

Performance of Topological Codes for Quantum Error Correction

Fern Harriet Erica Watson



Department of Physics
Imperial College London

*Thesis submitted in partial fulfilment of the
requirements for the degree of
Doctor of Philosophy in Physics
of
Imperial College London*

July 2015

I hereby declare that the work contained in this thesis is my own. Any information from other sources is appropriately referenced within the text.

The copyright of this thesis rests with the author and is made available under a Creative Commons Attribution Non-Commercial No Derivatives licence. Researchers are free to copy, distribute or transmit the thesis on the condition that they attribute it, that they do not use it for commercial purposes and that they do not alter, transform or build upon it. For any reuse or redistribution, researchers must make clear to others the licence terms of this work.

This thesis is dedicated to the memory of Sean Barrett.

Acknowledgements

My first and most heartfelt thanks go to my supervisor Dan Browne for his generosity. Dan was once described to me as ‘one of the nicest people working in the field’, and nothing I have learnt about him since contradicts this. Working with him has been inspiring and thought-provoking.

Special thanks go to Fay Dowker for the role she took during my most difficult time as a student. Her kindness has had a lasting impact on me, and she is in no small part responsible for me completing my studies.

In a similar vein I would like to thank Hussain Anwar. I have enjoyed working with him a great deal, and have learnt a lot from him, not just about physics. I would like to thank Earl Campbell, Ben Brown, James Auger and Tom Stace for enjoyable and instructive discussions.

Support has come in many forms over the years and from many people, of whom I mention just a few here. I would like to thank Kenneth Lessnoff, Alexandra Harper, Miranda Smith, Harriet Ballance and Stephen Milburn for their friendship and humour. I would also like to thank my uncle and aunt, Roy and Linda for their generosity.

I couldn’t have done this without the unwavering support of my parents. I thank them for their love and kindness throughout the years.

Finally I would like to thank James Millen for his unquestioning love and support, and for changing my life in so many good, interesting and exciting ways.

This work was made possible by the Centre for Doctoral Training in Controlled Quantum Dynamics. To complete the numerical work in this thesis I used the Imperial College High Performance Computing Service and UCL Legion High Performance Computing Facility (Legion@UCL).

Abstract

In this thesis we present three main contributions to the field of topological quantum error correcting codes. We focus on some of the properties of such codes required for fault-tolerant quantum computation.

Prior work has concentrated on determining error rate thresholds of particular models, but increasingly other parameters are gaining prominence. One of these is the overhead – the quantity of a named resource required to achieve a desired level of performance from the code. We characterise the qubit overhead of the toric code in a fault-tolerant setting. These results provide a general framework for determining the overhead for other code constructions with more complicated noise models.

Next we introduce a decoding algorithm, applicable to topological codes in a qudit architecture, specifically those where fault-tolerance is achieved through repeated syndrome measurements. It is computationally light and capable of decoding qudits of arbitrarily high dimension with negligible increase in its run time. The threshold of the decoder is limited by the percolation of the syndromes. Using local matching techniques we are able to overcome this limitation, increasing the threshold by almost a factor of two for high qudit dimensions.

Finally, we turn our attention to a second family of topological quantum codes: the colour codes. In three and higher spatial dimensions such codes can support transversal non-Clifford gates. We show, using a technique that we call a star-bipartition of the vertices of the lattice, that any existing qubit colour code lattice can be used to define a qudit colour code. By generalising the notion of triorthogonal matrices we derive analogous transversality properties in the qudit codes.

Overview

The following thesis contains work focussed on the properties of topological codes for quantum error correction, with a view to their use in fault-tolerant quantum computing schemes. Fault-tolerance is key, since the quantum systems and processes comprising the information processing task are unavoidably subject to a noisy environment. In order to store information in a robust way, quantum error correcting codes have been proposed, with a simple modification – namely repeated iterations of the error detection routine – allowing most codes to be realised fault-tolerantly.

We shall concentrate our efforts on the study of topological quantum error correcting codes, which have many desirable properties, making them probably the best candidates for experiments. All such error correcting codes can be realised in a 2D geometry, with the check operators (used for error detection) realised on a geometrically local subset of the qudits¹ comprising the code. The distance of the codes can be made arbitrarily large, meaning that they can correct any number of errors. Another important advantage is that numerical simulations indicate that they possess high error correction thresholds.

In Chapter 1 we shall provide a brief historical review of quantum error correction, making a connection to their classical counterparts and their limitations when a generalisation to the quantum domain was required. In this chapter we review the important framework of stabilizer codes and introduce the idea of quantum error correction using qudits, rather than the more commonly studied qubits. One of the two main families of topological codes, the surface code, is introduced in Chapter 2, with several variants detailed, including the planar code and a qudit surface code construction.

The first property of quantum error correcting codes we study is that of the logical error rate scaling. This metric becomes important after the threshold of the model² has been established, since we are interested in the case where the code is operating well below this threshold value. Previously, two asymptotic regimes of

¹We use qudit here to mean quantum systems of any dimension, including $d = 2$.

²The notion of ‘the model’ here encompasses the code, decoder and noise model chosen to study.

behaviour were established: in Chapter 3 we study these regimes and construct heuristic arguments to establish the parameter ranges over which they each provide a good approximation to the logical error rate scaling of the code. At the end of the chapter we provide an example calculation, showing how the two approximations can be inverted to give an estimate of the physical overhead of the code. Another interesting use for the results in this chapter could be to compare the efficacy of different decoders when the parameters are fixed by experimental limitations.

Quantum computation using qudits may offer some advantages, for example magic state distillation protocols are thought to be more efficient than for qubits. Furthermore, advances in experimental realisations have allowed many recent proof-of-principle experiments to demonstrate the viability of qudits in the laboratory for a variety of physical systems. This has led to an interest in the study of qudit error correcting codes. In Chapter 4 we study a decoder, the classical software that computes a correction operator from the syndrome resulting from the (quantum) error detection. The decoder is based on the simple intuition that at low error rates the errors are sparse, and give rise to small, sparse clusters of syndromes. We show that the decoder has a high threshold and a fast run time, which importantly is not sensitive to the dimension of the qudits, unlike some of the other decoders that have been proposed.

A second family of topological codes, the colour codes, offer a significant advantage over the surface codes. That is, in three and higher spatial dimensions a transversal non-Clifford gate can be realised, circumventing the need for magic state distillation altogether and potentially providing a significant reduction to the overhead of the computation. Error correcting thresholds are expected to be diminished significantly in such code due to their high-weight stabilizers, but it is possible that this could be mitigated somewhat by moving to qudit codes¹. In Chapter 5 we generalise all existing qubit colour codes to qudit constructions, for arbitrary spatial dimensions. We show that these codes can support a transversal non-Clifford gate from the m th level of the Clifford hierarchy, where $m \leq \mu$, the spatial dimension of the code.

¹The gauge colour code, rather than the colour code, may offer further practical advantages in this regard.

Contents

Contents	7
List of Figures	10
Nomenclature	12
1 Introduction to Quantum Error Correction	13
1.1 The Need for Fault-Tolerant Quantum Computing	14
1.1.1 Towards Quantum Error Correction	14
1.1.2 Quantum Computing on Encoded Information	16
1.2 The 3-Qubit Code	19
1.3 The Stabilizer Formalism	21
1.3.1 Stabilizer Quantum Error Correcting Codes	21
1.4 A Universal Fault-Tolerant Gate Set	24
1.4.1 Universality	24
1.4.2 Transversal Gates	26
1.4.3 Magic State Distillation	28
1.5 Qudit Stabilizer Codes	30
2 Topological Codes	32
2.1 Introduction	33
2.2 The Qubit Toric Code	34
2.2.1 An Introduction to Homology	36
2.2.2 Homology and the Toric Code	36
2.3 The Planar Code	40

2.3.1	The Qudit Surface Code	41
2.4	The Decoding Problem	44
2.4.1	Optimal Error Correction	45
2.4.2	Mapping to the Random Bond Ising Model	47
3	Logical Error Rate Scaling of the Toric Code	51
3.1	Introduction	52
3.2	Logical Failure Rates in Classical Coding Theory	54
3.3	Noise Models for the Toric Code	56
3.3.1	Error Correction	57
3.3.2	Simulating Noise and Error Correction in the Case of Perfect Syndrome Measurements	58
3.3.3	Simulating Noise and Error Correction in the Phenomenological Noise Model	59
3.4	The Universal Scaling Hypothesis	61
3.5	Determining the Threshold	63
3.5.1	Perfect Syndrome Measurements	63
3.5.2	The Threshold in the Phenomenological Noise Model	65
3.6	Evidence for the Universal Scaling Hypothesis	66
3.7	The Low Single-Qubit Error Rate Regime	68
3.8	The Validity of the Two Approximations	69
3.8.1	Testing the Range of Validity of the Universal Scaling Hypothesis	73
3.8.2	Testing the Range of Validity of the Low Error Rate Regime .	75
3.9	Comparison of the Overhead in the Two Regimes	76
3.10	Conclusions	82
4	A Fast Fault-Tolerant Decoder for Qubit and Qudit Surface Codes	85
4.1	Introduction	86
4.2	The Noise Model and Simulation Methods	88
4.3	The Phenomenological Noise Model	89
4.3.1	Simulating the Surface Code	90
4.4	HDRG Decoder with Noisy Syndromes	92
4.4.1	Decoder Construction	92

4.4.2	The Run Time of the HDRG Decoder	95
4.4.3	Thresholds Estimation and Percolation Limitation	96
4.4.4	Enhanced-HDRG Decoder	100
4.5	Conclusions	107
5	Qudit Colour Codes in All Spatial Dimensions	109
5.1	Introduction	110
5.2	Qubit Colour Codes	112
5.2.1	Colour Codes in Two Spatial Dimensions	112
5.2.2	Qubit Colour Codes in Higher Spatial Dimensions	114
5.3	Universality in Qudit Stabilizer Codes	115
5.3.1	Achieving Universality in Qudit Stabilizer Codes	116
5.4	Qudit Colour Codes	118
5.4.1	Star-bipartition	119
5.4.2	Qudit Colour Codes in Arbitrary Spatial Dimensions	120
5.5	m^* -orthogonal matrices and codes	123
5.5.1	Matrix Representation of Quantum Codes	123
5.5.2	Defining m^* -orthogonality	124
5.5.3	The m^* -orthogonality of qudit colour codes	126
5.6	Transversal Operators	128
5.6.1	Proof of lemma 9	132
5.7	Proof of the m^* -orthogonality Lemma	134
5.8	Conclusions	136
6	Summary and Outlook	139
	References	144

List of Figures

1.1	The circuit used to encode an unknown quantum state $ \psi\rangle$ into the 3-qubit code.	19
1.2	Implementing the T gate by magic state injection.	29
2.1	Representation of the stabilizer generators of the toric code.	34
2.2	Circuits for measuring the check operators of the toric code.	35
2.3	Cycles of the toric code corresponding to logical operators.	37
2.4	An example of error detection in the qubit toric code.	38
2.5	Examples of successful and unsuccessful error correction in the toric code.	39
2.6	The qubit surface code.	41
2.7	Vertex operator and plaquette operator of the qudit surface code. . .	42
2.8	An example of a distance 5 qudit planar surface code.	43
2.9	Examples of X -type errors and the syndrome transportation rule on the qudit surface code.	44
2.10	Three examples of underlying errors giving rise to the same syndrome pattern in the toric code.	45
2.11	An example error configuration on the toric code and the classical spins corresponding to the random bond Ising model.	48
2.12	The error chain corresponding to fixed antiferromagnetic couplings and the correction chain corresponding to excited bonds.	49
2.13	The transition from the ordered to disordered phase of the RBIM, and the Nishimori line.	50
3.1	A schematic of the logical error rate scaling of the toric code.	56

3.2	Flow diagram for the 3D HDRG decoder.	62
3.3	Data obtained from numerical simulations of the toric code failure rate close to threshold.	65
3.4	Dependence of the logical failure rate on the size of the lattice.	67
3.5	One way in which $\lceil L/2 \rceil$ errors lying along a minimum weight homologically non-trivial cycle will result in a logical error.	70
3.6	Data satisfying the condition $p > p_{\text{USH}}$ for the syndrome-noise-free error model.	74
3.7	Data satisfying the condition $p > p_{\text{USH}}$ for the phenomenological error model.	75
3.8	Logarithmic plot of all numerical data following the rescaling transformation $(L, p) \rightarrow x = (p - p_{c0})L^{1/\nu_0}$. The universal scaling fit is also shown.	76
3.9	The full set of rescaled data, coloured by lattice size. The low p analytic expression is shown for some small lattice sizes.	77
3.10	Rescaled data for the phenomenological noise model, as well as the low p analytic expression for this model, plotted for some small L values.	78
3.11	The range of validity of each of the regimes plotted as a function of the independent variables p and P_{fail}	79
3.12	A 3D plot of the overhead in each of the two regimes.	80
3.13	The overhead for the toric code in the presence of the phenomenological noise model for a range of parameters.	81
4.1	An illustrative picture of the data structure obtained in order to perform fault-tolerant error correction.	91
4.2	An illustration of the Manhattan distance metric.	93
4.3	The full HDRG decoding algorithm illustrated for a distance 5 code constructed from qudits with $d = 5$	98
4.4	An example for the collected simulation data used to estimate the threshold for qubits.	100
4.5	A summary of all qudit thresholds for different number of rounds of the initialisation step.	101

4.6	A summary of percolation thresholds for different number of rounds of the initialisation step.	101
4.7	An illustration of the first three initialisation levels.	102
4.8	The full decoding algorithm, with one initialisation level preceding it, illustrated for a distance 5 code constructed from qudits with $d = 5$. .	106
5.1	The two smallest instances of the 2D colour code defined on the hexagonal lattice.	114
5.2	The 15-qudit tetrahedral colour code.	121
5.3	A table illustrating the star-conjugate transversal implementation of some important logical gates and the spatial dimensions of the codes which support them.	132

Chapter 1

Introduction to Quantum Error Correction

Information stored in quantum systems is sensitive to the deleterious effects of the environment, from which it cannot be completely isolated. The goal of quantum computing is to coherently manipulate such quantum systems in order to perform information processing tasks – a goal which is impossible in the bare physical systems due to such uncontrolled interactions with the environment. Error correcting codes may be employed, using redundancy to encode the information in a larger Hilbert space. In this chapter we present a brief historical review of the key discoveries in the field of quantum error correction, culminating in the introduction of an important class of codes called stabilizer codes. We also introduce both the principle of universal quantum computation and some of the ways this can be achieved fault-tolerantly, that is, assuming that the elements of the circuit realising the computation are noisy. Finally, we show that stabilizer codes can be constructed for qudits: quantum systems defined as a superposition over d basis states.

1.1 The Need for Fault-Tolerant Quantum Computing

The ability to manipulate quantum systems lies at the heart of one of the most intriguing challenges currently facing physicists, namely quantum information processing. The potential of quantum computers is believed to lie in their ability to perform some specific algorithms much more efficiently than their classical counterparts. The landmark discoveries of Shor’s factoring algorithm [Shor, 1997] and Grover’s search algorithm [Grover, 1997] are two often-cited examples of algorithms that may demonstrate such a ‘quantum speed-up’.

One important consideration when any quantum system is under external control is its inherent instability in the presence of an *environment*. This generic term describes undesirable interactions due to any uncontrolled factors in the experiment. Take as an example a string of trapped ions – for a recent example of such an experimental setup realising a quantum error correcting code, see [Nigg et al., 2014]. In such a system there are many possible sources of error, for example stray electric fields, or imprecisely controlled laser pulses used to mediate interactions within or between ions.

Uncontrolled interactions with the environment can cause many different errors in different physical systems, for example leakage of information from the state to the environment, corruption of the information or complete loss of the quantum system¹. The exact form of the *noise model* used to represent the effect of the environment depends on the experimental setup under consideration, but it is possible nonetheless to develop generic methods to mitigate some of the potential errors.

1.1.1 Towards Quantum Error Correction

In developing a theory of quantum error correction it was natural to turn to the mature field of classical error correction for inspiration. Classical error correction is so well-established that the most distant man-made objects, Voyager I and II, have both been communicating with NASA via concatenated error correcting codes

¹Loss is likely to be the most prevalent error in photonic quantum systems, but is also present in other implementations if the qubits making up the computer leave the computational basis.

since 1989. The two space probes were initially launched in 1977 and use these codes to transmit data back to Earth – a distance of almost 2×10^{10} km. Indeed, classical codes are also used for many more prosaic purposes, for example encoding information in DVDs and barcodes, as well as in digital TV encoding. Many classical codes exist but the simplest is the *repetition code*, in which each bit is copied N times. After the storage time has elapsed the original information is recovered by selecting a decoding strategy (such as a majority vote).

However, there were some significant challenges in trying to adapt classical codes to the quantum setting. Perhaps the most obvious of these was the no-cloning theorem [Dieks, 1982; Wootters and Zurek, 1982], which states that an unknown quantum state cannot be copied. This simple observation, due to the linearity of quantum mechanics, eliminates the possibility of naïvely emulating simple classical repetition codes. Furthermore, classical codes are designed to correct bit-flip errors, but phase-flip errors have no analogue in classical computing¹.

The continuous nature of the errors also posed a challenge to the development of quantum codes [Chuang et al., 1995; Landauer, 1995; Unruh, 1995]. Fortunately there exists a strategy to ‘digitise’ these errors by making measurements. This projects them into either an error-free eigenstate of the measurement operator or an eigenstate with a large error. In making this measurement we can hope to use the information gained to learn something about the nature of the error, and in turn use this information in order to correct it. Developing a strategy to do this without destroying the quantum information itself presented a technical challenge.

In 1995 Shor proposed his famous 9-qubit code [Shor, 1995]. This code is able to protect one *encoded qubit* (often called a logical qubit) against a single arbitrary error by concatenating the 3-qubit bit-flip code, that will be discussed in Section 1.2, with a related 3-qubit code that can protect against a single phase-flip error. Shortly thereafter Steane introduced his 7-qubit quantum error correcting code [Steane, 1996a]. This code can also correct a single arbitrary error but the construction does not rely on concatenation and is more efficient than Shor’s code².

¹Classically, bits can flip from 0 to 1 and vice versa. The analogue in quantum information are bit-flip errors, which are of the form $\sigma_x(\alpha|0\rangle + \beta|1\rangle) = \beta|0\rangle + \alpha|1\rangle$. Phase-flip errors are those such as $\sigma_z(\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle - \beta|1\rangle$. Here σ_x and σ_z are Pauli operators defined in equation (1.1).

²Notably, the smallest code that can correct arbitrary single-qubit errors is the 5-qubit code [Laflamme et al., 1996].

Further error correcting codes were proposed quickly after Steane’s code [Calderbank and Shor, 1996; Steane, 1996b] and the CSS construction of quantum error correcting codes followed shortly thereafter. Named for their inventors, Calderbank, Shor and Steane, CSS codes take a pair of classical error correcting codes and use the quantum analogue of one of the codes to correct bit-flip errors and the quantum analogue of the other to correct phase-flip errors. By construction, bit-flip and phase-flip errors are inherently detected independently in CSS codes.

The stabilizer formalism introduced in [Calderbank et al., 1996] and refined into its current form in [Gottesman, 1997] is a powerful description of quantum error correcting codes. Such codes are represented by a set of operators known as the *stabilizers* of the code. By measuring these operators a certain set of errors can be detected and potentially corrected. All of the codes studied in subsequent chapters of this thesis are stabilizer codes. This class of codes will be introduced in more detail in Section 1.3.

1.1.2 Quantum Computing on Encoded Information

The development of error detecting and correcting codes is central to fault-tolerant quantum computing. Here the goal is to perform unitary operations directly on encoded quantum information even when all of the individual operations are subject to noise. Typically one estimates the tolerable noise rate in terms of the inverse of the total number of noisy operations. For example, in an early paper on fault-tolerance, Shor [Shor, 1996] demonstrated that a computation could be performed in the presence of noisy operations as long as the physical error rate was polylogarithmically small in the number of steps of the computation. By studying quantum Reed-Muller codes, Knill *et al.* [Knill et al., 1996] were able to demonstrate a similar result, that a set of fault-tolerant gates could be realised in such codes under the assumption of a polylogarithmically small error rate¹.

The first numerical estimate of the threshold was given by Steane [Steane, 1997]. He showed that for an error rate on the gates and measurements comprising the error detection circuit of $\sim 10^{-4}$ or less, active error correction can provide protection against such errors. However, the threshold theorem was first proved by Aharonov and

¹A polylogarithmic function is one defined as a polynomial of $\log x$ rather than x .

Ben-Or [Aharonov and Ben-Or, 1997], who were able to show that by concatenating codes, a constant threshold error rate of $\sim 10^{-6}$ can be achieved. Consequently, for any constant of rate errors below the threshold, active error correction can be used to improve the robustness to such errors. In numerical simulations of quantum error correcting codes, much higher thresholds have been reported, suggesting that fault-tolerant quantum error correction could be experimentally tractable.

Although in the literature it is common to use a single parameter to describe the single-qubit error rate, it is by no means clear that an interaction between the quantum system of n qubits and the environment should be described in such a simple way. To justify this, consider that the error models are necessarily based on many assumptions, for example that physical interactions are local. Under the additional assumption that the environment is isotropic, this implies that the environment will tend to give rise to errors that are identical and independently distributed (i.i.d.), with no correlations between the errors on different qubits. Any single-qubit error (with the exception of leakage errors) can be decomposed into a linear combination of a basis set of operations – here we shall choose the conventional single-qubit Pauli basis,

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad (1.1)$$

where $\sigma_x^2 = \sigma_y^2 = \sigma_z^2 = I$. Throughout this thesis we shall use the notation $X = \sigma_x$, $Y = \sigma_y$ and $Z = \sigma_z$ to represent the standard qubit Pauli matrices. Note that $Y = iXZ$. It follows that if a quantum error correcting code can correct the Pauli errors X and Z then it can correct any arbitrary single-qubit error.

Consider the depolarising channel, which is commonly parametrised for the quantum state ρ ¹ as

$$\mathcal{E}(\rho) = (1 - p)\rho + \frac{p}{3}(X\rho X + Y\rho Y + Z\rho Z). \quad (1.2)$$

Only the variable p , the rate at which physical errors occur, is required to describe

¹The density operator ρ is used to represent a mixed quantum state.

this model fully. If the code can correct the depolarising channel, as long as p is less than the threshold value, then it can correct arbitrary single-qubit errors.

The threshold is an important quantity for benchmarking the performance of a code since it is the maximum error rate that can be tolerated by that code, usually with respect to a given error model. The *logical error rate* (or logical failure rate) is the probability of the code failing for a given physical error rate. Below threshold it has been observed that the logical error rate can be exponentially suppressed (see Sections 3.2 and 3.4), whereas above the threshold, increasing the size of the code will increase the logical failure rate.

The early threshold lower-bound estimate by Aharonov and Ben-Or was groundbreaking because it proved that for a (small) constant error rate, quantum computing could still be performed. By numerically simulating error correcting schemes, much higher thresholds than those that are analytically proven are often revealed. A more recent example of a fault-tolerant quantum computing (FTQC) scheme is due to Raussendorf *et al.* [Raussendorf et al., 2006, 2007] which makes use of the topological quantum error correcting codes that we will introduce in Chapter 2. The threshold for this scheme obtained via simulations approaches 1%¹, which is realisable in many potential experimental implementations, for example: quantum dots [Veldhorst et al., 2014a,b], superconducting qubits [Barends et al., 2014] and trapped ions [Harty et al., 2014].

Some of the desirable properties for quantum error correcting codes related to FTQC schemes will be addressed in this thesis. We begin by considering the resources such schemes require, which should be quantified and minimised with respect to certain constraints. Such resources could be measured in computational time, number of qubits required, or some other metric, while the constraint is likely to be the desired accuracy of the computation. The codes should also have high thresholds so that an experimentally feasible error rate can be tolerated, and should ideally allow an arbitrary computation to be performed directly on the encoded information. Of course, this is not an exhaustive list of desiderata but provides some motivation for the studies presented in this thesis.

In this chapter we shall introduce stabilizer codes – one widely studied class

¹The value of the threshold is very sensitive to the exact error model – for a recent summary of results see [Stephens, 2014].

of quantum error correcting codes – and also discuss some ways in which encoded *universal* gate sets can be implemented. Finally, we shall introduce a higher dimensional¹ analogue of the qubit, known as the *qudit*. We shall briefly review the distinguishing features of qudit codes compared to their qubit counterparts.

1.2 The 3-Qubit Code

As a concrete example of an error correcting code, let us introduce a variant of the 3-qubit code that protects against a single bit-flip error. We begin with a single qubit in an unknown quantum state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where α and β are complex numbers satisfying $|\alpha|^2 + |\beta|^2 = 1$. We now take each of the basis states and replace them with an encoded basis state over 3 physical qubits:

$$\begin{aligned} |0\rangle &\rightarrow |0_L\rangle := |000\rangle, \\ |1\rangle &\rightarrow |1_L\rangle := |111\rangle. \end{aligned}$$

The circuit that performs this encoding is illustrated in figure 1.1. The encoded state is $|\psi_e\rangle = \alpha|0_L\rangle + \beta|1_L\rangle$. The error channel we consider applies bit-flip errors with probability p , with an error on each qubit of the code being equally likely.

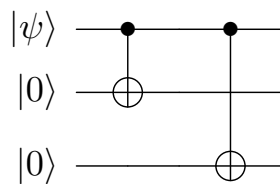


Figure 1.1: The circuit used to encode an unknown quantum state $|\psi\rangle$ into the 3-qubit code.

We begin by examining the case where up to one error has occurred and label the state $|\psi'_e\rangle$ in this case. The presence and location of the error is determined by

¹In this context, ‘higher dimensional’ means that the object is a superposition of $d > 2$ basis states.

making a series of projective measurements on the state. In this case, the projectors are:

$$\begin{aligned} M_0 &= |000\rangle\langle 000| + |111\rangle\langle 111|, \\ M_1 &= |100\rangle\langle 100| + |011\rangle\langle 011|, \\ M_2 &= |010\rangle\langle 010| + |101\rangle\langle 101|, \\ M_3 &= |001\rangle\langle 001| + |110\rangle\langle 110|, \end{aligned} \tag{1.3}$$

where the probability of a non-zero outcome for each projector M_i is

$$\langle \psi'_e | M_i^\dagger M_i | \psi'_e \rangle. \tag{1.4}$$

Making this measurement projects $|\psi'_e\rangle$ into one of four orthogonal states, where the exact state is labelled by the projector that had a nonzero outcome in the measurement. For example, if $\langle \psi'_e | M_2^\dagger M_2 | \psi'_e \rangle = 1$ then the projected state is $|\psi_p\rangle = \alpha |010\rangle + \beta |101\rangle$. To correct the error we apply X_k , where k labels the projector that was nonzero as well as the qubit we apply the correction to ($k = 0$ means no error occurred so there is no correction to apply). As long as our assumption that the number of bit-flip errors is ≤ 1 holds, then the error is corrected with certainty.

Of course, our assumption could be wrong and we illustrate the effects of such an error by example. Assume the code has suffered a bit-flip on the first two qubits, so it is in the state $|\psi'_e\rangle = \alpha |110\rangle + \beta |001\rangle$. Making the projective measurement will result in $\langle \psi'_e | M_3^\dagger M_3 | \psi'_e \rangle = 1$ which is indistinguishable from the case when a single bit-flip error occurs on the third qubit. Applying the correction operator indicated by the measurement outcome will result in a state of $|\psi_c\rangle = \alpha |111\rangle + \beta |000\rangle$. Using only the projective measurements in equation (1.3) this state cannot be distinguished from the original encoded state $|\psi_e\rangle$. Indeed, we see that the error and correction operators taken together have applied a *logical operation*, meaning that a bit-flip operation is applied to the encoded qubit. The logical bit-flip operator for this code is $X_1 \otimes X_2 \otimes X_3$, which sends $|0_L\rangle \leftrightarrow |1_L\rangle$.

This simple code offers no protection against other types of errors. For example a single phase-flip error is undetectable by this code, so it is not very robust even

for the simplest error models. A related 3-qubit code exists that protects against a single phase-flip, but to protect against bit-flip and phase-flip noise simultaneously the two codes can be concatenated, resulting in Shor’s 9-qubit code [Shor, 1995], which can protect against a single arbitrary error.

1.3 The Stabilizer Formalism

The 3-qubit code can also be viewed in another way. The uncorrupted code state $|\psi_e\rangle$ is the $+1$ eigenspace of a set of Pauli operators, the *stabilizers* of the code. This group of Pauli operators is *generated* by ZZI and IZZ ¹, hence the full set of stabilizers comprises all possible products of these operators. The stabilizers can be considered as parity check measurements, indicating the number of bit-flip errors (modulo 2) that have occurred on a particular pair of qubits. For instance, consider the situation where the measurement of ZZI yields an eigenvalue of -1 and the measurement of IZZ yields an eigenvalue of $+1$. The first outcome indicates an error on either the first or second qubit, while the second suggests no errors have occurred on either the second or third qubits. This pattern of measurement outcomes can therefore be interpreted as a bit-flip error having occurred on the first qubit, and hence an appropriate recovery operator can be inferred.

The *stabilizer formalism* provides a compact representation of a large class of quantum many-body states that can be used as error correcting codes. Moreover, this formalism can be used to represent the effect of Clifford operations² and Pauli measurements on the state. The stabilizer formalism and stabilizer error correcting codes were introduced in Ref. [Gottesman, 1997].

1.3.1 Stabilizer Quantum Error Correcting Codes

The Pauli operators (see equation (1.1)) generate the single-qubit Pauli group $\mathcal{P} = \langle X, Z \rangle$ up to a global phase of i^a for integer a , where ‘ $\langle \cdot \rangle$ ’ means the set of all possible products. The n -qubit Pauli group $\mathcal{P}^n$ is the n -fold tensor product of the single-qubit Pauli group $\mathcal{P}^{\otimes n}$.

¹We often omit the tensor product sign between operators so, for example, an operator such as ZZI should be read as the tensor product of these operators acting on three qubits, $Z_1 \otimes Z_2 \otimes I_3$.

²Clifford operations are introduced in Section 1.4.1.

The stabilizer group $\mathcal{S}$ of a general quantum state $|\psi\rangle$ is an abelian subgroup of $\mathcal{P}^n$ such that $i^a I \notin \mathcal{S}$ for $a \not\equiv 0 \pmod{4}$ and

$$S|\psi\rangle = +|\psi\rangle, \quad \forall |\psi\rangle \in \mathcal{C}, \quad \forall S \in \mathcal{S}. \quad (1.5)$$

Here $\mathcal{C}$ is the quantum error correcting code, in other words the space of vectors fixed by $\mathcal{S}$. A unique quantum state is defined by only n generators, whereas a quantum error correcting code is defined by fewer than n generators, allowing for information to be encoded in the unconstrained degrees of freedom. The dimension of the Hilbert space of a code is 2^k and this determines the number of encoded qubits, k .

A code is uniquely defined by its *generators* – a set of linearly independent operators that generate the stabilizer group. There are $n - k$ generators and the stabilizer group is found by taking all possible products of these generators, so that there are a total of 2^{n-k} elements of the stabilizer group $\mathcal{S} = \langle S_1, S_2, \dots, S_{n-k} \rangle$.

Logical Pauli operators of the code are defined as operators in the normalizer of $\mathcal{S}$ in $\mathcal{P}^n$, but not in $\mathcal{S}$. By definition, this means that they commute with the stabilizer group but are not elements of it. They act directly on the encoded information, causing the logical state to change. This may be desirable in the case of trying to perform operations directly on the encoded information, or undesirable as we saw in the 3 qubit code in Section 1.2 when an error and correction amounted to an undetected bit-flip operation on the encoded information.

Errors

Errors E are detected by measuring the generators S_i (sometimes called the check operators) of the code. The generators are not uniquely defined, with the only requirement being that they are independent. For example the sets $\langle S_1, S_2 \rangle$ and $\langle S_1 S_2, S_2 \rangle$ are equally valid sets of generators. Hence the check operators are selected such that their eigenvalues are likely to yield the most useful information about the errors that have occurred. They are Pauli operators so their eigenvalues are $\lambda_j = \pm 1 \forall j \in n - k$, and these outcomes give rise to the error *syndrome*. The syndrome is strictly the full vector $\mathbf{s}(E) = (\lambda_1, \lambda_2, \dots, \lambda_{n-k})$, but the term is often used to refer only to the non-trivial elements of this vector.

Due to the structure of the n -qubit Pauli group, all of its elements either commute

or anticommute with each of the others. Hence we conclude that a subset of all possible Pauli errors acting on the code will anticommute with at least one element of $\mathcal{S}$ and will therefore be detectable by the code. The logical operators of the code, by definition, commute with $\mathcal{S}$. Specifically, this means they commute with every element of $\mathcal{S}$ and therefore cannot be detected by the code. Indeed, any error that is undetectable by the code is equivalent to a logical operator (up to stabilizers) which will apply a logical operation, corrupting the encoded information. Note that the group of stabilizers, $\mathcal{S}$ is also undetectable, but they do not change the underlying quantum state and so are not considered to be errors, but rather correspond to the logical identity operation.

The code distance δ is the minimum weight¹ of an undetectable error. A stabilizer code can detect up to $(\delta - 1)$ single-qubit errors and correct up to $\lfloor (\delta - 1)/2 \rfloor$ errors. A *non-degenerate code* has unambiguous syndromes as long as $\lfloor (\delta - 1)/2 \rfloor$ or fewer errors have occurred. All of the codes we consider in subsequent chapters of this thesis are *degenerate codes* which means that all of the possible non-trivial syndrome configurations are ambiguous regardless of how few errors have occurred, see Chapter 2 for more details.

Determining the Correction Operator

Measuring the check operators of the code and determining the syndrome vector is not the end of the error correction protocol. A classical algorithm called a *decoder* must be employed to interpret the syndrome and return a correction operator. The decoder is trivial in some cases, for example the 3-qubit code described in Section 1.2. In Section 2.4 we discuss the decoding problem for a family of degenerate codes.

Stabilizer codes decompose errors into the Pauli basis, so in the construction of the correction operator we make use of the fact that all Pauli operators are self-inverse, meaning $\sigma^2 = I \ \forall \ \sigma \in \mathcal{P}^n$. Hence the error that has been identified can also be taken to be the correction operator. If the error was accurately identified then the original state is recovered. In fact, since all the error and recovery operations are in

¹For an operator $\mathcal{O} = \otimes_{i=1}^n P_j^k : k = \{0, 1\}$ defined as the tensor product of arbitrary single-qubit Pauli operators P_j on n qubits, the weight of the operator $|\mathcal{O}| = \sum k$. Equivalently it is the number of non-identity Pauli operators in the tensor product.

the n -qubit Pauli group it suffices to track the Pauli frame of the logical qubit¹ until a non-Clifford gate is applied or the state is read out, at which point a Pauli frame update must be applied [DiVincenzo and Aliferis, 2007; Knill, 2005].

1.4 A Universal Fault-Tolerant Gate Set

1.4.1 Universality

Quantum computations can be implemented by performing operations on a set of qubits initialised in a known quantum state². We consider only unitary dynamics on pure quantum states, so that an operation U acts as $U|\psi_i\rangle = \lambda_i|\psi_i\rangle$ for eigenstate $|\psi_i\rangle$ and eigenvalue λ_i . A unitary operator obeys $U^\dagger U = UU^\dagger = I$ and preserves the inner product between two states so the total information that can be extracted from the state is invariant under the action of a unitary operator.

In this thesis we will be interested in operations from three different levels of the *Clifford hierarchy* (CH). This hierarchy of quantum operations was introduced by Gottesman and Chuang [Gottesman and Chuang, 1999] and is defined recursively [Campbell et al., 2012; Howard and Vala, 2012] as

$$\mathcal{P}_{\ell+1}^n = \{U \mid UPU^\dagger P^\dagger \in \mathcal{P}_\ell^n \ \forall \ P \in \mathcal{P}_1^n\}, \quad (1.6)$$

where U is a unitary operator acting on n qubits. Note that $\mathcal{P}_\ell^n \subset \mathcal{P}_{\ell+1}^n$, so that the ℓ^{th} level of the CH is automatically contained in the $(\ell+1)^{\text{th}}$ level. In this definition of the CH, $\mathcal{P}_1^n = \mathcal{P}^n$ is the n -qubit Pauli group defined in Section 1.3 and the second level $\mathcal{P}_2^n$ is called the Clifford group. By definition, the Pauli group is invariant under conjugation by operators from the Clifford group, hence the Clifford group is the normalizer of the Pauli group. Only three gates are required to generate the Clifford

¹The Pauli frame is the overall correction operator accumulated from many rounds of error correction, which may include logical Pauli operators. While all logical and physical operations are Pauli it suffices to track this frame, since Pauli operators are unchanged (up to a global phase) when conjugated by another Pauli operator.

²This is the so-called *circuit model* of quantum computing. There is an equally powerful *measurement-based model* [Raussendorf and Briegel, 2001; Raussendorf et al., 2003] in which single-qubit measurements are performed on a *cluster state* to enact a computation.

group. These are the single-qubit Hadamard, H

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad (1.7)$$

the single-qubit phase, S

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad (1.8)$$

and the two-qubit entangling gate known as the controlled- X or CNOT gate

$$\text{CNOT} = \begin{pmatrix} I & 0 \\ 0 & X \end{pmatrix}. \quad (1.9)$$

A quantum computer prepared in Pauli eigenstates and implementing only stabilizer operations¹ is not universal for quantum computing. The Gottesman-Knill theorem [Gottesman, 1998] says that a quantum computer limited to such operations could be efficiently classically simulated and hence has no computational power beyond that which a classical computer can achieve. There are two well-known strategies to circumvent this limitation. The first, which we shall outline in Section 1.4.3, involves the preparation of specific states which are non-Pauli eigenstates – the so-called *magic states*. A quantum computer that can prepare such states and implement unitaries from the Clifford group is universal [Bravyi and Kitaev, 2005].

A second approach is to augment the Clifford group unitaries with at least one gate from the third level of the CH to achieve universal quantum computation. In this thesis we shall consider the T gate,

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{2i\pi/8} \end{pmatrix}, \quad (1.10)$$

because it has a straightforward fault-tolerant implementation [Nielsen and Chuang,

¹Stabilizer operations are: preparation of stabilizer states, Clifford unitaries (including those conditioned on the outcomes of previous measurements) and measurement in the Pauli basis.

2010].

The set of unitary operations on a single (encoded) qubit form a continuum described by the group $SU(2)$, the single-qubit unitaries with determinants of unity. It is by no means obvious that the discrete set of gates described can generate the necessary operations on such an encoded qubit. Fortunately the Solovay-Kitaev theorem [Kitaev, 1997a; Nielsen and Chuang, 2010] guarantees that the generating set H , S and T can efficiently approximate any desired single-qubit gate to arbitrary accuracy. The number of gates to approximate an arbitrary unitary to accuracy ϵ is $O(\log^c(1/\epsilon))$, for constant c known to lie in the range $1 \leq c < 4$. In the next section we describe some of the ways in which an encoded unitary from this generating set can be performed.

1.4.2 Transversal Gates

An error arising on an individual qubit in a quantum error correcting code cannot propagate to another qubit in the code unless the two qubits interact. Any encoded operation should be realised in such a way as to minimise interactions between qubits within a *code block*¹, to limit the spread of physical errors and in turn reduce the possibility of a logical error.

With this in mind, encoded single-qubit gates may be realised by a tensor product of operations on all of the individual qubits in the code block. Since the operator affects no interactions between qubits, this ensures that errors that have occurred do not spread within the code block during the operation. When an encoded two-qubit interaction is required, it should be performed between two code blocks, such that qubit k in the first block only ever interacts with the corresponding qubit k in the second block. Here, errors may spread in a benign way, in other words *between* code blocks, but not *within* them².

Gates acting in this way are called *transversal*. Fault-tolerance arises in such operations because the errors are bounded, and because each code block is decoded independently.

¹A code block is the fixed set of n physical qubits into which k logical qubits are encoded.

²If interactions between larger numbers of qubits are necessary, for example to realise a Toffoli (controlled-controlled-NOT) gate, they may be decomposed into a sequence of one- and two-qubit gates, which is possible according to the Solovay-Kitaev theorem.

An Example of a Transversal Gate

The $[[7,1,3]]$ Steane code¹ mentioned in Section 1.1.2 is capable of correcting a single arbitrary error. It is a CSS code which possesses a transversal Hadamard, phase and CNOT gate [Gottesman, 2010]. Here we will not expand on the explanation of the code, but merely demonstrate the transversality of one of these gates (the Hadamard).

