# A REVIEW ON QUANTUM ERROR CORRECTION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ÇETIN İLHAN KAYA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
PHYSICS

DECEMBER 2023

Approval of the thesis:

**A REVIEW ON QUANTUM ERROR CORRECTION**

submitted by **ÇETIN İLHAN KAYA** in partial fulfillment of the requirements for the degree of **Master of Science in Physics Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Seçkin Kürkçüoğlu
Head of Department, **Physics** _____

Assoc. Prof. Dr. Yusuf İpekoğlu
Supervisor, **Physics, METU** _____

**Examining Committee Members:**

Prof. Dr. Mehmet Emre Taşgın
Institute of Nuclear Sciences, Hacettepe University _____

Assoc. Prof. Dr. Yusuf İpekoğlu
Physics, METU _____

Prof. Dr. Sadi Turgut
Physics, METU _____

Date:06.12.2023

**I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.**

Name, Surname: Çetin İlhan Kaya

Signature :

## ABSTRACT

### A REVIEW ON QUANTUM ERROR CORRECTION

Kaya, Çetin İlhan
M.S., Department of Physics
Supervisor: Assoc. Prof. Dr. Yusuf İpekoğlu

December 2023, 204 pages

In this thesis, we review the quantum error correction codes which are essential for quantum information processing and quantum telecommunication systems to work fault-tolerantly. The concern of this study is to produce a catalog containing recipies for quantum error correction codes. Physical realizations of given codes are not concern of this work. We mostly concentrate on stabilizer codes which are the most studied family of quantum error correction codes. Moreover codes that might serve as quantum memories and methods to optimize quantum error correction codes using machine learning techniques are also discussed. For quantum memory candidate Haah cubic codes, we provide better equations describing commutation relations of the corner operators. Note that here we only consider QECCs useful for quantum industries.



Keywords: quantum error correction, quantum information theory, quantum computation, quantum information processing

# ÖZ

## KUANTUM HATA DÜZELTME ÜZERİNE DERLEME

Kaya, Çetin İlhan
Yüksek Lisans, Fizik Bölümü
Tez Yöneticisi: Doç. Dr. Yusuf İpekoğlu

Aralık 2023 , 204 sayfa

Bu tez, kuantum bilgi işleme ve kuantum telekomünikasyon sistemlerinin hataya dayanikli çalışmasını sağlamakta önemli rol oynayan kuantum hata düzeltme kodlarının derlemesidir. Bu çalışmanın hedefi, kuantum hata düzeltme kodlarının, tarifleriyle birlikte kataloglanmasıdır. Tez boyunca bu kodların fiziksel olarak hayata geçirilmelerinden bahsedilmeyecektir. Çoğunlukla, kuantum hata düzeltme kodlarının en iyi çalışılmış ailesi olan stabilizer kodlar üzerinde durulmuştur. Ayrıca quantum hafıza görevi görebilecek quantum hata düzeltme kodlarına ve quantum hata düzeltme kodlarının iyileştirilmesinde makine öğrenmesi kullanılan yöntemlere de değinilmiştir. Bir kuantum hafıza adayı olan Haah kübik kodlarda kullanılmak üzere, köşe operatörlerin bazı ilişkileri daha düzgün şekilde sunulmuştur. Bu çalışmada sadece, kuantum endüstrisinde kullanılmak üzere işlevsel kuantum hata düzeltme kodları incelenmiştir.

Anahtar Kelimeler: kuantum hata düzeltme, kuantum bilgi kuramı, kuantum hesaplama, kuantum bilgi işleme

This study is dedicated to my father Mehmet Kaya and my mother Sultan Uçar, whose unwavering love and encouragement have been the driving force behind my academic achievements and will continue to be my guiding light as I pursue my lifelong aspirations.

## ACKNOWLEDGMENTS

I acknowledge helpful discussions with Yusuf İpekoğlu, Osman Barış Malcıoğlu, Muhteşem Akif Korkmaz and Barış Yaradanakul. And also it is an obligation to acknowledge those from Max Planck to Anton Zeilinger for their contributions to quantum research which allow us to study such exciting topics.

## TABLE OF CONTENTS

# LIST OF TABLES

TABLES

## LIST OF FIGURES

FIGURES

## LIST OF ABBREVIATIONS

| | |
|---|---|
| 1D | 1 Dimensional |
| 2D | 2 Dimensional |
| 3D | 3 Dimensional |
| 4D | 4 Dimensional |
| CSS | Calderbank-Shor-Steane |
| QEC | Quantum Error Correction |
| QECC | Quantum Error Correction Code |
| AQEC | Autonomous Quantum Error Correction |
| AQECC | Autonomous Quantum Error Correction Code |
| AAQEC | Approximate Autonomous Quantum Error Correction |
| AAQECC | Approximate Autonomous Quantum Error Correction Code |
| QED | Quantum Error Detection |
| QC | Quantum Computing or Quantum Computer |
| PCM | Parity Check Matrix |
| CNOT | Controlled-Not |
| SPC | Single Parity Check |
| LOQC | Linear Optics Quantum Computing |
| CS | Controlled Phase-Flip |
| Prep | Preparation |
| QED | Quantum Error Detection |
| MAP | Maximum-a-Posteriori |
| LC | Local Clifford |
| LU | Local Unitary |
| MSC | Minimal Support Condition |

| | |
|---|---|
| GHZ | Greenberger-Horne-Zeilinger |
| RM | Reed-Muller |
| CCZ | Control-Control-Z |
| GF | Galois Field |
| CP | Controlled Phase |
| CCX | Control-Control-X (Toffoli) |
| MWM | Minimal Weight Matching |
| LDPC | Low Density Parity Check |
| QLDPC | Quantum Low Density Parity Check |
| QLDPCC | Quantum Low Density Parity Check Code |
| KLM | Knill-Laflamme-Milburn |
| RHS | Right Hand Side |
| LHS | Left Hand Side |
| CW | Clock Wise |
| CCW | Counter Clock Wise |
| SCQM | Self-Correcting Quantum Memory |
| TPCP | Trace-Preserving Completely Positive |
| PBC | Periodic Boundary Condition |
| ECLO | Error-Corrected Logical Operatpr |
| NISQ | Noisy Intermediate-Scale Quantum |
| VQC | Variational Quantum Circuit |
| VarQEC | Variational Quantum Error Correction |
| RL | Reinforcement Learning |
| PSM | Projective Simulation Model |
| SQUAB | Surface Quantum Architectures Benchmarking |
| PPO | Proximal Policy Optimization |
| ML | Machine Learning |

# CHAPTER 1

# INTRODUCTION

After discovery of classical information theory and computation, computer science provided humanity with various opportunities that has never been available before. Nearly 90 years passed since the *lambda calculus* [36] and in the meantime, computer parts got smaller and smaller. Transistor is the component where computation is carried by allowing the passage of current (state 1) or blocking it (state 0) depending on the base voltage and this is where 1s and 0s come from. Modern transistors are in the size of nanometers and shrinking them more is problematic due to the quantum tunneling. If the size is sufficiently small, even if transistor wants to block the voltage, electrons may tunnel their way resulting in computation error. For this reason, classical computation is about to reach its physical limits.

At this point, new idea of quantum computation takes place which was first proposed by Deutch [37] in 1985. The idea of making computation using quantum effects is receiving more interest ever since. It was purely hyphothetical at the beginnings but today we have some levels of quantum computation. In 2023, there are many companies and institutes work on quantum processors including Google, Amazon, Microsoft, IBM and so on. So far, IBM's Osprey is the most advanced with 433 qubits [38].

As far as quantum computers are concerned, making qubits to stay in the desired state during the computation is very hard. Although recent developments in the experimental part allowed quantum information to be preserved to some point and increased the fidelity values higher, experimental techniques are not enough to preserve it completely. Thankfully, applying error correction protocols, we can take the fidelities even further and make computation robust against noise.

An *error correcting code* can be defined as a way of storing either classical or quantum information in a set of bits (or qubits) in such a way that we can read/process that information even if there has been some unknown changes (errors) in the bits (or qubits).[18] There are many quantum error correction codes proposed beginning with [6] in 1995, but in this thesis, mostly stabilizer codes, quantum error correction codes that might serve as quantum memories and machine learning assisted quantum error correction codes are considered. The reason is that the stabilizer codes are most well-studied quantum error correction codes and can be very useful in the NISQ (*Noisy Intermediate-Scale Quantum*) era, while the codes like *H codes* allow us to posses higher *yield* which is the ratio of logical qubits created to physical qubits used in the process which probably useful for long-term quantum processors with high numbers of physical qubits. On the other hand quantum memory codes have critical importance which are suppose to story quantum information for long times without needing active quantum error correction application. And finally we consider the possible schemes using machine learning in both classical and NISQ devices to optimize quantum error correction codes.

Note that, in this chapter, we mostly follow [9].

## 1.1 Motivation and Problem Definition

Since quantum states are hard to preserve due to random quantum flactuations (or quantum errors), it is essential to develop new techniques to overcome this issue. The quest of suppressing quantum errors is mostly done by experimental research but it is not enough. To carry efficiency even further, quantum error correction is the key. However, since the field is relatively new, the resources for quantum error correction researchers are limited. In this thesis, we try to cover some methods that might be useful for researchers.

## 1.2 Proposed Methods and Models

In this thesis, no new method or model is offered. Instead, we will study existing techniques and catalog them. Most of the techniques uses methods in classical error correction and adjust them according to the quantum needs.

## 1.3 Contributions and Novelties

In this thesis, equations **??**, **??**, **??** and **??** are provided for commutation relations of corner operators which are used in Haah cubic codes that might serve as quantum memory. In the case that quantum memories are realized by Haah cubic codes, those equations might be useful.

## 1.4 Classical Noise

Before we start discussing quantum error correcting codes, we should mention some concepts from classical information theory and classical error correction.

In binary systems for example, we have information on states called *bits* which can have either state 0 or 1 depending on the voltage on the system. Physically, for a given threshold, 0 means current flowing through the wire is below that threshold and 1 means the current is above the threshold. Suppose we have states $B_i$ and $B_f$ being initial and final states in binary ($\mathbb{F}_2$). Then law of probability yields: [9]

$$\mathcal{P}(B_f = k) = \sum_j \mathcal{P}(B_f = k|B_i = j)\mathcal{P}(B_i = j) \tag{1.1}$$

where $\mathcal{P}(B_f = k|B_i = j)$ is the probability of transition from one state to another due to some noise. This definition will be replaced with the *fidelity* for qubits later. For now, let $\mathcal{P}(B_f = k|B_i = j) = p$, then probability of a state preserving its previous state is $(1 - p)$. If we denote the probabilities of bits being in state $j = 0, 1$ before and after the noise as $\mathcal{P}_{i,j}$ and $\mathcal{P}_{f,j}$ where i and f stands for initial and final, we have:

$$\begin{bmatrix} \mathcal{P}_{f,0} \\ \mathcal{P}_{f,1} \end{bmatrix} = \begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix} \begin{bmatrix} \mathcal{P}_{i,0} \\ \mathcal{P}_{i,1} \end{bmatrix} \tag{1.2}$$

or

$$\mathcal{P}_{\mathbf{f}} = \mathcal{N}\mathcal{P}_{\mathbf{i}} \tag{1.3}$$

where $\mathcal{P}_{\mathbf{i}}$ and $\mathcal{P}_{\mathbf{f}}$ are initial and final probability matrices while $\mathcal{N}$ is the matrix holding the probabilities of experiencing noise and preserving the initial state.
Note that we can call such an error causing state 0 to state 1 and vice versa as *bit flip error* and classically it is represented with $NOT$ operator. In addition, a bit may go through *erasure channel* which deletes the information entirely.

## 1.5 Markovian Processes

We may simply define a Markovian process as the system in which bits have no memories. In such systems, each event is independent from the one happened before. As an example, we may consider a series of coin flips. Notice that each tossing has probability $50\%$ $Heads$ and $50\%$ $Tails$ outcome for fair coins and that probability is not changing for different histories.

For the discussions that will mention Markovian processes, consider the following chain of errors changing state $B_0$ to $B_3$:

$$B_0 \xrightarrow{NOT} B_1 \xrightarrow{NOT} B_2 \xrightarrow{NOT} B_3 \tag{1.4}$$

if occuring a bit-flip error in each step are independent from eachother, this process is Markovian. Note that here we are not interested in what information states $B_i$ carry.

## 1.6 Classical Linear Codes

We describe a linear code $C$ with an $n \times k$ generator matrix $G$ where $n$ is the number of bits in code space and $k$ is the bits of information which we want to encode. $G$ maps codewords to their encoded versions by for a $k$ bits of codeword $\mathbf{x}$, encodes as $G\mathbf{x}$. Note that $\mathbf{x}$ here is column vector and the elements of $G$ are all $1s$ and $0s$. [9]

As an example, think of the repetition code which maps 1 bit to 3 repetitions. Its generator matrix $G = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ and $G[0] = (0,0,0)$ & $G[1] = (1,1,1)$.

A code using $n$ bits to encode $k$ bits is called an $[n,k]$ code. To ensure the encoding of all messages to be unique, we need linearly independent columns in $G$ because the columns of $G$ span the codewords corresponding to the vector space.

- In a general code, we need $2^k$ codewords with lengths $n$ to encode $k$ bits in $n$ bits. To specify the description of the code in total, we need $n2^k$ bits.
- For a linear code, only necessity is $nk$ bits of generator matrix. We just multiply the message of $k$ bits by generator matrix of $n \times k$ to get encoded message which takes $O(nk)$ operations.

Instead of generator matrix, we can use parity-check matrix that is an $[n,k]$ code containing all vectors $\mathbf{x}$ over $\mathbb{Z}_2 \ni H\mathbf{x} = 0$. Here $H$ is called *parity-check matrix* (PCM) and has dimensions $(n-k) \times n$. Kernel of the PCM is defined as the code. Since there exists $2^k$ possible codewords for a code encoding $k$ bits, $dim(Ker(H)) = k$ and therefore $H$ must have linearly independent rows while $G$ needs linearly independent columns.
In order to move between $G$ and $H$:

1. $H \rightarrow G$: Choose linearly independent $k$ vectors $\omega_j$ (where $j \in \mathbb{Z}$ from 1 to $k$) that are expected to span $Ker(H)$. And make $\omega_1$ to $\omega_k$ be the columns of $G$.
2. $G \rightarrow H$: Choose linearly independent $n-k$ vectors $\omega_j$ (where $j \in \mathbb{Z}$ from 1 to $n-k$) and columns of G are orthogonal to these vectors. Then make from $\omega_1^T$ to $\omega_{n-k}^T$ be the rows of $H$.

Consider $[3,1]$ code $G = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ as an example. Here, we want to build $H$ and to do so, we will require linearly independent vectors (3-1=2 number of them). And columns

of G have to be orthogonal to those vectors. Let them be $(1,1,0)\,\&\,(0,1,1)$ and define

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

Notice that $x = (1,1,1)$ and $x = (0,0,0)$ are the only codewords satisfying $Hx = 0$

Suppose the message we have is $x$ and we encoded it as $\omega = Gx$. At some point, an error $\mathscr{E}$ occured shifting $\omega \to \omega' = \omega \oplus \mathscr{E}$. We know that $H\omega = 0\ \forall\omega$, then $H\omega' = H\mathscr{E}$. We call $H\omega'$, error syndrome. Though $H\omega'$ is a function of $\omega'$, since $H\omega' = H\mathscr{E}$, it contains information about the error.

Suppose we have codewords $v$ and $\omega$ with $n$ bits each. We define the Hamming distance $d(v,\omega)$ between codewords by the bits they differ. And we define weight of a word $v$ by how much the word and the string of zeros differ ($wt(v) \equiv d(v,0)$)
Note that $d(v,\omega) = wt(v \oplus \omega)$. Also:

$$d(C) \equiv \min_{v,\omega \in C\ni v \neq \omega} d(v,\omega) \tag{1.5}$$

Meaning that using two codewords of a code $C$ and their minimum distance, we find the distance of the code. Following that $C$ is linear and $v\&\omega$ are codewords, $v \oplus \omega$ also a codeword. One observation is:

$$d(C) = \min_{v \in C, v \neq 0} (wt(v)) \tag{1.6}$$

So, we set $d \equiv d(C)$, then $C$ is said to be an $[n,k,d]$ code.

Note that an $[n,k,d]$ code can correct up to $t$ bits of error if its distance is at least $2t+1$ where $t \in \mathbb{Z}$. We achieve this by decoding $\omega'$ (erronous encoded message) as $\omega$ (codeword such that $d(\omega,\omega') \leq t$).

As an example on Hamming codes, let $r \geq 2$ & $r \in \mathbb{Z}$ and $H$ is a matrix with $2^r - 1$ bit strings of length $r$ for all of its columns which are all different from 0. This type of PCM is called $[2^r - 1, 2^r - r - 1]$ linear code. We call such a code as Hamming

code. [9]

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{1.7}$$

$[7, 4]$ code with $r = 3$. Notice that no two columns are the same meaning that they are not linearly dependent though the first 3 columns are. So $d = 3$. This code can correct an error on any single bit.

## 1.7 The Quantum Hamming Bound

Consider a non-degenerate code capable of encoding $k$ logical qubits using $n$ physical qubits. Suppose it corrects errors on $t$ qubits or less and errors occured in $e$ qubits (where $e \leq t$). Notice that in $n$ places, there are $\binom{n}{e}$ possible places for errors to happen. 3 possible error types Pauli $X, Y, Z$ give us in total $3^e$ possible errors.

On a $t$ or fewer qubits, we can find the total error number with:[9]

$$\sum_{e=0}^{t} \binom{n}{e} 3^e \tag{1.8}$$

where $e = 0$ stands for $\mathbb{I}$ which means no errors. All of these errors must be corresponding to a $2^k$-dimensional orthogonal subspace to encode $k$ qubits as non-degenerate and the $2^n$-dimensional space available to $n$ qubits must be filled by those subspaces, resulting in the quantum Hamming bound:

$$\sum_{e=0}^{t} \binom{n}{e} 3^e 2^k \leq 2^n \tag{1.9}$$

Assume that we'll encode 1 logical qubit into $n$ physical qubits. We do this in a way so that it corrects an error occuring on 1 qubit. Then quantum Hamming bound becomes:

$$\sum_{e=0}^{t} \binom{n}{e} 3^e 2^k \leq 2^n \rightarrow \binom{n}{0} 2 + \binom{n}{1} 3 \times 2 = 2(1+3n) \leq 2^n$$
$$\therefore n \geq 5 \tag{1.10}$$

Implying that the minimum required number of physical qubits for a protected 1 logical qubit is 5.

### 1.8 Gilbert-Varshamov bound

This bound suggests that we can find an $[n, k]$ error correction code where $n$ is large and code can protect $k$ bits of information from $t$ errors satisfying the following inequality:

$$\frac{k}{n} \geq 1 - \mathtt{H}(\frac{2t}{n}) \tag{1.11}$$

where

$$\mathtt{H}(x) \equiv -x \log x - (1-x) \log(1-x) \tag{1.12}$$

which is the *binary Shannon entropy*.
This is the condition that giving us the information whether the particular code parameter exists. In other words, it guarantees the existence of good codes so that we know how much information can be encoded into given $n$. [9]

### 1.9 the Dual construction

Consider a generator matrix $G$ and a PCM $H$ that construct an $[n, k]$ code $C$. Then we can define the dual $C^{\perp}$ of the code $C$ as follows: it has the generator matrix $H^T$ and PCM $G^T$, and contains all the codewords that are orthogonal to all of the codewords in $C$. And we say:[9]

- "weakly self-dual" to a code if $C \subseteq C^{\perp}$
- "strictly self-dual" to a code if $C = C^{\perp}$

## 1.10 System-Environment Interactions and Master equation

Assume we have a system $\rho$ interacting with its environment. Such interaction can be described as $\rho \otimes \rho_{env}$.

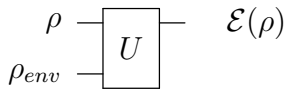



Figure 1.1: system-environment interaction [9]

After the action of unitary operator $U$ is complete, system and environment don't interact with each other anymore (which allows us to take partial trace). And at the end we have $\mathcal{E}(\rho)$ which represents the quantum operation on state $\rho$. Then reduced state can be written as: [9]

$$\mathcal{E}(\rho) = \mathrm{Tr}_{env}\left[U(\rho \otimes \rho_{env})U^{\dagger}\right] \tag{1.13}$$

Mostly, we assume that $\rho_{env} = |0\rangle\langle 0|$, but in chapter 3, we discuss QECCs that may serve as quantum memories in which system-environment interactions are important for real life solutions. We demand such systems to be passively protected against the thermal noise coming from environment.

To discuss such systems, we also need a description of noise for continuous time and non-unitary nature. For this end, we use *Lindbladian master equation* such as: [9][48]

$$\frac{d\rho}{dt} = -\frac{i}{\hbar}[H,\rho] + \sum_{j}(2L_j\rho L_j^{\dagger} - \{L_j^{\dagger}L_j,\rho\}) \tag{1.14}$$

where $L_j$'s are called *Lindblad operators* and they describe the coupling between system and the reservoir.

In general, for practical purposes, we begin with $\rho \otimes \rho_{env}$ state. A Hamiltonian containing system-environment coupling is followed by determining $L_j$'s via Born-Markov approximations. Note that as far as master equation is concerned, it is always $\mathrm{Tr}[\rho(t)] = 1$.

## 1.11 Quantum Noises

In this section, we look some examples of quantum noises. Due to their discrete nature, we need different approach to overcome them using techniques other than classical error correction. One thing to note here is that quantum errors are mediated through quantum fields, therefore fully suppressing them in hardware is physically impossible.

Let $\mathcal{E}$ be some quantum operation which acts on a state as follows: [9]

$$\mathcal{E}(\rho) = \sum_j E_j \rho E_j^\dagger \tag{1.15}$$

where $E_j$ are the Krauss operators such that $\sum_j E_j^\dagger E_j = 1$ and equation.1.15 is called *Krauss representation*. We will use this description to describe quantum noise types.

**Bit Flip Error** This channel flips the state in computational basis such that $|0\rangle$ turns into $|1\rangle$ and vice versa. We can think of it as a Pauli-$X$ acting on that qubit. Following the description in equation.1.15, we have:

$$\begin{aligned} E_0 &= \sqrt{p}\mathbb{I} \\ E_1 &= \sqrt{1-p}X \end{aligned} \tag{1.16}$$

or in equation.1.15 form:

$$\mathcal{E}_X(\rho) = p\rho + (1-p)X\rho X \tag{1.17}$$

**Phase Flip Error** Acts like bit flip but in Hadamard basis such that it turns $|+\rangle$ to $|-\rangle$ and vice versa. We can consider it as a Pauli-$Z$ acting on that qubit.

$$\begin{aligned} E_0 &= \sqrt{p}\mathbb{I} \\ E_1 &= \sqrt{1-p}Z \end{aligned} \tag{1.18}$$

or

$$\mathcal{E}_Z(\rho) = p\rho + (1-p)Z\rho Z \tag{1.19}$$

**Bit-Phase Flip Error** Error of bit and phase flips occurring at the same time and can be considered as a Pauli-$Y$ acting on the given qubit.

$$\begin{aligned} E_0 &= \sqrt{p}\mathbb{I} \\ E_1 &= \sqrt{1-p}Y \end{aligned} \tag{1.20}$$

or

$$\mathcal{E}_Y(\rho) = p\rho + (1-p)Y\rho Y \tag{1.21}$$

**Depolarizing Error** Turns quantum state into maximally mixed state in which one can consider as Pauli-$X, Y, Z$ are applied.

$$\begin{aligned} E_0 &= \sqrt{1-\frac{3p}{4}}\rho \\ E_1 &= \frac{\sqrt{p}}{2}X \\ E_2 &= \frac{\sqrt{p}}{2}Y \\ E_3 &= \frac{\sqrt{p}}{2}Z \end{aligned} \tag{1.22}$$

or

$$\mathcal{E}_D(\rho) = (1-\frac{3p}{4})\rho + \frac{p}{4}(X\rho X + Y\rho Y + Z\rho Z) \tag{1.23}$$

**Amplitude Damping Error** A quantum state loosing energy. For a photonic system, it turns $|1\rangle$ into $|0\rangle$ and have no effect on $|0\rangle$ since $|0\rangle$ usually corresponds to the ground state and amplitude damping is loosing energy to the environment. In the finite temperature case, it is called $T_1$ *relaxation* such that a high temperature spin system releases energy resulting in equilibrium with the environment. It has Krauss operators:

$$\begin{aligned} E_0 &= \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma} \end{bmatrix} \\ E_1 &= \begin{bmatrix} 0 & \sqrt{\gamma} \\ 0 & 0 \end{bmatrix} \end{aligned} \tag{1.24}$$

where $\gamma \in [0, 1]$ is the probability of amplitude damping.

**Phase Damping Error** In an evolution of a quantum state for an uncertain amount of time, quantum information is lost without loosing any energy. It is mostly known as $T_2$ *relaxation* and the operator causing it is called *phase kick*. This quantum noise has philosophical significance since it destroys cross terms of density matrix of a given state with superposition. In [9] exercise 8.31 we see that it causes non-diagonal terms to decay with $e^{-k(\Delta t)}$ where $k$ is some constant, leaving only pure states. Therefore we argue that phase damping prevents quantum effects to occur in daily classical life. Its Krauss operators are:

$$\begin{aligned} E_0 &= \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-\lambda} \end{bmatrix} \\ E_1 &= \begin{bmatrix} 0 & 0 \\ 0 & \sqrt{\lambda} \end{bmatrix} \end{aligned} \tag{1.25}$$

## 1.12 Knill-Laflamme Conditions

Knill-Laflamme conditions are a set of necessary and sufficient conditions for a QECC to be able to correct particular error set $\{E_j\}$. They are: [113]

Condition - 1)

$$\langle \overline{a} | E_i^\dagger E_j | \overline{a} \rangle = 0 \tag{1.26}$$

$\forall a, b \in \{0, 1\}, i \neq j$ and $E_i$ are linear operators describing environmental effects.

This condition implies that any error should carry logical sates to states that are orthogonal.

Condition - 2)

$$\langle \overline{a} | E_i^\dagger E_j | \overline{a} \rangle = \langle \overline{b} | E_i^\dagger E_j | \overline{b} \rangle \tag{1.27}$$

Meaning that corrupted logical states have the same length and inner product under projections.

## 1.13 Fidelity

*Fidelity* ($\mathcal{F}$) is the measure of how well a quantum state is preserved. First defined by Schumacher as the probability of a decoded message being similar to the encoded message in a given coding-decoding arrengement. Error probability was $(1 - \mathcal{F})$ which we can call *infidelity*. In classical sense, fidelity goes to 0 if less than $\mathtt{H}(A)$ bits/message are allowed, and goes to 1 if more than $\mathtt{H}(A)$ bits/message are allowed where $\mathtt{H}(A)$ is the Shannon entropy (from equation 1.12) of the message produced by some source A. [41]

Quantum mechanically, we can consider infidelity as transition probability from a pure state $\rho$ to another pure state $\sigma$ where $\rho = |\psi\rangle\langle\psi|$ and $\sigma = |\phi\rangle\langle\phi|$. Then fidelity is: [42]

$$\begin{aligned} \mathcal{F}(\rho, \sigma) &= \langle\phi|\psi\rangle \langle\psi|\phi\rangle \\ &= |\langle\psi|\phi\rangle|^2 \end{aligned} \tag{1.28}$$

which we can describe as the probability of a system staying in the same state. This equation measures the distance between two states in usual Hilbert space geometry. If one of the states is impure, say $\sigma$, then:

$$\mathcal{F}(|\psi\rangle\langle\psi|, \sigma) = \langle\psi|\sigma|\psi\rangle \tag{1.29}$$

## 1.14 The Outline of the Thesis

Chapter 1 of this thesis gives preliminary information for Quantum Error Correction. Those are mostly classical error correction and some bounds for QEC. In the Chapter 2, we discuss Stabilizer codes which are the most well studied topic of QEC. In Chapter 3, we investigate the QEC codes generating quantum memories. In Chapter 4, we briefly mention the QECCs assisted by machine learning techniques. Finally, in Chapter 5, we conclude the concepts discussed in this thesis.

# CHAPTER 2

# STABILIZER CODES

## 2.1 Introduction

This method allow us to work in many quantum systems with the operators that stabilizes the quantum state instead of working with the explicit description of the state. Here stabilize means that the eigenvalue of the operator is $+1$ whenever acted on that state such that: [9] (and we follow [9] throughout this section)

$$M\left|\psi\right\rangle = \left|\psi\right\rangle \quad \text{where } M \text{ is an operator and } \left|\psi\right\rangle \text{ is some quantum state} \tag{2.1}$$

Here we say that the state $|\psi\rangle$ is stabilized by operator $M$. Suppose $\mathcal{S}$ is a subgroup of the Pauli group on $n$ qubits $G_n$ and let each element of $\mathcal{S}$ to fix the set of $n$-qubit states $V_S$. We say $V_S$ is the vector space stabilized by $\mathcal{S}$, and since all elements of $V_S$ are stable under the operators in $\mathcal{S}$, we say $\mathcal{S}$ is the stabilizer of space $V_S$.

This method also allow us to describe errors in the qubits, operations, measurements in the computational basis and many quantum codes in a more compact way.

The trick of stabilizer formalism is the Pauli group $G_n$ on $n$ qubits. Note that for a single qubit, all Pauli matrices with $\pm 1$ and $\pm i$ factors has to be in the group so that group can be closed under matrix multiplication.

$$G_1 \equiv \{\pm\mathbb{I}, \pm i\mathbb{I}, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\} \tag{2.2}$$

And the Pauli group in $n$ qubits contains all $n$-fold tensor products of Pauli matrices.

Stabilizer codes are Hamiltonian-based codes that its Hamiltonian can be written in terms of commuting projectors. Note that the codespace of a stabilizer code is the ground-state space of the stabilizer codes Hamiltonian that contains equal linear combination of generators. [43]

As a simple example to stabilizer codes, consider $n = 3$ case with $\mathcal{S} = \{\mathbb{I}, Z_1Z_2, Z_2Z_3, Z_1Z_3\}$. We can see that this subspace which is fixed by $Z_1Z_2$ is spanned by $|000\rangle, |111\rangle, |110\rangle, |111\rangle$ while the subset fixed by $Z_1Z_3$ is spanned by $|000\rangle, |010\rangle$, $|101\rangle, |111\rangle$. Then we can look at the subspace stabilized by two operators (which we call *generators*) in $\mathcal{S}$ and say that $V_S$ is spanned by $|000\rangle$ and $|111\rangle$ which are the common in subspace fixed by $Z_1Z_2$ and subspace fixed by $Z_1Z_3$.

And for a group $G$ the set of elements $g_1, \cdots, g_l \in G$ are called *generators* if any $g \in G$ can be written in terms of products of those elements. Then we denote $G = \langle g_1, \cdots, g_l \rangle$

Note that the generator set of a group $G$ with size $|G|$ has at most $\log |G|$ elements. [9] And to check if a state is stabilized by a subgroup of $\mathcal{S}$ we can simply check the generators since if a state is stabilized by a set of operators, then their products also stabilize the state.

Also note that we can't use every subgroup of Pauli group for a non-trivial vector space. For instance, considering subgroup $G_1 \equiv \{\pm\mathbb{I}, \pm X\}$ for $-1\,|\psi\rangle = |\psi\rangle$, $|\psi\rangle$ must be zero giving trivial solution. Therefore, $\{\pm\mathbb{I}, \pm X\}$ stabilizes trivial vector space.

Non-trivial vector space $V_S$ is stabilized by $\mathcal{S}$ only if:

1. $[M, N] = 0 \quad \forall\, M, N \in \mathcal{S}$ (elements of $\mathcal{S}$ commute)

2. $-\mathbb{I} \notin \mathcal{S}$

Now we want generators to be independent so that removing any element $g_i$ results in smaller generated group such that:

$$\langle g_1, \cdots, g_{i-1}, g_{i+1}, \cdots, g_l \rangle \neq \langle g_1, \cdots, g_l \rangle \tag{2.3}$$

To check if generators are independent, we use check matrix (analogous to PCMs in classical linear codes). Suppose we have $\mathcal{S} = \langle g_1, \cdots, g_l \rangle$ and there exists an $l \times 2n$ matrix. The rows of that matrix are the generators. Columns from first to $j^{th}$ correspond to $X$ operators acting, $(j+1)^{th}$ to $(j+n)^{th}$ correspond to $Z$ operators. Consider following matrix:

$$\begin{array}{c} \\ g_1 \rightarrow \\ g_2 \rightarrow \\ \vdots \\ g_l \rightarrow \\ \\ \end{array} \left[\begin{array}{ccccccc|ccccccc} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & & & & & & & & & & & & & \\ 0 & & & \ddots & & & & & & \vdots & & & & \\ 0 & & & & & & & & & & & & & \\ \multicolumn{7}{c}{\underbrace{\qquad\qquad\qquad}_{Xs}} & & \multicolumn{6}{c}{\underbrace{\qquad\qquad\qquad\qquad\qquad}_{Zs}} \end{array}\right] \tag{2.4}$$

Here we see $\mathbb{I} \otimes \mathbb{I}$ is acting on first two qubits, it's trivial. Then we see in the $3^{rd}$ & $(3+7)^{th}$ columns that $Z$ operator is applied. Since $4^{th}$ & $5^{th}$ columns are 1 and $(4+7)^{th}$ & $(5+7)^{th}$ columns are 0 meaning we apply $X_4 X_5$. Notice that in the $6^{th}$ & $7^{th}$ qubits, we have 1s in both sides, it means that we apply Pauli-Y.

$$\therefore g_1 : \ \mathbb{I} \otimes \mathbb{I} \otimes Z \otimes X \otimes X \otimes Y \otimes Y = Z_3 X_4 X_5 Y_6 Y_7 \tag{2.5}$$

Note that subscript of operators represents the specific qubit it is acted on, and check matrix doesn't contain $\pm 1, \pm i$ factors.

Let $r(g)$ denotes $2n$-dimensional row vector of some generator $g$ and suppose $X' = \begin{bmatrix} 0 & \mathbb{I} \\ \mathbb{I} & 0 \end{bmatrix}$ where $\mathbb{I}$ are $n \times n$ and $X'$ is $2n \times 2n$. Then for given $g, g' \in \mathbb{P}_n$ ($n$-fold Pauli subgroup), $g$ and $g'$ commute iff

$$r(g) X' r(g')^T = 0 \tag{2.6}$$

**Proposition 2.1.1** *For a stabilizer $\mathcal{S} = \langle g_1, \cdots, g_l \rangle \ni -\mathbb{I} \notin \mathcal{S}$, all generators are independent if and only if the rows of the check matrix are not linearly dependent.*

**Proposition 2.1.2** *For a stabilizer $\mathcal{S} = \langle g_1, \cdots, g_l \rangle \ni -\mathbb{I} \notin \mathcal{S}$ and fixed $i \ni i = 1, \cdots, l$, $\exists g \in G_n \ni g g_i g^\dagger = -g_i$ and $g g_j g^\dagger = g_j \; \forall i \neq j$.*

**Proposition 2.1.3** *For a stabilizer $\mathcal{S} = \langle g_1, \cdots, g_{n-k} \rangle \ni -\mathbb{I} \notin \mathcal{S}$ where $[g_i, g_j] = 0$ and $g_i \ni G_n \forall i = 1, \cdots, n-k$, $V_S$ is a 2D vector space.*

**Unitary gates in Stabilizers** Suppose vector space $V_S$ is stabilized by $\mathcal{S}$. If we apply a unitary operator $U$, then $\forall g \in \mathcal{S}$:

$$U|\psi\rangle = Ug|\psi\rangle = UgU^\dagger U|\psi\rangle \quad \text{where } |\psi\rangle \in V_S \tag{2.7}$$

Meaning that the state $U|\psi\rangle$ is stabilized by $UgU^\dagger$, therefore the vector space $UV_S$ is stabilized by group $USU^\dagger = \{UgU^\dagger | g \in \mathcal{S}\}$. Moreover if $g_1, \cdots, g_l$ generate $\mathcal{S}$, then $Ug_1U^\dagger, \cdots, Ug_lU^\dagger$ generate $USU^\dagger$. So, checking the generators gives how the stabilizer is affected.

With this method, we can stabilize a quantum state $|0\rangle$ which is stabilized by $Z$ with $X$ by Hadamard operator since:

$$HXH^\dagger = Z, \quad HYH^\dagger = -Y, \quad HZH^\dagger = X \tag{2.8}$$

Considering this in $n$-qubit systems, say $|0\rangle^{\otimes n}$ which is stabilized by $\langle Z_1, \cdots, Z_n \rangle$ we see that after Hadamard, we superpose the states in computational basis to $|+\rangle^{\otimes n}$ which is stabilized by $\langle X_1, \cdots, X_n \rangle$. Following table shows the results of some unitary operations:

Note that since for a set of $U \ni UG_nU^\dagger = G_n$ is called normalizer of $G_n$ (denoted by $N(G_n)$), $H$, $S$ & CNOT gates sometimes are called normalizer gates.

**Theorem 2.1.1** *Suppose $U$ is a unitary operator acting on $n$ qubits. And if $g \in G_n$*

| | CNOT (control: 1, target:2) | | | | H | | S | | X | | Y | | Z | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Input | $X_1$ | $X_2$ | $Z_1$ | $Z_2$ | $X$ | $Z$ | $X$ | $Z$ | $X$ | $Z$ | $X$ | $Z$ | $X$ | $Z$ |
| Output | $X_1X_2$ | $X_2$ | $Z_1$ | $Z_1Z_2$ | $Z$ | $X$ | $Y$ | $Z$ | $X$ | $-Z$ | $-X$ | $-Z$ | $-X$ | $Z$ |

Table 2.1: Results of some unitary operations

*then $UgU^\dagger \in G_n$ is true. As a result $U$ may be contains $O(n^2)$ $H$, phase and CNOT gates.*

Note that $\pi/8$ and Toffoli gates are not normalizer gates.

Let $T \equiv \pi/8$ and $U \equiv$ Toffoli with $1^{st}$ and $2^{nd}$ qubits as control and $3^{rd}$ qubit target, then:

$$TZT^\dagger = Z \tag{2.9}$$

$$UZ_1U^\dagger = Z_1 \tag{2.10}$$

$$UZ_2U^\dagger = Z_2 \tag{2.11}$$

$$UZ_3U^\dagger = Z_3 \otimes \frac{1}{2}(\mathbb{I} + Z_1 + Z_2 - Z_1Z_2) \tag{2.12}$$

$$TXT^\dagger = \frac{1}{\sqrt{2}}(X + Y) \tag{2.13}$$

$$UX_1U^\dagger = X_1 \otimes \frac{1}{2}(\mathbb{I} + Z_2 + X_3 - Z_2X_3) \tag{2.14}$$

$$UX_2U^\dagger = X_2 \otimes \frac{1}{2}(\mathbb{I} + Z_1 + X_3 - Z_1X_3) \tag{2.15}$$

$$UX_3U^\dagger = X_3 \tag{2.16}$$

Therefore, the circuits containing $\pi/8$ and Toffoli gates are not suitable to analyse with stabilizer formalism. Stabilizer formalism is useful for circuits with normalizer gates.

**Measurement** Assume that $g$ is some tensor product of Pauli matrices and have no $-1$ or $\pm i$ factor in front without loss of generality. Assume system in the state $|\psi\rangle$ with $\mathcal{S} = \langle g_1, \cdots, g_n \rangle$ and we want to make a measurement of $g \in G_n$ (since Hermitian matrix $g$ acts as an observable). It's either:

a) $[g, g_j] = 0 \; \forall j = 1, \cdots, n$
$g$ commutes with all generators of stabilizer.

Since $g_j g |\psi\rangle = g g_j |\psi\rangle = g |\psi\rangle \; \forall g_j \ni j = 1, \cdots, n$ and $g_j \in \mathcal{S}$, then $g |\psi\rangle \in V_S$

and hence $g$ is multiple of $|\psi\rangle$.

Since $g^2 = \mathbb{I}$ then $g|\psi\rangle = \pm|\psi\rangle$ and therefore either $g$ or $-g$ must be in the stabilizer.

For the sake of discussion, assume $g \in \mathcal{S}$ ($-g \in \mathcal{S}$ proceeds analogously) then $g|\psi\rangle = |\psi\rangle$ giving measurement of $g$ $+1$ with prob 1. This measurement doesn't collapse the quantum state and therefore leaves stabilizer invariant.

b) $g$ anti-commutes with one or more generators of the stabilizer.
Assume $\mathcal{S} = \langle g_1, \cdots, g_n\rangle$ and $g$ anti-commutes with $g_1$, then we can also assume that $g$ and $\langle g_2, \cdots, g_n\rangle$ commute without loss of generality. If it doesn't commute with one of the generators, say $g_2$, we can verify that $g$ commutes with $g_1 g_2$ and replace the $g_2$ in the list with $g_1 g_2$.
For a $g$ anti-commuting with $g_1$ and commuting with the rest of the generators, $g$ has $\pm 1$ eigenvalues giving projective measurement outcomes $\pm 1$ are $(\mathbb{I} \pm g)/2$. So the probabilities of measurement are:

$$\mathcal{P}(+1) = tr\left(\frac{\mathbb{I}+g}{2}|\psi\rangle\langle\psi|\right) \quad \& \quad \mathcal{P}(-1) = tr\left(\frac{\mathbb{I}-g}{2}|\psi\rangle\langle\psi|\right) \tag{2.17}$$

using $g_1|\psi\rangle = |\psi\rangle$ and $gg_1 = -g_1 g$ gives:

$$\mathcal{P}(+1) = tr\left(\frac{\mathbb{I}+g}{2} g_1|\psi\rangle\langle\psi|\right) = tr\left(g_1\frac{\mathbb{I}-g}{2}|\psi\rangle\langle\psi|\right) \tag{2.18}$$

using property of trace:

$$\mathcal{P}(+1) = tr\left(\frac{\mathbb{I}-g}{2}|\psi\rangle\langle\psi| g_1^\dagger g_1\right) = \mathcal{P}(-1) \tag{2.19}$$

$$\rightarrow \mathcal{P}(+1) = \mathcal{P}(-1) = 1/2 \tag{2.20}$$

then:

- +1 result: new state of system:

$$|\psi^+\rangle \equiv \frac{1}{\sqrt{2}}(\mathbb{I}+g)|\psi\rangle \quad \text{with } \mathcal{S} = \langle g_1 g_2, \cdots, g_n\rangle \tag{2.21}$$

- -1 result: posterior state stabilized by $\mathcal{S} = \langle -g, g_2, \cdots, g_n\rangle$

**Theorem 2.1.2** *Gottesman-Knill theorem*
*Suppose a quantum computation including only H, CNOT, phase and Pauli gates, state preparation in computational basis and measurements of the observables from Pauli group. It is possible to simulate this computation on a classical computer.*

Note that a set of $m$ operations requires $O(n^2m)$ operations to simulate on a classical pc.

**Constructing Stabilizer codes** Let $C(\mathcal{S})$ be an $[n, k]$ stabilizer code on vector space $V_S$ which is stabilized by $\mathcal{S} \in G_n \ni -\mathbb{I} \notin \mathcal{S}$ where $\mathcal{S}$ has $n-k$ independent and commuting generators.
In principle, there are $2^k$ possible choise of logical computational basis states from orthonormal vectors in $C(\mathcal{S})$.
For practical purposes, we begin choosing operators $\overline{Z}_1, \cdots, \overline{Z}_k \in G_n \ni g_1, \cdots, g_{n-k}$, $\overline{Z}_1, \cdots, \overline{Z}_k$ forms a set of commuting and independent elements. Since $\overline{Z}_j$ represents the Pauli $Z$ operator on $j^{th}$ qubit, logical computational basis state $|X_1, \cdots, X_k\rangle_L$ has stabilizer:

$$\mathcal{S} = \langle g_1 \cdots, g_{n-k}, (-1)^{X_1}\overline{Z}_1, \cdots, (-1)^{X_k}\overline{Z}_k \rangle \tag{2.22}$$

If we define $\overline{X}_j$ as a product of Pauli matrices and under conjugation, this $\overline{X}_j$ turns $\overline{Z}_j$ into $-\overline{Z}_j$ while not touching all other $\overline{Z}_i$ and $g_i$'s $\forall i \neq j$. Then this $\overline{X}_j$ is the NOT gate on $j^{th}$ qubit satisfying $\overline{X}_j g_k \overline{X}_j^\dagger = g_k$, $[\overline{X}_j, \overline{Z}_i] = 0$,$\{\overline{X}_j and \overline{Z}_j\} = 0$.

Suppose an error $\mathscr{E} \in G_n$ corrupted the code $C(\mathcal{S})$. If this error anti-commutes with one of the elements of the stabilizer, it takes $C(\mathcal{S})$ to an orthogonal subspace and in theory, we can detect this error via suitable projective measurement.

If $\mathscr{E} \in \mathcal{S}$, we are also fine because in this case, error doesn't corrupt the space. However if $\mathscr{E}$ commutes with all generators in $\mathcal{S}$ but not present in $\mathcal{S}$ ($Eg = gE$ $\forall g \in \mathcal{S}$, $E \in \mathscr{E}$) it's a problem. We call the set $\mathscr{E} \in G_n \ni [E, g] = 0 \; \forall g \in \mathcal{S}$ as *centralizer of $\mathcal{S}$ in $G_n$* which is denoted by $Z(\mathcal{S})$. Note that centralizer is identical to normalizer of $\mathcal{S}$.

**Theorem 2.1.3** *Conditions for a stabilizer code*
*Let $C(\mathcal{S})$ be a stabilizer code with stabilizer group $\mathcal{S}$. Supposing $\{E_j\}$ as a generator set in $G_n \ni E_j^\dagger E_k \notin N(\mathcal{S})/\mathcal{S}\ \forall j,k$ we have that error set $\{E_j\}$ of $C(\mathcal{S})$ is correctable.*

Note that we can consider only errors $E_j \in G_n \ni E_j^\dagger = E_j$ and reduce the error correction conditions for stabilizer codes to $E_j E_k \ni N(\mathcal{S})/\mathcal{S}\ \forall j,k$ without loss of generality.
And also note that the theorem 2.1.3 dictates that a code with at least $2t+1$ distance can correct errors in up to $t$ qubits.

Before performing error correction, let $\mathcal{S} = \langle g_1, \cdots, g_{n-k} \rangle$ is the stabilizer of an $[n,k]$ stabilizer code while $\{E_j\}$ is the set of errors that we can correct. To correct errors, we begin with measuring the error-syndromes with generators $g_1$ to $g_{n-k}$.

If we have a single error $E_j$ occured, we measure $\beta_l \ni E_j g_l E_j^\dagger = \beta_l g_l$. To recover, we apply $E_j^\dagger$.
But if we have two distinct errors $E_j$ and $E_{j'}$ that result in the same syndrome, then $E_j P E_j^\dagger = E_{j'} P E_{j'}^\dagger$ giving $E_j^\dagger E_{j'} P E_{j'}^\dagger = P$ where $E_j^\dagger E_j \in \mathcal{S}$ and $P$ is the projection operator onto the code space. So, we can correct the error by applying $E_{j'}$ followed by $E_j^\dagger$.

Finally we give some last definitions thanks to theorem 2.1.3 which allows us to construct quantum analogue of distance.
We call the numbers of non-identity terms in the tensor product describing the error is called *weight*. As an example $wt(X_2 Y_5 Z_6) = 3$. The minimum weight of an element of $N(\mathcal{S})/\mathcal{S}$ is called the distance of some code $C(\mathcal{S})$. And if $C(\mathcal{S})$ is an $[n,k]$ stabilizer code with distance $d$, it is an $[n,k,d]$ stabilizer code and denoted with $[[n,k,d]]$.

## 2.2 Calderbank-Shor-Steane (CSS) Codes

CSS codes are probably the most well studied type of stabilizer codes. Let we have $[n, k_1]$ and $[n, k_2]$ classical linear codes of $C_1$ and $C_2$ respectively where $C_2 \subset C_1$ and both $C_1$ and $C_2^{\perp}$ can correct up to $t$ errors. Then we can construct an $[n, k_1 - k_2]$ quantum code that can correct errors on up to $t$ qubits.

