

The ABCs of the color code:
A study of topological quantum codes
as toy models for fault-tolerant quantum
computation and quantum phases of matter

Thesis by
Aleksander Marek Kubica

In Partial Fulfillment of the Requirements for the
degree of
Doctor of Philosophy

Caltech

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2018
Defended June 8, 2017

© 2018

Aleksander Marek Kubica
ORCID: 0000-0001-8213-8190

All rights reserved

*To my parents
Janina and Marek
with gratitude*

ACKNOWLEDGEMENTS

I am deeply grateful to my academic advisor, John Preskill, who is an exceptional scientist as well as a patient and understanding mentor. John's immense knowledge and his crystal-clear explanations of even the most complicated problems in physics have always been a source of inspiration for me. I am also thankful to Jason Alicea, Fernando Brandão and Alexei Kitaev for being on my thesis committee, finding time to discuss my work and providing useful feedback.

I would like to thank all the postdocs at IQIM who I had a chance to interact with — Gorjan Alagic, Mario Berta, Elizabeth Crosson, Nicolas Delfosse, Glen Evenbly, Philippe Faist, David Gosset, Tomas Jochym-O'Connor, Olivier Landon-Cardinal, Spiros Michalakis, Fernando Pastawski and Burak Şahinoğlu. The discussions we had were very inspiring and motivating.

I would like to thank Stephen Bartlett, Matthias Christandl, Daniel Gottesman, Michał Horodecki, Ray Laflamme, David Poulin, Rafał Demkowicz-Dobrzański and Krysta Svore for inviting me for many stimulating visits. I have always had a great time explaining my work to you as well as learning about your new results.

I have been lucky to meet many great people and amazing researchers, including Dave Aasen, Sergey Bravyi, Ben Brown, Dan Browne, Earl Campbell, Xie Chen, Steve Flammia, Jeongwan Haah, Nick Hunter-Jones, JędreK Kaniewski, Michael Kastoryano, Isaac Kim, Anna Kómár, Shaun Maguire, Charlie Marcus, Jenia Mozgunov, Oskar Painter, Leonid Pryadko, Grant Salton, Sujeet Shukla, Sam Roberts, Nathan Wiebe, Dom Williamson, Ted Yoder, Iliia Zintchenko and Will Zeng.

I would like to give special thanks to two of my good friends and collaborators, Michael Beverland and Beni Yoshida, and to Héctor Bombín. You have always been kind to me and showed me how to be a better researcher and a human being!

I am particularly indebted to all my Caltech friends — Wael Halbawi, Ruby Lopez-Zenteno, Carlos Perez-Arancibia, Krishna Shankar and Christos Thrampoulidis, as well as to my awesome housemate, Aidan Chatwin-Davies, who proofread many things I wrote. The time I spent in Pasadena was great mostly because of you!

I would like to thank Henry and Grazyna Bauer for their fellowship and Dr. David and Barbara Groce for travel support.

Last but not least, a big thank you to my family, in particular my parents Janina and Marek for unconditional support. I would not have made it this far without you!

ABSTRACT

This thesis is devoted to studying a class of quantum error-correcting codes — topological quantum codes. We explore the question of how one can achieve fault-tolerant quantum computation with topological codes. We treat quantum error-correcting codes not only as a compelling ingredient needed to build a quantum computer, but also as a useful theoretical tool in other areas of physics. In particular, we explore what insights topological codes can provide into challenging questions, such as the classification of quantum phases of matter.

In this thesis, we focus on a family of topological codes — color codes, which are particularly intriguing due to the rich physics they display and their computational power. We start by introducing color codes and explaining their basic properties. Then, we show how to perform fault-tolerant universal quantum computation with three-dimensional color codes by transverse gates and code switching. We later compare the resource overhead of the code-switching approach with that of a state distillation scheme. We discuss how to perform error correction with the toric and color codes, as well as introduce local decoders for those two families of codes. By exploiting a connection between error correction and statistical mechanics we estimate the storage threshold error rates for bit-flip and phase-flip noise in the three-dimensional color code. We finish by showing that the color and toric code families in d dimensions are equivalent in a sense of local unitary transformations and explore implications of this equivalence.

PUBLISHED CONTENT AND CONTRIBUTIONS

- [BKS18] Michael Beverland, Aleksander Kubica, and Krysta Svore. “The cost of universality: A comparative study of the overhead of state distillation and code switching in color codes”. In: *in preparation* (2018). A.K. and M.B. both contributed significantly to this work.
- [KB15] Aleksander Kubica and Michael Beverland. “Universal transversal gates with color codes: A simplified approach”. In: *Phys. Rev. A* 91 (2015), p. 032330. DOI: 10.1103/PhysRevA.91.032330. A.K. and M.B. both contributed significantly to this work.
- [KDP18] Aleksander Kubica, Nicolas Delfosse, and John Preskill. “Local decoders for topological toric and color codes in any dimensions”. In: *in preparation* (2018). A.K. was the main contributor to this work.
- [Kub+18] Aleksander Kubica et al. “Three-Dimensional Color Code Thresholds via Statistical-Mechanical Mapping”. In: *Phys. Rev. Lett.* 120 (2018), p. 180501. DOI: 10.1103/PhysRevLett.120.180501. A.K. was the main contributor to this work.
- [KYP15] Aleksander Kubica, Beni Yoshida, and Fernando Pastawski. “Unfolding the color code”. In: *New J. Phys.* 17 (2015), p. 083026. DOI: 10.1088/1367-2630/17/8/083026. A.K. was the main contributor to this work.

TABLE OF CONTENTS

Acknowledgements	v
Abstract	vii
Published Content and Contributions	viii
Table of Contents	ix
Chapter I: Introduction	1
1.1 Brief introduction to topological stabilizer codes	5
1.2 Thesis organization and contributions	9
Chapter II: Universal transversal gates with color codes: A simplified approach	13
2.1 Color code in two dimensions	15
2.2 Color code in higher dimensions	21
2.3 Transversal gates in color codes	30
2.4 Universal transversal gates with color codes	35
Chapter III: The cost of universality in surface and color codes: State distil- lation versus code switching	41
3.1 Overview of stabilizer and subsystem codes	42
3.2 Error correction with stabilizer codes	45
3.3 Non-Clifford gates from distillation	50
3.4 Non-Clifford gates from code switching	54
3.5 Error threshold analysis	59
3.6 Overhead comparison	63
Chapter IV: Local decoders for topological toric and color codes in any dimensions	73
4.1 Beyond Toom’s rule	81
4.2 Decoding of the toric code	85
4.3 Proof of threshold	89
4.4 Decoding of the color code	98
4.5 Thresholds of 3D toric and color code decoders	113
4.6 Discussion	117
Chapter V: Three-dimensional color code thresholds via statistical-mechanical mapping	119
5.1 3D stabilizer color code	121
5.2 Statistical-mechanical models	123
5.3 Duality of models for zero disorder	124
5.4 Heuristic estimates of the threshold	126
5.5 Proof of implication	127
5.6 Phase diagram	130
5.7 Finding phase transitions	131
5.8 Discussion	138
Chapter VI: Unfolding the color code	141

6.1 Topological color code without boundaries	144
6.2 Rigorous proof of the equivalence	162
6.3 Topological color code with boundaries	168
6.4 Transversal gates	178
Bibliography	183

Chapter 1

INTRODUCTION

*It is a privilege to be able to explore.
If you know what you're doing, don't do it!*
—H. Jeff Kimble

The field of quantum computing [Pre99; NC10] and the idea of the quantum computer can be traced back Feynman's question of simulating physics with computers [Fey82]. Feynman hypothesized that there could be two classes of computation: classical and quantum. At that time it was already known that a universal Turing machine could efficiently simulate any classical computation [Tur36; Chu36]. However, it was unclear if there could exist a universal quantum simulator powerful enough to efficiently simulate any quantum system. This question was positively resolved by introducing the notion of a quantum Turing machine, which provided a concrete realization of the idea of a universal quantum computer [Ben80; Deu85]. Since then, different models of quantum computation, such as adiabatic [Far+00], topological [Kit03; Fre+03], measurement-based [RB01] and quantum circuit [Deu89] models, have been proposed and proven to be equivalent to the quantum Turing machine.

Quantum computing promises a technological breakthrough if a quantum computer is ever built. As originally suggested, we could use such a machine to efficiently simulate physics of quantum many body systems and quantum field theories [JLP11], tasks believed to be computationally intractable with classical devices. A timely question to think of now is to understand what specific computational tasks and simulations can be performed on small quantum computers operating on a couple hundred noisy physical qubits. We expect that in the future we will most likely be able to circumvent engineering limitations and manufacture devices with millions of logical qubits. Such computational power will allow us to run any quantum algorithm, for instance Shor's famous factorization algorithm [Sho94; Sho99], which offers exponential speedup compared to known classical counterparts.

Building a quantum computer is, unfortunately, a daunting task. One of the main reasons is that resources needed to perform quantum computation, such as entangle-

ment, are very fragile to uncontrolled interactions with the environment. Decoherence tends to destroy quantum information that is in a superposition of states, which subsequently leads to a failure in the computation. Since we will never be able to manufacture ideal quantum (or even classical) computer components and perfectly isolate them to prevent the leakage of quantum information into the surrounding environment, there will always be some physical noise present at the hardware level. Fortunately, in order to reliably operate a quantum computer, it suffices to reduce the level of physical noise below a certain critical value, the accuracy threshold, which can be achieved by improving the quality and control of hardware components. This impressive and far-reaching result is captured the quantum threshold theorem [Sho96; AB97; KLZ98; Pre98], which asserts that an arbitrarily long quantum computation can be performed with reasonable overhead, provided the physical noise is smaller than the accuracy threshold and not too correlated.

One of the key ideas of protecting quantum information is to encode it into quantum error-correcting codes [Sho95; Ste96a], which are formed by highly entangled many-body states that are robust to typical errors. By doing so, one can achieve a level of effective noise affecting the encoded information, which we call the logical error rate, that is below the physical error rate for unencoded information. However, we do not only want to reliably store quantum information on a noisy quantum computer — we would like to perform quantum algorithms as well. In order to do that, we need to be able to implement logical operations on the encoded information. According to the Solovay-Kitaev theorem [Kit97; NC10], one can find a finite set of gates, called a universal set, such that any unitary can be efficiently approximated by using exclusively gates from that set. Thus, it would suffice to know how to implement logical operations which form a universal gate set. At the same time, we require that all the operations we apply to the system are fault-tolerant, i.e. do not spread errors in an uncontrollable way. This motivates a search for quantum error-correcting codes which simultaneously provide good protection against typical noise and admit a fault-tolerant implementation of logical gates which are universal.

One of the most promising approaches to fault-tolerant quantum computing is topological quantum codes and the scheme of topological quantum computation introduced by Kitaev [Kit03; Kit06; Kit01]. The ingenious insight of Kitaev was to realize that one can use anyons, i.e. quasiparticles in a two-dimensional system with unusual statistics [Wil82], to perform quantum computation. In this approach, errors are suppressed at the hardware level and operations are implemented by braiding

and fusing anyons. We remark that recently there has been a lot of experimental effort devoted to showing the existence of anyons, in particular Majorana fermions [Ali12]. Kitaev also proposed a class of many-body systems on a lattice realizing so-called quantum double models [Kit03] whose highly entangled ground states can be viewed as a quantum error-correcting code and their localized excitations as errors in the code. The corresponding codes, called topological quantum codes [Den+02; Bom13], have many desirable features, such as geometric locality and high physical error rate tolerance. Despite their simplicity, quantum double models and the corresponding topological quantum codes capture and describe a lot of interesting concepts, such as quantum phases of matter and topological order [Wen90]. We can also view those models as toy examples to study and explore the connection between condensed matter physics and quantum error correction.

The main goal of this thesis is to study topological quantum codes as leading candidates for realizing fault-tolerant quantum computation. Moreover, we explore the insights that topological quantum codes can provide into challenging problems of many-body physics. We focus a lot of attention on a particularly intriguing topological code called the color code. The questions addressed in this thesis can be grouped into three broad categories: fault-tolerant computation, error correction and quantum phases of matter. We illustrate the basic structure of the thesis in Fig. 1.1.

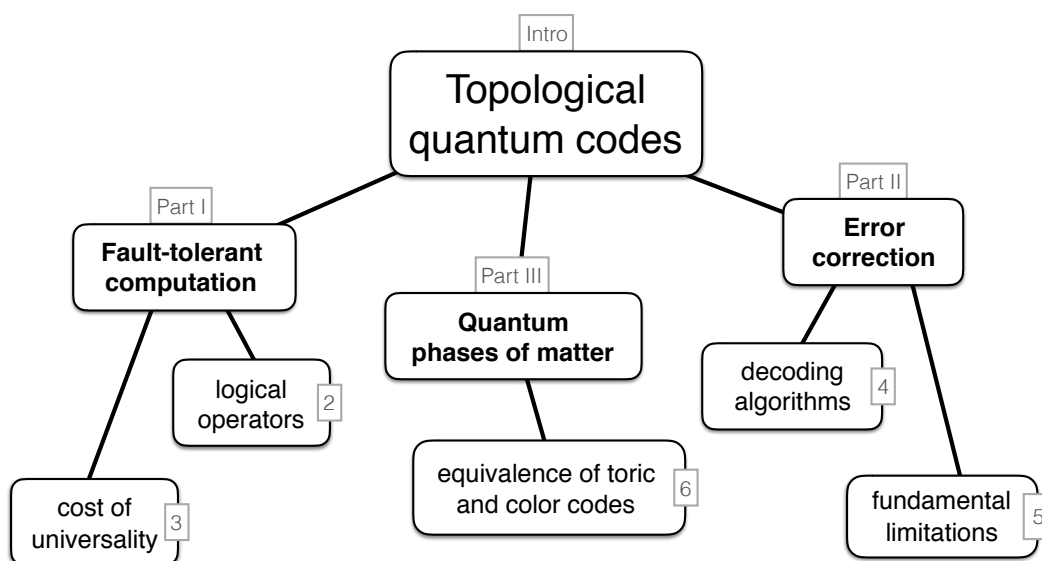


Figure 1.1: Structure of the thesis

Fault-tolerant computation

We investigate how one can perform fault-tolerant computation with topological codes. In particular, we provide methods of implementing logical operators in a very simple fault-tolerant way. We also analyze the resource overhead of achieving a universal gate set.

Error correction

We would like to understand how to reliably protect information against arbitrary noise by encoding it into topological quantum codes. We explore fundamental limitations on error-correction capabilities of topological codes and develop algorithms to perform error correction. In order to do that, we study phase transitions in new statistical-mechanical models and propose local decoders based on cellular automata.

Quantum phases of matter

We focus on a class of systems related to topological codes and described by stabilizer Hamiltonians. We prove the equivalence of two well-known models: the toric and color codes in $D \geq 2$ dimensions. Furthermore, we describe tools to study models with non-commuting Hamiltonians and try to understand what ingredients, such as symmetries, are needed to achieve error suppression.

This thesis might serve as a small step toward understanding the fundamental principles behind error correction, which eventually could help with engineering many-body systems suited for fault-tolerant quantum computation. We would like to emphasize that ideas from quantum error correction go beyond quantum computation and seem to permeate many fields of physics including condensed matter and quantum gravity. For instance, it was recently noticed that quantum error-correcting codes can provide new insight into the bulk/boundary correspondence [ADH15; Pas+15]. We believe that techniques and methods used in quantum error correction will soon become a standard toolset used by any physicist, and thus exploring them is an important and fascinating scientific quest.

In the rest of this chapter, we provide a brief introduction to topological stabilizer codes, which are the main focus of this thesis. We hope that the next section is understandable for everyone even without extensive knowledge of quantum mechanics. We also provide a concise summary of the results and contributions presented in the subsequent chapters of this thesis.

1.1 Brief introduction to topological stabilizer codes

Classical and quantum codes

Classical and quantum computation can suffer from noise due to uncontrolled interactions with the environment. In order to protect information from errors one can encode it using error-correcting codes [Sha01; MS77]. The simplest example of a code is the classical repetition code — one encodes a logical bit of information, either 0 or 1, into multiple physical bits, each either zero or one:

$$0 \rightarrow 00 \dots 0, \quad 1 \rightarrow 11 \dots 1. \quad (1.1)$$

Then, provided that a fewer than half of the physical bits are corrupted, one can still recover the encoded information by majority vote.

In order to store one quantum bit of information — a qubit, which is an arbitrary superposition of two quantum states $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ — we cannot immediately use an idea similar to the repetition code. Namely, due to the no-cloning theorem [WZ82], an arbitrary quantum state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ cannot be copied, where α and β are complex numbers satisfying $|\alpha|^2 + |\beta|^2 = 1$. Also, a quantum version of the repetition code

$$|0\rangle \rightarrow |00 \dots 0\rangle \quad |1\rangle \rightarrow |11 \dots 1\rangle \quad (1.2)$$

would protect the encoded qubit against the bit-flip Pauli $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ error

$$|0\rangle \xrightarrow{X} |1\rangle \quad |1\rangle \xrightarrow{X} |0\rangle \quad (1.3)$$

but would not prevent the other type of error, the phase-flip Pauli $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ error

$$|0\rangle \xrightarrow{Z} |0\rangle \quad |1\rangle \xrightarrow{Z} -|1\rangle. \quad (1.4)$$

Fortunately, there exist quantum error correcting codes that reliably protect encoded qubits against the bit- and phase-flip errors.

One of the simplest examples of a quantum code is the nine qubit code by Shor [Sho95]

$$|0\rangle \rightarrow \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle), \quad (1.5)$$

$$|1\rangle \rightarrow \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle), \quad (1.6)$$

which protects the encoded logical qubit against any single-qubit Pauli X , Z and $Y = iXZ$ errors. In fact, the nine qubit code allows for detection and correction of any single-qubit error, which can be an arbitrary CPTP map applied to any of the physical qubits. It is possible to extend Shor's construction and find quantum codes which can protect encoded information against more errors. In particular, we now focus on a systematic construction of a class of quantum error-correcting codes, introduced by Gottesman [Got96; Cal+97], called stabilizer codes.

Stabilizer codes

Stabilizer codes are quantum codes which are at the heart of many fault-tolerant quantum computation schemes. A stabilizer code is specified by the stabilizer group \mathcal{S} , which is an Abelian subgroup of the Pauli group $\mathcal{P}_n = \{I, X, Y, Z\}^{\otimes n}$ generated by tensor products of Pauli operators on n qubits. The code space associated with the stabilizer group \mathcal{S} is spanned by $+1$ eigenvectors of stabilizers $S \in \mathcal{S}$. For the code space to be non-trivial we require $-I \notin \mathcal{S}$. Operators which transform encoded states of the stabilizer code in the same way as Pauli operators would transform unencoded states are called logical Pauli operators. They can be found as the elements of the normalizer $\mathcal{N} = \{P \in \mathcal{P}_n | PSP^\dagger \in \mathcal{S}\}$ of the stabilizer group \mathcal{S} in the Pauli group \mathcal{P}_n . Two logical operators are equivalent if they differ by some stabilizer.

To diagnose errors affecting a stabilizer code we measure generators of the stabilizer group. The presence of errors is signalled by any non-trivial syndrome, which is the set of all stabilizers returning -1 measurement outcomes. Since two errors which differ by some stabilizer have the same syndrome, we cannot distinguish them by stabilizer measurements and thus treat them as equivalent. We can always perform error correction with stabilizer codes by choosing some Pauli operator consistent with the observed syndrome, which brings the encoded state back to the code space. However, the recovery might fail if the returned state in the code space is different from the original one. In other words, the recovery fails if and only if it implements any non-trivial logical Pauli operator on the code space, i.e. an element of the normalizer \mathcal{N} of \mathcal{S} that is not in \mathcal{S} . This motivates the definition of a notion of the distance of a quantum code to be the minimum weight of any non-trivial logical Pauli operator

$$d = \min_{P \in \mathcal{N} \setminus \mathcal{S}} |P|, \quad (1.7)$$

where $|P|$ denotes the weight of an operator $P \in \mathcal{P}_n$, i.e. the number of terms in the tensor product not equal to identity. The distance is related to the error-correcting

capabilities of a code. Namely, a code with distance d can correct arbitrary errors on any $\lfloor d/2 \rfloor$ qubits.

We illustrate notions from this section with the nine qubit code. The stabilizer group \mathcal{S} of the nine qubit code is generated by the following stabilizer generators

$$\mathcal{S} = \langle Z_1 Z_2, Z_2 Z_3, Z_4 Z_5, Z_5 Z_6, Z_7 Z_8, Z_8 Z_9, X_1 X_2 X_3 X_4 X_5 X_6, X_4 X_5 X_6 X_7 X_8 X_9 \rangle, \quad (1.8)$$

where X_i and Z_i denote Pauli X and Z operators acting on the qubit i . Two logical Pauli X and Z operators can be represented as

$$\bar{X} = X_1 X_2 X_3 \quad \bar{Z} = Z_1 Z_4 Z_7. \quad (1.9)$$

One can verify that the distance of the nine qubit code is $d = 3$.

Decoding of stabilizer codes

The problem of finding a suitable correction of errors in a stabilizer code, based on the observed syndrome, is called decoding. In general, optimal decoding, which reduces to finding the most probable equivalence class of errors for the observed syndrome, is computationally intractable [IP15]. However, there exist efficient methods for the decoding of some codes, for instance by finding the most probable error with the given syndrome [Den+02]. Also, the problem of decoding can be simplified for a class of stabilizer codes, called CSS codes [CS96; Ste96b], whose stabilizer group is generated by independent X - and Z -type stabilizer generators. Namely, the decoding of CSS codes can be separated into independent correction of Pauli X and Z errors.

We now consider the problem of decoding for a family of codes with growing distance. Since error correction might fail, we can define an effective error rate as the probability of unsuccessful decoding for a given code and physical error rate. We say that a code family has a threshold p_c if the effective error rate goes to zero in the limit of infinite distance for any physical error rate p smaller than p_c . Intuitively, the threshold p_c describes the maximum error rate that the code family can tolerate. We would like to emphasize that the property of non-zero threshold is exhibited only by certain families of codes, such as topological codes.

Topological stabilizer codes

Topological stabilizer codes [Kit03; BK98; FM01; BM06] are a class of stabilizer codes with geometrically local generators. In other words, physical qubits can

be placed on a manifold, for instance a torus, so that the code only has stabilizer generators with local support, see Fig. 1.2 for illustration. We further require that the number of qubits in the support of any stabilizer generator as well as the number of stabilizer generators supported on any given qubit are bounded. Topological codes protect information against local errors by encoding it into global degrees of freedom associated with the topology of the manifold. At the same time, topological codes require only geometrically local syndrome extraction and thus seem well suited for experimental implementations, for which locality is an important constraint.

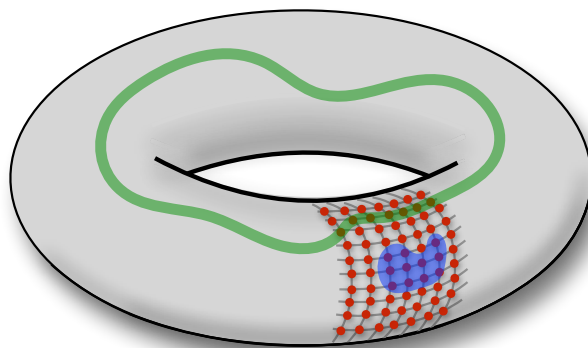


Figure 1.2: An illustration of a topological stabilizer code with qubits (red dots) placed on a torus. Stabilizer generators of the code (shaded in blue) are geometrically local, whereas non-trivial logical operators (shaded in green) are supported on non-contractible regions. Code properties, such as the number of logical operators, are related to the topology of the manifold.

The locality restriction that defines topological codes comes at a price. For instance, for any D -dimensional topological code where $D \geq 2$, the distance d and the number of encoded logical qubits k are related to the number of physical qubits n via the scaling

$$kd^{\frac{2}{D-1}} \leq O(n), \quad (1.10)$$

as shown by [BHM10]. At first, the scaling of the distance d with the total number of qubits n might suggest that topological codes do not provide good protection against high-weight errors. It turns out, however, that the contrary is true: topological codes can correct most of the likely errors and tend to have high threshold [Den+02; KBM09]. There is also a restriction on the computational power with topological codes in D dimensions if one requires quantum circuits to be local. Namely, for any topological code the group of logical gates implemented via local quantum circuits of constant depth must be contained in the D^{th} level of the Clifford hierarchy C_D ,

where $C_1 = \mathcal{P}_n$ and the D^{th} level is defined recursively as the set of all unitaries which map the Pauli group \mathcal{P}_n onto unitaries in the $(D - 1)$ -level [ZCC11; EK09; BK13; PY15; JKY18]. In particular, this restriction applies to transversal logical gates, i.e. code-preserving operations composed of separate unitaries applied to each physical qubit.

1.2 Thesis organization and contributions

The thesis is divided into three parts, which respectively focus on fault-tolerant computation, error correction and quantum phases of matter. In Chapters 2 and 3 we discuss fault-tolerant implementation of logical gates with the color code and the associated resource overhead. In Chapters 4 and 5 we analyze how to decode the toric and color codes and provide fundamental limitations on error-correcting capabilities of the latter. In Chapter 6 we prove that two models described by stabilizer Hamiltonians associated with the toric and color codes are equivalent in $D \geq 2$ dimensions. In the rest of this section we describe the main contributions of this thesis using more technical language.

Chapter 2 — Universal transversal gates with color codes: A simplified approach

In this chapter we provide an accessible introduction to color codes in two or more dimensions. We provide an explicit construction of a family of color codes in arbitrary dimensions and describe some of their crucial properties. Within this framework, we explicitly show how to transversally implement the generalized phase gate $R_n = \text{diag}(1, e^{2\pi i/2^n})$. We describe how to implement the Hadamard gate H fault tolerantly using code switching. In three dimensions, this yields, together with the transversal controlled-not (CNOT), a fault-tolerant universal gate set $\{H, \text{CNOT}, R_3\}$ without state distillation.

This chapter is based on [KB15].

Chapter 3 — The cost of universality in surface and color codes: State distillation versus code switching

In this chapter, we compare two methods of achieving a fault-tolerant universal gate set with topological color codes: state distillation and code switching. In our comparison, we focus on the qubit overhead of implementing a non-Clifford gate, the $T = \text{diag}(1, e^{2\pi i/8})$ gate. For state distillation, we use the 15-qubit Reed Muller code to distill logical T-states encoded into two-dimensional triangular color codes.

In the case of code switching, we transfer the logical state between two- and three-dimensional color codes and perform a transverse logical T gate in the 3D code. Our results indicate that the code switching technique offers no advantage over state distillation.

This chapter is based on [BKS18].

Chapter 4 — Local decoders for topological toric and color codes in any dimensions

In this chapter, we address a question of performing error correction in topological stabilizer codes with a local update rule. First, we propose the Sweep Rule, which is a generalization of Toom’s rule. The Sweep Rule shrinks k -dimensional domain walls on any d -dimensional locally Euclidean lattice built of d -simplices, where $d \geq 2$ and $k \in \{1, \dots, d - 1\}$. Based on the Sweep Rule, we design a local decoder for the d -dimensional toric code without point-like excitations. Then, we construct a new class of local decoders of the color code in $d \geq 2$ dimensions. The two key components of our construction include a local reduction of the problem of color code decoding to that of the toric code and using the Sweep Rule to decode the resulting toric code. Finally, we obtain a lower bound on the performance of decoders for the toric and color codes based on the Sweep Rule. Our results provide a rigorous proof of a non-zero threshold, and thus clarify the success of Toom’s rule as an error-correction method for topological stabilizer codes.

This chapter is based on [KDP18].

Chapter 5 — Three-dimensional color code thresholds via statistical-mechanical mapping

This chapter is devoted to finding the storage threshold error rates for bit-flip and phase-flip noise in the three-dimensional color code on the body-centered cubic lattice, assuming perfect syndrome measurements. In particular, by exploiting a connection between error correction and statistical mechanics, we estimate the threshold for 1D string-like and 2D sheet-like logical operators to be $p_{3\text{DCC}}^{(1)} \simeq 1.9\%$ and $p_{3\text{DCC}}^{(2)} \simeq 27.5\%$. We obtain these results by using parallel tempering Monte Carlo simulations to study the disorder-temperature phase diagrams of two new 3D statistical-mechanical models: the 4- and 6-body random coupling Ising models. Our estimates provide thresholds for optimal decoding of bit-flip and phase-flip noise for the 3D color code and thus can be used to benchmark performance of the color code decoders we describe in Chapter 4.

This chapter is based on [Kub+18].

Chapter 6 — Unfolding the color code

In this chapter we show that the color code on a d -dimensional closed manifold is equivalent to multiple decoupled copies of the d -dimensional toric code up to local unitary transformations and adding or removing ancilla qubits. This result not only generalizes the proven equivalence for the $d = 2$ case, but also provides an explicit recipe for how to decouple independent components of the color code, highlighting the importance of colorability in the construction of the code. Moreover, for the d -dimensional color code with $d + 1$ boundaries of $d + 1$ distinct colors, we find that the code is equivalent to multiple copies of the d -dimensional toric code which are attached along a $(d - 1)$ -dimensional boundary. In particular, for $d = 2$, we show that the (triangular) color code with boundaries is equivalent to the (folded) toric code with boundaries. We also find that the d -dimensional toric code admits logical non-Pauli gates from the d^{th} level of the Clifford hierarchy, and thus saturates the bound by Bravyi and König. In particular, we show that the logical d -qubit control- Z gate can be fault-tolerantly implemented on the stack of d copies of the toric code by a local unitary transformation.

This chapter is based on [KYP15].

Chapter 2

UNIVERSAL TRANSVERSAL GATES WITH COLOR CODES: A SIMPLIFIED APPROACH

To build a fully functioning quantum computer, it is necessary to encode quantum information to protect it from noise. In physical systems, one expects noise to act locally. Therefore, *topological codes* [Kit03; LW05; BM06; Bon+10], which naturally protect against local errors, represent our best hope for storing quantum information. However, a quantum computer must also be capable of processing this information. This motivates the search for topological codes allowing the implementation of a set of gates which (i) can operate in the presence of typical noise without corrupting the stored information, and (ii) can perform any computation on the encoded information. A theoretical framework has been developed around these ideas — a gate which is *fault-tolerant* does not propagate typical errors into uncorrectable errors [Sho96; Pre98], and therefore satisfies (i). A set of gates which is *universal* can generate any unitary on the code space with arbitrary precision [Kit97; NC10], and therefore satisfies (ii).

The known methods of implementing a universal, fault-tolerant gate set in topological codes typically require an enormous amount of overhead. For instance, magic state distillation [BK05] with the two-dimensional toric code requires many additional ancilla qubits [Fow+12], whereas computing by braiding non-abelian anyons [Kit03; Nay+08] requires additional time to move anyons around macroscopic loops [Bec+01]. These forms of overhead can make quantum processing orders of magnitude less efficient than storage alone in topological codes. This may render such approaches impractical given the experimental difficulty of scaling up quantum hardware [Fow+12; DS13; Wec+14]. In this chapter we focus on a new construction by Bombín [Bom15a], for a universal fault-tolerant gate set with topological color codes. This seems not to involve significant additional overhead, however a lattice of at least three dimensions is required, limiting the construction's practicality for reasons of architecture.

Following Bombín's construction, we use the simplest form of fault-tolerant gate — the *transversal gate*, which is a code-space preserving unitary composed of separate unitaries applied to each physical qubit. However, according to a no-go theorem by

Eastin and Knill [EK09], for any code which protects against arbitrary single-qubit errors, the set of transversal gates forms a finite group and therefore cannot be universal. Some recent approaches to circumvent this no-go theorem in order to implement a universal gate set with transversal gates have been put forward [JL14; PR13; ADP14].

In Ref. [Bom15a], Bombín applies the approach of *gauge fixing* [PR13; ADP14] to color codes in a d -dimensional lattice. Color codes were first introduced in two dimensions by Bombín and Martin-Delgado in Ref. [BM06]. They are *topological stabilizer codes* [Got96; Cal+97; Kit03; BK13], meaning they are defined on a lattice and have macroscopic distance together with geometrically local stabilizer generators. The main new conceptual contribution in Ref. [Bom15a] is that gauge fixing allows one to fault-tolerantly switch between a (stabilizer) color code on a d -dimensional lattice, in which CNOT and $R_d = \text{diag}\left(1, \exp\left(\frac{2\pi i}{2^d}\right)\right)$ are transversal, and a different (subsystem) color code on the same lattice, in which H is transversal. Critically, for $d \geq 3$, $\{H, \text{CNOT}, R_d\}$ forms a universal gate set. To the author's knowledge, this represents the first construction using gauge fixing to achieve a universal gate set in a topological code.

In Ref. [Bom15a], Bombín argues that for every $d \geq 2$, there exists a d -dimensional color code with a transversal implementation of $R_d \in \mathcal{P}_d \setminus \mathcal{P}_{d-1}$, which is the main technical contribution therein. At the same time, for any topological stabilizer code, Bravyi and König [BK13] showed that the group of logical gates implemented transversally must be contained in \mathcal{P}_d , the d^{th} level of the Clifford hierarchy¹ [GC99]. These results have been extended beyond the stabilizer code setting [PY15; Bev+16]. Color codes are the only family of topological stabilizer codes currently known to saturate the Bravyi-König classification in every dimension $d \geq 2$.

In this chapter, we provide a simplified yet rigorous presentation of the ideas in Ref. [Bom15a]. The organization is as follows. First, to build some intuition, we introduce color codes in two dimensions in Section 2.1. We explain how to transversally implement the gate set $\{H, \text{CNOT}, R_2\}$, which generates the Clifford group. Then, we describe the generalization of color codes to d dimensions in Section 2.2. Next, in Section 2.3 we discuss transversal gates in those codes with an emphasis on the phase gate R_n , and show that in certain d -dimensional color codes R_d is transversal. Our construction utilizes the bipartite property of the lattice

¹The Clifford hierarchy is defined sequentially for $j > 1$ according to $\mathcal{P}_j = \{\text{unitary } U | UPU^\dagger \in \mathcal{P}_{j-1} \forall P \in \mathcal{P}_1\}$, with \mathcal{P}_1 representing the Pauli group. Note that \mathcal{P}_2 is the well-known Clifford group.

allowing for a simpler verification than in Ref. [Bom15a]. Finally, in Section 2.4 we explain how to switch between color codes fault-tolerantly using the technique of gauge fixing. In particular, this allows one to implement a fault-tolerant universal gate set $\{H, \text{CNOT}, R_3\}$ in a color code in three dimensions.

2.1 Color code in two dimensions

In this section, we give an explicit construction of a stabilizer color code in two dimensions [BM06; Bom13]. We consider a 3-valent lattice formed as a tiling of a sphere, such that faces of the lattice are colored with three colors, where neighboring faces have distinct colors. Qubits are placed at the vertices of this lattice. To define a color code on this lattice, we associate an X - and a Z -type stabilizer generator with every face. This code encodes no logical qubits. A new code, which encodes a single logical qubit, can be formed through the removal of a single physical qubit. We describe the transversal implementation of the logical gates $\overline{\text{CNOT}}$, \overline{H} and \overline{R}_2 in the new code².

Color code with no encoded qubits

Color codes in two dimensions are CSS stabilizer codes [Got96; Cal+97], and are therefore specified by their stabilizer group \mathcal{S} generated by X - and Z -type stabilizer generators. The code space is the simultaneous $+1$ eigenspace of every stabilizer generator. In the construction, we use a two-dimensional lattice \mathcal{L}_0^* , obtained from a tiling of the 2-sphere, and satisfying the following requirements

- valence — every vertex is 3-valent, meaning it belongs to exactly 3 edges,
- colorability — faces can be colored with 3 colors: red, green and blue, such that every two faces sharing an edge have different colors.

An example of such a tiling of the 2-sphere is presented in Fig. 2.1(a). From these properties alone, one can show that the total number of vertices in \mathcal{L}_0^* is even. To see this, note that the Euler characteristic gives $V - E + F = 2$, where V , E and F denote the number of vertices, edges and faces in \mathcal{L}_0^* , respectively. Since every vertex is 3-valent, we obtain $E = \frac{3}{2}V$, and then $V = 2(F - 2)$, which is even.

At every vertex in \mathcal{L}_0^* we place a qubit. We refer to the set of all qubits by Q , whereas by $\mathcal{Q}(\Pi) \subset Q$ we denote the set of vertices of a face Π . Alternatively,

²We use a bar to indicate action on logical code space. The absence of a bar indicates action on physical qubits.

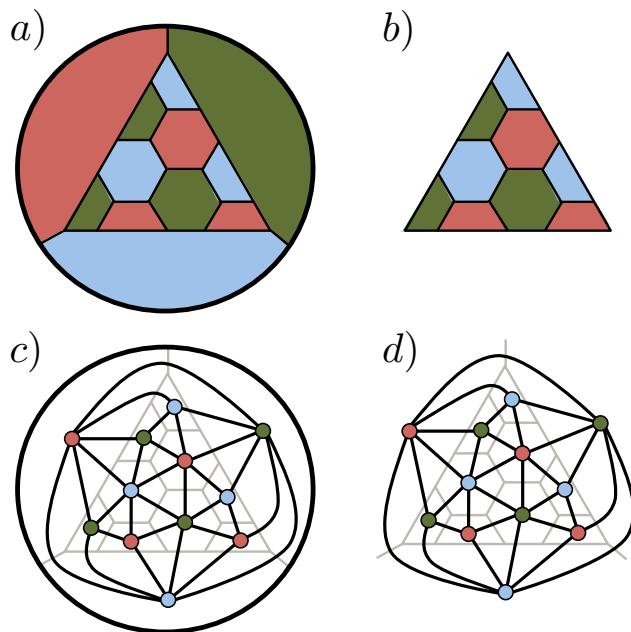


Figure 2.1: Construction of color codes in two dimensions. In (a) and (b), qubits are placed at vertices, and X - and Z -type stabilizer generators are associated with faces. In (c) and (d) (the dual picture), qubits are placed on faces, and X - and Z -type stabilizer generators are associated with vertices. (a) Take a lattice \mathcal{L}_0^* , which is a tiling of the 2-sphere with 3-colorable faces and 3-valent vertices. The surrounding circle is identified with a vertex v . The color code on \mathcal{L}_0^* encodes no logical qubits. (b) To obtain \mathcal{L}^* , remove from \mathcal{L}_0^* the vertex v , together with the three edges and three faces containing it. The color code on \mathcal{L}^* encodes one logical qubit. (c) (Dual) lattice \mathcal{L}_0 is obtained from \mathcal{L}_0^* by replacing faces, edges and vertices by vertices, edges and faces, respectively. All faces are triangles, and the vertices are 3-colorable. The color code on \mathcal{L}_0 encodes no logical qubits. (d) Lattice \mathcal{L} formed from \mathcal{L}_0 by removing a single face. No stabilizer generators are associated with those vertices belonging to the boundary of the removed face. The color code on \mathcal{L} encodes one logical qubit.

we can think of $Q(\Pi)$ as the set of qubits belonging to Π . To define the color code, it is sufficient to specify X - and Z -type stabilizer generators. For every face Π , we define an X -type stabilizer generator $X(\Pi)$ to be a tensor product of Pauli X operators supported on qubits $Q(\Pi)$, similarly for Z -type generators. Then, the stabilizer group \mathcal{S} is generated by

$$\mathcal{S} = \langle X(\Pi), Z(\Pi), \text{ for every face } \Pi \text{ in } \mathcal{L}_0^* \rangle. \quad (2.1)$$

To prove that this specifies a well-defined stabilizer code, we must verify that all the generators of \mathcal{S} commute. It is sufficient to check that for any two faces Π_1 and

Π_2 in \mathcal{L}_0^* , $X(\Pi_1)$ and $Z(\Pi_2)$ commute. First take the case $\Pi_1 \neq \Pi_2$. If Π_1 and Π_2 share no vertices, then $X(\Pi_1)$ and $Z(\Pi_2)$ trivially commute. If they share a vertex, then by 3-valence, they also share an edge. Moreover, due to 3-colorability, Π_1 and Π_2 cannot share two consecutive edges, and thus their intersection has to contain an even number of vertices,

$$|\mathcal{Q}(\Pi_1) \cap \mathcal{Q}(\Pi_2)| \equiv 0 \pmod{2}. \quad (2.2)$$

For the case $\Pi_1 = \Pi_2 = \Pi$, due to 3-colorability and 3-valence, the number of vertices belonging to a face Π is even,

$$|\mathcal{Q}(\Pi)| \equiv 0 \pmod{2}. \quad (2.3)$$

Therefore, we obtain commutation of $X(\Pi_1)$ and $Z(\Pi_2)$ for arbitrary Π_1 and Π_2 .

From the construction of the lattice, one obtains that each vertex belongs to exactly three faces, colored with three different colors. Thus, one can express the set of vertices in \mathcal{L}_0^* as the disjoint union³ of vertices belonging to red faces, and similarly for green and blue [BM06; Bom13], namely

$$\mathcal{Q} = \bigsqcup_{\Pi_R} \mathcal{Q}(\Pi_R) = \bigsqcup_{\Pi_G} \mathcal{Q}(\Pi_G) = \bigsqcup_{\Pi_B} \mathcal{Q}(\Pi_B), \quad (2.4)$$

where $\{\Pi_R\}$, $\{\Pi_G\}$ and $\{\Pi_B\}$ are the sets of all red, green and blue faces, respectively. This implies that not all the stabilizer generators we have defined are independent

$$\prod_{\Pi_R} X(\Pi_R) = \prod_{\Pi_G} X(\Pi_G) = \prod_{\Pi_B} X(\Pi_B), \quad (2.5)$$

$$\prod_{\Pi_R} Z(\Pi_R) = \prod_{\Pi_G} Z(\Pi_G) = \prod_{\Pi_B} Z(\Pi_B). \quad (2.6)$$

In fact, these are the only conditions [BM07a; Bom13] which relate the stabilizer generators to one another.

We can now verify that the color code which we have defined on the lattice \mathcal{L}_0^* encodes no logical qubits. As before, using the Euler characteristic we obtain $F - 2 = E - V$, and from 3-valence of vertices — $E = \frac{3}{2}V$. We have placed physical qubits at vertices, thus $|Q| = V$. There are $2F - 4$ independent stabilizer generators, since there are two stabilizer generators for every face and four conditions (2.5) and (2.6). The number of logical qubits is equal to the number of physical qubits minus the number of independent stabilizer generators, and we obtain

$$|Q| - (2F - 4) = V - 2(E - V) = 0. \quad (2.7)$$

³We use the *disjoint union* $A \sqcup B$ in place of the union $A \cup B$ of two sets A and B when their intersection is empty, $A \cap B = \emptyset$.

Color code with one logical qubit

To obtain a color code with one encoded logical qubit, we can remove one vertex from the lattice \mathcal{L}_0^* , together with three edges and three faces it belongs to, obtaining a new lattice \mathcal{L}^* (see Fig. 2.1b). By removing one vertex, we also discard six stabilizer generators associated with the removed faces, and thus the stabilizer generators no longer have to satisfy (2.5) and (2.6). One can check that this new code encodes one logical qubit, since there is one qubit more than independent stabilizer generators. By removing more vertices, one could encode more logical qubits, but we will not analyze that case. Note that the total number of qubits in \mathcal{L}^* is odd, $|Q| \equiv 1 \pmod{2}$, which plays an important role in our considerations.

On physical grounds, it is of interest to consider stabilizer codes with stabilizer generators which are low-weight and geometrically local. In the construction we have presented, this can be achieved if each face in the lattice \mathcal{L}^* is geometrically local and contains a small number of vertices, as in Fig. 2.1b. It can be shown that following this construction, the resulting color code has macroscopic distance [BM06], and therefore is a topological stabilizer code.

Later, when we discuss color codes in d dimensions, we follow a similar construction. We briefly outline the procedure here, deferring detailed discussion to Section 2.2. We start with a tiling of a d -sphere, place qubits at vertices and define (gauge group) generators to be supported on suitable cells. Then, we remove one vertex and all the cells containing it. In particular, we discard generators supported on the removed cells. Such a code encodes only one logical qubit [BM07a].

Transversal gates

In this chapter we consider stabilizer codes encoding only one logical qubit, with the stabilizer group \mathcal{S} . In this setting, a *transversal gate* \overline{U} on a single logical qubit is implemented as a tensor product of single physical qubit unitaries $U_1 \otimes \dots \otimes U_{|Q|}$, which preserves the code space. On the other hand, a logical gate on two logical qubits requires two copies of the code, in which case we say that the *overall* code space is the +1 eigenspace of the elements in $\mathcal{S} \otimes \mathcal{S}$. A transversal gate on two logical qubits is implemented as a tensor product of two qubit gates on pairs of corresponding qubits in both copies of the code, which preserves the overall code space. Observe that transversal gates are fault-tolerant since they do not spread errors within each copy of the code.

We now show that in the two-dimensional color code described in the previous

subsection, one can transversally implement the gate set $\{\overline{H}, \overline{\text{CNOT}}, \overline{R}_2\}$, which generates the (non-universal) Clifford group. The Clifford group, combined with computational basis state preparation and measurement, can be simulated efficiently on a classical computer [Got98; AG04]. For each gate, \overline{H} , $\overline{\text{CNOT}}$ and \overline{R}_2 , we verify that a particular transversal unitary implements the logical gate by showing that it has the correct action under conjugation on generators of the logical Pauli group, and that the stabilizer group is preserved⁴.

The two-dimensional color code is a CSS stabilizer code encoding a single logical qubit with logical Pauli operators $\overline{X} = X(Q)$ and $\overline{Z} = Z(Q)$. In addition it is a *self-dual CSS stabilizer code* — a code with the same support for X - and Z -type stabilizer group elements (for each face, there is an X - and a Z -type generator). This implies that the logical Hadamard gate can be implemented transversally, as under conjugation by $H(Q)$, $\overline{X} \mapsto H(Q)X(Q)H(Q)^\dagger = \overline{Z}$ and similarly $\overline{Z} \mapsto \overline{X}$. Moreover, $X(\Pi) \mapsto Z(\Pi)$, $Z(\Pi) \mapsto X(\Pi)$, and thus \mathcal{S} is preserved.

The logical gate $\overline{\text{CNOT}}$ can be implemented transversally between two identical copies of this color code by applying a physical gate CNOT to every pair of corresponding qubits in the first and the second copy. This can be verified by checking that under conjugation by $\overline{\text{CNOT}}$, $\overline{X} \overline{I} \mapsto \overline{X} \overline{X}$, $\overline{I} \overline{X} \mapsto \overline{I} \overline{X}$, $\overline{Z} \overline{I} \mapsto \overline{Z} \overline{I}$, $\overline{I} \overline{Z} \mapsto \overline{Z} \overline{Z}$ and $\mathcal{S} \otimes \mathcal{S}$ is preserved⁵.

To show that \overline{R}_2 can be implemented transversally, we use the fact that the set of vertices in \mathcal{L}^* is bipartite (see Fig. 2.2(a)). In other words, Q can be split into two subsets, T and $T^c := Q \setminus T$, such that vertices in T are connected only to vertices in T^c and vice versa. To prove this, first note that every face in \mathcal{L}_0^* has an even number of edges. Moreover, every cycle in \mathcal{L}_0^* (as a tiling of the 2-sphere) is contractible. This implies that every cycle in \mathcal{L}_0^* is a boundary of faces and is therefore even. Using the following lemma

Lemma 1 (Graph Bipartition) *A graph containing only even cycles is bipartite [Wil96].*

we see that \mathcal{L}_0^* must be bipartite, and so is the lattice \mathcal{L}^* due to its construction from \mathcal{L}_0^* .

⁴Preservation of the stabilizer group is a sufficient (but not necessary) condition that implies preservation of the code.

⁵Notice that generators of $\mathcal{S} \otimes \mathcal{S}$ are mapped under conjugation to a different generators, namely $X(\Pi) \otimes I(\Pi) \mapsto X(\Pi) \otimes X(\Pi)$, $Z(\Pi) \otimes I(\Pi) \mapsto I(\Pi) \otimes Z(\Pi)$, $I(\Pi) \otimes X(\Pi) \mapsto I(\Pi) \otimes X(\Pi)$ and $I(\Pi) \otimes Z(\Pi) \mapsto Z(\Pi) \otimes Z(\Pi)$.

Now, we can show that $R = R_2^k(T)R_2^{-k}(T^c)$ implements \bar{R}_2 , for some choice of integer k . We use the relations $R_2XR_2^\dagger = iXZ$ and $R_2ZR_2^\dagger = Z$. Since $|Q| \equiv 1 \pmod{2}$, then $|T| - |T^c| = 2|T| - |Q| \equiv \pm 1 \pmod{4}$, and picking $k = |T| - |T^c| \pmod{4}$ ensures that $k(|T| - |T^c|) \equiv 1 \pmod{4}$. With this choice of k , the action by conjugation of $R = R_2^k(T)R_2^{-k}(T^c)$ on the logical \bar{X} and \bar{Z} is

$$R\bar{X}R^\dagger = i^{k(|T|-|T^c|)}\bar{X}\bar{Z} = i\bar{X}\bar{Z}, \quad (2.8)$$

$$R\bar{Z}R^\dagger = \bar{Z}. \quad (2.9)$$

Furthermore, as every face Π in the lattice \mathcal{L}^* has an equal number of vertices in T and T^c , under the action of R the stabilizer generators $X(\Pi)$ and $Z(\Pi)$ become:

$$RX(\Pi)R^\dagger = i^{k(|T \cap \Pi| - |T^c \cap \Pi|)}X(\Pi)Z(\Pi) \quad (2.10)$$

$$= X(\Pi)Z(\Pi) \in \mathcal{S}, \quad (2.11)$$

$$RZ(\Pi)R^\dagger = Z(\Pi), \quad (2.12)$$

implying that the stabilizer group \mathcal{S} is preserved. This completes the verification that R implements \bar{R}_2 .

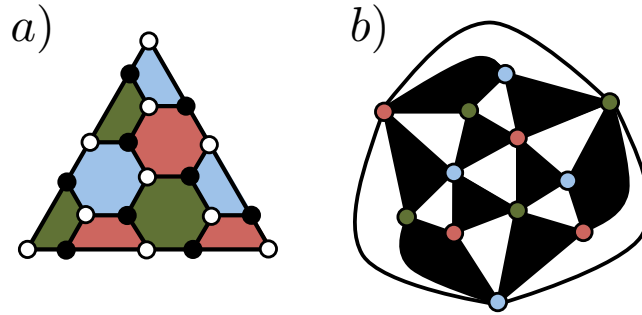


Figure 2.2: (a) The set of vertices of \mathcal{L}^* , the lattice used to define the color code, is bipartite — it can be split into two subsets: T (hollow circles), and its complement T^c (filled circles). Vertices in T are only connected to vertices in T^c and vice versa. The logical gate \bar{R}_2 can be implemented by applying R_2^k to qubits in T , and R_2^{-k} to qubits in T^c , where $k \equiv |T| - |T^c| \pmod{4}$. (b) The dual lattice \mathcal{L} . Faces are bipartite.

Dual lattice picture

We can alternatively express the construction of color codes in the dual lattice picture, which we use extensively in the later discussion for $d > 2$ dimensions. We use a two-dimensional (dual) lattice \mathcal{L}_0 , obtained from a tiling of the 2-sphere, and satisfying the following requirements

- all faces are triangles,
- vertices are 3-colorable, meaning two vertices belonging to the same edge are colored with different colors.

See Fig. 2.1(c) for a simple example. Note that these conditions are equivalent to the conditions of 3-valence of vertices and 3-colorability of faces required for the tiling \mathcal{L}_0^* of the 2-sphere, where lattices \mathcal{L}_0^* and \mathcal{L}_0 are dual to one another.

A qubit is placed on every face of \mathcal{L}_0 , and an X - and a Z -type stabilizer generator is associated with every vertex, meaning they are supported on qubits corresponding to faces containing that vertex. The resulting color code is exactly the same as that described in Section 2.1, and therefore has zero logical qubits. To encode a single logical qubit, one should remove a face from \mathcal{L}_0 , together with stabilizer generators associated with the vertices belonging to the removed face, see Fig. 2.1(d).

The bipartition of vertices in \mathcal{L}^* corresponds to a bipartition of faces in \mathcal{L} , meaning that faces can be split into two sets, T and its complement T^c , such that faces in T share an edge only with faces in T^c and vice-versa. See Fig. 2.2(b).

2.2 Color code in higher dimensions

Here we present a construction of color codes on d -dimensional lattices. In higher dimensions it is easier to describe the construction in the language of the dual lattice. The majority of this section is devoted to defining dual lattices satisfying certain conditions and analyzing their properties. The discussion is a generalization of that already presented for two dimensions. The basic idea of how to construct the dual lattice \mathcal{L} is to first tile a d -sphere with d -simplices to form a lattice \mathcal{L}_0 . We require that every vertex in \mathcal{L}_0 can be assigned one of $d + 1$ distinct colors and two vertices belonging to the same edge have different colors. The lattice \mathcal{L} , used to define the color code, is formed by removing one d -simplex from \mathcal{L}_0 .

Simplicial complexes and colorability

A d -simplex δ is a d -dimensional polytope which is a convex hull of its $d + 1$ affinely independent vertices v_0, v_1, \dots, v_d , namely

$$\delta = \left\{ \sum_{i=0}^d t_i v_i \mid 0 \leq t_i \wedge \sum_{i=0}^d t_i = 1 \right\}. \quad (2.13)$$

In particular, 0-simplices are vertices, 1-simplices are edges, 2-simplices are triangles, 3-simplices are tetrahedra and so on.

A convex hull of a subset of vertices of size $k + 1 \leq d + 1$ is a k -simplex σ , which we call a k -face of δ , and $\sigma \subset \delta$. For example, the faces of a 3-simplex (a tetrahedron) are: four 0-simplices, six 1-simplices, four 2-simplices and a single 3-simplex. More generally, δ contains $\binom{d+1}{k+1}$ k -faces, since every k -face is uniquely determined by the choice of $k + 1$ vertices spanning it. By $\Delta_k(\delta)$ we call the set of all k -faces of δ , namely

$$\Delta_k(\delta) = \{\sigma \subset \delta \mid \sigma \text{ is a } k\text{-simplex}\}. \quad (2.14)$$

Instead of having only one simplex, we can consider a collection of them. Moreover, we can create new objects, called simplicial complexes [Hat02], by gluing simplices along their proper faces of matching dimension. We restrict ourselves to simplicial complexes containing finitely many simplices. We will define a d -dimensional color code on a lattice \mathcal{L} obtained by gluing together d -simplices. The technical name for such a lattice is a homogeneous simplicial d -complex.

Although \mathcal{L} is formally a collection of simplices, by the same symbol we also denote the union of these simplices as a topological space. Notice that \mathcal{L} is a manifold with a boundary, which we can think of as being embedded in real space. We denote by $\partial\mathcal{L}$ the set of simplices belonging to the boundary of \mathcal{L} , where the boundary of \mathcal{L} is the set of points in the closure of \mathcal{L} not belonging to the interior of \mathcal{L} . Moreover, by $\Delta'_k(\mathcal{L})$ we understand a set of all k -simplices belonging to $\mathcal{L} \setminus \partial\mathcal{L}$. Note that $\Delta'_d(\mathcal{L}) = \Delta_d(\mathcal{L})$.

We say that a simplicial d -complex \mathcal{L} is $(d + 1)$ -colorable if we can introduce a function

$$\text{color} : \Delta_0(\mathcal{L}) \rightarrow \mathbb{Z}_{d+1}, \quad (2.15)$$

where $\mathbb{Z}_{d+1} = \{0, 1, \dots, d\}$ is a set of $d + 1$ colors, and two vertices belonging to the same edge have different colors. Moreover, by $\text{color}(\delta)$ we understand the set of colors assigned to all the vertices of a simplex δ , namely

$$\text{color}(\delta) = \bigsqcup_{v \in \Delta_0(\delta)} \text{color}(v). \quad (2.16)$$

An example of a 3-colorable, homogeneous, simplicial 2-complex is the lattice \mathcal{L} shown in Fig. 2.1(d). Note in particular that it is composed of nineteen 2-simplices (triangles). The exact shape of objects in \mathcal{L} is not important due to its topological nature — the lattice is not rigid and can be smoothly deformed. In this example, $\Delta'_0(\mathcal{L})$ consists of the set of 9 vertices (the three vertices in the boundary are

excluded). $\Delta'_1(\mathcal{L})$ is the set of 27 edges, (the three edges in the boundary are excluded). $\Delta'_2(\mathcal{L})$ is the set of all 19 triangular faces.

Definition of color code

Here we define color codes on a d -dimensional lattice \mathcal{L} , which must satisfy the following conditions

Condition 1 \mathcal{L} is a homogeneous simplicial d -complex obtained as a triangulation of the interior of a d -simplex.

Condition 2 \mathcal{L} is $(d + 1)$ -colorable.

One can obtain such a lattice \mathcal{L} from any $(d + 1)$ -colorable tiling of the d -sphere with d -simplices, followed by the removal of one d -simplex. In $d = 2$ dimensions, this is precisely the procedure described in Section 2.1. An explicit construction of a family of lattices satisfying these conditions is outlined in the later part of this section.

Qubits are placed on each and every d -simplex of \mathcal{L} , and thus the set of all qubits Q is equal to $\Delta_d(\mathcal{L})$. This motivates the next definition, namely for a simplex $\delta \subset \mathcal{L} \setminus \partial\mathcal{L}$ we define

$$Q(\delta) = \{\sigma \in \Delta_d(\mathcal{L}) \mid \sigma \supset \delta\}. \quad (2.17)$$

In other words, $Q(\delta)$ can be thought of as the set of qubits placed on d -simplices containing δ . We say that qubits $Q(\delta)$ are supported on δ . By saying that an operator is supported on δ we mean that it is supported on the set $Q(\delta)$, for example $X(\delta) := X(Q(\delta))$.

A color code is a CSS subsystem code [Pou05; Bac06]. Recall that a CSS subsystem code is specified by its gauge group \mathcal{G} . Each X -type gauge group generator $X(G^x)$ consists of Pauli X operators applied to qubits G^x ; similarly for Z -type generators. The stabilizer group $\mathcal{S} \subset \mathcal{G}$ is the group generated by all Pauli operators $X(S^x)$ and $Z(S^z)$ contained in \mathcal{G} , which commute with every element of \mathcal{G} . Note that $-I \notin \mathcal{S}$. The codewords are $+1$ eigenvectors of all elements of \mathcal{S} .

We define a d -dimensional color code [Bom15a] on the lattice \mathcal{L} , where $d = \dim \mathcal{L}$, as the CSS subsystem code with X - and Z -type gauge generators supported on $(d - 2 - z)$ - and $(d - 2 - x)$ -simplices in \mathcal{L} ,

$$\mathcal{G} = \langle X(\delta), Z(\sigma) \mid \forall \delta \in \Delta'_{d-2-z}(\mathcal{L}), \sigma \in \Delta'_{d-2-x}(\mathcal{L}) \rangle, \quad (2.18)$$

where $x + z \leq d - 2$. The X - and Z -type generators of the stabilizer group \mathcal{S} are supported on x - and z -simplices, namely

$$\mathcal{S} = \langle X(\delta), Z(\sigma) \mid \forall \delta \in \Delta'_x(\mathcal{L}), \sigma \in \Delta'_z(\mathcal{L}) \rangle. \quad (2.19)$$

We refer to this code by $CC_{\mathcal{L}}(x, z)$. When context makes the lattice unambiguous, we sometimes use $CC_d(x, z)$ to emphasize the dimensionality of the lattice, $\dim \mathcal{L} = d$. Note that the generators of the gauge and stabilizer groups are supported on simplices which do not belong to $\partial \mathcal{L}$, the boundary of the lattice \mathcal{L} .

To illustrate the language introduced in this section, we revisit the two-dimensional color code described in Sections 2.1 and 2.1. We begin with the lattice \mathcal{L} shown in Fig. 2.1d. Qubits are placed on 2-simplices (triangular faces). Since $x + z \leq \dim \mathcal{L} - 2 = 0$, there is only one color code on the two-dimensional lattice \mathcal{L} , namely $CC_{\mathcal{L}}(0, 0)$, which is a stabilizer code. Stabilizer generators are associated with 0-simplices (vertices). Note that no stabilizer generators are assigned to the three vertices belonging to the boundary of \mathcal{L} .

Properties of the lattice

Here we present some properties of any $(d + 1)$ -colorable homogeneous simplicial d -complex \mathcal{L} . We use these properties to verify that $CC_{\mathcal{L}}(x, z)$ is a valid code, and later that there is a transversal implementation of \overline{R}_n . We start with the following two lemmas

Lemma 2 (Intersection) *Let δ and σ be two simplices in $\mathcal{L} \setminus \partial \mathcal{L}$. If $Q(\delta) \cap Q(\sigma) \neq \emptyset$, then $Q(\delta) \cap Q(\sigma) = Q(\tau)$, where τ is the smallest simplex containing both δ and σ .*

Proof: If $Q(\delta) \cap Q(\sigma) \neq \emptyset$, then there exists $\epsilon \in \Delta_d(\mathcal{L})$ such that $\epsilon \supset \delta, \sigma$. Let $C = \text{color}(\delta) \cup \text{color}(\sigma)$ and set τ to be the unique $(|C| - 1)$ -simplex in ϵ , colored with the set of colors C . Clearly, τ is the smallest simplex containing δ and σ , and $Q(\delta) \cap Q(\sigma) = Q(\tau)$. \square

Lemma 3 (Even Support) *Let δ be a k -simplex not belonging to the boundary of the lattice, $\delta \in \Delta'_k(\mathcal{L})$, with $0 \leq k < d$. Then*

$$|Q(\delta)| \equiv 0 \pmod{2}. \quad (2.20)$$

Before we prove the (Even Support) Lemma 3, we explain its consequences. For $CC_d(x, z)$ to be a subsystem code, the stabilizer generators have to commute with each other, as well as with the gauge group generators. Notice that for two arbitrary X - and Z -type stabilizer generators to commute, the intersection of their supports has to contain even number of elements. Let X - and Z -type stabilizer generators be supported on $\delta \subset \Delta'_x(\mathcal{L})$ and $\sigma \subset \Delta'_z(\mathcal{L})$, respectively. If the intersection $\mathcal{Q}(\delta) \cap \mathcal{Q}(\sigma)$ is non-empty, then due to the (Intersection) Lemma 2 there exists a simplex τ such that $\mathcal{Q}(\delta) \cap \mathcal{Q}(\sigma) = \mathcal{Q}(\tau)$. Moreover, since δ is spanned by $x + 1$ vertices and σ by $z + 1$ vertices, then τ is spanned by at most $x + z + 2 \leq d$ vertices. Thus, τ is a k -simplex with $k < d$, and the (Even Support) Lemma 3 applies, $|\mathcal{Q}(\delta) \cap \mathcal{Q}(\sigma)| = |\mathcal{Q}(\tau)| \equiv 0 \pmod{2}$, showing that $X(\delta)$ and $Z(\sigma)$ commute. The commutation of stabilizer generators with the gauge generators follows similarly.

We can obtain the (Even Support) Lemma 3 as a corollary of the following

Lemma 4 (Disjoint Union) *Let \mathcal{L} be a simplicial d -complex which is $(d + 1)$ -colorable. Then, for a simplex $\delta \subset \mathcal{L} \setminus \partial\mathcal{L}$ and a chosen set of colors C , such that $\text{color}(\delta) \subset C \subset \mathbb{Z}_{d+1}$, there exists a partition of the set of qubits supported on δ into a disjoint union of sets of qubits supported on $(|C| - 1)$ -simplices containing δ , namely*

$$\mathcal{Q}(\delta) = \bigsqcup_{\substack{\sigma \supset \delta \\ \sigma \in \Delta'_{|C|-1}(\mathcal{L}) \\ \text{color}(\sigma) = C}} \mathcal{Q}(\sigma). \quad (2.21)$$

Proof: First note, that two different k -simplices δ_1 and δ_2 in $\mathcal{L} \setminus \partial\mathcal{L}$ colored with the same colors, $\text{color}(\delta_1) = \text{color}(\delta_2)$, cannot belong to the same l -simplex, $l \geq k$, thus do not share a qubit, $\mathcal{Q}(\delta_1) \cap \mathcal{Q}(\delta_2) = \emptyset$. Moreover, if $\mathcal{Q}(\epsilon) \subset \mathcal{Q}(\delta)$, where $\epsilon \in \Delta_d(\mathcal{L})$, then $\epsilon \supset \delta$ and there exists a unique simplex $\sigma \subset \epsilon$ colored with colors B . Since $\text{color}(\sigma) = C \supset \text{color}(\delta)$, then $\sigma \supset \delta$, which finishes the proof of the (Disjoint Union) Lemma 4. \square

In particular, the set of qubits supported on any k -simplex δ in $\mathcal{L} \setminus \partial\mathcal{L}$ with $k < d$ can be decomposed as a disjoint union of qubits supported on $(d - 1)$ -simplices σ containing δ and colored with a chosen set of d colors, $C \supset \text{color}(\delta)$. Notice, that $|\mathcal{Q}(\sigma)| = 2$ for any $\sigma \in \Delta'_{d-1}(\mathcal{L})$, which immediately yields

$$|\mathcal{Q}(\delta)| = \sum_{\substack{\sigma \supset \delta \\ \sigma \in \Delta'_{d-1}(\mathcal{L}) \\ \text{color}(\sigma) = C}} |\mathcal{Q}(\sigma)| \equiv 0 \pmod{2}, \quad (2.22)$$

showing the (Even Support) Lemma 3.

The property needed for the transversal implementation of the gate \overline{R}_n , presented in Section 2.3, can be encapsulated in the following lemma

Lemma 5 (Bipartition of Qubits) *The set of d -simplices in \mathcal{L} , $\Delta_d(\mathcal{L})$, is bipartite.*

Let us first explain the (Bipartition of Qubits) Lemma 5 — the d -simplices in \mathcal{L} can be split into two disjoint sets, where d -simplices in the first set share $(d - 1)$ -faces only with d -simplices from the second set, and vice versa.

Proof: First, construct a graph $G = (V, E)$ with the set of vertices $V = \Delta_d(\mathcal{L})$ and the set of edges $E = \Delta'_{d-1}(\mathcal{L})$. Two vertices $v, w \in V$ are connected by an edge $e \in E$ iff d -simplices corresponding to v and w share a $(d - 1)$ -face corresponding to e . Since for all $\delta \in \Delta'_{d-2}(\mathcal{L})$ the (Even Support) Lemma 3 gives $|\mathcal{Q}(\delta)| \equiv 0 \pmod{2}$, and every cycle in \mathcal{L} is contractible, we obtain that every cycle in the graph G is even. Using the (Graph Bipartition) Lemma 1 we immediately obtain that G is bipartite. This shows that the set of d -simplices in \mathcal{L} , which is equal to the set of qubits, $\Delta_d(\mathcal{L}) = \mathcal{Q}$, is bipartite. \square

Construction of a lattice in d dimensions

A recipe to obtain a lattice \mathcal{L} satisfying the Conditions 1 and 2 required to define color codes in d dimensions is as follows (see Fig. 2.3 for an example in $d = 2$).

1. Start with a d -simplex δ , with vertices which are colored with $d + 1$ colors \mathbb{Z}_{d+1} .
2. Construct a homogeneous simplicial d -complex \mathcal{K} from δ by dividing k -faces of δ into k -simplices. We also require that the coloring is preserved, i.e. every k -face $\sigma \subset \delta$ colored with $C = \text{color}(\sigma)$ is divided into k -simplices colored with C and the whole complex \mathcal{K} is $(d + 1)$ -colorable.
3. Place the d -complex \mathcal{K} inside a d -simplex τ colored with \mathbb{Z}_{d+1} .
4. For every k -face $\varrho \subsetneq \tau$ and for every $(d - k - 1)$ -simplex $\omega \subset \mathcal{K}$ obtained from a $(d - k - 1)$ -face $\sigma \subset \delta$ with complementary colors, $\text{color}(\omega) = \mathbb{Z}_{d+1} \setminus \text{color}(\varrho)$, attach a d -simplex spanned by ϱ and ω .

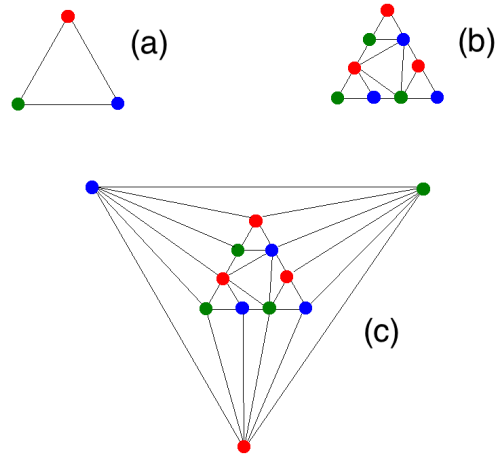


Figure 2.3: Construction of a color code in 2D with spatially local (stabilizer) generators. (a) Take a 2-simplex δ , with vertices colored in red, green and blue. (b) Divide δ into “smaller” simplices with matching colors. This is a 3-colorable homogeneous simplicial 2-complex \mathcal{K} . (c) Place \mathcal{K} inside a 2-simplex τ and attach 2-simplices between τ and \mathcal{K} . The resulting homogeneous simplicial 2-complex \mathcal{L} is 3-colorable, and thus we can define a color code on the lattice \mathcal{L} .

5. Choose \mathcal{L} to be the collection of all d -simplices added in Step 4, together with simplices belonging to \mathcal{K} and τ . This can be used to define a color code on the lattice \mathcal{L} as specified in Section 2.2.

Note that in the above recipe, step 2 is not fully specified. Any homogeneous simplicial d -complex \mathcal{K} obtained from a d -simplex δ will work, as long as \mathcal{K} is $(d + 1)$ -colorable. Such lattices always exist — below we give an explicit example of a family of lattices in any dimension $d \geq 2$. Following steps 3-5, we always obtain a lattice on which we can define a color code in d dimensions.

There is a systematic construction of a family of (fractal) color codes in d dimensions, for which there is an explicit recipe for \mathcal{K} . The resulting codes neither have spatially local generators nor have macroscopic distance, and do not result in color codes, which are topological stabilizer codes. The prescription is as follows.

1. The first member is defined on the lattice \mathcal{L}_1 , obtained from the recipe by setting \mathcal{K} to be a d -simplex.
2. The $i + 1$ member of the family is defined on the lattice \mathcal{L}_{i+1} , obtained from the recipe by setting $\mathcal{K} = \mathcal{L}_i$.

The first three members of the family of the two-dimensional (fractal) color codes are illustrated in Fig. 2.4.

In Ref. [Bom15a], a systematic construction in two and three dimensions for families of color codes with spatially local generators is presented. In two dimensions, \mathcal{K} is chosen to be a part of triangular lattice (as in Fig. 2.3), whereas in three dimensions \mathcal{K} is a part of the BCC lattice. Bombín's constructions result in topological color codes.