The generators of the code are

$$\begin{aligned} S_{X,1} &= X I I X I X X, & S_{Z,1} &= Z I I Z I Z Z, \\ S_{X,2} &= I X I X X I X, & S_{Z,2} &= I Z I Z Z I Z, \\ S_{X,3} &= I I X I X X X, & S_{Z,3} &= I I Z I Z Z Z, \end{aligned} \quad (1.11)$$

and the logical Pauli operators are

$$\bar{X} = X^{\otimes 7}, \quad \bar{Z} = Z^{\otimes 7}. \quad (1.12)$$

The action of the single-qubit Hadamard on the Pauli operators X and Z is

$$H X H^\dagger = Z, \quad H Z H^\dagger = X. \quad (1.13)$$

Defining the transversal Hadamard to be $\bar{H} = H^{\otimes 7}$ we see that, for example

$$\begin{aligned} \bar{H} S_{X,1} \bar{H}^\dagger &= H X H^\dagger \otimes H I H^\dagger \otimes H I H^\dagger \otimes H X H^\dagger \otimes H I H^\dagger \otimes H X H^\dagger \otimes H X H^\dagger, \\ &= Z I I Z I Z Z, \\ &= S_{Z,1}. \end{aligned}$$

Using the same argument we can show that all of the stabilizers transform correctly under the action of the transversal Hadamard, so that $S_{X,j}$ transforms to $S_{Z,j}$ and vice versa. We also find for the $[[7,1,3]]$ Steane code that $\bar{H} \bar{X} \bar{H}^\dagger = \bar{Z}$ and $\bar{H} \bar{Z} \bar{H}^\dagger = \bar{X}$, as expected. Hence the logical Hadamard can be implemented transversally in this code.

Indeed, there are two main CSS code families known, one of which possesses

¹The notation $[[n, k, \delta]]$ is commonly used for a code to indicate the number of physical qubits n , the number of logical qubits k and the code distance δ .

the Clifford group (or a subgroup of it) but no gates from higher in the CH, and the other has a non-Clifford gate but not the Hadamard gate. This points to a no-go theorem by Eastin and Knill [Eastin and Knill, 2009], who have shown that no quantum code capable of detecting errors can possess a universal transversal gate set¹. Depending on the code, an alternative fault-tolerant implementation of at least one of the gates from the universal set must be found. In the next section we shall outline one of the most famous methods of performing a fault-tolerant non-Clifford operation, although several alternatives exist.

1.4.3 Magic State Distillation

In most well-known codes, except for quantum Reed-Muller codes and colour codes (see Chapter 5), the T gate cannot be transversally implemented. Hence we require alternative methods to realise this logical unitary. The most common way to achieve a non-Clifford gate fault-tolerantly is through *magic state injection* [Bravyi and Kitaev, 2005].

Bravyi and Kitaev originally proposed magic state injection protocols for the state $|T\rangle = \cos \beta |0\rangle + e^{i\pi/4} \sin \beta |1\rangle$, where $\beta = \frac{1}{2} \arccos \frac{1}{\sqrt{3}}$, and for the eigenstate of the Hadamard operator $|H\rangle = \cos(\pi/8) |0\rangle + \sin(\pi/8) |1\rangle$. Both of these magic states are capable of injecting a non-Clifford phase gate. For simplicity, we will restrict our discussion to the ‘ H -type magic states’. Such states correspond to points lying the equatorial planes of the Bloch sphere, equidistant from the Pauli eigenstates in that plane. We shall consider the H -type magic state in the x - y plane

$$|\Theta\rangle = |0\rangle + e^{i\pi/4} |1\rangle, \quad (1.14)$$

which is required to implement the T gate given in equation 1.10.

Promotion of the set of Clifford operations to universality requires the encoded (unknown) state $|\psi\rangle$ to be entangled with an encoded ancilla prepared in a magic state. This ‘state injection’ can be realised using a simple quantum circuit, see figure 1.2, so we see that the complexity comes almost entirely in producing the magic state itself.

¹Earlier results proved that no stabilizer code can possess a universal transversal gate set [Chen et al., 2008; Zeng et al., 2007].

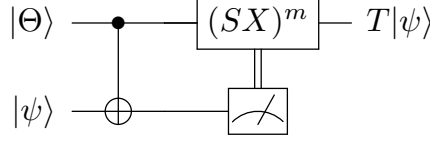


Figure 1.2: Implementing the T gate by magic state injection. After the CNOT gate, the second qubit is measured in the Z basis and the measurement outcome $m = \{0, 1\}$ indicates the correction operator that should be applied to the first qubit. This is also known as gate teleportation.

The preparation of such a state is non-trivial. Since it is not a stabilizer state it cannot simply be prepared by Clifford operations, instead it must be purified or *distilled*. In this thesis we describe only the $|\Theta\rangle$ magic state distillation protocol. We assume that we have access to a noisy preparation of the state that we shall call ρ_Θ , and that a large number of copies of this state can reliably be produced by repeating the noisy preparation procedure. Protocols using only Clifford operations exist that take several copies of ρ_Θ and purify them [Bravyi and Kitaev, 2005; Raussendorf et al., 2007].

The initial step in the distillation protocol is the production of a maximally entangled pair of qubits. One half of this Bell pair is encoded into a 15-qubit Reed-Muller code by measuring the code's X stabilizer generators. If the measurement outcomes are all trivial, the encoding is successful. This is followed by a transversal T gate and then measurement in the X basis of all 15 code qubits. The transversal T gate is implemented by the transversal application of the gate teleportation circuit in figure 1.2, using as a resource 15 copies of ρ_Θ rather than 15 copies of the pure state $|\Theta\rangle$.

If the protocol is successful, meaning that the measured stabilizer eigenvalues are all trivial, then the half of the Bell pair that was not encoded in the Reed-Muller code is projected into the state ρ'_Θ such that its fidelity with respect to the desired pure state is increased:

$$\sqrt{\langle\Theta|\rho'_\Theta|\Theta\rangle} > \sqrt{\langle\Theta|\rho_\Theta|\Theta\rangle}. \quad (1.15)$$

As we have implied, this is a probabilistic protocol and if it fails, meaning that

any of the measured stabilizer eigenvalues are non-trivial, the resulting state is discarded. Fortunately, the process can be concatenated to an arbitrary number of levels, resulting in a state that is arbitrarily pure. Such purity is costly in terms of the resource of the noisy magic state ancillas. However, recent progress in more efficient encodings has been made [Bravyi and Haah, 2012; Jones, 2013; Meier et al., 2012].

Magic state injection by gate teleportation is one of the best-known techniques for implementing a transversal non-Clifford gate. However, as we have mentioned, codes exist that have a transversal non-Clifford gate but not a transversal Hadamard gate. In such cases the logical Hadamard can be realised using magic state injection or a technique called *gauge fixing* [Paetznick and Reichardt, 2013]. Other techniques to achieve a universal fault-tolerant gate set exist, such as fault-tolerant ancilla preparation for the Toffoli gate [Shor, 1996], code concatenation [Jochym-O'Connor and Laflamme, 2014] and code conversion [Anderson et al., 2014], but these will not be discussed in this thesis.

We have now completed our brief introduction to fault-tolerant quantum computing. We conclude this section by bringing to the reader's attention the dominance of the qubit in all we have discussed thus far. This is a natural extension of the classical realm, where the bit likewise dominates. However, as in the classical domain, base 2 is not the only possible choice. In the next section we therefore introduce the notion of qudits, the quantum analogue of a base d classical bit.

1.5 Qudit Stabilizer Codes

Qubits have been studied as the basic building block for quantum computing and hence error correcting architectures because as 2-level systems they are the natural quantum analogue of classical bit. *Qudits* (d -level quantum systems) are the natural higher-dimensional generalisation of qubits, defined as a quantum superposition of d basis states. The primary advantage of qudit quantum error correcting codes is that the magic state distillation procedures are much more efficient [Campbell, 2014; Campbell et al., 2012]. There is also evidence for improved thresholds in some codes as the qudit Hilbert space dimension increases [Anwar et al., 2014], and we shall explore this more in Chapter 4.

Associated to each d -level quantum state is a Hilbert space $\mathcal{H}_d$. This is a complex vector space with an inner product and norm, and is spanned by the computational basis states $|\alpha\rangle$ where $\alpha \in \mathbb{Z}_d$. In the case of prime qudit dimensions, the d -element cyclic group $\mathbb{Z}_d = \{0, \dots, d-1\}$ can be conveniently identified with addition over integers modulo d . The fact it is a cyclic group means $(d-a) \in \mathbb{Z}_d \equiv -a \in \mathbb{Z}_d$.

The conventional single-qubit Pauli operators are generalised as:

$$X_d^a = \sum_{j \in \mathbb{Z}_d} |j \oplus a\rangle \langle j|, \quad Z_d^b = \sum_{j \in \mathbb{Z}_d} \omega^{bj} |j\rangle \langle j|, \quad (1.16)$$

where $\omega = e^{2\pi i/d}$ and the addition ‘ $\oplus$ ’ is taken to be modulo d throughout this thesis. We shall drop the subscript d unless it is necessary for clarity. Notice that now $X^d = Z^d = I$. Unlike qubit Pauli operators, these unitary operators are not Hermitian when $d > 2$, but they possess orthogonal eigenspaces with eigenvalues of the form ω^j , for some j . Hence, we can still interpret them as corresponding to physical observables with measurement outputs labelled by their complex eigenvalues.

The qudit Pauli operators obey the ω -commutation relation $X^a Z^b = \omega^{-ab} Z^b X^a$ for arbitrary $a, b \in \mathbb{Z}_d$. They generate the single-qudit Pauli group $\mathcal{P}_d = \langle X, Z \rangle$ up to a global phase of ω^j for $j \in \mathbb{Z}_d$. The n -qudit Pauli group $\mathcal{P}_d^n$ is the n -fold tensor product of the single-qudit Pauli group $\mathcal{P}_d^{\otimes n}$.

Similarly to the qubit case, the code space of a qudit stabilizer code is defined as the ‘+1’ eigenspace of an abelian subgroup $\mathcal{S} \in \mathcal{P}_d^n$, such that $\omega^j I \notin \mathcal{S}$ for non-zero j . The qudit syndrome measurements can have outcomes $\omega^j \quad \forall j \in \mathbb{Z}_d$, and we often denote these outcomes simply by j , which we call the *charge* of the syndrome¹.

In the next chapter we begin by introducing both qubit and qudit variants of a family of degenerate quantum error correcting codes that have useful properties for FTQC applications: the surface codes. In Chapter 3 we study some of the properties of the qubit version of the surface code. The decoders for such codes are nontrivial and in Chapter 4 we introduce a new decoder that performs well for both the qubit and qudit variants of the code. We conclude in Chapter 5 by generalising a second family of quantum error correcting codes, colour codes, from the qubit to the qudit domain.

¹This comes from the anyon (quasi-particle) description of topological codes. We do not consider this description at all in this thesis, but simply borrow this convenient terminology.

Chapter 2

Topological Codes

Topological codes are a class of stabilizer error correcting codes defined on lattices with useful properties. Consequently, they have become the leading candidates in the search for quantum error correcting codes to form part of a scheme to realise an experimentally implementable quantum computer. Their defining feature is that the encoded information is dependent not on local parameters but rather on the nature of the manifold in which the lattice is embedded. The information is hence associated with global degrees of freedom. Thus, each code family is parametrised by the lattice size, and increasing the lattice size increases the error protection offered by the code. Furthermore, the stabilizer generators are geometrically local. This offers two key benefits: firstly they are easily measured in an experimental setting, and secondly any noise introduced in measuring them remains local. By employing suitable recovery strategies that aim to minimise the chances of a local error being mapped to a global operation, the chances of failing to correct the errors are low as long as the code is operating below the error threshold. In this chapter we introduce several variants of one prominent example of a topological code: the surface code.

2.1 Introduction

So far we have discussed protection against errors and fault-tolerant implementations of gates for universal quantum computation. In this chapter we introduce surface codes: a class of codes invented in 1997 by Alexei Kitaev. There are two overwhelming benefits of this class of codes over the code constructions by Calderbank, Shor and Steane introduced in Chapter 1. First, the code distance scales very efficiently, for example as $O(n^{1/2})$ in 2D, where n is the number of qubits comprising the code. The second benefit of these codes is that their check operators are geometrically local. Since physics acts locally this makes the check operators easier to realise, thus this property increases the possibility of experimentally implementing such a code.

The surface code [Bravyi and Kitaev, 1998; Kitaev, 1997b, 2003] is one of a family of topological stabilizer codes, and is the basis for an approach to fault-tolerant quantum computing for which high thresholds have been reported [Fowler et al., 2012a; Raussendorf and Harrington, 2007; Raussendorf et al., 2007; Wang et al., 2003]. Surface codes are able to perform quantum error correction fault-tolerantly, that is, they are robust not only to physical errors on the individual qubits, but also to noise in the syndrome measurements.

The toric code [Dennis et al., 2002] is among the most extensively studied of this family of codes, revealing much insight into related topologically ordered systems. The great benefits that these codes offer is reflected in the degree of experimental interest, for example in superconducting qubits [Kelly et al., 2015]. A great deal of theoretical work has concentrated on calculating thresholds for various error models, for example models including lost qubits [Stace et al., 2009], and on the discovery and implementation of new classical decoding algorithms such as the soft-decision renormalisation group algorithm [Duclos-Cianci and Poulin, 2010a,b]. The toric code performs well, with high thresholds for some commonly studied noise models.

In this chapter we shall outline the qubit toric code construction due to Kitaev and while discussing the planar variant often known as the surface code we shall include its generalisation to a qudit form. The introduction of the other widely studied family of topological codes, colour codes, is left until Chapter 5.

2.2 The Qubit Toric Code

In the toric code, physical qubits reside on the edges of an $L \times L$ square lattice, as shown in figure 2.1. There are $n = 2L^2$ physical qubits comprising the code. Periodic boundary conditions are imposed and the lattice can be imagined to be embedded on the surface of a torus. The dual lattice is one in which the vertices of the primal lattice are replaced by plaquettes, and vice versa.

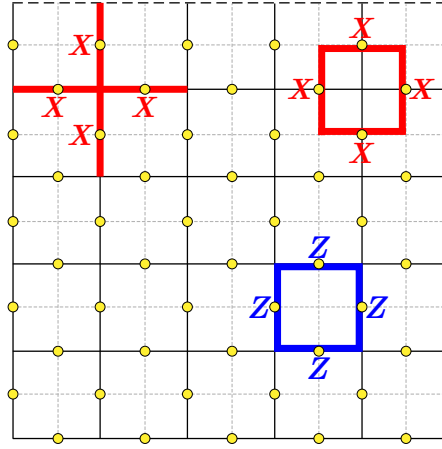


Figure 2.1: Representation of the stabilizer generators on an $L = 5$ toric code lattice. Qubits, shown as yellow circles, are placed on the edges of the lattice. Note that the top and bottom boundaries are identified with one another, as are the left and right boundaries. This is indicated by the black dashed lines. The dual lattice is shown using dashed grey lines. *Top*: A vertex operator on the primal lattice (left) and the dual lattice (right). *Bottom*: A plaquette operator on the primal lattice.

The toric code is described by a set of two types of commuting *stabilizer generators* – the so-called *vertex*, A_V , and *plaquette*, B_P , operators, defined as

$$A_V = \otimes_{i \in V} X_i, \quad B_P = \otimes_{i \in P} Z_i, \quad (2.1)$$

where X and Z are the conventional single-qubit Pauli operators, V indicates a vertex and P a plaquette of the primal lattice. The A_V operators therefore act on the four qubits surrounding a vertex of the lattice, and the B_P operators act on the four qubits contained in a plaquette, see figure 2.1. The measurement of these four-body

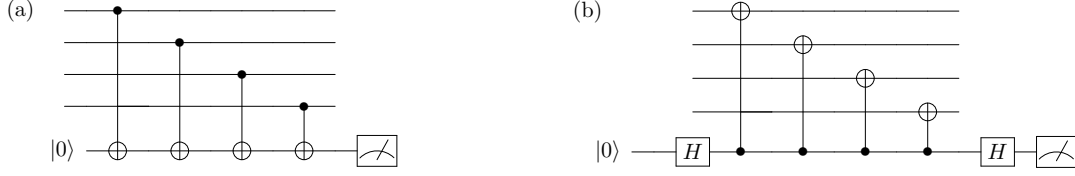


Figure 2.2: (a) The circuit for measuring a plaquette generator of the toric code, using four CNOT gates. The four top qubits are the data qubits and the one at the bottom is the ancilla qubit, prepared in the $|0\rangle$ state and measured in the Z (computational) basis. (b) The circuit for measuring a vertex generator of the toric code.

terms can be decomposed into four two-qubit CNOT gates with the addition of an ancilla, see figure 2.2 [Dennis et al., 2002].

We denote the logical encoded state of the toric code by $|\psi\rangle_{\text{toric}}$. In the absence of noise, measuring any element of $S = \{A_V, B_P\}$ on this state will yield a $+1$ eigenvalue:

$$S_i |\psi\rangle_{\text{toric}} = + |\psi\rangle_{\text{toric}}, \quad (2.2)$$

where $S_i \in S$. The stabilizer group $\mathcal{S}$ is generated by S with multiplication being the group action, $\mathcal{S} = \langle A_V, B_P \rangle$. All elements of the stabilizer group act trivially on the code space.

The product of all vertex generators $\prod A_V = I$ and similarly for the plaquette operators $\prod B_P = I$. Overall there are $s = 2(L^2 - 1)$ independent generators and $n = 2L^2$ physical qubits, hence there are $n - s = 2$ degrees of freedom in the system. This means that the code space is 2^2 -dimensional and so can encode two logical qubits. The number of encoded qubits is independent of n , the number of qubits comprising the code, hence the toric code protects a constant number of logical qubits regardless of its lattice size.

In the toric code the quantum information is protected in global, rather than local, degrees of freedom. Any physically realistic error model is assumed to act locally, so that the probability of the errors accessing global degrees of freedom is low, for a large enough system size (and for an error rate below threshold).

The symmetry between the *primal lattice* and the *dual lattice* (constructed by replacing plaquettes of the primal lattice by vertices and vice versa) shown in

figure 2.1, reveals a useful symmetry in the stabilizers of the toric code. On the dual lattice the A_V operators act on the qubits surrounding a plaquette, as shown in figure 2.1. By considering both the primal and dual lattices we can view all stabilizers as closed loops, meaning that all plaquette-type operators on the primal lattice have an analogous vertex-type operator on the dual lattice. It follows that all results calculated for either bit-flip or phase-flip errors are interchangeable with results for the other type of error.

2.2.1 An Introduction to Homology

It is often convenient when discussing the qubit surface code to employ the language of algebraic topology. A short introduction to the relevant ideas is provided for the reader, but it is necessarily not comprehensive. A good introduction to algebraic topology can be found in many textbooks, for example see [Henle, 1994].

Any 2D surface can be *cellulated* such that its surface is tiled – we shall limit the present discussion to a square cellulation of the surface. The cellulation defines n -cells on the surface, where n is the spatial dimension of the cell. Therefore in 2D there exist 0-cells corresponding to the vertices, 1-cells corresponding to the edges, and 2-cells corresponding to the plaquettes.

An n -chain assigns an element of a group, in this case $\mathbb{Z}_2$ since the code is comprised of qubits, to every n -cell. The meaning of such an assignment will become clear in the next section. The boundary of an n -chain is defined as a linear map ∂ ; the boundary of a n -chain is a $(n - 1)$ -chain, and a key property is that a boundary itself has no boundary, or $\partial^2 = 0$.

This leads on to the idea of a n -cycle, defined as a chain with no boundary. There are two important classes of cycles; homologically trivial n -cycles, which are the boundaries of $(n + 1)$ -chains, and homologically non-trivial cycles which are not boundaries. Two n -chains are *homologically equivalent* if they differ by an n -cycle. Homologically equivalent n -chains belong to the same *homology class* [Hatcher, 2002].

2.2.2 Homology and the Toric Code

In this formalism, all of the toric code stabilizers correspond to *homologically trivial cycles*. In figure 2.3 we show an example of a homologically trivial cycle that is

generated by multiplying two adjacent stabilizer generators together. We see that all homologically trivial cycles are equivalent to products of stabilizer generators and hence act trivially on the code space.

The *logical operators* are also represented by cycles of Pauli operators. However, these cycles wrap around the torus and are not homologically equivalent to stabilizers. The logical operators correspond to *homologically non-trivial cycles* and have a non-trivial effect on the code space. We denote this group of logical operators $\mathcal{L}$. We denote the set of logical operators which do not leave the code space invariant as $\mathcal{L} - \mathcal{S}$. The distance of a topological code corresponds to the weight of shortest possible logical operator, i.e. the distance is L .

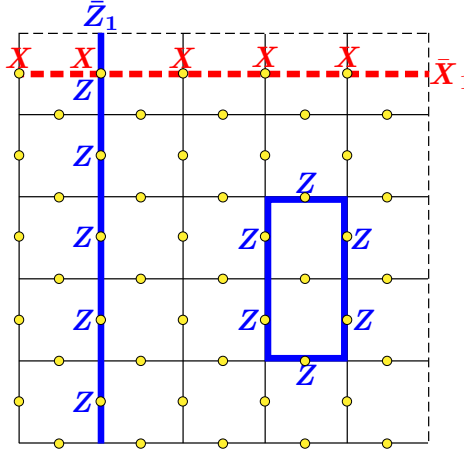


Figure 2.3: *Left:* $\bar{Z}_1$ is a minimum-weight homologically non-trivial cycle, equivalent to a logical operator acting on the encoded information. *Top:* The $\bar{X}_1$ operator, drawn as a cycle on the dual lattice (lattice not shown). The $\bar{X}_1$ logical operator shares a single physical qubit with $\bar{Z}_1$ and hence they anticommute. *Right:* An example of a homologically trivial cycle generated by multiplication of two adjacent plaquette operators.

There are two sets of $\bar{Z}$ and $\bar{X}$ logical operators addressing the two encoded qubits (overbar indicates a logical operation). One of these, labelled $\bar{Z}_1$, is shown in figure 2.3 spanning the lattice vertically. The corresponding $\bar{X}_1$ is also shown, and forms a closed horizontal loop on the dual lattice. By multiplying a logical operator by a subset of stabilizers we can continuously deform the minimum-weight cycle $\bar{Z}_1$

into any other operator spanning the lattice vertically. The set of operators that are equivalent up to stabilizer operations belong to the same homology class.

Errors are detectable if they anticommute with at least one element of the set of stabilizers $\mathcal{S}$. If we assume that the stabilizers are measured perfectly then it follows that if any non-trivial eigenvalues are observed, this indicates the presence of errors with certainty. The pattern of stabilizers that anticommute with a given error reveals some information about the location and most likely type of error, although it cannot uniquely identify the error. This ambiguity is due to the code degeneracy.

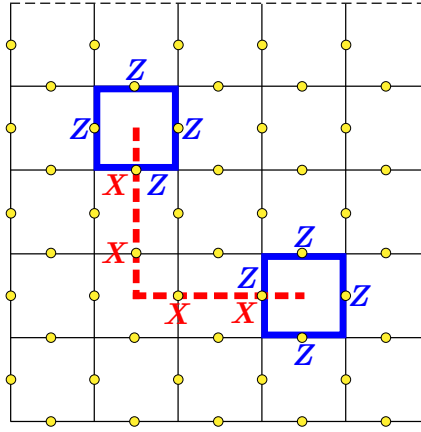


Figure 2.4: A string of X errors is shown as a dashed line on the dual lattice. Measuring the two B_P generators indicated yields -1 eigenvalues because the stabilizer and error chain anticommute at these locations. Note that if the X error chain forms a cycle then it will not be detectable.

The edges of the lattice are the 1-cells of the cellulation of the surface associated with the toric code lattice. The edges of the lattice are themselves associated with the qubits of the code. By labelling the absence of an error with the value ‘0’ and the presence of an error with the value ‘1’ we see that the set of all errors on the lattice is a 1-chain, labelled E . We use the notation introduced from algebraic topology to indicate the boundary of the chain of errors as ∂E . The errors commute with the stabilizers except at the boundary of the chain where they anticommute and the measured eigenvalues are non-trivial. The full set of stabilizer eigenvalues is called the *syndrome*. Figure 2.4 shows a string of X errors and the two plaquette operators that anticommute with it.

Once the syndrome has been established we employ a classical algorithm called a *decoder* to decide which correction operator, E' , to apply. Broadly speaking, the goal of the decoder is to identify 1-chains between non-trivial syndromes, representing the correction operator, such that the total operator $C = E + E'$ has the highest probability of being a homologically trivial cycle and thus a member of the stabilizer group. Failure of the decoding algorithm corresponds to the creation of a homologically non-trivial cycle, which corrupts the encoded information. Optimal decoding is discussed in more detail in Section 2.4.1 and a new decoding algorithm is introduced in Chapter 4.

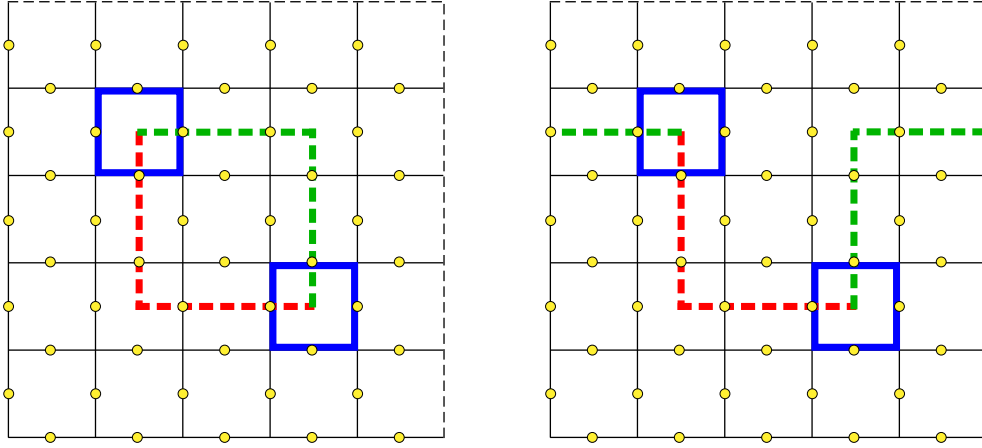


Figure 2.5: The true error chain E is marked in red, and the resulting non-trivial syndrome indicated as blue plaquettes. The applied correction chain E' is in green. (a) The error and correction chain forms a A_V stabilizer operator and so this correction has been successful. (b) The resultant operator $E + E'$ is a nontrivial cycle so the correction has been unsuccessful.

It has been observed that the code failure rate decreases exponentially with increasing L [Dennis et al., 2002; Raussendorf et al., 2007] as long as the physical error rate p is below the threshold value p_c . It is understood that this is the threshold with respect to the parameters of the simulation, namely the error model and decoder chosen. If $p > p_c$ then increasing the block size will decrease the code performance and the logical error rate will be enhanced with respect to the physical error rate. The meaning of the threshold will be discussed in more detail in Section 2.4.1.

2.3 The Planar Code

The toric code is an extremely simple and elegant model of a topological quantum error correcting code. Its main drawback is that its geometry is difficult to implement experimentally, so a planar variant of the code, usually known as the surface code, has been proposed [Bravyi and Kitaev, 1998; Freedman and Meyer, 1998].

Similarly to the toric code, in a planar surface code qubits are identified with the edges of an $L \times L$ lattice but in this case boundaries are present, as shown in figure 2.6. The surface code is a stabilizer code with two types of stabilizer generators $\mathcal{S} = \langle A_V, B_P \rangle$ defined on the lattice as in equation (2.1). Note that the two different types of boundary (*rough* and *smooth*) on the lattice lead to deformations of plaquette and vertex operators at the boundary, respectively. See figure 2.6 for examples of deformed stabilizers at a boundary.

The surface code with open boundaries as defined in this thesis supports one logical qubit¹. The logical operators are once again defined by homologically non-trivial string-like X (or Z) operators with no boundary. To satisfy this requirement, they must begin and end at the boundaries of the lattice defining the code. The logical $\bar{X}$ operators connect the two opposing smooth edges, whereas the logical $\bar{Z}$ operators connect the two rough edges. An example of each is shown in figure 2.6.

Once again we restrict the discussion to X errors, since results for Z -type errors will be analogous. A single X -type error is detected by two adjacent plaquettes, except when it occurs on a smooth boundary, see figure 2.6. Similarly, a contiguous chain of X errors ending at a smooth boundary gives rise to only one nontrivial syndrome, see figure 2.6

As we have outlined in this section, the differences between the toric and surface code models are not substantial. The recovery procedure is conceptually identical, and many decoders developed for the toric code can be readily adapted for the surface code.

In the next section we shall introduce the qudit surface code, that is, the surface code constructed using d -level quantum systems. As we shall see, many of the key ideas carry over.

¹A variant of the planar code allows an arbitrary number of qudits to be encoded in it, as pairs of holes (a hole is a location at which a stabilizer generator is not measured). This is a useful model when considering computation using the surface code [Bravyi and Raussendorf, 2007].

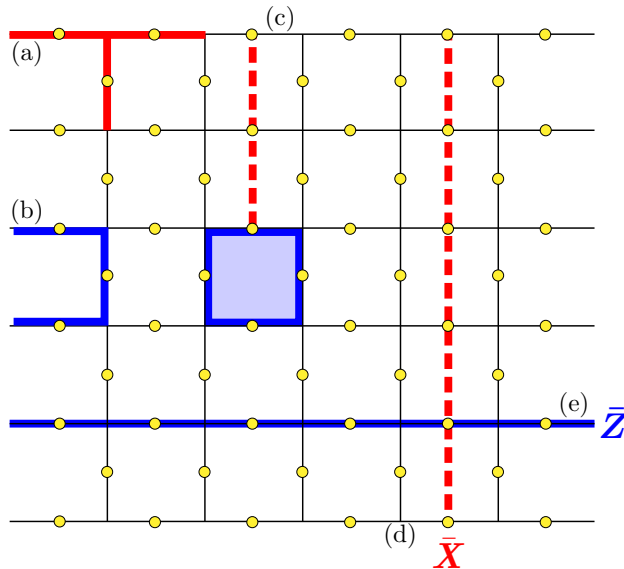


Figure 2.6: The surface code. As illustrated in this figure the top and bottom boundaries are smooth, while the left and right boundaries are rough. (a) An example of a deformed edge vertex operator. (b) An example of a deformed edge plaquette operator. (c) A string of X errors connecting to a boundary and hence giving rise to only one nontrivial syndrome. (d) An example of a logical X operator. (e) An example of a logical Z operator. The two logical operators have only one qubit in common and as a result they anticommute.

2.3.1 The Qudit Surface Code

The surface code can naturally be generalised to higher dimensional systems. Indeed, this generalisation is already present in Kitaev's seminal paper [Kitaev, 2003]. Despite the fact that we have introduced the surface code, for completeness we shall also provide an overview of the qudit planar code. For further details of the qudit construction see [Bullock and Brennen, 2007]. Recall that an introduction to the basic idea of qudit error correcting codes was given in Section 1.5.

We now identify d -level quantum systems, or qudits, with the edges of the same lattice introduced in Section 2.3. The qudit surface code is a stabilizer code with

two types of stabilizer generators $\mathcal{S} = \langle A_V, B_P \rangle$ defined on the lattice as

$$A_V = X_{e_1} \otimes X_{e_2}^{-1} \otimes X_{e_3}^{-1} \otimes X_{e_4} \quad \forall e \in V, \quad (2.3)$$

$$B_P = Z_{e_1}^{-1} \otimes Z_{e_2}^{-1} \otimes Z_{e_3} \otimes Z_{e_4} \quad \forall e \in P, \quad (2.4)$$

where $e \in V$ are the edges surrounding a vertex v of the lattice and $e \in P$ are the edges contained in a plaquette P , see figure 2.7. Once again A_V are known as the vertex operators and B_P as the plaquette operators. An example of each is shown in figure 2.8(a) and (b). As before, the rough and smooth boundaries lead to deformations of plaquette and vertex operators, respectively. See figure 2.8(c) for an example of a deformed stabilizer at a boundary.

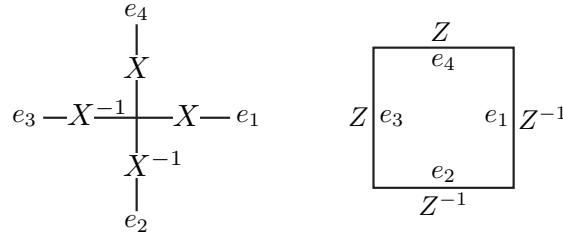


Figure 2.7: Vertex operator (left) and plaquette operator (right) of the qudit surface code.

The qudit surface code supports one *logical qudit*. An example of each type of logical operator is shown in figure 2.8(d) and (e). These operators, together with the stabilizer group, generate the group of logical qudit Pauli operators which map the code space to itself. As in the qubit case, the distance of this code is L .

Physical errors are detected by measuring the stabilizers, with X -type¹ and Z -type errors detected independently by the plaquette and the vertex operators, respectively, allowing us once again to restrict ourselves to a discussion of X -type errors only. A single X -type error is detected by two adjacent plaquettes, except when it occurs on a smooth boundary, see figure 2.9(a). A string of X -type errors is detected by plaquettes contiguously along the path of the string, as shown by the example in figure 2.9(b). This is in contrast to the qubit ($d = 2$) case where only

¹This is shorthand for errors of the type X^j for $j \in \mathbb{Z}_d \setminus \{0\}$, and similarly for errors we refer to as ‘ Z -type’.

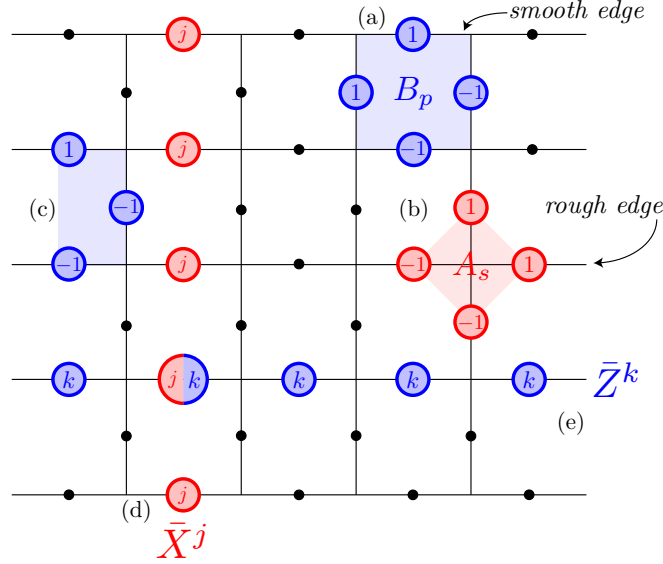


Figure 2.8: An example of a distance 5 qudit planar surface code. Qudits are shown as black dots, arranged on the edges of a lattice with two types of boundary: rough and smooth. For clarity, when an arbitrary X^j or Z^k Pauli operator acts on a physical qudit, we only include the exponents j and k on the edges of the figure. We use red for X^j errors and vertex operators, and blue for Z^k errors or plaquette operators. (a) and (b) An example of a single plaquette and vertex operator, respectively. (c) An example of a deformed rough edge plaquette operator (3-body operator). Note that the vertex operators are deformed at smooth edges. (d) and (e) An example of a pair of ω -commuting logical operators.

the end-points of the string give rise to non-trivial plaquette measurements. This observation suggests that in higher d the syndrome reveals more information about the path of the errors on the lattice. Indeed, it is this information that, if exploited correctly by the decoder, can lead to improved error correction performance, as shown by their higher threshold values, as d increases.

As explained in Section 2.2, generally speaking the aim of the decoder is to use the information given by the syndrome to return a correction operator that restores the code to its original state. In the next section we shall discuss the decoding problem and introduce the concept of *optimal decoding*. This in turn will motivate the requirement for alternative decoding strategies. One decoding algorithm is studied in detail in Chapter 4.

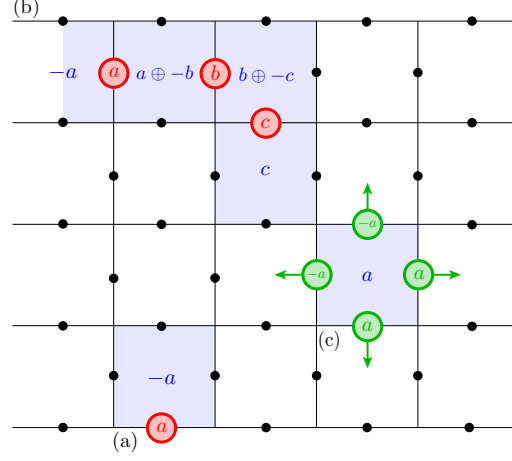


Figure 2.9: Examples of X -type errors and the syndrome transportation rule. (a) A single boundary error is only detected by one plaquette. (b) An arbitrary string of three errors and the corresponding intermediate plaquette measurement outcomes. (c) An example of how to transport the plaquette with outcome ‘ a ’ in any of the indicated directions by applying the relevant X -type operator shown in green.

2.4 The Decoding Problem

In this section we shall introduce the decoding problem for the qubit surface code. The related problem for the qudit surface code will be considered in some detail in Chapter 4.

Two important requirements for a decoder are that it should run in an efficient time and that it should have a high threshold, p_c . The requirement for efficient decoding is discussed further in Chapter 4 but essentially means that the time it takes to run should scale polynomially (or less) in the size of the input. The input to a decoder is the syndrome, the size of which depends on the number of check operators associated with the code, which in turn depends on the code distance, L . For error rates $p < p_c$ it is desirable that the logical failure rate should drop quickly – preferably exponentially – as L increases.

In practice however, for the surface code the information available (the syndrome) is insufficient to completely characterise the error that has occurred. A large number of different error configurations can lead to the same syndrome pattern, making the identification of the error, and hence the identification of a suitable correction chain,

difficult. An example of this syndrome degeneracy for the qubit case is illustrated in figure 2.10.

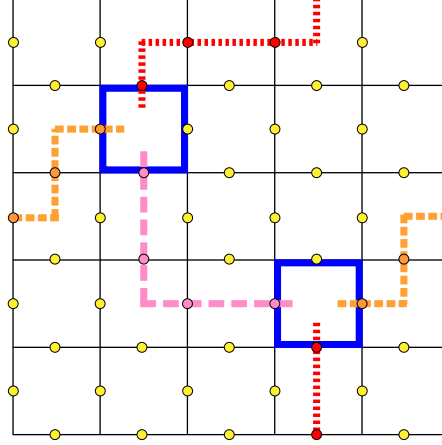


Figure 2.10: Non-trivial syndromes, indicated in blue, can be caused by many different underlying errors. Three different possible error chains giving rise to the same syndrome are indicated with different dashed lines. In total 2^n error chains exist for every syndrome configuration.

Fortunately we do not need to identify the exact error in order to mitigate its effect. Recall that all elements of the stabilizer act as the logical identity operator, hence if we apply a correction E' such that $\partial E' = \partial E$ and $E + E' \in \mathcal{S}$ then although the error chain has not been eliminated, it has no logical effect and the encoded quantum state is unaffected. Therefore the goal of a decoding algorithm should be to identify a correction operator in the same homology class h_i as the error.

2.4.1 Optimal Error Correction

The threshold p_c is the highest single-qubit error rate that can be tolerated by an error correcting code. Any error rates $p > p_c$ will cause the logical failure rate of the code to increase, rather than decrease, with increasing code distance. As such it is an important task to find the threshold for any combination of error correcting code, error model and decoding strategy. For simplicity we will restrict our discussion in this section to the qubit surface code.

The highest possible threshold for a fixed code and error model is known as the *optimal threshold* and is achieved by a *maximum likelihood* decoder. The idea behind this type of decoder is to identify the homology class that the original error belongs to. The decoder begins by identifying a chain of operators, E' , such that $\partial E'$ is consistent with the observed syndrome. We shall call this chain the *correction chain*.

The error distribution associated with an independent noise model¹ applies an X error to each qubit with probability p and the identity with probability $1 - p$, followed by Z errors applied with the same distribution. Therefore the probability of the correction chain E' occurring is

$$P[E'] = (1 - p)^{n - |E'|} (p)^{|E'|}. \quad (2.5)$$

Recall that n is the total number of physical qubits comprising the code and $|E'|$ is the weight of the correction chain, or the number of physical qubits on which it acts nontrivially.