Let $x, y$ be a codewords such that $x \in C_1$ and $y \in C_2$. Then let,

$$|x \oplus C_2\rangle \equiv \frac{1}{\sqrt{|C_2|}} \sum_{y \in C_2} |x \oplus y\rangle \tag{2.23}$$

Then let $x' \in C_1 \ \ni x - x' \in C_2$. So, $|x \oplus C_2\rangle = |x' \oplus C_2\rangle$ which implies that $|x \oplus C_2\rangle$ state depends only on the coset of $C_1/C_2$.
Notice that if x and x' are in different $C_2$ cosets, then we ensure that $x \oplus y \neq x' \oplus y' \ \forall y, y' \in C_2$, resulting that $|x \oplus C_2\rangle$ and $|x' \oplus C_2\rangle$ states are orthonormal.
Since the number of cosets $C_2$ in $C_1$ is $|C_1|/|C_2|$, $dim(CSS(C_1, C_2)) = |C_1|/|C_2| = 2^{k_1 - k_2}$. So, we say that $CSS(C_1, C_2)$ is an $[n, k_1 - k_2]$ quantum code which is defined as a vector space spanned by $|x \oplus C_2\rangle$ states for all $x \in C_1$. This code can detect and correct up to $t$ bit-flip and phase-flip errors.

To illustrate how CSS codes work, consider the state $|x \oplus C_2\rangle$, then bit and phase flip errors occured causing the state become:

$$\frac{1}{\sqrt{|C_2|}} \sum_{y \in C_2} (-1)^{(x \oplus y) \cdot e_2} |x \oplus y \oplus e_1\rangle \tag{2.24}$$

where $e_1$ is $n$ bit vector of bit flip error while $e_2$ is $n$ bit vector of phase flip error.
At this point, we need ancilla qubits to store the information coming from the syndrome measurement such that $|x \oplus y \oplus e_1\rangle$. Applying parity check matrix $H_1$ (for $C_1$) gives:

$$|x \oplus y \oplus e_1\rangle \, |H_1(x \oplus y \oplus e_1)\rangle = |x \oplus y \oplus e_1\rangle \, |H_1 e_1\rangle \tag{2.25}$$

Then,

$$\frac{1}{\sqrt{|C_2|}} \sum_{y \in C_2} (-1)^{(x \oplus y) \cdot e_2} |x \oplus y \oplus e_1\rangle |H_1 e_1\rangle \tag{2.26}$$

Now, we can measure the ancilla state and get the bit-flip $H_1 e_1$. Excluding the ancilla:

$$\frac{1}{\sqrt{|C_2|}} \sum_{y \in C_2} (-1)^{(x \oplus y) \cdot e_2} |x \oplus y \oplus e_1\rangle \tag{2.27}$$

We can guess $e_1$ with error syndrome $H_1 e_1$ since $C_1$ can correct up to $t$ errors which finishes the error-detection. And for recovery, we simply apply $NOT$ gate wherever bit-flip has occurred. Then the state is:

$$\frac{1}{\sqrt{|C_2|}} \sum_{y \in C_2} (-1)^{(x \oplus y) \cdot e_2} |x \oplus y\rangle \tag{2.28}$$

Next, we will deal with the phase flip. To find them, we first apply Hadamard gate to each qubit.

$$\frac{1}{\sqrt{|C_2| 2^n}} \sum_{z} \sum_{y \in C_2} (-1)^{(x \oplus y) \cdot (e_2 \oplus z)} |z\rangle \tag{2.29}$$

Note that the summation here is over all possible $n$ bit $z$ values. Let $z' \equiv z + e_2$, then we have:

$$\frac{1}{\sqrt{|C_2| 2^n}} \sum_{z} \sum_{y \in C_2} (-1)^{(x \oplus y) \cdot z'} |z' \oplus e_2\rangle \tag{2.30}$$

Notice if $z' \in C_2^{\perp}$ we have $\sum_{y \in C_2} (-1)^{y \cdot z'} = |C_2|$, but if $z' \notin C_2^{\perp}$ we have $\sum_{y \in C_2} (-1)^{y \cdot z'} = 0$

Therefore we can write the state as:

$$\frac{1}{\sqrt{2^n / |C_2|}} \sum_{z' \in C_2^{\perp}} (-1)^{x \cdot z'} |z' \oplus e_2\rangle \tag{2.31}$$

Notice this state looks like bit-flip case but with $e_2$, then like in bit-flip case, we introduce ancilla qubit, apply parity-check matrix $H_2$ (for $C_2^{\perp}$), get syndrome elementary steps, resulting state is:

$$\frac{1}{\sqrt{2^n/|C_2|}} \sum_{z' \in C_2^{\perp}} (-1)^{x \cdot z'} |z'\rangle \tag{2.32}$$

Again, to complete the error correction, we apply Hadamard gate to all qubits. We either calculate the results directly or use the fact that the Hadamard is inverse of itself and it is the same as applying to the $e_2 = 0$ case of 2.31 which gives us the $e_2 = 0$ case of $\frac{1}{\sqrt{|C_2|}} \sum_{y \in C_2} (-1)^{(x \oplus y) \cdot e_2} |x \oplus y\rangle$. Finally what we end up with is

$$\frac{1}{\sqrt{|C_2|}} \sum_{y \in C_2} |x \oplus y\rangle \tag{2.33}$$

which is the originally encoded state.

CSS codes proves the quantum version of Gilbert-Varshamov bound and guarantees us that the good quantum error correction codes exists.

### 2.2.1 Steane Code

Also known as [[7,1,3]] Steane code which is constructed using [7,4,3] Hamming code. Its PCM is

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{2.34}$$

Let $C_S$ denotes the Steane code and $C_1 \equiv C_S$ and $C_2 \equiv C_S^{\perp}$. But first, we check that $C_2 \subset C_1$ so that these codes define a CSS code or not. We know:

$$H(C_2) = G(C_1^T) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{2.35}$$

Notice that span of the rows of $H(C_2)$ also spans the rows of $H(C_1)$. And since $C_1$ and $C_2$ are kernels of $H(C_1)$ and $H(C_2)$ respectively, we ensured $C_2 \subset C_1$. Since

$d(C_S) = 3$ and $C_2^{\perp} = (C_S^{\perp})^{\perp} = C_S$, both distances of $C_1$ and $C_2$ are 3 meaning they can correct 1 bit of error and since $C_1$ is [7,4] code while $C_2$ is [7,3], we conclude that we can correct errors on a single qubit with $CSS(C_1, C_2)$ which is a [7,1] code. Denoting logical $|0\rangle$ with $|\bar{0}\rangle$ we have:

$$\begin{aligned}|\bar{0}\rangle =& \frac{1}{\sqrt{8}}(|0000000\rangle + |1010101\rangle + |0110011\rangle + |1100110\rangle + |0001111\rangle \\ &+ |1011010\rangle + |0111100\rangle + |1101001\rangle)\end{aligned} \tag{2.36}$$

The other logical codeword is found by an element of $C_1$ which is not in $C_2$, like $(1111111)$ giving:

$$\begin{aligned}|\bar{1}\rangle =& \frac{1}{\sqrt{8}}(|1111111\rangle + |0101010\rangle + |1001100\rangle + |0011001\rangle + |1110000\rangle \\ &+ |0100101\rangle + |1000011\rangle + |0010110\rangle\end{aligned} \tag{2.37}$$

### 2.2.2 Classical-Product Codes

A CSS technique which doesn't need that the classical linear codes which are used to construct quantum code to be self-orthogonal. [12]

**Theorem 2.2.1** *Let $H$ be a PCM of some $[n, k, d]$ code $C$ with a permutation $\pi \ni H(H\pi)^T = 0$. Then $H' = \begin{pmatrix} H & 0 \\ 0 & H\pi \end{pmatrix}$ matrix yields a check matrix for an [[n, 2k-n, d]] quantum code.*

Notice that in special case, when $\pi = \mathbb{I}$, $C$ becomes self-orthogonal implying that the CSS code with self-orthogonal condition is just a special case of theorem 2.2.1. Since the code $C\pi$ is orthonormal to $H$, $C\pi$ is subcode of $C$. So,

$$|x + C\pi\rangle = \frac{1}{\sqrt{|C\pi|}} \sum_{y \in C\pi} |x + y\rangle \quad \forall x \in C \tag{2.38}$$

with $\frac{|C|}{|C\pi|} = 2^{(2k-n)}$ number of them.

To find a suitable permutation $\pi$, we should search for all permutations on $n$ element which requires a heavy computer search. Note that there is no quantum code satisfying theorem 2.2.1 if there is no suitable permutation exists.

To illustrate the construction, we start by finding suitable permutation $\pi$ for an $[n, n/2, d]$ code $C$ with parity-check $H$ as

$$H = \begin{pmatrix} \mathbb{I}_{n/2} & \mathbb{S} \end{pmatrix} \tag{2.39}$$

where $\mathbb{I}_{n/2}$ is identity and $\mathbb{S}$ is a symmetric matrix.
Consider permutation $\pi$ which replaces $\mathbb{I}_{n/2}$ and $\mathbb{S}$ giving:

$$H\pi = \begin{pmatrix} \mathbb{S} & \mathbb{I}_{n/2} \end{pmatrix} \tag{2.40}$$

Notice we have $H(H\pi)^T = 0$ and $C\pi = C^{\perp}$. And it is shown that in [13], $C$ and $C\pi$ contain codes satisfying Gilbert-Varshamov bound for large $n$.
As an example, consider the Pauli group $\mathbb{P}_9 = \{g_1, g_2, \cdots, g_8\}$ where $g_i$ contains only $X$ operators $\forall i \leq 4$ and $Z$ operators $\forall i \geq 5$. Using theorem 2.2.1 we can construct $[[9, 1, 3]]$ quantum code as follows:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $g_1$ | $X$ | $\mathbb{I}$ | $\mathbb{I}$ | $\mathbb{I}$ | $X$ | $X$ | $\mathbb{I}$ | $\mathbb{I}$ | $X$ |
| $g_2$ | $\mathbb{I}$ | $X$ | $\mathbb{I}$ | $\mathbb{I}$ | $\mathbb{I}$ | $X$ | $X$ | $\mathbb{I}$ | $X$ |
| $g_3$ | $\mathbb{I}$ | $\mathbb{I}$ | $X$ | $\mathbb{I}$ | $\mathbb{I}$ | $\mathbb{I}$ | $X$ | $X$ | $X$ |
| $g_4$ | $\mathbb{I}$ | $\mathbb{I}$ | $\mathbb{I}$ | $X$ | $X$ | $\mathbb{I}$ | $\mathbb{I}$ | $X$ | $\mathbb{I}$ |
| $g_5$ | $Z$ | $\mathbb{I}$ | $Z$ | $\mathbb{I}$ | $\mathbb{I}$ | $Z$ | $Z$ | $\mathbb{I}$ | $\mathbb{I}$ |
| $g_6$ | $\mathbb{I}$ | $Z$ | $\mathbb{I}$ | $\mathbb{I}$ | $\mathbb{I}$ | $Z$ | $Z$ | $\mathbb{I}$ | $Z$ |
| $g_7$ | $\mathbb{I}$ | $\mathbb{I}$ | $\mathbb{I}$ | $Z$ | $Z$ | $\mathbb{I}$ | $Z$ | $\mathbb{I}$ | $Z$ |
| $g_8$ | $\mathbb{I}$ | $\mathbb{I}$ | $Z$ | $Z$ | $\mathbb{I}$ | $\mathbb{I}$ | $\mathbb{I}$ | $Z$ | $\mathbb{I}$ |

Table 2.2: Generators for Classical-Product Codes [12]

with following parity-check matrices:

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \quad \forall i \leq 4 \tag{2.41}$$

$$H' = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad \forall i \geq 5 \tag{2.42}$$

Here the code $C$ with PCM $H$ is a $[9, 5, 3]$ code. Since $H'$ is produced by the permutation of the columns of $H$ with $\pi$, then $H' = H\pi$, and since all the rows of $H$ are orthogonal to all the rows of $H'$, $H(H\pi)^T = 0$. So, by theorem 2.2.1, we obtain the check matrix for $[[9, 1, 3]]$ quantum code $C_9$ as

$$H_x = \begin{pmatrix} H & 0 \\ 0 & H\pi \end{pmatrix} \tag{2.43}$$

For a generalized discussion, we can redefine classical product code $C_x$ as follows: Suppose two classical codes $C_1 \subseteq \mathbb{F}_2^{n_1}$ and $C_2 \subseteq \mathbb{F}_2^{n_2}$ having parity-check matrices $H_1 \in \mathbb{F}_2^{m_1 \times n_1}$ and $H_2 \in \mathbb{F}_2^{m_2 \times n_2}$. Then the 2-fold product of them $C_x$ an its PCM $H_x$ are:

$$C_x = C_1 \otimes C_2 \subseteq \mathbb{F}_2^{n_1} \times \mathbb{F}_2^{n_2} \tag{2.44}$$

and

$$H_x \coloneqq \begin{pmatrix} H_1 \otimes \mathbb{I} \\ \mathbb{I} \otimes H_2 \end{pmatrix} \in \mathbb{F}_2^{(n_1 m_2 + m_1 n_2) \times n_1 n_2} \tag{2.45}$$

Here we have $dim(C_x) = k_x = k_1 k_2$, distance of $C_x$ $d_x = d_1 d_2$, length $n_x = n_1 n_2$ and check $m_x = n_1 m_2 + m_1 n_2$. With syndrome measurements, errors can be detected by applying $H_1$ to each row (which is equivalent to applying $H_1 \otimes \mathbb{I}$ to each bit) and $H_2$ to each column (or apply $\mathbb{I} \otimes H_2$ to all). For systematic encoders, the codewords $\omega \in \mathbb{F}_2^n$ are the concatenation of a set of parity check bits $c \in \mathbb{F}_2^{n-k}$ and the message $x \in \mathbb{F}_2^k$ such that $\omega = (x, c)$. Errors can be detected by checking $m_1 = n_1 - k_1$

or $m_2 = n_2 - k_2$ last bits of rows or columns if they are valid combinations of the message bits respectively. Here we can define *meta-checks* using $m_1 m_2$ bits such as:

$$\bar{m}_x \coloneqq (m_1 - r_1)n_2 + (m_2 - r_2)n_1 + r_1 r_2 \tag{2.46}$$

#### 2.2.2.1 Quantum Single Parity Check Product Codes

Suppose that we have an $[n, n-1, 2]$ SPC code $C_{SPC} \in \mathbb{F}_2^n$ giving $[n^2, (n-1)^2, 4]$ SPC product code. To encode, we add 0s and 1s to the ends of the columns and rows of a 2D $(n-1) \times (n-1)$ matrix, making even number of 1 in each row and column. To decode we check both columns and rows if 1s are even or odd. If a column or a row is odd, it means that there exists an error in that row/column and can be corrected by repeating the process.
The parity check matrix of SPC(n) is: [12]

$$H_{SPC(n)} = \begin{pmatrix} \mathbf{1} \otimes \mathbb{I}_n \\ \mathbb{I}_n \otimes \mathbf{1} \end{pmatrix} \tag{2.47}$$

where $\mathbf{1}$ is vector of 1s such that $\mathbf{1} = 111 \cdots 1$.

Note that since the rows of $H_{SPC(n)}$ are not self-orthogonal $\forall n$, dual code is not in SPC product code.

**Theorem 2.2.2** *$\exists$ an $[[n^2, n^2 - 4n + 2, 4]]$ quantum code where $n = 4j$ $\forall j \in \mathbb{Z}$ which corresponds to an SPC product code with length of $n$ components.*

Consider $n = 4$ case for example, with $[[16, 2, 4]]$ quantum single parity check prod-

uct code having parity check matrices:

$$H_{SPC(4)} = \begin{pmatrix} \mathbf{1} \otimes \mathbb{I}_4 \\ \mathbb{I}_4 \otimes \mathbf{1} \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.48}$$

$$H_{SPC(4)}\pi = \begin{pmatrix} \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \otimes \mathbb{I}_2 \\ \mathbb{I}_2 \otimes \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{2.49}$$

**Theorem 2.2.3** *There is no SPC product code constructed with theorem 2.2.1 if the component length $n$ of the code is odd.*

#### 2.2.2.2 Classical Binary Tensor Product Codes

Let $C_i$ be classical codes with PCMs $H_i \in \mathbb{F}_2^{m_i \times n_i}$ where $i = \{1, 2\}$ and associated tensor product code $C_\otimes$ [14] which are the following: [15]

$$H_{\otimes} \coloneqq H_1 \otimes H_2 \ \in \mathbb{F}_2^{m_1 \times n_1} \otimes \mathbb{F}_2^{m_2 \times n_2} \cong \mathbb{F}_2^{m_1 m_2 \times n_1 n_2} \tag{2.50}$$

$$C_{\otimes} \coloneqq (C_1^{\perp} \otimes C_2^{\perp})^{\perp} \subseteq \mathbb{F}_2^{n_1} \otimes \mathbb{F}_2^{n_2} \cong \mathbb{F}_2^{n_1 n_2} \tag{2.51}$$

with

$$r_{\otimes} = \operatorname{rank}(H_{\otimes}) = r_1 r_2 = \operatorname{rank}(H_1) \operatorname{rank}(H_2) \tag{2.52}$$

hence, the dimension of the code $C_{\otimes}$:

$$\begin{aligned} k_{\otimes} &= n_1 n_2 - r_1 r_2 = k_1 n_2 + n_1 k_2 - k_1 k_2 \\ d_{\otimes} &= \min(d_1 d_2) \end{aligned} \tag{2.53}$$

Tensor products creates codes with higher rates than their components. They increase the weights of the PCMs but the code distances don't change and don't have meta-checks.

#### 2.2.2.3 Asymmetric 2-fold Product Code

**Definition 2.2.1** *Let $C_i$ be CSS codes with PCMs $H_i^j \in \mathbb{F}_2^{m_i^j \times n_i}$ where $i = \{1, 2\}$ and $j = \{X, Z\}$ denoting associated $X$ and $Z$ checks. Theorem 2.2.1 yields:*

$$H_i^x (H_i^z)^T = 0 \ \forall i \in \{1, 2\} \tag{2.54}$$

*Then we define the asymmetric 2-fold product quantum CSS code $C_\alpha$ having parity-check matrices: [15]*

$$H_\alpha^x \coloneqq \begin{pmatrix} H_1^x \otimes \mathbb{I} \\ \mathbb{I} \otimes H_2^x \end{pmatrix} \quad \& \quad H_\alpha^z \coloneqq \begin{pmatrix} H_1^z \otimes H_2^z \end{pmatrix} \tag{2.55}$$

Note that the code we used here is more protective against $Z$ errors than $X$ errors, but one can easily construct a code more sensitive to $X$ errors by reversing the PCMs. This method may be useful for error channels such that $Z$ errors occur more frequently than $X$ errors. If the PCM is constructed with product, meta-checks are in the PCM but not if the PCM is constructed with tensor-product code construction.

| | General case | Special case when $C_1 = C_2$ |
|---|---|---|
| length | $n_\alpha = n_1 n_2$ | $n_\alpha = n^2$ |
| $X$ checks | $m_\alpha^x = m_1^x n_2 + m_2^x n_1$ | $m_\alpha^x = 2nm^x$ |
| $Z$ checks | $m_\alpha^z = m_1^z m_2^z$ | $m_\alpha^z = (m^z)^2$ |
| $X$ meta-checks | $\overline{m}_\alpha^x = (m_1^x - r_1^x)n_2 + (m_2^x - r_2^x)n_1 + r_1^x r_2^x$ | $\overline{m}_\alpha^x = 2n(m^x - r^x) + (r^x)^2$ |
| dimension | $k_\alpha = k_1^x k_2^x - r_1^z r_2^z$ | $k_\alpha = (k^x)^2 - (r^z)^2$ |
| $X$ pure distance | $\delta_\alpha^x = \min(\delta_1^x, \delta_2^x)$ | $\delta_\alpha^x = \delta^x$ |
| $Z$ pure distance | $\delta_\alpha^z = \delta_1 \delta_2$ | $\delta_\alpha^z = (\delta^z)^2$ |
| $X$ row weight | $\omega_{\alpha,r}^x = \max(\omega_{1,r}^x, \omega_{2,r}^x)$ | $\omega_{\alpha,r}^x = \omega_r^x$ |
| $Z$ row weight | $\omega_{\alpha,r}^z = \omega_{1,r}^z \omega_{2,r}^z$ | $\omega_{\alpha,r}^z = (\omega_r^z)^2$ |
| $X$ column weight | $\omega_{\alpha,r}^x = \omega_{1,c}^x + \omega_{2,c}^x$ | $\omega_{\alpha,r}^x = 2\omega_c^x$ |
| $Z$ column weight | $\omega_{\alpha,c}^z = \omega_{1,c}^z \omega_{2,c}^z$ | $\omega_{\alpha,c}^z = (\omega_c^z)^2$ |

Table 2.3: Properties of asymmetric 2-fold product code [15]

We cannot compute true code distances easily but we can set bounds from below such that $d^x \geq \delta^x$ and $d^z \geq \delta^z$. In this code, distance of $C_\alpha$ can be smaller than the product of distances, say $d_\alpha^z < d_1^z d_2^z$ that is not like classical case. The dimension of $C_\alpha$:

$$\begin{aligned} k_i^x &= \dim(\ker(H_i^x)) \\ r_i^z &= \operatorname{rank}(H_i^z) \end{aligned} \tag{2.56}$$

Noting that subscript r/c stands for row/column respectively, table 2.3 shows the properties of $C_\alpha$.

#### 2.2.2.4 Symmetric 2-fold Product CSS Codes

**Definition 2.2.2** *Let $C_i$ be CSS codes with parity-check matrices $H_i^j \in \mathbb{F}_2^{m_i^j \times n_i}$ where $i = \{1,2,3,4\}$ and $j = \{X, Z\}$ denoting associated $X$ and $Z$ checks. Theorem 2.2.1 yields:*

$$H_i^x (H_i^z)^T = 0 \;\; \forall i \in \{1,2,3,4\} \tag{2.57}$$

*Then we define the symmetric 2-fold product quantum CSS code $C_\beta$ having parity-check matrices: [15]*

$$H_\beta^x \coloneqq \begin{pmatrix} H_1^x \otimes H_2^x \otimes \mathbb{I} \otimes \mathbb{I} \\ \mathbb{I} \otimes \mathbb{I} \otimes H_3^x \otimes H_4^x \end{pmatrix} \quad \& \quad H_\beta^z \coloneqq \begin{pmatrix} H_1^z \otimes \mathbb{I} H_3^z \otimes \mathbb{I} \\ \mathbb{I} \otimes H_2^z \otimes \mathbb{I} H_4^z \end{pmatrix} \tag{2.58}$$

| | General case | Special case |
|---|---|---|
| length | $n_\beta = n_1 n_2 n_3 n_4$ | $n_\beta = n^4$ |
| $X$ checks | $m^x_\beta = m^x_1 m^x_2 n_3 n_4 + m^x_3 m^x_4 n_1 n_2$ | $m^x_\beta = 2n^2 (m^x)^2$ |
| $Z$ checks | $m^z_\beta = m^z_1 m^z_3 n_2 n_4 + m^z_2 m^z_4 n_1 n_3$ | $m^z_\beta = 2n^2 (m^z)^2$ |
| $X$ meta-checks | $\overline{m}^x_\beta = (m^x_1 m^x_2 - r^x_1 r^x_2) n_3 n_4$ $+(m^x_3 m^x_4 - r^x_3 r^x_4) n_1 n_2 + r^x_1 r^x_2 r^x_3 r^x_4$ | $\overline{m}^x_\beta = 2n^2((m^x)^2 - (r^x)^2) + (r^x)^4$ |
| $Z$ meta-checks | $\overline{m}^z_\beta = (m^z_1 m^z_3 - r^z_1 r^z_3) n_2 n_4$ $+(m^z_2 m^z_4 - r^z_2 r^z_4) n_1 n_3 + r^z_1 r^z_2 r^z_3 r^z_4$ | $\overline{m}^z_\beta = 2n^2((m^z)^2 - (r^z)^2) + (r^z)^4$ |
| dimension | $k_\beta = \prod_i n_i + \sum_{j\in\{X,Z\}} \prod_i r^j_i - r^x_1 r^x_2 n_3 n_4$ $-r^x_3 r^x_4 n_1 n_2 - r^z_1 r^z_3 n_2 n_4 - r^z_2 r^z_4 n_1 n_3$ | $k_\beta = n^4 + (r^x)^4 + (r^z)^4$ $-2n^2 (r^x)^2 - 2n^2 (r^z)^2$ |
| $X$ pure distance | $\delta^x_\beta = \min(\delta^x_1, \delta^x_3) \min(\delta^x_2, \delta^x_4)$ | $\delta^x_\beta = (\delta^x)^2$ |
| $Z$ pure distance | $\delta^z_\beta = \min(\delta^z_1, \delta^z_2) \min(\delta^z_3, \delta^z_4)$ | $\delta^z_\beta = (\delta^z)^2$ |
| $X$ row weight | $\omega^x_{\beta,r} = \max(\omega^x_{1,r}\omega^x_{2,r}, \omega^x_{3,r}\omega^x_{4,r})$ | $\omega^x_{\beta,r} = (\omega^x_r)^2$ |
| $Z$ row weight | $\omega^z_{\beta,r} = \max(\omega^z_{1,r}\omega^z_{3,r}, \omega^z_{2,r}\omega^z_{4,r})$ | $\omega^z_{\beta,r} = (\omega^z_r)^2$ |
| $X$ column weight | $\omega^x_{\beta,r} = \omega^x_{1,c}\omega^x_{2,c} + \omega^x_{3,c}\omega^x_{4,c}$ | $\omega^x_{\beta,r} = 2(\omega^x_c)^2$ |
| $Z$ column weight | $\omega^z_{\beta,c} = \omega^z_{1,c}\omega^z_{3,c} + \omega^z_{2,c}\omega^z_{4,c}$ | $\omega^z_{\beta,c} = 2(\omega^z_c)^2$ |

Table 2.4: Properties of symmetric 2-fold product CSS code [15]

Note that those parity-check matrices are product of the codes associated with the PCMs $H^z_1 \otimes H^z_3$ and $H^z_2 \otimes H^z_4$ for $H^z_\beta$ and $H^x_1 \otimes H^x_2$ and $H^x_3 \otimes H^x_4$ for $H^x_\beta$ implying that $H^x_\beta$ and $H^z_\beta$ are not full-rank matrices. To find dimension of the code:

$$\begin{aligned} k_\beta &= n_\beta - \mathrm{rank}(H^x_\beta) - \mathrm{rank}(H^z_\beta) \\ r^j_i &= \mathrm{rank}(H^j_i) \end{aligned} \tag{2.59}$$

where $i = \{1, 2, 3, 4\}$ and $j = \{X, Z\}$.

Table 2.4 shows the properties of $C_\beta$ for both general case and the special case when the component codes are equals.

#### 2.2.2.5 Symmetric D-fold Product CSS Codes

We can easily generalize the symmetric product CSS code construction to higher dimensions with the product of $D^2$ CSS codes. To illustrate, think of a set of 9 CSS

codes. Then we can construct PCMs of 3-fold product code: [15]

$$H^x_{(3)} \coloneqq \begin{pmatrix} H^x_1 \otimes H^x_2 \otimes H^x_3 \otimes \mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I} \\ \mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I} \otimes H^x_4 \otimes H^x_5 \otimes H^x_6 \otimes \mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I} \\ \mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I} \otimes H^x_7 \otimes H^x_8 \otimes H^x_9 \end{pmatrix} \tag{2.60}$$

$$H^z_{(3)} \coloneqq \begin{pmatrix} H^z_1 \otimes \mathbb{I} \otimes \mathbb{I} \otimes H^z_4 \otimes \mathbb{I} \otimes \mathbb{I} \otimes H^z_7 \otimes \mathbb{I} \otimes \mathbb{I} \\ \mathbb{I} \otimes H^z_2 \otimes \mathbb{I} \otimes \mathbb{I} \otimes H^z_5 \otimes \mathbb{I} \otimes \mathbb{I} \otimes H^z_8 \otimes \mathbb{I} \\ \mathbb{I} \otimes \mathbb{I} \otimes H^z_3 \otimes \mathbb{I} \otimes \mathbb{I} \otimes H^z_6 \otimes \mathbb{I} \otimes \mathbb{I} \otimes H^z_9 \end{pmatrix} \tag{2.61}$$

**Definition 2.2.3** *Let $H^p_i \in \mathbb{F}_2^{m^p_i \times n_i}$ where $i \in \{1, \cdots, D^2\}$ and $p \in \{X, Z\}$ be associated PCMs checking $X$ and $Z$ for given $p$ of the set of $D^2$ CSS codes $C_1, \cdots, C_{D^2}$. Then we define symmetric D-fold product CSS code $C_{(D)}$ with associated parity-check matrices:*

$$H^p_{(D)} \coloneqq \left[ \underset{j=0}{\overset{D-1}{Stack}} \right] \bigotimes_{i=1}^{D^2} \mathcal{H}^p_{ij} \tag{2.62}$$

*where $p \in \{X, Z\}$ such that*

$$\begin{aligned} \mathcal{H}^x_{ij} &= \begin{cases} H^x_i & \text{if } jD + 1 \leq i \leq (j+1)D \\ \mathbb{I}_{n_i} & \text{otherwise} \end{cases} \\ \mathcal{H}^z_{ij} &= \begin{cases} H^z_i & \text{if } (i-1) \equiv j (mod D) \\ \mathbb{I}_{n_i} & \text{otherwise} \end{cases} \end{aligned} \tag{2.63}$$

*where $[Stack]$ denotes stacking $(m_j \times n)$ matrices one above other and making an $(m \times n)$ matrix.*

We can get meta-checks $\overline{m}^p_{(D)}$ by $\overline{m}^p_{(D)} = m^p_{(D)} - (n_{(D)} - k^p_{(D)})$ where $p \in \{X, Z\}$ and $k^p_{(D)}$ are classical code dimensions related to $X$ and $Z$ parity-check matrices which can be calculated as:

$$k^x_{(D)} = \prod_{j=0}^{D-1} \left( \prod_{i=jD+1}^{(j+1)D} n_i - \prod_{i=jD+1}^{(j+1)D} r^x_i \right)$$
$$k^z_{(D)} = \prod_{j=0}^{D-1} \left( \prod_{\substack{(i-1)\equiv j \\ (mod D)}} n_i - \prod_{\substack{(i-1)\equiv j \\ (mod D)}} r^z_i \right) \tag{2.64}$$

### 2.2.3 Quantum Parity Codes

In this method, we take equal superpositions of all states with even parity as logical-0 while the ones with odd parity as logical-1 giving us the general qubit encoded with $n$ physical qubits as:[5]

$$|\psi\rangle^{(n)} = \alpha \, |\mathbf{even}\rangle^{(n)} + \beta \, |\mathbf{odd}\rangle^{(n)} \tag{2.65}$$

Applying $Z$-measurement and then a CNOT gate corrects the encoded state while reducing the available qubits in the system and giving us the following un-encoded state after necessary number of $Z$-measurements: [1]

$$|\psi\rangle^{(1)} = \alpha \, |\mathbf{0}\rangle + \beta \, |\mathbf{1}\rangle \tag{2.66}$$

In optical QC, a bosonic mode or quantum harmonic oscillator is a system with states $|0\rangle, |1\rangle, \cdots$ where $|0\rangle$ is ground state and $|\mathbf{0}\rangle$ represents the vacuum state that is the all possible modes in $|0\rangle$ state.

Some operators used in LOQC:

- annihilation operator:

$$\mathbf{a}^{(l)} \, |k\rangle_l = \sqrt{k} \, |k-1\rangle_l \, for \; k \geq 1 \quad \& \quad \mathbf{a}^{(l)} \, |0\rangle_l = 0$$

  where $l$ labels the mode.

- creation operator:

$$\mathbf{a}^{(l)\dagger}$$

- number operator:

$$\mathbf{n}^{(l)} \coloneqq \mathbf{a}^{(l)\dagger}\mathbf{a}^{(l)}$$

- phase shifter:

$$\mathbf{n}^{(l)} \coloneqq \mathbf{a}^{(l)\dagger}\mathbf{a}^{(l)}$$

- squeezer:

$$s^{(lm)} \coloneqq \mathbf{a}^{(l)}\mathbf{a}^{(m)} + \mathbf{a}^{(l)\dagger}\mathbf{a}^{(m)\dagger}$$

  This Hamiltonian can be prepared in its vacuum state $|0\rangle$

- beam splitter:

$$\mathbf{b}^{(lm)} \coloneqq \mathbf{a}^{(l)}\mathbf{a}^{m\dagger} + \mathbf{a}^{(l)\dagger}\mathbf{a}^{(m)}$$

Unitary operators related to the phase shifters $P_\theta^{(l)}$ and beam splitters $B_\theta^{(lm)}$:

$$U(P_\theta^{(1)}) = e^{i\theta}, \quad U(B_\theta^{(12)}) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \tag{2.67}$$

Encoding using two modes and one boson: $|0\rangle_q \rightarrow |0\rangle_a |1\rangle_b$ and $|1\rangle_q \rightarrow |1\rangle_a |0\rangle_b$ "bosonic qubit". There we use a single boson source to prepare the states $|1\rangle_a$ in mode $a$ and $|0\rangle_b$ in mode $b$. For example, we can use orthogonal polarization states of an optical mode.

$$CNOT = \begin{pmatrix} \mathbb{I} & 0 \\ 0 & X \end{pmatrix}, \quad CS = \begin{pmatrix} \mathbb{I} & 0 \\ 0 & Z \end{pmatrix}, H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

where 0 stands for the zero matrix of $2 \times 2$.

$$|t_n\rangle = \sum_{j=0}^{n} |1\rangle^{\otimes j} |0\rangle^{\otimes(n-j)} |0\rangle^{\otimes j} |1\rangle^{\otimes(n-j)} \tag{2.68}$$

entanglement due to teleportation. And it has a success probability of $\frac{1}{n+1}$. Note that normalization constants are omitted here.

$$|tp_n\rangle = \textstyle\sum_{j=0}^{n} |1\rangle^{\otimes j} |0\rangle^{\otimes(n-j)} |0\rangle^{\otimes j} |1\rangle^{\otimes(n-j)} |(n-j) \mod (2)\rangle_{a_1} |(n-j+1) \mod (2)\rangle_{a_2} \tag{2.69}$$

variant of $|t_n\rangle$ with a bosonic qubit ancilla containing the information about the parity of the states in superposition.

Now we can carry parity measurement on the space which is spanned by $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ without destructing the state. We can achieve it by teleporting the CS gate with the following instead of $|tp_n\rangle$:

$$|p'_n\rangle = \sum_{i,j:i+j=0(2)} |1\rangle^{\otimes j} |0\rangle^{\otimes(n-j)} |0\rangle^{\otimes j} |1\rangle^{\otimes(n-j)} |1\rangle^{\otimes i} |0\rangle^{\otimes(n-i)} |0\rangle^{\otimes i} |1\rangle^{\otimes(n-i)} \tag{2.70}$$

We can find if the number of bosons in two input modes are even or odd by measuring and find the conditional state using output modes after suitable phase shifts.

Now back into our concern here, which is the QEC, we can define physical encoding as $|0\rangle \equiv |H\rangle$ which is the horizontal polarization and $|1\rangle \equiv |V\rangle$ which is the vertical polarization. The advantage is that it allows us to apply single qubit unitary operations deterministically via passive linear optics instruments. Or instead of physical encoding, we can do *parity encoding* with notation $|\psi\rangle^{(n)}$ which stands for the logical state $|\psi\rangle$ that is encoded using $(n)$ physical qubits. It is:

$$|0\rangle^{(n)} \equiv \frac{1}{\sqrt{2}}(|+\rangle^{\otimes n} + |-\rangle^{\otimes n}) \quad \text{and} \quad |1\rangle^{(n)} \equiv \frac{1}{\sqrt{2}}(|+\rangle^{\otimes n} - |-\rangle^{\otimes n}) \tag{2.71}$$

We can perform two of the following operations on parity encoded qubits without difficulty:

- Rotation by some angle around x-axis of the Bloch sphere by applying following operator to the physical qubit that we are interested in:

$$X_\theta = \cos(\theta/2)\mathbb{I} + i\sin(\theta/2)X \tag{2.72}$$

- The other is applying $Z$ to all of the physical qubits because the states with odd parity acquire an overall phase flip.

Next important operation is the partial Bell state measurement which is the combination of two physical qubits on polarising beam splitter which are measured in

diagonal-antidiagonal basis at the end. When we see photon count at the each outputs of the beamsplitter, this is a successful event in which the state is projected onto the Bell states $|00\rangle + |11\rangle$ and $|00\rangle - |11\rangle$. But if we observe both photons at one of the outputs, it is unsuccessful projecting onto $|01\rangle$ and $|10\rangle$ which are seperable and in computational base. We can add $n$ physical qubits by this operation to parity encoded states with $|0\rangle^{(n+2)}$ resource which we may call "type-II fusion" ($f_{II}$) that is:

$$f_{II} \, |\psi\rangle^{(m)} \, |0\rangle^{(n+2)} \begin{cases} |\psi\rangle^{(m+n)} & \text{success} \\ |\psi\rangle^{(m-1)} \, |0\rangle^{(n+1)} & \text{failure} \end{cases} \tag{2.73}$$

The length of parity qubit $n$ is increased if successful (may be a phase flip correction is necessary after this process depending on the outcome), and a physical qubit is moved from parity encoded state to resource state such that $|0\rangle^{(n)} \rightarrow |0\rangle^{(n+1)}$

We apply parity encoding since non-deterministic gates fail and measure in the computational basis, and as we think of the photon loss as a measurement in computational basis such that we don't know the answer which can be thought as the bit flip version of the original state at worst.

**Reduntant encoding** The final layer of the encoding in which we solve the bit-flip possibility is redundancy code. Consider logical qubits at the highest level:

$$|\psi\rangle_L = \alpha \, |0\rangle_1^{(n)} \, |0\rangle_2^{(n)} \cdots |0\rangle_q^{(n)} + \beta \, |1\rangle_1^{(n)} \, |1\rangle_2^{(n)} \cdots |1\rangle_q^{(n)} \tag{2.74}$$

We can encode a parity qubit by trying fusing $|0\rangle \, |0\rangle_1^{(n)} \cdots |0\rangle_q^{(n)} +$
$|1\rangle \, |1\rangle_1^{(n)} \, |1\rangle_2^{(n)} \cdots |1\rangle_q^{(n)}$ resource state onto parity qubit $|\psi\rangle^{(n)}$ until it is a success, giving:

$$\begin{aligned} \alpha \, ( \, |0\rangle^{(n-k)} \, |0\rangle_1^{(n)} \cdots |0\rangle_q^{(n)} + |1\rangle^{(n-k)} \, |1\rangle_1^{(n)} \cdots |1\rangle_q^{(n)} + \beta \, \Big( |1\rangle^{(n-k)} \, |0\rangle_1^{(n)} \cdots |0\rangle_q^{(n)} \\ + |0\rangle^{(n-k)} \, |1\rangle_1^{(n)} \cdots |1\rangle_q^{(n)} ) \end{aligned} \tag{2.75}$$

where $k$ is the number of unsuccessful tries such that $k \in (0, n-1)$. Notice here we have $qn$ new photons coming from the resource and $n-k$ old photons making parity

qubits. If we measure those old photons in the computational basis and then apply a bit-flip operation, then we have the desined encoded state.

**Loss tolerant qubit memory** Consider 2 qubits as an example. We take the logical qubit as memory for a time. Within that time, a photon loss may occur. Then we take the logical qubit out of memory and send one of its parity qubits, say $P2$, to the encoder which adds another level of redundancy encoding to our logical qubit and make a measurement on $P2$ without destroying the state where the measurement tells if a photon has been lost in $P2$.

$$\text{The state after encoding: } |\psi\rangle_L = \alpha\,|0\rangle_1^{(n)}\,|1\rangle_2^{(n)}\,|1\rangle_3^{(n)} + \beta\,|1\rangle_1^{(n)}\,|0\rangle_2^{(n)}\,|0\rangle_3^{(n)} \quad (2.76)$$

Then the recovery is done by measuring the modes coming out of the encoder in diagonal basis which destroys the entanglement between the $P1$ and other qubits without destroying the logical value. Note that we get $P1$ even if a bit-flip occured in $P2$ but a phase shift may be necessary on $P1$ depending on the result of the measurement in diagonal basis. At last we encode $P1$, send it back to the memory and repeat the sequence.

This process is tolerant up to a photon loss per sequence but higher levels of tolerance can be achieved by increasing the redundancy code size in memory and generalizing the process. For example we can correct two photon loss with $3^{rd}$ qubit in 3 qubits by 3-qubit encoders.

**Threshold**

- Probability of parity qubit is encoded successfully without photon loss:

$$\mathcal{P}_{Q_s} = \sum_{i=1}^{n-1} \left(\frac{1}{2}\eta_1\eta_2\right)^i \eta_i^{n-i} \quad (2.77)$$

  where $n$ is the original parity qubit size, $\eta_1$ is the probability of finding an old photon that is $\eta_1 = \eta_d\eta_s\eta_m$ with $\eta_d$: detector efficiency, $\eta_s$: source efficiency, $\eta_m$:memory efficiency and $\eta_2$ is the probability of detecting a new photon that is $\eta_2 = \eta_d\eta_s$

- Probability of complete failure: All parity qubit photons lost without a successful disentanglement caused by a series of fusion failures and photon losses.

$$\mathcal{P}_{ff} = \sum_{j=1}^{n-1} \left(\frac{1}{2}\eta_1\eta_2\right)^{j-1} (1-\eta_1\eta_2)(1-\eta_1)^{n-j} + \left(\frac{1}{2}\eta_1\eta_2\right)^{n-1} (1-\eta_1) + R\sum_{j=0}^{n-2} \left(\frac{1}{2}\eta_1\eta_2\right)^{j+1} \sum_{k=0}^{n-2-j} \eta_1^k (1-\eta_1)^{n-1-j-k} \tag{2.78}$$

where $R = \sum_{k=1}^{q} \binom{q}{k} (1-\eta_2)^{kn}[1-(1-\eta_2)^n]^{q-k}$ which is the term for decoupling failures (as well as for new parity qubits)

- Probability of a photon loss in one of the encoding parity qubits but successfully destroying the entanglement between that qubit and others in the redundancy code:

$$\mathcal{P}_{Q_f} = 1 - \mathcal{P}_{Q_s} - \mathcal{P}_{ff} \tag{2.79}$$

Now to find the threshold for the memory, we should consider two cases that circuit succeeds:

i) one parity qubit is successfully encoded without a photon loss and successfully disentangles: $\mathcal{P}_{Q_s}[1 = (1-\eta_1)^n]$

ii) a photon loss occurs in a parity qubit but successfully disentangles (such that we encoded again different parity qubit with success) with probability $\mathcal{P}_{Q_f}\mathcal{P}_{Q_s}$.

Therefore, probability of one successful memory circuit sequence for $q$ parity qubits:

$$\mathcal{P}_E = \sum_{j=0}^{q-1} \mathcal{P}_{Q_f}^i \mathcal{P}_{Q_s}[1-(1-\eta_1)^m]^{q-1-j} \tag{2.80}$$

As far as encoding is concerned, there are two methods which we can use, the concatenation approach or the incremental approach.

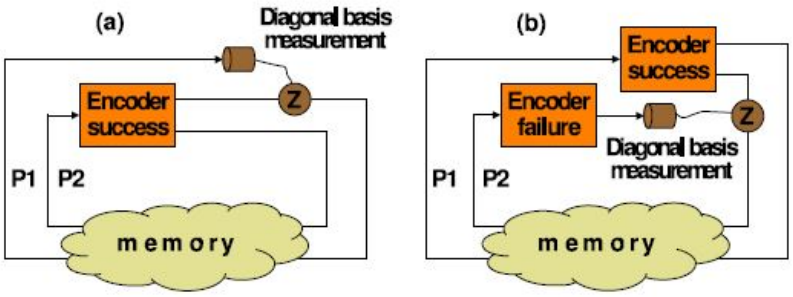



Figure 2.1: memory circuit from [4]

**Concatenation approach** In [8] it is argued that teleporter $T_{n/n+1} \ni n \in \mathbb{Z}$ with success probability $\frac{n}{n+1}$ needs a large successful entanglement resources for each qubit. Generally entanglement resource for teleportation:

$$|t_n\rangle = \frac{1}{\sqrt{n+1}} \sum_{j=0}^{n} |1\rangle^{\otimes j} |0\rangle^{\otimes (n-j)} |0\rangle^{\otimes j} |1\rangle^{\otimes (n-j)} \tag{2.81}$$

In the case of teleportation failure, we can think of it as the $Z$-measurement on the supposedly teleported qubit. So, we can increase the success rate of teleportation by using QEC protecting against $Z$-measurements. As an example, consider 2-qubit encoding:

$$|\mathbf{0}\rangle \to \frac{1}{\sqrt{2}}(|\mathbf{00}\rangle + |\mathbf{11}\rangle) \quad \& \quad |\mathbf{1}\rangle \to \frac{1}{\sqrt{2}}(|\mathbf{01}\rangle + |\mathbf{10}\rangle) \tag{2.82}$$

Therefore arbitrary $|\psi\rangle = \alpha |\mathbf{0}\rangle + \beta |\mathbf{1}\rangle$ becomes:

$$|\psi\rangle \to |\psi\rangle^{(2)} \frac{1}{\sqrt{2}}(\alpha(|\mathbf{00}\rangle_{AB} + |\mathbf{11}\rangle_{AB}) + \beta(|\mathbf{01}\rangle_{AB} + |\mathbf{10}\rangle_{AB})) \tag{2.83}$$

Here we can correct a single $Z$-measurement error if we know on which qubit the error occured and the result of $Z$-measurement. If the $Z$-measurement of a qubit is "0", it means that the other qubit is in the originally unencoded state while "1" means that we need to apply $X$ gate to the other qubit (the qubit that $Z$-measurement haven't applied)

To encode more than two qubits, say $2n$-qubits, we concatenate as:

$$|\psi\rangle^{(2n)} \frac{1}{\sqrt{2}}(\alpha(|\mathbf{00}\rangle^{(n)} + |\mathbf{11}\rangle^{(n)}) + \beta(|\mathbf{01}\rangle^{(n)} + |\mathbf{10}\rangle^{(n)})) \tag{2.84}$$

We can see that the qubits required is doubled for each concatenation from equation 2.84. But note that, concatenation also decreases the probability of failure of applying a gate.

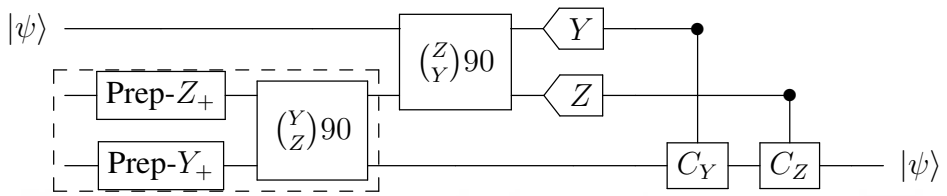
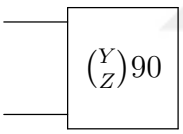



Figure 2.2: Teleportation Circuit [5]

Note that —— Prep- gate means the state preparation gate,
—— $Y$ means Pauli measurement, the block is the entanglement resource, donated by $|tx_n\rangle$,




is $(ZY)^{(12)}_{90^\circ}$ rotation. For example $(YZ)_{90^\circ} = e^{-\frac{i\pi}{4}Y\otimes Z}$

In this teleportation network, $C_Z = X_{180^\circ}$ if $S_2 = 1$ and $C_Y = Z_{180^\circ}$ if $S_1 = 1$ correcting possible $Z$-measure error may occur on the first qubit.

where $Z'$ is $(ZZZZ)_{90^\circ}$

To encode a state in KLM scheme [8], we need two non-deterministic gates $CS$ and $Z_{90}$ which can be created in an iterative manner in each concatenation level using basic gates which also needs iteration and resource states. We can find the probability of the recovery from $Z$-measurement error by calculating failure probability of teleporting a logical qubit with: [2]

$$F_z = \frac{f^2(2-f)}{1-f(1-f)} \tag{2.85}$$

where $F_z$ is the probability that $Z_{90}$ rotation fails and $f$ is the probability of teleporting one of the resource qubits failing.