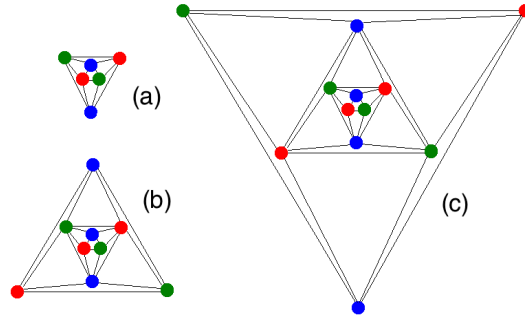


Figure 2.4: The family of (fractal) color codes in two dimensions. The first three members of the family — two-dimensional color codes encoding one logical qubit using (a) 7, (b) 13 and (c) 19 physical qubits.

Quantum Reed-Muller codes as color codes

There exists a family of codes known as the quantum Reed-Muller codes [MS77; Ste99; ADP14]. Here we are concerned with the subfamily of quantum Reed-Muller codes with members labeled uniquely by an integer $m \geq 3$ with parameters $[[2^m - 1, 1, 3]]$, i.e. encoding one logical qubit into $2^m - 1$ physical qubits, with a distance of three. We denote by $QRM(m)$ the m^{th} member of this subfamily. These codes are defined in terms of matrices M_i satisfying the recursion relations

$$M_1 = (1), \quad M_{i+1} = \begin{pmatrix} M_i & 0 & M_i \\ 0 \dots 0 & 1 & 1 \dots 1 \end{pmatrix}. \quad (2.23)$$

Note that the set of columns of M_m is the set of all non-zero binary vectors of length m . By M_m^\perp we denote a matrix dual to M_m , namely a matrix with rows being a basis of the kernel of M_m . Clearly, $M_m(M_m^\perp)^T = 0$. We can define $QRM(m)$ as the stabilizer code with the stabilizer group \mathcal{S}_m generated by rows of M_m and M_m^\perp with 0's and 1's replaced by I 's and X 's or Z 's, namely

$$\mathcal{S}_m = \langle M_m^X, (M_m^\perp)^Z \rangle. \quad (2.24)$$

We now show that $QRM(m)$ is the same as the stabilizer color code $CC_{m-1}(0, m-3)$ obtained from the construction described in preceding subsection by taking the simplicial complex \mathcal{K} to be a $(m-1)$ -simplex δ , $\mathcal{K} = \delta$. In other words, $QRM(m)$ is equal to the first member of the (fractal) color code family in $m-1$ dimensions (see Fig. 2.4 (a) for $m=3$ case). In particular, $QRM(3)$ is Steane's 7-qubit code and $QRM(4)$ is the 15-qubit Reed-Muller code.

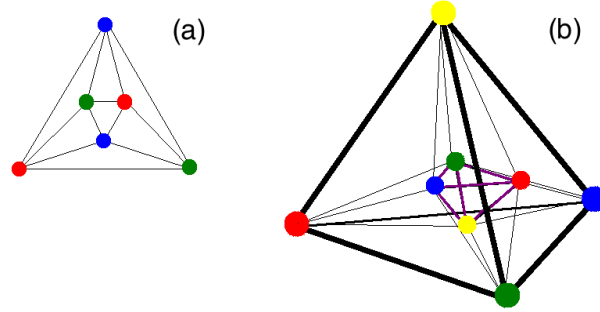


Figure 2.5: Quantum Reed-Muller code $QRM(m)$ as a special case of a (stabilizer) color code $CC_{m-1}(0, m-3)$ for (a) $m=3$ — Steane's 7-qubit code, and (b) $m=4$ — the 15-qubit Reed-Muller code. Steane's code with all the possible transversal gates has recently been implemented experimentally [Nig+14].

To prove this equivalence, it is sufficient to show that there is a one-to-one identification of physical qubits of $QRM(m)$ with those of $CC_{m-1}(0, m-3)$ such that the logical Pauli operators \bar{X} and \bar{Z} are identical, and that the X -type stabilizer generators are identical. Note that this completely specifies the stabilizer group \mathcal{S} , since the Z -type generator matrix is a dual to the X -type generator matrix. In particular, we show that the X -type generator matrix M'_m for $CC_{m-1}(0, m-3)$ is the same as M_m for $QRM(m)$ up to a permutation of columns.

Using the construction described in the preceding subsection, and taking the simplicial complex $\mathcal{K} = \delta$, where δ is a $(m-1)$ -simplex, results in a lattice \mathcal{L} , with $\dim \mathcal{L} = m-1$. The total number of $(m-1)$ -simplices in \mathcal{L} is $2^m - 1$. This is because we attach $(m-1)$ -simplices between every $(k-1)$ -face $\varrho \subset \tau$, for every $1 \leq k \leq m-1$, and the $(m-k-1)$ -face $\sigma \subset \delta$ colored with the complementary colors, $\text{color}(\sigma) = \mathbb{Z}_{d+1} \setminus \text{color}(\varrho)$. We can pick a subset of k vertices of τ in $\binom{m}{k}$ different ways and thus the number of newly attached $(m-1)$ -simplices is $\binom{m}{1} + \binom{m}{2} + \dots + \binom{m}{m-1} = 2^m - 2$. Therefore, including a qubit placed at δ , there are exactly $2^m - 1$ physical qubits in $CC_{\mathcal{L}}(0, m-3)$. On the other hand, there are

exactly m vertices in $\mathcal{L} \setminus \partial\mathcal{L}$, and thus there are m X -type stabilizer generators in $CC_{\mathcal{L}}(0, m-3)$. The weight of a column in M'_m , corresponding to a qubit supported on a $(m-1)$ -simplex π , is given by the number of X -type stabilizer generators supported on that qubit, i.e. the number of vertices belonging to π but not to $\partial\mathcal{L}$. There are exactly $\binom{m}{m-k}$ $(m-1)$ -simplices containing k vertices not belonging to $\partial\mathcal{L}$ and each of them contains a different set of k vertices. Thus, there are $\binom{m}{k}$ different columns of weight k in M'_m and the only way this can occur is if the columns of M'_m are the set of all non-zero binary vectors of length m . Thus, up to a relabeling of physical qubits, M'_m and M_m are identical. Also note that the logical operators of both codes are $\bar{X} = X(Q)$ and $\bar{Z} = Z(Q)$. Therefore the codes are the same.

2.3 Transversal gates in color codes

As mentioned in the introduction, transversal gates are fault-tolerant. In this section, we first review some relevant features of a class of CSS subsystem codes, which includes the color codes defined in Sectionrefsec:Ddim. Then, we examine transversal gates of codes in this class. We show that $\overline{\text{CNOT}}$ is transversal in any such code and under certain additional conditions the Hadamard and \bar{R}_n can be transversal, too. Finally, we show that the additional conditions are satisfied by certain color codes.

Subsystem codes

A CSS subsystem code [Pou05; Bac06] is specified by its gauge group \mathcal{G} , which is a subgroup of the Pauli group on physical qubits Q . Each X -type gauge group generator $X(G^x)$ consists of Pauli X operators applied to qubits G^x ; similarly for Z -type generators. The stabilizer group $\mathcal{S} \subseteq \mathcal{G}$ is the group generated by all Pauli operators $X(S^x)$ and $Z(S^z)$ contained in \mathcal{G} , which commute with every element of \mathcal{G} . Note that a stabilizer code is a special case of a subsystem code, for which $\mathcal{G} = \mathcal{S}$. The codewords are the $+1$ eigenvectors of all elements of \mathcal{S} . We say that two codewords are equivalent if they differ by application of a linear combination of elements of $\mathcal{G} \setminus \mathcal{S}$. This allows one to decompose the subspace of codewords into a tensor product of two spaces: *logical* qubits and *gauge* qubits. Elements of $\mathcal{G} \setminus \mathcal{S}$ have no effect on the state of the logical qubits, but may change that of the gauge qubits.

For a subsystem code, we say a unitary implements a *logical gate* if it preserves the space of all codewords, and has an action on the logical qubits which is independent of any action on the gauge qubits. A logical gate \bar{U} can be implemented on the logical qubits $|\psi\rangle$ as a *bare* gate U_{bare} which leaves gauge qubits $|g\rangle$ unchanged,

$U_{\text{bare}} : |\psi\rangle|g\rangle \mapsto (\bar{U}|\psi\rangle)|g\rangle$, or more generally as a *dressed* gate U_{dressed} , which can affect the gauge qubits too, $U_{\text{dressed}} : |\psi\rangle|g\rangle \mapsto (\bar{U}|\psi\rangle)|g'\rangle$.

Consider the class of CSS subsystem codes which

- encode one logical qubit,
- have bare logical \bar{X} and \bar{Z} implemented by $X(Q)$ and $Z(Q)$.

Note that these codes are defined on an odd number of physical qubits, $|Q| \equiv 1 \pmod{2}$, since \bar{X} and \bar{Z} anticommute.

We can define a pair of inequivalent (and not normalized) codewords, which are representatives of logical $|\bar{0}\rangle$ and $|\bar{1}\rangle$, namely

$$|\bar{0}\rangle|g_X\rangle = \sum_{X(G) \in \mathcal{G}} X(G)|\mathbf{0}\rangle, \quad (2.25)$$

$$|\bar{1}\rangle|g_X\rangle = \bar{X}|\bar{0}\rangle|g_X\rangle, \quad (2.26)$$

where $|\mathbf{0}\rangle$ is a state with every physical qubit set to $|0\rangle$, and $|g_X\rangle$ is a fixed state of the gauge qubits. One can verify that the states $|\bar{0}\rangle|g_X\rangle$ and $|\bar{1}\rangle|g_X\rangle$ are +1 eigenstates of \mathcal{S} , and satisfy $\bar{Z}|\bar{0}\rangle|g_X\rangle = |\bar{0}\rangle|g_X\rangle$, $\bar{Z}|\bar{1}\rangle|g_X\rangle = -|\bar{1}\rangle|g_X\rangle$. They are also +1 eigenstates of every X -type generator of \mathcal{G} . All equivalent codewords can be generated from $|\bar{0}\rangle|g_X\rangle, |\bar{1}\rangle|g_X\rangle$ by application of a linear combination of elements from $\mathcal{G} \setminus \mathcal{S}$. An alternative pair of representatives of logical $|\bar{0}\rangle$ and $|\bar{1}\rangle$ is

$$|\bar{0}\rangle|g_Z\rangle = \sum_{X(S) \in \mathcal{S}} X(S)|\mathbf{0}\rangle, \quad (2.27)$$

$$|\bar{1}\rangle|g_Z\rangle = \bar{X}|\bar{0}\rangle|g_Z\rangle, \quad (2.28)$$

which are +1 eigenstates of all Z -type generators of \mathcal{G} .

Transversal gates in subsystem codes

Consider a CSS subsystem code with one logical qubit, and \bar{X} and \bar{Z} implemented by $X(Q)$ and $Z(Q)$. To check that a physical unitary U implements a dressed logical gate \bar{U} in such a code, one can verify its action on $|\bar{0}\rangle|g\rangle$, and $|\bar{1}\rangle|g\rangle$ for every state $|g\rangle$ of the gauge qubits. Alternatively, it is sufficient to verify that U has the correct action by conjugation on \bar{X} and \bar{Z} , and that it preserves⁶ the gauge group \mathcal{G} .

⁶Note that preservation of the gauge group under the action of a physical unitary U is a sufficient, but not a necessary condition for U to implement a dressed logical gate.

The logical gate $\overline{\text{CNOT}}$ can be implemented transversally between two identical copies of a CSS subsystem code by applying a physical gate CNOT to every pair of corresponding qubits in the first and the second copy. This can be verified by checking that under conjugation by $\overline{\text{CNOT}}$, $\overline{XI} \mapsto \overline{XX}$, $\overline{IX} \mapsto \overline{IX}$, $\overline{ZI} \mapsto \overline{ZI}$, $\overline{IZ} \mapsto \overline{ZZ}$ and $\mathcal{G} \otimes \mathcal{G}$ is preserved⁷.

For a *self-dual* CSS subsystem code, namely a code with X - and Z -type gauge group generators supported on the same sets of qubits, $\mathcal{G} = \langle X(G_i), Z(G_i) \rangle$, a dressed logical Hadamard gate can be implemented transversally as $\overline{H} = H(Q)$. To see this, observe that under conjugation by $H(Q)$, $\overline{X} \mapsto \overline{Z}$, $\overline{Z} \mapsto \overline{X}$, $X(G) \mapsto Z(G)$ and $Z(G) \mapsto X(G)$, and thus \mathcal{G} is preserved.

The last logical gate we analyze is $\overline{R}_n = \text{diag} \left(1, e^{\frac{2\pi i}{2^n}} \right)$, for an integer $n > 0$. We aim to implement \overline{R}_n transversally as a bare logical gate by applying the same single-qubit unitary to some subset $T \subset Q$ of the physical qubits, and applying that unitary's inverse to the rest of the qubits $T^c := Q \setminus T$. Specifically, we now prove that \overline{R}_n is implemented by $R = R_n^k(T)R_n^{-k}(T^c)$, for some suitably chosen $k \in \{1, 2, \dots, 2^n - 1\}$, provided that T and \mathcal{G} satisfy

$$\forall X(G) \in \mathcal{G} : |T \cap G| \equiv |T^c \cap G| \pmod{2^n}. \quad (2.29)$$

First, pick k such that

$$k(|T| - |T^c|) \equiv 1 \pmod{2^n}. \quad (2.30)$$

The existence of k is guaranteed by Bezout's lemma, since $|Q|$ is odd, $|T| - |T^c| = 2|T| - |Q| \equiv 1 \pmod{2}$, and thus $\text{gcd}(2|T| - |Q|, 2^n) = 1$. Noting that $R_n^{\pm k}|0\rangle = |0\rangle$ and $R_n^{\pm k}X = e^{\pm \frac{2\pi i k}{2^n}} X R_n^{\mp k}$, we obtain

$$R|\overline{0}\rangle|g_X\rangle = \sum_{X(G) \in \mathcal{G}} R_n^k(T)R_n^{-k}(T^c)X(G)|\mathbf{0}\rangle \quad (2.31)$$

$$= \sum_{X(G) \in \mathcal{G}} e^{\frac{2\pi i k}{2^n}|T \cap G|} e^{-\frac{2\pi i k}{2^n}|T^c \cap G|} X(G)|\mathbf{0}\rangle \quad (2.32)$$

$$= \sum_{X(G) \in \mathcal{G}} X(G)|\mathbf{0}\rangle = |\overline{0}\rangle|g_X\rangle, \quad (2.33)$$

$$R|\overline{1}\rangle|g_X\rangle = R_n^k(T)R_n^{-k}(T^c)X(Q)|\overline{0}\rangle|g_X\rangle \quad (2.34)$$

$$= e^{\frac{2\pi i k}{2^n}|T|} e^{-\frac{2\pi i k}{2^n}|T^c|} X(Q)R|\overline{0}\rangle|g_X\rangle \quad (2.35)$$

$$= e^{\frac{2\pi i}{2^n}} X(Q)|\overline{0}\rangle|g_X\rangle = e^{\frac{2\pi i}{2^n}} |\overline{1}\rangle|g_X\rangle, \quad (2.36)$$

⁷Notice, that generators of $\mathcal{G} \otimes \mathcal{G}$ are mapped under conjugation to another set of generators, namely $X(G) \otimes I(G) \mapsto X(G) \otimes X(G)$, $Z(G) \otimes I(G) \mapsto I(G) \otimes Z(G)$, $I(G) \otimes X(G) \mapsto I(G) \otimes X(G)$ and $I(G) \otimes Z(G) \mapsto Z(G) \otimes Z(G)$.

which shows that R correctly implements logical \overline{R}_n when the gauge qubits are in the state $|g_X\rangle$. However, all other states of the gauge qubits can be reached by application of Z -type operators from $\mathcal{G} \setminus \mathcal{S}$, which all commute with R (since it is diagonal in the Z -basis). Therefore for any state $|g\rangle$ of the gauge qubits, it must be that $R : |\overline{0}\rangle|g\rangle \mapsto |\overline{0}\rangle|g\rangle, |\overline{1}\rangle|g\rangle \mapsto e^{\frac{2\pi i}{2^m}} |\overline{1}\rangle|g\rangle$, verifying that R implements the bare logical gate \overline{R}_n .

It may not be obvious that there exists a set $T \subset Q$ satisfying (2.29) for a given code. In the later parts of this chapter we show an explicit construction of T for color codes in d dimensions, with $n \leq d$. Notice that condition (2.29) can be inferred from the following condition

$$\left| T \cap \bigcap_{i=1}^m G_i \right| \equiv \left| T^c \cap \bigcap_{i=1}^m G_i \right| \pmod{2^{n-m+1}}, \quad (2.37)$$

where $m = 1, \dots, n$ and $\{X(G_1), \dots, X(G_m)\}$ is any subset of the X -type generators of the gauge group \mathcal{G} . To see the implication (2.37) \implies (2.29) notice, that for any $X(G) \in \mathcal{G}$, we can write it as product of generators, namely $X(G) = \prod_{i=1}^m X(G_i)$. Then

$$G = G_1 \vee G_2 \vee \dots \vee G_m, \quad (2.38)$$

where we used the symmetric difference of sets, $A \vee B := (A \setminus B) \cup (B \setminus A)$. Using the Inclusion-Exclusion Principle for symmetric difference⁸ we obtain

$$|T \cap G| = |T \cap (G_1 \vee G_2 \vee \dots \vee G_m)| \quad (2.39)$$

$$= \sum_i |T \cap G_i| - 2 \sum_{i \neq j} |T \cap (G_i \cap G_j)| + \quad (2.40)$$

$$4 \sum_{i \neq j \neq k} |T \cap (G_i \cap G_j \cap G_k)| - \dots \quad (2.41)$$

$$+ (-2)^{m-1} |T \cap (G_1 \cap G_2 \cap \dots \cap G_m)|, \quad (2.42)$$

and a similar expression for $|T^c \cap G|$. Clearly, if condition (2.37) holds, then $|T \cap G| - |T^c \cap G| \equiv 0 \pmod{2^n}$, showing (2.29). Moreover, condition (2.37) is easier to verify than condition (2.29), since we only need to check it for the X -type generators of \mathcal{G} , rather than for every X -type element of \mathcal{G} .

We can summarize the discussion of how to implement transversal \overline{R}_n in the following lemma

⁸For sets A_1, A_2, \dots, A_m , we have $|A_1 \vee A_2 \vee \dots \vee A_m| = \sum_i |A_i| - 2 \sum_{i \neq j} |A_i \cap A_j| + \dots + (-2)^{m-1} |A_1 \cap A_2 \cap \dots \cap A_m|$.

Lemma 6 (Sufficient Condition) Consider a CSS subsystem code encoding one logical qubit. Let the code be defined on a set of physical qubits Q , where $|Q|$ is odd and with bare logical operators $\bar{X} = X(Q)$ and $\bar{Z} = Z(Q)$. If there exists $T \subset Q$, such that for any $m = 1, \dots, n$:

$$\left| T \cap \bigcap_{i=1}^m G_i \right| \equiv \left| T^c \cap \bigcap_{i=1}^m G_i \right| \pmod{2^{n-m+1}}, \quad (2.43)$$

for every subset $\{X(G_1), \dots, X(G_m)\}$ of the X -type gauge generators of the code, then

$$R = R_n^k(T)R_n^{-k}(T^c) \quad (2.44)$$

implements logical \bar{R}_n , where k is a solution to $k(|T| - |T^c|) \equiv 1 \pmod{2^n}$ and $T^c = Q \setminus T$.

Transversal implementation of \bar{R}_n in color code

Here we show how to implement the logical gate \bar{R}_n transversely in the color code $CC_{\mathcal{L}}(x, z)$, for any integer $n \leq \dim(\mathcal{L})/(x+1)$. One applies $R = R_n^k(T)R_n^{-k}(T^c)$ for some integer k , where T and its complement $T^c = Q \setminus T$ correspond to the bipartite decomposition of qubits Q specified in the (Bipartition of Qubits) Lemma 5. We make use of the following property

Lemma 7 (Property of T) For any m -simplex σ in $\mathcal{L} \setminus \partial\mathcal{L}$ with $m < d$

$$|T \cap Q(\sigma)| = |T^c \cap Q(\sigma)|. \quad (2.45)$$

Proof: By the choice of the set T , every $(d-1)$ -simplex δ has one qubit in T , and one qubit in $T^c = Q \setminus T$, which is equivalent to $|T \cap Q(\delta)| = |T^c \cap Q(\delta)|$. Using the (Disjoint Union) Lemma 4, we can decompose the set of qubits $Q(\sigma)$ supported on an m -simplex σ , where $m < d$, as a disjoint union of qubits supported on $(d-1)$ -simplices colored with a chosen set of d colors $C \supset \text{color}(\sigma)$, and then we immediately obtain

$$|T \cap Q(\sigma)| - |T^c \cap Q(\sigma)| = \sum_{\substack{\delta \supset \sigma \\ \delta \in \Delta_{d-1}^V(\mathcal{L}) \\ \text{color}(\delta) = C}} |T \cap Q(\delta)| - |T^c \cap Q(\delta)| = 0, \quad (2.46)$$

which shows the (Property of T) Lemma 7. \square

Note that (2.43) in the (Sufficient Condition) Lemma 6 follows from the (Property of T) Lemma 7. To see this, observe first that every stabilizer generator $X(\delta_i)$ is supported on a x -simplex δ_i , thus $G_i = \mathcal{Q}(\delta_i)$ and we obtain

$$\bigcap_{i=1}^m \mathcal{Q}(\delta_i) = \emptyset \quad \text{or} \quad \bigcap_{i=1}^m \mathcal{Q}(\delta_i) = \mathcal{Q}(\tau), \quad (2.47)$$

where τ is a simplex colored with colors $C = \bigcup_{i=1}^m \text{color}(\delta_i)$, such that $\tau \supset \delta_1, \dots, \delta_m$. The case of an empty intersection is trivial. Since $|\text{color}(\delta_i)| = x + 1$, then obviously $|C| \leq m(x + 1) \leq d$, and thus τ is at most $(d - 1)$ -simplex. Using the (Property of T) Lemma 7 we obtain that for any $m = 1, \dots, n$:

$$\left| T \cap \bigcap_{i=1}^m \mathcal{Q}(\delta_i) \right| - \left| T^c \cap \bigcap_{i=1}^m \mathcal{Q}(\delta_i) \right| = \quad (2.48)$$

$$|T \cap \mathcal{Q}(\tau)| - |T^c \cap \mathcal{Q}(\tau)| = 0, \quad (2.49)$$

which implies (2.43). The (Sufficient Condition) Lemma 6 implies that R implements the logical \bar{R}_n . In particular, one can implement \bar{R}_d using the code $CC_d(0, d - 2)$, since $\dim \mathcal{L} = d$, $x = 0$ and thus $\lfloor \frac{\dim \mathcal{L}}{x+1} \rfloor = d$.

2.4 Universal transversal gates with color codes

A finite set of gates which is universal can be used to implement any logical unitary, with arbitrary precision. In particular, due to the Solovay-Kitaev [Kit97; NC10] theorem, the number of applied gates scales poly-logarithmically with the precision of approximation. Note that the set $\{\bar{H}, \overline{\text{CNOT}}, \bar{R}_n\}$ is universal for any integer $n > 2$.

In this section, we show how to achieve a universal transversal gate set with color codes by using the technique of *gauge fixing* to switch between different codes. This technique allows one to take advantage of the transversally implementable gates for different color codes. We first illustrate the method with a simple example of two 15-qubit codes [PR13; ADP14]. Then, we define a partial order between color codes. One can switch between color codes which are comparable to implement a universal gate set in three or higher dimensions.

Switching between codes using gauge fixing

First, let us define matrices H_1 and H_2 given by

$$H_1 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \quad (2.50)$$

$$H_2 = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \quad (2.51)$$

Moreover, for a binary matrix M , we define M^X to be a matrix obtained from M by the following substitutions, $0 \mapsto I$ and $1 \mapsto X$. Similarly for M^Z , we substitute $0 \mapsto I$ and $1 \mapsto Z$. Let \mathcal{C}_A be the stabilizer code with the stabilizer group \mathcal{S}_A generated by rows of H_1^X , H_1^Z and H_2^Z , which we denote by

$$\mathcal{S}_A = \langle H_1^X, H_1^Z, H_2^Z \rangle. \quad (2.52)$$

Let \mathcal{C}_B be the subsystem code with the stabilizer group \mathcal{S}_B and the gauge group \mathcal{G}_B chosen as follows

$$\mathcal{S}_B = \langle H_1^X, H_1^Z \rangle, \quad \mathcal{G}_B = \langle H_1^X, H_2^X, H_1^Z, H_2^Z \rangle. \quad (2.53)$$

We can consider both codes \mathcal{C}_A and \mathcal{C}_B to be defined on the same 15 physical qubits. One can check that \mathcal{C}_A represents the $[[15, 1, 3]]$ quantum Reed-Muller (stabilizer) code [MS77; Ste99; ADP14] and \mathcal{C}_B is a $[[15, 1, 3]]$ (subsystem) code, which can be thought of as the $[[15, 7, 3]]$ Hamming code, with six of the seven logical qubits treated as gauge qubits. Note also that $\mathcal{S}_B \subset \mathcal{G}_A = \mathcal{S}_A$ and \mathcal{G}_B has X - and Z -type generators supported on the same qubits (i.e. \mathcal{C}_B is a self-dual subsystem code).

Since the X -type generators of \mathcal{G}_B coincide with the X -type generators of \mathcal{S}_A , the codewords of \mathcal{C}_A and \mathcal{C}_B are the same when the latter has a gauge state $|g_Z\rangle$. In other words, codewords $|\bar{0}\rangle, |\bar{1}\rangle$ for \mathcal{C}_A are the same as codewords $|\bar{0}\rangle|g_Z\rangle, |\bar{1}\rangle|g_Z\rangle$ for \mathcal{C}_B , as defined in Eqs. (2.27) and (2.28). On the other hand the codewords $|\bar{0}\rangle|g_X\rangle, |\bar{1}\rangle|g_X\rangle$ for \mathcal{C}_B (as defined in Eqs. (2.25) and (2.26)), are not valid codewords for \mathcal{C}_A .

Now we show that $R_3^{\otimes 15}$ implements \overline{R}_3 transversally in \mathcal{C}_A . Consider any three of the four X -type generators for \mathcal{G}_A , and specify their support on subsets of qubits G_1, G_2, G_3 , which correspond to rows of H_1 . One can verify that $|G_a| = 8 \equiv 0 \pmod{2^3}$, $|G_a \cap G_b| = 4 \equiv 0 \pmod{2^2}$, and $|G_a \cap G_b \cap G_c| = 2 \equiv 0 \pmod{2}$, where $\{a, b, c\} = \{1, 2, 3\}$. Therefore by the (Sufficient Condition) Lemma 6, and by setting T to be an empty set, $T = \emptyset$, we see that $R_3^{\otimes 15}$ implements \overline{R}_3 transversally in the code \mathcal{C}_A . In contrast to the code \mathcal{C}_B , the extra X -type generators in $\mathcal{G}_B \setminus \mathcal{G}_A$ do not satisfy these conditions, and thus one cannot show that \overline{R}_3 is implemented transversally in \mathcal{C}_B .

It is straightforward to verify that \overline{H} is implemented transversally by $H^{\otimes 15}$ in \mathcal{C}_B . It swaps X and Z on any physical qubit, and therefore acts on the representative states as $H^{\otimes 15} : |\psi\rangle|g_Z\rangle \mapsto (\overline{H}|\psi\rangle)|g_X\rangle$. Since the state of the gauge qubits has changed, $H^{\otimes 15}$ is a dressed implementation of \overline{H} in \mathcal{C}_A . Clearly, $H^{\otimes 15}$ does not implement \overline{H} in \mathcal{C}_A , since it takes the state $|\psi\rangle|g_Z\rangle \in \mathcal{C}_A$ to $(\overline{H}|\psi\rangle)|g_X\rangle \notin \mathcal{C}_A$.

To implement \overline{H} fault-tolerantly in \mathcal{C}_A , we use the technique of *gauge fixing*. First, one should apply $H^{\otimes 15}$, resulting in mapping $|\psi\rangle|g_Z\rangle$ to $(\overline{H}|\psi\rangle)|g_X\rangle$, which is a codeword of \mathcal{C}_B , but not of \mathcal{C}_A . Then, to switch from code \mathcal{C}_B to \mathcal{C}_A , one should sequentially measure each of the six Z -type stabilizer generators generated by rows of H_2^Z , i.e. those in $\mathcal{S}_A \setminus \mathcal{S}_B$. Note that it is possible to fault-tolerantly measure the stabilizer generators in any stabilizer code [NC10]. If the measurement reveals that a particular Z -type generator is not satisfied, then one should apply an X -type Pauli operator which commutes with all generators in H_2^Z and H_1^Z , except for the violated stabilizer generator (with which it must anticommute). Such an X -type Pauli operator always exists. Following this application, the Z -type generator will no longer be violated. Therefore, after this is carried out for all six generators in H_2^Z , the state will have changed from $(\overline{H}|\psi\rangle)|g_X\rangle$ to $(\overline{H}|\psi\rangle)|g_Z\rangle$, as required. Specifically, we use the term *gauge fixing* to refer to the process of measuring and setting the gauge qubits to a desired state.

To recap, in the $[[15, 1, 3]]$ Reed-Muller code \mathcal{C}_A , one can implement \overline{H} fault-tolerantly with the following procedure

$$|\psi\rangle|g_Z\rangle \xrightarrow{H^{\otimes 15}} (\overline{H}|\psi\rangle)|g_X\rangle \xrightarrow{\text{gauge fixing}} (\overline{H}|\psi\rangle)|g_Z\rangle. \quad (2.54)$$

In combination with the transversal gates of \mathcal{C}_A , this allows one to implement a fault-tolerant universal gate set $\{\overline{H}, \overline{\text{CNOT}}, \overline{R}_3\}$. We will repeat essentially the same procedure for color codes later.

Partial order of color codes

Given a d -dimensional lattice \mathcal{L} , $\dim \mathcal{L} = d$, satisfying Conditions 1 and 2, we can catalog all color codes defined on \mathcal{L} . Namely, a pair of integers $x, z \geq 0$, such that $x+z \leq d-2$, corresponds to a color code, denoted as $CC_{\mathcal{L}}(x, z)$, with X - and Z -type gauge generators supported on $(d-2-z)$ - and $(d-2-x)$ -simplices. Note that the X - and Z -type stabilizer generators of $CC_{\mathcal{L}}(x, z)$ are supported on x -simplices and z -simplices, respectively. In two dimensions, $d = 2$, there is only one color code, $CC_2(0, 0)$ — a stabilizer code, with both X - and Z -type stabilizer generators supported on 0-simplices, whereas in three dimensions, $d = 3$, there are three color codes, $CC_3(1, 0)$, $CC_3(0, 1)$ — stabilizer codes, and $CC_3(0, 0)$ — a subsystem code.

One can define a partial order for subsystem color codes defined on the same lattice \mathcal{L} if each codeword of code \mathcal{C} is also a codeword of the other code \mathcal{C}' . In particular, we say that $\mathcal{C} > \mathcal{C}'$ holds if

- \mathcal{C} and \mathcal{C}' encode the same number of logical qubits, with identical bare logical Pauli operators,
- the gauge group \mathcal{G} of \mathcal{C} is contained in the gauge group \mathcal{G}' of \mathcal{C}' , $\mathcal{G} \subset \mathcal{G}'$.

Note that $\mathcal{G} \subset \mathcal{G}'$ implies $\mathcal{S}' \subset \mathcal{S}$, thus any codeword of \mathcal{C} is also a codeword of \mathcal{C}' , and since the bare Pauli operators for the logical qubit are the same in both codes, it actually represents the same logical codeword in both codes. Observe, that the partial order we have just defined can be succinctly expressed as

$$CC_{\mathcal{L}}(x, z) > CC_{\mathcal{L}}(x', z') \iff x \geq x' \wedge z \geq z', \quad (2.55)$$

as illustrated in Fig. 2.6. This follows from the observation that due to the (Disjoint Union) Lemma 4 the X -type gauge generators of $CC_{\mathcal{L}}(x, z)$, which are supported on $(d-2-z)$ -simplices, can be expressed as the product of the X -type gauge generators of $CC_{\mathcal{L}}(x', z')$ supported on $(d-2-z')$ -simplices, since $z \geq z'$. Similarly for Z -type gauge generators. We represent the family of color codes in Fig. 2.6, and show their partial order using arrows.

Universal fault-tolerant gate set in color codes

Here we apply the techniques just discussed to color codes defined on the same lattice \mathcal{L} . One can switch back and forth between two codes which are comparable, $CC_{\mathcal{L}}(x, z) < CC_{\mathcal{L}}(x', z')$, as follows

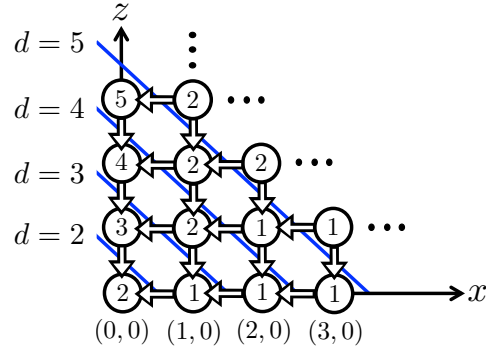


Figure 2.6: Family of color codes. For a given lattice \mathcal{L} , only color codes below the d^{th} diagonal line can be realized, where $d = \dim \mathcal{L}$ and the point (x, z) corresponds to the color code $CC_{\mathcal{L}}(x, z)$. This constraint holds, since x and z have to satisfy $x + z \leq d - 2$. An arrow from code \mathcal{C} to \mathcal{C}' indicates partial order between them, $\mathcal{C} > \mathcal{C}'$. The number placed at (x, z) indicates the maximum gate \overline{R}_n which can be implemented transversally with the stabilizer color code $CC_d(x, z)$, with $d = x + z + 2$, resulting in $n = \lfloor \frac{d}{x+1} \rfloor$.

- $CC_{\mathcal{L}}(x, z) \mapsto CC_{\mathcal{L}}(x', z')$: one does nothing, since codewords of $CC_{\mathcal{L}}(x, z)$ are codewords of $CC_{\mathcal{L}}(x', z')$,
- $CC_{\mathcal{L}}(x', z') \mapsto CC_{\mathcal{L}}(x, z)$: one can view the codewords of $CC_{\mathcal{L}}(x, z)$ as those for $CC_{\mathcal{L}}(x', z')$ with the additional gauge qubits present in $CC_{\mathcal{L}}(x, z)$ set to a particular state. To switch, one fixes the state of the additional gauge qubits to the appropriate state.

Given a three-dimensional lattice \mathcal{L} , $\dim \mathcal{L} = 3$, one can implement a universal gate set starting with a code $CC_{\mathcal{L}}(0, 1)$. As explained earlier, one can transversally perform the logical $\overline{\text{CNOT}}$ and \overline{R}_3 on that code. To form a universal gate set, it suffices to also implement logical \overline{H} . This gate cannot be implemented transversally in $CC_{\mathcal{L}}(0, 1)$, but can be achieved in $CC_{\mathcal{L}}(0, 0)$. Note that $CC_3(0, 0) < CC_3(0, 1)$, therefore any codeword in $CC_3(0, 1)$ is a valid codeword in $CC_3(0, 0)$. In particular, we can think of $|\psi\rangle \in CC_3(0, 1)$ as $|\psi\rangle|g\rangle \in CC_3(0, 0)$, where $|g\rangle$ is a state of the gauge qubits of $CC_3(0, 0)$. By applying $H(Q)$ we perform the logical \overline{H} on the logical qubits of $CC_3(0, 0)$, which also changes the state of the gauge qubits, namely

$$H(Q) (|\psi\rangle|g\rangle) = (\overline{H}|\psi\rangle) |g'\rangle. \quad (2.56)$$

Note that the resulting codeword $(\overline{H}|\psi\rangle) |g'\rangle \in CC_3(0, 0)$ is not a valid codeword of $CC_3(0, 1)$, since the gauge qubits are in the state $|g'\rangle \neq |g\rangle$. To return to $CC_3(0, 1)$, one needs to *fix the gauge qubits* to the correct state, namely $|g'\rangle \mapsto |g\rangle$, and we

obtain a codeword $\overline{H}|\psi\rangle|g\rangle \in CC_3(0, 1)$. Since $CC_3(0, 1)$ is a stabilizer code, it is possible to measure and correct the violated stabilizers in a fault-tolerant way, just as in Section 2.4. Therefore, to fix the gauge, one should first measure all Z -type stabilizer generators supported on 1-simplices, and then apply the appropriate X -type Pauli operators in order to correct any violated stabilizer generators. After this, assuming no errors have occurred, all the stabilizer generators for $CC_3(0, 1)$ are satisfied.

To summarize, we can perform the logical \overline{H} on $CC_3(0, 1)$ by first applying $H(Q)$ and subsequently fixing the gauge to return to the codespace of $CC_3(0, 1)$,

$$|\psi\rangle|g'\rangle \xrightarrow{H(Q)} (\overline{H}|\psi\rangle)|g'\rangle \xrightarrow{\text{gauge fixing}} (\overline{H}|\psi\rangle)|g\rangle. \quad (2.57)$$

Since $\overline{\text{CNOT}}$ and \overline{R}_3 can be performed transversally in $CC_3(0, 1)$, one can fault-tolerantly implement a universal gate-set $\{\overline{H}, \overline{\text{CNOT}}, \overline{R}_3\}$ in $CC_3(0, 1)$. This procedure can be directly generalized to fault-tolerantly implement the universal gate set $\{\overline{H}, \overline{\text{CNOT}}, \overline{R}_d\}$ with the code $CC_d(0, d - 2)$ in d dimensions.

*Chapter 3***THE COST OF UNIVERSALITY IN SURFACE AND COLOR CODES: STATE DISTILLATION VERSUS CODE SWITCHING**

Quantum error correction allows to use faulty quantum hardware to build a reliable quantum computer. It is a significant challenge for the field of quantum computing to develop good quantum error correction schemes, which lead to reliable, scalable storage and processing of quantum information with the weakest possible quality and minimum overhead requirements.

Topological stabilizer codes [BK13; Kit03; BM06], in particular the toric code and the color code, are the promising candidates for quantum computer architecture. With typically higher error correction thresholds than other schemes, such as concatenated codes [AB97; CJL16], topological codes can be used to reliably store quantum information with relatively imperfect physical qubits. The classical decoding is a process that diagnoses errors and proposes appropriate correction given the measurement outcomes can be done efficiently. Moreover, in both the toric and color codes, many logical gates can be implemented transversely [BM06; Den+02], and therefore reliably despite the inevitable presence of noise. These features, along with the fact that the stabilizer measurements required to implement the codes are geometrically local, make these codes excellent candidates for quantum hardware.

The fault-tolerant transverse gates in both the two-dimensional (2D) toric and color codes are in the Clifford group and therefore do not form a universal quantum gate set. A number of results rule out the possibility of designing similar 2D topological codes with a richer set of transverse gates [EK09; BK13; Bev+16; JKY18]. Magic state distillation [BK05; Kni04b; Kni04a] overcomes this shortcoming by implementing Clifford operations alone to distill from many noisy copies a high fidelity resource state, which later can be used to implement a non-Clifford gate. Unfortunately, despite significant recent improvements [BH12; Haa+17], the distillation overhead is formidable [Fow+12]. Code switching [PR13; ADP14] in 3D color codes is currently the only fault-tolerant alternative to distillation for topological schemes [Bom15a; KB15].

The experimental difficulty of moving to three-dimensional architectures could be justified if they result in significant overhead savings. Our goal is therefore to

extend previous studies which have focused on distillation [Fow+12; CO16] to estimate and compare the overhead of code switching. In particular, we set out to answer the following question: *Consider a hybrid architecture mainly consisting of 2D quantum-local regions for Clifford operations, along with a special 3D quantum-local region only to produce resource states. Are fewer qubits required in the special 3D region if used for code-switching or state distillation?*

By quantum local [Bom15b], we allow measurements of only geometrically local operators, but permit classical information to be passed without geometric restriction. The main result of this chapter is that even in the most favorable treatment of code-switching, we find no improvement in the overhead compared with magic state distillation using the standard Bravyi-Kitaev protocol. However, one may hope that better alternative forms of code switching could be found in future which outperform those presented here.

In order to estimate the overhead of universality in color codes, we need to implement a decoder of the 2D triangular color code. In particular, we adapted the 2D color code decoder from Ref. [Del14a] to the case of the code with boundaries and noisy syndrome extraction. We further optimize performance of the decoder in terms of the circuit-level threshold for the depolarizing channel, obtaining the value of 0.3 % for the color code on the hexagonal lattice.

The organization of this chapter is as follows. In Sec. 3.1, we outline stabilizer and subsystem codes, focusing on 2D and 3D color codes. In Sec. 3.2 we specify the circuit-level noise model we assume and various relevant aspects of error correction. In Sec. 3.3 and Sec. 3.4 we explain the two approaches to achieve topologically encoded magic states: state distillation and code switching. In Sec. 3.5, we find circuit-level thresholds for the 2D hexagonal lattice color code using an efficient decoder. Lastly, we compare the overhead of distillation and code switching in Sec. 3.6.

3.1 Overview of stabilizer and subsystem codes

A stabilizer code [Got97] is specified by the stabilizer group \mathcal{S} , which is an Abelian subgroup of the Pauli group \mathcal{P}_n on n qubits generated by Pauli X , Y and Z operators. The code space is defined to be the +1 eigenspace of operators in \mathcal{S} . Here we abuse notation and write $|\psi\rangle \in \mathcal{S}$ for a code state $|\psi\rangle$, i.e., we use \mathcal{S} to represent both the stabilizer group of Pauli operators, and the code space. When referring to this state at the logical level in the code space, we write $|\psi\rangle = |\overline{\psi}\rangle$. For the code space to be

non-trivial we require $-I \notin \mathcal{S}$. We say that the code encodes k logical qubits if the number of (independent) generators of the stabilizer group \mathcal{S} is $n - k$.

The distance d of the code is defined to be the minimum weight of a Pauli operator P which commutes with the elements in \mathcal{S} but does not belong to \mathcal{S} . In other words, we look for a minimum-weight operator in the normalizer $\mathcal{N}(\mathcal{S})$ of the stabilizer group \mathcal{S} which does not belong to \mathcal{S} , namely

$$d = \min_{P \in \mathcal{N}(\mathcal{S}) \setminus \mathcal{S}} |P|. \quad (3.1)$$

This can be generalized: a subsystem code is defined by its gauge group $\mathcal{G} \subset \mathcal{P}$, which can be non-Abelian. The center of the gauge group defines another stabilizer group $\mathcal{S}(\mathcal{G})$. The code states of $\mathcal{S}(\mathcal{G})$ are also code states of the subsystem code, i.e., $|\psi\rangle \in \mathcal{S}(\mathcal{G}) \implies |\psi\rangle \in \mathcal{G}$. The key conceptual idea of a subsystem code is that elements of \mathcal{G} do not affect the encoded information. Let $|\psi\rangle \in \mathcal{S}(\mathcal{G})$ be a code state and $U \in \mathcal{A}[\mathcal{G}]$ be a unitary in the algebra generated by \mathcal{G} . We say that the state $U|\psi\rangle \in \mathcal{S}(\mathcal{G})$ is logically equivalent to $|\psi\rangle$ for the subsystem code and write $U|\psi\rangle \sim |\psi\rangle$. This motivates our thinking of the code states as a tensor product between a logical subsystem and a gauge subsystem, namely

$$|\psi\rangle = |\bar{\psi}\rangle_L \otimes |\bar{0}\rangle_{\mathcal{G}}, \quad (3.2)$$

$$U|\psi\rangle = |\bar{\psi}\rangle_L \otimes (\bar{U}|\bar{0}\rangle_{\mathcal{G}}), \quad (3.3)$$

with $U \in \mathcal{A}[\mathcal{G}]$ acting non-trivially only on the gauge subsystem. A Pauli operator is a logical operator for \mathcal{G} iff it commutes with all the elements of $\mathcal{S}(\mathcal{G})$ but is not in \mathcal{G} . It is always possible to choose representations of the logical operators which commute with all elements of \mathcal{G} , which we call *bare* logical operators L . Then, for a unitary V in the bare logical operator algebra, $V \in \mathcal{A}[L]$, only the logical state is affected,

$$V|\psi\rangle = (\bar{V}|\bar{\psi}\rangle_L) \otimes |\bar{0}\rangle_{\mathcal{G}}. \quad (3.4)$$

Since \mathcal{G} can be non-Abelian, multiplying bare logical operators by elements of \mathcal{G} will give dressed representations of the logical operators which may not commute with all the elements of \mathcal{G} . The distance $d(\mathcal{G})$ of a subsystem code is defined as follows

$$d(\mathcal{G}) = \min_{P \in \mathcal{N}(\mathcal{S}) \setminus \mathcal{G}} |P|. \quad (3.5)$$

Note that a stabilizer code can be viewed as a subsystem code whose gauge group is Abelian, i.e., $\mathcal{G} = \mathcal{S}(\mathcal{G})$.

Topological stabilizer codes [BK13; Kit03; BM06] are a special case of stabilizer codes, which have geometrically local stabilizer generators. We can think of qubits of a topological stabilizer code as being embedded in a manifold, e.g. a torus. Properties of the topological stabilizer code, such as its distance or the number of logical qubits, are related to the topology of the manifold. The most famous example of a topological stabilizer code is the toric code. It is defined on a 2D lattice, i.e. a tessellation of a 2-manifold with or without boundaries, by placing qubits on edges and identifying vertices and faces with stabilizer generators of X - and Z -type; see Fig. 3.1(a). Another example of a topological stabilizer code is the color code. An advantage of the 2D color code is that all single logical qubit Clifford gates can be implemented transversely, i.e., as a tensor product of single-qubit unitaries. The code is defined on a 2D lattice, which satisfies two constraints

- 3-valence: each vertex belongs to three edges,
- 3-colorability: faces can be colored in three colors $\{r, g, b\}$ in such a way that two neighboring faces are of different color.

Qubits are placed at vertices and the stabilizer group of the color code is generated by X - and Z -face operators, which are tensor products of Pauli X and Z operators on qubits on corresponding faces. In particular, we will be interested in the 2D triangular color code on the hexagonal lattices.

The color code construction can be generalized to d dimensions [BM07a]. In particular, the 3D color code is defined on a 4-valent lattice with 4-colorable volumes, i.e., two neighboring volumes have different colors. Qubits are placed on vertices and the stabilizer group is generated by X - and Z -type operators associated with 3- and 2-cells; see Fig. 3.1(c)(d). The 3D color code lattice we focus on is the dual lattice of the body-centered cubic (bcc) lattice. Two key properties [Bom15a; ADP14; KB15] of the 3D color code are that it (i) admits a transverse non-Clifford logical gate, $T = \text{diag}(1, e^{i\pi/4})$, and (ii) allows information transfer to the 2D color code by local measurements and Pauli corrections; see Fig. 3.1(c) and (d). The transverse T gate follows from features of the bipartite lattice. We describe the dimensional jump in more detail in Sec. 3.4, and refer the reader to Refs. [Bom15a; KB15] for thorough derivations of the transverse properties of these code families.

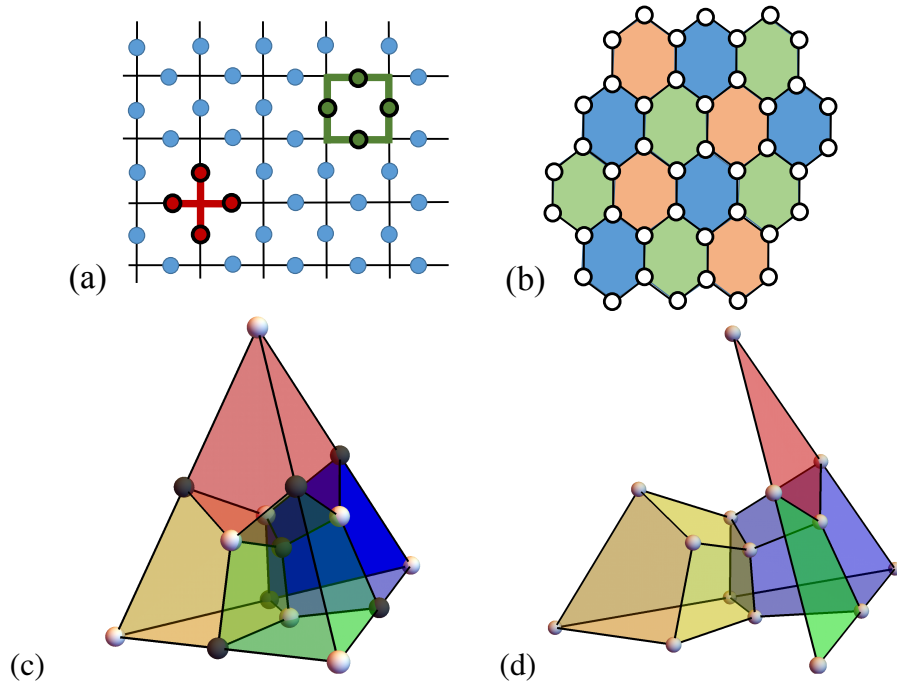


Figure 3.1: (a) The 2D toric code on a square lattice with qubits placed on the edges of the lattice. The X -vertex (respectively Z -face) stabilizers are products of four Pauli X (or Pauli Z) operators applied to qubits on edges incident at any given vertex (or on edges around any face), as indicated in red (green). (b) The hexagonal lattice is 3-valent and has 3-colorable faces. With qubits on vertices, the 2D color code's stabilizer generators are composed of either Pauli X or Z operators on all vertices for any given face. (c) The smallest 3D color code is the 15-qubit Reed-Muller code. Qubits are placed on the vertices of the 3D lattice, whose volumes are 4-colorable. The lattice consists of four volumes, one of which has been shaded for clarity. The 3D color code has a X -volume stabilizer supported on all the qubits of any volume in the lattice. For every face, there is a Z -face stabilizer supported on the qubits on the face. The lattice is bipartite, and to implement the logical \bar{T} , one applies T to the dark qubits and T^\dagger to the light qubits. (d) By measuring generators of the 2D color code, and applying appropriate Pauli operations, one can switch from the 3D to the 2D color code.

3.2 Error correction with stabilizer codes

Error correction can reliably protect quantum information from sufficiently weak (and not adversarial) noise. In this section, we outline how we model error correction using stabilizer codes.

Noise model

To make the analysis tractable, we need to make some simplifying assumptions about the noise present in the system. The depolarizing channel on a single- and

two-qubit density matrices $\rho^{(1)}$ and $\rho^{(2)}$ is defined as follows

$$\mathcal{E}_p^{(1)} : \rho^{(1)} \mapsto (1-p)\rho^{(1)} + \frac{p}{3} \sum_{P \in \{X,Y,Z\}} P\rho^{(1)}P, \quad (3.6)$$

$$\mathcal{E}_p^{(2)} : \rho^{(2)} \mapsto (1-p)\rho^{(2)} + \frac{p}{15} \sum_{\substack{P_1, P_2 \in \{I,X,Y,Z\} \\ P_1 \otimes P_2 \neq I \otimes I}} (P_1 \otimes P_2)\rho^{(2)}(P_1 \otimes P_2), \quad (3.7)$$

where the parameter p can be interpreted as the error probability. The depolarizing channel leaves a single-qubit state unaffected with probability $1-p$ and with probability $\frac{p}{3}$ applies one out of three errors X , Y or Z . Similarly, the depolarizing channel leaves a two-qubit state unaffected with probability $1-p$ and with probability $\frac{p}{15}$ applies one out of 15 non-trivial Pauli errors $P_1 \otimes P_2 \neq I$.

Now we explain how we model a noisy quantum circuit. We approximate every noisy gate, i.e., Pauli X , Y and Z operators, the Hadamard gate H , the phase gate T , the controlled-not gate CNOT, and the idle gate I , by an ideal gate followed by the depolarizing channel on qubits acted on by the gate; see Fig. 3.2. In particular, we fix the strength of the depolarizing channel for both single- and two-qubit gates to have the same error probability p . We model preparation of ancilla qubits in Z or X bases by ideal preparation followed by the bit-flip X or phase-flip Z errors with probability p , respectively. Similarly, we model single-qubit measurements in Z or X bases by ideal measurements followed by a flip of the classical bit representing the measurement outcome with probability p . We assume that every elementary operation, i.e., preparation of an ancilla qubit, implementation of a gate and measurement of a qubit, can be performed in one time step. These assumptions significantly simplify our analysis, however in real-life experiment the implementation time as well as the noise model and its strength heavily depend on the type of performed operation. For instance, entangling gates seem to be more noisy and take more time to execute than single-qubit gates.

Syndrome extraction

In order to diagnose errors in the system we have to perform stabilizer measurements. The set of all stabilizers which return -1 measurement outcome is called a syndrome. Any non-trivial syndrome indicates that some errors affected the system. We note that in a realistic setting we cannot measure stabilizers perfectly. Rather, we can only use noisy components which in principle can introduce even more errors into the system. Fortunately, reliable syndrome extraction is possible provided the strength and correlations of the noise are sufficiently weak.

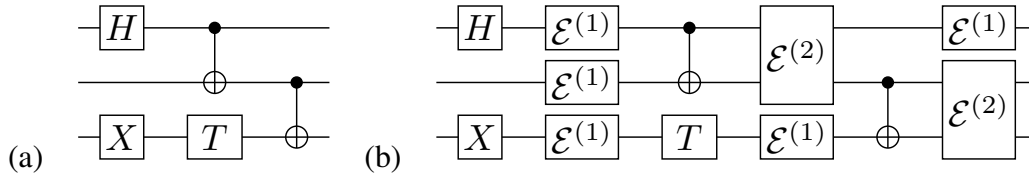


Figure 3.2: (a) An example of an ideal circuit on three qubits (idle gates not depicted). We assume that every gate, as well as preparation of ancilla qubits or single-qubit measurements take one time step. (b) We model a noisy circuit as composed of ideal gates followed by the depolarizing channel, with the probability of no error being $1 - p$ for both one- and two-qubit gates. We describe noisy preparation and measurement steps as ideal operations followed by the bit-flip X - or phase-flip Z -errors with probability p , depending on the choice of basis.



Figure 3.3: Ideal circuits used to measure weight-two stabilizers: (a) X -type and (b) Z -type. The bottom line represents an ancilla qubit, which is prepared in either $|0\rangle$ or $|+\rangle$ state. In our simulations we assume that every component of the circuits is affected by noise.

We model syndrome extraction with two quantum circuits measuring X - or Z -stabilizers, depicted in Fig. 3.3. We start by preparing an ancilla qubit in either the $|+\rangle$ or $|0\rangle$ state, then apply the CNOT gates between the ancilla qubit and all the qubits of the stabilizer, and finally measure the ancilla qubit in the corresponding X - or Z -basis. Since we focus our attention on a realistic scenario of syndrome extraction via noisy quantum circuits, we need to repeatedly measure each stabilizer in order to reliably infer the syndrome. We would like to emphasize that at each time step new errors can appear in the system and thus it is important to design a shallow circuit which does not propagate errors badly to extract the syndrome. When searching for such a circuit, we need to satisfy two restrictions:

- at each time step at most one operation can be applied to any given qubit,
- the circuit preserves the combined stabilizer group of the code and measurement ancilla qubits.

Decoding and thresholds

The problem of decoding can be stated as: *given the extracted stabilizer measurement syndrome, find and apply an appropriate correction.* Note that we do not need to identify the errors exactly: decoding succeeds if and only if our estimate of the error differs from the actual error by a stabilizer. For a given error model, in our case the depolarizing channel, optimal decoding finds the most probable equivalence class of errors consistent with the observed syndrome. In general, the complexity of optimal decoding can scale exponentially with the number of qubits and thus is computationally intractable in the regime of interest. However, one can consider efficient decoding strategies, such as the Minimum-Weight Perfect Matching algorithm for the 2D toric code which finds the most probable error. To decode the 2D color code we choose the projection decoder by Delfosse, which we describe in Sec. 3.5.

Let us consider a family of error-correcting codes of growing distance d , for instance the family of 2D triangular color codes. We say that a code family has a non-zero error-correction threshold p_c if for any physical error probability $p < p_c$ the probability of unsuccessful decoding $p_L(p, d)$ approaches 0 as the distance goes to infinity. If a code family has a non-zero threshold, then we can reliably store quantum information by first encoding it into a code of distance d from that family, and then performing repeated rounds of syndrome extraction followed by error correction. This way we can make the probability of logical error p_L , which we also call the effective error probability, to be arbitrarily small by choosing codes of sufficiently large distance d . For a given code family, the effective error probability p_L of the encoded qubits is a function of the physical error probability p and the distance d of the code and is expected to behave as

$$p_L(p, d) \approx \alpha \left(\frac{p}{p_c} \right)^{d/2}, \quad (3.8)$$

for p less than but comparable to p_c ; see [Fow+12; LAR11] for details. We will later use this heuristic scaling to estimate the probability of a logical error in the encoded information. For some choices of code family and decoding algorithm, there will be no finite error correction threshold. Although some codes could still be useful in particular regimes, usually we take the absence of a threshold to be a sign that an error correction scheme is not scalable.

We remark that the threshold for perfect syndrome extraction can be established by connecting to phase transitions in certain statistical-mechanical models. No

such connection exists for circuit-level noise however. One can prove lower bounds on circuit-level thresholds, however the bounds are typically quite loose. In order to estimate the circuit-level threshold, one typically uses numerical Monte Carlo simulations, see Sec. 3.5.

Universal computation

The Clifford group is the group of unitary gates which preserve the Pauli group when acting by conjugation. Gates in the Clifford group include X , Y , Z , S , H , and $CNOT$. We collectively refer to the set of all Clifford group elements, along with preparation of states and measurement in the Pauli basis as Clifford operations. A state is referred to as a stabilizer state if and only if it can be prepared by Clifford operations. The set of Clifford operations is not universal for quantum computation, and can be efficiently simulated on a classical computer [NC10]. However, the addition of any non-Clifford gate to the Clifford operations renders the resulting set universal, i.e., it generates a dense cover of the unitary group.

In most error correction schemes, the quantum information is stored in a stabilizer code, and many fault-tolerant gates are achieved by applying code-preserving transverse operations

$$U = U_1 \otimes U_2 \otimes \cdots \otimes U_n, \quad (3.9)$$

where each U_i is a unitary that acts on at most one qubit (in each code block). Transverse gates are automatically fault-tolerant as they do not spread errors. However, no quantum error correcting code can have a universal set of transverse gates [EK09; ZCC11; JKY18].

A wide range of schemes of practical interest are capable of fault-tolerantly applying all Clifford operations. To generate a non-Clifford gate to render such a scheme universal, a *resource state* (which cannot be formed by Clifford operations) can be used. For example, the so-called “magic state” $|T\rangle = T|+\rangle = (|0\rangle + e^{i\pi/4}|1\rangle)/\sqrt{2}$ can be used as in the circuit depicted in Fig. (3.4). There are a number of approaches to obtain the coveted fault-tolerant non-Clifford gate in such schemes. However, only two are known to be applicable to topological codes, magic state distillation and code switching, which we explain in Sec. 3.3 and Sec. 3.4, respectively.

Logical level circuit analysis

As we will discuss in Sec. 3.6, in order to distill magic states one needs to execute certain Clifford circuits. Due to the presence of the noise affecting physical qubits

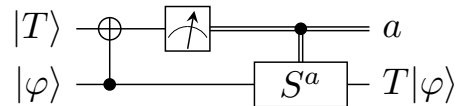


Figure 3.4: A T state $|T\rangle = (|0\rangle + e^{i\pi/4}|1\rangle)/\sqrt{2}$ can be used to implement the non-Clifford $T = \text{diag}(1, e^{i\pi/4})$ gate using Clifford operations. The measurement outcome a controls the application of the Clifford gate $S = \text{diag}(1, i)$ to yield the state $T|\varphi\rangle$ up to an unimportant global phase.

we would like to run those quantum circuits on the logical level. In other words, we require that every qubit involved in the quantum circuit is not just a physical qubit but a logical qubit of some code, say the 2D triangular color code of distance d . We choose the 2D color code because it can be used for both code switching (to the 3D color code) and distillation, by implementing logical Clifford gates transversely. We note that the error probability on the physical qubits in the system and the logical qubits is different. Namely, if p is the physical error probability, then according to Eq. (3.8) the effective error probability on the logical qubits is $p_L(p, d)$. To analyze the chances of successful implementation of the quantum circuit, we make a simplifying assumption that the effective noise in the circuit is captured by the depolarizing channel with error probability $p_L(p, d)$.

3.3 Non-Clifford gates from distillation

Although no encoded resource state $|\bar{R}\rangle$ can be generated using Clifford operations (which we will assume can be performed fault-tolerantly using the chosen error correction scheme), some number of encoded $|\bar{R}\rangle$ states with sufficiently weak noise (which could be produced by a non-fault-tolerant process) can be distilled by using only Clifford operations into fewer $|\bar{R}\rangle$ states but with improved fidelity [BK05]. The encoded resource state can then be used to fault-tolerantly implement a non-Clifford gate using a circuit such as that in Fig. 3.4. Any given state distillation protocol takes N_i copies of a resource state $|R\rangle$ affected by depolarizing noise with error probability p , and on average produces $N_f(p)$ copies with effective error probability $cp^m + \mathcal{O}(p^{m+1})$. The dependence of $N_f(p)$ on p is due to the fact that many protocols have a test which leads to the state being discarded upon failure. We also define $C(p) = N_i/N_f(p)$, the average number of input copies per output.

Here we outline three approaches to distill $|R\rangle$. To our knowledge, all known proposals can be viewed as variations of these three. Consider the distillation of a non-stabilizer resource state $|R\rangle$, which is stabilized by an operator $C_R|R\rangle = |R\rangle$,

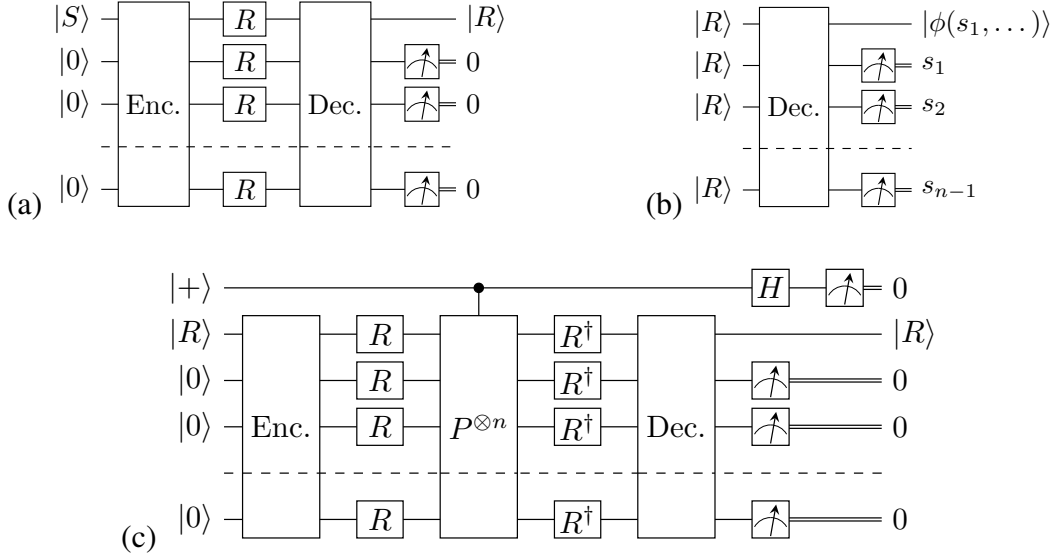


Figure 3.5: The three known classes of approaches to distill a resource state $|R\rangle$. The circuits are depicted for perfect input states, but noisy states would be used in practice, with post selection onto all zero measurement outcomes. *Enc.* and *Dec.* denote the (Clifford) encoding and decoding circuits. (a) A code with a transverse non-Clifford gate U_R is needed, such that $U_R|S\rangle = |R\rangle$ for a stabilizer state $|S\rangle$. (b) A code with a transverse gate C_R such that $C_R|R\rangle = |R\rangle$ is required, and the measurement of the stabilizers of the code will be probabilistic even when $|R\rangle$ input is free of noise, with post-selected state $|\varphi(0, \dots, 0)\rangle = |R\rangle$. (c) A code with a transverse gate C_R such that $C_R|R\rangle = |R\rangle$ is required, where $C_R = U_R P U_R^\dagger$ for a Pauli operator P .

and can be obtained from a stabilizer state $|S\rangle$ by the application of a non-Clifford gate U_R , i.e., $|R\rangle = U_R|S\rangle$. We will refer to C_R and U_R as the resource stabilizer and resource rotator, respectively. In all three approaches (see Fig. 3.5), each wire in the circuits represents a logical qubit of an error correction scheme that can fault-tolerantly perform all Clifford gates. For now we assume that these Clifford gates can therefore be performed perfectly, but we will relax this assumption later.

1. *Code projector then resource rotator* [LC13; BK05; BH12]. Find a code with a transverse logical $\overline{U_R}$ gate $\overline{U_R} = U_R \otimes U_R \otimes \dots \otimes U_R$. By preparing the code in the logical stabilizer state $|\overline{S}\rangle$, then using n copies of the noisy resource state $|R\rangle$ to implement the transverse gate as in Fig. 3.4, one prepares the (noisy) logical state $|\overline{R}\rangle$. By decoding $|\overline{R}\rangle$ and post-selecting on +1 outcomes of the code's stabilizers gives a distilled resource state with error probability reduced from p to $O(p^d)$ for code distance d ; see Fig. 3.5(a).

2. *Resource stabilizer then code projector* [BK05; Rei05]. Find a code with a transverse logical \overline{C}_R gate $\overline{C}_R = C_R \otimes C_R \otimes \cdots \otimes C_R$ as a transverse gate. Start with n noisy resource states $|R\rangle^{\otimes n}$, which by construction satisfies $\overline{C}_R |R\rangle^{\otimes n} = |R\rangle^{\otimes n}$, then measure the stabilizers of the code, with outcomes which are unaffected by the logical operator \overline{C}_R . If all outcomes are +1 then $|\bar{R}\rangle$ has been prepared, which can then be decoded to yield a distilled resource state $|R\rangle$ with error suppressed from p to $\mathcal{O}(p^d)$. The post-selection probability will not become unity given perfect resource states since the initial state is not in the code space, see right Fig. 3.5(b). This feature seems to render this approach less promising than the other two.

3. *Code projector then resource stabilizer* [Kni04b; Kni04a; MEK13; Jon13; DP15; CO16; Haa+17]. Consider a case in which $C_R |R\rangle = |R\rangle$ and $|R\rangle = U_R |S\rangle$, and $U_R C_R U_R^\dagger = P \in \mathcal{P}$ is a Pauli operator. Find a code which has $\overline{C}_R = C_R \otimes C_R \otimes \cdots \otimes C_R$ as a transverse gate. First encode one of the noisy $|R\rangle$ states in the code, giving $|\bar{R}\rangle$, plus noise terms. Then, construct a special gadget to measure the logical \overline{C}_R of the code, and post-select on the +1 outcome to reduce the contribution of the noise terms. The gadget involves first applying $U_R \otimes U_R \otimes \cdots \otimes U_R$, using n noisy $|R\rangle$ states as in Fig. 3.4, followed by the Clifford gate control- $(P \otimes P \otimes \cdots \otimes P)$, controlled by an ancilla state $|+\rangle$, before applying $U_R^\dagger \otimes U_R^\dagger \otimes \cdots \otimes U_R^\dagger$, using another n noisy $|R\rangle$ states. The resulting $|\bar{R}\rangle$ is decoded and kept as a distilled resource state only if the stabilizers of the code were all satisfied. Errors on the $|R\rangle$ states other than that uncoded are suppressed to the order of the code distance. Errors on the $|R\rangle$ that is encoded are not as suppressed, but can be boosted by (for example) repeating the measurement. See the lower panel of Fig. 3.5. ¹

Although for simplicity we have described each case with a code that encodes a single logical qubit $k = 1$, many of the best schemes make use of codes with $k > 1$ such that k/n is large, while maintaining a high post-selection success probability and non-trivial code distance d .

We will focus on the first type of distillation scheme from above. As a benchmark, consider 15-qubit code, which is defined in Fig. 3.1 and the surrounding text, where we explained the transverse implementation of the T gate.

¹It is worth pointing out that if R is a gate from the third level of the Clifford hierarchy such as T , then C_R will be in the Clifford group, but by using this approach with a code which implements a non-Clifford transverse gate C_R , distillation for higher order schemes should be possible.

To simplify the analysis of distillation protocols it is standard to imagine first applying a channel to diagonalize the state in the basis of the resource state. Continuing with the above notation, for a resource state $|R\rangle$ with a stabilizing operator C_R . Then, $C_R|R\rangle = |R\rangle$ and assume that $C_R|R^\perp\rangle = e^{\frac{2\pi i}{m}}|R^\perp\rangle$ for some integer m , and where $|R^\perp\rangle$ is orthogonal to $|R\rangle$. We can always write an arbitrary single qubit state as

$$\varrho = \varrho_{00}|R\rangle\langle R| + \varrho_{01}|R\rangle\langle R^\perp| + \varrho_{10}|R^\perp\rangle\langle R| + \varrho_{11}|R^\perp\rangle\langle R^\perp|. \quad (3.10)$$

The following twirling channel forces the density matrix into the simple form,

$$\begin{aligned} \varrho &\mapsto \sum_{k=1}^m C_R^k \varrho (C_R^k)^\dagger, \\ &= \sum_{k=1}^m \left(\varrho_{00}|R\rangle\langle R| + e^{\frac{2\pi i k}{m}} \varrho_{01}|R\rangle\langle R^\perp| + e^{-\frac{2\pi i k}{m}} \varrho_{10}|R^\perp\rangle\langle R| + \varrho_{11}|R^\perp\rangle\langle R^\perp| \right) \\ &= \varrho_{00}|R\rangle\langle R| + \varrho_{11}|R^\perp\rangle\langle R^\perp| = (1-p)|R\rangle\langle R| + p|R^\perp\rangle\langle R^\perp|, \end{aligned} \quad (3.11)$$

where $p = \langle R^\perp|\varrho|R^\perp\rangle$. Defining an error operator E which satisfies $|R^\perp\rangle = E|R\rangle$, we can represent the noise on a set of n such states as the binary vector $c \in \{0, 1\}^n$, where $E^{c_1} \otimes E^{c_2} \cdots \otimes E^{c_n}$, which occurs with probability $p^{|c|}(1-p)^{n-|c|}$ where $|c|$ is the bit string's Hamming weight. For example, for the state $|T\rangle$, the stabilizing operator is $HSHSH$ and the error operator is Z .