This correction chain can then be multiplied by each of the elements of the stabilizer group $\mathcal{S}$ in turn and the corresponding probability computed, so that the probability of the correction chain belonging to the homology class h_i is given by

$$P[E' \in h_i] = \sum_{S \in \mathcal{S}} P[E' S C_{(i)}], \quad (2.6)$$

where $C_{(i)}$ is an operator belonging to the i th homology class. For the trivial homology class h_0 we set $C_{(0)} = I$. The identification of the value $P_{\max}(E') = \max_i P[E' \in h_i]$ indicates that in order to maximise the likelihood of successful correction the original choice of E' should be modified by $C_{(i)}$ before it is applied as the correction operator.

Until recently no efficient implementation of an optimal decoder was known. This necessitated the optimal threshold to be calculated using numerical techniques. The optimal threshold was originally found using a mapping from the toric code to a statistical mechanics model called the *random bond Ising model* (RBIM), and considering the phase transition of this model [Dennis et al., 2002]. Indeed, such a relationship to condensed matter and statistical physics is one of the hallmarks

¹Although this noise model might appear simplistic it is commonly used for benchmarking decoders for CSS codes.

of topological quantum error correcting codes. The optimal threshold value of the independent noise model that we consider throughout this thesis has been found using such techniques to be $p_c = 0.1093$. This calculation has been performed many times, for example see [de Queiroz, 2009; Honecker et al., 2001].

Last year however, an exact efficient algorithm for the independent noise model was discovered [Bravyi et al., 2014]. This implementation of the optimal decoder found a threshold of $0.109 \leq p_c \leq 0.11$, a value consistent with the results from the mapping to the RBIM¹. In the next section we shall give a brief overview of the mapping from the toric code to the RBIM and how it can be used to calculate the optimal threshold of the toric code.

2.4.2 Mapping to the Random Bond Ising Model

The mapping from the toric code to the RBIM is explained in detail in [Dennis et al., 2002], here we summarise the key ideas. We begin by selecting an initial error configuration according to the error distribution described in Section 2.4.1, labelled E . We employ an indicator function q_e to show whether the qubit corresponding to the lattice edge e has suffered an error or not. The indicator function takes the value

$$q_e = \begin{cases} +1 & \text{if there is no error on the qubit,} \\ -1 & \text{if there is an error on the qubit.} \end{cases} \quad (2.7)$$

The mapping proceeds by overlaying an Ising model with nearest neighbour couplings on top of this lattice, so that the classical spins are associated with the vertices of the toric code, see figure 2.11. The couplings between two spins σ_i and σ_j are denoted by $\tau_{ij} = \pm 1$, where $\tau_{ij} = +1$ indicates a ferromagnetic coupling, meaning the spins will tend to align, and $\tau_{ij} = -1$ indicates an antiferromagnetic coupling, meaning the spins will tend to anti-align. We now enforce the condition $\tau_{ij} = q_e$, leading to a quenched random configuration of couplings in the model, with the spins tending to align except for those along the path of E which will tend to anti-align.

¹For other noise models, such as the depolarising noise model in equation 1.2, there is no efficient optimal decoder known at present, and approximate decoding methods must be used. Nonetheless, high thresholds have been reported, see for example [Bravyi et al., 2014; Duclos-Cianci and Poulin, 2010b; Harrington, 2004].

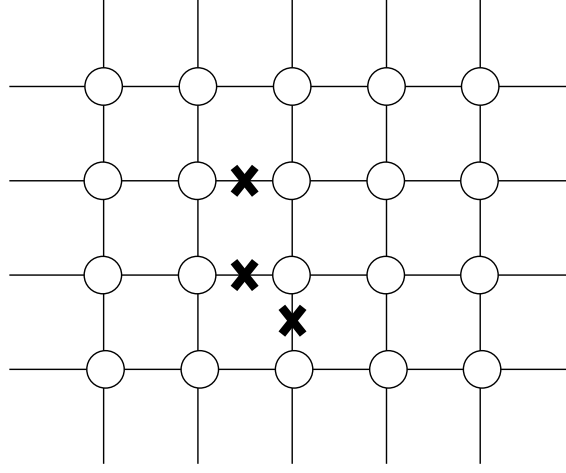


Figure 2.11: An example error configuration on the toric code is indicated by crosses, and the classical spins corresponding to the random bond Ising model are shown as a lattice of circles.

The end-points of E introduce frustrations in the Ising model that can only be removed by the introduction of *domain walls*. These are excited bonds whose underlying coupling is ferromagnetic but the spins are forced to anti-align. With the frustrations eliminated in this way, there are now regions defined by the quenched random antiferromagnetic bonds and the domain walls, within which the spins are flipped relative to the bulk, see figure 2.12.

The fluctuations of the domain walls are described by the random bond Ising model, a statistical mechanics model which has the partition function

$$Z(J, \tau) = \sum_{\{\sigma\}} \prod_{\langle i, j \rangle} e^{J\tau_{ij}\sigma_i\sigma_j}. \quad (2.8)$$

The sum is taken with respect to all spin configurations and the product is over all nearest neighbour pairs. Note that $2J = \beta$, the inverse temperature of the system, so that $\beta = 1/T$. This is set by the toric code error probability distribution

$$\beta = -\ln \frac{p}{1-p}, \quad (2.9)$$

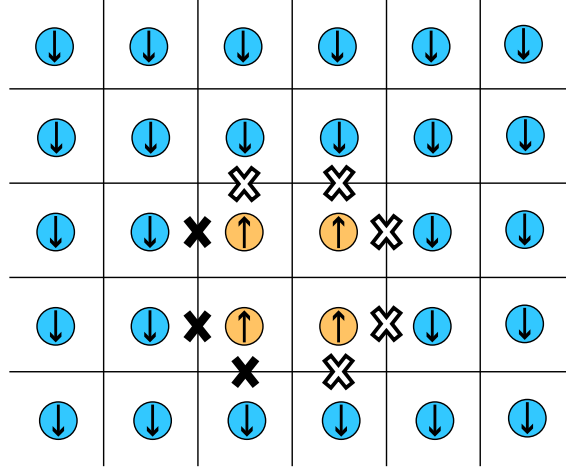


Figure 2.12: The dual toric code lattice is now shown. The error chain E , corresponding to fixed antiferromagnetic couplings, is indicated as filled crosses. The correction chain E' , corresponding to excited bonds, is indicated as open crosses. The classical spins are given nominal values and coloured according to these values for clarity. Note that if all of the arrows were reversed the figure would remain equally valid.

where p is the single-qubit error probability. This relationship defines the Nishimori line [Nishimori, 1981], and ensures that the original probability distribution informs the Boltzmann weight of states in the statistical mechanics model.

As the probability of antiferromagnetic bonds increases, the total number of quenched antiferromagnetic couplings in the system increases. The resulting domain wall fluctuations occur on a length scale up to the correlation length and the percentage of spins that are flipped increases. At the critical point T_c the correlation function diverges and the system undergoes a phase transition from an ordered to a disordered phase. The critical temperature can be determined numerically for different values of p via equation (2.9). This defines the order-disorder transition in the $p - T$ phase diagram, and the point where this transition coincides with the Nishimori line corresponds to the optimal threshold of the toric code.

In the next chapter we make use of this mapping, and other techniques, to construct a scaling hypothesis for the logical failure rate of the toric code as the bit-flip error rate p falls below the threshold. This is a quantity of interest because in more sophisticated models (for example the fault-tolerant model in which all of

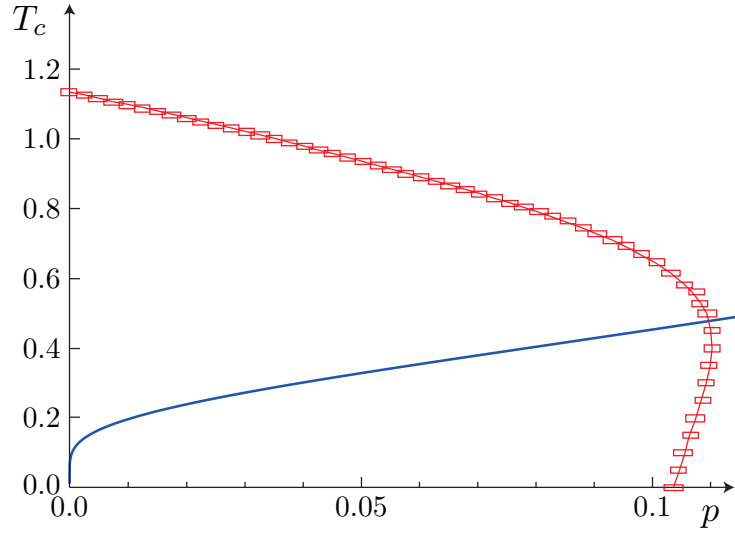


Figure 2.13: The data in red shows the transition from the ordered to disordered phase of the RBIM. The blue curve is the Nishimori line, so where the two curves meet is the optimal threshold for the toric code. Data taken from [Andrist et al., 2015].

the circuit elements used to perform the error correction are assumed to be noisy) it can be inverted to find estimates for the overhead, measured in terms of the physical size of the code or the number of qubits comprising it.

Chapter 3

Logical Error Rate Scaling of the Toric Code

To date, a great deal of attention has focused on characterising the performance of quantum error correcting codes via their thresholds, the maximum correctable physical error rate for a given noise model and decoding strategy. Practical quantum computers will necessarily operate below these thresholds meaning that other performance indicators become important. In this chapter we consider the scaling of the logical error rate of the toric code and demonstrate how, in turn, this may be used to calculate a key performance indicator: the overhead of the code. We use a perfect matching decoding algorithm to find the scaling of the logical error rate and find two distinct operating regimes. The first regime admits a universal scaling analysis due to a mapping to a statistical physics model. The second regime characterises the behaviour of the logical error rate in the limit of small physical error rate and can be understood by counting the error configurations leading to the failure of the decoder. We present a conjecture for the ranges of validity of these two regimes and use them to quantify the overhead – the total number of physical qubits required to perform error correction.

Some of the results in this chapter are not yet published. Many of the techniques were developed in an earlier version of this analysis [[Watson and Barrett, 2014](#)] which focussed on a simpler noise model, not including measurement errors.

3.1 Introduction

Topological quantum codes built from qubits play a central role in architectures for fault-tolerant quantum computing at the forefront of current research [Devitt et al., 2009; Fowler et al., 2012a]. The surface code and the related toric code introduced in Chapter 2 are prominent examples of such codes. Compared with other quantum error correcting codes, they possess the key experimental benefit of requiring only local interactions and yet, under realistic noise models, they have been shown to achieve the highest reported fault-tolerant thresholds [Stephens, 2014; Wang et al., 2011].

As discussed in Chapter 2, a high threshold is a very desirable property for an error correcting code since for all error rates below the threshold, increasing the number of physical qubits encoding the quantum information reduces the logical error rate. In a realistic setting the code must operate at an error rate well below the threshold. Other quantities then become important to characterise the performance of a quantum computer, for example the *code overhead*, the number of physical qubits comprising the code that are required to adequately protect the encoded quantum information. This is an important consideration for the practical implementation of fault-tolerant quantum computation and has recently begun to draw some attention [Bravyi and Vargo, 2013; Gottesman, 2013; Suchara et al., 2013].

The logical failure rate of the error correction, denoted here as P_{fail} , is a key metric of the performance of a code, since it describes the likelihood of failing to protect the encoded quantum information. In this work we seek closed-form expressions for the logical failure rate of the toric code for fixed code distance and physical error rate, p . The code distance is the minimum length of a logical operator that has a non-trivial effect on the code space, and in the case of the toric code such operators have a length equal to the lattice size L .

As a test-bed for this study, we choose the toric code. Although not a candidate for direct physical realisation it provides a simplified setting which nevertheless captures the key behaviours of topological codes. An additional benefit is that it is closely related to other, more physically realistic systems. We expect therefore that results for the logical error rate scaling of the toric code could be applied in a range of other physical systems – most obviously with noisy syndrome measurements, a

scenario that we shall also examine in this chapter. The techniques to determine the scaling of the logical error rate should also be analogous for the related planar code (with open, rather than periodic, boundary conditions), although the numerical results would be expected to differ from the toric code case. Furthermore, once the scaling has been determined it can be used to calculate the fault-tolerant overhead for the planar code using the methods presented in this chapter.

Below the threshold, the logical failure rate of a topological code is expected to reduce exponentially as we increase the code distance [Dennis et al., 2002]. Although the code performance improves rapidly with increasing L , in the lattice of the toric code the total number of physical qubits scales as $O(L^2)$. Manufacturing, storing, and manipulating resources with such a scaling is a non-trivial task with technology available at present. We should then ask not simply how large we can make the code, but how many physical qubits are required to achieve a desired error correction performance.

In order to answer this question, we examine the behaviour of the toric code in the presence of uncorrelated bit-flip and phase-flip noise. We numerically simulate the error correction procedure and use this to find closed-form approximate expressions for the failure rate as a function of the input parameters L and p and find two operating regimes. The first of these, which we will call the *universal scaling hypothesis* regime, extends ideas by Wang *et al.* [Wang et al., 2003] and uses rescaling arguments based on the mapping to the 2-dimensional random bond Ising model, or RBIM, introduced in Section 2.4.2. This approach provides a good estimate for P_{fail} when the error weight (the number of qubits an operator acts on non-trivially) is high and code distance is large.

Rescaling arguments apply in the thermodynamic limit, and close to criticality, where the correlation length of the RBIM also diverges and the appropriate length scale is the ratio of the lattice size to the correlation length, L/ξ . As p decreases there is a point at which finite-size effects begin to dominate and we no longer expect the universal scaling hypothesis to apply. This limit corresponds to low physical error rates, as well as small lattices.

The second approach to studying the scaling behaviour of the logical failure rate extends ideas by Raussendorf *et al.* [Raussendorf et al., 2007] and Fowler *et al.* [Fowler et al., 2012a] to find an analytic expression for P_{fail} in the limit $p \rightarrow 0$.

When the error weight is low and the code distance is small this expression gives a good estimate of the logical failure rate. We will refer to this as the *low p expression*.

Although we know the limits in which each of these approaches is valid, we would like to make some quantitative statements about the range of parameters for which each is applicable. We shall present a heuristic argument for the range of L and p for which each regime gives a good approximation to the numerical data.

3.2 Logical Failure Rates in Classical Coding Theory

Classical information theory was born with the publication of Shannon's seminal work in 1948 [Shannon, 1948]. The original formulation was in terms of communication through noisy channels – a situation that applies equally well in attempting to preserve information as it passes through a physical channel in space, or through a temporal channel while stored in an error correcting code.

At its most basic level, the classical error correction model protects bits against flipping from 0 to 1 and vice versa. Of course, many sophisticated error models exist, and other errors are possible (for example erasures) depending on the physical system under investigation. Indeed, classical error correcting codes also have tools that have no parallels in the quantum world, such as cloning and non-destructive measurement of unknown states, as discussed in Chapter 1. Nonetheless, there have been techniques for analysing and quantifying codes developed in the classical paradigm that are interesting to study in quantum codes.

In classical information theory it is known that for the maximum likelihood (optimal) decoder, the block error probability P_e (the probability of unsuccessful decoding) decreases exponentially as the block size (number of bits comprising the code) is increased. This can be expressed as

$$P_e \leq e^{-LE(p)}, \quad (3.1)$$

where $E(p)$ is called the *error exponent*. If we examine the dependence of the code failure rate on the block size, we would expect for a maximum likelihood decoder

that P_{fail} in the region where $p < p_c$ depends exponentially on the code block length L and on some function of p , the error rate [Richardson and Urbanke, 2008].

It has also been shown that for maximum likelihood decoders in low-density parity-check (LDPC) codes¹, in the limit of large block size L , P_e is a function of $z = L^{1/\nu}(C - R)$. This can be expressed as $P_e = f(z)$ where $f(z)$ is known as the *scaling exponent*. Here C is the code capacity – the maximum error rate at which information can reliably be stored, or the classical analogue of the threshold. The rate R is the number of uncorrupted qubits per physical qubit². Crucially, this law holds in the so-called ‘waterfall region’, where the contribution to P_e is dominated by failures that occur on large length scales [Amraoui et al., 2009; Mondelli et al., 2013].

Interestingly, as well as the region of behaviour described by the scaling exponent, there is a second region of behaviour of the code: the *error floor* region. Here small-scale errors, that is, the errors that represent the minimum distance of the code, dominate [Richardson et al., 2002]. This happens when the error rate $p \rightarrow 0$ and refers to what we will call the low error rate regime.

In this chapter we shall be interested in finding both the error exponent and the scaling exponent in the case of the toric code, as well as a description of the error floor region. These are illustrated in figure 3.1, where a schematic of the expected behaviour of the code as rescaled function of L and p is shown.

In the asymptotic limit of large lattice sizes, the failure of the code is expected to drop exponentially in L and the behaviour is dominated by large-scale errors. This corresponds to the waterfall region and is indicated by the black curve³.

For finite lattice sizes, at some point the small-scale errors dominate and the behaviour is described by the error floor region, where we have shown only the asymptotic approximation for this region for $L = 5$ in figure 3.1. We also expect a crossover region of behaviour, where neither of the asymptotic descriptions match the data well. In this chapter we shall study these two regimes, which may be considered as bounds on the region of interest. We shall concern ourselves with determining

¹For our purposes this means that the number of bits involved in each parity check operator is small with respect to the total number of bits in the code, and preferably constant.

²In the quantum setting presented in this chapter, the quantity of interest is the physical error rate, which is simply the complement of the rate in the quantum setting.

³The data in the figure has been rescaled using $x = (L/\xi)^{1/\nu_0}$, see section 3.4 so it does not immediately resemble the classical picture.

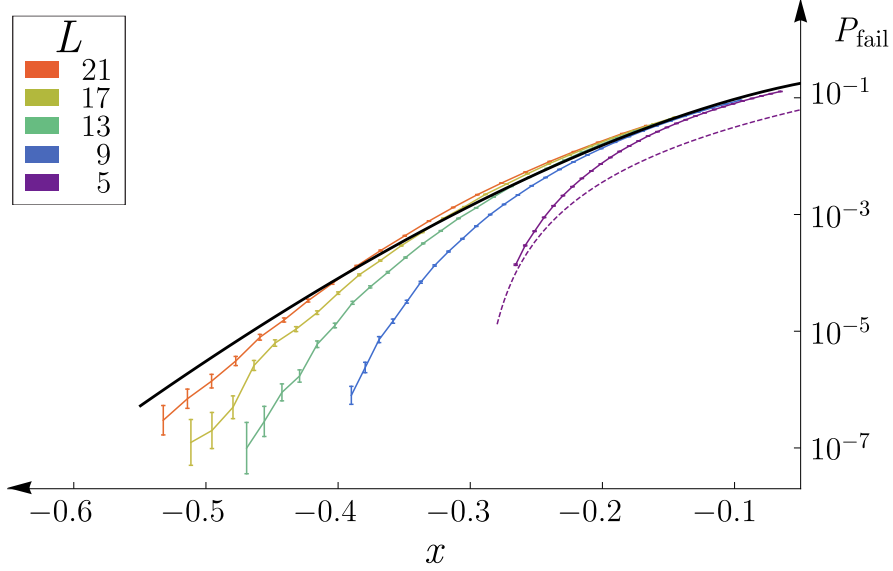


Figure 3.1: A schematic of the logical error rate scaling of the toric code, plotted as a function of the rescaled variable $x = (L/\xi)^{1/\nu_0}$. The universal scaling behaviour in the limit of large L is indicated in black, while the asymptotic behaviour in the limit of small physical error rates is plotted as a dashed line for $L = 5$ only. We observe that for finite lattice sizes the data is well-approximated by the universal scaling curve for a range of x , before breaking away from it and tending towards the ‘low p ’ regime.

over what parameter ranges the scaling laws provide a good approximation to the behaviour of the code, and we shall also seek to determine the functional forms for the two exponents of interest.

3.3 Noise Models for the Toric Code

In Section 2.2 we introduced the toric code. Here we shall describe the two noise models of interest and the techniques used for simulating them. One noise model is the simple independent noise model in Section 2.4.1 while the second noise model, that we shall refer to as *phenomenological*, incorporates the notion of noisy syndrome measurements. The phenomenological model is not a full gate error simulation, but suggests that the techniques we develop in this chapter would be applicable to such a detailed error model.

3.3.1 Error Correction

Optimal decoding, and the difficulties therein, for the toric code was discussed in Section 2.4.1. Several classes of sub-optimal efficient decoding algorithm exist, for example the soft-decision renormalisation group (SDRG) decoder by Duclos-Cianci and Poulin [Duclos-Cianci and Poulin, 2010a,b] or the hard-decision renormalisation group (HDRG) decoder by Bravyi and Haah [Bravyi and Haah, 2013]. We study a variant of the HDRG decoder in detail in Chapter 4. The one used in this chapter is a version of Edmonds' minimum-weight perfect matching algorithm (MWPMA) [Cook and Rohe, 1999; Edmonds, 1965]. This algorithm pairs the non-trivial syndromes via a correction chain that has the least weight possible while satisfying the condition that its boundary matches the error chain boundary, i.e. $\partial E = \partial E'$. This ensures that the total operator, $C = E + E'$, is a cycle. We denote the threshold for the MWPMA by p_{c0} . Numerical simulations suggest that $p_{c0} = 0.1031 \pm 0.0001$ [Wang et al., 2003].

Although this algorithm gives a high threshold [Wang et al., 2003], we shall consider a heuristic modification described in detail by Stace and Barrett [Stace and Barrett, 2010], that includes the effects of the degeneracy of E' and can give thresholds up to $p_{c0} \approx 0.106$. Degeneracy counts the number of possible paths that the chain can take, given that its boundary and weight are fixed. Matchings with higher degeneracy have a higher probability of arising so they may be *a priori* more likely than some matchings with a lower weight.

The degeneracy itself is simple to calculate for a given (minimum-weight) matching. For a path m between two non-trivial syndromes, a and b , the degeneracy of that path D_m is defined as the number of different combinations of the links in the matching. The product of all individual D_m is the total degeneracy of the matching, D_M .

To take degeneracy into account we compute the matching using the MWPMA, where the edge weights d_{ab} are modified by the effect of the degeneracy of that path. Then the weight passed to the algorithm becomes $d_{ab} - \tau \ln D_m$. Here τ is a weighting that we assign to the degeneracy term. The degeneracy is added in such a way due to entropic considerations, see [Stace and Barrett, 2010] for details. The decoding algorithm minimises the weighted degeneracy globally and this has been shown to lead to an improved threshold [Stace and Barrett, 2010]. We refer to this enhanced

version of the minimum-weight perfect matching simply as the PMA decoder.

3.3.2 Simulating Noise and Error Correction in the Case of Perfect Syndrome Measurements

An important tool in this thesis is the numerical simulation of the detection and correction of errors on a toric code. Repeating random trials allows us to examine the failure probability of the code over a wide range of parameters. As stated earlier, we consider uncorrelated bit-flip and phase-flip errors arising at a rate p . Although the independent noise model is not likely to be particularly close to the noise which occurs in physical systems, it has the advantage that it allows X and Z errors and their correction to be modelled independently. This so-called *uncorrelated noise model* is thus the standard noise model used to benchmark decoders.

The behavior of the toric code is simulated by placing an error with probability p on each individual qubit of the toric code lattice of linear dimension L , giving rise to a (usually disjoint) error chain E . The syndromes are measured¹ and the PMA decoder is used to determine the correction chain E' . These correction chains are added, modulo 2, to E and a parity check with each of the appropriate logical operators is used to determine the homology class of the total operator C . The result of this random sample indicates whether the error correction succeeds or fails.

The outcome of the Bernoulli trial (a single simulation of error correction) is assigned the value $n_f = 0$ if C is in the trivial homology class and $n_f = 1$ if it is in any of the non-trivial homology classes. To gather statistics we repeat this procedure N times for the same input parameters (L, p) . Of these N trials, $N_f = \sum_{i=1}^N n_{f,i}$ will have failed to perform error correction successfully. We therefore estimate the error correction failure probability as $P_{\text{fail}} \simeq N_f/N$ and the variance of such a distribution is $\sigma^2 \simeq P_{\text{fail}}(1 - P_{\text{fail}})/N$. The resulting data $P_{\text{fail}}(L, p)$ characterises the toric code performance.

¹For this simple error model we assume that the stabilizer measurements can be made perfectly. It follows that if any non-trivial eigenvalues are observed, this indicates the presence of errors with certainty.

3.3.3 Simulating Noise and Error Correction in the Phenomenological Noise Model

Since we will once again be considering the case where bit-flip and phase-flip errors are uncorrelated, it suffices to consider bit-flip or phase-flip noise models independently. However, in the phenomenological noise model we assume the stabilizer measurements are noisy – that is, their measured eigenvalues are incorrect with some probability q . We shall arbitrarily describe the method for simulating phase-flip noise and errors on the vertex operators A_V but by considering the dual lattice the same arguments apply for bit-flip noise. We shall explain our simulation methods briefly here, and then more fully for the qudit model in Chapter 4.

The strategy developed to cope with noisy stabilizer measurement values is to repeat the measurements in order to obtain more information [Dennis et al., 2002]. In the literature it is common to test decoders with a simple error model described by a single parameter p so that the probability of error in the syndrome measurement $q = p$. For ease of comparison, we shall follow this convention and use the same error model here. In this symmetric noise model where the physical noise rate and measurement noise rates are equal, we repeat the syndrome measurements L times, where L is the linear size of the lattice.

Between each round of syndrome measurements we assume that each physical qubit is independently subject to an error channel which applies error operator X with probability p , followed by an error channel which applies error operator Z with equal probability p . We then assume that the outcome of each syndrome measurement $j = \pm 1$ undergoes an error which maps j to $(-1)^k j$ for $k = \{0, 1\}$ with probability p .

We estimate the threshold via a Monte Carlo simulation. We shall study a distance L code for a variety of values of L . The simulation proceeds by first generating a 3D data structure of L time steps of the accumulated history of the physical qubit errors and the measurement errors. The corresponding syndrome measurement outcomes, taking into account both of these error sources, are then computed. In order to achieve the close analogy for the relationship between errors and syndromes in the 2D measurement-error-free and 3D general case, Dennis et al [Dennis et al., 2002] showed that it is most convenient to represent the history of the syndrome outcomes

as a 3D grid of *syndrome changes* from one time step to the next.

This 3D syndrome changes data is the input to the decoder. The PMA algorithm as described in Section 3.3.1 can be used directly in this 3D data structure, taking the syndrome changes data and returning a 3D correction operator F' . To convert this to a physical correction operator that can be applied in 2D we ignore time-like corrections since they are unphysical. We can combine the two-dimensional layers corresponding to each time step by taking a product of the operators on each qubit at each time step, projecting the 3D correction operator into 2D. This 2D correction operator E' corrects the accumulated errors at the last time step of the toric code. We say the decoder has succeeded when the sum (modulo 2) of the accumulated errors on the qudits and the returned correction operator is within the stabilizer of the code.

If we are operating below threshold then following the 3D decoding we expect almost all of the errors to have been corrected. It is possible however that a small number of errors will remain after the fault-tolerant decoding has been performed¹. In a realistic setting the error correction would proceed in this way, eliminating all but a small number of errors in each block of L time steps. At the point when the state is read out, these small errors can be accounted for by taking a majority vote on the measurements of the logical operators. For the purposes of the simulation however, we need to determine whether the fault-tolerant decoder has introduced a logical error. The conventional way to overcome this problem is to perform an additional round of error correction in 2D with noise-free syndrome measurements, after which we can be certain that all the errors are corrected and a parity check will reveal whether any logical errors have been introduced. A flow diagram for the steps of the decoder is shown in figure 3.2.

This whole procedure is repeated and the variance calculated in the same way as described in Section 3.3.2. Since the run time of the implementation of the PMA algorithm we used scales as $O(n^3)$, where n is the number of nodes in the graph, the lattice sizes that can be investigated when the syndrome measurements are noisy is reduced when compared with those that can be examined when the syndrome measurements are noise-free.

¹These generally arise from physical errors in (or close to) the final time step that are paired to the time-like boundary.

3.4 The Universal Scaling Hypothesis

In Chapter 2 we saw that there is a close relationship between the toric code and a statistical mechanics model. Indeed, in [Wang et al., 2003], Wang *et al.* used ideas from the theory of critical phenomena in finite-sized systems to show that there is a critical point in the failure probability of the toric code with noise-free syndrome measurements. To do this, they used a mapping from the toric code to the 2-dimensional random bond Ising model (RBIM), as already described in Section 2.4.2. The probability distribution of antiferromagnetic couplings in this model matches the probability distribution of errors in the toric code, hence a mapping between the two models can be constructed [Dennis et al., 2002; Harrington, 2004]. The RBIM has been extensively studied and it is known to undergo a phase transition from an ordered to a disordered phase as the concentration of antiferromagnetic bonds is increased. This implies a phase transition in the corresponding quantity of the toric code: its logical failure rate.

Wang *et al.* demonstrated that for the regime where $L \gg \xi$, where $\xi = |p - p_{c0}|^{-\nu_0}$ is the RBIM *correlation length*, we expect scale-invariant behavior. This argument leads to the conjecture that in this regime the failure probability of the toric code is a function only of L/ξ [Wang et al., 2003].

Below the threshold the failure rate has been found to depend exponentially on the system size [Dennis et al., 2002; Raussendorf et al., 2007], and also more generally in the presence of measurement errors [Fowler, 2012], so that

$$\ln P_{\text{fail}} \propto -L. \quad (3.2)$$

Numerical evidence for this will be provided later, in figure 3.4.

Together, the exponential dependence on L and the scaling hypothesis fix the functional form of P_{fail} to be

$$P_{\text{fail}} = Ae^{-a(L/\xi)} \quad (3.3)$$

$$= Ae^{-a|p-p_{c0}|^{\nu_0}L}. \quad (3.4)$$

In this expression A and a are constants that can be determined using numerical fitting techniques, see Sections 3.5 and 3.8.1.

In practice the toric code will be operating in the correctable ($p < p_{c0}$) regime

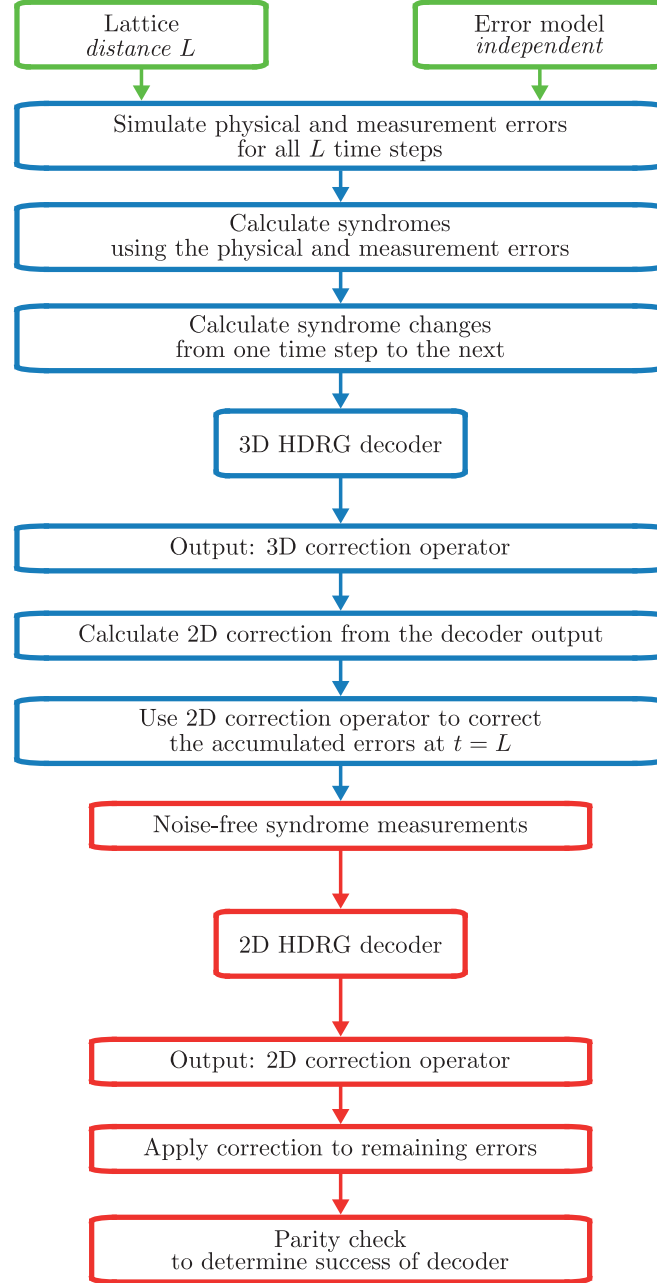


Figure 3.2: Flow diagram for the 3D HDRG decoder. The inputs to the model are coloured in green, with the main steps in the simulation of the 3D decoder in blue. Before the decoder is implemented, errors are simulated and the check operator outcomes determined. Syndrome changes are calculated, resulting in the effective localisation of syndromes to pairwise locations. The correction operator from the 3D decoder is projected into 2D to correct the error. An additional 2D decoder assuming perfect syndrome measurements, necessary only to assess the performance of the 3D decoder, is shown in red.

so we use the rescaled variable $x = (L/\xi)^{1/\nu_0}$ (alternatively this may be written as $x = (p - p_{c0})L^{1/\nu_0}$) and we can rewrite the universal scaling hypothesis as

$$P_{\text{fail}} = Ae^{-a|x|^{\nu_0}}. \quad (3.5)$$

We determine the values of A , p_{c0} and ν_0 from a fit to data close to the threshold, see Section 3.5.

In [Wang et al., 2003] there is also a statistical mechanics model described for the toric code in the case where the syndrome measurements are subject to noise, called the random plaquette gauge model (RPGM). It leads to the same universal scaling hypothesis as the one given in equation (3.4). Therefore we can use the same functional form for the ansatz for the scaling hypotheses of the two models.

3.5 Determining the Threshold

In order to determine the scaling exponent and validity of the universal scaling hypothesis, we must first establish the values of the threshold, p_{c0} , and critical exponent, ν_0 . The universal scaling hypothesis, equation (3.3), also relies on knowing the failure rate at threshold in the large L limit. In this section we show how these quantities are obtained from a fit to data close to the threshold.

3.5.1 Perfect Syndrome Measurements

The threshold for the stand-alone MWPMA decoding for the perfect syndrome error model has been calculated previously as $10.306 \pm 0.008\%$ [Wang et al., 2003]. Since we allow the degeneracy of the matching to affect the choice of correction chain, we repeat the calculation to obtain the threshold for our enhanced PMA decoder.

To find the logical failure rate P_{fail} we numerically simulate the error correction protocol, enhanced minimum-weight perfect matching (PMA), using the same method described in Section 3.6. We performed $N = 10^6$ simulations of the error correction procedure for p close to 10.3% and for odd lattice sizes in the range $5 \leq L \leq 25$. This set of data was only used for the purpose of finding the threshold and critical exponent, and is not the main data set used for the rest of the analysis in this

chapter.

The lattice sizes we use are far from the large L limit, so following the method from Wang *et al.* the fitting ansatz was constructed by taking a quadratic expansion in x around the threshold $x = 0$ and accounting for finite-size effects by adding a single non-universal term that is dependent on the lattice size [Wang et al., 2003]. The ansatz is:

$$P_{\text{fail}} = A + Bx + Cx^2 + DL^{-1/\mu}, \quad (3.6)$$

where A , B and C are expansion coefficients, D is the coefficient of the non-universal term, μ determines the dependence of the non-universal term on the lattice size¹, and

$$x = (p - p_{c0})L^{1/\nu_0}. \quad (3.7)$$

Here ν_0 is the critical exponent and p_{c0} is the threshold error rate for our PMA decoder.

Figure 3.3 shows the rescaled data with finite-size effects subtracted, and the fit to the data. The relevant parameters were found to be:

$$\begin{aligned} p_{c0} &= 0.1028 \pm 0.0002, \\ \nu_0 &= 1.530 \pm 0.006, \\ \mu &= 1.15 \pm 0.8, \\ A &= 0.246 \pm 0.006, \\ B &= 1.87 \pm 0.01, \\ C &= 2.16 \pm 0.06, \\ D &= -0.026 \pm 0.008. \end{aligned} \quad (3.8)$$

The threshold for our modified decoding algorithms was found to be in agreement with the value found by Wang *et al.* for the unmodified MWPMA [Wang et al., 2003]. This does not achieve the maximum threshold of $p_{c0} \simeq 10.6\%$ that is possible when the degeneracy of the matching is included [Stace and Barrett, 2010]. This is because in the simulations performed we allow only a weak dependence of the choice

¹We expect the non-universal term to be smaller for larger lattice sizes and vice versa.

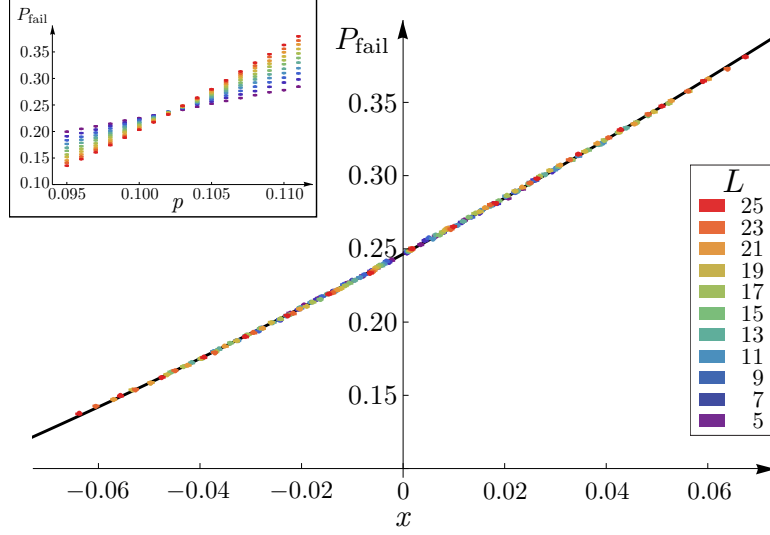


Figure 3.3: Data obtained from numerical simulations of the toric code failure rate close to threshold, rescaled using $x = (p - p_{c0})L^{1/\nu_0}$. Each data point represents $N = 10^6$ runs. The finite-size correction $DL^{-1/\mu}$ is subtracted from P_{fail} . All of the data collapses to a single curve and the threshold can be extracted as a fit parameter. *Inset:* The data prior to rescaling.

of matching on the degeneracy in our modified PMA decoder. This means that the choice of matching is only weakly dependent on the degeneracy of the matching and the effect on the threshold is small. The value of the critical exponent ν_0 found here is in agreement with the value found by Merz and Chalker when calculating the optimal threshold value [Merz and Chalker, 2002], although it does not agree with value found by Wang *et al.* for the MWPMA decoder.

The analysis presented in this section establishes the validity of the rescaling approach to the analysis for this choice of decoder by demonstrating that the scaling ansatz, equation (3.6) provides a good fit to the collapsed data close to the threshold.

3.5.2 The Threshold in the Phenomenological Noise Model

The threshold for the MWPMA decoding for the toric code subject to noisy syndrome measurements has been calculated previously as $2.93 \pm 0.02\%$ [Harrington, 2004; Stephens, 2014; Wang et al., 2003]. To find the logical failure rate P_{fail} we numerically

simulate the error correction protocol using the method described in Section 3.6. We performed $N = 10^5$ simulations of the error correction procedure for p close to 2.9% and for odd lattice sizes in the range $5 \leq L \leq 15$. This set of data was only used for the purpose of finding the threshold and critical exponent of this model.

Once again we fit the ansatz in equation 3.6 to the data and the relevant parameters were found to be:

$$\begin{aligned}
 p_{c0} &= 0.0292 \pm 0.0002, \\
 \nu_0 &= 1.03 \pm 0.03, \\
 \mu &= 0.59 \pm 1.0, \\
 A &= 0.048 \pm 0.005, \\
 B &= 1.41 \pm 0.06, \\
 C &= 14.3 \pm 2.4, \\
 D &= 0.04 \pm 0.4.
 \end{aligned} \tag{3.9}$$

The threshold for our modified decoding algorithms was again found to be in agreement with the value of $p_{c0} = 0.0293 \pm 0.0002$ found by Wang *et al.* for the unmodified MWPMA [Wang et al., 2003], because in the simulations performed we allow the same weak dependence of the choice of matching on the degeneracy in our modified PMA decoder as in the perfect syndrome measurement case. The value of the critical exponent ν_0 is also in agreement with the value found by Wang *et al.* of $\nu_0 = 1.0 \pm 0.05$ for the MWPMA decoder.

The analysis presented in this section establishes the validity of the rescaling approach to the analysis for this choice of decoder. We have demonstrated that data collapse occurs, and also shown that the scaling ansatz, equation (3.6) provides a good fit to the collapsed data close to the threshold.

