



# RELIABLE QUANTUM INFORMATION PROCESSING

By

Henry Lincoln Haselgrove  
B.Sc.(Hons), Flinders

A THESIS SUBMITTED TO THE UNIVERSITY OF QUEENSLAND  
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY  
SCHOOL OF PHYSICAL SCIENCES  
MAY 2006

Principal Advisor: *Prof. Michael A. Nielsen*  
Associate Advisor: *Prof. Gerard J. Milburn*

© Henry Lincoln Haselgrove, 2006.

Produced in L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>.

I hereby declare that this submission is my own work and to the best of my knowledge it contains no material previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any other degree or diploma at UQ or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by colleagues, with whom I have worked at UQ or elsewhere, during my candidature, is fully acknowledged.

I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.

Henry Lincoln Haselgrove, Candidate.

Michael A. Nielsen, Principal Advisor.



---

## Acknowledgements

---

I would like to begin by thanking my supervisor, Michael Nielsen. The opportunity to work closely with Michael is one I value very highly. Michael gave me encouragement to do my best, and gave all the support and guidance necessary for me to do just that. I wish to thank Michael for his painstaking corrections of the draft of this thesis.

During the first few months of my project, my “de-facto” supervisor was Tobias Osborne. I wish to acknowledge the enormous assistance he gave me then and since. I found his enthusiasm for quantum information highly infectious, and I have enjoyed the times I have spent collaborating with him.

In the last year and a half, I have been fortunate to collaborate closely with Chris Dawson. What would have been a gruelling and difficult investigation was made enjoyable by having shared it with Chris. I also wish to thank Chris for his comments on Chapter 2 of this thesis.

During my time as a PhD student I was an employee of the Defence Science and Technology Organisation. I wish to thank those in DSTO whose support made this project possible, including John Asenstorfer, John Kitchen, Warren Marwood, David Pulford, and Bruce Ward.

I was fortunate to spend time at a number of prestigious overseas institutions during my PhD. For their kind hospitality, I wish to thank Nick Bonesteel of Florida State University, Carl Caves of the University of New Mexico, and Ray Laflamme of the University of Waterloo. I would particularly like to thank John Preskill for hosting my five-month visit to the Institute of Quantum Information at the California Institute of Technology.

At UQ I have been surrounded by an amazing group of PhD students and postdocs. For their friendship, and for turning the last three and a half years from a valuable experience into an unforgettable one, I wish to thank Aggie Branczyk, Mick Bremner, Chris Dawson, Jennifer Dodd, Charles Hill, Andrew Hines, Vincent Loke, Tobias Osborne, Kenny Pregnell, David Poulin, Peter Rohde, Mohan Sarovar, and Christian Weedbrook.

Finally, I wish to thank my father, Frank, my mother, Pauline, and my stepmother Rainbow. Their love and support, although from afar, has meant a great deal to me.



---

## List of Publications

---

Below is a list of publications completed during my PhD. Those marked with a \* relate directly to this thesis.

- 1.\* C. M. Dawson, H. L. Haselgrove, and M. A. Nielsen. Noise thresholds for optical cluster-state quantum computation. *Phys. Rev. A* 73:052306 (2006). [eprint arXiv:quant-ph/0601066].
- 2.\* C. M. Dawson, H. L. Haselgrove, and M. A. Nielsen. Noise thresholds for optical quantum computers. *Phys. Rev. Lett.* 96:020501 (2006). [eprint arXiv:quant-ph/0509060].
3. C. M. Dawson, H. L. Haselgrove, A. P. Hines, D. Mortimer, M. A. Nielsen, T. J. Osborne. Quantum computing and polynomial equations over the finite field  $\mathbb{Z}_2$ . *Quant. Inf. Comp.* 5(2):102-112 (2005). [eprint arXiv:quant-ph/04080129].
- 4.\* H. L. Haselgrove. Optimal state encoding for quantum walks and quantum communication over spin systems. *Phys. Rev. A* 72:062326 (2005). [eprint arXiv:quant-ph/062326].
5. H. L. Haselgrove, M. A. Nielsen, T. J. Osborne. Entanglement, correlations, and the energy gap in many-body quantum systems. *Phys. Rev. A* 69:032303 (2004). [eprint arXiv:quant-ph/0308083].
6. A. M. Childs, H. L. Haselgrove, M. A. Nielsen. Lower bounds on the complexity of simulating quantum gates. *Phys. Rev. A* 68:052311 (2003). [eprint arXiv:quant-ph/0307190].
7. H. L. Haselgrove, M. A. Nielsen, T. J. Osborne. Practicality of time-optimal two-qubit Hamiltonian simulation. *Phys. Rev. A* 68:042303 (2004). [eprint arXiv:quant-ph/0303070].
- 8.\* H. L. Haselgrove, M. A. Nielsen, T. J. Osborne. Quantum states far from the energy eigenstates of any local Hamiltonian. *Phys. Rev. Lett.* 91:210401 (2003). [eprint arXiv:quant-ph/0303022].





---

## Abstract

---

*Quantum information processing* is the coherent manipulation of a quantum state for the purpose of performing an information processing task. It is known that certain tasks can in principle be performed much more efficiently using quantum information processing rather than ordinary classical processing. Examples include factoring, secure key distribution, and the simulation of quantum systems. However, significant theoretical and technological obstacles stand in the road to achieving these promised benefits. It is exceedingly difficult to construct devices that are able to achieve an extensive level of control of a quantum state while keeping that state well isolated from the effects of noise.

Although there is considerable experimental effort underway to build devices that are less noisy and more controllable, there is also a significant theoretical program aimed at finding schemes that make such physical limitations have less effect on the overall reliability of a device. The crowning achievement is the theory of *fault-tolerant quantum error-correction*, which shows that a noisy quantum device can be efficiently made to behave as though it were noise free, so long as the amount of noise present is below the “noise threshold”.

A central result of this thesis is the calculation of the noise threshold for optical cluster-state quantum computing. Optical cluster-state quantum computing is one of the most promising proposals for the physical implementation of a quantum computer (i.e., a generic quantum information processing device). Previous studies of the value of the threshold, that considered other physical implementations, do not apply to optical quantum computing due to the unusual features of the optical proposal such as non-deterministic gates and photon loss. We present the first detailed analysis of the value of the noise threshold for this proposal. Our analysis involves a number of innovations, including a method for error-correction known as telecorrection, whereby repeated error-syndrome measurements are guaranteed to agree due to the use of teleportation during the correction process.

This thesis also considers how to overcome limits to the amount of physical control able to be applied to a quantum device. We ask, if a quantum device can only control limited parts of its quantum state, can that device still be used to achieve a useful information processing task? We consider simple networks of interacting quantum spins where only a

few of the spins are controllable, and consider the problem of using this system for high-fidelity quantum communication (i.e., using the system as a simple “quantum wire”). Without control, such systems generally yield a very poor communication fidelity. We show that a very simple scheme that involves controlling a small number of the spins can be used to greatly increase the fidelity across the entire network. The scheme is designed using techniques of state encoding.

The thesis also considers the problem of engineering the interactions in a quantum device so that the ground state is a quantum error-correcting code. Such a problem is an important part of the proposal for *naturally fault-tolerant systems*, which have the ability to resist noise whilst using little or no external control. Our results prove that a certain important class of quantum error-correcting codes, the so called *nondegenerate codes*, cannot be the eigenstate of any physically-plausible quantum system. This result places significant restrictions on the design of naturally fault-tolerant devices, and sheds light on why current proposals for natural fault tolerance use codes that are *degenerate*.

---

# Contents

---

<b>List of Tables</b>	<b>xiii</b>
<b>List of Figures</b>	<b>xv</b>
<b>Notation and abbreviations</b>	<b>xvii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Quantum information theory . . . . .	2
1.2 Reliable quantum information processing . . . . .	4
1.3 Outline of the thesis . . . . .	5
<b>2 Quantum error-correction and fault-tolerance</b>	<b>9</b>
2.1 Classical error-correcting codes . . . . .	10
2.1.1 Maximum-likelihood decoding . . . . .	11
2.1.2 Classical repetition codes . . . . .	13
2.1.3 Classical linear codes . . . . .	13
2.1.3.1 Encoding linear codes . . . . .	14
2.1.3.2 Decoding linear codes . . . . .	15
2.1.3.3 Dual codes . . . . .	17
2.2 Quantum error-correcting codes . . . . .	17
2.2.1 Models for quantum noise . . . . .	19
2.2.2 Simple examples of quantum codes . . . . .	22
2.2.2.1 Quantum bit-flip code . . . . .	23
2.2.2.2 Phase-flip code . . . . .	25
2.2.2.3 Shor code . . . . .	26
2.2.2.4 Degenerate codes . . . . .	28
2.2.3 CSS codes . . . . .	29
2.2.3.1 Defining the CSS code states . . . . .	29
2.2.3.2 Circuits for creating CSS code states . . . . .	32
2.2.3.3 Logical operations on CSS-encoded qubits . . . . .	34
2.2.3.4 Error-correction circuits for CSS codes . . . . .	36

2.3	Fault tolerance . . . . .	40
2.3.1	Steane’s method for fault-tolerant error-correction . . . . .	43
2.3.2	Noise thresholds in fault-tolerant quantum systems . . . . .	49
2.3.3	Naturally fault-tolerant systems . . . . .	54
<b>3</b>	<b>Eigenstates of physically-plausible systems</b>	<b>57</b>
3.1	Introduction . . . . .	58
3.1.1	$L$ -local interactions . . . . .	59
3.2	Dimension-counting argument . . . . .	60
3.3	Simple test for physical plausibility . . . . .	61
3.4	Visualizing the manifold of allowed eigenstates . . . . .	64
3.5	Relationship between quantum codes and eigenstates of $L$ -local Hamiltonians	66
3.5.1	Nondegenerate codes . . . . .	67
3.5.2	Nondegenerate codes are “far away” from all plausible eigenstates .	68
3.5.3	Extension to $d$ -dimensional bodies . . . . .	70
3.5.4	Discussion . . . . .	71
3.6	Conclusion . . . . .	72
<b>4</b>	<b>State encoding for quantum communication over spin systems</b>	<b>73</b>
4.1	Introduction . . . . .	74
4.1.1	Assumptions and notation . . . . .	76
4.2	The optimal encoding scheme . . . . .	77
4.3	Dynamic control . . . . .	84
4.4	Conclusion . . . . .	90
4.5	Appendix A: Proof of Eq. (4.5) . . . . .	91
4.6	Appendix B: Interpretation of $\mathcal{C}_B(T)$ . . . . .	92
<b>5</b>	<b>Cluster states and optical quantum computation</b>	<b>95</b>
5.1	Cluster-state computation . . . . .	96
5.1.1	Basic elements of the cluster-state model . . . . .	96
5.1.2	Conversion from circuit to cluster (the hard way) . . . . .	98
5.1.3	The Pauli-frame method for feed-forward . . . . .	101
5.2	Optical quantum computing . . . . .	105
5.2.1	Basic elements . . . . .	106
5.2.2	The KLM scheme . . . . .	109
5.3	Optical cluster-state computation . . . . .	112
5.3.1	Effect of gate failures during cluster-state creation . . . . .	112

5.3.2	Nielsen's approach to efficient cluster creation . . . . .	113
5.3.3	The fusion gate . . . . .	115
<b>6</b>	<b>Noise thresholds for optical cluster-state quantum computation</b>	<b>119</b>
6.1	Introduction . . . . .	120
6.2	Physical setting . . . . .	122
6.3	Method for simulating a noisy cluster-state computation . . . . .	126
6.3.1	Example . . . . .	126
6.3.2	General description of the introduction and propagation of Pauli noise	127
6.4	Fault-tolerant protocol . . . . .	131
6.4.1	Introduction . . . . .	131
6.4.2	Broad picture of fault-tolerant protocol . . . . .	132
6.4.3	The cluster-based protocol . . . . .	132
6.4.3.1	Tools for optical cluster-state computing: microclusters, parallel fusion, and postselection . . . . .	134
6.4.3.2	Input states . . . . .	136
6.4.3.3	Ancilla creation . . . . .	137
6.4.3.4	Telecorrector creation . . . . .	140
6.4.3.5	Reduction of fusion gate failure and photon loss to Pauli errors . . . . .	143
6.4.3.6	Decoding . . . . .	145
6.4.3.7	Results of the optical cluster simulation . . . . .	147
6.4.4	The deterministic protocol . . . . .	150
6.4.4.1	Concatenation of protocols . . . . .	150
6.4.4.2	Effective noise model . . . . .	152
6.4.4.3	Telecorrection protocol . . . . .	154
6.4.4.4	Method for simulating the protocol . . . . .	155
6.4.4.5	Results of simulating the deterministic protocol . . . . .	156
6.4.5	Final Results . . . . .	161
6.4.6	Resource usage . . . . .	164
6.5	Conclusion . . . . .	166
6.6	Appendix: Telecorrection . . . . .	167
<b>7</b>	<b>Concluding remarks</b>	<b>171</b>
	<b>References</b>	<b>173</b>
	<b>Index</b>	<b>183</b>



---

## List of Tables

---

5.1	Rules for converting a quantum circuit to a cluster-state computation. . . .	105
6.1	The polynomial $E(\epsilon, \gamma)$ as fitted to the unlocated crash rate data, for the cluster-state protocol, using the 23-qubit code with memory noise enabled.	151
6.2	Estimated resource usage (number of Bell pairs consumed per computational operation) as a function of concatenation level, for noise parameters $(\epsilon, \gamma) = (4 \times 10^{-5}, 4 \times 10^{-4})$ , and using the 7-qubit code. . . . .	165





---

## List of Figures

---

2.1	Generic circuit for encoding one qubit in an $N$ -qubit quantum code. . . . .	18
2.2	Encoding circuit for the 3-qubit bit-flip code. . . . .	23
2.3	Error-correction circuit for the 3-qubit bit-flip code. . . . .	24
2.4	Encoding circuit for the 3-qubit phase-flip code. . . . .	26
2.5	Encoding circuit for the Shor code. . . . .	27
2.6	Error-correction circuit for the 9-qubit Shor code. . . . .	27
2.7	A circuit for creating the $ 0\rangle_L$ state for the Steane 7-qubit code. . . . .	33
2.8	A general circuit for encoding an arbitrary qubit state in the CSS code. . .	34
2.9	A circuit for measuring the $X$ syndrome of a noisy CSS-encoded qubit. . .	37
2.10	A circuit for measuring the $Z$ syndrome of a noisy CSS-encoded qubit. . .	39
2.11	An example unencoded circuit and corresponding encoded fault-tolerant version. . . . .	41
2.12	A comparison of the behaviour of error-correction procedures that are fault- tolerant and not fault-tolerant. . . . .	42
2.13	A logical operation that operates transversally is always fault-tolerant. . .	42
2.14	The non-fault-tolerant $X$ syndrome measurement circuit, for the 7-qubit Steane code. . . . .	44
2.15	Steane's fault-tolerant ancilla creation circuit, for the 7-qubit Steane code.	46
2.16	Complete fault-tolerant error-correction circuit for the 7-qubit Steane code	48
3.1	An example class of local interactions . . . . .	64
3.2	A random cross-section of the manifold of allowed eigenstates. . . . .	65
3.3	A random 3D cross-section of the manifold of allowed eigenstates . . . . .	66
4.1	The largest four singular values of $\tilde{U}^{(1)}(T)$ . . . . .	82
4.2	The optimal encoded state, evolved for a sequence of times $t$ . . . . .	82
4.3	The states $\vec{w}_2(75.75)$ and $\vec{w}_3(75.75)$ evolved for time=12. . . . .	83
4.4	The general setup, whereby control Hamiltonians $H_A(t)$ and $H_B(t)$ vary over time, to increase communication fidelity. . . . .	84
4.5	A modified version of Figure 4.4. . . . .	85





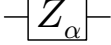

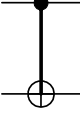
4.6	Control functions for a 104-spin $XY$ chain. . . . .	89
4.7	Control functions for a 29-spin $XY$ chain with randomly chosen coupling strengths. . . . .	90
5.1	Converting a qubit between polarization encoding and spatial encoding. . .	107
5.2	The beamsplitter with reflectivity $\cos^2 \theta$ , and its action on three example input states . . . . .	108
5.3	The KLM CPHASE gate with success probability $1/16$ . . . . .	110
5.4	Nielsen's scheme for building arbitrary-sized clusters. . . . .	114
5.5	The fusion gate. . . . .	115
6.1	Threshold region for the deterministic protocol using the 7-qubit Steane code. . . . .	159
6.2	Threshold region for the deterministic protocol using the 23-qubit Golay code. . . . .	160
6.3	Threshold region for the optical cluster protocol using the 7-qubit Steane code. . . . .	162
6.4	Threshold region for the optical cluster protocol using the 23-qubit Golay code. . . . .	163

---



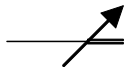
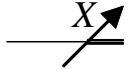
## Notation and Abbreviations

---

### *Gates*

Circuit notation	Description	Operator notation	Action
	Pauli $X$	$X$	$ 0\rangle\langle 1  +  1\rangle\langle 0 $
	Pauli $Y$	$Y$	$-i 0\rangle\langle 1  + i 1\rangle\langle 0 $
	Pauli $Z$	$Z$	$ 0\rangle\langle 0  -  1\rangle\langle 1 $
	Hadamard	$H$	$\frac{1}{\sqrt{2}}(X + Z)$
	Z-axis rotation	$Z_\alpha$	$e^{-i\alpha/2} 0\rangle\langle 0  + e^{i\alpha/2} 1\rangle\langle 1 $
	controlled-phase	CPHASE	$ 00\rangle\langle 00  +  01\rangle\langle 01 $ $+  10\rangle\langle 10  -  11\rangle\langle 11 $
	controlled-not	CNOT	$ 00\rangle\langle 00  +  01\rangle\langle 10 $ $+  10\rangle\langle 01  +  11\rangle\langle 11 $

### *Other circuit elements*

Circuit notation	Description
	quantum wire
	classical wire
	measurement, in a basis determined by context
	measurement in the $X$ basis

### ***Miscellaneous notation***

Term or notation	Description	(see page)
$(\ )^{\otimes n}$	$n$ -fold tensor product	
$C^\perp$	dual of the code $C$	17
$ +\rangle$	$\frac{1}{\sqrt{2}}( 0\rangle +  1\rangle)$	
$ -\rangle$	$\frac{1}{\sqrt{2}}( 0\rangle -  1\rangle)$	
KLM	Knill-Laflamme-Milburn (optical scheme)	109
CSS	Calderbank-Shor-Steane (code)	29
QECC	quantum error-correcting code	
QEC	quantum error-correction	

# Introduction

---

A number of remarkable phenomena occur as a result of placing a many-particle system in a pure, ordered quantum state. For example, if certain materials are cooled to their ground state they become superconductors. Similarly when certain gases are cooled they show the astonishing properties of a “superfluid”. A stream of neutral atoms that is sufficiently well-ordered will behave more like a laser beam than ordinary matter, and exhibit optical behaviour such as interference.

The study of *quantum information theory* has shown that if one is able to manipulate a pure many-particle quantum state in an appropriate way, an additional type of remarkable behaviour is possible: the ability to solve certain computational problems faster than is possible with an ordinary computer.

The question of exactly which computational problems can be solved faster on a “quantum computer” is still largely unknown. A handful of fast quantum algorithms have so far been developed, and there are ongoing efforts to find more. The progress to date is sufficiently encouraging that a large effort is underway to design and build a quantum computer, and other types of quantum information processing devices. This thesis is concerned with the problem of designing quantum information processing devices in a way that makes them operate reliably, despite the presence of noise or other physical limitations.

Keeping a quantum state free from noise is an extremely challenging task. If a quantum state is not completely shielded from unwanted interactions with the environment, then it will suffer *decoherence*. Decoherence is a process that adds randomness to a quantum state. As a system experiences more decoherence its behaviour will become in some sense “less quantum”, and so a quantum computer that suffers from too much decoherence will lose its ability to outpace ordinary classical computers [Aha99].

By contrast, the components of classical computers, which do not rely on quantum coherence to operate, can withstand considerable amounts of underlying noise. A signal in an ordinary computer is carried by the motion of conducting electrons. At room

temperature the motion of any given electron is essentially random, due to thermal noise. The formula  $\frac{1}{2}m_e v_{\text{rms}}^2 = \frac{3}{2}k_B T$  from classical statistical mechanics can be used to estimate the average speed of this random thermal motion at roughly one hundred kilometres per second. The presence of a signal will have a comparatively tiny effect on the electron's motion, typically an average drift of less than a millimetre per second. However, since all the many conducting electrons carry this same tiny drift velocity, the collective effect will be to amplify the signal and cause a relative suppression of the noise.

Unfortunately, it is generally not possible to use such simple types of collective motion to suppress noise in quantum devices. Rather, more sophisticated techniques, known as *quantum error-correction*, must be used. Such techniques form an important basis for the work described in this thesis.

The remainder of this introductory chapter is structured as follows. Sections 1.1 and 1.2 give a brief broad overview of quantum information theory, quantum algorithms, and quantum error-correction. All technical details are deferred to later chapters. Section 1.3 then outlines the structure and contents of the remaining chapters of this thesis.

## 1.1 Quantum information theory

Our understanding of how quantum information processing devices will operate comes from the study of *quantum information theory*. Quantum information theory is a relatively new field that combines information-theoretic concepts with the highly successful physical theory of quantum mechanics. Like many other physical theories, quantum mechanics is based on very simple rules, but the behaviour that results from these rules can be extremely complex, even for systems with only a small number of degrees of freedom. Broadly speaking, the aim of quantum information theory is to create new analytical tools for understanding the complex behaviour that arises in quantum systems.

For example, a successful tool developed in quantum information theory is the notion of *entanglement* as a physical resource<sup>1</sup>. Entanglement is a collective property of the state of many bodies, that relates to the presence of correlations of a manifestly quantum nature. Entanglement is understood to be a resource which is conserved under an important class of (“local unitary”) operations, can be inter-converted between different forms [BBPS96], and is consumed when performing certain types of quantum information processing tasks<sup>2</sup>.

<sup>1</sup>General discussions of the concept of entanglement as a physical resource can be found, for example, in Section 12.5 of [NC00] and in the introduction of [HDE<sup>+</sup>06].

<sup>2</sup>Examples include teleportation [BBC<sup>+</sup>93], certain schemes for quantum key distribution [BBM92, Eke91], and cluster-state computations [RB01].

It is known that if a quantum computer is to be powerful, its state must become highly entangled during the computation [JL03, Vid03].

Quantum information theory contains a wide range of other powerful techniques and results that provide insight into how information can be transformed in quantum systems. Some important examples include results regarding how quantum information can be compressed [Sch95], techniques for quantifying the information capacity of quantum channels [Sch96, SW97, Hol98], and a proof that quantum information cannot be cloned [WZ82, Die82].

Although the most obvious applications of quantum information theory relate to the study of quantum computers or other quantum information processing devices, there is a hope that quantum information theory will also provide important new insights into a wider class of complex quantum phenomena. For example, there is exciting progress being made towards new techniques for analysing condensed matter systems [Vid04, VC04]. In this work, a simple and efficient way is found to represent the quantum state of many-body systems, by taking advantage of the entanglement properties of such systems. It is expected that this insight will allow accurate, efficient numerical simulations to be performed (on ordinary classical computers) of important condensed matter systems such as high-temperature superconductors.

Perhaps the most astonishing result from quantum information theory is the existence of fast quantum algorithms. The most important example is Shor's factoring algorithm [Sho94]. The problem of factoring large integers is believed to be infeasible on classical computers. The infeasibility of factoring is the basis for the security of popular public-key cryptography systems. Amazingly, a quantum computer using Shor's algorithm will have the ability to factorize efficiently, in *polynomial time*<sup>3</sup>. (By comparison, the best known classical algorithm requires *super-polynomial time*). Certain other number-theoretic problems will also be efficiently solvable on a quantum computer, using variants of Shor's algorithm [ME99, EH00, Hal02, vDHI03].

Another class of fast quantum algorithms is based on *Grover's search algorithm* [Gro96]. The type of calculation that Grover's algorithm will speed up are those that would normally require a brute-force search to solve on a classical computer. A problem that would require searching through  $N$  possible solutions on a classical computer will take of order  $\sqrt{N}$  steps to solve on a quantum computer using Grover's algorithm. (Note that this speed-up is not as dramatic as that given by Shor's algorithm. Grover's algorithm cannot turn a super-polynomial-time solution into a polynomial-time solution.)

---

<sup>3</sup>That is, the amount of time required to run the algorithm scales as some polynomial in the size of the input.

There are two other important potential applications of quantum information processing that should be briefly noted. First, quantum computers will be very efficient at simulating other types of quantum systems [Zal98]. This could be particularly useful for purposes such as drug design. Second, the method of *quantum cryptography* can be used to allow cryptographic keys to be distributed securely [BB84].

## 1.2 Reliable quantum information processing

There are considerable practical difficulties that currently prevent the successful construction of a quantum computer. Broadly, these difficulties can be put into three categories: noise, controllability, and scalability. As discussed earlier in this introduction, quantum information processing devices are highly vulnerable to the effects of noise, and so it is necessary to carefully isolate quantum information processing devices from unwanted external disturbances. However, it is difficult to isolate a quantum system while at the same time applying the intricate level of control (i.e., gates, state preparations, and measurements, applied independently to each of the qubits) required to implement a quantum computation. The requirement of scalability will also be challenging, although less attention has been given to this issue so far since low-noise, controllable systems have not yet been demonstrated even for small numbers of qubits.

The detrimental effect that noise has on the overall reliability of a quantum device can be reduced by a sophisticated set of techniques known as *fault-tolerant quantum error-correction*. These techniques are reviewed in some detail in Chapter 2 of this thesis. So, the present discussion will be kept brief, and focused on some of the important challenges faced in improving current techniques for quantum error-correction. A series of theoretical results known as *noise threshold theorems* prove that, in principle, fault-tolerant error-correction can be used to efficiently make an arbitrarily long quantum computation work with arbitrarily high reliability, so long as the level of noise present is below some constant *noise threshold* value. That is, the noise threshold gives the maximum level of noise that can be efficiently and reliably tolerated in a quantum computer. The exact value of the noise threshold depends on the details of how the error-correction protocol is designed.

Fault-tolerant error-correction is a rapidly evolving field of research. There is much effort underway to find improved error-correction protocols, and to better understand the behaviour of existing schemes. One important aim is to find protocols having higher values of the noise threshold, with the ultimate aim being to increase the threshold beyond the level of noise present in real quantum devices. Recent progress along these lines has been



striking; in the last few years the highest-achieved value of the threshold has increased by almost a factor of ten [Ste03, Rei04, Kni05].

Effort is also required to make the resource-usage requirements of quantum error-correction techniques more practical. With current methods, when quantum error-correction is incorporated into a device, the complexity of the device must dramatically increase. The result is that almost all operations in an error-corrected device will be devoted to error-correction, and only a tiny fraction devoted to performing a useful information processing task. Thus, an important challenge is to simplify quantum error-correction protocols without sacrificing their ability to suppress noise.

It will also be necessary for the design and analysis of quantum error-correction protocols to be better tailored to the known properties of real physical systems. It is common to use quite abstract physical models when designing fault-tolerant error-correction protocols, and when determining the value of the noise threshold. Factors that are often neglected include geometric constraints that might limit which qubits can interact directly with each other, or unusual sources of noise that are particular to one type of physical system. More work is needed to determine how such factors affect the threshold, and how to best adjust the design of error-correction protocols between different physical systems. Some examples of recent work along these lines include studies of fault tolerance for semiconductor-spin devices [TED<sup>+</sup>05], ion traps [TMC<sup>+</sup>06], and generic two-dimensional systems [SDT06].

Most existing schemes for fault-tolerant quantum error-correction are closely related. They belong to a class of “active error-correction” protocols, which work by repeatedly creating certain *ancilla* states, and making these states interact with the data being corrected, after which the ancilla states are measured to determine the errors present on the data. There also exist radically different proposals for making a device resistant to noise, such as schemes for *natural fault-tolerance*. A naturally fault-tolerant device is one that achieves a resistance to noise without the need for complex external control. Further theoretical development of the concept of natural fault-tolerance would be valuable, since such techniques have the potential to dramatically simplify the construction of reliable quantum devices.

## 1.3 Outline of the thesis

The aim of this thesis is to develop new techniques for achieving reliable quantum information processing, and to provide new insights into existing techniques. The original

research contained in this thesis is naturally divided into three parts, and these correspond to three of the chapters (3, 4, and 6) in the body of the thesis. Two other chapters (2 and 5) are devoted to background material. Each chapter is outlined as follows.

Chapter 2 reviews the topics of quantum error-correction and quantum fault-tolerance. In their entirety, these fields are rather complex, due to the many details involved in the description of the various error-correction protocols and quantum codes that have so far been proposed. The aim of the chapter is to give an approachable, self-contained review of the aspects of these fields that are relevant to the later chapters of the thesis.

Chapter 3 presents new results concerning the design of naturally fault-tolerant systems. A naturally fault-tolerant system achieves resistance to noise due to the way that *static* interactions are arranged between bodies in the system. The idea is that the minimum-energy states (i.e., *ground states*) of the interactions form a quantum error-correcting code, so that when the system is cooled it will move to a state that belongs to the code. This makes the system resistant to noise without the need for complicated external control. Although there are several known examples of how to arrange interactions in a way that achieves natural fault-tolerance, there exists little in the way of general guiding principles for how to derive further examples. We prove that if a system is assumed to have physically-plausible interactions, then it can not possibly have a ground-state that is a *nondegenerate* code. (Nondegenerate codes are an important class of quantum error-correcting codes). This places significant restrictions on the design of naturally fault-tolerant systems, in particular that naturally fault-tolerant systems must be designed to use *degenerate* codes.

Chapter 4 considers the problem of dealing with errors that are not due to external noise but due to a lack of control over the internal dynamics of a system. Generally speaking, there are advantages in designing information-processing devices that don't require much external control. When there is less requirement for external control, it is usually easier to isolate qubits from external noise, and easier to scale systems to larger sizes. However, it is generally difficult to find ways of making a quantum device perform a useful information processing task when the external control of that device is severely restricted. We consider how a simple network of interacting quantum spins, where only a few of the spins are controllable, can be used to create a "quantum wire". That is, we consider how such systems can act as channel for high-fidelity quantum communication. Without control, such systems generally yield a very poor communication fidelity. We show that a simple scheme that involves controlling a small number of the spins can be used to greatly increase the fidelity across the entire network.

Chapters 5 and 6 are concerned with the recently proposed scheme for implementing a quantum computer via the method of *optical cluster-states*. This scheme is currently the most efficient way known of building a quantum computer from linear-optical components. However, little is known about how resilient this scheme is to the effects of noise. Existing calculations of the noise threshold do not apply to the optical cluster-state proposal, due to the unusual features that are present in optical cluster-state computing such as photon loss, nondeterministic gates, and cluster-state processing. We present the first detailed analysis of the value of the noise threshold for optical cluster-state quantum computing. Chapter 5 reviews optical cluster-state quantum computing. It contains introductions to the cluster-state model of quantum computation and the linear-optical implementation of quantum computing, and discusses how the two ideas are put together to create the optical cluster-state scheme. Chapter 6 then describes in detail our method for analysing the value of the noise threshold, and the results of this analysis. Much of the work described in this chapter involves the design of a new error-correction protocol that is tailored to the particular demands presented by the optical cluster-state implementation. Our protocol incorporates a number of innovations, including a method for teleported error-correction known as *telecorrection*. The final results of the chapter show that the value of the threshold is less than that given by previous calculations based on more abstract physical models, but the difference is less than an order of magnitude. Also, the threshold for photon-loss noise is found to be significantly greater than for other noise sources.

Concluding remarks are made in Chapter 7.



# Quantum error-correction and fault-tolerance

---

This chapter contains an introduction to the topics of quantum error-correction and fault-tolerant quantum computation. Rather than giving a complete description of either of these two fields, the intention of the present chapter is to give a gentle introduction to some of the ideas and techniques required for the later chapters of this thesis. Chapter 3 makes use of only relatively basic properties of error-correcting codes, whereas Chapter 6 makes detailed use of properties of a particular class of quantum codes known as Calderbank-Shor-Steane (CSS) codes, and of techniques by Steane for fault-tolerant quantum computation. (Chapters 4 and 5 are not directly concerned with quantum error-correction, although Chapter 4 does use quantum codes that have similarities with quantum error-correcting codes). For that reason, the current chapter goes into a fair amount of detail in reviewing the properties of CSS quantum codes, and in describing the fault-tolerant techniques of Steane. For a broader treatment of quantum error-correction and fault-tolerance, including historical perspectives and references to further literature, the reader is referred to, for example, Chapter 10 of [NC00].

The chapter is structured as follows. Section 2.1 reviews classical error-correction coding. Classical codes form a valuable point of comparison to quantum codes, and it is instructive to consider which concepts from classical coding theory can be carried over to the quantum case. I focus in particular on binary linear block codes, since codes of this type play an important role in the construction of CSS quantum codes.

In Section 2.2, I introduce quantum error-correction coding by first considering a simple description of generic quantum noise. Then I describe some of the simplest examples of quantum codes, including the bit-flip code, the phase-flip code, and the Shor code. I then describe in some detail properties of the CSS codes, including how to perform operations on the encoded qubits, and how to perform error correction.

Finally, Section 2.3 considers fault-tolerant quantum computation, the theory of how to make a quantum computer reliable when all its basic operations – including those used to

perform error-correction – are unreliable. I describe Steane’s method for performing fault-tolerant error-correction using CSS codes. I also discuss the concept of a *noise threshold* for fault-tolerant systems. I conclude the chapter with a brief mention of *naturally fault-tolerant* quantum systems.

*NOTE: This chapter was written solely for the purpose of inclusion in this thesis, and has not been published elsewhere. The contents of this chapter were written entirely by myself; however, as this is a review chapter, it doesn’t contain any of my own original research results.*

## 2.1 Classical error-correcting codes

Let’s start by considering how non-quantum (i.e., “classical”) information can be protected against noise by the use of error-correcting codes.

Broadly speaking, the purpose of an error-correcting code is to allow a message to be reliably sent through an unreliable communication channel. An error-correcting code works by transforming a message into a form that is more resistant to the effects of the channel. Usually, this is achieved by adding some kind of redundancy (such as repetition) to the message.

In order that we can consider some concrete examples of error-correcting codes, it is necessary to first introduce some formalities. We make the following assumptions about the channel, the message, and the abilities of the sender and receiver. Assume that the message “ $x$ ” is a string of  $k$  bits (i.e.,  $x \in \mathbb{Z}_2^k$ ). Also assume that, as far as the receiver knows, each of the  $2^k$  possible messages are equally likely. The sender is assumed to own a reliable computer, that is used to perform an *encoding* operation  $E(x)$  on the message before it is sent. The function  $E : \mathbb{Z}_2^k \rightarrow \mathbb{Z}_2^N$  maps  $k$ -bit strings into  $N$ -bit strings, where  $N > k$ . The encoded string  $E(x)$  is referred to as a *codeword*. The sender transmits each of the  $N$  bits of the codeword through the channel to the receiver.

At the other end of the channel, a “noisy version” of the  $N$  bits is received. We assume that the noise affects each bit independently, flipping the value of the bit with probability  $p$ . A channel which introduces noise this way is known as a *binary symmetric channel*. It is fairly common to consider this particular model of noise, since it is very simple yet still gives a reasonable approximation to many real noisy communication channels. The parameter  $p$  ranges from 0 to  $\frac{1}{2}$ , with 0 representing a perfect noise-free channel, and  $\frac{1}{2}$  representing a channel that obliterates the signal and outputs just random bits.

Let  $y$  denote the noisy  $N$ -bit string that is received. On average,  $y$  will differ from

$E(x)$  in  $Np$  different bit locations, due to the noise. Now, the receiver doesn't directly know which of the bits in  $y$  are affected by noise. Instead, the receiver must make a “best guess” about what noise occurred and what message was sent, based on the knowledge the receiver has about the encoding scheme  $E$  and about the general properties of the channel. The receiver computes  $D(y)$ , where  $D : \mathbb{Z}^N \rightarrow \mathbb{Z}^k$  is the function defined to give the most likely value for the message  $x$  given the received string  $y$ . The operation  $D(y)$  is referred to as the *decoding* operation. The receiver then assumes that the sent message was in fact  $D(y)$ , and the communication task is concluded.

Note that the general error-corrected communication procedure described above can be used to send messages longer than  $k$  bits by first breaking the message into blocks of length  $k$ , and repeating the procedure for each block. Codes that operate separately on different blocks in a message are referred to as *block codes*. This is opposed to the other main class of error-correcting codes known as *convolutional codes*, which I will not describe. (Similarly, my description of quantum codes in Section 2.2 will be limited to quantum block codes, and will not cover quantum convolutional codes. Block codes turn out to be particularly useful for fault-tolerant computing, whereas convolutional codes are less so.)

### 2.1.1 Maximum-likelihood decoding

Above I defined the decoding operation  $D(y)$  as calculating the most likely message  $x$  given a particular received bit string  $y$ . A decoding operation defined this way is known as a *maximum-likelihood decoder*. I now describe how maximum-likelihood decoding can be performed for the binary symmetric channel.

First we need two simple definitions: the Hamming weight and Hamming distance. Given a bit string  $a$ , the *Hamming weight*, denoted  $\text{wt}(a)$ , is equal to the number of bits in  $a$  which have a value 1. For example,  $\text{wt}(10011) = 3$ . The *Hamming distance*, denoted  $d(a, b)$ , for two equal-length bit-strings  $a$  and  $b$  is given by the number of positions at which  $a$  and  $b$  differ. For example,  $d(10011, 11111) = 2$ . Note that following relation holds:  $d(a, b) = \text{wt}(a + b)$ , where the sum is performed bitwise and modulo two.

Now, the definition of the decoding operation  $D(y)$  can be written more formally as

$$D(y) = \arg \max_x P(x|y), \quad (2.1)$$

where  $P(x|y)$  is defined to be the conditional probability of the original message being  $x$  given that the received string was  $y$ . The notation “ $\arg \max_x f(x)$ ” is used to mean the value of  $x$  which maximizes  $f(x)$ . (Note, if this is not a unique quantity, then we assume

that one of the possible values is chosen arbitrarily). Using Bayes' rule, this becomes

$$D(y) = \arg \max_x P(y|x) \frac{P(x)}{P(y)}. \quad (2.2)$$

Since the possible messages are selected with equal probability, then  $P(x)$  is the constant  $2^{-k}$ . Thus, the right hand side of Eq. (2.2) is of the form  $\arg \max_x [P(y|x)c]$ , where  $c$  is a positive value that does not depend on  $x$ . Thus, the factor  $c$  can be omitted from the  $\arg \max$ , giving

$$D(y) = \arg \max_x P(y|x). \quad (2.3)$$

Now, it can be shown that

$$P(y|x) = (1 - p)^{N-d(y,E(x))} p^{d(y,E(x))}. \quad (2.4)$$

To see this, note that if  $E(x)$  was sent but  $y$  was received, then channel noise must have transformed  $E(x)$  to  $y$ , meaning that a particular pattern of  $d(y, E(x))$  bit-flips occurred. The probability of such an event, given that each bit is flipped by the channel with probability  $p$ , is thus given by Eq. (2.4).

It turns out that the right hand side of Eq. (2.4) is a monotonically-decreasing function of  $d(y, E(x))$ , and so the expression  $P(y|x)$  in Eq. (2.3) can be replaced by  $d(y, E(x))$ , so long as the “arg max” is changed to an “arg min”. That is,

$$D(y) = \arg \min_x d(y, E(x)). \quad (2.5)$$

The advantage of Eq. (2.5) is that it gives a direct procedure for performing maximum-likelihood decoding. That is, to find  $D(y)$  simply loop over all the possible messages  $x$ , and for each case find the corresponding codeword  $E(x)$  and calculate the Hamming distance between  $E(x)$  and  $y$ . The choice of  $x$  that minimizes this Hamming distance is then taken as the value of  $D(y)$ . (In practice however, it can be difficult to search through all  $2^k$  messages when  $k$  is large. There do, however, exist a range of techniques that can help simplify this task somewhat.)

How often the correct message is obtained by the decoder depends on the noise parameter of the channel and also on the design of the code. The amount of noise that a code can successfully correct is indicated by an important property of the code known as the *minimum distance*. The minimum distance of a code is defined to be

$$d_{\min} = \min_{x \neq z} d(E(x), E(z)). \quad (2.6)$$

That is,  $d_{\min}$  is the minimum Hamming distance between any two codewords in the code. So long as less than  $\frac{1}{2}d_{\min}$  of the bits in a codeword are flipped by noise, then the message is guaranteed to be recovered perfectly using maximum-likelihood decoding.



### 2.1.2 Classical repetition codes

Consider an extremely simple class of codes known as the *repetition codes*. Repetition codes have a block length of  $k = 1$  (they are used to encode messages containing just one bit), and the encoding works by simply repeating the message bit  $N$  times. For example, for  $N = 5$  the encoding operation is  $E(0) = 00000$  and  $E(1) = 11111$ .

Maximum-likelihood decoding is performed by simply taking a “majority vote” of the received bits. So, if the received string has more 0s than 1s then the message is decoded to 0, and if the received string has more 1s than 0s the message is decoded to 1. For example,  $D(11000) = 0$  and  $D(11101) = 1$ . The larger the length  $N$  of the repetition code, the more noise it is able to withstand. For any amount of channel noise (except the maximum of  $p = \frac{1}{2}$ ), reliable communication will be possible by using a repetition code with sufficiently large  $N$ .

Nevertheless, repetition codes are not usually used in practice. The problem with repetition codes is that they have an inefficient *rate*. The rate of a block code is defined to be the fraction  $k/N$ , that is, the average number of message bits communicated per use of the channel. Generally speaking, given a particular noise parameter  $p$  and a particular desired reliability of communication, the repetition code that achieves this reliability will have a rate that is much poorer than other more sophisticated block codes that give the same reliability.

### 2.1.3 Classical linear codes

*Linear codes* are a special class of block codes. The biggest advantage that linear codes have over other block codes is their simple, compact description. The codewords of a linear code are specified indirectly via an  $N \times k$  matrix known as a *generator matrix*. Each codeword is found by selecting some subset of the  $k$  columns of the generator matrix, and adding them together modulo 2, to form an  $N$ -bit codeword. The  $2^k$  ways of doing this give the  $2^k$  different codewords required to encode arbitrary  $k$ -bit messages. Thus, an arbitrary linear code is specified by just the  $N \times k$  bits of the generator matrix, compared to a more general block code whose encoding operation requires  $N \times 2^k$  bits in order to be specified (that is,  $N$  bits for each of the  $2^k$  codewords).

Linear codes are usually analysed using tools and language borrowed from linear algebra. However, it is important to keep in mind that the linear algebra used in this context is all over the field  $\mathbb{Z}_2$ ; so all the basic arithmetic performed inside operations such as matrix products, inner products, linear combinations, etc., is performed over  $\mathbb{Z}_2$ . Many

of the results and techniques used in real or complex linear algebra are also applicable to linear algebra over  $\mathbb{Z}_2$ . One important difference however, is that the inner product of a nonzero vector with itself can be zero. (This is the case for example with the vector  $(1, 1, 1, 1)$ ). This fact causes certain techniques such as Gram-Schmidt orthogonalization to fail.

### 1 – Encoding linear codes

If  $x$  is a  $k \times 1$  vector containing the message bits, and  $G$  is the  $N \times k$  generator matrix of a linear code, then the encoding operation can be expressed as a matrix multiplication, as follows:

$$E(x) = Gx. \quad (2.7)$$

The matrix  $G$  maps the space  $\mathbb{Z}_2^k$  of possible messages into  $\mathbb{Z}_2^N$ . Hence, the set of codewords forms a  $k$ -dimensional vector subspace of  $\mathbb{Z}_2^N$  (assuming that  $G$  has been chosen to have linearly-independent columns), and so the set of codewords of a linear code is sometimes referred to as the *codespace*.

A simple example of a linear code is the repetition code, considered in the previous subsection. It's easy to see that the generator matrix for the  $N = 3$  repetition code is given by

$$G_{(\text{rep}3)} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}. \quad (2.8)$$

The example above encodes  $k = 1$  bits, and has minimum distance  $d_{\min} = 3$ . A more sophisticated example of a linear code is a *Hamming code*. There are different versions of the Hamming code having different sizes. The  $N = 7$  version has the following generator matrix,

$$G_{(\text{Hamming}7)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}, \quad (2.9)$$

and encodes  $k = 4$  bits into  $N = 7$  bits, with a minimum distance  $d_{\min} = 3$ . As we will see in Subsection 2.2.3, the 7-bit Hamming code turns out to be central to the construction of an important *quantum* code, known as the Steane code.

For any given linear code, there is some freedom as to how the generator matrix is written. The generator matrix can be altered by *elementary column operations*, and still represent the same code. An elementary column operation is where one column of the matrix is replaced by the sum of that column with another column. Moreover, the *rows* of the generator matrix may be permuted, and the resulting matrix will still represent a code with essentially the same properties, since permuting rows just changes the order in which bits in the codeword are sent through the channel. Now, it can be shown that by using an appropriate combination of the column and row operations described above, any generator matrix can be transformed into *standard form*. A generator matrix is in standard form when it can be written as  $G = \begin{pmatrix} I_k \\ A \end{pmatrix}$ , which means that the first  $k$  rows of  $G$  equal the  $k \times k$  identity matrix, and the last  $N - k$  rows equal some  $(N - k) \times k$  matrix, denoted  $A$ . (For example, the matrices in Eqs. (2.8) and (2.9) are both in standard form).

Having a generator matrix in standard form allows the effect of the encoding operation to be understood a little more clearly. When  $G = \begin{pmatrix} I_k \\ A \end{pmatrix}$ , then  $Gx = \begin{pmatrix} x \\ Ax \end{pmatrix}$ . So, the codeword that corresponds to a message  $x$  can be described as follows: the first  $k$  bits of the codeword are just an exact copy of  $x$ , and the remaining  $N - k$  bits contain a series of *parity-check bits*. The parity-check bits are redundant information about  $x$ . Each parity-check bit is equal to the parity (i.e., sum modulo 2) of some subset of the message bits.

## 2 – Decoding linear codes

For linear codes, maximum-likelihood decoding is usually achieved by using a technique called *syndrome decoding*. In the following few paragraphs I describe this technique. To simplify the description I assume that  $G$  is in standard form throughout.

To begin, denote the received bit string as  $y = \begin{pmatrix} a \\ b \end{pmatrix}$ , where  $a$  is the first  $k$  bits of  $y$ , and  $b$  is the last  $N - k$  bits. The first step in the procedure for syndrome decoding is for the receiver to calculate the *syndrome* of  $y$ , defined to be the vector equal to  $Aa - b$ . Note that the syndrome  $Aa - b$  can alternatively be expressed as  $Hy$ , where  $H$  is the *parity check matrix* of the code, equal<sup>1</sup> to

$$H = (A; I_{N-k}). \quad (2.10)$$

Now, from the discussion above regarding the standard form of  $G$ , it follows that the syndrome equals zero if and only if  $y$  is a valid codeword. (It's a rather remarkable

<sup>1</sup>If the generator matrix has not been given in standard form, the following more general definition of the parity check matrix can be used:  $H$  is any  $(N - k) \times N$  matrix with linearly-independent rows such that it satisfies  $HG=0$ .

property of linear codes that such a simple test exists for determining whether  $y$  is valid codeword – considering that for a more general block code it would be necessary to instead perform a laborious search over every codeword  $E(x)$  to test whether one of them matched  $y$ ). Thus, if the receiver finds that the syndrome  $Hy$  is zero, then it is most likely that  $y$  contains no errors, and the decoded message string should in this case be taken to be the first  $k$  bits of  $y$ .

On the other hand, if the receiver finds that  $Hy \neq 0$ , then the full syndrome-decoding procedure needs to be performed, as described below. The approach taken by syndrome decoding is to find the most likely pattern of errors introduced by the channel. That is, if we write  $y = Gx + e$ , where  $Gx$  is the encoded message and  $e$  is the error pattern, the aim of syndrome decoding is to find the most likely value of  $e$ , which in turn indicates the most likely value for  $x$ . The following function is defined to be the *syndrome decoder*:

$$D_S(Hy) = \arg \max_{e' \in \mathbb{Z}_2^N} P(e'|Hy). \quad (2.11)$$

In the above,  $P(e'|Hy)$  is the conditional probability that the error pattern equals  $e'$ , given that the syndrome is  $Hy$ . So,  $D_S(Hy)$  calculates the most likely error pattern given the syndrome  $Hy$ . Note that  $D_S(Hy)$  has the following relationship to the decoding function  $D(y)$  defined back in Eq. (2.1):

$$GD(y) = y - D_S(Hy). \quad (2.12)$$

Let's consider how the definition in Eq. (2.11) may be re-expressed in a way that is convenient to calculate. It can be shown that  $HGx = 0$  for all messages  $x$ , so we have that  $Hy = H(Gx + e) = He$ . This means that, knowing  $Hy$ , we need only consider potential error patterns  $e'$  that satisfy the condition  $He' = Hy$ , and so Eq. (2.11) can be re-written as follows:

$$D_S(Hy) = \arg \max_{e': He' = Hy} P(e'|Hy). \quad (2.13)$$

Using a line of reasoning similar to that used in Subsection 2.1.1, it can be shown that the  $e'$  which maximizes  $P(e'|Hy)$  is the one with minimum Hamming weight, so Eq. (2.13) can be re-written as follows:

$$D_S(Hy) = \arg \min_{e': He' = Hy} \text{wt}(e'). \quad (2.14)$$

Note that the set  $\{e' : He' = Hy\}$  of error patterns having the correct syndrome can be found by taking the codespace and adding  $y$  to each element. That is,

$$\{e' : He' = Hy\} = \{Gx + y : x \in \mathbb{Z}_2^k\}. \quad (2.15)$$

So, using the language of group theory, the set of vectors having a particular syndrome is a *coset* of the space of codewords. The vector  $D_S(Hy)$  is called the *coset leader* for the syndrome  $Hy$ . To summarize, the most likely error pattern,  $D_S(Hy)$ , can be calculated by computing  $Gx + y$  for every  $x \in Z_2^k$ , and finding which one has minimum Hamming weight.

On the face of it, computing the syndrome decoder  $D_S(Hy)$  may not seem easier than computing the ordinary decoding function  $D(y)$  of Eq. (2.1), since both involve a search over  $2^k$  terms. However, in speed-critical applications it is preferable to have the decoding function pre-computed in memory, for all possible inputs. The function  $D$  has  $2^N$  possible inputs whereas  $D_S$  has  $2^{N-k}$ , so there is potentially a huge memory saving in using  $D_S$  instead of  $D$ . A pre-computed version of the syndrome decoder is often called a *standard decoding array*.

### 3 – Dual codes

Consider one final definition relating to linear codes, that will be useful in our discussion of quantum codes. Say a linear code has generator matrix  $G_1$  and parity matrix  $H_1$ , and encodes  $k$  bits into  $N$ . Then the *dual* of this code is another linear code, defined to have generator matrix  $G_2 = H_1^T$  and parity check matrix  $H_2 = G_1^T$ , where  $T$  denotes matrix transpose. The dual code will encode  $N - k$  bits into  $N$ . Note that an equivalent way of defining the dual code is as follows: the set of codewords in the dual code consist of the set of all bit-strings that have a zero inner product with every codeword of the original code. Note also, it can be shown that the *dual of the dual* is equal to the original code.

As an example, consider taking the dual of the  $N$ -bit repetition code. The result is a code which encodes  $N - 1$  bits into  $N$  bits, and where each codeword is found by first writing down an  $(N - 1)$ -bit message, and adding a single parity bit equal to the sum of the message bits. Such a code has  $d_{\min} = 2$ , and can detect when a single error has occurred, but not correct it.

## 2.2 Quantum error-correcting codes

Now that we have finished reviewing classical error-correction, it is worthwhile to start this section by giving a warning about just how drastically different *quantum* error-correction is. Quantum theory places severe restrictions on how quantum information can be measured and copied, restrictions that do not exist in the classical case.

The no-cloning theorem [WZ82, Die82] shows that qubits cannot be copied (except when those qubits are being used to store classical information). So, the simple idea of the

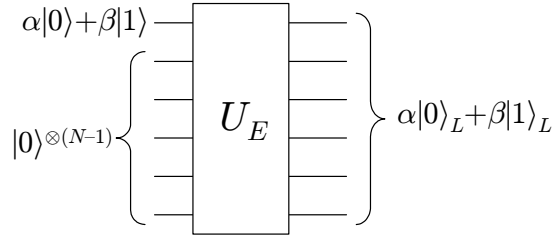


Figure 2.1: Generic circuit for encoding one qubit in an  $N$ -qubit quantum code.

classical repetition code, whereby the message is repeated many times, cannot generalize to the quantum case because the copying required for the encoding operation  $|\psi\rangle \rightarrow |\psi\rangle^{\otimes N}$  is impossible.

Other difficulties that are particular to quantum error-correction include (1) the fact that measuring a qubit destroys its state, (2) the fact that the state of a qubit is described as a point in a complex vector space (which is a continuous quantity, in contrast to discrete classical bits) and (3) that there are infinitely many different types of error that can affect a qubit, unlike the single bit-flip error of classical bits. Given all this, it's a remarkable fact that quantum error-correction is possible at all, and it's even more remarkable that certain quantum error-correcting codes have a close relationship to classical linear codes.

In a quantum code, redundancy is not achieved by duplicating a qubit; rather, the encoding operation can be thought of as “thinly spreading” a single copy of the qubit state across many other qubits. In broad terms, the procedure to encode a one-qubit message into  $N$  qubits can be described as follows (and is depicted in Figure 2.1). The sender takes the qubit containing the message, as well as an extra  $N - 1$  qubits that have been initialized to the  $|0\rangle$  state, and then applies some  $N$ -qubit unitary operation  $U_E$  (where the  $E$  stands for “encoder”) to the joint state of all the qubits. That is, if the message state is written as  $\alpha|0\rangle + \beta|1\rangle$ , then the encoding process can be written as

$$\begin{aligned} \alpha|0\rangle + \beta|1\rangle &\xrightarrow{\text{encoding}} U_E [(\alpha|0\rangle + \beta|1\rangle) \otimes |0\rangle^{\otimes(N-1)}] \\ &= \alpha|0\rangle_L + \beta|1\rangle_L, \end{aligned} \tag{2.16}$$

where  $|0\rangle_L$  and  $|1\rangle_L$  are the  $N$ -qubit states defined by  $|0\rangle_L \equiv U_E [|0\rangle \otimes |0\rangle^{\otimes(N-1)}]$  and  $|1\rangle_L \equiv U_E [|1\rangle \otimes |0\rangle^{\otimes(N-1)}]$ . (Here,  $L$  stands for “logical”, a term that is often used in this context instead of “encoded”).

In some sense, an encoded qubit is mathematically equivalent to a qubit that is not encoded, since either case is just an arbitrary superposition of two orthogonal vectors<sup>2</sup>.

<sup>2</sup>Admittedly, in the two cases the vectors belong to spaces of different dimension. However we can

In other words, the encoding operation is just a change of basis! There are of course important *physical* differences between encoded and unencoded qubits, in that the encoded basis states  $|0\rangle_L$  and  $|1\rangle_L$  are states of a system of  $N$  physical qubits, whereas the unencoded basis states  $|0\rangle$  and  $|1\rangle$  are states of just a single qubit. It is this difference which makes an encoded qubit more resilient to noise – but only when the noise satisfies certain conditions for being “physically reasonable”.

Note that in this section I make use of similar broad assumptions to that of the previous section on classical codes. That is, I assume the sender and receiver are each able to perform perfect noise-free operations on any state they control, and that noise is only introduced to a state when it passes through the channel between the sender and the receiver. Using these assumptions helps simplify the description of quantum error-correction techniques. Later, in Section 2.3, I consider the much more realistic case where *all* the components used to perform error-correction are themselves subject to noise.

### 2.2.1 Models for quantum noise

What types of noise should a quantum error-correcting code be able to correct? Of course, the answer to this question should be (ultimately) motivated by the physics of the situation – that is, be determined by the particular noise sources present in a system of interest. However, the task of understanding and accurately describing the noise processes in a quantum system can in practice be rather difficult. Luckily, it turns out that the precise details of the noise do not usually matter when designing a quantum code. Remarkably, when a quantum code is designed to protect against a certain type of simple noise known as the *independent depolarization channel*, such a code will automatically be able to protect against a large class of more complicated – and physically realistic – noise types.

Let’s consider a general way of describing physical noise processes. Imagine that an  $N$ -qubit system labelled  $D$  for “data” contains a quantum state that we are trying to protect, and that noise is introduced to  $D$  because of some unintended interaction between  $D$  and some external bodies that are collectively denoted  $E$  for “environment”. The interaction between  $D$  and  $E$  is described by a unitary operation  $U_{DE}$ , so that if the initial state of  $D$  and  $E$  are given by the density matrices  $\rho_D$  and  $\rho_E$ , then the joint state of  $D$  and  $E$  after the noise occurs is given by

$$\rho_D \otimes \rho_E \xrightarrow{\text{data-env. interaction}} U_{DE} (\rho_D \otimes \rho_E) U_{DE}^\dagger. \quad (2.17)$$

Assuming that the state of the environment is not directly accessible after the noise event, 

---

always imagine that the unencoded qubit is accompanied by another  $N - 1$  qubits in some fixed state.

then the appropriate way to describe the final state of  $D$  is to take the partial trace of the right hand side of Eq. (2.17) over the system  $E$ . Thus, the function  $\mathcal{E}(\rho_D)$ , defined to be the operation that maps the initial state on  $D$  to the final noisy state on  $D$ , is given by

$$\mathcal{E}(\rho_D) = \text{tr}_E \left[ U_{DE} (\rho_D \otimes \rho_E) U_{DE}^\dagger \right]. \quad (2.18)$$

It can be shown (see for example Section 8.2.3 of [NC00]) that  $\mathcal{E}(\rho_D)$  can be equivalently expressed in terms of operators acting only on the system  $D$ , via the so-called *operator-sum representation*,

$$\mathcal{E}(\rho_D) = \sum_k E_k \rho_D E_k^\dagger, \quad (2.19)$$

where the  $E_k$ , known as *operation elements*, are some set of operators<sup>3</sup> that satisfy  $\sum_k E_k^\dagger E_k = I$ . The operator-sum representation is a very flexible way of describing many different types of quantum operation, including unitary gates, state preparation, measurements<sup>4</sup>, and mixtures of all the above.