Code projector then resource rotator with the $[[15, 1, 3]]$ code

An encoding circuit for the 15-qubit code is shown in Fig. 3.6. For clarity, we have shown only one gate per time step, however the circuit length is reduced in practice by parallelizing. Almost all of the $CNOT$ gates in the encoding circuit commute. To implement the protocol, we set $|\psi\rangle = |+\rangle$, which allows the four $CNOT$ gates with the first qubit as the target to be removed.

If we assume the Clifford operations are perfect, then the only source of error is from the magic states themselves. Since the input is $\varrho = (1-p)|T\rangle\langle T| + pZ|T\rangle\langle T|Z$, we can think of the errors as being a Z -type Pauli inserted before the decoder in an otherwise perfect circuit from Fig. 3.6 with a probability $p^{|c|}(1-p)^{n-|c|}$ for c the bit string representing the support of the operator. We will obtain a false positive for this protocol iff the Z is a logical operator, in which case the output will be $Z|T\rangle$ instead of $|T\rangle$. As the code has distance $d = 3$ all weight one and two errors are detected. There are 35 weight 3 Z -type logical operators which each occur with probability $p^3(1-p)^{12}$, therefore the output is $\varrho' = (1-p')|T\rangle\langle T| + p'Z|T\rangle\langle T|Z$ with $p' = 35p^3 + O(p^4)$.

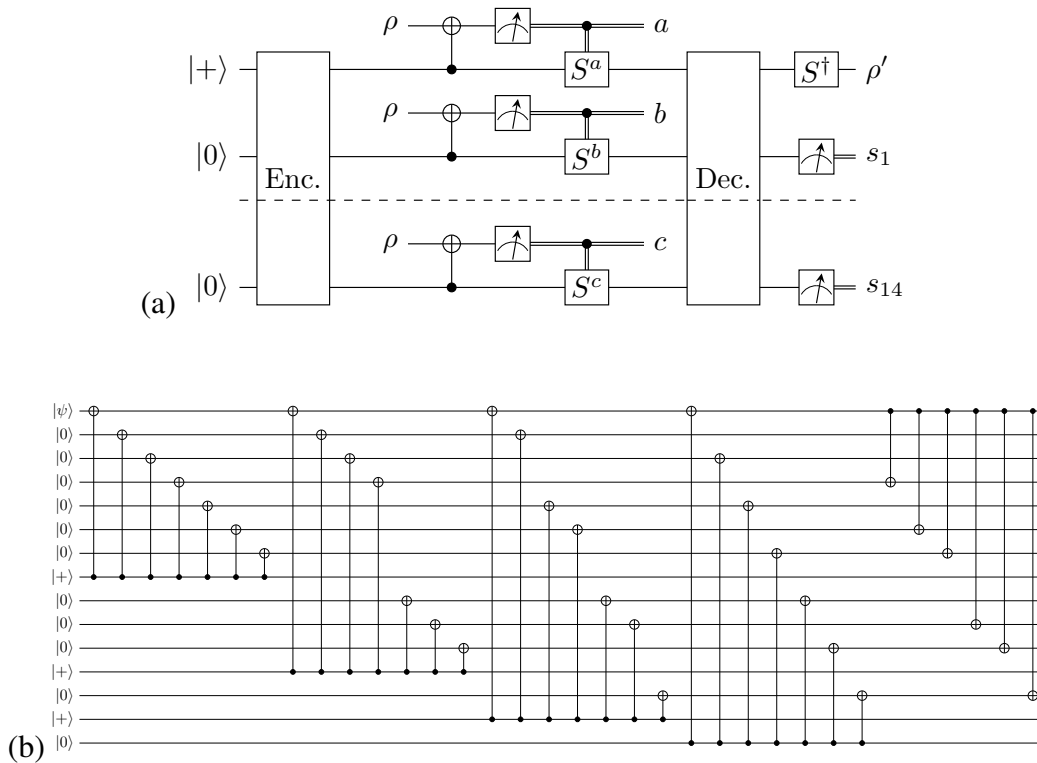


Figure 3.6: (a) A circuit to distill a $|T\rangle$ state using the fifteen qubit code. As the logical gate of the code is $\bar{T}^\dagger = T^{\otimes 15}$, the S^\dagger operation is required to return $|T\rangle = T|+\rangle$ rather than $T^\dagger|+\rangle$. The output ρ' is accepted if all of the stabilizers are satisfied (i.e. $s_i = 0$). If $\rho = (1 - p)|T\rangle\langle T| + pZ|T\rangle\langle T|Z$, then $\rho' = (1 - p')|T\rangle\langle T| + p'Z|T\rangle\langle T|Z$ with $p' = 35p^3 + \mathcal{O}(p^4)$. The transverse gate is implemented using $|T\rangle$ states using the gadget shown in Fig. 3.5. (b) A circuit *Enc.* which encodes the arbitrary state $|\psi\rangle$ into the 15 qubit code. To apply the decoding circuit *Dec.*, simply run *Enc.* in reverse.

3.4 Non-Clifford gates from code switching

If one can fault-tolerantly transfer information encoded in an error correction scheme which admits transverse Clifford operations to one which admits any transverse non-Clifford gate, a universal gate set can be achieved. Code switching is also known as gauge fixing or, in the context of topological codes of different spatial dimension, as dimensional jump. In this section we first give an explicit example of switching between the well-known Steane and 15-qubit Reed-Muller codes, which are the smallest 2D and 3D color codes. After a general description in terms of subsystem codes, we explain code switching in detail between arbitrarily large 2D and 3D color codes. Finally, we discuss the issue of fault tolerance of code switching.

Code switching between the 15-qubit and Steane codes

Suppose we begin with the density operator $\bar{\rho}_{\text{initial}}$ in the 15-qubit code space of Fig. 3.7(a). Our goal will be to end up with a final state encoded in the Steane code defined over those seven qubits making up the triangular red-green-blue facet in Fig. 3.7(a). The logical operators for both codes can be chosen to be $\bar{X} = X^{\otimes 7}$, $\bar{Y} = Y^{\otimes 7}$ and $\bar{Z} = Z^{\otimes 7}$ for all seven qubits in the red-green-blue facet. We can write the initial state as

$$\bar{\rho}_{\text{initial}} \propto c_0 \bar{I} + c_1 \bar{X} + c_2 \bar{Y} + c_3 \bar{Z}. \quad (3.14)$$

The Z type stabilizers of our target Steane code are already satisfied in the 15-qubit code. We force the X type stabilizers to be either +1 or -1 by simply measuring them. It is important that these measurements commute with $\bar{X}, \bar{Y}, \bar{Z}$. Any -1 outcome can be flipped to +1 by applying Z stabilizers of the original code, which commute with $\bar{X}, \bar{Y}, \bar{Z}$; see Fig. 3.7. The result is that all stabilizers of the Steane code are satisfied, giving a state $\bar{\rho}_{\text{final}}$ in the Steane code space. Moreover, since all the steps we took commute with \bar{X}, \bar{Y} and \bar{Z} , it must be that $\bar{\rho}_{\text{final}} = \bar{\rho}_{\text{initial}}$. An analogous procedure can be used to switch from the Steane code in Fig. 3.7(d) back to the 15-qubit code.

Code switching viewed as fixing a subsystem code's gauge state

Consider a subsystem code with the gauge group \mathcal{G} and the stabilizer group $\mathcal{S} = \mathcal{S}(\mathcal{G})$. Let two stabilizer codes \mathcal{S}_A and \mathcal{S}_B contain \mathcal{S} , i.e., $\mathcal{S} \subset \mathcal{S}_A$ and $\mathcal{S} \subset \mathcal{S}_B$. Assume that bare logical operators L for all three codes can be represented as the same operators. Note that any states $|\psi\rangle_A \in \mathcal{S}_A$ and $|\psi\rangle_B \in \mathcal{S}_B$ are also subsystem code states, i.e., $|\psi\rangle_A, |\psi\rangle_B \in \mathcal{G}$. Moreover, as L is common for all three codes, if $|\psi\rangle_A$ and $|\psi\rangle_B$ represent the same logical state in their respective stabilizer codes, then they must be logically equivalent in the subsystem code, namely

$$|\psi\rangle_A = |\bar{\psi}\rangle_L \otimes |\bar{\varphi}_A\rangle_{\mathcal{G}}, \quad (3.15)$$

$$|\psi\rangle_B = |\bar{\psi}\rangle_L \otimes |\bar{\varphi}_B\rangle_{\mathcal{G}}, \quad (3.16)$$

for different gauge states $|\bar{\varphi}_A\rangle_{\mathcal{G}}$ and $|\bar{\varphi}_B\rangle_{\mathcal{G}}$. We can therefore think about switching from \mathcal{S}_A to \mathcal{S}_B as fixing the gauge state of the subsystem code \mathcal{G} from $|\bar{\varphi}_A\rangle_{\mathcal{G}}$ to $|\bar{\varphi}_B\rangle_{\mathcal{G}}$. The process of code switching from \mathcal{S}_A to \mathcal{S}_B requires two steps.

1. Measure those generators of \mathcal{S}_B that are not in \mathcal{S} , which results in a syndrome σ .

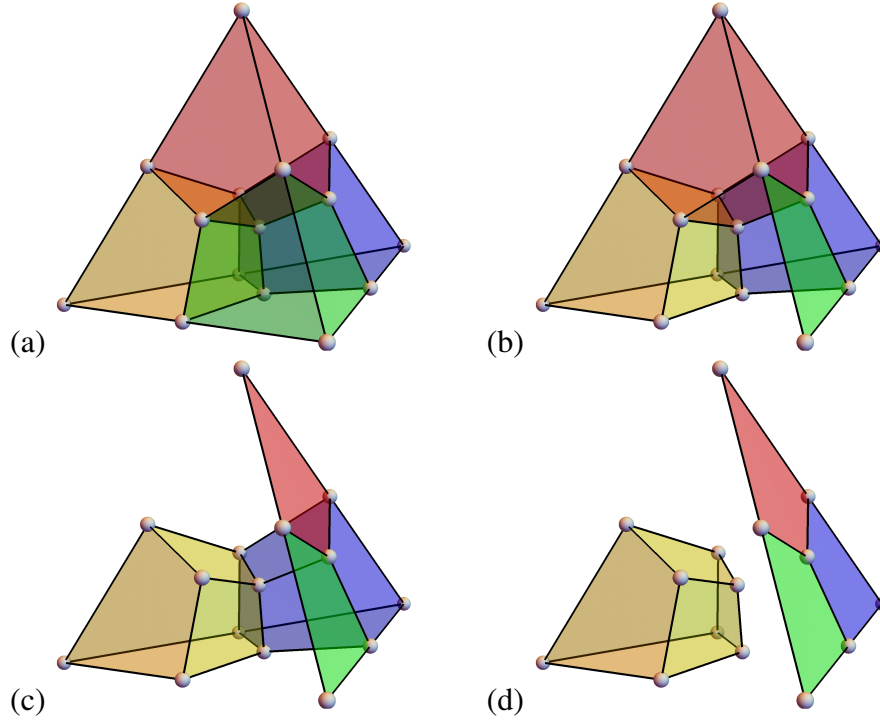


Figure 3.7: Conversion between code states $|\psi_{3D}\rangle$ and $|\psi_{2D}\rangle$ of the 3D and 2D color codes. (a) In the 3D code, X -stabilizers are supported on volumes, and Z -stabilizers on faces. (b) By measuring the X -face operator supported on the green face f_G on the rightmost boundary, we project $|\psi_{3D}\rangle$ into a $(+1)$ - or (-1) -eigenspace of $X(f_G)$. If the measurement outcome of $X(f_G)$ is -1 , then we apply an operator $Z(f_{RB})$ on the face f_{RB} separating the red and blue volumes and subsequently return to the $+1$ -eigenspace of $X(f_G)$. (c) We repeat the procedure and measure X -face operators for red and blue faces on the rightmost boundary, and conditioned on the measurement outcomes apply appropriate Z -face operators to ensure that the state is in the $+1$ -eigenspace of all measured X -face stabilizers. (d) The final state is a tensor product of some state $|\tilde{\psi}\rangle$ of eight qubits (yellow volume) and a code word $|\psi_{2D}\rangle$ of the 2D color code on seven qubits on the rightmost boundary. Since the logical operators of the two codes commute with all the X -face measurements and Z -face corrections in the conversion $|\psi_{3D}\rangle \mapsto |\tilde{\psi}\rangle \otimes |\psi_{2D}\rangle$, the unchanged encoded information is transferred from the 3D into the 2D color code.

2. In order to satisfy all the elements of \mathcal{S}_B , find and apply an element of \mathcal{G} which has the syndrome σ .

In order for code-switching to be fault-tolerant, we should reliably infer σ despite noise on the physical qubits along with errors in the measurements. We will return to this after describing a particular case of gauge-fixing: the dimensional jump for color codes.

Code switching between 2D and 3D color codes (dimensional jump)

In this case the relevant subsystem code is the 3D gauge color code. The gauge group \mathcal{G} is generated by all X and Z faces of the lattice \mathcal{L}_{3D} , which is dual to the bcc lattice. Note that X - and Z -type stabilizer generators of $\mathcal{S}(\mathcal{G})$ are identified with the volumes of \mathcal{L}_{3D} . Here, $\mathcal{S}_A = \mathcal{S}_{3D}$ is the stabilizer group for the 3D color code defined on \mathcal{L}_{3D} , and $\mathcal{S}_B = \langle \mathcal{S}_{\text{bulk}}, \mathcal{S}_{2D} \rangle$ is generated by two stabilizer groups on disjoint sets of qubits. In particular, \mathcal{S}_{2D} is the 2D color code stabilizer group on the boundary \mathcal{L}_{2D} of the lattice \mathcal{L}_{3D} , and $\mathcal{S}_{\text{bulk}}$ is a trivial stabilizer code (with no logical qubits) supported on $\mathcal{L}_{\text{bulk}} = \mathcal{L}_{3D} \setminus \mathcal{L}_{2D}$.

The generators of \mathcal{S}_{3D} are X -volumes, and Z -faces \mathcal{L}_{3D} . The generators of \mathcal{S}_{2D} are X -faces, and Z -faces in the facet \mathcal{L}_{2D} . The generators of $\mathcal{S}_{\text{bulk}}$ are the restriction of the generators of \mathcal{S}_{3D} to the bulk lattice $\mathcal{L}_{\text{bulk}}$, which amounts to all X -type 3-cells and Z -faces, along with additional X -faces along where the facet was removed. Finally, we can choose representatives of non-trivial logical operators $L = \{\bar{X}, \bar{Y}, \bar{Z}\}$ as products of Pauli X , Y and Z applied to all the qubits on the boundary \mathcal{L}_{2D} . We remark that those operators implement logical operators for both $\mathcal{S}_A = \mathcal{S}_{3D}$ and $\mathcal{S}_B = \mathcal{S}_{\text{bulk}} \cdot \mathcal{S}_{2D}$, as well as the bare logical operators for \mathcal{G} .

Following our general description from above, in order to implement a dimensional jump from the 2D to the 3D color code, we should have an initial state prepared in the 2D color code in the facet \mathcal{L}_{2D} , along with the qubits in $\mathcal{L}_{\text{bulk}}$ prepared in the unique code state of $\mathcal{S}_{\text{bulk}}$. Then, one should measure the Z -face operators near the facet. For any subset of these Z -face operators that return a -1 outcome, there will be a product of a set of X -face operators in \mathcal{L}_{2D} which anticommute with precisely that set and can therefore be applied to force all Z -face operators to have $+1$ outcome. The reverse process of a dimensional jump from 3D to 2D is very similar (and is depicted in Fig. 3.7).

Fault tolerance during code switching

Consider once more the example of switching from the 15-qubit code to the Steane code depicted in Fig. 3.7. We would like to argue that the process can be made fault-tolerant, such that a single error can occur during any location in the circuit used to implement the switching procedure without resulting in a logical error.

First imagine that the measurements are performed perfectly. Suppose that a single X type error appears on any of the 15 qubits at any stage in the process. This error will not alter the X -face measurements of the $r - g - b$ facet, and will be left as

an unpropagated X error at the end of the switching protocol, which (if acting on a qubit in the facet) will be corrected in the next round of error correction in the 2D code where our information is now stored. Therefore single X errors pose no problems for fault tolerance.

By the CSS nature of both the 2D and 3D color codes, it is sufficient to assume either a single Z error or no error at all occurs. A Z error can occur on a qubit either (1) in the facet, or (2) in the remaining 8 qubits of the bulk. Clearly, the above procedure fails for (1), since the Z will flip some the X -face measurements in the facet, causing us to apply the wrong Z -type gauge operators to ensure $+1$ X -face outcomes. To deal with this problem, we should identify the outcomes of the opposing X -faces of the bulk, along with the 3-cell X operator in the bulk (acting on all of its 8 qubits). Then, if the pairs of X -faces in the facet and their counterparts agree, we conclude that either no Z -error occurred, or the Z -error was in the bulk qubit furthest from the facet, and we proceed as if no error occurred. If the pairs of X -faces in the facet and their counterparts disagree, then we conclude that a Z error occurred either in the facet, or in one of the 7 adjacent qubits in the bulk. Moreover, the pattern of the disagreement in the pairs specifies which of the 7 pairs of qubits contain the Z error (but not whether the Z is on the facet or the bulk qubit of the pair). However, we can determine which one from the pair of Z operators was applied. Namely, if the X -volume measurement is -1 , then the Z operator was applied to the bulk. Therefore we have seen that the procedure can be made fault-tolerant (with perfect measurements) if we also measure X -stabilizers in the bulk. By implementing each of the measurements three times with verified cat states, and taking the majority vote, the entire process can be made fault-tolerant for faults in the measurement process.

A proof of quantum local fault-tolerance is given in [Bom14a] for code switching between the 2D color code and the 3D gauge color code. As usual in the topological setting, verified cat-states are not needed for measurements as the geometrically local measurement errors can be handled by the code (below a threshold error probability). We point out however that the T gate is not transverse in the 3D gauge color code. To our knowledge, there is no rigorous proof that that code-switching between the 2D and 3D stabilizer codes is fault-tolerant (with local quantum operations) for larger codes.

3.5 Error threshold analysis

As the toric and color codes are closely related [KYP15], one might hope to be able to reduce the problem of decoding of the color code to that of the toric code. This intuition is correct, as pointed out by Delfosse, who proposed the projection decoder for the 2D color code based on this strategy [Del14a]. In fact, it turns out that the color code in $d \geq 2$ dimensions can be decoded using any decoder of the corresponding d -dimensional toric code [KDP18]. In this section, we adapt the projection decoder for color codes on lattices with boundaries and imperfect syndrome extraction. We use this in numerical simulations to obtain the circuit-level threshold for the 2D triangular color code to be around 0.3%; see Fig. 3.8. These results are the first circuit-level noise analysis of the hexagonal lattice color code with an efficient decoder. Previous circuit-level noise analysis were performed for the square-octagon lattice color code, but with non-efficient [LAR11] decoders, or with a decoder that formally has no threshold [Ste14].

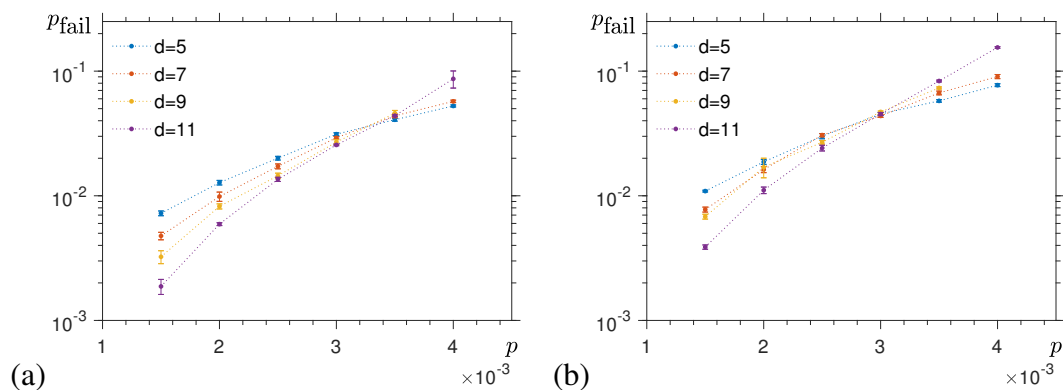


Figure 3.8: The failure probability $p_{\text{fail}}(p, d)$ of the 2D color code decoder from Ref. [Del14b] adapted to the case of the triangular color code with boundaries and noisy syndrome extraction. We can estimate the circuit-level thresholds for (a) the X -type and (b) Z -type correction by plotting $p_{\text{fail}}(p, d)$ for different d and finding their crossing point.

The (sublattice) projection decoder for the color code with boundaries

Let us consider the 2D color code on a lattice \mathcal{L} with boundaries, such as the 2D triangular color code. Since the color code is a self-dual CSS code, we will decode X - and Z -errors separately and focus on the correction of X -errors, since the correction of Z -errors is analogous. Assume that $\epsilon \subset \Delta_0(\mathcal{L})$ is the set of qubits affected by X -errors. Then, the syndrome $\sigma \subset \Delta_2(\mathcal{L})$ is the set of all faces neighboring odd number of errors; see Fig. 3.9(a) for illustration.

We construct the dual lattice \mathcal{L}^* , which has a vertex for every face of \mathcal{L} as well as one special vertex, called the boundary vertex, corresponding to all boundaries. Edges and faces of the dual lattice \mathcal{L}^* are obtained by replacing edges and vertices of \mathcal{L} , respectively. In particular, two vertices of \mathcal{L}^* are connected by an edge iff the corresponding two faces (or the face and the boundary) of \mathcal{L} share an edge. We note that a defining feature of a color code lattice is 3-colorability of its faces, and this implies 3-colorability of the vertices of the dual lattice \mathcal{L}^* . For every pair of colors $ij \in \{rg, rb, gb\}$ we define the projected lattice \mathcal{L}_{ij}^* to be composed of vertices of \mathcal{L}^* of color i or j and edges connecting them.

We now briefly discuss the projection decoder in the case of the lattice \mathcal{L} with boundaries and perfect syndrome extraction. The input of the decoder is the syndrome $\sigma \subset \Delta_2(\mathcal{L})$ and the output — an estimated error $\tilde{e} \subset \Delta_0(\mathcal{L})$ resulting in the syndrome σ . First we note that the syndrome σ corresponds to a subset of vertices $V \subset \Delta_0(\mathcal{L}^*)$. We call elements of V excitations and denote by V_{ij} the set of all excitations of color i or j . If the number of excitations in V_{ij} is odd, then we also include in V_{ij} the boundary vertex. Then, for every pair of colors ij we treat excitations in V_{ij} as the toric code excitations on the projected lattice \mathcal{L}_{ij}^* ; see Fig. 3.9(b)-(d). We can use any toric code decoder to find a suitable correction, i.e. a subset of edges $E_{ij} \subset \Delta_1(\mathcal{L}_{ij}^*)$ which corresponds to pairings of excitations in V_{ij} within the projected lattice \mathcal{L}_{ij}^* . In particular, we choose the Minimum-Weight Perfect Matching algorithm since it is an efficient toric code decoder. The last step of the decoder combines all three corrections for $ij \in \{rg, rb, gb\}$ and finds an estimated error \tilde{e} as a region enclosed by E_{rg} , E_{rb} and E_{gb} with minimum number of qubits in it; see Fig. 3.9(e). For a rigorous description of the projection decoder, see [Del14a; KDP18].

Adaptation of the (sublattice) projection decoder to noisy syndrome extraction

We are interested in decoding the 2D color code under circuit-level noise described in Sec. 3.2, i.e. we measure stabilizers with noisy quantum circuits. In order to reliably extract the syndrome and perform error correction one needs to repeat stabilizer measurements multiple times. Our input data consists of stabilizer measurements (possibly incorrect) at error correction (EC) steps labeled by integers and can be visualized as a (2+1)-dimensional lattice $\mathcal{K}^* = \mathcal{L}^* \times \mathbb{Z}$, where the extra dimension represents time; see Fig. 3.10. By an EC step, we simply mean a full cycle of stabilizer measurement circuits. Temporal edges, which vertically connect matching vertices in two copies of the 2D dual lattice \mathcal{L}^* at EC steps t and $t + 1$, correspond

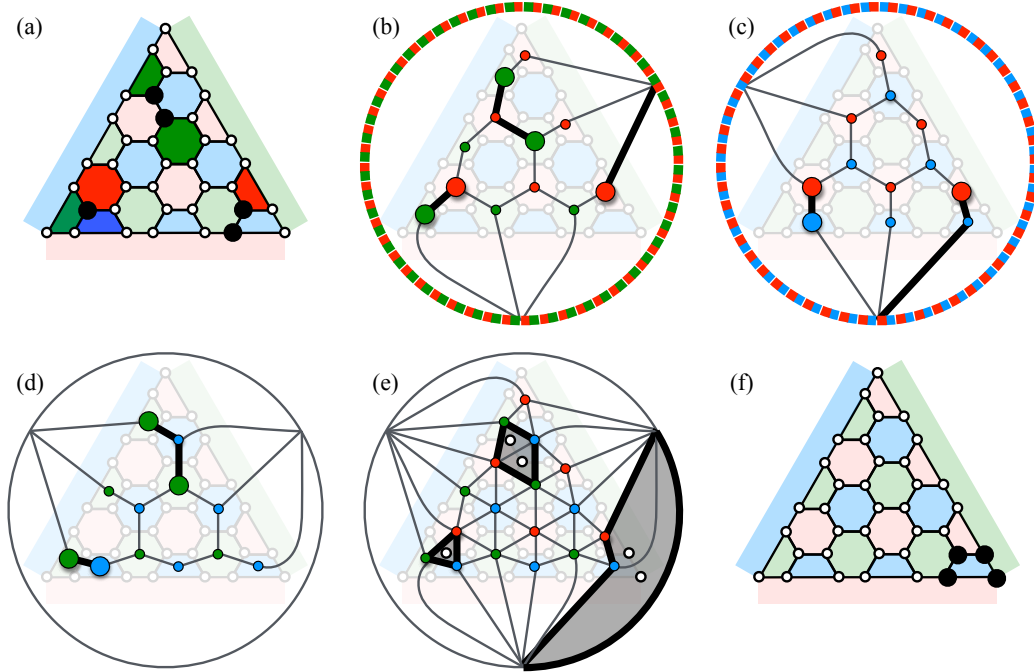


Figure 3.9: An illustration of how the projection decoder works. (a) The 2D triangular color code of distance $d = 7$ with errors $\epsilon \in \Delta_2(\mathcal{L})$ (black dots) and the corresponding syndrome $\sigma \in \Delta_0(\mathcal{L})$ (shaded faces). (b)-(d) Decoding on the projected lattices \mathcal{L}_{rg}^* , \mathcal{L}_{rb}^* and \mathcal{L}_{gb}^* , respectively. Note that in (b) and (c) (but not in (d)) the boundary vertex (depicted as the enclosing circle) is included in the set of excitations V_{rg} and V_{rb} . We find pairings of excitations which minimize the total length. (e) We find an error estimate $\tilde{\epsilon}$ as a region (shaded) with the minimal number of qubits (white dots), which is enclosed by combined corrections E_{rg} , E_{rb} and E_{gb} (thick black lines). (f) The decoding succeeds since the error ϵ combined with the estimate $\tilde{\epsilon}$ result in a stabilizer (black dots).

to stabilizer measurements at EC step t .

We propose a decoding strategy for the 2D color code with boundaries and noisy syndrome extraction, which is similar to the projection decoder we just described. For every pair of colors $ij \in \{rg, rb, gb\}$ we consider three (2+1)D lattices $\mathcal{K}_{ij}^* = \mathcal{L}_{ij}^* \times \mathbb{Z}$, each of which consists of copies of the 2D projected lattice \mathcal{L}_{ij} at different EC steps and corresponding vertices connected by temporal edges. Let $E^{(\sigma)}$ be a subset of temporal edges which correspond to violated stabilizers. Let $V^{(\sigma)}$ to be the set of vertices which are adjacent to exactly one edge in $E^{(\sigma)}$, i.e. elements of $V^{(\sigma)}$ are endpoints of segments from $E^{(\sigma)}$. We define $V_{ij}^{(\sigma)}$ be a subset of vertices of $V^{(\sigma)}$ of color i or j . The (sublattice) projection decoder with boundaries and noisy syndrome extraction consists of the following step.

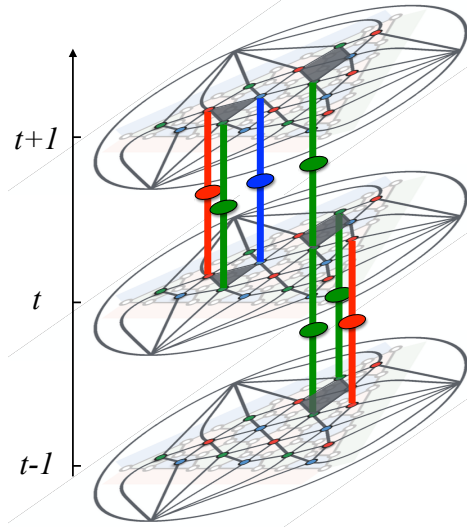


Figure 3.10: (2+1)D color code lattice. At each EC step, some errors might appear (shaded triangles). We find violated stabilizer (highlighted temporal edges) via noisy syndrome extraction. Notice that some stabilizer measurements are incorrect (the red and green at EC steps $t - 1$ and t , respectively)

1. Using the Minimum-Weight Perfect Matching find the pairing $E_{ij}^{(\sigma)}$ of elements in $V_{ij}^{(\sigma)}$ within the projected lattice \mathcal{K}_{ij}^* .
2. Combine obtained pairings, $\tilde{E} = E_{rg}^{(\sigma)} \sqcup E_{rb}^{(\sigma)} \sqcup E_{gb}^{(\sigma)}$ and decompose \tilde{E} as disjoint sum of maximal connected components, $\tilde{E} = \sqcup_i \epsilon_i$.
3. For every connected component ϵ_i :
 - find the minimal time window $[t_i^{(1)}, t_i^{(2)}]$ enclosing ϵ_i , i.e. $\epsilon \subset \mathcal{L}^* \times [t_i^{(1)}, t_i^{(2)}]$,
 - project ϵ_i onto the lattice \mathcal{L}^* at the EC step $t_i^{(2)}$ in order to obtain “flattened pairing” $\tilde{\epsilon}_i = \pi_i(\epsilon_i)$, where $\pi_i : \Delta_1(\mathcal{L}^* \times [t_i^{(1)}, t_i^{(2)}]) \rightarrow \Delta_1(\mathcal{L}^* \times \{t_i^{(2)}\})$ removes temporal edges and adds horizontal ones modulo two,
 - find a correction $\varphi_i \subset \Delta_2(\mathcal{L}^* \times \{t_i^{(2)}\})$ at EC step $t_i^{(2)}$ for $\tilde{\epsilon}_i$ as the minimal region enclosed by the “flattened pairing” $\tilde{\epsilon}_i$.

We make two additional technical remarks about the decoder. First, in step 1 we identify boundary vertices at all EC steps in \mathcal{K}_{ij}^* and if the number of vertices in $V_{ij}^{(\sigma)}$ is odd, then we include in $V_{ij}^{(\sigma)}$ the boundary vertex. Second, in step 2 we

distinguish boundary vertices at different EC steps and find connected components of \tilde{E} without temporal edges between boundary vertices.

We would like to establish the effective logical error probability $p_L(p, d)$ which describes the probability of logical information being changed in a single EC step. Assume that ϵ_t is the set of errors after EC step t . We define the cumulative error correction $\varphi_{\leq t}$ as all error corrections φ_i applied at EC steps $t_i^{(2)} \leq t$. Since syndrome extraction is noisy, we do not have a guarantee that at EC step t the cumulative error correction returns the encoded state to the code space. Most likely there is some residual noise $\varrho_t = \epsilon_t + \varphi_{\leq t}$ with the syndrome σ_t . We would obtain σ_t as the violated set of stabilizers if we could perform perfect syndrome extraction at time t . We define L_t as a logical operator which we would implement by decoding the syndrome σ_t of the residual error ϱ_t . This leads us to the effective logical error probability p_L to be proportional to the inverse of the average flip time. More explicitly, we write

$$p_L = \lim_{\Delta T \rightarrow \infty} \left(\frac{2}{\Delta T} \sum_{t=T}^{T+\Delta T} \text{flip}(t) \right)^{-1}, \quad (3.17)$$

where $\text{flip}(t)$ is the number of EC steps between the last logical flip and EC step t , and T is the EC step at which we begin to take data (which should be large enough such that the system reaches equilibrium). The factor of 2 in Eq. (3.17) ensures that if the system flips every k th EC step, then $p_L = 1/k$.

We remark that the “noisy projection decoder with boundaries” described here differs from the one discussed by Stephens [Ste14] in one important detail. Namely, we perform local “flattening” of pairings, which form a connected component. In contrast, Stephens flattens all pairings in the same time window of the length $O(d)$ at once and only then finds the correction. For any physical error probability p and for codes with sufficiently large code distance d the probability of successful correction is vanishingly small, i.e. the decoder does not have a non-zero threshold. Another remark is that in the discussion we assume that stabilizer measurements are performed infinitely many times at integer EC steps. In numerical simulations the number of EC steps $\Delta T \gg d^2$ is necessarily finite and we estimate the effective logical error probability as $\tilde{p}_{\text{eff}} = \left(\frac{2}{\Delta T} \sum_{t=T}^{T+\Delta T} \min\{\text{flip}(t), T + \Delta T - t\} \right)^{-1}$.

3.6 Overhead comparison

We would like to compare code switching with state distillation in the physically motivated setting of geometrically local operations on qubits arranged on the plane

(2D local operations). However the only known scalable example of topological code switching requires the 3D color code as described in Sec.3.4, which involves 3D local operations. To accommodate this in our setting we ask: *Consider a hybrid architecture mainly consisting of 2D quantum local regions for Clifford operations, along with special 3D quantum local regions only to produce resource states. Are fewer qubits required in the special 3D regions if used for code-switching or state distillation?* Although a special 3D region is not required for state distillation, a fair comparison with code-switching ought to permit the same physical operations to be used. Outside the special 3D region, we envisage an architecture based on the 2D color code, as this would connect straightforwardly to the 3D color code inside the special region via the dimensional jump technique described in Sec. 3.4. In 3.6 and 3.6 we explain how to estimate the number $N(p, p_{\text{target}})$ of physical qubits with error probability p required to produce an encoded resource state $|T\rangle$ (or a Clifford-equivalent state) of logical error probability p_L at most p_{target} . Then we use these techniques to compare various methods of code switching and distillation.

State distillation

We already mentioned that the Clifford distillation circuit is implemented at the logical level, i.e., every qubit in the circuit is a logical qubit of some base code of distance d . We assume that the noise affecting logical qubits is captured by the depolarizing noise of strength $p_L(p, d) \approx \alpha(p/p_c)^{d/2}$. The physical qubit overhead of every logical qubit in the distillation circuit is $N(d) \propto 3d^2/4$, since we choose the base code to be the 2D triangular color code. We denote by L_{dis} and N_R the number of locations in the distillation circuit and the number of required input resource states to produce one output state. Let the infidelity of each input resource state is q . If the output of the distillation circuit passes certain test, which occurs with probability $\text{Pr}_{\text{succ}}(q)$, then the infidelity of the output resource state is suppressed to $p_{\text{dis}}(q) = c \cdot q^m + O(q^{m+1})$, where c is some constant. We emphasize that in the discussion we neglect any noise introduced by the distillation circuit itself, which is a reasonable assumption if the circuit is implemented with the base code of sufficiently large distance.

We envision performing k rounds of distillation, in which the output resource state of one round is the input resource state into the next. Let us denote by $d^{(i)}$ and $p^{(i)}$ the distance of the base code and the infidelity of the input resource state $|\bar{T}\rangle$ in the i^{th} distillation round; see Fig. 3.11 for an illustration. We assume that we can grow the distance of the base code from $d^{(i)}$ to $d^{(i+1)}$ in between the rounds i and

$i + 1$ without introducing any additional errors. The error probability $p^{(i+1)}$ of the output of the i^{th} distillation round depends on the error probability $p^{(i)}$ of encoded magic states input into the i^{th} round, and on the physical error probability p and distance $d^{(i)}$ of the base code.

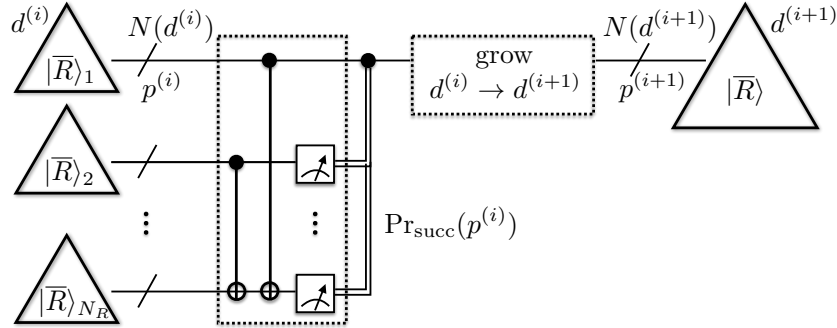


Figure 3.11: A schematic depiction of the i^{th} round of distillation. The input of the distillation circuit consists of N_R resource states $|\bar{R}\rangle$, each encoded in the base code of distance $d^{(i)}$ using $N(d^{(i)})$ physical qubits. The infidelity of input states is $p^{(i)}$. The distillation protocol succeeds with probability $\text{Pr}_{\text{succ}}(p^{(i)})$ and reduces the infidelity of the resource state to $p^{(i+1)}$. Then, the distance of the base code is increased from $d^{(i)}$ to $d^{(i+1)}$. The output resource state $|\bar{R}\rangle$ with parameters $(p^{(i+1)}, d^{(i+1)})$ is the input of the distillation round $i + 1$.

Before starting the first distillation round resource states are encoded in distance $d^{(1)}$ codes with error probability $p^{(1)}$, which depends on p . We require that the infidelity of the output resource state from the last round is below the target value, i.e., $p^{(k+1)} \leq p_{\text{target}}$. We want to find the distances $d^{(1)} < \dots < d^{(k)}$ of base codes used in distillation, which result in the minimum qubit overhead during the procedure. Note that the error probabilities $p^{(1)} > p^{(2)} > \dots > p^{(k+1)}$ have to decrease and satisfy the condition $p^{(i+1)} > a(p/b)^{d^{(i)}/2}$ for some constants a and b , which corresponds to the lower bound on the infidelity of the logical state in the path-counting regime.

Overhead given protocol parameters: We can estimate the average number of input states each round of distillation requires so that the last round returns one resource state with parameters $(p^{(k+1)}, d^{(k)})$. Since the distillation protocol in the last round succeeds with probability $\text{Pr}_{\text{succ}}(p^{(i)})$ thus on average we need $N_R/\text{Pr}_{\text{succ}}(p^{(k)})$ input resource states at the input of the last round, which in turn have to be successfully distilled in the round $k - 1$. In order to have $N_R/\text{Pr}_{\text{succ}}(p^{(k)})$ output states in the round $k - 1$, we need $N_R^2/(\text{Pr}_{\text{succ}}(p^{(k)})\text{Pr}_{\text{succ}}(p^{(k-1)}))$ resource states as the input of that round. In general, we need $N_R^{k+1-i}/(\prod_{j=i}^k \text{Pr}_{\text{succ}}(p^{(j)}))$ resource

states as the input of the i^{th} distillation round. Since every resource state needed in the i^{th} distillation round is encoded into the base code using $N(d^{(i)})$ physical qubits, thus the total number of physical qubits required at the start of that round is $N(d^{(i)})N_R^{k+1-i}/(\prod_{j=i}^k \text{Pr}_{\text{succ}}(p^{(j)}))$. Then, the qubit overhead of the distillation protocol can be found as the number of qubits needed in the most qubit-expensive distillation round, namely

$$N_{\text{dis}}(d^{(1)}, \dots, d^{(k)}; p^{(1)}, \dots, p^{(k)}) = \max_{i=1, \dots, k} N(d^{(i)})N_R^{k+1-i}/\left(\prod_{j=i}^k \text{Pr}_{\text{succ}}(p^{(j)})\right) \quad (3.18)$$

Estimating resource state error probabilities: First we consider how to estimate $p^{(1)}$, the error probability of inputs into the first distillation round. There are two main contributions to this error: (1) the encoding circuit using physical qubits with error probability p to encode the physical single-qubit state $|T\rangle$ into the smallest instance of the relevant code family², and (2) growing the code to reach the required distance $d^{(1)}$. We will neglect any errors added by code growing here and in what follows. As a fault has a probability p of occurring at any of the L_{en} locations in the encoding circuit, we estimate

$$p^{(1)} \approx L_{\text{en}} \cdot p. \quad (3.19)$$

To upper bound the infidelity $p^{(i+1)}$ of the output resource state of the i^{th} round, we note that the output has an error if an error occurs in the Clifford distillation circuit (occurring with probability $L_{\text{dis}} \cdot p_{\text{L}}(p, d^{(i)})$), or if there is an error associated with noisy inputs (occurring with probability $c \cdot (p^{(i)})^m$). We therefore obtain

$$p^{(i+1)} \leq 1 - \left[1 - L_{\text{dis}} \cdot p_{\text{L}}(p, d^{(i)})\right] \left[1 - c \cdot (p^{(i)})^m\right]. \quad (3.20)$$

and approximate $p^{(i)}$ by its upper bound.

We can sequentially estimate $p^{(1)}, \dots, p^{(k+1)}$ given the base code distances $d^{(1)}, \dots, d^{(k)}$ and the error rate p on the physical qubits. Finally, we find the distillation overhead $N_{\text{dis}}(p, p_{\text{target}})$ as the minimal qubit overhead for any possible k -round distillation scheme returning the output with infidelity below p_{target} , namely

$$N_{\text{dis}}(p, p_{\text{target}}) = \min_{k \in \mathbb{Z}_+} \min_{d^{(1)}, \dots, d^{(k)}} N_{\text{dis}}(d^{(1)}, \dots, d^{(k)}; p^{(1)}, \dots, p^{(k)}). \quad (3.21)$$

Hence, in order to estimate the overhead required for a distillation scheme, we need to know the distillation parameters L_{dis} , N_R , c , m , $\text{Pr}_{\text{succ}}(q)$, along with the properties of the underlying code $N(d)$, $p_{\text{L}}(p, d)$, L_{en} .

²For the color code on either the 4.8.8 or 6.6.6 lattice families, the smallest instance is the $[[7, 1, 3]]$ Steane code.

Code switching

Here we consider code switching involving three stabilizer codes \mathcal{S}_{2D} , \mathcal{S}_{3D} and $\mathcal{S}_{\text{bulk}}$, which we discussed in Sec. 3.4. Note that each of the first two codes encodes a single logical qubit and has distance d . The codes we consider require qubit overhead $N_{2D}(d)$, $N_{3D}(d)$ and $N_{\text{bulk}}(d) = N_{3D}(d) - N_{2D}(d)$, respectively. We take R_{2D} and R_{3D} to be the number of time steps in a single error correction round for \mathcal{S}_{2D} and \mathcal{S}_{3D} . Our analysis does not require that the codes referred to as 2D and 3D be implemented in two or three spatial dimensions, although in the cases of most interest they are. However, all Clifford operations should be transverse in \mathcal{S}_{2D} and \mathcal{S}_{3D} should admit a transverse T . Under equilibrium conditions with active error correction and circuit noise with error probability p , the logical error probabilities per time step are $p_{L,2D}(p, d)$ and $p_{L,3D}(p, d)$. We do not define $p_{L,\text{bulk}}$ since $\mathcal{S}_{\text{bulk}}$ encodes no logical qubits. We then assume code switching is implemented (under circuit noise with error probability p) with the following steps.

1. Over d error correction rounds (requiring $d \cdot R_{2D}$ time steps), an encoded state $|\overline{\mp}\rangle$ is prepared in \mathcal{S}_{2D} using $N_{2D}(d)$ qubits. The unique code state of $\mathcal{S}_{\text{bulk}}$ comes pre-prepared in $N_{\text{bulk}}(d)$ adjacent qubits, and is maintained throughout with error correction.
2. The generators of \mathcal{S}_{3D} are measured for d error correction rounds (requiring $d \cdot R_{3D}$ time steps), reliably inferring the subset of generators $\sigma_{\text{switch},3D} \in \mathcal{S}_{3D} \setminus \langle \mathcal{S}_{2D}, \mathcal{S}_{\text{bulk}} \rangle$.
3. The Pauli operator $E(\sigma_{\text{switch},3D}) \in \langle \mathcal{S}_{2D}, \mathcal{S}_{\text{bulk}} \rangle \setminus \mathcal{S}_{3D}$ which has the syndrome $\sigma_{\text{switch},3D}$ is applied. The system is now in the encoded state $|\overline{\mp}\rangle$ of \mathcal{S}_{3D} , up to residual noise. This takes no time steps since we can assume it is done off line by Pauli frame³ tracking.
4. The transverse \overline{T} gate of \mathcal{S}_{3D} is applied (in a single time step), resulting in the encoded resource state $|\overline{T}\rangle$ of \mathcal{S}_{3D} .
5. The generators of \mathcal{S}_{2D} and $\mathcal{S}_{\text{bulk}}$ are measured for d error correction rounds (requiring $d \cdot R_{2D}$ time steps), inferring the subset of generators $\sigma_{\text{switch},2D} \in \langle \mathcal{S}_{2D}, \mathcal{S}_{\text{bulk}} \rangle \setminus \mathcal{S}_{3D}$.

³Here Pauli tracking amounts to modifying the application of the transverse \overline{T} to be $E(\sigma_{\text{switch}})\overline{T}E(\sigma_{\text{switch}})^\dagger$.

6. The Pauli operator $F(\sigma_{\text{switch},2\text{D}}) \in \mathcal{S}_{3\text{D}} \setminus (\mathcal{S}_{2\text{D}} \cdot \mathcal{S}_{\text{bulk}})$ which has the syndrome $\sigma_{\text{switch},2\text{D}}$ is applied. This can be handled by Pauli frame tracking. The system is now in the encoded state $|\bar{T}\rangle$ of $\mathcal{S}_{2\text{D}}$, up to residual noise. The unique codestate of $\mathcal{S}_{\text{bulk}}$ is left in the $N_{\text{bulk}}(d)$ adjacent qubits, ready for the next use.

Now we explain $\sigma_{\text{switch},3\text{D}}$ which appears in the above steps. After the first round of these measurements, the system will already have been projected into a code equivalent to $\mathcal{S}_{3\text{D}}$ (for which $\sigma_{\text{switch},3\text{D}}$ of the generators of $\mathcal{S}_{3\text{D}}$ are flipped). There will also be residual Pauli noise P on the qubits, with an associated syndrome σ_P , and the reported measurement outcomes will be σ_{observed} such that

$$\sigma_{\text{observed}} = \sigma_{\text{switch},3\text{D}} + \sigma_P + \sigma_M, \quad (3.22)$$

with σ_M attributed to faulty measurements, and the fact that some of the Pauli operators in P were applied part-way through the measurement circuit. We assume that after d rounds, it is possible to reliably infer $\sigma_{\text{switch},3\text{D}}$, in addition to performing regular error correction. To our knowledge, it is an open question if this is the case. The explanation for $\sigma_{\text{switch},2\text{D}}$ is entirely analogous to that for $\sigma_{\text{switch},3\text{D}}$.

We expect $\mathcal{S}_{3\text{D}}$ to have a higher residual error rate affecting physical qubits than $\mathcal{S}_{2\text{D}}$. This can be modeled taking a first time step in (5) above with enhanced error probability $p_{\text{effective}}(p, d) > p$, followed by subsequent time steps with the usual error probability p . Taking the scheme to succeed only if no logical errors occur throughout the $d(2R_{2\text{D}} + R_{3\text{D}}) + 1$ time steps, we seek the smallest odd d for which the switching success probability

$$[1 - p_{\text{L},2\text{D}}(p_{\text{effective}}, d)] \times [1 - p_{\text{L},2\text{D}}(p, d)]^{2R_{2\text{D}} \cdot d - 1} \quad (3.23)$$

$$\times [1 - p_{\text{L},2\text{D}}(p, d)]^{R_{3\text{D}} \cdot d + 1} > 1 - p_{\text{target}}, \quad (3.24)$$

which we will call $d_{\text{min}}(p, p_{\text{target}})$. The overhead of code switching is then

$$N_{\text{sw}}(p, p_{\text{target}}) = N_{3\text{D}}(d_{\text{min}}(p, p_{\text{target}})). \quad (3.25)$$

Hence, in order to estimate the overhead required for code switching, we need to know $N_{2\text{D}}(d)$, $N_{3\text{D}}(d)$, $R_{2\text{D}}$, $R_{3\text{D}}$, $p_{\text{L},2\text{D}}(p, d)$, $p_{\text{L},3\text{D}}(p, d)$, $p_{\text{effective}}(p, d)$.

Plots and discussion

Here we describe the calculation in full detail using the 2D triangular color code on the hexagonal lattice with Bravyi-Kitaev distillation scheme. The well-known

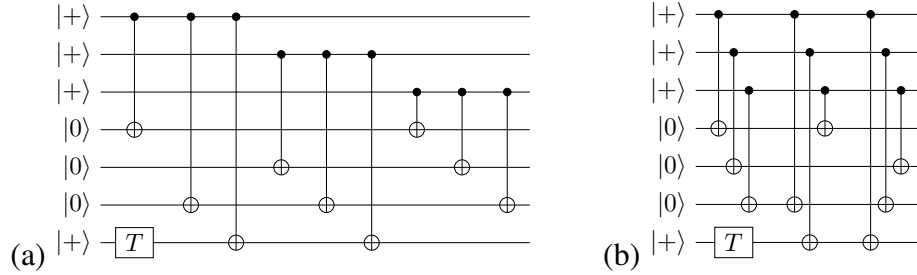


Figure 3.12: (a) Encoding circuit for Steane code. (b) A compressed version of the encoding circuit, which has a total of three time steps and seven qubits, giving a total circuit area of 21 in which a fault can occur.

parameters of this distillation scheme are $N_R = 15$, $c = 35$, $m = 3$, $\Pr_{\text{succ}}(q) = 1 - 15q$. We assume the distillation circuit in Fig. 3.6, which contains 34 CNOT gates, is implemented in a layered stack of 2D codes, with CNOT gates only possible between adjacent layers. We assume an average of $15/2$ swaps are needed to ensure the the correct layers are adjacent. Each swap requires 3 CNOT gates, therefore the total of $34 \cdot 15/2 \cdot 3 = 756$ logical gates are needed. As there are 15 logical qubits involved, we take the number of logical locations to be $L_{\text{dis}} = 15 \cdot 756 = 11475$.

For the triangular color code of distance d on the hexagonal lattice, the total number of vertices (corresponding to data qubits) is $|V| = 3(d^2 - 1)/4 + 1$, and the total number of faces (corresponding to two measurement qubits each) is $|F| = 3(d^2 - 1)/8$. Therefore, there is a total number of $N(d) = 3(d^2 - 1)/2 + 1$ qubits. From our threshold study in section 3.5, we take $p_L(p, d) = 0.03 \left(\frac{p}{0.003}\right)^{d/2}$. The encoding circuit into the Steane code can be performed in 3 time steps using 7 qubits. We therefore take $L_{\text{en}} = 21$; see Fig. 3.12.

Now we provide estimates of the required details for code switching between the triangular color code on the hexagonal lattice and the 3D color code on the dual-bcc lattice. Again, the overhead of the 2D color code is $N_{2D}(d) = 3(d^2 - 1)/2 + 1$. We take the length of the shortest measurement circuit for the 2D color code to be $R_{2D} = 7$. As above, $p_{L,2D}(p, d) = 0.03 \left(\frac{p}{0.003}\right)^{d/2}$. For the 3D code, there are $|V| = d(d^2 + 1)/2$ vertices (corresponding to data qubits), $|F| = 4d^3/3 + \mathcal{O}(d^2)$ faces (corresponding to Z measurement qubits), and $|C| = d^3/(12) + \mathcal{O}(d^2)$ volumes (corresponding to X measurement qubits). Each volume (except for the ones along the boundary of the lattice) contains 24 qubits, whereas the the faces either contain four or six qubits. For each volume, we can use either a single ancilla or multiple ancillas prepared in an unverified cat state to measure the qubits on the face belonging to that volume.

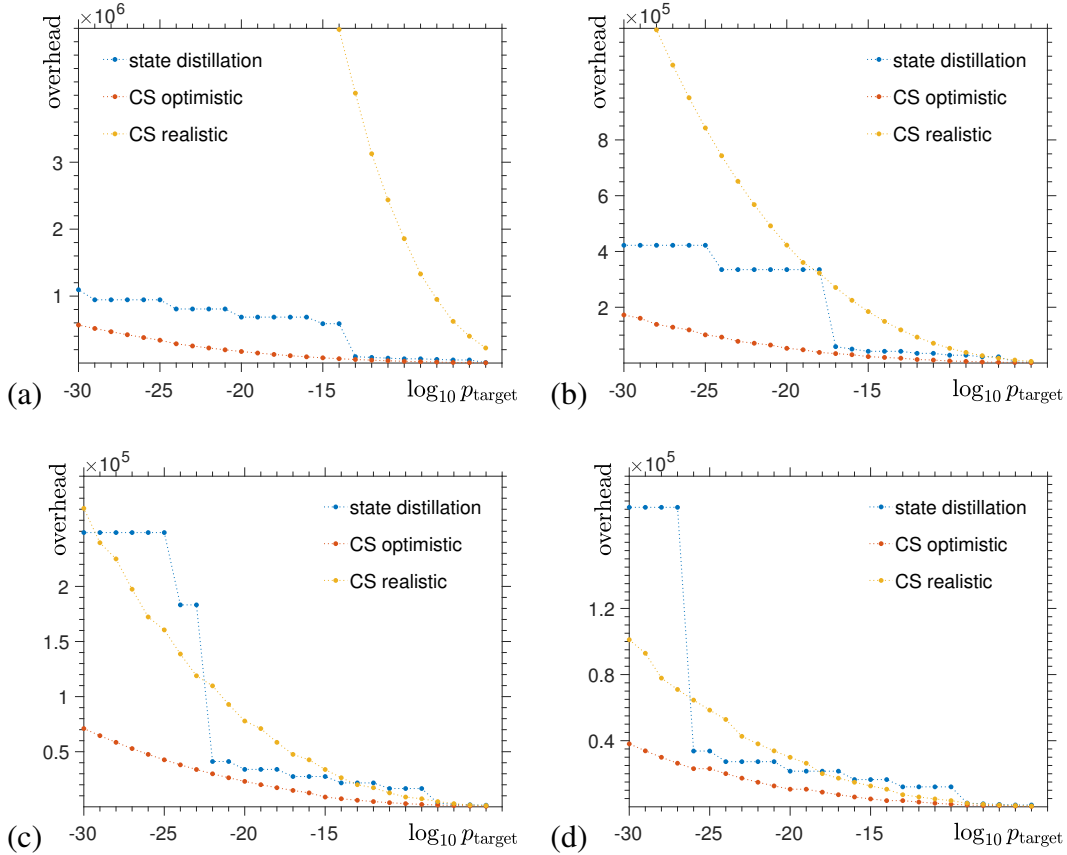


Figure 3.13: A comparison of the qubit overhead for state distillation (blue) and code switching in two scenarios: optimistic (red) and realistic (yellow). We set (a) $p = 0.0003$, (b) $p = 0.0001$, (c) $p = 0.00003$ and (d) $p = 0.00001$.

We imagine that four qubits are used per volume, and thus $R_{3D} = 24/4 = 6$ and we arrive at $N_{3D}(d) = |V| + |F| + 4|C| = 26d^3/12 + \mathcal{O}(d^2)$. Unfortunately, we do not know exactly what $p_{L,3D}(p, d)$ might be. Thus, we will consider two cases.

1. *Optimistic scenario:* To get an estimated lower bound of the code switching overhead, we can imagine very favorable properties for the 3D color code. We imagine that $p_{L,3D}(p, d) = p_{L,2D}(p, d)$, i.e., that the code performs as well as the 2D color code. If this were the case, then the 3D code would not be expected to result in significantly increased residual error after switching back to the 2D code, hence $p_{\text{effective}}(p, d) = p$.
2. *Realistic scenario:* It seems probable that the 3D color code has a very small circuit level threshold due to its high weight stabilizer generators. We know that the circuit-level threshold for the 6.6.6 color code is reduced by a factor of 35 compared to its threshold using an optimal decoder with perfect

measurements. Since the threshold for the 3D color code with an optimal decoder and ideal measurements is 0.02, the estimate for the circuit-level threshold we get is $p_{\text{th},3\text{D}} = 0.0006$. We take $p_{\text{L},3\text{D}}(p, d) = 0.005 \left(\frac{p}{0.0006}\right)^{d/2}$ and $p_{\text{effective}}(p, d) = 10p$.

We present the results of state distillation and code switching overhead comparison for different values of the target error probability p_{target} in Fig. 3.13. We notice that the code switching approach provides no advantage over a state distillation scheme in terms of the qubit overhead.

Chapter 4

LOCAL DECODERS FOR TOPOLOGICAL TORIC AND COLOR CODES IN ANY DIMENSIONS

Cellular automata are simple discrete models which can capture complex phenomena in the theory of computation, statistical physics and biology. An example of a cellular automaton is Toom's rule [Too80; Gri04], which describes the evolution of classical ± 1 spins placed on the faces of the 2D square lattice; see Fig. 4.1. At each time step, deterministic Toom's rule sets the value of the spin s_i to match the value of spins s_j and s_k on neighboring faces to the north and to the east if and only if these two neighboring spins are in the same state, i.e., $s_i \leftarrow \frac{1-s_j s_k}{2} s_i + \frac{1+s_j s_k}{2} s_j$. One can incorporate some non-deterministic element into the rule and then the resulting dynamics of the spin system is non-ergodic with the stationary state which (unlike in the Ising model) cannot be described by the Boltzmann distribution [BG85]. Yet another unexpected feature of the model is the existence of a non-zero measure region in its phase space where two stable phases can coexist [Too80]. Importantly, Toom's rule can be used to correct errors appearing in a three-dimensional cellular automaton that can simulate a universal Turing machine [GR88; Gác90; Gra01].

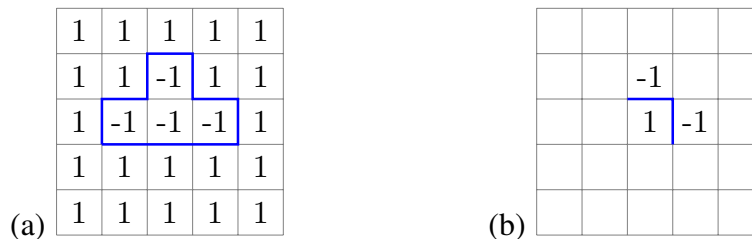


Figure 4.1: (a) The 2D square lattice with ± 1 spins on faces. The domain wall (blue) is a one-dimensional boundary which separates spins of different values. (b) Toom's rule sets the value of a spin s_i depending on the values s_j and s_k of its north and east neighbors. Namely, if s_j and s_k are the same, then set s_i to match s_j and s_k . Note that Toom's rule flips the spin s_i only if the north-east corner of that face (blue) contains the domain wall.

Let us discuss how deterministic Toom's rule can help with preserving the state of a classical memory in the presence of noise. For simplicity, we consider the 2D square lattice with periodic boundary conditions. We encode one bit of information in the system by setting all the spins to be either $+1$ or -1 . We would like to protect the

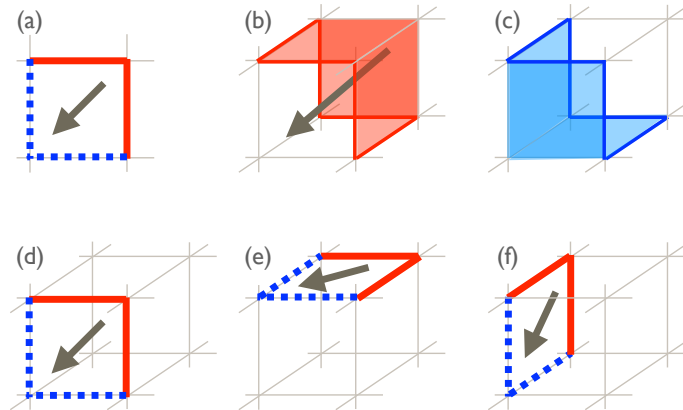


Figure 4.2: Toom's rule on the 2D square lattice (a) and the 3D cubic lattice (b)-(f). Edges and faces in red (solid) or blue (dashed) indicate configurations before and after the application of Toom's rule, respectively. For 1D loop-like domain walls in (a) and (d)-(f), the rule moves the specified corner of the domain wall (red two edges) in the direction of an arrow and results in two new edges (dashed blue). (b)-(c) For 2D surface-like domain walls, the rule moves the specified corner (red three faces) and results in three new faces (blue).

encoded bit against some random spin flips, $\pm 1 \mapsto \mp 1$. A straightforward approach is to check every spin, perform the majority vote and restore the spin values to the dominant value. However, collecting global information about the values of all the spins takes time during which more errors could occur. In order to avoid an accumulation of errors, it is natural to try to remove errors at every time step by implementing a local rule, such as Toom's rule. This provides a local error-correction method with performance almost as good as the optimal but non-local majority voting.

It is illuminating to think of Toom's rule as a local rule governing the movement of one-dimensional domain walls separating spins of different values. Namely, Toom's rule moves the north-east corner of the domain wall in the south-west direction by flipping a spin on the corresponding square face, see Fig. 4.2(a). Toom's rule can be generalized to the 3D cubic lattice (or in general to the hypercubic lattice in $d \geq 4$ dimensions) [BAS92]. In three dimensions, however, one could place spins on either cubic volumes or square faces of the lattice, which would result in two qualitatively different types of domain walls: 2D surface-like and 1D string-like. The local rules for how to move the domain walls on the 3D cubic lattice are illustrated in Fig. 4.2(b)-(f).

Unfortunately, a generalization of the Toom's rule beyond the 2D square lattice (or

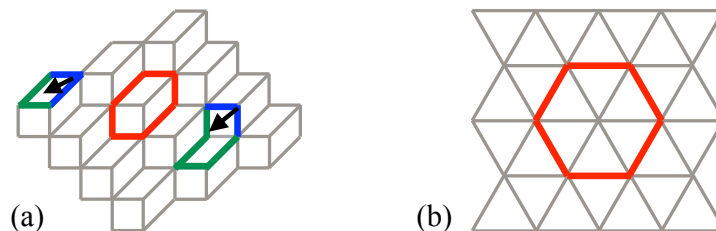


Figure 4.3: (a) A 2D lattice built of three different types of parallelograms. Toom’s rule can be viewed as a rule “*locally move NE corners of the domain wall in the SW direction*”. Namely, if the NE corner (blue) of a face supports a 1D domain wall (thick line), then flip the spin on that face (indicated by an arrow). Note that this simple rule fails to remove even small domain walls, such as the one depicted in red. (b) It is unclear whether a rule à la Toom can be introduced on triangulated lattices.

its higher-dimensional analogs) seems far from obvious. For instance, consider a 2D lattice built of three different types of parallelograms and choose two 1D domain walls as shown in Fig. 4.3(a). If one independently applies a simple rule “*locally move NE corners of faces containing the domain wall in the SW direction*” to every face of the lattice, then there exist configurations of loops which are not removed by such a rule. Moreover, it is not clear whether a rule à la Toom can be introduced on triangulated lattices, as exemplified in Fig. 4.3(b).

As suggested by Dennis et al. [Den+02], Toom’s rule can be used to protect quantum memory encoded into the topological quantum-error correcting code, the 4D toric code on the four-dimensional hypercubic lattice. Recent works [BT15; DBT17; PCC11] have numerically indicated that error correction schemes based on Toom’s rule may have a non-zero threshold p_c . In other words, correction of errors succeeds almost surely in the limit of infinite system size provided the rate of stochastic local errors is below p_c . These examples demonstrate the effectiveness of error correction with cellular automata, however fundamental reasons for the success of Toom’s rule as a decoder have remained poorly understood.

The main question we address in this chapter is how to define a local update rule à la Toom on any lattice and prove that an associated topological code decoder has a non-zero threshold. (i) We propose the Sweep Rule which shrinks k -dimensional domain walls on any reasonable lattice embedded in the d -dimensional Euclidean space for any $k \in \{1, \dots, d - 1\}$. In particular, the rule can be used on the d -dimensional torus. (ii) We design a local decoder, i.e. an error-correction scheme, for the d -dimensional toric code based on the Sweep Rule. (iii) We construct a new

class of local decoders of d -dimensional the color code. The two key components of our construction include a local reduction of the problem of color code decoding to that of the toric code and using the Sweep Rule to decode the resulting toric code. (iv) We obtain a lower bound on the performance of decoders for the toric and color codes based on the Sweep Rule. We also present numerical estimates of the thresholds of 3D topological codes. Our results provide a rigorous proof of a non-zero threshold, and thus clarify the success of Toom's rule as an error-correction method for topological stabilizer codes.

The chapter is structured as follows. In Section 4.1, we set the stage by introducing the notation, defining the Sweep Rule and showing its properties. Then, in Section 4.2 we discuss how to decode the toric code using the Sweep Rule. In Section 4.3 we prove that the Sweep Decoder based on the Sweep Rule has a non-zero threshold. Next, in Section 4.4 we explain how to decode the color code by using any toric code decoder and introduce a new class of local decoders of the color code. We finish in Section 4.5 by presenting results of the numerical simulations of proposed decoders and providing lower bounds on the color code threshold.

Description of the lattice

A d -dimensional lattice \mathcal{L} can be constructed by attaching d -dimensional cells to one another along their $(d - 1)$ -dimensional faces. We denote by $\Delta_k(\mathcal{L})$ the set of all k -cells of the lattice \mathcal{L} , where $k \in \{0, 1, \dots, d\}$. In particular, the sets $\Delta_0(\mathcal{L})$, $\Delta_1(\mathcal{L})$, $\Delta_2(\mathcal{L})$ and $\Delta_3(\mathcal{L})$ correspond to vertices, edges, two-dimensional faces and three-dimensional volumes of \mathcal{L} . We say that the lattice \mathcal{L} is a simplicial d -complex if for any k all of the k -cells of \mathcal{L} are just k -simplices; see [Gla72; Hat02].

In order to avoid technical difficulties we will be interested in lattices \mathcal{L} without boundaries which contain countably many cells and fill the d -dimensional Euclidean space \mathbb{R}^d . Moreover, each d -cell of \mathcal{L} is contained in some d -dimensional ball of constant radius. We remark that in the discussion of the toric and color codes we consider lattices \mathcal{L} , which are discretizations of the d -dimensional torus. However, we will be able to use the notions introduced in the rest of this section since the torus is locally Euclidean and we will only be interested in local properties of \mathcal{L} .

Now let us analyze the local structure of a d -dimensional lattice \mathcal{L} . Let $\kappa \in \Delta_k(\mathcal{L})$ be a k -simplex, where $k \in \{0, 1, \dots, d\}$. We abuse the notation and denote by $\Delta_l(\kappa)$ the set of all l -faces of κ , where $l \leq k$. The set of all n -simplices of \mathcal{L} which contain

κ as a k -face is called the n -star of κ and we denote it by

$$\text{St}_n(\kappa) = \{\nu \in \Delta_n(\mathcal{L}) \mid \kappa \in \Delta_k(\nu)\}. \quad (4.1)$$

We define $\text{Lk}_n(\kappa)$ to be the set of all n -simplices of \mathcal{L} which do not intersect with κ but belong to the same d -simplices as κ , i.e.,

$$\text{Lk}_n(\kappa) = \{\nu \in \Delta_n(\mathcal{L}) \mid \nu \cap \kappa = \emptyset \text{ and } \exists \delta \in \text{St}_d(\kappa) : \nu \in \Delta_n(\delta)\}. \quad (4.2)$$

We call $\text{Lk}_n(\kappa)$ the n -link of κ . We provide examples of stars and links in two and three dimensions in Fig. 4.4. It will also be useful to introduce the restriction $\sigma|_v$ of some subset of k -simplices $\sigma \subseteq \Delta_k(\mathcal{L})$ to the neighborhood of a vertex $v \in \Delta_0(\mathcal{L})$ as follows

$$\sigma|_v = \sigma \cap \text{St}_k(v). \quad (4.3)$$

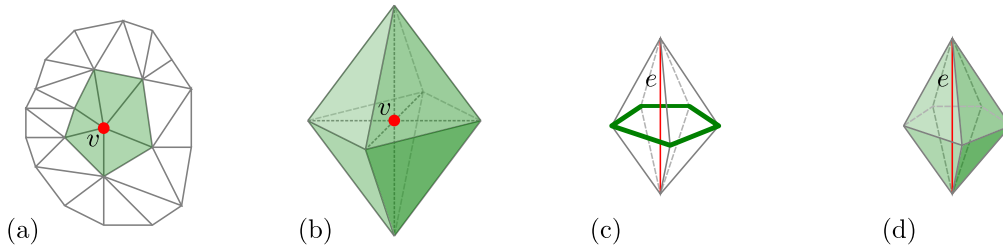


Figure 4.4: Examples of stars and links in 2D and 3D. In (a) and (b), we depict the 2-star $\text{St}_2(v)$ and the 3-star $\text{St}_3(v)$ of the vertex v (red), which correspond to the sets of six triangular faces and eight tetrahedra (shaded in green) containing v . (c) The 1-link $\text{Lk}_1(e)$ of the edge e (red) contains five edges (green), each of which belongs to the same tetrahedron as e but does not overlap with e . (d) The 3-star $\text{St}_3(e)$ of the edge e (red) corresponds to five tetrahedra (shaded in green) containing e .

A chain complex [Hat02] associated with a lattice \mathcal{L} is the underlying structure that makes possible the definition of the toric code in Sec. 4.2. Let us recall the definition of this structure. The role of the chain complex is to encode the relationship between any k -cell of the lattice \mathcal{L} and its $(k-1)$ -dimensional faces that belong to its boundary. Formally, the boundary of a k -cell κ is defined to be the set of all $(k-1)$ -faces of κ . Using the previous notations, it is the set $\Delta_{k-1}(\kappa)$. It is useful to extend the notion of a boundary and view it as a linear map in the following sense. Let the chain space $C_k(\mathcal{L})$ or simply C_k be the \mathbb{F}_2 -vector space with the set $\Delta_k(\mathcal{L})$ as a basis. The vectors of C_k are the formal sums of k -cells. Note that there is a one-to-one mapping between vectors in C_k and subsets of $\Delta_k(\mathcal{L})$, and thus we treat them interchangeably.