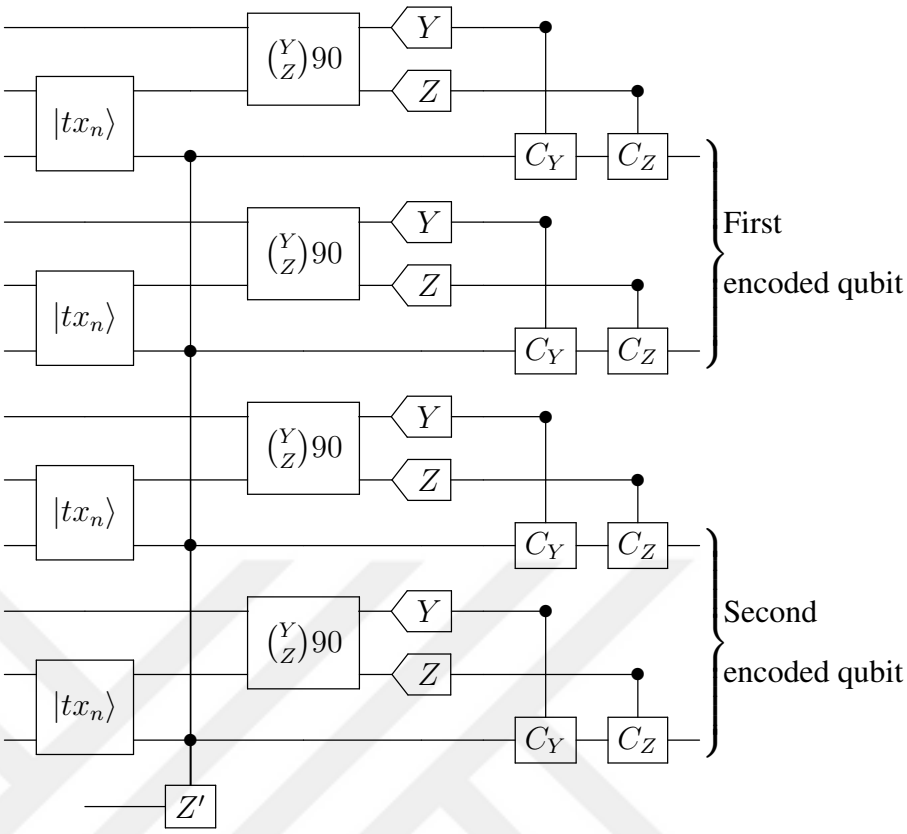



Figure 2.3: logical CS (controlled phase-flip) gate [5]

Then the success probability is simply $(1 - F_z)^2$ where we squared it due to the fact that we're using two encoded qubits. Notice that the success probability increases as the code is concatenated.

**Incremental approach** As an alternative to concatenation, [5] suggested that we can add qubits to the encoded state one by one. In this case, when a $Z$-measurement error occured, we remove one of the resource qubits out of the entanglement and correct the error by re-encoding via non-deterministic encoding circuit.

Going back to equation 2.66, we add an ancilla qubit in $(|\mathbf{0}\rangle + |\mathbf{1}\rangle)/\sqrt{2}$ to the encoded qubit as control and connect it to one of the physical qubits in the encoded qubit via CNOT gate.

Once we have a teleporter with sufficiently high success rate (since we're using non-

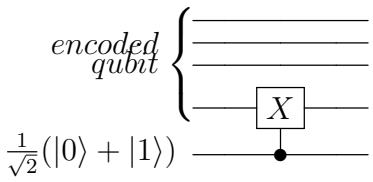



Figure 2.4: Incremental Circuit [5]

deterministic gate and it needs to be teleported) we can achieve high success rate by using this encoding and teleportation. Note that if we know one of the inputs of a gate (in this case the ancilla qubit), we can use it for resource preparation. Then we need one less teleporter which improves the success probability. If a $Z$-measurement occurs due to a fail in encoder gate, we loose one physical qubit and if not, we win one physical qubit. Therefore for a long-term gain, the success rate of the encoder must be higher than 50%.

We can calculate the probability of adding a new physical qubit to encoding by considering the attempt as a 1D lattice. When we set our beginning position as $m$ on lattice and $R$ and $L$ being boundaries corresponding adding a qubit and loosing all physical qubits in the encoding with probabilities $p$ and $(1-p)$ respectively. Arriving at $R$ before $L$ is:

$$\mathcal{P}_{R,m} = p\mathcal{P}_{R,m+1} + (1-p)\mathcal{P}_{R,m-1} \tag{2.86}$$

which has a trivial solution [5]

$$\mathcal{P}_{R,m} = \begin{cases} \frac{m-L}{R-L} & \text{if} \quad p = 1/2 \\ \frac{1-\beta^{m-L}}{1-\beta^{R-L}} & \text{if} \quad p \neq 1/2 \end{cases} \tag{2.87}$$

where $\beta = \frac{1-p}{p}$. Then, probability of adding a new qubit without loosing the encoding $\mathcal{P}_{add} \equiv \mathcal{P}_{R,0}$ with $L = -q$ and $R = 1$:

$$\mathcal{P}_{add} = \begin{cases} \frac{q}{q+1} & \text{if} \quad p = 1/2 \\ \frac{1-\beta^{q}}{1-\beta^{q+1}} & \text{if} \quad p \neq 1/2 \end{cases} \tag{2.88}$$

where $q$ is the number of physical qubits in the encoding in that instance.

Now consider $Z_{90}$ gate acting on a single encoded qubit. For this, first we add a qubit (with probability $\mathcal{P}_{add}$) followed by the gate which is deterministic and re-encode with probability $\mathcal{P}_{re}$. Starting from $m = 1$ with $L = 0$ and $R = 1$ that are loosing a qubit and successfully re-encoding to $q$ qubits we have:

$$\mathcal{P}_{re} = \begin{cases} 1/q & \text{if } \ p = 1/2 \\ \frac{1-\beta}{1-\beta^q} & \text{if } \ p \neq 1/2 \end{cases} \tag{2.89}$$

Requiring us, on average, $1/\mathcal{P}_{re}$ attempts. Note that since each failed attempt removes the qubit we added, we need to add new one, giving us the overall probability of success for single encoded-qubit gates as: $(\mathcal{P}_{add})^{1/\mathcal{P}_{re}}$

In CNOT case, we add a new phyiscal qubit to both control and target qubits, then attempt two teleportations with probability $\mathcal{P}_t$ (which is not necessarily be the same as the teleporters in the encoding with probability $p$). If any of those fail, we need to add a new qubit before again attempting the teleportation, if both successful, we only require the re-encoding of the target qubit (since the CNOT gate does nothing to the control). Both teleportations performed and re-encoding successfully is with probability $\mathcal{P}_t^2\mathcal{P}_{re}$ giving us the overall probability of success as $(\mathcal{P}_{add})^{1+1/(\mathcal{P}_t^2\mathcal{P}_{re})}$.

Assuming that all we apply is CNOT and with a lower bound on the number of non-deterministic gates performed gives us the total probability of computation with $n$ successful gates as:

$$\mathcal{P}_{tot} = \left[(\mathcal{P}_{add})^{1+1/(\mathcal{P}_t^2\mathcal{P}_{re})}\right]^n = (\mathcal{P}_{add})^{n+n/(\mathcal{P}_t^2\mathcal{P}_{re})} \tag{2.90}$$

In general,

$$n = \frac{\log(\mathcal{P}_{tot})}{(1 + \frac{1}{\mathcal{P}_t^2\mathcal{P}_{re}(q)})\log(\mathcal{P}_{add}(q))} \tag{2.91}$$

#### 2.2.3.1 [[9,1,3]] Shor Code

Suggested by Peter Shor in 1995 [6], one of the earliest codes, encodes as

$$|0\rangle \to \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle) = \frac{1}{2\sqrt{2}} |\overline{+}\rangle |\overline{+}\rangle |\overline{+}\rangle \equiv |\mathbf{0}\rangle$$
$$|1\rangle \to \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle) = \frac{1}{2\sqrt{2}} |\overline{-}\rangle |\overline{-}\rangle |\overline{-}\rangle \equiv |\mathbf{1}\rangle \tag{2.92}$$

In this method, we encode a qubit into 3-qubit circuit which is resilient against bit-flips and then encode those 3-qubits into another 3-qubit circuits with tolerance against phase-flips resulting a circuit with 9 physical qubits generating 1 logical qubit which is reboust against bit and phase flip errors.

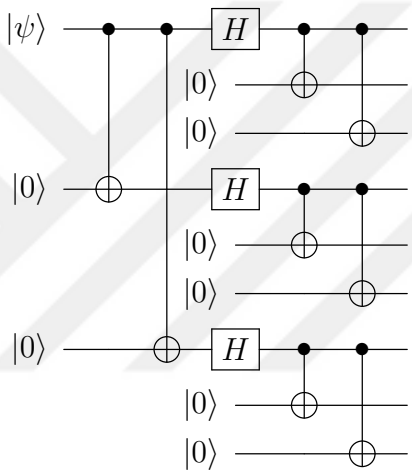



Figure 2.5: Shor's 9 qubit encoding circuit

To illustrate how code works, suppose there is a bit-flip error on the $1^{st}$ qubit of the $3^{rd}$ qubit group which is initially in $|+++\rangle = |\mathbf{0}\rangle$ state. After bit-flip we have $|{+}{+}\rangle \frac{1}{\sqrt{2}}(|100\rangle + |011\rangle)$. To detect the syndrome, we apply:

$$A_i \equiv x_1 \oplus x_2 \quad \text{and} \quad B_i \equiv x_1 \oplus x_3 \quad \forall i \in [1,3] \tag{2.93}$$

where we apply them in the $i^{th}$ group. Then we have $A_1 = x_1 \oplus x_2 = 1$, $B_1 = x_1 \oplus x_3 = 1$ which implies that there exists a bit-flip in $1^{st}$ qubit. Then we can simply correct it via $X$ operator.

Now assume we observe a phase-flip on $2^{nd}$ qubit group resulting state to evolve into

$|+-+\rangle$. In this case, we apply $A_i \equiv z_1 \oplus z_2$ and $B_i = z_1 \oplus z_3$ $\forall i \in [4, 6]$ where we consider $|-\rangle$ as 0 and $|+\rangle$ as 1. Then syndrome measurements yield: $A_4 = 1$ and $B_4 = 0$ pointing a phase-flip in $2^{nd}$ group.

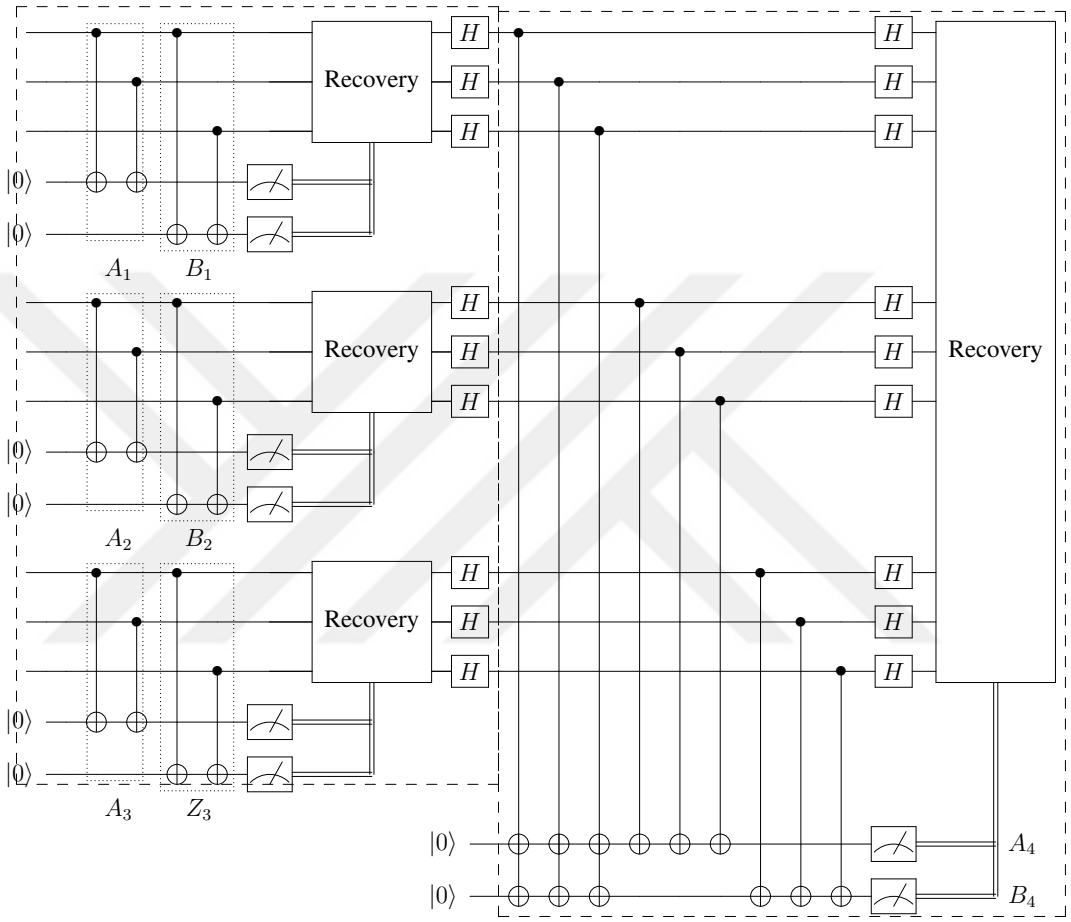



Figure 2.6: Shor's code encoding in detail

Note that the left block of the code detects and corrects bit-flip errors while the right block detects and corrects phase-flip errors. To decode, we simply reverse the encoding: [10]

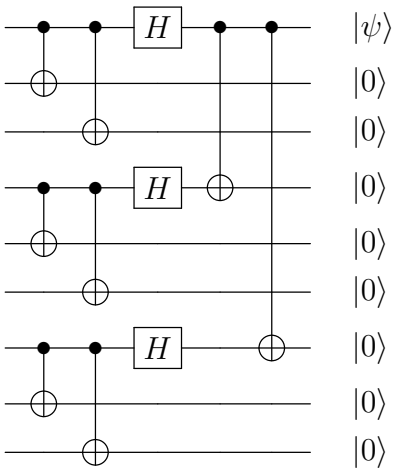



Figure 2.7: Shor's code decoding [10]

#### 2.2.3.2 Quantum Repetition Code

Quantum repetition codes are not actual quantum error correction codes due to the fact that they are not protective against phase and bit-flip errors at the same time. They can be use to protect against one leaving the quantum information vulnerable to the other. But without any concern for correcting errors, it can be used efficiently for detecting quantum errors. [7]

In the simplest case of 3-qubit bit-flip repetition code, we encode the quantum state $|\psi_{in}\rangle = \alpha|0\rangle + \beta|1\rangle$ as $|\psi\rangle = \alpha|000\rangle + \beta|111\rangle$. By applying projective measurement of $Z_1Z_2$ and $Z_2Z_3$ stabilizer operators, we can detect the bit-flip syndromes with majority-vote and correct the faulty qubit by applying $X$ gate. Since the most time-consuming and faulty step of QEC is the measurement step experimentally, it may be useful to replace it with a 3-qubit Toffoli gate. [3]

In the phase-flip case of 3-qubit repetition code, we encode the quantum state as $|\psi\rangle = \alpha|---\rangle + \beta|+++\rangle$. Line in bit-flip code but for phase errors, we make syndrome measurements with $X_1X_2$ and $X_2X_3$, and then correct the faulty qubit by

$Z$ operator.

Physical realization of quantum memory based on repetition code is the following: At the beginning quantum information is hold in one qubit and the other two are in the ground state. Then by CNOT gates, we make encoding. If we are dealing with phase-flip repetition code, encoding also requires the application of the Hadamard gates to each qubit. After encoding, errors occur during the waiting time. Again if it is phase-flip case, we apply Hadamards to each qubit followed by two CNOTs and finally to correct errors, we apply Toffoli gate.

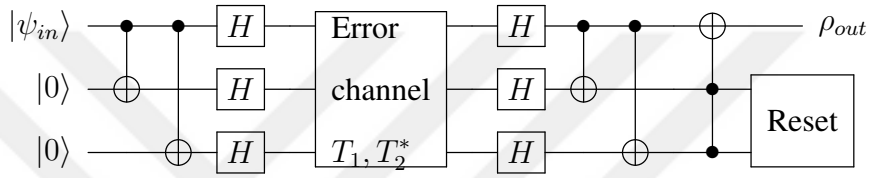



Figure 2.8: 3-qubit phase flip code with 1-qubit and CNOT gates [3]

where

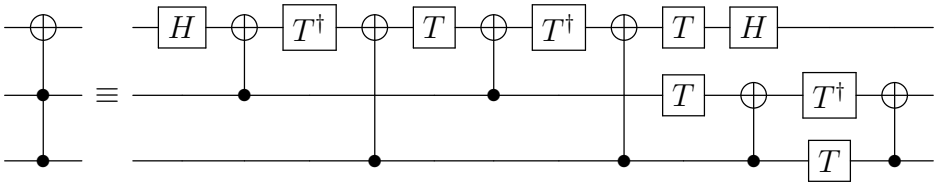



Figure 2.9: CCX (Toffoli) gate [3]

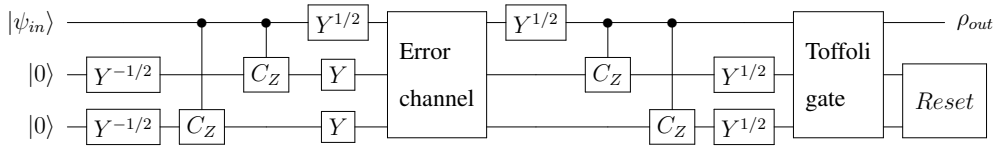



Figure 2.10: 3-qubit phase flip code with 1-qubit, CS and controlled-$S^{-1}$ gates [3]

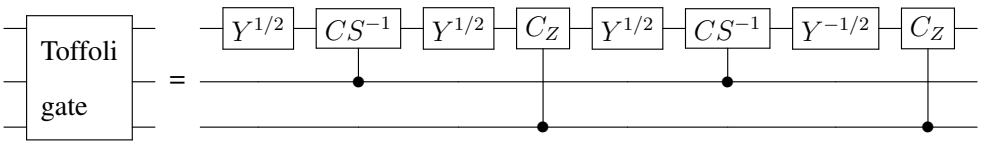



Figure 2.11: Toffoli gate [3]

In recent years, technological advancements in quantum dot qubits (superconducting qubits) promise overcoming the dephasing errors while leaving relaxation error as the dominant form of error. For this reason, we may look after the energy relaxation using repetetive codes as quantum error detection because energy relaxation can be thought of combinations of bit-flip ($X$-rotation) and bit-phase-flip ($Y$-rotation) errors which repetition codes can't protect against bot of these. [7]
To discuss $N$-qubit quantum repetetion code on zero temperature energy relaxation, let's simplify the discussion as follows:

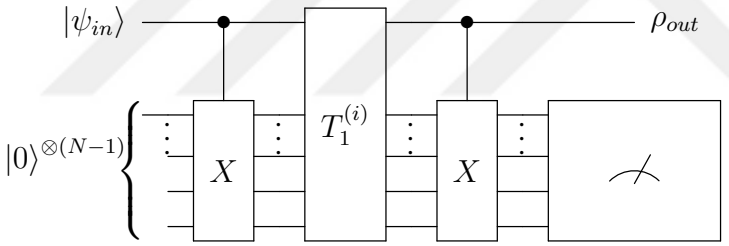



Figure 2.12: N-qubit Quantum Repetition Circuit [7]

Note that $T_1^{(i)}$ represents the zero-temperature energy relaxation happening in the $i^{th}$ qubit ($\ni\ i = 1, \cdots, N$) assuming that all qubits experience the same decoherence such that $T_1^{(i)} = T_1$, and $X$ block represents the block of CNOT gates.
Here, we encode the input state using $N-1$ number of CNOT gates giving us the encoded state $\alpha\,|0\rangle^{\otimes N} + \beta\,|1\rangle^{\otimes N}$ which then suffers relaxation $T_1^{(i)}$ in the $i^{th}$ qubit. Decoding via $N-a$ CNOT gates gives us $(\alpha\,|0\rangle + \beta\,|1\rangle)\,|0\rangle^{\otimes(N-1)}$ where we can detect errors by looking at the ancilla qubits $|0\rangle^{\otimes(N-1)}$. Notice without any error, we observe $|0\rangle^{\otimes(N-1)}$ as we measure ancillas in the computational basis, but if we see states being different than $|0\rangle$ means that there exists error. We can make correction and make $\rho_{out}$

closer to $|\psi_{in}\rangle$ or discard the non-$|0\rangle$ realizations and keep measurements with 0 result what we call *quantum error detection (QED)*. [7]

We can think of the encoding as the number of logical qubits as single qubit, two-qubits and N-qubits scenario.

**Single qubit encoding** After a time $t$, $|\psi_{in}\rangle$ evolves into:

$$\rho_{f_{in}} = \begin{pmatrix} |\beta|^2 e^{-t/T_1} & \alpha^*\beta e^{-t/2T_1} \\ \alpha\beta^* e^{-t/2T_1} & |\alpha|^2 + |\beta|^2(1-e^{-t/T_1}) \end{pmatrix} = A_r\rho_{in}A_r^\dagger + A_n\rho_{in}A_n^\dagger \tag{2.94}$$

where

$$A_r = \begin{pmatrix} 0 & 0 \\ \sqrt{p} & 0 \end{pmatrix}, \quad A_n = \begin{pmatrix} \sqrt{1-p} & 0 \\ 0 & 1 \end{pmatrix}, \quad p = 1-e^{-t/T_1}, \quad \rho_{in} = |\psi_{in}\rangle\langle\psi_{in}|$$

Note that the Kraus operators satisfy completeness such that $A_r^\dagger A_r + A_n^\dagger A_n = \mathbb{I}$ and if we look at the RHS of equation 2.94 what wee see are that left term is the relaxation with probability $\mathcal{P}_r = |\beta|^2 p$ while right term is the case of no relaxation with probability $\mathcal{P}_n = |\alpha|^2 + |\beta|^2(1-p) = 1-\mathcal{P}_r$ Then if no relaxation occurs, state evolves into:

$$|\psi_n\rangle = \frac{A_n|\psi_{in}\rangle}{\sqrt{\mathcal{P}_n}} = \frac{1}{\sqrt{\mathcal{P}_n}}\left(\alpha|0\rangle + \beta\sqrt{1-p}|1\rangle\right) \tag{2.95}$$

**Two qubit encoding** As two qubit encoding, we have $\alpha|00\rangle + \beta|11\rangle$. Then 2 qubits after time $t$ we have:

- no relaxation:
  $\frac{1}{\sqrt{\mathcal{P}_{nn}}}\left(\alpha|00\rangle + \beta\sqrt{1-p_1}\sqrt{1-p_2}|11\rangle\right)$ with probability $\mathcal{P}_{nn} = |\alpha|^2 + |\beta|^2(1-p_1)(1-p_2)$

- relaxation in the input qubit:
  $|01\rangle$ with probability $\mathcal{P}_{rn} = |\beta|^2 p_1(1-p_2)$

- relaxation in the ancilla:
  $|10\rangle$ with probability $\mathcal{P}_{nr} = |\beta|^2(1-p_1)p_2$
- relaxation in bath qubits:
  $|00\rangle$ with probability $\mathcal{P}_{rr} = |\beta|^2 p_1 p_2$

where $p_1 = 1 - e^{-t/T_1^{(1)}}$ and $p_2 = 1 - e^{-t/T_1^{(2)}}$. After decoding using a CNOT:

- $\frac{1}{\sqrt{\mathcal{P}_{nn}}}(\alpha|0\rangle + \beta\sqrt{1-p_1}\sqrt{1-p_2}|1\rangle) \otimes |0\rangle$ with probability $\mathcal{P}_{nn}$
- $|01\rangle$ with probability $\mathcal{P}_{rn}$
- $|10\rangle$ with probability $\mathcal{P}_{nr}$
- $|00\rangle$ with probability $\mathcal{P}_{rr}$

**N-qubit encoding** In $N$-qubit scenario: we encode as $\alpha|0\rangle^{\otimes N} + \beta|1\rangle^{\otimes N}$. Within $w^N$ possible outcomes, $2^{N-1}$ of them (due to measurement) correspond to either no relaxation in the input qubit (in which measurement results show the relaxed ancilla qubit) or relaxation in the input qubit (in which measurement results show 0). In all 0 measurement outcome cases we either have a main qubit with no relaxation:

$$|\psi_{none}\rangle = \frac{1}{\sqrt{\mathcal{P}_{none}}}(\alpha|0\rangle + \beta|1\rangle \prod_{j=1}^{N}\sqrt{1-p_j}) \tag{2.96}$$

where $\mathcal{P}_{none} = |\alpha|^2|\beta|^2\prod_{j=1}^{N}(1-p_j)$ is the probability corresponding to the no relaxation case,
or in $|0\rangle$ state in which all qubits relax with probability:

$$\mathcal{P}_{all} = |\beta|^2 \prod_{j=1}^{N} p_j \tag{2.97}$$

Note that, in [7] it is argued that 2 qubit encoding is enough for QED purposes.

### 2.2.4 Quantum Reed-Muller Codes

Let classical codes $C_1 = [n, k_1, d]$ and $C_2 = [n, k_2, d]$ with $C_2^{\perp} < C_1$ produces $[[n, k_1 + k_2 - n, d]]$ code using the optimum quantum code set that elements can

correct single error which Gottesman found [16] (that are $[[2^r, 2^r - r - 2, 3]]$ codes). Combine the classical Reed-Muller codes [17] we have:

$$[[n, k, d]] = [[2^r, 2^r - C(r,t) - 2\sum_{i=0}^{t-1} C(r,i), 2^t + 2^{t-1}]] \tag{2.98}$$

where $C(r,t) = \frac{r!}{t!(r-t)!}$. Consider the following:

$$G' = \left(\begin{array}{c|c} G_1 & 0 \\ 0 & G_2 \end{array}\right) \quad \text{and} \quad H = \left(\begin{array}{c|c} H_2 & 0 \\ 0 & H_1 \end{array}\right) \tag{2.99}$$

where $G_1$ and $G_2$ are generators of the classical codes $C_1$ and $C_2$ respectively. At this point, [18] suggests adding more rows $\mathtt{D}$ to $G_1$ such that they generate a classical code together with a minimum distance smaller than what $G_1$ alone generates and introduces a systematic way of changing signs of the code vectors resulting a generator in the following structure:

$$G' = \left(\begin{array}{c|c} G_1 & 0 \\ 0 & G_2 \\ \mathtt{D_x} & \mathtt{D_z} \end{array}\right) \tag{2.100}$$

where $\mathtt{D}$ stands for displacement matrix [18]

To have a quantum code with parameters in equation 2.98, consider the next in equation 2.100:

$$G_1 = G_2 = [2^r, k_{RM}(t,r), 2^{t+1}] \tag{2.101}$$

which is Reed-Muller code where $k_{RM}(t,r) = 2^r - \sum_{i=0}^{t} C(r,i)$.

Choose $\mathtt{D_x}$ in such a way that together with $G_1$, they produce the Reed-Muller code with distance $2^t$ and size $k_{RM}(t-1,r)$. Then quantum generator has number of rows in total:

$$n + k = k_{RM}(t,r) + k_{RM}(t-1,r) \tag{2.102}$$

deciding the $k$ in equation 2.98. Now, constructing $\mathtt{D_z}$ to increase the minimum distance to $2^t + 2^{t-1}$ from $2^t$ (which is the case when $\mathtt{D_z}$ is zero) such that:

$$\begin{aligned} {\mathtt{D_z}}^i &= {\mathtt{D_x}}^{i+1} \quad \text{if} \quad 1 \leq i < m \\ {\mathtt{D_z}}^m &= L_t({\mathtt{D_x}}^1) \end{aligned} \tag{2.103}$$

where $m = k_{RM}(t-1, r) - k_{RM}(t, r)$, superscript $j$ labels the $j^{th}$ row of the matrix and $L_t$ is rotating the bit sting to the left by $t$ places. (e.g. $L_3(00101101) = 01101001$)

As a result, since we constructed the generator matrix $G'$, we have the quantum code.

**some examples:**

- r=2, t=1 case:

$$\mathcal{G} = \left(\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \end{array}\right) \tag{2.104}$$

- r=3, t=1 case ([[8,3,3]] code): [78][79][80]

$$G'_{[[8,3,3]]} = \left(\begin{array}{cccccccc|cccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{array}\right) \tag{2.105}$$

$$H_{[[8,3,3]]} = \left(\begin{array}{cccccccc|cccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \end{array}\right) \tag{2.106}$$

- We can also obtain [[5,1,3]] perfect code by erasing a bit from [[6,0,4]] code which is

$$G'_{[[6,0,4]]} = \left(\begin{array}{cccccc|cccccc} 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \end{array}\right) \tag{2.107}$$

Note that, for $k = 0$, method fails. [18].

[19] showed that the code rates of a sequence of binary linear codes converge to an $r \in \{0, 1\}$, if its block lengths are strictly increasing, proving the Reed-Muller codes achieve capacity in the case of binary erasure channel under both bitwise *maximum-a-posteriori (MAP)* and block-MAP.

Also, [20] generalized the magic state distillation, which is a part of fault-tolerant computing, using quantum Reed-Muller codes for all prime dimensions. This allows us to prepare highly purified nonstabilizer states for a device with capability of carrying ideal stabilizer operations.

#### 2.2.4.1 [[$2^r - 1, 1, 3$]] Quantum Reed-Muller codes

We start the discussion of this topic by considering the conjecture offered by [21] which states that if any two stabilizer states are equivalent under local unitary (LU) operations, then they are also equivalent under local Clifford (LC) operations. Where if there is an LU operation $\mathcal{U}_n = \bigotimes_{i=1}^{n} U_i$ (mapping $|\psi'\rangle$ to $|\psi\rangle$) then the $n$-qubit states $|\psi\rangle$ and $|\psi'\rangle$ are local unitary equivalent: while if there exists an LU operation in the Clifford group $\mathcal{K}_n = \bigotimes_{i=1}^{n} K_i$ (mapping $|\psi'\rangle$ to $|\psi\rangle$ with $K_i \in \mathcal{L}_1$ for $i = 1, \cdots, n$),

then $|\psi\rangle$ and $|\psi'\rangle$ are local Clifford (LC) equivalent. [23]

Note that it has been shown in [22] that any stabilizer state is equivalent to some graph state under LC operations. Here what graph state is a kind of stabilizer state which is associated with graphs [25]. Suppose a graph $\mathcal{G}$ which has only two elements that are *vertices* (V) and *edges* (E). Each edge connects two endpoints (that are in the vertices' set) and the degree of a vertex is the number of edge endings on the given vertex. If there is a connection from each vertex to the next in the sequence by an edge, then we call that sequence a *path*, and if the starting and ending vertices of that path are the same it is called *cycle* whose length is the number of edges within that cycle.

Then, $\forall \mathcal{G}$ with vertex number $n$, $\exists n$ number of operators

$$R_a^{\mathcal{G}} \in \mathbb{P}_n \;\; \forall a = 1, 2, \cdots, n \;\; \ni \;\; R_a^{\mathcal{G}} = X_a \bigotimes_{\{a,b\} \in E} Z_b \tag{2.108}$$

Notice that they commute allowing the group generated by $\{R_a^{\mathcal{G}}\}_{a=1}^n$ is a stabilizer group $\mathcal{S}$ (stabilizing unique state $|\psi_{\mathcal{G}}\rangle$) and each $R_a^{\mathcal{G}}$ is called the standard generator associated with vertex $a$ of graph $\mathcal{G}$.

**Minimal Support Condition** The support of an element $R$ ($supp(R) \ni R \in \mathcal{S}(|\psi\rangle)$ is the set $\{R_i\}/\mathbb{I} \;\; \forall i \in \{1, \cdots, n\}$. Let $\omega = \{i_1, \cdots, i_k\} \subset \{1, \cdots, n\}$. Then a minimal support of $\mathcal{S}(|\psi\rangle)$ is a set $\omega \subseteq \{1, \cdots, n\} \ni \exists x \in \mathcal{S}(|\psi\rangle)$ with $supp(\omega)$, but there is no element such that its support is in $\omega$ strictly. This element $x$ is *minimal element*. Note that $A_\omega(|\psi\rangle)$ (the number of elements $R$) in $\mathcal{S}(|\psi\rangle)$ whose support is $\omega$) and we denote the subgroup $\mathcal{S}(|\psi\rangle)$ which is generated by the all minimal elements with $\mathcal{M}(|\psi\rangle)$. [23]

**Lemma 2.2.4** *Let $|\psi\rangle$ be a stabilizer state with $supp_{min}(\mathcal{S}(|\psi\rangle)) = \omega$. Then, $A_\omega(\mathcal{S}(|\psi\rangle)) = 1$, or 3 if $|\omega|$ is even. [21]*

$$\begin{aligned} A_\omega(\mathcal{S}(|\psi\rangle)) = 1 \quad &: \quad Z^{\otimes\omega} \\ A_\omega(\mathcal{S}(|\psi\rangle)) = 3 \quad &: \quad \{X^{\otimes\omega}, (-1)^{|\omega|/2} Y^{\otimes\omega}, Z^{\otimes\omega}\} \end{aligned} \tag{2.109}$$

Also, [24] proved that if $|\psi\rangle$ and $|\psi'\rangle$ are stabilizer states that are LU equivalent (i.e. $\mathcal{U}_n |\psi'\rangle = |\psi\rangle$), then the equivalence $\mathcal{U}_n$ must map the group generated by the all $\omega = supp_{min}(\mathcal{S}(|\psi\rangle))$ to the group that is corresponding and generated by the all minimal elements of $\omega = supp_{min}(\mathcal{S}(|\psi\rangle))$ for all minimal support $\omega$. Which yields [21]:

**Theorem 2.2.5** *Every stabilizer state $|\psi\rangle$, the LU equivalent to $|\psi'\rangle$, must also be LC equivalent to $|\psi'\rangle$ if $|\psi\rangle$ is a fully entangled stabilizer state with Pauli-X,Y,Z matrices occuring on every qubit in $\mathcal{M}(|\psi\rangle)$. [23]*

So, the condition that all Pauli-$X, Y, Z$ matrices occuring on every qubit $\mathcal{M}(|\psi\rangle)$ in theorem 2.2.5 is the *minimal support condition (MSC)*.

And if all Pauli-$X, Y, Z$ matrices occur on the $i^{th}$ qubit in n $\mathcal{M}(|\psi\rangle)$, then $U_i$ must be a Clifford operator where LU operation $\mathcal{U}_n = \bigotimes_{i=1}^{n} U_i$ mapping a stabilizer state $|\psi'\rangle$ to another stabilizer state $|\psi\rangle$. As a result, if MSC is satisfied for $|\psi\rangle$, then $\mathcal{U}_n$ is also an LC operator. [23]
[21] also showed that LU $\iff$ LC holds for $n$-GHZ states eventhough they don't have that structure.

**Theorem 2.2.6** *For any graph $\mathcal{G}$ with cycles of length neither 3 nor 4 holds LU $\iff$ LC equivalence for distance $\delta > 2$. [23]*

So, let's define the vertices we use for a graph $\mathcal{G}$:

- $V_1(\mathcal{G})$ = degree-1 vertices of $\mathcal{G}$
- $V_2(\mathcal{G}) = \{v | v$ connects to some $\omega \in V_1(\mathcal{G})\}$
- $V_3(\mathcal{G}) = \{v | v \notin V_1(\mathcal{G})$ and $v$ connects only to $\omega \in V_2(\mathcal{G})\}$
- $V_4(\mathcal{G}) = V(\mathcal{G})/(V_1(\mathcal{G}) \cup V_2(\mathcal{G}) \cup V_3(\mathcal{G}))$

In the figure above, we have partitions $V_1(A3) = \{7, 8, 9, 11, 12, 13\}$, $V_2(A3) = \{1, 4, 6, 10\}$, $V_3(A3) = \{5\}$, $V_4(A3) = \{2, 3\}$ and $V_1(B3) = \{10\}$, $V_2(B3) = \{3\}$,

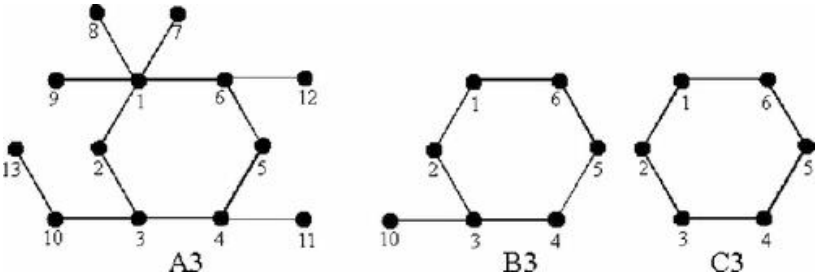



Figure 2.13: Partition example [23]

$V_3(B3) =$ , $V_4(B3) = \{1, 2, 4, 5, 6\}$ but $V_1(C3) = V_2(C3) = V_3(C3) =$ , $V_4(C3) = V(C3) = \{1, 2, 3, 4, 5, 6\}$ since $C3$ is a $\delta = 3$ graph.
Note that all $\delta = 2$ graphs are beyond MSC [23]

**The standard procedure** The case when $V_1 \cup V_2$. Our purpose have to convert the stabilizer states $|\psi_{\mathcal{G}}\rangle$ and $\left|\psi'_{\mathcal{G}}\right\rangle$ (which are LU-equivalent) into LC equivalent canonical forms. Then we go backwards from canonical forms to prove LU $\Longleftrightarrow$ LC for $|\psi_{\mathcal{G}}\rangle$. This procedure contains five steps:

i) Using LC operation, transforming into a new basis

ii) Encoding into repetition codes

iii) Showing $U_L \in \mathcal{L}_1$

iv) Construct a logical LC operation which relates $\left|\overline{\psi_{\mathcal{G}}}\right\rangle$ and $\left|\overline{\psi'_{\mathcal{G}}}\right\rangle$

v) Decoding $U_L^{(a)}$ in order to construct $K_n$

Notice that LU $\Longleftrightarrow$ LC holds since $|\psi_{A4}\rangle$ is a GHZ state. And note that the stabilizer for A4 has generator $\{XZ\mathbb{I}, ZXZ, \mathbb{I}ZX\}$.

[23] also provides the full algorithm to construct local Clifford operator $\mathcal{K}_n$ which corresponds to the given local unitary operation $\mathcal{U}_n$, that gives output of an LC operation for $\mathcal{K}_n$ (where $\mathcal{K}_n = \bigotimes_{i=1}^{n} K_i \ \ni \mathcal{K}_n \left|\psi'_{\mathcal{G}}\right\rangle = |\psi_{\mathcal{G}}\rangle$). And the inputs are: a stabilizer state $\left|\psi'_{\mathcal{G}}\right\rangle$, a graph $\mathcal{G}$ which is connected with no cycles of length either 3 or 4 and an LU operation $\mathcal{U}_n$ where $\mathcal{U}_n = \bigotimes_{i=1}^{n} U_i \ \ni \mathcal{U}_n \left|\psi'_{\mathcal{G}}\right\rangle = |\psi_{\mathcal{G}}\rangle$.

- To begin with, we make the partition of $V(\mathcal{G})$ into $V_1, V_2, V_3$ and $V_4$.

- Then we choose $K_i = U_i \; \forall i \in V_3 \cup V_4$

- Apply followings $\forall v_2 \in V_2$:

  - We calculate $B_{v_2} = U_{v_2}^\dagger Z_{v_2} U_{v_2}$ and find any $F_{v_2} \in \mathcal{L}_1 \;\ni F_{v_2} B_{v_2} F_{v_2}^\dagger = Z_{v_2}$ followed by calculating $\tilde{U}_{v_2} = U_{v_2} F_{v_2}^\dagger$.
    We find $\{\omega_1, \cdots, \omega_k\} \subset V_1 \ni \{\omega_j, v_2\} \in E(\mathcal{G}) \forall j \in [1, k]$ where $E(\mathcal{G})$ is the set of edges of graph $\mathcal{G}$.

  - Then for $j = 1, \cdots, k$:
    We find any $F_{\omega_j} \in \mathcal{L}_1 \ni F_{\omega_j} B_{\omega_j} F_{\omega_j}^\dagger = Z_{\omega_j}$ and calculate $\tilde{U}_{\omega_j} = H_{\omega_j} U_{\omega_j} F_{\omega_j}^\dagger$

  - Now we check if $\tilde{U}_{v_j}$ is diagonal or not.

    * If it is diagonal, we calculate $\tilde{K}_{v_2} = \tilde{U}_{v_2} \times \tilde{U}_{\omega_1} \times \cdots \times \tilde{U}_{\omega_k}$ where $\times$ stands for the standard matrix multiplication in $SU(2)$.
      Then we set $\tilde{K}_{\omega_j} = \mathbb{I}_{\omega_j} \; \forall j$, $K_{v_2} = \tilde{K}_{v_2} F_{v_2}$ and $K_{\omega_j} = H_{\omega_j} \tilde{K}_{\omega_j} F_{\omega_j} \; \forall j$

    * If it is not diagonal, we calculate $\tilde{K}_{v_2} = \tilde{U}_{v_2} X_{v_2} \times \tilde{U}_{\omega_1} X_{\omega_1} \times \cdots \times \tilde{U}_{\omega_k} X_{\omega_k}$
      Then we set $\tilde{K}_{\omega_j} = X_{\omega_j} \; \forall j$, $K_{v_2} = \tilde{K}_{v_2} F_{v_2}$ and $K_{\omega_j} = H_{\omega_j} \tilde{K}_{\omega_j} F_{\omega_j}$

  - Finally what we get is:

$$\mathcal{K}_n = \bigotimes_{i=1}^{n} K_i \quad \ni \mathcal{K}_n \left|\psi'_{\mathcal{G}}\right\rangle = \left|\psi_{\mathcal{G}}\right\rangle \tag{2.110}$$

#### 2.2.4.2 $[[2^r - 1, 2^r - 2r - 1, 3]]$ Hamming-based CSS codes

Hamming codes are $[[2^r - 1, 2^r - 2r - 1, 3]]$ quantum error correction code family for $r \geq 3$ and they are self-dual perfect CSS codes using $[2^r - 1, 2^r - r - 1, 3]$ classical Hamming code and CSS structure, we construct this first-order punctured Reed-Muller code $RM(r-2, r)$ [26]. Contains codes like $[[7, 1, 3]]$ code, $[[15, 7, 3]]$ code, $[[31, 21, 3]]$ code etc. As a brief example, consider $[[15, 7, 3]]$ code for its parity-check matrix:

$$H_{[[15,7,3]]} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{2.111}$$

This code uses 15 physical qubits. Although this method tolerates less noise compared to $[[7, 1, 3]]$ or $[[9, 1, 3]]$ codes, it is more efficient. [27] also introduced two flag qubits (ancillas) as a fault-tolerance method with low qubit overhead. Take $[[15, 7, 3]]$ code again, adding flag qubits we have the circuit:

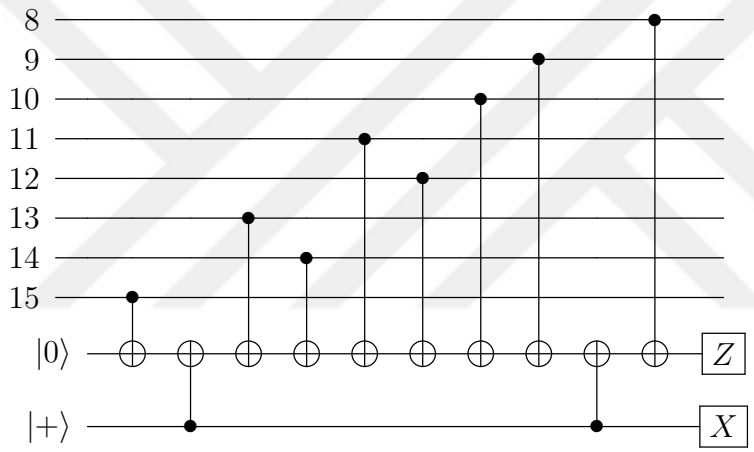



Figure 2.14: Flagged [[15,7,3]] Circuit [27]

For this, we take a subset $S$ and let $Z_S = \prod_{i \in S} Z_i$, then this circuit extracts $Z$ syndrome for $Z_{\{8,\cdots,15\}}$ by flag triggering due to single fault with one of the following errors:

$$\mathbb{I}, Z_8, Z_{\{8,9\}}, Z_{\{8,9,10\}}, Z_{\{8,9,10,12\}}, Z_{\{8,9,10,11,12\}}, Z_{\{8,9,10,11,12,14\}}, Z_{\{8,9,10,11,12,13,14\}} \tag{2.112}$$

And by their $X$ syndromes, those errors are distinguishable. Note that in what order we use CNOT gates is important. Take $[[7, 1, 3]]$ code:

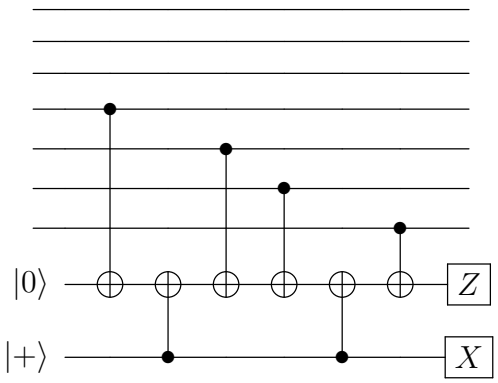



Figure 2.15: Flagged [[7,1,3]] Circuit [27]

Notice that this circuit extracts $\mathbb{III}ZZZZ$ syndrome in which any single gate fault causing data error of weight$\geq 2$ turns $X$ measurement to $|-\rangle$. Then possible errors are:

$$\mathbb{I}, Z_7, Z_6Z_7, Z_5Z_6Z_7 \sim Z_4 \tag{2.113}$$

Again distinguishable by their $X$ syndromes. Note that flag qubit makes the computation fault-tolerant.

### 2.2.5 Triorthogonal codes

**Definition 2.2.4** *A triorthogonal matrix is a binary matrix $G$ with size $m \times n$ iff*

$$\sum_{j=1}^{n} G_{a,j}G_{b,j} = 0 \mod (2) \quad \forall \textit{ row pairs } 1 \leq a < b \leq m \tag{2.114}$$

*and*

$$\sum_{j=1}^{n} G_{a,j}G_{b,j}G_{c,j} = 0 \mod (2) \quad \forall \textit{ row triples } 1 \leq a < b < c \leq m \tag{2.115}$$

*are satisfied.*

The submatrices of $G$ that are constructed with odd- and even-weight rows (meaning that the number of non-zero elements in the rows are odd and even respectively) will be denoted $G_{ii}$ and $G_i$ respectively. And we assume that $G_1$ contains first $k$ rows of $G$ $\forall k \geq 0$. Let $G', G'_i, G'_{ii} \subseteq \mathbb{F}_2^n$ spanned by the rows $G, G_i, G_{ii}$ respectively.

**Lemma 2.2.7** *Suppose $G$ is triorthogonal, then:*

*i)* $G_1$ *has linearly independent rows over* $\mathbb{F}_2$

*ii)* $G'_{ii} \cap G'_i = 0$

*iii)* $G'_{ii} = G' \cap G'^{\perp}$

*iv)* $G'^{\perp}_{ii} = G'_i \oplus G'^{\perp}$

For a matrix $G$ with $k$ odd-weight rows and $n$ columns sayisfying the equation 2.114, we can find a stabilizer code which encodes $k$ physical qubits into $n$ logical qubits. [28] Equation 2.115 ensures that the encoded state $\left|A^{\otimes k}\right\rangle$ can be prepared using transversal $T$-gate ($T^{\otimes n}$) on the encoded state $\left|+^{\otimes k}\right\rangle$. To see this, consider $n$-qubit unnormalized states:

$$|G_{ii}\rangle = \sum_{g \in G'_{ii}} |g\rangle \quad \& \quad |G\rangle = \sum_{g \in G'} |g\rangle \tag{2.116}$$

and define a state

$$\left|\overline{A^{\otimes k}}\right\rangle = \prod_{a=1}^{k} (\mathbb{I} + e^{i\pi/4} X(f^a)) \, |G'_{ii}\rangle \tag{2.117}$$

where $f_1, \cdots, f_k$ are rows of $G_1$.

**Lemma 2.2.8** *For a triorthogonal matrix $G$, one can find an operator $U$ in Clifford group containing $S$ gates and $\Lambda(Z)$ only, such that*

$$\left|\overline{A^{\otimes k}}\right\rangle = U T^{\otimes n} \, |G\rangle$$

**Lemma 2.2.9** *Suppose $G$ is a triorthogonal matrix with no zero columns. If $G_1$ is non-empty, then any $G_{ii}$ having 3 rows or less has one zero column or more.*

**Stabilizer codes based on triorthogonal matrices** For a triorthogonal matrix $G$ with $k$ odd-weight rows, we can define a CSS stabilizer code $CSS(X, G'_{ii}; Z, G'^{\perp})$. It has stabilizers $X(f)$ $\forall f \in G'_{ii}$ and $Z(g)$ $\forall g \in G'^{\perp}$. And $G'_{ii} \subseteq G'$ shows that stabilizers commute in pairwise.