Thus, by modelling noise on  $D$  as being some arbitrary interaction with an environment, the resulting description of the effect of the noise on  $D$  is an arbitrary quantum operation. We cannot hope to design a code that protects against an arbitrary quantum operation, so we need to introduce some further physically-motivated assumptions to restrict the form of the noise.

Most importantly, we assume that the noise conforms to an *independent error model*. That is, we assume that the noise on the  $N$  data qubits can be broken down into the tensor product of noise acting independently on each qubit:

$$\mathcal{E}(\rho_D) = (\mathcal{E}_1)^{\otimes N}(\rho_D), \quad (2.20)$$

where  $\mathcal{E}_1$  is some 1-qubit quantum operation. In effect, this assumes that each qubit interacts with its own separate (but identical) environment. Due to the fact that the real environment consists of a such an enormous number of degrees of freedom, the assumption that each qubit interacts with an independent part of the environment is often approximated well in practice.

A further assumption that we make is that the noise acting on each qubit is “not too strong”, in a way made precise as follows. Assume that the single-qubit noise operation

<sup>3</sup>It is relatively straightforward to relate the  $E_k$  with  $U_{DE}$  and  $\rho_E$ , as follows. Write an orthonormal basis for the system  $E$  as  $|e_k\rangle$ . Assume without loss of generality that  $\rho_E$  is the pure state  $|e_0\rangle$ , since the degrees of freedom involved in choosing a more general  $\rho_E$  can be absorbed into the unitary  $U_{DE}$ . Then, it follows easily that  $E_k = (I_D \otimes \langle e_k|) U_{DE} (I_D \otimes |e_0\rangle)$ , where  $I_D$  is the identity acting on system  $D$ .

<sup>4</sup>Note, the effect of measurements operations can only be described this way in the case that the measurement result is not known.



$\mathcal{E}_1$  can be written in the following form,

$$\mathcal{E}_1(\rho) = p \mathcal{E}_*(\rho) + (1 - p)\rho, \quad (2.21)$$

for some other single-qubit quantum operation  $\mathcal{E}_*$ , and for some  $0 < p < 1$ . So, the noise that occurs on a qubit can then be interpreted as follows: with probability  $p$  the error described by the operation  $\mathcal{E}_*$  occurs, and with probability  $(1 - p)$  no error occurs. Thus, the smaller the value of  $p$ , the weaker the noise. (Note that not all weak single-qubit noise operations can be written in this way; that is, as a mixture of a high-probability error-free term with a low-probability error term. However, it is possible for the argument that follows to be modified so that it uses a more general notion of weak noise. See Section 10.3.2 of [NC00] for further details).

Now, say that the initial state  $\rho_D$  is chosen to be a codeword of a quantum error-correcting code, where the code has the property that it can correct the error  $\mathcal{E}_*$  applied to as many as  $t$  (where  $t \geq 1$ ) of the  $N$  qubits. That is, we are assuming that there exists some recovery operation  $\mathcal{R}$ , such that if a codeword  $\rho_D$  is affected by the error  $\mathcal{E}_*$  on any subset of up to  $t$  of the qubits, then applying  $\mathcal{R}$  will return the state to  $\rho_D$ .

Now imagine that the independent noise of Eq. (2.20) is applied to the codeword  $\rho_D$ , followed by the recovery operation  $\mathcal{R}$ . It follows that the resulting state,  $\mathcal{R}(\mathcal{E}(\rho_D))$ , corresponds to one that, with a probability that scales with  $1 - O(p^{t+1})$ , is equal to the error-free state  $\rho_D$ , and with probability that scales with  $O(p^{t+1})$  contains an error that was not corrected. (This is because the probability that the operation  $\mathcal{E}(\rho_D)$  in Eq. (2.20) introduces the error  $\mathcal{E}_*$  to more than  $t$  of the qubits, scales as  $O(p^{t+1})$ ). So, a *non-encoded* qubit that is subject to the channel  $\mathcal{E}_1$  would suffer an error with probability  $p$ , but on the other hand an *encoded* state that is sent through the channel would have a resulting error probability after the recovery operation equal to  $O(p^{t+1})$ . Clearly then, for some sufficiently small value of  $p$ , the reliability is increased by having performed error-correction.

However, the assumption that the code corrects the error  $\mathcal{E}_*$  on up to  $t$  qubits is an unsatisfactory one, since it involves defining the properties of the code with respect to the details of a specific noise operation  $\mathcal{E}_*$ . The key to relaxing this assumption is the following result:

**Result 2.1** (Discretization of errors): *Suppose a quantum code has the property that it can correct errors of the following type: all possible tensor products of Pauli operators on some specific subset of the qubits. Then, such a code can correct any type of error applied to that subset of the qubits.*

So, for example, if a code can correct  $X$ ,  $Y$  and  $Z$  errors on a particular qubit, then it can correct *any* quantum operation  $\mathcal{E}_*$  on that same qubit. Result 2.1 can be proven by making use of the so-called *quantum error-correction conditions* (given as Theorem 10.1 in [NC00]), which are a formal statement of the necessary and sufficient conditions for a quantum code to be able to correct a given set of errors. I won't reproduce the proof of Result 2.1 here, but the essence of it is as follows. First, the quantum error-correction conditions can be used to prove that if a quantum code corrects some set of errors, then it can also correct any linear combination of those errors. The proof is completed by noting that the set of all tensor products of Pauli operators, on some number of qubits, forms a basis for all linear operators on the state space of those qubits, and so any error on that set of qubits can be expressed as a linear combination of Pauli errors.

Result 2.1 can also be verified directly for particular codes and recovery procedures, for example those given later in Subsection 2.2.3. As we'll see, the recovery procedures described in that section involve performing indirect measurements on the encoded state, in order to infer which Pauli errors have affected the state. If the errors are not Pauli, but some linear combination (that is, superposition or mixture) of Pauli operators, then the act of measuring will “collapse” the effective error down to a particular Pauli error. So, just by using a recovery procedure that operates under the assumption that errors consist of just Pauli operators, the errors will then automatically change themselves into that form.

Combining Result 2.1 with the preceding discussion on the independent error model, it follows that if a quantum code can correct all tensor products of Pauli errors applied to any subset of  $t$  qubits (where  $t > 1$ ), then the code will suppress the effective error probability (compared with a non-encoded qubit) for any channel that corresponds to an independent error model, so long as the noise strength is not too great.

Note that a simple type of independent noise model that is often used to judge the performance of quantum error-correcting codes is the *independent depolarizing channel*. This channel is defined by setting  $\mathcal{E}_*(\rho) = \frac{1}{4}(\rho + X\rho X + Y\rho Y + Z\rho Z)$  in Eqs. (2.20) and (2.21). That is, each qubit with probability  $p$  experiences a *depolarization error*, consisting of one of the Pauli operations applied at random.

## 2.2.2 Simple examples of quantum codes

It is instructive to consider some very simple examples of quantum codes, as a way of introducing some further ideas and techniques. In this subsection I describe the bit-flip code, the phase-flip code, and the Shor code.

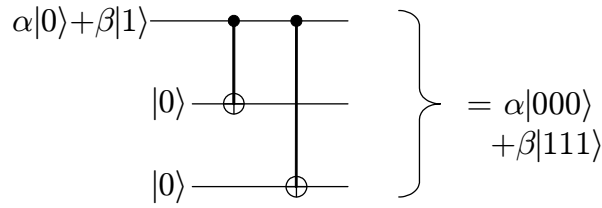


Figure 2.2: Encoding circuit for the 3-qubit bit-flip code.

### 1 – Quantum bit-flip code

The bit-flip code is not really a fully-fledged quantum error-correcting code, since it cannot correct arbitrary Pauli errors on even a single qubit. Instead, as its name implies, it is designed to correct just  $X$  errors (i.e., bit-flips). The smallest version of the bit-flip code encodes a single qubit into 3 qubits, as follows:

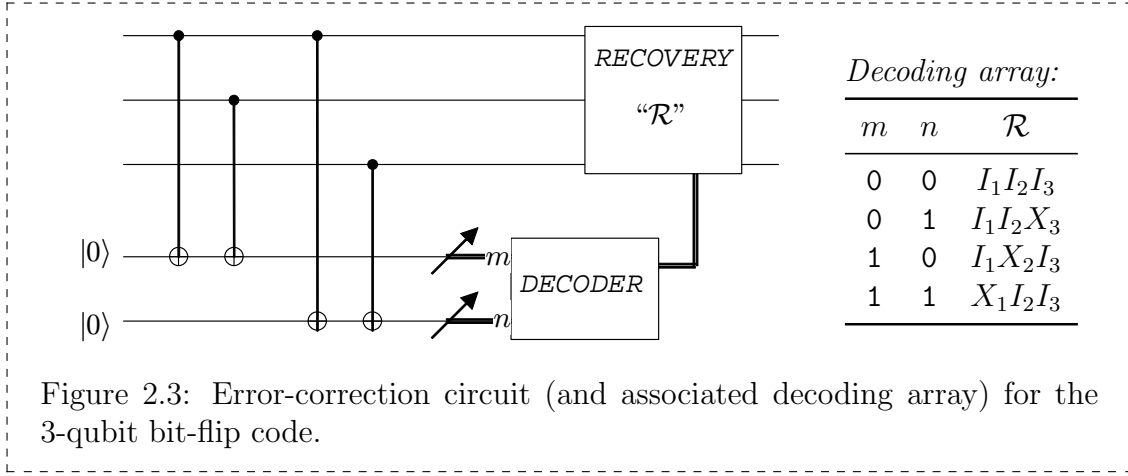
$$\alpha|0\rangle + \beta|1\rangle \xrightarrow{\text{bit-flip encoding}} \alpha|000\rangle + \beta|111\rangle. \quad (2.22)$$

The encoding process has the effect of replacing each of the computational basis states by its repetition-encoded version. As we'll see, this connection to the classical repetition code is especially important when it comes to performing error correction on the bit-flip code. A circuit to encode a qubit in the bit-flip code is shown in Figure 2.2.

It becomes quite clear that the code cannot handle  $Z$  errors, by considering that a  $Z$  error applied to any of the three qubits of the code state  $\alpha|000\rangle + \beta|111\rangle$  will produce another valid code state,  $\alpha|000\rangle - \beta|111\rangle$ , that corresponds to a different (incorrect) message.

Let's consider how the code deals with  $X$  errors. The most direct way to find which qubit is affected by an  $X$  error is to measure each of the three qubits in the computational basis. Then, for example, if the second qubit had been affected by an  $X$  error, resulting in the state  $\alpha|010\rangle + \beta|101\rangle$ , then the measured bit string would be either 010 or 101. This string could then be treated as though it were an ordinary repetition-encoded classical message, by using the majority-voting decoding method. This would correctly determine that the second qubit had indeed suffered a bit-flip. The problem with this procedure is that it not only measures the errors, but also the message itself (thus destroying the message). So this method of correction would only be useful in a circumstance where the message qubit was intended to be measured anyway, such as at the final step of a quantum computation.

More generally though, it is necessary that the measurements to determine the location of errors are performed *indirectly* on the code state, in a way that does not disrupt the



actual message. A circuit which achieves this is shown in Figure 2.3. In this circuit, the first three wires denote the received encoded message state. The circuit takes two *ancilla* qubits, initialized to the  $|0\rangle$  state, and interacts these with the encoded state via a specific series of four controlled-not gates. The ancilla qubits are then measured in the computational basis, and the results are used to determine what recovery operation should be performed on the encoded state.

To see how this works, first imagine that the input to the circuit in Figure 2.3 is not a code state, but simply a computational basis state,  $|y_1 y_2 y_3\rangle$ , for some bits  $y_1$ ,  $y_2$ , and  $y_3$ . Then, each controlled-not just adds the bit value at its control to the value at its target, and so when the two measurements are performed in Figure 2.3, the results will be  $m = y_1 + y_2$  and  $n = y_1 + y_3$ , and the remaining state on the first three qubits will be unchanged at  $|y_1 y_2 y_3\rangle$ . Now, instead imagine that the state input to the circuit is a bit-flip-encoded message, but modified by  $X$  errors, as follows:

$$|\psi\rangle = X^{e_1} \otimes X^{e_2} \otimes X^{e_3} (\alpha|000\rangle + \beta|111\rangle), \quad (2.23)$$

where  $(e_1, e_2, e_3)$  are three bits used to describe the pattern of  $X$  errors present on the input state. So now, the input state is a superposition of two different computational basis states. However, it's easy to check that each of the two states in this superposition give the same measurement outputs  $(m, n)$ , equal to the following linear combinations of the bits in the error pattern:

$$m = e_1 + e_2, \quad (2.24)$$

$$n = e_1 + e_3. \quad (2.25)$$

So, no information about the actual message is gained from the measurements  $m$  and  $n$ ,

and so after the two measurements are performed the state of the first three qubits will be preserved in the superposition described by Eq. (2.23).

To understand this in a slightly different way, notice that the measurement results  $(m, n)$  correspond to the *syndrome* – with respect to the 3-bit classical repetition code – of the error pattern  $(e_1, e_2, e_3)$ . In the case where some arbitrary computational basis state  $|y_1 y_2 y_3\rangle$  is input to the error-correction circuit in Figure 2.3, the results  $(m, n) = (y_1 + y_2, y_1 + y_3)$  equal the syndrome of the bit string  $(y_1, y_2, y_3)$ . Importantly, recall from Subsection 2.1.3 that the syndrome has the property that it contains all the available information about the errors present in the noisy classically-encoded string  $(y_1, y_2, y_3)$  but discards all information about the original message. As a result, the two terms in Eq. (2.23) share the same syndrome, which is the crucial factor that avoids having the superposition collapse when the measurements  $m$  and  $n$  are performed. So, whereas in classical error-correction finding the syndrome is a matter of convenience in simplifying the process of decoding, its use in the quantum bit-flip code is of a much more central importance.

After the two syndrome bits are measured in Figure 2.3, the most likely value of the error pattern  $(e_1, e_2, e_3)$  can be inferred using the method of syndrome decoding described in Subsection 2.1.3. The appropriate recovery operation,  $\mathcal{R}$ , then consists of an  $X$  operation applied to any qubit which is believed to have suffered an error. A decoding array, which details how  $\mathcal{R}$  is chosen for each syndrome  $(m, n)$ , is shown in Figure 2.3. It is easy to verify that this correction scheme can successfully correct any single  $X$  error on the input encoded state.

## 2 – Phase-flip code

If we take the definition of a bit-flip code, and simply interchange the roles of the  $X$  and  $Z$  bases (that is, interchange the roles of the single-qubit basis states  $|0\rangle \leftrightarrow |+\rangle$  and  $|1\rangle \leftrightarrow |-\rangle$ , and interchange the roles of the Pauli operators  $X \leftrightarrow Z$ ), we arrive at the definition of the *phase-flip* code. A qubit encoded in the 3-qubit version of the phase flip code is written as follows:

$$\alpha|0\rangle + \beta|1\rangle \xrightarrow{\text{phase-flip encoding}} \alpha|+++ \rangle + \beta|--- \rangle. \quad (2.26)$$

This code can correct a single phase-flip ( $Z$  error) but not  $X$  or  $Y$  errors. The encoding circuit is shown in Figure 2.4. It consists of the circuit for the bit-flip code, followed by a Hadamard gate  $H$  applied to each qubit (since  $H$  makes the appropriate transformations between  $X$  and  $Z$  bases,  $H|0\rangle = |+\rangle$  and  $H|1\rangle = |-\rangle$ ).

The correction circuit for the phase-flip code is not shown, but it is also a simple

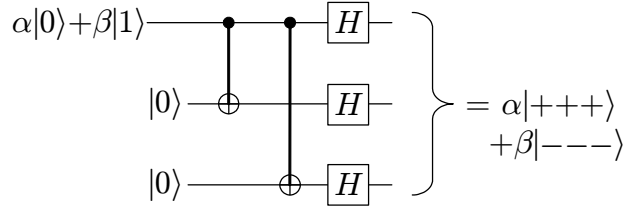


Figure 2.4: Encoding circuit for the 3-qubit phase-flip code.

modification from the bit-flip version of the circuit. Correction for the phase-flip code can be achieved by first applying  $H$  gates to the three qubits (which converts the phase-flip code to a bit-flip code and any  $Z$  errors to  $X$  errors), applying the bit-flip correction circuit, and then applying  $H$  gates again to convert back to the phase-flip code.

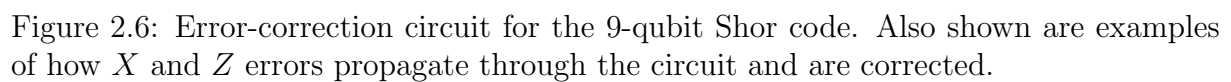
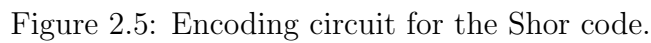
### 3 – Shor code

Given that the bit-flip code can correct a single  $X$  error (but cannot correct  $Z$  errors), and the phase-flip code can correct a single  $Z$  error (but cannot correct  $X$  errors), an obvious question to ask is, is there a way to combine these codes together so that both  $X$  and  $Z$  errors can be corrected? In fact, using a technique known as *code concatenation*, the bit-flip and phase-flip codes can be combined to form the *Shor code*, which corrects a single error of any type ( $X$ ,  $Y$ , or  $Z$ ).

The idea behind code concatenation is quite simple. Say we have two quantum codes,  $C_1$  and  $C_2$ , that encode a one-qubit message into  $N_1$  and  $N_2$  qubits respectively. The code which is the concatenation of these two codes also encodes one qubit, and is defined to have the following two-stage encoding circuit. First, the message is encoded using the code  $C_1$ , then each of the resulting  $N_1$  qubits in the encoded state are themselves individually encoded using the code  $C_2$ . Thus, the concatenated code has a size  $N_1 N_2$ . The encoding circuit for the Shor code, which is a concatenation of a phase-flip code with a bit-flip code, is given in Figure 2.5.

Consider the ability of the Shor code to correct the three types of Pauli errors, via the error-correction circuit shown in Figure 2.6. First, say that one of the nine qubits has suffered an  $X$  error. Since the Shor code consists of three blocks that each contain a bit-flip encoded state, one can simply apply the bit-flip correction circuit to each of these blocks, and the  $X$  error will be eliminated.

Now, assume instead that the original error had been a  $Z$  operation. This error will pass unaffected through the bit-flip correction circuits. Imagine then that after the bit-flip correction circuits, there is a un-encoding circuit applied to each bit-flip code block. The un-encoding circuit is simply the inverse of the encoding circuit, in this case for the bit-flip



code. A  $Z$  error input to one of the un-encoding circuits will pass on to the output. So, after the un-encoding circuits have been applied, the remaining three qubits will be in a phase-flip code, but subject to a single  $Z$  error. The error can then be eliminated by a final phase-flip error-correction circuit. (The corrected output state could then, if desired, be re-encoded into the Shor code, or completely unencoded to a single qubit.)

A  $Y$  error is equivalent to  $X$  and  $Z$  errors applied coincidentally to the same qubit, and for this reason it can be shown that a single  $Y$  error will also be corrected by the Shor code. So, from the results in Subsection 2.2.1, it follows that the Shor code can correct an arbitrary error on any single qubit, and can also be used to suppress any type of weak independent noise. Note that, by starting with larger versions of the phase-flip and bit-flip codes, one can construct larger versions of the Shor code able to correct more errors. In particular, for each odd  $n \geq 3$  there exists an  $n^2$ -qubit version of the Shor code that corrects  $\frac{1}{2}(n-1)$  errors.

#### 4 – Degenerate codes

The Shor code belongs to a special class of quantum codes known as *degenerate* codes. The interesting thing about degenerate codes is that the property they share has no obvious analogue in classical error-correction. If a quantum code is degenerate, then there exist two different patterns of Pauli errors that are both correctable by the code, but are indistinguishable by the receiver. Let me explain with an example. Consider again the error-correction circuit for the Shor code in Figure 2.6. Compare the following two cases: (1) a  $Z$  error has been applied to just the first input qubit, and (2) a  $Z$  error has been applied to just the second input qubit. In either case the error will pass through the bit-flip correction block, and end up as a  $Z$  error on the first input to the phase-flip correction block. Thus, the phase-flip correction block will measure the same syndrome in either case, and the end result will be that the receiver will have corrected the initial  $Z$  error without actually knowing which of the two cases it was.

The fact that the receiver cannot distinguish the two errors is not merely a result of the way the correction circuit is arranged, but is rather a fundamental property of the Shor code. It can be shown that if  $|\psi\rangle_L$  is a state belonging to the Shor code, then the following identity holds:  $Z_1|\psi\rangle_L = Z_2|\psi\rangle_L$ , where  $Z_1$  and  $Z_2$  represent the  $Z$  operator acting on the first and second qubits respectively. This identity means that it would be *impossible* for the receiver to distinguish the two errors  $Z_1$  and  $Z_2$  via any means.

In contrast, a *nondegenerate* code that corrects  $t$  errors has the property that all the possible patterns of up to  $t$  Pauli errors are distinguishable from one another. That is, if you take a state belonging to a nondegenerate code, and consider the set of different states



that would result from applying each of the possible patterns of up to  $t$  Pauli errors, then every state in that set would be orthogonal to all the others. Examples of nondegenerate codes include the CSS codes described in the next section.

### 2.2.3 CSS codes

Like the Shor code, which is notable for its connection to the classical repetition code, CSS codes also have a close connection to classical codes – in this case to a more sophisticated class of block linear codes. The way CSS codes are constructed makes them generally more efficient (in terms of rate, and number of errors corrected) than Shor codes. CSS codes are thought to be particularly well suited for use in quantum computing, since it's relatively easy to perform logical gates directly on a CSS-encoded qubit without needing to unencode it first (this is an important part of making a quantum computer *fault-tolerant*).

Note that for the sake of simplicity, my description of CSS codes will not be entirely general. I restrict the discussion to versions of the code that encode a single qubit in each block. Also, whereas normally the construction of any particular CSS code requires two separate classical linear codes to be specified, the construction I describe requires just one linear code to be specified, by implicitly assuming that the second linear code is always the *dual* of the other. The class of CSS codes that satisfy the two points mentioned above still encompasses many of the more well-known examples, such as the Steane 7-qubit code and the Golay 23-qubit code, which are the two codes I make use of in the work described in Chapter 6.

#### 1 – Defining the CSS code states

Let  $C$  be an  $N$ -bit classical linear code that encodes  $k$  bits and corrects  $t$  errors. Assume that  $C$  is chosen to satisfy the following two properties: first, that

$$k = \frac{N + 1}{2}, \quad (2.27)$$

and second, that

$$C^\perp \subset C. \quad (2.28)$$

In Eq. (2.28),  $C^\perp$  denotes the dual, as defined in Subsection 2.1.3, of the code  $C$ . So, Eq. (2.28) requires that all codewords in the dual of  $C$  also belong to  $C$ . Such a code is said to be *weakly self-dual*.

Then, the code  $C$  can be used to construct a CSS quantum code that encodes one qubit into  $N$  qubits and corrects  $t$  errors, as follows. In a CSS code, the encoded  $|0\rangle$

message state consists of an equal superposition of all codewords in  $C^\perp$ . That is,

$$|0\rangle_L = \frac{1}{\sqrt{|C^\perp|}} \sum_{y \in C^\perp} |y\rangle, \quad (2.29)$$

where  $|y\rangle$  is the  $N$ -bit computational basis state corresponding to the codeword  $y$ , and where  $|C^\perp| = 2^{\frac{N-1}{2}}$  is the number of codewords in  $C^\perp$ . The encoded  $|1\rangle$  message state is defined to be the state that results from applying the  $X$  operator to every qubit in the state  $|0\rangle_L$ . That is,

$$|1\rangle_L = X^{\otimes N} |0\rangle_L \quad (2.30)$$

$$= \frac{1}{\sqrt{|C^\perp|}} \sum_{y \in C^\perp} |\bar{y}\rangle, \quad (2.31)$$

where  $\bar{y}$  is defined to be the bitwise complement of the codeword  $y$  (that is,  $\overline{0001} = 1110$ , etc.).

An example linear code  $C$  that satisfies the conditions mentioned above is the 7-bit Hamming code, whose generator matrix is given in Eq. (2.9). The parity check matrix for the 7-bit Hamming code is

$$H_{(\text{Hamming7})} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}. \quad (2.32)$$

Recall that the set of codewords of the dual code  $C^\perp$  is given by the row-space of the code's parity check matrix. So, from Eqs. (2.29) and (2.30), the corresponding CSS code has basis states  $|0\rangle_L$  and  $|1\rangle_L$  equal to

$$\begin{aligned} |0\rangle_L = \frac{1}{\sqrt{8}} & \left[ |0000000\rangle + |1110100\rangle + |1101010\rangle + |0011110\rangle \right. \\ & \left. + |1011001\rangle + |0101101\rangle + |0110011\rangle + |1000111\rangle \right] \end{aligned} \quad (2.33)$$

and

$$\begin{aligned} |1\rangle_L = \frac{1}{\sqrt{8}} & \left[ |1111111\rangle + |0001011\rangle + |0010101\rangle + |1100001\rangle \right. \\ & \left. + |0100110\rangle + |1010010\rangle + |1001100\rangle + |0111000\rangle \right]. \end{aligned} \quad (2.34)$$

The quantum code defined by Eqs. (2.33) and (2.34) is known as the *Steane code*, after its inventor Andrew Steane [Ste96].

Note that the set of eight computational basis states that appear in Eq. (2.33) are disjoint from the set that appear in Eq. (2.34). The easy way to check this is to see that

all the computational basis states in Eq. (2.33) have an even Hamming weight, whereas all those in Eq. (2.34) have an odd Hamming weight. So,  $|0\rangle_L$  and  $|1\rangle_L$  are orthogonal to each other as required.

In fact, the same is true for *any* code  $C$  satisfying the conditions stated earlier in this subsection; that is the set of codewords  $y \in C^\perp$  all have even Hamming weight, and the bit-wise complement,  $\bar{y}$ , of these codewords all have odd Hamming weight. To prove this, note that the condition  $C^\perp \subset C$  implies that if  $y \in C^\perp$  then  $y \cdot y = 0$ . Now,  $y \cdot y = 0$  if and only if  $y$  has even Hamming weight, therefore any  $y \in C^\perp$  has even Hamming weight. Now, since the length  $N$  of the CSS code is odd (which is a consequence of Eq. (2.27)), then it follows that if  $y \in C^\perp$  then  $\bar{y}$  has an *odd* Hamming weight. These properties of evenness and oddness for  $y$  and  $\bar{y}$  respectively, when  $y \in C^\perp$ , will prove to be particularly useful during our later discussion on performing logical operations on encoded CSS qubits.

There is a related fact about the set of codewords in  $C^\perp$  that is worth mentioning at this point, namely that the following relation holds:

$$\overline{C^\perp} \cap C^\perp = C. \quad (2.35)$$

In the above,  $\overline{C^\perp}$  is used to denote the set containing the bitwise-complement of each of the codewords in  $C^\perp$ . So, for example, a consequence of Eq. (2.35) is that the set of all sixteen different bit-strings that appear in either Eq. (2.33) or Eq. (2.34), equals the set of codewords of the 7-bit Hamming code. Also, a general consequence is that the logical plus state,  $|+\rangle_L = \frac{1}{\sqrt{2}}(|0\rangle_L + |1\rangle_L)$ , may be written as an equal superposition of all codewords in  $C$ :

$$|+\rangle_L = \frac{1}{\sqrt{|C|}} \sum_{y \in C} |y\rangle. \quad (2.36)$$

Eq. (2.35) can be proven by first showing that  $\overline{C^\perp} \cap C^\perp \subset C$  and then showing that  $|\overline{C^\perp} \cap C^\perp| = |C|$ . To show the former, note that we need only show that  $\overline{C^\perp} \subset C$ , since we have already assumed that  $C^\perp \subset C$ . Now, from the definition of a dual code code, we have that

$$x \in C \iff x \cdot y = 0, \quad \forall y \in C^\perp. \quad (2.37)$$

So, consider an  $x \in \overline{C^\perp}$ , which can always be written in the form  $x = y' + (111 \dots 11)$  for some  $y' \in C^\perp$ . Then for any  $y \in C^\perp$  we have

$$x \cdot y = y' \cdot y + (111 \dots 11) \cdot y \quad (2.38)$$

$$= 0 + 0, \quad (2.39)$$

and hence  $x \in C$ . (In the above,  $y \cdot y' = 0$  due to the condition  $C^\perp \subset C$ , and  $(111 \dots 11) \cdot y = 0$  due to  $y$  having even Hamming weight). Hence,  $\overline{C^\perp} \subset C$ . Now, to see that

$|\overline{C^\perp} \cap C^\perp| = |C|$ , note that

$$\overline{C^\perp} \cap C^\perp = \text{span}\{C^\perp \cap 111 \dots 11\}, \quad (2.40)$$

and thus  $\overline{C^\perp} \cap C^\perp$  is a vector space having dimension one greater than the dimension of  $C^\perp$ . But the dimension of  $C^\perp$  and  $C$  respectively are  $\frac{N+1}{2} - 1$  and  $\frac{N+1}{2}$ . Thus, the spaces  $\overline{C^\perp} \cap C^\perp$  and  $C$  each have the same dimension, and so  $|\overline{C^\perp} \cap C^\perp| = |C|$ , and hence we arrive at Eq. (2.35).

## 2 – Circuits for creating CSS code states

There exist relatively simple quantum circuits for creating CSS code states. Let's first consider a circuit for creating the  $|0\rangle_L$  state. Now,  $|0\rangle_L$  is a superposition over all elements of the row-space of the parity check matrix  $H$  (for the code  $C$ ), and so  $|0\rangle_L$  can be written in the following form:

$$|0\rangle_L = \frac{1}{\sqrt{|C^\perp|}} \sum_{b \in \mathbb{Z}_2^{N-k}} |b_1 H_1 + \dots + b_{N-k} H_{N-k}\rangle, \quad (2.41)$$

where  $H_j$  denotes the  $j$ -th row of  $H$ , and  $b_j$  denotes the  $j$ -th element of the vector  $b$ . Recall from Subsection 2.1.3 that a parity check matrix can always be assumed to be in standard form; that is  $H = [A; I_{N-k}]$  for some  $(N-k) \times k$  matrix  $A$ , and where  $I_{N-k}$  is the  $(N-k) \times (N-k)$  identity matrix. (Note that the parity check matrix in Eq. (2.32) is in standard form). This allows Eq. (2.41) to be re-written as follows:

$$|0\rangle_L = \frac{1}{\sqrt{|C^\perp|}} \sum_{b \in \mathbb{Z}_2^{N-k}} |b_1 A_1 + \dots + b_{N-k} A_{N-k}\rangle |b\rangle, \quad (2.42)$$

where  $A_j$  denotes the  $j$ -th row of  $A$ . Now, it can be shown that a state of the form  $|b_1 A_1 + \dots + b_{N-k} A_{N-k}\rangle |b\rangle$  can be obtained by applying an appropriate series of controlled-not gates to the state  $|0\rangle^{\otimes k} |b\rangle$ , as follows:

$$|b_1 A_1 + \dots + b_{N-k} A_{N-k}\rangle |b\rangle = \left( \prod_{(m,n): A_{mn}=1} \text{CNOT}_{(k+m) \rightarrow (n)} \right) |0\rangle^{\otimes k} |b\rangle, \quad (2.43)$$

where  $\text{CNOT}_{(k+m) \rightarrow (n)}$  denotes a controlled-not gate applied to control qubit  $k+m$  and target qubit  $n$ . That is, for each entry in  $A$  that has a value 1, Eq. (2.43) applies a corresponding controlled-not gate between a pair of qubits determined by the row and column coordinates of that entry.

Thus, we can write Eq. (2.42) as

$$|0\rangle_L = \left( \prod_{(m,n): A_{mn}=1} \text{CNOT}_{(k+m) \rightarrow (n)} \right) \frac{1}{\sqrt{|C^\perp|}} \sum_{b \in \mathbb{Z}_2^{N-k}} |0\rangle^{\otimes k} |b\rangle, \quad (2.44)$$

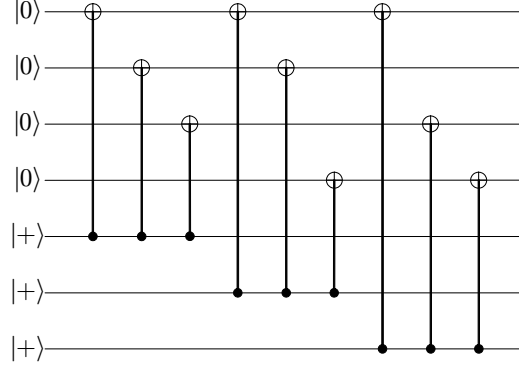


Figure 2.7: A circuit for creating the  $|0\rangle_L$  basis state as specified in Eq. (2.33) for the Steane 7-qubit code.

and since  $\frac{1}{\sqrt{|C^\perp|}} \sum_{b \in \mathbb{Z}_2^{N-k}} |b\rangle = |+\rangle^{\otimes(N-k)}$ , we finally obtain the following expression for  $|0\rangle_L$ ,

$$|0\rangle_L = \left( \prod_{(m,n): A_{mn}=1} \text{CNOT}_{(k+m) \rightarrow (n)} \right) |0\rangle^{\otimes k} |+\rangle^{\otimes(N-k)}, \quad (2.45)$$

which specifies a circuit composed of controlled-not gates and single-qubit state preparations, for creating the state  $|0\rangle_L$ . For example, in the case of the 7-qubit Steane code, this circuit is drawn in Figure 2.7. (Note that the ordering of the controlled-not gates does not matter, since it can be shown that two controlled-not gates commute whenever it is the case that the control of one is not applied to the same qubit as the target of the other).

Using a fairly straightforward extension to the circuit for creating  $|0\rangle_L$ , arbitrary qubit states  $\alpha|0\rangle + \beta|1\rangle$  can be encoded, as shown in Figure 2.8. In this circuit, the box containing “ $|0\rangle_L$ ” denotes the circuit for creating  $|0\rangle_L$ . A series of controlled-not gates are applied between the message state and each qubit in the  $|0\rangle_L$  code state. It can be shown that after these controlled-not gates have been applied, the state of the  $N + 1$  qubits is

$$|+\rangle(\alpha|0\rangle_L + \beta|1\rangle_L) + |-\rangle(\alpha|0\rangle_L - \beta|1\rangle_L). \quad (2.46)$$

So, when the first qubit is measured in the  $X$  basis, the state of the remaining qubits will collapse to either the desired state  $\alpha|0\rangle_L + \beta|1\rangle_L$ , or the state  $\alpha|0\rangle_L - \beta|1\rangle_L$ , depending on whether the measurement result  $m$  is 0 or 1 respectively. If  $m = 1$ , then the circuit applies a  $Z$  operation to each of the  $N$  qubits. It turns out that  $Z^{\otimes N}(\alpha|0\rangle_L - \beta|1\rangle_L) = \alpha|0\rangle_L + \beta|1\rangle_L$  (the reason for this will be made clear in the next subsection), and thus the circuit will then correctly give the state  $\alpha|0\rangle_L + \beta|1\rangle_L$  for either value of  $m$ .

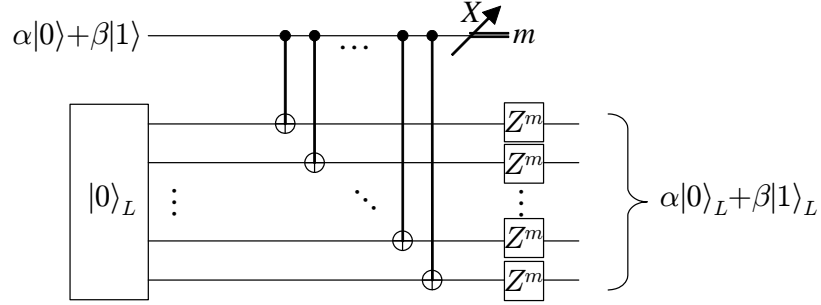


Figure 2.8: A general circuit for encoding an arbitrary qubit state in the CSS code.

### 3 – Logical operations on CSS-encoded qubits

Recall that I defined the encoded CSS state  $|1\rangle_L$  to be the result of applying  $X^{\otimes N}$  to  $|0\rangle_L$ . An important consequence of this definition is as follows: the effect that  $X^{\otimes N}$  has on an encoded qubit  $\alpha|0\rangle_L + \beta|1\rangle_L$  is equivalent to the effect that the  $X$  operation has on an unencoded qubit. That is,

$$X(\alpha|0\rangle + \beta|1\rangle) = \alpha|1\rangle + \beta|0\rangle \quad \text{and} \quad (2.47)$$

$$X^{\otimes N}(\alpha|0\rangle_L + \beta|1\rangle_L) = \alpha|1\rangle_L + \beta|0\rangle_L. \quad (2.48)$$

In other words,  $X^{\otimes N}$  behaves like an encoded version of the gate  $X$ . Accordingly, we refer to  $X^{\otimes N}$  as the “logical  $X$  gate” (or “encoded  $X$  gate”) for a CSS code, and write

$$X_L = X^{\otimes N}. \quad (2.49)$$

Eq. (2.49) is an example of where an encoded gate is implemented *transversally*. This refers to the fact that the encoded version of the  $X$  gate simply consists of the unencoded version of the gate applied separately to each qubit in the code. Note that usually, encoded gates cannot be implemented transversally, and instead most gates need to be implemented in a way that is significantly more complicated in comparison. Luckily however, it turns out that many of the most important gates have a transversal implementation when encoded with a CSS code. It turns out that, in addition to the  $X$  gate as we saw above, the CSS-encoded versions of all the following gates are transversal:  $Y$ ,  $Z$ , Hadamard, controlled-not and controlled-phase. I give the proofs below, for the case of the  $Y$  and  $Z$  gates. The proofs for the other cases follow similar lines, but are slightly more involved.

First consider the  $Z$  gate acting transversally on each of the two encoded basis states

$|0\rangle_L$  and  $|1\rangle_L$ . Now,

$$Z^{\otimes N}|0\rangle_L = \frac{1}{\sqrt{|C^\perp|}} \sum_{y \in C^\perp} (-1)^{\text{wt}(y)} |y\rangle \quad (2.50)$$

$$= \frac{1}{\sqrt{|C^\perp|}} \sum_{y \in C^\perp} |y\rangle \quad (2.51)$$

$$= |0\rangle_L, \quad (2.52)$$

where we used that  $(-1)^{\text{wt}(y)} = 1$ , since each  $y \in C^\perp$  has even Hamming weight. Similarly

$$Z^{\otimes N}|1\rangle_L = \frac{1}{\sqrt{|C^\perp|}} \sum_{y \in C^\perp} (-1)^{\text{wt}(\bar{y})} |\bar{y}\rangle \quad (2.53)$$

$$= -|1\rangle_L, \quad (2.54)$$

since each  $\bar{y}$  has odd Hamming weight. Thus,  $Z^{\otimes N}$  acts in the correct way on each encoded basis state, so we conclude that

$$Z_L = Z^{\otimes N}. \quad (2.55)$$

To prove that the encoded  $Y$  gate operates transversally, simply multiply together Eq. (2.49) and Eq. (2.55), and substitute  $XZ = -iY$  and  $X_L Z_L = -iY_L$ , to give

$$Y_L = i(-i)^N Y^{\otimes N}. \quad (2.56)$$

An example of a gate that can *not* be implemented transversally on CSS-encoded qubits is the so-called  $\pi/8$  gate, that is the gate corresponding to the single-qubit unitary operation  $|0\rangle\langle 0| + e^{i\pi/4}|1\rangle\langle 1|$ . The details of how such a gate is implemented can be found, for example, in Section 10.6.2 of [NC00]. I will not reproduce those details here though, but it is worth noting that the implementation of the  $\pi/8$  gate described in [NC00] manages to share one of the features that transversal gates have: it is *fault-tolerant*, so, like a transversal gate such as  $X^{\otimes N}$ , it will not cause existing errors to spread to other qubits in a code block.

It is important to be able to perform encoded *measurement* operations. Encoded versions of  $X$ ,  $Y$ , and  $Z$ -basis measurements all have what is essentially a transversal implementation. Consider, for example, a measurement in the  $Z_L$  basis, on an arbitrary encoded state  $\alpha|0\rangle_L + \beta|1\rangle_L$ . Imagine that each of the  $N$  qubits in this state are measured in the  $Z$ -basis, and that  $y$  denotes the vector of measurement results. From the discussion in Subsection 2.2.3.1, it is clear that  $y$  will equal one of the codewords in the code  $C$ . With probability  $|\alpha|^2$ ,  $y$  will have an even Hamming weight, in which case the measurement result should be interpreted as  $|0\rangle_L$ . Similarly, with probability  $|\beta|^2$ ,  $y$  will have an odd Hamming weight, in which case the measurement result should be interpreted as  $|1\rangle_L$ .

If the state being measured is noisy, then error-correction will need to be performed during the measurement procedure. This is simply a matter of performing classical error-correction on the measured bit-string  $y$ , with respect to the code  $C$ , before performing the test for evenness/oddness as described above. Recall that we assumed the code  $C$  corrects  $t$  errors. So, assuming that no more than  $t$  of the qubits being measured had errors, then no more than  $t$  of the resulting measurement bits in  $y$  will have errors, in which case performing classical error-correction on  $y$  will give the correct codeword of  $C$  (thus, importantly, having the correct Hamming weight), leading to an error free outcome for the logical measurement.

#### 4 – Error-correction circuits for CSS codes

I now describe a general procedure for performing error-correction on a CSS-encoded qubit.

Begin by noting that any of the four single-qubit Pauli operators  $I$ ,  $X$ ,  $Y$ , or  $Z$ , can be written in the form  $X^x Z^z$  for some appropriate choice of bit values  $x$  and  $z$ . (For example,  $X^0 Z^0 = I$ ,  $X^1 Z^1 = X$ ,  $X^0 Z^1 = Z$ , and  $X^1 Z^1 = -iY$ . The unimportant factor of  $-i$  is ignored in the case of  $-iY$ ). This allows us to write a CSS code-state that has been affected by an arbitrary pattern of Pauli errors, as follows:

$$|\psi\rangle = (X^{x_1} \otimes X^{x_2} \otimes \cdots \otimes X^{x_N})(Z^{z_1} \otimes Z^{z_2} \otimes \cdots \otimes Z^{z_N})(\alpha|0\rangle_L + \beta|1\rangle_L), \quad (2.57)$$

for some  $x, z \in \mathbb{Z}_2^N$ .

Consider an error-correction procedure performed on the state in Eq. (2.57), composed of the following broad steps:

1.  **$X$ -syndrome extraction.** A circuit which measures the “ $X$  syndrome” is applied. The  $X$  syndrome is defined to be  $s_x \equiv Hx$ , where  $H$  is the parity check matrix of the code  $C$  and where  $x$  is the vector that describes the  $X$  errors on the input state in Eq. (2.57).
2.  **$Z$ -syndrome extraction.** Similarly, a circuit which measures the “ $Z$  syndrome” ( $s_z \equiv Hz$ ) is applied.
3. **Maximum-likelihood decoding.** From the syndromes  $s_x$  and  $s_z$ , the most-likely values for the error patterns  $x$  and  $z$  are computed.
4. **Recovery.** The recovery operation  $(X^{\tilde{x}_1} \otimes \cdots \otimes X^{\tilde{x}_N})(Z^{\tilde{z}_1} \otimes \cdots \otimes Z^{\tilde{z}_N})$  is applied to the state undergoing correction, where  $\tilde{x}$  and  $\tilde{z}$  denote the most-likely error patterns as computed in step 3.



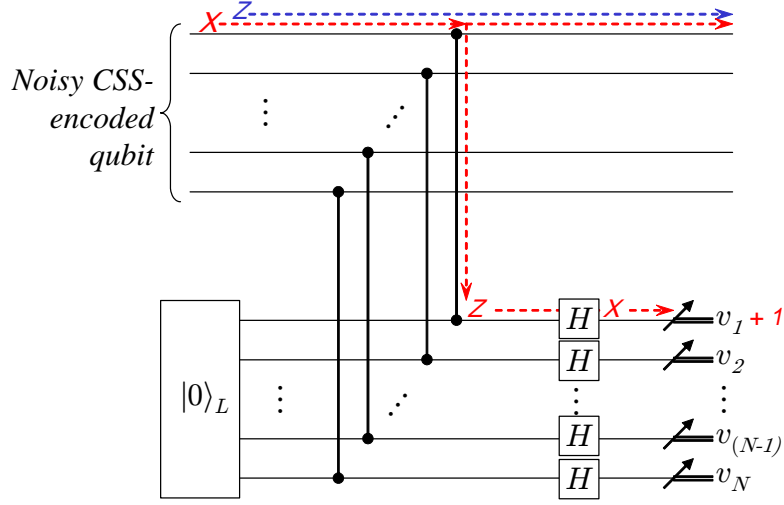


Figure 2.9: A circuit for measuring the  $X$  syndrome of a noisy CSS-encoded qubit. The box labelled  $|0\rangle_L$  denotes a circuit for creating the state  $|0\rangle_L$ . Also shown is an example of how  $X$  and  $Z$  errors propagate through the circuit.

In the first stage above,  $X$ -syndrome extraction is performed using the circuit shown in Figure 2.9. To understand how this circuit works, consider first what happens when the input is a noise-free CSS code state,  $\alpha|0\rangle_L + \beta|1\rangle_L$ . Notice that Figure 2.9 consists entirely of a series of *encoded* operations. That is, the circuit prepares an encoded  $|0\rangle_L$  state, then performs an encoded controlled-phase gate between that state and the input encoded qubit, then performs an encoded Hadamard gate, followed finally by an encoded  $Z$ -basis measurement. So, the effect that the circuit has on the message state can be understood in terms of the unencoded version of the circuit, as follows:

$$\begin{array}{c}
 \alpha|0\rangle + \beta|1\rangle \text{ --- } \bullet \text{ ---} \\
 |0\rangle \text{ --- } \bullet \text{ --- } [H] \text{ --- } \text{meter}
 \end{array} \quad (2.58)$$

It's easy to show that after the controlled-phase and Hadamard gates in Eq. (2.58) are applied, the first qubit remains unchanged in the state  $\alpha|0\rangle + \beta|1\rangle$ , and the second qubit has the state  $|+\rangle$ . (The computational-basis measurement that is then performed on the  $|+\rangle$  state will just yield 0 or 1 randomly).

Thus, when  $\alpha|0\rangle_L + \beta|0\rangle_L$  is input to the circuit in Figure 2.9, the output will equal  $\alpha|0\rangle_L + \beta|0\rangle_L$ , and the vector  $v$  of measurement results will randomly take one of the  $2^k$  codewords of  $C$ , that is  $v = Gw$  for some random  $w \in \mathbb{Z}_2^k$ .

Now, say that the input state contains Pauli errors; that is, the input is given by the

state  $|\psi\rangle$  in Eq. (2.57). Then the above argument can be modified to show that, again, the output state will equal the input state,  $|\psi\rangle$ . However, the vector  $v$  will in this case equal

$$v = Gw + x, \quad (2.59)$$

for a random  $w \in \mathbb{Z}_2^k$ , and where  $x$  is the input pattern of  $X$  errors. This is a result of the fact that the  $X$  errors in the input state *propagate* through the controlled-phase gates, Hadamard gates, and measurements, according to the following three commutation relations:

$$\begin{array}{c} \boxed{X} \\ \text{---} \bullet \text{---} \\ | \\ \text{---} \bullet \text{---} \end{array} = \begin{array}{c} \text{---} \bullet \text{---} \boxed{X} \\ | \\ \text{---} \bullet \text{---} \boxed{Z} \end{array} \quad (2.60)$$

$$\boxed{Z} \boxed{H} = \boxed{H} \boxed{X} \quad (2.61)$$

$$\text{---} \boxed{X} \text{---} \nearrow m = \text{---} \nearrow (m+1) \quad (2.62)$$

thus causing the input error pattern  $x$  to influence the measurement results, according to Eq. (2.59). Due to the random term  $Gw$  in Eq. (2.59), we can't directly determine the pattern  $x$  from the measurements  $v$ . However, if we apply the parity check matrix  $H$  to  $v$ , we get

$$Hv = HGw + Hx \quad (2.63)$$

$$= Hx, \quad (2.64)$$

(since  $HG = 0$ ), thus providing  $s_x \equiv Hx$ , the syndrome for the  $X$  errors.

Similarly, the circuit for performing  $Z$ -syndrome extraction is shown in Figure 2.10. It is almost identical to the circuit for  $X$ -syndrome extraction, except that the controlled-phase gates are replaced by controlled-not gates. In this case, the controlled-not gates have the desired effect of propagating  $Z$  errors downwards, via the following commutation relation:

$$\begin{array}{c} \boxed{Z} \oplus \text{---} \\ | \\ \text{---} \bullet \text{---} \end{array} = \begin{array}{c} \oplus \text{---} \boxed{Z} \\ | \\ \text{---} \bullet \text{---} \boxed{Z} \end{array} \quad (2.65)$$

As a result, applying  $H$  to the vector  $u$  of measurement results in Figure 2.10 will give the  $Z$ -syndrome:

$$Hu = Hz. \quad (2.66)$$

In the next stage of the error-correction procedure, the syndromes  $s_x$  and  $s_z$  are used to find  $\tilde{x}$  and  $\tilde{z}$ , the most-likely values for the  $X$  and  $Z$  error patterns. The precise procedure for performing this maximum-likelihood decoding will depend on the assumed noise

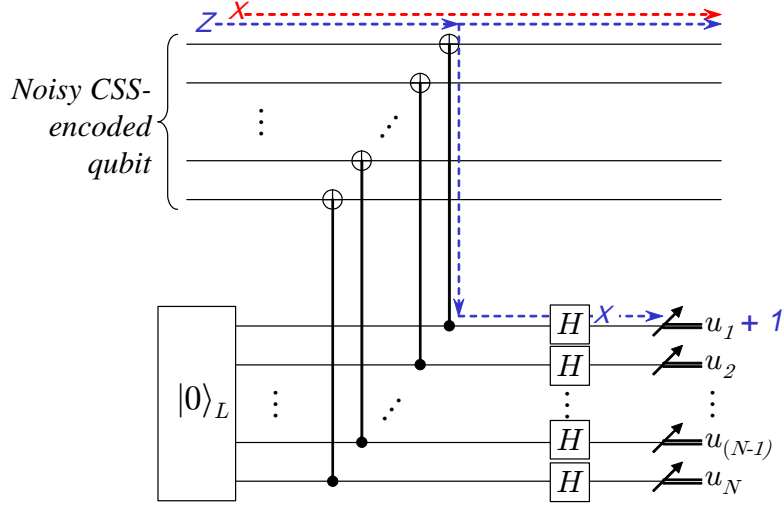


Figure 2.10: A circuit for measuring the  $Z$  syndrome of a noisy CSS-encoded qubit. Also shown is an example of how  $X$  and  $Z$  errors propagate through the circuit.

model. For example, say we assume that the state has been affected by the independent depolarizing channel. (Recall, this is where each qubit is independently depolarized with some probability  $p$ ). Then, the most-likely  $X$ -error pattern will depend on *both* the  $X$  and  $Z$  syndromes; that is, maximum-likelihood decoding for  $X$  errors will be performed using some function  $F$  of both  $s_x$  and  $s_z$ ,

$$\tilde{x} = F(s_x, s_z). \quad (2.67)$$

Note that by the symmetry between the way  $X$  and  $Z$  errors are applied and corrected, the maximum-likelihood decoding of the  $Z$  error pattern would use the same function, but with the inputs reversed:

$$\tilde{z} = F(s_z, s_x). \quad (2.68)$$

It may appear strange that knowledge of the  $Z$ -syndrome should be used to help determine the most likely  $X$ -error pattern. However, recall that when a qubit is depolarized, the four operators  $I$ ,  $X$ ,  $XZ$  and  $Z$  are applied with probability  $1/4$  each; so if a qubit has been effected by a  $Z$  error then there is a 50% chance it has also been effected by an  $X$  error. So, there will be significant correlations between the error patterns  $x$  and  $z$  when an independent depolarizing channel has been applied.

I won't go into the details of how the decoding function  $F(s_x, s_z)$  is computed for the depolarization channel. Note however, that there is a much simpler alternative to computing the true maximum-likelihood decoder: instead simply apply the syndrome decoder  $D_S$  (as defined for classical linear codes in Subsection 2.1.3.2) individually to the

two syndromes  $s_x$  and  $s_z$ . That is, we find the following “approximately most-likely” error patterns,

$$\tilde{x}' = D_S(s_x), \quad (2.69)$$

$$\tilde{z}' = D_S(s_z). \quad (2.70)$$

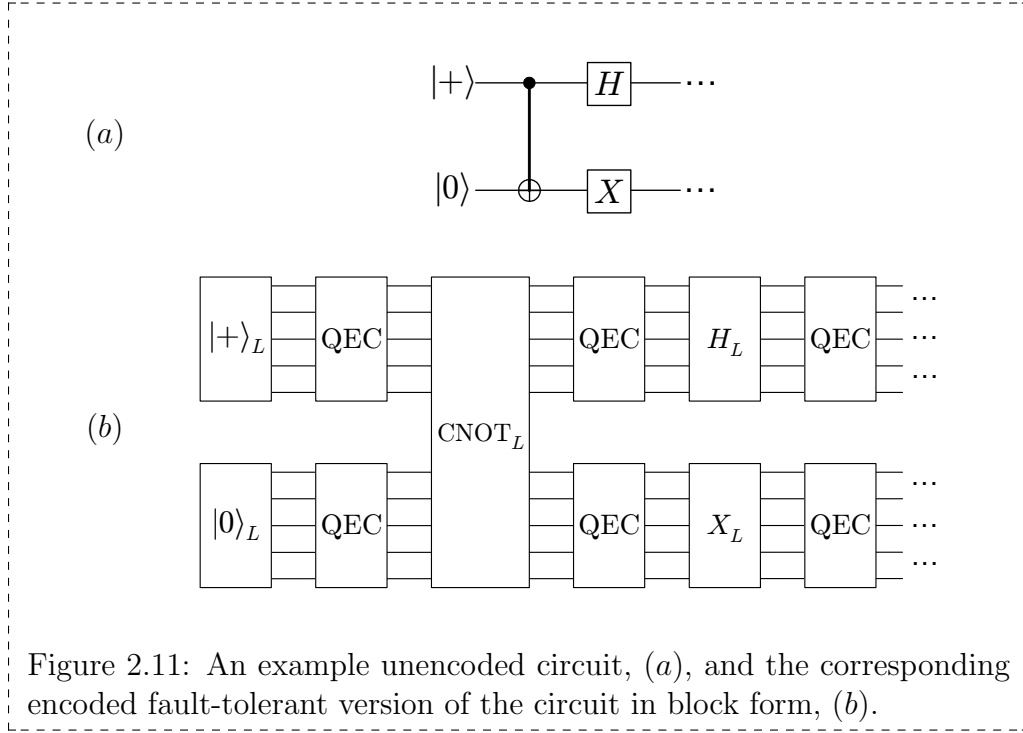
Now, from the properties of the syndrome decoder function  $D_S$ , we know that  $\tilde{x}'$  and  $\tilde{z}'$  will equal the actual error patterns  $x$  and  $z$  whenever  $\text{wt}(x) \leq t$  and  $\text{wt}(z) \leq t$  (where  $t$  is the number of errors that the classical code  $C$  can correct). This will be the case whenever no more than  $t$  of the qubits have been subject to Pauli errors. Thus, by choosing to apply a recovery operation based on the patterns  $\tilde{x}'$  and  $\tilde{z}'$ , the CSS code will be able to correct arbitrary Pauli errors on as many as  $t$  qubits.

Note, it can be shown that the decoding scheme given in Eqs. (2.69) and (2.70) has identical behaviour to the true maximum-likelihood decoders of Eqs. (2.67) and (2.68) whenever no more than  $t$  qubits contain Pauli errors. (But, when there are more than  $t$  errors, the two schemes will generally give different results). In fact, Eqs. (2.69) and (2.70) do precisely correspond to a maximum-likelihood decoder but for a different noise model – where  $X$  and  $Z$  Pauli errors are each introduced to the state independently with probability  $p$  (and thus such that a  $Y$  error only occurs with probability  $p^2$ ).

## 2.3 Fault tolerance


In this section, I review the concept of quantum *fault-tolerance*. A *fault-tolerant* device is one that operates reliably even when its individual components are unreliable. Fault tolerance will be an extremely important ingredient in the design of future quantum information processing devices, since all known practical methods for manipulating qubits are prone to significant amounts of noise.

One general approach for making a device fault-tolerant – and the one I shall focus on in this section – is to make it repeatedly perform error-correction on its own state as it runs. For example, say we want to build a device whose purpose is to perform the quantum circuit shown in Figure 2.11a. Suppose that all the components in the circuit, including the gates and the state preparations (and if they had been present, measurement operations also), are unreliable. That is, there is some probability that each of the components will introduce an error to its output. For the time being, let’s not worry about the details of how these errors are described; it suffices to think of a failed component as simply causing a random Pauli operation to be applied. Then, a fault-



tolerant version of the circuit in Figure 2.11a can be constructed according to the outline in Figure 2.11b. That is, each of the components in the original circuit are replaced by an encoded version. In addition, error-correction steps are placed between each of the encoded operations.

To be precise, the circuit in Figure 2.11b is an *error-corrected, encoded* version of the circuit in Figure 2.11a. This does not necessarily make it fault-tolerant. In order for an encoded circuit to be fault-tolerant, the logical operations and error-correction steps must be constructed in a way that individually satisfy certain requirements for fault-tolerance, as described further below. Then, so long as the failure probability of each component is not too large, the encoded circuit will be more reliable than the original unencoded one. On the other hand, as I demonstrate later in this section, an error-corrected encoded circuit that is *not* constructed fault-tolerantly will generally be *less* reliable than the unencoded one.