The boundary map $\partial_k : C_k \rightarrow C_{k-1}$ is a linear map specified for every basis element $\delta \in \Delta_k(\mathcal{L})$ by

$$\partial_k \kappa = \sum_{\nu \in \Delta_{k-1}(\kappa)} \nu. \quad (4.4)$$

The set of $d + 1$ chain spaces C_i for $i \in \{0, 1, \dots, d\}$ equipped with the boundary maps ∂_i is called a chain complex associated with \mathcal{L} and is denoted by

$$C_d \xrightarrow{\partial_d} C_{d-1} \xrightarrow{\partial_{d-1}} \dots \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0. \quad (4.5)$$

The defining property of a chain complex is the fact that the boundary of a boundary is empty, i.e., $\partial_i \circ \partial_{i+1} = 0$ for all $i \geq 1$. An element $\sigma \in \text{im } \partial_{k+1}$ is called a k -dimensional boundary. Later in the chapter we will also refer to σ as a k -dimensional domain wall. Moreover, an element of $\ker \partial_k$ is called a k -dimensional cycle. Note that every k -boundary is a k -cycle, but there might be a k -cycle which is not a k -boundary.

The structure of the chain complex in Eq. (4.5) is tightly connected with the boundary operator defined in Eq. (4.4). However, it will be convenient to introduce a notion of the generalized boundary operator $\partial_{k,n} : C_k \rightarrow C_n$ for all $k \neq n$ as a linear map defined on every basis element $\kappa \in \Delta_k(\mathcal{L})$ by

$$\partial_{k,n} \kappa = \begin{cases} \sum_{\sigma \in \Delta_n(\kappa)} \nu & \text{if } k > n, \\ \sum_{\sigma \in \text{St}_n(\kappa)} \nu & \text{if } k < n. \end{cases} \quad (4.6)$$

In this notation, $\partial_k = \partial_{k,k-1}$. We will keep in mind that in general we do not have a trivial composition of two generalized boundary operators, i.e., $\partial_{n,m} \circ \partial_{k,n} \neq 0$, unless the lattice has some extra structure or $k = n + 1 = m + 2$. Generalized boundary operators will be used in the discussion of the color code decoding in Sec. 4.4.

Partial order

Since the lattice \mathcal{L} contains countably many simplices and is locally embedded in the Euclidean space \mathbb{R}^d , we can find a vector $\vec{t} \in \mathbb{R}^d$ such that no edge $(u, w) \in \Delta_1(\mathcal{L})$ between two vertices u and w (treated as a vector from u to w in \mathbb{R}^d) is perpendicular to it, $\vec{t} \cdot (u, w) \neq 0$. We call the vector \vec{t} the sweep direction. We define a path $(u : w)$ between two vertices u and w of the lattice \mathcal{L} to be a set of edges $(v_1, v_2), (v_2, v_3), \dots, (v_{n-1}, v_n) \in \Delta_1(\mathcal{L})$, where $v_i \in \Delta_0(\mathcal{L})$ and $v_1 = u$, $v_n = w$. Since there could be many paths between u and w , the notation $(u : w)$ does not specify which path we consider. If the sign of the inner product $\vec{t} \cdot (v_i, v_{i+1})$

is the same for all edges in the path $(u : w)$, then we call the path causal and denote it by $(u \Downarrow w)$. We remark that any pair of the vertices of \mathcal{L} is connected by a path but there might not exist a causal path between them; see Fig. 4.5(a).

We observe that the sweep direction \vec{t} induces a binary relation \leq over the set of vertices $\Delta_0(\mathcal{L})$. Namely, we say that u precedes w , i.e., $u \leq w$ for $u, w \in \Delta_0(\mathcal{L})$, iff there exists a causal path $(u \Downarrow w)$ and $\vec{t} \cdot (v_i, v_{i+1}) > 0$ for any edge $(v_i, v_{i+1}) \in (u \Downarrow w)$. Thus, the set of vertices $\Delta_0(\mathcal{L})$ with the relation \leq forms a partially ordered set. Let $V \subset \Delta_0(\mathcal{L})$ be any finite subset of vertices. We require that there exist a unique infimum $\inf V$ (greatest lower bound) and a unique supremum $\sup V$ (least upper bound) of V .¹ Abusing the notation, we denote by $\inf \sigma$ and $\sup \sigma$ the infimum and supremum of the set of vertices $\Delta_0(\sigma)$, which belong to some subset σ of simplices of the lattice \mathcal{L} . We define the up-set $\Uparrow(v)$ and down-set $\Downarrow(v)$ of a vertex $v \in \Delta_0(\mathcal{L})$ as the collection of all simplices of \mathcal{L} whose vertices succeed and precede v , namely

$$\Uparrow(v) = \{\kappa \in \Delta_k(\mathcal{L}) \mid k \in \{0, 1, \dots, d\}, \forall u \in \Delta_0(\kappa) : v \leq u\}, \quad (4.7)$$

$$\Downarrow(v) = \{\kappa \in \Delta_k(\mathcal{L}) \mid k \in \{0, 1, \dots, d\}, \forall u \in \Delta_0(\kappa) : u \leq v\}. \quad (4.8)$$

For an introduction to ordered structures and partial orders, see e.g. [DP02].

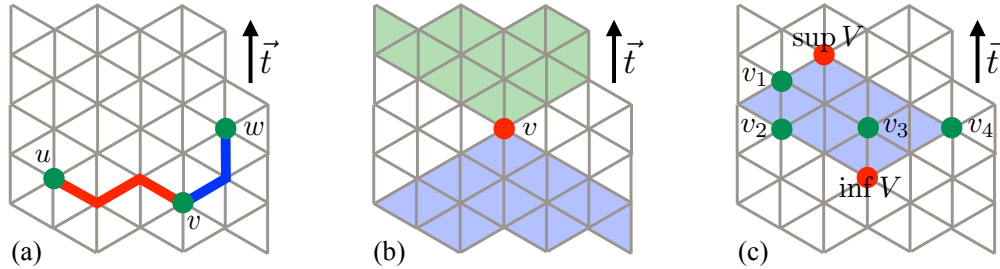


Figure 4.5: The sweep direction \vec{t} induces a partial order over the set of vertices of the triangular lattice \mathcal{L} . (a) Two vertices u and v are connected by a path $(u : v)$ (red), but there does not exist a causal path between them. Note that v and w are connected by a causal path $(u \Downarrow v)$ (blue). (b) The future light cone $\Uparrow(v)$ (shaded in green) and the past light cone $\Downarrow(v)$ (shaded in blue) of a vertex v . (c) The causal diamond $\diamond(V)$ (shaded in blue) of a subset of vertices $V = \{v_1, v_2, v_3, v_4\}$ can be found as the intersection of the future light cone of the infimum of V with the past light code of the supremum of V .

Informally, we can think of the partial order \leq between vertices of the lattice as a causality relation between points in the spacetime \mathbb{R}^d , where \vec{t} corresponds to the

¹This condition states that the partially ordered set $(\Delta_0(\mathcal{L}), \leq)$ is a lattice in the sense of order theory. However, we refrain from using this term in order to avoid confusion with the lattice \mathcal{L} defined as a simplicial complex. We also remark that the infimum and supremum of V are often denoted by $\bigwedge V$ and $\bigvee V$, respectively.

time direction. Then, we would say that u is in the past of v (or equivalently v is in the future of u) iff u precedes v , i.e., $u \leq v$ for $u, v \in \Delta_0(\mathcal{L})$. We could also treat the up-set $\uparrow(v)$ and down-set $\downarrow(v)$ as the future and past light cones of the vertex v , respectively. Lastly, we define a causal diamond $\diamond(V)$ of any finite subset of vertices $V \subset \Delta_0(\mathcal{L})$ to be the intersection of the future and past light cones of respectively the supremum and infimum of V , i.e.,

$$\diamond(V) = \uparrow(\inf V) \cap \downarrow(\sup V). \quad (4.9)$$

Note that uniqueness of the causal diamond of V is equivalent to uniqueness of the infimum and supremum of V . We illustrate the concepts of light cones and causal diamonds in Fig. 4.5(b)(c).

In the rest of the chapter we require that the lattice \mathcal{L} satisfy the following condition

- (local structure) for any vertex $v \in \Delta_0(\mathcal{L})$ and the k -dimensional boundary $\sigma \in \text{im } \partial_{k+1}$, if the restriction $\sigma|_v \subset \uparrow(v)$, then there exists a subset of $(k+1)$ -simplices $\varphi(v) \subset \text{St}_{k+1}(v) \cap \uparrow(v)$ in the future neighborhood of v satisfying (i) $(\partial_{k+1}\varphi(v))|_v = \sigma|_v$ and (ii) $\sup \varphi(v) = \sup \sigma|_v$.

In other words, the condition on the local structure of \mathcal{L} guarantees that if the restriction of any k -boundary $\sigma \in \text{im } \partial_{k+1}$ to the neighborhood of the vertex v is contained in the future light cone of v , then there exists a subset $\varphi(v)$ of $(k+1)$ -simplices in the future neighborhood of v whose restricted boundary $(\partial_{k+1}\varphi(v))|_v$ matches $\sigma|_v$ and the supremum of vertices in $\varphi(v)$ is the same as the supremum of vertices in $\sigma|_v$. Note that the latter condition happens to be equivalent to $\diamond(\varphi(v)) = \diamond(\sigma|_v)$.

The local structure condition, as well as uniqueness of the causal diamond of any finite subset of vertices $V \subset \Delta_0(\mathcal{L})$ can be easily checked for translationally-invariant lattices, such as the 3D bcc lattice depicted in Fig. 4.7(a). Indeed, let us identify the set of vertices of the body-centered cubic (bcc) lattice with the elements in $(2\mathbb{Z})^3 \cup (2\mathbb{Z} + 1)^3$ and choose the sweep direction to be $\vec{r} = (1, 1, 1) \in \mathbb{R}^3$. Then, we verify that the local structure condition is satisfied by exhaustively checking it for every possible choice of $\sigma|_v$. Moreover, we can explicitly find a unique infimum and supremum for any pair of vertices $u, v \in \Delta_0(\mathcal{L})$, which can be subsequently used to show by induction uniqueness of the infimum and supremum for any finite subset of vertices. Note that we will use the bcc lattice to study thresholds of the 3D color code in Sec. 4.5. For less regular lattices one might in principle need to

verify the local structure condition independently for every vertex of \mathcal{L} . However, we conjecture that it is satisfied by any finite homogeneous simplicial d -complex embedded in the Euclidean space \mathbb{R}^d with some fixed sweep direction $\vec{t} \in \mathbb{R}^d$.

Finally, we remark that the discussion of the partial order induced by the sweep direction \vec{t} translates into the setting when the lattice \mathcal{L} is defined on the d -dimensional torus. However, one has to exercise caution since in that case the partial order is well-defined only locally, i.e., within a d -dimensional ball of radius at most some fraction of the linear size of the torus. We will see in Sec. 4.3 that this technical point does not pose a problem in the proof of the threshold.

4.1 Beyond Toom's rule

As we have already seen, generalizing Toom's rule beyond the square lattice poses some challenge. In this section we introduce the Sweep Rule in $d \geq 2$ dimensions, which is a local update rule governing the dynamics of k -dimensional domain walls on any d -dimensional lattice for any $k \in \{1, \dots, d-1\}$. Here we consider lattices built of d -dimensional simplices. We remark that we focus on the combinatorial structure of the lattices since it is the key ingredient in the definition of topological quantum error-correcting codes.

Sweep Rule in 2D

We start by choosing a two-dimensional locally Euclidean lattice \mathcal{L} built of triangular faces and fixing the sweep direction $\vec{t} \in \mathbb{R}^2$. Recall that the sweep direction \vec{t} induces a partial order on the set of vertices $\Delta_0(\mathcal{L})$. We place a ± 1 spin on every triangular face of \mathcal{L} and denote by $\epsilon \subseteq \Delta_2(\mathcal{L})$ the set of all faces with -1 spins. We can find the boundary $\sigma = \partial_2 \epsilon$ of ϵ , which we also call the domain wall. We assume that we do not know ϵ , however we want to estimate it from the corresponding domain wall σ . We will see in Sec. 4.2 that ϵ and σ will be related to locations of errors and the observed syndrome in the toric code.

We are interested in a local update rule for the spin values, which can be simultaneously applied (possibly multiple times) to different local regions of the lattice \mathcal{L} and eventually allows us estimate ϵ . Let $\varrho \subseteq \Delta_2(\mathcal{L})$ denote a subset of faces, which keeps track of all the spins we have flipped. We perceive a rule which only uses local information about the domain wall σ to decide which spins to flip. By applying a rule in a local region, the values of some spins within that region are flipped, the domain wall σ is locally modified and the set of flipped spins ϱ is locally updated. Note that if at some point the domain wall disappears, namely $\sigma = 0$,

then ϱ provides an estimate of ϵ in the sense of having the same boundary as ϵ , i.e., $\partial_2 \varrho = \partial_2 \epsilon$.

Now we describe such a local rule, which we call the Sweep Rule. The Sweep Rule is defined for every vertex v of the lattice \mathcal{L} . Recall that we denote by $\sigma|_v$ the restriction of the domain wall σ to the edges incident to v . A vertex v is said to be extremal if $\sigma|_v$ is not empty and is contained in the future lightcone of v , namely $\sigma|_v \subset \uparrow(v)$; see Fig. 4.6. The basic idea behind the Sweep Rule is to identify a subset of faces $\varphi(v) \subseteq \text{St}_2(v)$ around the extremal vertex v , such that by flipping the values of spins on those faces we will move the domain wall σ away from v in the direction of \vec{t} . To be more precise, we find a subset of faces $\varphi(v) \subseteq \text{St}_2(v) \cap \uparrow(v)$ containing v and belonging to the future lightcone of v , such that the restriction of the boundary of $\varphi(v)$ locally matches the restriction of the domain wall σ , i.e., $(\partial_2 \varphi(v))|_v = \sigma|_v$. Then, the Sweep Rule flips spins on faces $\varphi(v)$ and updates the set of flipped spins $\varrho \leftarrow \varrho + \varphi(v)$. This results in the updated domain wall $\sigma \leftarrow \sigma + \partial_2 \varphi(v)$ being removed from the neighborhood of v . Note that the Sweep Rule does nothing if the vertex is not extremal. In Fig. 4.6 we illustrate three consecutive time steps when the Sweep Rule is simultaneously applied to every vertex of \mathcal{L} .

One can straightforwardly use the Sweep Rule to estimate the locations of -1 spins ϵ from the corresponding domain wall $\sigma = \partial_2 \epsilon$. Namely, (i) initialize $\varrho \leftarrow 0$, (ii) for every vertex $v \in \Delta_0(\mathcal{L})$ apply the Sweep Rule to find $\varphi(v)$ and update $\varrho \leftarrow \varrho + \varphi(v)$ and $\sigma \leftarrow \sigma + \partial_2 \varphi(v)$, (iii) repeat step (ii) until $\sigma = 0$. If the procedure terminates, then ϱ is a subset of faces whose boundary $\partial_2 \varrho$ matches $\partial_2 \epsilon$. In other words, by flipping the values of spins identified with the faces in ϱ we remove the initial domain wall $\partial_2 \epsilon$. Note that we are neither guaranteed that the procedure terminates nor to have $\varrho = \epsilon$.

We finally remark that in two dimensions, there is a unique choice of $\varphi(v)$ for the extremal vertex v . However, as we will see in the next subsection, in $d \geq 3$ dimensions there could be many possible $\varphi(v)$ with the boundary locally matching the domain wall. Thus, we will have to impose an extra condition on the choice of $\varphi(v)$, which, roughly speaking, guarantees that the local causal structure of the domain wall is preserved after flipping spins identified with $\varphi(v)$. This condition is automatically satisfied in two dimensions.

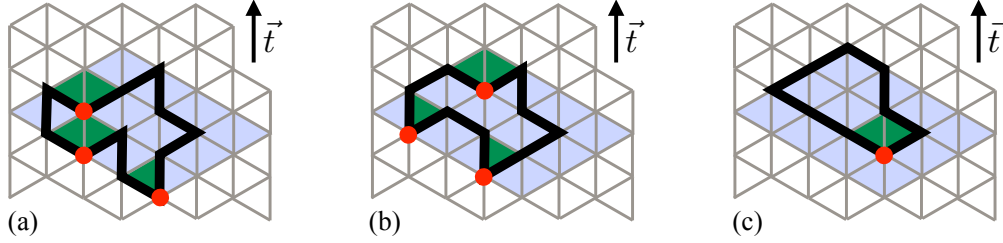


Figure 4.6: The action of the Sweep Rule in the two-dimensional triangular lattice. At each consecutive time step (a),(b) and (c), the Sweep Rule is applied to extremal vertices (red) of the domain wall σ (think black line). For each extremal vertex v we find a subset $\varphi(v)$ of neighbouring faces (shaded in green), which are in the future lightcone $\uparrow(v)$ of v and their boundary $\partial_2\varphi(v)$ locally matches the domain wall σ . By flipping spins on the faces in $\varphi(v)$ the domain wall is pushed away from v in the direction of \vec{t} . We note that the flipped spins as well as the domain wall always belong to the causal diamond $\diamond(\sigma)$ (shaded in blue) of the initial domain wall.

General Sweep Rule

Now we ready to generalize the Sweep Rule to be applicable to a d -dimensional lattice \mathcal{L} build of d -simplices, which is embedded in \mathbb{R}^d . We fix the sweep direction $\vec{t} \in \mathbb{R}^d$ as before, i.e., no edge is perpendicular to \vec{t} . First, we choose an integer $k \in \{1, \dots, d-1\}$ and place ± 1 spins on all $(k+1)$ -simplices of \mathcal{L} . Let $\epsilon \subseteq \Delta_{k+1}(\mathcal{L})$ be the set of all $(k+1)$ -simplices corresponding to -1 spins and $\sigma = \partial_{k+1}\epsilon$ be the corresponding domain wall, i.e., the k -dimensional boundary of ϵ . We can define a notion of an extremal vertex in the same way as in the two-dimensional case. Namely, a vertex v is extremal iff the restriction $\sigma|_v = \sigma \cap \text{St}_k(v)$ of the domain wall to the neighborhood of v is non-empty and is contained in the future lightcone $\uparrow(v)$ of v , i.e., $\sigma|_v \subseteq \text{St}_k(v) \cap \uparrow(v)$.

The Sweep Rule finds for every extremal vertex v a set of $(k+1)$ -simplices $\varphi(v) \subseteq \text{St}_{k+1}(v) \cap \uparrow(v)$ in the neighborhood of v , which are also contained in the future lightcone $\uparrow(v)$. The set of $(k+1)$ -simplices $\varphi(v)$ has to satisfy two conditions: (i) the boundary of $\varphi(v)$ locally matches the domain wall, i.e., $(\partial_{k+1}\varphi(v))|_v = \sigma|_v$, and (ii) the causal diamond of the restriction of the domain wall $\sigma|_v$ matches the causal diamond of $\varphi(v)$, i.e., $\diamond(\varphi(v)) = \diamond(\sigma|_v)$. The Sweep Rule can be succinctly formulated as follows.

As we claimed the Sweep Rule is local. In order to apply it to the vertex v , we only require the knowledge of the restriction $\sigma|_v$ of the domain wall and the set of $(k+1)$ -simplices $\text{St}_{k+1}(v) \cap \uparrow(v)$. The cardinality $|\text{St}_{k+1}(v) \cap \uparrow(v)|$ depends on the details of the lattice \mathcal{L} , but we are interested in cases where it is upper-bounded by

Algorithm 1: Sweep Rule

Require: a vertex v in the d -dimensional lattice \mathcal{L} , the sweep direction $\vec{t} \in \mathbb{R}^d$

Input: the k -dimensional domain wall $\sigma \in \text{im } \partial_{k+1}$ and the set of flipped spins $\varrho \subseteq \Delta_{k+1}(\mathcal{L})$

Output: locally updated σ and ϱ

find the restriction $\sigma|_v$ of the domain wall

if v is extremal, i.e., $\sigma|_v \subset \uparrow(v)$, then:

1. find a subset $\varphi(v) \subseteq \text{St}_{k+1}(v) \cap \uparrow(v)$, such that
 - (i) $(\partial_{k+1}\varphi(v))|_v = \sigma|_v$
 - (ii) $\diamond(\varphi(v)) = \diamond(\sigma|_v)$
2. locally update the domain wall $\sigma \leftarrow \sigma + \partial_{k+1}\varphi(v)$
3. locally update the set of flipped spins $\varrho \leftarrow \varrho + \varphi(v)$

return σ and ϱ

some constant. Thus, finding a subset $\varphi(v)$ can be done in constant time by checking all possible subsets of $\text{St}_{k+1}(v) \cap \uparrow(v)$ and finding the one satisfying conditions (i) and (ii). In some special cases, one can find $\varphi(v)$ more efficiently, as we discussed earlier in the context of two-dimensional lattices. Lastly, we comment why for any extremal vertex v and the domain wall σ we can always find $\varphi(v)$ satisfying conditions (i) and (ii). Recall that in the discussion of partial order in Sec. 4 we assumed that the lattice \mathcal{L} satisfies the local structure condition, which turns out to be equivalent to conditions (i) and (ii).

We remark that we can apply the Sweep Rule to any d -dimensional locally Euclidean lattice (not necessarily built of simplices) as long as it satisfies the local structure condition in Sec. 4. From that viewpoint, the Sweep Rule is a generalization of Toom's rule. For instance, if we choose the vector $\vec{t} = -(1, 1, 1) \in \mathbb{R}^3$, then the Sweep Rule on the 3D cubic lattice for one- and two-dimensional domain walls reduces to Toom's rule presented in Fig. 4.2(b)-(f).

The Sweep Rule governs the dynamics of the k -dimensional domain wall $\sigma \in \text{im } \partial_{k+1}$ in the lattice \mathcal{L} in the following sense. At each time step $n \geq 1$, we simultaneously apply the Sweep Rule to every vertex $v \in \Delta_0(\mathcal{L})$ and locally move the domain wall by flipping spins on the $(k+1)$ -simplices from $\varphi^{(n)}(v)$. Thus, the set of all spins flipped at time step n is equal to $\sum_{v \in \Delta_0(\mathcal{L})} \varphi^{(n)}(v)$, where we

set $\varphi^{(n)}(v) = 0$ if the vertex v is not extremal. Let us denote by $\sigma^{(n)}$ and $\varrho^{(n)}$ the updated domain wall and the set of flipped spins after n time steps, where we define $\sigma^{(0)} = \sigma$ and $\varrho^{(0)} = 0$. Then, we have $\varrho^{(n)} = \varrho^{(n-1)} + \sum_{v \in \Delta_0(\mathcal{L})} \varphi^{(n)}(v)$ and $\sigma^{(n)} = \sigma^{(n-1)} + \partial_{k+1}(\sum_{v \in \Delta_0(\mathcal{L})} \varphi^{(n)}(v))$. We can view the Sweep Rule as a prescription of how to move the domain wall by locally flipping spins in the neighborhood of every extremal vertex v so that the domain wall is pushed away from v in the direction specified by the sweep direction \vec{t} .

4.2 Decoding of the toric code

In this section, we describe decoding of CSS stabilizer codes, i.e., the problem of finding a suitable error-correction operator for errors in the system given the observed syndrome. We start by constructing a CSS chain complex for any CSS code. The formalism of chain complexes can not only be useful for decoding, but also exemplifies the intimate connection between the CSS codes and the systematic procedure of gauging and ungauging stabilizer symmetries [KY18]. We focus on the toric code, which is an example of a topological CSS code, and explain how to decode the 1D or higher-dimensional syndrome of the toric code in $d \geq 3$ dimensions using the Sweep Rule.

Decoding problem

Stabilizer codes are quantum error correcting codes [Sho95; Got96] which are at the heart of many fault-tolerant quantum computation schemes. A stabilizer code is specified by the stabilizer group \mathcal{S} , which is an Abelian subgroup of the Pauli group \mathcal{P}_n generated by tensor products of Pauli operators on n qubits. The code space associated with the stabilizer group \mathcal{S} is spanned by +1 eigenvectors of stabilizers $S \in \mathcal{S}$, and thus for the code space to be non-trivial we require $-I \notin \mathcal{S}$.

We focus our emphasis on CSS stabilizer codes [CS96; Cal+97], whose stabilizer group is generated by X - and Z -type stabilizer generators. A CSS code can be described by a CSS chain complex

$$\begin{array}{ccccc} D_2 & \xrightarrow{\tilde{\partial}_2} & D_1 & \xrightarrow{\tilde{\partial}_1} & D_0 \\ \text{Z-stabilizers} & & \text{qubits} & & \text{X-stabilizers} \end{array} \quad (4.10)$$

where D_2 , D_1 and D_0 are \mathbb{F}_2 -vector spaces with bases $\mathcal{B}_2 = Z$ -stabilizer generators, $\mathcal{B}_1 =$ physical qubits and $\mathcal{B}_0 = X$ -stabilizer generators, respectively. We choose linear maps $\tilde{\partial}_2$ and $\tilde{\partial}_1$, called the boundary operators, in such a way that

- the support of any Z -stabilizer $\omega \in D_2$ is given by $\tilde{\partial}_2\omega$,

- the X -type syndrome, which is the set of violated X -type stabilizer generators for some Z -error $\epsilon \in D_1$, can be found as $\tilde{\partial}_1 \epsilon$.

The boundary operators can be identified with the parity-check matrices H_Z^T and H_X of the CSS code. Note that the condition $\tilde{\partial}_1 \circ \tilde{\partial}_2 = 0$ is equivalent to the fact that any Z -stabilizer has a trivial X -syndrome. We can consider a dual chain complex

$$\begin{array}{ccccc} D_2^* & \xleftarrow{\tilde{\partial}_1^*} & D_1^* & \xleftarrow{\tilde{\partial}_0^*} & D_0^* \\ \text{Z-stabilizers} & & \text{qubits} & & \text{X-stabilizers} \end{array} \quad (4.11)$$

where dual vector spaces D_i^* are isomorphic to D_i , namely $D_i^* \cong D_i$ for $i = 0, 1, 2$, and the coboundary operators satisfy $\tilde{\partial}_1^* \circ \tilde{\partial}_0^* = 0$. Then, the support of any X -stabilizer $\omega^* \in D_0^*$ is $\tilde{\partial}_0^* \omega^*$, and Z -syndrome corresponding to any X -error $\epsilon^* \in D_1^*$ is given by $\tilde{\partial}_1^* \epsilon^*$.

Let us assume that X - and Z -errors appear at subsets of physical qubits $\epsilon^* \in D_1^*$ and $\epsilon \in D_1$, respectively. Since X - and Z -errors are diagnosed by measuring Z - and X -type stabilizers, we can choose to perform X - and Z -error correction separately, which further simplifies the discussion. To decode X - and Z -errors we need to guess from the observed syndrome of Z -type $\tilde{\partial}_1^* \epsilon^*$ and X -type $\tilde{\partial}_1 \epsilon$ which subsets of physical qubits $\varphi^* \in D_1^*$ and $\varphi \in D_1$ might have been affected by X - and Z -errors, respectively. Decoding succeeds iff the error and our guess differ by some stabilizer, namely there exist $\omega \in D_0^*$ and $\omega \in D_2$ such that $\epsilon^* + \varphi^* = \tilde{\partial}_0^* \omega^*$ and $\epsilon + \varphi = \tilde{\partial}_2 \omega$.

Toric code

Topological stabilizer codes [Kit03; BK98; BM06; BM07a; Bom13; FM01] are a special class of CSS stabilizer codes, which have geometrically local generators of the stabilizer group \mathcal{S} . One of the most studied examples of topological codes is the toric code [Kit03]. The d -dimensional toric code of type $k \in \{1, \dots, d-1\}$ can be defined on a d -dimensional lattice \mathcal{L} built of d -dimensional cells. We place one qubit at every k -cell κ in \mathcal{L} . For every $(k-1)$ -cell μ and $(k+1)$ -cell ν we define X - and Z -stabilizer generators $S_X(\mu)$ and $S_Z(\nu)$ to be the product of either Pauli X or Z operators on qubits in the neighborhood of μ and ν , namely

$$S_X(\mu) = \prod_{\kappa \in \text{St}_k(\mu)} X(\kappa), \quad S_Z(\nu) = \prod_{\kappa \in \Delta_k(\nu)} Z(\kappa). \quad (4.12)$$

The logical X - and Z -operators of the d -dimensional toric code can be chosen as Pauli X and Z operators, whose support forms $(d-k)$ - and k -dimensional objects.

The X - and Z -syndromes can be thought of as $(k - 1)$ - and $(d - k - 1)$ -objects, which we call excitations.² Note that the CSS chain complex associated with the toric code can be obtained by setting in Eq. (5.3) the vector spaces $D_2 = C_{k+1}$, $D_1 = C_k$, $D_0 = C_{k-1}$ and defining the boundary and coboundary operators as follows $\tilde{\partial}_2 = \partial_{k+1,k}$, $\tilde{\partial}_1 = \partial_{k,k-1}$, $\tilde{\partial}_0^* = \partial_{k-1,k}$, $\tilde{\partial}_1^* = \partial_{k,k+1}$. To summarize

$$\begin{array}{ccccc} C_{k+1} & \xrightarrow{\partial_{k+1,k}} & C_k & \xrightarrow{\partial_{k,k-1}} & C_{k-1} \\ Z\text{-stabilizers} & & \text{qubits} & & X\text{-stabilizers} \end{array} \quad (4.13)$$

To illustrate the toric code construction, let us analyze the three-dimensional case. The 3D toric code of type $k = 1$ is obtained by placing qubits on edges of a three-dimensional lattice \mathcal{L} and choosing X - and Z -stabilizer generators for every vertex v and face f of \mathcal{L} to be supported on qubits adjacent to v and f , namely

$$S_X(v) = \prod_{e \in \text{St}_1(v)} X(v), \quad S_Z(f) = \prod_{e \in \Delta_1(f)} Z(v). \quad (4.14)$$

The logical X and Z operators form 2D sheet-like and 1D string-like objects, whereas X - and Z -syndromes can be viewed as 0D point-like and 1D loop-like excitations.

Most likely error decoding of the toric code

The problem of decoding the 2D toric code has been extensively studied [Den+02; Fow+12; DP13; Har04; BH13; MKJ18], especially in the case of uncorrelated errors. We assume that each qubit is affected independently with the same probability p by X - and Z -errors, i.e., we consider the bit- and phase-flip noise models. For the sake of concreteness let us focus on correcting of Z -errors, since X -errors can be handled in a similar way. One of the best 2D toric code decoders is based on the Minimum-Weight Perfect Matching algorithm (MWPM), which finds the most likely Z -error consistent with the observed X -syndrome. The complexity of the MWPM is polynomial in the number of lattice constituents. The MWPM can only handle 0D point-like excitations but in $d \geq 3$ dimensions the toric code of type k on the lattice \mathcal{L} always has one- or higher-dimensional excitations, since either $k - 1 \geq 1$ or $d - k - 1 \geq 1$. Thus, one needs a generalization of the MWPM, which we call the k -Minimum-Weight Filling problem (MWF): for any k -dimensional boundary $\sigma \in \text{im } \partial_{k+1}$ find the smallest subset $\varrho \subseteq \Delta_{k+1}(\mathcal{L})$ of $(k + 1)$ -simplices of \mathcal{L} whose

²Observe that the X -syndrome can be thought of as the $(k - 1)$ -dimensional domain wall in the lattice \mathcal{L} . On the other hand, the Z -syndrome can be viewed as the $(d - k - 1)$ -dimensional domain wall in the dual lattice \mathcal{L}^* .

k -boundary matches σ . We can succinctly write

$$k\text{-MWF}(\sigma) = \arg \min_{\substack{\varrho \subseteq \Delta_{k+1}(\mathcal{L}) \\ \partial_{k+1} \varrho = \sigma}} |\varrho|. \quad (4.15)$$

Unfortunately, it is not known how to efficiently implement a solution to the k -MWF unless $k \in \{0, d-2, d-1\}$. We remark that the case of $k=0$ corresponds to the MWPM algorithm, and the case of $k=d-1$ is equivalent to finding “inside” and “outside” of the region in d dimensions enclosed by a $(d-1)$ -dimensional closed manifold. The case of $k=d-2$ was solved by Sullivan [Sul90] using an approach based on the max-flow min-cut theorem.

Local toric code decoder based on the Sweep Rule

We adopt a different approach to decoding the d -dimensional toric code (of type $k \neq 1$). Instead of most-likely error decoding, we introduce the Sweep Decoder based on the Sweep Rule. The Sweep Decoder is a local decoder (unlike the MWPM) which attempts to find an error correction $\varrho \subseteq \Delta_k(\mathcal{L})$ from the syndrome $\sigma \in \text{im } \partial_k$ by applying the Sweep Rule at most N times. Typically, we set N to be comparable with the linear size of \mathcal{L} ; see Sec. 4.3 for further discussion. We remark that for the Sweep Decoder to work the lattice \mathcal{L} has to satisfy certain conditions (which we already discussed in the previous section) and we need to specify the sweep direction $\vec{t} \in \mathbb{R}^d$.

Algorithm 2: Sweep Decoder

Require: the toric code of type $k \neq 1$ on the d -dimensional lattice \mathcal{L} , the Sweep Rule on \mathcal{L}

Input: $(k-1)$ -dimensional syndrome $\sigma \in \text{im } \partial_k$

Output: k -dimensional error correction $\varrho \subseteq \Delta_k(\mathcal{L})$

initialize $n = 0$, $\sigma^{(0)} = \sigma$ and $\varrho^{(0)} = 0$

unless $n > N$ or $\sigma^{(n)} = 0$ repeat:

1. update time step $n \leftarrow n + 1$
2. apply the Sweep Rule simultaneously to every vertex of \mathcal{L} to get updated $\sigma^{(n)}$ and $\varrho^{(n)}$

if $n \leq N$, then $\varrho = \varrho^{(n)}$, otherwise $\varrho = \text{FAIL}$

return ϱ

Note that if $\varrho \neq \text{FAIL}$, then the resulting error correction removes the input excitation

σ , namely

$$\partial_k \varrho = \partial_k \sum_{i=1}^n (\varrho^{(i)} + \varrho^{(i-1)}) = \sum_{i=1}^n (\sigma^{(i)} + \sigma^{(i-1)}) = \sigma^{(n)} + \sigma^{(0)} = \sigma. \quad (4.16)$$

The Sweep Decoder can fail for two reasons. First, the Sweep Decoder may not succeed at finding the error correction in N time steps, which results in $\varrho = \text{FAIL}$. Second, the correction ϱ combined with the actual error $\epsilon \subseteq \Delta_k(\mathcal{L})$ (whose syndrome $\partial_k \epsilon = \sigma$ we tried to decode) may implement a non-trivial logical operator, i.e., $\varrho + \epsilon \notin \text{im } \partial_k$. Lastly, we remark that the Sweep Decoder described here corrects Z -errors; correction of X -errors is described analogously but in the dual lattice \mathcal{L}^* and for $(d - k - 1)$ -dimensional excitations.

4.3 Proof of threshold

In order to argue that the decoder of the d -dimensional toric code of type $k \neq 1$ based on the Sweep Rule has a non-zero threshold one might be tempted to exploit similarities with the Ising model and use reasoning similar to Peierls' argument [Bro+16]. However, in this section we provide a rigorous proof based on the renormalization group (RG) ideas. Our proof is inspired by previous works [GR88; Har04; BH13]. In particular, the notation for chunk decomposition and the arguments about suppression of high-level chunks closely follow Ref. [BH13]. While it is natural to use the RG approach to analyze e.g. the performance of the Broom Decoder, which is an RG decoder, it is somehow unexpected to derive the existence of a threshold for the Sweep Rule.

Two notions of distance

In the proof, we need a notion of the distance and the causal distance. The distance $d(u, v)$ between two vertices u and v in the lattice \mathcal{L} is defined to be the length of the shortest path connecting u and v

$$d(u, v) = \min_{(u:v)} |(u : v)|, \quad (4.17)$$

whereas the causal distance $d_{\updownarrow}(u, v)$ is the length of the shortest causal path between u and v

$$d_{\updownarrow}(u, v) = \min_{(u \updownarrow v)} |(u \updownarrow v)|. \quad (4.18)$$

We note that there is always a path between u and v , but a causal path might not exist and in that case we set $d_{\updownarrow}(u, v) = \infty$. Also, the following inequality holds

$$d(u, v) \leq d_{\updownarrow}(u, v). \quad (4.19)$$

We define the distance $d(U, V)$ between two subsets of vertices U and V as the minimal distance between any two vertices of U and V , namely

$$d(U, V) = \min_{u \in U, v \in V} d(u, v). \quad (4.20)$$

It will be also useful to define the diameter of a subset of vertices V as the maximal distance between any two vertices of V , i.e.,

$$\text{diam}(V) = \max_{u, v \in V} d(u, v) \quad (4.21)$$

Properties of the Sweep Rule

Now we discuss some properties of the Sweep Rule which will be needed to a prove non-zero threshold of the Sweep Decoder.

Lemma 8 (Properties of the Sweep Rule) *Let \mathcal{L} be a d -dimensional lattice and $\sigma \in \text{im } \partial_{k+1}$ be the initial k -dimensional domain wall of finite size, where $k \in \{1, \dots, d-1\}$. At each time step we apply the Sweep Rule simultaneously to every vertex of \mathcal{L} . Then, after $n \geq 0$ time steps*

1. (Support Property) *the domain wall $\sigma^{(n)}$ stays within the causal diamond $\diamond(\sigma)$, i.e.,*

$$\sigma^{(n)} \subset \diamond(\sigma), \quad (4.22)$$

2. (Propagation Property) *for every vertex v of the domain wall $\sigma^{(n)}$ the causal distance between v and σ is at most n , i.e.,*

$$d_{\uparrow}(v, \sigma) \leq n, \quad (4.23)$$

3. (Removal Property) *the domain wall is removed, namely $\sigma^{(n)} = 0$, provided that n is at least the length of the longest causal path between the infimum $\inf \sigma$ and supremum $\sup \sigma$, i.e.,*

$$n \geq \max_{(\inf \sigma \uparrow \sup \sigma)} |(\inf \sigma \uparrow \sup \sigma)| \quad (4.24)$$

We remark that we can strengthen the Support Property by showing that $\sigma^{(n)} \subset \downarrow(\sup \sigma) \cap \bigcup_{v \in \Delta_0(\sigma)} \uparrow(v)$. Also, we can bake the bound in the Removal Property tighter by showing $n \geq \max_{v \in \Delta_0(\sigma)} \max_{(\sup \sigma \uparrow v)} |(\sup \sigma \uparrow v)|$. However, to prove a non-zero threshold it suffices to use weaker conditions in Eq. (4.22) and (4.24), which are simpler to state and explain.

Proof: We prove the properties of the Sweep Rule by induction. For $n = 0$ all of them trivially hold. In the rest of the proof, we use the following simple fact about causal diamonds: for any finite $U, W \subset \Delta_0(\mathcal{L})$ if $U \subseteq W$, then $U \subseteq \diamond(U) \subseteq \diamond(W)$.

Now we show the induction step for the Support Property. Let $V^{(n-1)} \subset \Delta_0(\sigma^{(n-1)})$ denote the set of extremal vertices of the domain wall $\sigma^{(n-1)}$ at time step $n - 1$. Note that at time step n for every extremal vertex $v \in V^{(n-1)}$ the Sweep Rule finds a certain subset $\varphi^{(n)}(v)$ of neighboring $(k + 1)$ -simplices, which locally matches $\sigma^{(n-1)}|_v$. Then, by flipping spins on $\varphi^{(n)}(v)$ the domain wall is locally modified and it becomes

$$\sigma^{(n)} = \sigma^{(n-1)} + \sum_{v \in V^{(n-1)}} \varphi^{(n)}(v). \quad (4.25)$$

Note that $\varphi^{(n)}(v)$ is chosen in such a way that $\diamond(\varphi^{(n)}(v)) = \diamond(\sigma^{(n-1)}|_v)$, and thus $\diamond(\varphi^{(n)}(v)) \subseteq \diamond(\sigma^{(n-1)}) \subseteq \diamond(\sigma)$. We conclude that

$$\diamond(\sigma^{(n)}) \subseteq \diamond\left(\diamond(\sigma^{(n-1)}) \cup \bigcup_{v \in V^{(n-1)}} \diamond(\varphi^{(n)}(v))\right) \subseteq \diamond(\sigma). \quad (4.26)$$

It is straightforward to prove the Propagation Property. Namely, every vertex v in the domain wall $\sigma^{(n)}$ either belongs to $\sigma^{(n-1)}$ or is connected to some vertex $u \in \Delta_0(\sigma^{(n-1)})$ via an edge $(u, v) \in \Delta_1(\mathcal{L})$, such that $(u, v) \cdot \vec{t} > 0$. Note that the latter case can arise when we locally modify $\sigma^{(n-1)}$ by flipping $(k + 1)$ -simplices around its extremal vertex u . Thus, by using the induction hypothesis and triangle inequality we arrive at

$$d_{\uparrow}(v, \sigma) \leq d_{\uparrow}(v, u) + d_{\uparrow}(u, \sigma) \leq n, \quad (4.27)$$

where we set $u = v$ if $v \in \Delta_0(\sigma^{(n-1)})$.

To show the Time Property we argue that the following integer-valued function

$$f_{\sigma}(n) = \max_{v \in \Delta_0(\sigma^{(n)})} \max_{(\sup \sigma \uparrow v)} |(\sup \sigma \uparrow v)|, \quad (4.28)$$

which is the length of the longest causal path between the supremum of σ and any vertex v in the domain wall $\sigma^{(n)}$, is a monotone of the Sweep Rule. In other words, the function $f_{\sigma}(n)$ is monotonically decreasing with n until the domain wall $\sigma^{(n)}$ is removed, and then we set $f_{\sigma}(n) = 0$. First, note that if v is a vertex of $\sigma^{(n)}$ which maximizes the function $f_{\sigma}(n)$, then it has to be extremal. Thus, at time step $n + 1$ the Sweep Rule modifies the domain wall in the neighborhood of v . In particular, v

is not included in $\sigma^{(n+1)}$, however some new vertices from the neighborhood of v , which are necessarily closer (in the sense of the longest causal path) to $\sup \sigma$ may be included. Thus, we conclude that $f_\sigma(n+1) < f_\sigma(n)$, as desired.

We observe that Time Property follows immediately from the monotone $f_\sigma(n)$. Namely, the initial value $f_\sigma(0)$ is upper-bounded by $\max_{(\inf \sigma \uparrow \sup \sigma)} |(\inf \sigma \uparrow \sup \sigma)|$. As long as the domain wall $\sigma^{(n)} \neq 0$, the monotone $f_\sigma(n)$ is decreased by at least one at each time step. Thus, after $n \geq \max_{(\inf \sigma \uparrow \sup \sigma)} |(\inf \sigma \uparrow \sup \sigma)|$ time steps we have $f_\sigma(n) = 0$ and the domain wall is guaranteed to disappear, which in turn shows the Time Property. \square

Properties of the lattice revisited

Before we proceed, let us define a discrete d -dimensional ball $B_v(r)$ of radius r centered at the vertex v of the d -dimensional lattice \mathcal{L} to be a collection of all k -simplices, whose distance from v is less than r , namely

$$B_v(r) = \{k \in \Delta_k(\mathcal{L}) \mid k \in \{0, 1, \dots, d\} \wedge d(v, k) < r\}. \quad (4.29)$$

Note that a unit ball $B_v(1)$ corresponds to the collection of all k -simplices containing v for any $k \in \{0, 1, \dots, d\}$, i.e., $B_v(1) = \bigsqcup_{k=0}^d \text{St}_k(v)$.

Now we briefly revisit the assumptions on the d -dimensional lattice \mathcal{L} we need to make in the rest of this section. The lattice \mathcal{L} , which is built of d -dimensional simplices, has to satisfy the following properties.

1. Causal structure:

- (i) for any finite subset of vertices $V \subset \Delta_0(\mathcal{L})$ there exists a unique causal diamond $\diamond(V)$,
- (ii) (local condition) for any $v \in \Delta_0(\mathcal{L})$ and $\sigma \in \text{im } \partial_k$ if $\sigma|_v \subset \uparrow(v)$, then there exists $\varphi(v) \subseteq \text{St}_k(v) \cap \uparrow(v)$ satisfying $(\partial_k \varphi(v))|_v = \sigma|_v$ and $\sup \varphi(v) = \sup \sigma|_v$.

2. Locally Euclidean:

- (i) for any ball $B_v(R)$ of radius R one finds a cover

$$\bigcup_{u \in U} B_u(r) \supset B_v(R) \quad (4.30)$$

with balls of radius $r < R$ indexed by $U \subset \Delta_0(\mathcal{L})$, such that $|U| \leq c_B (R/r)^d$ and c_B is a constant,

- (ii) for any finite subset of vertices $V \subset \Delta_0(\mathcal{L})$ the diameters of V and the causal diamond of V are comparable, i.e., there exists a constant c_D such that

$$\text{diam}(\diamond(V)) \leq c_D \cdot \text{diam}(V). \quad (4.31)$$

- (iii) for any pair of vertices $u \leq v$ the distance between them and the maximal length of any causal path between them are comparable, i.e., there exists a constant c_P such that

$$\max_{(u \updownarrow v)} |(u \updownarrow v)| \leq c_P \cdot d(u, v) \quad (4.32)$$

Note that $c_D \geq 1$ since $\diamond(V) \supset V$. Moreover, $c_P \geq d$ since by choosing $u = \inf \delta$ and $v = \sup \delta$ for any d -simplex δ we get $d(u, v) = 1$ and $\max_{(u \updownarrow v)} |(u \updownarrow v)| \geq d$.

We remark that the properties regarding the causal structure of the lattice \mathcal{L} (which we already discussed in the previous section) are needed if one wants to define the Sweep Rule on \mathcal{L} . Additionally, one requires the property of being locally Euclidean in order to prove that the Sweep Decoder has non-zero threshold for the toric code of type $k \in \{2, \dots, d-1\}$ defined on \mathcal{L} . Note that hyperbolic lattices do not satisfy the property of being locally Euclidean, and thus we cannot readily use the Sweep Decoder in that setting.

Chunk decomposition and connected components

Let $\epsilon \subseteq \Delta_k(\mathcal{L})$ be an error in the d -dimensional toric code of type k , i.e., the set of k -simplices identified with qubits affected by Pauli Z errors. We define a level-0 chunk $E^{[0]}$ to be an element of ϵ . In other words, a level-0 chunk corresponds to a single location of error. We recursively define a level- n chunk $E^{[n]} = E_1^{[n-1]} \sqcup E_2^{[n-1]}$ to be a disjoint union of two level- $(n-1)$ chunks $E_1^{[n-1]}$ and $E_2^{[n-1]}$, such that $\text{diam}(E^{[n]}) \leq Q^n/2$ for some constant Q . We define level- n error $E_n \subseteq \epsilon$ to be a union of all level- n chunks

$$E_n = \bigcup_i E_i^{[n]}. \quad (4.33)$$

Note that by definition $\epsilon = E_0$. Also, we have the following sequence of inclusions

$$\epsilon = E_0 \supseteq E_1 \supseteq \dots \supseteq E_m \supsetneq E_{m+1} = \emptyset, \quad (4.34)$$

which allows us to define $F_i = E_i \setminus E_{i+1}$ for $i = 0, 1, \dots, m$. Note that for any finite ϵ there exists a finite m satisfying Eq. (4.34). Lastly, we arrive at the following disjoint decomposition of the error

$$\epsilon = F_0 \sqcup F_1 \sqcup \dots \sqcup F_m. \quad (4.35)$$

We say that a subset of errors $M \subseteq \epsilon$ is an l -connected component if it cannot be split into two disjoint non-empty sets M_1 and M_2 separated by more than l . In other words, for any $M_1, M_2 \neq \emptyset$ if $M = M_1 \sqcup M_2$, then $d(M_1, M_2) \leq l$. One can show that, roughly speaking, the diameter of any connected component is not too big and different connected components are far from each other. This important observation is captured by the following lemma [BH13], whose proof we include for completeness.

Lemma 9 (Connected Components) *Let $Q \geq 6$ be some constant and a subset of errors $M \subseteq \epsilon$ be a Q^i -connected component of F_i . Then, $\text{diam}(M) \leq Q^i$ and $d(M, E_i \setminus M) > Q^{i+1}/3$.*

Proof: We prove the lemma by contradiction. Let us pick any $a \in \Delta_0(F_i)$ and assume that there exists $b \in \Delta_0(E_i)$, such that $Q^i < d(a, b) \leq Q^{i+1}/3$. Then, a and b cannot be in the same level- n chunk. Moreover, a and b belong to two different level- n chunks A and B , which are necessarily disjoint. Using triangle inequality and $Q \geq 6$ we get

$$\text{diam}(A \sqcup B) \leq \text{diam}(A) + d(A, B) + \text{diam}(B) \leq Q^i + Q^{i+1}/3 \leq Q^{i+1}/2. \quad (4.36)$$

This implies that $A \sqcup B$ is a level- $(i+1)$ chunk, and subsequently $a \in A \sqcup B \subseteq E_{i+1}$, which is in contradiction with $a \in F_i = E_i \setminus E_{i+1}$. We thus conclude that for any $b \in \Delta_0(E_i)$ we either have $d(a, b) < Q^i$ or $d(a, b) > Q^{i+1}/3$. The former case leads us to a conclusion that any Q^i -connected component $M \subseteq F_i$ has diameter at most Q^i . The latter case allows us to argue that the distance between M and $E_i \setminus M$ is more than $Q^{i+1}/3$. \square

We remark that the (Connected Components) Lemma 9 will be used to show that the Sweep Decoder removes different connected components independently of one another since they are sufficiently far apart.

Suppression of high-level chunks

We would like to show that the probability of observing a high-level chunk is doubly-exponentially suppressed in the level of the chunk. We do it by using results from the percolation theory, in particular the van den Berg and Kesten inequality [Gri99].

First, we discuss the probability of observing different error configurations $\epsilon \subseteq \Delta_k(\mathcal{L})$. We assume that with probability p every qubit is independently affected by

Pauli Z error, which is equivalent to the corresponding k -simplex being included in ϵ . Thus, in the case of a lattice \mathcal{L} with a finite number of k -simplices, the probability of the error configuration ϵ is given by $\text{pr}(\epsilon) = p^{|\epsilon|}(1-p)^{|\Delta_k(\mathcal{L})|-|\epsilon|}$. A collection of (error) configurations is called an event \mathcal{A} . We say that the event \mathcal{A} is increasing if $\epsilon \in \mathcal{A}$ implies $\epsilon' \in \mathcal{A}$ for any two configurations $\epsilon \subseteq \epsilon' \subseteq \Delta_k(\mathcal{L})$. The disjoint occurrence $\mathcal{A} \circ \mathcal{B}$ of two events \mathcal{A} and \mathcal{B} is defined as the collection of configurations $\epsilon = \epsilon_{\mathcal{A}} \sqcup \epsilon_{\mathcal{B}}$, which are a disjoint union of $\epsilon_{\mathcal{A}} \in \mathcal{A}$ and $\epsilon_{\mathcal{B}} \in \mathcal{B}$.

Let us consider a d -dimensional lattice \mathcal{L} and a randomly chosen error configuration $\epsilon \subseteq \Delta_k(\mathcal{L})$. We define the following events:

- $\mathcal{A}_{v,n}$: $B_v(Q^n/2)$ has a non-zero overlap with a level- n chunk of ϵ ,
- $\mathcal{B}_{v,n}$: $B_v(Q^n)$ contains a level- n chunk of ϵ ,
- $\mathcal{C}_{v,n}$: $B_v(Q^n)$ contains two disjoint level- $(n-1)$ chunks of ϵ ,
- $\mathcal{D}_{v,n}$: $B_v(Q^n)$ contains a level- $(n-1)$ chunk of ϵ .

By definition of chunks we have

$$\text{pr}(\mathcal{A}_{v,n}) \leq \text{pr}(\mathcal{B}_{v,n}) \leq \text{pr}(\mathcal{C}_{v,n}). \quad (4.37)$$

To relate the probabilities of events $\mathcal{C}_{v,n}$ and $\mathcal{D}_{v,n}$ we first note that the event $\mathcal{C}_{v,n} = \mathcal{D}_{v,n} \circ \mathcal{D}_{v,n}$ is the disjoint occurrence. Then, we can use the van den Berg and Kesten inequality: if \mathcal{A} and \mathcal{B} are two increasing events, then the probability $\text{pr}(\mathcal{A} \circ \mathcal{B})$ of the disjoint occurrence of \mathcal{A} and \mathcal{B} is upper-bounded by $\text{pr}(\mathcal{A})\text{pr}(\mathcal{B})$. Thus, we find $\text{pr}(\mathcal{C}_{v,n}) \leq \text{pr}(\mathcal{D}_{v,n})^2$. Now, consider a cover of the ball $B_v(Q^n)$ with balls of radius $Q^{n-1}/2$ indexed by $U \subset \Delta_0(\mathcal{L})$. Using the property that the lattice \mathcal{L} is locally Euclidean, we can find a cover with $|U| \leq (2Q)^d c_B$. Note that if $B_v(Q^n)$ contains a level- $(n-1)$ chunk, then there exists a vertex $u \in U$, such that the ball $B_u(Q^{n-1}/2)$ has non-zero overlap with that chunk. We remark that the latter condition describes the event $\mathcal{A}_{u,n-1}$. Thus, using union bound we arrive at

$$\text{pr}(\mathcal{D}_{v,n}) \leq \sum_{u \in U} \text{pr}(\mathcal{A}_{u,n-1}) \leq |U| \max_{u \in U} \text{pr}(\mathcal{A}_{u,n-1}). \quad (4.38)$$

Let us denote the probability of the event $\mathcal{A}_{v,n}$ maximized over the set of vertices $\Delta_0(\mathcal{L})$ by

$$p_{\mathcal{A},n} = \max_{v \in \Delta_0(\mathcal{L})} \text{pr}(\mathcal{A}_{v,n}). \quad (4.39)$$

Combining all previous inequalities and applying them recursively we arrive at

$$p_{\mathcal{A},n} \leq ((2Q)^d c_B p_{\mathcal{A},n-1})^2 \leq \dots \leq ((2Q)^d c_B)^{-2} (((2Q)^d c_B)^2 p_{\mathcal{A},0})^{2^n}. \quad (4.40)$$

Note that the event $\mathcal{A}_{v,0}$ describes the situation that at least one qubit in the neighborhood of v is affected by the error ϵ . Thus, $\text{pr}(\mathcal{A}_{v,0})$ is upper bounded by $|\text{St}_k(v)|p$ and subsequently $p_{\mathcal{A},0} \leq \max_{v \in \Delta_0(\mathcal{L})} |\text{St}_k(v)|p$. We finally conclude that for any given ball $B_v(Q^n/2)$ the probability $\text{pr}(\mathcal{A}_{v,n})$ of any level- n chunk of ϵ intersecting with $B_v(Q^n/2)$ is suppressed doubly exponentially in n as long as

$$p < p_{\text{th}} = \left(((2Q)^d c_B)^2 \max_{v \in \Delta_0(\mathcal{L})} |\text{St}_k(v)| \right)^{-1}. \quad (4.41)$$

Putting things together

Now we are ready to prove that the Sweep Decoder for the d -dimensional toric code of type $k \in \{2, \dots, d-1\}$ has a non-zero threshold, which is lower-bounded by p_{th} defined in Eq. (4.41). For concreteness, we consider a family of lattices \mathcal{L} on the d -dimensional torus of growing linear size $L \rightarrow \infty$, which satisfy the conditions from Sec. 4.3. Note that by the linear size of \mathcal{L} we mean the length of the shortest non-contractible path in \mathcal{L} . The toric code defined on the lattice \mathcal{L} has $\binom{d}{k}$ logical qubits and the corresponding logical Z operators can be represented as Pauli Z operators with support forming non-contractible k -dimensional sheets.

Let $\epsilon \subseteq \Delta_k(\mathcal{L})$ be a randomly chosen Z -type error, where each qubit is independently affected with probability $p < p_{\text{th}}$. We can find the disjoint decomposition of the error $\epsilon = F_0 \sqcup F_1 \sqcup \dots \sqcup F_m$, where m is the maximal level of any chunk of ϵ . The main idea behind the proof is to show that: (i) with high probability the maximal level of any chunk satisfies $m < m^* = \lfloor \log_Q(L/c_D) \rfloor$ and (ii) the Sweep Decoder successfully corrects any level- n chunk of the error ϵ for all $n < m^*$.

To show (i), we use the bound in Eq. (4.40). Namely, if there exists a level- m^* chunk of the error ϵ , then it has to intersect one of the balls from the set $\{B_v(Q^{m^*}/2)\}_{v \in \Delta_0(\mathcal{L})}$. Using union bound and Eq. (4.40) we find the following upper bound on the probability of ϵ containing a level- m^* chunk

$$\text{pr}(\text{level-}m^* \text{ chunk}) \leq \sum_{v \in \Delta_0(\mathcal{L})} \text{pr}(\mathcal{A}_{v,m^*}) \leq |\Delta_0(\mathcal{L})| p_{\mathcal{A},m^*} \leq \text{poly}(L) \cdot \exp(-\alpha L^\beta), \quad (4.42)$$

where $\alpha = c_D^{-\beta} \log(p_{\text{th}}/p)$, $\beta = \log_Q 2$ and we set $|\Delta_0(\mathcal{L})| = \text{poly}(L)$. This concludes the proof of (i).

To show (ii), we need to discuss the behavior of the Sweep Decoder. Let us choose a constant $Q = 6c_Dc_P$. At every time step $n = 1, 2, \dots$ the Sweep Decoder simultaneously applies the Sweep Rule to every vertex of the lattice and locally modifies the domain wall (corresponding to the syndrome $\sigma^{(n)} \in \text{im } \partial_k$, where we set $\sigma^{(0)} = \partial_k \epsilon$). Consider any non-empty subset of errors $M \subseteq \epsilon$, which is a Q^0 -connected component of F_0 . Then, within first $t_0 = c_Dc_PQ^0$ time steps the Sweep Rule removes the part $\partial_k M$ of the domain wall $\partial_k \epsilon$, which corresponds to M . Namely, using the (Connected Components) Lemma 9 we get that $\text{diam}(M) \leq Q^0$.³ Since \mathcal{L} is locally Euclidean, from Eq. (4.31) we get $c_D \text{diam}(M) \geq \text{diam}(\diamond(M)) \geq d(\inf M, \sup M)$, which combined with Eq. (4.32) results in the bound $|\inf M \Downarrow \sup M| \leq c_Dc_P \text{diam}(M) = t_0$ on the maximal length of any causal path within the causal diamond $\diamond(M)$. Note that $\diamond(\partial_k M) \subseteq \diamond(M)$, and thus from the Removal Property in Lemma 8 we obtain that $\partial_k M$ is guaranteed to be removed after $t_0 \geq \max_{(\inf \sigma \Downarrow \sup \sigma)} |\inf \sigma \Downarrow \sup \sigma|$ time steps.

Importantly, in the presented reasoning we use the fact that the distance between $\partial_k M$ and $\partial_k \epsilon \setminus \partial_k M$ is greater than $Q^1/3$. This fact follows from the (Connected Components) Lemma 9. Thus, the time evolution of the rest of the domain wall $\partial_k \epsilon \setminus \partial_k M$ due to the Sweep Rule does not affect the removal of $\partial_k M$. This follows from the fact that both $\partial_k \epsilon \setminus \partial_k M$ and $\partial_k M$ can only propagate over the distance at most $t_0 = c_Dc_PQ^0 \leq Q^1/6$ toward each other; see the Propagation Property in Lemma 8. Thus, they will not cover the total distance of more than $Q^1/3$, which is the separation between them.

We remark that the reasoning is applicable to Q^i -connected components of F_i for higher levels $i \geq 1$. We summarize our discussion in the following lemma, which can be analogously proven by induction on the level i .

Lemma 10 *Let $\epsilon \subseteq \Delta_k(\mathcal{L})$ be an error with the disjoint decomposition $\epsilon = F_0 \sqcup F_1 \sqcup \dots \sqcup F_m$ and choose $Q = 6c_Dc_P$. Then, for any Q^i -connected component M of F_i the corresponding part $\partial_k M$ of the domain wall $\partial_k \epsilon$ is removed by the Sweep Rule within first $t_i = c_Dc_PQ^i$ time steps. Moreover, the removal of $\partial_k M$ is not affected by any other part $\partial_k M'$ of the domain wall, irrespective of the level j of the Q^j -connected component M' of F_j .*

³Note that $\text{diam}(M) = 1$ implies that all the vertices of M belong to the same d -simplex δ . This, however, does not imply that the Sweep Rule can remove the corresponding part $\partial_k M$ of the domain wall in one step. Rather, at most $d - 1$ time steps may be required, as can be seen in the case of the one-dimensional domain wall visiting all vertices of δ in a sequence induced by the sweep direction \vec{i} .

We run the Sweep Decoder for $N = t_{m^*-1}$ time steps. The Lemma 10 guarantees that after N time steps any Q^n -connected component M of F_n is removed for all $n < m^*$. Moreover, for each M the Sweep Decoder finds (independently of the other connected components) a correction of the part $\partial_k M$ of the domain wall, which is contained in the causal diamond of $\diamond(M)$. This follows from the Support Property in Lemma 8. Note that the diameter of the causal diamond $\diamond(M)$ is smaller than the linear size of the system

$$\text{diam}(\diamond(M)) \leq c_D \cdot \text{diam}(M) \leq c_D Q^n < c_D Q^{m^*} \leq L, \quad (4.43)$$

where we use Eq. (4.31) and the (Connected Components) Lemma 9. Thus, any operator supported within $\diamond(M)$ cannot implement a non-trivial logical operator. This finishes the proof of (ii).

Lastly, we conclude that the Sweep Decoder can fail only if there exists a level- m^* chunk of the error ϵ . We arrive at an upper-bound on the decoding failure probability

$$\text{pr}(\text{fail}) \leq \text{pr}(\text{level-}m^* \text{ chunk}) \leq \text{poly}(L) \cdot \exp(-\alpha L^\beta), \quad (4.44)$$

which goes to zero in the limit of infinite size $L \rightarrow \infty$ for $p < p_{\text{th}}$, where p_{th} is a positive constant specified in Eq. (4.41). This finishes the proof of a non-zero threshold of the Sweep Decoder for the d -dimensional toric code of type $k \in \{2, \dots, d-1\}$.

4.4 Decoding of the color code

The color and toric codes are closely related in $d \geq 2$ dimensions [KYP15]. Thus, one would expect to be able to use toric code decoders to decode the color code. This intuition is indeed correct, as was shown in two dimensions [Del14b; BDP12]. However, the question in $d \geq 3$ dimensions remained mostly unexplored [BNB15]. Other 2D color code decoders have been studied [Wan+10; SR12], but it has been far from obvious whether one could use similar strategies to decode higher-dimensional color code. In this section, we describe how one can always locally reduce the problem of decoding the color code to that of the toric code. We would like to emphasize that we can use any toric code decoder for the reduced problem. We finish by introducing a new class of local decoders of the color code in d dimensions based on the Sweep Rule.

Color code

To define the color code [BM06; BM07a; KB15], we need a d -dimensional lattice \mathcal{L} built of d -simplices. In addition, we require that the vertices of \mathcal{L} are $(d+1)$ -

colorable, i.e., we can introduce a function

$$\text{color} : \Delta_0(\mathcal{L}) \rightarrow \mathbb{Z}_{d+1}, \quad (4.45)$$

where $\mathbb{Z}_{d+1} = \{0, 1, \dots, d\}$ is a set of $d + 1$ colors and any two vertices connected by an edge have different color. By $\text{color}(\kappa)$ we denote the set of colors of all the vertices $\Delta_0(\kappa)$ of a k -simplex κ .

The d -dimensional color code of type k is constructed by placing one qubit at every d -simplex δ . For every $(k - 1)$ -simplex μ and $(d - k - 1)$ -simplex ν we define X - and Z -stabilizer generators $S_X(\mu)$ and $S_Z(\nu)$ to be the product of either Pauli X or Z operators on qubits adjacent to μ and ν , namely

$$S_X(\mu) = \prod_{\delta \in \text{St}_d(\mu)} X(\delta), \quad S_Z(\nu) = \prod_{\delta \in \text{St}_d(\nu)} Z(\delta). \quad (4.46)$$

The representatives of logical X and Z operators of the d -dimensional color code can be chosen to form $(d - k)$ - and k -dimensional objects. The X - and Z -syndromes can be thought of as $(k - 1)$ - and $(d - k - 1)$ -objects, which we call excitations. Note that we can construct a CSS chain complex associated with the color code by setting in Eq. (5.3) the vector spaces $D_2 = C_{d-k-1}$, $D_1 = C_d$, $D_0 = C_{k-1}$ and defining the boundary and coboundary operators as follows $\tilde{\partial}_2 = \partial_{d-k-1,d}$, $\tilde{\partial}_1 = \partial_{d,k-1}$, $\tilde{\partial}_0^* = \partial_{k-1,d}$, $\tilde{\partial}_1^* = \partial_{d,d-k-1}$. To summarize

$$\begin{array}{ccccc} C_{d-k-1} & \xrightarrow{\partial_{d-k-1,d}} & C_d & \xrightarrow{\partial_{d,k-1}} & C_{k-1} \\ Z\text{-stabilizers} & & \text{qubits} & & X\text{-stabilizers} \end{array} \quad (4.47)$$

To illustrate the color code construction, we consider the three-dimensional case. The 3D color code of type $k = 1$ is obtained by placing qubits on tetrahedra of a three-dimensional lattice \mathcal{L} built of tetrahedra, whose vertices are 4-colorable. An example of such a lattice is the bcc lattice obtained from two interleaved cubic lattices; see Fig. 4.7(a). We choose X - and Z -stabilizer generators for every vertex v and edge e of \mathcal{L} to be supported on qubits adjacent to v and e , namely

$$S_X(v) = \prod_{t \in \text{St}_3(v)} X(t), \quad S_Z(e) = \prod_{t \in \text{St}_3(e)} Z(t), \quad (4.48)$$

The logical X - and Z -operators of the 3D color code can be supported within 2D sheet-like and 1D string-like regions, whereas X - and Z -syndromes form 0D point-like and 1D loop-like excitations.

We would like to emphasize that there is a qualitative difference between the toric and color code excitations. Similarly to the toric code, there are two types of color code excitations, electric and magnetic, which correspond to violated X - and Z -type stabilizers. In addition, there are $\binom{d+1}{k}$ species of $(k-1)$ -dimensional electric excitations in the color code compared to just one in the toric code. Different species of electric excitations are labelled by different subsets of k colors $C \subset \mathbb{Z}_{d+1}$, which are the colors of $(k-1)$ -simplices identified with violated stabilizers. In case of $(d-k-1)$ -dimensional magnetic excitations, we have $\binom{d+1}{d-k}$ species. The fact that there are different species of color code excitations plays a role when one locally creates or removes excitations. Namely, only certain processes are allowed, as can be easily seen in the case of zero-dimensional point-like electric excitations, i.e., $k=1$. A single-qubit Pauli Z error on a d -simplex $\delta \in \Delta_d(\mathcal{L})$ violates X -type stabilizers associated with vertices $\Delta_0(\delta)$, and thus creates or removes $d+1$ point-like electric excitations. However, a two-qubit Pauli Z error on neighboring qubits (on d -simplices sharing a $(d-1)$ -face) results in creation or removal of just two point-like excitations of the same color. The fact that one might not be able to locally remove certain incompatible species of excitations of the color code is one of the reasons why color code decoding seems to pose a harder challenge than toric code decoding.

Definition of the projected lattice

An important part of the Projection Decoder is the reduction of the decoding problem for the color code on the d -dimensional lattice \mathcal{L} , which is built of d -simplices and has $(d+1)$ -colorable vertices, to that for the toric code defined on the projected lattice \mathcal{L}_C . In this section we define the projected lattice \mathcal{L}_C obtained from \mathcal{L} by some local modifications of the lattice structure.

Let us pick a subset $C \subset \mathbb{Z}_{d+1}$ of $k+1$ colors. The projected lattice \mathcal{L}_C is a lattice containing all the vertices of \mathcal{L} of color in C . Moreover, for all $i=1, \dots, k$ all the i -simplices ι of \mathcal{L} , whose color is included in C , i.e., $\text{color}(\iota) \subseteq C$, are also included in \mathcal{L}_C . In other words, the first step of constructing \mathcal{L}_C is to delete from \mathcal{L} all the simplices whose color is not completely included in C . Note that this implies that we remove all i -simplices for $i > k$. Then, for $i = k+1, \dots, d$ and every removed $(d-i)$ -simplex of color in $\mathbb{Z}_{d+1} \setminus C$ we will successively attach a new i -cell and eventually construct the d -dimensional projected lattice \mathcal{L}_C . We illustrate the construction of the projected lattice with an example of the three-dimensional bcc lattice in Fig. 4.7.

Definition 1 (Projected Lattice) *Let \mathcal{L} be a valid d -dimensional color code lattice and $C \subset \mathbb{Z}_{d+1}$ be a subset of $k + 1$ colors, where $k \in \{1, \dots, d - 1\}$. The projected lattice \mathcal{L}_C is a cell d -complex constructed inductively from \mathcal{L} in the following way*

- *for $0 \leq i \leq k$: i -cells in \mathcal{L}_C are the same as i -simplices in \mathcal{L} of color included in C ,*
- *for $i = k + 1$: for every removed $(d - k - 1)$ -dimensional simplex $\delta \in \Delta_{d-k-1}(\mathcal{L})$ of color $\mathbb{Z}_{d+1} \setminus C$ add a new $(k + 1)$ -cell $\alpha(\delta)$ to \mathcal{L}_C by attaching a $(k + 1)$ -dimensional ball B^{k+1} to the k -link $\text{Lk}_k(\delta)$*
- *for $k + 2 \leq i \leq d$: every i -cell of \mathcal{L}_C corresponds to some $(d - i)$ -simplex $\delta \in \Delta_{d-i}(\mathcal{L})$ of color in $\mathbb{Z}_{d+1} \setminus C$ and is constructed by first finding all $(d - i + 1)$ -simplices of color in $\mathbb{Z}_{d+1} \setminus C$, which contain δ , and then attaching an i -ball B^i to the corresponding $(i - 1)$ -dimensional balls B^{i-1} (which have already been attached in the preceding inductive step).*

We remark that for $i \geq k$ each step of the inductive construction of the projected lattice \mathcal{L}_C can be viewed as a description of the i -skeleton of a cell complex corresponding to \mathcal{L}_C . Note that in this construction we use a fact that for any i -simplex ι of the d -dimensional lattice \mathcal{L} the corresponding $(d - i - 1)$ -link $\text{Lk}_{d-i-1}(\delta)$ is homeomorphic to a $(d - i - 1)$ -sphere and thus we can attach a $(d - i)$ -ball B^{d-i} to it. For more details, see [Hat02; Gla72].