3.6 Evidence for the Universal Scaling Hypothesis

In this section we give evidence that the numerical data meets the two conditions required for the universal scaling hypothesis, namely an exponential decay of the failure rate as L increases. The other requirement, scale invariance, has been

demonstrated in the previous section. Although in this section we consider only the case where the syndrome measurements are made perfectly, the same principle can be demonstrated in the presence of the phenomenological noise model.

To observe the dependence of P_{fail} on L and p we have generated a set of Monte Carlo data for $0.01 \leq p \leq 0.08$ and odd lattice sizes in the range $5 \leq L \leq 23$. We use the simulation method outlined in Section 3.3.2 with each simulation repeated $N = 10^7$ times using Kolmogorov's Blossom V minimum-weight perfect matching algorithm implementation [Kolmogorov, 2009]. We pass modified weights to the algorithm to account for degeneracy as described in Section 3.3.1.

In figure 3.4 we plot the logical failure rate on a logarithmic scale, as a function of the lattice size. The shaded portion of the figure indicates the region where this exponential relationship is not expected to hold according to a conjecture that will be explained in Section 3.8.

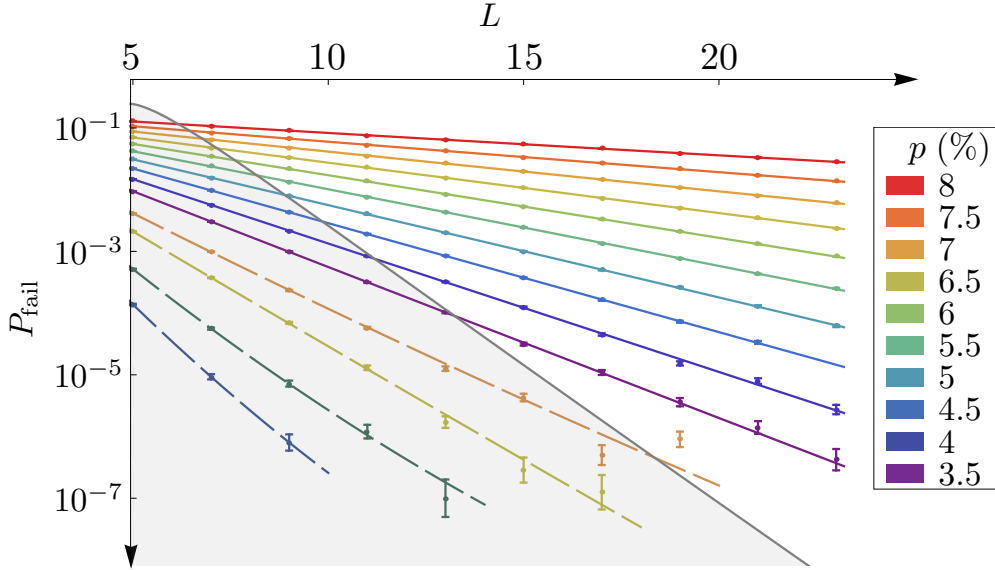


Figure 3.4: Dependence of the logical failure rate P_{fail} on the size of the lattice. Each data point represents $N = 10^7$ runs. The data is plotted on a logarithmic scale and linear fits to a selected set of the data between $p = 3.5\%$ and $p = 8\%$ are shown. The four data sets shown in the lower part of the plot (dashed lines) are examples of data with $p < 3.5\%$ for which linear fits could not be identified. In the grey region the linear relationship is expected to break down according to our validity conjecture, see Section 3.8.

Each set of data in figure 3.4 is fitted using a quadratic ansatz in L :

$$\ln P_{\text{fail}} = \alpha + \beta L + \gamma L^2. \quad (3.10)$$

For data in the range $0.035 \leq p \leq 0.08$ and $5 \leq L \leq 23$ the quadratic coefficient γ is typically 2–3 orders of magnitude smaller than the linear coefficient β . This is strong evidence for a linear fit to the (logarithmic) data, suggesting a fit of the form $P_{\text{fail}} \propto e^{-L}$, matching equation (3.2). For data with values of $p < 0.035$ the quadratic coefficient was comparable in magnitude to the linear coefficient. A selection of this data is also shown in figure 3.4, demonstrating that the behavior of the data for these values of physical error rate is ambiguous. Nevertheless, figure 3.4 establishes an exponential dependence of the logical failure probability on L for a wide range of the data.

The universal scaling hypothesis in equation (3.3) also requires the system to be scale invariant which implies that the behaviour of P_{fail} should depend only on the length scale L/ξ . This is demonstrated in figure 3.3 which shows the results of numerical simulations of the toric code failure rate close to threshold. Note that rescaling the numerical data using the variable $x = (L/\xi)^{1/\nu_0}$ leads to *data collapse*. This phenomenon describes the situation when data generated in different systems, in this case different lattice sizes, falls onto the same curve after an appropriate rescaling has been applied. It can be seen by comparing the data plotted as a function of both p and L in the figure inset to the main figure, in which all of the data collapses to a single curve.

3.7 The Low Single-Qubit Error Rate Regime

The universal scaling hypothesis is a good model for the logical failure rate when the lattice size is large and when there are sufficiently many errors. For a fixed lattice size, as p is reduced the universal scaling behavior should not be expected to hold indefinitely. Indeed, the numerical evidence suggests that when p becomes sufficiently small the scaling hypothesis fails. In the $p \rightarrow 0$ limit the behavior is

given by the low p analytic approximation:

$$P_{\text{fail}} \approx \frac{2L L!}{\lceil L/2 \rceil! \lfloor L/2 \rfloor!} p^{\lceil L/2 \rceil}. \quad (3.11)$$

This is justified by considering the uncorrectable error configurations in the $p \rightarrow 0$ limit and calculating P_{fail} directly. Restricting ourselves to low single-qubit error rates we consider the minimum number of errors that can cause the error correction to fail, $\lceil L/2 \rceil$. To cause the error correction to fail these errors must lie along a single minimum-weight homologically non-trivial cycle of the toric code. If they fall in this way the PMA will certainly apply the remaining $\lfloor L/2 \rfloor$ single-qubit operators required to ensure $C = E + E'$ is a logical operator. Figure 3.5 shows a sketch of how this can happen.

Thus the expression in equation (3.11) for the failure rate is constructed via a counting argument. The first factor, $2L$, is the number of minimum-weight homologically non-trivial cycles of the code that exist. The second is the binomial coefficient which counts the possible combinations of $\lceil L/2 \rceil$ errors along a cycle of weight L . Finally, we include a factor that accounts for the likelihood of exactly $\lceil L/2 \rceil$ errors occurring on a lattice constructed from $2L^2$ qubits, which is $p^{\lceil L/2 \rceil} (1-p)^{2L^2 - \lceil L/2 \rceil}$. The single-qubit error rate is small so we can neglect the final factor of $(1-p)^{2L^2 - \lceil L/2 \rceil}$ to obtain equation (3.11). In the low p limit the L dependence is $P_{\text{fail}} \propto e^{-\lceil L/2 \rceil}$ and we see that it is quantitatively different to the universal scaling regime, $P_{\text{fail}} \propto e^{-L}$.

A similar counting argument should apply when the syndrome measurements are noisy – the code distance is unchanged so the majority of the argument presented here should hold. However, the number of minimum-weight logical operators in the 3D data structure is increased from $2L$ to $2L^2$ hence the low p approximation is slightly modified in the presence of this phenomenological noise model:

$$P_{\text{fail}} \approx \frac{2L^2 L!}{\lceil L/2 \rceil! \lfloor L/2 \rfloor!} p^{\lceil L/2 \rceil}. \quad (3.12)$$

3.8 The Validity of the Two Approximations

The range of parameters we consider in our numerical simulations encompasses both the small p limit and the universal scaling limit. For small single-qubit error rates

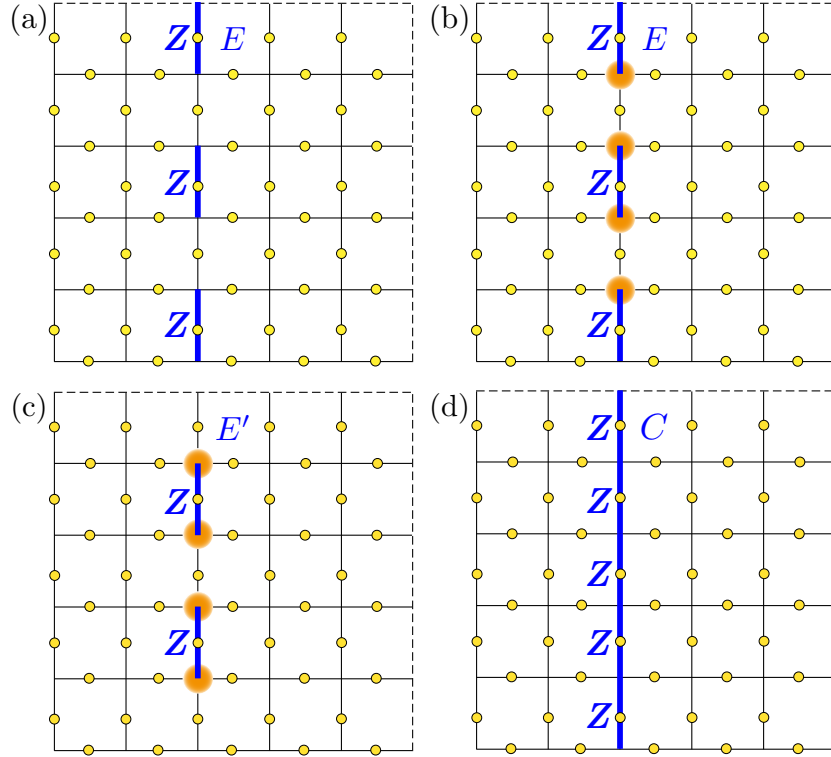


Figure 3.5: One way in which $\lceil L/2 \rceil$ errors lying along a minimum weight homologically non-trivial cycle will result in a logical error. The PMA decoder applies a correction chain that results in a non-trivial cycle, causing a logical failure. (a) The errors are distributed arbitrarily along one minimum-weight homologically non-trivial cycle of the lattice. (b) The syndromes that arise as a result of the error configuration are shown. (c) The minimum-weight perfect matching returns the correction chain E' with certainty. (d) The resultant cycle $C = E + E'$ is homologically non-trivial, which means that the error correction has failed.

the weight of the errors is typically much smaller than the code distance and the low p analytic expression is applicable. Conversely, for large L the number of errors can be much larger than the code distance and we expect a *universal scaling hypothesis* to apply. These regimes are distinct, as we see from their differing dependence on the code distance. Each of the two regimes will provide a good approximation to the numerical data over some region of parameter space. We shall now make a heuristic argument to quantify those regions, first considering the case where the syndrome measurements are noise-free and then explaining how it can be applied to the noise model incorporating noisy syndrome measurements.

In order to make a conjecture about the validity of the regimes, we consider the distribution of the number of errors that arise on a lattice of fixed size, at a known physical error rate. We will relate this distribution to $\lceil L/2 \rceil$, half the code distance. This number is significant to the PMA decoder because if the weight of the error chain, $|E|$, is less than this number then the error is certainly correctable. In the case when $|E| \geq \lceil L/2 \rceil$ a subset of the possible error configurations will lead to an incorrect pairing of syndromes, causing a logical failure. Errors obeying $|E| = \lceil L/2 \rceil$ are the spanning errors illustrated in figure 3.5.

The typical weight of errors on the lattice can be shown to be $2L^2p$. If $2L^2p < \lceil L/2 \rceil$ then the expected number of errors is less than half the code distance and logical errors are dominated by spanning chains, see figure 3.5. For a fixed p , as L increases this inequality is violated. When the number of errors is much greater than L but they are typically correctable, this is the universal scaling limit.

Requiring $p \gg 1/L$ (up to a numerical factor) leads to a relationship between L and p that determines a minimum single-qubit error rate for a given lattice size below which the universal scaling hypothesis breaks. We will now make explicit this relationship.

The single-qubit errors occur independently and at a rate p . The weight of the error that arises, $|E|$, obeys a binomial distribution with a mean that coincides with the typical error weight,

$$\mu = 2L^2p, \tag{3.13}$$

and a variance of:

$$\sigma^2 = 2L^2p(1-p). \quad (3.14)$$

According to the central limit theorem the binomial distribution can be approximated by a normal distribution for large enough lattice size.

For the universal scaling hypothesis, the condition we have proposed is that μ , the mean of the probability distribution, is large with respect to $\lceil L/2 \rceil$. This implies that the weight of the error chain that results is larger than $\lceil L/2 \rceil$ with high probability. We can write this as $\mu \gg \lceil L/2 \rceil$, or

$$\mu - n \frac{\sigma}{2} > \lceil \frac{L}{2} \rceil, \quad (3.15)$$

where n is the number of standard deviations above $\lceil L/2 \rceil$ we require the mean to lie. We make the arbitrary but natural choice of $n = 2$ for both the universal scaling hypothesis and corresponding condition for the low p expression.

Substituting equations (3.13) and (3.14) into equation (3.15) we obtain

$$2L^2p - \sqrt{2L^2p(1-p)} > \lceil \frac{L}{2} \rceil. \quad (3.16)$$

Solving for p and taking only the highest order terms, we arrive at the expression

$$p_{\text{USH}} \approx \frac{L^2 + \sqrt{2L^3} + 2L}{4L^3}. \quad (3.17)$$

This expression determines whether or not the behaviour can be considered to be within the universal scaling hypothesis regime.

The derivation for the equivalent validity condition for the low p expression, p_{LP} is not explicitly shown, but can be reproduced using a similar argument. Using similar methods but requiring $p \ll 1/L$, or

$$\mu + \sigma < \lceil \frac{L}{2} \rceil, \quad (3.18)$$

so that when this inequality is violated the single-qubit error rate above which the low p expression no longer provides a good approximation to the numerical data.

Using the same method as before, the point at which this occurs can be shown to be

$$p_{\text{LP}} \approx \frac{L^2 - \sqrt{2L^3 + 2L}}{4L^3}. \quad (3.19)$$

When $p \sim 1/L$ there is a ‘crossover’ region, in which the logical failure rate cannot be considered to be well approximated by either regime.

In the case of the phenomenological noise model, $|E|$ once again obeys a binomial distribution with a mean that coincides with the typical error weight,

$$\mu = 2L^3 p, \quad (3.20)$$

and a variance of:

$$\sigma^2 = 2L^3 p(1 - p). \quad (3.21)$$

Once again solving equation (3.15) and equation (3.18) using these definitions of the mean and variance we can derive conditions for the validity of the two regimes in the presence of noisy syndrome measurements.

3.8.1 Testing the Range of Validity of the Universal Scaling Hypothesis

Substituting p_{USH} given by equation (3.17) into the universal scaling hypothesis in equation (3.3) yields an expression for the minimum P_{fail} , for a fixed L , that belongs to the universal scaling regime. This expression is plotted as a grey line in figure 3.4 and hence the grey region indicates the region of parameter space where we do not expect the universal scaling hypothesis to hold. This supports the previous observation that most of the data we have obtained for $p < 3.5\%$ would lie outside the universal scaling region and therefore be poorly fit by equation (3.4).

We have fitted the universal scaling ansatz $P_{\text{fail}} = Ae^{-a|p-p_{c0}|^{\nu_0}L}$ (see equation (3.3)) to the data that falls outside this grey region. Recall that the values of A , p_{c0} and ν_0 are all determined from the appropriate fit to the data around threshold. From the fit to the data in the universal scaling regime we find $a = 32.31 \pm 0.13$ for the noise-free syndrome measurements.

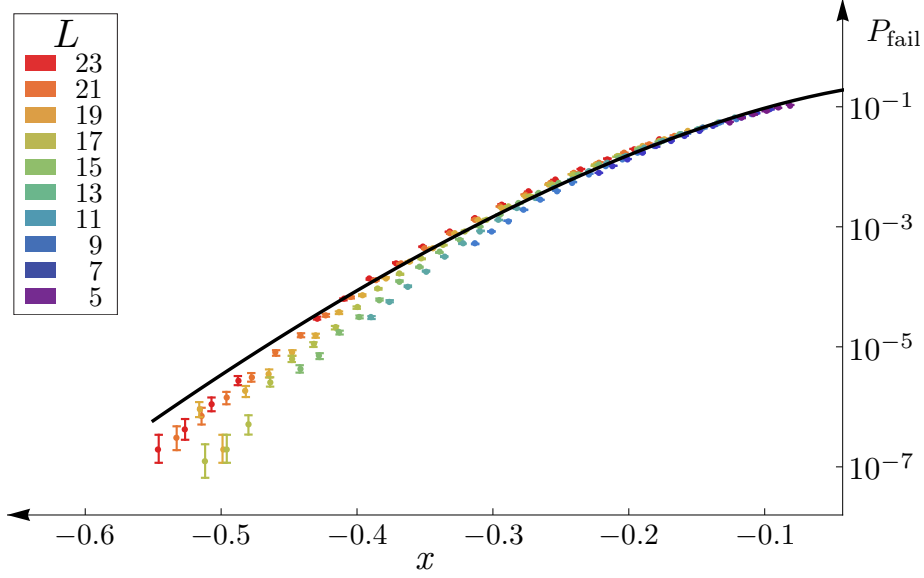


Figure 3.6: Data satisfying the condition $p > p_{\text{USH}}$, plotted on a logarithmic scale and coloured according to lattice size for the case of perfect syndrome measurements. Each data point represents $N = 10^7$ runs. Also shown in black is the fit of the ansatz, equation (3.3) with all values taken from the threshold fit (see Section 3.5) except for a which was extracted using a fit to the data set shown.

In the case of phenomenological noise model we fitted both A and a to the equivalent set of data and obtained values of $A = 0.066 \pm 0.003$ and $a = 40.4 \pm 0.6$. The data obeying the validity condition and the fit are shown in the case of perfect syndrome measurements in figure 3.6 and in the presence of noisy syndrome measurements in figure 3.7.

Let us now fix the code distance L and vary the single-qubit error rate to see how the full set of data behaves in relation to the universal scaling limit. For each fixed L in figure 3.8, reducing x corresponds to reducing p . When p becomes sufficiently small the scaling hypothesis fails and as expected the failure rate deviates below the universal scaling law.

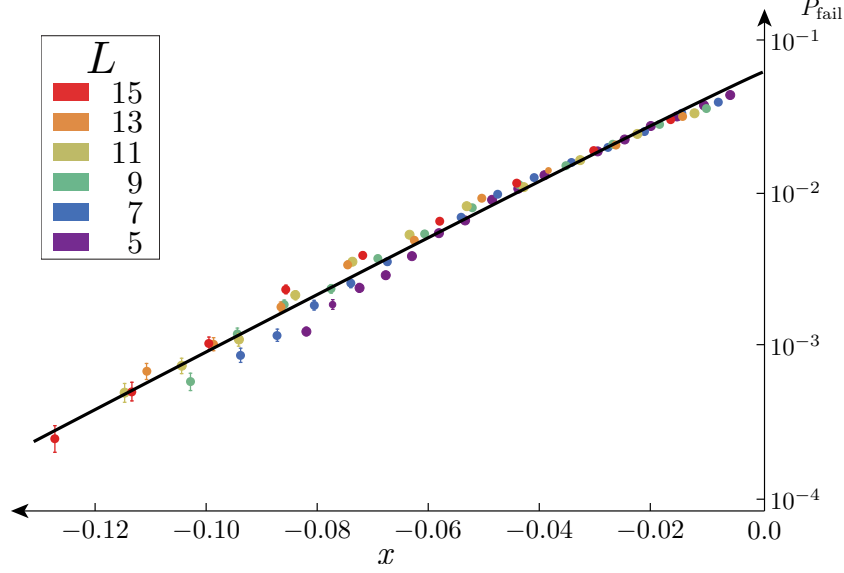


Figure 3.7: Data satisfying the condition $p > p_{\text{USH}}$, plotted on a logarithmic scale and coloured according to lattice size for the phenomenological noise model. Each data point represents $N = 10^5$ runs. Also shown in black is the fit of the ansatz, equation (3.3) with values taken from the threshold fit (see Section 3.5) except for A and a which were extracted using a fit to the data set shown.

3.8.2 Testing the Range of Validity of the Low Error Rate Regime

We have proposed that, in the low p limit, spanning errors of the type illustrated in figure 3.5 dominate when $2L^2p < \lceil L/2 \rceil$. This is the validity condition we use for the low p regime, see equation (3.19).

We can rewrite equation (3.11) in terms of L and the rescaling variable, x . Figure 3.9 shows this analytic expression plotted for some small values of L along with the numerical data. As the probability of errors decreases on a fixed lattice the mean number of errors will approach $\lceil L/2 \rceil$. As expected, the low p expression gives a good approximation for small lattice sizes and low physical qubit error rates. The data and low p analytic expression converge as x decreases, so for fixed lattice size as the physical error rate decreases the approximation improves.

Figure 3.10 shows the low p analytic expression for the phenomenological noise model in equation (3.12) plotted for some small values of L . The very low p data was

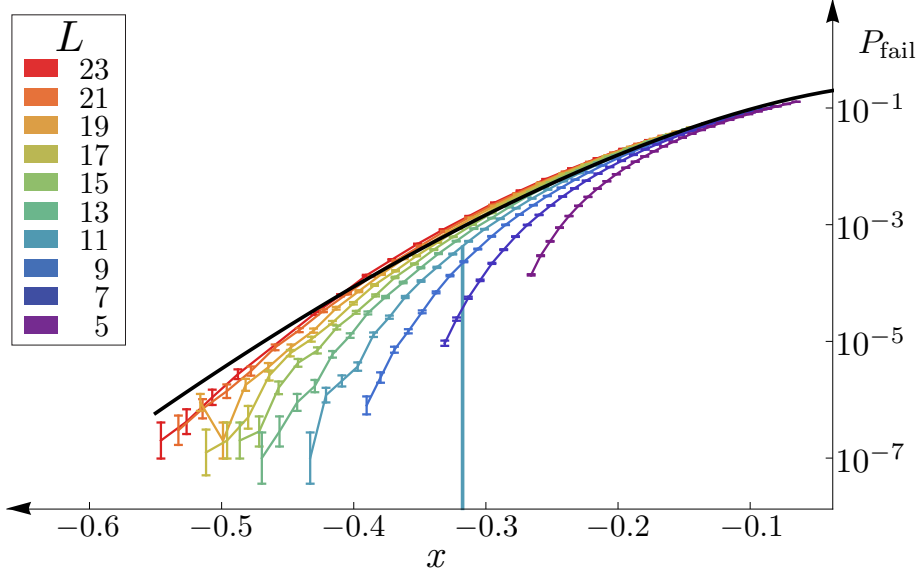


Figure 3.8: Logarithmic plot of all numerical data following the rescaling transformation $(L, p) \rightarrow x = (p - p_{c0})L^{1/\nu_0}$. The universal scaling fit is also shown in black. The data is coloured according to lattice size L . For fixed L , decreasing x corresponds to reducing p . As we do this the universal scaling hypothesis breaks at a point predicted by equation (3.17). This is indicated for a single lattice size ($L = 11$) as a vertical line.

obtained by increasing N , the number of runs, to 5×10^7 . Here we see convergence between the analytic expression and the data once again.

3.9 Comparison of the Overhead in the Two Regimes

So far we have concentrated on determining the logical error rate as a function of the lattice size and single-qubit error rate. Now we wish to demonstrate that it is possible to invert these relationships to find the overhead, Ω . This will be a function of the experimentally determined single-qubit error rate, p , and maximum tolerable logical failure rate P_{fail} .

Throughout this section we shall use the results from the phenomenological noise simulations. Although these results will not give accurate overheads because in reality

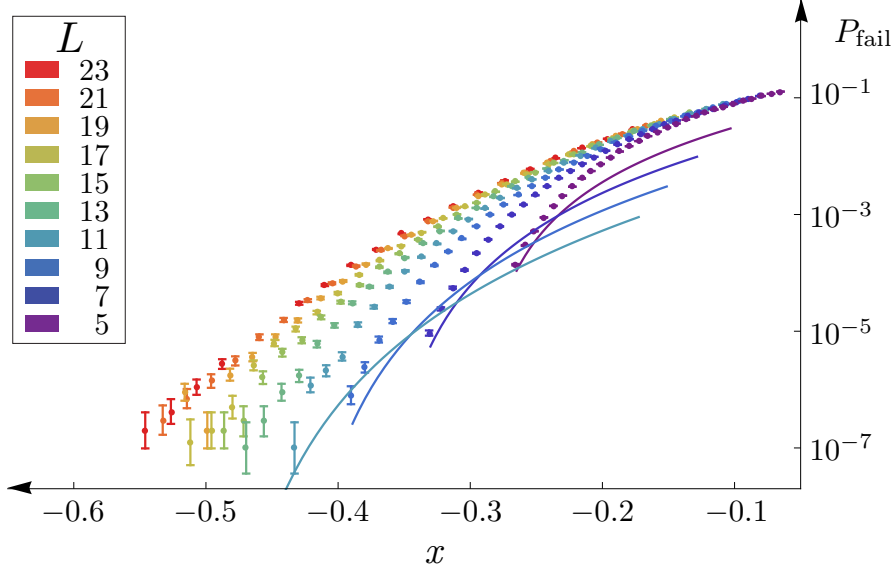


Figure 3.9: The full set of rescaled data, coloured by lattice size. The low p analytic expression, equation (3.11) is shown for some small lattice sizes. As x decreases the analytic expression tends towards the data. This numerical evidence suggests that the analytic expression is an underestimate of the failure rate for this range of parameters.

the noise model is much too simple to accurately reflect any physical implementation of the code, it will nonetheless give a more realistic picture than under the assumption that the syndromes can be measured perfectly. Indeed, the same techniques shown here will also be applicable to more physically realistic settings, for example a planar code with noisy stabilizer measurements. An extension of this model is discussed in detail in Chapter 4. Although the numerics will differ from those presented here, the methods used are expected to be directly analogous.

The first step in calculating the overhead is to determine which of the two regimes (universal scaling or low p) the code is operating within. To do this we use the expression for p_{USH} to find the maximum logical error rate for which the universal scaling hypothesis holds. Similarly we use p_{LP} to find the minimum logical error rate for which the low p expression holds. In figure 3.11 we plot these two bounds, and the regions of validity that they indicate. Figure 3.11 therefore shows the region of (P_{fail}, p) parameter space for which each of the regimes is expected to give a good

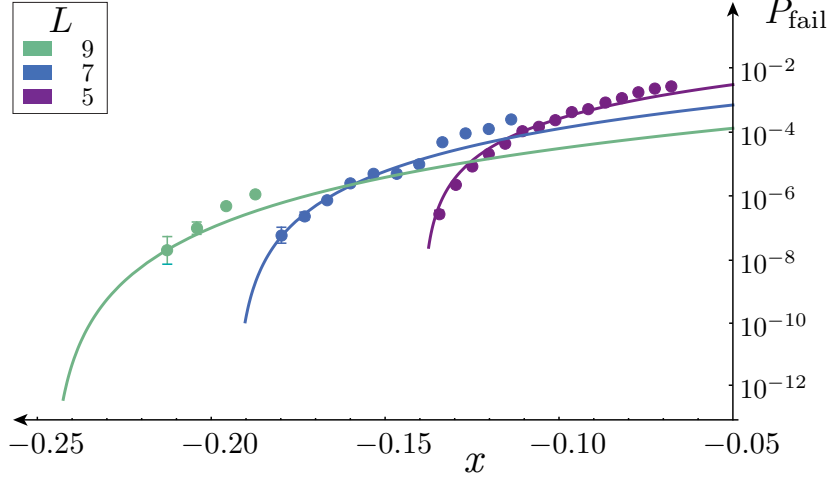


Figure 3.10: Rescaled data for the phenomenological noise model, as well as the low p analytic expression for the same model, equation (3.12), plotted for some small L values. This numerical evidence suggests that the analytic expression is a good estimate of the failure rate for most of the data plotted, and particularly for smaller lattice sizes.

approximation to the logical error rate. Once the correct regime has been identified, the overhead can be calculated.

In the universal scaling region the logical failure rate is $P_{\text{fail}} = Ae^{-a|p-p_{c0}|^\nu L}$. By using this to find the lattice size L as a function of P_{fail} and p , and recalling that there are $2L^2$ physical qubits comprising the toric code, we find the overhead in the universal scaling regime is given by:

$$\Omega_{\text{USH}}(P_{\text{fail}}, p) = \frac{2}{a^2} \left[\ln \left(\frac{A}{P_{\text{fail}}} \right) |p - p_{c0}|^{-\nu_0} \right]^2, \quad (3.22)$$

where the constants A and a have been determined from fits to the data, see Section 3.8.1. The remaining parameters, p_{c0} and ν_0 , can be determined from a fit to data generated close to threshold, see Section 3.5 for this calculation and for their numerical values.

The analytic expression for the low p regime, equation (3.12), can be simplified by assuming that $\lceil L/2 \rceil = \lfloor L/2 \rfloor = L/2$ and using Stirling's approximation $n! \approx (n/e)^n \sqrt{2\pi n}$. Inverting this simplified expression we obtain a solution for L that

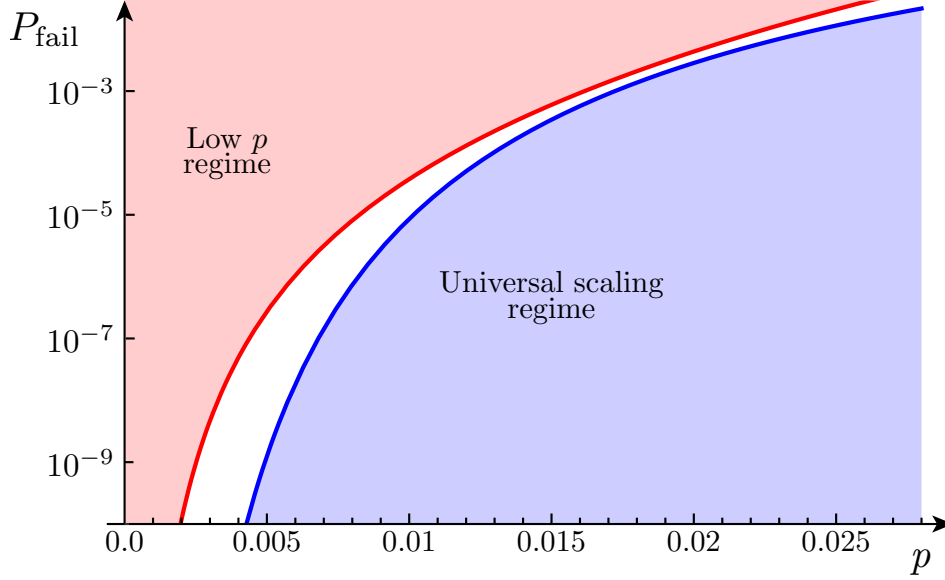


Figure 3.11: The range of validity of each of the regimes is indicated as a function of the independent variables p and P_{fail} . The red (upper-left) part of the plot represents the low p regime of validity and the blue (lower-right) part of the plot represents the universal scaling hypothesis range of validity. The uncoloured part of the plot is the crossover region between the two regimes.

uses the Lambert W function [Corless et al., 1996]. We can simplify this using the approximate form for the lower branch of the function [Veberic, 2010]. It follows that an approximate expression for the overhead in this regime is given by:

$$\Omega_{Lp}(P_{\text{fail}}, p) = 18 \left[\frac{\ln P_{\text{fail}}^{2/3} - \ln(-\ln P_{\text{fail}}^{2/3})}{\ln 4p} \right]^2. \quad (3.23)$$

Figure 3.12 shows a 3D plot of the overhead as a function of P_{fail} and p . There is a significant gap between the two plots for most of parameter space (see figure 3.13) and an increase in overhead is seen as p is increased, and as P_{fail} is decreased. Allowing a higher logical failure rate will naturally reduce the overhead required, as will reducing the single-qubit error probability.

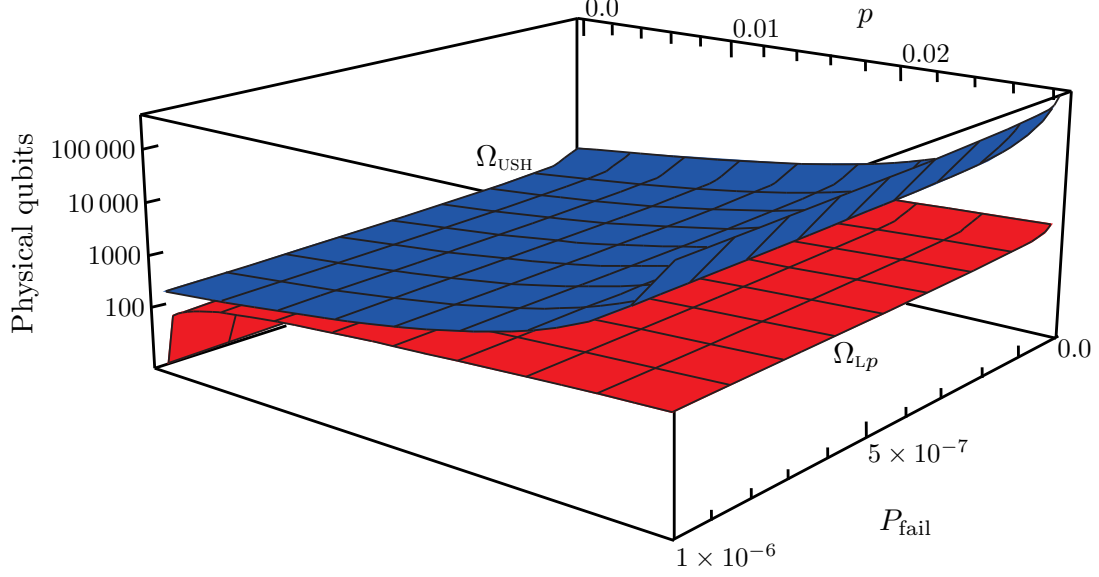


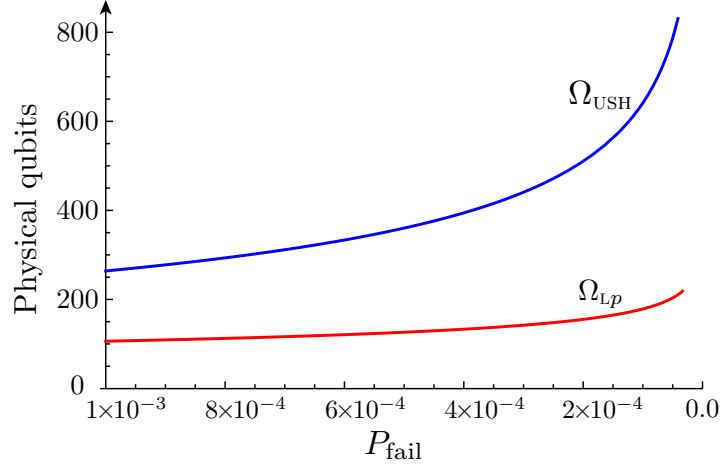
Figure 3.12: A 3D plot of the overhead, on a logarithmic scale, in each of the two regimes for $0 \leq p \leq 2.8\%$ and $10^{-9} \leq P_{\text{fail}} \leq 10^{-6}$. This plot reveals the gap between the two regimes over the whole region of parameter space considered. It also reveals drop in overhead as the single-qubit error rate is reduced, which is particularly striking for the low p regime.

As an example, let us estimate the overhead¹ using these results. Recent experimental results in a single trapped ion architecture report a single-qubit gate fidelity of 99.9999% and average error rate of 0.07% [Harty et al., 2014]. Thus we shall give a slightly larger margin of error and set the target logical success rate of the toric code to be 99.999%, and we shall make an estimate of the size of the overall error rate to be 1.5% since we are considering the state of a large number of qubits rather than a single trapped ion in isolation. In this case the code would be operating within the universal scaling regime and the overhead for the code is about 500 qubits.

Figure 3.13 shows the difference between the required overhead in the two different regimes. For the range of parameters considered the low p expression always gives an estimate of the overhead that lies below the value given by the universal scaling hypothesis.

¹Realistically speaking, here we are placing a lower bound on the overhead based on a full circuit-based error analysis, which would be a more realistic approximation to the true experimental noise model.

(a)



(b)

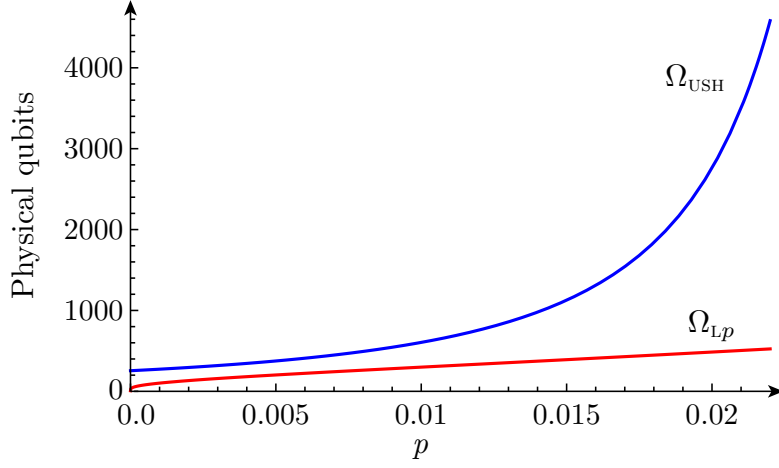


Figure 3.13: (a) The overhead for the toric code calculated for a physical error rate $p = 2\%$ for desired fidelities $10^{-9} \leq P_{\text{fail}} \leq 10^{-3}$. (b) The overhead for logical failure rate $P_{\text{fail}} = 10^{-7}$ and $0 \leq p \leq 2.5\%$. The plots can be considered to be practical lower bounds on the overhead for the parameters considered.

The low p expression tends to underestimate the logical failure rate for the range of numerical data simulated. Hence this may be considered to be a practical lower bound on the overhead required for those parameters and for the noise model considered in this chapter. Conversely, the universal scaling hypothesis is an overestimate of the logical failure rate for most of the numerical data, and hence can be considered to be a practical upper bound. We note that the numerical results bear little relation to the true resources that would be required, but the results presented provide a framework for that calculation.

3.10 Conclusions

We have found two distinct operating regimes of the toric code. In the universal scaling regime the data can be rescaled and an ansatz based on this scaling and the exponential dependence of the failure rate on L can be used to find an empirical expression for P_{fail} . In the low p regime, a counting argument gives rise to an analytic expression for the failure rate in the $p \rightarrow 0$ limit. We propose, using the probability distribution of the error weight for fixed (L, p) , heuristic conditions for the range of validity of each expression.

The expressions describing the two regimes have been inverted to calculate the system size required to achieve a desired logical success rate for a given single-qubit error rate. We have used the expressions for the logical failure rate to demonstrate techniques to calculate the overhead, $\Omega(P_{\text{fail}}, p)$. Although this was demonstrated in this chapter only for the model including noisy syndrome measurements, we have shown that the technique also works for the noise-free measurements case, see [Watson and Barrett, 2014].

We expect that the techniques we have demonstrated in this chapter will be applicable in a wide range of settings, in particular more physically realistic geometries such as the planar code. Furthermore, we expect that the methods we have demonstrated can be used to calculate the overhead of a fault-tolerant quantum memory – the data presented herein support the idea that these scaling laws are applicable in this case, although it would be even more instructive to see data for the full circuit noise model. Since all topological codes are based on similar principles, the techniques outlined in this chapter can be expected to be directly applicable

despite the fact that the numerics in these other cases will differ from those presented here.

Based on the numerical evidence, we claim that for most practical purposes the two regimes bound the required overhead. The numerical results presented in this chapter are dependent on the choice of the decoder. Similar scaling relationships would be expected for other decoding algorithms, particularly renormalisation group-based decoders such as those presented in [Bravyi and Haah, 2013; Duclos-Cianci and Poulin, 2010b].

This work raises several open questions. We believe that one of the most interesting is the question of using the resource overhead to compare the range of decoders that exist. A possible scenario is that the size of the topological code that can be realised will be fixed by technological limitations, hence a comprehensive comparison of the logical failure rate of all existing decoders over the whole region of (relevant) parameter space would be interesting and worthwhile. In that case, a comparison of the analysis presented in this chapter for all known decoders below threshold would reveal which should be implemented to minimise the logical failure rate.

Decoders with high thresholds usually require a longer run time than those with more modest thresholds. We expect a tradeoff between time and space resources, suggesting that those decoders with longer run times may have smaller physical qubit overheads. This is interesting, because although a high threshold is desirable, for practical implementations the run time and physical overhead are also important constraints. Therefore it seems that a balance between these three figures of merit may be of interest for practical quantum computation.