**Lemma 2.2.10** *$CSS(X, G'_{ii}; Z, G'^{\perp})$ code has $k$ logical qubits and we can choose its logical Pauli operators as:*

$$\overline{X}_a = X(f^a) \quad \& \quad \overline{Z}_a = Z(f^a) \qquad \forall a = 1, \cdots, k \tag{2.118}$$

*where $f^j$ is the $j^{th}$ row of $G_i$. The states defined in 2.116 and 2.117 $|G\rangle, |G_{ii}\rangle$ and $\left|A^{\otimes k}\right\rangle$ correspond to the encoded states $\left|+^{\otimes k}\right\rangle$, $\left|0^{\otimes k}\right\rangle$ and $\left|A^{\otimes k}\right\rangle$ respectively.*

The encoded state $|x\rangle \equiv |x_1, \cdots, x_k\rangle \ \forall x \in \mathbb{F}_2^k$ is

$$|\overline{x}\rangle = \overline{X}_1^{x_1} \cdots \overline{X}_k^{x_k} |G_0\rangle = \sum_{f \in G_0 + x_1 f^1 + \cdots + x_k f^k} |f\rangle \tag{2.119}$$

using $UT^{\otimes n} |f\rangle = \exp\left(\frac{i\pi}{4} \sum_{a=1}^{k} x_a\right) |f\rangle$ we get:

$$UT^{\otimes n} |\overline{x}\rangle = \exp\left(\frac{i\pi}{4} \sum_{a=1}^{k} x_a\right) |\overline{x}\rangle \tag{2.120}$$

which is a generalization of transversal $T$-gate to multiple logical qubits.
Triorthogonal codes are useful since we use them in magic state distillation. In [29] triorthogonal codes are divided into 38 subspaces for $n+k \leq 38$ and shown that there exists a relation between each triorthogonal code and a Reed-Muller polynomial with weight $n + k$.

### 2.2.6 H codes

Family of $[[n, n-4, 2]]$ CSS quantum stabilizer codes encoding even number of $k$ logical qubits using $k+4$ physical qubits and having transversal Hadamard operation. We denote H codes as $H_n$ where $n = k + 4$ physical qubits. The encoding rate $\eta_R = k/n \to 1$ as $n \to \infty$, so H codes are dense. [30] demonstrated that we can use concatenated H codes to distill encoded magic states with high-fidelity by consuming magic-state ancillas with low-fidelity (a procedure called multilevel distillation).

The stabilizer generators of H code are:

$$\begin{aligned} S_1 &= X_1X_2X_3X_4, \quad S_3 = X_1X_2X_5X_6 \cdots X_n, \\ S_2 &= Z_1Z_2Z_3Z_4, \quad S_4 = Z_1Z_2Z_5Z_6 \cdots Z_n \end{aligned} \tag{2.121}$$

where the subscripts indicate the physical qubit that given operator is acted on. And the logical operators are:

$$\overline{X}_i = X_1 X_3 X_{i+4} \quad \& \quad \overline{Z}_i = Z_1 Z_3 Z_{i+4} \tag{2.122}$$

Note that as the Hadamard gate exchanges $X$ and $Z$ operators in physical qubit level, transversal Hadamard does it in logical level. And also note that all H codes are distance-2, so, they are capable of detecting single Pauli error. Also note that for a universal quantum computing, the eigenstate with +1 eigenvalue $|H\rangle = \cos(\pi/8)\,|0\rangle + \sin(\pi/8)\,|1\rangle$ of Hadamard operator $H = (X+Z)/\sqrt{2}$ is a magic state. [30]

### 2.2.7 $[[n, n-2, 2]]$ even weight code

Following [18] in this topic, we discuss a stabilizer with distance-2 and even physical qubits $n$. Since this method encodes $n-2$ qubits, for larger $n$, the encoding rate $\eta_R = k/n = (n-2)/n \to 1$. Therefore, we can use it for block codes which are encoding multiple qubits. But since distance-2 codes cannot correct a general error, [31] it can be used of error detection [32]. Another possible usage is to fix located errors [34]: in a system with dominant error source caused by the qubits leaving the computational space, we can detect in which qubit the error is occured in principle. After that, we can correct the state via a distance-2 code. Lastly, we can use distance-2 codes for concatenation of distance-2 codes to get codes which can correct multiple errors.

The logical operators of $\overline{X}_i$ are $X_1 X_{i+1}$ where $i = 1, \cdots, n-2$ and $\overline{Z}_i$ are $Z_{i+1} Z_n$, then swapping the $(i+1)^{th}$ qubit with $(j+1)^{th}$ qubit results in swapping the $i^{th}$ qubit with $j^{th}$ encoded qubit. If performed carefully, eventhough it is not a transversal operation, swap can be done fault-tolerantly. And any error on one qubit won't propagate to the others as they are swapped too, but if we apply swap to the two qubits directly, then an error in swap gate might cause error in both of the qubits. We can solve this by introducing third ancilla qubit.

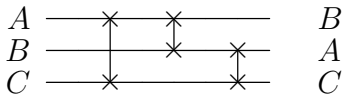



Figure 2.16: Ancilla with SWAP gates [31]

The circuit in fig.2.16 shows how it can be done with an ancilla [31].
Hadamard rotation converts $\overline{X}_i$ to $Z_1 Z_{i+1} = Z_2 \cdots Z_i Z_{i+2} \cdots Z_n$ (with multiplication by $M_2$) which is equivalent to $\overline{Z}_1 \cdots \overline{Z}_{i-1} \overline{Z}_{i+1} \cdots \overline{Z}_{n-2}$ while $\overline{Z}_i$ to $\overline{X}_1 \cdots \overline{X}_{i-1} \overline{X}_{i+1} \cdots \overline{X}_{n-2}$. And the $CNOT$ gate is a $CNOT$ between all of the encoded qubits in first block and their correspondings in the second block.

In [27], it is shown that in $[[n, n-2, 2]]$ codes, with two extra qubits, encoded $CNOT$ and encoded Hadamard gates are fault-tolerantly applied while encoded $CCZ$ gate requires four extra qubits $\forall n \geq 6$.

#### 2.2.7.1 [[4,2,2]] CSS code

It is a very famous CSS code offered by [32] with following encoding:

$$\begin{aligned} |0\rangle \rightarrow \left|\overline{0}\right\rangle &\equiv \frac{1}{2}(|00\rangle + |11\rangle)(|00\rangle + |11\rangle) \\ |1\rangle \rightarrow \left|\overline{1}\right\rangle &\equiv \frac{1}{2}(|00\rangle - |11\rangle)(|00\rangle - |11\rangle) \end{aligned} \tag{2.123}$$

where we encoded $\alpha |0\rangle + \beta |1\rangle$ as $\alpha \left|\overline{0}\right\rangle + \beta \left|\overline{1}\right\rangle$. [32] For more practical purposes, consider logical codewords in the computational ($Z$) basis: [33]

$$\begin{aligned} \left|\overline{00}\right\rangle &= \frac{1}{\sqrt{2}}(|0000\rangle + |1111\rangle), \quad \left|\overline{11}\right\rangle = \frac{1}{\sqrt{2}}(|0110\rangle + |1001\rangle) \\ \left|\overline{10}\right\rangle &= \frac{1}{\sqrt{2}}(|0101\rangle + |1010\rangle), \quad \left|\overline{01}\right\rangle = \frac{1}{\sqrt{2}}(|0011\rangle + |1100\rangle) \end{aligned} \tag{2.124}$$

and codewords in Hadamard basis (or along x):

$$|\overline{++}\rangle = \frac{1}{\sqrt{2}}(|++++\rangle + |----\rangle), \quad |\overline{--}\rangle = \frac{1}{\sqrt{2}}(|+--+\rangle + |-++-\rangle)$$
$$|\overline{+-}\rangle = \frac{1}{\sqrt{2}}(|+-+-\rangle + |-+-+\rangle), \quad |\overline{-+}\rangle = \frac{1}{\sqrt{2}}(|++--\rangle + |--++\rangle) \tag{2.125}$$

Logical Pauli operators:

$$\begin{aligned} Z_a &= Z \otimes Z \otimes \mathbb{I} \otimes \mathbb{I}, \quad Z_b = Z \otimes \mathbb{I} \otimes Z \otimes \mathbb{I} \\ X_a &= X \otimes \mathbb{I} \otimes X \otimes \mathbb{I}, \quad X_b = X \otimes X \otimes \mathbb{I} \otimes \mathbb{I} \end{aligned} \tag{2.126}$$

and the stabilizers to extract logical bit-flip and phase-flips:

$$\overline{X} = X \otimes X \otimes X \otimes X, \quad \overline{Z} = Z \otimes Z \otimes Z \otimes Z \tag{2.127}$$

Note that [34] showed that $[[4, 2, 2]]$ code is the smallest possible single-qubit code detecting errors, that are single qubit erasure and single-qubit error [32] while [35] showed that it can correct single amplitude damping with a loose quantum error correction criteria that can be useful for the systems with a known dominant quantum error source.

### 2.2.8 Divisible Quantum Codes

In classical error-correction, we say a linear code $C$ is divisible by $\Delta$ if every codeword weight is a multiple of $\Delta$ [46] where $\Delta$ is the divisor of $C$ denoted by $\Delta|C$. In other words, divisible codes are the codes whose weights of codewords have a common divisor $\Delta \geq 1$. This codes are created via the coset of $1^{st}$ order Reed-Muller code which is defined by quadratic forms. The basic idea is the stabilizer code with layers such that a universal set of fault-tolerant gates protects $N_1$ number of inner qubits via $N_2$ number of outer qubits. In this code, we follow [44]

Let $n \geq 1$, $N = 2^n$ and binary vectors :

$$\mathbf{a} = [a_1, a_2, \cdots, a_n] \quad \& \quad \mathbf{b} = [b_1, b_2, \cdots, b_n] \quad \ni a_i, b_j \in \{0, 1\} \forall i, j \leq n \tag{2.128}$$

Then define operators:

$$\begin{aligned} D(\mathbf{a},\mathbf{b}) &\coloneqq X^{a_1}Z^{b_1}\otimes\cdots\otimes X^{a_n}Z^{b_n} \\ E(\mathbf{a},\mathbf{b}) &\coloneqq i^{\mathbf{a}\mathbf{b}^T \mod (4)} D(\mathbf{a},\mathbf{b}) \end{aligned} \tag{2.129}$$

1,2 or 4 can be the order of $D(\mathbf{a},\mathbf{b})$ however $E(\mathbf{a},\mathbf{b}) = \mathbb{I}_N$

Define Pauli group on $n$-qubits:

$$\mathbb{P}_N \coloneqq \{i^K D(\mathbf{a},\mathbf{b}) | \mathbf{a},\mathbf{b}\in\mathbb{F}_2^n, K\in\mathbb{Z}_4\} \tag{2.130}$$

where $\mathbb{Z}_{2^l} = \{0,1,\cdots,2^l-1\}$

To describe the state evolution of a stabilizer code under a diagonal physical gate, we use:

$$\begin{aligned} U_Z = \sum_{\mathbf{v}\in\mathbb{F}_2^n} f(\mathbf{v})E(\mathbf{0},\mathbf{v}), \quad \text{where } f(\mathbf{v}) = \frac{1}{2^n}, \sum_{\mathbf{u}\in\mathbb{F}_2^n}(-1)^{\mathbf{u}\mathbf{v}^T} d_{\mathbf{u}} \\ \text{and} \quad \mathbf{v} = [v_1, v_2, \cdots, v_n] \in \mathbb{F}_2^n \quad \& \ \mathbf{u}\in\mathbb{F}_2^n \end{aligned} \tag{2.131}$$

To achieve the average logical channel induced by $U_Z$ on an $[[n,k,d]]$ $CSS(X,C_2;Z,C_1^\perp,\mathbf{y})$ where $\mathbf{y}$ is the binary vector which decides the signs of the elements of the $Z$-stabilizer group in CSS code $C$, we follow these steps:

i. prepare any code state $\rho_1$

ii. to obtain $\rho_2$, apply $U_Z$

iii. to measure $\rho_2$, obtain syndrome $\mathscr{E}$ with probability $\mathcal{P}_\mathscr{E}$ and the state after measurement $\rho_3$, use $X$-stabilizers

iv. to obtain $\rho_4$, apply a Pauli correction to $\rho_3$

If we denote $B_\mathscr{E}$ as the effective physical operator of syndrome $\mathscr{E}$, then evolution of code spaces:

$$\rho_4 = \sum_{\mathscr{E}\in\mathbb{F}_2^n/C_2^\perp} B_\mathscr{E}\rho_1 B_\mathscr{E}^\dagger \tag{2.132}$$

For a given $G_{C_1/C_2}$, there exists a unique vector set $\{\gamma_1, \gamma_2, \cdots, \gamma_k \in C_2^{\perp} \mid G_{C_1/C_2}\gamma_i = \hat{\mathbf{e}}_i \ \forall i = 1, \cdots, k\}$ where $\hat{\mathbf{e}}_{i=1,\cdots,k}$ are standard bases on $\mathbb{F}_2^k$.

As we expand $B_{\mathscr{E}}$ logical operator in terms of $Z$-logical Pauli operators $\mathscr{E}(\mathbf{0}, \gamma_{\mathscr{E}})E(\mathbf{0}, \gamma_{\mathscr{E}})$ which models the $Z$-logical Pauli correction of the decoder, we get the generator coefficients $A_{\mathscr{E}}, \gamma$.

Simplifying yields:

$$A_{\mathscr{E},\gamma} = \frac{1}{|C_1|} \sum_{\mathbf{u} \in C_1} (-1)^{(\mathscr{E} \oplus \gamma)\mathbf{u}^T} d_{\mathbf{u} \oplus \mathbf{y}} \quad \text{where } |C_1| = 2^{k_1} \text{which is the size of } C_1 \tag{2.133}$$

Then, if we put those generator matrices in a matrix where rows stand for $X$-syndromes and columns for $Z$-logicals we get:

$$M_{(\mathbb{F}_2^n/C_2^{\perp}, C_2^{\perp}/C_1^{\perp})} = \begin{bmatrix} \left[A_{\mathscr{E}=0,\gamma}\right]_{\gamma \in C_2^{\perp}/C_1^{\perp}} \\ \left[A_{\mathscr{E}=\mathscr{E}_1,\gamma}\right]_{\gamma \in C_2^{\perp}/C_1^{\perp}} \\ \vdots \\ \left[A_{\mathscr{E}=\mathscr{E}_{2^{k_2}-1},\gamma}\right]_{\gamma \in C_2^{\perp}/C_1^{\perp}} \end{bmatrix}_{\mathscr{E} \in \mathbb{F}_2^n} \tag{2.134}$$

If we fix $\mathscr{E} \in \mathbb{F}_2^n/C_2^{\perp}$ we get:

$$\begin{aligned} \left[A_{\mathscr{E},\gamma}\right]_{\gamma \in C_2^{\perp}/C_1^{\perp}} =& \frac{1}{|C_1|} \left[d_{\mathbf{u} \oplus \mathbf{y}}\right]_{\mathbf{u} \in C_1} H^{\mathscr{E}}_{(C_1, C_2^{\perp}/C_1^{\perp})} \\ & \text{where } H^{\mathscr{E}}_{(C_1, C_2^{\perp}/C_1^{\perp})} = \left[(-1)^{(\mathscr{E} \oplus \gamma)\mathbf{u}^T}\right]_{\mathbf{u} \in C_1, \gamma \in C_2^{\perp}/C_1^{\perp}} \end{aligned} \tag{2.135}$$

Note that the choice of $X$-syndromes and $Z$-logicals are not unique, we see that different choices have different global phase. Also, what generator coefficients do is using CSS code to group Pauli coeffecients of diagonal physical gate and adjust the signs of $Z$-stabilizers.

**Theorem 2.2.11** *[45] A codespace $CSS(X, C_2; Z, C_1^{\perp}, \mathbf{y})$ is protected by the physical diagonal gate $U_Z$ iff:*

- *either*

$$\sum_{\gamma \in C_2^{\perp}/C_1^{\perp}} \|A_{\mathbf{0},\gamma}\|^2 = \sum_{\gamma \in C_2^{\perp}/C_1^{\perp}} \overline{A_{\mathbf{0},\gamma}} A_{\mathbf{0},\gamma} = 1 \tag{2.136}$$

  *where* $\|\cdot\|$ *denotes the complex form*

- *or*

$$\left[A_{\mathscr{E}\neq\mathbf{0},\gamma}\right]_{\gamma} \in C_2^{\perp}/C_1^{\perp} = \mathbf{0} \tag{2.137}$$

So the induced logical operator:

$$U_Z^L = \sum_{\alpha \in \mathbb{F}_2^k} A_{\mathbf{0},g(\alpha)} E(\mathbf{0},\alpha) \tag{2.138}$$

where $g(\alpha) = \alpha G_{C_2^{\perp}/C_1^{\perp}}$ defines the bijective map $g : \mathbb{F}_2^k \to C_2^{\perp}/C_1^{\perp}$ and $G_{C_2^{\perp}/C_1^{\perp}}$ is a $Z$-logical operator matrix choice such that $G_{C_1/C_2} G_{C_2^{\perp}/C_1^{\perp}} = \mathbb{I}_k$

**Theorem 2.2.12** *For a given* $CSS(X, C_2; Z, C_1^{\perp}, \mathbf{y})$ *code, logical gate* $U_Z^L = \sum_{\alpha \in \mathbb{F}_2^k} e^{i\theta\alpha} |\alpha\rangle\langle\alpha|$ *is induced by a physical diagonal gate iff*

$$d_{\mathbf{u}\oplus\mathbf{y}} = e^{i\theta\alpha} \quad \textit{for} \;\; G_{C_2^{\perp}/C_1^{\perp}} \mathbf{u}^T = \alpha^T \tag{2.139}$$

**Theorem 2.2.13** *Theorem of Ax [47]*
*Suppose a generalized Reed-Muller code of order* $r$ $RM_q(r,m)$ *is defined over the* $GF(q)$*. Then this code is divisible by* $q^{(m/r)-1}$*, and this divisor is the highest power of* $p$ *capable of dividing the code if* $p$ *is a prime and divides* $q$*.*

Classical Reed-Muller codes are divisible codes that their codewords are evaluation functions $[h(\mathbf{x})]_{\mathbf{x}\in\mathbb{F}_2^m}$ of boolean functions $h \in \mathbb{F}_2$ with degree at most $r$ for a given Reed-Muller code $RM(r,m)$ in general. So, with the theorem of Ax (2.2.13) we see that all the weights in $RM(r,m)$ are divisible by $2^{(m-1)/r}$.

For Reed-Muller code of order 2 $RM(2,m)$, the codewords are the evaluation functions $[\epsilon\mathbf{1} \oplus L_{\mathbf{a}}(\mathbf{x}) \oplus Q_R(\mathbf{x})]_{\mathbf{x}\in\mathbb{F}_2^m}$ where $\epsilon \in \{0,1\}$, $Q_R(\mathbf{x})$ is a quadratic form and

$L_{\mathbf{a}} = \mathbf{a}\mathbf{x}^T$ is a linear function. Since we are interested in quadratic forms, the following property is defining quadratic form:

$$Q_R(\mathbf{x} \oplus \mathbf{y}) = Q_R(\mathbf{x}) \oplus Q_R(\mathbf{y}) \oplus \mathbf{x}R\mathbf{y}^T \tag{2.140}$$

where $R$ is a binary symplectic matrix in which the diagonal is all zeros.

Note that by writing $R$ as a superposition of a strictly upper triangular matrix and its transpose, we can set $Q_R(\mathbf{x}) = \mathbf{x}U_U\mathbf{x}^T$ where $U_U$ is strictly upper triangular. And $Q_R(\mathbf{x}) + L_\alpha(\mathbf{x})$ becomes a quadratic form of same binary symplectic matrix $R$ if $L_\alpha(\mathbf{x})$ is a linear function. So, the weights of coset $RM(1,m) + [Q_R(\mathbf{x})]_{\mathbf{x}\in\mathbb{F}_2^m}$ depend on the rank of $R$ only.

Consider $CSS(X, C_2; Z, C_1^\perp, \mathbf{y} = 0)$ code for $m \geq 4$ where $C_2 = C(m)$ is a simplex code with size $2^m - 1$ while $C_1 = \langle C_2, [\mathbf{1} \oplus x_i x_j]_{\mathbf{x}\in\mathbb{F}_2^m,\, \mathbf{x}\neq\mathbf{0}} | 1 \leq i \leq m-4,\ i < j \rangle$. Then

$$G_{C_1/C_2} = \begin{bmatrix} \mathbf{1} \\ (\mathbf{1} \oplus x_i x_j)_{\mathbf{x}\in\mathbb{F}_2^m, \mathbf{x}=\mathbf{0}} \\ \vdots \end{bmatrix}_{1\leq i\leq m-4,\, i<j} \tag{2.141}$$

which is the matrix generating $X$-logicals.

We observe that this CSS code is $[[n, 1 + \sum_{i=1}^{m-4}(m-i), 3]]$ by noticing that the minimum distance $d$ is also the minimum distance of Hamming code $C_2^\perp$.

**Theorem 2.2.14** *$CSS(X, C_2; Z, C_1^\perp, \mathbf{y} = 0)$ code is preserved by transversal gate $T^\dagger$ such that*

$$(T^\dagger)^{\otimes n} = U_Z = \sum_{\mathbf{u}\in\mathbb{F}_2^n} (e^{i\pi/4})^{wt_H(\mathbf{u})} |\mathbf{u}\rangle\langle\mathbf{u}| \tag{2.142}$$

*and this gate producees the following logical operator*

$$U_Z^L = \sum_{\alpha\in\mathbb{F}_2^k} d_\alpha |\alpha\rangle\langle\alpha|, \quad \textit{where } d_\alpha = \begin{cases} 1, & \textit{if } wt_H(\alpha) \textit{ is even} \\ e^{i\pi/4}, & \textit{if } wt_H(\alpha) \textit{ is odd} \end{cases} \tag{2.143}$$

$$U_Z^L \equiv \exp\left(\frac{i\pi}{8} Z \otimes Z \otimes \cdots \otimes Z\right) \tag{2.144}$$

Note that this theorem is valid for all $[[2^m - 1, 1 \leq k \leq 1 + \sum_{i=1}^{m-4}(m-i), 3]]$ CSS codes which are constructed by removing rows of the form $(\mathbf{1} \oplus x_i x_j)_{\mathbf{x}\in\mathbb{F}_2^m,\mathbf{x}\neq\mathbf{0}}$ from

$G_{C_1/C_2}$. One thing to notice here is that the only difference between equations 2.143 and 2.2.8 is the $e^{-i\pi/8}$ global phase and the can be achieved by applying a series of $CNOT$ gates from a $T$ gate by conjugation.

**Lemma 2.2.15**

$$\frac{\pi}{4}\binom{k}{1} - \frac{\pi}{2}\binom{k}{2} + \pi\binom{k}{2} = \begin{cases} 0 \mod (2\pi) & \textit{if } k \geq 1 \textit{ is even} \\ \frac{\pi}{4} \mod (2\pi) & \textit{if } k \geq 1 \textit{ is odd} \end{cases} \tag{2.145}$$

This lemma tells us that $U_Z^L$ at equation 2.143 can be split into:

- a $T$-gate on each logical qubit
- $CP^\dagger$ (controlled phase) gate on each logical pairs
- $CCZ$ on each logical qubit triples

**Designing stabilizer codes in layers** To design stabilizer codes in layers, we follow the following steps:

1. We derive all possible physical diagonal gates $U_Z'$ (on $N_1$ qubits) to produce a targeted logical gate starting with a stabilizer code on $N_1$ qubits.
2. We get the unique physical gate (in form equation 2.148) which preserves $[[5, 1, 3]]$ code and gives a logical $T$-gate.
3. By inserting $N_1$ qubits to a physical space of $N_2$ qubits which is larger, we made $N_1$ qubits to be the logical qubits of a stabilizer code on $N_2$ qubits.

So, the outer code is protected by the transversal diagonal gate on $N_2$ qubits and that gate also induces the target logical operator on the inner code.

As an example $(T^\dagger)^{\otimes 31}$ protects the $[[31, 5, 3]]$ code and produces a logical $T$-gate on the $[[5, 1, 3]]$ code which is in the inner layer.

To illustrate consider $[[5,1,3]]$ code and its generator matrix $G_S = [C|D]$

$$\text{where} \quad C = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix} \quad \& \quad D = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.146}$$

**Collary 2.2.1** *Consider $C_2 = \{\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_{2^{k_2-1}}\}$, a diagonal physical gate $U_Z = \sum_{\mathbf{u}\in\mathbb{F}_2^n} d_{\mathbf{u}} |\mathbf{u}\rangle\langle\mathbf{u}|$ and $CSS(X, C_2; Z, C_1^{\perp}, \mathbf{y})$ code. Physical diagonal gate $U_Z$ protects this CSS code iff $d_{\mathbf{u}_0\oplus\omega\oplus\mathbf{y}} = d_{\mathbf{u}_1\oplus\omega\oplus\mathbf{y}} = \cdots = d_{\mathbf{u}_{2^{k_2-1}}\oplus\omega\oplus\mathbf{y}}$ for each fixed $\omega \in C_1/C_2$. And the product logical operator:*

$$U_Z^L = \sum_{\alpha\in\mathbb{F}_2^k} d_{\mathbf{u}_0\oplus\alpha G_{C_1/C_2}\oplus\mathbf{y}} |\alpha\rangle\langle\alpha| \tag{2.147}$$

Note that collary 2.2.1 allows us to check if a physical gate protects a CSS code and to design CSS codes which are protected by desired physical diagonal gate.

Consider the generator matrix for a general stabilizer code $G_S = \begin{bmatrix} A & 0 \\ 0 & B \\ C & D \end{bmatrix}$ where $AandB$ are maximized. By this way, we kept the results same but the tower of classical codes is now $\langle A, B\rangle \subset B^{\perp}$ instead of $C_2 \subset C_1$.

Note that in this case $B = \{\mathbf{0}\}$ giving $B^{\perp} = \mathbb{F}_2^5$. For a coset $\langle C\rangle \in \mathbb{F}_2^5$, $\langle C\rangle$ is consists of all vectors with even weight and its nontrivial cosest contains all the vectors with odd weight.

Then collary 2.2.1 and new tower $\langle C\rangle \subset \mathbb{F}_2^5$ give the only physical diagonal gate protecting $[[5,1,3]]$ code and producing a logical $T$ as:

$$U_Z = \sum_{\alpha\in\mathbb{F}_2^k} d_\alpha |\alpha\rangle\langle\alpha|, \quad \text{where } d_\alpha = \begin{cases} 1, & \text{if } wt_H(\alpha) \text{ is even} \\ e^{i\pi/4}, & \text{if } wt_H(\alpha) \text{ is odd} \end{cases} \tag{2.148}$$

$$U_Z \equiv \exp\left(\frac{i\pi}{8} Z\otimes Z\otimes\cdots\otimes Z\right) \tag{2.149}$$

Now we construct the outer layer for $[[5,1,3]]$ code with $m = 5$ and $[[31,5,3]]$ CSS code protected by $(T^{\dagger})^{\otimes 31}$.

Let $G_{C_1} = \begin{bmatrix} G_{C_1/C_2} \\ G_{C_2} \end{bmatrix}$ where $G_{C_1/C_2} = \begin{bmatrix} \mathbf{1} \\ (\mathbf{1} \oplus x_1 x_j)_{\mathbf{x}\in\mathbb{F}_2^5,\, \mathbf{x}\neq\mathbf{0}} \end{bmatrix}_{j=2,\cdots,5}$ and $G_{C_2} = \Big[(x_i)_{\mathbf{x}\in\mathbb{F}_2^5,\, \mathbf{x}\neq\mathbf{0}}\Big]_{i=1,\cdots,5}$

We note that each matrix $R$ in $\langle R_i | i = 2, \cdots, 5\rangle$ has maximum rank of 2 if $R_i$, $i = 2, \cdots, 5$ is a symmetric binary matrix determined by $x_1 x_j$ quadratic form. Here $X$-logicals with odd weight determine cosets $C_2 + [Q_R(\mathbf{x})] + \mathbf{1}$ while $X$-logicals with even weight determine cosets $C_2 + [Q_R(\mathbf{x})]_{\mathbf{x}\in\mathbb{F}_2^5,\, \mathbf{x}\neq\mathbf{0}}$. And theorem 2.2.14 ensures us that $(T^\dagger)^{\otimes 31}$ protects the CSS code while producing logical operator in equation 2.148 when $m - (m-4) - 1 = 3$.

This family allows us to design stabilizer codes in layers. By doing so, we generate universal set of gates without teleporting the magic states. But since we need to induce logical gates on the inner code, the passage of encoding/decoding algorithms between layers is needed causing overhead of factorization depending on the algorithm complexity. Further research is necessary to optimize the gate synthesis by reducing the code switches in number.

## 2.3 Quantum Low-Density Parity Check Codes

Classical Low Density Parity Check (LDPC) codes are used in classical error correction and offer high performance with low cost. For quantum case, they are promising since they use few resouce. An $[[n, k, d]]$ QLDPCC with encoding rate $\eta_R = k/n$ is the same as $\eta = (n+k)/2n$ of $(2n, k+n)$ classical binary LDPC code.[77]

If we define a *good code* using the notion in classical coding theory as code having $k \in \Theta(n)$ and $d \in \Theta(n)$, good QLDPCCs having $d \geq \Theta(\sqrt{n} polylog(n))$ are discovered in 2020 which we will discuss in the following subsections. [64] note that, we will follow mostly [64] in QLDPCCs. For the mathematics required, reader may check Appendices.

### 2.3.1 Product Codes

#### 2.3.1.1 Hypergraph Product Codes

In this topic, we follow [53]. Before we start, let $\mathcal{G}_1 \otimes \mathcal{G}_2$ be the product graph of $C_1 \otimes C_2$ where $\mathcal{G}_1$ and $\mathcal{G}_2$ are Tanner graphs related to codes $C_1$ and $C_2$ respectively due to following definition:

**Definition 2.3.1** *Let $v_1$ and $v_2$ be the vertices of graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ respectively. $\mathcal{G}_1 \otimes \mathcal{G}_2$ gives a product graph with vertex set $(v_1, v_2)$ pairs, and edges connecting $(v_1, v_2)$ and $(v_1', v_2')$ if one of the followings is satisfied:*

*i) when $v_1 = v_1'$, $\{v_2, v_2'\}$ be an edge of $\mathcal{G}_2$*

*ii) when $v_2 = v_2'$, $\{v_1, v_1'\}$ be an edge of $\mathcal{G}_1$*

Note that $\mathcal{G}_1$ and $\mathcal{G}_2$ are called *hypergraphs* in this notion. Due to definition 2.3.1, we see that hypergraphs have edges with cardinality 2. In Tanner graph $\mathcal{G} = T(V, K, E)$ associated to PCM $H$, $V = \{1, \cdots, c\}$ (where $c$ is the column number of $H$) is called *variable node set*, $K = \{\otimes_1, \cdots, \otimes_r\}$ (where $r$ is the row number of $H$) is called *check node set* and $E$ is the set of edges (note that if $H_{ij} = 1$, then there exists an edge between $j$ and $\otimes_i$).

**Definition 2.3.2** *Consider Tanner graphs $\mathcal{G}_i = T(V_i, K_i, E_i)$ $\forall i = 1, 2$. For an induced subgraph $\mathcal{G}_1 \times \mathcal{G}_2$ having $V_1 \times V_2$ set of variable nodes and $K_1 \times V_2 \cup V_1 \times K_2$ set of check nodes, $\mathcal{G}_1 \otimes \mathcal{G}_2$ is called hypergraph product.*

**Definition 2.3.3** *(Product Code) Let $C_i$ be codes over $\mathbb{F}_2$ with lengths $n_i$ $(i = 1, 2)$. Then we can consider the codewords of product $C_1 \otimes C_2$ as $n_1 \times n_2$ binary matrices which are from $C_1 \otimes C_2$ iff $C_1$ has its columns and $C_2$ has its rows.*

Then the product has the dimension:

$$\dim(C_1 \otimes C_2) = \dim(C_1) \dim(C_2) \tag{2.150}$$

And the Tanner graph of $C_1 \otimes C_2$ is $\mathcal{G}_1 \otimes \mathcal{G}_2$ where $\mathcal{G}_1$ and $\mathcal{G}_2$ are Tanner graphs associated with $C_1$ and $C_2$.
Note that

$$T^T(V, K, E) = T(K, V, E) \tag{2.151}$$

and

$$\dim(C) = |V| - |K| + \dim(C^T) \tag{2.152}$$

So, for associated codes $C_x$ and $C_z$ of $C$ we have:

$$\begin{aligned} C_x(\mathcal{G}_1 \times \mathcal{G}_2)^T &= C_1^T \otimes C_2 \\ C_z(\mathcal{G}_1 \times \mathcal{G}_2)^T &= C_1 \otimes C_2^T \end{aligned} \tag{2.153}$$

or the products of $X$ and $Z$ we have:

$$\begin{aligned} \mathcal{G}_1 \times_x \mathcal{G}_2 &= (\mathcal{G}_1^T \otimes \mathcal{G}_2)^T \\ \mathcal{G}_1 \times_z \mathcal{G}_2 &= (\mathcal{G}_1 \otimes \mathcal{G}_2^T)^T \end{aligned} \tag{2.154}$$

Here, note that we induce $\mathcal{G}_1 \times \mathcal{G}_2$ by set of vertex nodes $V_1 \times V_2 \cup K_1 \times K_2$ and set of nodes $K_1 \times V_2$ and get the subgraph $\mathcal{G}_1 \times_x \mathcal{G}_2$. $\mathcal{G}_1 \times_z \mathcal{G}_2$ is similar with the same set of vertex nodes and set of nodes $V_1 \times K_2$. Also note that $C_x(\mathcal{G} \otimes \mathcal{G})$ and $C_z(\mathcal{G} \otimes \mathcal{G})$ are isomorphic.

#### 2.3.1.2 Fiber Bundle Codes

In subsystem codes, checks doesn't necessarily commute with each other, as a result we cannot interpret them as chain complexes. Fiber bundles' topology is used to construct *twisted product*, *lifted product* and *balanced product codes*. Here one thing to note is that twisted product and lifted products are probabilistic codes since they are constructed as products of classical codes (but we are not diving into probabilities of them violating or satisfying the required distances etc.) while balanced product codes are products of two quantum codes and constructed with deterministic algorithms which guarantees that the parameters hold. Note that in this section we follow Hastings-Haah-O'Donnell [50].

**Theorem 2.3.1** *For $N$ qubits and check distances $d_x = \Omega(\sqrt{N}/polylog(N))$ and $d_Z = \Omega(N^{3/4}/polylog(N))$ and generators with weight of at most $polylog(N)$ where all qubits are in $polylog(N)$ generators or less, there is a code family with $k = \Theta(\sqrt{N})$*

Though this codes are not in QLDPCCs, we can reduce its weight using bundle construction. We use two manifolds and take their product giving us fiber bundle. To illustrate, we can think of the Möbius strip. In that, we see it as product of some circle having some interval locally, yet it has reversing at the boundary (a twist). Due to twist, the strip is not homeomorphic to the product of that circle. Like in Möbius strip, fiber bundles seem like product locally but have more intricate structure globally. We have a base, a local product and a fiber generating automorphisms. In Möbius strip example, base is circle and fiber is the twist at the ends.

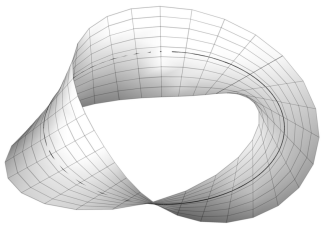
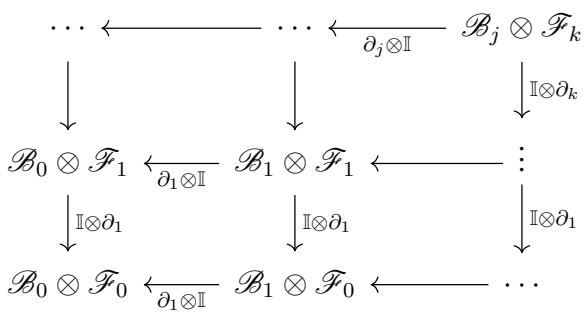


Figure 2.17: Mobius strip with 1 twist [49]

If we take base and fiber as circles, we have a torus. Consider a twist around the base of $\phi_0 \in \mathbb{R}$, some fixed angle. After twist, we still have a torus but with different geometry. Let $\theta \in \mathbb{R}$ be the angle for base and $\phi \in \mathbb{R}$ for fiber, we can identify as:

$$(\theta, \phi) \equiv (\theta, \phi + 2\pi) \equiv (\theta + 2\pi, \phi + \phi_0) \tag{2.155}$$

If we choose $\phi_0 = 0$, $d = min(m_B^{(1)}, m_F^{(1)}) = \sqrt{N/2}\ \forall m_B^{(1)} = m_F^{(1)}$ (where $m_{B,F}^{(i)}$ is the number of $i$-cells in base or fiber). A bundle $\mathscr{E}$ is product of some base complex $\mathscr{B}$ and fiber complex $\mathscr{F}$:



Then define bundles chain space:

$$\mathscr{E}_\mu = \bigoplus_{\alpha+\beta=\mu} \mathscr{B}_\alpha \otimes \mathscr{F}_\beta \tag{2.156}$$

For $\alpha$-cell of base $b^\alpha \in \mathscr{B}_\alpha$ and $\beta$-cell of fiber $f^\beta \in \mathscr{F}_\beta$, $\mu$-cells of the bundle $\mathscr{E}$ are pairs $(b^\alpha, f^\beta) \ni \alpha + \beta = \mu$. $\mu$-cell of such bundle is $(\alpha, \beta)$-cell and $\mathscr{E}$ is $(b + f)$-complex for $b$-complex $\mathscr{B}$ and $f$-complex $\mathscr{F}$.
Boundary map acting on every $\mathscr{B}_\alpha \otimes \mathscr{F}_\beta$:

$$\partial^{\mathscr{E}}_\mu\Big|_{(\alpha,\beta)} = \mathbb{I} \otimes \partial^{\mathscr{F}}_\beta + \partial^{\mathscr{B}}_\alpha \otimes \mathbb{I} \tag{2.157}$$

where $\partial^i \ni i \in \{\mathscr{B}, \mathscr{F}\}$ are boundary maps for base and fiber complexes. And in general we have $\partial^{\mathscr{E}}(\alpha, \beta) = (-1)^\alpha \mathbb{I} \otimes \partial^{\mathscr{F}}_\beta + \partial^{\mathscr{B}}_\alpha \otimes \mathbb{I}$ but not necessary for this thesis.

Consider an automorphism group $G$. The members of this group are permutings of $\beta$-cells with commuting boundary operators under the actions in $G$. Then fiber automorphism dictates:

$$g\partial f = \partial g f \quad \forall \beta,\ g \in G,\ f^\beta \in \mathscr{F}_\beta \tag{2.158}$$

We define twist as $\{(b, a) : b, a \text{ are cells } \ni a \in \partial b\} \xrightarrow{\phi} G$ where $\phi$ is the action for automorphism and, $\partial b$ and its support are identified. Note that this definition is applicable for an automorphism group $G$ satisfying equation 2.158. Then we can define twisted boundary map $\partial^{\mathscr{E}}$ by a connection $\phi$ in the case of no curvature:

$$\begin{aligned} \partial^{\mathscr{E}}_{(0,\beta)}(b^0 \otimes f) &= b^0 \otimes \partial f \\ \partial^{\mathscr{E}}_{(1,\beta)}(b^1 \otimes f) &= b^1 \otimes \partial f + \sum_{a^0 \in \partial b^1} a^0 \otimes \phi(b^1, a^0) f \end{aligned} \tag{2.159}$$

Notice that we are not using equation 2.157 anymore. Choose base as a 1-complex. Then, elements of $\mathscr{E}_{0,1}$ are vertical while $\mathscr{E}_{1,0}$ are horizontal. Using automorphism $h$ and changing the twist such that $\phi(b,a) \to h\phi(b,a)h^{-1}$, we can transform the fiber.

We use $\Pi_s : \mathscr{E}_s \to \mathscr{B}_s$ bundle projections in fiber bundle codes which can be defined as follows:

$$\begin{aligned} b^s \otimes f^0 &\to b^s \\ b^{s-t} \otimes f^t &\to 0 \ \forall t > 0 \end{aligned} \tag{2.160}$$

where $s$-and$(s-t)$-cells $b^s$and$b^{s-t}$ of base respectively while 0-and$t$-cells $f^0$and$f^t$ of fiber.

Now let cycle graph be the fiber (i.e. circle) which is a 1-complex having $m_F^{(0)}$ and $m_F^{(1)}$ number of 0-and1-cells respectively ($\ni m_F^{(0)} = m_F^{(1)} > 1$). Note that the circle gives rise to dihedral group which is a symmetry group for regular $r$-gon containing rotation and reflection, however we'll only use rotational symmetries. Also note that a circle with even weighted $\partial f^1$, and if each automorphism of fiber has a nontrivial effect on $H_1(\mathscr{F})$ while each even weighted 0-chain fiber gives some boundary, then that circle has an automorphism under which fundamental homology cycle remains invariant.

**Definition 2.3.4** *Consider a circle fiber $\mathscr{F}_1 \to \mathscr{F}_0$ and base $\mathscr{B}_1 \to \mathscr{B}_0$ are used to construct a twisted bundle complex $\mathscr{E}_2 \to \mathscr{E}_1 \to \mathscr{E}_0$. Then in dimension 1, cohomology and homology of this complex are related to the logical operators of our quantum CSS code that is a fiber bundle code.*

Let $m_F^{(0)} = m_F^{(1)} = m^2$ where $m \in \mathbb{Z}$, so all twists are $m$ multiples such that we use rotation group of order $m_F^{(1)}/m = m$. This allows us to lower bound the movement distance of the weights moving through a fiber so that two cells which are different from eachother due to a twist join.

Now, we use some random classical code as base giving us $m_B^{(1)}$ bits and $m_B^{(0)}$ parity checks as 1-and0-cells of base respectively. In Tanner graph, we have $m_B^{(0)}$ left

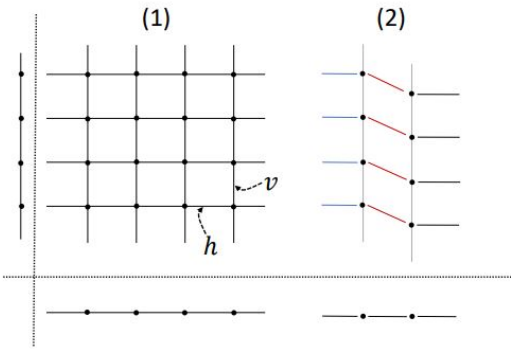



Figure 2.18: An example of a twist where vertical and horizontal edges are identified and the twist is colored in red. [50]

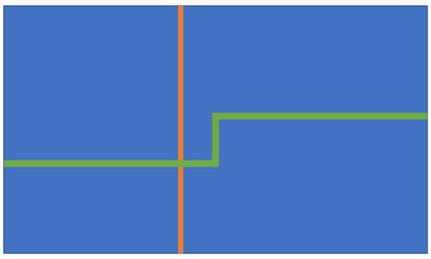


Figure 2.19: Notice vertical representative doesn't change but in horizontal, the distance increases where vertical and horizontal edges are identified. [51]

and $m_B^{(1)}$ right vertices. After some suitable choices bundle has qubit number $n = m_B^{(1)} m_F^{(0)} + m_B^{(0)} m_F^{(1)}$ and $(m_B^{(1)} + m_B^{(0)})(m_F^{(1)} + m_F^{(0)})$ number of cells.

**Definition 2.3.5** *We call base cell set supporting $e \in \mathscr{E}$, shadow of $e$. We can write a 1-chain bundle $e$ as a superposition of vertical and horizontal chains with*

$$\begin{aligned} h &= \sum_{b \in \{\textit{base 1-cells}\}} b \otimes f_b^0 \\ v &= \sum_{a \in \{\textit{base 0-cells}\}} a \otimes f_a^1 \end{aligned} \tag{2.161}$$

*where $h$ and $v$ are horizontal and vertical chains respectively.*
*Then horizontal shadow: $\{b : f_b^0 \neq 0\}$ and vertical shadow: $\{a : f_a^1 \neq 0\}$. We denote vertical shadow weight as $|e|_{vsw}$ etc. Consider a 0-chain bundle $e = \sum_a a \otimes f_a^0$ with shadow $\{a : f_a^0 \neq 0\}$. Then the shadow weight is denoted as $|e|_{sw}$.*

#### 2.3.1.2.1 Twist

Now we define the twist $\phi(b^1, v)$ as an automorphism where $v$ is a 0-cell and $b^1$ is a 1-cell $\ni b^1 \in \partial^T v$. Then $\exists$ a twist $\phi\, \forall v$ & $\forall b^1 \in \partial^T v$ meaning that there exists a twist for every bits in checks and for every check. [50]

Before we go with twists, we have to give a definition to a partitioned base code. To be partitioned, we demand a base code to have partitioned check vertices as $V_1, \cdots, V_k \ni$

$m_B^{(0)}/k \in \mathbb{Z}$ and every check $\nu \in [m_B^{(0)}]$ has a uniformly and randomly partitioned neighborhood $\partial^T \nu \in \{Tails_\nu, Heads_\nu\}$. We choose twists such that a twist $\phi_i = c\lambda \in \mathbb{Z}$ (where $c \in \{1, 2, \cdots \lambda - 1\}$) chosen from the set $\{\lambda, 2\lambda, \cdots, (\lambda-1)\lambda\}$ randomly and uniformly, for every set $V_i$. So for check node $\nu \in V_i$ the twist $\phi(b^1, \nu)$ is defined as: [50]

$$\phi(b^1, \nu) = \begin{cases} 0 & \text{if} \quad b^1 = Heads_\nu \\ \phi_\nu & \text{if} \quad b^1 = Tails_\nu \end{cases} \tag{2.162}$$

We begin defining $\vec{\mathscr{G}}_\phi$ twisted graph with vertices $v_i \in [\lambda]$ and edges $e_i$ connecting vertices $v_i$ and $v_{i+\phi_j/\lambda}$ $\forall j \in [k]$. Let base code $\mathscr{B}$ be partitioned and $\vec{\mathscr{G}}_\phi$ be twisted graph. Then we can define new code $\mathscr{B}(\vec{\mathscr{G}}_\phi) \in \mathbb{F}_2$ having $\lambda m_B^{(0)}$ parity checks and $\lambda m_B^{(1)}$ block length. And the parity checks of $i$-type with pairs $(p, q) \in \mathbb{F}_2^{m_B^{(1)}} \times \mathbb{F}_2^{m_B^{(1)}}$: [50]

$$\sum_{\alpha \in Heads_\nu} p_\alpha + \sum_{\beta \in Tails_\nu} q_\beta = 0 \quad (mod 2) \quad \forall \nu \in V_i \tag{2.163}$$

where $i \in [k]$.

#### 2.3.1.3 Lifted Product Codes

For lifted product codes, we follow [52]. Let for $\lambda \times \lambda$ matrices over $\mathbb{F}_2$, $\mathcal{R}$ be a ring with commuting elements. Then, matrices $A \in \mathcal{R}^{n \times m}$ and $B \in \mathcal{R}^{k \times l}$ can be written as matrices with $\lambda \times \lambda$ blocks in $\mathcal{R}$, $\mathbb{B}(A) \in \mathbb{F}^{\lambda n \times \lambda m}$ and $\mathbb{B}(B) \in \mathbb{F}^{\lambda k \times \lambda l}$. We call a matrix and its corresponding block matrix in binary as quasi-cyclic with lift $\lambda$.

As an example, let $T \in \mathcal{M}_{2\times 3}(\mathcal{R}_3)$ be:

$$T = \begin{pmatrix} x^2 & 1 + x^2 & 1 + x + x^2 \\ 0 & x & 1 + x \end{pmatrix} \tag{2.164}$$

Then with lift size $\lambda = 3$, its associated block matrix is:

$$\mathbb{B}(T) = \left(\begin{array}{ccc|ccc|ccc} 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{array}\right) \tag{2.165}$$

where $1 \equiv \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $x \equiv \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$ and $x^2 \equiv \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$. And the associated weight matrix is:

$$W(T) = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \end{pmatrix} \tag{2.166}$$

Notice that the elements of this weight matrix is the number of terms in matrix $T$.