We emphasize that in order to define the toric code of type k on the projected lattice \mathcal{L}_C we just need to have cells of dimension up to $k + 1$ in \mathcal{L}_C . However, we choose to include higher-dimensional cells in the projected lattice \mathcal{L}_C so that it can be treated on the same footing as \mathcal{L} , namely as a discretization of some d -dimensional manifold. Moreover, the local modifications of the lattice \mathcal{L} which we implement to construct \mathcal{L}_C do not change the topology of the lattice. In particular, we are interested in the structure of the k -cycles and k -boundaries in \mathcal{L}_C , since they determine the toric code logical subspace. First, let us verify that for any $(k + 1)$ -dimensional face $\alpha(\delta) \in \Delta_{k+1}(\mathcal{L})$ its boundary $\partial_{k+1}\alpha(\delta)$ is indeed a k -boundary in \mathcal{L}_C . Recall that $\alpha(\delta)$ is constructed by attaching a $(k + 1)$ -ball B^{k+1} to the k -link $\text{Lk}_k(\delta)$ of some removed $(d - k - 1)$ -simplex $\delta \in \Delta_{d-k-1}(\mathcal{L})$, and thus by definition $\partial_{k+1}\alpha(\delta) = \text{Lk}_k(\delta)$. Note that $\text{Lk}_k(\delta)$ is homeomorphic to a k -dimensional sphere, and thus $\partial_k(\partial_{k+1}\alpha(\delta)) = \partial_k\text{Lk}_k(\delta) = 0$, since the boundary of any discretization of a k -sphere is trivial. We also remark that k -cycles in \mathcal{L}_C which are not k -boundaries,

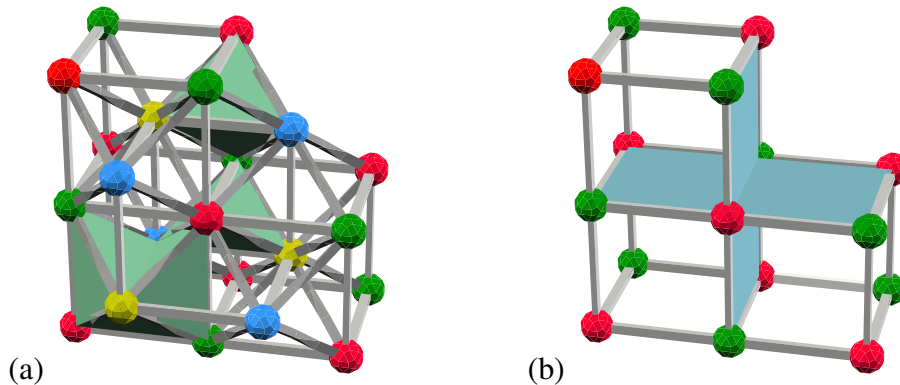


Figure 4.7: (a) The bcc lattice \mathcal{L} is a three-dimensional lattice constructed from two interleaved cubic lattices (one corresponding to vertices of color red or green, and the other to blue or yellow) by filling in tetrahedra (shaded in green). The vertices of the bcc lattice are 4-colorable. (b) The projected lattice \mathcal{L}_{RG} obtained from the bcc lattice \mathcal{L} by removing all vertices of color blue or yellow, as well as all the edges, faces and volumes containing them. For each removed edge of color blue-yellow we attach a square face (shaded in blue) to \mathcal{L}_{RG} , and for each removed vertex we add a cubic volume. We remark that the projected lattice \mathcal{L}_{RB} forms what we call the half-bcc lattice. The figures were made using vZome available at <http://vzome.com>.

i.e., the logical operators of the toric code on \mathcal{L}_C , are in one-to-one correspondence with certain elements from $\ker \partial_{d,k-1} \setminus \text{im } \partial_{d-k-1,d}$, which correspond to logical operators of the color code on \mathcal{L} with the associated color $\mathbb{Z}_{d+1} \setminus C$. We will discuss the structure of logical operators of the color code in Sec. 4.4.

Morphism between color and toric code chain complexes

Now we would like to relate decoding of the color code of type k defined on the lattice \mathcal{L} to that of the toric code of type k on the lattice \mathcal{L}_C , where $C \subset \mathbb{Z}_{d+1}$ is a subset of $k + 1$ colors. Namely, we are going to find a way to map stabilizers, errors and syndromes of the color code on the lattice \mathcal{L} onto stabilizers, errors and syndromes of the toric code on the lattice \mathcal{L}_C , respectively. In addition, the mapping should preserve relations between them, such as the syndrome of any stabilizer is trivial. This is not an obvious task, since we want to relate objects of different dimensionality, for example color code errors are identified with d -simplices, whereas toric code errors — with k -cells.

In order to relate the color and toric codes we introduce the projection π_C , which maps the chain complex of the color code of type k on the d -dimensional lattice \mathcal{L}

$$C_{d-k-1}(\mathcal{L}) \xrightarrow{\partial_{d-k-1 \rightarrow d}} C_d(\mathcal{L}) \xrightarrow{\partial_{d \rightarrow k-1}} C_{k-1}(\mathcal{L}) \quad (4.49)$$

onto the CSS chain complex of the toric code on the projected lattice \mathcal{L}_C

$$C_{k+1}(\mathcal{L}_C) \xrightarrow{\partial_{k+1}} C_k(\mathcal{L}_C) \xrightarrow{\partial_k} C_{k-1}(\mathcal{L}_C). \quad (4.50)$$

The projection π_C is defined in the following way.

Definition 2 (Projection) *The projection π_C is a triple of linear operators $(\pi_C^{(0)}, \pi_C^{(1)}, \pi_C^{(2)})$ defined as follows*

$$\pi_C^{(0)} : C_{k-1}(\mathcal{L}) \rightarrow C_{k-1}(\mathcal{L}_C), \quad (4.51)$$

$$\pi_C^{(0)}(\mu) = \begin{cases} \mu & \text{if } \text{color}(\mu) \subset C, \\ 0 & \text{otherwise,} \end{cases} \quad (4.52)$$

$$\pi_C^{(1)} : C_d(\mathcal{L}) \rightarrow C_k(\mathcal{L}_C), \quad (4.53)$$

$$\pi_C^{(1)}(\delta) = \delta|_C, \quad (4.54)$$

$$\pi_C^{(2)} : C_{d-k-1}(\mathcal{L}) \rightarrow C_{k+1}(\mathcal{L}), \quad (4.55)$$

$$\pi_C^{(2)}(\nu) = \begin{cases} \alpha(\nu) & \text{if } \text{color}(\nu) = \mathbb{Z}_{d+1} \setminus C, \\ 0 & \text{otherwise,} \end{cases} \quad (4.56)$$

where $\delta|_C$ is the k -simplex of color C , which belongs to the d -simplex $\delta \in \Delta_d(\mathcal{L})$.

Recall that $\alpha(\nu)$ is the $(k+1)$ -face of \mathcal{L}_C attached to the k -link $\text{Lk}_k(\nu)$, which corresponds to the $(d-k-1)$ -simplex ν removed from \mathcal{L} . The boundary of $\alpha(\nu)$ is by definition the k -link $\text{Lk}_k(\nu)$.

We claim that the projection π_C applied to the color code chain complex in Eq. (4.49) preserves its structure and results in the toric code chain complex in Eq. (4.50). This means, for instance, that if we map errors in the color code onto errors in the toric code and evaluate their toric code syndrome, then this syndrome will match the mapped color code syndrome. The structure that ensures that the projection π_C preserves the relations between stabilizers, errors and syndromes is a notion of a morphism of chain complexes [Del14b].

Lemma 11 (Morphism of Chain Complexes) *Let $C \subset \mathbb{Z}_{d+1}$ be a subset of $k+1$ colors, where $k \in \{1, \dots, d-1\}$. Let \mathcal{L} be a valid d -dimensional color code lattice. Then, the projection π_C is a morphism between chain complexes of the color code of type k on \mathcal{L} and the toric code of type k on \mathcal{L}_C . In other words, the following*

diagram is commutative

$$\begin{array}{ccccc}
C_{d-k-1}(\mathcal{L}) & \xrightarrow{\partial_{d-k-1,d}} & C_d(\mathcal{L}) & \xrightarrow{\partial_{d,k-1}} & C_{k-1}(\mathcal{L}) \\
\downarrow \pi_C^{(2)} & & \downarrow \pi_C^{(1)} & & \downarrow \pi_C^{(0)} \\
C_{k+1}(\mathcal{L}_C) & \xrightarrow{\partial_{k+1}} & C_k(\mathcal{L}_C) & \xrightarrow{\partial_k} & C_{k-1}(\mathcal{L}_C)
\end{array} \quad (4.57)$$

Proof: Let us pick $\delta \in \Delta_d(\mathcal{L})$ and consider the right side of the diagram. We want to show that $\pi_C^{(0)} \circ \partial_{d,k-1}(\delta) = \partial_k \circ \pi_C^{(1)}(\delta)$. Note that for any $n \in \{0, 1, \dots, k-1\}$ all n -simplices of δ of colors included in C belong to the k -simplex $\delta|_C = \pi_C^{(1)}(\delta)$ of δ , namely

$$\Delta_n(\delta|_C) = \{\nu \in \Delta_n(\delta) \mid \text{color}(\nu) \subset C\}. \quad (4.58)$$

Thus, we obtain

$$\pi_C^{(0)} \circ \partial_{d,k-1}(\delta) = \pi_C^{(0)} \left(\sum_{\mu \in \Delta_{k-1}(\delta)} \mu \right) = \sum_{\substack{\mu \in \Delta_{k-1}(\delta) \\ \text{color}(\mu) \subset C}} \mu = \sum_{\mu \in \Delta_{k-1}(\delta|_C)} \mu \quad (4.59)$$

$$= \partial_k(\delta|_C) = \partial_k \circ \pi_C^{(1)}(\delta), \quad (4.60)$$

which shows commutativity of the right side of the diagram.

Now we analyze the left side of the diagram. Let us pick $\delta \in \Delta_{d-k-1}(\mathcal{L})$ and consider two cases. In the first case, when $\text{color}(\delta) = \mathbb{Z}_{d+1} \setminus C$, all the k -simplices in the k -link of δ have color C . Note that the elements of the k -link $\text{Lk}_k(\delta)$ of δ are in one-to-one correspondence with the elements of the d -star $\text{St}_d(\delta)$ of δ , namely

$$\kappa \in \text{Lk}_k(\delta) \iff \exists \mu \in \text{St}_d(\delta) : \mu = \delta * \kappa, \quad (4.61)$$

where by $\mu = \delta * \kappa$ we denote that the simplex μ is spanned by the disjoint union of vertices of two simplices δ and κ . Then, we also have $\pi_C^{(1)}(\delta * \kappa) = \kappa$ and thus

$$\pi_C^{(1)} \circ \partial_{d-k-1,d}(\delta) = \pi_C^{(1)} \left(\sum_{\mu \in \text{St}_d(\delta)} \mu \right) = \pi_C^{(1)} \left(\sum_{\kappa \in \text{Lk}_k(\delta)} \delta * \kappa \right) \quad (4.62)$$

$$= \sum_{\kappa \in \text{Lk}_k(\delta)} \kappa = \partial_{k+1}(\alpha(\delta)) = \partial_{k+1} \circ \pi_C^{(2)}(\delta). \quad (4.63)$$

In the other case, i.e., $\text{color}(\delta) \neq \mathbb{Z}_{d+1} \setminus C$, there exists a simplex $\omega \subset \delta$ such that $\text{color}(\omega) = \text{color}(\delta) \cap C$. Then, as was shown in Ref. [KYP15], the d -star $\text{St}_d(\delta)$ of δ admits the following disjoint decomposition

$$\text{St}_d(\delta) = \bigsqcup_{\substack{\kappa \in \text{St}_k(\omega) \\ \text{color}(\kappa) = C}} \text{St}_d(\kappa), \quad (4.64)$$

where we use the fact that the lattice \mathcal{L} is built of d -simplices and its vertices are $(d + 1)$ -colorable. Moreover, for any n -simplex ν with $n \in \{0, 1, \dots, d - 1\}$ the cardinality of the d -star $\text{St}_d(\nu)$ is even, i.e., $|\text{St}_d(\nu)| \equiv 0 \pmod{2}$; see [KYP15] for the proof. Thus, $\sum_{\mu \in \text{St}_d(\nu)} \nu = 0$ and we get

$$\pi_C^{(1)} \circ \partial_{d-k-1,d}(\delta) = \pi_C^{(1)} \left(\sum_{\mu \in \text{St}_d(\delta)} \mu \right) = \pi_C^{(1)} \left(\sum_{\substack{\kappa \in \text{St}_k(\omega) \\ \text{color}(\kappa)=C}} \sum_{\mu \in \text{St}_d(\kappa)} \mu \right) \quad (4.65)$$

$$= \sum_{\substack{\kappa \in \text{St}_k(\omega) \\ \text{color}(\kappa)=C}} \sum_{\mu \in \text{St}_d(\kappa)} \pi_C^{(1)}(\mu) = \sum_{\substack{\kappa \in \text{St}_k(\omega) \\ \text{color}(\kappa)=C}} \sum_{\mu \in \text{St}_d(\kappa)} \kappa \quad (4.66)$$

$$= 0 = \partial_{k+1} \circ \pi_C^{(2)}(\delta), \quad (4.67)$$

where in the last step we use $\pi_C^{(2)}(\delta) = 0$ since $\text{color}(\delta) \neq \mathbb{Z}_{d+1} \setminus C$. This shows commutativity of the left side of the diagram in Eq. (4.57) and thus concludes the proof. \square

How to use any toric code decoder for the color code

We have just seen that the projection π_C is a morphism of chain complexes of the color and toric codes of type $k \in \{1, \dots, d - 1\}$ defined on \mathcal{L} and \mathcal{L}_C , respectively. Recall that $C \subset \mathbb{Z}_{d+1}$ denotes a subset of $k + 1$ colors. We now analyze the implications of this morphism for the problem of decoding the color code. Let $\epsilon \subseteq \Delta_d(\mathcal{L})$ be the set of qubits of the color code, which are affected by Pauli Z errors. Then, the corresponding X -type syndrome is $\sigma = \partial_{d,k-1}\epsilon$. We denote by $\sigma_C = \pi_C^{(0)}(\sigma)$ the projected syndrome, i.e., the subset of $(k - 1)$ -simplices of σ , whose color is included in C . Let us assume that for every possible choice of $k + 1$ colors $C \subset \mathbb{Z}_d$ including one specified color c^* we decode the projected syndrome σ_C of the toric code on the projected lattice \mathcal{L}_C . In other words, for every $C \subset \mathbb{Z}_{d+1}$ with $c^* \in C$ we find $\varrho_C \subseteq \Delta_k(\mathcal{L}_C)$ such that $\partial_k \varrho_C = \sigma_C$. We would like to infer the color code correction $\tau \subseteq \Delta_d(\mathcal{L})$ satisfying $\partial_{d,k-1}\tau = \sigma$ from the combined toric code correction $\varrho = \sum_C \varrho_C$, where the sum is over $C \subset \mathbb{Z}_{d+1}$ of $k + 1$ colors containing c^* . We can achieve that by a local procedure, which we call the Lift: for every vertex v in \mathcal{L} of color d locally “lift” the k -dimensional toric code correction to get the d -dimensional color code correction.

We can always find $\tau(v) \subseteq \text{St}_d(v)$ in the Lift, which satisfies the condition $(\partial_{d,k}\tau(v))|_v = \varrho|_v$. Namely, one can show that $\partial_k(\varrho|_v) \subseteq \text{Lk}_{k-1}(v)$ is a valid $(k - 1)$ -dimensional syndrome of the color code defined on the $(d - 1)$ -dimensional

Algorithm 3: Lift

Require: a vertex v in the d -dimensional color code lattice \mathcal{L}

Input: the k -dimensional combined toric code correction $\varrho = \sum_C \varrho_C \subseteq \Delta_k(\mathcal{L})$

Output: the d -dimensional local color code correction $\tau(v) \subseteq \text{St}_d(v)$

find a restriction of ϱ to the neighborhood of v , i.e., $\varrho|_v = \varrho \cap \text{St}_k(v)$

if $\varrho|_v \neq 0$, then find $\tau(v) \subseteq \text{St}_d(v)$ satisfying $(\partial_{d,k}\tau(v))|_v = \varrho|_v$, otherwise

$\tau(v) = 0$

return $\tau(v)$

sphere S^{d-1} around v , i.e., $S^{d-1} \simeq \bigsqcup_{i=0}^{d-1} \text{Lk}_i(v)$. By using an inductive argument, we can find the correction $\varsigma \subseteq \text{Lk}_{d-1}(v)$ for the decoding problem on S^{d-1} , since it is a decoding problem for the $(d-1)$ -dimensional color code. Finally, we set $\tau(v)$ to be the set of all d -simplices in the neighborhood of v spanned by v and ς , i.e., $\tau(v) = v * \varsigma$. We remark that one may find $\tau(v)$ by exhaustively considering all possible subsets of $\text{St}_d(v)$ and checking which one satisfies the condition specified in the Lift.

Algorithm 4: Projection Decoder

Require: the d -dimensional color code of type k on the lattice \mathcal{L} , any decoder of the d -dimensional toric code of type k

Input: the $(k-1)$ -dimensional color code syndrome $\sigma \in \text{im } \partial_{d,k-1}$

Output: the d -dimensional color code correction $\tau \subseteq \Delta_d(\mathcal{L})$

initialize $\tau = 0$ and choose any color $c^* \in \mathbb{Z}_{d+1}$

for every subset of $k+1$ colors $C \subset \mathbb{Z}_{d+1}$ containing c^* :

1. find the projected lattice \mathcal{L}_C and the projected syndrome $\sigma_C = \pi_C^{(0)}(\sigma)$

2. decode σ_C of the toric code on \mathcal{L}_C by finding $\varrho_C \subseteq \Delta_k(\mathcal{L}_C)$ such that

$$\partial_k \varrho_C = \sigma_C$$

find the combined toric code correction $\varrho = \sum_C \varrho_C$

for every vertex v of \mathcal{L} of color c^* :

1. apply the Lift to $\varrho|_v$ in order to find a local color code correction

$$\tau(v) \subseteq \text{St}_d(v)$$

2. update $\tau \leftarrow \tau + \tau(v)$

return τ

We would like to emphasize that the Lift is a fully local procedure (unlike the lifting

procedure discussed in [Del14b]), which does not depend on the toric code decoder used to find the combined toric code correction $\varrho = \sum_C \varrho_C$. We conclude that using the Lift we can define the Projection Decoder of the d -dimensional color code of type k based on any decoder of the d -dimensional toric code of type k . This resolves a long-standing question of how to relate decoding of the color and toric codes by providing an explicit procedure.

We remark that the Projection Decoder always removes all excitations from the color code. In other words $\partial_{d,k-1}\tau = \sigma$. However, the initial error ϵ combined with the correction τ may support non-trivial logical operator, i.e., $\epsilon + \tau \in \ker \partial_{d,k-1} \setminus \text{im } \partial_{d-k-1,d}$. We would like to understand when the Projection Decoder successfully corrects errors in the color code. We show the following result.

Lemma 12 (Successful Decoding) *Consider the color code of type $k \in \{1, \dots, d-1\}$ defined on a d -dimensional lattice \mathcal{L} . Let $\epsilon \subseteq \Delta_d(\mathcal{L})$ be the set of qubits affected by Z -type errors and $\sigma = \partial_{d,k-1}\epsilon$ be the corresponding X -type syndrome. Assume that for all subsets of $k+1$ colors $C \subset \mathbb{Z}_{d+1}$ the projected syndrome $\pi_C^{(0)}(\sigma)$ in the toric code on the projected lattice \mathcal{L}_C is successfully decoded. In other words, the toric code correction $\varrho_C \subseteq \Delta_k(\mathcal{L})$ combined with the projected error $\pi_C^{(1)}(\epsilon)$ forms some toric code stabilizer, i.e., $\varrho_C + \pi_C^{(1)}(\epsilon) \in \text{im } \partial_{k+1}$. Then, the Projection Decoder output $\tau \subseteq \Delta_d(\mathcal{L})$ successfully corrects the error ϵ in the color code, i.e., $\epsilon + \tau \in \text{im } \partial_{d-k-1,d}$.*

Before we prove (Successful Decoding) Lemma 12, we need to discuss the structure of the logical operators of the color code. We consider the color code of type k on the lattice \mathcal{L} , which is a discretization of the d -dimensional manifold without boundary, such as the d -dimensional torus. We can construct representatives of different logical Pauli Z operators as follows. First, we choose a subset of $d-k+1$ colors $\bar{C} \subset \mathbb{Z}_{d+1}$. Then, we find a k -dimensional non-contractible surface \mathcal{M} , such that it only contains $(k-1)$ -simplices of \mathcal{L} of color $\mathbb{Z}_{d+1} \setminus \bar{C}$ and cuts the subset $\omega(\mathcal{M}) \subset \Delta_{d-k}(\mathcal{L})$ of $(d-k)$ -simplices of \mathcal{L} of color \bar{C} ; see Fig. 4.8 for an illustration. Next, we define a subset $\lambda_{\mathcal{M},\bar{C}} \subset \Delta_d(\mathcal{L})$ as the set of all d -simplices of \mathcal{L} , which are in the neighborhood of $(d-k)$ -simplices of color \bar{C} cut by \mathcal{M} , namely

$$\lambda_{\mathcal{M},\bar{C}} = \bigsqcup_{\kappa \in \omega(\mathcal{M})} \text{St}_d(\kappa). \quad (4.68)$$

One can show that $\lambda_{\mathcal{M},\bar{C}} \in \ker \partial_{d,k-1} \setminus \text{im } \partial_{d-k-1,d}$ and thus the operator $\prod_{\delta \in \Delta_d(\lambda_{\mathcal{M},\bar{C}})} Z(\delta)$ is a representative of a non-trivial logical Z operator. We re-

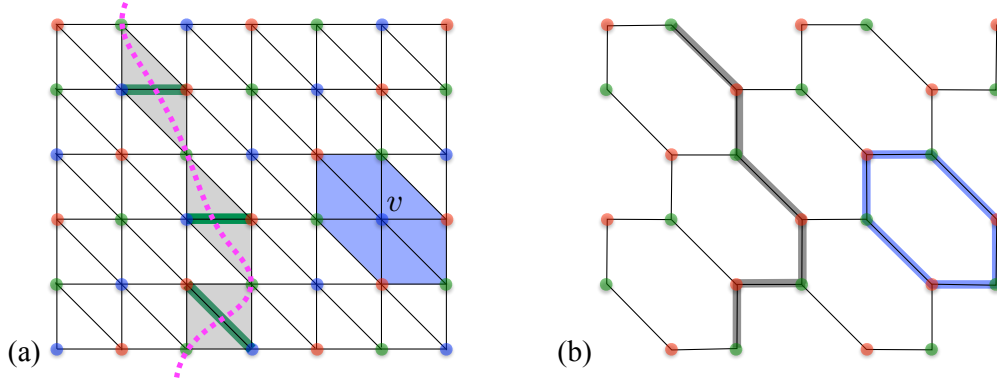


Figure 4.8: The 2D color code on the triangular lattice \mathcal{L} . (a) The support $\lambda_{\mathcal{M}, RB} \subset \Delta_2(\mathcal{L})$ of a non-trivial string-like logical operator can be found as a union of faces (shaded in grey) in the neighborhood of the set $\omega(\mathcal{M})$ of all the red-blue edges (thick green) cut by some non-contractible curve \mathcal{M} (dashed magenta). The support $\text{St}_2(v)$ of the stabilizer generator associated with a vertex v is the set of faces (shaded in blue), which contain v . (b) The projection $\pi_{RG}^{(1)}(\lambda_{\mathcal{M}, RB})$ (thick grey) of the logical operator is a non-contractible curve in the projected lattice \mathcal{L}_{RG} , i.e., it is a 1-cycle which is not a 1-boundary. On the other hand, the projection $\pi_{RG}^{(1)}(\text{St}_2(v))$ (thick blue) of the stabilizer is a contractible loop in \mathcal{L}_{RG} , i.e., it is a 1-boundary.

mark that any generator of the group of logical Z operators can be found in this way.

Let us apply the generalized boundary operator $\partial_{d,k}$ to $\lambda_{\mathcal{M}, \bar{C}}$. One can show that

$$\partial_{d,k} \lambda_{\mathcal{M}, \bar{C}} = \sum_{\kappa \in \omega(\mathcal{M})} \sum_{v \in \Delta_0(\kappa)} v * \text{Lk}_{k-1}(\kappa), \quad (4.69)$$

where $v * \text{Lk}_{k-1}(\kappa)$ denotes the set of k -simplices spanned by the vertex v and the $(k-1)$ -simplices from $\text{Lk}_{k-1}(\kappa)$. Note that k -simplices in $v * \text{Lk}_{k-1}(\kappa)$ are of color $C = (\mathbb{Z}_{d+1} \setminus \bar{C}) \sqcup \{\bar{c}\}$, where $\bar{c} = \text{color}(v)$. Thus, one can view $\partial_{d,k} \lambda_{\mathcal{M}, \bar{C}}$ as a sum of k -simplices which belong to $d-k+1$ non-contractible k -dimensional surfaces along \mathcal{M} in different projected lattices \mathcal{L}_C for all $\bar{c} \in \bar{C}$. In other words, the subset of k -simplices in $\partial_{d,k} \lambda_{\mathcal{M}, \bar{C}}$ restricted to the ones of color C , which corresponds to $\pi_C^{(1)}(\lambda_{\mathcal{M}, \bar{C}})$, is a k -cycle which is not a k -boundary in \mathcal{L}_C . We illustrate the discussion with a two-dimensional example in Fig. 4.8.

We can also apply the generalized boundary operator $\partial_{d,k}$ to the support of Z -type stabilizers of the color code of type k . Let us consider a stabilizer generator $S_Z(v) = \prod_{\delta \in \text{St}_d(v)} Z(v)$ identified with the $(d-k-1)$ -simplex v of color \bar{C} . One can show that

$$\partial_{d,k} \text{St}_d(v) = \text{Lk}_k(v) = \partial_{k+1} \alpha(v), \quad (4.70)$$

where $\alpha(\nu)$ is the $(k+1)$ -dimensional face in the projected lattice $\mathcal{L}_{\mathbb{Z}_{d+1} \setminus \bar{C}}$ identified with the removed $(d-k-1)$ -simplex ν of color \bar{C} . Thus, $\partial_{d,k} \text{St}_d(\nu) = \pi_C^{(1)}(\text{St}_d(\nu))$ is the k -boundary in the projected lattice \mathcal{L}_C , i.e., $\partial_{d,k} \text{St}_d(\nu) \in \text{im } \partial_{k+1}$. By linearity we conclude that the (generalized) k -boundary of the support of any Z -type stabilizer can be viewed as a sum of k -boundaries in different projected lattices \mathcal{L}_C for all subsets $C \subset \mathbb{Z}_{d+1}$ of $k+1$ colors.

We use the (Morphism of Chain Complexes) Lemma 11, as well as the aforementioned facts about the (generalized) k -boundary of logical operators and stabilizers of the color code in the following proof of the (Successful Decoding) Lemma 12.

Proof: In the decoding problem, we do not know the error ϵ but instead we know its syndrome $\sigma = \partial_{d,k-1} \epsilon$. Since the Projection Decoder returns the correction τ with the same syndrome as ϵ , i.e., $\partial_{d,k-1} \tau = \sigma$, we can write

$$\epsilon + \tau = \omega + \lambda, \quad (4.71)$$

where $\omega \in \text{im } \partial_{d-k-1,d}$ corresponds to some stabilizer and λ is a representative of one of the logical operators from $\ker \partial_{d,k-1} / \text{im } \partial_{d-k-1,d}$.

We proceed with a proof by contradiction. Let us assume that the Projection Decoder fails, i.e., $\epsilon + \tau \notin \text{im } \partial_{d-k-1,d}$, which is equivalent to λ supporting a non-trivial logical Z operator. Thus, for some subset $C \subset \mathbb{Z}_{d+1}$ of $k+1$ colors $\partial_{d,k} \lambda$ restricted to the projected lattice \mathcal{L}_C is not the k -boundary, i.e., $\pi_C^{(1)}(\lambda) \notin \text{im } \partial_{k+1}$. Let us apply the projection $\pi_C^{(1)}$ to the both sides of Eq. (4.71), which is equivalent to applying the generalized boundary operator $\partial_{d,k}$ and restricting our attention to k -simplices of color C . Since $\pi_C^{(1)}(\omega)$ is a k -boundary in \mathcal{L}_C and $\pi_C^{(1)}(\lambda)$ is not, then $\pi_C^{(1)}(\epsilon + \tau) \notin \text{im } \partial_{k+1}$ is not a k -boundary in \mathcal{L}_C . Lastly, $\pi_C^{(1)}(\epsilon + \tau) = \pi_C^{(1)}(\epsilon) + \rho_C$ leads to a contradiction with the assumption that the toric code decoder successfully corrects the projected syndrome $\pi_C^{(0)}(\sigma)$ in the projected lattice \mathcal{L}_C . This concludes the proof. \square

We remark that we can interpret the (Successful Decoding) Lemma 12 in the following way: successful decoding of the projected syndrome $\pi_C^{(0)}(\sigma)$ in the toric code on any projected lattice \mathcal{L}_C implies successful color code decoding of the initial syndrome σ . This fact allows us to lower-bound the threshold of the Projection Decoder for the color code on \mathcal{L} by (some function of) the threshold of the Sweep Decoder for the toric code on \mathcal{L}_C , as we will see in Sec. 4.5.

Decoding in 2D revisited and simplified

Now we illustrate color code decoding in the two-dimensional case, as well as describe a practical and very simple implementation of the decoder. We remark that the difficulty of color code decoding comes partly from the fact that for any zero-dimensional syndrome associated with a subset of vertices of the lattice \mathcal{L} we have to find a corresponding two-dimensional recovery operator identified with a subset of faces of \mathcal{L} . We contrast that with toric code decoding, where for the zero-dimensional syndrome we can straightforwardly find (e.g. by using the Minimum-Weight Perfect Matching algorithm) a recovery operator as a one-dimensional object, whose boundary is the given syndrome.

For simplicity, let us consider the triangular lattice \mathcal{L} with 3-colorable vertices, which is a discretization of the two-dimensional torus. For every triangular face we have one qubit associated with it. The stabilizer generators of X - and Z -type are defined for every vertex $v \in \Delta_0(\mathcal{L})$ as a product of Pauli X or Z operators on qubits on neighboring faces. We denote by $\epsilon \subseteq \Delta_2(\mathcal{L})$ the subset of faces, whose qubits are affected by Z errors, and by $\sigma \subseteq \Delta_0(\mathcal{L})$ the corresponding X -type syndrome, $\partial_{2,0}\epsilon = \sigma$; see Fig. 4.9(a).

Unlike in the original version of the 2D color code decoder in [Del14b], we start with only two projected lattices \mathcal{L}_{RG} and \mathcal{L}_{RB} . Note that \mathcal{L}_{RG} (respectively \mathcal{L}_{RB}) is obtained from \mathcal{L} according to the prescription in Sec. 4.4, namely we first remove all the vertices of color B (color G), as well as edges and faces incident to those vertices and then for every removed vertex we add a face; see Fig. 4.9(b). For each restricted syndrome, σ_{RG} and σ_{RB} , we find a subset of edges, respectively $\varrho_{RG} \subseteq \Delta_1(\mathcal{L}_{RG}) \subset \Delta_1(\mathcal{L})$ and $\varrho_{RB} \subseteq \Delta_1(\mathcal{L}_{RB}) \subset \Delta_1(\mathcal{L})$. We achieve that by treating σ_{RG} and σ_{RB} as if they were the syndromes of the toric code on \mathcal{L}_{RG} and \mathcal{L}_{RB} and using any toric code decoder. The final step is local lifting, which finds the correction $\tau \subseteq \Delta_2(\mathcal{L})$ by applying the Lift to every vertex of color R contained in $\varrho_{RG} + \varrho_{RB}$; see Fig. 4.9(c). We emphasize that the final step is the main improvement over the original version of the 2D color code decoder, which requires a global lifting procedure; see Fig. 4.9(d). The Lift is important in our generalization of the Projection Decoder to higher dimensions, since it allows us to construct a fully local color code decoder in 3D, as we will see in the next subsection.

We remark that one can further simplify decoding in two dimensions. Namely, one does not need to construct projected lattices at all! Rather, one restricts the syndrome to either σ_{RG} or σ_{RB} and then treats them as if they were the syndromes

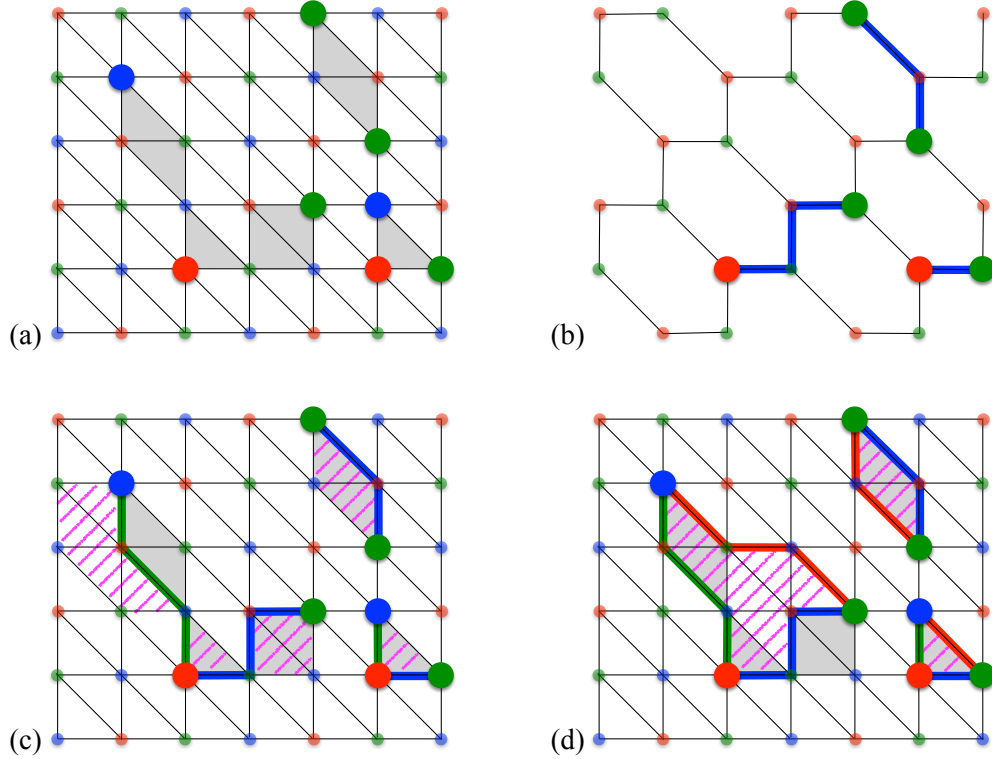


Figure 4.9: The Projection Decoder for the 2D color code on the triangular lattice \mathcal{L} . Note that qubits are placed on faces, whereas the stabilizers are associated with vertices of \mathcal{L} . (a) The qubits on the subset $\epsilon \subseteq \Delta_2(\mathcal{L})$ of faces (shaded in grey) are affected by Pauli Z errors, which result in the X-type syndrome $\sigma = \partial_{2,0}\epsilon$ (highlighted vertices). (b) We can use any toric code decoder to find the correction $\varrho_{RG} \subseteq \Delta_1(\mathcal{L}_{RG})$ (blue thick edges) of the projected syndrome σ_{RG} in the projected lattice \mathcal{L}_{RG} . (c) We find the color code correction $\tau \subseteq \Delta_2(\mathcal{L})$ (hatched in magenta) by a local procedure, i.e., applying the Lift to all red vertices which belong to $\varrho_{RG} + \varrho_{RB}$ (thick blue and green edges). Note that the initial error ϵ and the correction τ are not the same; rather, they form a stabilizer, i.e., $\epsilon + \tau \in \text{im } \partial_{0,2}$. (d) The original 2D color code decoder in [Del14b] requires decoding in three projected lattices \mathcal{L}_{RG} , \mathcal{L}_{RB} and \mathcal{L}_{RB} , as well as a global lifting procedure to find the correction (hatched in magenta).

of the toric code on the original lattice \mathcal{L} . Similarly as before, by using any toric code decoder for σ_{RG} and σ_{RB} we find one-dimensional objects $\widetilde{\varrho}_{RG} \subset \Delta_1(\mathcal{L})$ and $\widetilde{\varrho}_{RB} \subset \Delta_1(\mathcal{L})$. Note that this time $\widetilde{\varrho}_{RG}$ and $\widetilde{\varrho}_{RB}$ may contain vertices of color B or G , respectively. Thus, in the last step we not only apply the Lift to the vertices of color R contained in $\widetilde{\varrho}_{RG} + \widetilde{\varrho}_{RB}$, but also to all the vertices of color B and G contained in $\widetilde{\varrho}_{RG}$ and $\widetilde{\varrho}_{RB}$, respectively. We illustrate this significant simplification of the 2D Projection Decoder in Fig. 4.10.

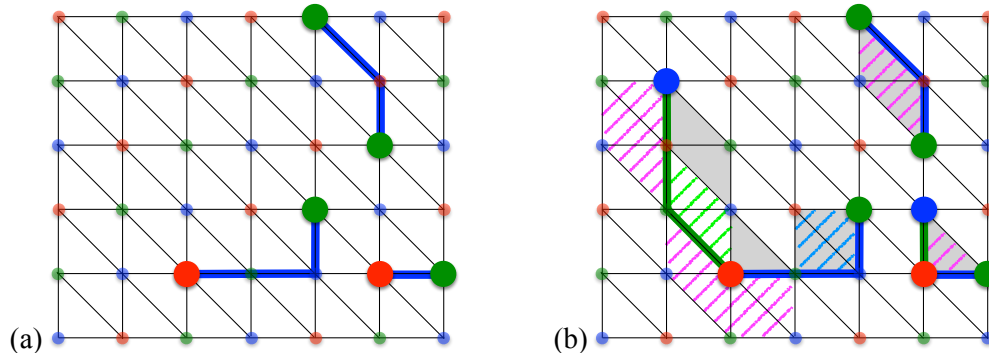


Figure 4.10: The simplified version of the Projection Decoder for the 2D color code. We assume the same error configuration $\epsilon \subseteq \Delta_2(\mathcal{L})$ as in Fig. 4.9. (a) We use any toric code decoder to find the correction $\widetilde{\mathcal{Q}}_{RG} \subset \Delta_1(\mathcal{L})$ (blue thick edges) of the projected syndrome σ_{RG} in the original lattice \mathcal{L} (not the projected lattice!). Note that there are some blue vertices which belong to $\widetilde{\mathcal{Q}}_{RG}$. (b) We find the color code correction $\tau \subseteq \Delta_2(\mathcal{L})$ by applying the Lift to all red vertices belonging to $\widetilde{\mathcal{Q}}_{RG} + \widetilde{\mathcal{Q}}_{RB}$, as well as to blue and green vertices belonging to $\widetilde{\mathcal{Q}}_{RG}$ and $\widetilde{\mathcal{Q}}_{RB}$. Note that the output of the Lift for vertices of color R , G and B is depicted as the faces hatched in magenta, light green and light blue, respectively.

Fully local color code decoder in 3D

We note that it seems impossible to have a strictly local (in the sense of space and time) decoder for two-dimensional topological stabilizer codes, since their syndrome is identified with zero-dimensional objects and they always have one-dimensional string-like logical operators [Bom14b; Yos11]. For instance, in the 2D toric code one could have a pair of violated stabilizers separated by an arbitrarily large distance due to a string-like error. If the decoder used only local information about the syndrome, it would not know the relative positioning of the pair of violated stabilizers and thus could not find a recovery operator. This observation is consistent with all known decoding strategies, which rely on non-local information, either explicitly, as in the case of the Minimum-Weight Perfect Matching algorithm, or implicitly, as in the case of the decoder by Harrington with the hierarchical structure of update rules.

It is possible, however, to have a fully local decoder of the color code in three dimensions, since one type of the syndrome necessarily forms one-dimensional loop-like objects. We remark that if one wanted a fully local decoder for both X - and Z -type of errors, then one would need to consider the 4D color code of type $k = 2$. For simplicity of discussion, we now present the three-dimensional case with $k = 2$ since it captures all the important aspects of local decoding based on the Sweep Rule.

Similarly as in the two-dimensional case, we do not need to construct projected lattices. Let $\epsilon \subseteq \Delta_3(\mathcal{L})$ be the set of qubits affected by Z errors and $\sigma = \partial_{3,1}\epsilon$ be the corresponding X -type syndrome. It suffices to consider three restrictions of the syndrome $\sigma_{RGB}, \sigma_{RGY}, \sigma_{RBY}$ and for each of them use the Sweep Decoder in the original lattice \mathcal{L} to find $\widetilde{\mathcal{Q}}_{RGB} \subset \Delta_2(\mathcal{L}), \widetilde{\mathcal{Q}}_{RGY} \subset \Delta_2(\mathcal{L})$ and $\widetilde{\mathcal{Q}}_{RBY} \subset \Delta_2(\mathcal{L})$. Finally, in order to find the correction $\tau \subseteq \Delta_3(\mathcal{L})$ we apply the local lifting procedure. As in the two-dimensional case, $\widetilde{\mathcal{Q}}_{RGB}$ may contain vertices of color Y since we do not use the projected lattice \mathcal{L}_{RGB} but the original lattice \mathcal{L} ; similarly for $\widetilde{\mathcal{Q}}_{RGY}$ and $\widetilde{\mathcal{Q}}_{RBY}$. Thus, we need to apply the Lift to all the vertices of color R , which belong to $\widetilde{\mathcal{Q}}_{RGB} + \widetilde{\mathcal{Q}}_{RGY} + \widetilde{\mathcal{Q}}_{RBY}$, as well as to all the vertices of color Y, B and G belonging to $\widetilde{\mathcal{Q}}_{RGB}, \widetilde{\mathcal{Q}}_{RGY}$ and $\widetilde{\mathcal{Q}}_{RBY}$, respectively.

For the convenience of the reader we summarize the simplified Projection Decoder of the Z -type errors in the 3D color code of type $k = 2$ on the lattice \mathcal{L} .

1. For every triple of colors $C \in \{RGB, RGY, RBY\}$ use the Sweep Decoder on the lattice \mathcal{L} for the projected syndrome σ_C in order to find $\widetilde{\mathcal{Q}}_C$.
2. Apply the Lift to every vertex of color R belonging to $\widetilde{\mathcal{Q}}_{RGB} + \widetilde{\mathcal{Q}}_{RGY} + \widetilde{\mathcal{Q}}_{RBY}$, as well as to every vertex of color $RGBY \setminus C$ belonging to $\widetilde{\mathcal{Q}}_C$.

Since this simplified version of the color code decoder is fully local and only uses the initial lattice \mathcal{L} (instead of at least three projected lattices), thus it is relatively easy to implement.

4.5 Thresholds of 3D toric and color code decoders

In this section, we present our numerical estimates of the error-correction threshold of the Projection Decoder for point-like excitations in the 3D color code on the bcc lattice; see Fig. 4.11. In particular, we implement the Projection Decoder based on the Minimum-Weight Perfect Matching algorithm and the local Lift. As a necessary step toward estimating the color code threshold, we find the thresholds of the Minimum-Weight Perfect Matching algorithm for point-like excitations in the 3D toric code on the cubic and half-bcc lattices. We also discuss an analytic lower bound on the threshold of the Projection Decoder in terms of the toric code threshold. Lastly, we numerically investigate the performance of the Sweep Decoder for the 3D toric code in the presence of faulty measurements.

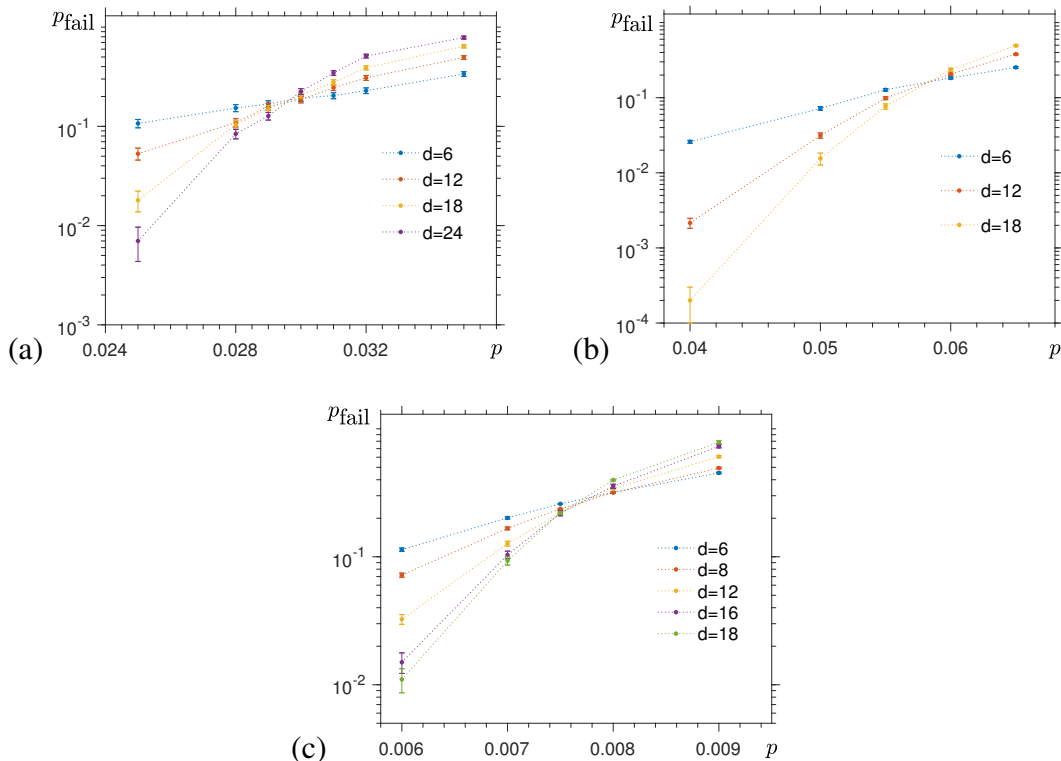


Figure 4.11: The failure probability $p_{\text{fail}}(p, d)$ of the Minimum-Weight Perfect Matching algorithm for point-like excitations in the 3D toric code on (a) the cubic lattice and (b) the half-bcc lattice as a function of the error rate p and the linear size of the lattice d . We can estimate the error-correction thresholds $q_{\text{cub}} \approx 0.0295$ and $q_{\text{half}} \approx 0.058$ by plotting $p_{\text{fail}}(p, d)$ for different d and finding their crossing point. (c) The failure probability $p_{\text{fail}}(p, d)$ of the Projection Decoder for the point-like excitations of the 3D color code on the bcc lattice. We estimate the threshold of the Projection Decoder to be $q_{\text{bcc}} \approx 0.0077$.

Analytic lower bound on color code threshold

We can find an analytic lower-bound on the error-correction threshold of the Projection Decoder for the color code in terms of the threshold of the corresponding toric code decoder. Namely, from the (Successful Decoding) Lemma 12 we know that if decoding of the toric code on different projected lattices succeeds, then the Projection Decoder of the color code successfully corrects the error. We assume that the errors appearing in the color code are captured by the X -type bit-flip or Z -type phase-flip noise models. Then, one can show that the effective noise model in the toric code on any projected lattice is also described by the bit-flip or phase-flip noise models. Importantly, no correlations between errors within the same projected toric code are introduced. However, the effective strength of the noise changes and it may vary for different qubits.

For concreteness, let us consider the 3D color code on the bcc lattice. Let p be the Z -type phase-flip error rate affecting the qubits of the color code. The projected lattices form either the cubic lattice or the half-bcc lattice (which is obtained e.g. by keeping vertices of color red or blue in Fig. 4.7). We can find that the effective physical error rates for the toric code on the cubic and half-bcc lattices are

$$p_{\text{cub}}(p) = 4p(1-p)^3 + 4p^3(1-p), \quad (4.72)$$

$$p_{\text{half}}(p) = 6p(1-p)^5 + 20p^3(1-p)^3 + 6p^5(1-p). \quad (4.73)$$

This follows from the fact that there is an error on the edge in the projected lattice iff there is an odd number of errors on tetrahedra containing that edge in the original lattice. From our numerical simulations presented in Fig. 4.11(a)(b) we estimate the toric code thresholds on the cubic and half-bcc lattices to be $q_{\text{cub}} \approx 0.0295$ and $q_{\text{half}} \approx 0.058$, respectively. We conclude that if the physical error rate p in the color code satisfies two conditions, $p_{\text{cub}}(p) \leq q_{\text{cub}}$ and $p_{\text{half}}(p) \leq q_{\text{half}}$, then the projected toric code decoders are guaranteed to succeed with high probability, and thus the color code decoder succeeds as well. This leads us to the following lower bound on the Projection Decoder threshold for the color code on the bcc lattice

$$q_{\text{bcc}} \geq \min\{p_{\text{cub}}^{-1}(q_{\text{cub}}), p_{\text{half}}^{-1}(q_{\text{half}})\}, \quad (4.74)$$

where $p_{\text{cub}}^{-1}(\cdot)$ denotes the inverse function of $p_{\text{cub}}(\cdot)$; similarly $p_{\text{half}}^{-1}(\cdot)$. We remark that the numerical value 0.0075 of the lower bound from Eq. (4.74) is consistent with our estimates of the Projection Decoder threshold $q_{\text{bcc}} \approx 0.0077$ from Fig. 4.11(c).

Toom's rule and the phenomenological noise model

So far in our discussion of various decoders we have assumed perfect syndrome extraction. However, a realistic scenario of error correction should accommodate for the possibility of incorrect syndrome measurements. Thus, in the rest of this section we numerically investigate the performance of local decoders for the phenomenological noise model. For concreteness, we focus on the Sweep Decoder based on Toom's rule for the 3D toric code (of type $k = 2$) on the cubic lattice \mathcal{L} with qubits on faces; see Fig. 4.2(d)-(f). We study the phenomenological noise model with the phase-flip error rate p matching the probability of the incorrect stabilizer measurement. Our numerical simulation can be succinctly described as follows.

1. Initialize the residual error $\tilde{\epsilon} \subseteq \Delta_2(\mathcal{L})$ by randomly choosing faces of \mathcal{L} with probability p .

2. For the imperfect correction cycle $i = 1, 2, \dots, N_{\text{cyc}}$:
 - (i) find the faulty syndrome $\tilde{\sigma} \subseteq \Delta_1(\mathcal{L})$ corresponding to the residual error $\tilde{\epsilon}$,
 - (ii) apply only one time step of the Sweep Decoder for $\tilde{\sigma}$ to find a one-time-step correction $\tilde{\varrho} \subseteq \Delta_2(\mathcal{L})$,
 - (iii) find a new error $\epsilon \subseteq \Delta_2(\mathcal{L})$ by randomly choosing faces of \mathcal{L} with probability p and update the residual error $\tilde{\epsilon} \leftarrow \tilde{\epsilon} + \tilde{\varrho} + \epsilon$.
3. Find the exact syndrome $\sigma = \partial_2 \tilde{\epsilon}$ and run the Sweep Decoder for σ to get the correction $\varrho \subseteq \Delta_2(\mathcal{L})$.
4. Check if the correction succeeds, i.e., $\tilde{\epsilon} + \varrho \in \text{im } \partial_2$.

We emphasize that the faulty syndrome $\tilde{\sigma}$ does not have to be a 1-boundary in \mathcal{L} since we allow for imperfect syndrome extraction. Subsequently, the Sweep Rule may not be able to find for some extremal vertex v a subset $\varphi(v) \subseteq \text{St}_2(v)$ of faces, whose spins should be flipped. In this case, i.e., when $\tilde{\sigma}|_v$ is not a restriction of any valid syndrome, we modify the Sweep Rule to do nothing.

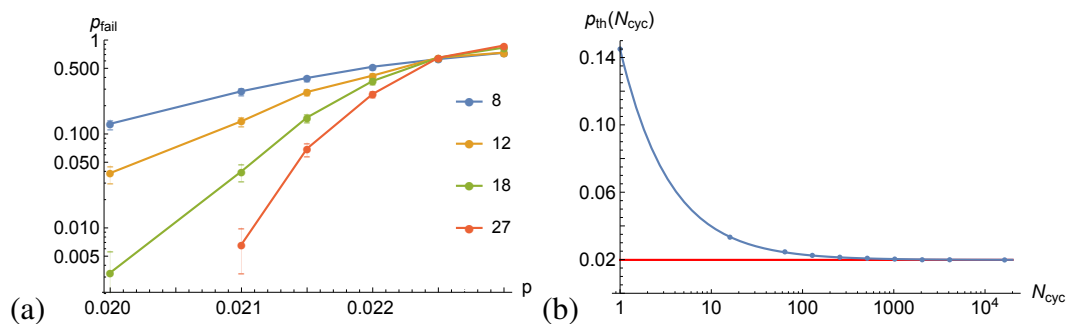


Figure 4.12: (a) The failure probability $p_{\text{fail}}(p, d)$ of the Sweep Decoder based on Toom's rule for the 3D toric code on the cubic lattice after $N_{\text{cyc}} = 2^7$ correction cycles, where p is the error rate and d is the linear size of the lattice. We estimate the threshold $p_{\text{th}}(N_{\text{cyc}}) \approx 0.0225$ from the crossing point of different curves. (b) The sustainable threshold p_{sus} is defined to be the limit of $p_{\text{th}}(N_{\text{cyc}})$ as $N_{\text{cyc}} \rightarrow \infty$. We estimate $p_{\text{sus}} \approx 0.02$ by fitting the numerical ansatz from Eq. (4.75) (blue line) to the data.

In order to analyze the performance of the Sweep Decoder for the phenomenological noise model, we fix the number of correction cycles N_{cyc} . Then, using Monte Carlo simulations we estimate the decoder failure probability $p_{\text{fail}}(p, d)$ and plot it as a function of the error rate p for different lattices of the linear size d ; see Fig. 4.12(a).

This allows us to identify the threshold $p_{\text{th}}(N_{\text{cyc}})$. Note that the threshold $p_{\text{th}}(0)$ after zero correction cycles corresponds to the threshold of the Sweep Decoder for perfect syndrome extraction. We are, however, interested in the so-called sustainable threshold $p_{\text{sus}} = \lim_{N_{\text{cyc}} \rightarrow \infty} p_{\text{th}}(N_{\text{cyc}})$ defined as the limit of $p_{\text{th}}(N_{\text{cyc}})$ as the number of correction cycles N_{cyc} goes to infinity [BNB15; Ter15]. We find that the threshold $p_{\text{th}}(N_{\text{cyc}})$ is very well approximated by the following numerical ansatz

$$p_{\text{th}}(N_{\text{cyc}}) \sim p_{\text{sus}}(1 - (1 - p_{\text{th}}(0)/p_{\text{sus}})N_{\text{cyc}}^{-\gamma}) \quad (4.75)$$

with the fitting parameters γ and p_{sus} . This allows us to estimate the sustainable threshold of the Sweep Decoder for the 3D toric code on the cubic lattice to be $p_{\text{sus}} \approx 0.02$; see Fig. 4.12(b).

4.6 Discussion

In this chapter, we proposed a simple local update rule for classical ± 1 spins, the Sweep Rule. It is a generalization of Toom's rule in the sense that it can be used to shrink k -dimensional domain walls on any locally Euclidean d -dimensional lattice, where $k \in \{1, \dots, d-1\}$. We used the Sweep Rule to design a local decoder of the toric code and rigorously prove its non-zero threshold. We could readily use this decoder for other topological code, the color code, since we found a generic reduction of the problem of color code decoding to that of the toric code. However, it would be worth thinking of a version of the Sweep Rule which would work natively for the color code syndrome. We believe that such a modified local rule necessarily uses the information about the colorability of the lattice.

In our discussion we mostly focused on decoding with perfect stabilizer measurements. We also numerically investigated the performance of local decoders for the phenomenological noise model. It seems that local decoders can be applied to the phenomenological noise model without much modification. Thus, a conceivable extension of our work could be to incorporate measurement errors in the proof of a non-zero threshold.

Finally, we remark that a question not immediately related to decoding but nevertheless worth exploring would be to study the spin dynamics generated by a non-deterministic version of the Sweep Rule. Namely, at each time step every spin is first updated according to the Sweep Rule, and then a probabilistic flip can happen depending on the value of the spin, as in the case of non-deterministic Toom's rule. In particular, one could explore the phase diagram of the underlying spin model and possibly find regions, where multiple stable phases coexist.

THREE-DIMENSIONAL COLOR CODE THRESHOLDS VIA STATISTICAL-MECHANICAL MAPPING

Some approaches to building scalable quantum computers are more practical than others due to their more favorable noise and resource requirements. The two-dimensional (2D) surface code approach [Kit03; BK98; Den+02] has very desirable features: (1) geometrically local syndrome measurements, (2) a high accuracy and (3) fault-tolerant Clifford gates with low overhead. Unfortunately, the surface code is not known to admit a (4) fault-tolerant non-Clifford gate with low overhead. The formidable qubit overhead cost of state distillation [BK05; Fow+12] for the necessary non-Clifford gate motivates the quest for alternatives to the surface code with all features (1)–(4).

Such alternatives may be sought in the general class of topological codes [Kit03; BK98; LW05; BM06; Bom13], which includes the surface code as a special case. By definition, topological codes require only geometrically local syndrome measurements and tend to have high accuracy thresholds. Topological codes often admit some fault-tolerant transversal gates (implemented by the tensor product of single-qubit unitaries), which have low overhead cost. However, no quantum error-detecting code (whether topological or not), has a universal transversal encoded gate set [ZCC11; EK09].

Here we focus on the 3D topological color codes [BM07a; Bom15a] closely related to the 3D toric code [KYP15], which come in two types. The stabilizer type has 1D string-like Z and 2D sheet-like X logical operators, and a logical non-Clifford gate $T = \text{diag}(1, e^{i\pi/4})$ is transversal. In the subsystem type, there are 1D string-like X and Z dressed logical operators, and all logical Clifford gates are transversal. Moreover, in the subsystem color code it is possible to reliably detect measurement errors in a single time step [Bom15b; BNB15]. By fault-tolerantly switching between the stabilizer and subsystem color codes [Bom15a; KB15], one can combine the desirable features (1), (3) and (4).

In this chapter, we address feature (2) for the 3D color codes by finding thresholds $p_{3\text{DCC}}^{(1)} \simeq 1.9\%$ and $p_{3\text{DCC}}^{(2)} \simeq 27.5\%$ for phase-flip Z and bit-flip X noise, respectively. Our results assume optimal decoders for independent X and Z noise

with perfect measurements, and thereby give fundamental error-correction bounds against which efficient, but suboptimal decoders (such as that studied in [BNB15]) can be compared. These thresholds are comparable to the analogous thresholds for the cubic lattice 3D toric code: $p_{3\text{DTC}}^{(1)} \simeq 3.3\%$ and $p_{3\text{DTC}}^{(2)} \simeq 23.5\%$ [OI98; Has+07; Ohn+04], but compare unfavorably to $p_{2\text{DTC}} \simeq 10.9\%$ for the square lattice 2D toric code [HPP01].

Our approach extends techniques known for other codes [Den+02; WHP03; KP15; Bom13; KBM09; Bom+12; And12] in order to relate the 3D color code thresholds to phase transitions in two new 3D statistical-mechanical models: the 4- and 6-body random coupling Ising models (RCIM). We use large-scale parallel tempering Monte Carlo simulations [HN96] and analyze specific heat, sublattice magnetization and Wilson loop operators to map the relevant parts of the disorder-temperature (p, T)-phase diagram, see Fig. 5.1. The 6-body RCIM is an example of a lattice gauge theory with a local $\mathbb{Z}_2 \times \mathbb{Z}_2$ symmetry, which makes this model both interesting and

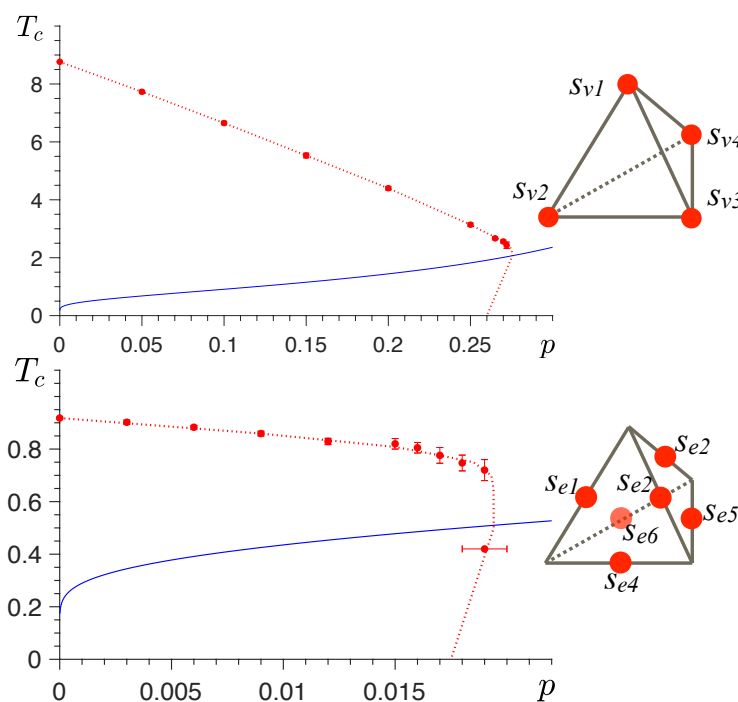


Figure 5.1: The disorder-temperature (p, T)-phase diagrams of the 4-body (top) and 6-body (bottom) 3D random coupling Ising models. Both models are defined on the 3D body-centered cubic lattice built of tetrahedra. The 4- and 6-body models have spins on vertices and edges, respectively. The error correction threshold p_c can be found as the intersection of the Nishimori line (blue line) with the anticipated phase boundary (red dotted line).

challenging to study.

5.1 3D stabilizer color code

Let \mathcal{L} be a three-dimensional lattice built of tetrahedra such that its vertices are 4-colorable, i.e. vertices connected by an edge are of different colors. An example of such a lattice is the body-centered cubic (bcc) lattice obtained from two interleaved cubic lattices, see Fig. 5.2(b). We denote by $\Delta_i(\mathcal{L})$ the set of all i -simplices of \mathcal{L} . Then, 0-simplices of \mathcal{L} are vertices, 1-simplices are edges, etc. We place one qubit at every tetrahedron $t \in \Delta_3(\mathcal{L})$. For every vertex $v \in \Delta_0(\mathcal{L})$ and edge $e \in \Delta_1(\mathcal{L})$ we define operators $S_X(v)$ and $S_Z(e)$ to be the product of either Pauli X or Z operators on qubits identified with tetrahedra in the neighborhood of the vertex v or edge e , namely

$$S_X(v) = \prod_{\substack{t \in \Delta_3(\mathcal{L}) \\ t \supset v}} X(t), \quad S_Z(e) = \prod_{\substack{t \in \Delta_3(\mathcal{L}) \\ t \supset e}} Z(t). \quad (5.1)$$

The 3D stabilizer¹ color code is defined by specifying its stabilizer group [Got96]

$$\mathcal{S} = \langle S_X(v), S_Z(e) | v \in \Delta_0(\mathcal{L}), e \in \Delta_1(\mathcal{L}) \rangle. \quad (5.2)$$

Using the colorability condition one can show that \mathcal{S} is an Abelian subgroup of the Pauli group not containing $-I$. The code space is the +1 eigenspace of all elements of \mathcal{S} and the lowest-weight logical X and Z operators of the 3D color code are 2D sheet-like and 1D string-like objects, see Fig. 5.2(a). In general, the color code can be defined in $d \geq 2$ dimensions on a lattice, provided it is a $(d + 1)$ -colorable simplicial d -complex [KB15].

Error correction in CSS codes

Since the color code is a CSS code [Cal+97], we choose to separately correct X - and Z -type errors, which simplifies the discussion. We also assume perfect measurements. For concreteness, we focus on X -error correction; Z -errors can be analyzed analogously².

The set of all Z -type stabilizers which return -1 measurement outcomes is called a Z -type syndrome. Note that any nontrivial Z -syndrome signals the presence of some X -errors in the system. Correction of X -errors in a CSS code can be succinctly

¹ The 3D subsystem color code has the gauge group $\mathcal{G} = \langle \prod_{\Delta_3(\mathcal{L}) \ni t \supset e} X(t), \prod_{\Delta_3(\mathcal{L}) \ni t \supset e} Z(t) | e \in \Delta_1(\mathcal{L}) \rangle$ and the stabilizer group $\mathcal{S} = \langle \prod_{\Delta_3(\mathcal{L}) \ni t \supset v} X(t), \prod_{\Delta_3(\mathcal{L}) \ni t \supset v} Z(t) | v \in \Delta_0(\mathcal{L}) \rangle$.

² For correction of Z -errors, exchange $X \leftrightarrow Z$ and redefine $\mathcal{B}_2 = \Delta_1(\mathcal{L})$, $\mathcal{B}_1 = \Delta_3(\mathcal{L})$, $\mathcal{B}_0 = \Delta_0(\mathcal{L})$, $\partial_2 e = \sum_{\Delta_3(\mathcal{L}) \ni t \supset e} t$ and $\partial_1 t = \sum_{\Delta_0(\mathcal{L}) \ni v \subset t} v$ for any $e \in \Delta_1(\mathcal{L})$ and $t \in \Delta_3(\mathcal{L})$.

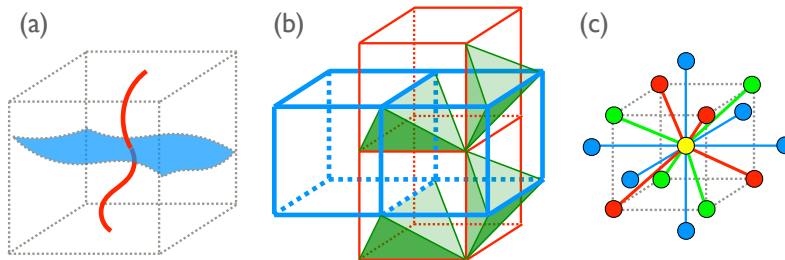


Figure 5.2: (a) The 3D stabilizer color code has both 1D string-like (red) and 2D sheet-like (blue) logical operators. (b) The bcc lattice can be constructed starting from two interleaved cubic lattices (red and blue) and filling in with tetrahedra (green). Not all tetrahedra are depicted. (c) The neighborhood of any vertex in the bcc lattice looks the same — every vertex belongs to 24 edges, 36 triangular faces and 24 tetrahedra. The bcc lattice is 4-colorable, i.e. every vertex is colored in red, green, blue or yellow, and no two neighboring vertices are of the same color.

described by introducing a chain complex [FM01; Del14b]

$$\begin{array}{ccccc}
 C_2 & \xrightarrow{\partial_2} & C_1 & \xrightarrow{\partial_1} & C_0 \\
 X\text{-stabilizers} & & \text{qubits} & & Z\text{-stabilizers}
 \end{array} \tag{5.3}$$

where C_2 , C_1 and C_0 are vector spaces over \mathbb{Z}_2 with bases $\mathcal{B}_2 = X$ -stabilizer generators, $\mathcal{B}_1 =$ physical qubits and $\mathcal{B}_0 = Z$ -stabilizer generators, respectively. The linear maps ∂_2 and ∂_1 , called boundary operators, are chosen in such a way that support of any X -stabilizer $\omega \in C_2$ is given by $\partial_2\omega$, and the Z -syndrome corresponding to any X -error $\epsilon \in C_1$ can be found as $\partial_1\epsilon$. Note that $\partial_1 \circ \partial_2 = 0$, since any X -stabilizer has a trivial Z -syndrome. One can think of the boundary operators as parity-check matrices H_X^T and H_Z of the CSS code. In the case of the 3D color code, C_2 , C_1 , C_0 are generated by vertices, tetrahedra, and edges respectively, i.e. $\mathcal{B}_2 = \Delta_0(\mathcal{L})$, $\mathcal{B}_1 = \Delta_3(\mathcal{L})$ and $\mathcal{B}_0 = \Delta_1(\mathcal{L})$. The boundary operators are defined to be $\partial_2 v = \sum_{\Delta_3(\mathcal{L}) \ni t \supset v} t$ and $\partial_1 t = \sum_{\Delta_1(\mathcal{L}) \ni e \subset t} e$ for any $v \in \Delta_0(\mathcal{L})$ and $t \in \Delta_3(\mathcal{L})$.

Let $\epsilon, \varphi \in C_1$ be two X -errors with the same Z -syndrome, $\partial_1\epsilon = \partial_1\varphi$. We say that ϵ and φ are equivalent iff they differ by some X -stabilizer $\omega \in C_2$, namely $\epsilon + \varphi = \partial_2\omega$. To correct errors, we need a decoder — an algorithm which takes the Z -syndrome $\sigma \in C_0$ as an input and returns a Z -correction φ which will restore all X -stabilizers to have +1 outcomes, i.e. $\partial_1\varphi = \sigma$. The decoder succeeds iff the actual error ϵ and the correction φ are equivalent. An optimal decoder finds a representative $\bar{\varphi}$ of the most probable equivalence class of errors $\bar{\varphi} = \{\varphi + \partial_2\omega \mid \forall \omega \in C_2\}$.

5.2 Statistical-mechanical models

In this section, we provide a brief derivation of the connection between optimal error-correction thresholds and phase transitions [Den+02; WHP03; KP15; Bom13; KBM09; Bom+12; And12]. In particular, we derive two new statistical-mechanical models relevant for the 3D color code.

We assume bit-flip noise, i.e. every qubit is independently affected by Pauli X error with probability p . The probability of an X -error $\epsilon \in C_1$ affecting the system is

$$\text{pr}(\epsilon) = \prod_{j \in \mathcal{B}_1} p^{[\epsilon]_j} (1-p)^{1-[\epsilon]_j} \propto \left(\frac{p}{1-p} \right)^{\sum_{j \in \mathcal{B}_1} [\epsilon]_j}, \quad (5.4)$$

where $[\epsilon]_j \in \mathbb{Z}_2$ denotes the j coefficient of ϵ in \mathcal{B}_1 basis, $\epsilon = \sum_{j \in \mathcal{B}_1} [\epsilon]_j j$.