In order for an individual error-correction block to be fault-tolerant it must satisfy the following condition: the failure of a single component within the error-correction block must never cause more than one error to be present on the output of that block. This requirement is illustrated in Figure 2.12. The symbol “” denotes the failure of a component within the error-correction block. Figure 2.12a shows the behaviour of a non-fault-tolerant error-correction procedure, where a single failure can lead to more than

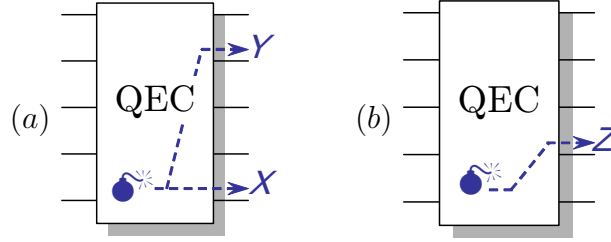


Figure 2.12: A cartoon comparing the behaviour of error-correction procedures that are (a) not fault-tolerant and (b) fault-tolerant.

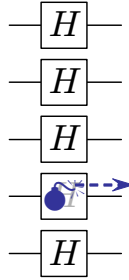


Figure 2.13: A logical operation that operates transversally is always fault-tolerant, since a failed component will only ever affect one output qubit.

one error. Figure 2.12b on the other hand shows behaviour that is fault-tolerant, in that the failure leads to just a single error in the output.


The condition described above is also identical to that required to make a logical operation fault-tolerant. Note, however, that for a logical operation that acts on two or more encoded qubits, such as a logical controlled-not gate, the condition should be applied separately to each of the encoded outputs. That is, a failed component in a two-qubit logical gate must not cause more than one error in the first output encoded qubit, or more than one error in the second output encoded qubit.

Recall from Subsection 2.2.3.3 that many important logical operations for CSS codes can be performed transversally. This includes encoded versions of the gates  $X$ ,  $Y$ ,  $Z$ ,  $H$ , controlled-not, and controlled-phase, as well as encoded Pauli-basis measurements. It is clear that any logical operation that is implemented transversally is automatically fault-tolerant. This fact is illustrated in Figure 2.13. However, the set of gates that can be performed transversally for CSS codes is not *universal* for quantum computation. That is, not all unitary transformations (i.e., quantum algorithms) can be constructed by composing just these gates together. The set does become universal, however, with the addition of the  $\pi/8$  gate. As mentioned in Subsection 2.2.3.3, there exist implementations of the  $\pi/8$  gate which, although are not transversal, are nonetheless fault-tolerant.

I'll omit further discussion on fault-tolerant logical operations, and concentrate instead on how to perform an error-correction procedure fault-tolerantly. The basic error-correction scheme for CSS codes that I described in Subsection 2.2.3.4 is not fault-tolerant; however, in the following subsection I describe a method by Steane [Ste97] for modifying the basic scheme so that it becomes fault-tolerant. The details of this method form an important foundation for the work that I describe in Chapter 6 of this thesis.

### 2.3.1 Steane's method for fault-tolerant error-correction

Steane's method for fault-tolerant error-correction takes the non-fault-tolerant scheme given in Subsection 2.2.3.4 and extends it in a way that makes it fault-tolerant. To begin, let's pinpoint the reason why the scheme in Subsection 2.2.3.4 is not fault-tolerant. Recall, error-correction for a CSS code is performed by first measuring the  $X$  and  $Z$ -error syndromes, then decoding these syndromes, then finally correcting the state by applying appropriate Pauli operations.

Consider the  $X$ -syndrome measurement procedure that was described in Subsection 2.2.3.4. Figure 2.14 shows the circuit for this procedure, explicitly for the case of the 7-qubit Steane code. The circuit is not fault-tolerant, a fact that can be seen clearly by considering what happens when one of the controlled-not gates fail in the  $|0\rangle_L$ -state-creation part of the circuit. Note, in this context of syndrome measurement, I shall henceforth refer to the circuit for creating the state  $|0\rangle_L$  as the “ancilla creation circuit”, and refer to the state  $|0\rangle_L$  as the “ancilla state”. Imagine that the gate marked by the symbol “” in Figure 2.14 fails, and say that this failure causes an  $X$  error on the output of that gate as shown. This error will then spread to become two errors as it propagates through the next controlled-not gate, according to the commutation relation

$$\begin{array}{c} \text{---} \oplus \text{---} \\ | \\ \boxed{X} \text{---} \bullet \text{---} \end{array} = \begin{array}{c} \text{---} \oplus \boxed{X} \text{---} \\ | \\ \text{---} \bullet \boxed{X} \text{---} \end{array} \quad (2.71)$$

Thus, the created ancilla state will contain two  $X$  errors. These errors will then propagate through the controlled-phase gates, according to the commutation relation in Eq. (2.60), to become two  $Z$  errors on the encoded data. Importantly, this pattern of errors introduced to the data is one that is not correctable by the 7-qubit code.

Similarly to the example above, several of the other controlled-not gates in Figure 2.14 will also cause an uncorrectable pattern of  $Z$  errors to propagate to the data, when they fail. The overall effect will be as follows: the probability that an uncorrectable error

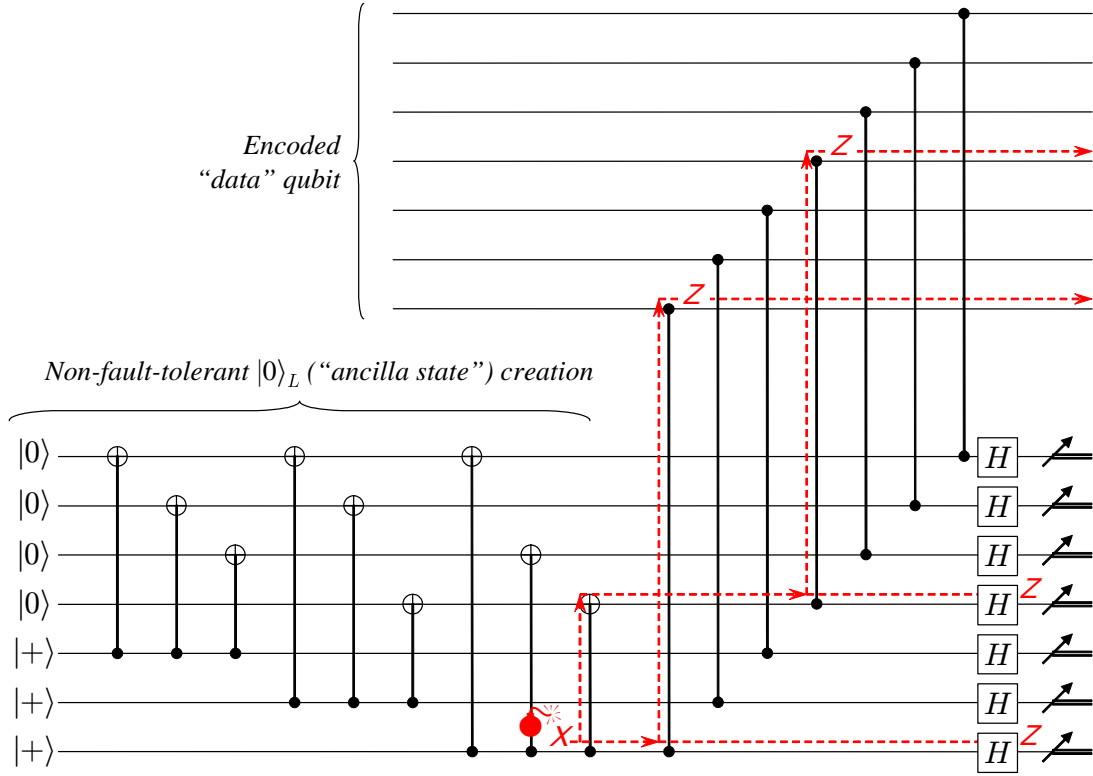


Figure 2.14: The non-fault-tolerant  $X$  syndrome measurement circuit, for the 7-qubit Steane code. A single gate failure can spread to become two or more  $Z$  errors on the state being corrected.

is introduced to the data by the syndrome measurement circuit will be *greater* than the probability of failure for a single unencoded gate. This means that applying an error-correction scheme that uses a non-fault-tolerant  $X$ -syndrome measurement will cause the effective error rate on the data to be *amplified*, not suppressed.

The issue is almost identical for the case of the  $Z$ -syndrome measurement circuit. Again, a single failure in the ancilla-creation circuit can lead to multiple  $X$  errors on the ancilla state, which in this case propagate to become  $X$  errors on the data. Thus, it is the lack of fault-tolerance of the ancilla-creation circuit which causes both the  $X$  and  $Z$ -syndrome measurement circuits to fail to be fault-tolerant. Accordingly, Steane's scheme for fault-tolerant error-correction is based on the use of an alternate, fault-tolerant, ancilla-creation procedure for syndrome measurement.

Note, it is only necessary to make the ancilla-creation circuit fault-tolerant with respect to  $X$  errors, not  $Z$  errors. That is, if we denote the noisy output of the ancilla-creation



circuit as

$$(X^{x_1} \otimes X^{x_2} \dots \otimes X^{x_N})(Z^{z_1} \otimes Z^{z_2} \dots \otimes Z^{z_N})|0\rangle_L, \quad (2.72)$$

then we only require that the following “X-fault-tolerance” condition holds: a single failure in the ancilla creation circuit shall not cause more than one element of the vector  $x$  in Eq. (2.72) to become nonzero. This weaker fault-tolerance condition is sufficient for the following reason: when the noisy ancilla in Eq. (2.72) is used for syndrome extraction, the pattern of errors that propagates to the data is equal to  $Z^{x_1} \otimes Z^{x_2} \dots \otimes Z^{x_N}$  (in the case of  $X$ -syndrome measurement) or  $X^{x_1} \otimes X^{x_2} \dots \otimes X^{x_N}$  (in the case of  $Z$ -syndrome measurement). That is, it is only the pattern “ $x$ ” that propagates to the data in both cases.

On the other hand, the pattern of  $Z$  errors on the ancilla doesn’t directly affect the data being corrected. Nevertheless, the  $Z$  errors on the ancilla do lead to errors in the *syndrome* measurements, and this can indirectly affect the data by leading to an incorrect decoding outcome. Steane’s scheme provides a simple way of dealing with these unreliable syndrome measurements. Each syndrome-measurement circuit is repeated  $N_R$  times in series on the same set of data, for some choice of  $N_R > 1$ . That is,  $N_R$   $X$ -syndromes are measured, followed by  $N_R$   $Z$ -syndromes. Only when some number  $N_A$  of the syndromes agree, where  $N_A$  is chosen such that  $1 < N_A \leq N_R$ , will the syndrome measurement be considered successful, and decoding performed. Steane has carried out extensive numerical investigations [Ste03] to find the best values of  $N_R$  and  $N_A$  in various circumstances.

Steane’s approach to making the ancilla-creation circuit fault-tolerant with respect to  $X$ -errors is to append a *verification circuit* to it, as shown in Figure 2.15 for the 7-qubit code. The verification circuit outputs a series of *verification bits*. If any of the verification bits equal 1, then the verification is deemed to have failed; and conversely if all verification bits equal 0 then the verification is deemed to have succeeded. Any ancilla that fails verification must be discarded (and its creation reattempted). It can be shown that an ancilla state that passes verification will only contain multiple  $X$  errors if more than one gate failed during the creation and verification circuits. So, if only the successfully-verified ancilla states are used in the syndrome-measurement procedures, then the syndrome measurements will be fault-tolerant as required. Figure 2.15 shows an example of the effect of a single failed gate on the creation and verification circuits. Although this failed gate causes multiple  $X$  errors on the output ancilla, it also causes three of the verification bits to equal 1, and so the ancilla state will be discarded.

Let’s consider in a little more detail how the verification circuit works. When it operates perfectly, the verification circuit is designed so that if some computational basis

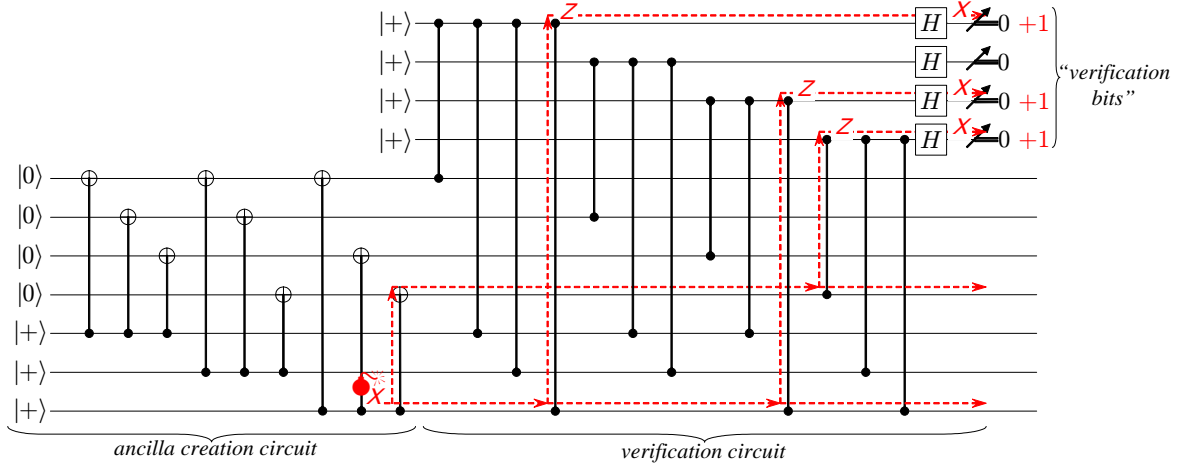


Figure 2.15: Steane's fault-tolerant ancilla creation circuit, for the 7-qubit Steane code.

state  $|y\rangle$  is input to the circuit then the verification bits will yield the vector  $H_{C^\perp}y$ , where  $H_{C^\perp}$  is the parity-check matrix of the code  $C^\perp$ . That is, the verification circuit measures the  $X$ -syndrome of its input, with respect to the code  $C^\perp$ . The arrangement of the gates in the verification circuit depends in a very simple way on the entries in the matrix  $H_{C^\perp}$ . To see this in the case of Figure 2.15, note that if  $C$  is the 7-bit Hamming code, then  $H_{C^\perp}$  is given by

$$H_{C^\perp} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}. \quad (2.73)$$

Then, the verification circuit in Figure 2.15 is constructed from Eq. (2.73) as follows. A number of “verification qubits” are initialized to the state  $|+\rangle$  (there is one verification qubit for each row of  $H_{C^\perp}$ ). For each location  $(m, n)$  in the matrix  $H_{C^\perp}$  that contains a “1”, a corresponding controlled-phase gate is applied between the  $m$ -th verification qubit and the  $n$ -th ancilla qubit. After all the controlled-phase gates have been applied, a Hadamard gate is applied to each verification qubit, and then each verification qubit is measured in the computational basis. It is easy to check that this circuit does in fact have the effect of measuring the  $X$ -syndrome for the code  $C^\perp$ .

Recall that the state  $|0\rangle_L$  is a superposition of all codewords in the code  $C^\perp$ . So, in the noise-free case, it follows that if the state  $|0\rangle_L$  is input to the verification circuit then all the verification bits will equal zero; and it also follows that this state will be preserved by the verification process. If the input to the verification circuit is instead the following

state,

$$|\psi\rangle = (X^{x_1} \otimes \cdots \otimes X^{x_N})|0\rangle_L, \quad (2.74)$$

i.e., equal to the state  $|0\rangle_L$  subject to some pattern of  $X$  errors, then it can be shown that the verification bits will all equal zero if and only if  $|\psi\rangle = |0\rangle_L$ . That is, the verification circuit can detect any nontrivial pattern of  $X$  errors on the ancilla state. I use the word “nontrivial” for the following reason: it can be shown that any pattern of  $X$ -errors that corresponds to a codeword of  $C^\perp$  will leave the state  $|0\rangle_L$  unchanged. The verification circuit will thus not detect these “trivial” error patterns.

We can thus show that the combined ancilla creation and verification process is fault-tolerant, as follows. If a gate fails during the ancilla-creation circuit but not during the verification circuit, then the resulting ancilla state is guaranteed to have no  $X$  errors if the verification bits are zero, according to the paragraph above. Say, on the other hand, that no gate fails during the ancilla creation, but that a gate fails during the verification process. This failure can create at most one  $X$  error on the ancilla. Any remaining gates in the verification circuit will not cause this error to multiply to create other  $X$  errors on the ancilla, a fact that can be seen by considering the rules for commuting Pauli operators through controlled-phase gates (i.e., using Eq. (2.60), and the fact that  $Z$  operations commute with controlled-phase gates). Thus, a single failed gate anywhere during the creation and verification process will not lead to more than one  $X$  error in the ancilla, and thus the overall procedure is fault-tolerant.

As a means of summarizing, the elements of the Steane protocol described above are combined together in Figure 2.16, showing a complete circuit for a fault-tolerant error-correction operation. Note that to simplify the diagram, I have assumed that all verification tests are successful. Also, the Hadamard gates before measurements have been omitted – this effectively assumes that all measurements in the diagram are performed in the  $X$  basis instead of the computational basis.

Also shown in Figure 2.16 is Steane’s approach for *parallelizing* the gates in the error-correction circuit, as indicated by the dashed vertical lines. Imagine that the physical system being used to implement the error-correction circuit has the ability to perform several gates in parallel (provided that any two gates acting in parallel do not operate on the same qubit). It makes sense to take maximal advantage of this ability, by packing as many gates together in parallel as possible, thus reducing the amount of time that each qubit is idle for (and accumulating noise). The dashed lines in Figure 2.16 are used to indicate groups of operations that are applied in parallel with each other. More information about how this grouping is chosen can be found in [Ste02].

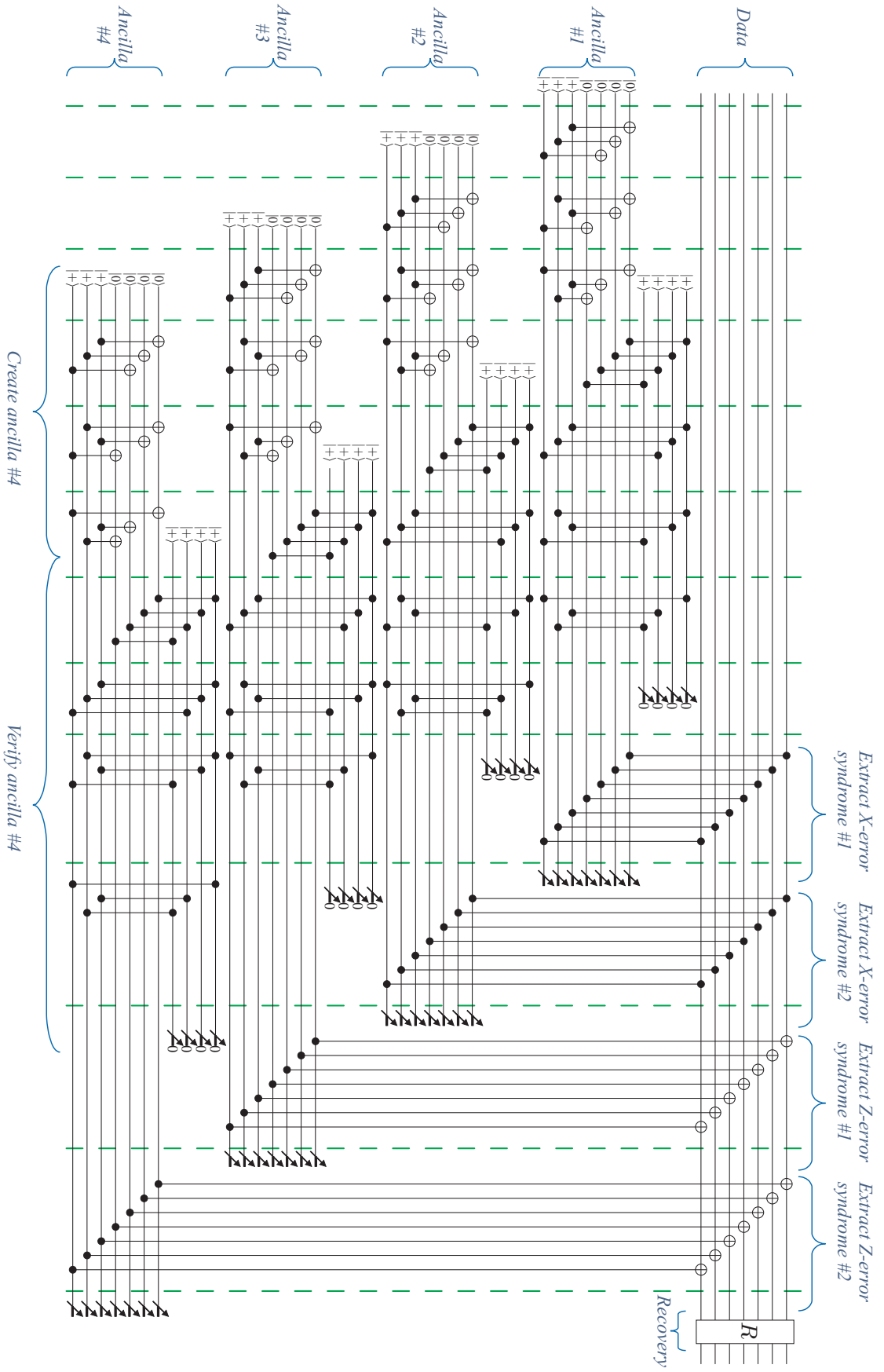


Figure 2.16: Complete fault-tolerant error-correction circuit for the 7-qubit Steane code, assuming two repetitions for each syndrome measurement, and assuming all verification tests pass. N.B.: all measurements are in the  $X$  basis.

### 2.3.2 Noise thresholds in fault-tolerant quantum systems

The purpose of this subsection is to review the concept of the *noise threshold* of a fault-tolerant quantum system. Broadly speaking, the noise threshold is the maximum level of noise that can be efficiently and reliably corrected by a particular fault-tolerant system.

For the purposes of this discussion, I'll continue to use a notion of noise along the lines of the last subsection. That is, imagine that each of the basic unencoded operations in the system have some probability of failure, and such a failure will cause a random error to be created on the qubit that the component acts on. To keep this discussion simple, let's assume that all the basic operations share the same probability of failure, denoted  $p$ . So, the parameter  $p$  describes the “noise level” present throughout the physical system.

When a circuit is replaced by its encoded version, the effective value of the noise level will change. That is, whereas an unencoded operation fails with probability  $p$ , the error-corrected encoded version of that operation will have some *effective failure probability*, which we denote  $p_{\text{eff}}$ . The effective failure probability is defined to be the probability that an uncorrectable error is created on the data, when a logical operation and its associated error-correction step are applied. Note that the occurrence of an uncorrectable error is often called a *crash*, and  $p_{\text{eff}}$  is thus also known as the *crash rate*.

It is  $p_{\text{eff}}$  that determines the overall reliability of an encoded circuit. A circuit is said to “succeed” when no crashes are caused by any of the logical operations in the circuit. Say that we wish to implement a circuit containing  $M$  logical operations, and we insist that the circuit has a probability of success of at least  $q$ . Then, we require that  $p_{\text{eff}}$  be small enough for the following condition to hold:

$$(1 - p_{\text{eff}})^M \geq q. \quad (2.75)$$

(Note, this ignores the fact that, in general, different types of logical operations will have different values for  $p_{\text{eff}}$ , even when the various types of *unencoded* operations have the same value of  $p$ . However, this complication is not particularly important for the discussion at hand). The important thing to take from Eq. (2.75) is that, for a particular desired overall reliability  $q$ , if we want the ability to implement circuits of arbitrary size  $M$  then we need to be able to achieve *arbitrarily low* values of  $p_{\text{eff}}$ . If the noise level  $p$  has some fixed value, then the only way to achieve variable values of  $p_{\text{eff}}$  is to vary the type of error-correction that is used.

It is thus useful to imagine that we have a *family* of different error-correction protocols, each having a different “strength” of ability to correct errors. The different protocols in the family would be used depending on what value of  $p_{\text{eff}}$  was required. Usually, the stronger

error-correcting protocols will make use of larger codes, and use correction circuitry that is correspondingly more complex. For example, stronger protocols can be constructed by *concatenating* other protocols together, as I describe later in this subsection.

When such a family of protocols has been constructed, an important question to ask is “for what values of the physical noise level,  $p$ , will arbitrarily low values of  $p_{\text{eff}}$  be achievable?”. This is where the concept of the *noise threshold* comes in. The noise threshold can be defined in broad terms as follows.

**Definition 2.1** (Noise threshold): *If a family of fault-tolerant error-correction protocols has a noise threshold equal to  $p_{\text{thresh}}$ , then by definition the following will hold:*

1. *If  $p < p_{\text{thresh}}$  (that is, if the physical noise level is “below the threshold”), then for all  $\eta > 0$  there will exist a protocol in the family that achieves an effective failure probability  $p_{\text{eff}} \leq \eta$ . Furthermore, the protocol that achieves this will be efficient (see text).*
2. *If  $p > p_{\text{thresh}}$ , then no protocol in the family will achieve  $p_{\text{eff}} < p_{\text{thresh}}$ .*

Note the use of the word “efficient” in the above definition. Basically, we require that as we decrease the desired value of  $p_{\text{eff}}$ , the complexity of the resulting error-correction protocol shouldn’t increase too rapidly. A reasonable working definition of “efficient” is that by increasing the complexity of an error-correcting protocol linearly the effective failure probability should decrease exponentially, whenever  $p$  is below the threshold.

So, by Definition 2.1, when a fault-tolerant system is characterized by a noise threshold, then that system will have radically different behaviour depending on whether the physical noise level is above or below the threshold. When the noise level is below the threshold, the system will be able to efficiently correct the noise, to arbitrary levels of reliability. On the other hand, when the noise level is above the threshold, the device will remain unreliable regardless of how strong the error-correction is.

Why should a noise threshold even exist? In the next few paragraphs, I’ll outline an argument to show why a nonzero threshold exists for Steane’s fault-tolerant error-correction protocol, and give a crude method for estimating its value in this case.

Say that a circuit has been converted to a fault-tolerant error-corrected version, by the general method that was illustrated in Figure 2.11 – that is, by replacing each operation by a fault-tolerant encoded version, and by placing a fault-tolerant error-correction procedure between each logical operation. Say that the error-correction procedure corresponds to Steane’s fault-tolerant protocol as described in the previous subsection, and say that the code being used is the 7-qubit Steane code.

Now, let's focus our attention on one particular logical gate within the circuit, and consider the value for the effective failure probability,  $p_{\text{eff}}$ , of this gate. Recall, the effective failure probability is the probability that an uncorrectable error pattern is created, when the logical gate is applied along with its adjacent error-correction procedure. The value of  $p_{\text{eff}}$  will be some function of the physical noise level  $p$ , i.e.,

$$p_{\text{eff}} = F(p), \quad (2.76)$$

for some function  $F$ . Say that there are a total of  $W$  unencoded operations that make up the circuits for the logical gate and error-correction procedure. Then  $F(p)$  can be expressed, using the law of total probability, as

$$F(p) = \sum_{n=0}^W P(\text{crash}|n)P(n), \quad (2.77)$$

where  $P(\text{crash}|n)$  is defined to be the probability that an uncorrectable error (crash) occurs given that  $n$  of the  $W$  gates have failed, and  $P(n)$  is defined to be the probability that  $n$  gates fail. The values  $P(\text{crash}|n)$  for the various  $n$  are constants, in that they depend only on the layout of the circuits for the logical operation and error-correction, not on the value of  $p$ . For conciseness in what follows, denote these constants as  $P(\text{crash}|n) \equiv D_n$ . Substituting the binomial distribution for  $P(n)$  in Eq. (2.77) gives us

$$F(p) = \sum_{n=0}^W D_n \binom{W}{n} p^n (1-p)^{W-n}. \quad (2.78)$$

Now, an error is uncorrectable by the 7-qubit code only when the error affects two or more qubits. But, by our assumption that the logical operation and error-correction circuits are fault-tolerant, it requires two or more gates to fail in order for two or more qubits to be affected by errors. It follows then that  $D_0 = D_1 = 0$ . So,

$$F(p) = \binom{W}{2} D_2 p^2 + O(p^3). \quad (2.79)$$

For small  $p$  it will be a good approximation to truncate this expression to order  $p^2$ , giving

$$F(p) \approx \binom{W}{2} D_2 p^2. \quad (2.80)$$

Now, to construct a family of fault-tolerant protocols, in order to show the existence of a noise threshold, we use the technique of *protocol concatenation*. Protocol concatenation is closely related to the idea of code concatenation discussed in Subsection 2.2.2.3. By definition, if a circuit is encoded using a protocol that is *concatenated to two levels*, then

the original circuit is transformed in the following way. First, the circuit is encoded using the usual procedure of Figure 2.11. Then, the circuits that implement each of the resulting logical operations and error-correction steps are themselves encoded, again using the procedure of Figure 2.11. In other words, a protocol that is concatenated to two levels is constructed by simply applying the usual encoding transformation twice to the circuit, instead of once. More generally, a protocol that is concatenated to  $L$  levels is constructed by applying the encoding transformation  $L$  times to the circuit.

The resulting circuit can then be thought of as operating at  $L + 1$  different “levels”. The highest level consists of the logical operations that make up the original circuit (for example, operations that carry out the algorithm of a quantum computation). Each of the other  $L$  lower levels consist of circuitry for implementing the logical operations of the next highest level, as well as the error-correction circuitry between these logical operations.

Importantly, this means there will be a different effective failure rate at each level. Define  $p_{\text{eff}}^{(j)}$  to be the effective failure rate of the logical operations at level  $j$ . (Note that the operations performed at level 1 are unencoded, so  $p_{\text{eff}}^{(1)} = p$ ). Recall that  $F$  is defined to be the function that gives the effective probability  $p_{\text{eff}}$  as a function of the noise level  $p$ , for the ordinary single-level Steane protocol. Ignoring some subtleties, it is reasonable to argue that in the multiply-concatenated case the same function  $F$  will give the effective failure probability for any level  $j + 1$ , as a function of the effective failure probability at the next lowest level  $j$ . (Basically, the argument is that the operations at level  $j$  will behave roughly as though they were unencoded operations, but with a physical noise level of  $p_{\text{eff}}^{(j)}$ ). Thus,

$$p_{\text{eff}}^{(j+1)} = F(p_{\text{eff}}^{(j)}) \quad (2.81)$$

$$\approx \binom{W}{2} D_2 \left[ p_{\text{eff}}^{(j)} \right]^2 \quad (2.82)$$

(where the approximate expression for  $F$  in Eq. (2.80) was used in the second line above). So, the effective failure probability at the highest level,  $L + 1$ , will be given by

$$p_{\text{eff}}^{(L+1)} \approx \left[ \binom{W}{2} D_2 \right]^L p^{2^L}. \quad (2.83)$$

As  $L \rightarrow \infty$ , the expression on the right hand side of Eq. (2.83) will converge to zero if and only if  $p < p_{\text{thresh}}$ , where

$$p_{\text{thresh}} = \frac{1}{\binom{W}{2} D_2}. \quad (2.84)$$

Thus we can conclude that Steane’s protocol is characterized by a nonzero noise threshold.

We can use Eq. (2.84) to obtain a crude numerical estimate of the noise threshold. The circuit for fault-tolerant error-correction shown in Figure 2.16 contains 204 unencoded



operations. We can use this number as an approximate value for  $W$ . (In a more accurate treatment, the gates that make up the adjacent logical operation should also be counted in  $W$ ). It is difficult to estimate  $D_2$  accurately, but for the sake of this calculation let's assume that an uncorrectable error will *always* occur when two gates fail; that is, take  $D_2 = 1$  as a pessimistic estimate. Then we have that the noise threshold for Steane's protocol is approximately  $\frac{1}{0.5 \times 204 \times 203 \times 1} \approx 5 \times 10^{-5}$ . (This can be compared with the more accurate estimate obtained by Steane [Ste03], equal to  $3 \times 10^{-3}$ ).

The above analysis also shows that, in principle, the multiply-concatenated Steane protocol is efficient (in terms of the level of reliability that can be achieved as a function of the complexity of the error-correction circuitry). Although the total number of unencoded operations required to implement the protocol to  $L$  levels scales as  $W^L$  (i.e., exponentially in  $L$ ), the effective failure rate decreases as the exponential of an exponential (or “doubly exponentially”) in  $L$ . In other words, the effective failure probability decreases exponentially as a function the complexity of the error-correction circuitry, a behaviour that can be considered as efficient.

The argument above for the existence of a noise threshold is clearly lacking in a number of ways. It is far from rigorous, and it gives a rather inaccurate estimate for the value of the threshold. However, there exist rigorous proofs for the existence of the noise threshold. Such *noise threshold theorems* were first proven independently by Aharonov and Ben-Or [ABO97, ABO99], Kitaev [Kit97c, Kit97b], and Knill, Laflamme and Zurek [KLZ96, KLZ98]. (A number of important variants have also been proven since, including those described in [TB04], [AGP06], and [DA06]). The different proofs consider a range of different physical assumptions, and in each case show that reliable, efficient quantum computation is possible so long as the noise level is below the threshold, where the threshold is a non-zero value that doesn't depend on the size of the computation.

Although the threshold theorems prove the existence of a noise threshold, they do not indicate precisely what value the threshold takes in each case. Usually it is necessary to perform a sophisticated numerical simulation in order to accurately determine the value. Chapter 6 of this thesis describes an example of such a numerical simulation. The original research described in that chapter is aimed at calculating the noise threshold for a particular physical implementation of quantum computation, namely the *linear-optical cluster-state model*.

### 2.3.3 Naturally fault-tolerant systems

There exist proposals for fault-tolerant error-correction that are radically different to Steane's protocol. Whereas Steane's protocol is used to create an "actively fault-tolerant system", there also exist proposals for constructing *naturally fault-tolerant systems*. Although the theory of natural fault-tolerance is less developed than that of active fault-tolerance, it is thought that techniques of natural fault-tolerance may have the potential to radically simplify the construction of fault-tolerant quantum systems.

An "active" fault-tolerant protocol, such as Steane's, requires that a complex series of operations be dynamically applied to the system, in order for error-correction to occur. On the other hand, the idea behind a naturally fault-tolerant system is that the complexity required to achieve error-correction should be built in to the physical construction of each of the individual components. The aim is for the natural behaviour of the components to be fault-tolerant, without the need for complex external control.

A useful analogy can be made with classical devices that are fault-tolerant, for example a hard drive<sup>5</sup>. A hard drive uses active error-correction to some degree, but the main reason that a hard drive is reliable is that the medium used to store information is naturally fault-tolerant. Each bit of information is stored on some number of magnetic domains, in a ferromagnetic material. A magnetic domain is a large collection of atoms that are all magnetically aligned. The orientation of this alignment determines the value of the bit that is stored. Due to the ferromagnetic interactions between atoms, it is energetically favourable for the atoms within a domain to remain aligned with each other. When noise causes some small fraction of the atoms to become misaligned, they will revert to their former alignment without disturbing the overall orientation of the domain. Thus, the information stored in a hard drive is naturally fault tolerant, due to the self-correcting properties that arise from the physical interactions.

There are various proposals for quantum natural fault-tolerance that work along analogous lines to the above example. A single encoded qubit is composed of many unencoded physical qubits that interact together in a specific way. The interactions are engineered so as to make it energetically favourable for the qubits to be in an error-correcting code state. That is, the system is constructed so that it has a degenerate ground state, where the space of states that belong to this degeneracy form some quantum error-correcting code. Such a design would then give the system the ability to self-correct. Examples of such proposals include the toric code by Kitaev [Kit97a] and the closely-related variant by Dennis, Kitaev, Landahl, and Preskill [DKLP02], and the subsystem code by Bacon

---

<sup>5</sup>This analogy is borrowed from Dave Bacon [Bac97]

[Bac05].

There are many complications involved in designing a set of interactions that lead to natural fault-tolerance. For example, the system must be made to have a large *energy gap* between the ground state and the first excited state, or else the logical qubit will be too susceptible to thermal fluctuations. Also, there should be an increasingly large energy “penalty” associated with states having increasingly many errors on them, so that it will generally be energetically favourable for errors to decrease.

A further important design criteria for such systems is that the interactions used are “physically realistic”. In the following chapter, I present new results which show that if physically-realistic interactions are used to build a naturally-fault tolerant quantum system, then this severely restricts the type of error-correcting code that the system can use.



## Eigenstates of physically-plausible systems

---

In many areas of physics and quantum information science it is crucial to be able to understand the energy eigenstates (that is, stationary states) of a many-body system. In this chapter, we show that making simple and quite reasonable assumptions about the interactions in a system can lead to severe restrictions on the possible eigenstates. The assumption we consider is  $L$ -locality, meaning simply that the interactions in a system are the sum of interactions that each involve at most  $L$  bodies. We show that no eigenstate of any nontrivial  $L$ -local Hamiltonian can be close to any state belonging to a certain important class of quantum error-correcting code states. This has implications not only for many-body physics, but for the prospect of creating naturally fault-tolerant quantum devices.

Note that this chapter contrasts with the other research chapters in this thesis in some important ways. First, it is the only chapter that deals directly with naturally fault-tolerant systems. Second, the main outcome of the chapter is a “negative” result, in that certain approaches to fault-tolerant devices are ruled out. The other research chapters, on the other hand, describe the construction of new protocols for reliable quantum information processing.

*NOTE: This chapter is based on a published journal article, [H. L. Haselgrove, M. A. Nielsen, and T. J. Osborne, Phys. Rev. Lett. 91:210401 (2003)]. Much of the text and maths from that article is included in this chapter, with varying degrees of modification. However, Sections 3.3 and 3.4 of this chapter were written entirely by myself for the sole purpose of inclusion in this thesis. The contribution of myself and my coauthors to the original article can be summarized as follows. I was responsible for a series of numerical calculations which originally suggested that most states could not be near the eigenstate of any  $L$ -local Hamiltonian. Osborne and I found the dimension-counting argument for explaining this fact. I found the condition for testing whether there exists an  $L$ -local Hamiltonian having a specified eigenstate. Osborne discovered the explicit class of states*

that could not be a  $L$ -local eigenstate. Nielsen created the proof of the general bound. Nielsen wrote most of the text of the article.

### 3.1 Introduction

When it comes to understanding the statics or dynamics of a quantum system, knowing the energy eigenstates is often key. Thus, a central problem in physics is to characterize eigenstates of Hamiltonians, and of a particular challenge is doing this characterization for complex many-body systems.

A range of different approaches exist for studying eigenstates, depending on what type of assumptions are being made about the system. In principle, direct numerical calculation of eigenstates can be performed if one has precise knowledge of the interactions in a system, by expressing those interactions as a Hamiltonian matrix and performing numerical diagonalisation. However, the large computational power required for this approach can easily become impractical for systems containing many bodies.

If *no* assumptions are made about the type of interactions in a many-body system, then quantum mechanics allows *any* state to be an eigenstate, by the choice of an appropriate Hamiltonian – for example, any state  $|E\rangle$  is an eigenstate of a corresponding Hamiltonian  $H = |E\rangle\langle E|$ . However, there is no guarantee that the interactions corresponding to a Hamiltonian such as  $|E\rangle\langle E|$  would be physically realistic. So a question that may be asked is: which states  $|E\rangle$  are physically realistic eigenstates?

The approach in this chapter is to consider an assumption,  $L$ -locality, that is general enough to encompass any imaginable physically plausible Hamiltonian, yet restrictive enough to still have interesting consequences for the eigenstates. We do not directly describe the resulting eigenstates, rather we obtain a set of states which are provably “far away” from all eigenstates of all nontrivial  $L$ -local Hamiltonians. Note that by nontrivial, we mean not a multiple of the identity. Hamiltonians which are a multiple of the identity effectively contain *no* interactions, and thus every state is a stationary state. Thus we exclude this uninteresting case, and only consider  $L$ -local Hamiltonians containing some interaction.

The particular set of states  $|\psi\rangle$  which we show to be far away from all eigenstates of all  $L$ -local Hamiltonians, are the *nondegenerate quantum error correcting codes correcting  $L$  errors*. It is interesting in itself that an abstract information-theoretic concept such as error-correcting codes should provide a useful perspective on many-body physics. But this connection has practical implications as well, for the prospect of creating a naturally

fault-tolerant quantum information processing device.

Recall from Subsection 2.3.3 that one of the main approaches to making a quantum system naturally fault-tolerant is to engineer the interactions in the system in a way that makes the ground states (minimum-energy eigenstates) belong to a quantum error-correcting code. Our results show that for such a scheme to work, it cannot be based on a *nondegenerate* code. This sheds a new light on why existing proposals (e.g., see [Kit97a] and [Bac05]) use codes which are highly degenerate.

The remainder of this chapter is structured as follows. This introductory section is completed by Subsection 3.1.1, being a brief comment on  $L$ -local interactions in classical and quantum mechanics. In Section 3.2 we give a simple dimension-counting argument showing that “most” quantum states are not the eigenstates of any physical Hamiltonian. Section 3.3 describes a simple test that can be used to determine if a given state  $|\psi\rangle$  is a physically-plausible eigenstate. Section 3.4 uses the results of Section 3.3 to obtain a visualization of the set of physically-plausible eigenstates. Then, in Section 3.5 we give a more powerful argument constructing quantum states  $|\psi\rangle$  “far away” from all the eigenstates  $|E\rangle$  of any non-trivial,  $L$ -local Hamiltonian.

### 3.1.1 $L$ -local interactions

Let’s begin with a definition of  $L$ -local interactions that is general enough to be applicable to both classical and quantum mechanics:

**Definition 3.1** ( $L$ -local interactions): *A system of  $N \geq L$  bodies contains  $L$ -local interactions when the total energy of the system is expressible as a sum of terms that are each a function of the state of at most  $L$  bodies.*

From this definition, it is apparent that the familiar forces of classical physics are *two*-local. For example, the electrostatic potential energy is given by a sum of terms that are each inversely proportional to the distance between a *pair* of bodies. Likewise, quantum mechanical theories are generally two-local as well. Note that examples of so-called *effective* theories exist where interactions are three- or four-local (see for example [ML04] and references therein). So, although proofs contained later in this chapter are performed for  $L$ -local interactions of arbitrary  $L$ , we shall keep in mind that  $L$  can generally be taken as 2, and sometimes 3 or 4.

For simplicity, we will be assuming for the most part in this chapter that the interacting bodies are all two-level objects (qubits). Our results easily generalize to higher-dimensional objects, as is discussed in Subsection 3.5.3. The most general expression for

a Hamiltonian on  $N$  qubits is:

$$H = \sum_{\sigma} h_{\sigma} \sigma, \quad (3.1)$$

where  $h_{\sigma}$  are real coefficients, and the  $\sigma$  denote all the possible  $N$ -fold tensor products of the Pauli matrices  $I, X, Y, Z$ . For example, for  $N = 2$  qubits,  $\sigma$  takes the value of each of the 16 tensor products  $I \otimes I, I \otimes X, I \otimes Y, \dots, Z \otimes Z$ . The Hamiltonian in Eq. (3.1) is  $L$ -local if and only if  $h_{\sigma}$  is zero for every  $\sigma$  of weight  $> L$ . The *weight* of  $\sigma$  refers to the number of terms in the tensor product that are not  $I$ .

## 3.2 Dimension-counting argument

We now give a counting argument showing that most quantum states cannot arise as an energy eigenstate of any local Hamiltonian. This counting argument has the advantage of simplicity, but also has some significant deficiencies, discussed below and remedied in Section 3.5. Suppose an  $N$ -body quantum system is described by an  $L$ -local Hamiltonian  $H$ , parameterized as in Eq. (3.1). The number of independent real parameters that describe  $H$  is simply the number of the coefficients  $h_{\sigma}$  that are not forced to be zero by  $L$ -locality. We denote this number as  $\#(N, L)$ . The value of  $\#(N, L)$  is given by the number of  $N$ -qubit Pauli products  $\sigma$  that act on  $L$  or less bodies, which can be expressed as

$$\#(N, L) = \sum_{j=0}^L \binom{N}{j} 3^j. \quad (3.2)$$

To see this, note that the different terms in the sum in Eq. (3.2) come from the interactions involving  $j = 0, 1, \dots, L$  bodies, respectively. For the  $j$ -body interactions, there are  $\binom{N}{j}$  ways of picking out a subset of  $j$  interacting systems, and given a particular subset there are  $3^j$  ways of choosing one of  $X, Y$ , or  $Z$  at each of the locations. When  $L \leq N/2$  we obtain a useful upper bound on  $\#(N, L)$  by noting that  $\binom{N}{j} \leq \binom{N}{L}$ , and  $3^j \leq 3^L$ :

$$\#(N, L) \leq (L+1) \binom{N}{L} 3^L. \quad (3.3)$$

For  $L = 2$  and  $L = 3$ , Eq. (3.2) can be expanded to the following polynomials:

$$\#(N, 2) = \frac{9N^2 - 3N + 2}{2}; \quad (3.4)$$

$$\#(N, 3) = \frac{9N^3 - 18N^2 + 15N + 2}{2}. \quad (3.5)$$



More generally, for any fixed  $L$ ,  $\#(N, L)$  is a polynomial of degree  $L$  in  $N$ .

Next, consider the set of states which can be obtained as the non-degenerate ground state<sup>1</sup> of some  $L$ -local Hamiltonian. This set can be parameterized by the  $\#(N, L)$  real parameters of the corresponding Hamiltonian. Since an arbitrary state of  $N$  qubits requires  $2 \times 2^N - 2$  real parameters to specify, then provided  $\#(N, L) < 2 \times 2^N - 2$  we see that there exists a state  $|\psi\rangle$  which cannot arise as the non-degenerate ground state of any  $L$ -local Hamiltonian. Comparing with the bound Eq. (3.3) we see that this is generically the case (except in the case where  $L$  approaches  $N$ , which as discussed earlier is unphysical except for very small  $N$ ). Thus generic quantum states will *not* be the ground state of any non-degenerate  $L$ -local Hamiltonian.

This argument proves the existence of quantum states which are not eigenstates of any non-degenerate,  $L$ -body Hamiltonian. However, there are many deficiencies with the argument. First, the argument only establishes the *existence* of such states, it does not tell us what they are. Second, while the argument shows that such a state cannot be an *exact* eigenstate, it does not provide any limitation on how close it can be to an eigenstate. Indeed, phenomena such as space-filling curves show that a manifold of small dimension can “fill up” a manifold of larger dimension so that every point in the manifold of larger dimension is arbitrarily close to a point in the manifold of smaller dimension. Third, the argument requires the eigenstates to be non-degenerate. This deficiency may be partially remedied by noting that the manifold of states arising as eigenstates of Hamiltonians with up to  $m$ -fold degeneracy is at most  $m \times \#(N, L)$ -dimensional. However, as  $m$  increases, the bound obtained by parameter counting becomes weaker and weaker.

### 3.3 Simple test for physical plausibility

Before we proceed to the stronger argument of Section 3.5, we first consider the question: “given an arbitrary state  $|\psi\rangle$ , how can we test whether there exists some nontrivial  $L$ -local Hamiltonian for which  $|\psi\rangle$  is an eigenstate?”. We shall see that there is a straightforward test, based on techniques from linear algebra, that can be applied to any  $|\psi\rangle$ . The description of this test provides a useful prelude to Section 3.5. Furthermore, it allows us to attempt to visualize the manifold of allowed  $L$ -local eigenstates, by calculating 2D and 3D cross-sections of that manifold. The visualization provides a way of developing some intuition about the set of physically-plausible eigenstates.

---

<sup>1</sup>We use the ground state for concreteness; the argument which follows applies equally to excited states. The fact that a Hamiltonian has many eigenstates will not affect the parameter count: we are counting real parameters, whereas the index to the particular eigenstate is a discrete parameter.

For the purposes of the present subsection, we will use a slightly modified notation as follows. Let the system Hamiltonian be

$$H = \sum_j^m h_j \sigma^{(j)} \quad (3.6)$$

where  $\sigma^{(j)}$ , for  $j=1, \dots, m$ , label the particular Pauli tensor products that are allowed to be nonzero in the Hamiltonian, based on our choice of locality restrictions. So if  $H$  is  $L$ -local then  $m = \#(N, L)$ , and  $\sigma^{(j)}$  enumerate the Pauli tensor products having weight no greater than  $L$ . However, we could just as easily assume some other type of locality constraint, like nearest-neighbour interactions on a grid, and the following analysis would still apply. Note that we require that the  $N$ -qubit identity operator always be included as one of the  $\sigma^{(j)}$ .

Suppose  $|E\rangle$  is an eigenstate of some nontrivial Hamiltonian of the form in Eq. (3.6). Without loss of generality we may assume the corresponding eigenvalue is zero<sup>2</sup>. Thus,  $|E\rangle$  satisfies the eigenvalue equation

$$H|E\rangle = 0|E\rangle \quad (3.7)$$

$$= \vec{0}. \quad (3.8)$$

Substituting in Eq. (3.6) gives

$$\sum_j h_j \sigma^{(j)} |E\rangle = \vec{0}. \quad (3.9)$$

Eq. (3.9) tells us that  $|E\rangle$  is an eigenstate of some nontrivial  $L$ -local Hamiltonian if and only if there exists some nonzero real linear combination of the vectors  $\sigma^{(j)}|E\rangle$  which gives the zero vector. The sum in Eq. (3.9) can be rewritten as matrix multiplication, giving

$$R\vec{h} = \vec{0}, \quad (3.10)$$

where  $\vec{h}$  is the  $m$ -by-1 vector with entries  $h_j$ , and  $R$  is a  $2^N$ -by- $m$  matrix with columns equal to  $\sigma^{(j)}|E\rangle$  as follows:

$$R \equiv \begin{pmatrix} \sigma^{(1)}|E\rangle & \cdots & \sigma^{(m)}|E\rangle \\ \downarrow & \cdots & \downarrow \end{pmatrix}. \quad (3.11)$$

Left-multiplying Eq. (3.10) by its Hermitian conjugate gives the following condition (holding if and only if Eq. (3.10) does):

$$\vec{h}^T R^\dagger R \vec{h} = 0. \quad (3.12)$$

---

<sup>2</sup>If the eigenvalue was  $E \neq 0$ , it could be rescaled to zero by replacing  $H$  by  $H - EI$ , leaving  $H$  still of the form of Eq. (3.6) and nontrivial as required.

Adding Eq. (3.12) to its complex conjugate, gives

$$\vec{h}^T [R^\dagger R + (R^\dagger R)^*] \vec{h} = 0 \quad (3.13)$$

(which holds if and only if Eq. (3.12) does, since both  $R^\dagger R$  and  $(R^\dagger R)^*$  are positive matrices). We can write this equation most simply as follows:

$$\vec{h}^T M \vec{h} = 0, \quad (3.14)$$

where  $M$  is the positive real-symmetric matrix with entries

$$M_{jk} = \text{Re} \langle E | \sigma^{(j)} \sigma^{(k)} | E \rangle. \quad (3.15)$$

Since  $M$  is positive, Eq. (3.14) has a nontrivial solution if and only if  $\det(M) = 0$ . To summarize, we have the following result:

**Result 3.1:** *If  $\sigma^{(j)}$ ,  $j = 1, \dots, m$ , is some collection of distinct tensor products of Pauli matrices on  $N$  qubits, such that the  $N$ -qubit identity operator is included, then a state  $|E\rangle$  is an eigenstate of some nontrivial Hamiltonian of the form  $H = \sum_{j=1}^m h_j \sigma^{(j)}$  if and only if  $\det(M) = 0$ , where  $M_{jk} = \text{Re} \langle E | \sigma^{(j)} \sigma^{(k)} | E \rangle$ . Furthermore, the set of vectors  $\vec{h} = (h_1, \dots, h_m)^T$  which cause  $|E\rangle$  to be a (zero-energy) eigenstate of  $H$  is exactly the nullspace of  $M$ .*

Thus, we have a simple way to tell if a given state  $|E\rangle$  could be the eigenstate of a nontrivial  $L$ -local Hamiltonian (or of some other class of Hamiltonians having a more general locality constraint), and a way of obtaining the set of all such Hamiltonians that yield  $|E\rangle$  as an eigenstate.

Let's briefly consider the question of how practical it is to evaluate the condition described in Result 3.1 – that is, the difficulty of computing the entries  $M_{jk}$  and the determinant  $\det(M)$ . The numerical calculation of an  $m$ -by- $m$  determinant can be computed quickly for values of  $m$  up to several thousand. For example, using MATLAB running on a Pentium-4 2.0GHz processor, a 2000-by-2000 matrix determinant takes approximately 7 seconds to compute. This matrix size corresponds roughly to the case of considering 2-local Hamiltonians on  $N = 21$  qubits. However, calculating the entries of  $M$  is typically more time-consuming than calculating the determinant, when  $|E\rangle$  is specified as an arbitrary  $2^N$ -element complex vector. For example, we found that in order to keep the total calculation time less than one hour it is necessary to restrict the problem size to  $N = 10$ , in the case of 2-local geometry. So, except in the case where  $|E\rangle$  has some special structure that can be exploited (such as for an error-correcting code state, as we

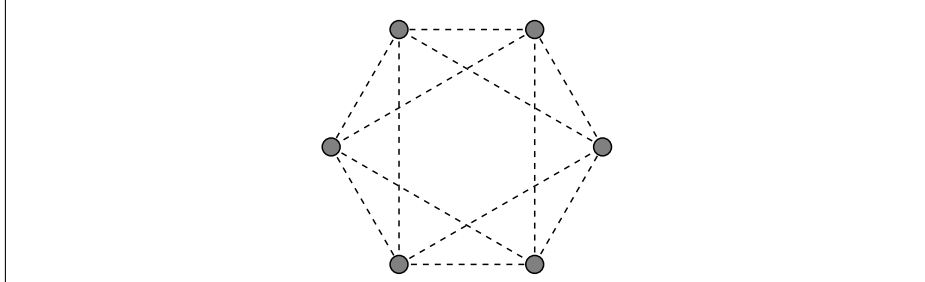


Figure 3.1: In this example, the class of Hamiltonians has arbitrary one and two-qubit interactions, except that qubits opposite each other are forbidden to directly interact.

will consider later in this chapter), Result 3.1 will be difficult to apply for large problem sizes.

Aside from speed of computation, another important practical consideration is the prevention of numerical underflow. We found that for typical examples of states  $|E\rangle$  that do not belong to the set of physically-plausible eigenstates, i.e., for which  $\det(M) \neq 0$ , the value of  $\det(M)$  is nonetheless usually very small, often near the lower limit of precision of the double-precision arithmetic used in MATLAB. Thus, numerical underflow errors can occur if  $\det(M)$  is evaluated directly. Note however that  $\det(M) = 0$  if and only if the smallest eigenvalue of  $M$  is zero, since  $M$  is positive. We found that, unlike the calculation of  $\det(M)$ , calculating the smallest eigenvalue of  $M$  is not prone to numerical problems. This modification will cause the overall computation time to be increased, but only marginally for the problem sizes we considered.

### 3.4 Visualizing the manifold of allowed eigenstates

With the result of Section 3.3 in hand, we now consider how the set of allowed eigenstates can be studied numerically, for a particular class of locality-restricted Hamiltonians. The example class of Hamiltonians we consider are those that are 2-local on 6 qubits, with the further restriction that when the qubits are arranged in a ring, opposite qubits don't directly interact. See Figure 3.1 for a depiction of this allowed interaction geometry. This particular configuration was chosen for two main reasons: (1) the dimensionality of the manifold of eigenstates was found to be only one less than the entire space of states, making it easier to find points on the manifold, and (2) the size of the system was large

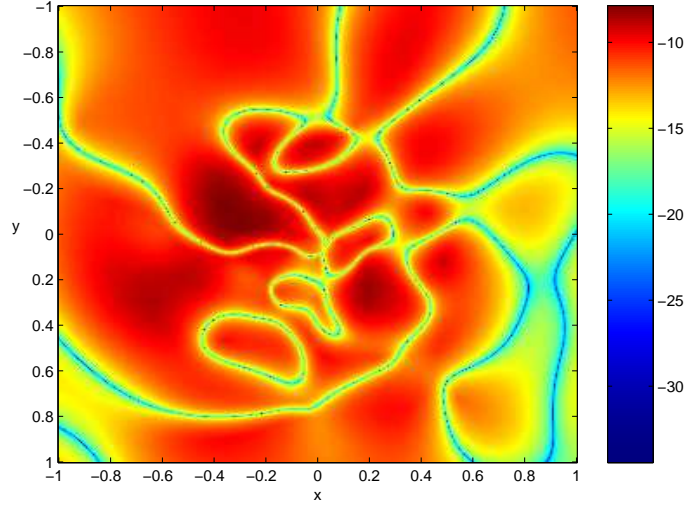


Figure 3.2: The colour scale indicates the logarithm of the smallest eigenvalue of the matrix  $M$  (as defined in Eq. (3.15)) evaluated over a random 2D cross-section of Hilbert space (parameterized by  $x$  and  $y$  as in Eq. (3.16)), for the class of interactions shown in Figure 3.1. The curves coloured by the lower-end of the colour scale thus depict a cross-section of the manifold of allowed eigenstates.

enough to be nontrivial but small enough to do fast calculations with.

The idea is to take a 2D or 3D “slice” through the Hilbert space of  $N$ -qubit states, and see how the manifold of allowed eigenstates intersects with that slice. By “a 2D slice through Hilbert space” we mean as follows. We represent a state in the computational basis, and fix all the coefficients except the coefficient of  $|1, \dots, 1\rangle$ . That one coefficient is allowed to vary, and is parameterized  $x + iy$ , where  $x$  and  $y$  are real. The resulting state is normalized. Thus, points on the slice are parameterized as

$$|\psi(x, y)\rangle = \frac{|\psi^{\text{fixed}}\rangle + (x + iy)|1, \dots, 1\rangle}{\|(|\psi^{\text{fixed}}\rangle + (x + iy)|1, \dots, 1\rangle)\|}. \quad (3.16)$$

Each choice of  $|\psi^{\text{fixed}}\rangle$  will thus define a different slice. In the present example,  $|\psi^{\text{fixed}}\rangle$  is chosen randomly, by selecting each coefficient from a complex zero-mean Gaussian distribution, normalizing the result, and setting the coefficient of  $|1, \dots, 1\rangle$  to zero.