Several of the limitations we faced have been addressed by Bravyi and Vargo in [Bravyi and Vargo, 2013]. Firstly they address the crossover region between the two regimes we have identified. Bravyi and Vargo have constructed a heuristic ansatz that interpolates between the dependence on L of the low p regime, $P_{\text{fail}} \propto e^{-\lceil L/2 \rceil}$, and the dependence expected for larger physical error rates, $P_{\text{fail}} \propto e^{-L}$. These functional forms match the two regimes we have identified, so the ansatz by Bravyi and Vargo could lead to a method for interpolating between them.

Another benefit of the technique by Bravyi and Vargo is that it provides a fit to the numerical data in the small and moderate p regimes. A significant limitation we faced was the availability of resources to run the Monte Carlo simulations of

the error correction procedure. For example, it was impossible to obtain data for $P_{\text{fail}} < 5 \times 10^{-7}$ due to the run time of the decoder. Bravyi and Vargo have discovered a new technique for probing very low error rates on surface codes [Bravyi and Vargo, 2013]. Obtaining data for very low logical error rates using this algorithm would help us to verify the conjecture of the range of validity of the low p expression, particularly for larger lattice sizes than we were able to test.

While heuristic approaches are very flexible, our universal scaling hypothesis has the following advantages. It addresses the large L limit and gives particularly good approximations to the numerical data for moderately large single-qubit error rates. The functional form for the universal scaling hypothesis, given in equation (3.3) is derived from the phase transition of the random bond Ising model, which is a model of statistical physics that the toric code error correction can be mapped to, meaning that it is not a heuristic expression. It is also easily invertible and its pre-factor, A , does not depend on the code distance. Ultimately the implementation of universal quantum computing that is found will set the input parameters that determine which of the regimes it operates within.

The PMA decoder we have studied in this chapter has the important advantage that it can obtain a threshold that is close to optimal and has an efficient run time with respect to the size of the input. It has two significant disadvantages, however. The first is that the run time depends on the third power of the number of qubits in the code, which means in the presence of noisy syndrome measurements it is prohibitively slow for larger lattice sizes. The second disadvantage of the PMA decoder is its applicability to only qubits. As we shall demonstrate in the final two chapters of this thesis, there is growing interest in quantum computing using qudits as the basic unit, and hence quantum error correction routines making use of qudits must be devised. The PMA algorithm is not suitable for such applications. Hence in Chapter 4 we introduce a fast decoder that is suitable for both qubit and qudit surface codes.

Chapter 4

A Fast Fault-Tolerant Decoder for Qubit and Qudit Surface Codes

*The surface code is one of the most promising candidates for combating errors in large scale fault-tolerant quantum computation. A fault-tolerant decoder is a vital part of the error correction process – it is the algorithm which computes the operations needed to correct or compensate for the errors according to the measured syndrome, even when the measurement itself is error prone. Previously decoders based on minimum-weight perfect matching have been studied. However, these are computationally expensive and not immediately generalisable from qubit to qudit codes. In this chapter we develop a fast fault-tolerant decoder for the surface code, capable of efficient operation for qubits and qudits of any dimension, generalising the decoder first introduced by Bravyi and Haah [Phys. Rev. Lett. **111**, 200501 (2013)]. We study its performance when both the physical qudits and the syndrome measurements are subject to generalised uncorrelated bit-flip noise (and the higher dimensional equivalent). We show that, with appropriate enhancements to the decoder and a high enough qudit dimension, a threshold at an error rate of more than 8% can be achieved.*

A published version of this work is also available [[Watson et al., 2015a](#)].

4.1 Introduction

Recent developments have shown that employing d -dimensional quantum systems, or qudits, as the building blocks for fault-tolerant schemes may offer some important advantages. For example, an integral part of many fault-tolerant schemes is the distillation of magic states [Bravyi and Kitaev, 2005], where generalisation to higher dimensions has resulted in improved distillation thresholds and lower overheads [Campbell, 2014; Campbell et al., 2012]. Moreover, threshold investigations of the qudit toric code with noise-free syndrome measurements have shown that, for a standard independent noise model, the error correction threshold increases significantly with increasing qudit dimension [Andrist et al., 2015; Anwar et al., 2014; Duclos-Cianci and Poulin, 2013], although it is difficult to fairly compare noise rates between systems of different dimension. It is inherently more challenging to realise qudit quantum systems experimentally, nonetheless recent work has demonstrated the ability to coherently control and perform operations in 16-dimensional atomic systems with high fidelity [Anderson et al., 2015; Smith et al., 2013].

In order to store the encoded information for an arbitrary length of time, active error detection must be performed periodically in order to prevent the errors from accumulating beyond the capability of the code to correct them. In every round of error correction the *syndrome* is obtained by measuring the stabilizer generators and then processed by the *decoder*. In a realistic environment both the physical systems and the stabilizer measurements are prone to errors, and hence the decoder must be able to take both of these types of errors into account [Dennis et al., 2002; Wang et al., 2003].

Decoders are often developed for the simpler case where measurement error is neglected. However, there is a well established and elegant method for generalising measurement-noise-free decoders for topological codes to the fully fault-tolerant setting [Dennis et al., 2002]. The noisy syndrome measurements are repeated, extending the two-dimensional surface representing the code to a three-dimensional data structure, where time represents an extra dimension. Remarkably, the change from two to three dimensions allows most decoder algorithms developed for noise-free measurements to be applied largely unchanged in this more general setting.

The most widely used decoding algorithm for topological codes remains the

minimum-weight perfect matching algorithm (MWPMA). However, this algorithm has a number of disadvantages. For a distance L surface code, with error-free measurements, the run time for a basic implementation of the MWPMA algorithm scales with $O(L^6)$, and for with error-prone measurements this run time increases to $O(L^9)$. Although this is a polynomial complexity, a 9th power scaling soon requires a significant classical computational resource. A more refined fault-tolerant implementation for the qubit surface code scales with $O(L^2)$ [Fowler et al., 2012b]. A second disadvantage is that the MWPMA algorithm is not suitable for qudit surface codes with $d > 2$. For these reasons, the development of alternative decoding algorithms is currently a very active research area [Anwar et al., 2014; Bravyi et al., 2014; Duclos-Cianci and Poulin, 2013, 2014; Herold et al., 2014; Hutter et al., 2014a,b; Wootton, 2015].

In this chapter we introduce a fault-tolerant decoding algorithm which addresses both of the disadvantages of the MWPMA. The algorithm, which extends the hard-decision renormalisation group (HDRG) decoder proposed by Bravyi and Haah [Bravyi and Haah, 2013], has a fast typical run time of $O(L^3)$ and can be applied to qudit surface codes of any dimension d .

A widely-studied qubit error model (described in Section 3.3.3) is the simple *uncorrelated noise* model where X -type and Z -type Pauli errors on individual code qubits and bit-flip errors on the syndrome measurement outcomes each occur independently with probability p . In the present chapter we shall introduce the qudit generalisation of this noise model and show that the decoder we introduce can, in the limit of large code distance, correct errors generated by this noise model up to some threshold error rate. That a decoder possesses a threshold for such a model is a necessary but not sufficient condition to show that it will also be effective in the presence of a full gate-based error model of fault-tolerance. Nevertheless we can have high confidence that it will generalise since there have been no examples where such a decoder has failed for an independent gate-error noise model. In light of this observation, and for brevity, we shall refer to this decoder as fault-tolerant.

We denote the error threshold for a given qudit dimension by $p_{\text{th}}^{(d)}$. For the uncorrelated noise model, the optimal threshold for the qubit toric code is known to be $p_c^{(2)} = 3.3\%$ [Ohno et al., 2004] while the threshold obtained with the MWPMA decoder is $p_{c0}^{(2)} = 2.9\%$ [Harrington, 2004; Wang et al., 2003].

The HDRG decoder we study here attains a threshold of $p_{\text{th}}^{(2)} = 2.2\%$ for the qubit planar code¹ and may also be used with qudit surface codes of any qudit dimension d . For the qudit generalisation of the uncorrelated noise model, the decoder achieves a threshold value which increases monotonically with d , until it reaches a saturated value of around 4.2%. Throughout the rest of this chapter, unless we specifically refer to qubits, we assume that the more general term ‘qudit’ refers to quantum systems of any dimension including the $d = 2$ case.

We show that this saturating behaviour is due to a syndrome percolation effect which upper-bounds the achievable threshold. To overcome the percolation threshold we have constructed a procedure executed *before* running the HDRG, which we call the *initialisation step* [Anwar et al., 2014]. The algorithm implemented in this ‘pre-decoding’ step disrupts the syndrome percolation and boosts the threshold to 8.3% for sufficiently high qudit dimension. We call the HDRG decoder when augmented with the initialisation step the *enhanced-HDRG decoder*.

4.2 The Noise Model and Simulation Methods

In this section we shall begin by reviewing the general requirements for a decoder in the 2D surface code. This will introduce part of the notation in a familiar setting. Next, we will give a formal description of the noise model and describe the method for the fault-tolerant simulation.

The qudit surface code, introduced in Section 2.3.1, is a stabilizer code. Recall that a single X -type error is detected by two adjacent plaquettes, except when it occurs on a smooth boundary, see figure 2.9(a). A string of X -type errors is detected by plaquettes contiguously along the path of the string, as shown by the example in figure 2.9(b). This is in contrast to the qubit case ($d = 2$) where only the end-points of the string give rise to non-trivial plaquette measurements. This observation suggests that in higher d the syndrome reveals more information about the path of the errors on the lattice. Indeed, it is this information that, if exploited

¹It is not clear that the toric code and planar code thresholds may be directly compared. Nevertheless, for the phenomenological noise model (see Section 4.3) it has been observed that the error correction threshold of the qubit toric and planar codes match in numerical simulations using the MWPMA decoder [Stephens, 2014; Wang et al., 2003].

correctly by the decoder, can lead to improved error correction performance, as shown by their higher threshold values as d increases.

Generally speaking, the aim of the decoder is to use the information contained in the syndrome to return a correction operator that restores the code to its original state. More formally, let us denote an arbitrary configuration of X -type errors on the 2D surface code by the set $\mathbf{e}$, and the corresponding plaquette measurement outcomes by the set $\mathbf{s} = \{s_{x,y}\}$, where $s_{x,y} \in \mathbb{Z}_d$ is the outcome of the measurement. The subscripts x, y are the coordinates of the plaquettes, and we choose our coordinate system such that $1 \leq x \leq L$ and $1 \leq y \leq (L - 1)$, where L is the linear size of the code. We will often refer to the outcome $s_{x,y}$ as the *charge* of the measurement. Then we say that a decoder $\mathcal{D}$ takes in the syndrome $\mathbf{s}$ and returns a correction configuration $\mathbf{e}'$. We denote this map by $\mathcal{D}(\mathbf{s}) \rightarrow \mathbf{e}'$.

The decoder succeeds if $\mathbf{e}\mathbf{e}' \in \mathcal{S}$, the group of stabilizer operations. The decoder fails if $\mathbf{e}\mathbf{e}' \in \mathcal{L} - \mathcal{S}$, the logical operations excluding the stabilizer group, in other words the group of operators that act non-trivially on the logical information.

4.3 The Phenomenological Noise Model

We described the noise model and simulations for the qubit toric code in Section 3.3.3. Here we generalise it to the qudit code, and introduce the formal notation which was unnecessary for the description in Chapter 3. Similarly to Chapter 3 we shall use a single parameter p to describe the noise model for both physical and measurement errors. As we explained when this simple noise model was first introduced, this model is commonly used when benchmarking decoders.

Between each round of syndrome measurements we assume that each physical qudit is independently subject to an error channel which applies error operator X^k such that $1 \leq k \leq (d - 1)$ with equal probability $p/(d - 1)$, followed by an error channel which applies error operator Z^k such that $1 \leq k \leq (d - 1)$ with equal probability $p/(d - 1)$. We then assume that the outcome of each syndrome measurement j undergoes an error which maps j to $j \oplus k$ for $1 \leq k \leq (d - 1)$ with equal probability $p/(d - 1)$. This is the qudit generalisation of the uncorrelated noise model.

As in the qubit code, under the assumption of the uncorrelated noise model, the

noise is symmetric for the physical errors and the measurement errors. We therefore repeat the measurements L times where L is the distance of the surface code. We shall call the coordinate in this direction t to mean a simulated time, or time-like direction. The resulting data structure can be viewed as a cuboid, with $1 \leq x \leq L$, $1 \leq y \leq (L - 1)$ and $1 \leq t \leq L$.

4.3.1 Simulating the Surface Code

The scheme for measuring the syndromes of the surface code and using this syndrome information to infer a correction operator is illustrated in figure 4.1. In this section we shall explain the protocol for determining the correction operator from the syndrome information in general terms, and in the next section we give the details of the HDRG decoder.

The simulation proceeds by first generating a 3D data structure for L time steps. This initial structure contains the accumulated history of both the physical qudit errors, as well as the measurement errors. The corresponding syndrome measurement outcomes, taking into account both of these error sources are computed. Next we must calculate the syndrome changes from one time step to the next.

Let us denote by $\mathbf{s}_t$ the set of syndrome outcomes at the t -th time step. The set of syndrome changes $\mathbf{s}'_t$ at time step t is then defined as the element-wise difference, modulo d between $\mathbf{s}_t$ and $\mathbf{s}_{t-1}$, i.e. $\mathbf{s}'_t = \mathbf{s}_t \ominus \mathbf{s}_{t-1}$, where ' $\ominus$ ' denotes subtraction modulo d and we assume the special case of $\mathbf{s}'_1 = \mathbf{s}_1$. Each set of syndrome changes corresponds to a 2D grid of integers, combined into a 3D structure with $t = 1$ at the bottom and $t = L$ at the top. We call this grid the syndrome changes history and denote it $\mathbf{S}'$. It is convenient to introduce a cartesian coordinate system to refer to the elements of $\mathbf{S}'$, i.e. $s'_{t,x,y}$ corresponds to the syndrome change at grid point (x, y) and at time step t .

The input to the decoder is the 3D syndrome changes history $\mathbf{S}' = \{\mathbf{s}_1, \mathbf{s}_2 \ominus \mathbf{s}_1, \dots, \mathbf{s}_L \ominus \mathbf{s}_{L-1}\}$. The decoder takes the syndrome changes history and returns a 3D correction operator $\mathbf{E}' = \{\mathbf{e}'_1, \mathbf{e}'_2, \dots, \mathbf{e}'_L\}$. To convert this to a physical correction operator that can be applied in 2D we ignore time-like edges and combine the two-dimensional layers corresponding to each time step to form a 2D correction operator $\tilde{\mathbf{e}}'$ that corrects the accumulated errors at the last time step of the surface code. The

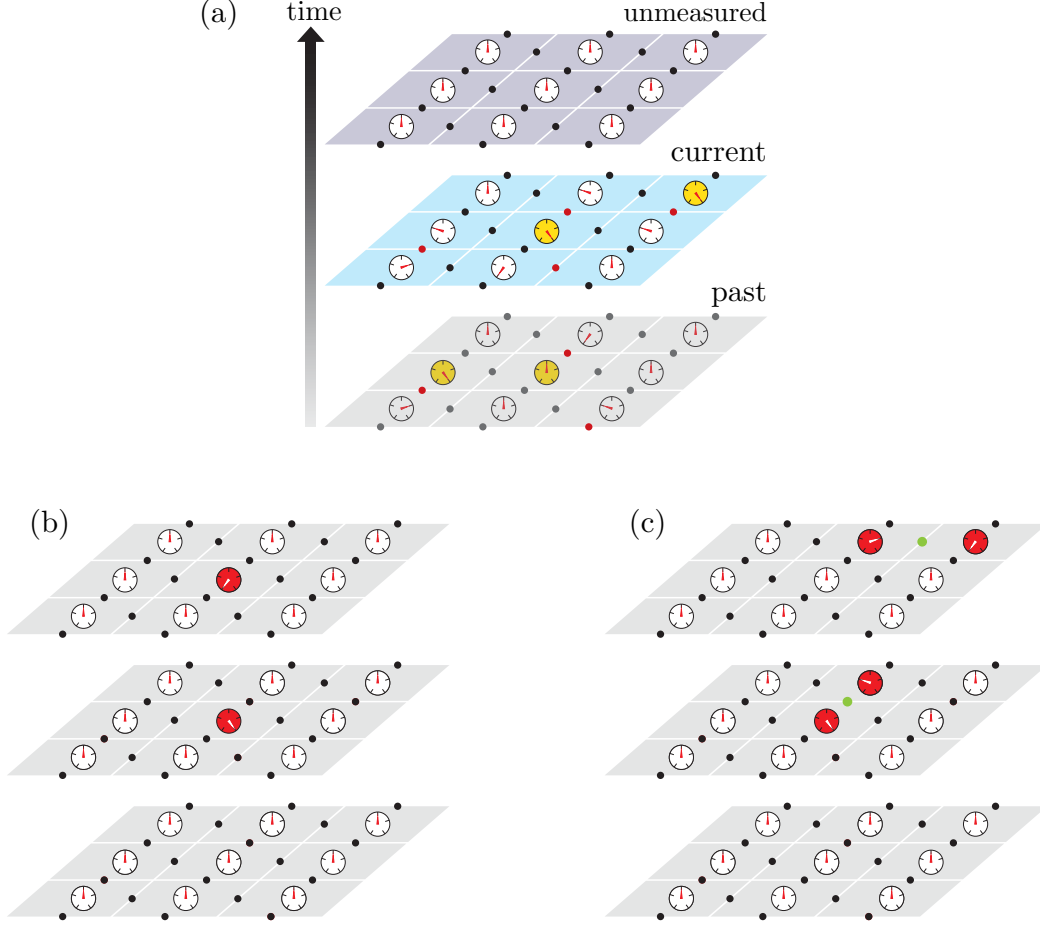


Figure 4.1: An illustrative picture of the data structure obtained in order to perform fault-tolerant error correction. Each layer represents a single time step where all the stabilizers are measured (only plaquettes are shown here for clarity – depicted by the meter with multiple outcomes) to obtain the syndrome. (a) The yellow meters represent locations where an error has occurred in the measurement procedure itself. After a specified number of time steps a full 3D history of the syndromes will have been collected. If operating below threshold the decoder then uses this data to infer a correction operator that returns the code its original state with high probability. (b) The syndrome changes history. Non-trivial syndromes are indicated in red for clarity. Here the syndrome implies that the error is in the syndrome measurement and no physical correction is applied. (c) The error causing the cluster shown has physical and time-like components. One possible correction operator is indicated in green.

resultant correction operator, $\tilde{\mathbf{e}}'$, is the product (modulo d) of the correction at each qudit location at each time step, i.e. $\tilde{\mathbf{e}}' = \prod_t \mathbf{e}'_t$. We say the decoder has succeeded when the product of the accumulated errors on the qudits and the returned correction operator is within the stabilizer of the code.

Similarly to the qubit simulation described in Section 3.3.3 we must simulate a round of error correction with perfect syndrome measurements in order to determine whether the decoder has returned an operator that is in $\mathcal{S}$ or in $\mathcal{L} - \mathcal{S}$. Once again we emphasise that in any physical implementation of the decoder this step would not be implemented, it is necessary only for the simulation.

4.4 HDRG Decoder with Noisy Syndromes

The HDRG decoder has a simple motivation behind its construction: when the error rate is sufficiently low we expect any errors arising on the surface code lattice to be sparse. This in turn means that syndromes are likely to occur in small, well-separated clusters. The HDRG decoder aims to identify clusters of syndromes generated by such sparse errors and correct them locally within a box enclosing each cluster. If these clusters have been correctly identified, and the clusters are each small enough that they do not span the lattice, then this strategy results in the decoder computing a correction operator that will correct all errors with high probability. In this section we shall give a formal definition of these concepts in the fault-tolerant setting described in Section 4.3.

4.4.1 Decoder Construction

The central concept required for the description of the HDRG decoder is that of a *metric* – a geometric distance function between any pair of elements from a set. Specifically, we wish to associate a metric between pairs of syndromes in the set $\mathbf{S}'$. The metric we use for the decoder presented in this chapter is the Manhattan distance, denoted here by δ , which maps two syndromes as follows:

$$\delta(s_{t,x,y}, s_{t',x',y'}) = |t' - t| + |x' - x| + |y' - y|. \quad (4.1)$$

The set of syndromes at a fixed distance δ from a chosen syndrome form an approximately octahedral structure around it in 3D – see figure 4.2 for an illustration for how this grows as δ increases.

We say that two syndromes are δ -connected if the Manhattan distance between them is less than or equal to δ . For a given metric value δ , we define a *cluster* $\mathcal{C}$ to be the set of non-trivial syndromes such that every element of the set is δ -connected to at least one other syndrome within that cluster. For a fixed δ , the syndrome changes history $\mathbf{S}'$ can always be partitioned into a set of disjoint clusters such that $\mathbf{S}' = \mathcal{C}_1 \cup \mathcal{C}_2 \cup \dots \cup \mathcal{C}_n$, for some integer n .

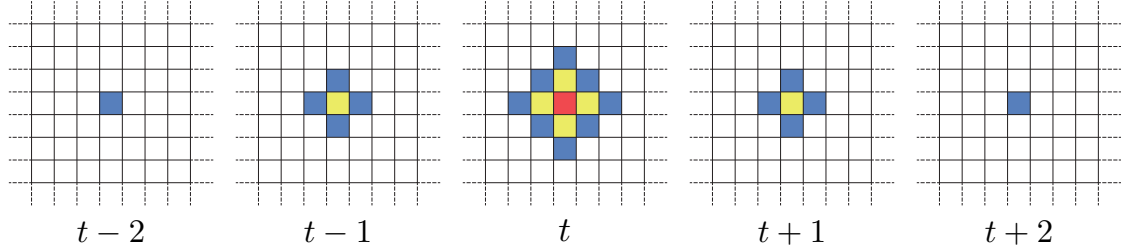


Figure 4.2: An illustration of the Manhattan distance metric. The figure shows the same portion of the lattice at each of five time steps from the syndrome changes history. The yellow plaquettes are 1-connected to the central red plaquette. Blue and yellow plaquettes are 2-connected to the central plaquette.

For each partitioning, or clustering, of $\mathbf{S}'$ corresponding to a fixed δ we classify clusters in one of three ways. This classification depends on the *cluster charge*, the sum over all the charges of the syndromes contained in the cluster $\oplus_{\mathcal{C}} s_{t,x,y}$, where the summation is performed modulo d .

- *Neutral clusters.* The total charge is zero. Such a cluster can be annihilated by fusing all of the syndromes contained within the cluster locally, meaning that the Pauli correction operator $\mathbf{e}'$ will have support only within a box enclosing the cluster.
- *Boundary-neutral clusters.* The charge is non-zero but the cluster is δ -connected to any of the three smooth boundaries (two spatial and one time-like). Clusters of this type can be annihilated by fusing the syndromes locally and then connecting the remaining charge to the nearest smooth boundary.

- *Charged clusters.* The charge is non-zero and the cluster is more than a distance δ from a smooth edge. No annihilation is possible and consequently the cluster is assumed to be part of a larger neutral cluster.

We note that the $t = L$ boundary is *both rough and smooth*. This simply means that we allow both vertex and plaquette syndromes to form boundary-neutral clusters if they are δ -connected to this boundary. Therefore boundary-neutral clusters can be defined in relation to two physical boundaries, either rough or smooth depending on the type of the syndrome, and one time-like boundary.

Here we explicitly explain how the decoder calculates $\mathbf{e}'$ for a given cluster by introducing the concept of syndrome *transportation*: a syndrome can be transported in any direction by applying the appropriate operator as illustrated by the example in figure 2.9(c). In doing this the charge at the initial syndrome location becomes zero, while the original charge a appears at one of the four neighbouring plaquettes. This is where the term ‘transportation’ arises – the syndrome value appears to move. Moreover, by transporting one syndrome to the location of a second, the two become *fused* into a single syndrome such that their charges are added (modulo d).

The HDRG decoder involves multiple levels of decoding to eliminate the non-trivial syndromes by fusing all the elements in $\mathbf{S}'$ and returning the resultant correction operator. Every decoding level ℓ is associated with a distance determining the connectivity of the disjoint clusters at that level. For the metric we have defined we will use $\delta = 2^\ell$ starting with $\ell = 0$ and running to a maximum length scale of $\ell = \lceil \log_2 L \rceil$. The algorithm terminates if all of the non-trivial syndromes are removed before the maximum level is reached. This means that the cluster connectivity increases exponentially as we increase the decoding levels linearly¹. At each level the neutral and boundary-neutral clusters are removed using the techniques listed above, leaving any charged clusters to be combined to form neutral clusters at subsequent levels. The correction operator identified at each level $\mathbf{E}'_\ell$ is tracked and the total correction calculated as $\mathbf{E}' = \mathbf{E}'_1 \mathbf{E}'_2 \dots \mathbf{E}'_{\ell_{\max}}$, where $\ell_{\max}$ is the level where the decoder terminated.

¹The exponential scaling of the metric improves the run time of the decoder at the expense of the threshold. However, the first two levels are identical when the metric increases linearly. This means that the threshold is not affected significantly by the choice of exponential scaling. Indeed, numerical tests indicate that the threshold values using exponentially and linearly increasing metrics, $p_{\text{th,e}}^{(d)}$ and $p_{\text{th,l}}^{(d)}$ respectively, satisfy $0.95p_{\text{th,e}}^{(d)} < p_{\text{th,l}}^{(d)} < 1.05p_{\text{th,e}}^{(d)}$ for all d tested.

The decoding procedure can now be summarised as follows, starting with $\ell = 0$.

1. *Clustering*: Identify all the disjoint δ -connected clusters at level ℓ .
2. *Neutral annihilation*: Fuse each neutral and boundary-neutral cluster locally and return a correction operator.
3. *Renormalise*: If there are clusters that have not been annihilated, then increment ℓ by 1 and return to step 1.

The decoder stops when there are no non-trivial syndromes remaining. The crucial feature of this decoder is that part of the total correction operator is fixed after each level of decoding. In classical coding theory, decoding algorithms exhibiting such a feature are referred to as a *hard-decision* decoders. An explicit example for a small lattice simulation is illustrated in figure 4.3.

4.4.2 The Run Time of the HDRG Decoder

The dominant parts of the decoding algorithm described in the previous section that contribute to the run time complexity are the identification of the δ -connected cluster of syndromes (we call this step *clustering*) and the determination of the Pauli operator that eliminates the syndrome (we call this step *fusion*). We shall look at each of these processes in turn and argue that for low error rates we expect a run time scaling of $O(L^3)$, and that even in the worse case this scaling will be no greater than $O(L^6)$.

Let us first consider clustering in the limit in which error rates are low and the errors are extremely sparse. In the clustering part of the algorithm at a given level ℓ , the algorithm searches a constant number of plaquettes around every non-trivial syndrome. The linear size of the metric at decoding level ℓ is 2^ℓ syndromes, so the volume it occupies is $\sim (2^\ell)^3$ hence the number of plaquettes searched for each non-trivial syndrome is $O(2^{3\ell})$. The total number of syndromes is $O(L^3)$ and the decoder will only need to run at the first level $\ell = 1$. Thus in this limit, the dependence of the run time complexity on L for this part of the algorithm will be $O(L^3)$.

In the worst case scenario we consider the most pessimistic estimates for the clustering step of the algorithm. In this case the decoder will run the maximum number of levels, $\ell = \lceil \log_2 L \rceil$. Again there are $O(L^3)$ syndromes and the dependence of the run time complexity on L for this part of the algorithm will thus be

$$\begin{aligned} \sum_{\ell=0}^{\lceil \log_2 L \rceil} 2^{3\ell} L^3 &\sim 2^{3 \log_2 L} L^3, \\ &\sim O(L^6). \end{aligned} \tag{4.2}$$

For the fusion part of the algorithm, the syndromes can all be moved to a single point in the box enclosing the cluster. To do this we use syndrome transportation, as described in Section 4.2. This will take a time that scales with the linear size of the enclosing box – the maximum linear size of the box scales with L so we expect the run time of the fusion to scale as $O(L)$ in this worst case scenario. Hence this part of the algorithm will take a time that scales like a polynomial of degree less than or equal to the clustering part of the algorithm. It also runs after the clustering so the two run times are additive rather than multiplicative.

Note that the modular arithmetic used in the decoder construction is only taking the modulus of a number with respect to d . The time complexity of such arithmetic depends only weakly on the size of d and the run time complexity of the decoder is therefore only weakly dependent on the qudit dimension.

4.4.3 Thresholds Estimation and Percolation Limitation

In order to estimate the error correction threshold we simulate the entire error correction process as described above. We begin by generating L time steps of errors and noisy syndromes¹, followed by syndrome extraction and then decoding. Finally a check is performed to determine whether the error correction succeeded or failed. The whole process is repeated for $N = 10^4$ runs for a fixed lattice size L and error rate p . We determine the threshold $p_{\text{th}}^{(d)}$ using the rescaling method outlined in Section 3.5 [Wang et al., 2003].

¹We consider the syndrome values to be $j \oplus k$ where j is the noise-free result and k is the error in the measured value. Hence just as we simulate the errors in the physical systems we can simulate the errors in the syndrome values in parallel.

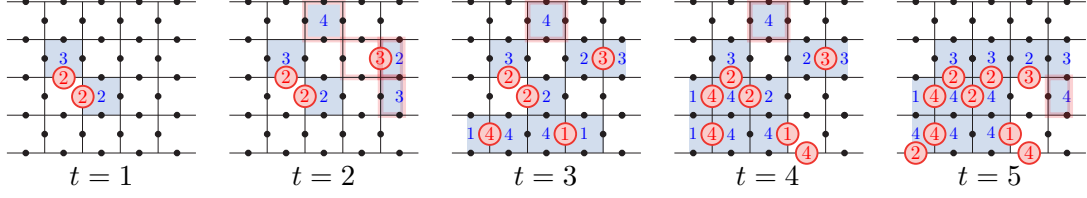


Figure 4.3(i) **Error and Syndrome Histories**: The distance 5 code is comprised of qudits with $d = 5$. The first step is to generate the full history of errors and noisy syndrome measurements $\mathbf{S}$ for $L = 5$ time steps. The red circles and squares indicate the location of errors. Notice how the errors accumulate at each time step. The goal of the decoder is to correct the final error configuration $t = 5$.

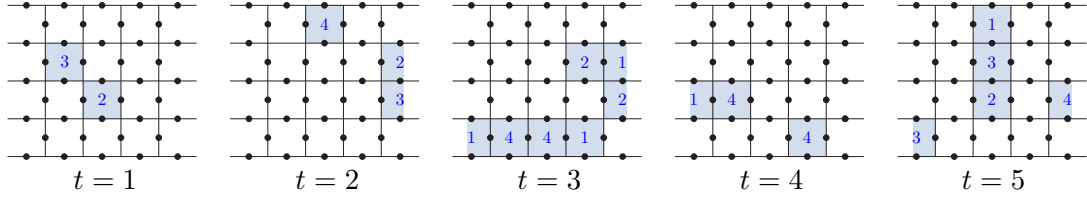


Figure 4.3(ii) **Syndrome Changes History**: The second step is to evaluate the syndrome changes history $\mathbf{S}' = \{\mathbf{e}_1, \mathbf{e}_2 \ominus \mathbf{e}_1, \mathbf{e}_3 \ominus \mathbf{e}_2, \mathbf{e}_4 \ominus \mathbf{e}_3, \mathbf{e}_5 \ominus \mathbf{e}_4\}$, where the subtraction is performed modulo d . The changes history is passed to the decoder which must infer a correction operator from the information in $\mathbf{S}'$ alone.

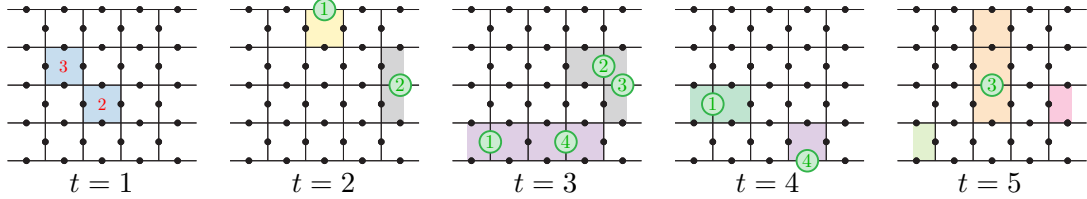


Figure 4.3(iii) **HDRG $\ell = 1$** : The first level of the HDRG decoder divides the set $\mathbf{S}'$ into disjoint 1-connected clusters (i.e. $\delta = 1$). There are three different types of clusters shown: *non-neutral*, *neutral*, and *boundary-neutral*. Specifically, there are two single-element non-neutral clusters shown in blue with their charge displayed. These clusters cannot be fused at this level. There are two neutral clusters in the bulk (grey and dark green), meaning that their total charge adds to zero (modulo 5). The elements of each neutral cluster are fused together to the vacuum. Finally, there are five boundary-neutral clusters (yellow, purple, light green, orange and pink). The total charge of these clusters does not add up to zero, but since they are 1-connected to one of the boundaries they can be fused with that boundary. When the cluster is fused with the time-like boundary (for example the pink cluster identified in the final time step), no physical correction is applied. The resultant correction operator from the fusion of the neutral and boundary-neutral clusters is shown in green.

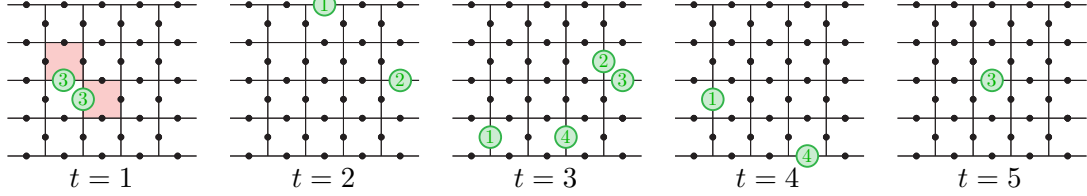


Figure 4.3(iv) **HDRG** $\ell = 2$: The only remaining non-neutral cluster from the previous level is now 2-connected (shown in red). Its elements are fused together and a local correction is returned.

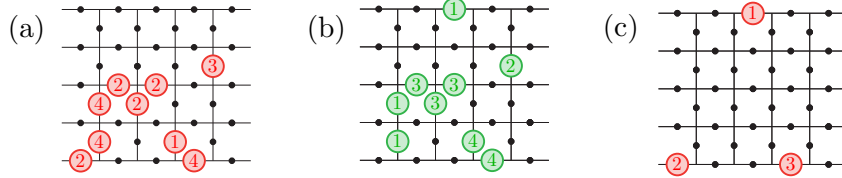


Figure 4.3(v) **Projected Correction**: (a) The final (physical) error layer from part (i) at $t = 5$. (b) The projected correction operator from corrections identified in parts (iii) and (iv), $\tilde{\mathbf{e}}' = \mathbf{e}'_1 \mathbf{e}'_2 \mathbf{e}'_3 \mathbf{e}'_4 \mathbf{e}'_5$. (c) The product of the accumulated error and the projected correction operators. The correction has resulted in a small number of remaining errors, which result primarily from matching clusters to the time-like boundary rather than a space-like one.

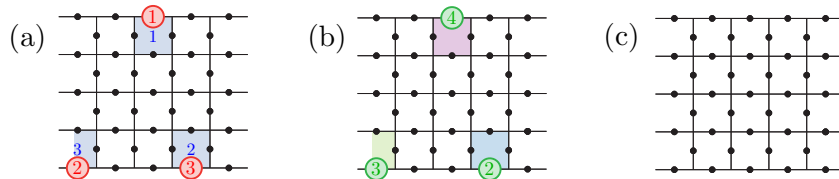


Figure 4.3(vi) **Noise-Free Decoding**: To confirm whether the decoder has succeeded or failed, we must perform an additional round of decoding with noise-free syndrome measurements. (a) The outcomes of the noise-free syndrome measurements. (b) Clustering and correction operators. (c) Result of syndrome noise-free decoding. In this instance all the errors have been eliminated, no logical error has been introduced and the decoding has succeeded.

Figure 4.3: The full decoding algorithm illustrated for a distance 5 code constructed from qudits with $d = 5$.

The success probability of the decoder for the qubit case is shown in figure 4.4, where we find a threshold value of $p_{\text{th}}^{(2)} = 0.0215 \pm 0.0006$. This allows us to compare our decoder with other fault-tolerant qubit decoders, for example the soft-decision renormalisation group decoder by Duclos-Cianci and Poulin achieves a threshold of $p_{\text{th}}^{(2)} = 0.019 \pm 0.004$ [Duclos-Cianci and Poulin, 2014].

Using the same technique of rescaling and fitting the function in equation (3.6) we can determine the threshold $p_{\text{th}}^{(d)}$ for further qudit dimensions. Although our HDRG decoder works for arbitrary qudit dimension d we choose to consider the first few prime dimensions, and in order to determine the asymptotic behaviour we also consider one very high qudit dimension, $d = 7919$, the 1000th prime number. The results are shown as the plot labelled ‘Initialisation levels 0’ in figure 4.5. The plot shows that the threshold achieved by the decoder increases monotonically with increasing qudit dimension, but quickly saturates to a value of $p_{\text{th}}^{(7919)} = 0.042 \pm 0.09$. Previous work performed on the noiseless syndrome measurement version of the HDRG in [Anwar et al., 2014] suggests that this saturation is due to a syndrome percolation effect.

This effect occurs when the syndromes span the lattice from one smooth edge to the other. By drawing a box around a percolating cluster we see that such a box must contain two opposing edges. Thus an operator with support within the box enclosing a spanning cluster could be homologically non-trivial – in other words it may span the lattice and enact a logical Pauli operation.

In order to verify this hypothesis, we performed a simulation of the syndrome percolation threshold. The simulation began by generating the qudit noise and noisy syndrome measurements for each qudit dimension in the same way as for the decoder simulation. However, once the syndrome changes were calculated, we did not employ the decoder but instead performed a simple check to determine whether any 1-connected clusters in $\mathbf{S}'$ percolated the lattice in the x or y directions¹. This information is summarised in the plot labelled ‘Initialisation levels 0’ in figure 4.6. The saturated value for the percolation thresholds for dimension 7919 is around 4.5%, agreeing with our prediction that the HDRG decoder thresholds are upper-bounded

¹The t direction was not checked since we wish to determine whether the percolating cluster is able to support a logical operator once it is collapsed to $\tilde{\mathbf{e}}'$, and any operators acting solely in the t direction are unphysical.

by the percolation threshold.

In the next section we show how to overcome this syndrome percolation effect and achieve improved qudit thresholds by using an initialisation step. This is an algorithm which is run before the HDRG to disrupt the percolating clusters.

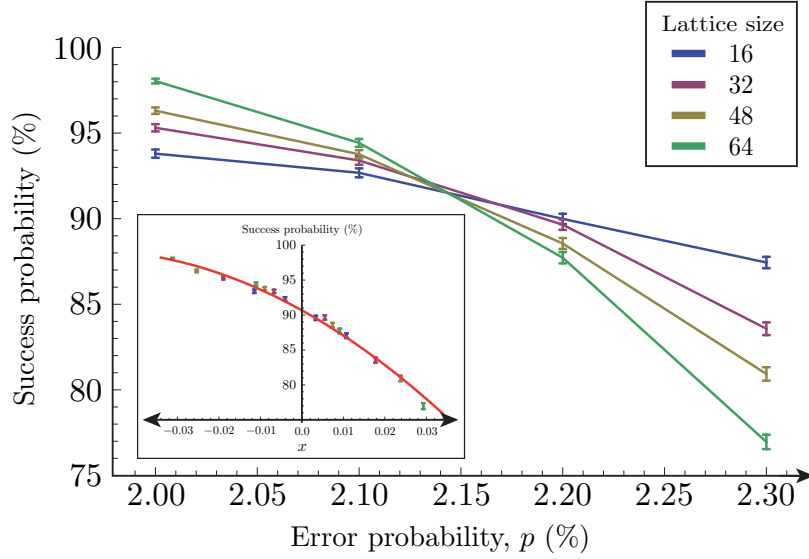


Figure 4.4: An example for the collected simulation data used to estimate the threshold for qubits. The inset figure shows the fitting of the function $P_{\text{succ}}(x) = A + Bx + Cx^2 + DL^{-1/\mu}$ to the rescaled data.

4.4.4 Enhanced-HDRG Decoder

The initialisation step is a subroutine that sweeps through all of the syndromes $\mathbf{S}'$ searching for *neutral sub-clusters* in order to disrupt the percolating clusters. Unlike the HDRG algorithm, the initialisation step does not divide the observed syndrome into disjoint clusters, but simply identifies and eliminates neutral sub-clusters locally.