For a general CSS code family with the chain complex in Eq. (5.3), the X -error correction threshold is the largest p_c such that for all $p < p_c$ the probability of successful decoding goes to 1 in the limit of infinite system size

$$\text{pr}(\text{succ}) = \sum_{\epsilon \in C_1} \text{pr}(\epsilon) \text{pr}(\text{succ}|\epsilon) \rightarrow 1. \quad (5.5)$$

With the optimal decoder, the conditional probability $\text{pr}(\text{succ}|\epsilon)$ equals 1 if ϵ belongs to the most probable error equivalence class consistent with the syndrome $\partial_1 \epsilon$, and 0 otherwise. The probability of equivalence class $\bar{\epsilon}$ is

$$\text{pr}(\bar{\epsilon}) = \sum_{\omega \in C_2} \text{pr}(\epsilon + \partial_2 \omega) \propto \sum_{\omega \in C_2} e^{-2\beta(p) \sum_{j \in \mathcal{B}_1} [\epsilon + \partial_2 \omega]_j}, \quad (5.6)$$

where we use Eq. (5.4) and introduce

$$\beta(p) = -\frac{1}{2} \log \frac{p}{1-p}. \quad (5.7)$$

To rewrite Eq. (5.6), we use two equalities

$$[\partial_2 \omega]_j \equiv \sum_{i \in \mathcal{B}_2 \wedge \partial_2 i \ni j} [\omega]_i \pmod{2}, \quad (5.8)$$

$$1 - 2[\epsilon + \partial_2 \omega]_j = (-1)^{[\epsilon]_j} (-1)^{[\partial_2 \omega]_j} = (-1)^{[\epsilon]_j} \prod_{i \in \mathcal{B}_2 \wedge \partial_2 i \ni j} (-1)^{[\omega]_i}. \quad (5.9)$$

By introducing new (classical spin) variables $s_i = (-1)^{[\omega]_i}$ for all $i \in \mathcal{B}_2$, we can replace the sum over $\omega \in C_2$ in Eq. (5.6) by a sum over different configurations $\{s_i = \pm 1\}$, yielding

$$\text{pr}(\bar{\epsilon}) \propto \sum_{\{s_i = \pm 1\}} e^{-\beta(p) H_\epsilon(\{s_i\})}, \quad (5.10)$$

where we introduce the Hamiltonian

$$H_\epsilon(\{s_i\}) = - \sum_{j \in \mathcal{B}_1} (-1)^{[\epsilon]_j} \prod_{\substack{i \in \mathcal{B}_2 \\ [\partial_2 i]_j = 1}} s_i. \quad (5.11)$$

We define the random coupling Ising model (RCIM) to be a classical spin $s_i = \pm 1$ random model with quenched couplings $(-1)^{[\epsilon]_j}$ described by $H_\epsilon(\{s_i\})$ in Eq. (5.11). The RCIM has two independent parameters: disorder strength p (i.e. the probability of negative couplings) and inverse temperature β . The partition function of the RCIM with disorder ϵ at temperature β^{-1} is given by

$$Z_\epsilon(\beta) = \sum_{\{s_i = \pm 1\}} e^{-\beta H_\epsilon(\{s_i\})}. \quad (5.12)$$

Note that for the proportionality $\text{pr}(\bar{\epsilon}) \propto Z_\epsilon(\beta)$ in Eq. (5.12) to hold one requires $\beta = \beta(p)$.

For the 3D color code, Eq. (5.11) leads to the following two new statistical-mechanical models

$$H_\epsilon^X(\{s_v\}) = - \sum_{t \in \Delta_3(\mathcal{L})} (-1)^{[\epsilon]_t} \text{img}_1, \quad (5.13)$$

$$H_\epsilon^Z(\{s_e\}) = - \sum_{t \in \Delta_3(\mathcal{L})} (-1)^{[\epsilon]_t} \text{img}_2, \quad (5.14)$$

relevant to correction of X - and Z -errors, respectively. Note that $H_\epsilon^X(\{s_v\})$ (respectively $H_\epsilon^Z(\{s_e\})$) contains 4-body (6-body) terms, which are products of vertex (edge) spins of every tetrahedron. We observe that for $p = 0$, i.e. the case with no disorder, these two models are mutually dual in the sense that the low-temperature expansion of each model matches the high-temperature expansion of the other [Weg71]. We are going to explain this duality in the following section.

5.3 Duality of models for zero disorder

We already mentioned that the 4- and 6-body RCIM described by Eqs. (5.13) and (5.60) are dual for $p = 0$, i.e., the case with no disorder. Here we say that two models are dual if the low-temperature expansion of the partition function of one model matches the high-temperature expansion of the partition function of the other and vice versa. For concreteness, we consider the spin models on the bcc lattice, however similar reasoning is applicable to any valid color code lattice. Let us follow the standard procedure [Weg71; Kar07] and write a series expansion for the partition function of the 4-body RCIM in two limits.

- Low temperature $\beta^X \gg 1$ — there are 8 ground states, which are related by the global $\mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_2$ symmetry to the ferromagnetic state with $s_v = +1$ for all spins and thus one can expand the partition function by including low energy excitations around the ground space to obtain

$$Z_0^X(\beta^X) = \sum_{\{s_v = \pm 1\}} e^{-\beta^X H_0^X(\{s_v\})} \propto e^{\beta^X |\Delta_3(\mathcal{L})|} (1 + |\Delta_0(\mathcal{L})| e^{-48\beta^X} + \dots). \quad (5.15)$$

- High temperature $\beta^X \ll 1$ — we treat spins independently and expand the partition function in powers of $\tanh \beta^X$ to obtain

$$Z_0^X(\beta^X) = \sum_{\{s_v = \pm 1\}} e^{-\beta^X H_0^X(\{s_v\})} = \sum_{\{s_v = \pm 1\}} \prod_{t \in \Delta_3(\mathcal{L})} \exp(\beta^X \text{tetra}) \quad (5.16)$$

$$= (\cosh \beta^X)^{|\Delta_3(\mathcal{L})|} \sum_{\{s_v = \pm 1\}} \prod_{t \in \Delta_3(\mathcal{L})} (1 + \text{tetra} \tanh \beta^X) \quad (5.17)$$

$$= 2^{|\Delta_0(\mathcal{L})|} (\cosh \beta^X)^{|\Delta_3(\mathcal{L})|} \quad (5.18)$$

$$\times (1 + E_4 (\tanh \beta^X)^4 + E_6 (\tanh \beta^X)^6 + \dots), \quad (5.19)$$

where $E_4 = \frac{3}{7} \Delta_1(\mathcal{L})$ and $E_6 = \frac{4}{7} \Delta_1(\mathcal{L})$ are the numbers of edges whose neighborhoods (i.e., 3-stars) contain 4 and 6 tetrahedra, respectively³. Note that contributions to the second and third terms in the expansion are due to selecting an edge $e \in \Delta_1(\mathcal{L})$ and multiplying all 4 or 6 (depending on the edge) weight-four spin operators identified with tetrahedra containing e , which results in all spin terms canceling each other.

In a similar way we can find a series expansion for the partition function of the 6-body RCIM on the bcc lattice in the limit of low and high temperatures.

- Low temperature $\beta^Z \gg 1$ — the number of lowest-energy configurations (related by local $\mathbb{Z}_2 \times \mathbb{Z}_2$ gauge transformations) is proportional to $2^{2|\Delta_0(\mathcal{L})|}$ and expanding around the ground space

$$Z_0^Z(\beta^Z) = \sum_{\{s_e = \pm 1\}} e^{-\beta^Z H_0^Z(\{s_e\})} \quad (5.20)$$

$$\propto 2^{2|\Delta_0(\mathcal{L})|} e^{\beta^Z |\Delta_3(\mathcal{L})|} (1 + E_4 e^{-8\beta^Z} + E_6 e^{-12\beta^Z} + \dots). \quad (5.21)$$

³ If we choose the coordinates of vertices of the bcc lattice to be triples of either integers or half-integers, then edges surrounded by 4 tetrahedra are aligned with directions $(1, 0, 0)$, $(0, 1, 0)$ and $(0, 0, 1)$, whereas edges surrounded by 6 tetrahedra are aligned with directions $(\pm 1, \pm 1, \pm 1)$.

- High temperature $\beta^Z \ll 1$ — expand the partition function in powers of $\tanh \beta^Z$ to obtain

$$Z_0^Z(\beta^Z) = (\cosh \beta^Z)^{|\Delta_3(\mathcal{L})|} \sum_{\{s_e = \pm 1\}} \prod_{t \in \Delta_3(\mathcal{L})} \left(1 + \text{tetrahedron} \tanh \beta^Z\right) \quad (5.22)$$

$$= 2^{|\Delta_1(\mathcal{L})|} (\cosh \beta^Z)^{|\Delta_3(\mathcal{L})|} (1 + |\Delta_0(\mathcal{L})| (\tanh \beta^Z)^{24} + \dots), \quad (5.23)$$

where contributions to the second term in the expansion are due to selecting a vertex $v \in \Delta_0(\mathcal{L})$ and multiplying all 24 weight-six spin operators identified with tetrahedra in the neighborhood (i.e., the 3-star) of v so that spin terms cancel each other.

We conclude that if $e^{-2\beta^Z} = \tanh \beta^X$ (or equivalently $e^{-2\beta^X} = \tanh \beta^Z$), then the terms in the series expansions in Eqn. (5.19) and (5.21) match each other up to a constant prefactor; similarly for terms in Eqn. (5.15) and (5.23). In particular, if there is only one phase transition in the first model at temperature T_c^X , then there is a unique phase transition in the dual model at temperature

$$T_c^Z = -2 \left(\log \tanh \frac{1}{T_c^X} \right)^{-1}. \quad (5.24)$$

This serves as a consistency check for our results. Indeed, for zero disorder $p = 0$ the critical temperatures $T_c^X = 8.77(2)$ and $T_c^Z = 0.918(3)$ for the 4- and 6-body RCIM are related according to Eq. (5.24) within the statistical uncertainty.

5.4 Heuristic estimates of the threshold

For a family of topological CSS codes, such as the toric or color codes, one can use the following, *nonrigorous* reasoning to arrive at the *heuristic estimates* of thresholds p^X and p^Z for the bit-flip and phase-flip noise models with optimal decoders and perfect measurements. Let L denote the linear system size, $n = n(L)$ be the number of physical qubits, $N_X = N_X(L)$ be the number of independent X -type stabilizers, and p be the phase-flip error rate. Thus, the average number of Z -type errors in the system is np , the number of different error patterns is $\binom{n}{np}$, and the number of different X -type syndromes is 2^{N_X} . As long as the error rate $p \leq p^Z$ we expect the following inequality to hold

$$\text{number of } Z\text{-error configurations} \lesssim \text{number of } X\text{-syndrome configurations}, \quad (5.25)$$

which after taking the logarithm of both sides and using Stirling's approximation leads to $nH(p) \approx \log_2 \binom{n}{np} \lesssim N_X$, where $H(p) = -p \log_2 p + (1-p) \log_2(1-p)$ is

the Shannon entropy. Thus, the heuristic estimate of the phase-flip threshold p^Z is obtained as the solution of the equation

$$H(p^Z) = \lim_{L \rightarrow \infty} \frac{N_X}{n}, \quad (5.26)$$

such that $p^Z \leq 1/2$; one similarly obtains the estimate of the bit-flip threshold p^X .

Let us consider the 3D color code on the bcc lattice of linear size L , which contains $V = 2L^3$ vertices, $E = 14L^3$ edges, $F = 24L^3$ faces and $T = 12L^3$ tetrahedra. Since the number of physical qubits is $n = T$ and the number of independent X -stabilizers is $N_X = V - 3$, thus by solving Eq. (5.26) we arrive at the heuristic estimate of the threshold $p_{3\text{DCC}}^Z \simeq 0.0246$. Similarly, we obtain the heuristic estimate of the threshold $p_{3\text{DCC}}^X \simeq 0.2644$, where we use the fact that the number of independent Z -stabilizers is $N_Z = E - 2V - 6$, see [KYP15]. We emphasize that the threshold estimates $p_{3\text{DCC}}^Z$ and $p_{3\text{DCC}}^X$ are not rigorous and should serve only as an intuition about the actual values $p_{3\text{DCC}}^{(1)} \simeq 1.9\%$ and $p_{3\text{DCC}}^{(2)} \simeq 27.5\%$. We remark that by applying the same reasoning to the 3D toric code on the cubic lattice we find the thresholds for the phase- and bit-flip noise models to be $p_{3\text{DTC}}^Z \simeq 0.0615$ and $p_{3\text{DTC}}^X \simeq 0.1740$, which should be compared with the actual values $p_{3\text{DTC}}^{(1)} \simeq 3.3\%$ and $p_{3\text{DTC}}^{(2)} \simeq 23.5\%$.

5.5 Proof of implication

Here we show that successful decoding implies diverging average energy cost of introducing any non-trivial domain wall. We used this fact in the derivation of statistical-mechanical models to relate the threshold p_c of optimal error correction to the critical point p_N on the Nishimori line. Note that this implication allows us to only infer that $p_c \leq p_N$. However, we expect that successful decoding be possible throughout the ordered phase and thus these two values should coincide.

Lemma 13 *Consider a CSS code described by the chain complex in Eq. (5.3). Let $H_1 = \ker \partial_1 / \text{im } \partial_2$ be the first homology group of finite cardinality, $|H_1| < \infty$. If the probability of successful optimal X -error correction goes to 1 in the limit of infinite system size*

$$\text{pr}(\text{succ}) = \sum_{\epsilon \in \mathcal{C}_1} \text{pr}(\epsilon) \text{pr}(\text{succ}|\epsilon) \rightarrow 1, \quad (5.27)$$

then the average free energy cost of introducing any non-trivial domain wall $\lambda \in \ker \partial_1 \setminus \text{im } \partial_2$ diverges

$$\langle \Delta_\lambda \rangle = \sum_{\epsilon \in \mathcal{C}_1} \text{pr}(\epsilon) \Delta_\lambda(\epsilon) \rightarrow \infty. \quad (5.28)$$

Proof: Let $\bar{\epsilon} = \{\epsilon + \partial_2 \omega \mid \omega \in C_2\}$ denote the equivalence class of errors for $\epsilon \in C_1$ and $\mathcal{E} = \{\bar{\epsilon}, \dots\}$ be the set of all equivalence classes. We define a representative of the most probable equivalence class of errors consistent with the syndrome $\sigma \in C_0$ to be

$$\varrho(\sigma) = \arg \max_{\substack{\epsilon \in C_1 \\ \partial_1 \epsilon = \sigma}} \text{pr}(\bar{\epsilon}). \quad (5.29)$$

The conditional probability of successful decoding using the optimal (maximum likelihood) decoder is given by

$$\text{pr}(\text{succ} \mid \epsilon) = \begin{cases} 1 & \text{if } \epsilon \in \overline{\varrho(\partial_1 \epsilon)}, \\ 0 & \text{otherwise.} \end{cases} \quad (5.30)$$

Thus, we have

$$\text{pr}(\text{succ}) = \sum_{\epsilon \in C_1} \text{pr}(\epsilon) \text{pr}(\text{succ} \mid \epsilon) = \sum_{\sigma \in \text{im } \partial_1} \text{pr}(\overline{\varrho(\sigma)}). \quad (5.31)$$

By rewriting the sum over all equivalence classes of errors $\bar{\epsilon} \in \mathcal{E}$ as the sum over all possible syndromes $\sigma \in \text{im } \partial_1$ and different representatives $\lambda' \in H_1$ of the homology group we arrive at

$$1 = \sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) = \sum_{\sigma \in \text{im } \partial_1} \sum_{\lambda' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda'}) \quad (5.32)$$

$$= \text{pr}(\text{succ}) + \sum_{\sigma \in \text{im } \partial_1} \sum_{0 \neq \lambda' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda'}). \quad (5.33)$$

We want to show two inequalities

$$\text{pr}(\text{succ}) \geq \sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \frac{\text{pr}(\bar{\epsilon})}{\sum_{\lambda' \in H_1} \text{pr}(\bar{\epsilon} + \lambda')} \geq 2\text{pr}(\text{succ}) - 1. \quad (5.34)$$

In order to show the first inequality (5.34) note that

$$\text{pr}(\text{succ}) = \sum_{\sigma \in \text{im } \partial_1} \sum_{\lambda'' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda''}) \frac{\text{pr}(\overline{\varrho(\sigma)})}{\sum_{\lambda' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda'})} \quad (5.35)$$

$$\geq \sum_{\sigma \in \text{im } \partial_1} \sum_{\lambda'' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda''}) \frac{\text{pr}(\overline{\varrho(\sigma) + \lambda''})}{\sum_{\lambda' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda'})} \quad (5.36)$$

$$= \sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \frac{\text{pr}(\bar{\epsilon})}{\sum_{\lambda' \in H_1} \text{pr}(\bar{\epsilon} + \lambda')}, \quad (5.37)$$

where we use $\text{pr}(\overline{\varrho(\sigma)}) \geq \text{pr}(\overline{\varrho(\sigma) + \lambda''})$ for all $\sigma \in \text{im } \partial_1$ and $\lambda'' \in H_1$. The second inequality (5.34) follows from

$$\text{pr}(\text{succ}) = \sum_{\sigma \in \text{im } \partial_1} \text{pr}(\overline{\varrho(\sigma)}) \quad (5.38)$$

$$= \sum_{\sigma \in \text{im } \partial_1} \text{pr}(\overline{\varrho(\sigma)}) \frac{\text{pr}(\overline{\varrho(\sigma)})}{\sum_{\lambda' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda'})} \quad (5.39)$$

$$+ \sum_{\sigma \in \text{im } \partial_1} \text{pr}(\overline{\varrho(\sigma)}) \frac{\sum_{0 \neq \lambda' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda'})}{\sum_{\lambda' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda'})} \quad (5.40)$$

$$\leq \sum_{\sigma \in \text{im } \partial_1} \sum_{\lambda'' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda''}) \frac{\text{pr}(\overline{\varrho(\sigma) + \lambda''})}{\sum_{\lambda' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda'})} \quad (5.41)$$

$$+ \sum_{\sigma \in \text{im } \partial_1} \sum_{0 \neq \lambda' \in H_1} \text{pr}(\overline{\varrho(\sigma) + \lambda'}) \quad (5.42)$$

$$= \sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \frac{\text{pr}(\bar{\epsilon})}{\sum_{\lambda' \in H_1} \text{pr}(\bar{\epsilon} + \lambda')} + (1 - \text{pr}(\text{succ})). \quad (5.43)$$

If $\text{pr}(\text{succ}) \rightarrow 1$, then from inequalities (5.34) we infer that

$$\sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \frac{\text{pr}(\bar{\epsilon})}{\sum_{\lambda' \in H_1} \text{pr}(\bar{\epsilon} + \lambda')} \rightarrow 1, \quad (5.44)$$

and thus for $\lambda \in \ker \partial_1 \setminus \text{im } \partial_2$ we have

$$\sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \frac{\text{pr}(\bar{\epsilon} + \lambda)}{\sum_{\lambda' \in H_1} \text{pr}(\bar{\epsilon} + \lambda')} \rightarrow 0. \quad (5.45)$$

In the last step we used the following inequalities

$$0 \leq \sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \frac{\text{pr}(\bar{\epsilon} + \lambda)}{\sum_{\lambda' \in H_1} \text{pr}(\bar{\epsilon} + \lambda')} \leq \sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \frac{\sum_{0 \neq \lambda' \in H_1} \text{pr}(\bar{\epsilon} + \lambda')}{\sum_{\lambda' \in H_1} \text{pr}(\bar{\epsilon} + \lambda')} \quad (5.46)$$

$$= 1 - \sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \frac{\text{pr}(\bar{\epsilon})}{\sum_{\lambda' \in H_1} \text{pr}(\bar{\epsilon} + \lambda')}. \quad (5.47)$$

We rewrite $\langle \Delta_\lambda \rangle$ in the following way

$$\langle \Delta_\lambda \rangle = \sum_{\epsilon \in \mathcal{C}_1} \text{pr}(\epsilon) \Delta_\lambda(\epsilon) = \sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \log \frac{\text{pr}(\bar{\epsilon})}{\text{pr}(\bar{\epsilon} + \lambda)} \quad (5.48)$$

$$= \sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \log \frac{\text{pr}(\bar{\epsilon})}{\sum_{\lambda' \in H_1} \text{pr}(\bar{\epsilon} + \lambda')} - \sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \log \frac{\text{pr}(\bar{\epsilon} + \lambda)}{\sum_{\lambda' \in H_1} \text{pr}(\bar{\epsilon} + \lambda')} \quad (5.49)$$

Using the inequality $\log x \geq 1 - \frac{1}{x}$ to lower-bound the first term and Jensen inequality for the second term we obtain

$$\langle \Delta_\lambda \rangle \geq (1 - |H_1|) - \log \sum_{\bar{\epsilon} \in \mathcal{E}} \text{pr}(\bar{\epsilon}) \frac{\text{pr}(\overline{\epsilon + \lambda})}{\sum_{\lambda' \in H_1} \text{pr}(\overline{\epsilon + \lambda'})} \rightarrow \infty. \quad (5.50)$$

□

5.6 Phase diagram

We describe how to map out the (p, T) -phase diagrams of the two RCIMs, $H_\epsilon^X(\{s_v\})$ and $H_\epsilon^Z(\{s_e\})$. The discontinuity in energy density across a first order phase transition allows for straightforward identification of the phase boundary in the regime of low disorder. However, more reliable order parameters are required to probe a (higher-order) phase transition close to the critical point on the Nishimori line. Moreover, an appropriate order parameter takes symmetries of the model into account. Note that flipping a subset of spins $\{s_i\}_{i \in I}$, i.e. $s_i \mapsto -s_i$ for $i \in I$, is a symmetry if it leaves the Hamiltonian describing the model invariant.

The 4-body RCIM in Eq. (5.13) has a global $\mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_2$ symmetry. An example of a symmetry operation is a simultaneous flip of vertex spins on all red and blue vertices, since it leaves every term of $H_\epsilon^X(\{s_v\})$ unchanged. Due to this symmetry, the total magnetization is not a good order parameter, however the sublattice magnetization of spins of a single color is. Instead of using the sublattice magnetization directly, more precise estimations are obtained by considering the finite-size scaling of the spin-spin correlation function [PC99]. Near phase transition, for fixed disorder strength p and temperatures T close to the critical temperature $T_c(p)$, the correlation length ξ_L is expected to scale as

$$\xi_L(p, T)/L \sim f(L^{1/\nu}(T - T_c(p))), \quad (5.51)$$

where L is the linear system size, f is a scaling function and ν is the correlation length critical exponent [Gol92]. We can estimate $T_c(p)$ by plotting $\xi_L(p, T)/L$ as a function of temperature T for different system sizes L and finding their crossing point, see Fig. 5.4(a)(b). If no crossing is observed, then we conclude that there is no phase transition.

The 6-body RCIM in Eq. (5.60) describes a lattice gauge theory with a local (gauge) $\mathbb{Z}_2 \times \mathbb{Z}_2$ symmetry. An example of a symmetry operation is a flip of edge spins on edges from a single yellow vertex to all neighboring red and blue vertices, see

Fig. 5.2(c). Due to Elitzur's theorem [Eli75], the gauge symmetry rules out existence of any local order parameter. We define a Wilson loop operator [Wil74; Kog79]

$$W(\gamma) = \prod_{e \in \gamma} s_e, \quad (5.52)$$

which is a product of edge spins along a loop $\gamma \subset \Delta_1(\mathcal{L})$. For $W(\gamma)$ to be gauge-invariant the loop γ can only be composed of edges connecting vertices of two (out of four possible) colors. The phase transition can be identified by analyzing scaling of the thermal expectation value of $W(\gamma)$ averaged over different disorder configurations

$$\langle W(\gamma) \rangle = \sum_{\epsilon \in \Delta_3(\mathcal{L})} \text{pr}(\epsilon) \sum_{\{s_e\}} W(\gamma) \frac{e^{-\beta H_\epsilon^Z(\{s_e\})}}{Z_\epsilon(\beta)}. \quad (5.53)$$

Namely, in the limit of large square loops [CJR79; WHP03; Ohn+04], $-\log \langle W(\gamma) \rangle$ scales linearly with the loop's perimeter $P(\gamma)$ in the ordered (Higgs) phase, whereas in the disordered (confinement) phase scales linearly with the minimum area $A(\gamma)$ enclosed by γ , see Fig. 5.4(d)-(f).

We find the (p, T) -phase diagrams of the 4- and 6-body RCIM by the performing parallel tempering Monte Carlo simulations [HN96]; see Fig. 5.1. We test equilibration of the system by logarithmic binning of the data; we define the system to equilibrate when the last three bins agree within statistical uncertainty [KBM09; And12]. Since we simulate systems of finite size, a careful analysis of finite-size effects is necessary.

5.7 Finding phase transitions

In order to map the disorder-temperature phase diagrams of the 4- and 6-body RCIM in Fig. 5.1 we need to reliably identify phase transitions. Here we describe in detail how we achieve that by analyzing specific heat, the spin-spin correlation function and the Wilson loop operator.

Specific heat

For a second-order phase transition, the specific heat $c(T)$ as a function of temperature T is expected to have a discontinuity near a phase transition at temperature T_c in the limit of infinite system size $L \rightarrow \infty$. However, for a system of finite linear size L , the peak of the specific heat $c_L(T)$ appears at temperature $T_c(L) = \arg \max_T c_L(T)$ shifted from that in the infinite system by an amount

$$\left| \frac{T_L - T_c}{T_c} \right| \propto L^{-1/\nu}, \quad (5.54)$$

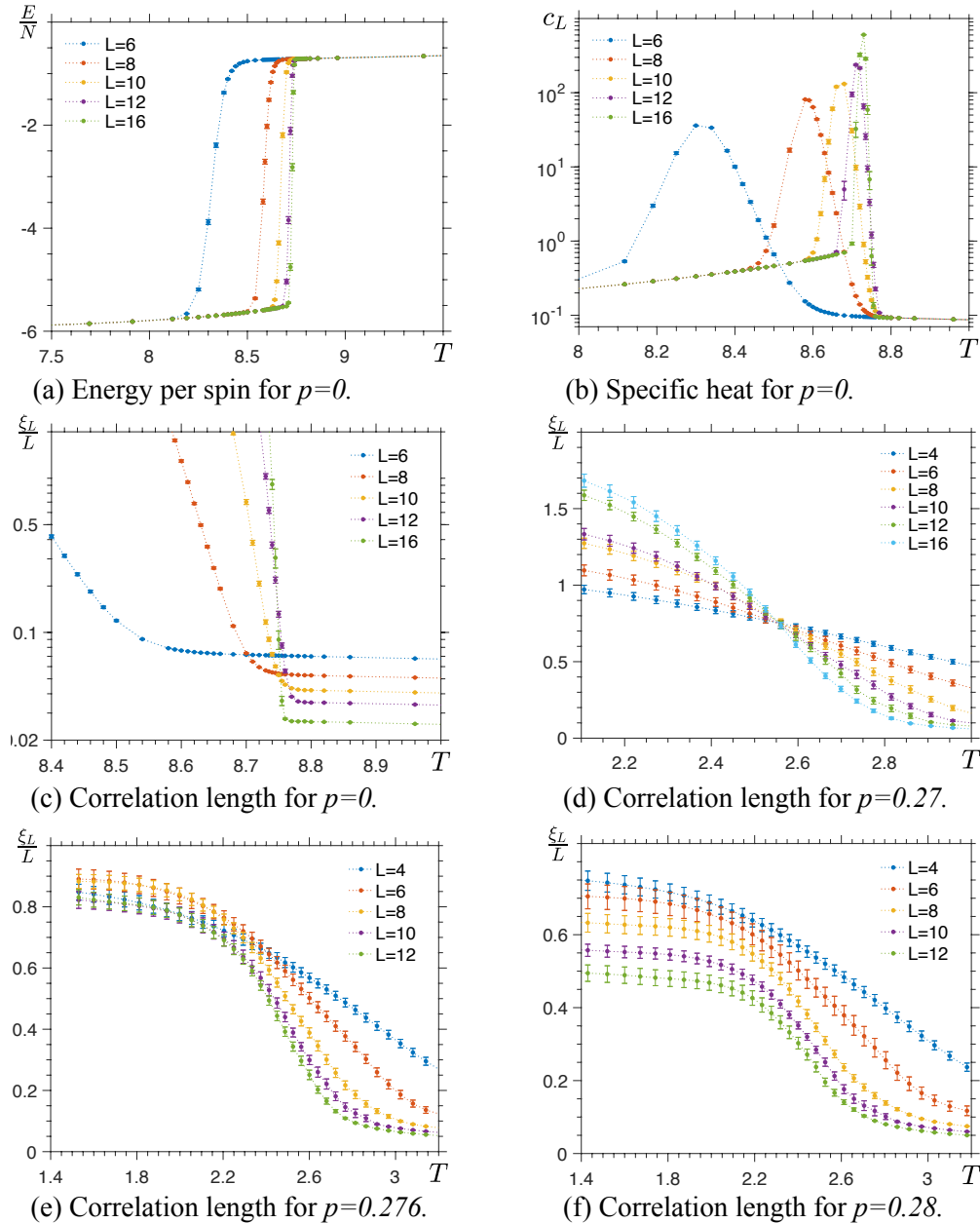


Figure 5.3: Results for the 3D 4-body RCIM. (a) The discontinuity in energy per spin E/N suggests a first-order phase transition. (b) By finding peaks of specific heat c_L for different system sizes L and exploiting finite-size scaling we estimate the critical temperature of a phase transition to be $T_c = 8.77(2)$. (c) The normalized correlation length ξ_L/L does not seem to be described well by the scaling ansatz in Eq. (5.51) possibly due to a transition being first-order. (d) We identify $T_c = 2.56(4)$ as the intersection of normalized spin-spin correlation functions ξ_L/L for different L . (e) Close to the critical point on the Nishimori line $p_N = p_{3\text{DCC}}^{(2)}$ detecting a phase transition and estimating its critical temperature becomes difficult. (f) For $p = 0.28$ there is no indication of a phase transition.

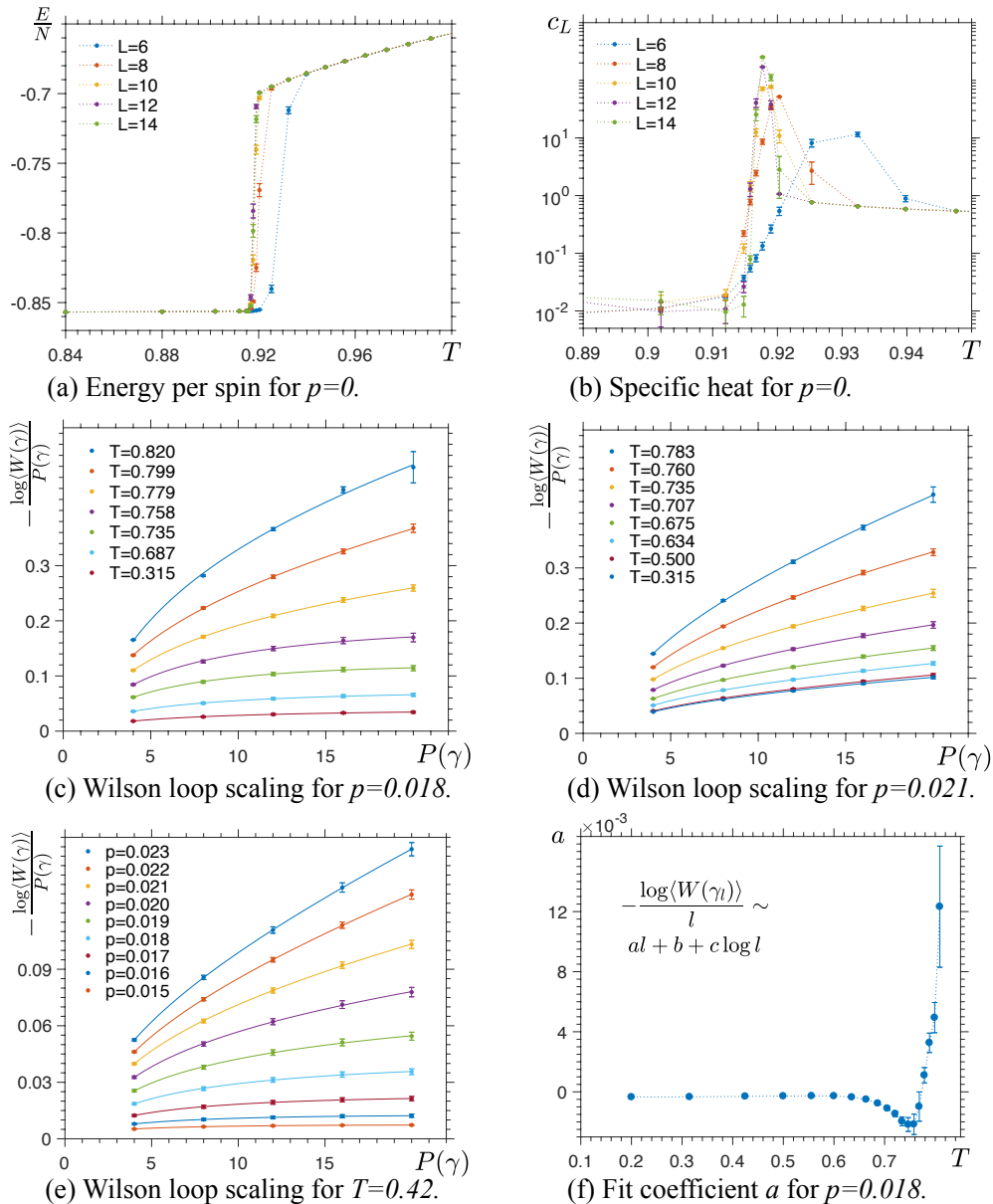


Figure 5.4: Results for the 3D 6-body RCIM. (a) The discontinuity in energy per spin E/N suggests a first-order phase transition. (b) By finding peaks of specific heat c_L for different system sizes L and exploiting finite-size scaling we estimate the critical temperature of a phase transition to be $T_c = 0.918(3)$. In (c)-(e) we check if the Wilson loop operator $W(\gamma)$ satisfies the perimeter law by plotting $-\log\langle W(\gamma) \rangle / P(\gamma)$ as a function of perimeter $P(\gamma)$ of the square loop γ . Solid lines show a numerical ansatz $aP(\gamma) + b + c \log P(\gamma)$ with fitting parameters a, b, c . The parameter a allows to identify a phase transition. A change of scaling in (c) and (e) signals a phase transition at $T = 0.75(3)$ and $p = 0.019(1)$, respectively, whereas in (d) there is no indication that the system undergoes a phase transition. (f) By finding a fit $-\log\langle W(\gamma_l) \rangle / l \sim al + b + c \log l$ we can identify the critical temperature $T_c = 0.75(3)$ of a transition as a location where $a = 0$.

where ν is the correlation length critical exponent [Gol92]. A similar scaling behavior has been established for first-order phase transitions [CLB86; Bin87; LK91; BK92]. Thus, we find the critical temperature T_c by fitting a function

$$T_c(L) \sim aL^{-b} + T_c \quad (5.55)$$

to the position of the specific heat peaks for different system sizes and evaluating $T_c(L = \infty)$.

Correlation function

One might not be able to identify a phase transition of higher order by looking at the specific heat. Rather, one needs to analyze the behavior of e.g. the order parameter correlation length ξ . In particular, for the system of finite size L and with fixed disorder strength p we define the two-point finite-size correlation length ξ_L as a function of temperature T

$$\xi_L(T) = \frac{1}{2 \sin(k_0/2)} \sqrt{\frac{\langle \chi(\vec{0}) \rangle}{\langle \chi(\vec{k}_0) \rangle} - 1}, \quad (5.56)$$

where $\langle \chi(\vec{k}) \rangle = \sum_{\epsilon \in \Delta_3(\mathcal{L})} \text{pr}(\epsilon) \chi(\vec{k})$, \vec{k} is the wavevector and $\vec{k}_0 = (2\pi/L, 0, 0)$. In the above, we use the thermal expectation value of the wavevector-dependent sublattice magnetic susceptibility

$$\chi(\vec{k}) = \sum_{\{s_v\}} \frac{1}{N} \left(\sum_{u \in U} s_u e^{i\vec{k} \cdot \vec{r}_u} \right)^2 \frac{e^{-\beta H_\epsilon^X(\{s_v\})}}{Z_\epsilon(\beta)}. \quad (5.57)$$

where \vec{r}_u denotes the position of the vertex spin s_u in a sublattice $U \subset \Delta_0(\mathcal{L})$ of single-color vertices. Near a phase transition at temperature T_c , the normalized correlation length is expected to scale as

$$\frac{\xi_L(T)}{L} \sim f(L^{1/\nu}(T - T_c)), \quad (5.58)$$

where f is a dimensionless scaling function and ν is the correlation length critical exponent. We can estimate T_c by plotting $\xi_L(T)/L$ as a function of temperature T for different system sizes L and finding their crossing point. If there is no crossing, then we conclude that there is no phase transition.

Wilson loop operator

When the system under consideration has a local (gauge) symmetry, one cannot use a local order parameter to detect a phase transition. Rather, one needs to consider

gauge-invariant quantities, such as the Wilson loop operator $W(\gamma)$ in Eq. (5.52). Suppose γ is a square loop. We denote by $P(\gamma)$ and $A(\gamma)$ the perimeter of γ and the minimal area enclosed by γ , respectively. The scaling of the averaged Wilson loop operator $\langle W(\gamma) \rangle$ in the limit of large loops changes between the ordered (Higgs) and disordered (confinement) phases. Namely,

- in the disordered phase: $\langle W(\gamma) \rangle \sim \exp(-\text{const} \cdot A(\gamma))$,
- in the ordered phase: $\langle W(\gamma) \rangle \sim \exp(-\text{const} \cdot P(\gamma))$.


We consider a system of finite size L and denote by γ_l a square loop of linear size $l \leq L/2$. Since $A(\gamma_l) \propto l^2$ and $P(\gamma_l) \propto l$, then $\log \langle W(\gamma_l) \rangle$ should scale either quadratically or linearly in l , depending on the phase of the system. Due to finite-size effects, there are some corrections to the area and perimeter scaling. In particular, we numerically find that

$$-\frac{\log \langle W(\gamma_l) \rangle}{l} \sim al + b + c \log l, \quad (5.59)$$

where a, b, c are some constants. We identify the disordered phase as the region where the fitting parameter a is positive, $a > 0$.

Classical Ising gauge theory

As an example of using specific heat and the scaling of the Wilson loop operator to identify a phase transition we study a known model, the three-dimensional random plaquette Ising model (RPIM), see Fig. 5.6. The RPIM is a generalization of the \mathbb{Z}_2 Ising gauge theory, which is relevant for studying the optimal error correction threshold for 1D string-like operators in the 3D toric code [Den+02]. The RPIM is a statistical-mechanical model with classical spins $s_e = \pm 1$ placed on edges $e \in \Delta_1(C)$ of the cubic lattice C and disorder $\epsilon \subset \Delta_2(C)$. The Hamiltonian describing the RPIM

$$H_\epsilon^{\text{RPIM}}(\{s_e\}) = - \sum_{f \in \Delta_2(C)} (-1)^{[\epsilon]_f} \text{img} \quad (5.60)$$


contains 4-body terms, which are products of four edge spins around every square face $f \in \Delta_2(C)$ of the lattice C . We set $[\epsilon]_f = 1$ if $f \in \epsilon$, otherwise $[\epsilon]_f = 0$. We observe that $H_\epsilon^{\text{RPIM}}(\{s_e\})$ has a local \mathbb{Z}_2 symmetry, generated by flips of spins on all edges incident on any vertex $v \in \Delta_0(C)$. The Wilson loop operator $W(\gamma_l)$ is a gauge-invariant quantity, where γ_l is a square loop of linear size l . The disorder-temperature phase diagram of the 3D RPIM is shown in Fig. 5.5.

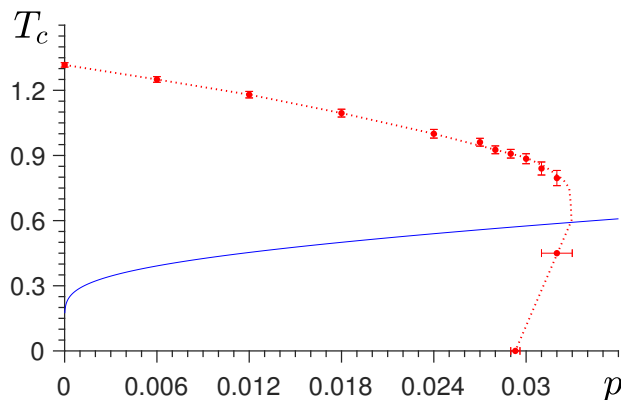


Figure 5.5: The disorder-temperature (p, T) -phase diagram of the 3D random plaquette Ising model on the cubic lattice. The intersection of the Nishimori line (blue) with the anticipated phase boundary (red dotted line) gives the 3D toric code threshold $p_{3\text{DTC}}^{(1)} \simeq 3.3\%$ for optimal error correction associated with 1D string-like logical operators (and point-like excitations). Note that the location of a phase transition for $T = 0$ was found in [WHP03].

Numerical simulation details

The numerical complexity of simulating the statistical-mechanical models, such as the 4- and 6-body RCIM and the RPIM, increases with the disorder strength p , which is reminiscent of a spin glass behavior. To speed up simulations we use the parallel tempering technique. The parallel tempering technique requires simultaneous simulation of multiple copies $k = 1, \dots, N_T$ of the system with the same disorder ϵ but different spin configurations $\{s_i\}_k$ and temperatures $T_1 < \dots < T_{N_T}$. After performing single-spin Metropolis updates for all spins in every copy of the system, swaps of spin configurations $\{s_i\}_k \leftrightarrow \{s_i\}_{k+1}$ of copies at neighboring temperatures T_k and T_{k+1} are allowed with probability

$$\text{pr}(k \leftrightarrow k + 1) = \exp\left((E_k - E_{k+1})\left(\frac{1}{T_k} - \frac{1}{T_{k+1}}\right)\right), \quad (5.61)$$

where E_k and E_{k+1} denote energies of spin configurations $\{s_i\}_k$ and $\{s_i\}_{k+1}$. We choose temperatures $T_1 < \dots < T_{N_T}$ is such a way that the exchange rate $\{s_i\}_k \leftrightarrow \{s_i\}_{k+1}$ estimated by counting the number of (successful) swaps of systems at neighboring temperatures is constant up to statistical fluctuations; for more in-depth discussions see e.g. [Kat+06]. Equilibration of the system is tested by a logarithmic binning of data. Namely, the total number of time steps of the simulation is $1 + t_1 + t_2 + \dots + t_\tau = 2^\tau$, where $t_i = 2^{i-1}$ and one time step consists of an update of every spin in all N_T copies of the system followed by (attempted) swaps $\{s_i\}_k \leftrightarrow \{s_i\}_{k+1}$ of spin configurations. We say that the system equilibrates if the

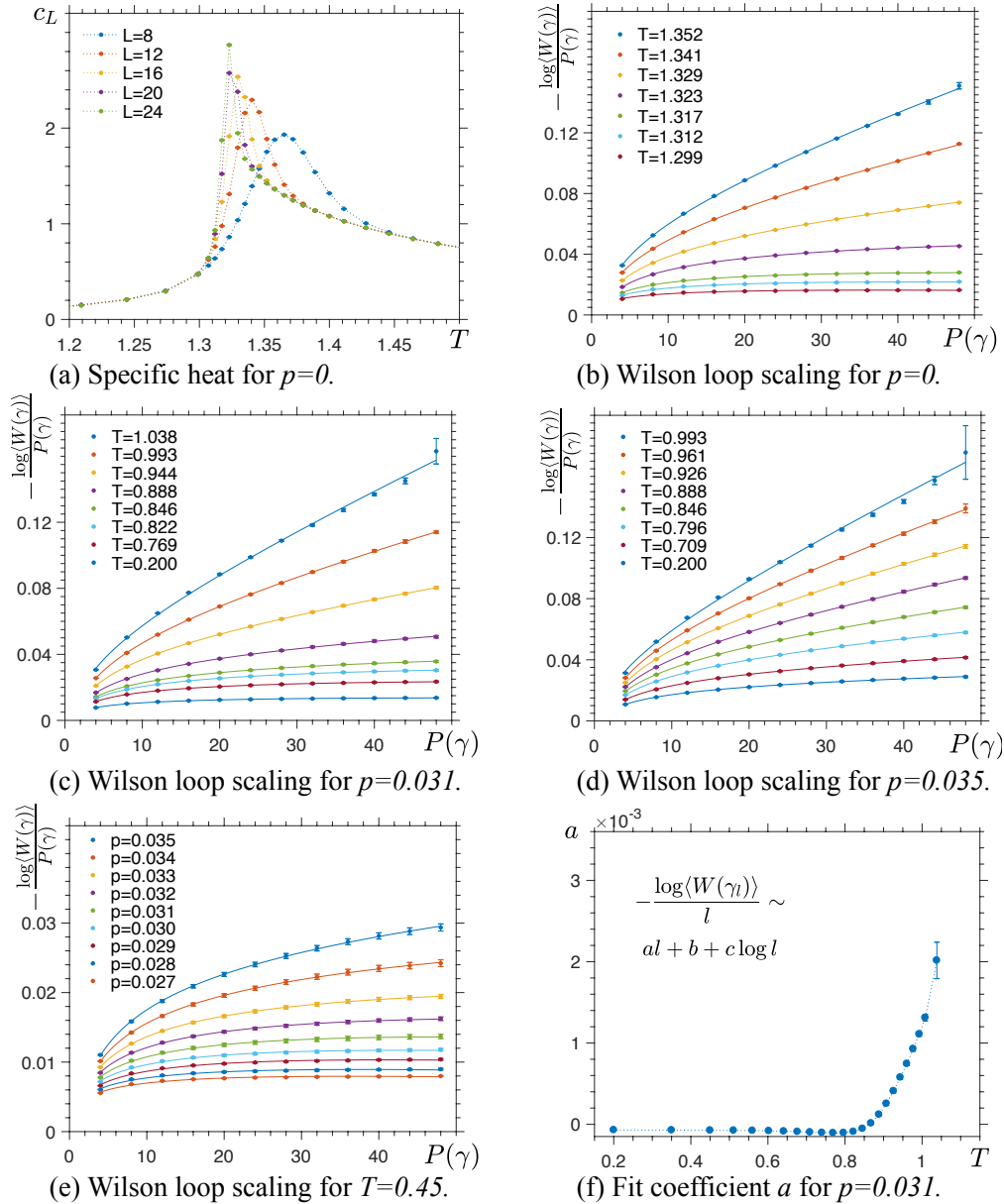


Figure 5.6: Results for the 3D RPIM. (a) We can estimate the critical temperature $T_c = 1.316(4)$ of a phase transition by finding the peak positions of specific heat c_L for different system sizes L and exploiting finite-size scaling. In (b)-(d) we check whether the Wilson loop operator $W(\gamma)$ satisfies the perimeter law by plotting $-\log\langle W(\gamma) \rangle/P(\gamma)$ as a function of perimeter $P(\gamma)$ of the square loop γ for different temperatures T and $L = 24$. (e) For fixed temperature $T = 0.45$ we analyze scaling of $-\log\langle W(\gamma) \rangle/P(\gamma)$ for different disorder values p . (f) We find a fit $-\log\langle W(\gamma_l) \rangle/l \sim al + b + c \log l$ to the data in (c) and plot the fit coefficient a as a function of temperature T . We identify the critical temperature $T_c = 0.84(3)$ of a phase transition in (c) as a location where $a = 0$. In (b),(c) and (e) we see a change of scaling as the system undergoes a phase transition at $T_c = 1.317(6)$, $T_c = 0.84(3)$ and $p_c = 0.032(1)$, respectively. In (d) there is no indication of a transition.

quantities of interest, such as the correlation length or the Wilson loop operator, evaluated based on the data from the last three periods of time $t_{\tau-2}$, $t_{\tau-1}$ and t_{τ} agree up to statistical uncertainty, see e.g. [KBM09; And12]. Numerical simulation details for the 4-body RCIM, the 6-body RCIM and the RPIM are presented in Table 5.1.

To estimate statistical error bars of quantities analyzed in the simulation we use the bootstrap technique. The main idea behind the bootstrap technique is to repeat sampling from the existing data set $D = \{d_1, \dots, d_N\}$ and evaluating a quantity of interest $q = q(D)$. In particular, for $i = 1, \dots, n$ we perform the following steps

1. from the data set D randomly choose N data points $d_{i(j)}$, where $i(j) \in \{1, \dots, N\}$,
2. evaluate the quantity $q_i = q(D_i)$ from the data set $D_i = \{d_{i(1)}, \dots, d_{i(N)}\}$.

Note that in step 1 we allow to choose the same data point multiple times. The relevant quantity q is estimated to be

$$q = \bar{q} \pm \sqrt{\sum_{i=1}^n \frac{(\bar{q} - q_i)^2}{n-1}}, \quad (5.62)$$

where $\bar{q} = \frac{1}{n} \sum_{i=1}^n q_i$.

5.8 Discussion

The 3D color code is a zero-rate code, thus from the quantum Gilbert-Varshamov bound [Gil52; Var57; CS96] we obtain the inequality $H(p_{3\text{DCC}}^{(1)}) + H(p_{3\text{DCC}}^{(2)}) \leq 1$ relating the phase- and bit-flip thresholds, where $H(p) = -p \log_2 p - (1-p) \log_2 (1-p)$ is the Shannon entropy. Our numerical estimates $p_{3\text{DCC}}^{(1)} \simeq 1.9\%$ and $p_{3\text{DCC}}^{(2)} \simeq 27.5\%$ satisfy that constraint.

The X -stabilizers detecting Z -errors are the same for the 3D stabilizer and subsystem color codes. Since the subsystem code has X - and Z -generators of identical support, its phase- and bit-flip thresholds for perfect measurements and optimal decoding are both equal to $p_{3\text{DCC}}^{(1)}$. For the 3D color code on the bcc lattice, the threshold of the (efficient) projection decoder $p_{\text{proj}}^{(1)} \simeq 0.75\%$ is less than a half of $p_{3\text{DCC}}^{(1)}$, justifying a search for better decoders.

Quantum codes can motivate studies of unusual systems, such as spin models on fractal lattices [KY14; YK14]. We hope our work provides a first step toward better

p	L_{\max}	N_{ϵ}	τ	N_T	T_{\min}	T_{\max}
0.000	16	500	20	55	2.40	12.80
0.050	16	500	20	42	2.30	11.40
0.100	16	500	20	41	2.20	10.15
0.150	16	500	20	42	2.10	8.42
0.200	16	500	20	41	2.00	6.80
0.250	16	500	20	42	1.90	4.97
0.265	16	500	20	34	1.80	3.53
0.270	16	500	20	34	1.60	3.32
0.272	12	500	20	34	1.60	3.30
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
0.280	12	500	20	34	1.33	3.18
0.000	14	250	20	47	0.20	1.28
0.003	12	250	20	44	0.20	1.25
0.006	12	250	20	42	0.20	1.22
0.009	12	250	20	39	0.20	1.17
0.012	12	250	20	38	0.20	1.14
0.015	10	250	22	37	0.20	1.08
0.016	10	250	22	37	0.20	1.08
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
0.023	10	250	22	37	0.20	1.10
0.000	24	500	19	51	0.40	2.08
0.006	24	500	19	43	0.40	1.95
0.012	24	500	19	41	0.40	1.77
0.018	24	500	19	43	0.35	1.64
0.024	24	500	19	42	0.30	1.49
0.027	24	250	19	43	0.20	1.28
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
0.035	24	250	19	43	0.20	1.28

Table 5.1: Numerical simulation parameters for: the 4-body RCIM (top), the 6-body RCIM (middle), and the IGT (bottom). L_{\max} and N_{ϵ} denote the linear size of the biggest simulated system and the number of randomly chosen disorder samples. N_T denotes the number of temperatures in the range $[T_{\min}, T_{\max}]$ chosen in a way that the exchange rate of spin configurations is approximately constant. 2^{τ} is the number of equilibration steps, where one equilibration step consists of an update of every spin in all N_T copies of the system followed by swaps $\{s_i\}_k \leftrightarrow \{s_i\}_{k+1}$ of spin configurations.

understanding of the 3D random coupling Ising models. We conjecture the existence of a spin-glass phase [BY86] in the 6-body RCIM. A future extension of this work might incorporate measurement errors which would require the study of 4D RCIMs and thus use more computational resources. If successful, this research program could provide a deeper understanding of single-shot error correction [Bom15b; BNB15] from the standpoint of statistical mechanics.

Chapter 6

UNFOLDING THE COLOR CODE

Quantum error-correcting codes [Sho96; Pre98] are vital for fault-tolerant realization of quantum information processing tasks. Of particular importance are topological quantum codes [Kit03; Den+02] where quantum information is stored in non-local degrees of freedom while the codes are characterized by geometrically local generators. An essential feature of such codes is to admit a fault-tolerant implementation of a universal gate set as this would guarantee that the physical errors propagate in a benign and controlled manner. Thus, the search for novel quantum error-correcting codes and the classification of fault-tolerantly implementable logical gates in these codes have been central problems in quantum information science [EK09; BK13; PR13; PY15; Bev+16].

The quest of analyzing topological quantum codes is also closely related to the central problem in quantum many-body physics, namely the classification of quantum phases [Sac99; CGW10]. A fruitful approach is to view topological quantum codes as exactly solvable toy models which correspond to representatives of gapped quantum phases. This approach has led to a complete classification of translation symmetric two-dimensional stabilizer Hamiltonians [Yos11; Bom14b], as well as to the discovery of novel three-dimensional topological phases which do not fit into previously known theoretical framework [Haa11; Yos13].

Topological color codes [BM07b] are important examples of topological stabilizer codes that admit transversal implementation of a variety of logical gates, which may not be fault-tolerantly implementable in other topological stabilizer codes. In two spatial dimensions, the color code admits transversal implementation of all logical Clifford gates. In three and higher dimensions, the color code admits transversal implementation of logical non-Clifford gates [BM09]. A naturally arising question is to identify the physical properties allowing the extension of the set of transversally implementable logical gates with respect to other topological codes.

Given two codes with different sets of fault-tolerantly implementable logical gates, one may naturally expect that they correspond to different topological phases of matter. However, physical properties of color codes and toric codes are known to be very similar. For instance, both of the codes have logical Pauli operators with similar

geometric shapes, which leads to essentially identical braiding properties of anyonic excitations from the viewpoint of long-range physics. Furthermore, it has been proven that translation symmetric stabilizer codes, supported on a two-dimensional torus, are equivalent to multiple decoupled copies of the two-dimensional toric code up to local unitary transformations and adding or removing ancilla qubits [Yos11]. This result implies that the two-dimensional color code supported on a torus is equivalent to two decoupled copies of the toric code, and thus they belong to the same quantum phase [BDP12].

However, the aforementioned results do not consider the effect of boundaries on the classification of quantum phases [BK98; KK12; BSW11]. In fact, the color code admits transversal implementation of computationally useful logical gates only if it is supported on a system with appropriately designed boundaries. Perhaps, the presence of boundaries may render additional computational power to topological quantum codes and may result in richer structure of topological phases of matter. Complete understanding of the relation between the color code and the toric code will be the necessary first step to clarify the connection between boundaries and achievable fault-tolerant logical gates, and its implications to the classification of quantum phases.

Summary of main results

In this chapter, we establish a connection between the color code and the toric code in the presence or absence of boundaries, and study fault-tolerantly implementable logical gates in these two codes. Our first result, presented in Section 6.1, focuses on the equivalence between the color code and the toric code on d -dimensional lattices without boundaries, $d \geq 2$.

Result 1 (Closed manifold) The topological color code on a d -dimensional closed manifold (without boundaries) is equivalent to multiple decoupled copies of the d -dimensional toric code up to local unitary transformations and adding or removing ancilla qubits.

This extends the known results from [Yos11; Bom14b; BDP12] to the family of color codes in arbitrary dimensions. While previous results are limited to either translation symmetric systems or do not provide an explicit method of transformations, we provide a specific construction of how to map the color code into multiple decoupled toric code components. The recipe emphasizes the importance of colorability in the

construction of the color code. Our result implies that the topological color code belongs to the same quantum phase as two copies of the toric code, according to the definition widely accepted in the condensed matter physics community [CGW10].

In Section 6.3, we analyze the case of topological codes with boundaries and present the following result.

Result 2 (Boundaries) The d -dimensional topological color code with boundaries is equivalent to d copies of the d -dimensional toric code which are attached along a $(d - 1)$ -dimensional boundary.

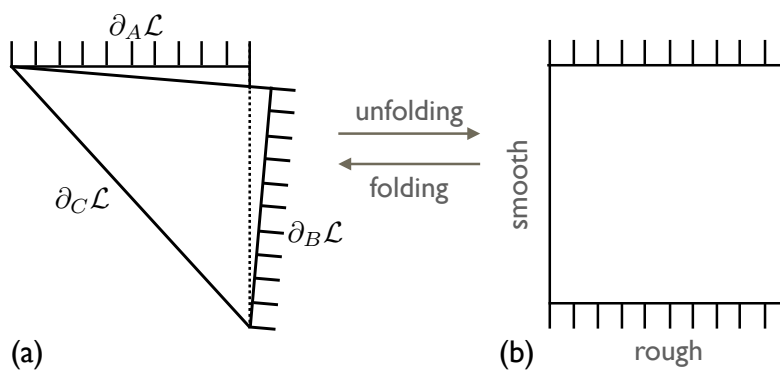


Figure 6.1: The topological color code (a) with three boundaries $\partial_A \mathcal{L}$, $\partial_B \mathcal{L}$ and $\partial_C \mathcal{L}$ viewed as the folded toric code (b) with two smooth and two rough boundaries. The boundary $\partial_A \mathcal{L}$ of color A is equivalent to a pair of boundaries — smooth in the front and rough in the rear layer; similarly $\partial_B \mathcal{L}$. The boundary $\partial_C \mathcal{L}$ is the fold.

In two dimensions, we find that the (triangular) color code with three boundaries is equivalent to the toric code with boundaries (i.e. the surface code) which is folded (see Fig. 6.1). We find that the $d > 2$ version of the color code with point-like excitations and $d + 1$ distinctly colored boundaries is equivalent to d copies of the toric code which are attached along a $(d - 1)$ -dimensional boundary. On this $(d - 1)$ -dimensional boundary, a composite point-like electric charge composed of all d electric charges from the different copies of the toric code may condense. Other boundaries are decoupled and allow condensation of a single electric charge associated with a specific copy.

In Section 6.4, we study logical non-Clifford gates fault-tolerantly implementable in the d -dimensional toric code. Our third result concerns the implementability of the d -qubit control- Z gate, i.e. a gate which applies a -1 phase only if all d qubits are in a $|1\rangle$ state.

Result 3 (Logical gate) A stack of d copies of the d -dimensional toric code with point-like excitations admits fault-tolerant implementation of the logical d -qubit control- Z gate by local unitary transformations.

In particular, we find that transversal application of physical $R_d = \text{diag}(1, e^{2\pi i/2^d})$ phase gates in the d -dimensional topological color code is equivalent to the logical d -qubit control- Z gate acting on d copies of the toric code. Note that the d -qubit control- Z gate belongs to the d -th level of the Clifford hierarchy, but is outside of the $(d - 1)$ -th level. Thus, a stack of d copies of the d -dimensional toric code saturates the bound by Bravyi and König on fault-tolerant logical gates which are implementable by local unitary transformations [BK13]. For a definition of the Clifford hierarchy, see [GC99; BK13; PY15].

We believe that our findings will shed light on the techniques of code deformation [BM09] and lattice surgery [Hor+12; LR14], allowing for computation with fewer physical qubits, higher fault-tolerant error suppression and shorter time. The ability to transform and relate different codes may turn out to be crucial in analyzing the available methods of computation with topological codes. In particular, we might be able to improve the decoding scheme for the color code proposed in Ref. [Del14b], and generalize it to any dimension. Also, our findings may lead to a systematic method of composing known quantum codes to construct new codes with larger set of fault-tolerant logical gates. Finally, an interesting future problem is to apply the disentangling unitary to the gauge color codes [Bom15a; KB15].

Our discussion mostly concerns the d -dimensional topological color code and the toric code with point-like excitations as it is the most interesting case from the viewpoint of transversal non-Clifford gates. For the sake of simplicity, we present proof sketches relying on many figures. We also provide rigorous proofs which require the language of algebraic topology [Gla72; Hat02], however they might be technically challenging and could obscure the main ideas presented in this chapter.

6.1 Topological color code without boundaries

In this section, we show that the d -dimensional topological color code supported on a closed manifold is equivalent to multiple decoupled copies of the toric code.

Brief introduction to the color code and the toric code

We begin by briefly reviewing the construction of the topological color code and the toric code. The starting point to define either the toric code or the color code is

a two-dimensional lattice \mathcal{L} . We can think of \mathcal{L} as a homogeneous cell 2-complex, i.e. a collection of vertices V , edges E and faces F , glued together in a certain way. In general, \mathcal{L} can be defined on a manifold with boundaries, but in this section we restrict our attention to closed manifolds.

The toric code in two dimensions is defined on a lattice \mathcal{L} by placing one qubit on every edge, and associating X - and Z -type stabilizer generators with vertices and faces of \mathcal{L} , namely

$$\forall v \in V : X(v) = \bigotimes_{e \supset v} X(e), \quad \forall f \in F : Z(f) = \bigotimes_{e \subset f} Z(e). \quad (6.1)$$

Here, $X(e)$ and $Z(e)$ denote Pauli X and Z operators on the qubit placed on the edge $e \in E$; see Fig. 6.2(a) for an example. We denote such a code, as well as its stabilizer group by $TC(\mathcal{L})$. One can verify that X - and Z -type stabilizer generators commute.

The color code is defined on a lattice \mathcal{L} , which satisfies two additional conditions:

- valence — each vertex belongs to exactly three edges,
- colorability — there is a coloring¹ of faces of \mathcal{L} with three colors, A , B and C , such that any two adjacent faces have different colors.

For instance, the honeycomb lattice satisfies the valence and colorability conditions; also see Fig. 6.2(b). In the case of the color code, we place one qubit at every vertex, and associate X - and Z -type stabilizer generators with every face of \mathcal{L} , namely

$$\forall f \in F : X(f) = \bigotimes_{v \subset f} X(v), \quad \forall f \in F : Z(f) = \bigotimes_{v \subset f} Z(v). \quad (6.2)$$

To verify that X - and Z -type stabilizers commute, one uses the valence and colorability conditions. We denote such a code, as well as its stabilizer group by $CC(\mathcal{L})$.

We can generalize the definition of the toric code and the color code to d dimensions by considering a d -dimensional lattice (i.e. a homogeneous cell d -complex) \mathcal{L} . There are $d - 1$ different ways of defining the toric code on \mathcal{L} — place qubits on m -cells, $m = 1, 2, \dots, d - 1$, and associate X - and Z -type stabilizer generators with

¹Note that due to the valence condition, this coloring is unique up to permutation of colors for any connected component of the lattice \mathcal{L} .

$(m - 1)$ - and $(m + 1)$ -cells, respectively. In the case of the color code, the additional conditions are that \mathcal{L} is $(d + 1)$ -valent and its d -cells are $(d + 1)$ -colorable. There are $d - 1$ ways of defining the color code on \mathcal{L} — place qubits on vertices, and associate X - and Z -type stabilizer generators with m - and $(d + 2 - m)$ -cells, where $m = 2, 3, \dots, d$. For a rigorous definition of the toric code and the color code in d dimensions see Sec. 6.2.

In the main body of the chapter, we restrict our attention to the color code and the toric code with point-like excitations, which significantly simplifies the discussion. In particular, the color code has X - and Z -type stabilizers associated with d -cells and 2-cells (faces), whereas the toric code has qubits placed on edges.

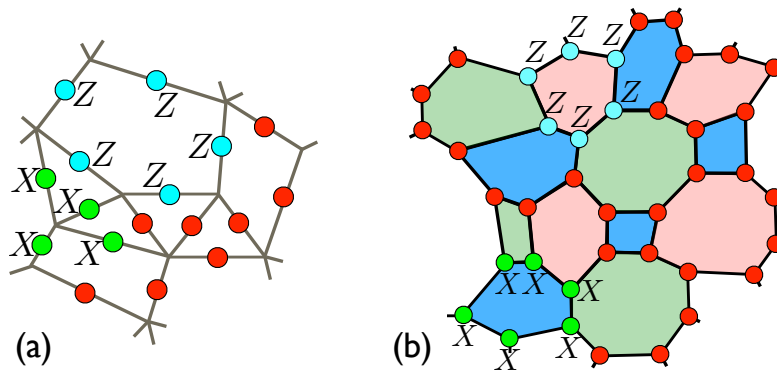


Figure 6.2: (Color online) The toric code and the color code in two dimensions. (a) The toric code has qubits (red dots) placed on edges, and X -vertex (green) and Z -face (blue) stabilizer generators. (b) The color code has qubits placed on vertices, and X -face and Z -face stabilizer generators. Note that the color code can only be defined on a 3-valent and 3-colorable lattice.

Equivalence in two dimensions

In this subsection, we prove that the two-dimensional color code supported on a closed manifold (without boundaries) is equivalent to two copies of the toric code.

Theorem 1 *Let $CC(\mathcal{L})$ be the two-dimensional topological color code defined on a lattice \mathcal{L} without boundaries, colored in A , B and C . There exists a local Clifford unitary U , and two lattices \mathcal{L}_A and \mathcal{L}_B obtained from \mathcal{L} by shrinking faces of color A and B , respectively, such that*

$$U[CC(\mathcal{L})]U^\dagger = TC(\mathcal{L}_A) \otimes TC(\mathcal{L}_B). \quad (6.3)$$

Moreover, one can choose U to be

$$U = \bigotimes_{f \in C} U_f, \quad (6.4)$$

where C represents the set of all faces in \mathcal{L} colored with C , and U_f is a Clifford unitary acting only on qubits of the face f .

Here, the tensor product $TC(\mathcal{L}_A) \otimes TC(\mathcal{L}_B)$ indicates that the stabilizer group can be factored into two independent stabilizer groups associated with two decoupled copies of the toric code on the lattices \mathcal{L}_A and \mathcal{L}_B . We shall refer to \mathcal{L}_A and \mathcal{L}_B supporting two decoupled copies of the toric code as *shrunk lattices* (see Refs. [BM07a; Bom13]).

As described in the theorem, the disentangling unitary transformation U has a tensor product structure, $U = \bigotimes_{f \in C} U_f$. Thus, U is a local unitary transformation, and two systems belong to the same quantum phase.

The proof of the Theorem 1 consists of three steps:

1. performing certain local unitary U_f at each and every face f of color C in \mathcal{L} ,
2. checking that the stabilizer generators $CC(\mathcal{L})$ are mapped by $U = \bigotimes_{f \in C} U_f$ into two sets of generators $TC(\mathcal{L}_A)$ and $TC(\mathcal{L}_B)$ supported on disjoint sets of qubits,
3. visualizing two codes $TC(\mathcal{L}_A)$ and $TC(\mathcal{L}_B)$ as codes defined on lattices \mathcal{L}_A and \mathcal{L}_B obtained from \mathcal{L} by local deformations.

Step 1: Let us pick a face f of \mathcal{L} colored in C . Since \mathcal{L} is 3-colorable and 3-valent, the face f has even number of vertices, $2n$. Moreover, we can color every edge in two distinct colors of faces it separates. Let us enumerate vertices of f in counter-clockwise order in such a way that the edge (1, 2) between vertices 1 and 2 has color AC . We would like to find a unitary transformation U_f of the Hilbert space \mathcal{H}_V of (color code) qubits placed on vertices into the Hilbert space \mathcal{H}_E of (toric code) qubits placed on edges² such that some operators on \mathcal{H}_V are mapped into certain

²Note that since the number of vertices of f is equal to the number of edges of f , then $\mathcal{H}_V \simeq \mathcal{H}_E \simeq (\mathbb{C}^2)^{\otimes 2n}$. Moreover, to perform such a transformation, one does not need any ancilla qubits.

operators on \mathcal{H}_E . In particular, we would require the following mappings to hold

$$Z_j Z_{j+1} \rightarrow Z_{(j,j+1)} \quad (j = 1, \dots, 2n - 1), \quad (6.5)$$

$$\left(\bigotimes_{j=1}^{2n} X_j \right) \cdot Z_{2n} Z_1 \rightarrow Z_{(2n,1)}, \quad (6.6)$$

$$X_j X_{j+1} \rightarrow X_{(j-1,j)} X_{(j+1,j+2)} \quad (j = 1, \dots, 2n - 2), \quad (6.7)$$

$$\left(\bigotimes_{j=1}^{2n} X_j \right) \cdot X_j X_{j+1} \rightarrow X_{(j-1,j)} X_{(j+1,j+2)} \quad (j = 2n - 1, 2n), \quad (6.8)$$

where X_j represents Pauli X operator on a qubit on the vertex j , while $X_{(j,j+1)}$ represents Pauli X operator on a qubit on the edge $(j, j+1)$ and $2n+1 \equiv 1$; similarly for Z_j and $Z_{(j,j+1)}$. The conditions imposed on U_f by Eqs. (6.5)–(6.8) for the face f with six vertices are illustrated in Fig. 6.3.

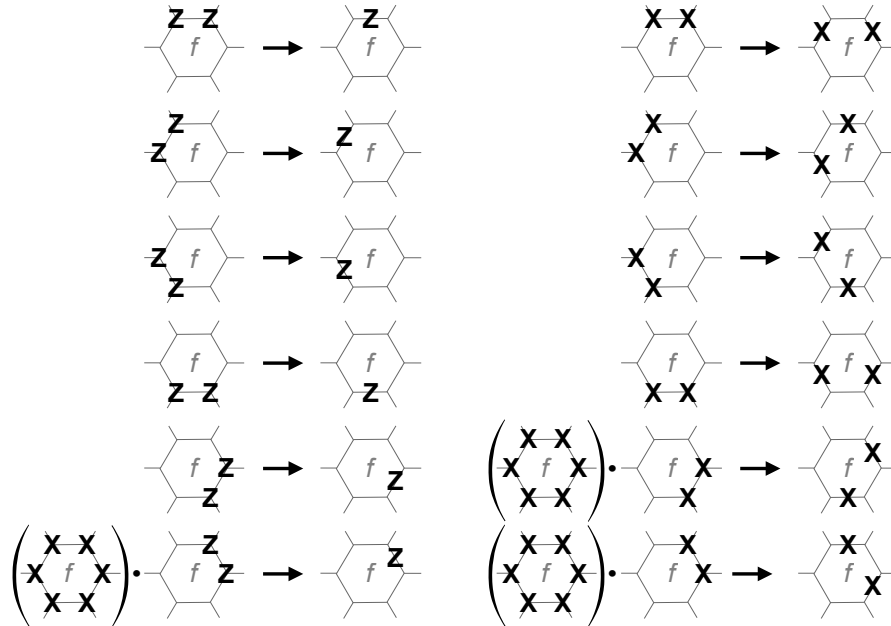


Figure 6.3: Transformation of the operators of the color code $CC(\mathcal{L})$ supported on qubits of the face f colored in C under the disentangling unitary transformation U_f .