To satisfy orthogonality condition

$$H_x H_z^* = 0 \tag{2.167}$$

we have:

$$\begin{bmatrix} A \otimes \mathbb{I}_{m_B} & \mathbb{I}_{m_A} \otimes B \end{bmatrix} \begin{bmatrix} \mathbb{I}_{n_A} \otimes B \\ A \otimes \mathbb{I}_{n_B} \end{bmatrix} = 0 \tag{2.168}$$

must be satisfied:

$$(A \otimes \mathbb{I}_{m_B})(\mathbb{I}_{n_A} \otimes B) + (\mathbb{I}_{m_A} \otimes B)(A \otimes \mathbb{I}_{n_B}) = 0 \tag{2.169}$$

since $A$ and $B$ are element-wise commuting matrices, we use mixed product formula

$$(X \otimes Y)(X' \otimes Y') = (XX' \otimes YY') \tag{2.170}$$

then we have:

$$(A \otimes \mathbb{I}_{m_B})(\mathbb{I}_{n_A} \otimes B) = (\mathbb{I}_{m_A} \otimes B)(A \otimes \mathbb{I}_{n_B}) = A \otimes_{\mathcal{R}} B \tag{2.171}$$

For ring $\mathcal{R}$ with characteristic 2, equation 2.169 is satisfied. Then all $A$ and $B$ matrices that their elements commute gives CSS code having $\mathbb{B}(H_x)$ and $\mathbb{B}(H_z)$ as PCMs. Such codes are called *lifted product codes* and denoted by $LP(A, B)$ while $H_x$ and $H_z$ are PCMs of $LP(A, B)$.

#### 2.3.1.4 Balanced Product Codes

In balanced product codes, we follow [54]. Unlike fiber bundle and lifted product codes, balanced product codes are explicit meaning that it can be produced in polynomial time using some algorithm which is deterministic. So, the bounds are not violated in any case.

In [54] it is stated that $[[n, k, d_x, d_z]]$ QLDPCC family of $k \in \Theta(N^{2/3})$, $d_x \in \Omega(N^{1/3})$ and $d_z \in \Theta(N)$ can be turned into an $[[n, k, d]]$ QLDPCC with $k \in \Theta(N^{4/5})$ and $d \in \Omega(N^{3/5})$ by distance balancing.

Let $C$ and $D$ be quantum codes with common symmetry under the actions of some group $G$. Then the new space $C \otimes_G D$ is balanced product.

Topologically speaking, for spaces $X$ and $Y$, $X \times_G Y$ balanced product gives a fiber bundle with base $\frac{X}{G}$ and fiber $Y$. We can also interpret this for a carthesian product $\frac{X}{G} \times Y$, a twist $Y$ along $\frac{X}{G}$. So, we can generate a balanced product code using a twist applied on the checks of some hypergraph product code.

So, let $X$ and $Y$ be two spaces and $G$ be a group such that $G$ acts on $X \times Y$ with:

$$g \cdot (x, y) = (x \cdot g^{-1}, g \cdot y) \tag{2.172}$$

where $G$ acts on $X$ and $Y$ freely from right and left respectively.
The space $X \times_G Y \equiv \frac{X \times Y}{G}$ is called *balanced product* and it indicates $G$-orbits under equation 2.172. Note that in this section, we denote $X \times_G Y$ elements as $[x, y]$ and it satisfies $[x \cdot g, y] = [x, g \cdot y]$.
We use balanced product to generate fiber bundles from principal bundles with quotient map

$$\pi_Y : \quad X \times_G Y \to \frac{X}{G}, \quad [x, y] \to xG \tag{2.173}$$

Here, $\pi_Y$ is a fiber bundle with base $\frac{X}{G}$ and fiber $Y$.

Now, we define trivialisation. When we cover a base by suitable homeomorphisms with $G$ and open subsets, we call this as *trivialisation of a bundle* $\pi$ where homeomorphisms are for a base covering set $\{U_i\}$:

$$\psi_i : \quad \pi^{-1}(U_i) \to U_i \times G \tag{2.174}$$

We get a transition map when two sets $U_i$ and $U_j$ intersects:

$$\phi_{i,j} : \quad U_i \cap U_j \to G \tag{2.175}$$

which is the continuous version of twist $\phi$. Note that we can use a continuous twist $\phi_{i,j}$ to recover the bundle $\pi : \ X \to \frac{X}{G}$ from $\{U_i, \phi_{i,j}\}$ data if we get $U_i \times G$ spaces together along $(U_i \cap U_j) \times G$ intersections or we can go backwards from data to fiber bundle by getting $U_i \times Y$ spaces together with continuous twist $\phi_{ij}(x)$ on $Y$ where $Y$ is the fiber.

Now we look for the discrete version for 1D cell complexes. For a graph $X$ and finite group $G$, we define *quotient condition* as from vertices $v$ and $vg$, there is no edges where $G$ acts freely on the 0-and1-cells (vertices and edges), let orbits of $G$ of $V$ and $E$ be the vertices and edges of $X$ respectively. Then 0-and1-cells of $\frac{X}{G}$:

$$\left(\frac{X}{G}\right)_0 = \{V\} \quad \& \quad \left(\frac{X}{G}\right)_1 = \{E\} \tag{2.176}$$

So, for adjacent vertices in $X$ $v \in V$ and $v' \in V'$, vertices in $\frac{X}{G}$ are adjacent. In another words, any orientation defined on $X$ also gives an orientation defined on the

quotient graph $\frac{X}{G}$. Note that the quotient map covers $\frac{X}{G}$ with $|G|$-fold, and also can be understood as a principal bundle. Map is:

$$\pi : \quad X \to \frac{X}{G}, \quad v \to vG \tag{2.177}$$

If we do a trivialisation on this map, we choose for all orbits representatives $R \in \{v \in V\}$ giving us twist $\phi_R$:

$$\phi_R : \quad \left(\frac{X}{G}\right)_1 \to G, \quad E = (V, V') \to \phi_R(E) \tag{2.178}$$

If $v, v' \in R$, then $\exists! \phi_R(E) \in G \ni (v, v'\phi_R(E))$ is a 1-cell $\in X$. Note that if $X$ is not twisted, we can simplify $\phi_R$ locally for a suitable choice of trivialisaion $R$. Moreover using quotient graph and $\phi_R$ we can reconstruct the bundle.

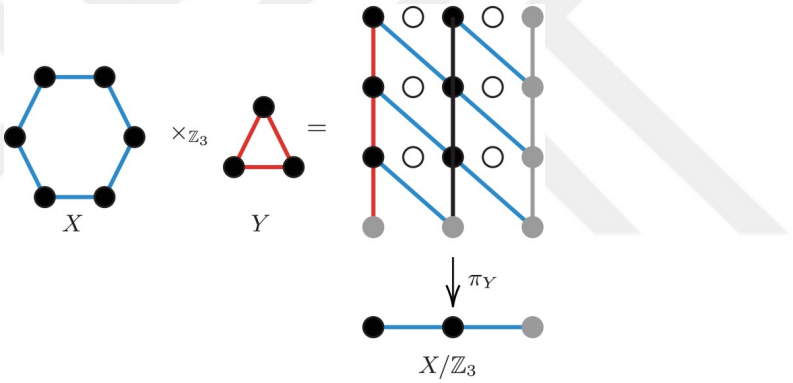



Figure 2.20: $C(X \times_G Y) = C(X) \otimes_G C(Y)$. Here grey points are identified with the points of their opposite sides such that the points on the right side are identified with the points on the left and the points on the down side are identified with the ones on the upper side. Also, vertices, edges and faces are associated with the $X$-checks, qubits and $Z$-checks respectively.[54]

Consider fig.2.20 in which we see two cyclic graphs $X$ and $Y$ with lengths of 6 and 3 respectively. When we take balanced product of $X$ and $Y$ as $X \times_G Y$ where $G = \mathbb{Z}_3$, we get the third graph that is the twisted $[[12, 2, 3]]$ toric code. Then projecting with fiber $Y$ and base $\frac{X}{G}$ we get the length-2 cyclic graph.

Suppose a group $G$ acts on vector spaces $A$ and $B$ from left and right respectively.

Then the balanced product of $A$ and $B$ is:

$$A \otimes_G B = \frac{A \otimes B}{\langle ag \otimes b - a \otimes gb \rangle} \tag{2.179}$$

which is the same as the tensor product over $G$ is:

$$A \otimes_G B = A \otimes_{\mathbb{F}_2[G]} B \quad \text{where} \quad \mathbb{F}_2 G = \left\{ \sum_{g \in G} u_g g \mid u_g \in \mathbb{F}_2 \right\} \tag{2.180}$$

We can apply the same definition to chain complexes $C$ and $D$ with right and left actions of $G$ respectively. (meaning $G$ acts on individual spaces $C_i$ and $D_i$ and commutes with all boundary operations). Then we can form total complex of the double complex $C \boxtimes_G D$, $C \otimes_G D$.
Let $\partial^h$ and $\partial^v$ be horizontal and vertical boundary operators acting on $C_\alpha \otimes_G D_\beta$ and $C_\alpha \otimes D_\beta$ respectively, such that

$$\partial^v = \partial^C \otimes \mathbb{I}_D \quad \& \quad \partial^h = \mathbb{I}_C \otimes \partial^D \tag{2.181}$$

Then we define followings:

- *balanced product double complex:*

$$(C \boxtimes_G D)_{\alpha,\beta} = C_\alpha \otimes_G D_\beta \tag{2.182}$$

- *balanced product complex:*

$$C \otimes_G D = Tot(C \boxtimes_G D) \tag{2.183}$$

Note that we call a balanced product complex as balanced product code when it is in natural basis. Now suppose that bases of $C_i$ and $D_i$ are $B_i^{(C)}$ and $B_i^{(D)}$ which allow us to define natural basis of $(C \otimes_G D)_k$ as:

$$(B^{(C)} \times_G B^{(D)})_k = \biguplus_{i+j=k} B_i^{(C)} \times_G B_j^{(D)} \tag{2.184}$$

Assume that $G$ is finite and order of odd number, the Künneth formula becomes:

$$H_n(C \otimes_G D) = \bigoplus_{p+q=n} H_p(C) \otimes_G H_q(D) \tag{2.185}$$

Breuckmann and Eberhardt used highly symmetrical Sipser-Spielman codes and a repetition code in [54] with cyclic symmetry to construct QLDPCCs with balanced products. Also they provide relation to lifted product codes and fiber bundle codes as follows,

Firstly, to see how twisted product codes are related to the balanced product codes, let's consider some complex $C$ with two-terms and an Abelian group $G$ which can act freely on all bases of $C$. Then there exists a suitable $\phi$ sayisfying:

$$C \otimes_G D = B \otimes_\phi F \tag{2.186}$$

where $B_i = \frac{C_i}{\langle cg - c \rangle}$ and $F = D$.

On the other hand, if $G$ is a commutative group with a free action on vector space $C_i$, $D_i$ the balanced product turns into a lifted product:

$$C \otimes_G D = LP(\partial^C, \partial^D) \tag{2.187}$$

with $R = \mathbb{F}_2 G$, $C = (R^p \xrightarrow{\partial^C} R^q)$ and $D = (R^r \xrightarrow{\partial^D} R^s)$ where $p, q, r, s \in \mathbb{Z}$, $\partial^C \in R^{p \times q}$ and $\partial^D \in R^{r \times s}$. And the check matrices of the lifted product becomes:

$$H_x = \begin{bmatrix} \partial_C \otimes \mathbb{I}_r & \mathbb{I}_p \otimes \partial^D \end{bmatrix} \quad \& \quad H_z^* = \begin{bmatrix} \mathbb{I}_q \otimes \partial_D \\ \partial^C \otimes \mathbb{I}_s \end{bmatrix} \tag{2.188}$$

### 2.3.2 Surface Codes

Surface codes started with Kitaev's construction of toric codes [57]. In toric codes, we have some level of automated error correction. Since toric codes dictates local Hamiltonians, subspace of their ground states which are degenerate yields this sense of automation. [61] [62]

We construct surface codes, in general, by the cellulation of given compact surface

which is orientable. Then we have $|E|$ number of edges where qubits are located and such codes encode $2g$ qubits where $g$ is genus. [90] [62]

First we look at planar quantum codes which are constructed on projective planes followed by toric codes, hyperbolic surface codes, Freedman-Meyer-Luo codes and Haah's codes.

#### 2.3.2.1 Planar Quantum Codes

Following Freedman and Meyer [62], we will see that $\mathbb{R}P^2$ projective plane yields a QECC with $k = 1$ in its cellulations. Note that the construction offered by Freedman and Meyer is dual to Kitaev's and since Kitaev's is more widespread, we stick with the Kitaev's construction.

Consider a surface with cellulation $\mathscr{C}$ and let this $\mathscr{C}$ splits the surface into set of vertices $V$, edges $E$ and faces $F$. Due to homology, we know that boundaries of each face $f \in F$ are edges such that $\partial f = e \in E$ where $\partial$ is the boundary operator. Similarly, for edges $e \in E$ we have $\partial e = v \in V$. And let $\tilde{E}_f \subset E$ and $\tilde{E}_v \subset E$ be the set of edges with boundaries as faces and vertices respectively. Then we define the stabilizers as:

$$\begin{aligned} A_v &:= \bigotimes_{e \in E} X^{\delta(e \in \tilde{E}_v)} \\ B_p &:= \bigotimes_{e \in E} Z^{\delta(e \in \tilde{E}_f)} \end{aligned} \tag{2.189}$$

where $\delta(\cdot) = \begin{cases} 1, & True \\ 0, & False \end{cases}$ satisfying $\prod_v A_v = \prod_p B_p = \mathbb{I}$. (See fig.2.26)

Let $C_1(\mathscr{C}; \mathbb{Z}_2)$ be the 1-chains of $\mathscr{C}$ and $Z_1(\mathscr{C}; \mathbb{Z}_2)$ be the chains having no boundary such that $Z_1(\mathscr{C}; \mathbb{Z}_2) \subset C_1(\mathscr{C}; \mathbb{Z}_2)$ all having coefficients from $\mathbb{Z}_2$. Moreover let $B_1(\mathscr{C}; \mathbb{Z}_2)$ be the boundaries of faces in $\mathscr{C}$, and $\mathscr{C}$ has homology group $H_1(\mathscr{C}; \mathbb{Z}_2) =$

$\frac{Z_1(\mathscr{C};\mathbb{Z}_2)}{B_1(\mathscr{C};\mathbb{Z}_2)}$. And for $\mathbb{R}P^2$ that is the real projective plane, we have 1 essential cycle. Then the $1^{st}$ homology group of $\mathbb{R}P^2$ is $H_1(\mathbb{R}P^2;\mathbb{Z}_2) = \mathbb{Z}_2$. Since we have 1 essential cycle, for all code subspaces associated to $\mathbb{R}P^2$, $k = 1$ and $D = 2$.

As an example, suppose there is a phase-flip error on edge $e$. We can detect this error with $A_v$ staying at the boundary vertices. Then we can correct the error by applying Pauli-$Z$. Note that we can correct this error if the vertex at which we detected the error and the vertex at which we applied Pauli-$Z$ doesn't contain an essential cycle. In bit-flip case, same things happens with $B_p$ in dual cellulation $\mathscr{C}^*$ followed by correction Pauli-$X$.

We can generalize the procedure as follows:
Let $c_1 \in C_1(\mathscr{C};\mathbb{Z}_2)$ and $c_1^* \in C_1(\mathscr{C}^*;\mathbb{Z}_2)$ be the chains on which phase-flip and bit-flip errors occured. We can detect errors on the boundaries of $c_1$ and $c_1^*$ by $A_v$ and $B_p$ stabilizers (if we have odd number of errors near stabilizers) which give us $-1$ eigenvalues. Then we choose some chain $c_2 \in C_1(\mathscr{C};\mathbb{Z}_2)$ and $c_2^* \in C_1(\mathscr{C}^*;\mathbb{Z}_2)$ on the boundary of error, applying Pauli-$Z$ and Pauli-$X$ respectively solves the problem only if $c_1 + c_2$ (or $c_1^* + c_2^*$ in the dual case) does not have essential cycle. We say the code distance in $\mathscr{C}$ (or $\mathscr{C}^*$ in the dual case) is minimum essential cycle length.

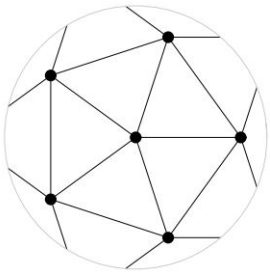


Figure 2.21: Planar projection of [[15,1,3]] code [62]

In fig.2.21 we see cellulation of $\mathbb{R}P^2$ having $[[15, 1, 3]]$ code parameters. Notice that in this tesselation of triangles we see 15 edges on which qubits stands, smallest essential cycle possible is 3 which is the distance. Also if we move vertices to the middle of edges and connecting them gives the dual cellulation in which the shortest essential

cycle is 5. Note that antipodal points are identified.

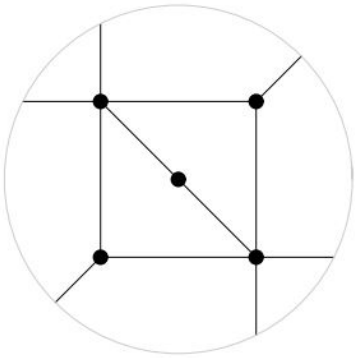


Figure 2.22: Planar projection of [[9,1,3]] code at the beginning [62]

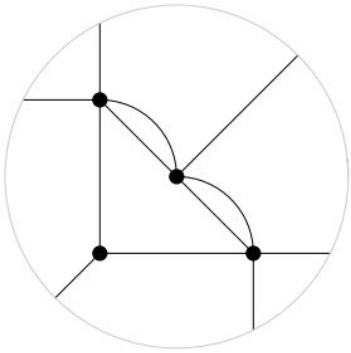


Figure 2.23: Planar projection of [[9,1,3]] code after identification of vertices [62]

In fig.2.22 we see $[[9, 1, 3]]$ code generated by the irregular cellulation. Notice that identifying two vertices gives the fig.2.23 in which still we use 9 qubits and have distance 3 but two codes are not equivalent.

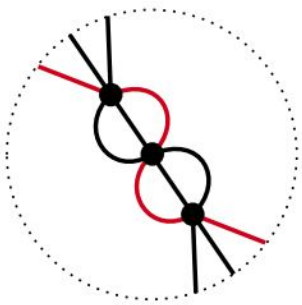


Figure 2.24: Planar projection of [[9,1,3]] code after second identification [64]

Identifying one more vertex gives fig.2.24 where we have again same distance and number of qubits but this code is inequivalent to both the codes in fig.2.22 and fig.2.23. Note that in figures 2.21, 2.22, 2.23 and 2.24, vertices show $X$-checks and faces show $Z$-checks. In fig.2.24, red line shows the logical-$Z$. Notice the code in fig.2.24 is the Shor's code.

#### 2.3.2.2 Toric Codes

In 1996, Kitaev offered the most well studied surface code in [57] which we call *toric codes*. Square tesselations generating a lattice with periodic boundary conditions wrapped around a torus gives us the structure in which qubits are placed on the edges. The qubits on upper and lower boundaries are identified while right and left boundary qubits are identified as well. Therefore, we have the torus shape.

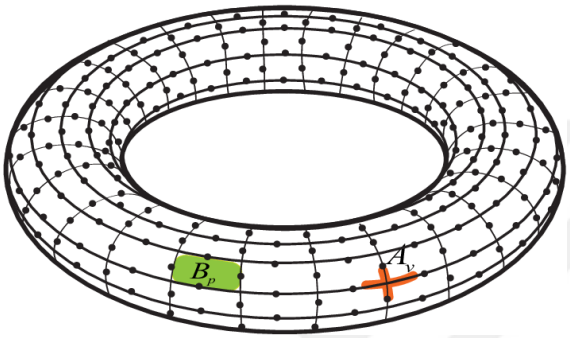



Figure 2.25: Toric code where dots are qubits and checks are shown [59]

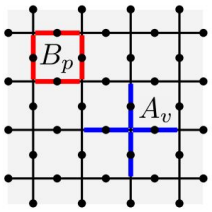



Figure 2.26: Plaquette ($B_p$) and vertex ($A_v$) operators which are constructed by applying Pauli-Z and Pauli-X on the edges of a face and the neighboring edges of a vertex respectively. [60]

Note that we have $2L^2$ edges in an $L \times L$ lattice meaning that $n = 2L^2$ and code parameters $[[n, k, d]] = [[2L^2, 2, L]]$.

In toric codes, we have two type of generators called *plaquette* and *vertex* operators. Plaquette operators are 4 Pauli-$Z$ operators acting on the edges of a square and vertex operators are 4 Pauli-$X$ operators acting on the neighboring qubits of a vertex such that

$$A_v = \bigotimes_{i \in v} X_i \quad \text{and} \quad B_p = \bigotimes_{i \in p} Z_i \tag{2.190}$$

where $v$ is the set of qubits neighboring the vertex and $p$ is the set of qubits surrounding the plaquette. Notice that this definition is the simplification of definitions in equation 2.189. In general, we have a subspace of degenerate ground states con-

structed with local Hamiltonians, and the Hamiltonian of toric codes is: [58]

$$H_{TC} = -\sum_{v} A_v - \sum_{p} B_p \tag{2.191}$$

Toric checks commute with their own kind (plaquettes with other plaquettes and vertices with other vertices) as well as neighboring plaquette and vertex operators. Therefore all checks commute. One important property of this checks arise from the fact that Pauli operators gives $\mathbb{I}$ once they applied twice resulting neighboring plaquette operators provide a new operator where Pauli operators acted on the boundaries while inner qubits remains invariant. This particular property enables us to construct stabilizer loops or logical operators.

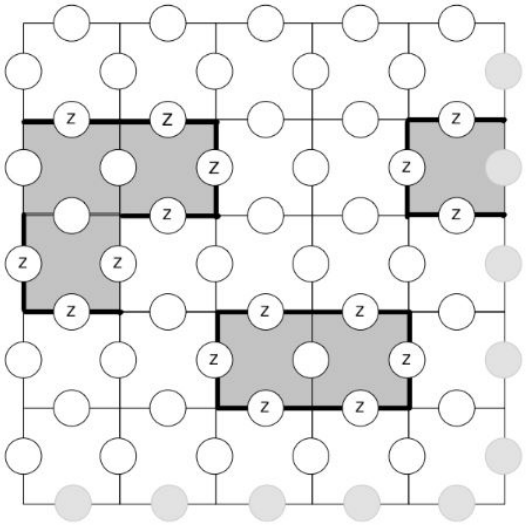


Figure 2.27: Generating $Z$-loops using plaquette operators [59]

Same is true for vertex operators but in the dual lattice which can be obtained by moving the lattice to half cell right and half cell down (the dashed lattice in Fig.2.29). Or we can think of it as changing vertex and plaquette operators while maintaining the boundary conditions, lattice size and the places of qubits (with orthogonal orientation to the places in primal lattice).

Note that there is $L^2$ plaquette (vertex) operator in an $L \times L$ (dual) lattice and therefore we have set of $L^2 - 1$ independent generators. Then we can find the number of encoded qubits for such toric code by substracting the number of independent generators from the number of qubits as $k = 2L^2 - 2(L^2 - 1) = 2$.

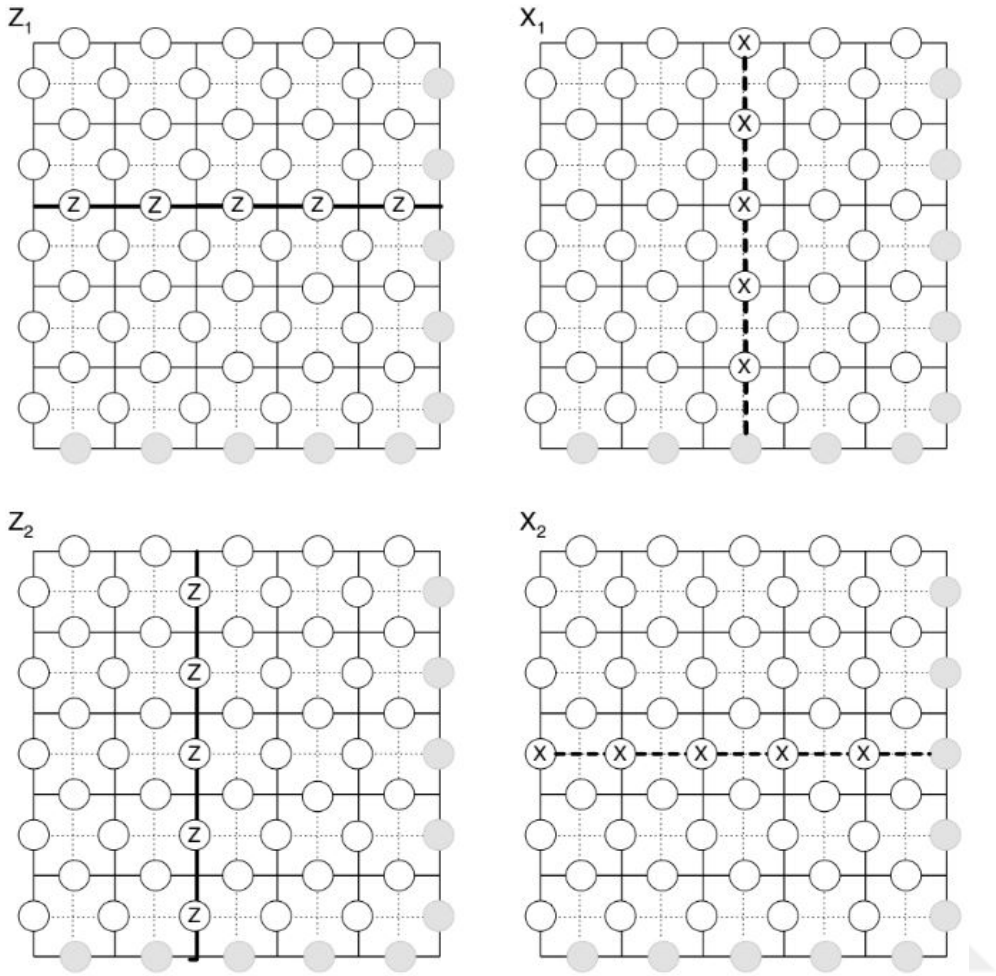



Figure 2.28: Logical operators [59]

The encoded logical operators are the loops around the torus. For example consider a $Z$-string which is the logical $Z$ operator and we apply a plaquette somewhere on that loop. Resulting new generator is another loop with different path adjusted by the plaquette we applied. The same is true for logical $X$ and vertex operator. Notice that since the code distance is minimum logical operator weight, and logical operator is a loop on the torus, then for an $L \times L$ toric code, distance is $L$. Also note that we can detect string of errors in this code. To illustrate consider a $Z$-error string (in Fig.2.29). We can detect that error by applying vertex operators at its ends giving us $-1$ eigenvalue. Then we can correct the syndrome by applying necessary set of plaquette operators.

We can generalize toric codes to $D$-dimensions following [81] in which Hastings

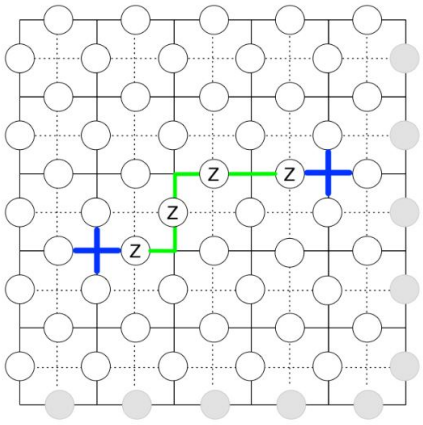



Figure 2.29: Z-string error and vertex checks at the ends [59]

et.al. discussed (p,q) toric codes such that p+q=D. For example ordinary 2D toric code is (1,1) while 4D toric code, qubits lie on the edges or the 1-cells while in 4D toric code which is (2,2), we have qubits on faces or 2-cells. For positive scalar $\kappa_A$ and $\kappa_B$ we have:

$$H = \kappa_A H_A + \kappa_B H_B \tag{2.192}$$

where

$$H_A = -\sum_{\{(p-1)-cells\}} \prod_{(p-1)-cell \ni \partial i)} X_i \quad \text{and} \quad H_B = -\sum_{\{(p+1)-cells\}} \prod_{i \in \partial((p+1)-cell)} Z_i \tag{2.193}$$

#### 2.3.2.2.0.1 Anyonic Model

We can model toric codes using 4 quasi-particles called *anyonic model*.[82] Those particles are

1. *vacuum particle* "1"
   meaning that there are no excitations in the lattice

2. *electric charge* "e" such that

$$|\phi\rangle = E\,|\psi\rangle \quad \text{at vertex: } A_v\,|\phi\rangle = -\,|\phi\rangle \tag{2.194}$$

3. *magnetic charge* "m" such that

$$B_p\,|\phi\rangle = -\,|\phi\rangle \tag{2.195}$$

4. *dion* "$\epsilon$"

   combination of electric and magnetic charges

In anyonic systems, combinations of particles are regulated by *fusion rules*. For particles $a$ and $b$ we denote fusion by $a \times b$ and fusion rules are:

$$\begin{aligned} a \times 1 &= a \\ e \times m &= \epsilon \\ a \times a &= 1 \\ \forall a \in &\{1, e, m, \epsilon\} \end{aligned} \tag{2.196}$$

We call anyon models obeying fusion rule *Abelian* and those models that don't obey called *non-Abelian*. Note that anyons are created in pairs at the ends of the string-like operators. Unlike anyons in Quantum Field Theory which require particle-antiparticle pairs to create vacuum, in toric code anyons can create it using same kinds. String-like and logical operators can be considered as the trajectories of anyons and pair production respectively. According to fusion rule, an anyon moves on a non-trivial path which is wrapped around the torus to annihilate at the point where it started. In fig.2.30(a) we see two anyons. Note that changing genus number changes degeneracy to $2^{2g}$ allowing $2g$ qubits to be encoded. In fig.2.30(b) we see an error which is two anyons created with a system which needs no energy.

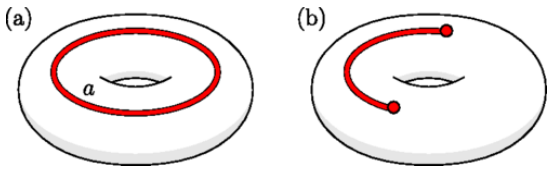



Figure 2.30: Anyons on a torus [82]

#### 2.3.2.3 Hyperbolic Surface Codes

We begin our discussion of hyperbolic surface codes by the bound given by [65] for 2D Euclidean surface codes:

$$kd^2 \leq cn \tag{2.197}$$

where $c$ is some constant. Due to this inequality, we see that both encoding rate ($\eta_R = k/n$) and distance are bounded such as:

$$\begin{aligned} \eta_R &\leq \frac{c}{d^2} \\ d^2 &\leq \frac{cn}{k} \end{aligned} \tag{2.198}$$

Note that this bound is not applicable for surfaces with negative curvature. For closed 2D surface codes and color codes, [67] showed the following inequality:

$$kd^2 \leq cn(\log k)^2 \tag{2.199}$$

where the row weight of the PCM decides the constant $c$. So, we see that the distance of color and LDPC surface codes can be logarithmic. However, [68] code having a minimum distance growing with $O(\sqrt{n})$ and [69] code family have minimum distance of $\Omega(\sqrt{n\sqrt{\log(n)}})$ which will be discussed later. And note that in this section, we follow [66].

Following the tradeoff in [67], we can encode more logical qubits using hyperbolic surface codes that is tilings of some 2D surface which gives us qubits with checks having $O(1)$ distance from eachother. Note that this structure is not isometric (distances are not preserved). Poincaré disc model is the best way of visualizing this structure which is the projection onto a unit disc (radius with 1). As an example, in fig.2.31 we see a $\{5, 4\}$-tiling of some hyperbolic plane in which 4 pentagons meet at each vertex. Here, we lost all distance and area information (meaning it is non-isometric) but all information of angles are preserved (it is conformal). All lines become arcs such that arcs intersects with eachother and boundaries with $\pi/2$ angle. Notice that the distances seem smaller as we go to the boundaries in some fractalic manner, yet all distances and areas of pentagons are the same.

Note that we need $\mathbb{H}^2$ to be isometric and have genus $g > 1$ so that we can encode several qubits. Moreover we can decompose our tiles since tiling is associated to a group whose elements are elementary triangular tiles. So, we decide $k$ by choosing a normal subgroup from the tiling group.

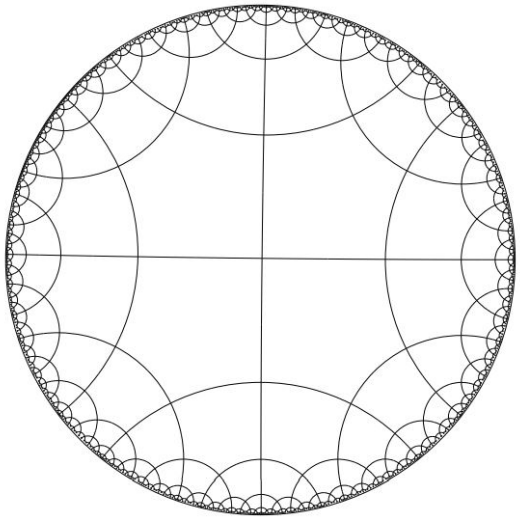


Figure 2.31: Poincaré disc of hyperbolic plane $\mathbb{H}^2$ with {5,4}-tiling [66]

Now we introduce Schläfli symbol $\{p, q\}$ which shows the structure of the tiling such that $q$ $p$-gons meet at each vertex. Notice that here, we have $p$ $Z$-stabilizer weight and $q$ $X$-stabilizer weight.

Consider Pauli subgroup $N(S)$ (where $N$ is the normalizer) in which type-$X$ and $Z$ elements have injection relation with cycles and cocycles. Non-boundary cycle operators act non-trivially while stabilizers act trivially on logical qubits. Due to $\delta_i = \partial_{i+1}^T$, we have $\dim H_1$ couples of non-(co)boundary (co)cycles. Such couples have odd overlap and other couples have even overlap. Those nontrivial loop couples are logical-$X$ and $Z$s. We can also introduce another measure (which is equivalent to distance), *combinatorial systole* ($csys$) which is the smallest length of the nontrivial cycle for a given tile.

Or we can simply calculate $n$ and $k$ by using the number of edges and how many restrictions are imposed by stabilizers as:

$$
\begin{aligned}
& n = \dim C_1 \\
& \quad \& \\
& k = n - \dim B^1 - \dim B_1 \\
& \quad = \dim C_1 - \dim Z_1^{\perp} - \dim B_1 \\
& \quad = \dim C_1 - (\dim C_1 - \dim Z_1) - \dim B_1 \\
& \quad = \dim H_1
\end{aligned}
\tag{2.200}
$$

Chains $E_x$ and $E_z \in C_1$ are associated with the Pauli errors and the support of a chain indicates the location of error. Now consider $E_z$. Since it is detected via $X$-stabilizers and those are located on vertices, we can say that this syndrome is $supp(\partial E_z)$. Or similar arguments can be carried for $E_x$ which gives us $\delta E_x$ as syndrome.
Let $P_z$ be some Pauli operator of $Z$-type. Then when we apply $P_z$ (with $supp(P_z) \in C_1$ whose boundary and syndrome matches), and turned back into code state. At this point, we look at the chain sum $E_z + supp(P_z)$ and check if it is in the boundary space $B_1$. Then applying a stabilizer solves our problem and logical qubits are unaffected. However if there is some non-trivial loop in $E_z + supp(P_z)$, then there is a damage on the encoded information. Correction of $E_x$ is done in the similar way with dual.
When there is an error, we have to pick and connect couples to minimize the possibility of logical operator occurring. Problem depends on how we describe the error and we expect errors with high weight have low probability of occurrence. Suppose chain $supp(P_z)$ has the minimum weight and there is no inconsistency between the chain and the syndrome. Then this problem is *minimum weight matching (MWM)* and we can solve this using blossom algorithm.

One thing to note here is that $S_z$ and $B_1$, $S_x$ and $B^1$, $N(S_x)_z$ and $Z_1$, $N(S_z)_x$ and $Z^1$, $n$ and $\dim(C_1)$, $k$ and $\dim(H_1)$, $d$ and $\min(csys, csys^*)$ are corresponding to each other from stabilizer codes and homology respectively.

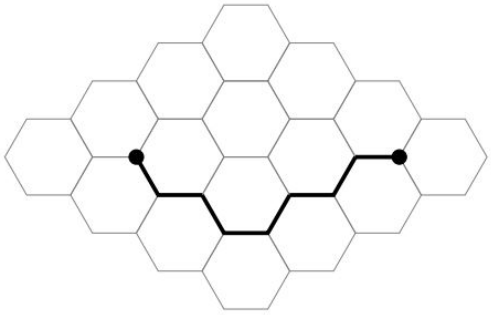


Figure 2.32: $E_z$ error chain with vertex boundaries [66]

#### 2.3.2.3.1 Tilings of Closed Surfaces

To describe surface tiles, in this topic we follow [66].

##### 2.3.2.3.1.1 Euclidean Tilings

Let denote 2D Euclidean plane with $\mathbb{E}^2$. Under composition, mapping set from $\mathbb{E}^2$ onto itself which has a property of preserving distances forms "group of isometries" $Isom(\mathbb{E}^2)$ with elements of either rotations, glide reflections (combination of reflections and translations) or translations.

Let $G \leq Isom(\mathbb{E}^2)$, then for $P \in \mathbb{E}^2$ where $P$ is a point, all the points that are mapped from $P$ by the elements of $G$ forms a set called the orbit of $P$ under $G$. As an example consider integer number group translated to the left. Then orbit of $P = (x, y) \in \mathbb{E}^2$ is $\{(x - a, y) | a \in \mathbb{Z}\}$.
A part of the plane containing at most one representative from each $G$-orbit within is called a fundamental domain of a group $G$, let's denote it with $\mathtt{F}$.

Let an action $g \in G$ is applied to the points on the Euclidean plane. Then this action also effects $\mathtt{F}$ of $G$ which denoted by $\mathtt{F} * g$. We call $e \in G \ni \mathtt{F} * e = \mathtt{F}$ as identity element with suitable action for group multiplication. Also latter means that $\mathtt{F} * (gh) = (\mathtt{F} * g) * h$.
Then define *tilings* (or *faces*) as set of finite plane parts covering Euclidean plane with non-overlapping or overlapping at the boundaries. Orbit of $\mathtt{F}$ produces tiling

naturally.

Tiling that the tiles are regular polygons (with edge-to-edge placing) is called *regular tilings* which are labelled with $\{p, q\}$ (Scläfli symbol) where $p$ is the number of sides of the polygons and at each vertex $q$ number of polygons meet. Note that the dual of $\{p, q\}$-tiling is $\{q, p\}$ and it is well defined.

Now introduce Wythoff's kaleidoscopic construction in which a regular tiling $\{p, q\}$ is related to a group of isometries $G = G_{p,q}$. In a regular $p$-gon, symmetry axes causes $2p$ symmetries by reflection and split the $p$-gon into $2p$ triangles with internal angles $\pi/2, \pi/q \; and \; \pi/p$.

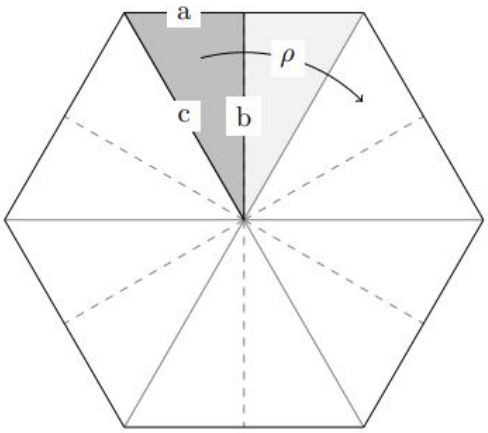



Figure 2.33: Rotation on a face [66]

Now consider fig.2.33 where reflections on $a, b$ and $c$ edges generate $G_{p,q}$ such that $a = c\sin(\pi/p)$ and $b = c\sin(\pi/q)$ and applying reflection twice gives identity ($a^2 = b^2 = c^2 = e$). Also for two reflections applied sequentially due to two lines with $\pi/k$ angle between them ($k \in \mathbb{Z}^+$) is the same as applying $2\pi/k$ rotation around the point where those two lines intersects, resulting that $k$ rotations is identity. Following is called the presentation of $G_{p,q}$:

$$G_{p,q} = \left\langle a, b, c \middle| a^2 = b^2 = (ab)^2 = (bc)^p = (ca)^q = e \right\rangle \tag{2.201}$$

To think of this group, we may consider all possible codewords containing $\{a, b, c, a^{-1}, b^{-1}, c^{-1}$. We can think group multiplication as concatenation of codewords while "subword" is the word such that group multiplication with the word gives identity ($g^{-1}g = e$).

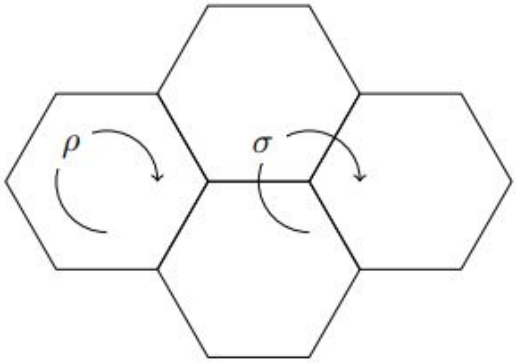



Figure 2.34: Rotation on a lattice [66]

In order to get a subgroup of $G_{p,q}$ spanned by elements preserving orientation, we use double reflections such that $\rho = bc$ and $\sigma = ca$ giving us $G^+_{p,q}$ that is:

$$G^+_{p,q} = \left\langle \rho, \sigma \middle| \rho^p = \sigma^q = (\rho\sigma)^2 = e \right\rangle \tag{2.202}$$

where $\rho$ is $2\pi/p$ rotation on a face and $\sigma$ is $2\pi/q$ rotation on a lattice CW. Note that we can cover whole map using a triangle and its $b$-reflection as $G^+_{p,q}$ fundamental domain since triangle generated by $G^+_{p,q}$-orbit covers only half of the plane. If we let the fundamental domain of $G^+_{p,q}$ be as a triangle, say $\mathtt{F}_0$, all other triangles can be considered as the projections of $\mathtt{F}_0$ onto $G^+_{p,q}$. In other words, for $g \in G^+_{p,q}$, $\mathtt{F} = g * \mathtt{F}_0$ where $\mathtt{F}$ is some triangle.

For a cyclic subgroup of $G^+_{p,q}$ (i.e. $\langle\rho\sigma\rangle = \langle e, \rho\sigma, (\rho\sigma)^2, \cdots, (\rho\sigma)^{p-1}\rangle$), edges of $p$-gons generated by $\{p, q\}$-tiling and triangles generated by $\rho\sigma$ are 1-to-1 (i.e. $g\langle\rho\sigma\rangle = \langle g, g\rho\sigma, g(\rho\sigma)^2, \cdots, g(\rho\sigma)^{p-1}\rangle$) meaning that those triangles are the left cosets of $\langle\rho\sigma\rangle$. Same relation also available for $\langle\rho\rangle$ and $\langle\sigma\rangle$ for faces and vertices respectively. Note that the lattice topology is encoded in $G^+_{p,q}$.

#### 2.3.2.3.1.2 Quotient Surfaces

Let $G \leq Isom(\mathbb{E}^2)$, then $\mathbb{E}^2/G$ defines a new space giving a set $\{P * g | g \in G\}$ $\forall P \in \mathbb{E}^2$. Assume that $\exists \varepsilon > 0 \ni \forall P \in \mathbb{E}^2$, then discs with radius $\varepsilon$ (called *injectivity radius* $R_{inj}$) in the neighborhood of all $P$ in orbit $P * G$ have no overlaps making $g \in G$

*discontinuous*. And if $g$ doesn't contain fixed points, it is *fixed-point free*. Therefore we don't have degeneracy and there is F of $G$ nearby all $P \in \mathbb{E}^2$. Then we say $\mathbb{E}^2$ covers $\mathbb{E}^2/G$ and $G$ is covering group. Note that the geometry within the disc with radius $R_{inj}$ is the same as the geometry of $\mathbb{E}^2$ and loops in $\mathbb{E}^2$ provides boundary for the faces they are located around. And assuming that all edges have unit length, $R_{inj}$ gives us a lower bound on distance $d$.
If $G \leq G^+_{p,q}$, then we have a regular tiling.
If $\mathbb{E}^2/G$ is isotropic, $i$-cells of tiling on $\mathbb{E}^2/G$ are labelled by $f \in G^+_{p,q}$ similarly. And if $fgf^{-1} \in G \; \forall f \in G^+_{p,q}$, $G$ is called *normal subgroup of* $G^+_{p,q}$ in which action elements don't change the labels.
If $G \leq G^+_{p,q}$ (or $G$ doesn't have glide-reflections), then $\mathbb{E}^2/G$ is orientable.
We can label vertices in $\mathbb{E}^2/G$ tilings with set in the form $f \langle\sigma\rangle G = \{f\sigma^n g | n \in \{1, \cdots, p\}, g \in G\}$ where $f \in G^+_{p,q}$, similar approach for edges with $\langle\rho\sigma\rangle G$ cosets and faces with $\langle\rho\rangle G$ cosets is also applicable. Note that if cosets $\langle\rho\rangle G$, $\langle\sigma\rangle G$ and $\langle\rho\sigma\rangle G$ share an element, $i$-cells of $\mathbb{E}^2/G$ is incident.

In a group, coset number of a subgroup is called *index of the subgroup* and $\mathbb{E}^2/G$ has a finite area if index of $G \in G^+_{p,q}$ is given finite.
So, to find orientable, closed, Euclidean surfaces, we look for the finite index and normal subgroup of $G^+_{p,q}$ with no fixed points.

**Low-Index Subgroup Algorithm** Now following the relation constructed above, we look how we can find normal subgroups of $G$. For given $G^+_{p,q}$, [73] shows an algorithm to find all normal subgroups $G$ and for this method, we follow this paper.

To begin with let $G^+_{p,q} = \langle \Xi, \mathcal{R} \rangle$ where for finite set $\Xi$ be the set of generators and $\mathcal{R}$ be the set of relators. Using algorithm offered in [74] for every conjugacy class in subgroups of $G^+_{p,q}$ we find a representative. Note that this doesn't go forever, instead we keep our search up to a chosen index number, call $J$. What algorithm does is basically searching through a Cayley tree backwards in which nodes correspond to the index number of subgroups of $G$. Each such generator is named after the level of the search, $k$-generator in level $k$. At each level we use coset enumeration for

finding necessary amount of right cosets of subgroup $G$ of that level and generate a coset table for $G$. In that table, rows represents the cosets while columns represents the generators of $\Xi$ and their inverses. When all entries in the coset table have definitions which satisfy the relators and generators, coset table is closed. Encountering with same coset with different indices is called *coincidence*, and in such situation, we replace the bigger number with the smaller one.

Since our search goes by row-by-row, left to the right in each row, new coset definitions are in normal order. So, if we see coset $i$ in column $c_i$ and the first entry of row $r_i$, and if $j > i > 1$, then $r_j > r_i$ or $r_j = r_i \wedge c_j > c_i$ is true.

For a stage in which the number of right cosets of that subgroup is bigger than $J$, then there exists at least one coincidence. So, we create new branches connecting to new nodes when beginning to the search of next level. To do that we have to identify the pair of cosets by forcing $Gg_i = Gg_j$ which is the same as adding $g_i g_j^{-1}$ to a set for $G$ containing generators. We denote those branches with $i = j$ and order them in the manner of increasing $j$'s.

Since we don't want to generate same group again, we also check for the position of the node in the tree. For a node, if there still exists possible forced coincidences: then we create new branches.

We take generators as output and new subgroup is found when the coset-table becomes closed while we do coset enumeration at a node. If we transversed the entire tree, than algorithm halts.

| n | k | d | csys | csys* | translations |
|---|---|---|---|---|---|
| 60 | 8 | 4 | 6 | 4 | $((\sigma\rho^{-1})^2\rho^{-1})^2$ |
| 160 | 18 | 6 | 8 | 6 | $\sigma\rho^2(\sigma\rho^{-1})^2\rho^{-1}\sigma^{-2}\rho^{-2}\sigma\rho^{-1}$ |
| 360 | 38 | 8 | 8 | 8 | $(\sigma\rho^2\sigma)^2(\rho^{-1}\sigma^{-2}\rho^{-1})^2$ |
| 1800 | 182 | 10 | 10 | 10 | $(\sigma\rho^{-1})^{10},\ \sigma\rho^2\sigma^2\rho^{-1}\sigma(\rho^2\sigma^{-1})^2(\rho\sigma^{-1})^2\sigma^{-1}\rho^{-2}\sigma\rho^{-1}$ |
| 1920 | 194 | 10 | 12 | 10 | $\sigma\rho^2\sigma^2\rho(\rho\sigma^{-1})^4\rho^{-1}(\rho^{-1}\sigma)^3\rho^{-1}$ |

Table 2.5: {5,4}-tiling compactification examples [66]

In the above table, we see possible translations allows us to compactify the lattice

and its dual. Note that once we translate a lattice and it becomes compact by one translation, combinatorial systole is the same as the length of that translation.