As with the decoder, the initialisation step has ‘levels’ defined by a metric. However, sub-clusters are more than δ -connected plaquettes, they are 1-connected *paths* of plaquettes, where the charge of the sub-cluster is counted along the entire path. This is because of the fact illustrated in figure 2.9, that a connected path of physical errors will result in a connected neutral path of syndromes.

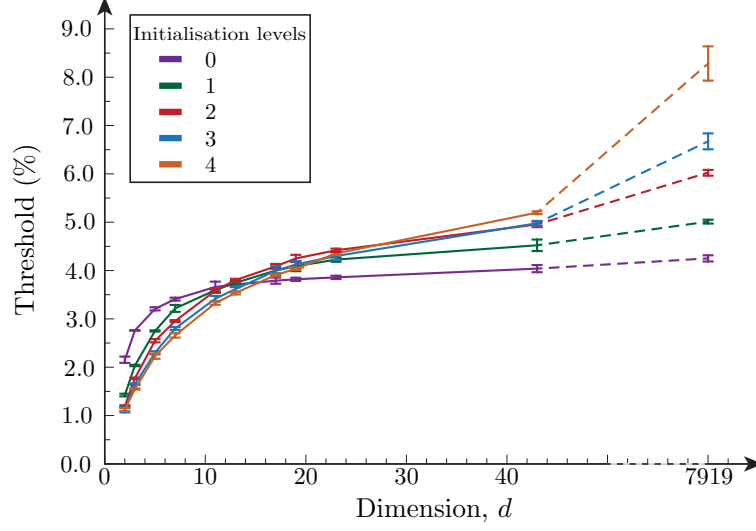


Figure 4.5: A summary of all qudit thresholds for different numbers of rounds of the initialisation step. We have chosen the 1000th prime dimension ($d = 7919$) to represent the asymptotic limit. Although for small qudit dimensions the initialisation step disrupts the syndrome too much and reduces the threshold, we see that in the asymptotic limit there is a clear advantage to using this technique.

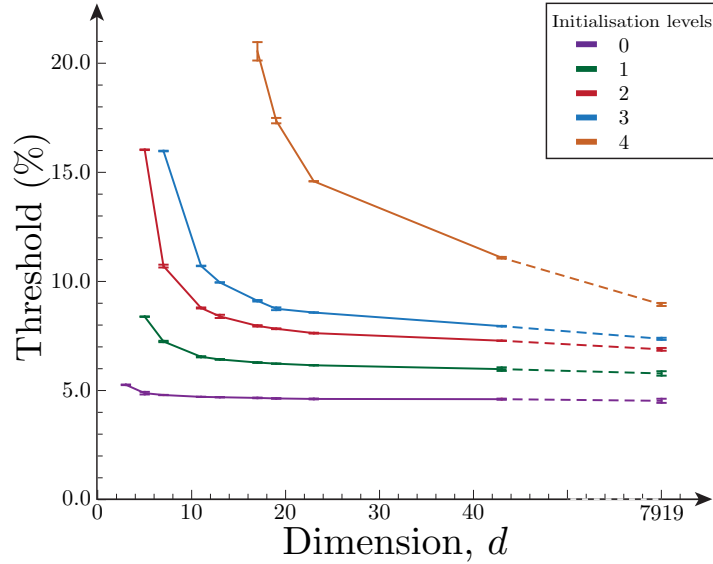


Figure 4.6: A summary of percolation thresholds for different number of rounds of the initialisation step. The initialisation step disrupts the percolation for low qudit dimensions meaning that in some cases a threshold cannot be identified.

An important idea needed to understand the initialisation levels is that of *degeneracy* of paths. If there are $\pm h$ steps in the x direction, $\pm v$ steps in the y direction and $\pm z$ steps in the t direction of the path then its degeneracy is given by

$$D = \frac{(h + v + z)!}{h! v! z!}. \quad (4.3)$$

The initialisation levels are defined sequentially by distance from the central syndrome, and the degeneracy of the paths, favouring those paths with equal distance but higher degeneracy as more likely. In figure 4.7 we show the first three initialisation levels.

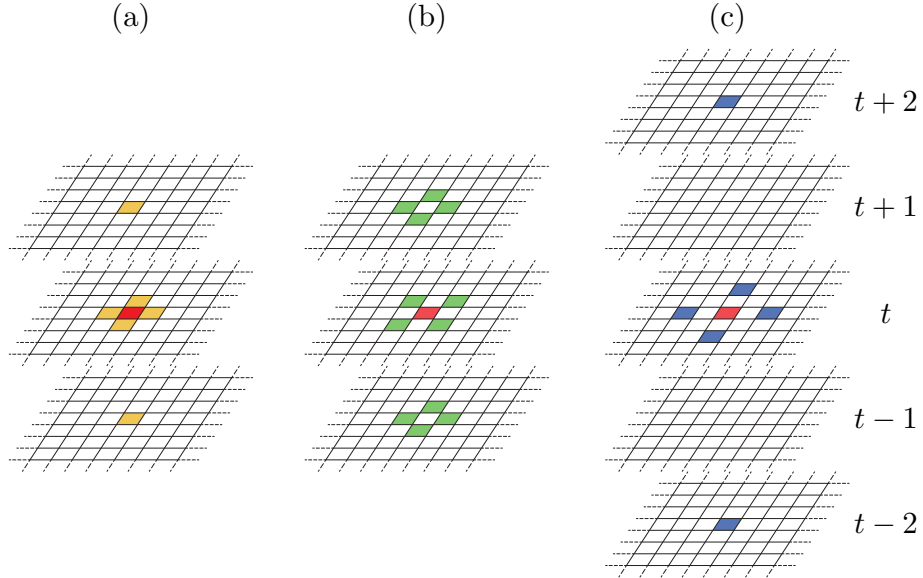


Figure 4.7: An illustration of the first three initialisation levels. (a) First initialisation level. Orange plaquettes are 1-connected to the central red plaquette. (b) Second initialisation level. Green plaquettes are 2-connected to the central plaquette and paths between the central plaquette and any green plaquette have a degeneracy of 2, meaning that there are two distinct paths between the red plaquette and any green plaquette. (c) Third initialisation level. Blue plaquettes are 2-connected to the central plaquette and paths between the central plaquette and any blue plaquette have a degeneracy of 1.

For a given syndrome $s_{t,x,y}$ we denote by $\mathcal{Q} = q_1, q_2, \dots, q_n$ the set of syndromes with the same distance δ from $s_{t,x,y}$, and whose paths connecting to $s_{t,x,y}$ have the

same degeneracy. We denote by p_i a possible path connecting $s_{t,x,y}$ to q_i and refer to each path as a sub-cluster at the initialisation level k . In this step we do not pair syndromes with boundaries – the reason being that the identification of any syndrome that is δ -connected to the boundary would be considered a boundary-neutral sub-cluster, which is too crude even for this ‘quick-and-dirty’ initialisation step. Therefore we consider $\mathcal{Q}$ to be only the set of *syndromes*, excluding any boundaries.

The *initialisation step* $\mathcal{J}_k$ of *depth* k consists of running all the levels $\mathcal{J}_1, \mathcal{J}_2, \dots, \mathcal{J}_k$ where the depth k is determined by the initialisation level. The initialisation procedure $\mathcal{J}_j$ consists of the following steps, beginning with the first non-trivial syndrome:

1. *Neutral sub-cluster*: search over all the paths p_i to each syndrome $q_i \in \mathcal{Q}$. If a neutral path (sub-cluster) is identified go to step 2. If all the paths are searched and none of them are neutral, increment the syndrome index by 1 and repeat step 1.
2. *Sub-cluster annihilation*: annihilate the neutral sub-cluster by fusing the syndromes within the sub-cluster i.e. along the path.

We refer to the HDRG decoder when augmented with initialisation at a certain depth as the enhanced-HDRG. An explicit example for a single initialisation level on a small lattice simulation is illustrated in figure 4.8.

The initialisation step is not efficient because the number of paths to search over increases factorially as the depth increases, but for small numbers of levels the number of sub-clusters to search over is still not too high. For example, at the first level of initialisation, in the worst case there will be 6 paths to check for each element in the bulk of $\mathbf{S}'$ (corresponding to the 6 neighbouring syndromes, see figure 4.7(a)). In general, the initialisation step has an overhead of $C_i L^3$ where C_i is the number of paths for each syndrome for the i th initialisation level. Specifically, $C_i = 6, 24, 6,$ and 48 , for the first four initialisation levels respectively.

We repeated the simulations described for the HDRG decoder once again for the enhanced-HDRG decoder, with up to four initialisation levels. The resulting thresholds are summarised in figure 4.5. We observe that increasing the number of initialisation steps is beneficial for high d , where we see that the asymptotic threshold achieved for four initialisation steps is around 8.3%, almost twice the threshold achieved by the HDRG decoder alone.

Although the improved thresholds for high d suggest that we are successfully able to disrupt the syndrome percolation using this technique, we still observe some saturation of the thresholds. To test this, we performed syndrome percolation simulations using the initialisation step prior to the test for percolation. The results are summarised in figure 4.6. We see that the percolation threshold still upper-bounds the enhanced-HDRG thresholds for the corresponding initialisation step.

Despite its success for very high qudit dimensions the enhanced-HDRG is not useful for low qudit dimensions, where the initialisation step disrupts the syndromes in a way that results in a *lower* threshold. This can be understood as a result of using a decoding strategy that is too local – the neutral sub-clusters identified are in fact fragments of larger errors and the syndromes do not contain enough information to reconstruct them correctly.

We can see that for low qudit dimensions there are only a small number of values a syndrome can take. This means the probability of finding a neutral sub-cluster is high regardless of whether it has been generated by a disjoint error or is in fact part of a larger cluster. Hence the probability of mis-identifying neutral sub-clusters and thus computing an erroneous correction operator is higher for low qudit dimensions than for higher qudit dimensions.

This suggests that the syndromes for very high qudit dimensions contain enough information to allow many rounds of initialisation to keep improving the threshold. For smaller qudit dimensions however, we see there is an optimal number of initialisation rounds that should be performed, for example for $d = 17$ we found that two initialisation levels is optimal.

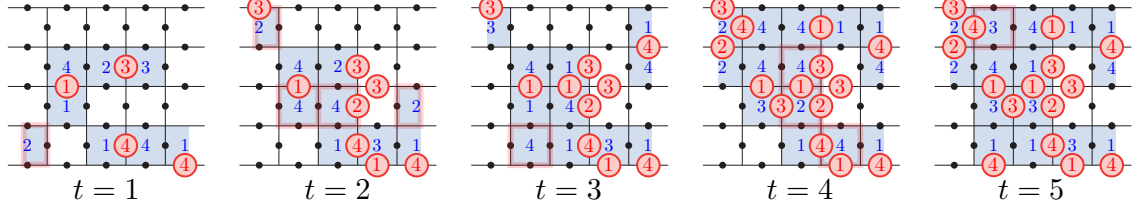


Figure 4.8(i) **Error and Syndrome Histories**: The distance 5 code is comprised of qudits with $d = 5$. The generation of the error and syndrome histories, similar to figure 4.3(i).

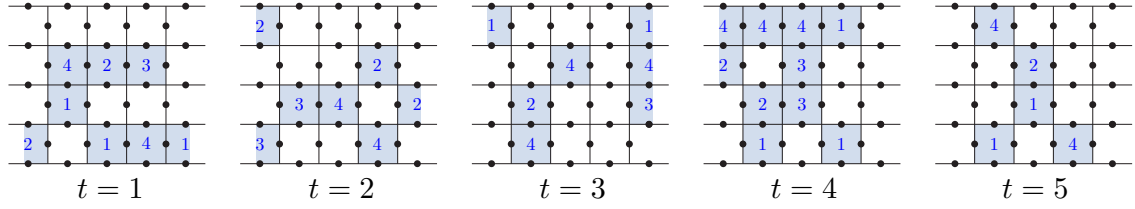


Figure 4.8(ii) **Syndrome Changes History**: The syndrome changes history $\mathbf{S}'$, similar to figure 4.3(ii). Notice how $\mathbf{S}'$ contains a 1-connected percolating cluster of syndromes from one space-like smooth edge to the other.

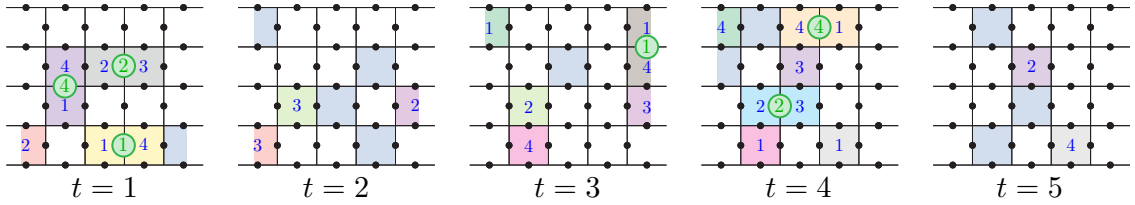


Figure 4.8(iii) **Initialisation $\mathcal{J}_1$** : At the first level of initialisation there are only 6 sub-clusters to search around each plaquette. The algorithm searches over every non-trivial syndrome and checks its 6 neighbouring plaquettes sequentially to see if any of them form a two-element neutral sub-cluster. Once a neutral pairing is found, the two plaquettes are fused together and a single correction operator is returned. Note that in this step we do not pair plaquettes to the physical or the time-like boundary. In the figure each neutral sub-cluster is coloured differently.

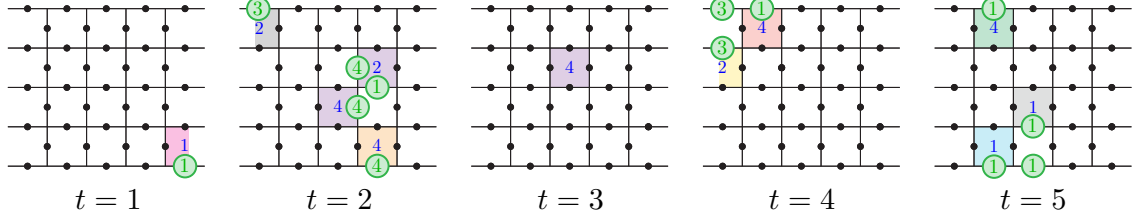


Figure 4.8(iv) **HDRG Decoding**: Notice that the initialisation step has disrupted the percolating cluster. In this step, neutral and boundary-neutral clusters have been identified by running two levels of the HDRG decoder. The correction operator returned by the decoder is shown in green.

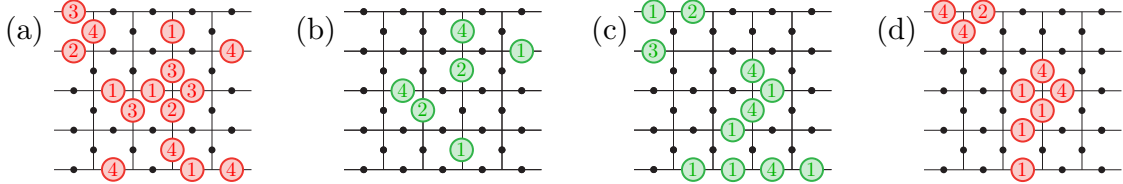


Figure 4.8(v) **Projected Correction**: (a) The accumulated layer of errors from part (i) at $t = 5$. (b) The projected correction operator from running the initialization J_1 identified in part (iii). (c) The projected correction operator obtained from the HDRG decoder identified in part (iv). (d) The resultant errors after taking the operator product of the two correction layers and the accumulated layer of errors.

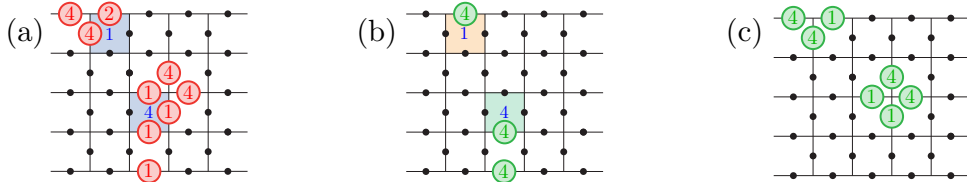


Figure 4.8(vi) **Noise-free Decoding**: (a) Noise-free syndrome measurements. (b) Clustering and correction operators. (c) The result of noise-free decoding. In this case the resultant operators are all members of the stabilizer group so once again the decoding has been successful.

Figure 4.8: The full decoding algorithm, with one initialisation level preceding it, illustrated for a distance 5 code constructed from qudits with $d = 5$.

4.5 Conclusions

We have presented a modified version of the HDRG decoder that was first introduced by Bravyi and Haah in [Bravyi and Haah, 2013] and studied its decoding performance for the surface code with noisy syndrome measurements. This study demonstrates the flexibility of the HDRG and its applicability to more complex noise models than those previously considered.

The main difference in our version is the use of a more refined metric which has led to an improved threshold. We have chosen the Manhattan distance metric δ , whereas Bravyi and Haah considered the d_∞ (box-like) metric. In our investigations we discovered that the majority of the syndromes are cleared at the first level of decoding. This means that having a more refined metric matters more at $\ell = 0$ than it does at higher decoding levels. The δ metric ensures that the clusters at the first decoding level are as closely connected as possible by allowing a single syndrome to be connected only to its 6 nearest neighbour plaquettes.

We found that, similarly to the measurement noise-free setting, for all but the smallest qudit dimensions, syndrome percolation places an upper bound on the decoder threshold for the HDRG decoder. We have demonstrated that this can be overcome by adopting an extra initialisation step which scans for locally neutral sub-clusters. In doing this and eliminating any it finds, it breaks up the percolating lattice and allows the decoder to succeed above the percolation threshold. This has a particularly stark effect for high dimensions, increasing the threshold by almost a factor of two for just 4 initialisation levels.

The uncorrelated noise model chosen here was adopted for ease of comparison with other decoders. However, an uncorrelated noise model is unlikely to be encountered in experiment. When the dimension is high, in an isotropic depolarising noise model, there would be a high correlation between the presence of X -type and Z -type errors. A decoder which used this information might achieve significantly higher thresholds. Nevertheless, we expect the decoder presented here to possess an error threshold for any noise model acting independently and identically distributed (i.i.d.) with respect to individual qudits and also non-i.i.d. noise models where the correlation between qudit errors is limited. Testing these possibilities is a pertinent open question.

A remarkable feature of this decoder is the independence of its run time com-

plexity with respect to qudit dimension. This is in stark contrast to other known qudit decoders. For example, the soft-decision renormalisation group decoder in the fault-tolerant setting [Duclos-Cianci and Poulin, 2014] has a straightforward implementation in higher dimensions but comes with a cost of a polynomial overhead in d which means its applicability is limited to low dimensions.

The surface code is a useful test-bed for many ideas in quantum error correction, and importantly it is experimentally viable. However, a second prominent class of topological stabilizer codes, called colour codes, exist. They have favourable properties for fault-tolerant quantum computing, namely that in 3 spatial dimensions they are capable of supporting a transversal non-Clifford gate. In the next chapter we introduce colour codes and generalise them to qudits and prove that the transversality of the non-Clifford gate remains in the qudit construction.

Chapter 5

Qudit Colour Codes in All Spatial Dimensions

*Two-level quantum systems, qubits, are not the only basis for quantum computation. Advantages exist in using qudits, d-level quantum systems, as the basic carrier of quantum information. In this chapter we show that colour codes – a class of topological quantum codes with remarkable transversality properties – can be generalised to the qudit paradigm. In recent developments it was found that in three spatial dimensions a qubit colour code can support a transversal non-Clifford gate, and that in higher spatial dimensions additional non-Clifford gates can be found, saturating Bravyi and König’s bound [Phys. Rev. Lett. **110**, 170503 (2013)]. Furthermore, by using gauge fixing techniques, an effective set of Clifford gates can be achieved, removing the need for state distillation. We show that the qudit colour code can support the qudit analogues of these gates, and show that in higher spatial dimensions a colour code can support a phase gate from higher levels of the Clifford hierarchy which can be proven to saturate Bravyi and König’s bound in all but a finite number of special cases. The methodology used is a generalisation of Bravyi and Haah’s method of triorthogonal matrices [Phys. Rev. A **86**, 052329 (2012)] .*

A published version of this work is also available [[Watson et al., 2015b](#)].

5.1 Introduction

As we have seen in earlier chapters, quantum technologies are often developed in the qubit paradigm. Qubits are a natural choice because binary is the language of classical technologies. However, even here, despite the prevalence of binary, its supremacy is questionable. Indeed, the use of balanced ternary, a 3-state classical logic has been proposed [Knuth, 1997]. In this chapter we present a qudit generalisation of a powerful class of quantum error correcting codes, the colour codes [Bombin and Martin-Delgado, 2006, 2007b]. Along with surface codes, they constitute the most successful topological codes. Qubit colour codes have several advantages over qubit surface codes, and in this chapter we show these features can be transferred over into the qudit setting.

We expect many of the results derived for qudit surface codes to hold for the colour codes. Taking as an example the threshold results found for the HDRG decoder in the qudit surface code in Chapter 4, we would expect that the thresholds for the same error models on the qudit colour codes would scale in a similar way. In the present chapter we concentrate on some more abstract properties of the codes in order to prove results related to the set of gates that can be implemented transversally on them. We expect this work, combined with further investigations into the error correcting properties of the qudit colour codes, to yield information about the best way to optimise the quantum error correction codes selected as the basis of quantum computers. Such analysis will be discussed in more detail in the concluding remarks in the next chapter.

There has been some prior investigation into qudit colour codes for prime-power dimensions [Sarvepalli, 2010] and colour codes based on more general groups [Brell, 2015], although these works were restricted to 2D topologies. In this chapter we generalise colour codes to any Hilbert space dimension d and spatial dimension μ , except where μ factorial is a multiple of d ¹. Specifically, given any lattice suitable for constructing qubit colour codes [Bombin and Martin-Delgado, 2007a; Bombin et al., 2013], we show how to use the same lattice to construct a qudit colour code. For qubits, a non-Clifford gate can be implemented in colour codes in 3 and higher

¹In this chapter we also consider two important examples of cases where $\mu!$ and d share a common factor. In particular, in Section 5.6.1 we consider the 3D code for $d = 3$ and 6.

spatial dimensions transversally, that is, by a tensor product of local unitary gates, an inherently fault-tolerant procedure [Knill et al., 1996]. We have introduced the notion of transversality in Section 1.4.2. To avoid confusion between the spatial dimension of the lattice and the Hilbert space dimension of the qudit, we denote the former by the letter μ and the latter by d .

It has been shown by Bravyi and König [Bravyi and König, 2013] that a quantum error correcting code in μ spatial dimensions can support a gate with constant depth from at most the μ th level of the Clifford hierarchy, see equation (1.6). The fact that colour codes can be shown to saturate this bound with transversal gates is a very promising feature, and when combined with gauge fixing techniques, for example those outlined in [Anderson et al., 2014; Jochym-O'Connor and Laflamme, 2014; Paetznick and Reichardt, 2013], enables universal quantum computation without the need for magic state distillation (see Chapter 1). The structure of the qudit Clifford group is very different from its qubit counterpart [Appleby, 2005; Howard and Vala, 2012]. Nevertheless, we will demonstrate in this chapter that 3D colour codes also provide transversal non-Cliffords in the qudit case.

The main technique which we employ in this chapter is a bipartition of the vertices in the graph that defines the code [Bombin et al., 2013; Bombin, 2015; Kubica and Beverland, 2014], into unstarred and starred vertices. We call this the star-bipartition, to distinguish it from the other important colourings which label the colour codes. The commutation properties of the stabilizer and logical operators of the colour codes in the qubit setting can be reduced to the fact that the pairs of operators $X \otimes X$ and $Z \otimes Z$ commute. The star-bipartition we introduce replaces operators with their complex conjugate on a subset of the vertices, for example, replacing the above operators with $X \otimes X^*$ with $Z \otimes Z^*$, respectively. Crucially, this latter pair of operators does commute for qudits of any dimension d , and this becomes the starting point for a generalisation of the colour codes from qubits to qudits.

Furthermore, the star-bipartition provides a general framework for constructing transversal gates from higher levels of the Clifford hierarchy. While elements of this technique can be seen in earlier work [Bombin et al., 2013; Bombin, 2015; Kubica and Beverland, 2014; Sarvepalli, 2010], this is the first time that it has been exploited systematically for qudit colour codes. The second key technical component of this

chapter is a generalisation of the triorthogonal matrix technique by Bravyi and Haah [Bravyi and Haah, 2012]. These matrices define quantum error correcting codes with transversal non-Clifford gates from the third level of the Clifford hierarchy (CH). In this chapter we generalise these results to matrices that define codes capable of supporting gates from arbitrarily high levels of the CH.

We begin this chapter by introducing the qubit colour code. Later in the chapter we shall derive triorthogonality conditions (and generalisations thereof) for matrices defining qudit codes under the star-bipartition assumption. Hence we shall show that colour codes obey these conditions and can be constructed such that their transversality properties saturate the Bravyi and König bound.

5.2 Qubit Colour Codes

Colour codes [Bombin and Martin-Delgado, 2006, 2007b] are a class of topological qubit stabilizer codes that may be defined on a topological space of any spatial dimension $\mu \geq 2$ [Bombin and Martin-Delgado, 2007a]. The μ -dimensional manifold is cellulated into a lattice made up of objects called k -cells for all spatial dimensions $0 \leq k \leq \mu$. For example, vertices are 0-cells, edges are 1-cells and a 2-cell is a cell defined on 2-dimensional objects, the faces (or plaquettes) of the lattice.

In this section we shall give a brief description of the key features of the qubit construction of colour codes in two and higher spatial dimensions. These constructions will build intuition for the μ -dimensional qudit construction, given in Section 5.4, and which is the main subject of this chapter.

5.2.1 Colour Codes in Two Spatial Dimensions

The colour code is defined on a lattice $\mathcal{L}_c$ embedded in a 2D orientable manifold, with the qubits associated with the vertices of the lattice. Specifically, in 2D the lattice is trivalent so three edges meet at a vertex. As a consequence, both its plaquettes and edges are 3-colourable, and it is guaranteed that all cycles in the corresponding graph are even. Importantly, this means that all plaquettes contain an even number of vertices. There are three known regular colour code lattices in 2-dimensions [Landahl et al., 2011], all of which obey these valency and colourability conditions, and all of

which have similar properties. One key difference is that the qubit overhead varies amongst these lattices in order to achieve a code of a fixed distance. In this section we shall consider only the hexagonal lattice for simplicity.

Each plaquette of the lattice P supports two stabilizer generators,

$$S_{X,P} = \otimes_{i \in P} X_i, \quad S_{Z,P} = \otimes_{i \in P} Z_i. \quad (5.1)$$

Here the $S_{X,i}$ are the X -stabilizer generators and the $S_{Z,j}$ are the Z -stabilizer generators. Due to the lattice construction these operators may be defined on disjoint sets of vertices, in which case they trivially commute. If they have shared vertices, the construction ensures the overlap is on a single edge or on a whole plaquette. In both of these cases the intersection contains an even number of qubits and hence they commute. There are no stabilizers associated with the edges or vertices of $\mathfrak{L}_c$.

The colour code can be embedded in a topologically non-trivial 2-dimensional surface but the most widely studied colour code in 2 dimensions is the *triangular* code, which has a trivial topology equivalent to that of a disc. When deformed to a triangle, the code has three edges, each coloured differently and associated with a boundary. String-like operators on the code also have a colour associated to them, and as with the planar code the end-points of the operator may be within the bulk of the code, or the operator may terminate at a boundary. The key difference is that string-like operators in the colour code may only terminate at the boundary of the same colour as the operator, see figure 5.1.

The triangular colour code supports a single encoded qubit. There are several possible choices of operator basis for this qubit but in this thesis we choose to define the pair of logical Pauli operators transversally,

$$\bar{X} = \otimes_{i \in \mathfrak{L}_c} X_i, \quad \bar{Z} = \otimes_{i \in \mathfrak{L}_c} Z_i. \quad (5.2)$$

By multiplying these logical operators by a subset of stabilizers they can be localised to an edge of the lattice. Therefore to ensure the anticommutation of these operators we require each of the edges of the planar code to contain an odd number of qubits. We also see that the minimum weight of such an operator is the length of an edge, hence the edge length corresponds to the distance of the triangular colour code.

The two smallest instances of the triangular colour code are shown in figure 5.1.

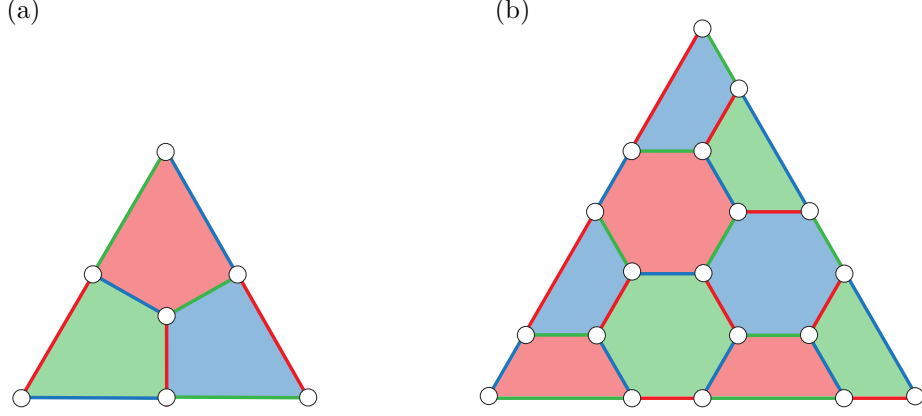


Figure 5.1: The two smallest instances of the 2D colour code defined on the hexagonal lattice. Qubits are indicated as white circles on the vertices of the lattice. (a) The distance 3 code. (b) The distance 5 code. Note that the edges are coloured blue/red, red/green and green/blue, corresponding to green, blue and red boundaries respectively. The meaning of the colour label with respect to a boundary is that strings of that colour can end on that boundary.

Since they are CSS codes, one can show that the Clifford group gates can be realised in the 2D colour code. Using the Eastin-Knill theorem [Eastin and Knill, 2009] this implies that a non-Clifford gate cannot be implemented transversally in such a code.

5.2.2 Qubit Colour Codes in Higher Spatial Dimensions

In [Bombin and Martin-Delgado, 2007a] it was shown that qubit colour codes can be generalised to arbitrary spatial dimensions. The μ -dimensional colour code is defined on a lattice that is $(\mu + 1)$ -valent such that the μ -cells and 1-cells (edges) of the lattice are $(\mu + 1)$ -colourable. By convention the X -stabilizers are defined on the μ' -cells C and the Z -stabilizers are defined on the $(\mu - \mu' + 2)$ -cells P of the lattice, where $2 \leq \mu' \leq \mu$

$$S_{X,C} = \otimes_{i \in C} X_i, \quad S_{Z,P} = \otimes_{i \in P} Z_i. \quad (5.3)$$

The stabilizer generators are guaranteed to commute in this construction. Those of the same type (X or Z) trivially commute, while those of different types commute since they always share an even number of qubits. Note that due to the freedom in the choice of μ' for a given spatial dimension μ several different colour codes may be defined. Regardless of this choice, the logical operators are defined transversally, see equation (5.2).

Such codes can be constructed [Bombin, 2015] by starting with a closed hyperspherical lattice and then removing a vertex to ‘puncture’ the surface. Alternatively, they may be constructed directly by defining a suitable boundary on a regular lattice structure [Watson et al., 2015b]. The punctured lattice encodes a single qubit and has the attractive feature of a transversal non-Clifford T gate, for a suitable choice of μ' . The qubit colour codes have many useful properties, for more details see [Bombin and Martin-Delgado, 2006, 2007a; Bombin, 2015].

In this chapter we generalise qubit colour codes to a qudit construction and show that a colour code in μ spatial dimensions can support a transversal gate from the μ th level of the Clifford hierarchy as long as $\mu' \neq \mu - \mu' + 2$ and $\mu!a_\mu \neq 0$ modulo d (where ‘!’ means factorial), where the meaning of the constant a_μ shall be explained in Section 5.3.1.

5.3 Universality in Qudit Stabilizer Codes

Qudit stabilizer codes were introduced in Section 1.5. In Section 1.4.1 we introduced the Clifford hierarchy (CH), with the definition in equation (1.6) holding for qudits as well as qubits. Throughout the rest of this chapter, unless stated otherwise, the qudit dimension d is assumed to be any integer greater than two.

In prime qudit dimensions three Clifford group gates H , S and $\Lambda(X)$ are known to generate the whole Clifford group [Clark, 2006; Gottesman, 1999]. H is the qudit Hadamard gate (also known as the discrete Fourier transform, see equation (1.7) for the qubit analogue),

$$H = \frac{1}{\sqrt{d}} \sum_{j,k \in \mathbb{Z}_d} \omega^{jk} |j\rangle \langle k|, \quad (5.4)$$

where $\omega = e^{2\pi i/d}$.

The S gate is the generalisation of the qubit S gate (see equation (1.8)),

$$S = \sum_{j \in \mathbb{Z}_d} \omega^{j^2} |j\rangle \langle j|, \quad (5.5)$$

and $\Lambda(X)$ is the controlled- X gate (also known as the SUM gate),

$$\Lambda(X) = \sum_{j,k \in \mathbb{Z}_d} |j\rangle_c |k \oplus j\rangle_t \langle j|_c \langle k|_t, \quad (5.6)$$

where c and t are the control and target qudits, respectively. The qubit analogue of this gate is given in equation (1.9).

As in the qubit case, the set of gates $\mathcal{P}_l$ in the CH contains all gates from lower levels of the hierarchy, $\mathcal{P}_m$ for $m < l$. To refer to gates in level l of the hierarchy but not level $l - 1$ we say that the level l of the hierarchy is the lowest level of the hierarchy for which the gate is a member.

5.3.1 Achieving Universality in Qudit Stabilizer Codes

In prime dimensions, it is known that it suffices to supplement the Clifford group with just one non-Clifford gate from the third level of the CH in order to obtain a universal set of gates [Campbell et al., 2012]. Such a gate is not unique, and in the qubit case, the T gate $\text{diag}(1, e^{i\pi/4})$ is usually chosen for this purpose, see equation (1.10). For the qudit case, we choose the following particularly convenient definition for the T gate, which is valid in all dimensions except when $d = 2, 3, 6$ [Howard and Vala, 2012; Zhu, 2010],

$$T = \sum_{j \in \mathbb{Z}_d} \omega^{j^3} |j\rangle \langle j|. \quad (5.7)$$

It is a consequence of lemmas 1 and 2 (below) that for $d \neq 2, 3, 6$ this gate is non-Clifford and inhabits the third level of the CH. For $d = 2$ the T gate takes a different form, given in equation 1.10, while for $d = 3$ and $d = 6$, the gate defined in equation (5.7) is not non-Clifford, indeed it reduces to the Pauli Z since $j^3 = j$ modulo 3 and modulo 6. However, the following definition provides a suitable alternative T gate for these dimensions. The gate is non-Clifford and in the third

level of the CH:

$$T_{3,6} = \sum_{j \in \mathbb{Z}_d} \gamma^{j^3} |j\rangle \langle j|, \quad (5.8)$$

where $\gamma^3 = \omega$ and where the function in the exponent j^3 is evaluated in regular arithmetic (or equivalently modulo $3d$). When we refer to ‘the T gate’ in this chapter we will always mean a gate of the form of equation (5.7) or (5.8), depending on the qudit dimension under consideration. For notational convenience we suppress the dependence of T on d , since it will always be clear from the context which T gate we require.

Notice how the T gate has a cubic power in the exponent of ω (or γ), in contrast to the quadratic power in the case of the S gate, see equation (5.5). In general, there is a close correspondence between the order of the polynomial in the exponent of ω and the lowest level of the CH the phase gate belongs to. Let us define the following family of phase gates R in terms of a polynomial function $f_r(j)$ of degree r , such that $r \leq d$ with coefficients $a_m \in \mathbb{Z}_d$, so $f_r(j) := \sum_{m=0}^r a_m j^m$:

$$R_{f_r} = \sum_{j \in \mathbb{Z}_d} \omega^{f_r(j)} |j\rangle \langle j|. \quad (5.9)$$

One can then prove the following useful lemmas.

Lemma 1. *For all d , all $r \leq d$ and all functions $f_r(j)$, the gate R_{f_r} is in the r th level of the Clifford hierarchy.*

Proof. The proof of this is simple and concise. We begin by calculating

$$R_{f_{r-1}} = R_{f_r} X R_{f_r}^\dagger X^\dagger, \quad (5.10)$$

where f_{r-1} is a new function $f_{r-1}(j) = f_r(j+1) - f_r(j) = ra_r j^{r-1} + \dots$. Expanding out, the degree r terms cancel, so the leading term is $ra_r j^{r-1}$ and all of the other terms are of degree $r-2$ or smaller. We now observe that if $r=1$, R_{f_r} is a Pauli operator. Using the definition of the Clifford hierarchy in equation (1.6), the lemma follows by induction. $\square$

Lemma 2. *For all d , all $r \leq d$ and all functions $f_r(j)$ satisfying $r!a_r \not\equiv 0 \pmod{d}$, the gate R_{f_r} is not in the $(r-1)$ th level of the Clifford hierarchy.*

Proof. We begin this proof by noting that if the gate is in the r th level of the hierarchy, then applying the inductive transformation in the previous proof $r - 1$ times must return a Pauli operator. It follows that applying it a total of r times results in an operator proportional to the identity.

Consider an operator in the r th level of the CH conjugated by X according to equation (5.10), resulting in $R_{f_{r-1}}$, where $f_{r-1}(j) = ra_r j^{r-1} + \dots$. By iterating the conjugation by X we obtain $f_{r-2}(j) = r(r-1)a_r j^{r-2} + \dots$. Repeating this $r - 1$ times we finally obtain $f_1(j) = r!a_r j + c$, so that if the corresponding operator is proportional to the identity then the function is constant, meaning that $r!a_r = 0 \pmod{d}$. This violates the statement above, and implies that the original operator R_{f_r} is in the $(r - 1)$ th level (or lower) of the Clifford hierarchy. $\square$

These two lemmas provide us with a simple way to generate gates at all levels of the Clifford hierarchy as required. They fail when $r!a_r = 0 \pmod{d}$, which was the case above for $r = \mu = 3$ and $d = 2, 3, 6$. In those exceptional cases, gates can be discovered by moving to higher roots of unity as illustrated by equation (5.8).

5.4 Qudit Colour Codes

In Section 5.2 we introduced the qubit colour codes. Here we shall describe the qudit generalisation of the colour codes in μ spatial dimensions. In the next section we prove their transversality properties.

Generalising the qubit codes introduced in Section 5.2, a lattice $\mathfrak{L}_c$ is called a μ -colex (colex is short for ‘colour complex’) whenever

1. it is a cellulation of an orientable manifold without a boundary; and
2. every vertex has $(\mu + 1)$ neighbours $((\mu + 1)$ -valency); and
3. the μ -cells are $(\mu + 1)$ -colourable.

Any μ -colex defines a qubit colour code. To encode information we must puncture the μ -colex by removing a vertex, along with any edges and cells containing that vertex, to create a boundary. For example, the smallest 3-dimensional colour code

with a boundary is a 15-qubit code defined on the lattice illustrated in figure 5.2(a). In such a colour code, the stabilizer group is generated by face operators (with Z operators assigned to the vertices around the face or 2-cell) and cell operators (with X operators assigned to the vertices around each 3-cell), see figure 5.2(b) and (c).

The stabilizer generators are guaranteed to commute in this construction. Those of the same type (X or Z) trivially commute, while those of different types commute since they always meet at an edge, which by definition is associated with two vertices. As already remarked, $X \otimes X$ and $Z \otimes Z$ commute only in $d = 2$. However, by replacing one operator of each pair with its complex conjugate (with respect to the computational basis) we produce a pair of operators that commute for all d . These are $X \otimes X^*$ and $Z \otimes Z^*$ ¹. Thus we see that as long as a pair of X - and Z -type stabilizers have in common an equal number of starred qudits, they will commute.

To define colour codes in all qudit dimensions, we need, therefore, to find a μ -colex construction which allows us to take advantage of the commutation of $X \otimes X^*$ and $Z \otimes Z^*$. This can be achieved by identifying and exploiting a bipartition or bicolouring of the graph defining the lattice. To avoid confusion with other colourings of the lattice important for colour codes we call this the star-bipartition.