We claim that there exists a Clifford unitary U_f which satisfies Eqs. (6.5)–(6.8). The proof of existence of such a unitary transformation is presented later. Note that under the unitary U_f the operators on the qubits on vertices of f (up to the stabilizer $\bigotimes_{j=1}^{2n} X_j$) transform into the operators on the qubits placed on edges of f in the

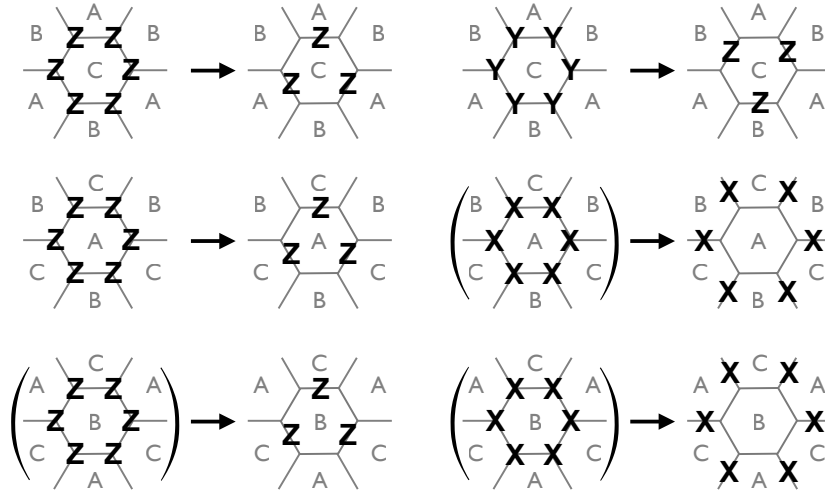


Figure 6.4: The effect of applying the disentangling unitary transformation U to the stabilizer group of the color code $CC(\mathcal{L})$. The parentheses indicate that the stabilizer of the color code might be multiplied by the X -face stabilizers on certain neighboring faces of color C , depending on the disentangling procedure, i.e. Step 1.

Step 3:

We would like to show that the stabilizer generators $TC(\mathcal{L}_A)$ and $TC(\mathcal{L}_B)$ define the toric code on two lattices, \mathcal{L}_A and \mathcal{L}_B , obtained from \mathcal{L} by local deformations. A recipe for the shrunk lattice \mathcal{L}_A is as follows:

- Vertices of \mathcal{L}_A are centers of A faces in \mathcal{L} .
- Edges of \mathcal{L}_A are BC edges in \mathcal{L} .
- Faces of \mathcal{L}_A are B and C faces in \mathcal{L} .

In short, one obtains \mathcal{L}_A by shrinking A faces to points while expanding B and C faces. [BM07a; Bom13]. Similarly, \mathcal{L}_B is obtained by shrinking B faces. Examples of shrunk lattices are depicted in Fig. 6.5 for the case of the hexagonal lattice \mathcal{L} . In this case, one obtains two copies of the toric code supported on triangular lattices.

The stabilizer generators $TC(\mathcal{L}_A)$ and $TC(\mathcal{L}_B)$ are respectively supported on \mathcal{L}_A and \mathcal{L}_B lattices. In particular,

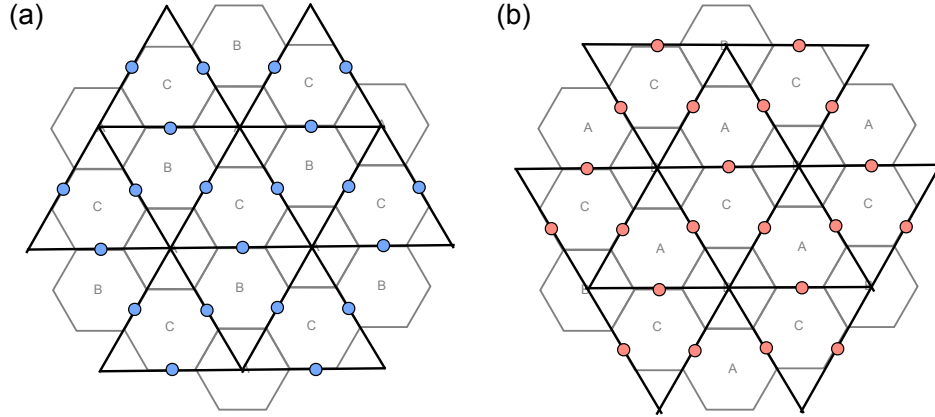


Figure 6.5: (Color online) Fragments of the shrunk lattices: (a) \mathcal{L}_A and (b) \mathcal{L}_B , obtained from \mathcal{L} by shrinking A and B faces, respectively. Qubits are placed on edges, and the stabilizer generators are X -vertex and Z -face operators.

- the X -vertex stabilizers in $TC(\mathcal{L}_A)$ (respectively $TC(\mathcal{L}_B)$) are obtained from X -face stabilizers³ of $CC(\mathcal{L})$ on A (respectively B) faces,
- the Z -face stabilizers in $TC(\mathcal{L}_A)$ (respectively $TC(\mathcal{L}_B)$) are obtained from Z -face stabilizers³ on B faces (respectively A) and Y -face (respectively Z -face) stabilizers on C faces.

To summarize, the unitary $U = \bigotimes_{f \in C} U_f$ transforms the generators of the stabilizer group $CC(\mathcal{L})$ of the color code into generator sets for two stabilizer groups $TC(\mathcal{L}_A)$ and $TC(\mathcal{L}_B)$, which define the toric code on two disjoint lattices \mathcal{L}_A and \mathcal{L}_B obtained from \mathcal{L} by shrinking either A or B faces. This concludes the proof of the equivalence in two-dimensions.

Note that the equivalence between the two-dimensional color code and copies of the toric code has been proven for systems with translation symmetries [Yos11; Bom14b]. Our results not only generalize the previous results to the color code on an arbitrary lattice \mathcal{L} on a closed manifold, but also present an explicit construction of the local unitary and shrunk lattices. This leads to new observations for topological color codes with boundaries, which are presented in Section 6.3.

³ Up to multiplication by X -face stabilizers on certain neighboring faces of color C .

Isomorphism between Pauli subgroups

In this subsection, we prove the existence of the disentangling unitary transformation $U = \bigotimes_{f \in C} U_f$. We begin by developing some useful technical tools concerning subgroups of the Pauli operator group. Consider a system of n qubits and two subgroups of Pauli operators $O_1, O_2 \subseteq \mathbf{Pauli}(n)$, where $\mathbf{Pauli}(n)$ is the Pauli operator group on n qubits. We shall neglect complex phases in O_1, O_2 . We say that O_1 and O_2 are *isomorphic* to each other iff there exists a Clifford unitary transformation U such that

$$UO_1U^\dagger = O_2. \quad (6.10)$$

Let $Z(O_1)$ and $Z(O_2)$ be centers of O_1 and O_2 , respectively. Then, the following lemma holds [YC10]:

Lemma 14 (Isomorphic Groups) *Two subgroups of Pauli operators $O_1, O_2 \subset \mathbf{Pauli}(n)$ are isomorphic iff*

$$G(O_1) = G(O_2), \quad G(Z(O_1)) = G(Z(O_2)), \quad (6.11)$$

where $G(O)$ represents the number of independent generators of $O \subset \mathbf{Pauli}(n)$.

Let $\{g_j\}$ and $\{h_j\}$ be two sets of independent generators for two isomorphic groups O_1 and O_2 . We say that $\{g_j\}$ and $\{h_j\}$ satisfy the same commutation relations if

$$\forall i, j : g_i g_j g_i^\dagger g_j^\dagger = h_i h_j h_i^\dagger h_j^\dagger. \quad (6.12)$$

We have the following lemma.

Lemma 15 (Clifford Transformation) *Let O_1 and O_2 be two isomorphic groups generated by two sets of independent generators, $\{g_j\}$ and $\{h_j\}$. If $\{g_j\}$ and $\{h_j\}$ have the same commutation relations, then there exists a Clifford unitary transformation U such that*

$$U g_j U^\dagger = h_j \quad \forall j. \quad (6.13)$$

Proof: Let us find a set of independent generators for O_1 , which we call canonical:

$$O_1 = \left\langle \begin{array}{cccccc} A_1, & \dots, & A_{n_1}, & A_{n_1+1}, & \dots, & A_{n_2} \\ A_{n_2+1}, & \dots, & A_{n_1+n_2} & & & \end{array} \right\rangle, \quad (6.14)$$

where $n_2 \geq n_1$, and two Pauli operators A_i and A_j commute unless they are in the same column, in which case they anti-commute by definition. Note that n_1 is the number of pairs of anticommuting canonical generators, whereas n_2 is the number of commuting canonical generators. Observe that any canonical generator can be written as a product of generators $\{g_j\}$.

For a binary vector $\vec{a} = (a_1, \dots, a_{n_1+n_2})$, we define

$$\mathcal{O}_1(\vec{a}) = \prod_{j=1}^{n_1+n_2} g_j^{a_j}, \quad \mathcal{O}_2(\vec{a}) = \prod_{j=1}^{n_1+n_2} h_j^{a_j}. \quad (6.15)$$

Then, there exists a set of independent $n_1 + n_2$ binary vectors $\vec{a}^{(j)}$ such that

$$A_j = \mathcal{O}_1(\vec{a}^{(j)}). \quad (6.16)$$

Let $B_j = \mathcal{O}_2(\vec{a}^{(j)})$. Since commutation relations of $\{g_j\}$ and $\{h_j\}$ are identical, then B_j are canonical generators for \mathcal{O}_2 :

$$\mathcal{O}_2 = \left\langle \begin{array}{cccccc} B_1, & \dots, & B_{n_1}, & B_{n_1+1}, & \dots, & B_{n_2} \\ B_{n_2+1}, & \dots, & B_{n_1+n_2} & & & \end{array} \right\rangle \quad (6.17)$$

Then, as shown in Ref. [YC10], there exists a Clifford unitary U such that

$$UA_jU^\dagger = B_j \quad \forall j \in \{1, \dots, n_1 + n_2\}. \quad (6.18)$$

Such a unitary transformation also satisfies

$$Ug_jU^\dagger = h_j \quad \forall j \in \{1, \dots, n_1 + n_2\}, \quad (6.19)$$

which completes the proof of the (Clifford Transformation) Lemma 15 \square

We are ready to show the existence of a Clifford unitary U_f , which satisfies the rules in Eqs. (6.5)–(6.8). First, let us introduce the notion of the overlap group of the stabilizer group [YC10]. For a given subset of qubits, denoted by Q , the overlap group on Q is defined as the group generated by the restriction of generators of the stabilizer group \mathcal{S} onto Q . Namely,

$$\mathcal{O}_Q = \langle u|_Q \mid u \in \mathcal{S} \rangle, \quad (6.20)$$

where $u|_Q$ represents a restriction of u onto Q (see Fig. 6.6). Note that the overlap group is not necessarily Abelian and is defined up to a global phase.

The key idea in the proof of existence of U is that the overlap groups for the color code and the toric code for the set of C faces are isomorphic. In particular, let us

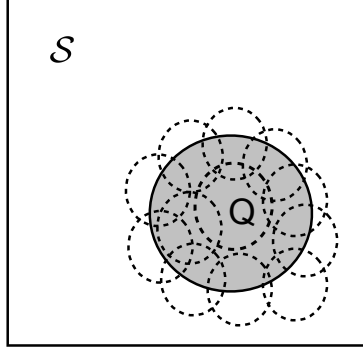


Figure 6.6: The overlap group of the stabilizer group \mathcal{S} on the region Q is defined as the group generated by the restriction of the generators of \mathcal{S} onto Q . Dotted circles represent the stabilizer generators of \mathcal{S} with support intersecting Q .

consider a C face $f \in \mathcal{L}$ with $2n$ vertices, and two corresponding faces $f^A \in \mathcal{L}_A$ and $f^B \in \mathcal{L}_B$ derived from f . Then, the overlap group of $CC(\mathcal{L})$ on f is generated by

$$\mathcal{O}_f = \langle Z_j Z_{j+1}, X_j X_{j+1} \mid j \in \{1, \dots, 2n\} \rangle, \quad (6.21)$$

whereas the overlap group of $TC(\mathcal{L}_A)$ and $TC(\mathcal{L}_B)$ on $f^A \sqcup f^B$ is generated by

$$\mathcal{O}_{f^A \sqcup f^B} = \langle Z_{(j,j+1)}, X_{(j-1,j)} X_{(j+1,j+2)} \mid j \in \{1, \dots, 2n\} \rangle. \quad (6.22)$$

Observe that both \mathcal{O}_f and $\mathcal{O}_{f^A \sqcup f^B}$ have $4n - 2$ independent generators and their centers are generated by 2 independent operators. Namely,

$$G(\mathcal{O}_f) = G(\mathcal{O}_{f^A \sqcup f^B}), \quad (6.23)$$

$$G(Z(\mathcal{O}_f)) = G(Z(\mathcal{O}_{f^A \sqcup f^B})). \quad (6.24)$$

Using the (Isomorphic Groups) Lemma 14, we obtain that \mathcal{O}_f and $\mathcal{O}_{f^A \sqcup f^B}$ are isomorphic.

Let us choose a set of independent generators for \mathcal{O}_f as follows

$$g_j = Z_j Z_{j+1} \quad (j = 1, \dots, 2n - 1), \quad (6.25)$$

$$g_{2n} = \left(\bigotimes_{i=1}^{2n} X_i \right) Z_{2n} Z_1 \quad (6.26)$$

$$g_{j+2n} = X_j X_{j+1} \quad (j = 1, \dots, 2n - 2). \quad (6.27)$$

We then label a set of independent generators for $\mathcal{O}_{f^A \sqcup f^B}$ in the following way

$$h_j = Z_{(j,j+1)} \quad (j = 1, \dots, 2n), \quad (6.28)$$

$$h_{j+2n} = X_{(j-1,j)} X_{(j+1,j+2)} \quad (j = 1, \dots, 2n - 2). \quad (6.29)$$

By direct calculation one can verify that $\{g_j\}$ and $\{h_j\}$ have the same commutation relations. Thus, from the (Clifford Transformation) Lemma 15, there exists a Clifford unitary U_f such that

$$U_f g_j U_f^\dagger = h_j \quad \forall j \in \{1, \dots, 4n - 2\}. \quad (6.30)$$

Therefore, the local Clifford unitary $U = \bigotimes_{f \in \mathcal{C}} U_f$ transforms $CC(\mathcal{L})$ into $TC(\mathcal{L}_A) \otimes TC(\mathcal{L}_B)$, and this completes the proof of Theorem 1.

One might find the labelings in Eqs. (6.25)–(6.29) arbitrary. Yet, once we have chosen g_j for $j = 1, \dots, 2n$, it is not difficult to find the right labeling for $j = 2n + 1, \dots, 4n - 2$ by checking the commutation relations. Note that the choice of $g_{2n} = \left(\bigotimes_{j=1}^{2n} X_j \right) Z_{2n} Z_1$ is crucial to ensure that the generators $\{g_j\}_{j=1}^{2n}$ are independent.

Three (or more) dimensions

A similar equivalence between the topological color code and the toric code holds in any dimension. It can be summarized in the following theorem.

Theorem 2 (Equivalence) *Let $CC(\mathcal{L})$ be the stabilizer group of the topological color code defined on a d -dimensional lattice \mathcal{L} without boundaries, which is $(d + 1)$ -valent and colored with C_0, \dots, C_d . Let X - and Z -type stabilizer generators be supported on respectively d -cells and 2 -cells, where $d \geq 2$. Then, there exists a local Clifford unitary U such that*

$$U[CC(\mathcal{L}) \otimes \mathcal{S}]U^\dagger = \bigotimes_{j=1}^d TC(\mathcal{L}_j), \quad (6.31)$$

where \mathcal{S} represents the stabilizer group of decoupled ancilla qubits, and $TC(\mathcal{L}_j)$ – the stabilizer group of the toric code defined on the shrunk lattice \mathcal{L}_j derived from \mathcal{L} by local deformations, i.e. shrinking d -cells of color C_j . Moreover, one can choose the disentangling unitary U to be of the form

$$U = \bigotimes_{c \in C_0} U_c, \quad (6.32)$$

where C_0 is the set of d -cells of color C_0 in \mathcal{L} , and U_c is a Clifford unitary acting only on qubits on vertices of the d -cell c .

Note that the color code qubits are placed on vertices, whereas the toric code qubits are placed on edges. Thus, for every d -cell c colored in C_0 , we shall add $E - V$

ancilla qubits, where V and E denote the number of vertices and edges in c . We can assume that ancilla qubits are stabilized by single-qubit Pauli Z operators. Since the lattice \mathcal{L} is $(d + 1)$ -valent, then $E = dV/2$ and $E - V \geq 0$ for $d \geq 2$. In particular, ancilla qubits are required for the three- or higher-dimensional cases.

Since the color code and the toric code in d dimensions support excitations whose braiding properties are similar (there exists an isomorphism between anyon labels for two codes), the equivalence should not be very surprising. The study of topological invariants gives valuable insight into the equivalence of models. It has been argued that two topologically ordered systems with isomorphic anyon labels and modular matrices belong to the same topological phase [CGW10; HW05; LW05]. This hypothesis has been proven for two-dimensional stabilizer Hamiltonians with translation symmetries [Yos11]. Also, this hypothesis has been tested for the two-dimensional Levin-Wen model in Ref. [KK12], where a construction of a transparent domain wall between two Levin-Wen models (with tensor unitary categories satisfying certain equivalence conditions) was presented.

The idea of the mapping is a straightforward generalization of the proof of Theorem 1 presented in Section 6.1B. First, we perform a local Clifford unitary, whose existence is guaranteed by the (Clifford Transformation) Lemma 15. Then, we analyze how the stabilizer generators of the color code transform under such a unitary. Finally, we check that the stabilizers can be split into d sets, each of them defining a copy of the toric code on a lattice obtained by deforming the initial lattice \mathcal{L} . For the sake of clarity, we focus on $d = 3$. We also first present the construction of shrunk lattices, before explaining how to construct a local Clifford unitary transforming the color code into d decoupled copies of the toric code.

In three dimensions, the lattice \mathcal{L} has volumes colored with four colors, A , B , C and D . Recall that we can assign colors to faces and edges, too. Namely, a face has two colors of two volumes it belongs to, whereas an edge has three colors (of three volumes it belongs to). We obtain three shrunk lattices \mathcal{L}_A , \mathcal{L}_B and \mathcal{L}_C by shrinking volumes of color A , B and C , respectively. In particular, \mathcal{L}_A consists of

- vertices — centers of A volumes in \mathcal{L} ,
- edges — BCD edges in \mathcal{L} ,
- faces — BC , BD and CD faces in \mathcal{L} ,
- volumes — B , C and D volumes in \mathcal{L} .

For an example, see Fig. 6.7. Similarly for other shrunk lattices \mathcal{L}_B and \mathcal{L}_C . In general, a d -dimensional lattice \mathcal{L} is colored with $d + 1$ colors, C_0, \dots, C_d , and one obtains the shrunk lattice \mathcal{L}_i , where $i = 1, \dots, d$, by shrinking d -cells of color C_i . Namely, \mathcal{L}_i consists of

- vertices — centers of d -cells in \mathcal{L} of color C_i ,
- edges — edges in \mathcal{L} of color $\{C_0, \dots, C_d\} \setminus \{C_i\}$,
- faces — faces in \mathcal{L} of color $\{C_0, \dots, C_d\} \setminus \{C_i, C_j\}$ for all $j \neq i$.

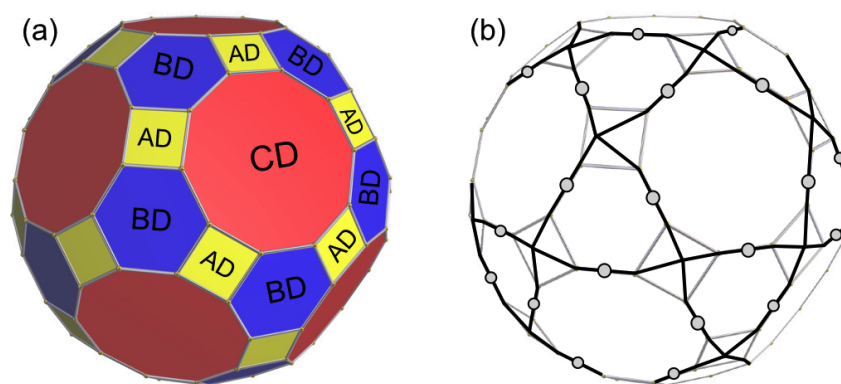


Figure 6.7: (Color online) (a) The boundary ∂c of a volume c of color D in the lattice \mathcal{L} . Note that ∂c can be viewed as a 3-colorable and 3-valent lattice on a closed manifold (a sphere), with faces colored in AD , BD and CD . (b) A volume in the shrunk lattice \mathcal{L}_A derived from c after shrinking volumes of color A . Note that qubits are placed on (a) vertices and (b) edges. The figures were created using Robert Webb's Stella software (<http://www.software3d.com/Stella.php>).

We construct the disentangling unitary U as a tensor product of local Clifford unitaries, $U = \bigotimes_{c \in \mathcal{D}} U_c$, where \mathcal{D} is the set of all volumes of color D . Let us consider a volume c of color D . The overlap group \mathcal{O}_c of the stabilizer group of the color code on c is generated by Z -edge operators and X -face operators, for each and every edge and face belonging to c . Namely,

$$\mathcal{O}_c = \left\langle \begin{array}{c} \mathbf{Z} \quad \mathbf{Z} \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \end{array}, \begin{array}{c} \mathbf{X} \quad \mathbf{X} \\ \text{---} \text{---} \\ \mathbf{X} \quad \mathbf{X} \\ \text{---} \text{---} \\ \mathbf{X} \quad \mathbf{X} \end{array} \right\rangle. \quad (6.33)$$

Let $\mathcal{H}_V \simeq (\mathbb{C}^2)^{\otimes V}$ and $\mathcal{H}_E \simeq (\mathbb{C}^2)^{\otimes E}$ be the Hilbert spaces of color code qubits and toric code qubits, respectively placed on vertices and edges of the volume c . Since $E - V > 0$, we need to add $E - V$ ancilla qubits to qubits on vertices to match the

dimensionality of Hilbert spaces, $\mathcal{H}_V \otimes \mathcal{H}_{ancilla} \simeq \mathcal{H}_E$, where $\mathcal{H}_{ancilla}$ is the Hilbert space of ancilla qubits. Let $\mathcal{S}_c = \langle Z_i \mid \forall i \in \{1, \dots, E - V\} \rangle$ be the stabilizer group of the ancilla qubits, where Z_i is the Pauli Z operator acting on the ancilla qubit i . We would like to construct a Clifford unitary U_c which maps the group $\mathcal{O}_c \otimes \mathcal{S}_c$ of operators on the Hilbert space $\mathcal{H}_V \otimes \mathcal{H}_{ancilla}$ into the group

$$\mathcal{O}_c^{TC} = \left\langle \begin{array}{c} \mathbf{Z} \\ \diagup \quad \diagdown \\ \diagdown \quad \diagup \\ \mathbf{Z} \end{array}, \begin{array}{c} \mathbf{X} \quad \mathbf{X} \\ \diagdown \quad \diagup \\ \diagup \quad \diagdown \\ \mathbf{X} \quad \mathbf{X} \end{array} \right\rangle \quad (6.34)$$

of operators on \mathcal{H}_E according to the rules

$$\left(\begin{array}{c} \mathbf{Z} \quad \mathbf{Z} \\ \diagup \quad \diagdown \\ \diagdown \quad \diagup \end{array} \right) \rightarrow \begin{array}{c} \mathbf{Z} \\ \diagup \quad \diagdown \\ \diagdown \quad \diagup \end{array}, \quad \left(\begin{array}{c} \mathbf{X} \quad \mathbf{X} \\ \diagdown \quad \diagup \\ \diagup \quad \diagdown \end{array} \right) \rightarrow \begin{array}{c} \mathbf{X} \quad \mathbf{X} \\ \diagdown \quad \diagup \\ \diagup \quad \diagdown \end{array}. \quad (6.35)$$

The parenthesis indicate that the mapping holds up to multiplication by the elements of the center $Z(\mathcal{O}_c \otimes \mathcal{S}_c)$.

Let us analyze what happens to the stabilizer group of the color code and the stabilizer group of ancilla qubits, $CC(\mathcal{L}) \otimes_{c \in \mathcal{D}} \mathcal{S}_c$, after applying the unitary $U = \otimes_{c \in \mathcal{D}} U_c$. One can verify that

- X -vertex stabilizers of $TC(\mathcal{L}_A)$, $TC(\mathcal{L}_B)$ and $TC(\mathcal{L}_C)$ are obtained from X -volume stabilizers⁴ in $CC(\mathcal{L})$ of color A , B and C , respectively,
- Z -face stabilizers in $TC(\mathcal{L}_A)$ are obtained from Z -face stabilizers⁴ of color BD , CD and BC ; similarly for $TC(\mathcal{L}_B)$ and $TC(\mathcal{L}_C)$,
- the elements in the center $Z(\mathcal{O}_c \otimes \mathcal{S}_c)$ are mapped into the center $Z(\mathcal{O}_c^{TC})$.

Moreover, the generators of the group $U \left(CC(\mathcal{L}) \otimes_{c \in \mathcal{D}} \mathcal{S}_c \right) U^\dagger$ are supported on either \mathcal{L}_A , or \mathcal{L}_B , or \mathcal{L}_C , and thus one obtains three decoupled copies of the toric code.

The last thing we need to justify is the existence of U_c consistent with the rules in Eq. (6.35). We start with showing that $\mathcal{O}_c \otimes \mathcal{S}_c$ and \mathcal{O}_c^{TC} are isomorphic. Clearly, $\mathcal{O}_c \otimes \mathcal{S}_c, \mathcal{O}_c^{TC} \subset \mathbf{Pauli}(n = E)$. First, let us look at the independent generators of $\mathcal{O}_c \otimes \mathcal{S}_c$. There are $V - 1$ independent operators of type $\begin{array}{c} \mathbf{Z} \quad \mathbf{Z} \\ \diagup \quad \diagdown \\ \diagdown \quad \diagup \end{array}$, denoted by $\{g_i\}_{i=1}^{V-1}$, supported on edges of a (spanning) tree $T \subset E$ of the graph $G = (V, E)$. To see

⁴ Up to multiplication by elements of the center $Z(\mathcal{O}_c \otimes \mathcal{S}_c)$ for any neighboring volume c of color D .

this, note that edge operators are independent as long as the corresponding edges do not form a closed loop. This implies that independent edge generators correspond to a tree consisting of $V - 1$ edges in the graph G . In the case of operators of type



, there are exactly two independent relations between them, namely

$$\prod_{f \in \mathcal{AD}} \begin{array}{c} \text{X-X} \\ \text{X-f-X} \\ \text{X-X} \end{array} = \prod_{f \in \mathcal{BD}} \begin{array}{c} \text{X-X} \\ \text{X-f-X} \\ \text{X-X} \end{array} = \prod_{f \in \mathcal{CD}} \begin{array}{c} \text{X-X} \\ \text{X-f-X} \\ \text{X-X} \end{array}, \quad (6.36)$$

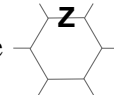
where the products are taken over all X -face operators associated with faces of c of color AD , BD , and CD , respectively. Thus, there are $F - 2$ independent X -face operators. We set $F - 3$ generators $\{g_i\}_{i=V}^{V+F-4}$ to be X -face operators, associated with all faces of c but three — one of each color AD , BD and CD . We also set $g_{V+F-3} = \bigotimes_{v \in V} X(v)$, where $\bigotimes_{v \in V} X(v)$ is the X -volume operator on c . Including $E - V$ single-qubit Pauli Z stabilizer generators $\{g_i\}_{i=V+F-2}^{E+F-3}$ for ancilla qubits, there are


$$(V - 1) + (F - 2) + (E - V) = E + F - 3 \quad (6.37)$$

independent generators of $O_c \otimes S_c$, and thus $G(O_c \otimes S_c) = E + F - 3$. Note that since

$$Z(O_c \otimes S_c) = \left\langle \begin{array}{c} \mathbf{z-z} \\ \mathbf{z-z} \\ \mathbf{z-z} \end{array}, \bigotimes_{v \in V} X(v), Z_i \right\rangle, \quad (6.38)$$

then $G(Z(O_c \otimes S_c)) = (F - 2) + 1 + (E - V)$.


In the case of O_c^{TC} , there are E independent generators of type . Observe

that there are only three independent relations between generators of type , namely

$$\prod_{f \in \mathcal{AD}} \begin{array}{c} \text{X-X} \\ \text{X-f-X} \\ \text{X-X} \end{array} = \prod_{f \in \mathcal{BD}} \begin{array}{c} \text{X-X} \\ \text{X-f-X} \\ \text{X-X} \end{array} = \prod_{f \in \mathcal{CD}} \begin{array}{c} \text{X-X} \\ \text{X-f-X} \\ \text{X-X} \end{array} = I, \quad (6.39)$$

and thus $G(O_c^{TC}) = E + (F - 3)$. Since the group O_c^{TC} has single qubit Pauli Z operators as generators, the center $Z(O_c^{TC})$ can only be generated by Z -type operators,

$$Z(O_c^{TC}) = \left\langle \begin{array}{c} \mathbf{z-z} \\ \mathbf{z-z} \\ \mathbf{z-z} \end{array} \right\rangle. \quad (6.40)$$

There are $2F$ operators of type , and they satisfy three independent relations, namely a product of all Z -face operators with qubits placed on edges not colored in

i , for $i \in \{A, B, C\}$. Thus $G(Z(\mathcal{O}_c^{TC})) = 2F - 3$ and using Euler characteristic for c , $V - E + F = 2$, we obtain $G(Z(\mathcal{O}_c^{TC})) = G(Z(\mathcal{O}_c \otimes \mathcal{S}_c))$. From the (Isomorphic Groups) Lemma 14 we obtain that $\mathcal{O}_c \otimes \mathcal{S}_c$ and \mathcal{O}_c^{TC} are isomorphic.

We have already chosen independent generators $\{g_i\}$ of $\mathcal{O}_c \otimes \mathcal{S}_c$. We choose independent generators $\{h_i\}$ of \mathcal{O}_c^{TC} as follows:

- for $i = 1, \dots, V - 1$: $g_i = \begin{array}{c} \mathbf{Z} \text{---} \mathbf{Z} \\ | \quad | \\ \text{---} \text{---} \\ | \quad | \\ \text{---} \text{---} \end{array} \rightarrow h_i = \begin{array}{c} \mathbf{Z} \\ | \\ \text{---} \text{---} \\ | \quad | \\ \text{---} \text{---} \end{array},$
- for $i = V, \dots, V + F - 4$: $g_i = \begin{array}{c} \mathbf{X} \text{---} \mathbf{X} \\ | \quad | \\ \mathbf{X} \text{---} \mathbf{X} \\ | \quad | \\ \mathbf{X} \text{---} \mathbf{X} \end{array} \rightarrow h_i = \begin{array}{c} \mathbf{X} \quad \mathbf{X} \\ | \quad | \\ \mathbf{X} \text{---} \mathbf{X} \\ | \quad | \\ \mathbf{X} \quad \mathbf{X} \end{array},$
- for $i = V + F - 3$: $g_i = \bigotimes_{v \in V} X(v) \rightarrow h_i = \bigotimes_{e \in E} Z(e),$
- for $i = V + F - 2, \dots, E + F - 3$: $g_i = Z_i \rightarrow h_i \in Z(\mathcal{O}_c^{TC}),$

where Z_i is the Pauli Z operator on the ancilla qubit i . We would like to emphasize that the choice of $\{h_i\}_{i=V+F-2}^{E+F-3}$ does not matter, as long as they belong to the center $Z(\mathcal{O}_c^{TC})$ and $\{h_i\}_{i=1}^{E+F-3}$ is the set of independent operators. One can verify that $\{g_i\}$ and $\{h_i\}$ have the same commutation relations, and thus from the (Clifford Transformation) Lemma 15, there exists a Clifford unitary U_c such that

$$U_c g_i U_c^\dagger = h_i \quad \forall i \in \{1, \dots, E + F - 3\}. \quad (6.41)$$

Moreover, the choice of generators $\{g_i\}$ and $\{h_i\}$ guarantees that the rules in Eq. (6.35) are satisfied. This concludes the proof of the (Equivalence) Theorem 2 in $d = 3$ dimensions.

Finally, we present a proof sketch for a higher-dimensional case (for a rigorous proof, see the next section). The color code is defined on a d -dimensional lattice \mathcal{L} with d -cells colored in C_0, C_1, \dots, C_d . Let c be a d -cell in \mathcal{L} of color C_0 with V vertices, E edges and F ($d - 1$)-cells. Let \mathcal{O}_c be the overlap group of the stabilizer group $CC(\mathcal{L})$ of the color code on c and \mathcal{S}_c be the stabilizer group of $E - V$ ancilla qubits. Note that $\mathcal{O}_c \otimes \mathcal{S}_c$ is generated by Z -edge operators, X -type ($d - 1$)-cell-like operators and single Pauli Z operators on ancilla qubits. Thus, $G(\mathcal{O}_c \otimes \mathcal{S}_c) = (V - 1) + (F - d + 1) + (E - V) = E + F - d$. Let \mathcal{O}_c^{TC} be defined as a group of operators on qubits placed on the edges of c . Namely, \mathcal{O}_c^{TC} is generated by single qubit Pauli Z operators on edges and X -vertex-like operators with support on all edges radiating out of ($d - 1$)-cells of c . Note that there are

d independent relations between X -vertex-like operators, namely a product of all X -vertex-like operators associated with $(d - 1)$ -cells of certain color is identity. Thus, $G(\mathcal{O}_c^{TC}) = E + F - d$. By relating the number of independent generators of $Z(\mathcal{O}_c \otimes \mathcal{S}_c)$ and $Z(\mathcal{O}_c^{TC})$ to the number of i -cells of c , for $i = 0, 1, \dots, d$, and the Betti numbers of c , we can prove $G(Z(\mathcal{O}_c \otimes \mathcal{S}_c)) = G(Z(\mathcal{O}_c^{TC}))$ (see the next section for more details). From the (Isomorphic Groups) Lemma 14 we obtain that $\mathcal{O}_c \otimes \mathcal{S}_c$ and \mathcal{O}_c^{TC} are isomorphic. We then choose independent generators $\{g_i\}$ and $\{h_i\}$ of $\mathcal{O}_c \otimes \mathcal{S}_c$ and \mathcal{O}_c^{TC} as follows

- $\{g_i\}_{i=1}^{V-1}$ — independent Z -edge operators related to a spanning tree $T \subset E$ of the graph $G = (V, E) \rightarrow \{h_i\}_{i=1}^{V-1}$ — single qubit Pauli Z operators on qubits placed on edges associated with the spanning tree T ,
- $\{g_i\}_{i=V}^{F+V-d-1}$ — independent X -type $(d - 1)$ -cell operators associated with all $(d - 1)$ -cells of c except for d of them, namely one $(d - 1)$ -cell for each colors $C_0C_1, C_0C_2, \dots, C_0C_d \rightarrow \{h_i\}_{i=V}^{F+V-d-1}$ — X -vertex-like operators with support on edges radiating out of $F - d$ corresponding $(d - 1)$ -cells of c ,
- $g_{i=F+V-d} = \bigotimes_{v \in V} X(v) \rightarrow h_{i=F+V-d} = \bigotimes_{e \in E} Z(e)$,
- $\{g_i\}_{i=F+V-d+1}^{E+F-d}$ — single Pauli Z operators on ancilla qubits $\rightarrow \{h_i\}_{i=F+V-d+1}^{E+F-d} \in Z(\mathcal{O}_c^{TC})$ — elements of the center of \mathcal{O}_c^{TC} chosen in such a way that all the operators $\{h_i\}$ are independent.

One can verify that $\{g_i\}$ and $\{h_i\}$ have the same commutation relations. From the (Clifford Transformation) Lemma 15, there exists of a local Clifford unitary U_c such that

$$U_c g_i U_c^\dagger = h_i \quad \forall i \in \{1, \dots, E + F - d\}. \quad (6.42)$$

By applying the disentangling unitary transformation $U = \bigotimes_{c \in C_0} U_c$ to the stabilizer group $CC(\mathcal{L})$ of the color code and the stabilizer group $\mathcal{S} = \bigotimes_{c \in C_0} \mathcal{S}_c$ of the ancilla qubits, one obtains the stabilizer groups of the toric code supported on d decoupled lattices $\mathcal{L}_1, \dots, \mathcal{L}_d$, namely

$$U[CC(\mathcal{L}) \otimes \mathcal{S}]U^\dagger = \bigotimes_{j=1}^d TC(\mathcal{L}_j), \quad (6.43)$$

which concludes the proof of the (Equivalence) Theorem 2.

6.2 Rigorous proof of the equivalence

Here, we briefly revisit the equivalence of the color code and (multiple decoupled copies of) the toric code in d dimensions without the restriction of point-like excitations. In particular, we focus on the construction of lattices supporting the decoupled copies of the toric code, which can be succinctly described using some notions from algebraic topology. The discussion in this section is presented in the language of the dual lattice unless mention otherwise.

Basic definitions of combinatorial geometry

We start with some basic notions in combinatorial geometry. A d -simplex δ is a convex hull of $d + 1$ affinely independent vertices v_0, v_1, \dots, v_d , namely

$$\delta = \left\{ \sum_{i=0}^d t_i v_i \mid 0 \leq t_i \wedge \sum_{i=0}^d t_i = 1 \right\}. \quad (6.44)$$

There is a combinatorial definition of a simplex, which we adopt for the rest of the discussion. Namely, a d -simplex δ is the power set of the set of vertices $V = \{v_0, \dots, v_d\}$ spanning it, $\delta = \mathcal{P}(V)$. A subset $W \subset V$ of size $k + 1 \leq d + 1$ spans a k -simplex $\sigma = \mathcal{P}(W)$, and we call σ a k -face of δ . We denote the set of all k -faces of δ by $\Delta_k(\delta)$. Note that $\Delta_k(\delta) = \mathcal{P}_{k+1}(V)$, where $\mathcal{P}_k(V)$ denotes the collection of subsets of V of cardinality k .

Let $V = \bigsqcup_{i=1}^k W_i$ be a decomposition of the set of vertices V into the union of k disjoint sets W_1, \dots, W_k . Let $\delta = \mathcal{P}(V)$ and $\sigma_i = \mathcal{P}(W_i)$. Then, we can represent δ as a Cartesian product of its faces $\sigma_1, \dots, \sigma_k$, namely

$$\delta = \sigma_1 \times \dots \times \sigma_k. \quad (6.45)$$

We say that \mathcal{L} is a simplicial d -complex if it is a set of simplices satisfying the following conditions

- every face of a simplex in \mathcal{L} is also in \mathcal{L} ,
- the intersection of two simplices in \mathcal{L} is a face of both of them,
- the dimension of the largest simplex in \mathcal{L} is d ,

If in addition

- for every $k < d$, every k -simplex in \mathcal{L} is a face of a d -simplex in \mathcal{L} ,

then \mathcal{L} is homogeneous. By $\Delta_k(\mathcal{L})$ we denote the set of all k -simplices belonging to \mathcal{L} . An n -skeleton of \mathcal{L} , denoted by $\text{skel}_n(\mathcal{L})$, is a collection of all k -faces of \mathcal{L} for all $k \leq n$, namely $\text{skel}_n(\mathcal{L}) = \bigsqcup_{k=0}^n \Delta_k(\mathcal{L})$.

We might generalize the notion of a simplex to a cell. Namely, a (closed) d -cell δ is the image of a d -dimensional (closed) ball B^d under an attaching map. Similarly to the combinatorial definition of a simplex, we want to think about δ as a collection of all its k -faces, for all $k = 0, 1, \dots, d$. We can define a cell complex⁵ in an analogous way to a simplicial complex, allowing for the faces to be cells.

From now on, we only consider complexes containing finitely many simplices (cells). Although a homogeneous simplicial (cell) d -complex \mathcal{L} is defined as a collection of simplices (cells), by the same symbol we also denote the union of these simplices (cells) as a topological space. In general, \mathcal{L} is a manifold with a boundary embedded in real space, but for the rest of the discussion we assume \mathcal{L} has no boundary. We also assume \mathcal{L} is a homogeneous simplicial d -complex unless stated otherwise.

The n -star of $\delta \in \Delta_k(\mathcal{L})$, denoted by $\text{St}_n(\delta)$, is the set of all n -simplices in \mathcal{L} which contain δ as a face, namely

$$\text{St}_n(\delta) = \{\sigma \in \Delta_n(\mathcal{L}) \mid \sigma \supset \delta\}. \quad (6.46)$$

Note that $\sigma \in \text{St}_n(\delta) \iff \delta \in \Delta_k(\sigma)$.

The n -link of $\delta \in \Delta_k(\mathcal{L})$, denoted by $\text{Lk}_n(\delta)$, is the set of all n -simplices in \mathcal{L} which are the n -faces of d -simplices containing δ , but do not intersect with δ , namely

$$\text{Lk}_n(\delta) = \{\sigma \in \Delta_n(\mathcal{L}) \mid \sigma \subset \tau \in \text{St}_d(\delta) \wedge \sigma \cap \delta = \emptyset\}. \quad (6.47)$$

Observe that for a k -simplex δ in \mathcal{L} there is a one-to-one mapping between the elements of $\text{Lk}_{d-k-1}(\delta)$ and $\text{St}_d(\delta)$, namely

$$\sigma \in \text{Lk}_{d-k-1}(\delta) \xleftrightarrow{\delta \times \sigma = \tau} \tau \in \text{St}_d(\delta). \quad (6.48)$$

We say that \mathcal{L} is $(d + 1)$ -colorable if there exists a function

$$\text{color} : \Delta_0(\mathcal{L}) \rightarrow \mathbb{Z}_{d+1}, \quad (6.49)$$

where $\mathbb{Z}_{d+1} = \{0, 1, \dots, d\}$ is the set of $d + 1$ colors, and two vertices connected by an edge have different colors. We define $\text{color}(\delta)$ to be the set of colors assigned to

⁵For a rigorous definition of a CW complex, see Ref. [Hat02].

vertices of a simplex δ , namely

$$\text{color}(\delta) = \bigsqcup_{v \in \Delta_0(\delta)} \text{color}(v). \quad (6.50)$$

Now, we are ready to define the color code and the toric code in d dimensions. The color code is a stabilizer code with the stabilizer group $CC_k(\mathcal{L})$ defined on a $(d+1)$ -colorable homogeneous simplicial d -complex \mathcal{L} , where $k \in \{0, \dots, d-2\}$. One qubit is placed at each and every d -simplex in \mathcal{L} , and X - and Z -type stabilizer generators are associated with $(d-k-2)$ - and k -simplices as follows

$$\forall \delta \in \Delta_{d-k-2}(\mathcal{L}) : X(\delta) = \bigotimes_{\sigma \in \text{St}_d(\delta)} X(\sigma), \quad (6.51)$$

$$\forall \delta \in \Delta_k(\mathcal{L}) : Z(\delta) = \bigotimes_{\sigma \in \text{St}_d(\delta)} Z(\sigma), \quad (6.52)$$

where $X(\sigma)$ is the Pauli X operator on a qubit placed at $\sigma \in \Delta_d(\mathcal{L})$; similarly $Z(\sigma)$.

The toric code is a stabilizer code with the stabilizer group $TC_k(\mathcal{L})$ defined on a homogeneous cell d -complex \mathcal{L} , where $k \in \{1, \dots, d-1\}$. One qubit is placed at each and every k -cell in \mathcal{L} , and X - and Z -type stabilizer generators are associated with $(k+1)$ - and $(k-1)$ -cells in the following way

$$\forall \delta \in \Delta_{k+1}(\mathcal{L}) : X(\delta) = \bigotimes_{\sigma \in \Delta_k(\delta)} X(\sigma), \quad (6.53)$$

$$\forall \delta \in \Delta_{k-1}(\mathcal{L}) : Z(\delta) = \bigotimes_{\sigma \in \text{St}_k(\delta)} Z(\sigma). \quad (6.54)$$

Equivalence revisited

Let us revisit the (Equivalence) Theorem 2. Note that for the sake of simplicity we assumed earlier that the color code on a d -dimensional lattice \mathcal{L} has point-like excitations, and this corresponds to the $CC_{d-2}(\mathcal{L})$ case. Now we state the equivalence between the color code and the toric code in full generality in the following theorem.

Theorem 3 *Let the topological color code $CC_k(\mathcal{L})$ be defined on a $(d+1)$ -colorable homogeneous simplicial d -complex \mathcal{L} without boundary, where $0 \leq k \leq d-2$. Then, there exists a local Clifford unitary U such that*

$$U [CC_k(\mathcal{L}) \otimes \mathcal{S}_1] U^\dagger = \bigotimes_{N \in \mathcal{P}_{d-1-k}(\mathbb{Z}_d)} TC_{k+1}(\mathcal{L}_N) \otimes \mathcal{S}_2 \quad (6.55)$$

where \mathcal{S}_1 and \mathcal{S}_2 represent the stabilizer groups of decoupled ancilla qubits, and $TC_{k+1}(\mathcal{L}_N)$ is a copy of the toric code defined on a homogeneous cell $(k+2)$ -complex \mathcal{L}_N obtained from \mathcal{L} by removing all simplices with faces of colors in N . Moreover, one can choose the disentangling unitary U to be of the form

$$U = \bigotimes_{\substack{\delta \in \Delta_0(\mathcal{L}) \\ \text{color}(\delta) = \{d\}}} U(\delta), \quad (6.56)$$

where $U(\delta)$ is a Clifford unitary acting only on qubits placed on d -simplices in $\text{St}_d(\delta)$, and some ancilla qubits associated with δ .

Note that after the disentangling one obtains $\binom{d}{d-1-k} = \binom{d}{k+1}$ decoupled copies of the toric code, enumerated by different choices of the subset N of $d-1-k$ colors from \mathbb{Z}_d . Moreover, one might need to locally add ancilla qubits either to the color code, or the toric code depending on the simplicial d -complex \mathcal{L} . Clearly, the (Equivalence) Theorem 2 is a special case of Theorem 3, with $k = d-2$, $\mathcal{S}_2 = \emptyset$ and $C_0 = d$. The rest of this section is devoted to the construction of the cell complexes supporting decoupled copies of the toric code and the explanation of how to find a local Clifford unitary U .

To obtain \mathcal{L}_N from \mathcal{L} , where $N \in \mathcal{P}_{d-1-k}(\mathbb{Z}_d)$, one follows the following procedure.

1. Take the $(k+1)$ -skeleton $\text{skel}_{k+1}(\mathcal{L})$ of \mathcal{L} and construct a new $(k+1)$ -skeleton, $\text{skel}'_{k+1}(\mathcal{L})$, by removing from $\text{skel}_{k+1}(\mathcal{L})$ all simplices with faces of colors in N , namely

$$\text{skel}'_{k+1}(\mathcal{L}) = \{\sigma \in \text{skel}_{k+1}(\mathcal{L}) \mid \text{color}(\sigma) \subset \mathbb{Z}_{d+1} \setminus N\}. \quad (6.57)$$

2. For every $\tau \in \Delta_{d-2-k}(\mathcal{L})$, such that $\text{color}(\tau) = N$, attach a $(k+2)$ -cell to $\text{Lk}_{k+1}(\tau) \subset \text{skel}'_{k+1}(\mathcal{L})$. Resulting $(k+2)$ -skeleton is \mathcal{L}_N .

Note that in Step 2 we used a fact that $\text{Lk}_{k+1}(\tau)$ is homeomorphic to a $(k+1)$ -sphere, and thus we can attach a $(k+2)$ -ball to $\text{Lk}_{k+1}(\tau)$ (see Ref. [Gla72] for a proof and an illustrative discussion on combinatorial manifolds).

The disentangling unitary U in Eq. (6.56) has a tensor product structure. Thus, let us have a closer look at one of its constituents, $U(\delta)$, where δ is a 0-simplex in \mathcal{L} of color $\{d\}$. Let $U(\delta)$ be a Clifford unitary transforming the Hilbert space of color code qubits placed on d -simplices in $\text{St}_d(\delta)$ and $A = |\text{St}_{k+1}(\delta)| - |\text{St}_d(\delta)|$ ancilla

qubits⁶, $\mathcal{H}(\text{St}_d(\delta)) \otimes \mathcal{H}_{\text{ancilla}}$, into the Hilbert space $\mathcal{H}(\text{St}_{k+1}(\delta))$ of toric code qubits placed on $(k+1)$ -simplices in $\text{St}_{k+1}(\delta)$. Let $\mathcal{O}_\delta := \mathcal{O}_{\text{St}_d(\delta)}$ be the overlap group of the color code $CC_k(\mathcal{L})$ on the set of qubits $\text{St}_d(\delta)$, and \mathcal{S}_δ be the stabilizer group generated by single-qubit Pauli Z operators on the ancilla qubits, namely

$$\mathcal{O}_\delta = \left\langle \bigotimes_{\alpha \in \text{St}_d(\sigma)} X(\alpha), \bigotimes_{\alpha \in \text{St}_d(\tau)} Z(\alpha) \mid \forall \sigma \in \text{St}_{d-1-k}(\delta), \tau \in \text{St}_{k+1}(\delta) \right\rangle, \quad (6.58)$$

$$\mathcal{S}_\delta = \langle Z_i \mid \forall i \in \{1, \dots, A\} \rangle. \quad (6.59)$$

Let \mathcal{O}_δ^{TC} be a group of operators acting on qubits placed on $(k+1)$ -simplices in $\text{St}_{k+1}(\delta)$ defined as follows

$$\mathcal{O}_\delta^{TC} = \left\langle \bigotimes_{\alpha \in \text{Lk}_{k+1}(\sigma) \cap \text{St}_{k+1}(\delta)} X(\alpha), Z(\tau) \mid \forall \sigma \in \text{Lk}_{d-2-k}(\delta), \tau \in \text{St}_{k+1}(\delta) \right\rangle, \quad (6.60)$$

We require that $U(\delta)$ maps the generators of $\mathcal{O}_\delta \otimes \mathcal{S}_\delta$ into the generators of \mathcal{O}_δ^{TC} according to the following rules

$$\forall \sigma \in \text{Lk}_{d-2-k}(\delta) : g_i = \left(\bigotimes_{\alpha \in \text{St}_d(\sigma \times \delta)} X(\alpha) \right) \rightarrow h_i = \bigotimes_{\alpha \in \text{Lk}_{k+1}(\sigma) \cap \text{St}_{k+1}(\delta)} X(\alpha), \quad (6.61)$$

$$\forall \tau \in \text{St}_{k+1}(\delta) : g'_i = \left(\bigotimes_{\alpha \in \text{St}_d(\tau)} Z(\alpha) \right) \rightarrow h'_i = Z(\tau), \quad (6.62)$$

$$i \in \{1, \dots, A\} : g''_i = Z_i \rightarrow h''_i \in Z(\mathcal{O}_\delta^{TC}). \quad (6.63)$$

where the parenthesis indicate that the mapping holds up to multiplication by elements of the center $Z(\mathcal{O}_\delta \otimes \mathcal{S}_\delta)$ and we choose $\{h_i, h'_i, h''_i\}$ to be independent. Note that we had to add $A = |\text{St}_{k+1}(\delta)| - |\text{St}_d(\delta)|$ ancilla qubits to guarantee that $\mathcal{O}_\delta \otimes \mathcal{S}_\delta, \mathcal{O}_\delta^{TC} \in \mathbf{Pauli}(n = |\text{St}_{k+1}(\delta)|)$. One can check that $\{g_i, g'_i, g''_i\}$ and $\{h_i, h'_i, h''_i\}$ have the same commutation relations. The existence of the unitary $U(\delta)$ follows from the (Clifford Transformation) Lemma 15, given $\mathcal{O}_\delta \otimes \mathcal{S}_\delta$ and \mathcal{O}_δ^{TC} are isomorphic. We will show this fact invoking the (Isomorphic Groups) Lemma 14.

We want to verify that $G(\mathcal{O}_\delta \otimes \mathcal{S}_\delta) = G(\mathcal{O}_\delta^{TC})$ and $G(Z(\mathcal{O}_\delta \otimes \mathcal{S}_\delta)) = G(Z(\mathcal{O}_\delta^{TC}))$. First note that the elements of the center $Z(\mathcal{O}_\delta^{TC})$ are generated by only Z -type operators which are derived from k -simplices in $\text{St}_k(\delta)$, namely

$$Z(\mathcal{O}_\delta^{TC}) = \left\langle \bigotimes_{\substack{\tau \in \text{St}_{k+1}(\sigma) \\ \text{color}(\tau) = \text{color}(\sigma) \sqcup \{n_1\}}} Z(\tau) \mid \forall \sigma \in \text{St}_k(\delta), n_1 \in \mathbb{Z}_d \setminus \text{color}(\sigma) \right\rangle. \quad (6.64)$$

Since for any $\sigma \in \text{St}_k(\delta)$ we can choose $n_1 \in \mathbb{Z}_d \setminus \text{color}(\sigma)$ in $\binom{d-k}{1}$ ways, then there are $\binom{d-k}{1} |\text{St}_k(\delta)|$ generators of $Z(\mathcal{O}_\delta^{TC})$. Note that not all of them are independent.

⁶If $A < 0$, then $U(\delta)$ is a map between $\mathcal{H}(\text{St}_d(\delta))$ and $\mathcal{H}(\text{St}_{k+1}(\delta)) \otimes \mathcal{H}_{\text{ancilla}}$.

Rather, they have to satisfy certain relations, which we call \mathcal{R}_1 , namely

$$\forall \varrho_1 \in \text{St}_{k-1}(\delta), \{n_1, n_2\} \subset \mathbb{Z}_d \setminus \text{color}(\varrho_1) : \quad (6.65)$$

$$\prod_{\substack{\sigma \in \text{St}_k(\varrho_1) \\ \text{color}(\sigma) \subset \text{color}(\varrho_1) \sqcup \{n_1, n_2\}}} \left(\bigotimes_{\substack{\tau \in \text{St}_{k+1}(\sigma) \\ \text{color}(\tau) = \text{color}(\varrho_1) \sqcup \{n_1, n_2\}}} Z(\tau) \right) = I. \quad (6.66)$$

Since for any $\varrho_1 \in \text{St}_{k-1}(\delta)$ we can choose $\{n_1, n_2\} \subset \mathbb{Z}_d \setminus \text{color}(\varrho_1)$ in $\binom{d-k+1}{2}$ ways, then there are $\binom{d-k+1}{2} |\text{St}_{k-1}(\delta)|$ relations \mathcal{R}_1 . Note that not all relations \mathcal{R}_1 are independent. They have to satisfy $\binom{d-k+2}{3} |\text{St}_{k-2}(\delta)|$ relations \mathcal{R}_2 obtained for any choice of $\varrho_2 \in \text{St}_{k-2}(\delta)$ and $\{n_1, n_2, n_3\} \subset \mathbb{Z}_d \setminus \text{color}(\varrho_2)$. But relations \mathcal{R}_2 are not independent, and so on. Proper counting of independent relations between generators of $Z(\mathcal{O}_\delta^{TC})$ gives the following alternating sum $|\mathcal{R}_1| - |\mathcal{R}_2| + |\mathcal{R}_3| - \dots + (-1)^{k-1} |\mathcal{R}_k|$. Once the constraints have been properly accounted for, since $G(Z(\mathcal{O}_\delta^{TC}))$ is equal to the number of generators minus the number of independent relations between them, then we obtain

$$G(Z(\mathcal{O}_\delta^{TC})) = \binom{d-k}{1} |\text{St}_k(\delta)| - \binom{d-k+1}{2} |\text{St}_{k-1}(\delta)| + \dots + (-1)^k \binom{d}{k+1} |\text{St}_0(\delta)|. \quad (6.67)$$

Using the fact that the toric code on an n -sphere does not encode logical qubits, we obtain that the number of independent X -type generators of \mathcal{O}_δ^{TC} is equal to $|\text{St}_{k+1}(\delta)| - G(Z(\mathcal{O}_\delta^{TC}))$. Thus, including $|\text{St}_{k+1}(\delta)|$ independent Z -type generators,

$$G(\mathcal{O}_\delta^{TC}) = 2|\text{St}_{k+1}(\delta)| - G(Z(\mathcal{O}_\delta^{TC})). \quad (6.68)$$

To analyze the number of independent generators of $\mathcal{O}_\delta \otimes \mathcal{S}_\delta$ and its center $Z(\mathcal{O}_\delta \otimes \mathcal{S}_\delta)$, we use results from the Appendix D in Ref. [BM07a]. First, let us rephrase our problem in the language of the primal lattice. Note that a 0-simplex δ corresponds to a d -cell c , qubits are placed on vertices of c , and X - and Z -type stabilizers are supported on qubits on vertices of $(k+2)$ - and $(d-k)$ -faces of c . Let ∂c be the boundary of c , which can be thought of as a d -colorable and d -valent homogeneous cell $(d-1)$ -complex. Let us denote by C_i the number of i -faces of c , where $i = 0, \dots, d$. Clearly, $C_i = |\text{St}_{d-i}(\delta)|$. The overlap group of the color code on the qubits of c is thus generated by X - and Z -type operators on $(k+1)$ - and $(d-k-1)$ -faces of c . Note that $(k+1)$ - and $(d-k-1)$ -faces of c can be thought of as faces of ∂c , and thus the number of independent generators of $\mathcal{O}_\delta \otimes \mathcal{S}_\delta$ is

$$G(\mathcal{O}_\delta \otimes \mathcal{S}_\delta) = C_{k+1} - I(d-1, k+1) + C_{d-k-1} - I(d-1, d-k-1) + (C_{d-k-1} - C_0), \quad (6.69)$$

where we included $A = C_{d-k-1} - C_0$ single-qubit Pauli Z operators on the ancilla qubits. By $I(d-1, i)$ we denote the number of independent relations between operators on i -cells of ∂c . In particular (see Eq. (D14) in Ref. [BM07a]),

$$I(d-1, s) = \binom{d-1}{s-1} \sum_{i=0}^{d-1-s} (-1)^i h_{s+i} + \sum_{i=0}^{d-s-2} \binom{s+i}{s-1} (-1)^i C_{s+i+1} \quad (6.70)$$

$$= \binom{d-1}{s-1} (-1)^{d-1-s} + \sum_{i=0}^{d-s-2} \binom{s+i}{i+1} (-1)^i C_{s+i+1}, \quad (6.71)$$

where h_i is the i -th Betti number of ∂c . Since ∂c is homeomorphic to a $(d-1)$ -sphere, then $h_i = 1$ if $i = 0, d-1$; otherwise $h_i = 0$. The center $Z(\mathcal{O}_\delta \otimes \mathcal{S}_\delta)$ is generated by X - and Z -type operators on $(k+2)$ - and $(d-k)$ -faces of c , and single-qubit Pauli Z operators on ancilla qubits. Thus⁷,

$$G(Z(\mathcal{O}_\delta \otimes \mathcal{S}_\delta)) = C_{k+2} - I(d-1, k+2) + C_{d-k} - I(d-1, d-k) + (C_{d-k-1} - C_0). \quad (6.72)$$

We can express Eqs. (6.67) and (6.68) in terms of C_i 's, namely

$$G(Z(\mathcal{O}_\delta^{TC})) = \binom{d-k}{1} C_{d-k} - \binom{d-k+1}{2} C_{d-k+1} + \dots + (-1)^k \binom{d}{k+1} C_d \quad (6.73)$$

$$= \sum_{i=0}^k (-1)^i \binom{d-k+i}{i+1} C_{d-k+i}, \quad (6.74)$$

$$G(\mathcal{O}_\delta^{TC}) = 2C_{d-k-1} - G(Z(\mathcal{O}_\delta^{TC})). \quad (6.75)$$

There are many relations between the number of i -cells of ∂c , which is a d -colorable and d -valent homogeneous cell $(d-1)$ -complex homeomorphic to a $(d-1)$ -sphere. In particular, the following identities hold

$$-\binom{d-1}{s} \chi + (-1)^s C_0 + \sum_{i=0}^{s-1} (-1)^i \binom{d-2-i}{d-1-s} C_{d-1-i} + \sum_{i=s+1}^{d-1} (-1)^i \binom{i-1}{s} C_i = 0, \quad (6.76)$$

for any $s = 0, \dots, d-1$, where $\chi = 1 + (-1)^{d-1}$ (see Eq. (D16) in Ref. [BM07a]). One can straightforwardly verify that $G(\mathcal{O}_\delta \otimes \mathcal{S}_\delta) - G(\mathcal{O}_\delta^{TC}) = 0$ and $G(Z(\mathcal{O}_\delta \otimes \mathcal{S}_\delta)) - G(Z(\mathcal{O}_\delta^{TC})) = 0$, since they are obtained from Eq. (6.76) by setting $s = k$ and $s = k+1$, respectively. This finishes the proof that $\mathcal{O}_\delta \otimes \mathcal{S}_\delta$ and \mathcal{O}_δ^{TC} are isomorphic.

6.3 Topological color code with boundaries

Realistic physical systems have boundaries. Moreover, the transversal implementability of logical gates in the topological color code crucially depends on

⁷If $k = 0, d-2$, then we set $I(d-1, d) = 0$.

the choice of boundaries. In this section we show that the color code defined on a d -dimensional lattice with $d + 1$ boundaries of $d + 1$ distinct colors is equivalent to d copies of the toric code attached together at a $(d - 1)$ -dimensional boundary. We also briefly describe how the choice of boundaries of the color code determines if the copies of the toric code are attached or decoupled. We then discuss such boundaries from the viewpoint of condensation of excitations.

Physical intuition behind folding

We begin with presenting some physical intuition concerning why the toric code with two smooth and two rough boundaries needs to be folded if one hopes for transversal logical non-Pauli gates such as the Hadamard gate \overline{H} . Let us recall known results about gapped boundaries of the toric code. In two spatial dimensions, the toric code may have two types of boundaries, *smooth* and *rough* [BK98]. The rough boundaries are defined as the boundaries with open edges (see Fig. 6.8). Similarly to the toric code without boundaries, there are X -vertex and Z -face stabilizers, although Z -face stabilizers have to be modified along the rough boundaries. An X -type (Z -type) string-like logical operator can only start from and end on smooth (rough) boundaries. One says that the electric charge e , i.e. the violated X -vertex stabilizer, condenses on the rough boundary and the magnetic flux m , i.e. the violated Z -face stabilizer, is confined since single e , unlike m , can be created or absorbed on the rough boundary. Similarly, m condenses and e is confined on the smooth boundary.

Consider the two-dimensional toric code with two smooth and two rough boundaries as depicted in Fig. 6.8(a). Since there is only one pair of anti-commuting logical operators, \overline{X} and \overline{Z} , the code encodes a single logical qubit. There is one crucial difference between the toric code and the color code (with boundaries) — the latter admits transversal implementation of the Hadamard gate \overline{H} while the former does not. Recall that the Hadamard gate swaps Pauli X and Z operators. Suppose that the Hadamard gate can be implemented by a local unitary operator U . Let \overline{X} and \overline{X}' be two equivalent implementations of the logical X operator, supported on string-like horizontal regions (see Fig. 6.8(a)). Then, $U\overline{X}U^\dagger$ implements the logical Z operator, which has to anti-commute with \overline{X}' . On the other hand, since U is a local unitary, then $U\overline{X}U^\dagger$ and \overline{X}' have no overlap, and thus they commute, leading to a contradiction. We conclude that the logical Hadamard gate cannot be implemented by a local unitary operator in the toric code with boundaries. This is a simple version of the argument presented in Ref. [Bev+16].

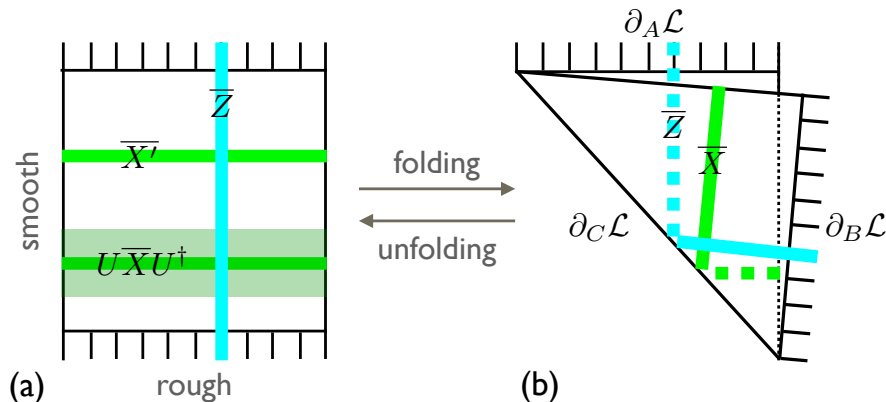


Figure 6.8: (Color online) Origami of the toric code with boundaries. (a) Blue line, starting from and ending on rough boundaries, represents the logical Z operator. Green lines, starting from and ending on smooth boundaries, represent the logical X operator. (b) The color code with three boundaries, $\partial_A \mathcal{L}$, $\partial_B \mathcal{L}$ and $\partial_C \mathcal{L}$, obtained by folding the toric code with two smooth and two rough boundaries. After folding, two logical operators \bar{X} and \bar{Z} are supported on overlapping regions.

We note that if the logical Hadamard is transversal, then both logical X and Z operators must have representations which are supported on overlapping regions. By folding the toric code, both logical X and Z operators can be supported on overlapping regions, as shown in Fig. 6.8(b). Thus, for the logical Hadamard to be transversal folding of the toric code is indeed necessary.

Unfolding in two dimensions

We now return to the analysis of the topological color code $CC(\mathcal{L})$ supported on a (3-valent and 3-colorable) two-dimensional lattice \mathcal{L} with the Euler characteristic⁸ $\chi = 2 - 2g - b$ and the boundary $\partial \mathcal{L} = \bigsqcup_{i=1}^n \partial_i \mathcal{L}$, where $\partial_i \mathcal{L}$ is the (maximum) connected component of the boundary $\partial \mathcal{L}$ of certain color. For conciseness, we simply refer to $\partial_i \mathcal{L}$ as a boundary. We say that the boundary $\partial_i \mathcal{L}$ is of color C_1 if all the faces adjacent to $\partial_i \mathcal{L}$ have colors C_2 and C_3 , where $\{C_1, C_2, C_3\} = \{A, B, C\}$. One can show that the color code $CC(\mathcal{L})$ encodes $\mu - 2\chi + 2(\delta_{n,0} + \delta_{n,b})$ logical qubits, where μ is the number of two-valent vertices belonging to $\partial \mathcal{L}$ and $\delta_{i,j}$ is the Kronecker delta. In particular, one important case corresponds to the triangular color code (with three boundaries of color A , B and C as shown in Fig. 6.9(a); see also Fig. 6.8(b)), which encodes one logical qubit regardless of the system size, and has transversal logical Hadamard \bar{H} and the phase gate \bar{R}_2 .

⁸We can think of \mathcal{L} as a tiling of a 2-manifold \mathcal{M} with boundary, and then the Euler characteristic is $\chi = 2 - 2g - b$, where g is the genus of \mathcal{M} and b is the number of connected components of $\partial \mathcal{M}$.

We would like to understand how the color code $CC(\mathcal{L})$ with boundaries transforms under the disentangling unitary $U = \bigotimes_{f \in C} U_f$ described in Section 6.1B. In the bulk, the disentangling unitary U transforms the stabilizers of the color code into stabilizers of the toric code supported on two decoupled lattices \mathcal{L}_A and \mathcal{L}_B , obtained from \mathcal{L} by shrinking faces of color A and B , respectively. On the other hand, the stabilizers of the color code supported on qubits near the boundaries may transform into stabilizers supported on both shrunk lattices \mathcal{L}_A and \mathcal{L}_B , depending on the colors of $\partial\mathcal{L}$. Unless there is no boundary of color C , we cannot transform the color code $CC(\mathcal{L})$ into the toric code supported on two decoupled lattices, $TC(\mathcal{L}_A) \otimes TC(\mathcal{L}_B)$. Rather, the toric code is defined on a lattice $\mathcal{L}_A \# \mathcal{L}_B$ obtained by *attaching*⁹ \mathcal{L}_A and \mathcal{L}_B , i.e. identifying some of their boundaries. Namely,

$$U[CC(\mathcal{L})]U^\dagger = TC(\mathcal{L}_A \# \mathcal{L}_B). \quad (6.77)$$

In the rest of this subsection we analyze the triangular color code (see Fig. 6.9), but the discussion is applicable to the color code on any homogeneous cell 2-complex with boundary, which is 3-colorable and 3-valent.

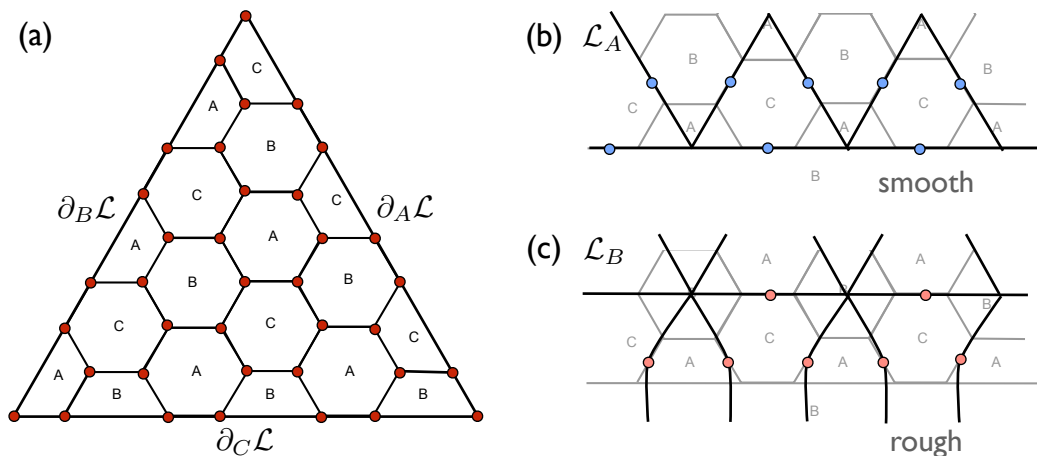


Figure 6.9: (Color online) (a) The (triangular) color code on a two-dimensional lattice \mathcal{L} with the boundary $\partial\mathcal{L}$ comprising three components of color A , B and C , namely $\partial\mathcal{L} = \partial_A\mathcal{L} \sqcup \partial_B\mathcal{L} \sqcup \partial_C\mathcal{L}$. Qubits are represented by dots. (b) A fragment of the lattice \mathcal{L}_A derived from \mathcal{L} by shrinking faces of color A . The smooth boundary arises in \mathcal{L}_A on the boundary $\partial_B\mathcal{L}$. (c) A fragment of the lattice \mathcal{L}_B derived from \mathcal{L} by shrinking faces of color B . The rough boundary arises in \mathcal{L}_B on the boundary $\partial_B\mathcal{L}$.