#### 2.3.2.3.1.3 Surfaces with Curvature

Curvature of an arc with radius $R$ is $1/R$ while curvature of a surface at a point $P$ can be found via two normal planes generating maximum and minimum radii $R_1$ and $R_2$ respectively. Here we define curvature $K$ which is the Gaussian curvature which is measured for following surfaces as:

- sphere: $K = 1/R^2$
  two arcs show the same direction
- saddle: $K < 0$
  two arcs show different directions (with different signs)
- Euclidean surface: $K = 0$

Note that (since we're dealing with triangles) internal angles of triangle adds up to: ($\Sigma_\Delta$: sum of internal angles of triangle)

- $\Sigma_\Delta > \pi$ for sphere
- $\Sigma_\Delta = \pi$ for Euclidean plane
- $\Sigma_\Delta < \pi$ for surface with $K < 0$

For a unit sphere $\mathbb{S}^2$, we can divide a regular $p$-gon into $2p$ triangles and regular $\{p, q\}$-tiling gives internal angles of $\pi/2,\ \pi/p\ \pi/q$. To have $\Sigma_\Delta > \pi$, $1/p + 1/q > 1/2$ must be satisfied leaving us with the possible tilings $\{3, 3\},\ \{3, 4\},\ \{3, 5\},\ \{4, 3\}$, and $\{5, 3\}$ which corresponds to the Platonic perfect solids.
For an Euclidean surface, $\{p, q\}$-tiling causes $\Sigma_\Delta = \pi$ condition to become $1/p + 1/q = 1/2$ leaving us with $\{3, 6\},\ \{4, 4\}$, and $\{6, 3\}$ tiles as regular.
For a surface with $K < 0$, $\Sigma_\Delta < 0$ dictates us $1/p + 1/q < 1/2$ and this condition is satisfied for infinite number of $p$ and $q$ values. Note that we call a surface with

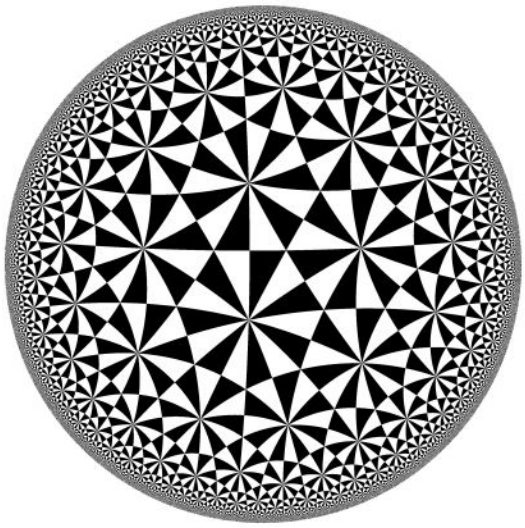


Figure 2.35: Poincaré disc of the $\{7,3\}$-tiling of $\mathbb{H}^2$. Note that blacks and whites stands for the fundamental domains such that same colors correspond to the same element of $G^+_{7,3}$ [66]

negative constant curvature $K = -1$ as hyperbolic plane $\mathbb{H}^2$.
Since sphere has only contractible loops, quotient surface it can produce is projective plane and it is not orientable. We generate it in such a way that a reflection due to a plane intersection with the sphere is taken out. [62]

We can use two non-colinear translations to generate closed quotient surface in Euclidean surface since all translations commute following $K = 0$. Surface $\mathbb{E}^2/G$ is torus for covering group $G$ with only translations while if $G$ has glide-reflections as well, surface $\mathbb{E}^2/G$ turns into a Klein bottle.

Killing-Hopf theorem allows us to express any closed surface with constant curvature as $\mathbb{S}^2/G$, $\mathbb{E}^2/G$ or $\mathbb{H}^2/G$.

#### 2.3.2.3.2 Hyperbolic Surface Having Open Boundaries

We can try to construct hyperbolic surface codes with open boundaries and since they

are associated with some planar graph, they might be better than the hyperbolic codes once they have better distance and encoding rate. In [66] it is shown that such code can be achieved by two ways. We pick one of them and give an example to it.

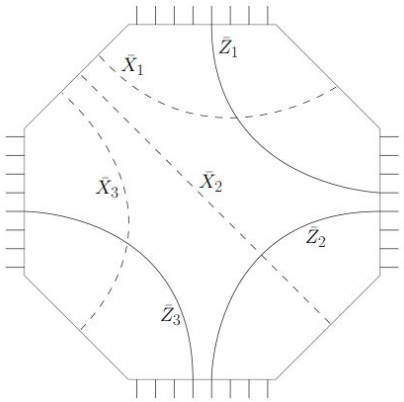



Figure 2.36: An octogon encoding 3 qubits with smooth and rough boundaries in an alternating manner. [66]

To begin with, let's consider a tiled torus having multiple genera. Then we cut this to create a surface as in the fig.2.36. Note that in that figure, boundary edges with fringes represents the *rough boundaries* while others represents the *smooth boundaries* which we can use to encode many qubits using that kind of surface if the lengths of the boundaries are sufficiently large. The surface boundary is divided into $2k$ parts and the encoding provides logical qubit number of $k-1$. In such structure, logical operators $\overline{X}$ and $\overline{Z}$ runs from one smooth boundary to another and one rough boundary to another respectively.
One upside of this code class once the qubits are represented on $\mathbb{E}^2$ is that the qubits can get closer or distant.

In [66] a bound for homological surface code family is given if they have boundaries as:

$$kd \leq cn \tag{2.203}$$

where $c$ is some constant. Notice that following this bound, distance of $\log(n)$ and encoding rate $\eta_R$ which is constant are forbidden.

#### 2.3.2.3.3 Example: A Small Planar Code

To generate a planar graph, we apply a partial regular tiling on hyperbolic plane. And to encode logical qubits, we modify the boundary. Before starting, we define single $p$-gon as level-1 $p$-gon and for each reflection, call it level-2, level-3 etc. As an example, consider $\{5, 5\}$-tiling:

i) Starting from a single level-1 $p$-gon, we reflect it to create level-2 $p$-gons. Reflecting those level-2 $p$-gons we get level-3 and so on. We do this for $m$ times to get faces from level-1 to level-$m$ and what we end up with is $\mathcal{G} = (V, E)$ planar graph. Here, each face is $Z$-check (or plaquette operator) and each vertex is $X^*$-check. Note that $Z$-checks act on boundary qubits on that face while $X^*$-checks acts on qubits next to the vertex. In the inner region, weight of $X$-checks is say, $q$. On the boundaries, for odd $q$, weight of $X$-checks is $(q-1)$, for even $q$, weight is either $q$ or 2.

   Notice that since there is no linear dependency between $Z$-checks and 1 between $X$-checks, the number of linearly independent $Z$-and$X$-checks are $F$ and $V - 1$ giving us $E = V - 1 + F$ under condition $\chi = 1$ ($\chi = V - E + F$ is the Euler characteristic) where $E = n$. Therefore this graph $\mathcal{G}$ encodes no qubits. To graph to encode qubits, we need to modify it. Note that this graph contains only smooth boundaries in which $X$-error string begins. In this graph, errors can be removed via $X^*$-checks, giving us no logical operators. In the case of rough boundaries, $Z$-error string may begin. We can prevent checks from removing strings by simply alternating between smooth and rough boundaries as in fig.2.36

ii) In the next step, we want to create rough boundaries. To begin with, we need to find the number of edges in which reflection process occur $E_{bound}$ with $|E_{bound}|$. To produce a code with qubits having equal distances, we divide $E_{bound}$ into $2k$ (where $k > 2$) portions of set giving us distinct smooth or rough boundaries with subset size $|E_{bound}|/2k$.

   In this code, logical-$X$, $\overline{X}_i$ ($i = 1, \cdots, k-1$) is from $i^{th}$ to $(i+1)^{th}$ smooth boundaries (or smooth boundaries between $i^{th}$ and $(i+1)^{th}$ rough boundaries)

while $\overline{Z}_i$ ($i = 1, \cdots, k-1$) is from $i^{th}$ and $(i+1)^{th}$ rough boundaries. Note that this code encodes $k-1$ qubits. To exploit the space in which qubits are distributed with best possible way, we choose $k$ in a way that shortest path going through the bulk and along perimeter are nearly the same.

In an edge region $E_{bound}^{region} \subseteq E_{bound}$ we create rought boundary with 3 steps.

Firstly, we remove all weight-2 $X$-checks (in fact we can remove more than weight-2 but that causes a smaller lattice) having 2 edges surrounded by $E_{bound}^{region}$. Note that we also remove an $X$-check with one edge in $E_{bound}^{region}$. Second, we look for qubits with only one plaquette is acting on and remove them thus having modified plaquettes at the boundary (weight-3 in toric code). By erasing an $X$-check we allowed a $Z$-string to run from that edge where we removed the $X$-check. Note that such string must only run in between rough boundaries. Also note that due to the facts that we removed an $X$-check and $ZZ$-checks with weight-2 commute with checks we have, we add a $ZZ$-check having weight-2 to stabilizer.

Another way of generating a graph that encodes $k-1$ logical qubits is that starting from some graph $\mathcal{G}$ that encodes no qubits, instead of reflection, rotation is used to generate next level $p$-gons. Rotating a $p$-gon by $2\pi/q$ about its vertices is the method here. In this method, we remove $X$-checks in the place where rough boundaries are created. So we don't need to introduce $ZZ$-checks having weight 2. Then we remove the qubits on which only one plaquette acts.

#### 2.3.2.3.4 Properties of Hyperbolic Codes

**Theorem 2.3.2** *Gauss-Bonnet Theorem:*

$$\int_{\mathcal{M}} K \, dA = 2\pi(2-2g) \tag{2.204}$$

*where $K$ is the Gaussian curvature, $\mathcal{M}$ is a orientable closed surface and $g$ is the genus of that surface.*

Since here we are dealing with the hyperbolic surfaces, Gauss-Bonnet theorem for $K = -1$ yields:

$$\begin{aligned} g &= \frac{area(\mathcal{M})}{4\pi} + 1 \\ area(\mathcal{M}) &= 4\pi(g-1) \end{aligned} \tag{2.205}$$

Then, are of regular hyperbolic $p$-gon $\Gamma$ is: [70]

$$area(\Gamma) = (p-2)\pi - \sum_{j=1}^{p} \theta_j \tag{2.206}$$

where $\theta_j$ are the internal angles of $\Gamma$. For $\{p, q\}$-tiles of $\mathcal{M}$, $\theta_i = 2\pi/q$ resulting:

$$area(\mathcal{M}) = \pi F(\pi - \frac{2p}{q} - 2) \tag{2.207}$$

where $F$ is the number of faces in $\{p, q\}$-tiling of $\mathcal{M}$. Then by equation 2.206 and equation 2.207, for $[[n, k, d]]$ code generated by $\{p, q\}$-tiling of surface $\mathcal{M}$ we have:

$$\eta_R \equiv \frac{k}{n} = 1 + \frac{2}{n} - \frac{2}{p} - \frac{2}{q} \tag{2.208}$$

where $\eta_R$ is the coding rate or yield. Notice that the coding rate gets bigger for higher $p$ and $q$ values. Smallest possible tiling of $p + q$ is $\{4, 5\}$-tiling and its dual is $\{5, 4\}$-tiling.

For smallest number of vertices, define an upper bound $\tilde{C}(p, q)$ on a regular hyperbolic map. Then for $\mathbb{H}^2/G$ of tiling $\{p, q\}$ such that its combinatorial systole is bigger than $r \in \mathbb{N}$ and number of edges $E \leq \frac{q}{2}\tilde{C}^r$ we have: [71]

$$\tilde{C} \leq \frac{1}{2}(5pq)^{16pq} \tag{2.209}$$

which provides a lower bound for distance with a logarithmically growing function $\tilde{C}(p, q)$.

For upper bound we have: [72]

$$d \leq \frac{p}{2} \log_{\sqrt{pq}}(2E) \tag{2.210}$$

In order to compute the distance of a given hyperbolic code, we can use the following algorithm:

i) Using Gaussian elemination, compute basis of $Z^1 = ker(\delta_1)$ where $\delta_1$ is the coboundary operator, in $\mathcal{O}(E^3)$ steps

ii) Take two copies of the lattice $L$ and $L'$

iii) Define new lattice $\Lambda$ for all elements $t$ in basis of $Z^1$

- If $\epsilon = (u, v)$ and $\epsilon' = (u', v')$ are in the support of $t$, we remove them from $L$ and $L'$ where $\epsilon$ and $\epsilon'$ are the edges in $L$ and $L'$ respectively.
- Add new edges from $u$ to $v'$ and $v$ to $u'$

iv) Compute the distance between $v$ and $v'$ in $\Lambda$ (i.e. using Dijkstra's algorithm in $\mathcal{O}(E\log(E))$ steps)

v) Determine systole via looping base points $v$ and $k$ elements of cocycle basis $t$ over in $\mathcal{O}(E^3 + E^2k\log(E))$ steps

For cosystole, we can do the same procedure in dual lattice.

#### 2.3.2.4 Freedman-Meyer-Luo Codes

Once record holder quantum code family until 2020, having $[[n, 2, \Omega(\sqrt{n}(\log(n))^{1/4})]]$ code parameters. [64] To illustrate, let's consider an example:
We take a Carthesian product within $[0, 1]$ on a closed hyperbolic surface $\Sigma_g$ where $g$ is the genus. We call smallest non-contractable loop on $\Sigma_g$ as $1 - systole$ ($sys_1(\Sigma_g)$). Two ends of $\Sigma_g \times [0, 1]$ are identified using $\sqrt{sys_1(\Sigma_g)}$ length twist (concept of twist is described in Fiber Bundle Codes). Then $1 - systole$ of generated 3D manifold is $\sqrt{sys_1(\Sigma_g)}$. We remove non-contractible loops with $sys_1(\Sigma_g)$ length due to $\Sigma_g$, resulting 3-manifold $\Sigma_g \times [0, 1]$. Note that the volume of this 3-manifold $\Sigma_g \times [0, 1]$ and the hyperbolic surface area $\Sigma_g$ is proportional due to the fact that the interval we've chosen is in unit length. Then the area using Gauss-Bonnet-Chern theorem: [75]

$$area(\Sigma_g) = 2\pi\chi(\Sigma_g) = \Theta(g) \tag{2.211}$$

So, we construct a 4D manifold via a loop $S^1 \ni |S^1| = \frac{g}{\sqrt{\log(g)}}$ and $\Sigma_g \times [0, 1]$ Carthesian producted resulting a 4-manifold with volume:

$$vol_4((\Sigma_g \times [0, 1]) \times S^1) = \frac{g^2}{\log(g)} \tag{2.212}$$

After uniformly tesselating the manifold, we have a homological quantum code such that we consider 1,2,3-cells as $X$-checks, qubits and $Z$-checks respectively and non-contractible surfaces within the manifold as logical operators. Area of non-contractible surfaces are 2-systoles with size $sys_2((\Sigma_g \times [0,1]) \times S^1) = \Theta(g)$ causing a bound on distance.

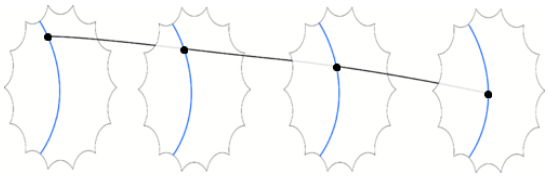


Figure 2.37: Freedman-Meyer-Luo code construction [64]

The geodesic we see (blue line) in fig.2.37 has length $\Theta(\log(g))$ and the points on the surface are moved $\Theta(\sqrt{\log(g)})$ along geodesic. Here black dot is the position of a point moving along geodesic due to $\Sigma_g \times [0,1]$ product. At the end of that product, we see beginning and end of the interval which have been identified resulting in twisted $\Sigma_g \times S^1$.

#### 2.3.2.5 Surface Analysis

So far, in surface codes, we looked at how to construct a surface but one may ask for its implementation. In this section, we study how we can implement a surface design as a quantum circuit, how to read errors on it and how to make simulations of QECCs.

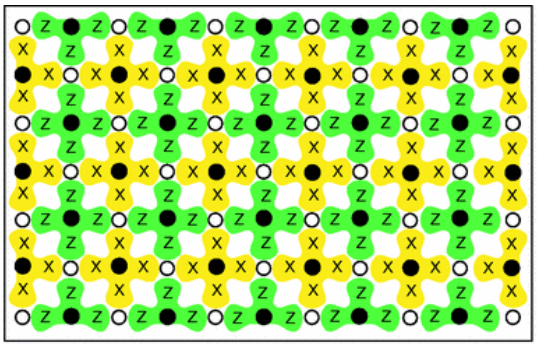


Figure 2.38: Surface indicating $X$ and $Z$-checks, data qubits and ancillas [76]

Consider fig.2.38 in which we see $X$-checks and $Z$-checks. Note that circles are qubits such that (•) are ancillary qubits and (○) are data qubits. In a tiling surface for example, when we say "edges corresponds to qubits", we mean data qubits and ancillas are on the vertices. Moving into dual construction turns data qubits into ancillas and vice versa. When checks observes an error on a data qubit, we see that in an ancilla. Observing two syndromes in the neighboring ancillas of a data qubit shows us that there is an error occured on that data qubit. On the other hand, if we observe an error on an ancilla alone in a cycle while not observing it in the next or vice versa and so on means that there is an error on that ancilla. In other words, we observe data qubit errors in space while ancilla errors in time. [76]

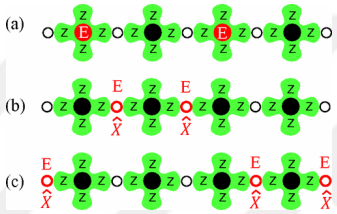



Figure 2.39: (a) syndrome measurement, (b) and (c) possible errors causing (a) [76]

In fig.2.39(a), we see a syndrome measurement while (b) and (c) are the possible error configurations. When we observe a syndrome on two ancillary qubits, this means that there is an error string in between them. And if we have error probability $p$, we can calculate the probability of (b) occuring as $p^2$ while (c) as $p^3$. Since $p < 1$, we can argue that configuration in (b) is most likely. Of course (b) causing the error is most likely but there still a chance that (c) caused it and when we choose (b) to correct the error, we make a misidentification. This misidentification happens with probability $p^{d_e}$ where $d_e$ is the *error dimension* and it is

$$d_e = \begin{cases} (d+1)/2, & \text{for odd } d \\ d/2, & \text{for even } d \end{cases} \tag{2.213}$$

Before we look how QEC simulations are done, we should look at how to convert a tiled surface into a quantum circuit. In fig.2.40, we see a $Z$-check in upper side with a,b,c and d data qubits and ancilla in the middle. What we do is basically applying a $CNOT$ gate from data qubits targetted to the ancilla to observe bit-flips. Notice

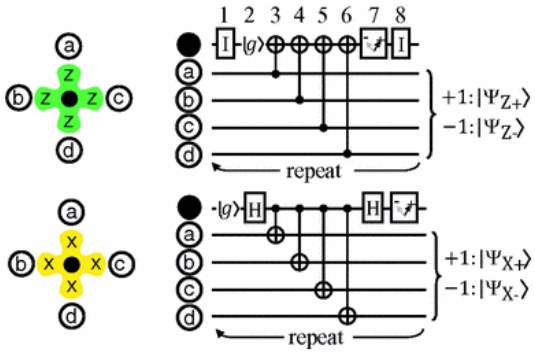



Figure 2.40: circuit implementation of $X$- and $Z$-checks [76]

that we have $\mathbb{I}$ in 1st and 8th steps which means doing nothing or waiting, this is for synchronizing the $Z$-checks to the $X$-checks. Same process is done in the $X$-check but with $H$ gates applied in order to change the basis from computational basis to Hadamard basis, so we observe phase-flip errors. Note that $CNOT$ gates look like controlling the ancilla while targeting to the data qubits, but in Hadamard basis, information is run from target to the control.

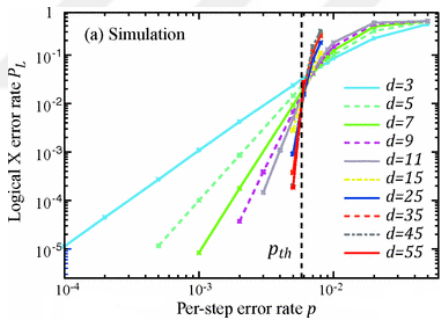



Figure 2.41: Graph of a simulation of a surface code [76]

Now to make simulation, assume that there is a probability of error $p$ at each step of the circuits in fig.2.40. Probability of a logical error is then:

$$P_L = \frac{d!d}{(d-1)!d_e!} p_e^{d_e} \tag{2.214}$$

where $p_e$ is the probability of error for each cycle, i.e. in fig.2.40 we have $p_e = 8p$. Then, for different distances, we repeat the simulation and we find threshold probability $p_{th}$ from the intersection of the lines (see the graph in fig.2.41. Note that for $p < p_{th}$, $P_L \sim p^{d_e}$.

# CHAPTER 3

# QUANTUM MEMORY CODES

## 3.1 Introduction

In this chapter, we look at the QECCs that might serve as quantum memories which we call *self-correcting quantum memory (SCQM)*. To say "memory", the quantum information is stored on the system for long time without need of extra effort. For this reason usually quantum states are protected via macroscopic energy gaps between the ground and excited states. Other mechanisms are possible as well, like entropic protection that we will discuss later in this chapter. Although no QECC for establishing a quantum memory demonstrated so far, but there are some candidates that might work with further research, which we will mention some of them. When we say quantum memory, it is required that the protection has to be passive, without the necessity of active QEC protecting the information; however we can still use active QEC during readout.

### 3.1.1 Caltech Rules for SCQM

There are some necessary conditions that a quantum memory has to satisfy. To understand it, suppose a model of self-correcting quantum memory in dimension of $D$. This model has to satisfy the followings: [92]

i) Spins in the system have to be finite dimensional with embedding in $\mathbb{R}^D$ having density which is also finite.

ii) Suppose a Hamiltonian describing finite number of interactions between particles with limited interaction range and strength. The system should evolve with

such Hamiltonian.

iii) Model has to have nontrivial codespace, that is its ground state space which is degenerate should encode one or more qubits.

iv) In thermodynamic limit, logical states should be rebust against the small changes in the system, called *perturbative stability*.

v) Decoding algorithm should be able to decode the encoded qubits in polynomial time.

vi) When the system-reservoir interaction in finite temperature with weak-coupling Markovian limit, encoded qubits should have lifetime growing exponentially as the spin number in the system increases.

Note that a Hamiltonian having a gap is not necessary condition though many self-correcting models require this.

### 3.1.2 Model for Finite Temperature

We make assumptions to investigate the change of quantum memory in thermal environment. First, assume that the reservoir is Markovian and the interaction between the environment and the memory is weak. Therefore once an information on the memory moved to the reservoir, it is lost. Moreover assume that the actions of reservoir on the memory components is local. So, qubit-reservoir couplings are independent from each other. [82]

We can study how quantum memory evolves by looking into the quantum memory and the reservoir it is coupled to, but for practical purposes we can just consider how memory is evolved. To be able to do that, memory has to evolve into its *Gibbs state* which is:[82]

$$\rho_\beta = \sum_j \frac{e^{-\beta\varepsilon_j}}{\mathcal{Z}} |e_j\rangle\langle e_j| \quad \ni \mathcal{Z} = \sum_j \langle e_j|e^{-\beta H}|e_j\rangle \tag{3.1}$$

where $\varepsilon_j$ are the energy eigenvalues, $\mathcal{Z}$ is canonical partition function and $\beta = 1/T$. Note that $|e_j\rangle$ form an orthonormal basis set for Hamiltonian of memory. Now let's think of some $\mathscr{E}$ event happening during some thermal evolution such that $\langle e_f|\mathscr{E}|e_i\rangle = 1$ where $|e_i\rangle$ and $|e_f\rangle$ are the eigenstates of memory with subscripts standing for initial and final states with energy difference in between:[82]

$$\omega_{\mathscr{E}} = (\varepsilon_f - \varepsilon_i) \tag{3.2}$$

So, for an event $\mathscr{E}$ during thermal evolution, frequency dependent rate equation is:[82]

$$\gamma(\omega_{\mathscr{E}}) = \frac{\omega_{\mathscr{E}}}{1 - e^{-\beta\omega_{\mathscr{E}}}} \tag{3.3}$$

When we look at the above equation, we realize that increasing the energy of the system is suppressed exponentially. If equation 3.3 satisfies the detailed balance condition:[83]

$$\gamma(\omega_{\mathscr{E}}) = e^{\beta\omega_{\mathscr{E}}}\gamma(-\omega_{\mathscr{E}}) \quad \forall\mathscr{E} \tag{3.4}$$

then the system evolves into Gibbs state. Notice that equation 3.3 satisfies detailed balance condition. In the meantime, we find master equation by taking both memory and the Hamiltonian that evolves the system into account. This Hamiltonian acts on the reservoir such that: [82]

$$H = H_M \otimes \mathbb{I}_R + \mathbb{I}_M \otimes H_R + \sum_j W_j \otimes f_j \tag{3.5}$$

where subscripts $M$ and $R$ indicates the memory and reservoir. In $\sum_j W_j \otimes f_j$ term which describes the memory-reservoir interaction, $W_j$ and $f_j$ act on memory and reservoir respectively while both of them are Hermitian. The master equation is:[82]

$$\frac{d\rho}{dt} = i[H_M, \rho] + \mathcal{L}(\rho) \tag{3.6}$$

where Liouvillian $\mathcal{L}$ is the description of memory-reservoir interaction so that[82]

$$\mathcal{L}(\rho) = \sum_{j,\omega\geq 0} \mathcal{L}_{j\omega}(\rho) \tag{3.7}$$

such that

$$\mathcal{L}_{j\omega}(\rho) = \hat{g}_j(\omega)\left(V_j^\dagger[\rho, V_j] + [V_j^\dagger, \rho]V_j + e^{-\beta\omega}(V_j[\rho, V_j^\dagger] + [V_j, \rho]V_j^\dagger)\right) \tag{3.8}$$

where $\hat{g_j(\omega)}$ is the reservoir power spectrum and Fourier transforming $V_j = V_j(\omega)$ gives $W_j$ such that $U(t)W_jU^\dagger(t) = \sum_\omega V_j(\omega)e^{-i\omega t}$ for $U(t) = e^{-iH_M t}$. If we take the interaction terms acting on memory $W_j$'s as one qubit Pauli matrices $X_k$ and $Z_k$, then the density matrix is always diagonal on its own eigenbasis:[82]

$$\rho(t) = \sum_k p_k(t) \left|e_k\right\rangle\!\left\langle e_k\right| \tag{3.9}$$

Master equation can be rewritten in the following form:[82]

$$\frac{dp_l}{dt} = \sum_{k\neq l}(\Gamma_{k\to l}p_k - \Gamma_{l\to k}p_l) \tag{3.10}$$

where $\Gamma_{k\to l}$ are transition rate from $k$ state to $l$ state.

Suppose that an error process $\mathscr{E}$ changes the system such that $k$ eigenstate turns into $l$ eigenstate via energy $\omega_{\mathscr{E}}$ giving us transition rates:[82]

$$\Gamma_{k\to l} = \gamma(\omega_{\mathscr{E}}) \propto \frac{\hat{g}(|\omega_{\mathscr{E}}|)}{|1 - e^{-\beta\omega_{\mathscr{E}}}|} \tag{3.11}$$

In order to achieve thermalization, we must have ergodic $W_j$'s meaning that the reservoir can access and interact with all possible memory configurations. To satisfy ergodicity, the only operator can commute with both the Hamiltonian and the all of the interaction terms in the set $\{W_j\}$ must be $\mathbb{I}$. For instance if the $W_j$'s are one qubit Pauli operators, then we achieved the ergodicity due to the fact that only operator to commute with all of the set members is $\mathbb{I}$ even though $H$ commutes with all.[82]

### 3.1.3 Coherence Time

To measure how well a quantum memory work, we can look at the amount of time in which encoded quantum information is decodable with high probability when thermal evolution is introduced and we call it *coherence time* $\tau$. Coherence time is mostly related to the inverse reservoir temperature $\beta$ and system size. We can fix the rest with equation 3.3.[82]

To illustrate how we will calculate $\tau_{qmem}$, consider a toric code with 4 qubits with Hamiltonian:[82]

$$H_{TC4} = -\frac{\Delta}{2}(S_x + S_z) \tag{3.12}$$

where

$$S_x = \bigotimes_{j=1}^{4} X_j \quad \& \quad S_z = \bigotimes_{j=1}^{4} Z_j \tag{3.13}$$

are stabilizers. Note that we call this 4-qubit states as *GHZ (Greenberg-Horne-Zeilinger) states* in which we have even number of qubits in state 1 and the logical states are defined on two qubits on which one-qubit Pauli operators are acted one by one.[82]

Suppose that $\overline{X} = X_1 X_3$, we know that $X_1$ will be applied followed by $X_3$. Since $X_1$ anticommutes with $S_z$, due to equation 3.12 it causes $\Delta$ energy cost and due to equation 3.3 it takes $e^{\beta\Delta}$ time. Then $X_2$ is applied and its process is relaxation process which is much faster, therefore we neglect it. What we have at the end is something similar to Arrhenius' law:[82]

$$\tau \sim e^{\beta\varepsilon} \tag{3.14}$$

where $\varepsilon$ is the energy creating the logical error which is equals to the gap $\Delta$ for small toric codes. Now, we want to have better coherence time than equation 3.14.[82]

### 3.1.4 Energy Barrier

One way to achieve passively protected QEC codes is to have macroscopic energy barriers such that there should be huge necessity of energy to create an excitation in the system. Consider logical ground state $|\psi\rangle$ such that the system has Pauli Hamiltonian which commute and logical error $\overline{X}$ causing excited state $\overline{X}|\psi\rangle$ where $\overline{X}$ can be described by:[82]

$$\overline{X} = \prod_{i=1}^{M} U_i \tag{3.15}$$

where $U_i$ are caused by reservoir and applied sequentially to generate $\overline{X}$. Denoting the energy required for each component $U_i$ of the logical error as $\varepsilon_i$, we can look for the most energy intensive step $\max_i(\varepsilon_i)$ and try to minimize it by looking at every possible sequence that creates $\overline{X}$. Once we achieve this, we call the activation energy that is the minimum of maximum energy barrier and denote $\varepsilon_B \equiv \min_j(\max_i(\varepsilon_i))$ where $j$ runs through all possible sequences to generate given logical error.[82]

As an example use toric code Hamiltonian (equation 2.191) in which logical error is generated via two anyons moving on some non-contractible loop. We can create this

by flips to some qubits on some non-contractible loop followed by completing the loop by applying flip to those which we didn't flip at first. The first way of creating logical error requires energy only at the beginning by creation of anyons giving us $\max_i(\varepsilon_i) = 2$ while second way requires $\max_i(\varepsilon_i) = L$ since it completes a loop, so $\min_j(\max_i(\varepsilon_i)) = 2$ giving $\varepsilon_B = 2$. In quantum memories, $\varepsilon_B$ grows with system size and $\tau_{qmem}$ grows with Arrhenius' law.[82]

### 3.1.5 Free Energy

Looking at energy barrier for $\tau_{qmem}$ is not always reasonable, so, instead we may look at its entropic character. One way is to look at its *free energy* using which we analyse how system evolves in temperatures that can't be neglected.[82]

$$F = E - \frac{S}{\beta} \tag{3.16}$$

where $O$ is the offset caused by entropy and a given event occurs with energy cost $F$. Since free energy approach takes the effects caused by entropy, provides better understanding on coherence time as:[82]

$$\tau \sim e^{\beta F} \tag{3.17}$$

### 3.1.6 Simulating Finite-Temperature Effects

In this subsection, we look at how we can make Monte Carlo simulations for given finite-temperature model.[82]

Let $|\psi(t)\rangle$ be an eigenstate of commuting Pauli Hamiltonian $H$. To model noise, we take a noise as congruent events turning eigenstates of the Hamiltonian into each other. Denoting an event with $\mathscr{E}$, we try to get such events satisfying $|\psi(t+\delta t)\rangle = \mathscr{E}\,|\psi(t)\rangle$ for some $\delta t$. Note that usually ground state is taken as $t = 0$ case and the simulation is carried out for a time duration $t_{max}$ and $\mathscr{E}$ acts similar to $X_i$, $Y_i$ or $Z_i$ since we've working with commuting Pauli Hamiltonian. Then due to equation 3.3 solved for $H$ and $|\psi(t)\rangle$ gives the probability distribution for event $\mathscr{E}$:[82]

$$\mathcal{P}_{\mathscr{E}} = \frac{\gamma(\omega_{\mathscr{E}})}{\sum_{\mathscr{E}} \gamma(\omega_{\mathscr{E}})} \tag{3.18}$$

where the total rate runs over every possible $\mathscr{E}$. The interval between each $\mathscr{E}$ is $\delta t$ and it is exponentially distributed random variable:[82]

$$\delta t = -\frac{\ln(\mathscr{r})}{\sum_{\mathscr{E}} \gamma(\omega_{\mathscr{E}})} \tag{3.19}$$

where random variable $\mathscr{r} \in (0, 1]$ is uniformly chosen, and let the total rate be $\mathtt{R} \equiv \sum_{\mathscr{E}} \gamma(\omega_{\mathscr{E}})$. So, after a time $\delta t$, event $\mathscr{E}$ is occurred and we have state $|\psi(t + \delta t)\rangle = \mathscr{E} |\psi(t)\rangle$. Whenever a step is calculated, we check total time $t_{tot} = t + \delta t < t_{max}$. If so, next step is calculated, if not simulation stops. Averaging over the data, we get $\tau$. We can define $\tau$ as the average of time it takes to decoder which we apply repeatedly to fail once or the time that the success rate of decoder falls below some threshold value.[82]

### 3.1.7 Thermodynamic Stability Criteria

In this section, following [96] we analyze SCQMs in thermodynamic limit. Results Chesi,et.al. found are, for a quantum memory at nonzero temperature, some bound on relaxation rate, and for stabilizer codes they showed that logical operators have vanishing thermal expectation values.

#### 3.1.7.1 Thermal Fragility

To consider a perturbation in Hamiltonian, let

$$H_\eta = H - \eta \cdot \overline{P} \tag{3.20}$$

where $\overline{P} = (\overline{X}, \overline{Y}, \overline{Z})$ and $\eta$ is the perturbation term. Note that $\eta$ is a field of symmetry-breaking and due to that logical operators have expectation value that is finite.

Assume that $\eta = \eta\hat{z}$ and denote degenerate eigenkets of Hamiltonian having energy $E_z$ and $\overline{Z}$ eigenvalue $\lambda = \pm 1$ as $|\zeta, \lambda\rangle$. Notice that $H |\zeta, \lambda\rangle = E_\zeta |\zeta, \lambda\rangle$ and $\overline{Z} |\zeta, \lambda\rangle = \lambda |\zeta, \lambda\rangle$. Then the canonical partition function is $\mathcal{Z}_\eta = \mathrm{Tr}\big(e^{-\beta H_\eta}\big)$ and setting temperature $T \equiv 1/\beta$ yields: [96]

$$\mathcal{Z}_\eta = \sum_{\zeta,\lambda} e^{-\beta E_\zeta} \langle \zeta, \lambda | e^{\beta \eta \overline{Z}} | \zeta, \lambda \rangle \tag{3.21}$$

since $\sum_\zeta \langle \zeta, \lambda | e^{\beta \eta \overline{Z}} | \zeta, \lambda \rangle = 2\cosh(\beta\eta)$ we have:

$$\mathcal{Z}_\eta = \mathrm{Tr}\left(e^{-\beta H}\right) \cosh(\beta\eta) \tag{3.22}$$

In [97] we see that $\left\langle \overline{Z} \right\rangle_\eta$ don't depend on $H$ and gives only energy level degeneracy.

$$\left\langle \overline{Z} \right\rangle_\eta = \frac{\sum_{\zeta,\lambda} e^{-\beta E_\zeta} \langle \zeta, \lambda | \overline{Z} e^{\beta \eta \overline{Z}} | \zeta, \lambda \rangle}{\sum_{\zeta,\lambda} e^{-\beta E_\zeta} \langle \zeta, \lambda | e^{\beta \eta \overline{Z}} | \zeta, \lambda \rangle} = \tanh(\beta\eta) \tag{3.23}$$

Since equation 3.23 is independent of $H$ and tends to go zero for small $\eta$, the average tends to go zero in the thermodynamic limit. For system with physical qubit number $n$:

$$\lim_{\eta \to 0} \lim_{n \to \infty} \left\langle \overline{Z} \right\rangle_\eta = 0 \tag{3.24}$$

So, the expectation value of the logical operator of thermal state (that is $e^{-\beta H_\eta}$) is not nonzero since field of symmetry-breaking is not sufficiently strong. Using the proof of equation 3.24 we see that the argument is valid for stabilizer codes so that in the presence of vanishing symmetry-breaking field $\eta$ and finite temperature, all logical operators have vanishing thermal expectation value. [96]

#### 3.1.7.2 Error Corrected Logical Operators

Now, following suitable choice of symmetry-breaking field, we try to focus the weight of $e^{-\beta H_\eta}$ around $\left|\overline{0}\right\rangle$. Consider $\left|\overline{0}\right\rangle \ni \overline{Z} \left|\overline{0}\right\rangle = \left|\overline{0}\right\rangle$ and $E \left|\overline{0}\right\rangle \ni \{E, \overline{Z}\} = 0$ where $E \left|\overline{0}\right\rangle$ is a state with single-qubit error. Note that in the low temperature, probability of observing $E \left|\overline{0}\right\rangle$ state is low, but this changes as we reach the thermodynamic limit, therefore it is better to consider *error-corrected logical operators (ECLOs)* (or named as in [85] *dressed quantum operators*) instead of $\left\langle \overline{Z} \right\rangle$ as discussed in [85]. Let's define a *trace-preserving completely positive (TPCP)* transformation that makes error-correction for operator in Hilbert space: [96]

$$\Phi_{EC}(O) = \sum_\zeta \Pi_\zeta Cr^\dagger(\zeta) O Cr(\zeta) \Pi_\zeta \tag{3.25}$$

where $Cr(s)$ is the operator correcting Pauli operators with syndrome $\zeta$, $O$ is the operator and $\Pi_\zeta$ is the projection operator onto codespace such that $\Pi_\zeta = E\Pi_0 E^\dagger$. So, ECLOs are:[96]

$$\overline{X}^{EC} = \Phi_{EC}(\overline{X}) \quad \& \quad \overline{Z}^{EC} = \Phi_{EC}(\overline{Z}) \tag{3.26}$$

Note that $\overline{X}^{EC}$ and $\overline{Z}^{EC}$ don't need to be Pauli operators but they satisfy $\overline{X}^{EC^2} = \overline{Z}^{EC^2} = \mathbb{I}$ and $\overline{X}^{EC}\overline{Z}^{EC} = \overline{Z}^{EC}\overline{X}^{EC}$. For eigenvalues $\lambda_z(\zeta), \lambda_x(\zeta) \in \{-1, +1\}$ such that $Cr^\dagger(\zeta)\overline{X}Cr(\zeta) = \lambda_x(\zeta)\overline{X}$ and $Cr^\dagger(\zeta)\overline{Z}Cr(\zeta) = \lambda_z(\zeta)\overline{Z}$ we have:

$$\overline{X}^{EC} = \overline{X}\sum_\zeta \lambda_x(\zeta)\Pi_\zeta \quad \text{and} \quad \overline{Z}^{EC} = \overline{Z}\sum_\zeta \lambda_z(\zeta)\Pi_\zeta \tag{3.27}$$

Note that ECLOs commute with all generators in the stabilizer group and the aforementioned discussions show us that expectation values of ECLOs dissappear when there is symmetry-breaking field:[96]

$$\lim_{\eta\to 0}\lim_{n\to\infty}\left\langle \overline{Z}^{EC}\right\rangle_\eta = 0 \tag{3.28}$$

As an example consider 2D Ising model with spontaneous symmetry-breaking:

$$H_b = -J\sum_{\langle i,j\rangle} Z_i Z_j - b\sum_{i\in\Lambda} Z_i \tag{3.29}$$

where the sum $\langle i, j\rangle$ is over pair of nearest neighbour sites, $b$ is external magnetic field and on full lattice $\Lambda$ $i$ and $j$ labels faces. Then at low temperatures we have:[96]

$$\lim_{b\to 0}\lim_{n\to\infty}\langle Z_j\rangle_b \neq 0 \quad \forall j \text{ lattice sites} \tag{3.30}$$

Note that it is better to choose a perturbation with extensive symmetry-breaking since while $\eta$ symmetry-breaking field can act on a single site on the lattice, $b$ symmetry-breaking field can act on all sites. As a result, summation of different versions of logical operators as symmetry-breaking field is better choice. We can get extensive operator by multiplying $\overline{X}$ with stabilizer code elements.[96]

#### 3.1.7.3 Relaxation Rate in Quantum Memory Hamiltonians

Let the quantum memory system Hamiltonian $H$ and the algebra $\mathcal{A}$ defines the operators which act on $H$. [96]

**Definition 3.1.1** *Suppose there exists some projection operator* $\Pi$ *onto a subspace of* $\mathcal{H}$ *and operator* $O \in \mathcal{A} \ni [\Pi, O] = 0$. *If* $[E, O]\Pi = 0 \; \forall E \in \mathcal{E}$ *(where* $E \in \mathcal{E} \ni \mathcal{E} \subset \mathcal{A}$ *is the set of errors) is satisfied, then* $O$ *is robust against all* $E$ *errors in* $\Pi$ *subspace.*

Note that we call $\tilde{X}, \tilde{Z} \in \mathcal{A}$ satisfying $\tilde{X}^2 = \tilde{Z}^2 = \mathbb{I}$ and $\{\tilde{X}, \tilde{Z}\} = 0$ without the necessity of being a Pauli product or single qubit operator as *Pauli-like observable.*[96]

Now suppose we describe the change in the system with Markovian master equation:

$$\frac{d\rho}{dt} = -i[H, \rho] + \mathcal{L} \tag{3.31}$$

where Liouvillian operator

$$\mathcal{L} : \mathcal{A} \to \mathcal{A} \text{ and } \mathcal{L} = \sum_{k,\omega} \mathcal{L}_{k,\omega} \tag{3.32}$$

such that $\mathcal{L}_{k,\omega}(\rho) = L_{k,\omega}\rho L_{k,\omega}^{\dagger} - \frac{1}{2}\{\rho, L_{k,\omega}^{\dagger} L_{k,\omega}\}$. Here $L_{k,\omega}$ are the quantum jump operators carrying energy $\omega$ to the reservoir. If $\mathcal{E}_{\mathcal{L}} \subset \mathcal{A}$ be the set containing the quantum jump operators from $\mathcal{L}$, Markovian master equation 3.31 becomes after integration:

$$\rho(t)\Phi_t(\rho(0)) \quad \& \quad \Phi_t = e^{-it[H,\bullet]+t\mathcal{L}} \tag{3.33}$$

Note that after defining trace norm as $\|O\|_1 \equiv \mathrm{Tr}\left(\sqrt{OO^{\dagger}}\right)$ where $O$ is some operator, we can find $\mathcal{L}$ strength with:

$$\|\mathcal{L}\|_1 = \max_{V \in \mathcal{A}} \|\mathcal{L}(V)\|_1 \tag{3.34}$$

applied to $\|V\|_1 \leq 1$ where $V = V^{\dagger}$.[96]

**Theorem 3.1.1** *If* $\mathcal{L}$ *is a Liouvillian operator and* $\mathcal{E}_{\mathcal{L}}$ *is the quantum jump operator set of that Liouvillian, then Liouvillian operators' fixed point* $\mathcal{L}(\rho_\beta) = 0$ *is* $\rho_\beta \sim e^{-\beta H}$ *Gibbs state. Let on a subspace* $\Pi$ *we have* $\tilde{X}, \tilde{Z} \in \mathcal{A}$ *operators that are Pauli-like and robust against* $\mathcal{E}_{\mathcal{L}}$ *error set, and assume* $[\tilde{X}, H] = [\tilde{Z}, H] = [\Pi, H] = 0$. *Then there are some TPCP maps that are encode and decode:*

$$\Phi_{in} : \mathbb{L}(\mathbb{C}^2) \to \mathcal{A} \quad \& \quad \Phi_{out} : \mathcal{A} \to \mathbb{L}(\mathbb{C}^2) \tag{3.35}$$

*satisfying followings:*

$$\|\Phi_t \circ \Phi_{in}(q) - \Phi_{in}(q)\|_1 \leq 8t\|\mathcal{L}\|_1 \,\mathrm{Tr}(\mathbb{I} - \Pi)\rho_\beta \tag{3.36}$$

*and*

$$\Phi_{out} \circ \Phi_{in}(q) = q \tag{3.37}$$

*where $q$ is a single-qubit state and $t \geq 0$. [96]*

For a system with $n$-qubits and $\|\mathcal{L}\|_1 \leq poly(n)$, qubit can be stored safely with:

$$\tau_{qmem} \sim \frac{1}{poly(n)\epsilon_{qmem}} \tag{3.38}$$

where $\tau_{qmem}$ is *storage time* and $\epsilon_{qmem} = \mathrm{Tr}(\mathbb{I} - \Pi)\rho_\beta$ is *relaxation rate*. Note that RHS of equation 3.36 can be used as an upper bound for how well an initial state $q$ is approximated by $\Phi_{out} \circ \Phi_t \circ \Phi_{in}(q)$[96]

We can use equation 3.38 as a storage time bound if:

i) $\epsilon_{qmem} \leq e^{-n^\kappa}$ where $\kappa > 0$
   Systems having macroscopic energy barriers (proportional to $n^\kappa$) around the states that are orthogonal to $\Pi$ can be considered this way. As an example, 4D toric code is one of them which we will discuss in the next section.

ii) $\epsilon_{qmem}$ is small polynomially while sufficiently large degree give fast-growing $\tau_{qmem}$ with $n$
   In this one, energy barriers grow with $n$. For instance, $\epsilon_{qmem} \leq n^{-\kappa\beta} = e^{-\kappa\beta\log(n)}$ where $\kappa > 0$ meaning that the growth depends on the temperature. If $T < T_c$ then, $\epsilon_{qmem}$ decays in such a way that $\tau_{qmem}$ grows with $n$. In [98], $\tau_{qmem}$ grows polynomially using some self-consistent gap and its logarithmic divergence at room temperature. As a result, it might be possible to achieve quantum memory with less spatial dimensions.[96]

In [99] Davies gives a description for system-reservoir with weak coupling that has the interaction Hamiltonian:

$$H_{int} = \sum_{j=1}^{M} A_j \otimes B_j \tag{3.39}$$

where operators $A_k$ and $B_k$ are acting on few system qubits and reservoir respectively. Here $A_k$ is coupling operator which pumps $\omega$ amount of system energy to the

reservoir. Defining Liouvillian operator as:[96]

$$\mathcal{L}(\rho) = \sum_{j} \sum_{\omega} h(j,\omega)(L_{j,\omega}\rho L_{j,\omega}^{\dagger} - \frac{1}{2}\{\rho, L_{j,\omega}^{\dagger} L_{j,\omega}\}) \tag{3.40}$$

where $h(j,\omega)$ is (obtained by transforming $B_k$s' autocorrelation function using the Fourier transform, or) jump operator probability satisfying detailed balance condition (equation 3.4) as $h(j,-\omega) = e^{-\beta\omega}h(j,\omega)$ and we can consider jump operator as the part of coupling operator carrying energy to the reservoir such that $A_j(t) \equiv e^{iHt}A_j e^{-iHt} = \sum_{\omega} L_{j,\omega}e^{-i\omega t}$. So Fourier summation of jump operators gives time-dependent coupling operator. Here we note that detailed balance condition for $h(j,\omega)$ (which we need to include reservoir temperature into the equation) tells us that Gibbs state $\rho_\beta$ is equilibrium state of $\mathcal{L}$.[96]

Now, we use theorem 3.1.1 for Davies master equation. Here we want $\tilde{X}$ and $\tilde{Z}$ be reboust agaainst $\mathcal{E}_{int} = \{A_j\}$ error set which contains the errors caused by couplings between system and reservoir. Note that we assume that for all $\omega$, $[H,\Pi] = [H,\tilde{X}] = [H,\tilde{Z}] = 0$. Then $[\tilde{X}, A_j]\Pi = 0$ giving us $[\tilde{X}, L_{j,\omega}]\Pi = [\tilde{Z}, L_{j,\omega}] = 0$.
Assuming $\|A_j\| \leq 1 \ \forall j,\omega$ we have $\|\mathcal{L}\|_1 \leq 2M \max_{j,\omega}(h(j,\omega))$ where $M$ is the number of terms in $H_{int}$ (equation 3.39). Define $h_{max} \equiv \max_{j,\omega}(h(j,\omega))$ and assume that $h(j,\omega) \leq h_{max}$ normalizes $B_j$s.
Supposing on a subspace $\Pi$, $\tilde{X}$ and $\tilde{Z}$ Pauli-like operators robust against couplings between system and reservoir and $[H,\tilde{X}] = [H,\tilde{Z}] = [H,\Pi] = 0$, and following theorem 3.1.1, we can store a qubit protected from errors with time:[96]

$$\tau_{qmem} \sim \frac{1}{M h_{max} \epsilon_{qmem}} \tag{3.41}$$

Note that for $A_j$, $M \in \mathcal{O}(n)$.