For a range of  $x$  and  $y$ , the matrix  $M$  in Eq. (3.15) is calculated using  $|E\rangle = |\psi(x, y)\rangle$ . Recall that  $|\psi(x, y)\rangle$  is an allowed eigenstate if and only if  $\det(M) = 0$ , or equivalently if and only if the smallest eigenvalue of  $M$  is zero (since  $M$  is positive). In Figure 3.2, the logarithm (base  $e$ ) of the smallest eigenvalue of  $M$  is plotted as a function of  $x$  and  $y$ , for a particular random  $|\psi^{\text{fixed}}\rangle$ . The plot shows a remarkable pattern of curves formed by the points where the smallest eigenvalue of  $M$  approaches zero. (Although the

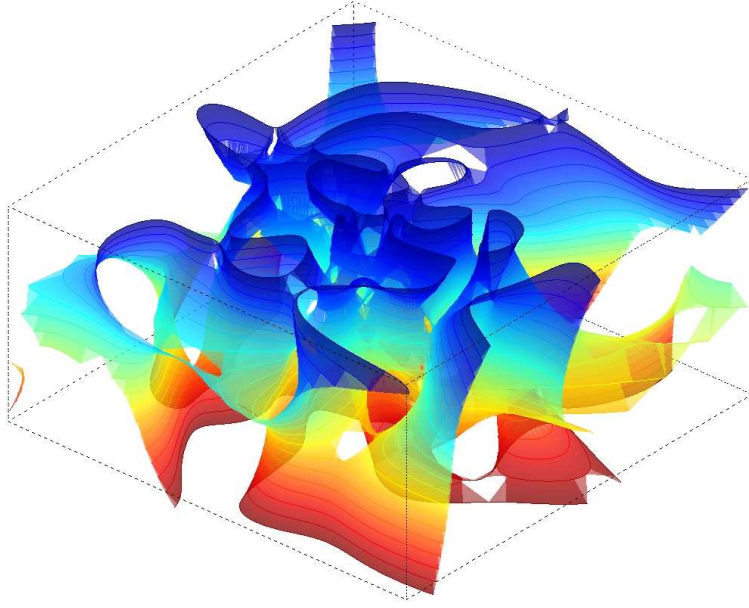


Figure 3.3: A random 3D cross-section of the manifold of allowed eigenstates, for the same example as in Figure 3.2. A colour gradient along the  $z$  axis is used to aid the eye.

minimum value that the smallest eigenvalue of  $M$  attains on this plot is  $\approx e^{-35}$ , sampling at higher resolutions near the curve yielded values approaching arbitrarily close to zero). The example gives a glimpse of how complex the manifold of physically-plausible states is, even for this simple example.

A 3D slice of the manifold can also be plotted.  $|\psi(x, y, z)\rangle$  is defined as  $|\psi(x, y)\rangle$  above but with the real part of the coefficient of  $|1, \dots, 1, 0\rangle$  given by  $z$ . Many 2D slices with different  $z$  values can be calculated, and for each the curve of  $\det(M) = 0$  is found. The curves are joined together between slices and plotted in three dimensions to give Figure 3.3.

### 3.5 Relationship between quantum codes and eigenstates of $L$ -local Hamiltonians

Section 3.2 used a dimension-counting argument to show that “most” states are not physically-plausible eigenstates. A much stronger argument can be made about the set of physically-plausible eigenstates by using the theory of quantum error-correcting codes

(QECCs). We now briefly review the relevant elements of the theory of QECCs, and explain a simple observation motivating the connection between  $L$ -local Hamiltonians and QECC states. Then in Subsection 3.5.2 we develop a stronger quantitative version of the argument.

### 3.5.1 Nondegenerate codes

Recall, degenerate and nondegenerate codes were defined in Subsection 2.2.2.4 of this thesis. In the present subsection, we shall briefly review the properties of nondegenerate QECCs relevant to the later proof, and discuss some of the known results regarding the existence of such codes.

A code encoding  $k$  qubits into  $N$  qubits is a  $2^k$ -dimensional subspace of the  $2^N$ -dimensional state space of  $N$  qubits. It is convenient to give the code space a label,  $V$ . A nondegenerate code is said to correct  $t$  errors if the subspaces  $\sigma V$  are all orthogonal to one another, for the set of  $N$ -qubit Pauli tensor products  $\sigma$  of weight  $\leq t$ . Thus if an error consisting of an unknown Pauli operation of weight  $\leq t$  occurs to a state in the code space, it is possible to perform a measurement to determine which subspace  $\sigma V$  the state has moved to, and thus which error  $\sigma$  occurred, and then apply the operator  $\sigma$  to correct the state.

Note that it is possible to find codes not of this type. In particular, for a class of codes known as *degenerate codes* correcting  $t$  errors, different errors  $\sigma$  and  $\sigma'$  of weight  $\leq t$  may have *identical* effects on the codespace (and thus be indistinguishable from each other) yet still be correctable.

Since we will be using the properties of nondegenerate codes of various lengths  $N$  to prove something about eigenstates of  $N$ -body  $L$ -local Hamiltonians, we should pause to consider the question: do codes that have such properties exist, for a range of different  $N$  and  $L$ ? In fact, there are many useful bounds on the existence of nondegenerate codes, and we now describe an example of such. The “quantum Gilbert-Varshamov bound” shows that a nondegenerate code encoding  $k$  qubits into  $N$  qubits, and correcting errors on up to  $t$  qubits, exists whenever<sup>3</sup>:

$$\#(N, 2t) < \frac{2^{2N} - 1}{2^{N+k} - 1} \quad (3.17)$$

---

<sup>3</sup>A different form of the Gilbert-Varshamov bound was originally stated in [EM96, Got97]. Gottesman [Got] points out that the earlier bound requires a slight correction, which we have given here [Got03]. This correction makes little difference to our results, and is mentioned only for complete accuracy.

In the limit of large  $N$  this becomes [CRSS97]

$$\frac{k}{N} < 1 - H\left(\frac{2t}{N}\right) - \frac{2t}{N} \log(3), \quad (3.18)$$

where  $H(x) \equiv -x \log(x) - (1-x) \log(1-x)$  is the binary entropy, and all logarithms are taken to base 2.

The Gilbert-Varshamov bound applies even when  $k = 0$ . Thus there exists a 1-dimensional quantum code — that is, a quantum state,  $|\psi\rangle$  — such that the states  $\sigma|\psi\rangle$  are all orthogonal to one another. This is true for  $\sigma$  up to weight  $t$  for any  $t$  satisfying

$$\#(N, 2t) < \frac{2^{2N} - 1}{2^N - 1}. \quad (3.19)$$

In the large  $N$  limit, this becomes  $t/N < 0.0946$ . Summarizing, the quantum Gilbert-Varshamov bound tells us that there exists a quantum state  $|\psi\rangle$  such that the states  $\sigma|\psi\rangle$  form an orthonormal set for  $\sigma$  of weight at most  $t$ , for any  $t$  satisfying  $\#(N, t) < (2^{2N} - 1)/(2^N - 1)$ .

### 3.5.2 Nondegenerate codes are “far away” from all plausible eigenstates

Let us return to the problem of Hamiltonians and eigenstates. Recall from Section 3.3 (particularly Eq. (3.9)) that a state  $|\psi\rangle$  is an eigenstate of a nontrivial  $L$ -local Hamiltonian if and only if some nonzero real linear combination of the vectors  $\sigma|\psi\rangle$ , for  $\sigma$  having weight  $\leq L$ , gives the zero vector. But we have just seen in Subsection 3.5.1 that if  $|\psi\rangle$  belongs to a nondegenerate QECC code correcting  $L$  errors then the vectors  $\sigma|\psi\rangle$  will be orthogonal, and so no nonzero linear combination (real or complex) of those vectors will give the zero vector. Thus we have the following result:

**Result 3.2:** *A state  $|\psi\rangle$  belonging to a nondegenerate quantum error-correcting code correcting  $L$  errors is not an eigenstate of any nontrivial  $L$ -local Hamiltonian.*

The above argument addresses two of the problems with the parameter counting argument. Namely, finding concrete examples of states  $|\psi\rangle$  that cannot be exact eigenstates of  $L$ -local Hamiltonians, and dealing with degeneracies in  $H$ . However, it leaves the most significant problem open, namely, proving bounds on how close  $|\psi\rangle$  can be to an eigenstate of  $H$ . Remarkably, the answer turns out to be “not very”, as we now prove.

Suppose an  $N$ -body  $L$ -local quantum system is described by a non-trivial Hamiltonian  $H$ . We suppose  $H$  acts on qubits; the extension to other systems is straightforward.



Suppose  $|E\rangle$  is any energy eigenstate for the system, with corresponding energy  $E$ , and let  $H' \equiv H - EI$  be a rescaled Hamiltonian such that  $|E\rangle$  has energy 0. Note that  $H' = \sum_{\sigma} h'_{\sigma} \sigma$ , where  $h'_I = h_I - E$ , and  $h'_{\sigma} = h_{\sigma}$  for all other  $\sigma$ . Let  $|\psi\rangle$  be a state such that  $\sigma|\psi\rangle$  forms an orthonormal set for  $\sigma$  of weight up to  $L$ , such as a nondegenerate QECC state correcting  $L$  errors. We use the operator norm defined as  $\|A\| \equiv \max_{|\phi\rangle: \langle\phi|\phi\rangle=1} \|A|\phi\rangle\|$ . Now, we have

$$\|H'(|\psi\rangle - |E\rangle)\| \leq \|H'\| \|\psi\rangle - |E\rangle\|. \quad (3.20)$$

Substituting  $H'|E\rangle = 0$ , we obtain:

$$\|\psi\rangle - |E\rangle\| \geq \frac{\|H'|\psi\rangle\|}{\|H'\|}. \quad (3.21)$$

We can assume  $\|H'\| \neq 0$ , since we have assumed that  $H$  is non-trivial, i.e., it is not a scalar multiple of the identity. Now, since the states  $\sigma|\psi\rangle$  are orthonormal for all  $\sigma$  with weight at most  $L$ , we see that:

$$\|H'|\psi\rangle\| = \sqrt{\sum_{\sigma} h_{\sigma}^2} \quad (3.22)$$

$$= \|\vec{h}'\|_2, \quad (3.23)$$

where  $\|\cdot\|_2$  is the Euclidean, or  $l_2$ , norm of a vector. Furthermore, by the triangle inequality,

$$\|H'\| \leq \sum_{\sigma} |h'_{\sigma}| \|\sigma\| = \sum_{\sigma} |h'_{\sigma}| \quad (3.24)$$

$$= \|\vec{h}'\|_1, \quad (3.25)$$

where  $\|\cdot\|_1$  denotes the  $l_1$  norm of a vector, i.e., the sum of the absolute value of the components, and where we used the fact that  $\|\sigma\| = 1$ . Substituting Eqs. (3.23) and (3.25) into Eq. (3.21), we obtain

$$\|\psi\rangle - |E\rangle\| \geq \frac{\|\vec{h}'\|_2}{\|\vec{h}'\|_1}. \quad (3.26)$$

Now, the Cauchy-Schwarz inequality,  $|\vec{x} \cdot \vec{y}| \leq \|\vec{x}\|_2 \|\vec{y}\|_2$ , can be used to show the following general relationship between the  $l_1$  and  $l_2$  norms of an  $m$ -dimensional vector:

$$\|\vec{x}\|_1 \leq \|\vec{x}\|_2 \sqrt{m}. \quad (3.27)$$

Thus,

$$\|\vec{h}'\|_1 \leq \sqrt{\#(N, L)} \|\vec{h}'\|_2, \quad (3.28)$$

which, combined with Eq. (3.26), gives the general bound

$$\| |\psi\rangle - |E\rangle \| \geq \frac{1}{\sqrt{\#(N, L)}}. \quad (3.29)$$

Applying Eq. (3.3) gives us

$$\| |\psi\rangle - |E\rangle \| \geq \left[ (L+1) \binom{N}{L} 3^L \right]^{-1/2}. \quad (3.30)$$

Eq. (3.30) provides a constant lower bound on the distance of  $|\psi\rangle$  from any energy eigenstate  $|E\rangle$  of  $H$ , completely independent of any details about  $H$ , other than the fact that it is a non-trivial,  $L$ -local Hamiltonian, acting on  $N$  qubits.

### 3.5.3 Extension to $d$ -dimensional bodies

The result in Eq. (3.29) carries over directly to systems of interacting *qudits* (i.e.,  $d$ -dimensional quantum objects), provided the operator basis  $\sigma$  we expand in is unitary. The only difference is that the value of  $\#(N, L)$  is somewhat larger for qudit systems. In particular, for  $d$ -dimensional systems the bound on  $\#(N, L)$  in Eq. (3.3) is modified as follows:

$$\#(N, L) \leq (L+1) \binom{N}{L} (d^2 - 1)^L. \quad (3.31)$$

Combining Eq. (3.29) with Eq. (3.31), we may summarize with the following result:

**Result 3.3:** *Let  $H$  be a non-trivial  $L$ -local Hamiltonian acting on  $N$   $d$ -dimensional objects, and let  $|E\rangle$  be an eigenstate of  $H$ . Let  $|\psi\rangle$  be a state such that the states  $\sigma|\psi\rangle$  are orthonormal for all  $\sigma$  of weight up to  $L$ . (For example,  $|\psi\rangle$  might be a nondegenerate QECC correcting errors on up to  $L$  qubits.) Then the following chain of inequalities holds:*

$$\| |\psi\rangle - |E\rangle \| \geq \frac{1}{\sqrt{\#(N, L)}} \quad (3.32)$$

$$\geq \left[ (L+1) \binom{N}{L} (d^2 - 1)^L \right]^{-1/2}. \quad (3.33)$$

It should be mentioned that the bound in Eq. (3.33) becomes trivial in the limit of very large  $N$ , i.e. for macroscopic systems. We speculate that in that limit there would not exist a state far away from eigenstates of all  $L$ -local Hamiltonians,  $L \geq 2$ .

### 3.5.4 Discussion

We have seen that nondegenerate QECC states are interesting examples of states that cannot be close to any eigenstate of any nontrivial  $L$ -local Hamiltonian. A corollary of this result is that no nontrivial  $L$ -local Hamiltonian has a *ground state* (minimum-energy state) which is close to a nondegenerate QECC state. Understanding the ground state of a quantum system is of particular interest because the ground state is the only thermal-equilibrium state which can be “pure” (that is, noise free). Superconductivity and superfluidity are just two examples of important phenomena which occur as a result of the special properties of quantum ground states. In quantum information processing, cooling a system to its ground state is central to a number of procedures, including state preparation, adiabatic quantum computation, and natural fault tolerance. What implications do our results have for such procedures?

Our results show that there are severe limitations to which quantum codes may be used in a naturally fault-tolerant device. If an  $N$ -qubit naturally fault-tolerant device is constructed using  $L$ -local interactions, then it is impossible to arrange the system so that its ground state is equal to an  $N$ -qubit nondegenerate QECC that corrects  $L$  errors. This is quite a restriction, since nondegenerate QECCs include the important class of CSS codes. Thus, the degeneracy of the quantum codes appearing in proposals such as [Kit97a] and [Bac05] is not a coincidence, but rather an essential feature necessary for the system to be naturally resilient to multiple errors.

Likewise, suppose one did not wish to create a full naturally fault-tolerant system, but simply to use cooling as a way of preparing a state (for example, preparing the encoded ancilla states for Steane’s active error-correction scheme). Our results show that there are significant restrictions on how nondegenerate code states can be prepared in this way, a fact that is somewhat surprising in light of the fact that such states can be efficiently prepared by a circuit-based quantum computer. Thus it would appear that cooling, as a procedure for state preparation, is much less powerful than the ability to perform quantum gates. This would seem to contradict the known equivalence [KKR04] between the circuit-model of quantum computing and the model of adiabatic quantum computing using 2-local interactions. (In the model of adiabatic quantum computing, the system is kept at its ground state at all times). However, our results are not in fact directly applicable to adiabatic quantum computing, since in the adiabatic model many auxiliary qubits are added which are not present in the equivalent quantum circuit. Our results only apply when an  $N$ -qubit system is being used to create an  $N$ -qubit code state (that is, the same “ $N$ ” for the size of the system and code).

Thus, the possibility is left open for a nondegenerate code correcting  $L$  errors to be (near to) the eigenstate of a  $L$ -local Hamiltonian, so long as auxiliary qubits are used. In practice, how might that work? One possibility would be to use the technique of *effective interactions*, whereby  $L$ -local interactions can be made to behave (approximately) like interactions of weight  $> L$ , by using auxiliary qubits that interact in some appropriate way. (See Section 6.2 of [KKR04] for an example of effective interactions used in a quantum information theory context.) Further work is needed to study how effective interactions might be useful in naturally fault tolerant systems.

### 3.6 Conclusion

To conclude, we have found interesting examples of quantum states far from the eigenstates of any non-trivial  $L$ -local Hamiltonian. Surprisingly, the states we construct can still be prepared efficiently on a quantum computer. Our construction has implications for the physics of locally interacting many-body systems, and for the theory of naturally fault-tolerant systems for quantum information processing.

# State encoding for quantum communication over spin systems

---

This chapter considers the following question: can a network of interacting quantum spins be used to perform some useful quantum information processing task, when the amount of external control that can be applied to the network is extremely limited? In particular we focus on a recent proposal for using a simple chain of interacting spins as a medium for high-fidelity quantum communication. The idea is that such a chain could act as a “quantum wire” in future quantum devices. In general, the natural dynamics of a spin chain yields a very poor-fidelity channel – even when there is no external noise. We present a general scheme for significantly increasing the communication fidelity, by using a very simple form of external control on parts of the spin chain. The derivation of the control scheme is based on techniques of quantum state encoding.

Our scheme for quantum communication is applicable to any spin system having interactions that conserve  $z$ -spin. The sender and receiver are assumed to directly control several spins each, and the sender encodes the message state onto the larger state-space of her control spins. We show how to find the encoding that maximizes the fidelity of communication, using a simple method based on the singular-value decomposition. Then we show that this solution can be used to increase communication fidelity in a rather different circumstance: where no encoding of initial states is used, but where the sender and receiver control exactly two spins each and vary the interactions on those spins over time. The methods presented are computationally efficient, and numerical examples are given for systems having up to 300 spins.

The results in this chapter demonstrate that quantum coding is not only useful for suppressing random noise, but also for helping control the dynamics of a noise-free system. In fact, we do not consider the effects of external noise at all in this chapter. This fact is an important contrast to the other research chapters in this thesis, which deal with designing systems that are reliable against the effects of noise. As a result, the codes

constructed in this chapter are rather different from the standard error-correcting codes considered in other parts of this thesis.

*NOTE: This chapter is based on a published journal article, [H. L. Haselgrove, Phys. Rev. A 72:062326 (2005)]. The text, maths, and figures of that article are reproduced in this chapter with only minor changes.*

## 4.1 Introduction

Quantum communication, the transfer of a quantum state from one place or object to another, is an important task in quantum information science[DiV00]. The problem of communicating quantum information is profoundly different to the classical case[Pre98, NC00]. For example, quantum communication could not possibly be achieved by just measuring an unknown state in one place, and reconstructing in another. Rather, an entire system of source, target, and medium must evolve in a way that maintains quantum coherence.

In this chapter we consider an idealized system of interacting spin-1/2 objects, isolated from the environment. The aim is to use the system's natural evolution to communicate a qubit state from one part of the system to another. The motivation is that such a system could be used as a simple “quantum wire” in future quantum information-processing devices. The most obvious configuration to choose is a simple one-dimensional open-ended chain, with interactions between nearest-neighbour spins, in which case we want the chain's evolution to transfer a qubit state from one end to the other. The methods in this chapter apply to this simple type of chain, and also to spin networks on arbitrary graphs.

A number of interesting proposals exist for quantum communication through spin chains. In [Bos03], the 1D “Heisenberg” chain was considered, with coupling strengths constant over the length of the chain and with time. The idea was to initialize all spins in the “down” state, except the first spin, which was given the state of the qubit to be sent. After the system was allowed to evolve, the spin at the far end of the chain would then contain the sent state, to some level of fidelity. Simulations were carried out for a range of chain lengths, and it was shown that the fidelity was high only for very small chains.

In [CDEL04], a 1D spin chain with so-called “XY” couplings was considered. Here, the coupling strengths were constant over time, but were made to vary over the length of the chain in a specific way. Like [Bos03], the first spin was initialized in the state to be sent, with all other spins initialized to “down”. It was shown that this scheme allows a

*perfect* state transfer to the far spin site, for any length of chain.

In [OL04], a scheme was presented for high-fidelity quantum communication over a *ring* of spins with nearest-neighbour Heisenberg couplings, using coupling strengths constant over the length of the ring and over time. The sender and receiver are located diametrically opposite to one another. The authors showed that excitations travel around the ring in a way that can be described using a concept from classical wave theory, the dispersion relation. Using this insight, they constructed a scheme where the sender, who controls several adjacent spins, constructs an initial state that is a Gaussian pulse having a particular group velocity chosen to minimize the broadening of the pulse over time. Using this state for the encoding of the  $|1\rangle$  basis of the qubit message, and the all-down state as the encoding of  $|0\rangle$ , an arbitrary qubit can be sent with high fidelity over rings of any size, so long as the number of spins that the sender controls is at least the cube root of the total number.

Motivated by the results in [OL04], we pose the following problem. Say we are given the Hamiltonian for a system of interacting spins, where the graph of the interactions is not necessarily a ring structure, but is completely arbitrary. Also, the strength and type of interaction along each graph edge is arbitrary (so long as total  $z$  spin is conserved). The sender Alice controls some given subset of the spins, and the receiver Bob controls some other given subset. How does Alice encode the qubit to be sent onto the spins she controls, in order to maximize the fidelity of communication? We know from [OL04] that the Gaussian pulse provides a near-optimal fidelity for the case of a Heisenberg ring (and is optimal in the limit of large ring sizes). What about other shapes of spin network? Can we find a general solution?

We provide a simple and efficient method for finding the maximum-fidelity encoding of the  $|1\rangle$  message basis state, for a general  $z$ -spin-conserving spin system. (We assume that the encoding for the  $|0\rangle$  basis state is fixed to the all-down state). So, unlike the schemes in [Bos03], [CDEL04], and [OL04], which use systems with interactions that have specific strengths and conform to a specific graph, our scheme is designed to “make the most” of whatever arbitrary system is given to us. We give a numerical example of our method, for a system of 300 spins (where Alice and Bob each control 20 spins), showing a near-perfect average fidelity.

We give a second scheme for increasing fidelity, that does not use encoding of initial states, but relies on Alice and Bob dynamically controlling the interactions on their control spins. Here, the number of control spins is fixed at two each for Alice and Bob. We give a straightforward method for deriving control functions, that give a fidelity (and

communication time) equal to the values that would result if Alice and Bob had instead each controlled many more spins (with static interactions) and used the optimal initial-state encoding scheme. This method has the combined benefits of being applicable to arbitrary  $z$ -spin-conserving spin-chains, yet having a fixed two-spin “interface” with Alice and Bob. We give numerical examples, and plot the derived control functions, for a 104-spin and a 29-spin system, showing a near-perfect fidelity in each case.

In the remainder of this introductory section, we briefly describe the assumptions behind our schemes, and define our notation. Section 4.2 describes our method of deriving the optimal message encoding. Section 4.3 describes our scheme for increasing fidelity via dynamic control. Concluding remarks are made in Section 4.4.

### 4.1.1 Assumptions and notation

Each of the systems considered in this chapter consist of a collection of interacting spin-1/2 objects. For convenience, we mainly use qubit notation rather than the equivalent spin notation. So the “down state”  $|\downarrow\rangle$  and “up state”  $|\uparrow\rangle$  are equated with the computational basis states  $|0\rangle$  and  $|1\rangle$  respectively. For the spin operators along the  $x$ ,  $y$  and  $z$  axes, we use the Pauli  $X$ ,  $Y$ , and  $Z$  operators, as used elsewhere in this thesis and defined in the List of Notation.

The solution presented in this chapter relies on two main assumptions, which we now list. Firstly, the system Hamiltonian must commute with  $Z^{\text{tot}}$ , which we define to be the  $z$ -component of the total spin operator

$$\vec{\sigma}^{\text{tot}} \equiv (X^{\text{tot}}, Y^{\text{tot}}, Z^{\text{tot}}) \equiv \sum_j \vec{\sigma}_j, \quad (4.1)$$

where  $\vec{\sigma}_j$  is the vector of Pauli operators ( $X, Y, Z$ ) acting on the  $j$ -th spin. Secondly, the spin system must be initialized to the all-down state  $|\downarrow\rangle \otimes \cdots \otimes |\downarrow\rangle$ , before the communication is carried out. Note that the schemes in [Bos03],[CDEL04] and [OL04] also make use of these two assumptions. The first condition is satisfied by any system whose interactions are invariant under rotations about the  $z$  axis. Of course, any external magnetic field will destroy this invariance, unless the field lines are parallel to the  $z$  axis. The Heisenberg and  $XY$  interactions are examples of interactions that conserve  $Z^{\text{tot}}$ . The second constraint might be rather difficult to achieve in practice. One possibility would be to initially apply a strong polarizing magnetic field in the  $z$  direction, over the entire system, and let the system relax to its ground state.

Note that in a system of  $N$  qubits, the  $2^N$  different computational basis states are all eigenstates of  $Z^{\text{tot}}$ , where the eigenvalue has one of  $N + 1$  possible values, given by



the number of  $|0\rangle$ s minus the number of  $|1\rangle$ s. So the state-space of  $N$  spin-1/2 objects can be broken into  $N + 1$  subspaces of different well-defined  $z$ -component of total spin. We use  $\mathcal{H}^{(n)}$ ,  $n = 0, \dots, N$ , to denote these subspaces. That is,  $\mathcal{H}^{(n)}$  is defined to be the eigenspace of  $Z^{\text{tot}}$  spanned by the  $\binom{N}{n}$  computational basis states having  $n$  qubits in the  $|1\rangle$  state and the rest in the  $|0\rangle$  state.

Since the system Hamiltonian  $H$  commutes with  $Z^{\text{tot}}$ , a state in  $\mathcal{H}^{(n)}$  will remain in  $\mathcal{H}^{(n)}$  under the evolution of  $H$ .  $\mathcal{H}^{(0)}$  is one-dimensional; it is spanned by the all-zero state  $|0\rangle \otimes \dots \otimes |0\rangle$ . So this state is a stationary state of  $H$ .

## 4.2 The optimal encoding scheme

Say that Alice wishes to send the qubit state  $\alpha|0\rangle + \beta|1\rangle$ . In our scheme, she does so by preparing the state  $\alpha|\mathbf{0}\rangle_A + \beta|1_{ENC}\rangle_A$ , where  $|\mathbf{0}\rangle_A$  is the all-zero state on her spins, and  $|1_{ENC}\rangle_A$  is some state orthogonal to  $|\mathbf{0}\rangle_A$  (the “ENC” stands for “encoded”). (Note that Alice doesn’t necessarily know  $\alpha$  and  $\beta$ . She would presumably prepare the state by some unitary operation acting on her spins and some external spin containing the state  $\alpha|0\rangle + \beta|1\rangle$ .) We assume that the entire spin chain is initialized to the all-zero state, so immediately after Alice prepares the abovementioned state on her spins, the state of the whole system is

$$|\Psi(0)\rangle \equiv (\alpha|\mathbf{0}\rangle_A + \beta|1_{ENC}\rangle_A) \otimes |\mathbf{0}\rangle_{\bar{A}}, \quad (4.2)$$

where  $\bar{A}$  refers to all spins that Alice does not control. The whole spin system is allowed to evolve for a time  $T$ , giving the state  $|\Psi(T)\rangle = e^{-iHT}|\Psi(0)\rangle$ . Using the fact that  $|0\rangle \otimes \dots \otimes |0\rangle$  is a stationary state,  $|\Psi(T)\rangle$  can be written (up to some global phase) as

$$|\Psi(T)\rangle = \beta\sqrt{1 - \mathcal{C}_B(T)}|\eta(T)\rangle + |\mathbf{0}\rangle_{\bar{B}}(\alpha|\mathbf{0}\rangle_B + \beta\sqrt{\mathcal{C}_B(T)}|\gamma(T)\rangle_B), \quad (4.3)$$

for some nonnegative  $\mathcal{C}_B(T)$ , some normalized  $|\gamma(T)\rangle_B$  orthogonal to  $|\mathbf{0}\rangle_B$ , and for some normalized  $|\eta(T)\rangle$  that is orthogonal to all states of the form  $|0\rangle_{\bar{B}} \otimes |v\rangle_B$ .

We now show that  $\mathcal{C}_B(T)$  can be used as a measure of success. Comparing Eqs. (4.2) and (4.3), we see that  $\mathcal{C}_B(0) = 0$ . If  $\mathcal{C}_B(T)$  reaches 1 for some later  $T$ , a perfect-fidelity quantum communication has resulted. This is because Bob will then have the state  $\alpha|\mathbf{0}\rangle_B + \beta|\gamma\rangle_B$  on the qubits he controls, which can be “decoded” by a unitary operation into the state  $\alpha|0\rangle + \beta|1\rangle$  of a single spin, since  $|\mathbf{0}\rangle_B$  and  $|\gamma\rangle_B$  are orthogonal. If  $\mathcal{C}_B(T)$  is less than 1, the unitary decoding by Bob will leave him with a qubit state  $\rho$  that is generally different to the message state. That is, the measure of state fidelity

$F \equiv (\alpha|0\rangle + \beta|1\rangle)^\dagger \rho (\alpha|0\rangle + \beta|1\rangle)$  between the message  $\alpha|0\rangle + \beta|1\rangle$  and  $\rho$ , will generally be less than one whenever  $\mathcal{C}_B(T) < 1$ . However, the value of  $F$  is highly dependent on the message state — for example, if  $\alpha = 1$  then  $F = 1$  regardless of the value of  $\mathcal{C}_B(T)$ .

$\mathcal{C}_B(T)$ , on the other hand, is a message-independent measure of the fidelity of communication. Consider  $\bar{F}$ , defined to be the state fidelity  $F$  averaged over all message states. For encodings  $|1_{ENC}\rangle$  that belong to the  $\mathcal{H}^{(1)}$  subspace, we have

$$\bar{F} = \frac{1}{2} + \frac{1}{3}\sqrt{\mathcal{C}_B(T)} + \frac{1}{6}\mathcal{C}_B(T), \quad (4.4)$$

which is a monotonic function of  $\mathcal{C}_B(T)$  [OL04]. So, in this case maximizing the *average* state fidelity is equivalent to maximizing  $\mathcal{C}_B(T)$ . More generally, for  $|1_{ENC}\rangle$  not in  $\mathcal{H}^{(1)}$ , the expression in Eq. (4.4) provides a reasonably tight lower bound on  $\bar{F}$ :

$$\frac{1}{2} + \frac{1}{3}\sqrt{\mathcal{C}_B(T)} + \frac{1}{6}\mathcal{C}_B(T) \leq \bar{F} \leq \frac{1}{2} + \frac{1}{3}\sqrt{\mathcal{C}_B(T)} + \frac{1}{6}. \quad (4.5)$$

A proof of this bound may be found in Appendix 4.5. The precise value of  $\bar{F}$  will then depend on  $|\eta(T)\rangle$  and the full specification of Bob's decoding unitary.

A further argument for using  $\mathcal{C}_B(T)$  as a measure of communication fidelity comes from considering the system's ability to transfer *quantum entanglement* from Alice to Bob. Suppose that Alice, instead of sending a message which is a pure quantum state  $\alpha|0\rangle + \beta|1\rangle$ , sends a state which is maximally entangled with some additional spin that Alice possesses. (The additional spin does not interact when the system evolves). If the communication is perfect, the result must be that Bob's decoded message becomes maximally entangled with Alice's additional spin. So more generally, when the communication is not perfect, the amount of entanglement generated between Alice and Bob would be a good measure of communication fidelity. In fact, the entanglement generated, measured by the *concurrence*, is equal to  $\sqrt{\mathcal{C}_B(T)}$  (a proof of this fact is outlined in Appendix 4.6). This is independent of  $|\eta(T)\rangle$ , or the full specification of Bob's decoding unitary, or whether  $|1\rangle_{ENC}$  belongs to  $\mathcal{H}^{(1)}$ .

To recap, when the Hamiltonian commutes with  $Z^{\text{tot}}$  and the state is initialized to  $|\mathbf{0}\rangle$ , the problem of achieving a high communication fidelity can be boiled down to choosing an appropriate initial encoding  $|1_{ENC}\rangle$  for the  $|1\rangle$  qubit basis state. We seek a state  $|1_{ENC}\rangle_A \otimes |\mathbf{0}\rangle_{\bar{A}}$  that has the property that it evolves to (or near to) a state of the form  $|\mathbf{0}\rangle_B \otimes |\gamma\rangle_B$ , or in other words such that  $\mathcal{C}_B(T) \approx 1$  for some  $T$ . Alice's choice for the “encoding” of the  $|0\rangle$  qubit basis state is fixed to the all-zero state. With perfect fidelity that basis state will evolve to the all-zero state on Bob's spins. (Note that in some cases it may be possible to increase fidelity further by allowing an encoding for  $|0\rangle$  other than

the all-zero state. We ignore such a possibility, in order to keep the method for finding the encoding simple and efficient. The simplification is used likewise in [OS04].)

We now show that the encoding  $|1_{ENC}\rangle$  which maximizes  $\mathcal{C}_B(T)$  for a given  $T$  can be found by performing the singular value decomposition on a modified version of the evolution matrix  $e^{-iHT}$ .

Let  $\mathcal{A}$  be the vector subspace of states of the form  $|1_{ENC}\rangle_A \otimes |\mathbf{0}\rangle_{\bar{A}}$ , such that  ${}_A\langle\mathbf{0}|1_{ENC}\rangle_A = 0$ . Similarly, let  $\mathcal{B}$  be the vector subspace of states of the form  $|\mathbf{0}\rangle_{\bar{B}} \otimes |\gamma\rangle_B$ , such that  ${}_B\langle\mathbf{0}|\gamma\rangle_B = 0$ . In other words,  $\mathcal{A}$  reflects all the possible encodings that Alice could use for the  $|1\rangle$  qubit basis state (regardless of the fidelity they would achieve).  $\mathcal{B}$  is the set of states that we would like some state in  $\mathcal{A}$  to evolve to; a state in  $\mathcal{A}$  that evolves to one in  $\mathcal{B}$  represents an encoding for  $|1\rangle$  that gives  $\mathcal{C}_B(T) = 1$  and thus a perfect average fidelity.

Let  $P_{\mathcal{A}}$  and  $P_{\mathcal{B}}$  be the projectors onto the subspaces  $\mathcal{A}$  and  $\mathcal{B}$ . Let  $U(T) \equiv e^{-iHT}$  be the time-evolution operator. From Eqs. (4.2) and (4.3), we can write

$$\mathcal{C}_B(T) = \|P_{\mathcal{B}}U(T)|1_{ENC}\rangle_A \otimes |\mathbf{0}\rangle_{\bar{A}}\|^2, \quad (4.6)$$

where  $\|\cdot\|$  denotes the  $l_2$ -norm. This means that for a particular total communication time  $T$ , choosing the optimal initial encoding for the  $|1\rangle$  state is a matter of finding the normalized  $|\psi\rangle \in \mathbb{C}^{2^N}$  that maximizes  $\|P_{\mathcal{B}}U(T)P_{\mathcal{A}}|\psi\rangle\|$ . The maximum value is given by the largest singular value of  $\tilde{U}(T) \equiv P_{\mathcal{B}}U(T)P_{\mathcal{A}}$ , and the corresponding optimal  $|\psi\rangle$  is the first right-singular-vector of  $\tilde{U}(T)$  [HJ91]. Recall, the SVD (singular value decomposition) of  $\tilde{U}(T)$  is

$$\begin{aligned} \tilde{U}(T) &= VSW^\dagger \\ &= \begin{pmatrix} \vec{v}_1 & \vec{v}_2 & \cdots \\ \downarrow & \downarrow & \cdots \end{pmatrix} \begin{pmatrix} s_1 & & \\ & s_2 & \\ & & \ddots \end{pmatrix} \begin{pmatrix} \vec{w}_1^* \rightarrow \\ \vec{w}_2^* \rightarrow \\ \vdots \end{pmatrix}, \end{aligned} \quad (4.7)$$

where  $s_1 \geq s_2 \geq \cdots \geq 0$  are the *singular values*, the orthonormal  $\vec{w}_j$  are the *right singular vectors*, and the orthonormal  $\vec{v}_j$  are the *left singular vectors* of  $\tilde{U}(T)$ . Numerical packages such as MATLAB have in-built routines for easily calculating the SVD. So, we have that  $\mathcal{C}_B(T)$  has its maximum value,  $s_1^2$ , when Alice chooses the initial state  $|1_{ENC}\rangle_A \otimes |\mathbf{0}\rangle_{\bar{A}} = \vec{w}_1$  to encode  $|1\rangle$ .

Other parts of the decomposition could be useful as well. Say Alice wants to transmit two qubits *simultaneously* to Bob. If she uses the all-down state to encode the  $|00\rangle$  basis state, then she should use the encodings  $\vec{w}_1$ ,  $\vec{w}_2$ , and  $\vec{w}_3$  for the other three basis states

$|01\rangle$ ,  $|10\rangle$ , and  $|11\rangle$ . Then, so long as  $s_3 \approx 1$ , the two qubits would be simultaneously communicated with high fidelity.

The vectors  $\vec{w}_j$  and the values  $s_j$  are also the eigenvectors and square-root eigenvalues respectively of  $(P_B \tilde{U}(T) P_A)^\dagger P_B \tilde{U}(T) P_A = P_A \tilde{U}^\dagger(T) P_B \tilde{U}(T) P_A$ . Now,  $Z^{\text{tot}}$  commutes with  $P_A \tilde{U}^\dagger(T) P_B \tilde{U}(T) P_A$  because it commutes with each of  $P_A$ ,  $P_B$ ,  $\tilde{U}(T)$ , and  $\tilde{U}^\dagger(T)$  separately. So the  $\vec{w}_j$  will all have well-defined total  $Z$  spin (or can be chosen to, wherever ambiguities exist because of degeneracies in the  $s_j$ ). This is important when it comes to calculating these solutions efficiently. Instead of performing the full  $2^N$  by  $2^N$  matrix exponential and SVD, the calculation can be done separately for each of the smaller subspaces  $\mathcal{H}^{(n)}$ , starting each calculation with the  $\binom{N}{n}$ -by- $\binom{N}{n}$  part of the Hamiltonian that acts on the  $\mathcal{H}^{(n)}$  subspace.

Alice cannot create a state with more than  $|A|$  qubits in the “one” state, where  $|A|$  is the number of spins she controls. So, in fact the calculation only needs to be done over the  $\mathcal{H}^{(1)}, \dots, \mathcal{H}^{(|A|)}$  subspaces (in other words, the singular values corresponding to states in other subspaces will always be zero).

In practice we have found that the optimal solution  $\vec{w}_1$  often belongs to the  $\mathcal{H}^{(1)}$  subspace. (In particular, we calculated the optimal solution for a range of different values of  $T$  for various 8 and 9-spin systems, and found that only for a very small minority of the values of  $T$ , for each system, was the solution *not* in the  $\mathcal{H}^{(1)}$  subspace). A rudimentary argument for this can be made as follows. Looking for solutions in  $\mathcal{H}^{(n)}$  means optimizing over Alice’s  $\binom{|A|}{n}$  degrees of freedom (of the space  $\mathcal{A} \cap \mathcal{H}^{(n)}$ ), in order to make the final state land in or near a  $\binom{|B|}{n}$ -dimensional target space  $\mathcal{B} \cap \mathcal{H}^{(n)}$ . This must be achieved despite the fact that the Hamiltonian is “trying” to move the state through a much larger  $\binom{N}{n}$ -dimensional space  $\mathcal{H}^{(n)}$ . Over the various values of  $n = 1, \dots, |A|$ , the dimensionality of  $\mathcal{A} \cap \mathcal{H}^{(n)}$  and  $\mathcal{B} \cap \mathcal{H}^{(n)}$  as a fraction of the dimensionality of  $\mathcal{H}^{(n)}$  is largest when  $n = 1$ . In other words, when  $n = 1$ , the size of the target space, and amount of control available of the initial state, is largest as a fraction of the dimensionality of the entire subspace  $\mathcal{H}^{(n)}$ .

So, in general we can restrict all the calculations to the  $N$ -dimensional subspace  $\mathcal{H}^{(1)}$ , and there will still be a good chance that we will arrive exactly at the globally-optimal encoding  $\vec{w}_1$ . Ignoring solutions in the other subspaces will increase the efficiency of calculation considerably, especially for large chains.

The evolution of a state in the  $\mathcal{H}^{(1)}$  subspace can also be interpreted as a *continuous quantum walk* of a particle over a graph. (For an introduction to quantum walks, see for example [Kem03] and references therein). The graph is simply the graph of interactions

between spins in the Hamiltonian  $H$ , and the state  $|\mathbf{0}\rangle_{\bar{j}} \otimes |1\rangle_j$  corresponds to the particle being at vertex  $j$  of that graph. So our methods for increasing communication fidelity are, equivalently, methods for guiding a quantum walk from one part of a graph to another. We point out this connection because of the significant interest currently in using quantum walks for solving computational problems (see for example [CCD<sup>+</sup>03, Amb04, OS04] and references therein).

To demonstrate the use of the SVD optimal-encoding technique, we now consider a numerical example. Imagine that Alice and Bob are joined by a 300-site open-ended chain with nearest-neighbour couplings given by the antiferromagnetic Heisenberg interaction, with coupling strengths all equal to 1. That is,

$$H = \sum_{j=1}^{299} \vec{\sigma}_j \cdot \vec{\sigma}_{j+1}. \quad (4.8)$$

Assume that Alice and Bob control the first and last 20 spins respectively.

In light of the earlier discussion, we restrict our optimization to the  $\mathcal{H}^{(1)}$  subspace, and thus ignore all singular vectors in other subspaces. A MATLAB program is used to carry out the following calculations. First, the 300 by 300 matrix  $H^{(1)}$ , defined to be the part of  $H$  that acts on  $\mathcal{H}^{(1)}$ , is constructed. Then, the SVD of

$$\tilde{U}^{(1)}(T) = P_{\mathcal{B} \cap \mathcal{H}^{(1)}} e^{-iH^{(1)}T} P_{\mathcal{A} \cap \mathcal{H}^{(1)}} \quad (4.9)$$

is calculated for a range of values of  $T$ . The optimal value for communication time is not known beforehand, so this repetition of the calculation for different values of  $T$  is needed in order to find a reasonable tradeoff between communication time and fidelity.

The four largest singular values,  $s_1, \dots, s_4$ , of  $\tilde{U}^{(1)}(T)$  are plotted in Figure 4.1. Over the range of  $T$  shown,  $s_1(T)$  has its maximum of 0.99999 at  $T = 75.75$ . So, this system can transmit a qubit with near-perfect fidelity, over a time interval of 75.75. The graph shows that  $s_2(75.75)$  and  $s_3(75.75)$  are also very close to 1, so in fact *two qubits* could be transmitted simultaneously with high fidelity in this example, using the encodings  $|\mathbf{0}\rangle$ ,  $\vec{w}_1(75.75)$ ,  $\vec{w}_2(75.75)$  and  $\vec{w}_3(75.75)$  for the two-qubit basis states.

Let's look at the actual optimally encoded states that are generated in this example. We visualize a state in  $\mathcal{H}^{(1)}$  by plotting the square magnitude of the coefficients  $\psi_j$ , where  $\psi_j$  is the coefficient of the basis state that has the  $j$ -th spin in the  $|1\rangle$  state:

$$\mathcal{H}^{(1)} \ni |\psi\rangle = \sum_{j=1, \dots, N} \psi_j |1\rangle_j \otimes |\mathbf{0}\rangle_{\bar{j}}. \quad (4.10)$$

In Figure 4.2 we show the evolution of the state  $\vec{w}_1(75.75)$ . That is, we set  $|\psi(0)\rangle = \vec{w}_1(75.75)$ , and  $|\psi(t)\rangle = e^{-iH^{(1)}t}|\psi(0)\rangle$ , and plot the magnitudes  $|\psi_j|^2$  for a sequence of

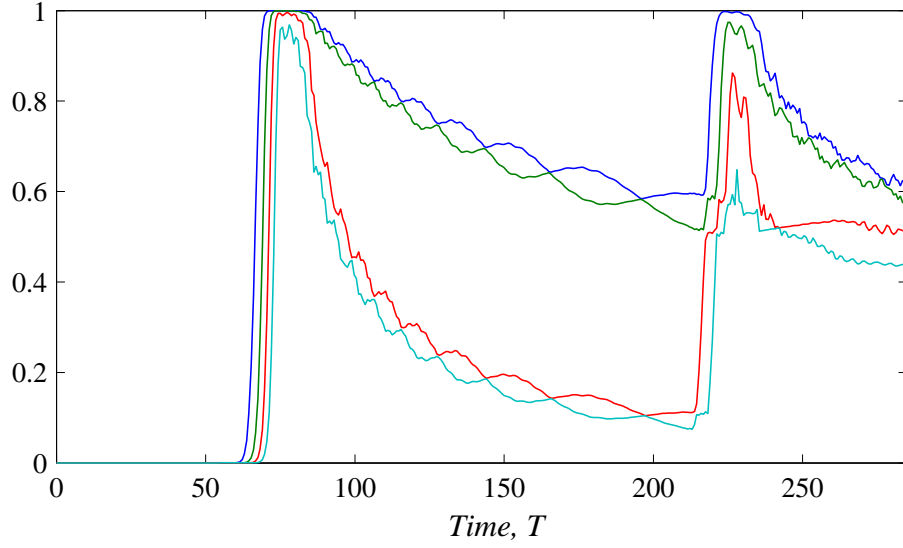


Figure 4.1: The largest four singular values of  $\tilde{U}^{(1)}(T)$ , for a range of communications times  $T$ . Open-ended Heisenberg chain,  $N=300$ , and Alice and Bob each control 20 sites.

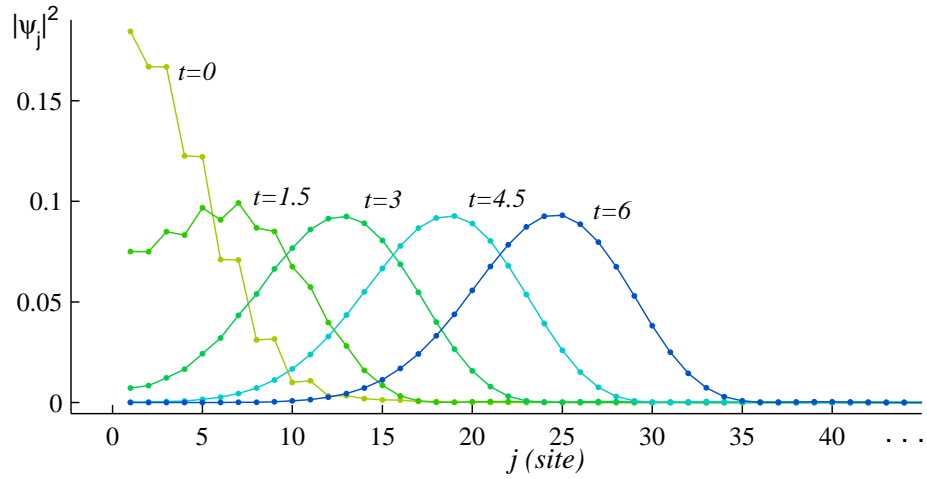


Figure 4.2: The optimal encoded state  $\vec{w}_1(75.75)$ , evolved for a sequence of times  $t$ .

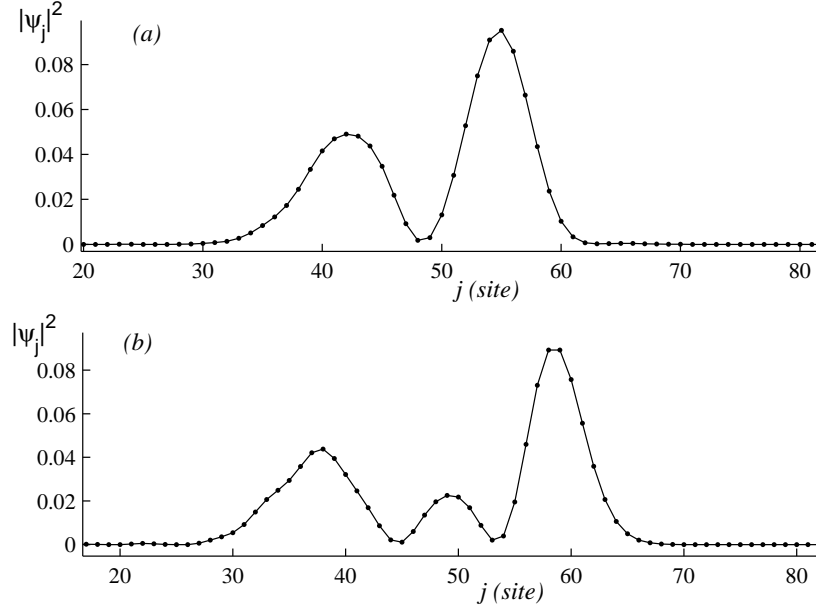


Figure 4.3: The states (a)  $\vec{w}_2(75.75)$  and (b)  $\vec{w}_3(75.75)$  evolved for time=12.

equally-spaced times  $t$ . As is necessarily the case, the  $t = 0$  state has non-zero coefficients only on Alice's spins,  $j = 1 \dots 20$ . The state deforms itself into a Gaussian shape quite quickly. This is interesting in comparison with the results in [OL04]. Whilst Gaussian initial states were shown to give a high fidelity on a Heisenberg ring, the best initial states for an open-ended Heisenberg chain are ones that deform into Gaussians. From the total communication time in this example, the group velocity of the pulse is roughly 3.95 (defining the distance between neighbouring spins to be 1). Thus, in the open-ended Heisenberg chain we have found the same phenomenon that appeared in the Heisenberg ring in [OL04], notably that the system has a preferred group velocity that gives a minimum dispersion and thus maximum fidelity. This explains the fact that in Figure 4.1 the singular values drop for  $T$  greater than 75.75, and rise again to a near-maximum at  $T \approx 225 \approx 3 \times 75.75$ : the high-fidelity communication for  $T \approx 225$  is also operating at the preferred group velocity, but the wave packet is traversing the chain three times, after bouncing from each end.

Curiously, the lower solutions  $\vec{w}_2(75.75)$  and  $\vec{w}_3(75.75)$  seem to evolve into a sum of two and three Gaussians respectively (see Figure 4.3). Animations of these evolutions are available online [Has04].

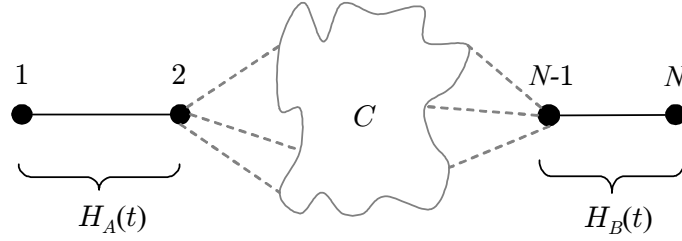


Figure 4.4: The general setup, whereby Alice and Bob vary the Hamiltonians  $H_A(t)$  and  $H_B(t)$  on their control spins over time, so as to increase communication fidelity through the network  $C$ .

### 4.3 Dynamic control

The scheme we presented in Section 4.2 utilizes the evolution of a system having completely static interactions. The control that Alice and Bob have over the chain is only for an instant at the beginning and end of the procedure, and so their only degrees of freedom for increasing the fidelity lie in the encoding they use. For very long chains, the number of control sites needed to give a high fidelity might become impractically large, as suggested by the results in [OL04]. In this section, we consider the advantage that can be gained by allowing Alice and Bob to control their spins throughout the procedure, by modulating the strength of interactions on those spins. The advantages of the scheme are that the number of control spins are fixed at four, and that suitable functions for Alice and Bob to use to vary the interaction strengths are easily derived from a simple extension of the SVD approach already described.

This type of control scheme is an example of a fundamental problem in quantum information processing, that of determining how to use the limited physical control that one has of a quantum system, in such a way as to achieve the dynamics that are required. For the task at hand, our method provides a practical and efficient way of finding an appropriate dynamical control.

A general schematic for the system is shown in Figure 4.4. Alice is now in control of just two spins, numbered 1 and 2, and, likewise, Bob controls two spins  $N - 1$  and  $N$ . The other spins in the system, numbered 3 to  $N - 2$ , are collectively denoted  $C$ . The graph of the interactions connecting Alice and Bob's spins can be arbitrary, except that spin 1 must directly couple only to 2, and spin  $N$  must directly couple only to  $N - 1$ . Like the previous scheme, we require that the system Hamiltonian  $H$  commutes with  $Z^{\text{tot}}$ , and that all spins are initialized to  $|0\rangle$  before the procedure starts. So again the problem is that of finding a way of sending the  $|1\rangle$  basis state with high fidelity.



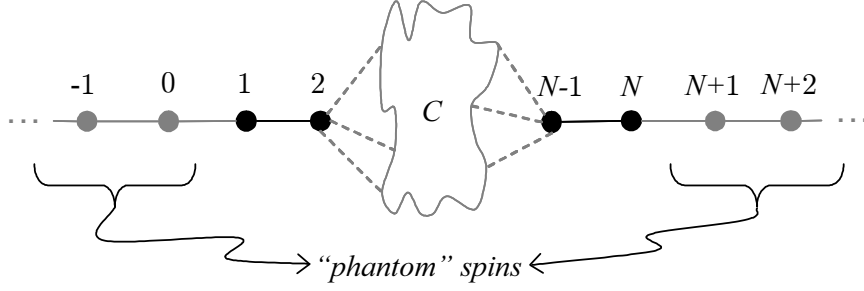


Figure 4.5: A modified version of Figure 4.4. Spin labels now range from  $(1 - N_P)$  to  $(N + N_P)$ , where  $N_P$  is the number of spins added to each side.

The protocol works as follows. At time  $t = 0$ , Alice transfers the qubit state she wishes to send onto spin 1. Then, over the time interval  $0 \leq t \leq T$  she varies the interactions between spins 1 and 2 according to some two-qubit Hamiltonian which we shall denote  $H_A(t)$ . Over the same interval, Bob varies the interactions between his control spins according to the two-qubit Hamiltonian  $H_B(t)$ . The total system Hamiltonian at time  $t$  is thus

$$H(t) = H_A(t) + H_{2 \dots (N-1)} + H_B(t), \quad 0 \leq t \leq T, \quad (4.11)$$

where  $H_{2 \dots (N-1)}$  denotes the constant part of Hamiltonian connecting spin 2 to spin  $N - 1$  through the network  $C$ . At time  $T$ , Bob's spin  $N$  will contain the sent qubit state, to a level of fidelity that depends on the choice of control Hamiltonians  $H_A(t)$  and  $H_B(t)$ .

We parameterize  $H_A(t)$  and  $H_B(t)$  in terms of real-valued control functions denoted  $J_A^R(t)$ ,  $J_A^I(t)$ ,  $J_B^R(t)$ , and  $J_B^I(t)$ , as follows:

$$\begin{aligned} H_A(t) = & J_A^R(t)[X_1 X_2 + Y_1 Y_2] \\ & + J_A^I(t)[X_1 Y_2 - Y_1 X_2] \end{aligned} \quad (4.12)$$

$$\begin{aligned} H_B(t) = & J_B^R(t)[X_N X_{N-1} + Y_N Y_{N-1}] \\ & + J_B^I(t)[X_N Y_{N-1} - Y_N X_{N-1}]. \end{aligned} \quad (4.13)$$

It will also be convenient to define two complex-valued versions of the control functions,  $J_A(t) \equiv J_A^R(t) + iJ_A^I(t)$  and  $J_B(t) \equiv J_B^R(t) + iJ_B^I(t)$ .

How do we choose the control functions in a way that gives a high average communication fidelity? Our solution is to imagine a modified version of the system as follows. First, the time varying interactions are made static, for instance by setting  $J_A(t) = J_B(t) = 1$ , and second, a number of “phantom” spins are added to both Alice and Bob's set of control spins (such as in Figure 4.5).

This modified system is of the correct form for applying the SVD method from the previous section. We can find the state  $|1_{ENC}\rangle_A$  on Alice's extended set of control spins that will evolve optimally for communication to Bob's extended set of control spins. A simulation of the evolution of this state on the modified system may be easily performed on a classical computer. From this, one can derive functions  $J_A(t)$  and  $J_B(t)$  that cause the unencoded message state  $|1\rangle$  on the original system to evolve in exactly the same way as the encoded message on the modified system, on all spins in the range 2 to  $N - 1$ . That is, the system and its modified version will share the same average communication fidelity. Thus, one can exploit the degrees of freedom of the encoding over many spins, while physically controlling just four.

We now describe the derivation of  $J_A(t)$  and  $J_B(t)$  in detail. Let  $|\phi(t)\rangle$  denote the evolution of the  $|1\rangle$  message on the system in Figure 4.4. That is,

$$|\phi(0)\rangle = |1\rangle_1 \otimes |\mathbf{0}\rangle_{\bar{1}} \text{ , and} \quad (4.14)$$

$$\frac{d|\phi(t)\rangle}{dt} = -iH(t)|\phi(t)\rangle. \quad (4.15)$$

Since  $|\phi(t)\rangle$  is in  $\mathcal{H}^{(1)}$ , we can write

$$|\phi(t)\rangle = \sum_{j=1}^N \phi_j(t) |1\rangle_j \otimes |\mathbf{0}\rangle_{\bar{j}}. \quad (4.16)$$

Assuming that  $N_P$  spins have been added to each side to create the modified system of Figure 4.5, and that all newly created bonds have an  $XY$  interaction of unit strength, the modified Hamiltonian is:

$$\begin{aligned} \tilde{H} = & \sum_{j=1-N_P}^1 [X_j X_{j+1} + Y_j Y_{j+1}] + H_{2\dots(N-1)} \\ & + \sum_{j=N-1}^{N+N_P-1} [X_j X_{j+1} + Y_j Y_{j+1}]. \end{aligned} \quad (4.17)$$

We let  $|\psi(t)\rangle$  denote the evolution of the encoded  $|1\rangle$  message basis on the modified system:

$$|\psi(t)\rangle = e^{-i\tilde{H}t} (|1_{ENC}\rangle_{(1-N_P)\dots 1} \otimes |\mathbf{0}\rangle_{2\dots(N+N_P)}). \quad (4.18)$$

Note that we are assuming in the above that the “modified Alice” has encoded the message over spins in the range  $(1 - N_P)$  to 1.