⁹We would like to point out similarities between the attaching procedure we describe and welding defined in Ref. [Mic14].

Let us describe how to obtain the lattice $\mathcal{L}_A\#\mathcal{L}_B$ supporting the toric code. Recall that in the bulk, \mathcal{L}_A and \mathcal{L}_B are obtained from \mathcal{L} by shrinking faces of color A and B . Let $\partial_A\mathcal{L}$, $\partial_B\mathcal{L}$ and $\partial_C\mathcal{L}$ be the boundaries of color A , B and C , respectively. We find that shrunk lattices \mathcal{L}_A and \mathcal{L}_B are decoupled along $\partial_A\mathcal{L}$ and $\partial_B\mathcal{L}$, but are identified along $\partial_C\mathcal{L}$. In particular,

- on the boundary $\partial_A\mathcal{L}$: the lattice \mathcal{L}_B has open edges (rough boundary), whereas \mathcal{L}_A — no open edges (smooth boundary),
- on the boundary $\partial_B\mathcal{L}$: the lattice \mathcal{L}_A has open edges (rough boundary), whereas \mathcal{L}_B — no open edges (smooth boundary),
- on the boundary $\partial_C\mathcal{L}$: since the disentangling unitary U does not affect the qubits placed on vertices belonging to $\partial_C\mathcal{L}$, both lattices \mathcal{L}_A and \mathcal{L}_B share these qubits.

See Fig. 6.9 and Fig. 6.10(a)(b) for examples of how smooth and rough boundaries arise in the disentangling procedure. Note that along $\partial_C\mathcal{L}$, the lattices \mathcal{L}_A and \mathcal{L}_B are identified. This implies that the electric excitation e on \mathcal{L}_A can be transformed into the excitation e on \mathcal{L}_B by going through the boundary $\partial_C\mathcal{L}$; similarly for the magnetic excitation m .

We can visualize the lattice $\mathcal{L}_A\#\mathcal{L}_B$ by horizontally flipping \mathcal{L}_B and attaching it to \mathcal{L}_A (see Fig. 6.10(c)). Observe that starting from the color code $CC(\mathcal{L})$ with three boundaries, performing the disentangling unitary $U = \bigotimes_{f \in C} U_f$ and unfolding the resulting lattice $\mathcal{L}_A\#\mathcal{L}_B$, one obtains a single copy of the toric code $TC(\mathcal{L}_A\#\mathcal{L}_B)$ with two smooth and two rough boundaries. We can summarize the discussion by the following theorem.

Theorem 4 (Unfolding) *The (triangular) color code $CC(\mathcal{L})$ on a two-dimensional lattice \mathcal{L} with three boundaries, $\partial_A\mathcal{L}$, $\partial_B\mathcal{L}$ and $\partial_C\mathcal{L}$, is equivalent to one folded copy of the toric code $TC(\mathcal{L}_A\#\mathcal{L}_B)$ defined on a lattice $\mathcal{L}_A\#\mathcal{L}_B$ with two smooth and two rough boundaries. Moreover, $\mathcal{L}_A\#\mathcal{L}_B$ is constructed by attaching two lattices \mathcal{L}_A and \mathcal{L}_B (derived from \mathcal{L} by shrinking faces of color A and B , respectively) along the boundary $\partial_C\mathcal{L}$.*

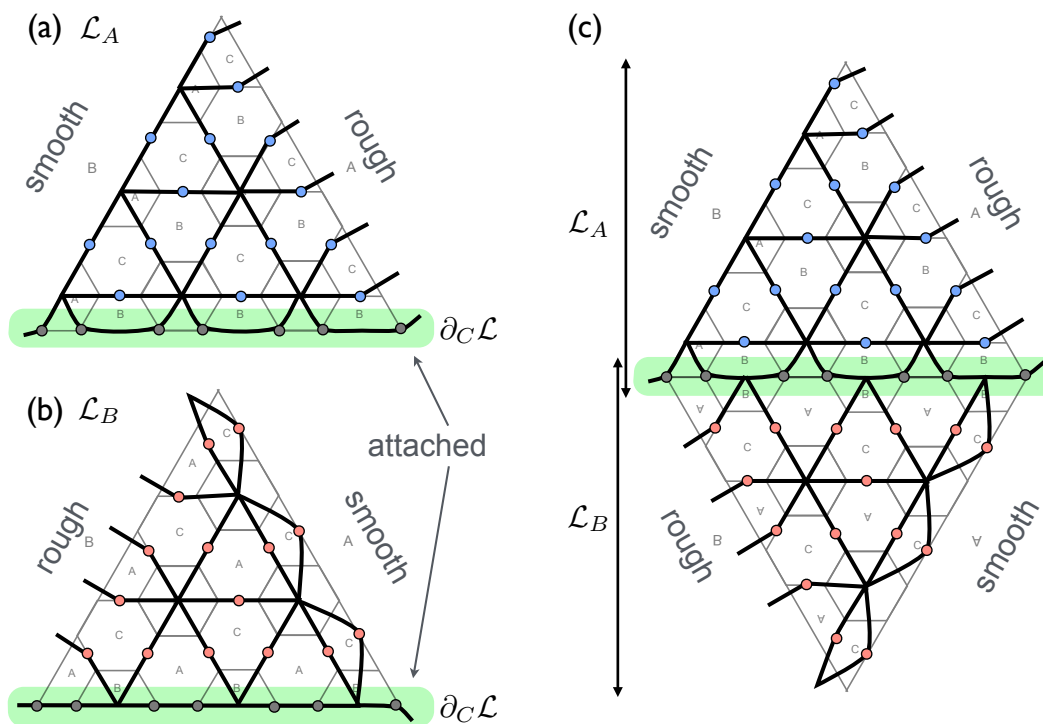


Figure 6.10: (Color online) Attaching two lattices: (a) \mathcal{L}_A and (b) \mathcal{L}_B by identifying qubits along the boundary $\partial_C \mathcal{L}$. (c) Unfolded toric code $TC(\mathcal{L}_A \# \mathcal{L}_B)$. Blue qubits belong to the lattice \mathcal{L}_A , whereas red qubits belong to the (flipped) lattice \mathcal{L}_B .

Three (or more) dimensions

The toric code on a d -dimensional lattice with boundaries, $d \geq 3$, does not differ substantially from the two-dimensional model — qubits are placed on edges, and X - and Z -type stabilizer generators are associated with vertices and faces. There are two types of boundaries, rough and smooth, which may absorb point-like electric charges and $(d-1)$ -dimensional magnetic fluxes, respectively. Moreover, string-like logical Z (respectively $(d-1)$ -dimensional membrane-like logical X) operators can only start from and end on rough (respectively smooth) boundaries (see Fig. 6.14(b)).

The color code can be defined on a $(d+1)$ -valent and $(d+1)$ -colorable d -dimensional lattice \mathcal{L} with the boundaries $\partial \mathcal{L} = \bigsqcup_{i=1}^n \partial_i \mathcal{L}$, where each (maximum) connected component $\partial_i \mathcal{L}$ has one out of $d+1$ colors, C_0, \dots, C_d . We say that $\partial_i \mathcal{L}$ is of color C_j if all d -cells adjacent to $\partial_i \mathcal{L}$ have colors different from C_j . Qubits are placed on vertices, and X - and Z -type stabilizer generators are associated with d -cells and 2-cells (faces), respectively. For the sake of clarity, in the rest of this subsection we focus on the three-dimensional color code $CC(\mathcal{L})$ defined on a tetrahedron-like lattice \mathcal{L} with four boundaries of color A, B, C and D (see Fig. 6.11(a)).

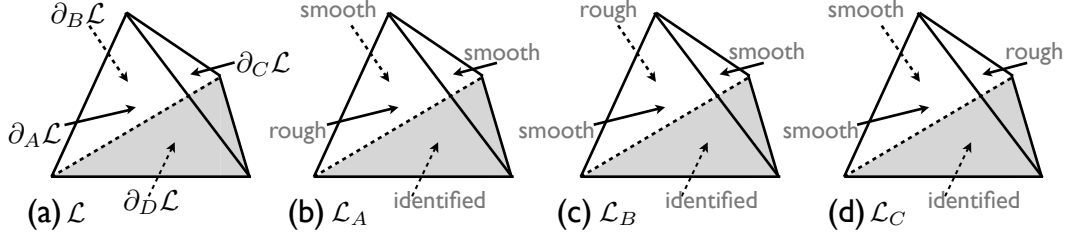


Figure 6.11: (a) A tetrahedron-like lattice \mathcal{L} with boundaries $\partial_A \mathcal{L}$, $\partial_B \mathcal{L}$, $\partial_C \mathcal{L}$ and $\partial_D \mathcal{L}$. Three shrunk lattices: (b) \mathcal{L}_A , (c) \mathcal{L}_B , (d) \mathcal{L}_C derived from \mathcal{L} . The shaded boundary represents the attaching boundary $\partial_D \mathcal{L}$.

We would like to analyze what happens to $CC(\mathcal{L})$ if we apply the disentangling unitary $U = \bigotimes_{c \in \mathcal{D}} U_c$ described in Section 6.1D. In the bulk, the disentangling unitary U transforms the stabilizers of the color code into stabilizers of the toric code supported on three decoupled lattices, \mathcal{L}_A , \mathcal{L}_B and \mathcal{L}_C , obtained from \mathcal{L} by shrinking volumes of color A , B and C , respectively. Since \mathcal{L}_A , \mathcal{L}_B and \mathcal{L}_C share qubits along the boundary $\partial_D \mathcal{L}$, we cannot transform the color code $CC(\mathcal{L})$ into the toric code supported on three decoupled lattices, $TC(\mathcal{L}_A) \otimes TC(\mathcal{L}_B) \otimes TC(\mathcal{L}_C)$. Rather, the toric code is defined on a lattice $\mathcal{L}_A \# \mathcal{L}_B \# \mathcal{L}_C$ obtained by attaching three shrunk lattices along the boundary $\partial_D \mathcal{L}$. We then obtain

$$U[CC(\mathcal{L}) \otimes \mathcal{S}]U^\dagger = TC(\mathcal{L}_A \# \mathcal{L}_B \# \mathcal{L}_C), \quad (6.78)$$

where \mathcal{S} is the stabilizer group of the ancilla qubits. Note that U does not transform qubits on vertices belonging to the boundary $\partial_D \mathcal{L}$.

Let us have a closer look at the shrunk lattices and the identified boundary. \mathcal{L}_i has one rough boundary $\partial_i \mathcal{L}$, and two smooth boundaries $\partial_j \mathcal{L}$ and $\partial_k \mathcal{L}$, where $\{i, j, k\} = \{A, B, C\}$ (see Fig. 6.11). Recall that shrunk lattices share only qubits placed on vertices of the identified boundary $\partial_D \mathcal{L}$. To obtain $\mathcal{L}_A \# \mathcal{L}_B \# \mathcal{L}_C$ one attaches the shrunk lattices by identifying the qubits placed on vertices of $\partial_D \mathcal{L}$ (see Fig. 6.12). Note that a single-qubit Pauli Z operator on a qubit on the boundary $\partial_D \mathcal{L}$ causes three X -vertex stabilizers to be violated, i.e. one in each of three shrunk lattices. Put another way, such an operator creates a triple of electric charges, e_A , e_B and e_C . This implies that the composite electric charge $e_A e_B e_C$ can condense on the boundary $\partial_D \mathcal{L}$. We focus on condensation of excitations on the boundaries in the next subsection.

The discussion here can be straightforwardly generalized to d dimensions, yielding the equivalence between the color code and the toric code with boundaries. We

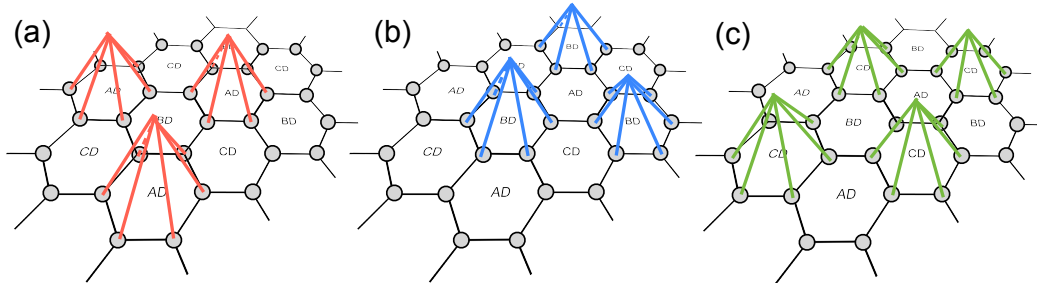


Figure 6.12: (Color online) The identified boundary $\partial_D \mathcal{L}$ of three shrunk lattices: (a) \mathcal{L}_A , (b) \mathcal{L}_B and (c) \mathcal{L}_C . The shrunk lattices are attached by identifying qubits on vertices of $\partial_D \mathcal{L}$.

conclude with the following theorem.

Theorem 5 (Attaching) *Let $CC(\mathcal{L})$ be the color code on a d -simplex-like lattice \mathcal{L} with $d + 1$ boundaries $\partial_0 \mathcal{L}, \dots, \partial_d \mathcal{L}$, where $\partial_i \mathcal{L}$ has color C_i . Then, there exists a local Clifford unitary $U = \bigotimes_{c \in C_0} U_c$ (described in Section 6.1D) such that*

$$U[CC(\mathcal{L}) \otimes \mathcal{S}]U^\dagger = TC(\#_{i=1}^d \mathcal{L}_i), \quad (6.79)$$

where \mathcal{S} is the stabilizer group of the ancilla qubits. The toric code $TC(\#_{i=1}^d \mathcal{L}_i)$ is defined on the lattice $\#_{i=1}^d \mathcal{L}_i$ obtained by attaching lattices $\mathcal{L}_1, \dots, \mathcal{L}_d$ along the boundary $\partial_0 \mathcal{L}$, where \mathcal{L}_i is derived from \mathcal{L} by shrinking d -cells of color C_i , and has one rough boundary, $\partial \mathcal{L}_i$.

Let us mention that the lattice \mathcal{L} from Theorem 5 can be obtained from a certain tiling of a d -sphere (see Ref. [KB15] for details).

Condensation of anyonic excitations

It is instructive to interpret the equivalence between the color code and the toric code with boundaries from the viewpoint of condensation of anyonic excitations. In the two-dimensional toric code, the anyonic excitations are: electric e — a single violated X -vertex stabilizer, magnetic m — a single violated Z -face stabilizer, and fermionic $\epsilon = e \times m$ — a composite excitation obtained by fusing e and m . The label 1 corresponds to the vacuum (no excitations).

The gapped boundaries of two-dimensional systems are classified by maximum sets of mutually bosonic excitations which may condense [KK12; LG12; BJQ13; Lev13; LWW15]. In the case of a single layer of the toric code, possible sets of anyons which

may condense on the boundaries are $\{1, e\}$ and $\{1, m\}$. Note that ϵ has fermionic self-statistics and thus cannot condense on the gapped boundaries. The sets $\{1, e\}$ and $\{1, m\}$ correspond to rough and smooth boundaries, respectively [BK98]. On the other hand, the folded toric code has three boundaries (see Fig. 6.8(b)). If we denote by e_i, m_i and ϵ_i the excitations in the front ($i = 1$) and rear ($i = 2$) layer of the folded toric code, then we can associate the boundaries with the sets of condensing anyons. Namely,

$$\partial_A \mathcal{L} \leftrightarrow \{1, e_1, m_2, e_1 m_2\} \quad (6.80)$$

$$\partial_B \mathcal{L} \leftrightarrow \{1, e_2, m_1, e_2 m_1\} \quad (6.81)$$

$$\partial_C \mathcal{L} \leftrightarrow \{1, e_1 e_2, m_1 m_2, \epsilon_1 \epsilon_2\} \quad (6.82)$$

As depicted in Fig. 6.13(a), two electric charges e_1 and e_2 created on boundaries $\partial_A \mathcal{L}$ and $\partial_B \mathcal{L}$ can be jointly annihilated (or created) on $\partial_C \mathcal{L}$.

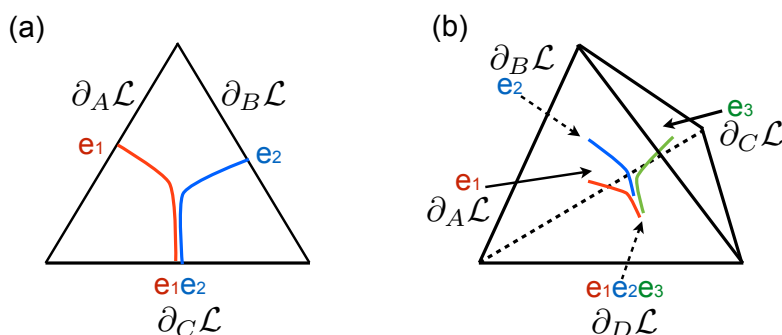


Figure 6.13: (Color online) Condensation of electric charges in (a) two and (b) three dimensions. Observe that single electric charges can condense on all but one boundary, which is the identified boundary. On the identified boundary, a composite electric charge (a) $e_1 e_2$ and (b) $e_1 e_2 e_3$ can be created or annihilated.

By associating the boundaries with the sets of condensing anyons we can find the correspondence between anyonic excitations in the toric code and the color code. We can label excitations in the color code by i_P , where $P \in \{X, Z\}$ indicates the type of the violated stabilizer, and $i \in \{A, B, C\}$ indicates the color of the face associated with the violated stabilizer. Observe that not all six excitations are independent. For instance, a single qubit Pauli X operator on a vertex v creates excitations, A_Z , B_Z and C_Z , on three neighboring faces sharing v . This implies that in the bulk the following fusion channels exist

$$A_X \times B_X \times C_X = 1, \quad A_Z \times B_Z \times C_Z = 1. \quad (6.83)$$

Note that excitations i_X and i_Z can only condense on the boundary $\partial_i \mathcal{L}$. This leads to the following isomorphism between labels of anyonic excitations of the toric and color codes

$$e_1 \leftrightarrow A_X, \quad e_2 \leftrightarrow B_X, \quad m_1 \leftrightarrow B_Z, \quad m_2 \leftrightarrow A_Z. \quad (6.84)$$

In $d > 2$ dimensions, the excitations of the color code are point-like electric charges and $(d - 1)$ -dimensional magnetic fluxes. Let us first focus on condensation of electric charges. We find that the boundaries of the d -dimensional color code on a d -simplex-like lattice are given by

$$\partial_0 \mathcal{L} \leftrightarrow \{e_1 e_2 \dots e_d\}, \quad (6.85)$$

$$\partial_i \mathcal{L} \leftrightarrow \{e_i\} \quad \text{for } i = 1, \dots, d. \quad (6.86)$$

(See Fig. 6.13 for two- and three-dimensional examples). Yet, none of the magnetic fluxes can individually condense on the boundary $\partial_0 \mathcal{L}$. Rather, any pair of fluxes can condense on $\partial_0 \mathcal{L}$, and thus we might think of the fluxes as being equivalent. To sum up, we find the following condensations of $(d - 1)$ -dimensional magnetic fluxes:

$$\partial_0 \mathcal{L} \leftrightarrow \{m_i m_j \mid \forall i \neq j\}, \quad (6.87)$$

$$\partial_i \mathcal{L} \leftrightarrow \{m_j \mid \forall j \neq i\}. \quad (6.88)$$

One may observe that, as expected, the set of condensing magnetic and electric excitations on every boundary is mutually bosonic.

We would like to emphasize that while the gapped boundaries in $(2 + 1)$ -dimensional TQFTs have been thoroughly classified [KK12; BSW11], the understanding of the gapped boundaries in higher-dimensional TQFTs is still incomplete. Characterization of condensing excitations in the color code may provide instructive examples that help with classification of the gapped boundaries in higher-dimensional TQFTs. Namely, different boundaries of various colors in the color code may lead to a rich variety of gapped boundaries in the corresponding toric code models. Moreover, logical action of the transversal \tilde{R}_n operator (see Eq. (6.89)) on the code space crucially depends on the choice of boundaries in the color code. Thus, one may be able to characterize gapped boundaries by analyzing the logical action of transversal operators, and vice versa.

6.4 Transversal gates

We have seen that the color code is equivalent to (multiple copies of) the toric code, both in the presence or the absence of boundaries. Our findings hint that there might be non-trivial logical gates from the d -th level of the Clifford hierarchy in the d -dimensional toric code which admit fault-tolerant implementation. In this section, we show that one can implement by local unitary transformations the logical d -qubit control- Z gate on the stack of d copies of the d -dimensional toric code with point-like excitations.

Transversal R_d operator and boundaries

Let us start with reviewing the transversal implementation of the physical phase gate $R_n = \text{diag}(1, e^{2\pi i/2^n})$ in the color code [Bom15a; KB15]. Consider the topological color code $CC(\mathcal{L})$ on a d -dimensional lattice \mathcal{L} , which is $(d+1)$ -valent and $(d+1)$ -colorable. It is known that the graph $G = (V, E)$ of vertices and edges of \mathcal{L} is bipartite, namely the set of vertices V can be split into two subsets, T and T^c , such that $V = T \sqcup T^c$ and vertices in T are connected only to vertices in T^c , and vice versa. Then, regardless of the lattice \mathcal{L} , the following unitary operator preserves the code space

$$\widetilde{R}_d := \bigotimes_{j \in T} R_d(j) \bigotimes_{j \in T^c} R_d^{-1}(j). \quad (6.89)$$

Here, we adopt a convention that \widetilde{R}_d denotes a transversal operator implemented by physical R_d gates or their powers. When the lattice \mathcal{L} is d -simplex-like (see Section 6.3C and Fig. 6.11), then \widetilde{R}_d implements the logical R_d gate in the code space. For other choices of boundaries, the action of \widetilde{R}_d in the code space does not necessarily coincide with the logical R_d gate.

For the sake of simplicity, in the rest of this section we shall consider the d -dimensional color code supported on a d -hypercube-like lattice \mathcal{L} colored with C_0, \dots, C_d (see Fig. 6.15(a) and Fig. 6.14(a)). In particular, we choose \mathcal{L} to have the opposite boundaries colored in the same color. Namely, we assume that two boundaries perpendicular to the direction \hat{j} have color C_j , where $j = 1, \dots, d$. One can show that the color code $CC(\mathcal{L})$ encodes d logical qubits. In order to do this, consider the disentangling unitary $U = \bigotimes_{c \in C_0} U_c$, which is a tensor product of local unitaries supported on d -cells of color C_0 (see Section 6.1D and Section 6.3C). Then, U transforms the color code $CC(\mathcal{L})$ into d decoupled copies of the toric

code,

$$U[CC(\mathcal{L}) \otimes \mathcal{S}]U^\dagger = \bigotimes_{i=1}^d TC(\mathcal{L}_i) \quad (6.90)$$

where \mathcal{S} is the stabilizer group of the ancilla qubits and the lattice \mathcal{L}_i is derived from \mathcal{L} by shrinking d -cells of color C_i . Moreover, \mathcal{L}_i is a d -hypercube-like lattice with two rough boundaries which are perpendicular to the direction \hat{i} and all the other boundaries smooth. Thus, for $i = 1, \dots, d$, the toric code $TC(\mathcal{L}_i)$ encodes one logical qubit, with a string-like logical Z operator in the direction \hat{i} , and a $(d - 1)$ -dimensional membrane-like logical X operator perpendicular to \hat{i} .

With the above choice of boundaries, \widetilde{R}_d does not implement the logical R_d gate in the code space. One verifies this by observing that $\widetilde{R}_d^2 = I$ in the code space of the color code. Rather, we find that \widetilde{R}_d implements the logical d -qubit control- Z gate on the stack of d copies of the toric code. (Note that a similar observation holds for the color code supported on a d -torus, i.e. a d -hypercube-like lattice with periodic boundary conditions). We devote the rest of this section to describe this finding.

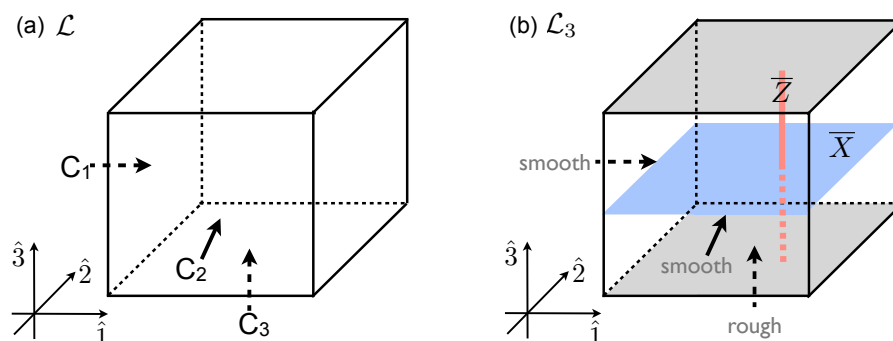


Figure 6.14: (Color online) (a) The color code $CC(\mathcal{L})$ on a three-dimensional cube-like lattice \mathcal{L} with pairs of boundaries perpendicular to the direction \hat{i} colored with C_i . (b) The toric code $TC(\mathcal{L}_3)$ on a cube-like lattice \mathcal{L}_3 derived from \mathcal{L} by shrinking 3-cells of color C_3 . Note that \mathcal{L}_3 has two rough boundaries (shaded) and $TC(\mathcal{L}_3)$ encodes one logical qubit with a string-like logical Z operator (red) connecting two opposite rough boundaries and a membrane-like logical X operator (blue).

Transversal d -qubit control- Z gate in the toric code

We discuss the two-dimensional case first. The topological color code on a square-like lattice \mathcal{L} with four boundaries of color C_1 and C_2 encodes two logical qubits (see Fig. 6.15(a)). We label by $\overline{X}^{(i)}$ and $\overline{Z}^{(i)}$ the logical Pauli X and Z operators, which are perpendicular or parallel to the direction \hat{i} , respectively, for $i = 1, 2$. Since

a unitary operator R_2 transforms Pauli X into XZ and leaves Z unchanged, then the logical operators transform under the conjugation by \widetilde{R}_2 as follows

$$\begin{aligned} \overline{X^{(1)}} &\rightarrow \overline{X^{(1)}} \overline{Z^{(2)}}, & \overline{Z^{(1)}} &\rightarrow \overline{Z^{(1)}}, & \overline{X^{(2)}} &\rightarrow \overline{Z^{(1)}} \overline{X^{(2)}}, & \overline{Z^{(2)}} &\rightarrow \overline{Z^{(2)}}. \end{aligned} \quad (6.91)$$

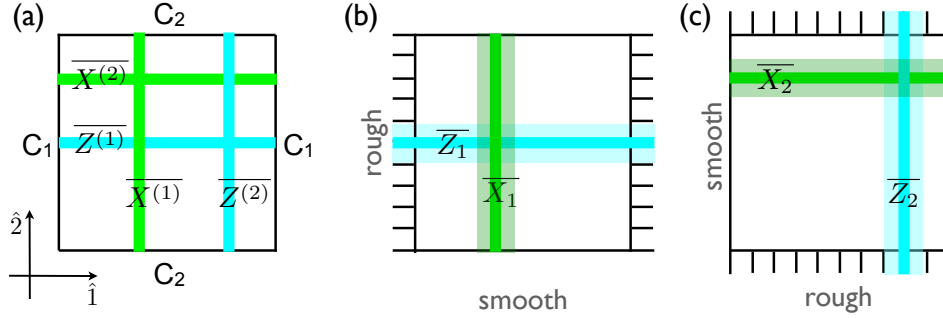


Figure 6.15: (Color online) (a) The topological color code $CC(\mathcal{L})$ on a square-like lattice \mathcal{L} with four boundaries of color C_1 and C_2 encodes two logical qubits, with logical operators $\overline{X^{(i)}}$ and $\overline{Z^{(i)}}$ for $i = 1, 2$. The toric code $TC(\mathcal{L}_i)$ (b) for $i = 1$ and (c) for $i = 2$ derived from \mathcal{L} by shrinking faces of color C_i encodes one logical qubit with logical operators \overline{X}_i and \overline{Z}_i .

Note that since there is no boundary of color C_0 , then the disentangling unitary U (see Eq. (6.90)) transforms the color code $CC(\mathcal{L})$ into two decoupled copies of the toric code, $TC(\mathcal{L}_1)$ and $TC(\mathcal{L}_2)$. The mapping defines an isomorphism between logical operators of the color code and the toric code (see Fig. 6.15). Namely,

$$\overline{X^{(1)}} \leftrightarrow \overline{X}_1 \otimes I, \quad \overline{X^{(2)}} \leftrightarrow I \otimes \overline{X}_2, \quad \overline{Z^{(1)}} \leftrightarrow \overline{Z}_1 \otimes I, \quad \overline{Z^{(2)}} \leftrightarrow I \otimes \overline{Z}_2, \quad (6.92)$$

where $\overline{P}_1 \otimes \overline{P}_2$ denotes an operator which acts as a logical P_1 operator on the first copy $TC(\mathcal{L}_1)$ of the toric code, and as P_2 on the second copy $TC(\mathcal{L}_2)$. Thus, one can immediately deduce the effect of $U\widetilde{R}_2U^\dagger$ on logical operators of $TC(\mathcal{L}_1)$ and $TC(\mathcal{L}_2)$

$$\overline{X}_1 \otimes I \rightarrow \overline{X}_1 \otimes \overline{Z}_2, \quad I \otimes \overline{X}_2 \rightarrow \overline{Z}_1 \otimes \overline{X}_2, \quad \overline{Z}_1 \otimes I \rightarrow \overline{Z}_1 \otimes I, \quad I \otimes \overline{Z}_2 \rightarrow I \otimes \overline{Z}_2. \quad (6.93)$$

This implies that the action of \widetilde{R}_2 in the color code is equivalent (up to the local Clifford unitary U) to the logical control- Z gate between two copies of the toric code.

Let us consider a d -dimensional case, $d \geq 2$. The d -qubit control- Z gate is a generalization of the control- Z gate and is defined in the computational basis as

$$C^{\otimes d-1} Z |x_1, \dots, x_d\rangle = (-1)^{x_1 \dots x_d} |x_1, \dots, x_d\rangle. \quad (6.94)$$

Note that the action of $C^{\otimes d-1}Z$ is invariant under permutation of qubits. Similar to the phase gate R_d , the d -qubit control- Z belongs to the d -th level of the Clifford hierarchy¹⁰ but is outside the $(d-1)$ -th level, which can be seen from the following relations

$$\mathsf{K}[R_n, X] = e^{-2\pi i/2^n} R_{n-1} \propto R_{n-1}, \quad (6.95)$$

$$\mathsf{K}[C^{\otimes n-1}Z, X \otimes I^{\otimes n-1}] = I \otimes C^{\otimes n-2}Z \quad (6.96)$$

where $n \geq 2$ and the commutator is defined as $\mathsf{K}[A, B] = ABA^\dagger B^\dagger$.

We label logical X and Z operators in the color code by $\overline{X^{(i)}}$ and $\overline{Z^{(i)}}$ for $i = 1, \dots, d$. Namely, $\overline{Z^{(i)}}$ is a string-like logical operator parallel to the direction \hat{i} (i.e. connecting two opposite boundaries of color C_i) and $\overline{X^{(i)}}$ is a $(d-1)$ -dimensional membrane-like logical operator perpendicular to the direction \hat{i} . We define the operator \widetilde{R}_i recursively for $i = d-1, \dots, 1$ as follows

$$\widetilde{R}_{d-1} = \mathsf{K}\left[\widetilde{R}_d, \overline{X^{(1)}}\right], \quad (6.97)$$

$$\widetilde{R}_{d-2} = \mathsf{K}\left[\widetilde{R}_{d-1}, \overline{X^{(2)}}\right], \quad (6.98)$$

$$\vdots \quad (6.99)$$

$$\widetilde{R}_1 = \mathsf{K}\left[\widetilde{R}_2, \overline{X^{(d-1)}}\right] = \overline{Z^{(d)}}. \quad (6.100)$$

Note that the above relations hold for any permutation of colors C_1, \dots, C_d . Let \overline{X}_j and \overline{Z}_j be logical X and Z operators in the toric code $TC(\mathcal{L}_j)$. Then, the following correspondence holds

$$\overline{X^{(j)}} \leftrightarrow \overline{X}_j, \quad \overline{Z^{(j)}} \leftrightarrow \overline{Z}_j. \quad (6.101)$$

We can verify that using \widetilde{R}_d one can implement the logical d -qubit control- Z gate on the stack of d copies of the toric code. Namely,

$$I \otimes \overline{C^{\otimes d-2}Z} \propto \mathsf{K}\left[\overline{C^{\otimes d-1}Z}, \overline{X}_1\right], \quad (6.102)$$

$$I^{\otimes 2} \otimes \overline{C^{\otimes d-3}Z} \propto \mathsf{K}\left[I \otimes \overline{C^{\otimes d-2}Z}, \overline{X}_2\right], \quad (6.103)$$

$$\vdots \quad (6.104)$$

$$I^{\otimes d-1} \otimes \overline{Z} \propto \mathsf{K}\left[\overline{CZ}, \overline{X}_{d-1}\right]. \quad (6.105)$$

In the above equations proportionality indicates the same action of the operators on the code space. Since the disentangling unitary U is a local unitary transformation,

¹⁰The Clifford hierarchy is defined recursively: \mathcal{P}_1 is the Pauli group and $\mathcal{P}_j = \{\text{unitary } U | UPU^\dagger \in \mathcal{P}_{j-1} \forall P \in \mathcal{P}_1\}$ for $j > 1$.

$U\widetilde{R}_dU^\dagger$ is a local unitary transformation implementing the logical $\overline{C^{\otimes d-1}Z}$ gate on the stack of d copies of the toric code. We summarize the discussion in this section by the following theorem.

Theorem 6 (Transversal Implementation) *Consider a $(d+1)$ -colorable and $(d+1)$ -valent d -hypercube-like lattice \mathcal{L} with pairs of boundaries perpendicular to the direction \hat{i} colored in C_i for $i = 1, \dots, d$. Let \mathcal{L}_i be a lattice derived from \mathcal{L} by shrinking all d -cells of color C_i . Then, the logical d -qubit control- Z gate can be implemented on $TC(\mathcal{L}_1), \dots, TC(\mathcal{L}_d)$ — the stack of d -copies of the toric code, by a local unitary transformation*

$$\overline{C^{\otimes d-1}Z} \propto U\widetilde{R}_dU^\dagger, \quad (6.106)$$

where \widetilde{R}_d is a transversal R_d gate (see Eq. (6.89)) implemented in the color code $CC(\mathcal{L})$ and U is the disentangling unitary transforming $CC(\mathcal{L})$ into $\bigotimes_{i=1}^d TC(\mathcal{L}_i)$.

Observe that the implementation of $\overline{C^{\otimes d-1}Z}$ seems to require a set of d lattices $\{\mathcal{L}_i\}$ which satisfy certain constraints, i.e. are derived from \mathcal{L} described in the (Transversal Implementation) Theorem 6. In general, it may not be clear whether there exists a local unitary transformation implementing $\overline{C^{\otimes d-1}Z}$ in d copies of the toric code. Yet, one can freely deform the lattices on which the toric code is supported by local operations. Specifically, consider the toric code $TC(\mathcal{L})$ on a d -dimensional lattice \mathcal{L} . We claim that one can transform $TC(\mathcal{L})$ into $TC(\mathcal{L}')$ by local unitary transformations (and adding or removing ancilla qubits), where \mathcal{L}' is a lattice derived from the original lattice \mathcal{L} by adding or removing edges. Such local deformations of the lattices allow us to obtain d copies of the toric code with $\overline{C^{\otimes d-1}Z}$ implementable by local unitary transformations as long as the boundaries of d copies of the toric code are appropriately arranged. In particular, this implies that three copies of the three-dimensional toric code admit fault-tolerant implementation of a logical non-Clifford gate, which saturates the bound by Bravyi and König in three dimensions.

Koniec i bomba, a kto czytał, ten trąba!
—Witold Gombrowicz, “Ferdydurke”

BIBLIOGRAPHY

- [AB97] Dorit Aharonov and Michael Ben-Or. “Fault Tolerant Quantum Computation with Constant Error”. In: *STOC '97 Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*. Nov. 1997, pp. 176–188. DOI: 10.1145/258533.258579. arXiv: 9611025.
- [ADH15] Ahmed Almheiri, Xi Dong, and Daniel Harlow. “Bulk locality and quantum error correction in AdS/CFT”. In: *Journal of High Energy Physics* 2015.4 (Apr. 2015), p. 163. DOI: 10.1007/JHEP04(2015)163.
- [ADP14] Jonas T. Anderson, Guillaume Duclos-Cianci, and David Poulin. “Fault-tolerant conversion between the Steane and Reed-Muller quantum codes”. In: *Phys. Rev. Lett.* 113 (2014), p. 080501. DOI: 10.1103/PhysRevLett.113.080501.
- [AG04] Scott Aaronson and Daniel Gottesman. “Improved simulation of stabilizer circuits”. In: *Phys. Rev. A* 70 (2004), p. 052328. DOI: 10.1103/PhysRevA.70.052328.
- [Ali12] Jason Alicea. “New directions in the pursuit of Majorana fermions in solid state systems”. In: *Reports on Progress in Physics* 75.7 (July 2012), p. 076501. DOI: 10.1088/0034-4885/75/7/076501. arXiv: 1202.1293.
- [And12] Ruben Sarard Andrist. “Understanding topological quantum error-correction codes using classical spin models”. PhD thesis. 2012.
- [Bac06] Dave Bacon. “Operator quantum error-correcting subsystems for self-correcting quantum memories”. In: *Phys. Rev. A* 73 (2006), p. 012340. DOI: 10.1103/PhysRevA.73.012340.
- [BAS92] Albert Lszl Barabasi, Mariela Araujo, and H. Eugene Stanley. “Three-dimensional Toom model: Connection to the anisotropic Kardar-Parisi-Zhang equation”. In: *Physical Review Letters* 68 (1992), pp. 3729–3732. DOI: 10.1103/PhysRevLett.68.3729.
- [BDP12] H. Bombin, Guillaume Duclos-Cianci, and David Poulin. “Universal topological phase of two-dimensional stabilizer codes”. In: *New Journal of Physics* 14 (Mar. 2012). DOI: 10.1088/1367-2630/14/7/073048. arXiv: 1103.4606.
- [Bec+01] David Beckman et al. “Causal and localizable quantum operations”. In: *Phys. Rev. A* 64 (2001), p. 052309. DOI: 10.1103/PhysRevA.64.052309.

- [Ben80] Paul Benioff. “The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines”. In: *Journal of Statistical Physics* 22.5 (May 1980), pp. 563–591. DOI: 10.1007/BF01011339.
- [Bev+16] Michael E Beverland et al. “Protected gates for topological quantum field theories”. In: *Journal of Mathematical Physics* 57.2 (2016), p. 022201.
- [BG85] Charles H. Bennett and G. Grinstein. “Role of irreversibility in stabilizing complex and nonergodic behavior in locally interacting discrete systems”. In: *Physical Review Letters* 55.7 (Aug. 1985), pp. 657–660. DOI: 10.1103/PhysRevLett.55.657.
- [BH12] Sergey Bravyi and Jeongwan Haah. “Magic-state distillation with low overhead”. In: *Physical Review A* 86.5 (2012), p. 052329.
- [BH13] Sergey Bravyi and Jeongwan Haah. “Quantum Self-Correction in the 3D Cubic Code Model”. In: *Physical Review Letters* 111.20 (Nov. 2013), p. 200501. DOI: 10.1103/PhysRevLett.111.200501. arXiv: 1112.3252.
- [BHM10] Sergey Bravyi, Matthew B. Hastings, and Spyridon Michalakis. “Topological quantum order: Stability under local perturbations”. In: *Journal of Mathematical Physics* 51.9 (Sept. 2010), p. 093512.
- [Bin87] K Binder. “Theory of first-order phase transitions”. In: *Rep. Prog. Phys.* 50 (1987), pp. 783–859. DOI: 10.1088/0034-4885/50/7/001.
- [BJQ13] Maissam Barkeshli, Chao-Ming Jian, and Xiao-Liang Qi. “Theory of defects in Abelian topological states”. In: *Phys. Rev. B* 88 (23 Dec. 2013), p. 235103. DOI: 10.1103/PhysRevB.88.235103.
- [BK05] Sergey Bravyi and Alexei Kitaev. “Universal quantum computation with ideal Clifford gates and noisy ancillas”. In: *Physical Review A* 71 (2005), p. 022316. DOI: 10.1103/PhysRevA.71.022316.
- [BK13] Sergey Bravyi and Robert König. “Classification of Topologically Protected Gates for Local Stabilizer Codes”. In: *Phys. Rev. Lett.* 110 (2013), p. 170503. DOI: 10.1103/PhysRevLett.110.170503.
- [BK92] Christian Borgs and Roman Kotecky. “Finite-size effects at asymmetric first-order phase transitions”. In: *Physical Review Letters* 68 (1992), pp. 1734–1737. DOI: 10.1103/PhysRevLett.68.1734.
- [BK98] S. B. Bravyi and A. Yu. Kitaev. “Quantum codes on a lattice with boundary”. In: (1998), p. 6. arXiv: 9811052 [quant-ph].
- [BKS18] Michael Beverland, Aleksander Kubica, and Krysta Svore. “The cost of universality: A comparative study of the overhead of state distillation and code switching in color codes”. In: *in preparation* (2018).

- [BM06] H. Bombin and M. A. Martin-Delgado. “Topological Quantum Distillation”. In: *Physical Review Letters* 97 (2006), p. 180501.
- [BM07a] H. Bombin and M. Martin-Delgado. “Exact topological quantum order in D=3 and beyond: Branyons and brane-net condensates”. In: *Physical Review B* 75 (2007), p. 075103. doi: 10.1103/PhysRevB.75.075103.
- [BM07b] H. Bombin and M.A. Martin-Delgado. “Topological Computation without Braiding”. In: *Phys.Rev.Lett.* 98 (2007), p. 160502.
- [BM09] H Bombin and M A Martin-Delgado. “Quantum measurements and gates by code deformation”. In: *J. Phys. A: Math. Theor.* 42.9 (2009), p. 095302.
- [BNB15] Benjamin J. Brown, Naomi H. Nickerson, and Dan E. Browne. “Fault-Tolerant Error Correction with the Gauge Color Code”. In: *Nature Communications* 7 (2015), p. 4. doi: 10.1038/ncomms12302.
- [Bom+12] H. Bombin et al. “Strong Resilience of Topological Codes to Depolarization”. In: *Physical Review X* 2 (2012), p. 021004. doi: 10.1103/PhysRevX.2.021004.
- [Bom13] H. Bombin. “An Introduction to Topological Quantum Codes”. In: *Topological Codes*. Ed. by Daniel A. Lidar and Todd A. Brun. Cambridge University Press, 2013.
- [Bom14a] H Bombin. “Dimensional Jump in Quantum Error Correction”. In: *arXiv preprint arXiv:1412.5079* (2014).
- [Bom14b] Héctor Bombín. “Structure of 2D Topological Stabilizer Codes”. In: *Communications in Mathematical Physics* 327.2 (July 2014), pp. 387–432. doi: 10.1007/s00220-014-1893-4. arXiv: 1107.2707.
- [Bom15a] Hector Bombin. “Gauge color codes: Optimal transversal gates and gauge fixing in topological stabilizer codes”. In: *New Journal of Physics* 17 (2015). doi: 10.1088/1367-2630/17/8/083002.
- [Bom15b] Héctor Bombín. “Single-Shot Fault-Tolerant Quantum Error Correction”. In: *Physical Review X* 5 (2015), p. 031043. doi: 10.1103/PhysRevX.5.031043.
- [Bon+10] P. H. Bonderson et al. “A blueprint for a topologically fault-tolerant quantum computer”. In: (2010). arXiv: 1003.2856.
- [Bro+16] Benjamin J. Brown et al. “Quantum memories at finite temperature”. In: *Reviews of Modern Physics* 88.4 (Nov. 2016). doi: 10.1103/RevModPhys.88.045005. arXiv: arXiv:1411.6643v1.

- [BSW11] Salman Beigi, Peter W. Shor, and Daniel Whalen. “The Quantum Double Model with Boundary: Condensations and Symmetries”. In: *Commun. Math. Phys.* 306.3 (2011), pp. 663–694. doi: 10.1007/s00220-011-1294-x.
- [BT15] Nikolas P. Breuckmann and Barbara M. Terhal. “Constructions and Noise Threshold of Hyperbolic Surface Codes”. In: (June 2015). arXiv: 1506.04029.
- [BY86] K. Binder and A. P. Young. “Spin glasses: Experimental facts, theoretical concepts, and open questions”. In: *Reviews of Modern Physics* 58 (1986), pp. 801–976. doi: 10.1103/RevModPhys.58.801.
- [Cal+97] A. Calderbank et al. “Quantum Error Correction and Orthogonal Geometry”. In: *Physical Review Letters* 78 (1997), pp. 405–408. doi: 10.1103/PhysRevLett.78.405.
- [CGW10] Xie Chen, Zheng-Cheng Gu, and Xiao-Gang Wen. “Local unitary transformation, long-range quantum entanglement, wave function renormalization, and topological order”. In: *Phys. Rev. B* 82 (26, 2010), p. 155138.
- [Chu36] Alonzo Church. “An Unsolvable Problem of Elementary Number Theory”. In: *American Journal of Mathematics* 58.2 (1936), p. 345. doi: 10.2307/2371045. arXiv: arXiv:1011.1669v3.
- [CJL16] Christopher Chamberland, Tomas Jochym-O’Connor, and Raymond Laflamme. “Thresholds for Universal Concatenated Quantum Codes”. In: *Phys. Rev. Lett.* 117 (1 June 2016), p. 010501. doi: 10.1103/PhysRevLett.117.010501.
- [CJR79] Michael Creutz, Laurence Jacobs, and Claudio Rebbi. “Monte Carlo study of Abelian lattice gauge theories”. In: *Physical Review D* 20 (1979), pp. 1915–1922. doi: 10.1103/PhysRevD.20.1915.
- [CLB86] Murty S. S. Challa, D. P. Landau, and K. Binder. “Finite-size effects at temperature-driven first-order transitions”. In: *Physical Review B* 34 (1986), pp. 1841–1852. doi: 10.1103/PhysRevB.34.1841.
- [CO16] Earl T Campbell and Joe O’Gorman. “An efficient magic state approach to small angle rotations”. In: *Quantum Science and Technology* 1.1 (2016), p. 015007.
- [CS96] A. Calderbank and Peter Shor. In: *Physical Review A* 54.2 (1996), pp. 1098–1105. doi: 10.1103/PhysRevA.54.1098.
- [DBT17] Kasper Duivendoorn, Nikolas P. Breuckmann, and Barbara M. Terhal. “Renormalization group decoder for a four-dimensional toric code”. In: (Aug. 2017). arXiv: 1708.09286.
- [Del14a] Nicolas Delfosse. “Decoding color codes by projection onto surface codes”. In: *Phys. Rev. A* 89 (1 Jan. 2014), p. 012317.

- [Del14b] Nicolas Delfosse. “Decoding color codes by projection onto surface codes”. In: *Physical Review A* 89 (2014), p. 012317. doi: 10.1103/PhysRevA.89.012317.
- [Den+02] Eric Dennis et al. “Topological quantum memory”. In: *Journal of Mathematical Physics* 43 (2002), p. 4452. doi: 10.1063/1.1499754.
- [Deu85] D. Deutsch. “Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer”. In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 400.1818 (1985), pp. 97–117. doi: 10.1098/rspa.1985.0070. arXiv: 0407008 [arXiv:quant-ph].
- [Deu89] D. Deutsch. “Quantum Computational Networks”. In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 425.1868 (1989), pp. 73–90. doi: 10.1098/rspa.1989.0099. arXiv: 1108.0910.
- [DP02] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2002. doi: 10.1017/CB09781107415324.004. arXiv: arXiv:1011.1669v3.
- [DP13] Guillaume Duclos-Cianci and David Poulin. “Kitaev’s Z_d -code threshold estimates”. In: *Physical Review A* 87.6 (June 2013), p. 062338. doi: 10.1103/PhysRevA.87.062338. arXiv: 1302.3638.
- [DP15] Guillaume Duclos-Cianci and David Poulin. “Reducing the quantum-computing overhead with complex gate distillation”. In: *Physical Review A* 91.4 (2015), p. 042315.
- [DS13] M H Devoret and R J Schoelkopf. “Superconducting circuits for quantum information: an outlook.” In: *Science* 339 (2013), pp. 1169–74. doi: 10.1126/science.1231930.
- [EK09] B Eastin and E Knill. “Restrictions on transversal encoded quantum gate sets”. In: *Phys. Rev. Lett.* 102 (2009), p. 110502. doi: 10.1103/PhysRevLett.102.110502.
- [Eli75] S. Elitzur. “Impossibility of spontaneously breaking local symmetries”. In: *Physical Review D* 12 (1975), pp. 3978–3982. doi: 10.1103/PhysRevD.12.3978.
- [Far+00] Edward Farhi et al. “Quantum Computation by Adiabatic Evolution”. In: (Jan. 2000). doi: quant-ph/0001106. arXiv: 0001106 [quant-ph].
- [Fey82] RP Feynman. “Simulating physics with computers”. In: *International journal of theoretical physics* (1982).
- [FM01] Michael H. Freedman and David A. Meyer. “Projective plane and planar quantum codes”. In: *Found. Comp. Math.* 1 (2001), pp. 325–332.

- [Fow+12] Austin G. Fowler et al. “Surface codes: Towards practical large-scale quantum computation”. In: *Physical Review A* 86 (2012), p. 032324. DOI: 10.1103/PhysRevA.86.032324.
- [Fre+03] M Freedman et al. “Topological quantum computation”. In: *Bulletin of the American* (2003).
- [Gác90] Peter Gács. “A Toom rule that increases the thickness of sets”. In: *Journal of Statistical Physics* 59.1-2 (Feb. 1990), pp. 171–193. DOI: 10.1007/BF01015567. arXiv: 0102009 [math-ph].
- [GC99] Daniel Gottesman and Isaac L. Chuang. “Quantum Teleportation is a Universal Computational Primitive”. In: *Nature* 402 (1999), pp. 390–393. DOI: 10.1038/46503.
- [Gil52] Edgar N Gilbert. In: *Bell Labs Technical Journal* 31.3 (1952), pp. 504–522.
- [Gla72] Leslie C. Glaser. *Geometric combinatorial topology*. New York: Van Nostrand Reinhold Company, 1972.
- [Gol92] N Goldenfeld. *Lectures on phase transitions and critical phenomena*. Addison-Wesley Publishing Company, 1992.
- [Got96] Daniel Gottesman. “Class of quantum error-correcting codes saturating the quantum Hamming bound”. In: *Physical Review A* 54 (1996), pp. 1862–1868. DOI: 10.1103/PhysRevA.54.1862.
- [Got97] Daniel Gottesman. “Stabilizer Codes and Quantum Error Correction”. In: (1997). arXiv: quant-ph/9705052.
- [Got98] Daniel Gottesman. “The Heisenberg Representation of Quantum Computers”. In: (1998). arXiv: quant-ph/9807006.
- [GR88] Peter Gács and John Reif. “A simple three-dimensional real-time reliable cellular array”. In: *Journal of Computer and System Sciences* 36.2 (Apr. 1988), pp. 125–147. DOI: 10.1016/0022-0000(88)90024-4.
- [Gra01] Lawrence F. Gray. “A reader’s guide to Gacs’s “positive rates” paper”. In: *Journal of Statistical Physics* 103.1-2 (2001), pp. 1–44. DOI: 10.1023/A:1004824203467.
- [Gri04] G. Grinstein. “Can complex structures be generically stable in a noisy world?” In: *IBM Journal of Research and Development* 48.1 (Jan. 2004), pp. 5–12. DOI: 10.1147/rd.481.0005.
- [Gri99] Geoffrey Grimmett. *Percolation*. Springer-Verlag, 1999.
- [Haa+17] Jeongwan Haah et al. “Magic State Distillation with Low Space Overhead and Optimal Asymptotic Input Count”. In: (2017). arXiv: quant-ph/1703.07847.

- [Haa11] J. Haah. “Local stabilizer codes in three dimensions without string logical operators”. In: *Phys. Rev. A* 83 (2011), p. 042330.
- [Har04] Jim Harrington. “Analysis of quantum error-correcting codes: symplectic lattice codes and toric codes”. PhD thesis. California Institute of Technology, 2004.
- [Has+07] Martin Hasenbusch et al. “Magnetic-glassy multicritical behavior of the three-dimensional $\pm J$ Ising model”. In: *Physical Review B* 76 (2007), p. 184202. doi: 10.1103/PhysRevB.76.184202.
- [Hat02] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.
- [HN96] Koji Hukushima and Koji Nemoto. “Exchange Monte Carlo Method and Application to Spin Glass Simulations”. In: *Journal of the Physical Society of Japan* 65 (1996), pp. 1604–1608. doi: 10.1143/JPSJ.65.1604.
- [Hor+12] Clare Horsman et al. “Surface code quantum computing by lattice surgery”. In: *New. J. Phys.* 14.12 (2012), p. 123011.
- [HPP01] A. Honecker, M. Picco, and P. Pujol. “Nishimori point in the 2D +/- J random-bond Ising model”. In: *Phys. Rev. Lett.* 87 (2001), p. 047201. doi: 10.1103/PhysRevLett.87.047201.
- [HW05] M. B. Hastings and Xiao-Gang Wen. “Quasiadiabatic continuation of quantum states: The stability of topological ground-state degeneracy and emergent gauge invariance”. In: *Phys. Rev. B* 72 (25, 2005), p. 045141.
- [IP15] Pavithran Iyer and David Poulin. “Hardness of decoding quantum stabilizer codes”. In: *IEEE Transactions on Information Theory* 61.9 (2015), pp. 5209–5223. doi: 10.1109/TIT.2015.2422294. arXiv: 1310.3235.
- [JKY18] Tomas Jochym-O’Connor, Aleksander Kubica, and Theodore J. Yoder. “Disjointness of Stabilizer Codes and Limitations on Fault-Tolerant Logical Gates”. In: *Physical Review X* 8.2 (May 2018), p. 21047. doi: 10.1103/PhysRevX.8.021047.
- [JL14] Tomas Jochym-O’Connor and Raymond Laflamme. “Using Concatenated Quantum Codes for Universal Fault-Tolerant Quantum Gates”. In: *Phys. Rev. Lett.* 112 (2014), p. 010505. doi: 10.1103/PhysRevLett.112.010505.
- [JLP11] Stephen P. Jordan, Keith S. M. Lee, and John Preskill. “Quantum Algorithms for Quantum Field Theories”. In: *Science* 336.6085 (Nov. 2011), pp. 1130–1133. doi: 10.1126/science.1217069. arXiv: 1111.3633.
- [Jon13] Cody Jones. “Multilevel distillation of magic states for quantum computing”. In: *Physical Review A* 87.4 (2013), p. 042305.

- [Kar07] M Kardar. *Statistical physics of fields*. Cambridge University Press, 2007.
- [Kat+06] Helmut G. Katzgraber et al. “Feedback-optimized parallel tempering Monte Carlo”. In: *Journal of Statistical Mechanics: Theory and Experiment* 2006 (2006), p. 12. DOI: 10.1088/1742-5468/2006/03/P03018.
- [KB15] Aleksander Kubica and Michael Beverland. “Universal transversal gates with color codes: A simplified approach”. In: *Phys. Rev. A* 91 (2015), p. 032330. DOI: 10.1103/PhysRevA.91.032330.
- [KBM09] Helmut G. Katzgraber, H. Bombin, and M. A. Martin-Delgado. “Error Threshold for Color Codes and Random Three-Body Ising Models”. In: *Physical Review Letters* 103 (2009), p. 090501. DOI: 10.1103/PhysRevLett.103.090501.
- [KDP18] Aleksander Kubica, Nicolas Delfosse, and John Preskill. “Local decoders for topological toric and color codes in any dimensions”. In: *in preparation* (2018).
- [Kit01] A Yu Kitaev. “Unpaired Majorana fermions in quantum wires”. In: *Physics-Uspekhi* 44.10S (Oct. 2001), pp. 131–136. DOI: 10.1070/1063-7869/44/10S/S29. arXiv: 0010440 [cond-mat].
- [Kit03] A Y Kitaev. “Fault-tolerant quantum computation by anyons”. In: *Annals Phys.* 303 (2003), pp. 2–30. DOI: 10.1016/S0003-4916(02)00018-0.
- [Kit06] Alexei Kitaev. “Anyons in an exactly solved model and beyond”. In: *Annals of Physics* (2006). arXiv: 0506438v3 [arXiv:cond-mat].
- [Kit97] A Yu Kitaev. “Quantum computations: algorithms and error correction”. In: *Russian Mathematical Surveys* 52.6 (1997), p. 1191.
- [KK12] Alexei Kitaev and Liang Kong. “Models for Gapped Boundaries and Domain Walls”. In: *Commun. Math. Phys.* 313.2 (2012), pp. 351–373. DOI: 10.1007/s00220-012-1500-5.
- [KLZ98] Emanuel Knill, Raymond Laflamme, and Wojciech H. Zurek. “Resilient quantum computation: error models and thresholds”. In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 454.1969 (Feb. 1998), pp. 365–384. DOI: 10.1098/rspa.1998.0166. arXiv: 9702058 [quant-ph].
- [Kni04a] E. Knill. “Fault-Tolerant Postselected Quantum Computation: Schemes”. In: *arXiv preprint arXiv:0402171* (2004).
- [Kni04b] E. Knill. “Fault-Tolerant Postselected Quantum Computation: Threshold Analysis”. In: *arXiv preprint arXiv:0404104* (2004).

- [Kog79] JB Kogut. “An introduction to lattice gauge theory and spin systems”. In: *Reviews of Modern Physics* (1979).
- [KP15] Alexey A. Kovalev and Leonid P. Pryadko. “Spin glass reflection of the decoding transition for quantum error correcting codes”. In: *Quantum Information and Computation* 15 (2015), p. 0825.
- [Kub+18] Aleksander Kubica et al. “Three-Dimensional Color Code Thresholds via Statistical-Mechanical Mapping”. In: *Phys. Rev. Lett.* 120 (2018), p. 180501. DOI: 10.1103/PhysRevLett.120.180501.
- [KY14] Aleksander Kubica and Beni Yoshida. “Precise estimation of critical exponents from real-space renormalization group analysis”. In: *arXiv preprint* (Feb. 2014), p. 8. arXiv: 1402.0619.
- [KY18] Aleksander Kubica and Beni Yoshida. “Ungauging quantum error-correcting codes”. In: (May 2018), pp. 1–33. arXiv: arXiv:1805.01836v1.
- [KYP15] Aleksander Kubica, Beni Yoshida, and Fernando Pastawski. “Unfolding the color code”. In: *New J. Phys.* 17 (2015), p. 083026. DOI: 10.1088/1367-2630/17/8/083026.
- [LAR11] Andrew J Landahl, Jonas T Anderson, and Patrick R Rice. “Fault-tolerant quantum computing with color codes”. In: *arXiv preprint arXiv:1108.5738* (2011).
- [LC13] Andrew J. Landahl and Chris Cesare. “Complex instruction set computing architecture for performing accurate quantum Z rotations with less magic”. In: (Nov. 2013). arXiv:1302.3240.
- [Lev13] Michael Levin. “Protected Edge Modes without Symmetry”. In: *Phys. Rev. X* 3 (2 May 2013), p. 021009. DOI: 10.1103/PhysRevX.3.021009.
- [LG12] Michael Levin and Zheng-Cheng Gu. “Braiding statistics approach to symmetry-protected topological phases”. In: *Phys. Rev. B* 86.11 (Sept. 10, 2012), pp. 115109–.
- [LK91] Jooyoung Lee and J. M. Kosterlitz. “Finite-size scaling and Monte Carlo simulations of first-order phase transitions”. In: *Physical Review B* 43 (1991), pp. 3265–3277. DOI: 10.1103/PhysRevB.43.3265.
- [LR14] Andrew J. Landahl and Ciaran Ryan-Anderson. arXiv:1407.5103. 2014.
- [LW05] Michael Levin and Xiao-Gang Wen. “String-net condensation: A physical mechanism for topological phases”. In: *Physical Review B* 71 (2005), p. 045110. DOI: 10.1103/PhysRevB.71.045110.
- [LWW15] T. Lan, J. Wang, and X. G. Wen. In: *Phys. Rev. Lett.* 114, 076402 (2015).

- [MEK13] Adam M Meier, Bryan Eastin, and Emanuel Knill. “Magic-state distillation with the four-qubit code”. In: *Quantum Information and Computation* 13.3&4 (2013), pp. 0195–0209.
- [Mic14] Kamil P Michnicki. “3-d topological quantum memory with a power-law energy barrier”. In: *Phys. Rev. Lett.* 113 (2014), p. 130501. doi: 10.1103/PhysRevLett.113.130501.
- [MKJ18] Nishad Maskara, Aleksander Kubica, and Tomas Jochym-O’Connor. “Advantages of versatile neural-network decoding for topological codes”. In: (Feb. 2018). arXiv: 1802.08680.
- [MS77] FJ MacWilliams and NJA Sloane. *The theory of error-correcting codes*. North-Holland, 1977.
- [Nay+08] Chetan Nayak et al. “Non-Abelian anyons and topological quantum computation”. In: *Rev. Mod. Phys.* 80 (3 Sept. 2008), pp. 1083–1159. doi: 10.1103/RevModPhys.80.1083.
- [NC10] MA Nielsen and IL Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2010.
- [Nig+14] Daniel Nigg et al. “Experimental Quantum Computations on a Topologically Encoded Qubit”. In: (2014). arXiv: 1403.5426.
- [Ohn+04] Takuya Ohno et al. “Phase structure of the random-plaquette Z2 gauge model: Accuracy threshold for a toric quantum memory”. In: *Nuclear Physics B* 697 (2004), pp. 462–480. doi: 10.1016/j.nuclphysb.2004.07.003.
- [OI98] Yukiyasu Ozeki and Nobuyasu Ito. “Multicritical dynamics for the $\pm J$ Ising Model”. In: *Journal of Physics A: Mathematical and General* 31 (1998), p. 5451. doi: 10.1088/0305-4470/31/24/007.
- [Pas+15] Fernando Pastawski et al. “Holographic quantum error-correcting codes: toy models for the bulk/boundary correspondence”. In: *Journal of High Energy Physics* 2015.6 (Mar. 2015), p. 149. doi: 10.1007/JHEP06(2015)149. arXiv: 1503.06237.
- [PC99] Matteo Palassini and Sergio Caracciolo. “Universal Finite-Size Scaling Functions in the 3D Ising Spin Glass”. In: *Physical Review Letters* 82 (1999), pp. 5128–5131. doi: 10.1103/PhysRevLett.82.5128.
- [PCC11] Fernando Pastawski, Lucas Clemente, and Juan Ignacio Cirac. “Quantum memories based on engineered dissipation”. In: *Physical Review A - Atomic, Molecular, and Optical Physics* 83.1 (Oct. 2011). doi: 10.1103/PhysRevA.83.012304. arXiv: 1010.2901.
- [Pou05] David Poulin. “Stabilizer Formalism for Operator Quantum Error Correction”. In: *Phys. Rev. Lett.* 95 (2005), p. 230504. doi: 10.1103/PhysRevLett.95.230504.

- [PR13] Adam Paetznick and Ben W. Reichardt. “Universal Fault-Tolerant Quantum Computation with Only Transversal Gates and Error Correction”. In: *Phys. Rev. Lett.* 111 (2013), p. 090505. doi: 10.1103/PhysRevLett.111.090505.
- [Pre98] J. Preskill. “Reliable quantum computers”. In: *Proc. Roy. Soc. Lond.* 454 (1998), pp. 385–410. doi: 10.1098/rspa.1998.0167.
- [Pre99] John Preskill. *Lecture notes for Physics 219: Quantum computation*. 1999.
- [PY15] Fernando Pastawski and Beni Yoshida. “Fault-tolerant logical gates in quantum error-correcting codes”. In: *Phys. Rev. A* 91.1 (Jan. 8, 2015), pp. 012305–.
- [RB01] Robert Raussendorf and Hans J. Briegel. “A One-Way Quantum Computer”. In: *Physical Review Letters* 86.22 (May 2001), pp. 5188–5191. doi: 10.1103/PhysRevLett.86.5188.
- [Rei05] Ben W Reichardt. “Quantum universality from Magic States Distillation applied to CSS codes”. In: *Quantum Information Processing* 4.3 (2005), pp. 251–264.
- [Sac99] S. Sachdev. *Quantum Phase Transitions*. Cambridge University Press, Cambridge, 1999.
- [Sha01] C. E. Shannon. “A mathematical theory of communication”. In: *ACM SIGMOBILE Mobile Computing and Communications Review* 5.1 (Jan. 2001), p. 3. doi: 10.1145/584091.584093. arXiv: 9411012.
- [Sho94] P.W. Shor. “Algorithms for quantum computation: discrete logarithms and factoring”. In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. IEEE Comput. Soc. Press, 1994, pp. 124–134. doi: 10.1109/SFCS.1994.365700. arXiv: 9605043.
- [Sho95] Peter W. Shor. “Scheme for reducing decoherence in quantum computer memory”. In: *Physical Review A* 52.4 (Oct. 1995), R2493–R2496. doi: 10.1103/PhysRevA.52.R2493. arXiv: 0506097.
- [Sho96] P.W. Shor. “Fault-tolerant quantum computation”. In: *Proceedings of 37th Conference on Foundations of Computer Science*. IEEE Comput. Soc. Press, 1996, pp. 56–65. doi: 10.1109/SFCS.1996.548464.
- [Sho99] Peter W. Shor. “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”. In: *SIAM Review* 41.2 (Jan. 1999), pp. 303–332. doi: 10.1137/S0097539795293172. arXiv: 9508027 [quant-ph].
- [SR12] Pradeep Sarvepalli and Robert Raussendorf. “Efficient decoding of topological color codes”. In: *Physical Review A - Atomic, Molecular, and Optical Physics* 85.2 (Nov. 2012). doi: 10.1103/PhysRevA.85.022317. arXiv: arXiv:1402.3037v1.

- [Ste14] Ashley M Stephens. “Efficient fault-tolerant decoding of topological color codes”. In: *arXiv preprint arXiv:1402.3037* (2014).
- [Ste96a] A. Steane. “Multiple-Particle Interference and Quantum Error Correction”. In: *Proc. Roy. Soc. Lond.* 452 (1996), pp. 2551–2577. DOI: 10.1098/rspa.1996.0136.
- [Ste96b] A. Steane. “Multiple-Particle Interference and Quantum Error Correction”. In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 452.1954 (Nov. 1996), pp. 2551–2577. DOI: 10.1098/rspa.1996.0136. arXiv: 9601029 [quant-ph].
- [Ste99] Andrew Steane. “Quantum Reed-Muller Codes”. In: *IEEE Trans. Info. Th.* 45 (1999), pp. 1701–1703. DOI: 10.1109/18.771249.
- [Sul90] John Matthew Sullivan. “A crystalline approximation theorem for hypersurfaces”. PhD thesis. Princeton University, 1990.
- [Ter15] Barbara M Terhal. “Quantum Error Correction for Quantum Memories”. In: *Rev. Mod. Phys.* 87 (2015), p. 307. arXiv: arXiv:1302.3428v1.
- [Too80] A L Toom. “Stable and Attractive Trajectories in Multicomponent Systems”. In: *Multicomponent Systems*. Vol. 6. 1980, pp. 549–575.
- [Tur36] AM Turing. “On computable numbers, with an application to the Entscheidungsproblem”. In: *Proceedings of the London mathematical society* (1936).
- [Var57] RR Varshamov. In: *Dokl. Akad. Nauk SSSR*. Vol. 117. 5. 1957, pp. 739–741.
- [Wan+10] David S. Wang et al. “Graphical algorithms and threshold error rates for the 2d colour code”. In: *Quantum Information and Computation* 10 (July 2010), p. 780. arXiv: 0907.1708.
- [Wec+14] Dave Wecker et al. “Gate-count estimates for performing quantum chemistry on small quantum computers”. In: *Phys. Rev. A* 90 (2014), p. 022305. DOI: 10.1103/PhysRevA.90.022305.
- [Weg71] Franz J. Wegner. “Duality in Generalized Ising Models and Phase Transitions without Local Order Parameters”. In: *Journal of Mathematical Physics* 12 (1971), p. 2259. DOI: 10.1063/1.1665530.
- [Wen90] X. G. Wen. “Topological orders in rigid states”. In: *International Journal of Modern Physics B* 04.02 (Feb. 1990), pp. 239–271. DOI: 10.1142/S0217979290000139. arXiv: arXiv:1011.1669v3.
- [WHP03] Chenyang Wang, Jim Harrington, and John Preskill. “Confinement-Higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory”. In: *Annals of Physics* 303 (2003), pp. 31–58. DOI: 10.1016/S0003-4916(02)00019-2.

- [Wil74] Kenneth G. Wilson. “Confinement of quarks”. In: *Physical Review D* 10 (1974), pp. 2445–2459. DOI: 10.1103/PhysRevD.10.2445.
- [Wil82] Frank Wilczek. “Quantum mechanics of fractional-spin particles”. In: *Physical Review Letters* 49.14 (Oct. 1982), pp. 957–959. DOI: 10.1103/PhysRevLett.49.957.
- [Wil96] Robin J. Wilson. *Introduction to Graph Theory*. Longman, 1996.
- [WZ82] WK Wootters and WH Zurek. “A single quantum cannot be cloned”. In: *Nature* (1982).
- [YC10] Beni Yoshida and Isaac L. Chuang. “Framework for classifying logical operators in stabilizer codes”. In: *Phys. Rev. A* 81 (4, 2010), p. 052302.
- [YK14] Beni Yoshida and Aleksander Kubica. “Quantum criticality from Ising model on fractal lattices”. In: *arXiv preprint* (Apr. 2014). arXiv: arXiv:1404.6311.
- [Yos11] Beni Yoshida. “Classification of quantum phases and topology of logical operators in an exactly solved model of quantum codes”. In: *Annals of Physics* 326.1 (Jan. 2011), pp. 15–95. DOI: 10.1016/j.aop.2010.10.009. arXiv: 1007.4601.
- [Yos13] Beni Yoshida. “Exotic topological order in fractal spin liquids”. In: *Phys. Rev. B* 88.12 (Sept. 12, 2013), pp. 125122–.
- [ZCC11] Bei Zeng, Andrew Cross, and Isaac L. Chuang. “Transversality versus universality for additive quantum codes”. In: *IEEE Transactions on Information Theory* 57 (2011), pp. 6272–6284. DOI: 10.1109/TIT.2011.2161917.