We note that a similar construction was defined (for prime power qudit dimension only) in [Sarvepalli, 2010]. Here we go further and show that the star-bipartition is the starting point for an identification of a broad family of transversal gates on these codes, including non-Clifford gates saturating Bravyi and König's bound [Bravyi and König, 2013].

5.4.1 Star-bipartition

The star-bipartition is a bipartition of the colour code lattice $\mathfrak{L}_c$ with no boundaries. It thus divides the set of lattice vertices into starred and unstarred vertices where neighbouring vertices always belong to different sets. Here we state the following useful bipartition lemmas.

¹Throughout this chapter, a starred operator denotes the complex conjugate of the operator with respect to the computational basis. The sharp-eyed reader will note that $X^* = X$. We write the star explicitly on a real matrix here, and throughout, to emphasise symmetry and to simplify notation.

Lemma 3 (Star-bipartition lemma). *Let $\mathfrak{L}_c$ be a μ -colex, then its vertices can be 2-coloured into starred and unstarred sets, $v_\star$ and $v_\bullet$, respectively.*

Using ‘ $|\cdots|$ ’ to denote the number of elements in a set, we also have

Lemma 4 (Starring of cells lemma). *Let $\mathfrak{L}_c$ be a μ -colex, so that its vertices are partitioned into $v_\bullet$ and $v_\star$ according to lemma 3. It follows that*

1. $|v_\star| = |v_\bullet|$; and
2. any k -cell C with $0 < k \leq \mu$ contains a equal number of vertices from each partition, so $|C \cap v_\star| = |C \cap v_\bullet|$.

Lemmas 3 and 4, or variants thereof, were already proved in prior research [Bombin and Martin-Delgado, 2006, 2007a; Bombin, 2015; Kubica and Beverland, 2014; Watson et al., 2015b] so we do not include a proof here.

The above results concern a lattice without a boundary. However, if one punctures the code by removing a starred vertex then we have

Corollary 1. *Let $\mathfrak{L}_c$ be puncturing of a μ -colex, with the inherited bipartition into $v_\star$ and $v_\bullet$. Then*

1. $|v_\bullet| = |v_\star| + 1$; and
2. any k -cell C with $0 < k \leq \mu$ contains a equal number of vertices from each partition, so $|C \cap v_\star| = |C \cap v_\bullet|$.

Property 1 immediately follows from point 1 of lemma 4 as a single starred vertex has been removed. Property 2 is identical to its partner in lemma 4. The punctured lattice only keeps cells that did not contain the punctured vertex, and so the property is directly inherited.

5.4.2 Qudit Colour Codes in Arbitrary Spatial Dimensions

Before we write down the stabilizer generators for these codes, we shall use the star-bipartition to define notation for an important family of transversal operators which we call *star-conjugate transversal*. Throughout this chapter, when discussing colour codes we identify qudits with the vertices of the lattice defining the code. Let

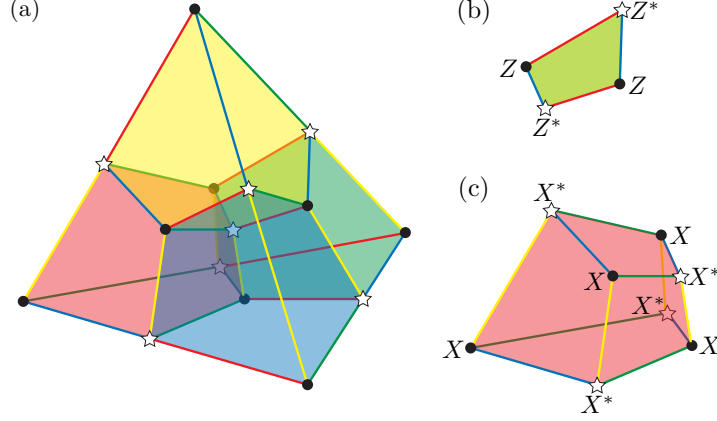


Figure 5.2: (a) The smallest instance of the qudit 3D colour code with the X -type stabilizers coloured red, green, blue and yellow. The 1-cells (edges) of the code are also coloured. The 15 vertices which represent the qudits are coloured so the set $v_\bullet$ is represented by black circles and the set $v_\star$ is represented by white stars. (b) A single Z -stabilizer of the tetrahedral 3D colour code. The plaquette can take the colour yellow if considered as a face of the green 3-cell, and vice versa. (c) A single X -stabilizer of the tetrahedral colour code.

v denote a set of vertices (and thus the corresponding qudits). Let us introduce the notation

$$U[v] = U^{\otimes |v|} \quad (5.11)$$

to denote the tensor product of unitary U acting on each qudit identified with the vertices in v .

We call an operator *star-conjugate transversal* when it has the following form

$$\tilde{U} = U[v_\bullet] \otimes U^*[v_\star]. \quad (5.12)$$

In other words, $\tilde{U}$ consists of U applied to all unstarred vertices and U^* applied to all starred vertices.

To define qudit colour codes we need to introduce two types of stabilizer generators which are defined with respect to cells of different dimensions within the lattice. As in the qubit case, we associate X -type stabilizer generators with the μ' -cells of the lattice and Z -type stabilizers with the $(\mu - \mu' + 2)$ -cells [Bombin and Martin-Delgado,

2007a]. In this section, for simplicity of presentation, we will take the example of $\mu' = 3$ so the Z -type stabilizers are associated with plaquettes (2-cells) and X -type stabilizers are associated with 3-cells, although the construction holds in the more general case, which we shall consider later. For example in 3D, the X -stabilizers act on the vertices contained in a 3-cell of the lattice, and the Z -stabilizers act on the vertices contained in a 2-cell (plaquette). This can be seen in figure 5.2.

Setting $\mu' = 3$ the stabilizer generators therefore take the form

$$\begin{aligned} S_{X,C} &= X[v_\bullet \cap C] \otimes X^*[v_\star \cap C], \\ S_{Z,P} &= Z[v_\bullet \cap P] \otimes Z^*[v_\star \cap P], \end{aligned} \tag{5.13}$$

for all 2-cells P , and 3-cells C of the lattice. Recall that we write the conjugate operator X^* explicitly to emphasis the ubiquity of the starring, even though X is a real operator in the computational basis and $X^* = X$. It is not just the $S_{Z,P}$ operators where starring is non-trivial, but many logical operators also require starring.

Let us denote the group generated by $S_{X,C}$ operators S_X , and the group generated by $S_{Z,P}$ operators S_Z . All elements of S_X trivially commute, as do all elements of S_Z . It remains to show that all elements of S_X commute with all elements of S_Z . This follows from the fact that $X \otimes X^*$ commutes with $Z \otimes Z^*$. The above construction ensures that whenever cell C and cell P overlap they overlap on an equal number of starred and unstarred vertices, which is a point discussed further in Section 5.5.3. Hence these stabilizer generators commute as required. Note that the qudit $d = 2$ case is included in our definition.

We noted above that, in the constructions we consider, the code will contain one more unstarred qudit than there are starred qudits. We can thus define the logical encoded Pauli operators for the code as

$$\begin{aligned} \bar{X} &= \tilde{X} = X[v_\bullet] \otimes X^*[v_\star], \\ \bar{Z} &= \tilde{Z} = Z[v_\bullet] \otimes Z^*[v_\star]. \end{aligned} \tag{5.14}$$

These logical operators are transversal like their qubit counterparts, see equation 5.2, where the star-bipartite lattice structure means that the qudit logical Pauli operators have some additional structure. The fact that $|v_\bullet| = |v_\star| + 1$ ensures that the logical operators satisfy the same commutation properties as X and Z . One can verify that

these operators commute with the stabilizer operators defined above by recalling that each 2-cell P and 3-cell C contains an equal number of starred and unstarred vertices, see lemma 4.

5.5 $m^\star$ -orthogonal matrices and codes

Bravyi and Haah [Bravyi and Haah, 2012] introduced a powerful framework for defining codes with a transversal non-Clifford gate – codes defined in terms of *triorthogonal matrices*. Here we introduce a significant generalisation to Bravyi and Haah’s approach, extending to qudits codes with transversal gates from higher up in the Clifford hierarchy.

5.5.1 Matrix Representation of Quantum Codes

We begin by reviewing how Bravyi and Haah represent CSS codes with matrices. They use a matrix G over $\mathbb{Z}_2$ that has linearly independent rows under modulo 2 arithmetic. The matrix G is broken up into two blocks G_1 (with k rows) and G_0 (with s rows). This defines a quantum code with k logical qubits. The elementary logical basis state $|0_L\rangle$ is written

$$|0_L\rangle = \frac{1}{\sqrt{2^s}} \sum_{f \in \text{span}[G_0]} |f\rangle = \frac{1}{\sqrt{2^s}} \sum_{y \in \mathbb{Z}_2^s} |y^T \cdot G_0\rangle, \quad (5.15)$$

where ‘ $\cdot$ ’ denotes matrix multiplication modulo 2, and we take advantage of the fact that the rows of G_0 are linearly independent to represent the terms in the superposition by a row vector y transposed and multiplied with G_0 . For the other logical computational basis states $|x_L\rangle$, where $x \in \mathbb{Z}_2^k$, we have

$$|x_L\rangle = \frac{1}{\sqrt{2^s}} \sum_{y \in \mathbb{Z}_2^s} |y^T \cdot G_0 \oplus x^T \cdot G_1\rangle, \quad (5.16)$$

where addition of row vectors is elementwise modulo 2 as acknowledged by the symbol $\oplus$. Note that in the special case that $k = 1$, as for all the colour codes we consider, x is a scalar (or a 1×1 row vector) and the transpose operation is trivial. For qudits, quantum codes are defined by a matrix G taking elements from $\mathbb{Z}_d$, so

that

$$|x_L\rangle = \frac{1}{\sqrt{d^s}} \sum_{y \in \mathbb{Z}_d^s} |y^T \cdot G_0 \oplus x^T \cdot G_1\rangle. \quad (5.17)$$

The key differences are that we now sum over all $y \in \mathbb{Z}_d^s$ and that arithmetic is modulo d .

In the CSS formalism, a quantum code has a stabilizer group which is generated by a product of two distinct generating sets, the X -stabilizer generators, which are tensor products of X and I alone, and the Z -stabilizer generators, which are tensor products of Z and I alone. In addition, one must define logical operators, and in the CSS formalism, logical encoded X operators consist of a tensor product of X and (optionally) I alone, and the logical encoded Z consists of a tensor product of Z and (optionally) I alone. Once the X -stabilizer generators and logical $\bar{X}$ operator (or operators if there are multiple encoded qubits) are defined, the remaining Z -stabilizer generators are fixed by their conjugation relations. In the qudit setting, G_0 will again define the X -stabilizer generators for the code, via the mapping k to X^k , e.g. the row vector $(0, 0, 0, 1, 2, 3)$ defines the operator $I \otimes I \otimes I \otimes X \otimes X^2 \otimes X^3$, and G_1 will similarly define the logical X operators for the code.

5.5.2 Defining m^* -orthogonality

Bravyi and Haah [Bravyi and Haah, 2012] prove that a so-called triorthogonal code supports a transversal non-Clifford gate in the 3rd level of the Clifford hierarchy. We now generalise Bravyi and Haah's construction. First, we focus on generalising from qubits to qudits. We defined qudit colour codes using a star-bipartition, and it is useful to incorporate that into the definition. We introduce the star sign-flip matrix F . Each column of matrix G corresponds to a qudit, and let us order these columns such that the first p columns correspond to unstarred qudits, and the latter $n - p$ columns to starred qudits. We then define the $n \times n$ matrix F :

$$F = \text{diag}(\underbrace{1, \dots, 1}_{|v_\bullet| \text{ entries}}, \underbrace{-1, \dots, -1}_{|v_\star| \text{ entries}}), \quad (5.18)$$

where the first $|v_\bullet|$ elements on the diagonal are 1 and the remaining elements on the diagonal are -1 . We note that the ordering of the columns here is arbitrary, so we do not lose any generality by ordering the columns in this manner. The purpose of F will be to flip the sign of the entries of row vectors corresponding to the starred qudits, e.g. if $g = \{[g]_1, [g]_2, \dots, [g]_n\}$ is a row vector, we define $g \cdot F$ as the row vector where $[gF]_j = [g]_j$ for the first p elements (corresponding to unstarred qudits) and $[gF]_j = [-g]_j$ for the remaining elements (corresponding to starred vertices). We also define the weight $|\dots|$ of a row vector as $|g| = \sum_j [g]_j$.

We now will define a significant generalisation of triorthogonal matrices, which we call $m^\star$ -orthogonality (pronounced ‘ m star orthogonality’).

Definition 1. *An $n \times n$ matrix G over $\mathbb{Z}_d$ with a bipartition of columns into $\{v_\bullet, v_\star\}$ is $m^\star$ -orthogonal if both of the following conditions hold.*

1. *The weight of every elementwise product of any m rows of GF (including repeated rows) is equal to 0, except for the following case:*
2. *For each row of submatrix G_1F , the weight of the row vector raised to the m^{th} power elementwise is 1.*

Recall F is defined as in equation (5.18), and the matrices’ columns are ordered to respect to the star-bipartition $\{v_\bullet, v_\star\}$.

Notice the above makes no mention of modular arithmetic. If we had instead defined the weights modulo d , we would have a weaker notion of $m^\star$ -orthogonality. Colour codes turn out to satisfy this stronger notion, and so this definition suffices here. We note that the results below rely on the stronger form to deal with the exceptional cases, e.g. $d = 3, 6$ for the T gate (see Section 5.6.1). We also remark on the weaker notion to clarify that for $d = 2$ it exactly corresponds to the Brayvi-Haah definition of triorthogonality. Other applications of weak $m^\star$ -orthogonality include quantum Reed-Muller codes [Campbell, 2014; Campbell et al., 2012].

A more symbolic statement of $m^\star$ -orthogonality will prove useful in subsequent sections. We make use of $u \circ v$ symbol to denote element-wise products of vectors u and v , such that it has elements

$$[g_a \circ g_b]_j = [g_a]_j [g_b]_j, \quad (5.19)$$

which generalises for an arbitrary number of vectors, e.g. for three vectors

$$[g_a \circ g_b \circ g_c]_j = [g_a]_j [g_b]_j [g_c]_j. \quad (5.20)$$

Let us now consider a reformulation of points 1 and 2 of Def. 1. For any set of m rows, we have a list of m row indices $\{a_1, a_2, \dots, a_m\}$. m^* -orthogonality demands that all the bitwise products of m row vectors obey

$$\sum_j [(g_a \circ g_b \circ \dots \circ g_m)F]_j = 0, \quad (5.21)$$

unless all g_i in the bitwise product are identical and come from G_1 , in which case the summation in equation (5.21) equals unity.

5.5.3 The m^* -orthogonality of qudit colour codes

The concept of m^* -orthogonality is a powerful tool for studying the properties of qudit colour codes due to the following lemma

Lemma 5. *A qudit colour code for any $d \geq 2$ defined on a lattice in spatial dimension μ where X -stabilizer generators are defined by μ' -cells with $\mu' \leq \mu$, and with a star-bipartition $\{v_\bullet, v_\star\}$ defined by the bipartition of the lattice, is an m^* -orthogonal code for all $m \leq \mu'$.*

To prove this, let us first prove a convenient lemma.

Lemma 6. *Any m^* -orthogonal matrix G which includes, as one of its rows, the all-ones vector is also m'^* -orthogonal for all $m' < m$.*

Proof. This follows since an elementwise product of m vectors including the all-ones vector is equal to the elementwise product of $m - 1$ vectors excluding the all-ones vector. $\square$

Since the logical X for all colour codes is the transversal X acting on all qudits, the matrix G_1 for these codes contains the all-ones row. Hence to prove lemma 5 we now only need to prove that a colour code in μ spatial dimensions is m^* -orthogonal, where $m = \mu'$.

Proof. Recall that the X -generators of the code are defined by μ' -cells. Taking as an example the 3D lattice in figure 5.2, the X -stabilizer generators are defined by 3-cells. Consider the following geometric properties of these cells. When q distinct μ' -cells intersect non-trivially, where they meet defines a cell of smaller dimension. For a general lattice, there is no further restriction on the dimension of this cell. However, for a μ -colex it is well known [Bombin and Martin-Delgado, 2006, 2007a; Bombin, 2015; Kubica and Beverland, 2014] that the intersection of q objects of dimension μ' yields either an empty set or a cell of dimension $\mu' - q + 1$. For instance, the intersect of two neighbouring μ' -cells defines a $(\mu' - 1)$ -cell, where three such μ' -cells meet defines a $(\mu' - 2)$ -cell, and so on. Where μ' μ' -cells meet defines a 1-cell, or lattice edge. For example, in 3D, two adjacent 3-cells meet at a face, and three adjacent 3-cells meet at an edge (four adjacent 3-cells meet at a point).

We use this geometric fact to prove the theorem. Any product of m row vectors in G_0 has a geometric representation as the intersection of the vertices of these cells. This corresponds to either an empty cell or cell of dimension no less than $\mu - m + 1$. Each cell in the lattice, of any dimension greater than zero, has an equal number of starred and unstarred vertices (see corollary 1). When we multiply the product row vector by F we invert the sign of the columns corresponding to the starred vertices, but this corresponds to precisely half of the non-zero elements in the vector. Hence the weight of the resultant vector is zero provided $\mu - m + 1 > 0$. Therefore, the rows of G_0 satisfy the conditions for m^* -orthogonality whenever $m \leq \mu$.

The other case to consider is the case where all m vectors in the product are the all-ones vector. The element-wise product results trivially in the all-ones vector, corresponding to the logical operator of the colour code, or G_1 in the matrix defining the code. The number of unstarred qudits is one more than the starred qudits (see corollary 1), hence the weight of this vector after multiplying by F is 1. Together with lemma 6 this proves lemma 5. $\square$

Note that the bounds in this lemma are tight. A code whose X -stabilizer generators are defined by μ' -cells is not $(\mu' + 1)^*$ -orthogonal because $(\mu + 1)$ μ' -cells meet at a point, and a single vertex represents a vector which does not have zero weight.

5.6 Transversal Operators

Our attention now turns to diagonal transversal gates of the form of equation (5.9). The qubit colour codes are the only family of topological codes known to support transversal gates saturating the Bravyi-König bound. We prove in this section that (apart from a minority of special cases which need to be treated individually) the qudit codes we have defined also saturate this bound.

Theorem 1. *The qudit colour codes on a μ -colex, where X -stabilizer generators are defined by $\mu' \leq \mu$ cells and where $\mu'! \not\equiv 0 \pmod{d}$, support transversal gates in the μ' th level of the Clifford Hierarchy (CH) (which are not in the $(\mu - 1)$ th level of the CH), saturating the Bravyi-König bound.*

The qualification $\mu'! \not\equiv 0 \pmod{d}$ is due to lemma 2, namely that when $\mu'! \equiv 0 \pmod{d}$, gates of the form of equation (5.9) are in the $(\mu' - 1)$ th level of the CH (and therefore do not saturate the Bravyi-König bound) as discussed in Section 5.3.

For the most important $\mu = 3$ case, however, we additionally prove transversality for the exceptional $d = 3$ and $d = 6$ cases, and can thus state the unqualified result:

Theorem 2. *The qudit colour codes on a μ -colex, where X -stabilizer generators are defined by $\mu' \leq \mu$ cells support transversal non-Clifford gates in the 3rd level of the Clifford Hierarchy (CH) for all $\mu' \geq 3$.*

To prove theorem 1, we show that gates of the form of equation (5.9) are transversal in all m^* -orthogonal codes for polynomial degree $r = m$. To prove theorem 2 we also prove that gates of the form in equation (5.8) are transversal in 3^* -orthogonal codes. The theorems then follow by virtue of lemmas 1, 2 and 5, together with the known results for qubit colour codes [Bombin and Martin-Delgado, 2007b]. In particular, the transversal gates we consider are of star-conjugate transversal form, see equation 5.12.

Lemma 7 (m^* -orthogonality lemma). *An m^* -orthogonal code has a star-conjugate transversal implementation of the unitary gate R_{f_m} such that*

$$\tilde{R}_{f_m} = R_{f_m}[v_\bullet] \otimes R_{f_m}^*[v_\star] \quad (5.22)$$

implements R_{f_m} on the code space.

We prove this lemma in Section 5.7. Here we present a proof of the special and important case of $r = 3$. The general proof follows the same approach, but is a little notationally unwieldy, therefore readers interested in only the 3D case may wish to skip the general proof.

Lemma 8. *A 3^* -orthogonal code has a star-conjugate transversal implementation of the unitary gate $T = R_{j_3}$ such that:*

$$\tilde{T} = T[v_\bullet] \otimes T^*[v_\star] \quad (5.23)$$

implements T on the code space.

Proof. We now examine the conditions under which the star-conjugate transversal $\tilde{T}$ implements a logical $\bar{T}$. For this to be true, the phases for each computational basis component of each logical code word in equation (5.17), must agree with the phases defining the gate in equation (5.7). Noting that the same phase applies to all terms in the sum in equation (5.17), we can write

$$\tilde{T} |(x, y)^T \cdot G\rangle = \omega^{x^3} |(x, y)^T \cdot G\rangle, \quad (5.24)$$

where $x \in \mathbb{Z}_d$ and $y \in \mathbb{Z}_d^s$, where s is the number of stabilizer generators. In other words we fix the number of logical qubits $k = 1$.

Applying equations (5.7) and (5.23) to the state on the left hand side of equation (5.24), we find

$$\begin{aligned} \tilde{T} |(x, y)^T \cdot G\rangle &= (T[v_\bullet] \otimes T^*[v_\star]) |(x, y)^T \cdot G\rangle, \\ &= \prod_{u \in v_\bullet} T_u |(x, y)^T \cdot G\rangle \prod_{w \in v_\star} T_w^* |(x, y)^T \cdot G\rangle, \\ &= \prod_j \sum_k \omega^{((x, y)^T \cdot G \cdot F)_j^3} |k\rangle \langle k| (x, y)^T \cdot G\rangle, \end{aligned}$$

where the index j labels the qudits in the code and k_j is the j th element of the code word. When raising k_j to the third power this operation is performed bitwise, hence we recover the following relationship between the phases on both sides of

equation (5.24):

$$\sum_j [((x, y)^T \cdot G \cdot F) \circ ((x, y)^T \cdot G \cdot F) \circ ((x, y)^T \cdot G \cdot F)]_j = x^3 \pmod{d}. \quad (5.25)$$

It is useful to define $z = (x, y)$ so

$$\sum_j [(z^T \cdot G \cdot F) \circ (z^T \cdot G \cdot F) \circ (z^T \cdot G \cdot F)]_j = x^3 \pmod{d}. \quad (5.26)$$

We can also write this as

$$\sum_j [(z^T \cdot G \cdot F)^{\circ 3}]_j = x^3 \pmod{d}, \quad (5.27)$$

where we make use of the notation ‘ $(.)^{\circ 3}$ ’, which means a circle-product (see equation (5.19)) taken between three elements.

Now define $z = \sum_a z_a e_a$ where z_a is the value of the a th position of the vector z , and where e_a are the basis vectors over $\mathbb{Z}_2^{s+1}$. For example $e_1 = (1, 0, \dots, 0)$, $e_2 = (0, 1, \dots, 0)$ and so on. Re-writing the left hand side of equation (5.27) we obtain

$$\begin{aligned} \sum_j [(z^T \cdot G \cdot F)^{\circ 3}]_j = \\ \sum_j \left[\left(\sum_{a=1}^{s+1} z_a e_a^T \cdot G \cdot F \right) \circ \left(\sum_{b=1}^{s+1} z_b e_b^T \cdot G \cdot F \right) \circ \left(\sum_{c=1}^{s+1} z_c e_c^T \cdot G \cdot F \right) \right]_j \pmod{d}. \end{aligned}$$

Note that $e_a^T \cdot G$ is the a th row of the matrix G , which we denote g_a . Therefore

$$\begin{aligned} \sum_j [(z^T \cdot G \cdot F)^{\circ 3}]_j = \\ \sum_j \left[\left(\sum_{a=1}^{s+1} z_a g_a \cdot F \right) \circ \left(\sum_{b=1}^{s+1} z_b g_b \cdot F \right) \circ \left(\sum_{c=1}^{s+1} z_c g_c \cdot F \right) \right]_j \pmod{d}. \end{aligned}$$

We can re-write this as

$$\begin{aligned} \sum_j [(z^T \cdot G \cdot F)^{\circ 3}]_j &= \sum_j \sum_{a,b,c=1}^{s+1} [(z_a g_a \cdot F) \circ (z_b g_b \cdot F) \circ (z_c g_c \cdot F)]_j \pmod{d}, \\ &= \sum_j \sum_{a,b,c=1}^{s+1} [(g_a \circ g_b \circ g_c) \cdot F]_j z_a z_b z_c \pmod{d}. \end{aligned}$$

Summing over j gives

$$\sum_j [(z^T \cdot G \cdot F)^{\circ 3}]_j = \sum_{a,b,c=1}^{s+1} |(g_a \circ g_b \circ g_c) \cdot F| z_a z_b z_c \pmod{d}.$$

We break the sum into two parts, so that

$$\begin{aligned} \sum_j [(z^T \cdot G)^{\circ 3} \cdot F]_j \\ = \sum_a |(g_a \circ g_a \circ g_a) \cdot F| z_a^3 + \sum_{\{a,b,c | \neg(a=b=c)\}} |(g_a \circ g_b \circ g_c) \cdot F| z_a z_b z_c \pmod{d}. \end{aligned}$$

Now we use the definition of m^* -orthogonal matrices. This ensures that the first term is equal to $x^3 \pmod{d}$ and the second term is equal to zero. Hence equation (5.25) is satisfied, which completes the proof of lemma 8. $\square$

Using a simplified proof of the one presented here, we can show that a 2^* -orthogonal code supports a transversal S gate. It follows from lemma 6 that any 3^* -orthogonal code is also a 2^* -orthogonal code, so this suffices to show that the star-bipartite colour codes also support a transversal S gate. For larger m , the proof follows the same lines but is with more cumbersome algebra. The proof of lemma 7 which we present in Section 5.7 completes the proof of theorem 1. To prove theorem 2 we require one further lemma:

Lemma 9. *A 3^* -orthogonal qudit code where $d = 3$ or $d = 6$ has a transversal implementation of the unitary gate $T_{3,6}$ defined in equation (5.8) such that*

$$\tilde{T}_{3,6} = T_{3,6}[v_\bullet] \otimes T_{3,6}^*[v_\star]. \quad (5.28)$$

Logical operator	Defined in equation	Star-conjugate transversal implementation	Spatial dimension
H	Eqn. (5.4)	$H[v_\bullet] \otimes H^*[v_\star]$	2
$\Lambda(X)$	Eqn. (5.6)	Applied blockwise transversally	≥ 2
S	Eqn. (5.5)	$S[v_\bullet] \otimes S^*[v_\star]$	≥ 2
T	Eqn. (5.7)	$T[v_\bullet] \otimes T^*[v_\star]$	≥ 3
R_{f_r}	Eqn. (5.9)	$R_{f_r}[v_\bullet] \otimes R_{f_r}^*[v_\star]$	$\geq r$

Figure 5.3: A table illustrating the star-conjugate transversal implementation of some important logical gates and the spatial dimensions of the codes which support them for $d \neq 2, 3, 6$. The diagonal gates S , T and R_{f_r} can only be implemented for spatial dimensions for which $\mu!a_\mu \neq 0 \pmod{d}$.

The proof is presented below, completing the proof of theorem 2. Finally, we remark that, as CSS codes, all colour codes admit a transversal $\Lambda(X)$. In figure 5.3 we tabulate the star-conjugate transversal gates in qudit colour codes of different spatial dimensions.

5.6.1 Proof of Lemma 9: Transversal T Gate for $d = 3$ and $d = 6$

In this section we shall prove that a 3^* -orthogonal qudit code where $d = 3$ or $d = 6$ has a transversal implementation of the unitary gate $T_{3,6}$ defined as

$$T_{3,6} = \sum_{j \in \mathbb{Z}_d} \gamma^{j^3} |j\rangle \langle j|, \quad (5.10)$$

where $\gamma^3 = \omega$ and where the function in the exponent j^3 is evaluated in regular arithmetic (or equivalently modulo $3d$). We shall prove lemma 9 for $d = 3$ and $d = 6$ simultaneously, for separate proofs see [Watson et al., 2015b].

Proof. We begin by noting that $\gamma = e^{2\pi i/3d}$ and hence our calculation will make use of addition modulo $3d$. Proceeding as before we start with:

$$\tilde{T}_{3,6} |z^T \cdot G\rangle = \gamma^{x^3} |z^T \cdot G\rangle. \quad (5.30)$$

The phases on both sides of this equation match if

$$\sum_j ((z^T \cdot G \cdot F)^{\circ 3})_j = x^3 \pmod{3d}. \quad (5.31)$$

Once again, here we make use of the notation ‘ $(.)^{\circ 3}$ ’, which means a circle-product (see equation (5.19)) taken between three elements. This expression is a mix of modulo d and modulo $3d$ arithmetic so we must proceed with care. The matrix multiplication is mod d , whereas all other arithmetic is mod $3d$. Re-writing the left hand side as

$$\begin{aligned} & \sum_j ((z^T \cdot G \cdot F)^{\circ 3})_j \\ &= \sum_j \sum_{a,b,c=1}^{s+1} ([g_a \cdot F]_j z_a \pmod{d}) \circ ([g_b \cdot F]_j z_b \pmod{d}) \circ ([g_c \cdot F]_j z_c \pmod{d}), \end{aligned}$$

we can use the following identity to express the modulo d reduction in terms of standard addition

$$a \pmod{n} = a - \lfloor \frac{a}{n} \rfloor n. \quad (5.32)$$

Setting aside the modulo $3d$ arithmetic for the moment and evaluating this expression first in standard arithmetic, the left hand side now expands to the following form:

$$\begin{aligned} \sum_j ((z^T \cdot G \cdot F)^{\circ 3})_j &= \sum_j \sum_{a,b,c=1}^{s+1} \left([g_a \cdot F]_j z_a - d \lfloor \frac{[g_a \cdot F]_j z_a}{d} \rfloor \right) \\ &\quad \circ \left([g_b \cdot F]_j z_b - d \lfloor \frac{[g_b \cdot F]_j z_b}{d} \rfloor \right) \circ \left([g_c \cdot F]_j z_c - d \lfloor \frac{[g_c \cdot F]_j z_c}{d} \rfloor \right), \\ &= \sum_j \sum_{a,b,c=1}^{s+1} [(g_a \circ g_b \circ g_c) \cdot F]_j z_a z_b z_c \\ &\quad - 3d [(g_a \circ g_b) \cdot F]_j z_a z_b \lfloor \frac{[g_c \cdot F]_j z_c}{d} \rfloor \\ &\quad + 3d^2 [g_a \cdot F]_j z_a \lfloor \frac{[g_b \cdot F]_j z_b}{d} \rfloor \lfloor \frac{[g_c \cdot F]_j z_c}{d} \rfloor \\ &\quad - d^3 \lfloor \frac{[g_a \cdot F]_j z_a}{d} \rfloor \lfloor \frac{[g_b \cdot F]_j z_b}{d} \rfloor \lfloor \frac{[g_c \cdot F]_j z_c}{d} \rfloor. \end{aligned}$$

The last three terms in this summation are integer multiples of $3d$ and hence equal $0 \pmod{3d}$ for both $d = 3$ and $d = 6$. Thus if we reimpose modulo $3d$ arithmetic we are left with:

$$\sum_j ((z^T \cdot G \cdot F)^{\circ 3})_j \pmod{3d} = \sum_j \sum_{a,b,c=1}^{s+1} [(g_a \circ g_b \circ g_c) \cdot F]_j z_a z_b z_c \pmod{3d}. \quad (5.33)$$

Finally, via m^* -orthogonality we recover

$$\sum_j ((z^T \cdot G \cdot F)^{\circ 3})_j \pmod{3d} = x^3 \pmod{3d}, \quad (5.34)$$

completing the proof. $\square$

Note that we are using the fact that the definition of m^* -orthogonality is stated in regular arithmetic which implies the modulo $3d$ orthogonality used here.

5.7 Proof of the m^* -orthogonality Lemma

In this section we prove the following lemma, repeated from Section 5.6. We have already proven the special case that a 3^* -orthogonal code can support a star-conjugate transversal T gate, in other words a phase gate from the third level of the CH. We now generalise this to m^* -orthogonal codes and show that such codes have star-conjugate transversal R_{f_m} gates – phase gates from the m^{th} level of the CH. Recall that according to the theorem by Bravyi and König such a code must be realised in m or higher spatial dimensions.

Lemma 7 (m^* -orthogonality lemma). *An m^* -orthogonal code has a star-conjugate transversal implementation of the unitary gate R_{f_m} such that*

$$\tilde{R}_{f_m} = R_{f_m}[v_\bullet] \otimes R_{f_m}^*[v_\star] \quad (5.24)$$

implements R_{f_m} on the code space.

Proof. For equation (5.22) to be true, for all $z = (x, y)$ we have

$$\tilde{R}_{f_m} |z^T \cdot G\rangle = \omega^{z_j^m} |z^T \cdot G\rangle. \quad (5.38)$$

Applying equations (5.9) and (5.22) to the state on the right hand side of equation (5.38), the phases on both sides of the equation are equal if

$$\sum_j [(z^T \cdot G \cdot F)^{\circ m}]_j = x^m \pmod{d}. \quad (5.39)$$

Recall that $(\cdot)^{\circ m}$ means a bitwise product taken between m elements. Proceeding using the same notation as in Section 5.6 we find

$$\begin{aligned} \sum_j ((z^T \cdot G \cdot F)^{\circ m})_j &= \sum_j \left[\left(\sum_a z_a g_a \cdot F \right)^{\circ m} \right]_j \pmod{d}, \\ &= \sum_j \left[\sum_{\vec{a} \in [1, \dots, k+s]^m} z_{\vec{a}} g_{\vec{a}} \cdot F \right]_j \pmod{d}, \end{aligned} \quad (5.40)$$

where $[a, \dots, b]^c$ is a vector of length c with elements that take values between a and b , and where $g_{\vec{a}}$ is the bitwise product of m rows of G , so

$$g_{\vec{a}} = g_{a_1} \circ g_{a_2} \circ \dots \circ g_{a_m}. \quad (5.41)$$

We can re-write this as

$$\sum_j ((z^T \cdot G \cdot F)^{\circ m})_j = \sum_j \sum_{\vec{a} \in [1, \dots, k+s]^m} [g_{\vec{a}} \cdot F]_j z_{\vec{a}} \pmod{d}. \quad (5.42)$$

Summing over j gives

$$\sum_j ((z^T \cdot G \cdot F)^{\circ m})_j = \sum_{\vec{a} \in [1, \dots, k+s]^m} |g_{\vec{a}} \cdot F| z_{\vec{a}} \pmod{d}. \quad (5.43)$$

We now use the definition of an m^* -orthogonal code. This means that all terms satisfying $\neg(a_1 = \dots = a_m)$ in the summation on the right hand side vanish. This implies that the only non-zero contribution is represented by $\vec{a} = (a, a, \dots, a)$, an m -element vector where every element is a . Recall also that such terms vanish unless $a = 1$, in other words it is the m -fold elementwise product of the vector in G_1 . We

note that $z_{\vec{a}} = z_a^m$, leaving only

$$\begin{aligned} \sum_j ((z^T \cdot G \cdot F)^{\circ m})_j &= |g_{\vec{a}} \cdot F| z_a^m \pmod{d}, \\ &= x^m \pmod{d}, \end{aligned} \tag{5.44}$$

which proves that the desired equation (5.39) is satisfied. $\square$

We have now shown that the colour codes can be generalised to qudits via what we call the star-bipartite construction, for all spatial dimensions. This relies on the bipartition of the vertices of the lattice in which the code is embedded, and a framework in which any operator acting transversally on the lattice (or as a product operator on a subset of the lattice) inherits a ‘complex conjugate’ at locations corresponding to vertices within one of the partitions – we call this the star-conjugate implementation of the operator.

Furthermore, we have generalised the notion of triorthogonal codes, capable of supporting a transversal gate from the third level of the CH. We call the generalised codes m^* -orthogonal and show that within the star-bipartite framework they can realise a gate from the m th level of the CH¹. Finally, we have argued that a colour code in any spatial dimension μ is a m^* -orthogonal code, for $m \leq \mu$.

5.8 Conclusions

We have shown how any existing qubit colour code can be generalised to support a qudit colour code in arbitrary spatial dimensions. To do this we introduced the notion of a star-conjugate lattice and showed how this construction allows the Pauli, Clifford phase, controlled- X and non-Clifford phase gates to be implemented transversally in a code with arbitrarily high distance. The fault-tolerant universal gate set is completed by the addition of a star-conjugated transversal Hadamard gate, which can be implemented using the technique of *gauge fixing*.

Gauge fixing allows additional transversal gates to be implemented without

¹This is true up to the requirement that $\mu'! \neq 0 \pmod{d}$ in general, although we have shown that in 3D this does not prevent us from realising transversal non-Clifford gates for $d = 3, 6$ and we believe that it is possible to identify similar constructions in higher spatial dimensions.

compromising the error correcting properties of the code. It works by fault-tolerantly transferring to a subsystem code that can implement the requisite transversal gate, and then transforming back to the original code [Paetznick and Reichardt, 2013]. In the context of colour codes, gauge fixing and subsystem codes have been used to create a new family of codes, the gauge colour codes. Such codes possess a transversal Hadamard gate and have other favourable properties [Bombin, 2015; Brown et al., 2015]. It is beyond the scope of this thesis to describe gauge colour codes in detail, but this construction can be straightforwardly applied to the generalised colour codes introduced in the present chapter [Watson et al., 2015b].

There are still many open questions remaining – here we shall outline some of the ones we find most interesting. Although a fault-tolerant decoder has been implemented for a 2D code, no decoder has, to our knowledge, been proposed for stabilizer colour codes in higher spatial dimensions. In [Bombín, 2015] a decoder for a 3D gauge colour code is discussed, and such a decoder has recently been implemented [Brown et al., 2015]. It would be interesting to develop a decoder for the qudit colour codes presented in this chapter and to study their error correction properties. The proposal for a self-correcting quantum memory in a 6D colour code may also be generalisable to qudits [Bombin et al., 2013].

The construction of the transversal gates in the colour codes presented in this work holds for colour codes in any spatial dimension μ . In many physical realisations of quantum error correcting codes, it is not currently feasible to achieve even a two dimensional system [Kelly et al., 2015; Nigg et al., 2014]. Nonetheless, proposals exist that mean the effective spatial dimension of a system is governed by connection of optical fibres, where increasing the connectivity of nodes results in an increased number of effective spatial dimensions [Nickerson et al., 2013, 2014]. It would be beneficial to better understand the effect of the ability to realise gates from higher in the Clifford hierarchy on the efficiency of the quantum algorithm being realised.

The construction of the transversal gates in the colour codes presented in this work holds for qudits of any dimension, d . However, for non-prime d it is not clear that S , H and $\Lambda(X)$ are sufficient to generate the Clifford group. We desire a better understanding of universality and the Clifford hierarchy when using systems of non-prime dimension.

Finally, there are many Reed-Muller codes known for qudits for which no equiva-

lent colour code is known [Campbell, 2014; Campbell et al., 2012]. Is it possible to use one of these to construct a topological code similar to a colour code and study its properties? Here, we found codes where the distance is purely topological in nature, and so depends on the lattice and not the qudit dimension. In contrast, qudit Reed-Muller codes provide an improved distance that is algebraic rather than topological in origin. The potential exists for qudit quantum codes which abstract topological ideas to a more general setting to generate novel codes of arbitrary code distance with a rich family of transversal logical gates.

Chapter 6

Summary and Outlook

The work presented in this thesis has concentrated on some specific properties of topological quantum error correcting codes with applications to fault-tolerant quantum computing schemes. The codes we have focussed on are amongst the most experimentally viable, since their check operators are geometrically local to implement and their distances scale favourably with the number of qudits¹ comprising the code. Furthermore they are known to have some of the highest theoretical thresholds, meaning they are robust to a high rate of experimental noise.

The first contribution we have made in this thesis, presented in Chapter 3, is a framework for determining the *physical overhead* – that is, the size of the code (measured by the number of qudits) required in order to provide the desired error protection performance. The overhead value depends on the code, decoder and noise model as well as the physical error rate and the desired fidelity. We have presented an example of the calculation necessary to determine the overhead for the qubit toric code in the presence of two different, but related, noise models. The examples provide a framework in which to generalise these calculations to different topological codes and other noise models, as well as to other decoders, particularly those based on renormalisation group methods.