It is important for the procedure at hand that we choose  $|1_{ENC}\rangle$  from the  $\mathcal{H}^{(1)}$  subspace (whereas before this restriction was just a way of making the solution much faster to

compute), and we write

$$|\psi(t)\rangle = \sum_{j=1-N_P}^{N+N_P} \psi_j(t) |1\rangle_j \otimes |\mathbf{0}\rangle_{\bar{j}}. \quad (4.19)$$

The aim is to chose control functions  $J_A(t)$  and  $J_B(t)$  in such a way as to force  $\phi_j(t) = \psi_j(t)$ , for all the spins in the range  $j = 2, \dots, N-1$ , and for all  $t$  in the interval  $[0, T]$ . That is, we know the way the optimal encoded state evolves over the modified chain, and we want to make the  $|1\rangle$  state in the physical system evolve in exactly the same way, over all spins except 1 and  $N$ . In this way, the physical system will carry a qubit across its length with the same fidelity as the encoded modified system does. Now, the interactions on the spins from site 3 to site  $N-2$  are the same in the physical chain as in the modified chain. That is, the differential equations for the  $\psi_j(t)$  are the same as those for the  $\phi_j(t)$ , for  $j = 3, \dots, N-2$ . For example,

$$\begin{aligned} \frac{d\psi_j(t)}{dt} &= -if_j(\psi_2, \dots, \psi_{N-1}) \quad \text{and} \\ \frac{d\phi_j(t)}{dt} &= -if_j(\phi_2, \dots, \phi_{N-1}), \end{aligned} \quad (4.20)$$

for  $j = 3, \dots, N-2$ , for some set of functions  $f_j(\cdot)$  that depend on the Hamiltonian  $H_{2\dots(N-1)}$ . Also, the initial conditions are the same between the  $\psi_j$  and the  $\phi_j$ , for that range of spins:  $\psi_j(0) = \phi_j(0) = 0$ .

Spins in the range 3 to  $N-2$  only directly couple outside that range to spins 2 and  $N-1$ . Thus, it follows that if we can use our control functions to force  $\phi_2(t) = \psi_2(t)$  and  $\phi_{N-1}(t) = \psi_{N-1}(t)$ , over the time range  $t = 0, \dots, T$ , then we will have  $\psi_j(t) = \phi_j(t)$  for all  $t$  in that time range, and for *all*  $j = 2, \dots, N-1$ , as desired. Since  $\phi_2(0) = \psi_2(0) = 0$ , and  $\phi_{N-1}(0) = \psi_{N-1}(0) = 0$ , the requirement reduces to forcing  $\frac{d\phi_2(t)}{dt} = \frac{d\psi_2(t)}{dt}$ , and  $\frac{d\psi_{N-1}(t)}{dt} = \frac{d\phi_{N-1}(t)}{dt}$ , over the time range  $t = 0, \dots, T$ .

Now,

$$\frac{d\psi_2(t)}{dt} = -2i\psi_1(t) - if_2(\psi_2, \dots, \psi_{N-1}), \quad \text{and} \quad (4.21)$$

$$\frac{d\phi_2(t)}{dt} = -2iJ_A(t)\phi_1(t) - if_2(\phi_2, \dots, \phi_{N-1}), \quad (4.22)$$

for some  $f_2(\cdot)$  that depends on  $H_{2\dots(N-1)}$ , and where we have used that

$$\begin{aligned} \langle 0|_1 \langle 1|_2 H_A(t) |1\rangle_1 |0\rangle_2 &= 2J_A^R(t) + 2iJ_A^I(t) \\ &= 2J_A(t). \end{aligned} \quad (4.23)$$

So, assuming that at time  $t$  the condition  $\phi_j(t) = \psi_j(t)$  holds for  $j = 2, \dots, N-1$ , then the condition  $\frac{d\phi_2(t)}{dt} = \frac{d\psi_2(t)}{dt}$  is achieved by setting

$$J_A(t) = \frac{\psi_1(t)}{\phi_1(t)}. \quad (4.24)$$

Similarly,  $\frac{d\phi_{N-1}(t)}{dt} = \frac{d\psi_{N-1}(t)}{dt}$  is achieved by setting

$$J_B(t) = \frac{\psi_N(t)}{\phi_N(t)}. \quad (4.25)$$

Thus, the practical task of numerically calculating  $J_A(t)$  and  $J_B(t)$  involves simulating the evolution of both the  $\phi_j$  and  $\psi_j$  states on the original and modified systems respectively, over the time interval  $[0, T]$ , and evaluating Eqs. (4.24) and (4.25).

It should be noted that Eqs. (4.24) and (4.25) will never become infinite. In fact,  $|J_A(t)|$  and  $|J_B(t)|$  will be at most 1. This is a simple consequence of conservation of probability. Since  $\psi_j(t) = \phi_j(t)$  over the bulk of the chain ( $j = 2, \dots, N-1$ ) and Alice and Bob's sides only interact via the bulk of the chain for both the physical and modified systems, we have that

$$\sum_{j=1-N_P}^1 |\psi_j|^2 = |\phi_1|^2, \text{ and} \quad (4.26)$$

$$\sum_{j=N}^{N+N_P} |\psi_j|^2 = |\phi_N|^2, \quad (4.27)$$

from which it follows that  $|\psi_1(t)| \leq |\phi_1(t)|$  and  $|\psi_N(t)| \leq |\phi_N(t)|$ . So,  $|J_A(t)| \leq 1$  and  $|J_B(t)| \leq 1$ , if they are defined. If  $J_A(t)$  (or  $J_B(t)$ ) is undefined ( $0/0$ ), it means that the requirement of  $\frac{d\phi_2(t)}{dt} = \frac{d\psi_2(t)}{dt}$  ( respectively  $\frac{d\phi_{N-1}(t)}{dt} = \frac{d\psi_{N-1}(t)}{dt}$  ) is satisfied *regardless* of the value of  $J_A(t)$  (respectively  $J_B(t)$ ) for that  $t$ , in which case the value of the control function can be chosen arbitrarily at that time.

We now plot the derived control functions  $J_A(t)$  and  $J_B(t)$  for two simple example XY chain systems. We used numerical integration in these examples, in calculating the evolution of the  $\phi_j(t)$  due to the time-varying Hamiltonian. We divided the total evolution into a number of discrete time steps, where the approximation was made that the Hamiltonian remains constant throughout each step. The value of  $J_A(t)$  and  $J_B(t)$  for a step was calculated from the state of the system at the previous step. We used 2000 time steps, which gave a final fidelity in the physical chain within two significant figures of the correct value given by the evolution of the static modified system. Note that for each of the two examples, the derived functions  $J_A(t)$  and  $J_B(t)$  were real-valued, corresponding

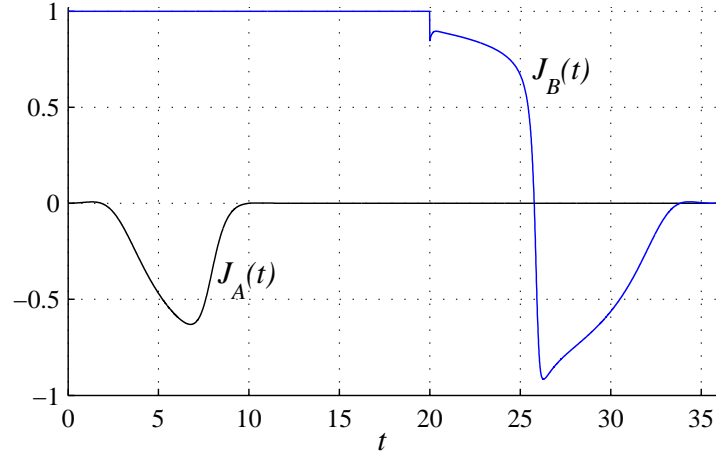


Figure 4.6: Control functions for a 104-spin  $XY$  chain, where the non-controlled coupling strengths all have the same strength, 1.

to control of only the  $XX + YY$  component of the Hamiltonian on the control spins. For more complicated interaction graphs, or interactions other than  $XY$ , this property would not generally hold.

The first example is a chain 104 spins long (ie. 100 non-controlled spins, plus the four control spins), with all the non-controlled coupling strengths set to the same value, 1. The control functions were derived by using a modified chain 144 spins long (that is, 20 phantom spins added to each side) with all coupling strengths set to 1, and the total communication time  $T$  chosen to be 36. Figure 4.6 shows that the resulting  $J_A(t)$  and  $J_B(t)$  are quite simple and well behaved. The fidelity measure  $\mathcal{C}_B(T)$  is 1.0, to 6 decimal places. This can be compared with the fidelity in the same 104-spin system but without the time-dependent control, that is with  $J_A(t)$  and  $J_B(t)$  fixed at 1: over a time interval  $0 < t < 1000$ , the value of  $\mathcal{C}_B(t)$  is at most 0.2809.

The second example is an  $XY$  chain 29 spins long, but where the non-controlled coupling strengths are randomly sampled uniformly from the interval  $[0.95, 1.05]$ . This is as if the chain has been manufactured with random imperfections in the coupling strengths, but these coupling strengths have been somehow measured after the manufacturing process and are known to Alice and Bob. (A shorter chain was chosen in this example, compared with the previous example, in order that a near-perfect fidelity would still result. We have observed that when random couplings are used, the achievable fidelity will decrease as a function of the chain length). We derived control functions using a modified system with 25 phantom spins added to each side, where the new arbitrary coupling strengths

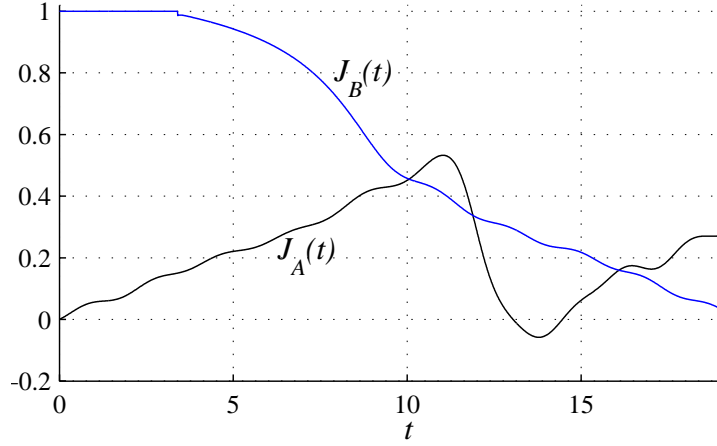


Figure 4.7: Control functions for a 29-spin  $XY$  chain, where the non-controlled coupling strengths were chosen randomly from the interval  $[0.95, 1.05]$ .

are set to 1. The communication time  $T$  was chosen to be 19.5. Figure 4.7 shows control functions which are a little more complicated in this case, but still rather smooth. The fidelity measure is  $\mathcal{C}_B(T) = 0.99625$ . In comparison, in the non-controlled version of this system, with  $J_A(t) = J_B(t) = 1$ , the value of  $\mathcal{C}_B(t)$  does not exceed 0.496 over the interval  $0 < t < 1000$ . Animations of both examples are available online [Has04].

## 4.4 Conclusion

We have considered the problem of communicating a quantum state over an arbitrary  $Z^{\text{tot}}$ -conserving spin system. Our first scheme used a static system Hamiltonian, and utilized the fact that the sender and receiver control several spins each, to increase fidelity by performing state encoding. We showed that choosing the optimal state encoding is a simple matter of performing a SVD on a modified evolution matrix, given our fixed choice of encoding for the  $|0\rangle$  message state.

We have also shown that if the sender and receiver have control of just two spins each, but can vary the interactions on these four spins over time, then they can achieve a fidelity that is equal to if they each controlled many more spins on a static system and used the optimal state encoding. We have given a practical method of deriving suitable control functions. The advantage of this scheme is the “fixed interface” that Alice and Bob have with the chain. That is, if the chain is altered, the only change that Alice and Bob need make is to their control functions, rather than to the number of spins they control.

It should be noted that the systems we have considered are idealized to a high degree. In particular, we haven't considered the effects of external noise, or the effect of having a Hamiltonian that only *approximately* commutes with  $Z^{\text{tot}}$ , or the case where Alice and Bob have only an approximate knowledge of the system Hamiltonian.

## 4.5 Appendix A: Proof of Eq. (4.5)

In this appendix we sketch a proof of Eq. (4.5). This inequality shows that the average fidelity  $\bar{F}$  of a decoded message (where the average is taken uniformly over all message states on the block sphere) is bounded tightly by two monotonic functions of  $\mathcal{C}_B(T)$ . Hence,  $\mathcal{C}_B(T)$  is a reasonable measure of average fidelity.

The state  $|\Psi(T)\rangle$  of the entire chain at time  $T$  is given by Eq. (4.3). We imagine that Bob performs a decoding unitary, denoted  $U_{\text{dec}}$ , on the spins he controls.  $U_{\text{dec}}$  is defined to act as follows:  $U_{\text{dec}}|\mathbf{0}\rangle_B = |\mathbf{0}\rangle_B$  and  $U_{\text{dec}}|\gamma(T)\rangle_B = |0\dots 01\rangle_B$ , where  $|0\dots 01\rangle_B$  is the  $|1\rangle$  state on spin  $N$  and the all-zero state on Bob's other spins. The state of spin  $N$  after decoding is

$$\rho_N = \text{tr}_{\bar{N}} \left( U_{\text{dec}} |\Psi(T)\rangle \langle \Psi(T)| U_{\text{dec}}^\dagger \right) \quad (4.28)$$

$$= (\alpha|0\rangle + \beta\sqrt{\mathcal{C}_B(T)}|1\rangle)(\alpha\langle 0| + \beta\sqrt{\mathcal{C}_B(T)}\langle 1|)^\dagger + \tilde{\rho}|\beta|^2(1 - \mathcal{C}_B(T)) \quad (4.29)$$

where  $\tilde{\rho} \equiv \text{tr}_{\bar{N}}(U_{\text{dec}}|\eta(T)\rangle\langle\eta(T)|U_{\text{dec}}^\dagger)$ . The fidelity between  $\rho_N$  and the message is given by

$$F = (\alpha|0\rangle + \beta|1\rangle)^\dagger \rho_N (\alpha|0\rangle + \beta|1\rangle) \quad (4.30)$$

$$\begin{aligned} &= |\alpha|^4 + |\beta|^4 \mathcal{C}_B(T) + 2|\alpha|^2|\beta|^2\sqrt{\mathcal{C}_B(T)} \\ &\quad + |\alpha|^2|\beta|^2(1 - \mathcal{C}_B(T))\langle 0|\tilde{\rho}|0\rangle \\ &\quad + |\beta|^4(1 - \mathcal{C}_B(T))\langle 1|\tilde{\rho}|1\rangle \\ &\quad + 2|\beta|^2(1 - \mathcal{C}_B(T))\text{Re}(\alpha^*\beta\langle 0|\tilde{\rho}|1\rangle). \end{aligned} \quad (4.31)$$

Thus to find the average value of  $F$  over all messages  $\alpha|0\rangle + \beta|1\rangle$ , we must find the average values of  $|\alpha|^4$ ,  $|\beta|^4$ ,  $|\alpha|^2|\beta|^2$ , and  $|\beta|^2\text{Re}(\alpha^*\beta)$  over the Bloch sphere. These averages are

given by the following integrals respectively:

$$\frac{1}{2} \int_0^\pi d\theta \cos^4\left(\frac{\theta}{2}\right) \sin(\theta) = \frac{1}{3} \quad (4.32)$$

$$\frac{1}{2} \int_0^\pi d\theta \sin^4\left(\frac{\theta}{2}\right) \sin(\theta) = \frac{1}{3} \quad (4.33)$$

$$\frac{1}{2} \int_0^\pi d\theta \sin^2\left(\frac{\theta}{2}\right) \cos^2\left(\frac{\theta}{2}\right) \sin(\theta) = \frac{1}{6} \quad (4.34)$$

$$\frac{1}{4\pi} \int_0^{2\pi} d\phi \cos(\phi) \int_0^\pi d\theta \sin^3\left(\frac{\theta}{2}\right) \cos\left(\frac{\theta}{2}\right) = 0. \quad (4.35)$$

Thus,

$$\bar{F} = \frac{1}{2} + \frac{1}{3} \sqrt{\mathcal{C}_B(T)} + \frac{1}{6} \mathcal{C}_B(T) + \frac{1}{6} (1 - \mathcal{C}_B(T)) \langle 1 | \tilde{\rho} | 1 \rangle, \quad (4.36)$$

and since  $(1 - \mathcal{C}_B(T)) \langle 1 | \tilde{\rho} | 1 \rangle$  may take values in the range  $[0, 1]$ , we arrive at Eq. (4.5) as required.

## 4.6 Appendix B: Interpretation of $\mathcal{C}_B(T)$

Here we outline a proof of the claim in Section 4.2 regarding the connection between  $\mathcal{C}_B(T)$  and the system's ability to transmit entanglement from Alice to Bob. This connection helps establish  $\mathcal{C}_B(T)$  as a good measure of communication fidelity.

**Proposition:** *Suppose that Alice sends a state which is maximally entangled with some additional spin that Alice possesses. (After the maximally entangled state is created, the additional spin is assumed to not interact during the remainder of the communication procedure). Then, after the communication procedure of Section 4.2 is carried out, the entanglement (measured by concurrence) between Alice's additional spin and Bob's decoded message, equals  $\sqrt{\mathcal{C}_B(T)}$ .*

**Proof:** Note that it does not matter which maximally-entangled state is used — all such states are equivalent up to a local unitary on the additional spin, and such a local unitary could not possibly affect the way entanglement is transferred through the system.

Let the additional spin “+” and the spin “M” containing the message have the maximally entangled state  $\frac{1}{\sqrt{2}}(|0\rangle_+ |0\rangle_M + |1\rangle_+ |1\rangle_M)$ . Thus, after Alice performs her encoding, the entire state is:

$$|\Phi(0)\rangle = \frac{1}{\sqrt{2}} [ |0\rangle_+ |0\rangle_A |0\rangle_{\bar{A}} + |1\rangle_+ |1_{ENC}\rangle_A |0\rangle_{\bar{A}} ]. \quad (4.37)$$

After the system evolves for time  $T$ , the state becomes

$$\begin{aligned} |\Phi(T)\rangle = \frac{1}{\sqrt{2}} \Big[ & |0\rangle_+ |0\rangle_{\bar{B}} |0\rangle_B + |1\rangle_+ (\sqrt{1 - \mathcal{C}_B(T)} |\eta(T)\rangle \\ & + \sqrt{\mathcal{C}_B(T)} |0\rangle_{\bar{B}} |\gamma(T)\rangle_B \Big]. \end{aligned} \quad (4.38)$$



Then Bob performs a decoding unitary,  $U_{\text{dec}}$ , on the spins he controls. As in Appendix 4.5,  $U_{\text{dec}}$  is defined to act as follows:  $U_{\text{dec}}|\mathbf{0}\rangle_B = |\mathbf{0}\rangle_B$  and  $U_{\text{dec}}|\gamma(T)\rangle_B = |0\dots 01\rangle_B$ . After Bob's decoding, the joint state of Alice's additional spin and Bob's decoded spin is:

$$\begin{aligned}\rho_{+/N} &= \text{tr}_{+/N}(U_{\text{dec}}\text{tr}_{\bar{B}}(|\Phi(T)\rangle\langle\Phi(T)|)U_{\text{dec}}^\dagger) \\ &= \frac{1}{2}\left[(1 - \mathcal{C}_B(T))|1\rangle\langle 1| \otimes \tilde{\rho} + \mathcal{C}_B(T)|11\rangle\langle 11| \right. \\ &\quad \left. + |00\rangle\langle 00| + \sqrt{\mathcal{C}_B(T)}(|00\rangle\langle 11| \right. \\ &\quad \left. + |11\rangle\langle 00|)\right],\end{aligned}\tag{4.39}$$

where  $\tilde{\rho} \equiv \text{tr}_{+/N}(U_{\text{dec}}\text{tr}_{\bar{B}}(|\eta(T)\rangle\langle\eta(T)|)U_{\text{dec}}^\dagger)$ , and where  $\text{tr}_{(\cdot)}(\cdot)$  is the partial trace performed over the spins indicated.

Concurrence is a measure of entanglement between two qubits [Woo98]. The value of concurrence for a density matrix  $\rho_{+/N}$  is equal to

$$E(\rho_{+/N}) = \max\{0, \sqrt{\lambda_1} - \sqrt{\lambda_2} - \sqrt{\lambda_3} - \sqrt{\lambda_4}\},\tag{4.40}$$

where the  $\lambda_j$ s are the eigenvalues, in nonincreasing order, of the matrix  $\rho_{+/N}(Y \otimes Y)\rho_{+/N}^*(Y \otimes Y)$ , where  $*$  represents complex conjugation in the computational basis. It can be shown that

$$\begin{aligned}\lambda_1 &= \frac{1}{4} \left( \sqrt{\tilde{\rho}_{11}\mathcal{C}_B(T) + 1 - \tilde{\rho}_{11}} + \sqrt{\mathcal{C}_B(T)} \right)^2 \\ \lambda_2 &= \frac{1}{4} \left( \sqrt{\tilde{\rho}_{11}\mathcal{C}_B(T) + 1 - \tilde{\rho}_{11}} - \sqrt{\mathcal{C}_B(T)} \right)^2 \\ \lambda_3 &= 0 \\ \lambda_4 &= 0,\end{aligned}\tag{4.41}$$

where  $\tilde{\rho}_{11} = \langle 0|\tilde{\rho}|0\rangle$ . Thus, using the fact that  $\mathcal{C}_B(T)$  and  $\tilde{\rho}_{11}$  each lie in the interval  $[0, 1]$ , we have

$$E(\rho_{+/N}) = \sqrt{\mathcal{C}_B(T)},\tag{4.42}$$

as required.  $\square$



# Cluster states and optical quantum computation

---

This chapter contains introductions to two topics: the cluster-state model of quantum computation, and the linear-optical implementation of quantum computation. These two topics, which at first glance do not appear closely related, are both of central importance to work described in Chapter 6. The intention is not to provide a complete treatment of either of the two subjects, but rather to focus on the aspects relevant to Chapter 6.

Section 5.1 introduces the cluster-state model of quantum computing. The key to this model is the definition of a particular highly entangled state known as a *cluster state*. If one is given a cluster state, then it can be used to perform *any* quantum computation (up to a certain size depending on the size of the state) by simply measuring individual qubits in the state. This is of immense theoretical interest because it shows that, in essence, almost the entire power of quantum computation can be contained in an entangled resource state.

Section 5.2 introduces linear-optical quantum computing, one of the most promising proposals for how to physically implement a quantum computer. In a linear-optical quantum computer, each qubit is represented by the state of a single photon, and operations are performed on qubits using passive linear optics in addition to photodetector measurements and single-photon sources. Appealing features of the scheme include the fact that photons can be fairly well isolated from the environment, the implementation of single-qubit unitary gates is simple, and read-out can be performed with relative ease.

Finally Section 5.3 discusses the considerable advantages that result from applying the cluster state model to linear-optical quantum computing. Schemes developed by Nielsen [Nie04] and Browne and Rudolph [BR05] demonstrate that the difficulty in implementing a linear-optical quantum computation can be reduced dramatically if the computation is recast in the cluster-state model.

*NOTE: This chapter was written solely for the purpose of inclusion in this thesis, and has not been published elsewhere. The contents of this chapter were written entirely by*

myself; however, as this is a review chapter, it doesn't contain any of my own original research results.

## 5.1 Cluster-state computation

In this section I describe the cluster-state model of quantum computation. I begin by defining cluster states, and giving a broad description of the stages of a cluster-state computation. Then I show how a series of circuit identities can be used to demonstrate a mathematical equivalence between the cluster-state model and circuit model. Finally I introduce a set of notation, and a corresponding set of rules for interpreting the notation, that allow for a compact and convenient description of arbitrary cluster-state computations. The notation also allows for a very simple method for converting from the circuit model to the cluster-state model.

Note that further details on the cluster-state model which are omitted from this section can be found in the following references: [Nie05],[ND05], [RB01], and [RBB03].

### 5.1.1 Basic elements of the cluster-state model

At the heart of the cluster-state model of quantum computation is the definition of a special class of entangled quantum states known as *cluster states*. Cluster states can be defined for systems of any number of qubits.

A simple one-to-one correspondence exists between  $N$ -qubit cluster states and graphs on  $N$  vertices, and so to specify a cluster state it is usually most convenient to specify the corresponding graph<sup>1</sup>. The state which is associated to a particular graph is then defined as follows.

**Definition 5.1** (Cluster state): *Given an  $N$ -vertex graph  $G$ , associate each of the vertices to a corresponding qubit in an  $N$ -qubit system. Then, the cluster state associated with the graph  $G$  is the state obtained by the performing this graph-dependent preparation procedure:*

1. Initialize each of the  $N$  qubits in the state  $|+\rangle$ .
2. For each edge in graph, apply a CPHASE gate between the corresponding pair of qubits.

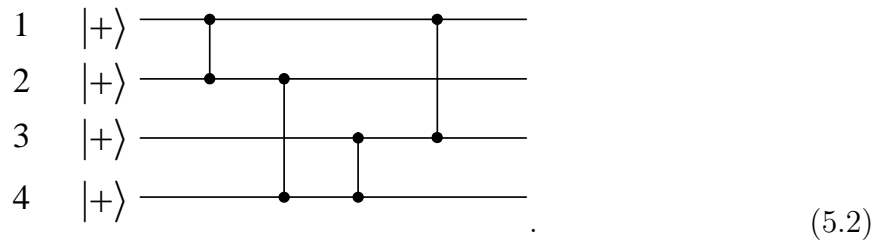
---

<sup>1</sup>Note that our definition of cluster state is more general than that taken by some authors. Some authors reserve the term cluster state for only those states corresponding to square-lattice graphs, and refer to states corresponding to all other graphs as “graph states”.

So, as a simple example, the graph



specifies the cluster state that would result from applying the following circuit:



Note that Definition 5.1 does not specify the order in which the CPHASE gates should be applied. This does not matter, since CPHASE gates commute with each other, and thus the same state will result from any ordering. Although I have defined cluster states by specifying a particular preparation procedure involving  $|+\rangle$  and CPHASE gates, it is of course possible to use other quite different procedures to prepare the same state. An important case in point is a method for building cluster states optically, where so-called “fusion-gates” are used in the preparation procedure. This method is covered in detail later in the chapter.

In a cluster-state computation, the cluster state can be thought of as playing the role of an initial resource, which is “used up” as the computation proceeds. Broadly speaking, a cluster-state computation involves three stages:

1. **Preparation.** A cluster state is created.
2. **Measurement.** An adaptive sequence of single-qubit measurements is performed on qubits in the cluster state. (This stage effectively “runs” the computation).
3. **Read-out.** The results of the computation are found by measuring the remaining cluster qubits not measured in stage 2, and performing a simple post-processing on these results that is a function of the measurement results of stage 2.

Importantly, this procedure is equivalent in computational power to the standard circuit-model of quantum computation. That is, any quantum circuit can be simulated by a cluster-state computation, using resources that scale polynomially in the depth and breath

of the circuit<sup>2</sup>. In the following subsection I sketch an argument showing this equivalence, by outlining a general procedure for converting a quantum circuit to a cluster-state computation.

Note that for simplicity I consider a version of the cluster-state model that differs from Raussendorf and Briegel’s original design [RB01]. In the original cluster-state model, the preparation of the cluster state (step 1 above) does not depend on the details of the computation being performed (just on the depth and breadth of the computation) and it is rather steps 2 and 3 that determine the computation performed. However in our presentation, all three steps of the cluster computation will depend on the details of the computation being performed. An advantage of the treatment in [RB01] is the way it directly demonstrates the remarkable fact that a single generic cluster state can act as the main resource for simulating any desired quantum circuit. A cluster state can in some sense capture the entire “power” of quantum computation.

### 5.1.2 Conversion from circuit to cluster (the hard way)

In this subsection, I outline a set of rules for converting a quantum circuit to a cluster-state computation. In practice (and as described in Subsection 5.1.3) there are other ways of performing the conversion which are easier, but the method presented as follows has the advantage of making explicit the mathematical equivalence between the circuit model and cluster-state model. The idea behind the conversion process is to apply a series of transformations to a quantum circuit so that it eventually has the form of a cluster-state computation as described broadly in the previous subsection.

For simplicity, I shall assume that the circuit to be converted satisfies three properties described below. This will not affect the generality of the argument, since standard techniques can be used to easily transform any other circuit to one satisfying the properties. The three properties which the circuit must satisfy are as follows:

1. At the beginning of the circuit all qubits are initialized in the  $|+\rangle$  state.
2. Only the following gates are allowed: (a) CPHASE gates, and (b) the set of single-qubit gates parameterized  $He^{-i\alpha Z/2}$ , where  $H$  is the Hadamard gate,  $Z$  is the Pauli  $Z$  gate, and  $\alpha$  is a real parameter. (The notation  $Z_\alpha \equiv e^{-i\alpha Z/2}$  will be adopted for brevity.) Note that this set of gates is *universal*, i.e., any unitary operation can be expressed in terms of these gates.

---

<sup>2</sup>Likewise, any cluster-state computation can be simulated with a quantum circuit, using resources that scale polynomially with the size of the cluster. This is a direct consequence of the way we have defined the three stages of a cluster-state computation in terms of circuit operations.

A simple example of a circuit in the correct form is:



The first step in converting a circuit of the above form to a cluster-state computation is to substitute all single qubit gates in the circuit according to the following circuit identity:



In the right hand side of Eq. (5.4),  $m = 0, 1$  is the result from measuring the first qubit in the  $Z$  basis, after it has first interacted with a second qubit in the  $|+\rangle$  state and undergone a single-qubit gate  $HZ_\alpha$ . Depending on the value of  $m$ , either an  $X$  gate or the identity is applied to the second qubit as shown. (I will refer to gate such as  $X^m$  whose operation is determined by the value of a classical bit  $m$  as being “classically controlled”). The proof of Eq. (5.4) is straightforward, and will be omitted. The right hand side of Eq. (5.4) is known as the *transport circuit*, since it has the combined effect of applying the gate  $HZ_\alpha$  and transferring the output to the second qubit.

Applying the substitution Eq. (5.4) to our example circuit (5.3) gives



In general at this stage of conversion the circuit will have the following feature: gates of the form  $HZ_\alpha$  will occur only immediately before a  $Z$ -basis measurement. As such, we can imagine that gates of form  $HZ_\alpha$  do not actually occur; rather we can think of the combined effect of a  $HZ_\alpha$  gate followed by a  $Z$ -basis measurement as simply being a measurement in a variable basis  $(HZ_\alpha)^\dagger|0\rangle$ ,  $(HZ_\alpha)^\dagger|1\rangle$ . Notice that if it weren't for the

classically-controlled  $X$  gates, the circuit at this stage would essentially be of the form of a cluster state computation (that is, consist of the three stages listed in Subsection 5.1.1).

Accordingly, the next step of the conversion removes the classically-controlled  $X$  gates. This is done by repeatedly applying rules (given below) that cause Pauli gates to commute rightwards through all other operations in the circuit. This ultimately replaces each original  $X^m$  gate in the circuit by an equivalent manipulation of later measurement result bits and measurement bases. The relevant rules for commuting Pauli gates rightwards are as follows. First, an arbitrary Pauli gate  $X^m Z^n$  can be moved rightwards through a CPHASE gate by way of the following circuit identity:

$$\begin{array}{c} \boxed{X^m Z^n} \\ \text{---} \bullet \text{---} \\ | \\ \text{---} \bullet \text{---} \end{array} = \begin{array}{c} \text{---} \bullet \text{---} \boxed{X^m Z^n} \\ | \\ \text{---} \bullet \text{---} \boxed{Z^m} \end{array} . \quad (5.6)$$

Second, Pauli gates can move rightwards through a combined  $HZ_\alpha$  gate and  $Z$ -basis measurement as follows:

$$\boxed{X^m Z^n} \boxed{HZ_\alpha} \begin{array}{c} Z \\ \nearrow \end{array} = \boxed{HZ_{(-1)^m \alpha}} \begin{array}{c} Z \\ \nearrow \end{array} \bigoplus_n . \quad (5.7)$$

That is, an  $X$  gate on the left can be substituted for a sign flip of the parameter  $\alpha$ , and a  $Z$  gate on the left can be substituted for a bit flip of the measurement result. Third, moving  $X^m Z^n$  rightwards through an  $X$ -basis measurement involves flipping the measurement result if  $n$  is odd. Applying these commutation rules for our example circuit gives the following circuit, which is finally in the form of a cluster-state computation:

$$\begin{array}{c} \begin{array}{|c|} \hline \text{Stage 1} \\ \hline \end{array} \begin{array}{|c|} \hline \text{Stage 2} \\ \hline \end{array} \begin{array}{|c|} \hline \text{Stage 3} \\ \hline \end{array} \\ \hline \begin{array}{c} |+\rangle \text{---} \bullet \text{---} \boxed{HZ_\alpha} \begin{array}{c} Z \\ \nearrow \end{array} m \\ |+\rangle \text{---} \bullet \text{---} \\ |+\rangle \text{---} \bullet \text{---} \boxed{HZ_\beta} \begin{array}{c} Z \\ \nearrow \end{array} n \\ |+\rangle \text{---} \bullet \text{---} \end{array} \end{array} \begin{array}{c} \begin{array}{c} \bigoplus_n \\ \nearrow X \end{array} \\ \bigoplus_m \end{array} . \quad (5.8)$$

The stages denoted in Circuit (5.8) are the three stages of cluster-state computation listed in Subsection 5.1.1. The graph of the cluster state that is created in stage 1 of Circuit



(5.8) is



which, by no accident, matches the shape of the original circuit in Eq. (5.3).

In our example conversion, the substitution rule in Eq. (5.7) was not used. However, in general this rule is necessary. A consequence is that in general some measurement results in stage 2 of the computation will affect the basis in which other measurements are performed, via the conditional sign-flip applied to the  $\alpha$  parameter in Eq. (5.7). This dependence of measurement operations on earlier results is referred to as “classical feed-forward”. The requirement for feed-forward means that in general the measurements in stage 2 cannot all be performed simultaneously. Instead, cluster nodes will usually need to be measured in the same left-to-right order that the corresponding gates appear in the original circuit. An important exception is measurements that have an  $\alpha$  parameter which is an integer multiple of  $\frac{\pi}{2}$ . Such measurements can be performed at any time during stage 2, because changing the sign of  $\alpha$  will not affect the physical measurement basis. (To see this, first note that replacing a  $Z_\alpha$  with a  $Z_{-\alpha}$  is equivalent to inserting an extra  $Z_{-2\alpha}$  operation in the circuit. If  $\alpha = k\frac{\pi}{2}$  where  $k$  is an integer, then this extra operation is  $Z_{-k\pi} = Z^k$ . The effect of the extra Pauli operation  $Z^k$  is to conditionally flip the measurement result bit, rather than change the axis of measurement.)

So, at the expense of restricting a cluster-state computation to simulate only circuits containing the gates CPHASE and  $HZ_{k\frac{\pi}{2}}$ ,  $k \in \mathbb{Z}$ , it is possible to highly parallelize the computation by performing every operation in stages 2 and 3 simultaneously. Unfortunately, this restricted set of gates is not universal for quantum computation. Nevertheless, the circuits that can be created by composing these gates correspond to the important *Clifford group* of operations. Significantly, most common types of quantum error correction circuitry belong to the Clifford group. Circuits of this type will thus become highly parallelized when converted to a cluster computation. The work presented in Chapter 6 makes extensive use of this property in the construction of cluster computations that perform fault-tolerant quantum error correction.

### 5.1.3 The Pauli-frame method for feed-forward

The description of cluster-state computing to this point has emphasized the connection to circuit-model computation. The next goal is to simplify the presentation of the cluster-

state model by giving a description that is self-contained and not reliant on the circuit model. Central to this description are two constructions: (1) a graphical way of specifying a cluster-state computation, that specifies the graph of the cluster state as well as the bases and order of measurements, and (2) a set of simple bookkeeping rules for correctly performing classical feed-forward of measurement outcomes and adjustment of final readout bits. These bookkeeping rules take the place of the rules given in Subsection 5.1.2 for commuting the classically-controlled  $X$  gates rightwards. To begin, I describe the graphical specification of a cluster-state computation, and follow with a description of the associated bookkeeping rules.

An example of the graphical specification of a cluster-state computation is as follows:

$$\begin{array}{ccccc}
 \textcircled{1} & & \textcircled{2} & & \textcircled{\phantom{0}} \\
 HZ_{\alpha} & \text{---} & HZ_{\pm\gamma} & \text{---} & \\
 & & | & & \\
 \textcircled{1} & & \textcircled{2} & & \textcircled{\phantom{0}} \\
 HZ_{\beta} & \text{---} & HZ_{\pm\delta} & \text{---} & 
 \end{array} . \tag{5.10}$$

The overall shape of the diagram in Eq. (5.10) specifies the cluster state to be created at stage 1 of the computation, in the obvious way. Additionally, inside each node may be written up to two pieces of information. First, the number in the upper part of a node denotes the order in which that node is measured. Thus, all nodes labelled 1 must be measured before all nodes labelled 2, and so forth. Nodes that do not have an order label may be measured at any time after creation of the cluster state, regardless of when other nodes are measured. (For example, a cluster-state computation that performs a Clifford group operation will usually not require any order labels to be specified.)

The operator written in the lower part of a node specifies the basis in which the node is measured. So “ $HZ_{\alpha}$ ” means that the node should be measured in the basis  $(HZ_{\alpha})^{\dagger}|0\rangle, (HZ_{\alpha})^{\dagger}|1\rangle$ . When a “ $\pm$ ” sign precedes the parameter  $\alpha$ , the sign must be chosen as a function of earlier measurement results, according to the rules given later in this subsection. When a measurement basis operator is not shown it should be taken to be  $HZ_0$  (corresponding to a measurement in the  $X$  basis,  $[|0\rangle \pm |1\rangle]/\sqrt{2}$ ).

To simplify the later description of feed-forward rules, assume that each edge in the graphical description of the computation is drawn either vertically or horizontally. By design, horizontal and vertical edges do not have the same effect on a cluster-state computation. This relates to the fact that horizontal and vertical edges correspond to very different components in a quantum circuit: wires and CPHASE gates respectively. (A consequence, for example, is that taking a graphical description of a cluster-state compu-

tation and rotating it 90 degrees will *not* give an equivalent computation). In addition, assume that each node in the graph has no more than one edge connecting it to the left, and no more than one edge connecting it to the right. Apart from these restrictions the graph can be arbitrary, so there is no need for the graph to be planar, for example.

It is convenient to categorize each qubit in a cluster state as being either *terminating* or non-terminating. A terminating qubit is defined to be one that is not connected to the right by any cluster edge. Thus, for example, the two rightmost qubits in Cluster (5.10) are terminating. Measurements of terminating qubits usually correspond to the final readout of a cluster computation (subject to adjustment due to feed-forward).

Let's consider a set of rules that can be used to correctly feed forward measurement results in stages 2 and 3 of a cluster-state computation. Useful to the discussion will be the definition of a data structure called the *Pauli frame*. The Pauli frame associates an operator  $X^x Z^z$ , for some integers  $x, z$  modulo 2, to each cluster qubit. That is, the Pauli frame is a tensor product of Pauli operators on the cluster qubits, but such that overall phases are ignored. (Note, global phases will routinely be ignored in any discussion related to the manipulation of Pauli frames, so, e.g.,  $XZ$  is regarded as equivalent to  $ZX$  or  $Y$ .) At the beginning of the computation the Pauli frame is initialized to the tensor product of identities,  $I \otimes \cdots \otimes I$ , and is then modified as each of the measurement results in stage 2 are obtained.

There are three rules for using the Pauli frame. When a qubit is measured, Rule (1) governs how to adjust the measurement basis based on the current Pauli frame. After any measurement, the Pauli frame must be updated according to Rule (2). If a measurement corresponds to a final readout of the computation, the result must be adjusted according to Rule (3).

*Rule (1): choosing the measurement basis.* For a qubit which has a “ $\pm$ ” sign in the specification of the measurement basis operator, the  $\pm$  sign should be replaced by  $(-1)^x$ , where the Pauli frame on the qubit being measured is  $X^x Z^z$ . So, e.g., if the specified measurement basis operator is  $HZ_{\pm\alpha}$ , then the qubit should be measured in the basis  $(HZ_{(-1)^x\alpha})^\dagger|0\rangle$ ,  $(HZ_{(-1)^x\alpha})^\dagger|1\rangle$ .

*Rule (2): updating the Pauli frame after measurement.* This rule is divided into two parts, which are applied depending on whether the qubit is connected vertically or horizontally by a cluster edge to another qubit. Suppose the Pauli frame of the qubit being measured is  $X^{x_1} Z^{z_1}$ , and the measurement result is  $m$ . Suppose also that the qubit connected to the right (if present) has Pauli frame  $X^{x_2} Z^{z_2}$ , and the

qubit connected vertically (if present) has Pauli frame  $X^{x_2}Z^{z_2}$ . Update Rule (a) is performed first, followed by Rule (b).

- (a) If there is a qubit connected vertically, and if Rule 2(a) has not already been applied for this particular vertical edge, then update the Pauli frame as follows:

$$z'_1 = z_1 + x_3 \quad (5.11)$$

$$z'_3 = z_3 + x_1. \quad (5.12)$$

- (b) If a qubit is connected horizontally to the right, update the Pauli frame as follows:

$$x'_2 = x_2 + z_1 + m \quad (5.13)$$

$$z'_2 = z_2 + x_1. \quad (5.14)$$


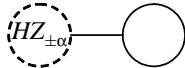
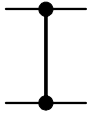


*Rule (3): adjusting final readout measurements.* Suppose a terminating qubit (i.e., a qubit containing the final readout of the computation) is measured in the  $X$  basis, and the outcome is  $m$ . Then, the adjusted outcome is  $m+z$ , where  $X^xZ^z$  is the Pauli frame of the qubit being measured (after Rule 2(a) has been applied, if necessary).

The three rules above can be derived straightforwardly from the commutation relations in Eqs. (5.6) and (5.7) in the previous subsection, but the details of this derivation are tedious so I will omit them.

Implicit in the rules above is an assumption that qubits are measured in a left-to-right order. If qubits are measured in some other order (because, for example, a Clifford-group computation is being performed) then Rule (2) for updating the Pauli frame must still be applied in a left-to-right order, which would usually mean delaying the application of Rule (2) until all measurement results to the left have been obtained.

Note finally that the task of converting from a quantum circuit to a cluster-state computation becomes almost trivial when we take advantage of the formalism introduced in this subsection. Assuming we start with a circuit of the appropriate form described in Subsection 5.1.2, the conversion involves simply substituting each operation in the circuit for an appropriate element of the cluster-state notation, as summarized in Table (5.1) and described as follows. For each operation in the circuit that prepares the  $|+\rangle$  state, we substitute a single node in the cluster. For each single-qubit gate  $HZ_\alpha$  we substitute a node, add an edge joining the nearest node to the left, and write the measurement-basis operator  $HZ_{\pm\alpha}$  in the left-neighbouring node. Each CPHASE is substituted for a vertical

Table 5.1: Summary of the substitution rules for converting a quantum circuit to a cluster-state computation.

Circuit operation	Corresponding cluster element
$ +\rangle$ —	
— $HZ_{\alpha}$ —	
	
	N/A

cluster edge. The two nodes that this vertical edge must join are found by finding the nearest single-qubit gate or preparation operation to the left, for each of the qubits the CPHASE gate acts upon in the circuit, and then finding the nodes that were substituted for those operations.  $X$ -basis measurements in the circuit do not need to be explicitly converted, since the results can be inferred from the measurement of terminating cluster qubits.

## 5.2 Optical quantum computing

This section is an introduction to optical quantum computing, with the focus being on linear-optical quantum computing.

The earliest proposals for optical quantum computing [Mil89, YKI88] combined linear-optical elements with nonlinear elements to produce universal quantum computation. Although linear elements such as beamsplitters and phase delays are relatively easy to implement, it is thought that the nonlinear elements required these proposals are impractical if not impossible.

A breakthrough was achieved by Knill, Laflamme and Milburn (KLM) [KLM01], who constructed a scheme which did away with the need for coherently-acting nonlinear el-

ements. Instead, their scheme achieves quantum computation using just linear optics together with photodetectors, single-photon sources and classical control. Experimental demonstrations [PFJF03, OPW<sup>+</sup>03, SJP<sup>+</sup>04, GPW<sup>+</sup>04, ZZC<sup>+</sup>04] of several of the basic elements of KLM have now been achieved.

The remainder of this section is structured as follows. I begin by describing some of the basic elements of optical quantum computing, including how linear optical elements can be used to achieve single-qubit gates, and follow with an overview of the KLM scheme.

### 5.2.1 Basic elements

Let's consider a physical implementation of quantum computation where each qubit is represented by a single photon. (It is also possible to imagine representing a qubit with a many-photon laser pulse, but I won't discuss this idea here. See, for example, [BvL05] and references therein for details.) Of all the possible distinguishable states that a photon could exist in, two states are selected to represent the basis states  $|0\rangle$  and  $|1\rangle$  of the qubit. The two most commonly-mentioned ways of making this selection are known as *spatial encoding* and *polarization encoding*.

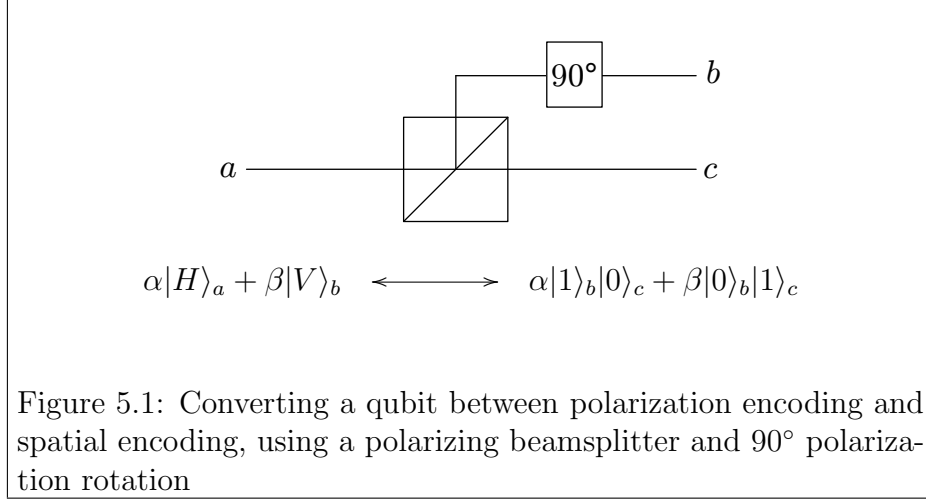
When a qubit is spatially encoded, it is in the state  $|0\rangle$  or  $|1\rangle$  depending on which of two spatially-separated paths the photon is travelling on. For example, the two paths could simply be separate optical fibres. Note that all other degrees of freedom of the photon, like polarization and pulse shape, must be constant and not depend on which of the two paths are taken. In a graphical depiction of an optical circuit, a spatially-encoded qubit is drawn as two lines, as follows:

$$\begin{array}{l} a \text{ —————} \\ b \text{ —————} \end{array} \quad (5.15)$$

The wires labelled  $a$  and  $b$  depict the two spatially-separated single-photon states defining the qubit. Note that it is most appropriate to describe  $a$  and  $b$  as being two separate photon *modes*. Use of this language reflects the fact that, physically, any number of photons could potentially share the state defined by  $a$  (or  $b$ ). In ket notation,  $n$  photons occupying the mode  $a$  is denoted  $|n\rangle_a$ . So, the state of an arbitrary spatially-encoded qubit is

$$\alpha|0\rangle + \beta|1\rangle \xrightarrow{\text{spatial enc.}} \alpha|1\rangle_a|0\rangle_b + \beta|0\rangle_a|1\rangle_b. \quad (5.16)$$

When a qubit is polarization-encoded, the qubit basis states  $|0\rangle$  and  $|1\rangle$  correspond to two orthogonal polarization states of a photon, denoted  $H$  for horizontal and  $V$  for



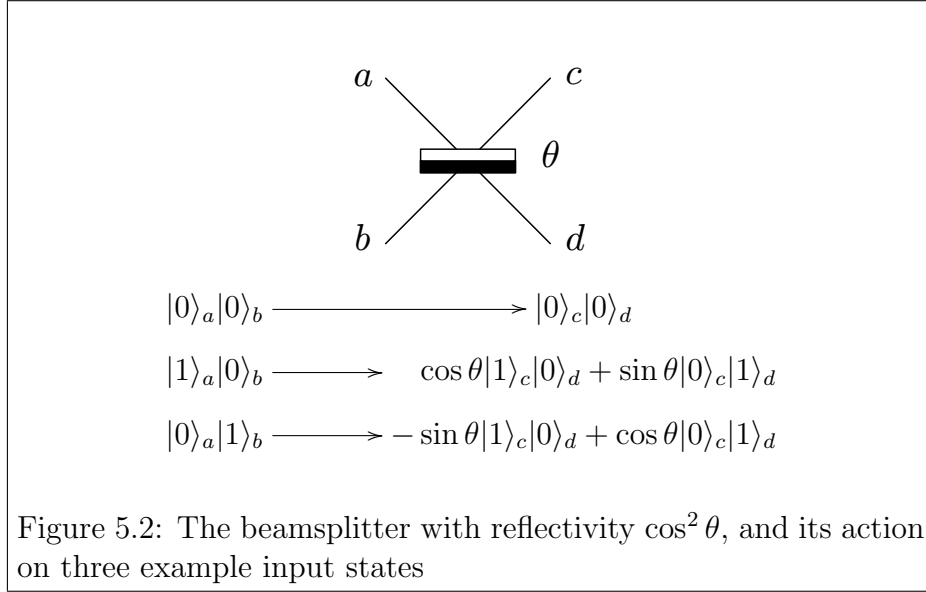
vertical. A polarization-encoded qubit is usually denoted by single line in an optical circuit, and the corresponding ket notation is

$$\alpha|0\rangle + \beta|1\rangle \xrightarrow{\text{polariz. enc.}} \alpha|H\rangle_a + \beta|V\rangle_a \quad (5.17)$$

In principle it is easy to convert between spatial and polarization encodings, by using a polarizing beamsplitter (which reflects horizontally-polarized light and transmits vertically-polarized light) together with a 90° polarization rotation, as shown in Figure 5.1. Given this ease of conversion, I shall hereafter focus my description on the case of spatially-encoded qubits.

One of the most notable advantages of optical quantum computing is that single-qubit gates can be performed with relative ease. In the case of spatial encoding, any single-qubit gate can be performed by an appropriate combination of *phase delays* and *beamsplitters*, in a manner outlined as follows. A phase delay applied to an optical mode effectively changes the path length of that mode, causing the state to pick up an extra phase  $e^{-i\phi}$  where  $\phi$  is proportional to both the change in path length and the number of photons in the mode. When a phase delay is applied to the mode representing the  $|0\rangle$  qubit basis state, it can easily be seen that the effect on the qubit's state is a  $Z$  rotation,  $e^{-i\theta Z/2}$  (up to an unimportant global phase). Note that this ease of performing  $Z$  rotations can be a blessing and a curse: unwanted  $Z$  rotations will occur unless path lengths are stabilized to well within one wavelength of light.

Other single-qubit gates are achieved with the inclusion of a beamsplitter, a device which partially reflects and partially transmits a beam of light. A beamsplitter is defined by a parameter  $\theta$  which governs its reflectivity: a proportion  $\cos^2 \theta$  of the intensity of the beam is reflected (and the rest is transmitted, assuming a lossless device). Figure 5.2



shows the graphical notation we adopt for the beamsplitter, together with a listing of the beamsplitter's action on zero and one-photon basis states. This action amounts to a  $Y$  rotation  $e^{-iY\theta}$  on a spatially encoded qubit. The  $Y$  rotation from a beamsplitter can be combined with  $Z$  rotations from phase delays to construct any arbitrary single-qubit gate (see, for example, Theorem 4.1 of [NC00]).

It will be useful to note one additional transformation rule for beamsplitters, as follows. A state with one photon in each of the input modes undergoes the transformation

$$\begin{aligned}
 |1\rangle_a |1\rangle_b &\xrightarrow{\text{beamsplitter}} -\sqrt{2} \cos \theta \sin \theta |2\rangle_c |0\rangle_d + \sqrt{2} \cos \theta \sin \theta |0\rangle_c |2\rangle_d \\
 &\quad + (\cos^2 \theta - \sin^2 \theta) |1\rangle_c |1\rangle_d.
 \end{aligned} \tag{5.18}$$

The above transformation rule is useful to know when considering implementations of two-qubit interactions. Two-qubit interactions will be discussed in later subsections.

Other basic elements required for optical quantum computation are photodetectors and single-photon sources. A photodetector is a highly sensitive measuring device that detects whether or not an optical mode is occupied by a photon. Optical quantum computing schemes usually require photodetectors which are *number-discriminating*, meaning that the device must not only measure if a mode is occupied, but also how many photons are in that mode. All photodetector measurements are destructive, in the sense that the photons in the mode are always lost during the measurement process.

The photon sources in an optical quantum computer must satisfy some rather challenging requirements. On demand, the source must be able to emit a single photon in a particular mode. Each photon emitted at different times and by the different sources in



the computer must all share the same properties such as carrier frequency, pulse shape, and polarization. Several schemes are being developed to satisfy these requirements, using effects such as conditional spontaneous parametric down-conversion, Raman emission to a cavity mode, and triggered decay in semiconductor quantum dots. For more details on the requirements and potential implementations of photon sources, see for example Section IV.B of [KMN<sup>+</sup>05] and references therein.

### 5.2.2 The KLM scheme

The elements described in the previous subsection are clearly sufficient to perform any single-qubit operation (state preparation, gates, and measurement) on optical qubits. Knill, Laflamme and Milburn showed how these same elements can be used to also achieve two-qubit interactions (and thus universal quantum computation), removing the need for nonlinear optical elements.

The KLM scheme provides an optical circuit for implementing a nondeterministic CPHASE gate. By nondeterministic, we mean that the constructed gate has an inherent probability of failure, even in the absence of external noise. In essence, this nondeterminism is the price that must be paid for using only linear optics.

To be precise, the KLM scheme provides not just a single optical circuit for implementing a CPHASE gate, but a whole family of such circuits, having varying values for the probability of success. The simplest circuit in the family has a success probability  $1/16$ , and is shown for the case of spatially-encoded qubits in Figure 5.3. The circuit takes a mode from each of the input qubits, interacts them via a beamsplitter, interacts the resulting modes with two ancilla photons via further beamsplitters, and measures four of the resulting modes with a photodetector. If certain specific measurement results are obtained as indicated in the figure, we say the gate has succeeded, and it can be shown that the output of the circuit is then equal to the input state transformed by a CPHASE operation. Any other combination of photodetector outcomes is considered a failure. The effect of a failure is equivalent to a computational-basis measurement on the two input qubits (subject to some caveats<sup>3</sup>), thus amounting to an irreversible corruption of the input state.

A gate which fails with probability  $15/16$  is clearly not much use as a direct substitute for an ordinary deterministic gate. However, KLM showed that the basic CPHASE circuit

---

<sup>3</sup>The exact effect depends on which of the photodetector outcomes is observed. In one case, the effect is precisely a computational-basis measurement, but in others it is more accurate to describe the effect as a *partial* measurement. In every case though, the effect can be turned into a full computational-basis measurement by directing the four output modes of the circuit into additional photodetectors.

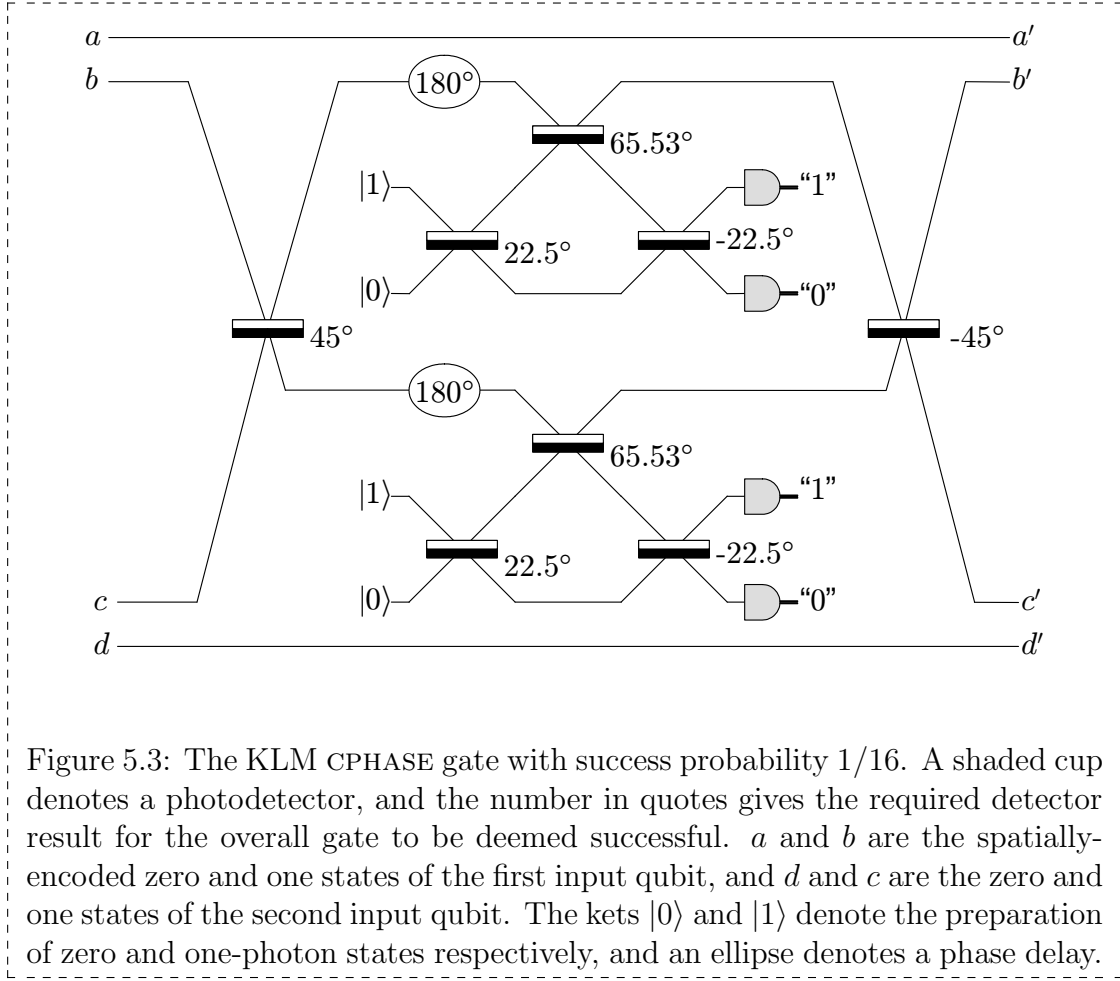


Figure 5.3: The KLM CPHASE gate with success probability  $1/16$ . A shaded cup denotes a photodetector, and the number in quotes gives the required detector result for the overall gate to be deemed successful.  $a$  and  $b$  are the spatially-encoded zero and one states of the first input qubit, and  $d$  and  $c$  are the zero and one states of the second input qubit. The kets  $|0\rangle$  and  $|1\rangle$  denote the preparation of zero and one-photon states respectively, and an ellipse denotes a phase delay.

construction of Figure 5.3 can be used as the basis for more elaborate constructions having higher success probability. These constructions incorporate the idea of *gate teleportation*, first introduced by Nielsen and Chuang [NC97] and developed by Gottesman and Chuang [GC99].