Now consider a Hamiltonian with a gap $\Delta$ that is constant. Here what we mean by gap is the energy difference between ground and excited states. For reservoir temperature $T \ll \Delta$, Boltzman factor $e^{-\beta\Delta}$ rules any excitation from the ground state giving us:[96]

$$\frac{1}{\tau_{qmem}} \leq \mathcal{O}(n)e^{-\beta\Delta} \tag{3.42}$$

### 3.1.8 Thermal Stability in High Dimensions

In this section we study the stability of finite-temperature systems in higher dimensions following [84] which is the discovery of stability increasing with dimensionality. We begin our discussion with classical Ising model, then move to quantum cases.

Consider a 2D lattice $L \times L$ having $L^2$ number of spins located on the vertices where spins interact with their nearest-neighbour via:

$$E(\mathbf{s}) = -\frac{1}{2} \sum_{<i,j>} s_i s_j \tag{3.43}$$

where $s_i = \pm 1$ is the spin value for $+1$ up or $-1$ down and $< i, j >$ labels the couple of vertices which are at the ends of edges. Note that $\mathbf{s}$ stores information of all spins on the lattice. So, this Hamiltonian can be considered as a classical memory with binary encoding of a bit on the magnetization defined as $\overline{s} \equiv \sum_i s_i / L^2$. Here we have ground states $\overline{s} = \pm 1$ associated with $s_i = \pm 1 \; \forall i$. In fig.3.1 we see the regions

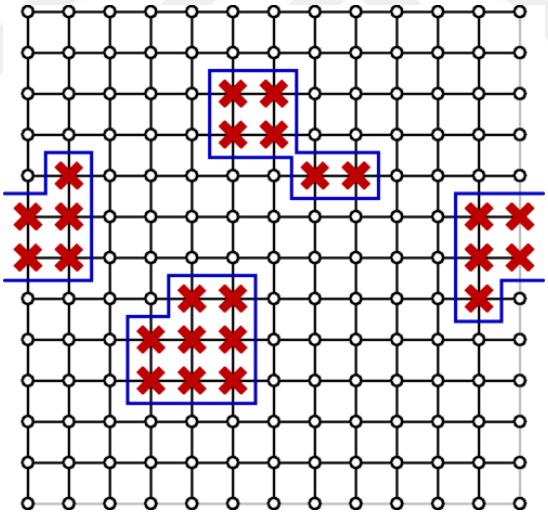


Figure 3.1: Droplets [82]

where information is flipped called *droplets*. Following that Hamiltonian cause some energy cost due to a neighbouring couple of opposite signed bits, droplets cost energy that is growing with its boundary length. We call a side of a boundary *Peierls contour* $b$.

For all practical purposes, we see that thermodynamic limit causes ground state prob-

ability to converge at zero. However since we're interested in the signs of the magnetization, information in bits is stored. So, all we need to look at is whether the thermal average value of the magnetizations absolute value is nonzero when we have finite but appropriately low temperature, in thermodynamic limit:

$$\langle|\overline{s}|\rangle = \sum_{s\in K} |\overline{s}| p(\mathbf{s}) \tag{3.44}$$

where system being in a specific configuration according to some thermal Gibbs distribution probability is $\mathcal{P}(\mathbf{s}) = e^{-\beta E(\mathbf{s})}/\mathcal{Z} \ni$ partition function $\mathcal{Z} = \sum_{\mathbf{s}\in K} e^{-\beta E(\mathbf{s})}$ with all possible configurations forming set $K$.

To get $\langle|\overline{s}|\rangle$ suppose that we have a lattice with most of the spins are in $+1$ state ($\overline{s} > 0$). Then we can call the regions of down states as "inside of the boundary". Now let $\langle N_-\rangle$ be thermal average of down state number and $l$ be the length of contour. Notice that if the length of the contour is fixed, then maximum possible number of spins within that contour would be $l^2/8$ following $l \sim 2L$ for a lattice with PBCs, just below the limit down states becoming dominant. Then we have:

$$N_-(\mathbf{s}) \leq \sum_b \frac{l_b^2}{8} \delta_b(\mathbf{s}) \tag{3.45}$$

where $\delta_b(\mathbf{s}) = \begin{cases} 1, & \text{if } b \in \mathbf{s} \\ 0, & \text{otherwise} \end{cases}$ and $l_b$ is the length of the boundary of contour b.

Now we construct another bound for loops that are possible. Consider a walker starting from a point in $L^2$ possibilities and moving in either north, south, west or east giving us one in four, followed by a direction which is not the opposite of what have been done in last time to prevent moving back. All closed loops are counted $l$ times since we can begin from $l$ faces which contour encounters, giving us $4 \cdot 3^{l-1} L^2/l$. Then suppressing $l_b = l$ by Boltzmann factor such that $\langle\delta_b\rangle \leq e^{-\beta l}$:

$$\langle N_-\rangle \leq \frac{L^2}{6} \sum_{l=4,6,\cdots} l 3^l e^{-\beta l} \tag{3.46}$$

In the limit of infinite volume:

$$\langle N_-\rangle \lesssim 27 L^2 e^{-4\beta} \frac{2 - 9e^{-2\beta}}{(1 - 9e^{-2\beta})^2} \tag{3.47}$$

where $e^{-2\beta} < 1$. For configurations of $\overline{s} < 0$, symmetry imposes $\langle N_+\rangle = \langle N_-\rangle$. Now divide $K$ into subsets where $\overline{s} > 0$: $K_+$ and $\overline{s} < 0$: $K_-$. Since $\overline{s} = 0$ configurations

are not magnetized, they can be neglected. Then we have:

$$\langle|\overline{s}|\rangle = \sum_{s\in K_+} |\overline{s}|p(\mathbf{s}) - \sum_{s\in K_-} |\overline{s}|p(\mathbf{s}) \tag{3.48}$$

To our arguments valid $|K_+| \geq L^2/2$ giving

$$\sum_{s\in K_+} |\overline{s}|p(\mathbf{s}) \geq \frac{1}{2} - \frac{\langle N_-\rangle}{L^2} \tag{3.49}$$

and similarly

$$\sum_{s\in K_-} |\overline{s}|p(\mathbf{s}) = \frac{\langle N_+\rangle}{L^2} = \frac{\langle N_-\rangle}{L^2} \tag{3.50}$$

resulting in:

$$\langle|\overline{s}|\rangle \geq \frac{1}{2} - \frac{2\langle N_-\rangle}{L^2} \tag{3.51}$$

implying if we have finite $\beta$, we have nonzero results not depending on system size. Assuming infinite volume, magnetization in the model stays stable.

## 3.2 4D Toric Code

We discussed 2D toric codes in surface codes section in chapter 1 in detail. Now we discuss toric codes defined in 4D with slight differences. In 4D, toric code has $N \times N \times N \times N$ lattice. Qubits are located on the faces. There is 6 face orientation for each vertex, such that $n = 6N^4$. We define PBCs on 4D. Half of the local stabilizers associated with 1D edges while the rest corresponds to 3D cubes. There is a $X^{\otimes 6}$ operator for every 1D edge where the edge lies on one of the 6 2D faces. Also there is a $Z^{\otimes 6}$ operator for every 3D cube which has 6 2D faces. Any two edge and cube stabilizers overlap if the edge belongs to the cube. This overlap happens on two faces. As a result, all stabilizers commute. There are 6 logical operator pair that anticommute with each other while commuting the other stabilizers and there is a pair for every 6 plane orientation. [91]

4D toric code have Hamiltonian: [82]

$$H_{4DTC} = -\sum_e A_e - \sum_c B_c \tag{3.52}$$

where

$$A_e = \prod_{e \in \partial f} X_f \quad \& \quad B_c = \prod_{f \in \partial c} Z_f \tag{3.53}$$

### 3.2.1 PBC Notation in 4D Lattice

We can identify every vertex on a 4D lattice with four-vector $\mathbf{v} = \langle v_0, v_1, v_2, v_3 \rangle \in \mathbb{Z}_N^4$. There are 4 edges $\hat{e}$, 6 faces $\hat{f}$ and 4 cubes $\hat{c}$ (having lower corner identified with $\mathbf{v}$) for every vertex $\mathbf{v}$. We can describe those orientations with binary four-vectors: [91]

$$\hat{e}, \hat{f}, \hat{c} \in \{(v_0, v_1, v_2, v_3) |\ v_i \in \{0, 1\}\} \tag{3.54}$$

where for edge $\hat{e}$: $\sum_{i=0}^{3} v_i = 1$, for face $\hat{f}$: $\sum_{i=0}^{3} v_i = 2$, for cube $\hat{c}$: $\sum_{i=0}^{3} v_i = 3$. Note that we can identify physical qubits with $[\mathbf{v}, \hat{f}]$ tuple in which $\hat{f}$ and $\mathbf{v}$ labels the orientation of the plane and lower side corner respectively.[91]
Stabilizers possessing 6 engaging physical qubits are:

- $X$-type edge stabilizers:

$$E_{\mathbf{v},\hat{e}} = \bigotimes_{\hat{e} \subset \hat{f}} X_{\mathbf{v},\hat{f}} \otimes X_{\mathbf{v}-\hat{f}+\hat{e},\hat{f}} \tag{3.55}$$

- $Z$-type cube stabilizers:

$$C_{\mathbf{v},\hat{c}} = \bigotimes_{\hat{f} \subset \hat{c}} Z_{\mathbf{v},\hat{f}} \otimes Z_{\mathbf{v}+\hat{c}-\hat{f},\hat{f}} \tag{3.56}$$

And for every face orientation $\hat{f}$ there is a logical operator pair that anticommute such as:

$$\overline{Z}_{\hat{f}} = \bigotimes_{n,m=1}^{N} Z_{n\hat{e}_1 + m\hat{e}_2, \hat{f}} \quad \& \quad \overline{X}_{\hat{f}} = \bigotimes_{n,m=1}^{N} X_{n\hat{e}_3 + m\hat{e}_4, \hat{f}} \tag{3.57}$$

where $\hat{e}_1 + \hat{e}_2 \equiv \hat{f}$ and $\hat{e}_3 + \hat{e}_4 \equiv \hat{f}^{\perp}$. Then $\overline{X}_{\hat{f}}$ and $\overline{Z}_{\hat{f}}$ anticommute when they encounter on qubit $(\mathbf{0}, \hat{f})$[91]

### 3.2.2 Quantum Toom's Rule in 4D

This tool is useful for a dissipation mechanism which prevents errors to pile up and recovery procedure for information which erases the errors so that information can

be restored. In classical 2D version, it flips the bit if the southern neighbours are different from the bit of interest. On the other hand in 4D, we phase-flip the qubit if the edge $X$-stabilizer in the south of the qubit is not satisfied and bit-flip if the cube $Z$-stabilizer in the south of the qubit is not satisfied. So, we make local rotation based on neighbouring stabilizer state. The jump superoperator that describes the update rule is:[91]

$$J^X_{\mathbf{v},\hat{f}}(\rho) = X_{\mathbf{v},\hat{f}}\Pi^z_{\mathbf{v},\hat{f}}\rho\Pi^z_{\mathbf{v},\hat{f}}X_{\mathbf{v},\hat{f}} + \Pi^{z\perp}_{\mathbf{v},\hat{f}}\rho\Pi^{z\perp}_{\mathbf{v},\hat{f}} \tag{3.58}$$

for every qubit $(\mathbf{v}, \hat{f})$. Here $\Pi^z_{\mathbf{v},\hat{f}}$ is the projection operator onto the subspace in which bit-flip acted on the qubit $(\mathbf{v}, \hat{f})$ and $\Pi^{z\perp}_{\mathbf{v},\hat{f}}$ is the orthogonal subspace. If we set $\hat{f} = \hat{e}_1 + \hat{e}_2$ the projector turns into:

$$\Pi^z_{\mathbf{v},\hat{f}} = \frac{1}{4}(1 - E_{\mathbf{v},\hat{e}_1})(1 - E_{\mathbf{v},\hat{e}_2}) \tag{3.59}$$

which can be done for the other projector as well.[91]

### 3.2.3 Error Correction Operators and Recovery

For jump superoperators can be considered as recovery operators we have the commutation relation:

$$[J^X_{\mathbf{v},\hat{f}}, J^Z_{\mathbf{v}',\hat{f}'}] = 0 \tag{3.60}$$

Note that due to the neighbouring plaquette, same type recovery operators may not commute. To define recovery operator in terms of local recovery rules, we need to specify application order explicitly. Then we can define error-corrected operators $X^{EC}_{\hat{f}}$ and $Z^{EC}_{\hat{f}}$ as:

$$\mathrm{Tr}\Big(X^{EC}_{\hat{f}}\rho\Big) = \mathrm{Tr}\Big(\overline{X}_{\hat{f}}J\rho\Big) \quad \& \quad \mathrm{Tr}\Big(Z^{EC}_{\hat{f}}\rho\Big) = \mathrm{Tr}\Big(\overline{Z}_{\hat{f}}J\rho\Big) \tag{3.61}$$

or

$$X^{EC}_{\hat{f}} = \overline{J}(\overline{X}_{\hat{f}}) \quad \& \quad Z^{EC}_{\hat{f}} = \overline{J}(\overline{Z}_{\hat{f}}) \tag{3.62}$$

Meaning that if we evaluate less noisy state, error-corrected logical observables give protected results and they coincide with logical operators in the noiseless subspace. [91]

### 3.2.4 Master Equation

Here we study the master equation that contain locally depolarizing noise term with strength $\Gamma_\epsilon$ and Lindblad terms that are expected to prevent error clusters to bet bigger. We begin with simulated master equation: [91]

$$\frac{d\rho}{dt} = \mathcal{L}\rho = \Gamma\mathcal{L}_{4D-Toom}\rho + \Gamma_\epsilon\mathcal{L}_{dep}\rho \tag{3.63}$$

where $\mathcal{L}_{4D-Toom}$ is dissipative protection:

$$\begin{aligned}\mathcal{L}_{4D-Toom}\rho = &\sum_{\mathbf{v},\hat{f}} L^z_{\mathbf{v},\hat{f}}\rho(L^z_{\mathbf{v},\hat{f}})^\dagger - \frac{1}{2}\{(L^z_{\mathbf{v},\hat{f}})^\dagger L^z_{\mathbf{v},\hat{f}}, \rho\}_+ \\ &+ L^x_{\mathbf{v},\hat{f}}\rho(L^x_{\mathbf{v},\hat{f}})^\dagger - \frac{1}{2}\{(L^x_{\mathbf{v},\hat{f}})^\dagger L^x_{\mathbf{v},\hat{f}}, \rho\}_+\end{aligned} \tag{3.64}$$

and protecting Lindblad operators are:

$$L^x_{\mathbf{v},\hat{f}} = X_{\mathbf{v},\hat{f}}\Pi^z_{\mathbf{v},\hat{f}} \quad \& \quad L^z_{\mathbf{v},\hat{f}} = Z_{\mathbf{v},\hat{f}}\Pi^x_{\mathbf{v},\hat{f}} \tag{3.65}$$

note that Lindblad operators in equation 3.65 corresponds to quantum jump superoperators $J^{\{X,Z\}}_{\mathbf{v},\hat{f}}$ which are Toom-like. [91]

### 3.2.5 Toric Code at Finite Temperature

Now we discuss the thermal effects on toric codes at finite temperature. It will be discussed in two regions, small and large critical system sizes. Here the critical system size depends on the cost of energy for an excitation and the temperature. Required energy for an excitation is usually considered as its mass and it is equivalent to the strength of the interaction ($\Delta$). Note that the anyons of toric code during thermal equilibrium has average density $\rho \sim e^{-\beta\Delta}$

For a system having size $L$, anyon couple number is: [82]

$$\langle N\rangle \sim \frac{L^2\rho}{2} = \frac{L^2 e^{-\beta\Delta}}{2} \tag{3.66}$$

According to above equation 3.66, we divide toric code thermal dynamics into two regions with $\langle N\rangle \lesssim 1$ and $\langle N\rangle \gg 1$.

#### 3.2.5.1 System with Small Size

For a system with $L \lesssim e^{\beta\Delta/2}$, equation 3.66 gives $\langle N \rangle \lesssim 1$ which we consider as small system. In such case, we can neglect the probability of more than one anyon couple occurs. So, a logical error will most probably caused by one anyon couple. Toric codes with small size at low temperature should provide good memory. Denoting the time for pair production $\tau_{pp}$ and anyons moving up to distance $L/2$ (see fig.3.2(b)) $\tau_m$, we have coherence time: [82]

$$\tau_{small} = \tau_{pp} + \tau_m \tag{3.67}$$

Note that for an anyon couple exceeds $L/2$ distance, encoded information is not

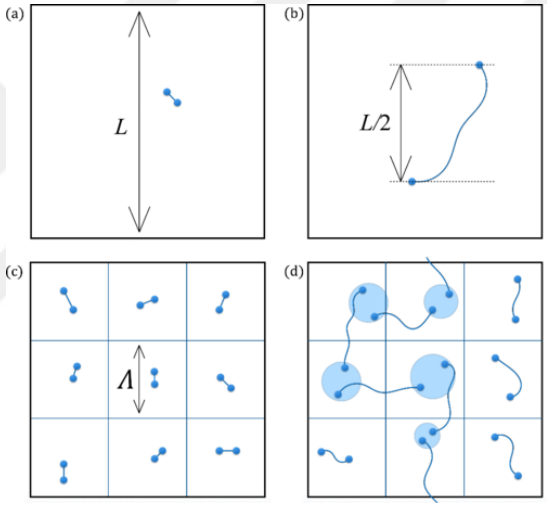



Figure 3.2: Anyon pairs in small and large system sizes [82]

retrievable. Since what matters is only the distance between the anyons which is governed by unbiased random walk, we can fix an anyon and look for the relative motion of the other and simulate the walk in $(L/2)^2$ steps for $L/2$ distance in $2D$. Using equation 3.19 we find step time as: [82]

$$\delta t = \frac{1}{8\gamma(0)} \tag{3.68}$$

where 8 in the denominator is due to possible paths that anyon may follow. So, we have:

$$\tau_m \simeq \frac{1}{8\gamma(0)} \left(\frac{L}{2}\right)^2 = \frac{\beta L^2}{32} \tag{3.69}$$

For pair production time, we have to consider pairs that annihilate using random walk as well as total rate, since some anyon couples annihilate each other before reaching distance $L/2$. Following energy required for pair production is $2\Delta$, we have a total rate $\mathtt{R}_0 = 2L^2\gamma(-2\Delta)$ giving us time as $1/\mathtt{R}_0 = 1/(2L^2\gamma(-2\Delta)) \sim e^{2\beta\Delta}/L^2$. To calculate random walk, let $\mathcal{P}(L,\beta)$ be the probability of anyon couple reaches $L/2$ before annihilating each other. Note that for a random walker, probability of walker not going back into where it started in first $t$ steps is approximately $1/\ln(t)$. Then for $(L/2)^2$ steps, we have $1/(2\ln(N/2))$. In addition, we have to take couples that annihilate each other immediately after the production into account which is $1/(1+\tilde{c}_0\beta)$ where $\tilde{c}_0$ is a constant while $\tilde{c}_0\beta$ indicates the relative probability of nearest neighbour anyons annihilate each other, giving us: [82]

$$\mathcal{P}(L,\beta) \sim \frac{1}{\ln(L/2)}\frac{1}{1+\tilde{c}_0\beta} \tag{3.70}$$

Therefore we have:

$$\tau_{pp} \simeq \frac{1}{\mathtt{R}_0}\frac{1}{\mathcal{P}(L,\beta)} \sim (1+\tilde{c}_0\beta)\frac{\ln(L/2)e^{2\beta\Delta}}{L^2} \tag{3.71}$$

So, coherence time is:

$$\tau_{small} = \tau_{pp} + \tau_m = (1+\tilde{c}_0\beta)\frac{\ln(L/2)e^{2\beta\Delta}}{L^2} + \frac{\beta L^2}{32} \tag{3.72}$$

Notice that in small size systems, coherence time mostly depends on $\tau_{pp}$ due to exponential term of $\beta$ while $\tau_m$ is only in $\mathcal{O}(\beta)$. [82]

#### 3.2.5.2 System with Large Size

We consider systems that the equation 3.66 gives $\langle N\rangle \sim L^2\rho/2 \gg 1$ as large size systems in which several anyon couples are produced in a uniformly distributed manner. In such systems, equilibrium density $\rho \sim e^{-\beta\Delta}$ and anyon couples occupy $2/\rho$ area which can be approximated as $L' = (2/\rho)^2 \sim e^{\beta\Delta/2}$ (see fig.3.2(c)). As some of the couples have distance $L'$, decoder has significant probability of failing and some series of errors can spread throughtout the entire system (see fig.3.2(d)).
In large system, error is created by pair production followed by diffusion in $L' \times L'$ region and we say system is failed when anyons from neighbouring regions come

close together. We can estimate the coherence time of large system with the time in the case of failure of a typical region allowing us to use the results of small system case by setting $L' = L/2$ resulting: [82]

$$\tau'_m \sim \beta L'^2 \simeq \beta e^{\beta\Delta} \tag{3.73}$$

and

$$\tau'_{pp} \sim \frac{e^{2\beta\Delta}}{L'^2}(1+5\beta)\ln(L') \tag{3.74}$$

Now let $L' = \tilde{c}'_0 e^{\beta\Delta/2}$ to make $\tau'_{pp}$ depending only on $\beta$ where $\tilde{c}'_0$ consists of constants in $\rho$, and if we put this $L'$ back into equation 3.74 we get: [82]

$$\tau'_{pp} \sim e^{\beta\Delta}(1+\tilde{c}_1\beta+\tilde{c}_2\beta^2) \tag{3.75}$$

where $\tilde{c}_1$ and $\tilde{c}_2$ are some constants. Notice that both $\tau'_m$ and $\tau'_{pp}$ are exponentially dependent on $\beta$, as a result we don't neglect any of them. So we have:

$$\tau_{large} \simeq \tau'_{pp} + \tau'_m \simeq e^{\beta\Delta}(1+\tilde{c}'_1\beta+\tilde{c}_2\beta^2) \tag{3.76}$$

where $\tilde{c}'_1$ is some new constant with motional terms within. Note that in this model, we simplified the thermal effects in such a way that speed of particles diffusing or probability of pairs annihilate themselves are not predicted well. Therefore we may ignore terms that are not exponential in $\beta$ by setting $\tilde{c}'_1 = \tilde{c}_2 = 0$ we get coherence time: [82]

$$\tau_{large} \sim e^{\beta\Delta} \tag{3.77}$$

### 3.2.6 Thermal Dynamics

If the quantum system that we want to solve contains many-body, analyzing it is very hard and thus we try to simplify it following [85] in which we use observables $X^{EC}$ and $Z^{EC}$ that are anticommute with each other. Let we have observable $O$, we can find its decay rate via: [82]

$$\lambda(O) = -\operatorname{Tr}\big(\rho_B O^\dagger \mathcal{L}(O)\big) \tag{3.78}$$

where $\lambda$ is decay rate of observables, $\mathcal{L}$ is Liouvillian and $O$ is the observable such that $\operatorname{Tr}(\rho_B O) = 0$ and $\operatorname{Tr}\big(\rho_B O^\dagger O\big) = 1$. For one qubit Pauli operators, Alicki et.al.[85]

showed that: [82]

$$\mathcal{F}(\rho(t)) \equiv \langle\psi|\rho(t)|\psi\rangle \geq \frac{1}{2}(e^{\lambda(\overline{X}^{EC})t} + e^{\lambda(\overline{Z}^{EC})t}) \tag{3.79}$$

where $\mathcal{F}$ is the fidelity of some decaying encoded state $\rho(t)$ with initial state $\rho(0) = |\psi\rangle\langle\psi|$. Note that ECLOs we used here ($\overline{X}^{EC}$ and $\overline{Z}^{EC}$) are mentioned before in equation 3.26. After this point all we have to do is to bound the decay rates from above using Peierls' argument we've discussed before. Error loops can be corrected via $\Phi_{EC}(O)$ with high probability. Peierls' argument also shows that loop-excitations having length larger than that can be corrected are suppressed exponentially. Moreover considering the effect of Liouville on ECLOs via Peierls' argument shows the same errors are again exponentially suppressed. Therefore 4D toric code is self-correcting for suitably low-temperatures. [82]

## 3.3 3D Quantum Memory

In this section, we discuss Haah's cubic code [86] which provides partial self-correction. We introduce better notation for corner operator commutation relations compared to the ones in [86]. As far as our knowledge, equation 3.91 has not been published so far. Although a quantum memory in 3D would be difficult for manipulation and readout, for the case which we don't find any other option than 3D quantum memories, that equation might be helpful.

### 3.3.1 Partial Self-Correction

We call systems having polynomial $\tau_{qmem}$ which grows with system size which is temperature-dependent and can reach to a cut-off size, and present a scaling that is super-exponential to an inverse-temperature as partial self correcting codes. Such codes have energy barrier scaling with error size logarithmically. [82]

Haah [86] discovered that cubic lattice with no string operators having qubits less than three at each vertex provides partial self correction which we discuss in the next section in detail. Here we can summarize condition of having no strings as not being

able to carry the non-trivial errors through the lattice without causing new errors. In such codes, due to the required energy to create logical error is high, they yield good protection as far as thermal noise is concerned. Codes having no string property have logarithmically growing energy barriers for error length $l_e$ such that necessary energy to create such error occupying volume with radius $l_e/2$ is: [82]

$$\varepsilon_e \gtrsim c\Delta \log(l_e) \tag{3.80}$$

where $c > 0$ is some constant and $\Delta$ is the energy gap. Assuming average distance between errors grows with with $\beta$ such as $L' \sim e^{\beta\Delta/D}$ where $D$ is dimensionality and following memory undergoing decoherence demanding error size being comparable with the average distance between errors $l_e \sim L'$ gives at decoherence point: [82]

$$\varepsilon_e \sim \frac{c\Delta^2\beta}{D} \tag{3.81}$$

Using this and Arrhenius' law we get:

$$\tau \sim e^{c\Delta^2\beta/D} \tag{3.82}$$

Likewise for small system such that $L \lesssim L'$, we have:

$$\varepsilon_e \sim c\Delta \log(L) \tag{3.83}$$

putting it into Arrhenius' law we get:

$$\tau \sim L^{c\Delta\beta} \tag{3.84}$$

which is polynomial in size of the system having linear $\beta$ in the exponent. [82]

### 3.3.2 Haah cubic Code

Here we discuss Haah's cubic code in a new framework in which we reorganize the cubic structure on a carthesian coordinates giving us single equation of commutation relations for corner operators than the original work in [86] (which has 13 equations).

Consider a stabilizer code which acts locally on a cubic lattice of $D$-dimensions $\mathbb{Z}^D$ and let single qubit is placed on every site. Note that once we coarse-grain the lattice

or put 1 or more qubits on every site, then we see that for a unit hypercube with $2^D$ qubits on each site, a generator can act on qubits. Also, we only consider $X$ and $Z$ type generators (that are $2m \times 2^D$ vectors). And we want that the codes don't have string logical operators. We can think of string logical operators as logical operators with a shape of straight line and act nontrivially.[86]

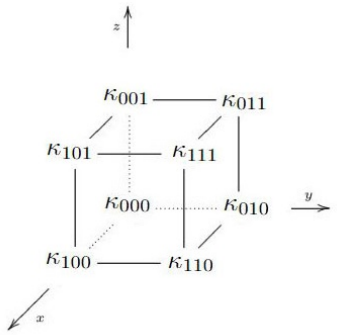



Figure 3.3: Corner operators where subscripts labels the positions on the cube

Note that non-CSS codes should satisfy $\kappa^j_{abc} = \kappa^{j\oplus 1}_{a\oplus 1 b\oplus 1 c\oplus 1}$ where $\kappa$ is the corner operator, $a, b, c \in \mathbb{F}_2$ and $j \in \{X, Z\} \ni$ if $j = X$ then $j \oplus 1 = Z$ and vice versa. Or in other words, a generator on a corner is equals to the generator at the other corner of the diagonal along the cube of different type of Pauli operator.
Let $n'$ be the qubit number on a site. Then stabilizer code acting on an open or periodic lattice locally satisfies:[86]

$$kd = \mathcal{O}(L^3) \tag{3.85}$$

If the number of generators $t < n'$, we have $k \geq (n' - t)L^3$. But to reach macroscopic code distance we need to ensure that $t \geq n'$[86]

Since we ignore boundary effects and consider only thermal stability, we take stabilizer codes defined on infinite lattice of $\mathscr{L} = \mathbb{Z}^3$ into account. And for a logical operator of single-site $O$, we require that $O \in \mathcal{S}$. Therefore we take $O$ as identity up to phase and denote it as:[86]

$$O[v]_p = \cdots \otimes O \otimes O \otimes \cdots \tag{3.86}$$

where $v$ is the line Pauli operator is repeated on and $p$ is the point it passes through.

And the support line for the nontrivial Pauli operator on a site set:[86]

$$supp(O[v]_p) = \{p + nv \in \mathbb{Z}^3 | n \in \mathbb{Z}\} \tag{3.87}$$

Note that if $\|v\|_\infty = 1$ then period of $O[v]$ is 1 and all logical operators having period of 1 is the necessary condition for no string logical operators being in the code. [86] Since those restrictions causes k=0, we want product of corner operators gives identity which means nontrivial algebraic constraint for corner operators in CSS but no further restriction for non-CSS since they already satisfy.
Notice that a stabilizer code generated by some symmetric transformation on a unit cube or by changing the basis of all sites is the same as the original one, leaving us 17 CSS and 1 non-CSS cubic codes reported by [86] as in the table 3.1. [86]

Haah also provided conditions for cubic codes as follows:[86]

i) In an infinite lattice $\mathscr{L}$ of simple cubes, there should be either 1 or 2 qubits on each site.

ii) There should be two operator types which are acting on the corners and those operators generate $\mathcal{S}$ such that $\mathcal{S}$ is invariant under translations. In the case of CSS codes, product of operators acting on corners be identity. And in non-CSS case, two operator types should be the spatial inverse of each other.

iii) Single-site $O$ should be $\mathbb{I}$ up to phase if $O \in \mathcal{S}^\perp$.

iv) $\gamma \in \mathcal{S}^\perp$ with $\gamma_\infty = 1$ (meaning that $\gamma$ has period of 1) is $\mathbb{I}$ up to phase.

Note that $\gamma_\infty = 1$ means that for example support of $l$ is along a diagonal of a face or diagonal of the body or a coordinate axis. To illustrate let's consider commutations of corner operators.[86]

Let $\omega$ be an $n \times n$ commutation relation matrix such that $\omega \in \mathbb{F}_2$ which is skew-symmetric. In [86] this matrix is used to tell commutation relations, but in this study, we provide a better form.

| | $\kappa_{000}$ | $\kappa_{001}$ | $\kappa_{010}$ | $\kappa_{011}$ | $\kappa_{100}$ | $\kappa_{101}$ | $\kappa_{110}$ | $\kappa_{111}$ |
|---|---|---|---|---|---|---|---|---|
| 0 | $\mathbb{II}$ | ZX | ZZ | XZ | ZY | XX | XY | Z$\mathbb{I}$ |
| 1 | $\mathbb{II}$ | $\mathbb{I}$Z | $\mathbb{I}$Z | Z$\mathbb{I}$ | $\mathbb{I}$Z | Z$\mathbb{I}$ | Z$\mathbb{I}$ | ZZ |
| 2 | ZZ | Z$\mathbb{I}$ | Z$\mathbb{I}$ | $\mathbb{I}$Z | Z$\mathbb{I}$ | $\mathbb{I}$Z | Z$\mathbb{I}$ | ZZ |
| 3 | $\mathbb{II}$ | $\mathbb{I}$Z | ZZ | $\mathbb{I}$Z | ZZ | $\mathbb{I}$Z | Z$\mathbb{I}$ | ZZ |
| 4 | $\mathbb{II}$ | Z$\mathbb{I}$ | $\mathbb{I}$Z | $\mathbb{I}$Z | Z$\mathbb{I}$ | $\mathbb{I}$Z | Z$\mathbb{I}$ | ZZ |
| 5 | $\mathbb{II}$ | $\mathbb{I}$Z | Z$\mathbb{I}$ | $\mathbb{I}$Z | $\mathbb{II}$ | Z$\mathbb{I}$ | ZZ | ZZ |
| 6 | ZZ | $\mathbb{I}$Z | $\mathbb{I}$Z | $\mathbb{II}$ | Z$\mathbb{I}$ | Z$\mathbb{I}$ | ZZ | $\mathbb{II}$ |
| 7 | $\mathbb{II}$ | ZZ | $\mathbb{I}$Z | $\mathbb{II}$ | Z$\mathbb{I}$ | Z$\mathbb{I}$ | $\mathbb{I}$Z | ZZ |
| 8 | $\mathbb{II}$ | Z$\mathbb{I}$ | $\mathbb{I}$Z | $\mathbb{I}$Z | $\mathbb{I}$Z | Z$\mathbb{I}$ | ZZ | Z$\mathbb{I}$ |
| 9 | ZZ | $\mathbb{I}$Z | $\mathbb{I}$Z | $\mathbb{II}$ | ZZ | Z$\mathbb{I}$ | ZZ | $\mathbb{I}$Z |
| 10 | ZZ | Z$\mathbb{I}$ | $\mathbb{I}$Z | Z$\mathbb{I}$ | Z$\mathbb{I}$ | Z$\mathbb{I}$ | ZZ | $\mathbb{I}$Z |
| 11 | $\mathbb{II}$ | ZZ | Z$\mathbb{I}$ | $\mathbb{I}$Z | $\mathbb{II}$ | Z$\mathbb{I}$ | $\mathbb{I}$Z | ZZ |
| 12 | $\mathbb{II}$ | $\mathbb{I}$Z | Z$\mathbb{I}$ | $\mathbb{II}$ | ZZ | Z$\mathbb{I}$ | ZZ | $\mathbb{I}$Z |
| 13 | $\mathbb{II}$ | ZZ | $\mathbb{I}$Z | $\mathbb{II}$ | $\mathbb{I}$Z | Z$\mathbb{I}$ | Z$\mathbb{I}$ | ZZ |
| 14 | $\mathbb{II}$ | $\mathbb{I}$Z | $\mathbb{I}$Z | ZZ | ZZ | Z$\mathbb{I}$ | ZZ | $\mathbb{I}$Z |
| 15 | ZZ | Z$\mathbb{I}$ | $\mathbb{I}$Z | $\mathbb{II}$ | $\mathbb{II}$ | Z$\mathbb{I}$ | ZZ | $\mathbb{I}$Z |
| 16 | ZZ | ZZ | $\mathbb{I}$Z | $\mathbb{II}$ | $\mathbb{II}$ | Z$\mathbb{I}$ | $\mathbb{I}$Z | Z$\mathbb{I}$ |
| 17 | Z$\mathbb{I}$ | ZZ | $\mathbb{I}$Z | Z$\mathbb{I}$ | $\mathbb{I}$Z | Z$\mathbb{I}$ | Z$\mathbb{I}$ | ZZ |

Table 3.1: Corner operators of given cubic codes. Note that codes 1,2,3 and 4 have no logical string operators while codes 0, 11, 12, 13, 14, 15, 16 and 17 have. [86]

In this approach, we can find commutation relations among corner operators along any direction by defining the position vector $\boldsymbol{\alpha}$, the direction vector $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ vector. Note that $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ satisfy followings:

1.

$$wt(\boldsymbol{\gamma}_j) = 1 \quad \forall j \neq 0 \tag{3.88}$$

where $\boldsymbol{\gamma}_0 = (000)$

2. for corner operators meeting at an $i$-cell,

$$wt\left(\sum_{j=0}^{i}\boldsymbol{\gamma}_j\right)=i \tag{3.89}$$

3.

$$\boldsymbol{\beta}\oplus\sum_{j}\boldsymbol{\gamma}_j=(111) \tag{3.90}$$

where $\boldsymbol{\gamma}_l\neq\boldsymbol{\gamma}_k\ \forall l\neq k$

Then commutation relations in 3D can be found via

$$\sum_{j=0}^{2^i-1}[\kappa_{\boldsymbol{\alpha}\oplus\mathbf{f}_j},\kappa_{\boldsymbol{\alpha}\oplus\boldsymbol{\beta}\oplus\mathbf{f}_j}]=0 \tag{3.91}$$

where $\mathbf{f}_j=(\delta_{1j}+\delta_{3j})\boldsymbol{\gamma}_1\oplus(\delta_{2j}+\delta_{3j})\boldsymbol{\gamma}_2$ with $\oplus$ in modulo 2.

Note that equations 3.88, 3.89 and 3.90 dictate in which direction commutation relations can be found. For example, for corner operators meeting at a vertex (0-cell) we have $i=0$, then equation 3.89 gives $wt\left(\sum_{j=0}^{0}\boldsymbol{\gamma}_j=0\right)$ inducing $\boldsymbol{\gamma}=(000)$. Then due to equation 3.90 we have $\boldsymbol{\beta}=(111)$ which implies that commutation occurs in the direction of body diagonals.

To illustrate how it works, let's consider corner operators meeting at a face (2-cell) for corner operator $\kappa_{100}$, along $\hat{x}$ direction. Notice position vector $\boldsymbol{\alpha}=(100)$ and direction vector $\boldsymbol{\beta}=(100)$. Then, due to equation 3.90, $\sum_j\boldsymbol{\gamma}_j=(011)$. So, we choose $\boldsymbol{\gamma}_1=(010)$ and $\boldsymbol{\gamma}_2=(001)$. As a result, equation 3.91 becomes:

$$\sum_{j=0}^{2^2-1}[\kappa_{(100)\oplus\mathbf{f}_j},\kappa_{(100)\oplus(100)\oplus\mathbf{f}_j}]=\sum_{j=0}^{3}[\kappa_{(100)\oplus\mathbf{f}_j},\kappa_{(000)\oplus\mathbf{f}_j}]=0 \tag{3.92}$$

Here $\mathbf{f}_0=(000)$, $\mathbf{f}_1=\boldsymbol{\gamma}_1=(010)$, $\mathbf{f}_2=\boldsymbol{\gamma}_2=(001)$ and $\mathbf{f}_3=\boldsymbol{\gamma}_1\oplus\boldsymbol{\gamma}_2=(011)$. At the end we get:

$$[\kappa_{100},\kappa_{000}]+[\kappa_{110},\kappa_{010}]+[\kappa_{101},\kappa_{001}]+[\kappa_{111},\kappa_{011}]=0 \tag{3.93}$$

Now for logical operators, consider $\lambda=\begin{pmatrix}0&\mathbb{I}\\\mathbb{I}&0\end{pmatrix}$, $p=\{X,Z\}$ and $O[\boldsymbol{\beta}]$ which is a logical operator with period 1 along some $\boldsymbol{\beta}$ axis. $O[\boldsymbol{\beta}]$ is a $p_a$ type logical operator

iff it satisfies:

$$\lambda(\kappa_{\boldsymbol{\alpha}\oplus\mathbf{f}_j}^{p_{a\oplus 1}}\kappa_{\boldsymbol{\alpha}\oplus\boldsymbol{\beta}\oplus\mathbf{f}_j}^{p_{a\oplus 1}}, O) = 0 \quad \forall j = 0,1,2,3 \text{ and } a = 0,1 \tag{3.94}$$

Note that if two cubic codes have the same commutation relation matrix $\omega$, then a symplectic transformation on one gives the other. Also, $X\&Z$-type generators have relation based on spatial inversion such that

$$\kappa_{aij}^{x} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \kappa_{a'ij}^{z} \quad \text{where } \; i,j,a,a' \in \mathbb{F}_2 \ni a' = a \oplus 1 \tag{3.95}$$

which implies that we can find a logical operator of $Z$-type by applying spatial inversion about origin to a logical operator of $X$-type and then symplectic transform to all sites. So, for CSS we can just look at logical $X$ operators and stabilizer generators of $Z$-type.

Haah shows how we can reduce a 3D logical operator into logical operators with less dimensions following cleaning theorem and using Eqn.3.95 [86]. But before we start, we should give the definition of being connected:

**Definition 3.3.1** *We call a set of sites connected if there exists nontrivially acting stabilizer generators for each consecutive site which goes as sequential pairs from the beginning to the end connecting some path. [86]*

To illustrate we will look at code 0 in fig.3.4. Consider a finite logical operator $\Gamma$ and some finite box $\beta$:

$$\beta = \{(x,y,z) | x_0 \leq x \leq x_1, y_0 \leq y \leq y_1, z_0 \leq z \leq z_1\} \subseteq \mathscr{L} \tag{3.96}$$

Let's assume that $\beta$ supports $\Gamma$.

Now consider the vertex $v_1 = (x_1, y_1, z_0)$ in $B$ with largest $x$ & $y$ and smallest $z$ possible. Note that it has to commute with its body diagonal opposites that are $\kappa_{001} = ZX$ in $Q_0$ & $\kappa_{001} = XY$ in $Q_0^P$ and it has to be the linear combination of them. What we want to do here is to shrink $\beta$ which is the support of $\Gamma$. We have 4 cases to consider:[86]

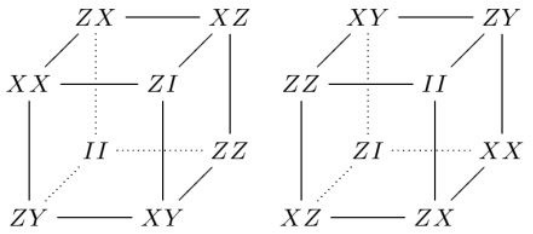



Figure 3.4: $Q_0$ (in the left) & $Q_0^P$ (in the right) [86]

1. $v_1 = \mathbb{III}$

   In this case, we can shrink the support trivially.

2. $v_1 = XY$

   To make $v_1 = \mathbb{III}$ we need to multiply $Q_0$ inside $\beta$ and then shrink the support.

3. $v_1 = ZX$

   This time, we will multiply with $Q_0^P$ inside the support giving us $v_1 = \mathbb{III}$ and then shrink $\beta$.

4. $v_1 = YZ$

   Last case requires us to multiply the support by the $Q_0 Q_0^P$ product to shrink $\beta$.