The theme of the investigation presented in Chapter 4 was to develop a computationally light decoder, which unlike the MWPMA decoder used in the earlier investigation, is applicable to fault-tolerant qudit surface codes. As well as a fast run time that is insensitive to the dimension of the physical system, the decoder

¹We use qudit here to mean quantum systems of any dimension, including $d = 2$.

possesses a high threshold. In fact, by increasing the qudit dimension and using an enhancement to the decoder that we call the ‘initialisation step’, we were able to show that thresholds of over 8% could be obtained for the model considered.

Finally, in Chapter 5 we turned our attention to a second family of topological codes, namely the colour codes. We developed a flexible method to generalise these codes to qudit constructions. We studied the transversality properties of these qudit codes and were able to show that in $\mu \geq 3$ spatial dimensions such codes can support a transversal non-Clifford gate, saturating Bravyi and König’s bound for all qudit dimensions and codes in all spatial dimensions. We have also noted that a universal set of fault-tolerant gates can be completed by switching between the stabilizer colour codes and the gauge colour codes in 3 and higher spatial dimensions. Indeed, the gauge colour code, rather than the straightforward colour code, may offer other practical advantages.

One way in which the work presented in Chapters 3 and 4 of this thesis could be extended is to consider more physically realistic noise models. Although the methods presented seem to be equally valid for the fully fault-tolerant model, a full circuit-based noise model of the error detection protocols could be studied. We expect that the framework presented in Chapter 3 would be easily generalised to this noise model, with only the numerical results varying significantly. Changing the focus of the investigation from the toric code to the surface code, which may be easier to experimentally implement, should also be straightforward. The decoder we presented in Chapter 4 should be very flexible and although we expect the threshold to be suppressed due to the relative increase in errors, we believe that no changes should be necessary to the decoder itself.

The ultimate goal for physicists working on the challenge of quantum information processing is the experimental realisation of a fully scalable universal quantum computer. This area of research has drawn interest from industry, with investment from Google in superconducting qubit architectures and also from the UK government who have invested in the NQIT network of quantum technologies hubs. As a result, several experimental schemes for quantum error correction that have been proposed and tested [Barends et al., 2014; Kelly et al., 2015; O’Gorman et al., 2014] will be pushing forward their aim of realising a fault-tolerant quantum computing scheme in the near future. It would be of great benefit to such implementations to characterise

the resource costs and limitations of their experimental designs using an analysis such as the one we shall outline below. Before we detail this analysis, let us first consider the likely evolution of the cost of resources for quantum computing.

Initially we might imagine that the noise is not well understood or well controlled, and that fault-tolerant quantum computers (or some reduced ‘proof of principle’ computer, perhaps incapable of implementing a full universal gate set but demonstrating a transversal set of Pauli or Clifford operations) would be operating below, but close to, the relevant error threshold. At this stage the qudits are likely to be relatively rather expensive to manufacture. As the technology is developed and refined the errors will become better controlled and the computer will begin to operate at an error rate well below the threshold for all potential decoders. It is also likely that the cost-per-qudit will decrease.

With this evolution in mind, it would be interesting to perform a full fault-tolerant resource analysis to determine the tradeoffs amongst them. These resources include, but are not limited to, error thresholds, decoder run time and overheads. The overhead will not be limited to only the qudit overhead discussed in this thesis, but also, for example, the number of non-Clifford gates that must be performed. If, for example, the T gate is performed via state injection, this overhead may be measured in terms of the total number of noisy ancillas required to produce the purified magic states.

The analysis of the overheads is likely to be framed in terms of a specific algorithm, which will give rise to an estimate of the necessary fidelity of each logical element of the circuit. The results will be parametrised by this fidelity as well as the physical noise rates¹. Naturally the resource analysis would also depend on the choice of code, decoder and method for achieving universality (gauge fixing, magic state distillation and so on), as well as the qudit dimension.

Now we see how such analysis may be useful as the technology changes, or indeed, for selecting a system that will be robust to such changes. Optimising the resources early on, in the ‘proof-of-principle’ stage is likely to mean selecting a decoder with a high threshold and preferably a relatively low qudit overhead. Here we see that the computer can potentially tolerate a slow decoder run time, notably if there are no

¹Since we are discussing a realistic implementation we now allow for the possibility that the noise model is parametrised by more than one variable.

non-Clifford gates and the Pauli frame update can be delayed until the state is read out.

Later the computer will be operating well below the threshold so its exact value will not be such an important quantity. At this point the decoder can be selected to improve the run time at some cost to the value of the threshold. As the technology becomes more scalable, the method chosen for achieving universality will become important. Particularly in the case of magic state distillation this may affect the choice of dimension of the qudits used, although this will be largely determined by the physical system itself, which may not be generalisable to qudits with $d > 2$. We see that, of course, the knowledge of the end-goal of the fully scalable quantum computer should inform the choice of which technology to invest in earlier, during the ‘proof-of-principle’ experiments.

Naturally these investigations should not be confined to surface codes but should also encompass colour codes and gauge colour codes as well. The possibilities here include 2D colour codes with magic state injection to achieve universality, colour codes with dimensional jumps [Bombin, 2014], 3D colour codes and gauge colour codes, as well as higher dimensional (gauge) colour codes. So far we have not discussed the implications of a higher dimensional colour code but in some potential experimental setups, for example see [Nickerson et al., 2013, 2014], this is a matter of connecting qubits via an optical fibre network and can be done in such a way as to simulate any spatial dimension.

A comparison of the resources required here is currently more challenging because few decoders are known for colour codes in $\mu > 2$ spatial dimensions. It has been shown that it is straightforward to adapt the HDRG decoder to correct the 3D single-shot fault-tolerant gauge colour code in the qubit case at least [Bombín, 2015; Brown et al., 2015], but as with the surface code a range of decoders with different properties are likely to exist. Such a resource comparison within and between these two prominent code families is likely to be revealing.

Naturally this thesis poses many questions, but we believe it provides a framework in which further answers can be provided, and motivates a unified study of the properties of error correcting codes for fault-tolerant quantum computing. There are many parameters to control and the comparisons between the two code families are not necessarily straightforward. The tradeoffs between all of the parameters

are also subtle but a detailed understanding of the resource landscape would be instructive and may enhance experimental work. Ultimately the implementation of universal quantum computing that is found will set some of the input parameters that determine which of the schemes is selected. However, some initial theoretical input could help to guide the experimental research in a way that would be beneficial from the outset.

References

- D. Aharonov and M. Ben-Or. Fault-tolerant quantum computation with constant error. In *Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing*, STOC '97, pages 176–188, New York, NY, USA, 1997. ACM. ISBN 0-89791-888-6. doi: 10.1145/258533.258579. URL <http://doi.acm.org/10.1145/258533.258579>. 17
- A. Amraoui, A. Montanari, T. Richardson, and R. Urbanke. Finite-length scaling for iteratively decoded LDPC ensembles. In *IEEE Transactions on Information Theory (Volume:55, Issue:2)*, pages 473 – 498, Feb 2009. doi: 10.1109/TIT.2008.2009580. 55
- B. E. Anderson, H. Sosa-Martinez, C. A. Riofrío, Ivan H. Deutsch, and Poul S. Jessen. Accurate and robust unitary transformations of a high-dimensional quantum system. *Phys. Rev. Lett.*, 114:240401, Jun 2015. doi: 10.1103/PhysRevLett.114.240401. URL <http://link.aps.org/doi/10.1103/PhysRevLett.114.240401>. 86
- Jonas T. Anderson, Guillaume Duclos-Cianci, and David Poulin. Fault-tolerant conversion between the Steane and Reed-Muller quantum codes. *Phys. Rev. Lett.*, 113:080501, Aug 2014. doi: 10.1103/PhysRevLett.113.080501. URL <http://link.aps.org/doi/10.1103/PhysRevLett.113.080501>. 30, 111
- Ruben S. Andrist, James R. Wootton, and Helmut G. Katzgraber. Error thresholds for abelian quantum double models: Increasing the bit-flip stability of topological quantum memory. *Phys. Rev. A*, 91:042331, Apr 2015. doi: 10.1103/PhysRevA.91.042331. URL <http://link.aps.org/doi/10.1103/PhysRevA.91.042331>. 50, 86

-
- Hussain Anwar, Benjamin J. Brown, Earl T. Campbell, and Dan E. Browne. Fast decoders for qudit topological codes. *New Journal of Physics*, 16(6):063038, 2014. URL <http://stacks.iop.org/1367-2630/16/i=6/a=063038>. 30, 86, 87, 88, 99
- D. M. Appleby. Symmetric informationally complete positive operator valued measures and the extended Clifford group. *Journal of Mathematical Physics*, 46(5):052107, 2005. doi: <http://dx.doi.org/10.1063/1.1896384>. URL <http://scitation.aip.org/content/aip/journal/jmp/46/5/10.1063/1.1896384>. 111
- R. Barends, J. Kelly, A. Megrant, A. Veitia, D. Sank, E. Jeffrey, T. C. White, J. Mutus, A. G. Fowler, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, C. Neill, P. O'Malley, P. Roushan, A. Vainsencher, J. Wenner, A. N. Korotkov, A. N. Cleland, and John M. Martinis. Superconducting quantum circuits at the surface code threshold for fault tolerance. *Nature*, 508(7497):500–503, 04 2014. 18, 140
- H. Bombin and M. A. Martin-Delgado. Topological quantum distillation. *Phys. Rev. Lett.*, 97:180501, Oct 2006. doi: 10.1103/PhysRevLett.97.180501. URL <http://link.aps.org/doi/10.1103/PhysRevLett.97.180501>. 110, 112, 115, 120, 127
- H. Bombin and M. A. Martin-Delgado. Exact topological quantum order in $d = 3$ and beyond: Branyons and brane-net condensates. *Phys. Rev. B*, 75:075103, Feb 2007a. doi: 10.1103/PhysRevB.75.075103. URL <http://link.aps.org/doi/10.1103/PhysRevB.75.075103>. 110, 112, 114, 115, 120, 121, 127
- H Bombin, R W Chhajlany, M Horodecki, and M A Martin-Delgado. Self-correcting quantum computers. *New Journal of Physics*, 15(5):055023, 2013. URL <http://stacks.iop.org/1367-2630/15/i=5/a=055023>. 110, 111, 137
- Hector Bombin. Dimensional jump in quantum error correction. *arXiv:quant-ph/1412.5079*, Dec 2014. 142
- Hector Bombin. Gauge color codes: optimal transversal gates and gauge fixing in topological stabilizer codes. *New Journal of Physics*, 17(8):083002, 2015. URL <http://stacks.iop.org/1367-2630/17/i=8/a=083002>. 111, 115, 120, 127, 137

-
- Héctor Bombín. Single-shot fault-tolerant quantum error correction. *Phys. Rev. X*, 5:031043, Sep 2015. doi: 10.1103/PhysRevX.5.031043. URL <http://link.aps.org/doi/10.1103/PhysRevX.5.031043>. 137, 142
- Hector Bombin and M. A. Martin-Delgado. Topological computation without braiding. *Phys. Rev. Lett.*, 98:160502, Apr 2007b. doi: 10.1103/PhysRevLett.98.160502. URL <http://link.aps.org/doi/10.1103/PhysRevLett.98.160502>. 110, 112, 128
- Sergey Bravyi and Jeongwan Haah. Magic-state distillation with low overhead. *Phys. Rev. A*, 86:052329, Nov 2012. doi: 10.1103/PhysRevA.86.052329. URL <http://link.aps.org/doi/10.1103/PhysRevA.86.052329>. 30, 112, 123, 124
- Sergey Bravyi and Jeongwan Haah. Quantum self-correction in the 3D cubic code model. *Phys. Rev. Lett.*, 111:200501, Nov 2013. doi: 10.1103/PhysRevLett.111.200501. URL <http://link.aps.org/doi/10.1103/PhysRevLett.111.200501>. 57, 83, 87, 107
- Sergey Bravyi and Alexei Kitaev. Universal quantum computation with ideal Clifford gates and noisy ancillas. *Phys. Rev. A*, 71:022316, Feb 2005. doi: 10.1103/PhysRevA.71.022316. URL <http://link.aps.org/doi/10.1103/PhysRevA.71.022316>. 25, 28, 29, 86
- Sergey Bravyi and Robert König. Classification of topologically protected gates for local stabilizer codes. *Phys. Rev. Lett.*, 110:170503, Apr 2013. doi: 10.1103/PhysRevLett.110.170503. URL <http://link.aps.org/doi/10.1103/PhysRevLett.110.170503>. 111, 119
- Sergey Bravyi and Robert Raussendorf. Measurement-based quantum computation with the toric code states. *Phys. Rev. A*, 76:022304, Aug 2007. doi: 10.1103/PhysRevA.76.022304. URL <http://link.aps.org/doi/10.1103/PhysRevA.76.022304>. 40
- Sergey Bravyi and Alexander Vargo. Simulation of rare events in quantum error correction. *Phys. Rev. A*, 88:062308, Dec 2013. doi: 10.1103/PhysRevA.88.062308. URL <http://link.aps.org/doi/10.1103/PhysRevA.88.062308>. 52, 83, 84

-
- Sergey Bravyi, Martin Suchara, and Alexander Vargo. Efficient algorithms for maximum likelihood decoding in the surface code. *Phys. Rev. A*, 90:032326, Sep 2014. doi: 10.1103/PhysRevA.90.032326. URL <http://link.aps.org/doi/10.1103/PhysRevA.90.032326>. 47, 87
- Sergey B. Bravyi and Alexei Yu. Kitaev. Quantum codes on a lattice with boundary. *arXiv:quant-ph/9811052*, Nov 1998. 33, 40
- Courtney G. Brell. Generalized color codes supporting non-abelian anyons. *Phys. Rev. A*, 91:042333, Apr 2015. doi: 10.1103/PhysRevA.91.042333. URL <http://link.aps.org/doi/10.1103/PhysRevA.91.042333>. 110
- Benjamin J. Brown, Naomi H. Nickerson, and Dan E. Browne. Fault tolerance with the gauge color code. *arXiv:1503.08217*, Mar 2015. 137, 142
- Stephen S. Bullock and Gavin K. Brennen. Qudit surface codes and gauge theory with finite cyclic groups. *Journal of Physics A: Mathematical and Theoretical* 40 (13):3481, 2007. URL <http://stacks.iop.org/1751-8121/40/i=13/a=013>. 41
- A. R. Calderbank and Peter W. Shor. Good quantum error-correcting codes exist. *Phys. Rev. A*, 54:1098–1105, Aug 1996. doi: 10.1103/PhysRevA.54.1098. URL <http://link.aps.org/doi/10.1103/PhysRevA.54.1098>. 16
- A.R. Calderbank, E. M Rains, P. W. Shor, and N. J. A. Sloane. Quantum error correction via codes over GF(4). *arXiv:quant-ph/9608006*, Aug 1996. 16
- Earl T. Campbell. Enhanced fault-tolerant quantum computing in d -level systems. *Phys. Rev. Lett.*, 113:230501, Dec 2014. doi: 10.1103/PhysRevLett.113.230501. URL <http://link.aps.org/doi/10.1103/PhysRevLett.113.230501>. 30, 86, 125, 138
- Earl T. Campbell, Hussain Anwar, and Dan E. Browne. Magic-state distillation in all prime dimensions using quantum Reed-Muller codes. *Phys. Rev. X*, 2:041021, Dec 2012. doi: 10.1103/PhysRevX.2.041021. URL <http://link.aps.org/doi/10.1103/PhysRevX.2.041021>. 24, 30, 86, 116, 125, 138

-
- Xie Chen, Hyeyoun Chung, Andrew W. Cross, Bei Zeng, and Isaac L. Chuang. Subsystem stabilizer codes cannot have a universal set of transversal gates for even one encoded qudit. *Phys. Rev. A*, 78:012353, Jul 2008. doi: 10.1103/PhysRevA.78.012353. URL <http://link.aps.org/doi/10.1103/PhysRevA.78.012353>. 28
- I. L. Chuang, R. Laflamme, P. W. Shor, and W. H. Zurek. Quantum computers, factoring, and decoherence. *Science*, 270(5242):1633–1635, 1995. doi: 10.1126/science.270.5242.1633. URL <http://www.sciencemag.org/content/270/5242/1633.abstract>. 15
- Sean Clark. Valence bond solid formalism for d -level one-way quantum computation. *Journal of Physics A: Mathematical and General*, 39(11):2701, 2006. URL <http://stacks.iop.org/0305-4470/39/i=11/a=010>. 115
- William Cook and André Rohe. Computing minimum-weight perfect matchings. *INFORMS Journal on Computing*, 11(2):138–148, 1999. doi: 10.1287/ijoc.11.2.138. URL <http://pubsonline.informs.org/doi/abs/10.1287/ijoc.11.2.138>. 57
- Robert M Corless, Gaston H. Gonnet, David E.G. Hare, David J. Jeffrey, and Donald E. Knuth. On the LambertW function. *Advances in Computational mathematics*, 5(1):329–359, 1996. URL <http://link.springer.com/article/10.1007/BF02124750>. 79
- S. L. A. de Queiroz. Location and properties of the multicritical point in the Gaussian and $\pm J$ Ising spin glasses. *Phys. Rev. B*, 79:174408, May 2009. doi: 10.1103/PhysRevB.79.174408. URL <http://link.aps.org/doi/10.1103/PhysRevB.79.174408>. 47
- Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43(9):4452–4505, 2002. doi: <http://dx.doi.org/10.1063/1.1499754>. URL <http://scitation.aip.org/content/aip/journal/jmp/43/9/10.1063/1.1499754>. 33, 35, 39, 46, 47, 53, 59, 61, 86
- Simon J. Devitt, Austin G. Fowler, Ashley M. Stephens, Andrew D. Greentree, Lloyd C.L. Hollenberg, William J. Munro, and Kae Nemoto. Architectural design for a topological cluster state quantum computer. *New Journal of Physics*, 11:083032, 2009. 52

-
- D. Dieks. Communication by EPR devices. *Physics Letters A*, 92(6):271 – 272, 1982. ISSN 0375-9601. doi: [http://dx.doi.org/10.1016/0375-9601\(82\)90084-6](http://dx.doi.org/10.1016/0375-9601(82)90084-6). URL <http://www.sciencedirect.com/science/article/pii/0375960182900846>. 15
- David P. DiVincenzo and Panos Aliferis. Effective fault-tolerant quantum computation with slow measurements. *Phys. Rev. Lett.*, 98:020501, Jan 2007. doi: 10.1103/PhysRevLett.98.020501. URL <http://link.aps.org/doi/10.1103/PhysRevLett.98.020501>. 24
- G. Duclos-Cianci and D. Poulin. A renormalization group decoding algorithm for topological quantum codes. In *Information Theory Workshop (ITW), 2010 IEEE*, pages 1–5, Aug 2010a. doi: 10.1109/CIG.2010.5592866. 33, 57
- Guillaume Duclos-Cianci and David Poulin. Fast decoders for topological quantum codes. *Phys. Rev. Lett.*, 104:050504, Feb 2010b. doi: 10.1103/PhysRevLett.104.050504. URL <http://link.aps.org/doi/10.1103/PhysRevLett.104.050504>. 33, 47, 57, 83
- Guillaume Duclos-Cianci and David Poulin. Kitaev’s z_d -code threshold estimates. *Phys. Rev. A*, 87:062338, Jun 2013. doi: 10.1103/PhysRevA.87.062338. URL <http://link.aps.org/doi/10.1103/PhysRevA.87.062338>. 86, 87
- Guillaume Duclos-Cianci and David Poulin. Fault-tolerant renormalization group decoder for abelian topological codes. *Quantum Information and Computation*, 14:0721–0740, 2014. 87, 99, 108
- Bryan Eastin and Emanuel Knill. Restrictions on transversal encoded quantum gate sets. *Phys. Rev. Lett.*, 102:110502, Mar 2009. doi: 10.1103/PhysRevLett.102.110502. URL <http://link.aps.org/doi/10.1103/PhysRevLett.102.110502>. 28, 114
- Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17(3): 449–467, 1965. 57
- Austin G. Fowler. Proof of finite surface code threshold for matching. *Phys. Rev. Lett.*, 109:180502, Nov 2012. doi: 10.1103/PhysRevLett.109.180502. URL <http://link.aps.org/doi/10.1103/PhysRevLett.109.180502>. 61

-
- Austin G. Fowler, Matteo Mariantoni, John M. Martinis, and Andrew N. Cleland. Surface codes: Towards practical large-scale quantum computation. *Phys. Rev. A*, 86:032324, Sep 2012a. doi: 10.1103/PhysRevA.86.032324. URL <http://link.aps.org/doi/10.1103/PhysRevA.86.032324>. 33, 52, 53
- Austin G. Fowler, Adam C. Whiteside, and Lloyd C. L. Hollenberg. Towards practical classical processing for the surface code. *Phys. Rev. Lett.*, 108:180501, May 2012b. doi: 10.1103/PhysRevLett.108.180501. URL <http://link.aps.org/doi/10.1103/PhysRevLett.108.180501>. 87
- Michael H. Freedman and David A. Meyer. Projective plane and planar quantum codes. *arXiv:quant-ph/9810055*, Oct 1998. 40
- Daniel Gottesman. *Stabilizer Codes and Quantum Error Correction*. PhD thesis, California Institute of Technology, 1997. URL <http://thesis.library.caltech.edu/2900/2/THESIS.pdf>. 16, 21
- Daniel Gottesman. The Heisenberg representation of quantum computers. *arXiv:quant-ph/9807006*, 1998. 25
- Daniel Gottesman. Fault-tolerant quantum computation with higher-dimensional systems. *Chaos, Solitons & Fractals*, 10(10):1749 – 1758, 1999. ISSN 0960-0779. doi: [http://dx.doi.org/10.1016/S0960-0779\(98\)00218-5](http://dx.doi.org/10.1016/S0960-0779(98)00218-5). URL <http://www.sciencedirect.com/science/article/pii/S0960077998002185>. 115
- Daniel Gottesman. An introduction to quantum error correction and fault-tolerant quantum computation. In Samuel J. Lomonaco, editor, *Quantum Information Science and Its Contributions to Mathematics: American Mathematical Society Short Course, January 3-4, 2009, Washington, DC*. American Mathematical Society, 2010. 27
- Daniel Gottesman. What is the overhead required for fault-tolerant quantum computation? *arXiv:quant-ph/1310.2984*, Oct 2013. 52
- Daniel Gottesman and Isaac L. Chuang. Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature*, 402(6760):390–393, 11 1999. URL <http://dx.doi.org/10.1038/46503>. 24

-
- Lov K. Grover. Quantum mechanics helps in searching for a needle in a haystack. *Phys. Rev. Lett.*, 79:325–328, Jul 1997. doi: 10.1103/PhysRevLett.79.325. URL <http://link.aps.org/doi/10.1103/PhysRevLett.79.325>. 14
- James William Harrington. *Analysis of quantum error-correcting codes: symplectic lattice codes and toric codes*. PhD thesis, California Institute of Technology, 2004. URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.83.6167&rep=rep1&type=pdf>. 47, 61, 65, 87
- T.P. Harty, D.T.C. Allcock, C.J. Ballance, L. Guidoni, H.A. Janacek, N.M. Linke, D.N. Stacey, and D.M. Lucas. High-fidelity preparation, gates, memory, and readout of a trapped-ion quantum bit. *Phys. Rev. Lett.*, 113:220501, Nov 2014. doi: 10.1103/PhysRevLett.113.220501. URL <http://link.aps.org/doi/10.1103/PhysRevLett.113.220501>. 18, 80
- Allen Hatcher. *Algebraic Topology*. Cambridge University Press, Cambridge, 2002. URL <http://www.math.cornell.edu/~hatcher/AT/ATchapters.html>. 36
- Michael Henle. *A Combinatorial Introduction to Topology*. Dover (New York), 1994. 36
- M. Herold, E. T. Campbell, J. Eisert, and M. J. Kastoryano. Cellular-automaton decoders for topological quantum memories. *arXiv:1406.2338*, 2014. 87
- A. Honecker, M. Picco, and P. Pujol. Universality class of the Nishimori point in the 2D $\pm J$ random-bond Ising model. *Phys. Rev. Lett.*, 87:047201, Jul 2001. doi: 10.1103/PhysRevLett.87.047201. URL <http://link.aps.org/doi/10.1103/PhysRevLett.87.047201>. 47
- Mark Howard and Jiri Vala. Qudit versions of the qubit $\pi/8$ gate. *Phys. Rev. A*, 86: 022316, Aug 2012. doi: 10.1103/PhysRevA.86.022316. URL <http://link.aps.org/doi/10.1103/PhysRevA.86.022316>. 24, 111, 116
- Adrian Hutter, Daniel Loss, and James R. Wootton. Improved HDRG decoders for qudit and non-abelian quantum error correction. *arXiv:1410.4478*, 2014a. 87

- Adrian Hutter, James R. Wootton, and Daniel Loss. Efficient Markov chain Monte Carlo algorithm for the surface code. *Phys. Rev. A*, 89:022326, Feb 2014b. doi: 10.1103/PhysRevA.89.022326. URL <http://link.aps.org/doi/10.1103/PhysRevA.89.022326>. 87
- Tomas Jochym-O'Connor and Raymond Laflamme. Using concatenated quantum codes for universal fault-tolerant quantum gates. *Phys. Rev. Lett.*, 112:010505, Jan 2014. doi: 10.1103/PhysRevLett.112.010505. URL <http://link.aps.org/doi/10.1103/PhysRevLett.112.010505>. 30, 111
- Cody Jones. Multilevel distillation of magic states for quantum computing. *Phys. Rev. A*, 87:042305, Apr 2013. doi: 10.1103/PhysRevA.87.042305. URL <http://link.aps.org/doi/10.1103/PhysRevA.87.042305>. 30
- J. Kelly, R. Barends, A. G. Fowler, A. Megrant, E. Jeffrey, T. C. White, D. Sank, J. Y. Mutus, B. Campbell, Yu Chen, Z. Chen, B. Chiaro, A. Dunsworth, I. C. Hoi, C. Neill, P. J. J. O'Malley, C. Quintana, P. Roushan, A. Vainsencher, J. Wenner, A. N. Cleland, and John M. Martinis. State preservation by repetitive error detection in a superconducting quantum circuit. *Nature*, 519(7541):66–69, 03 2015. URL <http://dx.doi.org/10.1038/nature14270>. 33, 137, 140
- A.Y. Kitaev. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys*, 52:11911249, 1997a. 26
- A.Yu. Kitaev. Quantum error correction with imperfect gates. In O. Hirota, A.S. Holevo, and C.M. Caves, editors, *Quantum Communication, Computing, and Measurement*, pages 181–188. Springer US, 1997b. ISBN 978-1-4613-7716-0. doi: 10.1007/978-1-4615-5923-8_19. URL http://dx.doi.org/10.1007/978-1-4615-5923-8_19. 33
- A.Yu. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, 2003. doi: [http://dx.doi.org/10.1016/S0003-4916\(02\)00018-0](http://dx.doi.org/10.1016/S0003-4916(02)00018-0). URL <http://www.sciencedirect.com/science/article/pii/S0003491602000180>. 33, 41
- E. Knill. Quantum computing with realistically noisy devices. *Nature*, 434(7029):39–44, 03 2005. URL <http://dx.doi.org/10.1038/nature03350>. 24

- E. Knill, R. Laflamme, and W. Zurek. Threshold accuracy for quantum computation. *arXiv:quant-ph/9610011*, 1996. 16, 111
- Donald E. Knuth. *Art of Computer Programming*, volume 2: Seminumerical Algorithms. Addison-Wesley Professional, 3rd edition, 1997. ISBN 978-0-201-89684-8. URL <http://www.informit.com/articles/article.aspx?p=2221791>. 110
- Vladimir Kolmogorov. Blossom V: a new implementation of a minimum cost perfect matching algorithm. *Mathematical Programming Computation*, 1(1):43–67, 2009. URL <http://link.springer.com/article/10.1007/s12532-009-0002-8>. 67
- Aleksander Kubica and Michael E. Beverland. Universal transversal gates with color codes - a simplified approach. *arXiv:quant-ph/1410.0069*, Oct 2014. 111, 120, 127
- Raymond Laflamme, Cesar Miquel, Juan Pablo Paz, and Wojciech Hubert Zurek. Perfect quantum error correcting code. *Phys. Rev. Lett.*, 77:198–201, Jul 1996. doi: 10.1103/PhysRevLett.77.198. URL <http://link.aps.org/doi/10.1103/PhysRevLett.77.198>. 15
- Andrew J. Landahl, Jonas T. Anderson, and Patrick R. Rice. Fault-tolerant quantum computing with color codes. *arXiv:quant-ph/1108.5738*, 2011. 112
- Rolf Landauer. Is quantum mechanics useful? *Philosophical Transactions of the Royal Society A*, 353:367–376, Dec 1995. doi: 10.1098/rsta.1995.0106. 15
- Adam M. Meier, Bryan Eastin, and Emanuel Knill. Magic-state distillation with the four-qubit code. *arXiv:1204.4221 [quant-ph]*, Apr 2012. 30
- F. Merz and J. T. Chalker. Two-dimensional random-bond Ising model, free fermions, and the network model. *Phys. Rev. B*, 65:054425, Jan 2002. doi: 10.1103/PhysRevB.65.054425. URL <http://link.aps.org/doi/10.1103/PhysRevB.65.054425>. 65
- M. Mondelli, S.H. Hassani, and R. Urbanke. Scaling exponent of list decoders with applications to polar codes. In *Information Theory Workshop (ITW), 2013 IEEE*, pages 1–5, Sept 2013. doi: 10.1109/ITW.2013.6691215. 55
- Naomi H. Nickerson, Ying Li, and Simon C. Benjamin. Topological quantum computing with a very noisy network and local error rates approaching one percent.

-
- Nat Commun*, 4:1756, 04 2013. URL <http://dx.doi.org/10.1038/ncomms2773>. 137, 142
- Naomi H. Nickerson, Joseph F. Fitzsimons, and Simon C. Benjamin. Freely scalable quantum technologies using cells of 5-to-50 qubits with very lossy and noisy photonic links. *Phys. Rev. X*, 4:041041, Dec 2014. doi: 10.1103/PhysRevX.4.041041. URL <http://link.aps.org/doi/10.1103/PhysRevX.4.041041>. 137, 142
- Michael A Nielsen and Isaac L Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2010. 25, 26
- D. Nigg, M. Müller, E. A. Martinez, P. Schindler, M. Hennrich, T. Monz, M. A. Martin-Delgado, and R. Blatt. Quantum computations on a topologically encoded qubit. *Science*, 345(6194):302–305, 2014. doi: 10.1126/science.1253742. URL <http://www.sciencemag.org/content/345/6194/302.abstract>. 14, 137
- Hidetoshi Nishimori. Internal energy, specific heat and correlation function of the bond-random Ising model. *Progress of Theoretical Physics*, 66(4):1169–1181, 1981. 49
- Joe O’Gorman, Naomi H. Nickerson, Philipp Ross, John J. L. Morton, and Simon C. Benjamin. A silicon-based surface code quantum computer. *arXiv:1406.5149*, Jun 2014. 140
- Takuya Ohno, Gaku Arakawa, Ikuo Ichinose, and Tetsuo Matsui. Phase structure of the random-plaquette gauge model: accuracy threshold for a toric quantum memory. *Nuclear Physics B*, 697(3):462 – 480, 2004. ISSN 0550-3213. doi: <http://dx.doi.org/10.1016/j.nuclphysb.2004.07.003>. URL <http://www.sciencedirect.com/science/article/pii/S055032130400481X>. 87
- Adam Paetznick and Ben W. Reichardt. Universal fault-tolerant quantum computation with only transversal gates and error correction. *Phys. Rev. Lett.*, 111:090505, Aug 2013. doi: 10.1103/PhysRevLett.111.090505. URL <http://link.aps.org/doi/10.1103/PhysRevLett.111.090505>. 30, 111, 137

-
- Robert Raussendorf and Hans J. Briegel. A one-way quantum computer. *Phys. Rev. Lett.*, 86:5188–5191, May 2001. doi: 10.1103/PhysRevLett.86.5188. URL <http://link.aps.org/doi/10.1103/PhysRevLett.86.5188>. 24
- Robert Raussendorf and Jim Harrington. Fault-tolerant quantum computation with high threshold in two dimensions. *Phys. Rev. Lett.*, 98:190504, May 2007. doi: 10.1103/PhysRevLett.98.190504. URL <http://link.aps.org/doi/10.1103/PhysRevLett.98.190504>. 33
- Robert Raussendorf, Daniel E. Browne, and Hans J. Briegel. Measurement-based quantum computation on cluster states. *Phys. Rev. A*, 68:022312, Aug 2003. doi: 10.1103/PhysRevA.68.022312. URL <http://link.aps.org/doi/10.1103/PhysRevA.68.022312>. 24
- Robert Raussendorf, Jim Harrington, and Kovid Goyal. A fault-tolerant one-way quantum computer. *Annals of Physics*, 321(9):2242 – 2270, 2006. ISSN 0003-4916. doi: <http://dx.doi.org/10.1016/j.aop.2006.01.012>. URL <http://www.sciencedirect.com/science/article/pii/S0003491606000236>. 18
- Robert Raussendorf, Jim Harrington, and Kovid Goyal. Topological fault-tolerance in cluster state quantum computation. *New Journal of Physics*, 9(6):199, 2007. URL <http://stacks.iop.org/1367-2630/9/i=6/a=199>. 18, 29, 33, 39, 53, 61
- T. Richardson, A. Shokrollahi, and R. Urbanke. Finite-length analysis of various low-density parity-check ensembles for the binary erasure channel. In *IEEE International Symposium on Information Theory, 2002. Proceedings.*, 2002. doi: 10.1109/ISIT.2002.1023273. 55
- Tom Richardson and Rüdiger Urbanke. *Modern coding theory*. Cambridge University Press, 2008. 55
- P. Sarvepalli. Topological color codes over higher alphabet. In *Information Theory Workshop (ITW), 2010 IEEE*, pages 1–5, Aug 2010. doi: 10.1109/CIG.2010.5592860. 110, 111, 119
- Claude Elwood Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379423, 1948. 54

- Peter W. Shor. Scheme for reducing decoherence in quantum computer memory. *Phys. Rev. A*, 52:R2493–R2496, Oct 1995. doi: 10.1103/PhysRevA.52.R2493. URL <http://link.aps.org/doi/10.1103/PhysRevA.52.R2493>. 15, 21
- Peter W. Shor. Fault-tolerant quantum computation. In *37th Annual Symposium on Foundations of Computer Science, 1996. Proceedings.*, pages 56 – 65, Oct 1996. doi: 10.1109/SFCS.1996.548464. 16, 30
- 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. doi: 10.1137/S0097539795293172. URL <http://epubs.siam.org/doi/abs/10.1137/S0097539795293172>. 14
- A. Smith, B. E. Anderson, H. Sosa-Martinez, C. A. Riofrío, Ivan H. Deutsch, and Poul S. Jessen. Quantum control in the Cs $6S_{1/2}$ ground manifold using radio-frequency and microwave magnetic fields. *Phys. Rev. Lett.*, 111:170502, Oct 2013. doi: 10.1103/PhysRevLett.111.170502. URL <http://link.aps.org/doi/10.1103/PhysRevLett.111.170502>. 86
- Thomas M. Stace and Sean D. Barrett. Error correction and degeneracy in surface codes suffering loss. *Phys. Rev. A*, 81:022317, Feb 2010. doi: 10.1103/PhysRevA.81.022317. URL <http://link.aps.org/doi/10.1103/PhysRevA.81.022317>. 57, 64
- Thomas M. Stace, Sean D. Barrett, and Andrew C. Doherty. Thresholds for topological codes in the presence of loss. *Phys. Rev. Lett.*, 102:200501, May 2009. doi: 10.1103/PhysRevLett.102.200501. URL <http://link.aps.org/doi/10.1103/PhysRevLett.102.200501>. 33
- A. M. Steane. Error correcting codes in quantum theory. *Phys. Rev. Lett.*, 77:793–797, Jul 1996a. doi: 10.1103/PhysRevLett.77.793. URL <http://link.aps.org/doi/10.1103/PhysRevLett.77.793>. 15
- A. M. Steane. Active stabilization, quantum computation, and quantum state synthesis. *Phys. Rev. Lett.*, 78:2252–2255, Mar 1997. doi: 10.1103/PhysRevLett.78.2252. URL <http://link.aps.org/doi/10.1103/PhysRevLett.78.2252>. 16

- Andrew Steane. Multiple-particle interference and quantum error correction. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 452(1954):2551–2577, 1996b. 16
- Ashley M. Stephens. Fault-tolerant thresholds for quantum error correction with the surface code. *Phys. Rev. A*, 89:022321, Feb 2014. doi: 10.1103/PhysRevA.89.022321. URL <http://link.aps.org/doi/10.1103/PhysRevA.89.022321>. 18, 52, 65, 88
- M. Suchara, A. Faruque, C.-Y. Lai, G. Paz, F. T. Chong, and J. Kubitowicz. Comparing the overhead of topological and concatenated quantum error correction. *arXiv:quant-ph/1312.2316*, Dec 2013. 52
- W. G. Unruh. Maintaining coherence in quantum computers. *Phys. Rev. A*, 51: 992–997, Feb 1995. doi: 10.1103/PhysRevA.51.992. URL <http://link.aps.org/doi/10.1103/PhysRevA.51.992>. 15
- D. Veberic. Having fun with Lambert $W(x)$ function. *arXiv:cs.MS/1003.1628*, Mar 2010. 79
- M. Veldhorst, J. C. C. Hwang, C. H. Yang, A. W. Leenstra, B. de Ronde, J. P. Dehollain, J. T. Muhonen, F. E. Hudson, K. M. Itoh, A. Morello, and A. S. Dzurak. An addressable quantum dot qubit with fault-tolerant control-fidelity. *Nat Nano*, 9 (12):981–985, 12 2014a. URL <http://dx.doi.org/10.1038/nnano.2014.216>. 18
- M. Veldhorst, C.H. Yang, J.C.C. Hwang, W. Huang, J.P. Dehollain, J.T. Muhonen, S. Simmons, A. Laucht, F.E. Hudson, K.M. Itoh, A. Morello, and A.S. Dzurak. A two qubit logic gate in silicon. *arXiv:1411.5760*, 2014b. 18
- Chenyang Wang, Jim Harrington, and John Preskill. Confinement-Higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory. *Annals of Physics*, 303(1):31 – 58, 2003. ISSN 0003-4916. doi: [http://dx.doi.org/10.1016/S0003-4916\(02\)00019-2](http://dx.doi.org/10.1016/S0003-4916(02)00019-2). URL <http://www.sciencedirect.com/science/article/pii/S0003491602000192>. 33, 53, 57, 61, 63, 64, 65, 66, 86, 87, 88, 96
- David S. Wang, Austin G. Fowler, and Lloyd C. L. Hollenberg. Surface code quantum computing with error rates over 1%. *Phys. Rev. A*, 83:020302, 2011. 52

- Fern H. E. Watson, Hussain Anwar, and Dan E. Browne. Fast fault-tolerant decoder for qubit and qudit surface codes. *Phys. Rev. A*, 92:032309, Sep 2015a. doi: 10.1103/PhysRevA.92.032309. URL <http://link.aps.org/doi/10.1103/PhysRevA.92.032309>. 85
- Fern H. E. Watson, Earl T. Campbell, Hussain Anwar, and Dan E. Browne. Qudit color codes and gauge color codes in all spatial dimensions. *Phys. Rev. A*, 92:022312, Aug 2015b. doi: 10.1103/PhysRevA.92.022312. URL <http://link.aps.org/doi/10.1103/PhysRevA.92.022312>. 109, 115, 120, 132, 137
- Fern H.E. Watson and Sean D. Barrett. Logical error rate scaling of the toric code. *New Journal of Physics*, 16(9):093045, 2014. URL <http://stacks.iop.org/1367-2630/16/i=9/a=093045>. 51, 82
- W. K. Wootters and W. H. Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, 10 1982. URL <http://dx.doi.org/10.1038/299802a0>. 15
- James Wootton. A simple decoder for topological codes. *Entropy*, 17(4):1946, 2015. ISSN 1099-4300. doi: 10.3390/e17041946. URL <http://www.mdpi.com/1099-4300/17/4/1946>. 87
- Bei Zeng, Andrew Cross, and Isaac L. Chuang. Transversality versus universality for additive quantum codes. *arXiv:0706.1382*, Sept 2007. 28
- Huangjun Zhu. SIC POVMs and Clifford groups in prime dimensions. *Journal of Physics A: Mathematical and Theoretical*, 43(30):305305, 2010. URL <http://stacks.iop.org/1751-8121/43/i=30/a=305305>. 116