I won't go into the details of how the teleported CPHASE gate works, but the basic idea is as follows. Much of the circuit for implementing the teleported version of the gate is devoted to creating a particular entangled resource state. The particular part of the circuit devoted to resource-state creation uses many instances of the probability- $1/16$  CPHASE gate, but importantly does not involve the actual input qubits. Thus, if one of the probability- $1/16$  gates fail, as they usually will, then the resource-creation circuit is simply restarted without any detrimental effect on the input qubits. Eventually the resource state is created successfully, and it is then used to complete the teleported CPHASE. This is done by making the resource state interact with the input qubits in a specific way, and then performing photodetector measurements on all but four of the

resulting optical modes.

A teleported CPHASE gate nevertheless has some probability of failure, and again these failure events are flagged by certain values for the photodetector outcomes, and the effect is an unintentional computational-basis measurement on one or both of the inputs. However, the failure probability can be made arbitrarily low, if a sufficiently large resource state is used. For each positive integer  $n$ , there exists a version of the teleported CPHASE circuit which has a success probability  $n^2/(n+1)^2$  and involves the use of a  $4n$ -mode resource state.

On the face of it, this may not seem so bad. For instance, a CPHASE gate with a 95% probability of success can be achieved at the expense of using a 160-mode resource state. However, the circuit to construct the resource state involves applying a number  $n^2$  of the probability-1/16 CPHASE gates, which all must succeed or else the circuit must be restarted. So, factoring in the average number of times the circuit must be restarted, it's easy to show that the teleported CPHASE gate has a complexity which scales *exponentially* in the parameter  $n$ . As a result, even the 95%-probability version of the circuit has a cost which is for all practical purposes infinite.

KLM showed that this explosion in circuit complexity can in principle be largely avoided by using error-correcting codes. The idea is that one only needs to use a relatively small value for the parameter  $n$ , and then use efficient error-correction techniques to amplify the effective success probability.

However, even using such error-correction techniques, the complexity of the KLM gate remains daunting. For example, it has been estimated [HGMR04] that in order to achieve an error-corrected teleported CPHASE with effective success probability of 95%, roughly 10,000 optical elements need to be applied. Moreover, these figures assume that all optical elements are perfect, and not effected by external noise. Certain types of external noise, such as qubit depolarization, will be greatly *amplified* by the KLM error-correction scheme.

If and when a linear-optical quantum computer is built, its design is likely to differ significantly from the original KLM scheme. Since the KLM scheme was published, a range of researchers have been successful in improving various aspects. For example, simplifications of the basic low-probability KLM gate have been found by Ralph et al. [RWMM02] and by Knill [Kni02], and a more efficient version of the teleported KLM gate has been found by Franson et al. [FDF<sup>+</sup>02]. An extremely promising approach for further simplifying linear-optical quantum computing involves applying the techniques of cluster-state computing. This is the subject of the next section.

### 5.3 Optical cluster-state computation

Section 5.1 presented the cluster-state model as an abstract alternate model for quantum computation. The purpose of the current section is to describe how the cluster-state model might be of significant practical interest too, in the implementation of an optical quantum computer.

For KLM gates to be useful in a reliable implementation of circuit-model computation, the success probability of each gate must be made relatively high, and as discussed in the last section this requirement leads to very complicated optical circuits. However, Nielsen [Nie04] has shown that when KLM gates are used to create the initial state in a cluster-state computation, each gate need only have a relatively low probability of success. The result is that implementing a computation using the cluster-state model will take far fewer optical elements than the equivalent circuit-model implementation. Note that recent experiments [WRR<sup>+</sup>05] have demonstrated the construction of simple optical cluster states.

The remainder of this section is structured as follows. First I describe a simple property of cluster states that makes them exceptionally robust against the nondeterministic failure of optical gates. Then I briefly outline Nielsen's approach for taking advantage of this property, via a procedure for efficiently building optical cluster states. Finally I describe an improvement to Nielsen's approach due to Browne and Rudolph [BR05]. The key to their scheme is the use of a very simple nondeterministic gate known as the *fusion gate* in the creation of optical cluster states.

I do not go into great detail of the protocols of either Nielsen or Browne and Rudolph. Instead, in Chapter 6 I describe in detail a new protocol which combines some elements of both approaches.

#### 5.3.1 Effect of gate failures during cluster-state creation

Recall from the previous section that a KLM teleported CPHASE gate that uses a  $4n$ -mode resource state has a probability of failure equal to  $1 - n^2/(n+1)^2$ , and the effect of a failure is a heralded (known to the experimenter) computational-basis measurement on one or both of the inputs. This raises the question, how can the effect of inadvertent measurements be dealt with efficiently during a quantum computation? In general, when one qubit in a many-qubit entangled state is measured, there is no simple way of reversing the effect of that measurement other than to create the entire state again. Consider though what happens when a cluster-state qubit is measured in the computational basis. The

precise effect is described as follows:

**Result 5.1:** *When one qubit of a cluster state is measured in the computational basis, then the remaining state on the other qubits is also a valid cluster state, but with the Pauli operation  $Z^m$  applied to all neighbours of the measured qubit, where  $m$  is the measurement result. The graph of the new cluster state is equal to the original graph, except with all edges to the measured qubit deleted.*

This result may be obtained by repeatedly applying the following circuit identity, once for each edge connected to the measured node:

The diagram shows a circuit identity. On the left, two horizontal lines represent qubits. A vertical line with dots at both ends connects them, representing a CPHASE gate. The bottom qubit then has a measurement symbol (a diagonal line with an arrow) labeled  $Z$  and  $m$ . On the right, after an equals sign, the top qubit has a box labeled  $Z^m$  before the measurement symbol on the bottom qubit. The entire equation is labeled (5.19) on the right.

Eq. (5.19) shows that after a CPHASE gate has been applied (for example during creation of a cluster state), the application of a computational-basis measurement on one of the qubits will undo the effect of the CPHASE on the other qubit, subject to a possible  $Z$  operation.

Thus, imagine that an experimenter has created a very large cluster state, when one of the qubits is inadvertently measured in the computational basis. The effect of this measurement can be reversed by simply re-preparing the measured node in the state  $|+\rangle$ , re-creating the bonds to the measured node, and (if necessary) applying some Pauli operations.

Of course, if nondeterministic CPHASE gates are used to replace the missing edges, then these gates could fail too, causing in turn more edges to be lost. So some care must be taken to ensure that applying a CPHASE will on average grow a cluster rather than shrink it.

### 5.3.2 Nielsen's approach to efficient cluster creation

Nielsen [Nie04] used the simple observation described in the previous subsection to create an efficient method for building optical cluster states. Let's briefly review Nielsen's scheme.

The scheme applies to any desired cluster state having the property that each node has at most one vertical edge connected to it. (This is in addition to the usual restriction that each node has at most one horizontal edge to the left and one horizontal edge to the

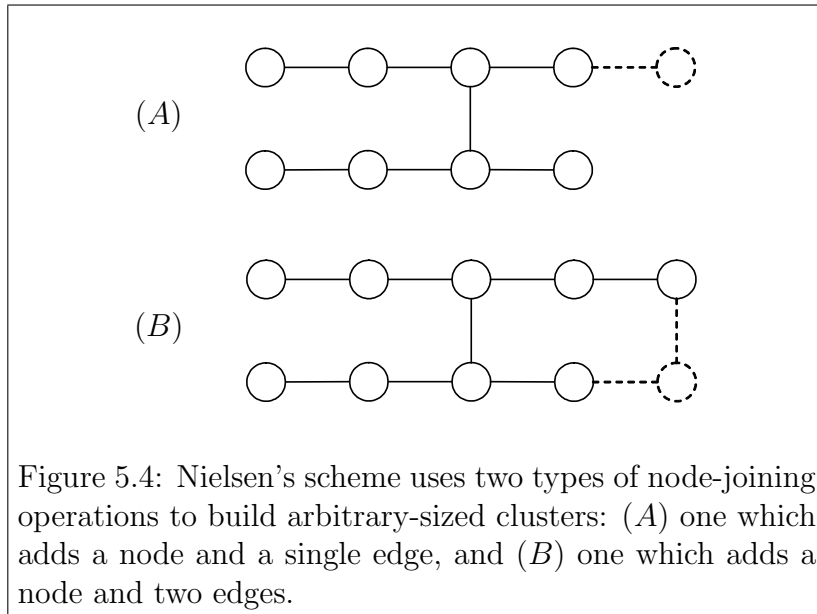


Figure 5.4: Nielsen's scheme uses two types of node-joining operations to build arbitrary-sized clusters: (A) one which adds a node and a single edge, and (B) one which adds a node and two edges.

right). When a cluster state satisfies this property, it's not hard to see that the procedure for creating the cluster can then be broken down into a series of steps which are each either of type "A" or "B" as shown in Figure 5.4. Operation *A* is defined to be one which adds a node to the cluster by creating a single edge to it, and operation *B* is defined to be one which adds a node to the cluster by creating two edges to it.

When either *A* or *B* is carried out successfully, exactly one node is added to the cluster. On the other hand, the failure of a CPHASE gate during operation *A* causes a node to be *removed* from the cluster, and failures during operation *B* cause as many as two nodes to be removed. So, the process of creating the cluster state is essentially a random walk. For this to be an efficient method for creating large clusters, we need to make sure the random walk has a statistical bias towards the desired direction of increasing the cluster size.

Nielsen considered the statistics that result from using the probability-4/9 version of the KLM gate throughout. In this case, the expected number of nodes added by operation *A* is  $1/3$ , and the expected number of nodes added by operation *B* is  $-1/9$  (that is, on average operation *B* causes nodes to be *lost*). However, it is reasonable to assume that for every time operation *B* is applied, operation *A* will have been applied at least once, so the overall average number of nodes added per joining operation throughout the construction of the entire cluster will be at least  $1/2 \times (1/3 - 1/9) = 1/9$ .

Thus, on average it will take approximately  $9N$  joining steps to successfully create an  $N$ -node cluster state using this method. Once the cluster is created, the computation can

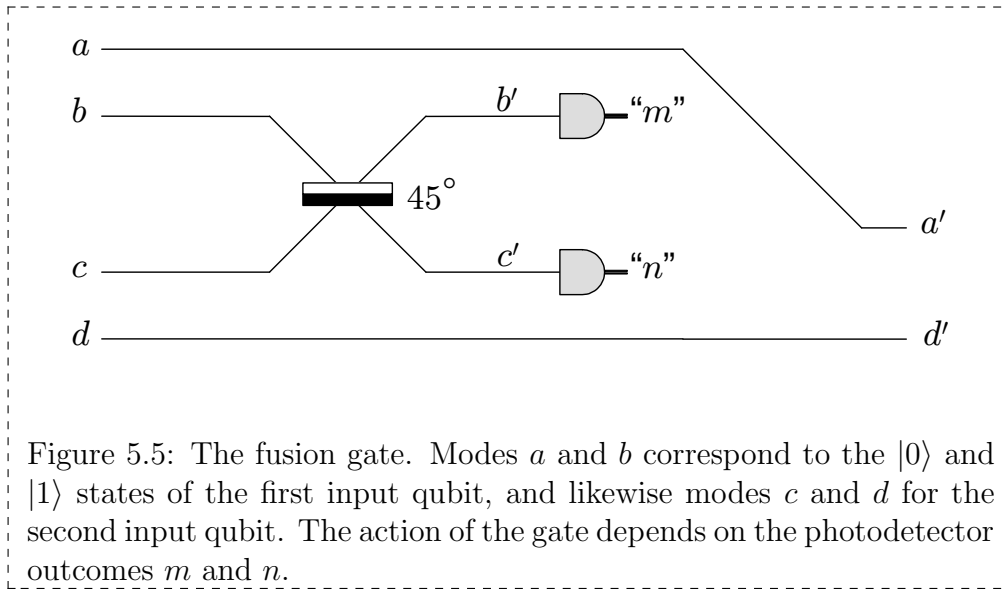


Figure 5.5: The fusion gate. Modes  $a$  and  $b$  correspond to the  $|0\rangle$  and  $|1\rangle$  states of the first input qubit, and likewise modes  $c$  and  $d$  for the second input qubit. The action of the gate depends on the photodetector outcomes  $m$  and  $n$ .

be completed deterministically using single-qubit measurements.

So one can use just a simple low-probability version of the KLM gate, yet still achieve an arbitrary quantum computation in an essentially deterministic manner, with a resource cost that scales gracefully with the computation size. The result is an implementation of optical quantum computing that is dramatically simpler than previous schemes.

Note that versions of the KLM gate with success probability less than  $4/9$  can also be used in Nielsen's scheme, if the procedure for building the cluster is modified appropriately. Roughly speaking, to ensure that the expected number of nodes added per joining operation remains greater than zero it is necessary that each joining operation attempts to add many nodes at a time instead of just one.

### 5.3.3 The fusion gate

Browne and Rudolph [BR05] simplified Nielsen's scheme by showing that cluster-state computing can be performed optically without using the KLM gate. It is possible to instead use the much simpler *fusion gate*. Although Browne and Rudolph define both "type-I" and "type-II" versions of the fusion gate, there exist simple modifications to their protocol that make the type-II version unnecessary. So, I shall review just the type-I version, and I shall henceforth refer to it as simply *the fusion gate*.

The optical circuit for the fusion gate is shown in Figure 5.5, for spatially-encoded qubits. The action which the fusion gate performs on the two input qubits can be derived as follows. Consider an arbitrary two-qubit state input to the gate, written as follows in

both ordinary qubit notation and spatially-encoded notation:

$$\begin{aligned} & \alpha|00\rangle + \beta|11\rangle + \gamma|01\rangle + \delta|10\rangle \\ & \xrightarrow{\text{spatial enc.}} \alpha|1010\rangle_{abcd} + \beta|0101\rangle_{abcd} + \gamma|1001\rangle_{abcd} + \delta|0110\rangle_{abcd}. \end{aligned} \quad (5.20)$$

(In the above, the notation  $|1010\rangle_{abcd}$  is short for  $|1\rangle_a|0\rangle_b|1\rangle_c|0\rangle_d$ , and likewise for the other terms.) Consider how each of the elements in the fusion gate circuit affect this state. The effect of the beamsplitter can be found by applying the transformation rules in Figure 5.2 and in Eq. (5.18). The result is the following state:

$$\begin{aligned} & \xrightarrow{\text{beamsplitter}} |10\rangle_{b'c'} \frac{1}{\sqrt{2}} (-\alpha|10\rangle_{a'd'} + \beta|01\rangle_{a'd'}) \\ & + |01\rangle_{b'c'} \frac{1}{\sqrt{2}} (\alpha|10\rangle_{a'd'} + \beta|01\rangle_{a'd'}) \\ & + |00\rangle_{b'c'} \gamma|11\rangle_{a'd'} \\ & + (|02\rangle_{b'c'} - |20\rangle_{b'c'}) \frac{\delta}{\sqrt{2}} |00\rangle_{a'd'}. \end{aligned} \quad (5.21)$$

Next, modes  $b'$  and  $c'$  are measured by the photodetectors. As is apparent from Eq. (5.21), there are five possible outcomes for the pair of measurement results  $(m, n)$ . If  $m = n = 0$  then the experimenter learns that the input to the gate was the computational basis state  $|01\rangle$ . Similarly if  $(m, n) = (0, 2)$  or  $(2, 0)$ , the experimenter learns that the input to the gate was the computational basis state  $|10\rangle$ . In other words, in these cases the action of the fusion gate is effectively a computational-basis measurement on the two input qubits, and in analogy to the KLM gate such an outcome is considered to be a failure of the gate.

Either of the other possible photodetector results  $(m, n) = (0, 1)$  or  $(1, 0)$  is considered to be a success, and in this case the output modes  $a'$  and  $d'$  will contain a valid spatially-encoded qubit. From the first two terms in Eq. (5.21) it is apparent that the state of the output qubit (up to normalization) will relate to the state of the input qubits via the operator  $|0\rangle\langle 00| + |1\rangle\langle 11|$  or  $-|0\rangle\langle 00| + |1\rangle\langle 11|$  for the two cases  $(m, n) = (0, 1)$  or  $(1, 0)$  respectively. For simplicity we can take the operation performed to be  $|0\rangle\langle 00| + |1\rangle\langle 11|$  in *both* cases, by assuming the experimenter applies a  $Z$  operation to the output for the case  $(m, n) = (1, 0)$ .

What is the effect on a cluster state when two of its nodes are input to a fusion gate? It can be shown that in this case the probabilities for success and failure of the gate are each  $1/2$  (whereas when a non-cluster state is input to a fusion gate, the probabilities of success and failure will be variable, and depend on the input state itself). From Result 5.1 the



effect of failure will be to remove both nodes from the cluster. The effect of a success can be stated as follows:

**Result 5.2:** *Suppose a fusion gate acts successfully on two qubits  $A$  and  $B$  of an  $N$ -qubit cluster state. Assuming that the input cluster state does not contain an edge joining  $A$  with  $B$ , then the state after the fusion gate is a valid  $(N-1)$ -qubit cluster state, with a graph identical to the input graph except that the nodes  $A$  and  $B$  are replaced by a single node  $C$  and all edges originally connected to  $A$  or  $B$  now connect to  $C$ .*

So, the fusion gate is very appropriately named! Its effect is to fuse two nodes of a cluster state, preserving the original links to those nodes. For example, a successful fusion between the two inner nodes of the four-qubit cluster state “ $\bigcirc - \bigcirc \quad \bigcirc - \bigcirc$ ” will yield the state “ $\bigcirc - \bigcirc - \bigcirc$ ”, whereas a failed fusion gate will yield “ $\bigcirc \quad \bigcirc$ ”.

The proof of Result 5.2 is outlined as follows. First note that it is straightforward to show that the fusion gate satisfies the following circuit identities, where  $F$  denotes a successful fusion gate:

$$\text{Circuit 1} = \text{Circuit 2} = \text{Circuit 3} \quad (5.22)$$

and

$$\text{Circuit 4} = \text{Circuit 5} \quad (5.23)$$

Now, consider the cluster state that the fusion gate is acting on, and imagine that it has been created by a circuit composed of CPHASE gates and  $|+\rangle$ -state preparations. Suppose the fusion gate is then appended to end of this circuit, acting on two of the qubits,  $A$  and  $B$ . Eq. (5.22) can then be used repeatedly to commute the fusion gate left past any CPHASE gates which act on either  $A$  or  $B$ , and the result will be to reattach all such CPHASE gates to the output qubit,  $C$ , of the fusion gate. Finally, Eq. (5.23) is used to commute the fusion gate past the two  $|+\rangle$ -state preparation operations on  $A$  and  $B$  at the left-hand side of the circuit, replacing them with a single  $|+\rangle$ -state preparation on  $C$ . It is clear that the resulting circuit creates a cluster state with nodes  $A$  and  $B$  fused together, as described in Result 5.2.

Fusion gates can be used to build cluster states of arbitrary size, so long as the experimenter has available one additional resource: a supply of Bell states. A Bell state is a maximally entangled state of two qubits, and is equivalent (up to local operations) to

the two-qubit cluster state  $\bigcirc\text{---}\bigcirc$ . (Note, it is not clear how a good source of Bell states might be implemented experimentally. This is currently an area of active investigation. See for example [SYA<sup>+</sup>06],[GRW06].)

So, for example, a three-link cluster chain  $\bigcirc\text{---}\bigcirc\text{---}\bigcirc$  can be created by the successfully fusing together two Bell pairs, and a five-link chain can then be created by fusing two three-link chains, and so on. More complex clusters can be built with similar ideas. Of course, during such joining procedures some of the attempted fusion gates will fail, so the procedure must be capable of efficiently dealing with such failures. Browne and Rudolph showed that efficient procedures do in fact exist for using the fusion gate to build large clusters suitable for quantum computation.

To summarize, Browne and Rudolph's scheme for optical quantum computation has the striking advantage of replacing the highly complex KLM gate with the simple fusion gate. Whereas tens of thousands of optical elements are used per KLM CPHASE in a circuit-model implementation of optical quantum computing, the same computation implemented via a cluster-state computation in Browne and Rudolph's scheme causes each original CPHASE gate to be replaced by only tens of optical elements (as estimated in [BR05]).

In Chapter 6, I describe in detail a new scheme for using fusion gates to build optical cluster states for quantum computation. The design criteria for this new scheme include the ability to deal efficiently with not only nondeterministic fusion gate failures but also external noise. Satisfying both these criteria turn out to be a rather difficult trade-off. Dealing with nondeterministic gate failure is best done by gradually building up the cluster state over many time steps, such as in Nielsen's random walk scheme. However to reduce the opportunity for external noise to be introduced it's best to minimize the time that each cluster qubit is idle. A reasonable compromise can be found by using techniques such as *microclusters*, *parallel fusion*, and *postselection*, as described in Chapter 6.

# Noise thresholds for optical cluster-state quantum computation

---

This chapter describes a detailed investigation of the value of the noise threshold for optical cluster-state quantum computation. We perform simulations to model the behaviour of an optical cluster-state error-correction protocol, where all optical components are modelled as simultaneously experiencing both photon loss and depolarizing noise. The depolarizing noise is a general proxy for all types of local noise other than photon loss noise. The strengths of the two types of noise are varied, and the resulting reliability of the error-correction protocol is measured. The main result of the chapter is a *threshold region* of allowed pairs of values for the two types of noise. Roughly speaking, our results show that scalable, reliable optical quantum computing is possible in the combined presence of both noise types, provided that the photon loss probability is  $< 3 \times 10^{-3}$  and the depolarization probability is  $< 10^{-4}$ . The fault-tolerant protocol described in this chapter involves a number of innovations, including a method for syndrome extraction known as telecorrection, whereby repeated syndrome measurements are guaranteed to agree.

*NOTE: This chapter is based on a published paper, [C. M. Dawson, H. L. Haselgrove, and M. A. Nielsen, “Noise thresholds for optical cluster-state quantum computation”, Phys. Rev. A 72:052306 (2006)]. A condensed version of that paper has also been published, [C. M. Dawson, H. L. Haselgrove, and M. A. Nielsen, Phys. Rev. Lett. 96:020501 (2006)]. Most of the contents of the paper published in Physical Review A (other than background material therein) has been reproduced in this chapter with only minor modifications.*

*My contribution to the collaboration can be summarized as follows. Dawson and I were jointly responsible for the design of the optical cluster error-correction protocol, and for the development of the simulation techniques. I was responsible for developing the maximum-likelihood decoding routine and designing the deterministic teleported error-correction protocol. I created the method for extracting a threshold curve from the raw*

*simulation results. I wrote one of the two independent versions of the simulator program. (My version consisted of approximately 12,000 lines of code, excluding whitespace but including comments). Most of the long version of the paper was written jointly by Nielsen and myself. Exceptions are that Nielsen was solely responsible for most of the introduction and conclusion, and I was solely responsible for the appendix on telecorrection, the section analysing resource usage, the section describing the deterministic protocol, and the results sections.*

## 6.1 Introduction

We have seen in Chapter 5 that there exist promising proposals for combining the cluster-state model of quantum computation with techniques of linear-optical quantum computation. However, for such proposals to be considered credible approaches to fully scalable quantum computation, it is necessary to consider the effects of noise. In particular, it is necessary to establish that a *noise threshold theorem* holds for the optical cluster-state proposals. Recall from Section 2.3 that a noise threshold theorem proves the existence of a nonzero *noise threshold* value, such that provided the amount of noise per elementary operation is below this level, it is possible to efficiently perform a quantum computation of arbitrary length, to arbitrary accuracy, using appropriate error-correction techniques. In the standard quantum circuit model of computation such a threshold has been known to exist since the mid-1990s (for references, see Subsection 2.3.2 of this thesis, or Chapter 10 of [NC00]). However, the optical cluster-state proposals are not based on the circuit model, but rather on the cluster-state model of computation, and thus *a priori* it is not obvious that a similar noise threshold need hold.

Fortunately, recent work [ND05, Rau03, AL05] has shown that the fault-tolerance techniques developed for the circuit model can be adapted for use in the cluster-state model, and used to prove the existence of a noise threshold for noisy cluster-state computing. The earliest work [ND05, Rau03] established the *existence* of a threshold for clusters, without obtaining a value. [AL05] argued that in a specific noise model, the cluster threshold is no more than an order of magnitude lower than the threshold for circuits. The most recent work [RHG05b] combines ideas from cluster-state computing with topological error-correction to obtain a cluster threshold. However, neither [AL05] nor [RHG05b] are of direct relevance to the optical cluster-state proposal, since they make use of deterministic entangling gates, which are not available in linear optics, and the noise model does not include any process analogous to photon loss.

In the work described in this chapter, the value of the noise threshold for optical cluster state computing is studied in detail. We use numerical simulations to estimate the threshold for a particular fault-tolerant protocol, for two different quantum codes.

Our threshold analysis is tailored to the dominant sources of noise in optical implementations of quantum computing. In particular, our simulations involve three different sources of noise: (a) the inherent nondeterminism of the entangling gates used to build up the cluster; (b) photon loss; and (c) depolarizing noise. The strength of noise source (a) is regarded as essentially fixed, while the strengths of (b) and (c) are regarded as variables that can be changed by improved engineering. Note that most existing work on thresholds (e.g., [AL05, RHG05b, Kni05, Ste03]) in either clusters or circuits focuses on abstract noise models based on depolarizing noise, and neglects sources (a) and (b).

Noise sources (a) and (b) likely dominate actual experiments, and our protocol attempts to cope with these very efficiently. The protocols for decoding and correction can be made to take advantage of the knowledge the experimenter has of the locations of error types (a) and (b). For example, the well-known Steane 7-qubit code is usually used to correct a depolarization error on a single qubit. A more efficient use of the code is possible, in which it is used to correct photon loss or nondeterministic gate failure errors on as many as *two* qubits.

Although noise sources (a) and (b) will dominate, sources of noise other than (a) and (b) will also be present in experiments, and so it is important that our fault-tolerant protocol and analysis also deals with those. This is why we include noise source (c), as a proxy for all additional noise effects. Of course, in practice it is unlikely that depolarizing noise will be a particularly good model for the other noise sources. However, as discussed in Section 2.2, standard results in the theory of quantum error-correction show that the ability to correct depolarizing noise implies the ability to correct essentially all reasonable physical noise models, and so depolarization is a good proxy for those other effects.

A prior work [VBR05] (c.f. [RHG05a]) has calculated a threshold for optical quantum computation when the only source of noise is photon loss. In real experiments noise sources other than photon loss are present, such as dephasing, and protocols such as [VBR05, RHG05a] will amplify the effects of such noise at the encoded level. Thus, even if the original noise strength is very weak, encoding may amplify the noise to the point where it dominates the computation. By contrast, our protocol protects against both photon loss and depolarizing noise, and by standard fault-tolerance results thus automatically protects against arbitrary local noise, including dephasing (in any basis), amplitude damping, etc.

Because our model includes multiple noise parameters, we do not obtain a single value

for the threshold, as in most earlier work. Instead, we obtain a threshold *region* of noise parameters for which scalable quantum computing is possible. The main outcome of this chapter is a series of threshold regions, with the different regions corresponding to varying assumptions regarding the relative noise strength of quantum memory, and the use of different quantum error-correcting codes. Qualitatively, we find that our fault-tolerant protocols are substantially more resistant to photon loss noise than they are to depolarizing noise, with threshold values of approximately  $6 \times 10^{-3}$  for photon loss noise (in the limit of zero depolarization noise), and  $3 \times 10^{-4}$  for depolarizing noise (in the limit of no photon loss). When both types of noise are present in the system, a typical value in the threshold region has a strength of  $3 \times 10^{-3}$  for photon loss noise, and a depolarization probability of  $10^{-4}$ .

Our fault-tolerant protocol involves a number of innovations in addition to those already described, including: (1) the development of special techniques to deal with the inherent non-determinism of the entangling optical gates; (2) heavy use of the ability to parallelize cluster-state computations [RBB03], and the ability to do as much of the computation off-line as possible; and (3) as a special case of the previous point, we develop a new method for doing fault-tolerant syndrome measurement which we call *telecorrection*. This has the striking property that repeated measurements of the syndrome are *guaranteed to agree* (which helps increase the threshold), unlike in standard protocols, where measurements only sometimes agree.

The structure of the chapter is as follows. Section 6.2 describes our assumptions about the physical setting: what physical resources are allowed, what quantum gates can be performed, and what noise is present in the system. Section 6.3 describes briefly how we simulate noisy cluster-state computations. This is a surprisingly subtle topic, due to the multiple noise sources in our model, which is why it merits a separate section. Section 6.4 describes the details of the fault-tolerant protocol that we simulate, and presents the results of our simulations, including threshold regions for two different quantum codes. Section 6.5 concludes.

## 6.2 Physical setting

In this section we describe in detail both what physical operations we assume can be done, and our model of noise.

**Physical operations:** We assume the following basic elements are available. First, a source of polarization-entangled Bell pairs (specifically, the state  $[|0\rangle \otimes (|0\rangle + |1\rangle) +$

$|1\rangle \otimes (|0\rangle - |1\rangle)/2$  in qubit notation). Physically, these can be produced in a number of different ways, but the details don't matter to our analysis. Second, single-qubit gates can be performed on the optical qubits. Physically, this can be done using linear optics, following KLM. Third, the fusion gate of Browne and Rudolph can be applied. Fourth, efficient polarization-discriminating photon counters capable of distinguishing 0, 1 and 2 photons are available. These can be used to effect measurements in the computational basis, and are also used to verify the success or failure of the fusion gate. Note that having single-qubit gates and computational basis measurements allows us to effect single-qubit measurements in an arbitrary basis. Single-qubit gates do not appear explicitly in our protocol, rather only as part of single-qubit measurements.

For simplicity, we assume that all the basic operations take the same amount of time, and we thus describe our protocol in terms of a sequence of *time steps*. In a time step, each qubit may be involved in at most one operation. As a consequence, an important additional element that must be available is the quantum memory “gate”, during which an optical qubit ideally does nothing for a time step, but may still be affected by noise. Physically, it's not yet clear what the best way of implementing such a quantum memory will be.

We've described the basic elements in our model of quantum computation. However, a number of important additional assumptions are made about how these elements can be put together. First, we assume that any two qubits can be interacted directly. This is reasonable, given the ease of moving photonic qubits from one location to another. Second, we assume the ability to perform operations on all the qubits in parallel. Third, we assume the availability of classical computation, communication, and feed-forward, all on a timescale fast compared with the unit time step. The feed-forward requirement is particularly demanding, since it requires us to decide which qubits interact in a time-step, based on the results of measurements in the previous time-step. To some extent, this requirement is imposed merely to simplify our simulations, and it seems likely that the requirement can be relaxed, but this remains a topic for further investigation.

**Noise model:** We now describe our model of the physical sources of noise. As stated in the introduction, our protocol deals with three kinds of noise: (a) the inherent nondeterminism of the fusion gates; (b) photon loss; and (c) depolarizing noise. We now describe these in more detail.

The noise due to the inherent non-determinism of the fusion gate was described in Chapter 5. Recall that with probability 50% the gate succeeds, causing the fusion of two cluster nodes, while with probability 50% it fails, and the two qubits are measured in the

computational basis.

We assume a single parameter  $\gamma$  controls the strength of the photon loss.  $\gamma$  is the probability per qubit per time step of a photon being lost. We assume this probability is independent of the state of the qubit, and that photon loss affects every basic operation in our protocol, as follows:

- Bell-state preparation: After the state has been prepared, each of the two qubits independently experiences photon loss with probability  $\gamma$ .
- Memory, single-qubit, and fusion gates: Before the gate each input qubit experiences photon loss with probability  $\gamma$ . In the case of the fusion gate, which has two inputs, we assume the loss probabilities are independent. Later in the chapter we also investigate the case where the photon loss rate for memory gates has been suppressed relative to the other gates.
- Measurement: Before measurement we assume photon loss occurs with probability  $\gamma$ . Physically, this can model both the loss of photons from the relevant optical modes, and also detector inefficiencies.

It is worth noting that detector inefficiencies are currently much worse than other sources of photon loss, and it could be argued that detector inefficiency and other photon loss rates should be treated as independent parameters (or, alternately, that all other photon loss rates be set to zero). However, it is clear that turning off or turning down photon loss noise in locations other than before measurement can only improve the threshold, and so we have used the more pessimistic model described above. In fact, it can be shown that photon loss occurring in locations other than before measurements propagates to become equivalent to photon loss before measurements. Thus, the model in which photon loss occurs only before measurement should have a threshold for photon loss noise several times higher than the results we report.

Note also that we have chosen a model of photon loss during Bell-state preparation that acts independently on each qubit in the pair. It would perhaps be more physically realistic for loss to occur in a manner that is highly correlated between the two qubits in the Bell pair (i.e., making it more likely that both photons in the pair are lost as opposed to just one). However, the design of our fault-tolerant protocol ensures that we can always detect situations where both photons in a Bell state are lost, and thus this type of coincidental loss has no negative effect on the threshold. So, our choice of uncorrelated photon loss is the more pessimistic of the two alternatives.



Similarly to photon loss, we assume a single *depolarizing parameter*  $\epsilon$  controls the strength of the depolarizing noise. We assume depolarization affects every basic operation, as follows:

- Bell-state preparation: After the state has been prepared, the joint state of the two qubits is depolarized as follows: with probability  $1 - \epsilon$  no error occurs, while with respective probabilities  $\epsilon/15$  one of the 15 non-identity Pauli product operators  $IX, XX$  etcetera, are applied.
- Memory and single-qubit gates: Before each gate we depolarize as follows: with probability  $1 - \epsilon$  no error occurs, while with respective probabilities  $\epsilon/3$  one of the 3 non-identity Pauli operators  $X, Y$  and  $Z$  are applied.
- Fusion gate: The joint state of the two qubits is depolarized with parameter  $\epsilon$  (in the same way as described for Bell-state preparation above) before being input to the gate.
- Measurement: Before measurement the qubit is depolarized with parameter  $\epsilon$  (in the same way as described for memory and single-qubit gates above).

Note that in our noise model noise occurs before or after operations. In a real physical setting, noise will also occur during gate operations. However, standard fault-tolerance techniques (see, e.g., [KLZ98]) can be used to show that noise during an operation can be regarded as completely equivalent to noise before or after that operation.

The noise model we have described is obviously an approximation to reality, and is incomplete in various ways. For example, it is difficult to justify on physical grounds using the same two noise strength parameters for all operation types. Also, additional noise sources that may have an effect in real implementations include dark counts, dephasing, and non-local correlations. However, it can be shown that the fault-tolerant protocol we implement automatically provides protection against such noise sources. We haven't done a detailed investigation of the threshold for these noise sources, or for the case of different noise strengths for different operation types, but believe that the results would be in qualitative agreement with the results of this chapter.

## 6.3 Method for simulating a noisy cluster-state computation

In this section we explain how to simulate a noisy cluster-state computation. We do not yet describe the details of the fault-tolerant protocol, leaving those to the next section. However, the protocol is simulated using essentially the techniques we now describe.

We concentrate on the case when the errors are solely Pauli-type errors. It turns out that a simple modification of the techniques used to describe these errors can be used to describe the non-deterministic failure of fusion gates, or photon loss. However, we defer this discussion to the next section as it depends on some details of the fault-tolerant protocol.

### 6.3.1 Example

We begin with a toy example of a noisy cluster-state computation in which noise is introduced at just a single location, and we study how this affects the remainder of the computation. This example will motivate our later abstractions and the data structures used to model noise.

The example is a two-qubit cluster-state computation:

$$\begin{array}{c} \text{---} \bigcirc \text{---} \bigcirc \text{---} \\ | \quad H \quad | \end{array} \quad (6.1)$$

We imagine that the two qubits of the cluster are perfectly prepared. After preparation, we suppose a single Pauli  $Z$  error corrupts the first qubit, so the actual physical state of the cluster is related to the ideal state by an overall error  $Z \otimes I$ . Now we suppose a perfect  $H$  operation and computational basis measurement is carried out on the first qubit, yielding an outcome  $m = 0$  or  $1$ . It will be convenient to regard the combined Hadamard and measurement as a single operation, a perfect measurement in the  $X$  basis. This completes our example computation.

At the end of the computation, the experimenter believes the resulting state of the second qubit is  $X^m H|+\rangle$ . However, a calculation shows that the actual state is  $X^{m+1} H|+\rangle$ . Mathematically, there are two different ways we can think about this resulting state:

- Measurement of the first qubit propagates the  $Z$  error on that qubit to the second qubit and causes it to become a physical  $X$  error on that qubit.
- Measurement of the first qubit causes the  $Z$  error on that qubit to turn into an  $X$  error in the Pauli frame of the second qubit, but eliminates all physical errors.

While these points of view are equivalent, we will take the second point of view, as it turns out that in more complex examples, it is this point of view which gives the simplest description of what is going on.

This analysis can be repeated for the case where, instead of a  $Z$  error, we had a single  $X$  error occur on the first qubit. However, this case is more trivial, because the  $X$  error followed by the perfect  $X$  basis measurement is equivalent to a perfect  $X$  basis measurement alone, and thus the resulting state is  $X^m H|+\rangle$ , as expected by the experimenter. Thus, in this case the effect of measurement is simply to eliminate the physical error.

### 6.3.2 General description of the introduction and propagation of Pauli noise

Our simulations of Pauli noise in cluster-state computation are based on generalizations of the concepts introduced in the previous example. There are two basic data structures that we keep track of. The first is the *physical error* in the state of the cluster. This is a tensor product of Pauli matrices, one for each cluster qubit. This begins as the identity at every qubit, and we will describe below how it is modified as noise and gate operations occur.

The second data structure is the *error in the Pauli frame* of the cluster. Again, this is a tensor product of Pauli matrices, one for each qubit in the cluster. It begins as the identity at every qubit, and will be modified during the simulation according to rules described below.

It is notable that our description of noisy cluster-state computation is thus based entirely on products of Pauli operators. What makes this description possible is that all the operations we simulate are Clifford-group operations, and this ensures that the errors remain Pauli errors at all times. It is also worth noting that in our simulations we do not keep track of the actual state of the cluster, nor of the Pauli frame, but only of the errors in each. This is because the aim of our fault-tolerance simulations is to determine various statistics associated to these errors, and the actual state of the cluster is not of direct importance.

Note that in our description, physical errors and Pauli frame errors are not generally interchangeable, since they undergo different propagation rules (described later in this subsection) and thus may have different effects on the final state of the computation. Errors in the Pauli frame are introduced only as a result of noise-affected measurements in transport circuits, and propagate as a result of the Pauli frame update rules (described in Subsection 5.1.3) that the experimenter applies. Physical errors describe noise on the

state itself, and propagate according to how the Pauli matrices commute through the various quantum operations performed on the state.

As we have described, the physical error and Pauli frame error are products of Pauli operators on all the remaining cluster qubits. It is often convenient to focus on one or just a few cluster qubits rather than the entirety. For this purpose we will refer to *local physical errors* and *local Pauli frame errors*, which are just the corresponding Pauli operators for a specified qubit or qubits. It will also be convenient to describe such local errors either in matrix form as  $X^x Z^z$ , or in terms of the pair  $(x, z)$ , and we will use these descriptions interchangeably. So, for example, we may refer to either  $X^x$  or simply  $x$  as the  $X$  error. We will routinely ignore global phase factors in our description of errors, so, e.g.,  $XZ$  and  $ZX$  are regarded as equivalent.

It is important to remember the definition of a *terminating qubit* from Subsection 5.1.3. Recall, a cluster qubit is terminating if it has no horizontal bonds. The significance of terminating qubits in a cluster-state computation is that measurement of these qubits reveals the outcomes of the computation. By contrast, measurement of non-terminating qubits merely reveals information which can be used to propagate quantum information to other parts of the cluster.

We now describe the rules for updating both our data structures for each of the possible operations that can occur during a noisy cluster-state computation.

*Update rule for depolarization event:* The physical error is updated by matrix multiplication by the appropriate randomly-chosen error (e.g.,  $X, Y$  or  $Z$ ). The error in the Pauli frame is not affected.

*Update rule when a non-terminating qubit is measured in the  $X$  basis:* It is easiest to describe this by describing two separate cases: the case when there is a single horizontal bond attached to the qubit being measured, to the right; and the case where both vertical and horizontal bonds are present.

Suppose the qubit being measured has a single horizontal bond attached, to the right. Suppose before the measurement the local physical error on the qubit being measured is  $X^{x_{1p}} Z^{z_{1p}}$ , and the local Pauli frame error is  $X^{x_{1f}} Z^{z_{1f}}$ . Suppose the corresponding values for the qubit on the right are  $X^{x_{2p}} Z^{z_{2p}}$  and  $X^{x_{2f}} Z^{z_{2f}}$ . After the measurement the updated values for the local physical and Pauli frame errors of the qubit on the right are as follows:

$$x'_{2p} = x_{2p} \tag{6.2}$$

$$z'_{2p} = z_{2p} \tag{6.3}$$

$$x'_{2f} = x_{2f} + z_{1p} + z_{1f} \tag{6.4}$$

$$z'_{2f} = z_{2f} + x_{1f}. \tag{6.5}$$

These rules are derived from our description of the transport circuit and the rules for updating the Pauli frame in Subsection 5.1.3, along essentially the same lines as the example in Subsection 6.3.1. As in the example, we see that  $X$  physical errors on the qubit being measured are eliminated, and  $Z$  physical errors propagate to become  $X$  errors in the Pauli frame. Similar reasoning shows that  $X$  errors in the Pauli frame of the qubit being measured propagate to become  $Z$  errors in the Pauli frame of the attached qubit, and vice versa for  $Z$  errors in the Pauli frame.

Suppose the qubit being measured has a vertical bond, and a rightward horizontal bond. Suppose we label the qubits 1 (qubit being measured), 2 (qubit to the right), and 3 (qubit attached by vertical bond). We will denote the values for the local physical error and local Pauli frame error by  $x_{jp}, z_{jp}$  and  $x_{jf}, z_{jf}$ , respectively, where  $j$  labels the qubit. We update these in two stages, with the update method derived from the two stages for updating the Pauli frame when a vertical bond is present, as described in Subsection 5.1.3. The first stage is associated to the vertical bond. We set:

$$x'_{1p} = x_{1p} \quad (6.6)$$

$$z'_{1p} = z_{1p} \quad (6.7)$$

$$x'_{1f} = x_{1f} \quad (6.8)$$

$$z'_{1f} = z_{1f} + x_{3f} \quad (6.9)$$

$$x'_{3p} = x_{3p} \quad (6.10)$$

$$z'_{3p} = z_{3p} \quad (6.11)$$

$$x'_{3f} = x_{3f} \quad (6.12)$$

$$z'_{3f} = z_{3f} + x_{1f}. \quad (6.13)$$

The local physical and Pauli frame errors for qubit 2 are not changed during this step. For the second stage we behave as though the vertical bond has been deleted, and use our new values for the physical and Pauli frame errors as input to the update rules for the case of a horizontal bond, Equations (6.2)-(6.5).

*Update rule for measuring terminating qubits in the  $X$  basis:* We first describe the update rules for the case when the terminating qubit has no vertical bonds attached. The update rule is to compute the *total error*, which we define as the product of the local physical and Pauli frame errors on that qubit. The qubit is then deleted from the cluster, and its local physical and Pauli frame errors are deleted from the corresponding data structures. The total error  $X^x Z^z$  determines whether or not the measurement outcome (e.g., of syndrome information) contains an error. Since the measurement is in the  $X$

basis, the error in the measurement is simply  $z$ . The aim of our fault-tolerant simulations will be to determine various statistics associated to this total error.

Consider now the case when the terminating qubit has a vertical bond also, before being measured in the  $X$  basis. In this case we simply follow the rules of Equations (6.6)-(6.13) for updating the errors, and then treat the qubit as though the vertical bond has been deleted, and apply the rules described earlier for treating a terminating qubit.

*Update rule for measuring a qubit in the  $Z$  basis:* In our protocols we only ever do such a measurement on non-terminating qubits, and so restrict our attention to this case. In an ideal cluster-state computation the effect of a  $Z$  measurement with outcome  $m = 0$  or  $1$  is effectively to remove that qubit from the cluster, and apply  $Z^m$  to all neighbouring qubits (see Result 5.1 in Subsection 5.3.1). An experimenter getting a result  $m$  can therefore update the Pauli frame of neighbouring qubits by multiplying each by an extra factor of  $Z^m$ .

To describe the update rule in this case, we define the total error to be  $x_t = x_p + x_f$ ,  $z_t = z_p + z_f$ , where subscript  $ps$  denote physical errors, and subscript  $fs$  denote Pauli frame errors. The error in the measurement outcome will be  $x_t$ , since  $Z$  flips do not affect  $Z$  basis measurements. So the update rule is merely to discard the local physical and Pauli frame errors from our overall physical error and Pauli frame error, and to introduce an additional  $Z^{x_t}$  Pauli frame error on all neighbouring qubits.

*Update rules for the fusion gate:* We separate our analysis into cases when the fusion gate is unsuccessful and successful. When unsuccessful the fusion gate results in a  $Z$  basis measurement being applied to the qubits we are attempting to fuse. This case can be described by the rules stated above for  $Z$  basis measurements.

When the fusion gate is successful we update as follows. We label the qubits being fused as qubit 1 and 2. It turns out that in our fault-tolerant protocol we *never* fuse qubits which have Pauli frame errors. Thus we can assume that the initial errors on the qubits being fused are simply  $x_{jp}, z_{jp}$ , where  $j = 1, 2$  labels the qubit. For distinctness we will call the physical and Pauli frame errors of the output qubit  $x_{3p}$  and  $z_{3p}$ ; the 3 is merely for clarity, and does not indicate the creation of a new physical qubit. The update rule is as follows:

- For each qubit neighbouring qubit 1, we add  $x_{2p}$  to the  $Z$  physical error.
- Vice versa, for each qubit neighbouring qubit 2, we add  $x_{1p}$  to the  $Z$  physical error.
- $x_{3p} = x_{1p} + x_{2p}$ ,
- $z_{3p} = z_{1p} + z_{2p}$ .

These rules follow straightforwardly from the definition of the fusion gate.

## 6.4 Fault-tolerant protocol

### 6.4.1 Introduction

In this section we describe in detail our fault-tolerant protocol, and the threshold results we obtain. We begin in this subsection with a brief discussion of antecedents to our work. We begin describing the technical details of the protocol in the next subsection.

A large body of numerical work aimed at determining the threshold has been performed by various researchers. Especially notable is the work by Steane [Ste03], who did the first detailed numerical investigations of the threshold, and the recent work by Knill [Kni05], who has established the best known thresholds in the standard quantum circuit model. Many of our techniques are based on those described by Steane. We will also see below that there is some overlap with the techniques of Knill. Of course, the very different nature of optical cluster-state computation demands many new techniques, and care must be taken in comparing the values of thresholds in this model and the standard quantum circuit model.

As explained in the introduction, Nielsen and Dawson [ND05] proved the existence of a threshold for optical cluster-state computing (c.f. [Rau03, AL05, RHG05b]). The basic idea of the construction in [ND05] is to show that if we take a quantum circuit, convert it into a fault-tolerant form using multiple layers of concatenated coding, and then simulate the circuit using optical cluster states, the resulting noisy optical cluster-state computation is itself fault-tolerant. This proof relied on an important theorem of Terhal and Burkard [TB04] establishing a threshold for non-Markovian noise in the standard circuit model.

When we began the work described in this chapter, our intention was to apply the procedure described in [ND05] to a fault-tolerant circuit protocol similar to that considered by Steane [Ste03] (i.e., the protocol reviewed in Subsection 2.3.1 of this thesis). In the event, our protocol involves substantial improvements over this basic procedure, is manifestly fault-tolerant, and gives a much better threshold. The improvements include: optimizing our treatment of photon loss and non-determinism; exploiting the ability to premeasure and parallelize parts of the cluster-state computation; and taking advantage of the ability to premeasure clusters in order to improve ancilla creation. All these improvements are described in detail below.

## 6.4.2 Broad picture of fault-tolerant protocol

In this subsection we outline our fault-tolerant protocol. The protocol is split into two main parts.

The first part is a cluster-based simulation of a variant of Steane’s fault-tolerant protocol. We have modified Steane’s protocol to deal with the non-deterministic nature of the optical gates, and introduced several cluster-based tricks to improve the threshold. This protocol and the results of our simulations are described in Subsection 6.4.3.

The second part is a deterministic gate-based protocol, whose purpose will be explained in the paragraphs below. This protocol is also based on Steane’s methods, again with some substantial variations. This protocol and the results of our simulations are described in Subsection 6.4.4.

The reason for using the two protocols is that the actual cluster threshold is obtained by concatenating a single encoded level of the cluster protocol with multiple levels of the deterministic protocol. This works because our multiply concatenated fault-tolerant cluster protocol is equivalent to building up a fault-tolerant implementation through multiple levels of concatenation in the circuit model, and then replacing each gate in the bottom level by a clusterized equivalent.

To obtain the overall behaviour of such a protocol, it is not feasible to directly simulate the multiply concatenated computation. Instead, we do one simulation of the clusterized protocol at just a single level of encoding, and another of the deterministic protocol, again at a single level of encoding. We then make an argument allowing us to use the data obtained from these two protocols to estimate the overall behaviour if multiple layers of concatenation had in fact been used. The details of how this is done are described in Subsection 6.4.5.

## 6.4.3 The cluster-based protocol

Our cluster-based protocol performs multiple rounds of clusterized quantum error-correction, effectively implementing a fault-tolerant quantum memory. Following previous numerical work on the threshold (e.g. [Ste03, Kni05]) we do not simulate dynamical operations at the encoded level. However, our simulations could easily be varied to implement encoded Clifford group operations. For the codes we consider, encoded Clifford group operations have a transversal implementation, and so including them would only marginally add to the operation count of the already complex error-correction protocol, and would thus leave the noise threshold essentially unchanged.



Computational universality requires the ability to perform at least one type of encoded non-Clifford group operation. However, this is difficult to simulate, and previous workers [Kni05, SDT06] have argued that it changes the threshold very little. A version of the argument is briefly summarized as follows. So long as the physical noise rate is below the threshold for Clifford group operations, the techniques of *state injection* [Kni05, Kni04] and *magic state distillation* [BK06] can be used to reliably apply encoded non-Clifford group operations as well. In [BK06] it is shown that when one has the ability to perform perfect Clifford group operations, universal quantum computation becomes possible given one additional operation: the ability to create copies of a certain noisy qubit state  $\rho$ . An example of a suitable  $\rho$  is the state  $|\pi/8\rangle \equiv \cos(\pi/8)|0\rangle + \sin(\pi/8)|1\rangle$ , or more generally a depolarized version of that state,

$$\rho = (1 - \alpha)|\pi/8\rangle\langle\pi/8| + \alpha(|0\rangle\langle 0| + |1\rangle\langle 1|)/2, \quad (6.14)$$

so long as  $\alpha$  is less than  $\approx 0.28$ . If  $\alpha > 0$ , that is if  $\rho$  is noisy, then the perfect Clifford group operations are used to distill many copies of  $\rho$  into noise-free copies of the state  $|\pi/8\rangle$ , which then can be used in the construction of perfect non-Clifford group operations. The condition  $\alpha \lesssim 0.28$  is required for this distillation process to operate successfully. If physical error rates are below the threshold for reliable Clifford group computation, then we can achieve the perfect Clifford group operations required for the distillation process by operating the distillation process on encoded qubits with sufficient levels of concatenation. Furthermore, Knill has shown that encoded versions of the necessary state  $\rho$  in Eq. (6.14) can be created at arbitrary levels of concatenation using a simple technique known as *state injection*. He has argued that given a physical error rate near the threshold for Clifford group operations, the state injection process would create a  $\rho$  having a depolarization parameter  $\alpha$  significantly less than the upper limit of  $\approx 0.28$  required for perfect distillation. Thus, to summarize, being below the threshold for Clifford group operations implies being below the threshold for universal quantum computation, and so we concentrate on the simpler task of analysing the former. Note that the above arguments were originally given in the context of a much different error-correction protocol (and noise model) to that which we consider, however the reasoning can be carried over directly to the case of optical cluster states.

Our simulations extract various statistics regarding failure modes of our fault-tolerant protocol. Thus we do multiple trials of the protocol in order to estimate these statistics. A single trial involves the simulation of multiple rounds of quantum error-correction applied to a single encoded logical qubit. This is all done within the optical cluster-state model of computation, with the noise model as described in Section 6.2.

The major elements of a single trial are as follows: (1) the input state; (2) the input to a round of quantum error-correction; (3) the preparation of the ancilla states used to extract error syndromes; (4) the preparation and use of the *telecorrector* cluster enabling interactions between the encoded data and the ancilla; (5) the reduction of photon loss and non-determinism to Pauli-type errors; and (6) decoding.

We will now describe each of these elements in detail. First, however, we discuss some special tools which are used repeatedly in multiple elements of our cluster-state computation.

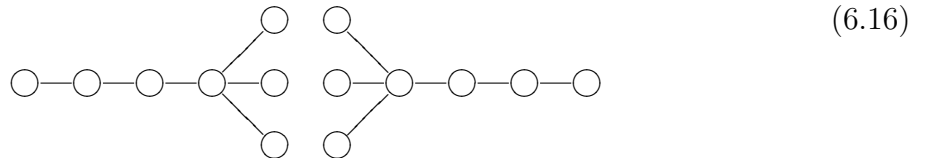
### 1 – Tools for optical cluster-state computing: microclusters, parallel fusion, and postselection

Chapter 5 described how to clusterize quantum circuits, and how to implement cluster-state computation optically. However there are three useful additional tools which we use repeatedly through the entire protocol, and which deserve special mention: *microclusters*, *parallel fusion*, and *postselection*.

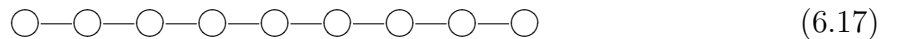
A microcluster is a star-shaped cluster, for example:



The central node in the microcluster is known as the *root node*, while the other nodes are *leaf nodes*. Nielsen [Nie04] showed that microclusters can be used as a tool to build up larger clusters. In particular, the use of microclusters ensures that these larger clusters always have multiple leaf nodes, which can be used to enhance the probability with which we fuse two larger clusters:



We can attempt three simultaneous fusion gates between adjacent leaf nodes of the two clusters. With a probability that goes rapidly to one as the number of leaves increases, at least one of these fusion gates succeeds, fusing the two clusters:



When more than one fusion gate succeeds, we can obtain the same fused cluster, simply by measuring redundant fused nodes in the computational basis<sup>1</sup>. We call this process of using leaves to fuse the two clusters with high probability *parallel fusion*.

We try to create microclusters in a way that meets two complementary aims: (1) we wish to create them rapidly, in order to minimize the effects of noise; and (2) we wish to use the fewest physical resources possible in creating the microclusters. Our microcluster creation protocol is designed with both these goals in mind; somewhat better thresholds could be obtained at the expense of using more resources.

When the number of leaves is a power of two, e.g.,  $k = 2^m$ , we create the microcluster as follows. We begin with  $2^m$  one-leaf microclusters, which are just Bell pairs:



We then fuse pairs of the one-leaf microclusters in order to create two-leaf microclusters:



We continue in this way, repeatedly fusing the root nodes of pairs of microclusters, obtaining microclusters with ever more leaves. For the 4-leaf case, the process terminates at the next stage:



The protocol when the number of leaves is not a power of two is a straightforward modification.

When preparing the microclusters, the fusion gates will inevitably sometimes fail. However, by doing a large number of attempts to create the microcluster in parallel, we can ensure that with very high probability at least one of these attempts will be successful.

---

<sup>1</sup>In fact, the operations we need to do can even be accomplished without removing any redundant nodes, and this is the approach we take in our simulations. In particular, imagine that  $k$  of the simultaneous fusions succeed, resulting in  $k$  qubits in a position where there should be just one. It can be shown that this cluster state is stabilized by (that is, is a  $+1$  eigenstate of) a tensor product of  $X$ s on any even number of those  $k$  qubits. This shows that if we were to later measure one of the  $k$  qubits in the  $X$  basis, as part of the normal running of the cluster, then the state of each of the other  $k - 1$  extra qubits would collapse to an eigenstate of  $X$ , thus automatically disentangling them from the cluster without the need for further measurements. Note that there is a potential advantage in measuring the extra qubits anyway, in the  $X$  basis, to verify the measurement outcome of the first qubit. However, we do not perform this type of verification in the simulations.

For simplicity, in our simulations we assume that fusion gates are *always* successful during microcluster creation (but not in general). This is justified because the experimenter can always postselect during microcluster creation. With this postselection, the expected number of Bell pairs consumed per  $k$ -leaf microcluster is  $k^2$ , and it takes  $\log_2(k) + 1$  time-steps to create the microcluster.

Our use of postselection in microcluster creation is merely one place at which we use postselection. It can be used whenever performing manipulations on clusters that do not contain any of the data being processed. This will include ancilla and telecorrector creation, which actually contain the bulk of the operations performed in our computation. This is extremely convenient, for it enables us to assume that non-deterministic operations have been performed successfully, at the expense of requiring the experimenter to perform a number of attempts at such operations in parallel, and to postselect on the successful operations. It will be important for us to keep track of the scaling involved in such postselection, to ensure that no exponential overheads are incurred.

## 2 – Input states

The trials we simulate consist of multiple rounds of clusterized quantum error-correction. To describe how these rounds occur we must first specify the form of the state which is input to a round. The first round of error-correction is, of course, somewhat special, since it is the initial state of the entire computation. Nonetheless, it has the same general form as the inputs to any other round. Therefore, we begin by describing the general case, before discussing some caveats specific to the initial state of the entire trial.

The state of our optical cluster-state computer at the start of any given round is of the following form:


(6.21)

This is not (quite) a cluster state. To describe the state in the ideal case, consider the following two-stage preparation procedure<sup>2</sup>: (1) prepare the boxed qubits (i.e. the root

<sup>2</sup>This is, of course, not the actual procedure used to obtain the state, but merely a convenient way of

nodes) in the encoded state of the corresponding qubit; and (2) attach bonds to the leaves according to the standard definition. We will make use of the leaves in the manner described earlier, to enhance the probability of fusing this input cluster to the telecorrector state (described later), which is used to effect the error-correction. As pictured, we have three leaves per root node, however in simulations this number may be varied.

Of course, in practice, the actual state will be related to this ideal state by a Pauli frame, and possibly also affected by noise in the Pauli frame, and on the physical qubits. These deviations are described using the techniques we have already introduced. Furthermore, in practice the root nodes will typically have been premeasured, and so won't actually be physically present. However, as was mentioned in Section 5.1, it is often convenient to carry out the analysis as though operations were done in a different order than is actually the case physically, and so we will sometimes describe the computation as though the root nodes (and the associated local Pauli frames) are present at the beginning of the round.

At the beginning of the entire trial, we assume the input is a noise-free state of the form depicted in Equation (6.21). Of course, in practice, the actual state at the beginning of the computation will be noisy. However, this noise-free assumption is justified on the grounds that the initial state does not actually matter, since our goal is to estimate the rate *per round* at which crashes are introduced into the encoded data. Following Steane [Ste03], we perform some number of “warm-up” rounds of error-correction before beginning to gather data on this crash rate, in order to avoid transient effects due to the particular choice of initial state. The reason for starting with a noise free state is because it is a reasonable approximation to the actual (noisy) state of the computer after many rounds, and thus the transient effects can be expected to die out relatively quickly compared with many other possible starting states.

### 3 – *Ancilla creation*

Each round of quantum error-correction involves the creation of some number of verified ancilla states, which are used to extract syndrome bits. These states are analogous to the ancilla states used in Steane's protocol, described in Subsection 2.3.1. The exact number of ancilla states required may vary from round to round; we describe later the details of how they are integrated into the computation.

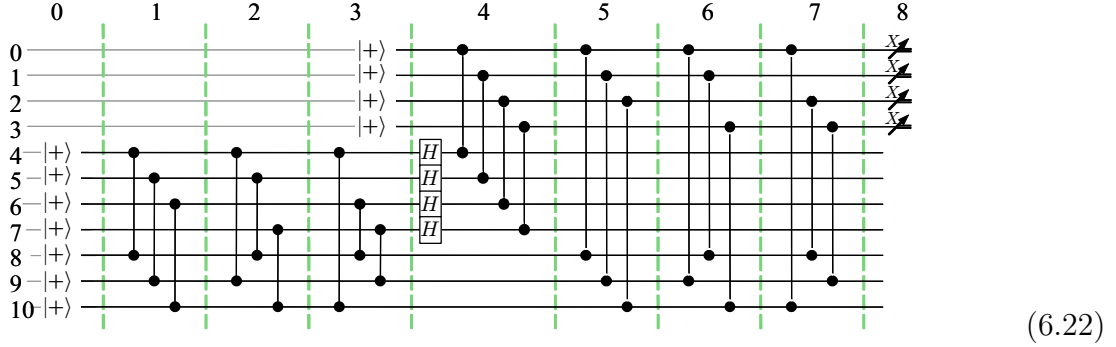
In this section we describe the cluster-state computation used to prepare a single ancilla. This computation is essentially a clusterized version of Steane's ancilla creation circuit. We will describe this for the case of the Steane 7-qubit code, but the procedure

---

describing what the state is.

generalizes in a straightforward manner to many other CSS codes, including the 23-qubit Golay code used in some of our simulations<sup>3</sup>.

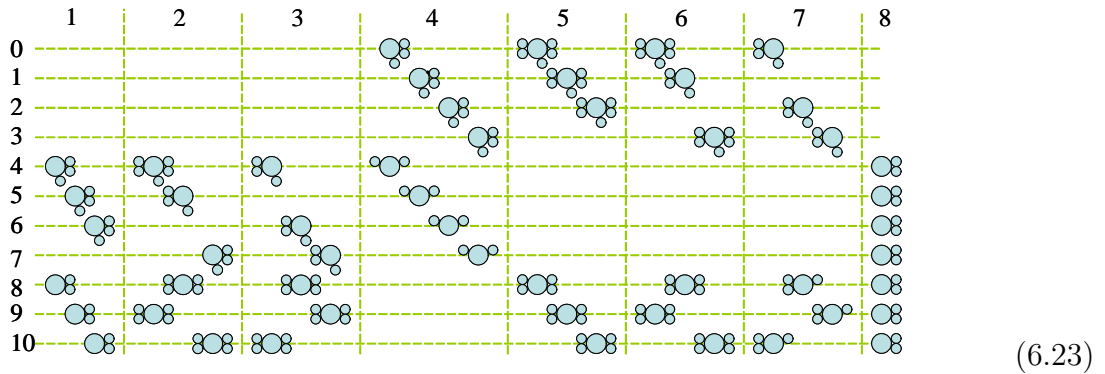
Following the description of Steane's protocol in Subsection 2.3.1, we can create a fault-tolerant ancilla for the 7-qubit code using a quantum circuit of the form:



Note that, for convenience, the above circuit is drawn a little differently to the versions shown in Subsection 2.3.1. In particular, the  $|0\rangle$ -state preparations have been replaced with  $|+\rangle$ -state preparations, and the controlled-not gates have been replaced by controlled-phase gates. To compensate for these changes, a series of Hadamard gates have been added.

We clusterize the circuit following the standard procedures (as described in Subsection 5.1.3) for clusterization, but optimized in order to meet two complementary goals: (1) we do many operations in parallel, in order to reduce the effects of noise; and (2) careful use of postselection, in order to prevent a blow out in resource usage.

We begin the clusterization by creating an array of microclusters:

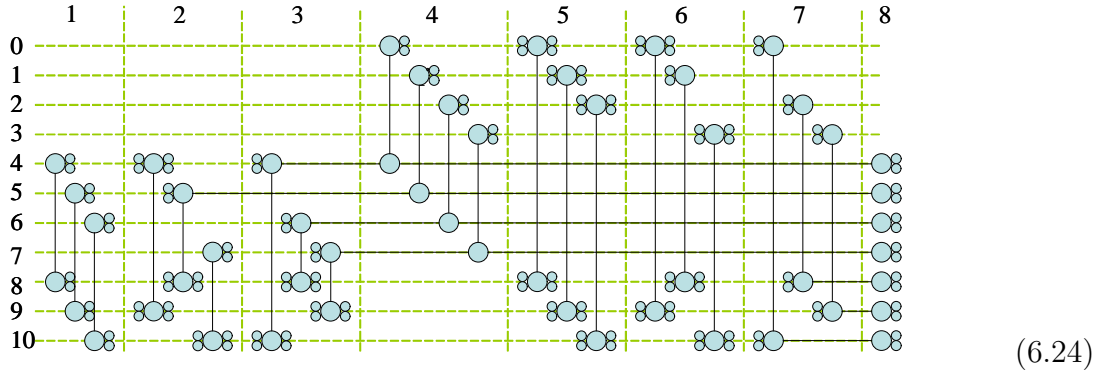


For clarity we have abridged our microcluster notation, omitting the bonds, and just drawing the nodes; the large circles are root nodes, while the small circles are leaf nodes.

<sup>3</sup>The 23-qubit Golay code is derived from the classical binary Golay code, whose defining parity check matrix is given in, for example, Sec. 5.3.3 of [LX04] and online at [Ter].

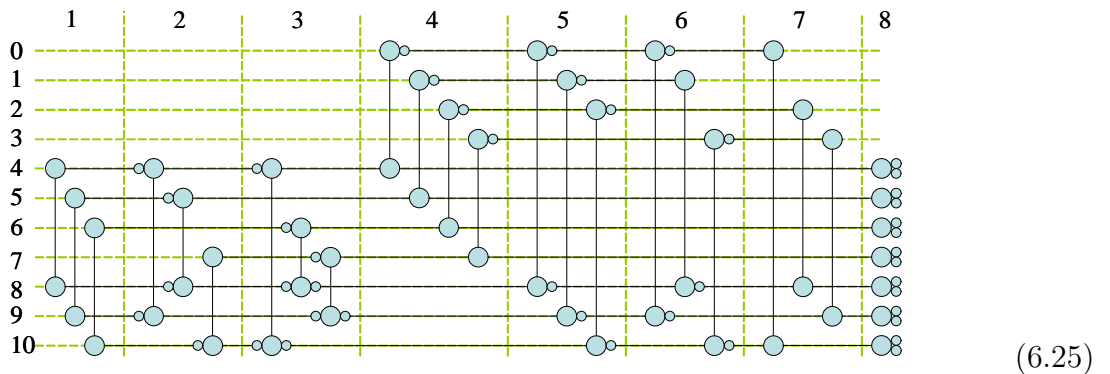
The co-ordinates in our microcluster array (denoted by the dashed lines and numbered labels) correspond directly to the co-ordinates in the Steane circuit. The only exception is the final column of the microcluster array, which corresponds to the output of the cluster-state computation. Nontrivial gates in the Steane circuit are replaced by microclusters, while memory steps do not require additional microclusters, and so we omit these where possible.

Our next goal is to create the following bonded microcluster array:



We do this in two steps. The first step is to attempt creation of all the *vertical* bonds, by fusion of appropriate leaves and roots. By postselection we can assume that all of these fusions were successful and no photon loss was detected. In reality, the experimenter will need to create a larger array of microclusters, and attempt all the fusions simultaneously, discarding wherever unsuccessful. The second step is to create the horizontal bonds, again by fusions of the appropriate leaves and roots, and using postselection to ensure success.

The final step is to obtain the cluster:



using parallel fusion to add the remaining horizontal bonds. The reason we use parallel fusion at this stage is to reduce the cost of postselection. The horizontal bonds added at this stage connect large parts of the cluster, and so a failure of any will result in the need

to start over, and thus it is important to ensure a high probability of success, in order to reduce resource usage.

Note that, as illustrated, parallel fusion involves 2 attempted connections. However, the number of attempted connections is a variable of our simulation, and in practice we have been using 3. Varying this figure will affect both the noise threshold and the resource usage. A value of 1 is the best choice with respect to the noise threshold, since the microclusters used would be smallest in this instance, thus creating less opportunities for noise to be introduced. The corresponding resource overhead would be particularly bad though, due to the very small probability ( $\frac{1}{2^{29}} \approx 2 \times 10^{-9}$  for the 7-qubit code) of fusion gates in the final step of ancilla cluster creation all succeeding. Using 3 attempts per parallel fusion, the probability of success of the final step increases to  $\frac{7^{29}}{8^{29}} \approx 0.02$ . If the number of attempts per parallel fusion is made too large, the benefit to the resource usage due to the higher probability of success is outweighed by the expense of creating large microclusters at the beginning. We have not performed a detailed analysis of the optimal choice for this parameter, rather we have chosen 3 as a reasonable trade-off between noise performance and resource usage.

To conclude the ancilla preparation, we simultaneously measure all remaining qubits in the  $X$  basis, except those qubits in column 8, applying the standard rules for Pauli frame propagation. To verify the ancilla, we postselect on the measurement results of the terminating qubits in rows 0, 1, 2, 3 all being 0. The resulting state is identical to the state illustrated in another context in Equation (6.21), with the encoded state being a  $|+\rangle$ . By contrast, the output of Steane's circuit-based procedure is an encoded  $|0\rangle$ . The difference between our protocol and Steane's is due to the presence of the extra horizontal bond between columns 7 and 8, which effects an encoded Hadamard operation. This will be compensated by a subsequent encoded Hadamard operation, described below.

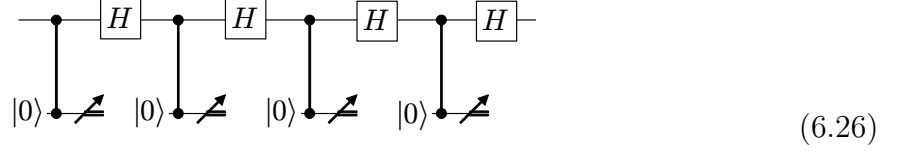
#### 4 – *Telecorrector creation*

To perform error correction we need to interact the data in our cluster-state computer with the ancilla states in order to extract the error syndrome. We do this using a special cluster state which we call a *telecorrector*, which incorporates both multiple ancilla states, as well as the cluster-based machinery to effect the necessary interactions. The telecorrector arises by clusterizing Steane's protocol, but, as we shall describe, the cluster protocol enables several modifications to improve the quality of the syndrome extraction. As in the previous section, our description is adapted to the Steane 7-qubit code, but is easily modified for many other CSS codes.

Our clusterized method of syndrome extraction is based on the following quantum

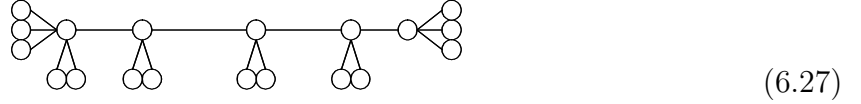


circuit for syndrome extraction



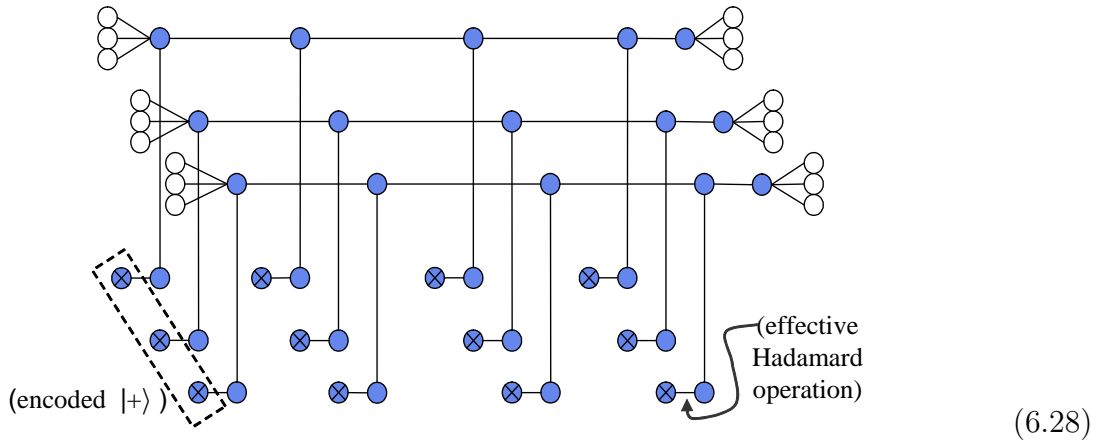
where operations are being performed on encoded qubits,  $|0\rangle$  is fault-tolerant ancilla creation, and the measurement is a transversal  $X$  basis measurement on constituent physical qubits in the code. Circuit (6.26) is analogous to Steane's circuit, except that the number of syndrome extractions is fixed, and the syndrome extractions are performed in a different order:  $X, Z, X, Z$ , in contrast to Steane, who extracts all  $X$  information first, followed by all  $Z$ . The reasons for these differences are explained below.

Telecorrector creation begins with the creation of 7 copies of the following state:



This state can be created in the obvious way using postselected microcluster fusion. The leaves on the left-hand end will eventually be used to attach to a single qubit of the encoded data using parallel fusion. The leaves and root node on the right-hand end will contain the output of this round of error-correction, and become the input to the next round of error-correction. The remaining leaves will be used to fuse to ancilla states.

Simultaneous with the creation of Cluster (6.27), we create four verified ancilla states, using the technique described in Section 6.4.3.3. We then fuse the ancillas with the leaves on Cluster (6.27) to create the state:



(The meaning of the shaded qubits will be explained below). Note that we have illustrated this as though only three qubits are involved in the code: the case of 7 (or more) qubit codes is similar, but the diagram would be larger and more cluttered.

The next step is to measure all the shaded qubits in Cluster (6.28) in the  $X$  basis, leaving only the left-most and right-most leaves, for later use in attaching the data, and future rounds of error-correction.

Applying the propagation rules for the Pauli frame, it can be shown that the pattern of measurement outcomes from the shaded qubits completely determines whether or not the repeated syndrome measurements will agree.

This is a remarkable property, since it enables us to determine whether the repeated syndrome measurements will agree *before* the state has even interacted with the data. Furthermore, we can take advantage of this by postselecting on obtaining a set of measurement outcomes that ensure this *preagreeing syndrome* property. We call the postselected state with this preagreeing syndrome property the *telecorrector*.

Once prepared, we use parallel fusion to attach the telecorrector to the data, and then  $X$  basis measurements to complete this part of the cluster-state computation. Standard propagation rules are used to update the Pauli frame, and to determine the final syndrome extracted from this procedure. We describe in the next section how this syndrome information is decoded in order to perform correction.

The preagreeing syndrome property is responsible for the different number and order of syndrome extractions in our protocol as compared with Steane's. Steane needs to extract many syndromes (more as the code gets larger) in order to make it likely that some large subset of those syndromes agree. In any round where syndromes don't agree, correction cannot take place, and the round just adds more noise to the data. We avoid this issue by using the preagreeing syndrome property, thus reducing the number of locations at which noise can be introduced into the data.

The preagreeing syndrome property also accounts for the order in which we extract syndromes. In Steane's protocol, the order of syndrome extractions is all  $X$  extractions in succession followed by all  $Z$  extractions, so as to maximize the chance of obtaining syndromes that agree. By extracting syndromes in the order  $X, Z, X, Z$  we reduce the chance of agreeing syndromes (for a small cost in resource usage) but gain the ability to detect and postselect against additional types of noise. In particular,  $X$  errors that propagate from the second ancilla to become  $X$  errors on the data will be detectable via a disagreement of the first and third syndromes. Likewise,  $X$  errors that propagate from the third ancilla to become  $Z$  errors on the data will cause the second and fourth syndromes to disagree.

### 5 – *Reduction of fusion gate failure and photon loss to Pauli errors*

During the preparation of the ancilla and telecorrector states we used postselection to avoid dealing with fusion gate failure and photon loss. This has the advantage both of improving the threshold, and also means that our simulations do not need to describe these errors. However, when the telecorrector is joined to the data, it is no longer possible to postselect against these types of error, and we must find some way of modeling them in our simulations.

By following a suitable experimental protocol, it turns out that both these types of errors can be reduced to a (located) Pauli-type error, which we already know how to model in our simulations. The purpose of this section is to describe this reduction.

In practice, we believe the protocol for reduction we describe is likely to slightly worsen the behaviour of the cluster-state computation. The reason for introducing the reduction is therefore not to improve the threshold, but rather to simplify our simulations, and the statistics that we gather. In actual experiments, the special steps in the protocol described in this section would not need to be performed, and the threshold would be slightly higher than our simulations indicate.

We begin with a description of how we treat fusion gate failure. The discussion of photon loss will follow similar lines.

Suppose we are attempting to connect the telecorrector to the data using parallel fusion, and all attempts fail, for a particular horizontal row of qubits. The result of such a failure will be missing horizontal bonds in the cluster. That is, instead of obtaining the desired cluster

$$\begin{array}{c} \otimes \text{---} \circ \text{---} \circ \text{---} \dots \\ \phantom{\otimes \text{---} \circ \text{---}} \vdots \end{array} \quad (6.29)$$

we obtain

$$\begin{array}{c} \otimes \quad \circ \quad \circ \text{---} \dots \\ \phantom{\otimes \quad \circ \quad} \vdots \end{array}, \quad (6.30)$$

where the crossed node indicates a root node of the input data. We can think of this as two located (i.e., known by the experimenter) CPHASE errors. (Note that in reality, the central bare node in Equation (6.30) is not present, due to the destructive measurement occurring with fusion failure. For the sake of the present argument, we shall imagine that the experimenter has brought in a new  $|+\rangle$  state in this instance.) Unfortunately, our error model for simulations doesn't allow us to describe CPHASE errors directly. Although

we could imagine adding such an error to our list of possible error types, the propagation rules turn out to be rather complex, and we wish to avoid this if possible.

Suppose, however, that when parallel fusion fails, the experimenter does the following:

- Depolarizes all three nodes of Equation (6.30) (that is, replaces the state of the three qubits by the maximally mixed state).
- Notes the location of the row in which the failure occurred, for use in decoding.
- Carries out the rules for propagating the Pauli frame, as though the fusion gate had succeeded, and the horizontal bonds created.

Once the experimenter has performed the depolarization, it does not make any physical difference whether the CPHASE errors occurred or not, and so we can imagine they have not occurred. The only remaining errors are Pauli-type errors, and so can be simulated in the standard way.

Note that the effect of the intentional depolarization as described is to randomize the results of later measurements performed on the three qubits. Thus, our reduction of fusion failures to Pauli errors could be equivalently achieved by the experimenter randomizing the measurement results, without actually performing the depolarization. Or simpler still, as a consequence of the propagation rules for the Pauli frame, the randomization of the three measurement results could be replaced by a randomization of the Pauli frame of the left-hand root node. Thus, in our simulation, when a failed parallel fusion between the data and telecorrector occurs, we simply randomize the description of the Pauli frame error of the left-hand root node. This completes the description of our procedure for modeling parallel fusion failure when attaching the telecorrector to the data.

Consider now the case of photon loss. Suppose a photon loss is detected after fusion of the data and the telecorrector. Recall from Section 6.2 that photon loss may occur in a number of physically distinct ways: immediately after Bell pair creation, before a memory step, before a fusion gate, and between the fusion gate and measurement. The effect of the photon loss may depend on which of these possible ways it arose during the computation. In particular, the effect may be described in the various cases as either CPHASE errors as in the case of failed fusion, or simply a successfully created cluster followed by a single photon-loss error. The experimenter would not know which of these cases had occurred.

To cope with this, we modify the protocol so that the effects (in any of these cases) can be simulated by a Pauli-type error. In the modified protocol the experimenter does the following after detecting a photon loss immediately after the data and telecorrector have been fused.

- If a missing photon has been detected, the experimenter randomizes the local Pauli frame of the corresponding data qubit, i.e., the left-hand root node.
- Notes the location at which the photon loss occurred, for use in decoding.
- Carries out the rules for propagating the Pauli frame as though the horizontal bond between data and telecorrector had been successfully created.

This is simulated in the obvious way: when a photon loss is detected after fusion of data and telecorrector, we randomize the Pauli frame error of the corresponding data qubit, and apply the standard propagation rules<sup>4</sup>. The justification for following this procedure is very similar to fusion gate failure, but requires the consideration of more separate cases, corresponding to the different possible points of photon loss. We omit the details.

Note that a significant advantage of the frequent measurements performed in the cluster model is that photon loss is detected before it has a chance to propagate to adversely affect other parts of the computation. This is particularly useful as postselection can be used to ensure that ancillas are free of photon loss noise, which helps improve the threshold.

## 6 – Decoding

We use a non-standard technique for syndrome decoding, designed to take advantage of the knowledge the experimenter has of the locations of errors caused by photon loss and nondeterminism. Our technique is a maximum likelihood procedure for decoding arbitrary combinations of located and unlocated errors.

We take advantage of the fact (see, e.g., Exercise 10.45 on page 467 of [NC00]) that a code able to correct  $t$  unlocated errors is also able to correct  $2t$  located errors. This is particularly advantageous for optical cluster-state computation, since parallel fusion failure and photon loss errors are likely the dominant types of noise.

Both the codes we will use in simulations (Steane 7-qubit and Golay 23-qubit) are CSS codes with the property that decoding of the  $X$  and  $Z$  errors can be performed separately using an identical procedure. Our description will be for the case of  $X$  decoding; the  $Z$  follows similar lines.

The decoding routine has the following inputs: the measured  $X$ -error syndrome, obtained by applying the classical parity check matrix to the vector of total errors of the ancilla measurement outcomes; and a list of locations (qubit indices within the code block)

---

<sup>4</sup>One slight simplification we make in our simulations is to assume that photon loss may occur even following *failed* fusion gates. This can only worsen the thresholds obtained from our simulations, but the effect is negligible.

at which located errors have occurred during the round. The outputs to the decoding routine are: a list of locations where  $X$  flips should be made in order to correct the data; and a flag signaling a *located crash*.

The located crash flag indicates that the correction has likely failed, and the logical state of the data has effectively experienced a random  $X$  operation (i.e. an  $X$  *crash*). This situation arises when two different patterns of  $X$  errors are found to have equal maximum likelihood, but differ from each other by a logical  $X$  operation. The located crash flag is not used directly, but will be used to assist decoding at the next level of concatenation, by identifying encoded blocks which are known to have experienced an error. By feeding information in this way to higher levels of concatenation, we are increasing the overall noise-threshold performance of the protocol.

Before describing our maximum likelihood decoder, we first give a simple model for the relative likelihood of errors. The total  $X$  error pattern on the data is a product of  $X$ s due to unlocated errors, and  $X$ s due to located errors. The measured syndrome is assumed to be the bitwise exclusive or of the syndromes of the two error patterns. The likelihood of a pattern of unlocated  $X$ s is assumed to be a decreasing function of weight, but not a function of how the errors are positioned. The likelihood of a pattern of  $X$  errors due to located errors is uniform across all patterns which have  $I$  wherever located errors have not occurred. This is due to our reduction of located errors to depolarization. For example, if located errors have occurred on three qubits, then the resulting  $X$  error pattern on those qubits due to the located errors will be either  $III$ ,  $IIX$ ,  $\dots$ ,  $XXX$  with equal probability, and  $I$  on other qubits.

To decode, we loop over all possible values for the located error pattern, and for each one we determine the most likely unlocated error pattern. For a particular located error pattern, the most likely unlocated error pattern is found by first finding its syndrome, by taking the exclusive or of the measured syndrome with the syndrome of the located error pattern. Then from this syndrome, the most likely unlocated error pattern is found via a standard decoding array. As the loop is repeated over all located error patterns, we keep track of which “most likely unlocated error pattern” has the overall minimum weight, and is thus most likely overall. If this minimum is unique, then the data is corrected<sup>5</sup> by first correcting for this minimum weight unlocated error pattern, then correcting for the corresponding located error pattern. The located crash flag is set to “false”.

---

<sup>5</sup>Note that in both the cluster-based and deterministic protocols we do not ever physically apply the corrections. Instead, by “correcting” the data we simply mean that we keep track of the corrections that must be applied, and propagate them forward through the computation to be compensated at the end, much as we treat the Pauli frame.

Otherwise, if the minimum is not unique, we arbitrarily choose one of the minimum weight patterns and corresponding located error pattern, and correct accordingly. We compare the correction performed against the corrections associated with each of the other minima. If they are all equivalent up to stabilizer operations of the code, then we set the located crash flag to “false”. Otherwise we set the located crash flag to “true”.

### 7 – Results of the optical cluster simulation

To determine the threshold for a concatenated error correction protocol, we must analyse how the effective error rates vary as more levels of concatenation are added. We now give results of this analysis for the lowest level of concatenation – the cluster-based protocol of Subsection 6.4.3. We simulated this protocol with the aim of categorizing the function that maps the physical noise parameters  $(\epsilon, \gamma)$  to the logical error rates, or *crash* rates, defined below. Likewise, in Subsection 6.4.4 we describe simulations which categorize the similar function for a deterministic circuit-based protocol, representing higher levels of concatenation. In Subsection 6.4.5, the results are combined to give the threshold region for concatenated cluster-state optical quantum computing.

Two of the collaborators in this project, C. M. Dawson and H. L. Haselgrove, each created a version of the simulator, and no program code was shared between the two versions. This duplication was done so that agreement between the results of the two simulators could act as a verification that the simulators were bug free. The programming languages C++ and C were used for the most part (and to a lesser extent, Python and MATLAB).

At the end of a round of simulated cluster-based error correction, we say that the round has caused a *located crash* whenever either the  $X$  or  $Z$  decoding steps in that round has reported a located crash. Note that the imagined experimenter would be aware of located crashes occurring. In addition, we define an *unlocated crash* as follows. We take the pattern of total Pauli errors on the root nodes of the data, and consider the result of a perfect (noise-free) round of correction. If perfect correction would result in a pattern of Pauli errors equivalent to an encoded  $X$ ,  $Y$ , or  $Z$  Pauli operation, then we say the data has experienced an unlocated crash. Note that errors on the leaves of the data are not taken into account when we test for an unlocated crash. Such errors are not completely ignored, as they will instead propagate to the next round of error correction and affect the next crash rate test.

We performed four separate sets of simulations, in order to compare the use of two different codes and two different settings for memory noise. The four configurations were: Steane 7-qubit code with and without memory noise enabled; and the Golay 23-qubit

code with and without memory noise enabled. In the cases where memory noise was disabled, we did not apply photon loss or depolarization noise during memory operations. Comparing the results with memory noise enabled and disabled gives an indication of how significant the effect of memory noise is on the performance of the protocol.

For each of the four configurations noted above, we chose a number of settings for the noise parameters  $(\epsilon, \gamma)$ , and for each we ran a many-trial Monte Carlo simulation. Each trial of the Monte Carlo simulation consisted of two successive rounds of the error correction protocol, and the outcome of the trial was determined by whether the second of the two rounds caused a crash. The purpose of the first “warm-up” round is to reduce the transient effects due to our choice of (noise free) initial conditions. We found that including more than one warm-up round did not make a statistically significant change to the results. However, not including a warm-up round did affect results considerably.

The number of parallel attempts per leaf-to-leaf fusion during ancilla cluster creation and during the joining of ancilla to the telecorrector was set to 3 throughout. Recall, as far as the noise performance is concerned, the fewer attempts per leaf-to-leaf fusion the better during the above mentioned cluster building steps. However we chose 3 as a compromise between noise performance and resource usage.

The number of parallel attempts per leaf-to-leaf fusion when joining a telecorrector to the data cluster was set at five throughout. Here, fewer attempts is not necessarily better for noise performance, because when all attempts fail, a located error is introduced to the data. We found that using any figure above five gave a consistently worse final threshold, whereas a figure less than five gave a worse threshold for small values of  $\gamma$  but a slightly better threshold for larger  $\gamma$  values.

The various outcomes of each trial are tallied as follows. For all the trials for which the first round does not cause a crash, we count: (1) the number  $N_U$  of trials for which the second round causes an unlocated crash but not a located crash, (2) the number  $N_L$  of trials for which the second round causes a located crash, and (3) the number  $N_N$  of trials for which no crashes occur.

From the values  $N_U$ ,  $N_L$ , and  $N_N$ , the unlocated and located crash rates  $E$  and  $\Gamma$  are estimated as follows:

$$E = \frac{N_U}{N_U + N_N}, \quad (6.31)$$

$$\Gamma = \frac{N_L}{N_U + N_N + N_L}. \quad (6.32)$$

Note that we omit  $N_L$  from the denominator of  $E$  since we only compute the unlocated crash rate conditional on no located crash having occurred. The estimated standard error



for  $E$  and  $\Gamma$  respectively are

$$\sigma_E = \frac{\sqrt{N_U}}{N_U + N_N}, \quad (6.33)$$

$$\sigma_\Gamma = \frac{\sqrt{N_L}}{N_U + N_N + N_L}. \quad (6.34)$$

Both these expressions arise from the fact that if we sample  $N$  times to estimate the probability  $p$  of an event occurring, then the standard deviation in the estimate is  $\sqrt{p(1-p)/N}$ . When  $p$  is small, as it is in our case, we may neglect the  $1-p$  term to obtain a standard deviation of  $\sqrt{p/N}$ .

The two versions of the simulator program code were compared as follows. For 65 different settings of  $(\epsilon, \gamma)$ , the values  $(E, \Gamma)$  were estimated from each simulator using a sample size of at least  $10^6$ . This was done for the 7-qubit code, both for memory noise disabled and enabled. For the resulting 130 different values, we compared the results obtained by the two simulators, and the largest difference observed was 3.1 times the estimated standard error. In other words, the two independently-created simulators showed excellent agreement, and this provides additional evidence that they are free of serious bugs.

One of the versions of the simulator code was used to gather final results. We denote the particular choices of the input noise parameters as  $(\epsilon_i, \gamma_i)$ ,  $i = 1, \dots, D$ , the corresponding crash rate estimates as  $E_i$  and  $\Gamma_i$ , and the corresponding standard errors as  $\sigma_i^E$  and  $\sigma_i^\Gamma$ . For the 7-qubit code, approximately  $10^7$  samples were run for each of 59 different settings of the noise parameters  $(\epsilon_i, \gamma_i)$ , for both disabled and enabled memory noise. (Note that the particular choices used for  $(\epsilon_i, \gamma_i)$  are shown as small circles on the threshold plots in the final results subsection, 6.4.5).

For the 23-qubit code, samples were gathered for 43 different noise parameter settings, for both enabled and disabled memory noise. The sample sizes ranged from  $4 \times 10^4$  to  $3 \times 10^7$  for disabled memory noise, and from  $3 \times 10^5$  to  $2 \times 10^7$  for enabled memory noise. The smaller sample sizes correspond to highest noise rate settings, where the simulation becomes much slower (due to noisy ancilla and telecorrectors being discarded more often, an effect which is much more pronounced for the 23-qubit code compared with the 7-qubit code).

We fit polynomials to the data using weighted least-squares fitting. A polynomial  $E(\epsilon, \gamma)$  is fitted to the values  $E_i$  by minimizing the following residual:

$$R_E = \sum_{i=1}^D \frac{(E(\epsilon_i, \gamma_i) - E_i)^2}{(\sigma_i^E)^2}. \quad (6.35)$$

Likewise, the polynomial  $\Gamma(\epsilon, \gamma)$  is fitted to the values  $\Gamma_i$  by minimizing the residual:

$$R_\Gamma = \sum_{i=1}^D \frac{(\Gamma(\epsilon_i, \gamma_i) - \Gamma_i)^2}{(\sigma_i^\Gamma)^2}. \quad (6.36)$$

All terms up to order five were included in the polynomial  $E(\epsilon, \gamma)$ , with the exception of terms of order 0 in  $\epsilon$ . The reason for the excluded terms is that we know  $E(0, \gamma) = 0$ . In the polynomial  $\Gamma(\epsilon, \gamma)$ , all terms up to order six were included for the 23-qubit code results, and terms up to order five were included for the 7-qubit code. In each case the chosen orders of five or six were the minimum that gave a “good” fit to the data for all four configurations of code and memory noise. We considered a good fit to be when the residual divided by the number of data points  $D$  was roughly of order 1 (in practice the value ranged from 0.42 to 1.43 for the eight polynomials fitted). Such a condition indicates that the differences between the observed values and the fitted polynomial could reasonably be accounted for solely by the errors due to the finite sample size.

It would be rather cumbersome to give all the fitted polynomials obtained, in isolation from the procedure in Subsection 6.4.5 to convert this information to a threshold region. Rather, as an example of the results, we give the coefficients of the polynomial  $E(\epsilon, \gamma)$  for the case of the 23-qubit Golay code with memory noise enabled, in Table 6.1.

#### 6.4.4 The deterministic protocol

In this subsection we describe the simulation of our deterministic (circuit based) error-correction protocol. Much of the detail of the protocol is given in Appendix 6.6. The main purpose of the present subsection is to explain how the deterministic protocol fits together with the cluster-based protocol, describe the effective noise model used for simulating the deterministic protocol, and to give the methods and results of these simulations.

##### *1 – Concatenation of protocols*

To perform a threshold analysis, one usually imagines that a fault-tolerant error-correction protocol is concatenated with itself many times. That is, the encoded qubits corrected by the circuit at the lowest level of concatenation are themselves used to build up a circuit for error correction at a higher level of encoding, and so on. Then, by definition, a physical error rate is “below the threshold” if the rate of logical errors (crashes) at the highest level of encoding can be reduced arbitrarily close to zero by using sufficiently many levels of concatenation. Usually, to simplify analysis, the error correction circuit and noise model at every level are taken to be identical, and the rate of noise per gate at one level is taken to be the rate of crashes per error-correction round at the next lowest level. With these

Table 6.1: The polynomial  $E(\epsilon, \gamma)$  as fitted to the unlocated crash rate data, for the cluster-state protocol, using the 23-qubit code with memory noise enabled.

Monomial	Coefficient
$\epsilon$	0.003357
$\epsilon\gamma$	2209
$\epsilon\gamma^2$	$-3.630 \times 10^6$
$\epsilon\gamma^3$	$1.868 \times 10^9$
$\epsilon\gamma^4$	$-8.421 \times 10^{10}$
$\epsilon^2$	2009
$\epsilon^2\gamma$	$-2.133 \times 10^7$
$\epsilon^2\gamma^2$	$2.979 \times 10^{10}$
$\epsilon^2\gamma^3$	$-2.573 \times 10^{12}$
$\epsilon^3$	$-3.578 \times 10^7$
$\epsilon^3\gamma$	$2.348 \times 10^{11}$
$\epsilon^3\gamma^2$	$-2.9574 \times 10^{13}$
$\epsilon^4$	$7.098 \times 10^{11}$
$\epsilon^4\gamma$	$-2.341 \times 10^{14}$
$\epsilon^5$	$-2.472 \times 10^{14}$

set of assumptions, the task of determining if a particular noise rate is below the threshold becomes that of simulating just the lowest level of concatenation, and testing whether the crash rate is below the physical noise rate.

In the quantum computation that we are simulating, only the lowest level of concatenation uses the cluster based protocol described in subsection 6.4.3. For the second and higher levels of concatenation, we can effectively regard it as though a circuit-based deterministic protocol is being used, since the encoded gates available to higher levels of concatenation are deterministic. Steane's fault-tolerant protocol would be a suitable choice for the higher levels of concatenation, but rather we have chosen to use the telecorrection protocol of Appendix 6.6 for the reasons we outline in that appendix.

To motivate the ensuing description of the effective noise model used in the simulations of the deterministic protocol, we discuss the way in which a gate or other operation at one level of concatenation is built from the error-correction protocol at the next lower level of concatenation. The operations used in the telecorrector circuit are: CPHASE and CNOT gates; preparation of  $|0\rangle$  and  $|+\rangle$ ;  $X$ -basis measurements; and memory. First we discuss how these operations in the level  $L \geq 3$  of concatenation are built from level  $L - 1$ .

The memory operation at level  $L$  is simply one round of error correction at level  $L - 1$ . Accordingly, in our noise model for memory operations at level  $L$ , the various noise types are introduced with probabilities given by the crash rates of a round of level  $L - 1$  error correction (the details will be made clear later).

Each of the two types of gates used in the telecorrection circuit at level  $L$  are implemented by first applying a round of error correction to the inputs of the gate, then applying the encoded gate consisting of the level  $(L - 1)$  gate applied transversally to each qubit in the code. The error correction stage contains many more gates than the actual encoded gate, thus we assume that the majority of the noise introduced by a gate is due to the error correction step. Accordingly, in our noise model for gates at level  $L$ , noise is introduced to each of the gate inputs according to the model for memory noise (that is, again given by the crash rates of a level  $L - 1$  correction round).

The preparation of  $|0\rangle$  or  $|+\rangle$  at level  $L$  can be implemented by preparing the level  $L - 1$  state transversally on each qubit in the code, followed by a round of error correction. Again, we assume that most of the noise is due to the error-correction step, and at level  $L$  our model introduces noise after preparation operations according to the model for a step of memory noise.

An  $X$ -basis measurement at level  $L$  is implemented by measuring each level  $L - 1$  qubit transversally in the  $X$  basis, then performing classical error correction on the results. So in contrast to the other operations, measurement does not involve a quantum error correction round at the lower level, but rather a noise free classical correction. Accordingly, our noise model assigns a much lower rate of noise to measurements at level  $L$  relative to the rates of the other operations.

Similar arguments can be made for operations at level 2 built from the cluster protocol of the lowest level. Thus, we will take the rates of noise introduced to gates, memory and preparation at level 2 equal to the crash rates due to a round of clusterized error correction, but make the noise due to measurement significantly less.

We now specifically state the effective noise model used at a level  $L \geq 2$  of concatenation, following the arguments above.

## **2 – Effective noise model**

In the simulation of the level  $L \geq 2$  error correction circuit, we model the encoded qubits that this circuit acts upon as though they were physical qubits. That is, at every stage of the simulation of the circuit, the error description is a Pauli error,  $I$ ,  $X$ ,  $Y$  or  $Z$ , associated with each of the qubits. The details of the errors on lower level qubits are not directly simulated. As in the cluster-based protocol, the circuit is divided into time-steps.

Each qubit in the level- $L$  circuit can undergo one operation per time-step. The length of a time-step corresponds to the time taken for a complete round of error correction and an encoded operation to be performed at level  $L - 1$ .

Our model involves four types of noise, unlocated  $X$  and  $Z$  Pauli errors, and located  $X$  and  $Z$  Pauli errors. Unlocated and located errors are designed to represent the unlocated and located crashes occurring at level  $L - 1$ . When a qubit experiences an unlocated  $X$  Pauli error, it undergoes an  $X$  operation, unknown to the experimenter, and similarly for unlocated  $Z$  Pauli errors. When a qubit experiences a located  $X$  error, it undergoes an  $X$  operation with probability  $1/2$ . The experimenter will know that a located  $X$  error has occurred, but not whether the corresponding  $X$  Pauli error has actually been applied.  $Z$  located errors are similar.

When we say that unlocated noise is applied with a probability  $p$ , we mean that both unlocated  $X$  and  $Z$  Pauli errors are applied with equal probability, and independently, such that the total probability that an error was applied is  $p$ . Similarly for located noise applied with probability  $q$ . We choose this model of independent  $X$  and  $Z$  errors because of a numerical observation that the rate of  $Y$  crashes is much less than the combined rate of  $X$  and  $Z$  crashes, for both our cluster-based and deterministic protocols. Although observed  $X$  and  $Z$  crash rates are not entirely independent, we have nonetheless chosen an independent noise model, which empirically appears to provide a good approximation to the observed behaviour.

We now describe how noise is introduced by each operation. Let  $p$  and  $q$  be the rates of unlocated and located crashes respectively for an error correction round at level  $L - 1$ .

- Memory and gates: Before the gate or memory, the following noise is applied to the input qubit, or in the case of two-qubit gates is applied independently to each input. Unlocated noise is applied with probability  $p$ , and, independently, located noise is applied with probability  $q$ .
- Preparation: After the preparation, unlocated noise is applied with probability  $p$ , and, independently, located noise is applied with probability  $q$ .
- Measurement: Before the measurement, unlocated noise is applied with probability  $p/10$ , and, independently, located noise is applied with probability  $q/10$ .

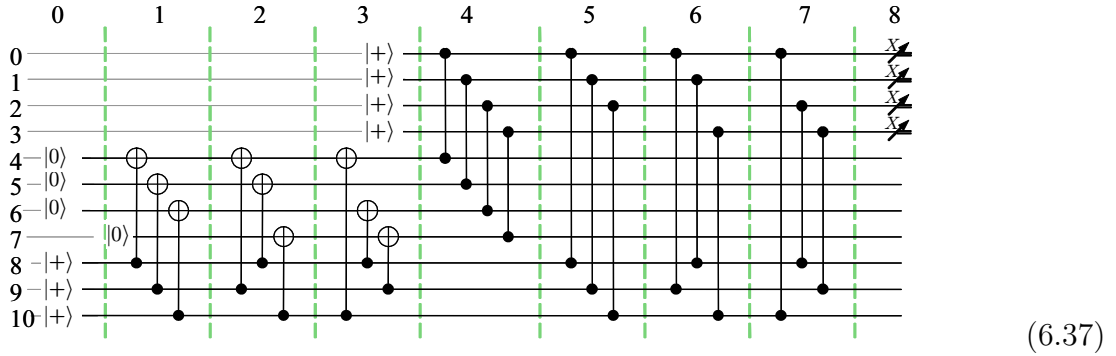
The value of one tenth for the relative strength of measurement noise is somewhat arbitrary. In reality, the relative strength of measurement noise would decrease for higher levels of concatenation, and would be generally less than one tenth. We believe that

our choice to fix the value at one tenth makes little difference to the simulation results, compared with a more accurate treatment.

### 3 – Telecorrection protocol

We simulate the protocol described in Appendix 6.6, in particular using the layout of Circuit (6.61). We now briefly describe some further pertinent details of the protocol not given in the appendix, namely the circuit used for ancilla creation/verification, the procedure for post-selection during telecorrection creation, and the decoding procedure.

The circuit used to create and verify encoded  $|0\rangle$  ancilla states, denoted by the operation “ $|0\rangle$ ” in Circuit (6.61), uses the design of Steane. For example, for the seven qubit code, the circuit is:



where the measurements are post-selected to have outcome “0”.

Note that in the case of the 23-qubit code (not shown), our circuit for ancilla creation and verification has the advantage of taking 8 fewer time steps than that used by Steane [Ste03] for the same code. This is due to the fact that we start with a version of the classical Golay code having a reordering of the 23 bits in the code. By reordering bits in the code (i.e., permuting columns in the check matrix) and then reexpressing the check matrix in standard form, it is possible to change the maximum column and row weight of the check matrix, which has the effect of changing the number of time steps in the creation and verification circuits. After trying many random bit-reorderings, we found that the number of time steps in the circuit could be made as low as 17, compared with Steane’s 25.

The telecorrector-creation part of the protocol, indicated by the boxed region in Circuit (6.61), is performed many times in parallel, and post-selected to give a successfully created telecorrector state. Here, “successful” means that syndromes of like type agree, and that no located noise occurred during the creation circuit.

During the protocol, the data and one half of the telecorrector are measured, in order to effectively apply two successive encoded transport circuits. Each of the two encoded

measurements consists of  $X$ -basis measurements on each of the qubits in the code, followed by classical error correction performed on the measurement results. In each case, the correction procedure involves: (1) calculating the syndrome associated with the measurement results, (2) determining which of the individual measurement results within the encoded measurement were subject to located noise, and (3) using the results of the first two steps as input to the decoding procedure of Subsection 6.4.3.6.

#### **4 – Method for simulating the protocol**

A simulation trial begins with the state of the quantum computer being noise-free. Thus, the description of the initial state is a Pauli error of  $I$  on each data qubit<sup>6</sup>. Then, some number of repeated telecorrection rounds are simulated. As each operation in the circuit is simulated, the Pauli error description of the qubits are updated stochastically based on the unlocated noise model, and Pauli errors are propagated as appropriate for the operation. The propagation rules for each operation are:

- Preparation: Pauli error is reset to  $I$ .
- Measurement: Measurement in the  $X$  basis causes the  $Z$  part of the Pauli error on a qubit to propagate to the measurement result, and the  $X$  part of the error to be eliminated.
- CNOT gate: A Pauli error of  $X^{x_t}Z^{z_t}$  on the target and  $X^{x_c}Z^{z_c}$  on the control are transformed as

$$x'_t = x_t + x_c \tag{6.38}$$

$$z'_t = z_t \tag{6.39}$$

$$x'_c = x_c \tag{6.40}$$

$$z'_x = z_c + z_t. \tag{6.41}$$

- CPHASE gate: A Pauli error of  $X^{x_1}Z^{z_1}$  and  $X^{x_2}Z^{z_2}$  on the two inputs are transformed as

$$x'_1 = x_1 \tag{6.42}$$

$$z'_1 = z_1 + x_2 \tag{6.43}$$

$$x'_2 = x_2 \tag{6.44}$$

$$z'_2 = z_2 + x_1. \tag{6.45}$$

---

<sup>6</sup>Note that the Pauli frame, used in the optical cluster protocol, does not form part of the deterministic protocol. Thus we do not keep track of Pauli frame errors when simulating the deterministic protocol.

To speed up simulations, located noise is not introduced where it will later be post-selected away. Located noise which cannot be post-selected away occurs due to the following operations in the protocol: the transversal CPHASE between the data and the one half of the telecorrector; the memory step on the other half of the telecorrector during the aforementioned transversal CPHASE; and the measurements of the data and one half of the telecorrector. A straightforward analysis of error locations shows that the effect of all these located noise events is statistically equivalent to applying a located error at the start of the round with a suitable probability. We omit the details of this analysis, but note that for simplicity in simulation we used this simplified error model.

### ***5 – Results of simulating the deterministic protocol***

As in the simulations of the cluster-based protocol, we aim to categorize the function which maps input noise parameters, in this case the unlocated noise rate  $p$  and located noise rate  $q$ , to the logical error rates, being the unlocated crash rate  $P$  and located crash rate  $Q$ . From knowledge of this map for both the cluster-based and deterministic protocol, the overall threshold region can be determined.

Again we performed separate sets of simulations using the 7-qubit Steane code with and without memory noise enabled, and using the 23-qubit Golay code with and without memory noise enabled. Note that in the case where memory noise is disabled, we still apply memory noise on the bottom half of the telecorrector during the timestep in which the data and top half of the telecorrector are interacting with the CPHASE gate. This location in the circuit is where any encoded gate would be performed between correction rounds, and so we apply noise here in every circumstance so that the noise due to this encoded operation is taken into account.

For a particular choice of code and memory noise setting, we chose a number of settings for the parameters  $(p, q)$ , and for each we ran a many-trial Monte Carlo simulation. As for the cluster-state simulations, each trial of the Monte Carlo simulation consisted of two successive rounds of the error correction protocol, with statistics gathered on the rate of crashes introduced by the second round. Again, including more than two rounds did not appear to affect results.

The definition of unlocated and located crashes for a round of the deterministic protocol is virtually identical to that given in subsection 6.4.3.7. Similarly, the tallies  $N_L$ ,  $N_U$  and  $N_N$  for the various trial outcomes share the same definition as in subsection 6.4.3.7.



The unlocated and located crash rates  $P$  and  $Q$  are estimated as follows:

$$P = \frac{N_U}{N_U + N_N}, \quad (6.46)$$

$$Q = \frac{N_L}{N_U + N_N + N_L}. \quad (6.47)$$

The estimated standard error for each quantity is

$$\sigma^P = \frac{\sqrt{N_U}}{N_U + N_N}, \quad (6.48)$$

$$\sigma^Q = \frac{\sqrt{N_L}}{N_U + N_N + N_L}. \quad (6.49)$$

The results of two independently-written simulators were compared, as in the case of the optical cluster state protocol, as a check on whether the results were bug free. Estimates of the quantities  $P$  and  $Q$  were compared between the two versions of the simulator, using the 7-qubit code and a sample size of approximately  $10^6$ , for 68 different noise settings with memory noise disabled and 86 different noise settings for memory noise enabled. Comparisons of a lesser sample size were also carried out for the 23-qubit code. The largest discrepancy found during all comparisons equated to 3.2 times the estimated standard deviation. Thus the two simulators showed excellent agreement, and this provides additional evidence that they are free of serious bugs.

Final results were gathered using one of the versions of the simulator. Denote the choices of the input noise parameters as  $(p_i, q_i)$ ,  $i = 1, \dots, D$ , the corresponding crash rate estimates as  $P_i$  and  $Q_i$ , and the corresponding standard errors as  $\sigma_i^P$  and  $\sigma_i^Q$ . Polynomials were fitted to the data using weighted least-squares fitting. A polynomial  $P(p, q)$  is fitted to the values  $P_i$  by minimizing the following residual:

$$R_P = \sum_{i=1}^D \frac{(P(p_i, q_i) - P_i)^2}{(\sigma_i^P)^2}. \quad (6.50)$$

Likewise, the polynomial  $Q(p, q)$  is fitted to the values  $Q_i$  by minimizing the residual:

$$R_Q = \sum_{i=1}^D \frac{(Q(p_i, q_i) - Q_i)^2}{(\sigma_i^Q)^2}. \quad (6.51)$$

All terms up to order six were included in the polynomial  $P(p, q)$ , with the exception of terms of order 0 in  $p$ . In  $Q(p, q)$ , all terms up to order five and eight respectively were included when using the 7 and 23-qubit codes, except terms of order 0 in  $q$ . The reason for the excluded terms is that we know  $P(0, q) = 0$  and  $Q(p, 0) = 0$ . The orders were chosen using a similar criteria as for optical cluster protocol.

To present the results of the deterministic simulations, we calculate a threshold region with respect to the noise parameters at the second level of concatenation. (Thus, we are temporarily ignoring the effect of the optical cluster protocol at the lowest level). Define the map  $g : (p, q) \rightarrow (P(p, q), Q(p, q))$ , where  $P$  and  $Q$  are the fitted polynomials. If  $(p, q)$  are the effective unlocated and located noise rates at the second level of concatenation, then the unlocated and located crash rates at the  $k$ -th level may be estimated by computing  $g^{(k-1)}(p, q)$ . Provided this tends towards  $(0, 0)$  as  $k \rightarrow \infty$  the point  $(p, q)$  is inside the threshold region for the deterministic protocol. It is possible to test many thousands of points very quickly using this method, giving the threshold to high resolution.

The threshold regions for the simulations using the 7 qubit code are shown in Figure 6.1. For each of the points  $(p_i, q_i)$  shown by the circles, between  $10^7$  and  $2 \times 10^7$  trials were run. Threshold regions for the simulations using the 23 qubit code are shown in Figure 6.2. The number of trials run per point  $(p_i, q_i)$  ranged from approximately  $2 \times 10^5$  to  $4 \times 10^7$ . For the upper plot in Figure 6.2 we have estimated the error in the threshold due to the finite sample size of the simulations. This rough estimate of the error was obtained by repeating the polynomial fitting a further 20 times, using the same set of data  $(P_i, Q_i)$ , but subject to additional additive Gaussian noise of standard deviation  $(\sigma_i^P, \sigma_i^Q)$ . The largest and smallest values of the threshold obtained through this process are plotted as the dashed lines. The estimated error for the other three plots in Figures 6.1 and 6.2 is not shown, but is smaller in these cases.

The threshold with respect to unlocated noise can be compared to circuit-model thresholds obtained by other authors (keeping in mind though that noise models and resource usage vary substantially between different authors). Our best threshold for unlocated noise for the four plots in Figures 6.1 and 6.2 is approximately  $8 \times 10^{-3}$ , for the 23-qubit code with no memory noise. This compares with a threshold of  $3 \times 10^{-3}$  obtained by Steane [Ste03],  $9 \times 10^{-3}$  by Reichardt [Rei04], and  $3 \times 10^{-2}$  by Knill [Kni05].

A feature of our threshold plots worth noting is the dramatically larger threshold for located noise (up to 0.25 for the Golay code) as compared to that of unlocated noise. Thus, the use of post-selection in the protocol combined with a purpose-built decoding routine has had a dramatic positive effect on the threshold for unlocated noise.

Note also that all threshold regions in Figures 6.1 and 6.2 show an unexpected feature: the threshold for unlocated noise actually *improves* when a small amount of located noise is added. Presumably, the presence of located noise converts some crashes from unlocated to located, which are then more efficiently dealt with by higher levels of concatenation. So, although it would seem a somewhat absurd notion that adding noise should ever improve

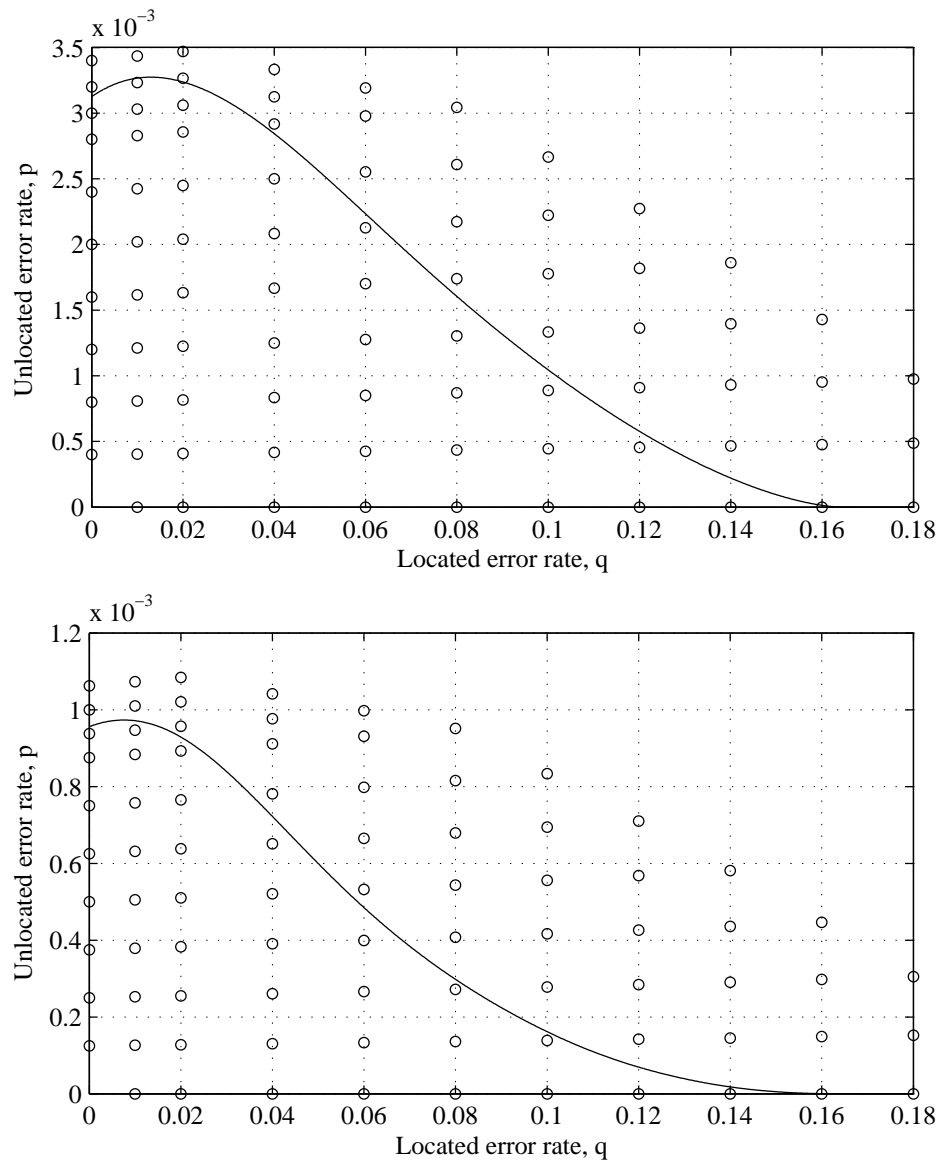


Figure 6.1: Threshold region (below the solid line) for the deterministic protocol using the 7-qubit Steane code. Memory noise is disabled top, and enabled bottom. Circles indicate the noise parameter values for which the simulation was run.

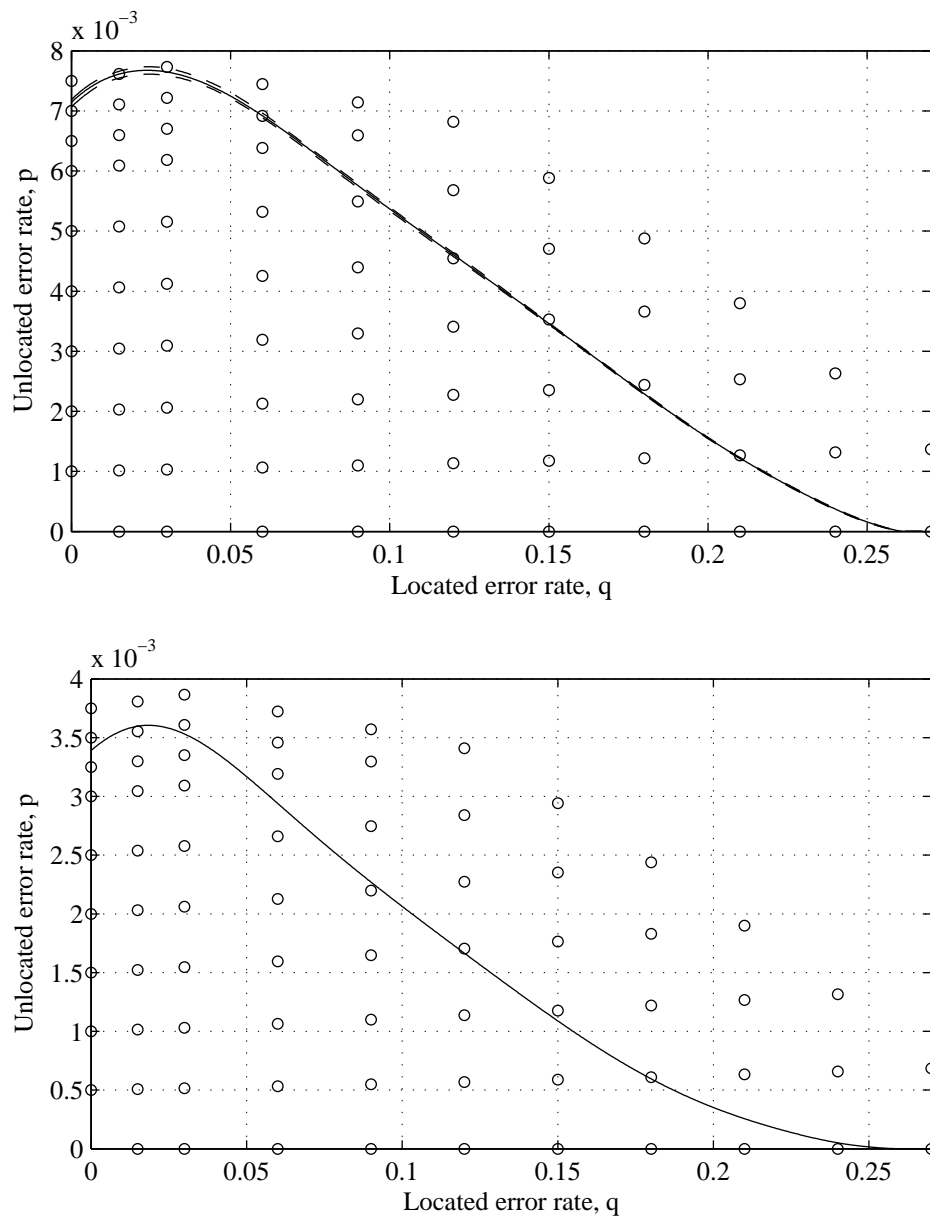


Figure 6.2: Threshold region (below the solid line) for the deterministic protocol using the 23-qubit Golay code. Memory noise is disabled top, and enabled bottom. Dashed lines in upper figure show error due to finite sample size.

the reliability of an error-correction protocol, such behaviour in this case highlights how advantageous it can be to pass information (i.e., crash locations) from one level to another in a concatenated protocol. Such behaviour appears somewhat similar to the well-known phenomenon of stochastic resonance, whereby adding noise to a system may in some circumstances actually improve the signal-to-noise ratio in observations made on that system.

### 6.4.5 Final Results

In this subsection we give the final threshold results for optical cluster-state quantum computing, with respect to the physical error rates of our noise model.

Under  $k$  layers of concatenation, our error-correction protocol consists of one level of the optical cluster protocol concatenated with  $k - 1$  levels of the deterministic protocol. Define the maps  $f : (\epsilon, \gamma) \rightarrow (E(\epsilon, \gamma), \Gamma(\epsilon, \gamma))$  and  $g : (p, q) \rightarrow (P(p, q), Q(p, q))$ , where  $E$  and  $\Gamma$  are the polynomials obtained for the optical cluster protocol in Subsection 6.4.3.7 and  $P$  and  $Q$  are the polynomials obtained for the deterministic protocol in Subsection 6.4.4.5. If  $\epsilon$  is the depolarization parameter and  $\gamma$  is the photon loss rate (defined in Section 6.2) then the unlocated and located crash rates at level  $k$  may be estimated by computing  $(g^{(k-1)} \circ f)(\epsilon, \gamma)$ . If this tends to  $(0, 0)$  as  $k \rightarrow \infty$  then the physical noise rates  $(\epsilon, \gamma)$  are below the threshold.

Note that in deriving the results in this section, we are imagining that the same code (either 7-qubit or 23-qubit) is used at every level of concatenation. This need not be the case, and in general it is possible to imagine a situation where the code choice is made independently at each level.

The threshold regions using the 7-qubit and 23-qubit codes are shown in Figures 6.3 and 6.4 respectively. In the upper plot in Figure 6.4 we have estimated the error in the threshold due to the finite sample size of the simulations, using a method similar to that of Subsection 6.4.4.5. The estimated error is not shown in the other three plots, but is smaller in these cases.

The best of the four thresholds is given by the 23-qubit code with no memory noise. In this case, the protocol can simultaneously protect against a depolarization strength of  $4 \times 10^{-4}$  and photon loss rate of  $10^{-2}$ , approximately. As expected, these values are poorer than for the concatenated circuit-based protocol, due to the overhead associated with clusterization of the optical protocol. We consider these values encouraging, especially given the nondeterministic nature of the optical two-qubit interactions.

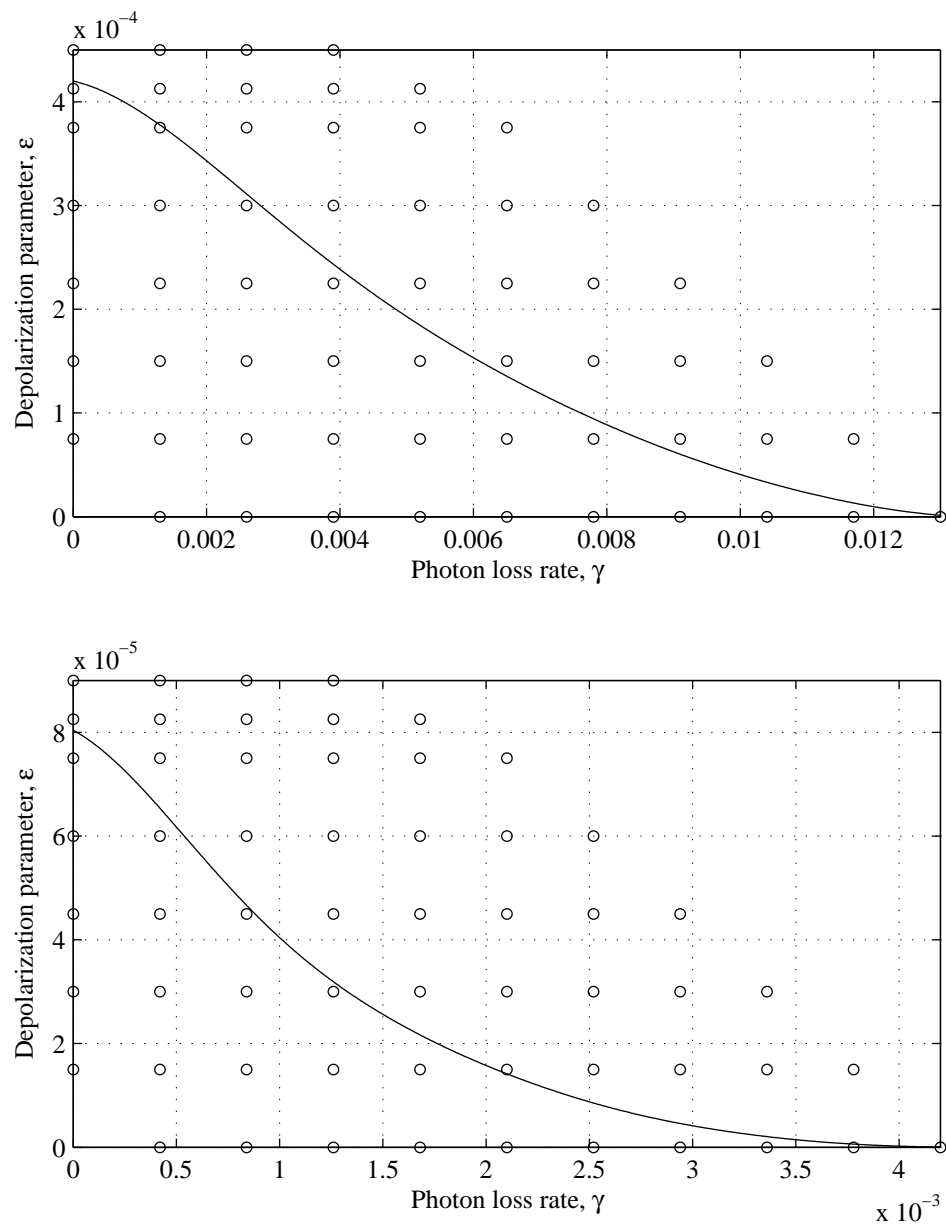


Figure 6.3: Threshold region (below the solid line) for the optical cluster protocol using the 7-qubit Steane code. Memory noise is disabled top, and enabled bottom. Circles are located at the noise parameter values for which the cluster simulation was run.

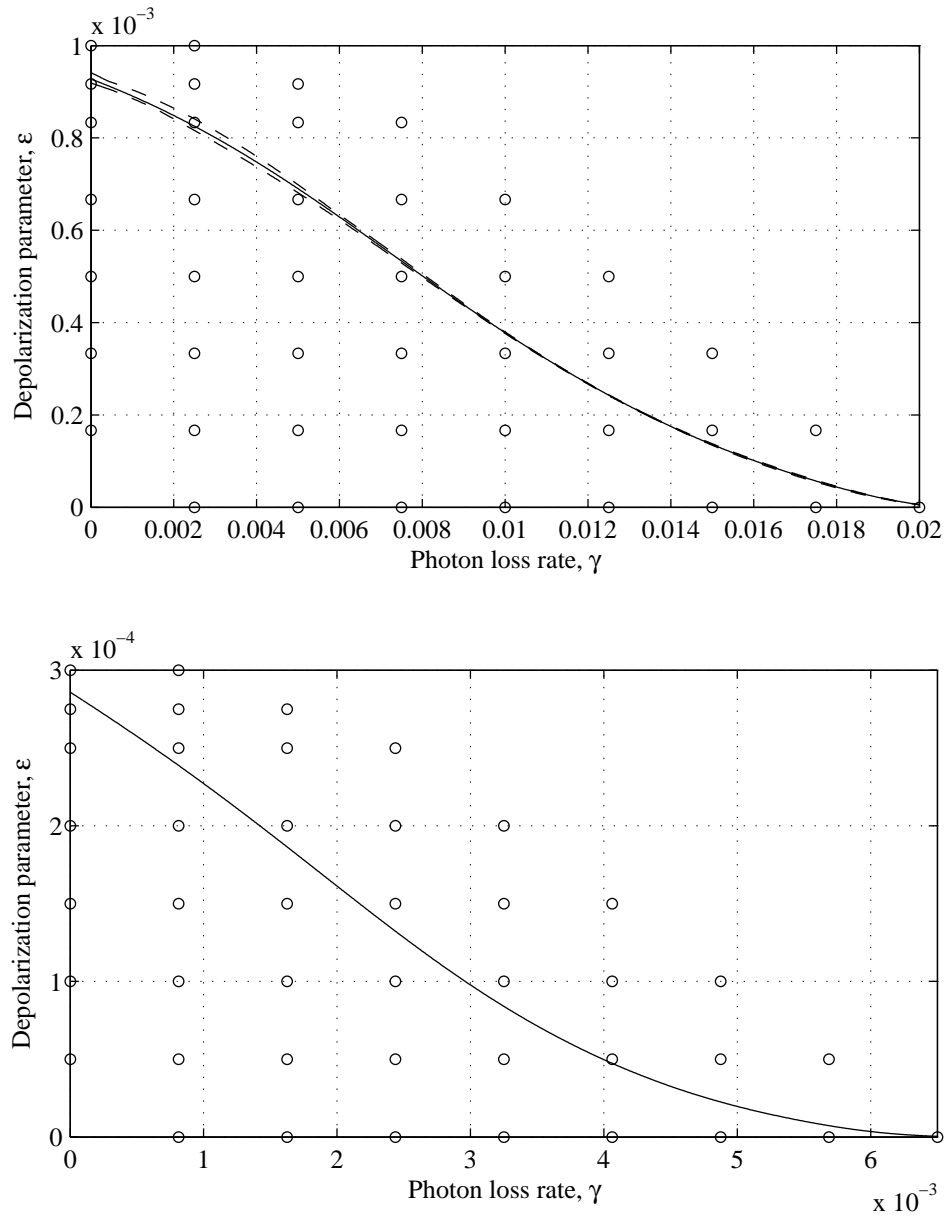


Figure 6.4: Threshold region (below the solid line) for the optical cluster protocol using the 23-qubit Golay code. Memory noise is disabled top, and enabled bottom. Dashed lines in the upper figure show error due to finite sample size.

### 6.4.6 Resource usage

In this subsection we perform a simple analysis of resource usage. This analysis is performed for the Steane 7-qubit code, and for a particular physical noise rate. Ideally, a fuller analysis would consider the (rather complex) question of how resource usage varies with physical noise rates, code choice, and other variable aspects of the protocol such as number of parallel fusions. However, the present analysis is merely aimed at giving a very rough idea of resource requirements.

For a measure of resource usage, we count the average number of Bell pairs consumed per encoded operation. This measure can also be considered as a rough indication of the usage of the other basic operations (fusion gate, measurement, memory), since in the protocol these operations are always very closely associated with Bell pair creation and vice versa.

By “per encoded operation” in the description of the resource usage measure above, we are referring to an operation at the highest level of concatenation (that is, an actual logical gate of the computation being carried out). Henceforth, we refer to such operations as “computational operations”. The resource usage figure will thus depend on the number of levels of concatenation. In turn, the number of levels of concatenation required will depend on the desired level of reliability of the final output of the computation, and the total number of computational operations performed. For the sake of the present analysis, let us define a “reliable” computation to be as follows: with probability at least  $\frac{1}{2}$  all computational operations are crash-free (with respect to the highest level of encoding). Assuming noise rates are below the threshold, adding more levels of concatenation will give a lower probability of crash per computational operation, and thus increase the maximum number of computational operations allowed such that the output will be reliable. If the total crash probability per computational operation is  $p_c$ , then the output will be reliable if the number of computational operations is less than

$$\frac{\log(\frac{1}{2})}{\log(1 - p_c)}. \quad (6.52)$$

In Table 6.2, the results of the analysis are shown, for the 7-qubit code with memory noise enabled. The chosen physical noise parameters are  $(\epsilon, \gamma) = (4 \times 10^{-5}, 4 \times 10^{-4})$ , corresponding to a point roughly in the centre of the threshold region. Each row of the table corresponds to a different number of levels of concatenation. The effective rates of unknown and known crashes at each level are shown in the columns  $p$  and  $q$ . These values were obtained by iterating the polynomials generated from the numerical simulations<sup>7</sup>.

<sup>7</sup>For the purposes of this analysis, we disallowed further low-order terms in the polynomial that by



Table 6.2: Estimated resource usage (number of Bell pairs consumed per computational operation) as a function of concatenation level, for noise parameters  $(\epsilon, \gamma) = (4 \times 10^{-5}, 4 \times 10^{-4})$ , and using the 7-qubit code.

Level	$p$	$q$	Maximum reliable computation length	Bell pairs used per computational op.
1	0.00046	0.0097	68	$1.5 \times 10^{11}$
2	0.00022	0.0027	$2.4 \times 10^2$	$9.3 \times 10^{13}$
3	$4.4 \times 10^{-5}$	0.00036	$1.7 \times 10^3$	$5 \times 10^{16}$
4	$1.5 \times 10^{-6}$	$9.9 \times 10^{-6}$	$6.1 \times 10^4$	$2.6 \times 10^{19}$
5	$1.6 \times 10^{-9}$	$9.4 \times 10^{-9}$	$6.3 \times 10^7$	$1.4 \times 10^{22}$
6	$1.9 \times 10^{-15}$	$9.8 \times 10^{-15}$	$5.9 \times 10^{13}$	$7.1 \times 10^{24}$

The maximum computation length for each level was calculated from Equation (6.52) with  $p_c = p + q$ . The value for Bell pairs consumed per computational operation, at a particular level  $L$ , is given by the number of Bell pairs consumed per error-correction step at level 1, multiplied by appropriate scale-up factors for each of the levels  $2, \dots, L$ . The scale-up factor at some level  $l$  is the expected number of level  $l - 1$  error-correction steps used to implement a level  $l$  error-correction. These factors were estimated by the simulator in a straightforward way (the details of the estimation procedure are not given).

Thus, we see that to get a reliable computation consisting of a significant amount of operations (say  $10^9$ ), the protocol as it stands has the very demanding requirement of approximately  $10^{23}$  Bell pairs per operation. That this figure is so large can be partly explained by our liberal use of post-selection in the various parts of the protocol. Since our main aim in this work is to find the threshold for optical quantum computing, our protocol was designed with optimization of the threshold the primary goal, and thus optimization of resource usage was a lesser priority. A number of simple modifications to the protocol would reduce the resource usage by a few orders of magnitude at least, while only having a small detrimental effect on the threshold. Such modifications would include increasing the number of attempts per parallel fusion so that clusters are discarded less often, and spreading cluster-building procedures over more time steps so that smaller clusters are discarded if a step fails. Nonetheless, resource usage is certainly a significant problem

---

the principles of fault tolerance should be zero. This was done with the aim of increasing the accuracy for very small parameter values.

both for our protocol and others (especially those that heavily rely on post-selection such as [Rei04] and [Kni05]).

## 6.5 Conclusion

We have performed a detailed numerical investigation of the fault-tolerant threshold for optical cluster-state quantum computing. Our work considers a noise model which allows for both photon loss and depolarizing noise. Depolarizing noise is used as a general proxy for all types of local noise other than photon loss, and standard results in the theory of error-correction ensure that the ability to protect against depolarization ensure the ability to protect against other types of noise, including dephasing, amplitude damping, etc.

Our main result has been a *threshold region* of allowed pairs of values for the photon loss and depolarizing noise. Roughly speaking, our results show that scalable, reliable optical quantum computing is possible in the combined presence of both noise types, provided that the photon loss probability is  $< 3 \times 10^{-3}$  and the depolarization probability is  $< 10^{-4}$ . To achieve such threshold values requires very substantial overheads in order to accurately perform long computations. Future work will need to not only improve the threshold, but also reduce the overhead required to do fault-tolerant computation, improve the accuracy of the noise model used in simulations, and address the pseudothreshold phenomenon identified in [STD05, SCCA05].

Our noise model is in contrast to previous investigations of the threshold for optical quantum computing, which have focused on the case in which photon loss is the *sole* source of noise. While photon loss will certainly be an important source of noise in real implementations, other sources of noise such as dephasing will also be present (at lower levels), and techniques which protect solely against photon loss will have the effect of greatly amplifying those other sources of noise. Thus, while the earlier loss-only thresholds are of considerable theoretical interest, they do not provide physically meaningful thresholds.

We note that our threshold results might be applicable to implementations of quantum computing other than linear optics – in particular to any scheme that contains nondeterministic two-qubit interactions, loss noise and depolarization noise. For example, in the scheme by Barret and Kok [BK05] for quantum computing with matter qubits, two-qubit interactions are nondeterministic, with a heralded failure rate of 50%. In analogy to photon loss, the scheme can also exhibit “loss” when an atom jumps out of the qubit space into a higher energy level. It is likely that our threshold results would agree at least

qualitatively with the thresholds for such a system.

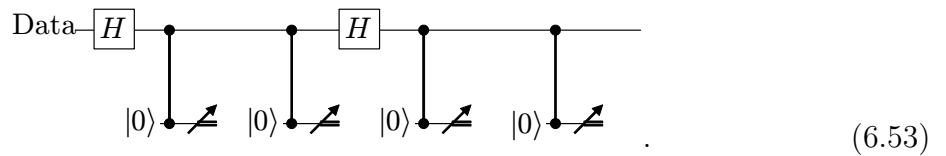
## 6.6 Appendix: Telecorrection

This appendix presents the idea of the telecorrector in a simple form, without the baggage of clusters and nondeterminism. Although use of the telecorrector arises naturally from the cluster state model, there are good reasons to consider it in the circuit model as well. First, it provides a different way of thinking about quantum error correction: most of the difficulty of a fault tolerant round of quantum error correction can be reduced to the creation of a single  $2N$ -qubit resource state ( $N$  being the size of the code). This is in contrast to the normal requirements of an error correction round – the creation of a variable number (at least four) copies of an  $N$ -qubit ancilla state. While we shall consider a particular way of generating the telecorrector, based on a teleported Steane syndrome extraction circuit, it is an interesting open problem to consider better methods for creation.

The second reason for considering telecorrection in the circuit model is for practical use in our simulations. Our deterministic error correction protocol, used for the second and higher levels of concatenation, uses circuit-model telecorrection instead of the standard Steane approach. The benefit is an improved threshold, due to the ability to post-select for agreeing syndromes and against located noise types during telecorrector creation.

Note that the idea of combining error correction and teleportation has been used previously by Knill [Kni05], however the details of our telecorrection procedure and Knill's procedure differ significantly in the details.

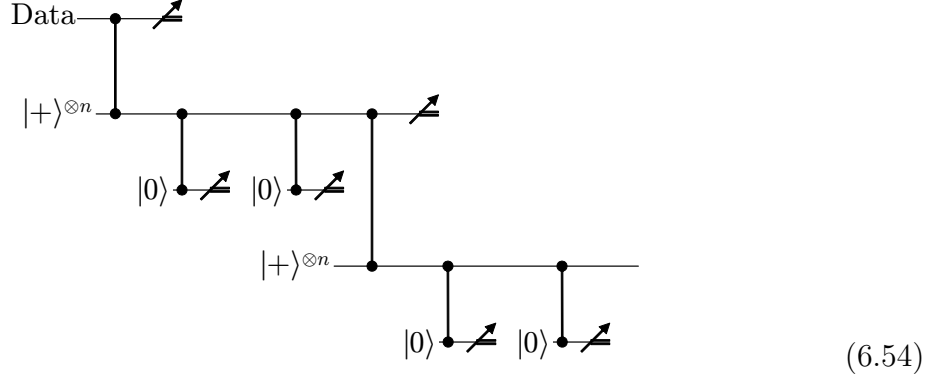
We now derive a circuit for fault-tolerant telecorrection. Begin with the following circuit for Steane's repeated syndrome extraction:



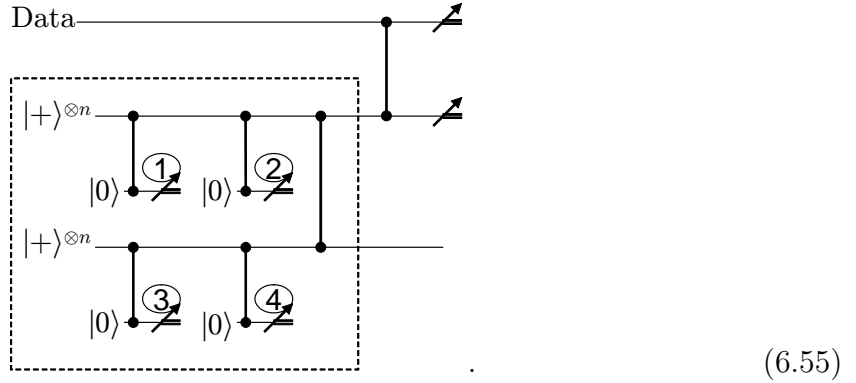
In the circuit, wires and gates represent encoded qubits and encoded operations respectively, and  $|0\rangle$  represents Steane's fault-tolerant ancilla creation circuit. The circuit performs two  $Z$  syndrome extractions followed by two  $X$  syndromes, and generalizes to more than two extractions of each in the obvious way.

We replace each of the encoded Hadamard operations in Circuit (6.53) by the transport

circuit of Equation (5.4), to give



where we have omitted showing the necessary classical feed-forward associated with the transport steps. We commute various operations to finally give



The dashed box encloses the telecorrector creation circuit. Measurements 1 and 2 correspond to  $Z$  syndrome measurements of the original circuit, and measurements 3 and 4 correspond to  $X$  syndrome measurements. However, these measurements do not directly give the syndromes of the input data, since they must be adjusted due to the output of the transport-circuit measurements. The details of this will be derived below. Note however that we can determine if the syndromes of like type will agree, and post-select for this, before the telecorrector has interacted with the data.

To understand exactly what state the telecorrector is, we now consider the evolution of Circuit (6.55), in the case of noise-free operations. The telecorrector creation circuit begins with the state  $|+\rangle^{\otimes N} \otimes |+\rangle^{\otimes N}$ . We note that the state  $|+\rangle^{\otimes N}$  can be written, without normalization, as

$$|+\rangle^{\otimes N} = \sum_{s=0}^{2^{(N-1)/2}-1} \vec{X}(s) |+\rangle_L, \quad (6.56)$$

where  $s$  labels  $X$ -syndromes of the code (we are assuming the code encodes one qubit, hence there are  $2^{(N-1)/2}$   $X$ -syndromes),  $\vec{X}(s)$  is some tensor product of  $X$ s and  $I$ s having syndrome  $s$ , and  $|+\rangle_L$  is the encoded  $|+\rangle$  state. Each  $|+\rangle^{\otimes N}$  in the circuit undergoes two  $X$  syndrome extractions, each consisting of a controlled phase with an encoded  $|0\rangle$  ancilla and subsequent measurement. The first  $X$  syndrome extraction performed on a  $|+\rangle^{\otimes N}$  randomly collapses it to one of the terms in Equation (6.56). In the noise-free case, the second syndrome extraction has no effect. Thus, the state of the telecorrector after all syndrome extractions, but before the controlled phase connecting the two halves, is

$$\left(\vec{X}(s_z)|+\rangle_L\right) \otimes \left(\vec{X}(s_x)|+\rangle_L\right), \quad (6.57)$$

where  $s_z$  and  $s_x$  represent the syndrome measurement results from the top and bottom halves of the circuit respectively.

Next we apply the controlled phase between the two halves of the telecorrector creation circuit. This gives the state

$$\left(\vec{X}(s_z)\vec{Z}(s_x) \otimes \vec{X}(s_x)\vec{Z}(s_z)\right) \left| \bigcirc - \bigcirc \right\rangle, \quad (6.58)$$

where we have commuted the controlled phase through the  $X$  operations, and the ket is the encoded two-node cluster state. Thus, a noise-free telecorrector state is an encoded two-node cluster state, up to known Pauli operations.

We now consider the remaining operations in Circuit (6.55) that complete the telecorrection of the data. The controlled phase between the telecorrector and data gives the state

$$\left(\vec{Z}(s_z) \otimes \vec{X}(s_z)\vec{Z}(s_x) \otimes \vec{X}(s_x)\vec{Z}(s_z)\right) \left| \psi - \bigcirc - \bigcirc \right\rangle, \quad (6.59)$$

where the ket is the state (on encoded qubits) obtained by applying a controlled phase between the data state, denoted  $\psi$ , and a two-node cluster state.

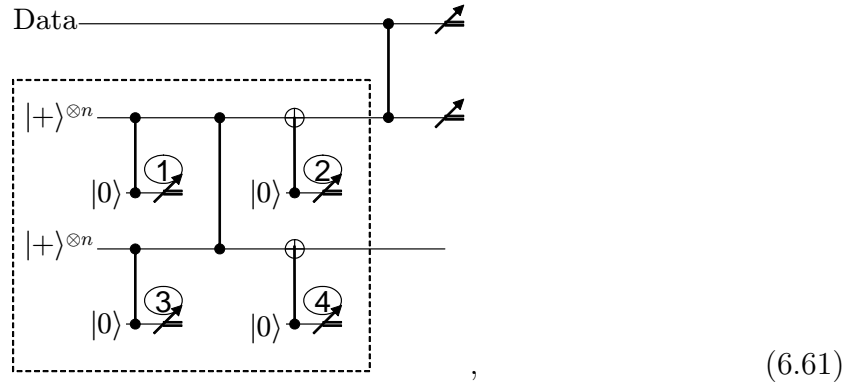
The final two measurements are then performed. These are encoded  $X$ -basis measurements on the data and one half of the telecorrector. An encoded  $X$ -basis measurement is performed by measuring each physical qubit in the  $X$  basis, adjusting the measurement results to remove the effects of known Pauli operations ( $\vec{Z}(s_z)$  in the case of the measurement of the data, and  $\vec{Z}(s_x)$  in the case of the measurement of the top half of the telecorrector), and performing classical error correction on the resulting bit string. For the codes we consider, the measurement outcome is 0 if the corrected bit string has even weight, and 1 otherwise. The corrections performed during the two encoded measurements have the effect of eliminating any errors present in the input state, and also certain errors introduced by telecorrector creation, subject to the weight of those errors being not too large.

Let the measurement result on the data and telecorrector be  $m_1$  and  $m_2$  respectively. Then, the output state is

$$\vec{X}(s_x)\vec{Z}(s_z)(Z^{\otimes n})^{m_1}(X^{\otimes n})^{m_2}|\psi\rangle, \quad (6.60)$$

to which we apply the appropriate Pauli operators, giving the final output of the telecorrector, the error-corrected version of the state  $|\psi\rangle$ .

Finally, note that the following straightforward modification to the telecorrection circuit,



provides an improved noise-threshold performance compared with Circuit (6.55). In Circuit (6.61), measurements 1 and 4 correspond to  $Z$  syndrome extraction in Steane's protocol, and measurements 2 and 3 correspond to  $X$  syndrome extraction. The circuit has the property that the post-selection for preagreeing syndromes will eliminate a larger class of errors than is the case for Circuit (6.55). For example,  $X$  errors which propagate from either ancilla 1 and 3 to become  $Z$  errors on the telecorrector will very likely cause syndromes 2 or 4 to disagree with 3 or 1 respectively. Also, certain types of noise caused by a failed controlled phase between the two halves of the telecorrector will also cause disagreeing syndromes.

## Concluding remarks

---

In this thesis we have achieved a number of new results regarding the design of reliable quantum information processing devices. The results concern the construction of naturally fault-tolerant devices, the design of reliable quantum wires, and the design and analysis of protocols for fault-tolerant optical cluster-state computing.

Natural fault-tolerance has the potential to radically simplify the construction of reliable quantum devices. We have identified an important new design criteria for such systems: they must use *degenerate* quantum codes. However, a caveat in our results does leave one door open for the use of nondegenerate codes in naturally fault-tolerant systems – it may be possible use nondegenerate codes if *effective interactions* are used. Our results also have implications to wider many-body physics. The statics and dynamics of many-body systems are often understood in terms of the eigenstates of the system's Hamiltonian. We have proven that no eigenstate of any nontrivial physically-plausible Hamiltonian can be near to a state that belongs to a nondegenerate code.

We have seen that a very simple control scheme can turn a channel with poor communication fidelity into one with a high fidelity. A simple network of statically-interacting spins is not normally able to perform a useful quantum information processing task. However, such a system can be made to act as a high-fidelity quantum wire, by controlling just four of the spins. The appropriate control scheme can easily be found for any given network, by utilizing techniques of quantum coding. Generally speaking, the advantage of achieving a quantum information processing task by using only simple control is that the system will be easier to isolate from external noise, and easier to scale to larger sizes. Note that we did not consider the effect of external noise on our scheme. An interesting topic for further investigation is whether such simple control schemes are still useful when external noise is present. Perhaps combining our method of control with ideas of natural fault-tolerance would be fruitful in this case.

This thesis contains the most detailed analysis to date of the effect of noise on optical quantum computers. We considered one of the most promising of the proposed varieties

of optical quantum computing, the so-called optical cluster-state scheme. We constructed a new protocol for optical cluster-state error-correction, and then simulated the effects of noise on this protocol. From these results we derived the noise threshold for optical cluster-state quantum computing. We found that optical quantum computers can reliably deal with the simultaneous effects of both photon loss noise and depolarization noise, so long as the rate of photon loss per basic operation is  $< 3 \times 10^{-3}$  and the rate of depolarization error per basic operation is  $< 10^{-4}$ . This shows that the threshold for depolarization noise is less than that calculated previously for more abstract physical models. However, this difference is surprisingly small given the extra noise sources present in the optical cluster-state scenario: photon loss, and an inherent failure rate of 50% for two-qubit gates. Our error-correction protocol was able to achieve such large threshold values due to the novel ways in which we exploited properties of the cluster-state model. The ability to parallelize and pre-measure during cluster-state error-correction proved particularly useful. Our protocol currently has substantial resource requirements, which will need to be reduced before the protocol can be used in a practical setting.



---

## References

---

- [ABO97] D. Aharonov and M. Ben-Or. Fault tolerant computation with constant error. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*, pages 176–188, 1997. . . . . 53
- [ABO99] D. Aharonov and M. Ben-Or. Fault-tolerant quantum computation with constant error rate. *arXiv:quant-ph/9906129*, 1999. . . . . 53
- [AGP06] P. Aliferis, D. Gottesman, and J. Preskill. Quantum accuracy threshold for concatenated distance-3 codes. *Quant. Inf. Comput.*, 6:97–105, 2006. *arXiv:quant-ph/0504218*. . . . . 53
- [Aha99] D. Aharonov. Quantum to classical phase transition in noisy quantum computers. *Phys. Rev. A*, 62(6):062311, 1999. . . . . 1
- [AL05] P. Aliferis and D. W. Leung. Fault-tolerant quantum computation with graph states. *arXiv:quant-ph/0503130*, 2005. . . . . 120, 121, 131
- [Amb04] A. Ambainis. Quantum walk algorithm for element distinctness. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 22–31. IEEE Computer Society, 2004. *arXiv:quant-ph/0311001*. 81
- [Bac97] Dave Morris Bacon. *Decoherence, Control, and Symmetry in Quantum Computers*. PhD thesis, University of California, Berkeley, 1997. *arXiv:quant-ph/0305025*. . . . . 54
- [Bac05] D. Bacon. Operator quantum error correcting subsystems for self-correcting quantum memories. *arXiv:quant-ph/0506023*, 2005. . . . . 55, 59, 71
- [BB84] C. H. Bennett and G. Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Proceedings of IEEE International Conference on Computers, Systems and Signal Processing*, pages 175–179, New York, 1984. IEEE. Bangalore, India, December 1984. . . . . 4

- 
- [BBC<sup>+</sup>93] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, and W. K. Wootters. Teleporting an unknown quantum state via dual classical and EPR channels. *Phys. Rev. Lett.*, 70:1895–1899, 1993. .... 2
- [BBM92] C. H. Bennett, G. Brassard, and N. D. Mermin. Quantum cryptography without Bell’s theorem. *Phys. Rev. Lett.*, 68(5):557–559, 1992. .... 2
- [BBPS96] C. H. Bennett, H. J. Bernstein, S. Popescu, and B. Schumacher. Concentrating partial entanglement by local operations. *Phys. Rev. A*, 53(4):2046–2052, 1996. arXiv:quant-ph/9511030. .... 2
- [BK05] S. D. Barrett and P. Kok. Efficient high-fidelity quantum computation using matter qubits and linear optics. *Phys. Rev. A*, 71:060310, 2005. arXiv:quant-ph/0408040. .... 166
- [BK06] S. Bravyi and A. Kitaev. Universal quantum computation with ideal clifford gates and noisy ancillas. *Phys. Rev. A*, 71:022316, 2006. arXiv:quant-ph/0403025. .... 133
- [Bos03] S. Bose. Quantum communication through an unmodulated spin chain. *Phys. Rev. Lett.*, 91:207901, 2003. arXiv:quant-ph/0212041. .... 74, 75, 76
- [BR05] D. E. Browne and T. Rudolph. Resource-efficient linear optical quantum computation. *Phys. Rev. Lett.*, 95(1):010501, 2005. .... 95, 112, 115, 118
- [BvL05] S. L. Braunstein, , and P. van Loock. Quantum information with continuous variables. 77:513–577, 2005. arXiv:quant-ph/0410100. .... 106
- [CCD<sup>+</sup>03] A. M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann, and D. A. Spielman. Exponential algorithmic speedup by quantum walk. In *STOC*, pages 59–68, 2003. arXiv:quant-ph/0209131. .... 81
- [CDEL04] M. Christandl, N. Datta, A. Ekert, and A. J. Landahl. Perfect state transfer in quantum spin networks. *Phys. Rev. Lett.*, 92:187902, 2004. arXiv:quant-ph/0309131. .... 74, 75, 76
- [CRSS97] A. R. Calderbank, E. M. Rains, P. W. Shor, and N. J. A. Sloane. Quantum error correction and orthogonal geometry. *Phys. Rev. Lett.*, 78:405–8, 1997. 68

- 
- [DA06] J. Preskill D. Aharonov, A. Y. Kitaev. Fault-tolerant quantum computation with long-range correlated noise. *Phys. Rev. Lett.*, 96:050504, 2006. arXiv:quant-ph/0510231. .... 53
- [Die82] D. Dieks. Communication by EPR devices. *Phys. Lett. A*, 92(6):271–272, 1982. .... 3, 17
- [DiV00] D. P. DiVincenzo. The physical implementation of quantum computation. *Fortschritte der physik — progress of physics*, 48(9–11):771–783, 2000. arXiv:quant-ph/0002077. .... 74
- [DKLP02] E. Dennis, A. Y. Kitaev, A. Landahl, and J. Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43:4452–4505, 2002. arXiv:quant-ph/0110143. .... 54
- [EH00] M. Ettinger and P. Høyer. On quantum algorithms for noncommutative hidden subgroups. *Advances in Applied Mathematics*, 25:239–251, 2000. . 3
- [Eke91] A. K. Ekert. Quantum cryptography based on Bell’s theorem. *Phys. Rev. Lett.*, 67(6):661–663, 1991. .... 2
- [EM96] A. Ekert and C. Macchiavello. Error correction in quantum communication. *Phys. Rev. Lett.*, 77:2585, 1996. arXiv:quant-ph/9602022. .... 67
- [FDF<sup>+</sup>02] J. D. Franson, M. M. Donegan, M. J. Fitch, B. C. Jacobs, and T. B. Pittman. High-fidelity quantum logic operations using linear optical elements. *Phys. Rev. Lett.*, 89(13):137901, 2002. arXiv:quant-ph/0202160. .... 111
- [GC99] D. Gottesman and I. L. Chuang. Quantum teleportation is a universal computational primitive. *Nature*, 402:390–392, 1999. arXiv:quant-ph/9908010. 110
- [Got] D. Gottesman. Errata for [Got97], at URL [perimeterinstitute.ca/researchers/people/dgottesman](http://perimeterinstitute.ca/researchers/people/dgottesman). .... 67
- [Got97] D. Gottesman. *Stabilizer Codes and Quantum Error Correction*. PhD thesis, California Institute of Technology, Pasadena, CA, 1997. arXiv:quant-ph/9705052. .... 67, 175
- [Got03] D. Gottesman. Private communication, 2003. .... 67

- 
- [GPW<sup>+</sup>04] S. Gasparoni, J.-W. Pan, P. Walther, T. Rudolph, and A. Zeilinger. Realization of a photonic CNOT gate sufficient for quantum computation. *Phys. Rev. Lett.*, 93:020504, 2004. arXiv:quant-ph/0404107. .... 106
- [Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *28th ACM Symposium on Theory of Computation*, page 212, New York, 1996. Association for Computing Machinery. .... 3
- [GRW06] A. Gilchrist, K. J. Resch, and A. G. White. A semiconductor source of triggered entangled photon pairs? *arXiv:quant-ph/0602018*, 2006. .... 118
- [Hal02] S. Hallgren. Polynomial time quantum algorithms for Pell’s equation and the principal ideal problem. In *Proc. 34<sup>th</sup> Annual ACM Symposium on the Theory of Computation*, pages 653–658. ACM Press, 2002. .... 3
- [Has04] H. L. Haselgrove. Online animations. 2004. Online animations available at <http://www.physics.uq.edu.au/people/hlh/comms>. .... 83, 90
- [HDE<sup>+</sup>06] M. Hein, W. Dr, J. Eisert, R. Raussendorf, M. Van den Nest, and H.-J. Briegel. Entanglement in graph states and its applications. *arXiv:quant-ph/0602096*, 2006. To appear in the Proceedings of the International School of Physics “Enrico Fermi” on “Quantum Computers, Algorithms and Chaos”, Varenna, Italy, July, 2005. .... 2
- [HGMR04] A. J. F. Hayes, A. Gilchrist, C. R. Myers, and T. C. Ralph. *J. Opt. B*, 6:533–541, 2004. .... 111
- [HJ91] R. A. Horn and C. R. Johnson. *Topics in matrix analysis*. Cambridge University Press, Cambridge, 1991. .... 79
- [Hol98] A. S. Holevo. The capacity of the quantum channel with general signal states. *IEEE. Trans. Inf. Theory*, 44(1):269–273, 1998. .... 3
- [JL03] R. Jozsa and N. Linden. On the role of entanglement in quantum computational speed-up. *Proc. Roy. Soc. Lond. A*, 459(2036):2011–2032, 2003. arXiv:quant-ph/0201143. .... 3
- [Kem03] J. Kempe. Quantum random walks: an introductory overview. *Contemporary Physics*, 44(4):307–327, 2003. arXiv:quant-ph/0303081. .... 80

- 
- [Kit97a] A. Y. Kitaev. Fault-tolerant quantum computation by anyons. *arXiv:quant-ph/9707021*, 1997. .... 54, 59, 71
- [Kit97b] A. Y. Kitaev. Quantum computations: algorithms and error correction. *Russ. Math. Surv.*, 52(6):1191–1249, 1997. .... 53
- [Kit97c] A. Y. Kitaev. Quantum error correction with imperfect gates. In A. S. Holevo O. Hirota and C. M. Caves, editors, *Quantum Communication, Computing, and Measurement*, pages 181–188, New York, 1997. Plenum Press. .... 53
- [KKR04] J. Kempe, A. Kitaev, and O. Regev. The complexity of the local Hamiltonian problem. In *Proc. 24th FSTTCS*, pages 373–383, 2004. *arXiv:quant-ph/0406180*. .... 71, 72
- [KLM01] E. Knill, R. Laflamme, and G. J. Milburn. A scheme for efficient quantum computation with linear optics. *Nature*, 409(6816):46–52, January 2001. 105
- [KLZ96] E. Knill, R. Laflamme, and W. Zurek. Accuracy threshold for quantum computation. *arXiv:quant-ph/9610011*, 1996. .... 53
- [KLZ98] E. Knill, R. Laflamme, and W. H. Zurek. Resilient quantum computation: error models and thresholds. *Proc. Roy. Soc. A*, 454(1969):365–384, 1998. *arXiv:quant-ph/9702058*. .... 53, 125
- [KMN<sup>+</sup>05] P. Kok, W. J. Munro, K. Nemoto, T. C. Ralph, J. P. Dowling, and G. J. Milburn. Linear optical quantum computing. *arXiv:quant-ph/0512071*, 2005. 109
- [Kni02] E. Knill. Quantum gates using linear optics and postselection. *Phys. Rev. A*, 66(5):052306, 2002. *arXiv:quant-ph/0110144*. .... 111
- [Kni04] E. Knill. Fault-tolerant postselected quantum computation: Threshold analysis. *arXiv:quant-ph/0404104*, 2004. .... 133
- [Kni05] E. Knill. Quantum computing with realistically noisy devices. *Nature*, 434(7029):39–44, 2005. .... 5, 121, 131, 132, 133, 158, 166, 167
- [LX04] S. Ling and C. Xing. *Coding Theory – A First Course*. Cambridge University Press, 2004. .... 138
- [ME99] M. Mosca and A. Ekert. The hidden subgroup problem and eigenvalue estimation on a quantum computer. *arXiv:quant-ph/9903071*, 1999. .... 3

- 
- [Mil89] G. J. Milburn. Quantum optical Fredkin gate. *Phys. Rev. Lett.*, 62(18):2124, 1989. .... 105
- [ML04] A. Mizel and D. A. Lidar. Three and four-body interactions in spin-based quantum computers. *Phys. Rev. Lett.*, 92:077903, 2004. arXiv:quant-ph/0401081. .... 59
- [NC97] M. A. Nielsen and I L. Chuang. Programmable quantum gate arrays. *Phys. Rev. Lett.*, 79(2):321–324, 1997. .... 110
- [NC00] M. A. Nielsen and I. L. Chuang. *Quantum computation and quantum information*. Cambridge University Press, Cambridge, 2000. 2, 9, 20, 21, 22, 35, 74, 108, 120, 145
- [ND05] Michael A. Nielsen and C. M. Dawson. Fault-tolerant quantum computation with cluster states. *Phys. Rev. A*, 71(5):042323, 2005. .... 96, 120, 131
- [Nie04] Michael A. Nielsen. Optical quantum computation using cluster states. *Phys. Rev. Lett.*, 93(4):040503, 2004. .... 95, 112, 113, 134
- [Nie05] Michael A. Nielsen. Cluster-state quantum computation. *arXiv:quant-ph/0504097*, 2005. to appear in *Rev. Math. Phys.* .... 96
- [OL04] T. J. Osborne and N. Linden. The propagation of quantum information through a spin system. *Phys. Rev. A*, 69:052315, 2004. arXiv:quant-ph/0312141. .... 75, 76, 78, 83, 84
- [OPW<sup>+</sup>03] J. L. O’Brien, G. J. Pryde, A. G. White, T. C. Ralph, and D. Branning. Demonstration of an all-optical quantum controlled-not gate. *Nature*, 426(6964):264–267, 2003. .... 106
- [OS04] T. J. Osborne and S. Severini. Quantum algorithms and covering spaces. *arXiv:quant-ph/043127*, 2004. .... 79, 81
- [PFJF03] T. B. Pittman, M. J. Fitch, B. C. Jacobs, and J. D. Franson. Experimental controlled-NOT logic gate for single photons in the coincidence basis. *Phys. Rev. A*, 68(3):032316, 2003. .... 106
- [Pre98] J. Preskill. *Physics 229: Advanced mathematical methods of physics — Quantum computation and information*.

- 
- California Institute of Technology, Pasadena, CA, 1998.  
<http://www.theory.caltech.edu/people/preskill/ph229/>. . . . . 74
- [Rau03] R. Raussendorf. *Measurement-based quantum computation with cluster states*. PhD thesis, Ludwig-Maximilians Universität München, 2003.  
<http://edoc.ub.uni-muenchen.de/archive/00001367>. . . . . 120, 131
- [RB01] R. Raussendorf and H. J. Briegel. A one-way quantum computer. *Phys. Rev. Lett.*, 86(22):5188–5191, 2001. . . . . 2, 96, 98
- [RBB03] R. Raussendorf, D. E. Browne, and H. J. Briegel. The one-way quantum computer — a non-network model of quantum computation. *Phys. Rev. A*, 68(2):022312, 2003. . . . . 96, 122
- [Rei04] Ben W. Reichardt. Improved ancilla preparation scheme increases fault-tolerant threshold. *arXiv:quant-ph/0406025*, 2004. . . . . 5, 158, 166
- [RHG05a] T. C. Ralph, A. J. F. Hayes, and A. Gilchrist. Loss-tolerant optical qubits. *Phys. Rev. Lett.*, 95:100501, 2005. *arXiv:quant-ph/0501184*. . . . . 121
- [RHG05b] R. Raussendorf, J. Harrington, and K. Goyal. A fault-tolerant one-way quantum computer. *arXiv:quant-ph/0510135*, 2005. . . . . 120, 121, 131
- [RWMM02] T. C. Ralph, A. G. White, W. J. Munro, and G. J. Milburn. Simple scheme for efficient linear optics quantum gates. *Phys. Rev. A*, 65(1):012314, 2002. *arXiv:quant-ph/0108049*. . . . . 111
- [SCCA05] K. M. Svore, A. W. Cross, I. L. Chuang, and A. V. Aho. Pseudothreshold or threshold? - more realistic threshold estimates for fault-tolerant quantum computing. *arXiv:quant-ph/0508176*, 2005. . . . . 166
- [Sch95] Benjamin Schumacher. Quantum coding. *Phys. Rev. A*, 51:2738, 1995. . . 3
- [Sch96] B. W. Schumacher. Sending entanglement through noisy quantum channels. *Phys. Rev. A*, 54:2614, 1996. . . . . 3
- [SDT06] K. M. Svore, D. P. DiVincenzo, and B. M. Terhal. Noise threshold for a fault-tolerant two-dimensional lattice architecture. *arXiv:quant-ph/0604090*, 2006. . . . . 5, 133

- [Sho94] P. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proc. 35<sup>th</sup> Annual Symposium on Foundations of Computer Science*, page 124, Los Alamitos, CA, 1994. IEEE Computer Society Press. 3
- [SJP<sup>+</sup>04] K. Sanaka, T. Jennewein, J.-W. Pan, K. Resch, and A. Zeilinger. Experimental nonlinear sign shift for linear optics quantum computation. *Phys. Rev. Lett.*, 92(1):017902, 2004. .... 106
- [STD05] K. M. Svore, B. M. Terhal, and D. P. DiVincenzo. Local fault-tolerant quantum computation. *Phys. Rev. A*, 72(2):022317, 2005. arXiv:quant-ph/0410047. .... 166
- [Ste96] A. M. Steane. Error correcting codes in quantum theory. *Phys. Rev. Lett.*, 77:793, 1996. .... 30
- [Ste97] A. M. Steane. Active stabilisation, quantum computation and quantum state synthesis. *Phys. Rev. Lett.*, 78:2252–2255, 1997. arXiv:quant-ph/9611027. 43
- [Ste02] A. Steane. Fast fault-tolerant filtering of quantum codewords. *arXiv:quant-ph/0202036*, 2002. .... 47
- [Ste03] A. M. Steane. Overhead and noise threshold of fault tolerant quantum error correction. *Phys. Rev. A*, 68(4):042322, 2003. 5, 45, 53, 121, 131, 132, 137, 154, 158
- [SW97] B. Schumacher and M. D. Westmoreland. Sending classical information via noisy quantum channels. *Phys. Rev. A*, 56(1):131–138, 1997. .... 3
- [SYA<sup>+</sup>06] R. M. Stevenson, R. J. Young, P. Atkinson, K. Cooper, D. A. Ritchie, and A. J. Shields. A semiconductor source of triggered entangled photon pairs. *Nature*, 439:179–182, 2006. .... 118
- [TB04] B. M. Terhal and G. Burkard. Fault-tolerant quantum computation for local non-Markovian noise. *Phys. Rev. A*, 71(1):012336, 2004. arXiv:quant-ph/0402104. .... 53, 131
- [TED<sup>+</sup>05] J. M. Taylor, H.-A. Engel, W. Dür, A. Yacoby, C. M. Marcus, P. Zoller, and M. D. Lukin. Fault-tolerant architecture for quantum computation using electrically controlled semiconductor spins. *Nature Physics*, 1:177–183, 2005. 5



- 
- [Ter] David Terr. Golay code. from *Mathworld*, URL <http://mathworld.wolfram.com/GolayCode.html>. . . . . 138
- [TMC<sup>+</sup>06] D. D. Thaker, T. S. Metodi, A. W. Cross, I. L. Chuang, and F. T. Chong. Quantum memory hierarchies: efficient designs to match available parallelism in quantum computing. *arXiv:quant-ph/0604070*, 2006. . . . . 5
- [VBR05] M. Varnava, D. E. Browne, and T. Rudolph. Loss-tolerant one-way quantum computation – a horticultural approach. *arXiv:quant-ph/0507036*, 2005. 121
- [VC04] F. Verstraete and J. I. Cirac. Renormalization algorithms for quantum-many body systems in two and higher dimensions. *arXiv:quant-ph/0407066*, 2004. 3
- [vDHI03] W. van Dam, S. Hallgren, and L. Ip. Quantum algorithms for some hidden shift problems. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pages 489–498, 2003. . . . . 3
- [Vid03] G. Vidal. Efficient classical simulations of slightly entangled quantum computations. *Phys. Rev. Lett.*, 91(14):147902, 2003. *arXiv:quant-ph/0301063*. 3
- [Vid04] G. Vidal. Efficient simulation of one-dimensional quantum many-body systems. *Phys. Rev. Lett.*, 93:040502, 2004. *arXiv:quant-ph/0310089*. . . . . 3
- [Woo98] W. K. Wootters. Entanglement of formation of an arbitrary state of two qubits. *Phys. Rev. Lett.*, 80(10):2245–2248, 1998. . . . . 93
- [WRR<sup>+</sup>05] P. Walther, K. J. Resch, T. Rudolph, E. Schenck, H. Weinfurter, V. Vedral, M. Aspelmeyer, and A. Zeilinger. Experimental one-way quantum computing. *Nature*, 434:169–176, 2005. . . . . 112
- [WZ82] W. K. Wootters and W. H. Zurek. A single quantum cannot be cloned. *Nature*, 299:802–803, 1982. . . . . 3, 17
- [YKI88] Y. Yamamoto, M. Kitagawa, and K. Igeta. In *Proc. 3rd Asia-Pacific Phys. Conf.*, Singapore, 1988. World Scientific. . . . . 105
- [Zal98] C. Zalka. Efficient simulation of quantum systems by quantum computers. *Fort. Der. Physik*, 46(6-8):877–879, 1998. . . . . 4

- [ZZC<sup>+</sup>04] Z. Zhao, A.-N. Zhang, Y.-A. Chen, H. Zhang, J.-F. Du, T. Yang, and J.-Wei. Pan. Experimental demonstration of a non-destructive controlled-NOT quantum gate for two independent photon-qubits. *Phys. Rev. Lett.*, 94:030501, 2004. arXiv:quant-ph/0404129. .... 106

---

# Index

---

- active fault-tolerance, 54
- ancilla creation
  - clusterized, 137
  - Steane's circuit, 45
- beamsplitter, 107
- binary symmetric channel, 10
- bit-flip code, 23
- Clifford group, 101
- cluster state, 96
- cluster-state computation, 97
  - conversion from quantum circuit, 104
  - equivalence with circuit model, 98
  - parallelization in, 101
- codespace (classical), 14
- concatenation
  - of fault-tolerant protocols, 51, 150
  - of quantum codes, 26
- concurrence, 78, 93
- coset leader, 17
- CSS codes, 29
  - code states, 29
  - correction procedure, 36
  - decoding, 38
  - encoding, 33
  - fault-tolerant implementation, 43
  - logical operations, 34
  - syndrome extraction, 37
- decoding
  - classical, 11
  - linear codes, 15
  - optical cluster protocol, 145
- degenerate codes, 28
- discretization of errors, 21
- dual codes, 17
- entanglement, 2, 78
- error-correcting codes
  - classical, 10
  - quantum, 17
- fault tolerance, 40
- fault-tolerant ancilla creation, 45
- ferromagnetism, 54
- fusion gate, 115
- generator matrix, 13
- Grover's algorithm, 3
- Hamming code, 14
- Hamming distance, 11
- Hamming weight, 11
- Heisenberg interaction, 81
- independent depolarization channel, 19
- independent error model, 20
- KLM scheme, 109
- L-local interactions, 59
- linear codes, 13
  - decoding, 15
  - encoding, 14

- magic state distillation, 133
- MATLAB, 63, 79, 81, 147
- maximum-likelihood decoding, 11
- microcluster, 134
- natural fault-tolerance, 54
- no-cloning theorem, 17
- noise threshold, 49
- nondegenerate codes, 28, 67
- operation elements, 20
- operator-sum representation, 20
- parallel fusion, 134
- parity check matrix, 15
- Pauli frame, 103
  - rules for updating, 103
- phantom spins, 85
- phase-flip code, 25
- photon loss errors, 143
- polarization encoding (photonic qubit), 106
- postselection, 136
- preagreeing syndromes, 142
- quantum communication, 74
- quantum error-correction conditions, 22
- quantum walk, 80
- repetition code, 13
- Shor code, 26
- Shor's factoring algorithm, 3
- singular value decomposition, 79
- spatial encoding (photonic qubit), 106
- standard form (generator matrix), 15
- state injection, 133
- Steane's fault-tolerant protocol, 43
- syndrome (classical), 15
- telecorrector
  - circuit-based, 168
  - clusterized, 140–142
- XY interaction, 86