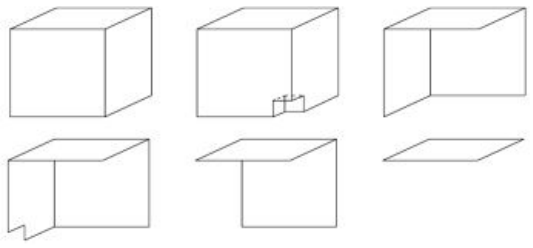


Figure 3.5: Deformation procedure in code 0, 1, 2 and 4 [86]

Note that the above procedures deforms the support in such a way that it has one less site (see 2nd in fig.3.5) and we can do it many times. Only thing we have to look for is that result from deforming the $\beta$ contains a unit cube. This ensures us that multiplying with linear combinations of $Q_0$ & $Q_0^P$ have effect only on sites in the support. So, what we end up with is 3 rectangles with a small width (see 3rd in

fig.3.5) that are:[86]

$$\begin{aligned} R_x &= \{(x_0, y, z) \mid y_0 \leq y \leq y_1, z_0 \leq z \leq z_1\}, \\ R_y &= \{(x, y_0, z) \mid x_0 \leq x \leq x_1, z_0 \leq z \leq z_1\}, \\ R_z &= \{(x, y, z_0) \mid x_0 \leq x \leq x_1, y_0 \leq y \leq y_1\} \end{aligned} \tag{3.97}$$

where subscripts of $R$'s indicates the axis that site is perpendicular to. Now consider another vertex $v_2 = (x_1, y_0, z_0)$ which is in $R_y$. Since this edge has corner operators which are sufficiently independent due to its position on $R_y$, it is required to commute with $\kappa_{001}$ and $\kappa_{011}$. Notice that $\kappa_{001} = ZX$ & $\kappa_{011} = XZ$ in $Q_0$ and $\kappa_{001} = XY$ & $\kappa_{011} = ZY$ in $Q_0^P$ giving us $v_2 = \mathbb{III}$ (see the 4th in fig.3.5). Note that if we can do the aforementioned process, we call that edge as *good for erasing*.[86] Furthermore, we can do the same for $R_x$ and $R_z$, then what we have at the end is $\mathbb{I}$ by multiplying the required stabilizer inside the support.[86]

In this topology, a logical operator with 1D support can be called as *string logical operator* which is capable of stretch with constant width arbitrarily in a lattice of infinite size.[86] We can do deformation similar to discussed before to string logical operators as well. Haah showed that codes 1,2,3 & 4 have no string logical operators and such codes have the property of self-correction therefore a promising candidate for realization of quantum memories. [86]

### 3.4 Entropically Protected Quantum Memory

In this section, we look at method offered in [87] to overcome thermal noise via optimization of $F$ in equation 3.17 and allowing better coherence time. [88] This method is useful for systems in which energy barrier is constant. Especially due to no-go theorem offered by Bravyi and Terhal [89] stating that no 2D model can support macroscopic energy barriers, therefore incapable of protecting quantum information against thermal noise. Entropic barrier might allow us to produce 2D quantum memories.[87]

To begin with, consider $N$-level spin system of $L \times L$ lattice in Kitaev's model $\mathbb{Z}_N$.[90] For Pauli operators $X, Z \ni \ X\left|j\right\rangle = \left|j + 1(mod N)\right\rangle,\ Z\left|j\right\rangle = e^{2\pi i j/N}\left|j\right\rangle$

we have Hamiltonian:[87]

$$H = \sum_v \mathbf{J} \cdot \mathbf{\Pi}_v + \sum_p \mathbf{J} \cdot \mathbf{\Pi}_p \tag{3.98}$$

where local projection operators

$$\Pi_v^a = \frac{1}{N} \sum_{j=1}^{N} e^{2\pi i j a/N} A_v^j \quad \& \quad \Pi_v^a = \frac{1}{N} \sum_{j=1}^{N} e^{2\pi i j a/N} B_p^j \tag{3.99}$$

commute locally $\forall v \& p$ where $a = 1, 2, \cdots, N-1$. Here strength of interactions $\mathbf{J} = (J_1, \cdots, J_{N-1})$ and $\mathbf{\Pi}_v = (\Pi_v^1, \cdots, \Pi_v^{N-1})$, $\mathbf{\Pi}_p = (\Pi_p^1, \cdots, \Pi_p^{N-1})$.
Note that we use $N^2$-fold degeneracy imposed by the model for logical encoding. For this method, we consider anyonic model with electric and magnetic anyons $e_j$ and $m_j$ respectively to be integers where $j = 1, \cdots, N-1$. For primary lattice, we have electric anyons while for the dual we have magnetic anyons. For type-a anyons, anyon number $n_a$ decides energy eigenvalues as:[87]

$$E = \sum_j (n_{e_j} + n_{m_j}) J_j \tag{3.100}$$

In fig.3.6(c)&(d) we see that anyons created at the ends of string operators that can be splitted when $N > 2$ (fig.3.7(d)) such that those splittings subjected to the fusion rules for $\mod N$ that is

$$e_k \times e_l = e_{k \oplus l} \quad \& \quad m_k \times m_l = m_{k \oplus l} \tag{3.101}$$

where $\oplus$ is addition in modulo $N$. Using Ohmic reservoir, spin-boson interactions of errors in Davies limit gives us thermal evolution with equation (following equation 3.3)

$$\gamma(\omega) = \frac{\omega}{1 - e^{-\beta\omega}} \tag{3.102}$$

causes the state to turn into Gibbs state. [87]
Conserving the net charge, an error $e_j$ splits into two errors $e_k$ and $e_l$ which have separation in space and we denote this process with $e_j \to e_k \times e_l$. We exploit this behaviour by introducing defect lines that maps electric charge $e_j$ into $e_{j \oplus j}$ and magnetic charge $m_j$ into $m_{j \oplus j}$ once they crossed it in negative and positive directions respectively. Note that crossing in the opposite direction reverses the mapping.[87] In fig.3.6(g)&(h) these lines changes the Hamiltonian via projectors. In primal face,

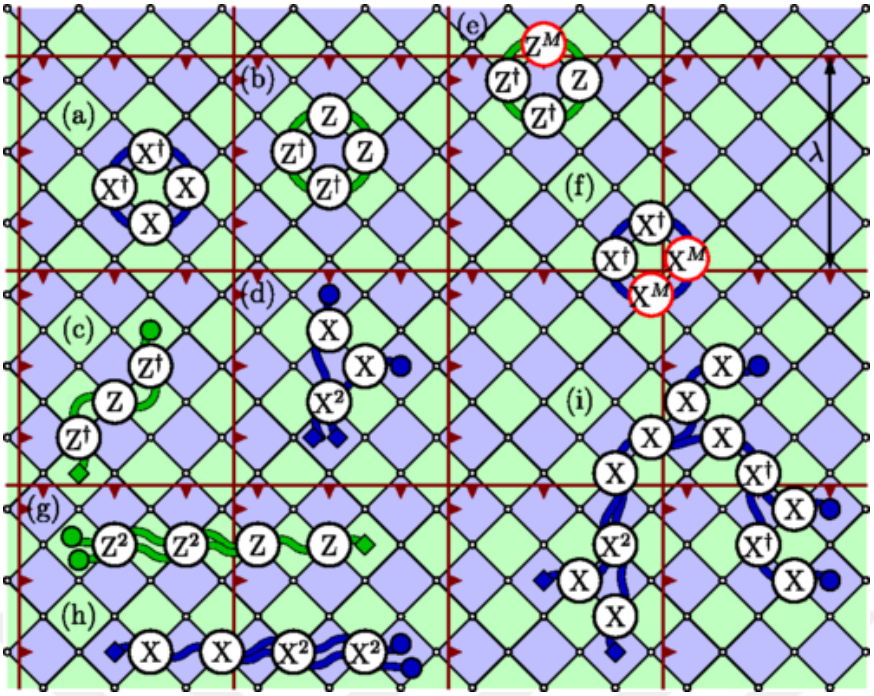



Figure 3.6: Anyon movements along 2D lattice with defect grids [87]

defect line gives $supp(A_v)$ power $M$ if it is in the framed side (see fig.3.6(f)). In dual face, defect line makes the power of $supp(B_p)$ $M$ if it is in the unframed side (see fig.3.6(e)).[87]

For instance, consider $\mathbb{Z}_5$ ($N = 5$) and particles $e_j$ having masses $M_j$:

$$2M_1 = 2M_4 \leq M_2 = M_3 \tag{3.103}$$

In this case, $e_2 \rightarrow e_1 \times e_1$ and $e_3 \rightarrow e_4 \times e_4$ will most likely undergo. For grid number g $= 2$, errors with low mass $e_1$ and $e_4$ will turn into errors with high mass $e_2$ and $e_3$ once they cross the defect lines causing energy penalty $J_H - J_L$ where subscripts in interaction strengths stand for high and low masses.[87]

As an example, let's consider an error in mid-low temperature. Suppose an error $e_1$ with mass $M_1$ is created (see fig.3.7(a)) and passed the defect line which suppressed the low-mass errors energetically. Such energy penalty decreases the chance of system to decohere. In fig.3.7(b) we see $e_1$ turning into $e_2$ after passing through defect line which is unlikely. However if the initial error was $e_2$ which has high mass, it will decay as $e_2 \rightarrow e_1 \times e_1$ (see fig.3.7(c)). Then two $e_1$ errors cross next defect line with an energy penalty causing them to recombine (see fig.3.7(d)) which we can increase

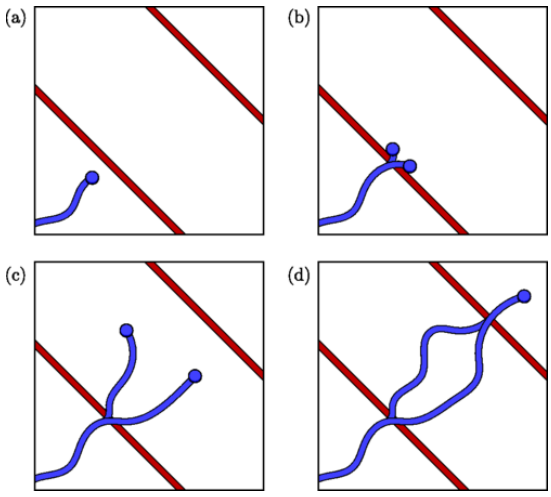



Figure 3.7: Errors passing through defect lines [82]

its probability by adjusting the distance between defect lines $\lambda$.[87]
Note that although it is not probable, a high mass error may pass through a defect line with no energy penalty. However we can make this even less likely by increasing the defect line seperation $\lambda$. Moreover, if the temperature is so low, low-mass errors most likely recombine to high-mass errors and pass through the defect line without any energy penalty. This probability increases exponentially with $\beta$ compared to low-mass passing through since recombination costs $J_H - 2J_L$ energy while low-mass crossing costs $J_H - J_L$. So, we need appropriately low-temperatures that is not that cold. Luckily, we can overcome this issue by increasing defect line seperation $\lambda$ which decreases the probability of recombination.[87]
Another important point is that defect lines can't provide divergent quantum memory coherence time for Abelian models; however non-Abelian cases might be different in which more study is required.[87]

## 3.5 Single Cavity Memory

In this method, we use single cavity mode instead of multi-qubit system to store quantum information. We exploit the fact that in a cavity mode, dominant error type is photon damping and increasing the photon number in the cavity doesn't add any channels. As a result with a single syndrome measurement, we can decide whether

there is an error due to linearity. Basically, we encode logical qubits into coherent states' superposition that has multiple components.
Denoting ground state as $|g\rangle$ and excited state as $|e\rangle$, a qubit state $|\Psi\rangle = c_g |g\rangle + c_e |e\rangle$ where $c_g$ and $c_e$ are constants that are associated to the ground and excited states respectively. We can map this state into the state [94]

$$\left|\psi_\alpha^{(0)}\right\rangle = c_g \left|\mathcal{C}_\alpha^+\right\rangle + c_e \left|\mathcal{C}_{i\alpha}^+\right\rangle \tag{3.104}$$

which is multicomponent coherent state. For later use, let's define:

$$\begin{aligned}
\left|\psi_\alpha^1\right\rangle &= c_g \left|\mathcal{C}_\alpha^-\right\rangle + ic_e \left|\mathcal{C}_{i\alpha}^-\right\rangle \\
\left|\psi_\alpha^2\right\rangle &= c_g \left|\mathcal{C}_\alpha^+\right\rangle - c_e \left|\mathcal{C}_{i\alpha}^+\right\rangle \\
\left|\psi_\alpha^3\right\rangle &= c_g \left|\mathcal{C}_\alpha^-\right\rangle - ic_e \left|\mathcal{C}_{i\alpha}^-\right\rangle
\end{aligned} \tag{3.105}$$

Here $|\mathcal{C}_\alpha^+\rangle$ and $\left|\mathcal{C}_{i\alpha}^+\right\rangle$ are:

$$\begin{aligned}
\left|\mathcal{C}_\alpha^\pm\right\rangle &= N(|\alpha\rangle \pm |-\alpha\rangle) \\
\left|\mathcal{C}_{i\alpha}^\pm\right\rangle &= N(|i\alpha\rangle \pm |-i\alpha\rangle) \\
&\forall \alpha \in \mathbb{C}
\end{aligned} \tag{3.106}$$

where normalizing factor $N \approx 1/\sqrt{2}$ and coherent state $|\alpha\rangle$ forms a quasiorthogonal set $A = \{|\alpha\rangle, |-\alpha\rangle, |i\alpha\rangle, |-i\alpha\rangle\} \ni |\langle j_k|j_l\rangle| \ll 1 \ \forall k, l = 1, \cdots, 4$ denoting the which element of $A$ is the state $|j\rangle$ is. We encode $\left|\overline{0}\right\rangle$ as $|\mathcal{C}_\alpha^+\rangle$ and $\left|\overline{1}\right\rangle$ as $\left|\mathcal{C}_{i\alpha}^+\right\rangle$ having properties:[94]

i) photon loss maps $|\psi_\alpha^n\rangle$ into

$$\frac{a\,|\psi_\alpha^n\rangle}{\|a\,|\psi_\alpha^n\rangle\|} = \left|\psi_\alpha^{(n+1) \mod 4}\right\rangle \tag{3.107}$$

where $a$ is annihilation operator

ii) $|\psi_\alpha^n\rangle$ evolves into $\left|\psi_{\alpha e^{-\kappa t/2}}^n\right\rangle$ after a time $t$ if there is no quantum jumps where $\kappa$ is the decay rate of the cavity.

iii) parity operator

$$\mathtt{P} = e^{i\pi a^\dagger a} \tag{3.108}$$

can be considered as quantum jump indicator such that

$$\langle\psi_\alpha^n|\mathtt{P}|\psi_\alpha^n\rangle = (-1)^n \tag{3.109}$$

Mathematically what this method does is:[94]

$$U_{enc}(c_g|g\rangle + c_e|e\rangle)\otimes|0\rangle = |g\rangle\otimes(c_g\left|\mathcal{C}_\alpha^+\right\rangle + c_e\left|\mathcal{C}_{i\alpha}^+\right\rangle) + \mathcal{O}(e^{-|\alpha|^2}) \quad \forall c_g, c_e \tag{3.110}$$

Notice that we have $\mathcal{O}(e^{-|\alpha|^2})$ term due to the fact that $|\mathcal{C}_\alpha^+\rangle$ and $\left|\mathcal{C}_{i\alpha}^+\right\rangle$ are not orthogonal, but disappears with photon number $|\alpha|^2$ exponentially and due to gate efficiency, we can neglect this error term.[94]

To understand suppose we observed $j$ jumps in time $t$ turning initial state $|\psi_\alpha^0\rangle$ into $\left|\psi_{\alpha e^{-\kappa t/2}}^{(j) \mod 4}\right\rangle$ via non destructive parity measurement. We can do this with various methods such as measurement based QEC [93] but that needs measurements that are both reliable and quick. Another way needs qubit reset which is applied fast with high fidelity, which we cover in this section called *autonomous QEC (AQEC)*. For this, we need to find a unitary operation for correction satisfying:[94]

$$\begin{aligned} U_{corr}(|g\rangle\otimes\left|\psi_{\alpha e^{-\kappa t/2}}^{\pm}\right\rangle) &= \frac{1}{\sqrt{2}}(|g\rangle \pm |e\rangle)\otimes\left|\mathcal{C}_\alpha^+\right\rangle \\ U_{corr}(|g\rangle\otimes\left|\psi_{i\alpha e^{-\kappa t/2}}^{\pm}\right\rangle) &= \frac{1}{\sqrt{2}}(|g\rangle \pm |e\rangle)\otimes\left|\mathcal{C}_{i\alpha}^+\right\rangle \end{aligned} \tag{3.111}$$

where we neglected $\mathcal{O}(e^{-|\alpha|^2})$. What this operator should do is to carry the entropy of the system to the ancillas by encoding $c_g|g\rangle + c_e|e\rangle$ state into $|\psi_\alpha^0\rangle$. After unitary, resetting yields the initial state by getting rid of the entropy.
Let the time interval of QEC cycle is $t_\omega$ and assume that within $t_\omega$ only one or no jumps occur. Then before correction, state is $\left|\psi_{\alpha e^{-\kappa t_\omega/2}}^{(0)}\right\rangle$ or $\left|\psi_{\alpha e^{-\kappa t_\omega/2}}^{(1)}\right\rangle$ while after correction we get $\left|\psi_\alpha^{(0)}\right\rangle$. Throughout the QEC cycle we need:[94]

- Displacement operator $\mathcal{D}_\alpha$ acting on the cavity state
- Conditional phase shift operators for cavity
  - $\mathcal{P}$ turning state in the form $|e,\alpha\rangle$ into $|e,-\alpha\rangle$
  - $\sqrt{\mathcal{P}}$ turning state in the form $|e,\alpha\rangle$ into $|e,-i\alpha\rangle$

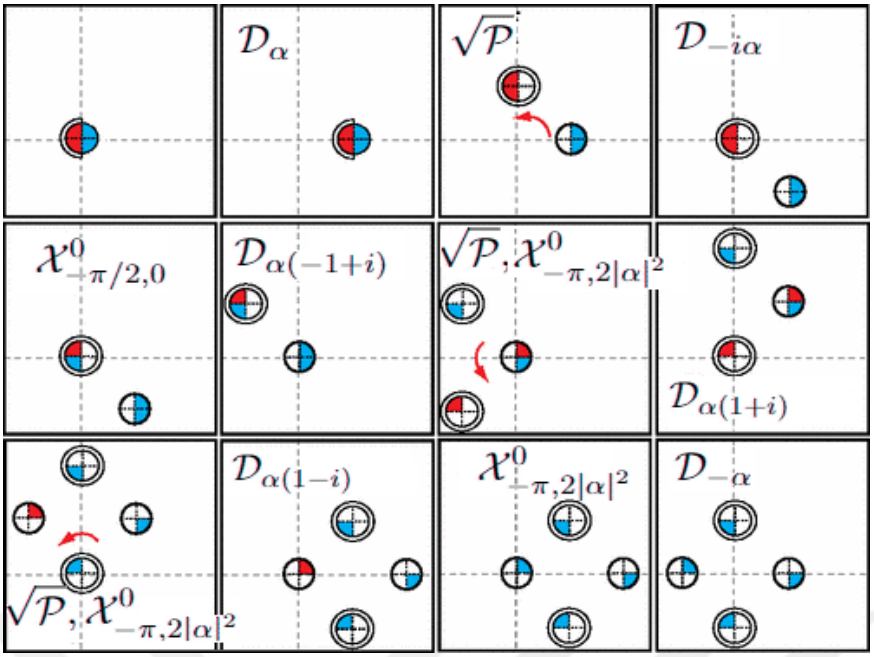



Figure 3.8: Fresnel diagram of the operators we use, which describes the cavity modes. Here, where the center of the circle stands shows the amplitude $\alpha$, single circle on the outside represents $c_g$ while double representing $c_e$, the blue color is for $|g\rangle$ and red color is for $|e\rangle$, and how much part of the circle is filled shows total weight for every component $\left|N_{c_{g,e}}\right|^2$ such as half filled meaning that $\left|N_{c_{g,e}}\right|^2 = \frac{1}{2}$ [94]

and $|g, \alpha\rangle$ is invariant under both $\mathcal{P}$ and $\sqrt{\mathcal{P}}$

- Conditional rotation operator $\mathcal{X}^0_{\theta,\eta}$:
  If the cavity is in $|0\rangle$ state, then conditional rotation operator applies a rotation by
  $\exp\{\theta(e^{i\eta}|e\rangle\langle g| - e^{-i\eta}|g\rangle\langle e|)/2\}$

Those operations are illustrated in fig.3.8 which is in sequential order for encoding. Note that $\eta = 2|\alpha|^2$ is to make up the accumulated phase. For circuits, let's denote $\beta \equiv \alpha(-1+i)$ and $\overline{n} \equiv |\alpha|^2$ In circuit form encoding and decoding is: [95]

Now for the correction, We carry entropy of the cavity to ancilla and reset the ancillary qubit. Then to make up the jump happened during $t_\omega$ repump the energy into coherent component. Finally we again encode the cavity. Consider following figures and corresponding circuits, after defining damped amplitude following $t_\omega$ $\alpha' = e^{-\kappa t_\omega/2}$,

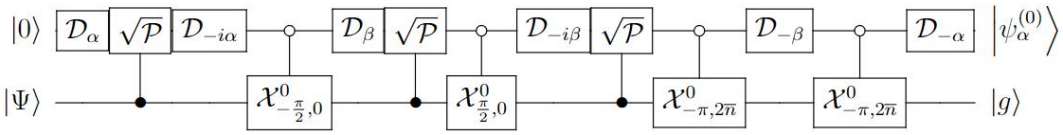


Figure 3.9: Encoding circuit [95]

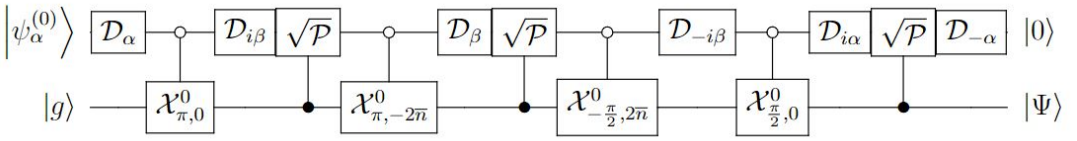


Figure 3.10: Decoding circuit [95]

giving $\overline{n}' = |\alpha'|^2$, $\beta' = \alpha'(-1+i)$ and $\beta'_d = (\beta' - \beta)/2$:[95]

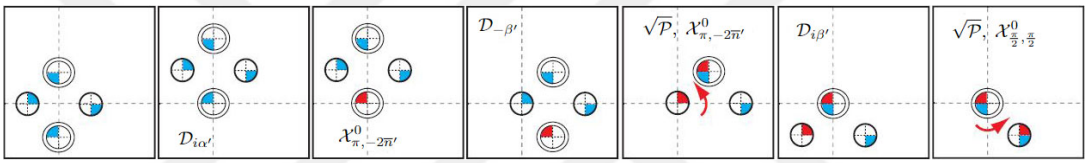


Figure 3.11: Fresnel diagram of entropy transfer step [95]

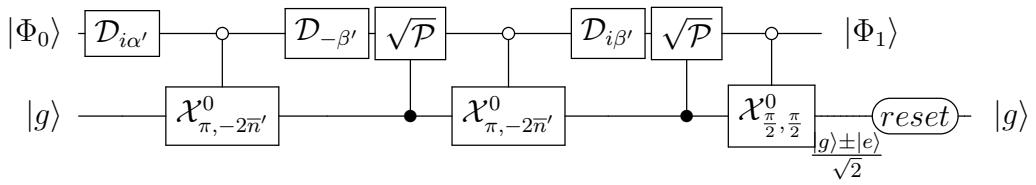


Figure 3.12: Entropy transfer circuit [95]

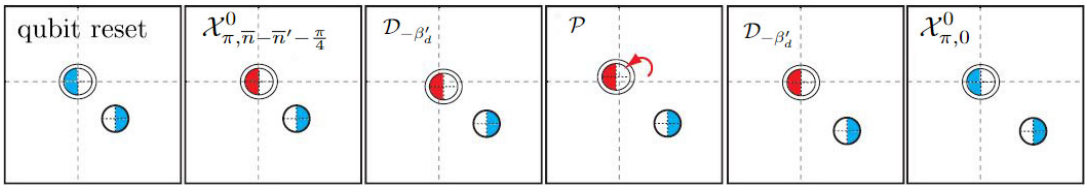


Figure 3.13: Fresnel diagram of energy repump step [95]

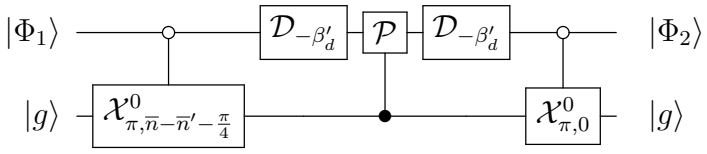



Figure 3.14: Energy repump circuit [95]

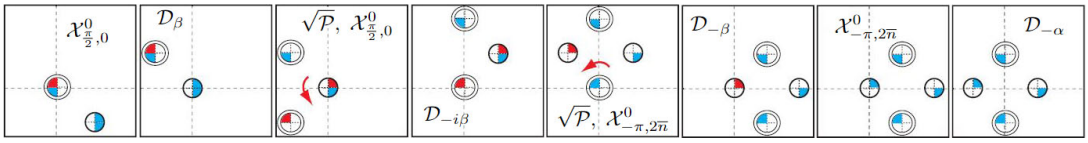



Figure 3.15: Fresnel diagram of re-encoding step [95]

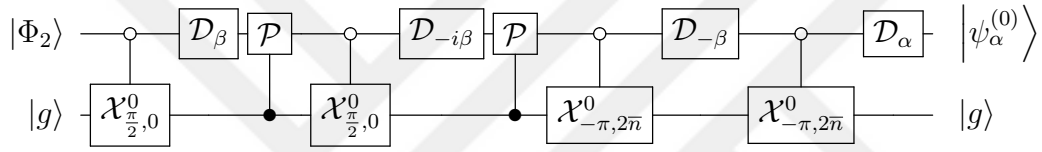



Figure 3.16: Reencoding circuit [95]

where

$$
\begin{aligned}
|\Phi_0\rangle &\equiv c_g \left|\mathcal{C}^{\pm}_{\alpha'}\right\rangle + i^{\frac{1\pm1}{2}} c_e \left|\mathcal{C}^{\pm}_{i\alpha'}\right\rangle \\
|\Phi_1\rangle &\equiv c_g \left|\alpha'(1-i)\right\rangle + c_e e^{i\pi/4} e^{i\overline{n}'} \left|0\right\rangle \\
|\Phi_2\rangle &\equiv c_g \left|\alpha(1-i)\right\rangle + c_e e^{i\overline{n}} \left|0\right\rangle
\end{aligned}
\tag{3.112}
$$

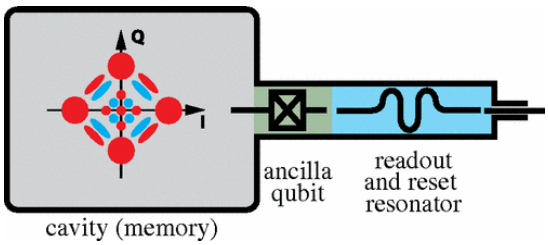



Figure 3.17: Single cavity quantum memory structure [94]

Note that in the superposition phase, error is encoded in fig.3.11 last frame and removed in the first frame of fig.3.13, but not be seen in the diagrams. Notice that the

reset is in different order than equation 3.111, this is because the experiments showed that this sequence is more efficient. [94]

## 3.6 Engineered Dissipation Quantum Memories

For this type of quantum memory, we need a codespace which contains degenerate energy levels that are robust against perturbations in the Hamiltonian. Also the system has to be robust against unwanted couplings between the system and the environment. Since in quantum memory Hamiltonians we assume weak coupling between thermal reservoir and the system, we can use thermalizing master equation for the system evolution with Born-Markov approximation. Note that in this section we follow [91]. Local couplings in general have initial states that decay into a unique Gibbs state, but for some observables, this decay rate shrinks as the subsystem number $N$ grows. This allows quantum information to be preserved in slowly decaying anticommuting many-body observable pairs.

We begin with considering logical qubit such that its physical qubits coupled with a dissipative environment and the following master equation:

$$\frac{d\rho}{dt} = \mathcal{L}_{diss}(\rho) + \mathcal{L}_{err}(\rho) \tag{3.113}$$

where $\mathcal{L}_{diss}$ is the term for engineered dissipation called Liouvillian, $\mathcal{L}_{err}$ is the error terms. We can write $\mathcal{L}_{diss}$ with Lindblad form equation 1.14 where Lindblad operators stand for dissipation or we can write it in a more suitable form:

$$\frac{d\rho}{dt} = \sum_j \Gamma_j [T_j(\rho) - \rho] \tag{3.114}$$

where $T_j$ are TPCP maps for which we can write $\rho$ with stochastic expansion:

$$\rho(t) = e^{-\Gamma t} \sum_{k=0}^{\infty} \frac{T^k \rho(0)}{k!} \tag{3.115}$$

where $T(\rho) = \sum_j \Gamma_j T_j(\rho)$ and $\Gamma = \sum_j \Gamma_j$ which might be useful for Monte Carlo simulations.

### 3.6.1 Concatenated QECC dissipation

We split QEC procedure into jumps by encoding QECC in multiple Lindblad operators. We can construct dissipative quantum memory with many-body by writing dissipation Liouvillian as superposition of recoveries (generated by concatenated QECC) of different logical qubits. Though as the number of qubits increases implementing dissipation gets harder, we can overcome this by using:

$$\mathcal{L}_{diss}(\rho) = \Gamma \sum_{l,k} \delta^{M-1}(\mathcal{R}_{l,k}(\rho) - \rho) \tag{3.116}$$

where concatenation level labeled by $l = 0, 1, \cdots, M-1$ and recovery operators $\mathcal{R}_{l,n}$ are used on $k$ qubits and $\Gamma$ is rate. We define recovery superoperator for stabilizer codes:

$$\mathcal{R}(\rho) = \sum_{(m)} R^{(m)} \Pi^{(m)} \rho \Pi^{(m)\dagger} R^{(m)\dagger} \tag{3.117}$$

where $\Pi^{(m)}$ are projection operators onto orthogonal subspaces of syndromes satisfying $\sum_{(m)} \Pi^{(m)} = \mathbb{I}$, $R^{(m)}$ are Pauli tensor products giving recovery operators that are unitary and $L^{(m)} = R^{(m)}\Pi^{(m)}$ are Lindblad operators for $M$-level encoding, $\mathbf{v} \in \mathbb{Z}_n^M$ is $n^l$ physical qubit set and $|\mathbf{v}|$ component number of $\mathbf{v}$. Then the master equation becomes:

$$\frac{d\rho}{dt} = \sum_{|\mathbf{v}|=M} \mathcal{D}_{err,\mathbf{v}}(\rho) + \sum_{|\mathbf{v}|<M} \mathcal{D}_{corr,\mathbf{v}}(\rho) \tag{3.118}$$

where $\mathcal{D}_{err,\mathbf{v}}$ superoperator for one-qubit $\mathbf{v}$ are the error terms such that $\|\mathcal{D}_{err,\mathbf{v}}\| \leq \Gamma_{err}$ & $\mathcal{D}_{corr,\mathbf{v}}$ is protective dissipation superoperator that is

$$\mathcal{D}_{corr,\mathbf{v}}(\rho) = \Gamma_{corr,\mathbf{v}}[\mathcal{R}_{\mathbf{v}}(\rho) - \rho] = \sum_j (L_{\mathbf{v}}^{(j)} \rho L_{\mathbf{v}}^{(j)\dagger} - \frac{1}{2}\{L_{\mathbf{v}}^{(j)\dagger} L_{\mathbf{v}}^{(j)}, \rho\}) \tag{3.119}$$

where $L_{\mathbf{v}}^{(j)} = \sqrt{\Gamma_{corr,\mathbf{v}}} R_{\mathbf{v}}^{(j)} \Pi_{\mathbf{v}}^{(j)}$.

In [91] it is shown that if $\Gamma_\epsilon$ (local noise rate) is small enough, compared to the used qubit number, the rate of information that is encoded is lost with a rate exponentially small such that

$$\Gamma_\epsilon < \Gamma_\epsilon^* = \frac{\delta^2 \Gamma}{n^2} \tag{3.120}$$

where $n$ is the physical qubits in the concatenated code and $\delta$ is the strength of many-body terms which is inversely proportional to the body number (i.e. $\delta = 1/5$ for

5-qubit code). For error rate less than error threshold, relaxation rate:

$$\frac{1}{\tau} \leq \Gamma_\epsilon \delta^M \left( \frac{\Gamma_\epsilon}{\Gamma_\epsilon^*} \right)^{2^M - 1} \tag{3.121}$$

Note that since dissipative master equation has recovery operators that are nonlocal need that different qubits in different positions on the lattice couple with the same reservoir, the number of them scales polynomially to this construction to work. For practical purposes, we need a design with spatially local dissipation terms.

### 3.6.2 An Example on 2D Lattice

Now let's discuss a toy model in which only local dephasing error occurs. To begin with the local dephasing:

$$\mathcal{L}_{dep}(\rho) = \Gamma_\epsilon^z \sum_{i=1}^{N} Z_i(\rho) Z_i - \rho \tag{3.122}$$

Since we're dealing with only one type of error, we can use classical memory notion for quantum case beginning with logical observables:

$$\overline{Z} \equiv \bigotimes_j Z_j \quad \& \quad \overline{X} \equiv \theta \left( \sum_j X_j \right) \tag{3.123}$$

where $\theta$ is heavyside step function. Since $[\overline{Z}, \mathcal{L}_{dep}] = 0$, $\overline{Z}$ is protected against dephasing while $\overline{X}$ is effected due to the $\rho$ term that gives $\pm 1$ eigenvalue for $\sum_j X_j$ (for instance states with very small magnetization along the $X$-axis). Dissipation ensures the preservation of $\overline{X}$ via maintaining bigger part of $\rho$ within some subspace having high concentration of X magnetization. For majority voting of nearest neighbour master equation:

$$\mathcal{L}_{NN}(\rho) = \Gamma \sum_{\langle s,r,t \rangle} L_{s,r,t} \rho L_{s,r,t}^\dagger - \frac{1}{2} \{ L_{s,r,t}^\dagger L_{s,r,t}, \rho \} \tag{3.124}$$

where $s$ covers all sites and have two different nearest neighbours $r$ and $t$ we have the Lindblad operators:

$$L_{s,r,t} \equiv Z_s \frac{1 - X_s \otimes X_r}{2} \frac{1 - X_s \otimes X_t}{2} \tag{3.125}$$

Note that Lindblad operators are wanted to be commuting with $\overline{Z}$ and nothing altered but $\overline{X}$ within some part of $\rho$ having the weakest magnetization along $X$-direction. In classical case, we can look at $\overline{X}$ stability via magnetization metastability. In an $L \times L$ lattice with PBCs, by fixing $r$ and $t$ as north and east neighbours, we end up with Toom's rule that yields exponentially protected timing while PBC is not experimentally realistic. In [91] on the other hand, numerical simulation based on an $L \times L$ square lattice in 2D without PBCs carried where $r$ and $t$ are not fixed in a position near $s$. What they found is that the effect of $\mathcal{L}_{NN}$ is on relaxation time of $\overline{X}$ is improvement and decline on $\overline{Z}$ relaxation time with $1/n$. (See fig.3.18)

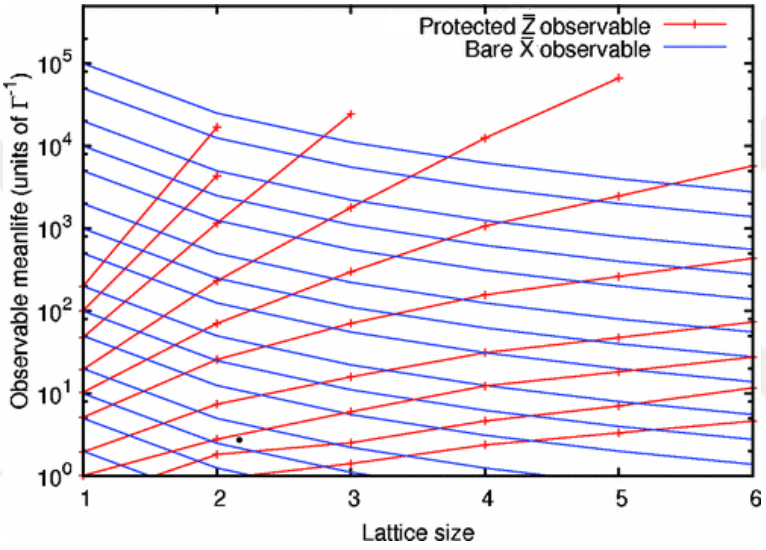



Figure 3.18: Relaxation times of logical operators with $\mathcal{L}_{NN}$ [91]

### 3.6.3 Dissipative Gadgets

To gain full control as far as dissipation is concerned, we need Lindblad jump operators that are independent and have weak influence on each other. To achieve such gadgets, we need many-body Hamiltonians generated via gadgets of perturbation theory and spontaneous dissipation specifically qubits having decoherence. *Independent rates of variation approximation* [100] is applied on the qubits that experienced decoherence in which we assume that the coupling between the system and the reservoir is faster than any other couplings. We can consider such damped qubits as resource for quantum dynamics similar to intialized qubits being resource for quantum circuit.

[91]

We can use system-ancillary qubit coupling such as:

$$H = \omega(L \otimes \sigma^+ + L^\dagger \otimes \sigma^-) \tag{3.126}$$

where $\sigma^- = |0\rangle\langle 1|$ and $\sigma^+ = |1\rangle\langle 0|$ are lowering and raising operators respectively, and ancilla having $\gamma$ damping rate couple yields a system with effective dissipation dynamics whose Lindblad operator is $\omega\sqrt{2/\gamma}L$.

## 3.7 Photonic Ising Model

For this section, we follow [101] in which we assumed that system is always initialized with $\rho_{in} = |\psi\rangle\langle\psi|$ where $|\psi\rangle \in \rho$ such that $\rho$ is spanned by logical states $\left|\overline{0}\right\rangle, \left|\overline{1}\right\rangle \in \mathcal{H}$ Hilbert space.
Consider a Liouvillian

$$\mathcal{L} = \mathcal{L}_e + \mathcal{L}_r \tag{3.127}$$

where error kicks out of the codespace and recovery generator takes those that are kicked from the codespace back to the codespace. Note that any state within the codespace is steady state of recovery generator such that $\mathcal{L}_r(\rho_i) = 0$. If this error lasts for time $t$, initial state evolves such that:

$$\rho_i \to \rho_m(t) = \rho^{\mathcal{L}t}(\rho_i) \tag{3.128}$$

Then we apply single-shot decoding channel $\mathcal{E}_r$ giving us the final state such that

$$\rho_f(t) = \mathcal{E}e^{\mathcal{L}t}(\rho_i) \tag{3.129}$$

To have self-correcting system, what we want from this system is that the initial and final states have difference exponentially small such as:

$$1 - \mathrm{Tr}(\rho_i\rho_f(t)) = \mathcal{O}(e^{-\gamma M}) \ni M \to \infty \tag{3.130}$$

where $\gamma > 0$ is some time-independent constant and $M$ is some system parameter. If equation 3.130 is satisfied, then for finite time $t$, quantum memory having system specified by $\mathcal{L}$ is protected.
In [102] a relation between $\mathbb{Z}_2$ symmetry breaking and QEC. It turns out that quantum

information is protected by a phase with broken symmetry in the case of either bit or phase flips. We can think of the outcome which is a classical bit of information that is robust against errors, as some qubit with biased error channel.
In [101], what Lieu et.al. try to achieve is to generate a protected qubit using two classical bits which are glued together. The system will be protected in the case of bit-flips by some Ising-like dissipator (in which qubits align locally) passively. And in the case of phase-flips, photonic cat code is responsible for passive protection of the system via driven-dissipative stabilization.

### 3.7.1 Photonic Cat Code

Suppose we have a photonic cavity operating with dissipation. The system has drive and loss of two photons as in:

$$H = \lambda(a^2 + (a^\dagger)^2) \tag{3.131}$$

which is the Hamiltonian of rotating frame with drive strength $\lambda$ where $a$ is the annihilation operator. We can define dissipators as $L_2 = \sqrt{\kappa_2}a^2$ two photon loss dissipator with $\kappa_2$ being two-photon loss rate, and $L_1 = \sqrt{\kappa_1}a$ single photon loss dissipator with $\kappa_1$ being single-photon loss rate. Note that parity operator in equation 3.108 causes $\mathbb{Z}_2$ symmetry due to $[H, \mathtt{P}] = [L_2, \mathtt{P}] = 0$ but this is violated by steady states defined as:

$$\rho_{ss} = |\psi\rangle\langle\psi| \quad \text{where } |\psi\rangle = c_0 |\alpha_e\rangle + c_1 |\alpha_o\rangle \ \ni |c_0|^2 + |c_1|^2 = 1 \tag{3.132}$$

Here $|\alpha\rangle$ denotes the coherent states with subscripts even or odd such that $|\alpha_e\rangle \sim |\alpha\rangle + |-\alpha\rangle$, $|\alpha_o\rangle \sim |\alpha\rangle - |-\alpha\rangle$. We have $\alpha = \sqrt{N}e^{-i\pi/4} \ \ni N \equiv \lambda/\kappa_2$ photons. Note that we define logical states as: $\left|\overline{0}\right\rangle \equiv |\alpha_e\rangle$ and $\left|\overline{1}\right\rangle \equiv |\alpha_o\rangle$.

When a photon damping $L_d = \sqrt{\kappa_d}a^\dagger a$ creates a phase-flip with error rate $e^{-\gamma N}$ (with constant $\gamma$), this error is ineffective on cat codes. When $N$ is large, states with broken symmetry ($|\pm\alpha\rangle \approx (|\alpha_e\rangle \pm |\alpha_o\rangle)/\sqrt{2}$) have lower risk of phase-flip such that they have lifetime that is exponentially long. In cat qubits, decoherence is occurs mostly due to single-photon loss $L_1$ induced bit-flip error turning steady state into a classical

bit such as:

$$\rho_{ss} \approx c\,|\alpha\rangle\langle\alpha| + (1-c)\,|-\alpha\rangle\langle-\alpha| \tag{3.133}$$

where $c \in [0,1]$. We can generalize whether if there is a passive protection by looking at the relation between perturbation and parity such that:

- if they commute (i.e. $[L_d, \mathtt{P}] = 0$), there is a passive protection
- if they don't commute (i.e. $\{L_1, \mathtt{P}\} = 0$) there is no passive protection.

### 3.7.2 2D Ising-Model

Suppose we have PBCs defined on $M \times M$ square lattice with 2D Ising model having Hamiltonian:

$$H_{Is} = -\sum_{x,y=1}^{M} (Z_{x,y}Z_{x+1,y} + Z_{x,y}Z_{x,y+1}) \tag{3.134}$$

where $Z_{x,y}$ is Pauli-$Z$ on site $(x,y)$. In this model, ground states are the ferromagnetic states with logical state $\left|\overline{0}\right\rangle \equiv |\downarrow\downarrow\cdots\downarrow\rangle$ and $\left|\overline{1}\right\rangle \equiv |\uparrow\uparrow\cdots\uparrow\rangle$. Note that $Z\,|\downarrow\rangle = |\downarrow\rangle$ and $Z\,|\uparrow\rangle = -\,|\uparrow\rangle$.

Now let H=0 by which thermalizing local dissipators can be described and think of dissipators which is constructed by taking the product of Pauli-$X$ and projection operators onto some domain-wall configuration. By doing so, we locally get a spin to shift its sign according to majority rule. For thermal state taken from the classical Ising in 2D as steady state

$$\rho_{ss} = \frac{e^{-\beta H_{Is}}}{\mathrm{Tr}(e^{-\beta H_{Is}})} \tag{3.135}$$

where $\beta = \frac{1}{8}\ln\left(\frac{\kappa+\Delta}{\Delta}\right)$ we have the dissipators (whose superscripts indicates how many domain walls projectiors check):

$$\begin{aligned} L^{(3)}_{x,y} &= \sqrt{\tilde{\kappa}} X_{x,y} \Pi^{+}_{x,y;\rightarrow} \Pi^{-}_{x,y;\uparrow} \Pi^{-}_{x-1,y;\rightarrow} \Pi^{-}_{x,y-1;\uparrow} \\ L^{(4)}_{x,y} &= \sqrt{\kappa} X_{x,y} \Pi^{+}_{x,y;\rightarrow} \Pi^{-}_{x,y;\uparrow} \Pi^{-}_{x-1,y;\rightarrow} \Pi^{-}_{x,y-1;\uparrow} \end{aligned} \tag{3.136}$$

where $\tilde{\kappa} = \sqrt{\Delta\kappa + \Delta^2} - \Delta$ and projection operators that project onto specific local spin arrangement are $\Pi^{\pm}_{x,y;\rightarrow} = (1 \pm Z_{x,y}Z_{x+1,y})/2$ and $\Pi^{\pm}_{x,y;\uparrow} = (1 \pm Z_{x,y}Z_{x,y+1})/2$. Note that we omitted rotational invariance jumps. And for all lattice sites, bit-flip rate

that is uniform is:

$$L'_{x,y} = \sqrt{\Delta} X_{x,y} \tag{3.137}$$

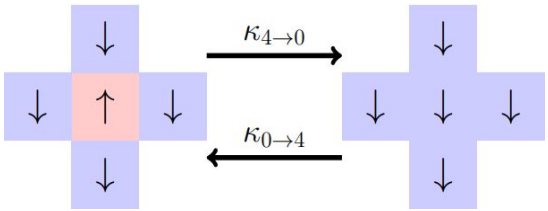



Figure 3.19: Classical spin configurations and related transition rates [101]

In fig.3.19 we see the transitioning from the situation with 4 domain wall (configuration on the left) to 0 domain wall case with transition rates which obey the detailed balance (equation 3.4) such that $(\kappa_{4\to0})/(\kappa_{0\to4}) = e^{8\beta}$

Note that we used equation 3.135 but this is not the only way. Parity operator always commute with any given dissipator. In other words, state parity is conserved in dynamical systems due to strong $\mathbb{Z}_2$ symmetry.[103]

Near the thermodynamic limit of phase with broken symmetry (or in low temperatures), we can keep the qubit in $\rho_{ss}$. However strong symmetry of $\mathbb{Z}_2$, if there is a phase-flip error ($L_i \sim Z_i$), turns into weak symmetry (such that $[\mathcal{L}, \mathtt{P}] = 0$, $\mathtt{P}(\rho) = \mathtt{P}\rho\mathtt{P}^\dagger$ ) meaning that $\rho_{ss}$ can store just a bit of information which is classic. This steady state is:

$$\rho_{ss} \approx c\left|\overline{0}\right\rangle\!\left\langle\overline{0}\right| + (1-c)\left|\overline{1}\right\rangle\!\left\langle\overline{1}\right| \quad \forall c \in [0,1] \tag{3.138}$$

If we make an analogy with cat qubits, when there is a single-photon loss, ferromagnetic states in the Ising model decohere due to phase-flip.

### 3.7.3 2D Photonic-Ising Model

Now we combine those two models by which protect the information with one model against an error that other model can't protect against. Assume we constructed an $M \times M$ lattice using photonic cavities with a two-photon drive and loss processes in each reading Hamiltonian:

$$H_{x,y} = \lambda(a_{x,y}^2 + (a_{x,y}^\dagger)^2) \tag{3.139}$$

with dissipator $L_{2,x,y} = \sqrt{\kappa_2} a_{x,y}^2$ where $a_{x,y}$ is the annihilation operator acting on $(x, y)$ site. Next, neighbouring cavities have parity-parity interaction with:

$$H_s = -\sum_{i,j} \mathtt{P}_i \mathtt{P}_j \tag{3.140}$$

where $\mathtt{P}_i$ is acting on site $i$. Just like in the Ising model, these kinds of interactions have tendency to align neighbouring parities when the temperatures are low. Local dissipators of causing such situations:

$$\begin{aligned} L_{x,y}^{(3)} &= \sqrt{\tilde{\kappa}_{nn}} a_{x,y} \Pi_{x,y;\rightarrow}^{+} \Pi_{x,y;\uparrow}^{-} \Pi_{x-1,y;\rightarrow}^{-} \Pi_{x,y-1;\uparrow}^{-} \\ L_{x,y}^{(4)} &= \sqrt{\kappa_{nn}} a_{x,y} \Pi_{x,y;\rightarrow}^{-} \Pi_{x,y;\uparrow}^{-} \Pi_{x-1,y;\rightarrow}^{-} \Pi_{x,y-1;\uparrow}^{-} \end{aligned} \tag{3.141}$$

where $\tilde{\kappa}_{nn} = \sqrt{\kappa_1 \kappa_{nn} + \kappa_1^2} - \kappa_1$ while projectors are $\Pi_{x,y;\rightarrow}^{\pm} = (1 \pm \mathtt{P}_{x,y} \mathtt{P}_{x+1,y})/2$ and $\Pi_{x,y;\uparrow}^{\pm} = (1 \pm \mathtt{P}_{x,y} \mathtt{P}_{x,y+1})/2$. On $(x, y)$ site, parity operator in equation 3.108 turns into $\mathtt{P}_{x,y} = e^{i\pi a_{x,y}^{\dagger} a_{x,y}}$. When there is no error ($\kappa_1 = 0$), steady states are:

$$|\psi\rangle = c_0 |\alpha_e\rangle |\alpha_e\rangle \cdots |\alpha_e\rangle + c_1 |\alpha_o\rangle |\alpha_o\rangle \cdots |\alpha_o\rangle \quad \text{with } |c_0|^2 + |c_1|^2 = 1 \tag{3.142}$$

As we discussed in previous sections of this chapter, to a logical error to happen, it has to pass a macroscopic energy barrier. In this method, a local single-photon loss has to overcome the large number of domain walls for a logical bit-flip. Note that this probability decays exponentially when the size of lattice growing. On the other hand for a logical phase-flip to occur, state $|\alpha_e\rangle \pm |\alpha_o\rangle$ has to turn into $|\alpha_e\rangle \mp |\alpha_o\rangle$ with $L_d$, but this also has low probability due to the fact that in the phase space, well seperated states $|\alpha\rangle$ and $|-\alpha\rangle$ have also an unstable fixed point in between. [104] Therefore for large $N$ logical phase-flip probability also decays exponentially.

### 3.7.4 Implementation

We can try to get something similar to equation 3.141 with digital automation called *Toom's rule*. This technique has no need for measurements but needs logical gate series that are fault-tolerant. We need digital step number to average photon number ratio to be linear in scale so that we can apply this method. We can consider only a single time step and the implementation of the local decoder at that step is enough, we don't need to consider whole steps.

Let's begin with placing each cavity on a lattice, an ancillary cavity initialized to $|\alpha_o\rangle$. First thing we do is applying a unitary gate $U$ to encode on some site. Then ancillary cavity changes its state according to the neighbouring cavities and the cavity that we applied $U$ on as:

$$U = \Pi \otimes (|\alpha_e\rangle \langle\alpha_o| + |\alpha_o\rangle \langle\alpha_e|) + \Pi^{\perp} \otimes (|\alpha_o\rangle\langle\alpha_o| + |\alpha_e\rangle\langle\alpha_e|) \tag{3.143}$$

where $\Pi$ projects onto the domain walls' local configuration and $\Pi^{\perp}$ is orthogonal subspace projection operator. In short what $U$ do is:

- Shift ancillary cavity state $|\alpha_o\rangle$ to $|\alpha_e\rangle$ if an error is detected
- Do nothing if no errors are detected.

Secondly, we apply $CNOT$ gate to both where control is the ancillary cavity and target is the lattice cavity of interest.
Thirdly, in order to remove the entropy from the ancillary cavity, we use a transmon. We couple this transmon and the ancillary cavity via strong dispersive coupling, then reset the ancilla back to initial $|\alpha_o\rangle$ state. When the chosen autonomous decoder is Ising-type, phase flips are suppressed due to fault-tolerance of cavity-cavity gates.
Notice that $U$ is local at every site and in the neighbourhood of every lattice cavity, and $U$'s at different sites commute. Then we can generalize the operation on a single site to the whole lattice, meaning that we can apply $U$ to all sites before the reset.

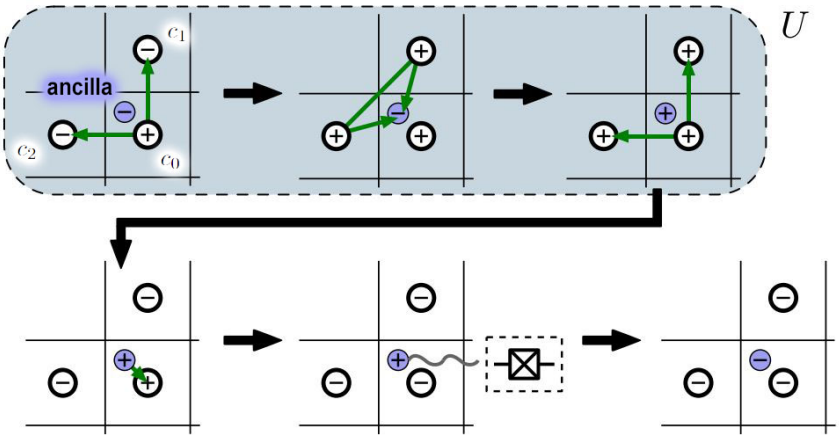



Figure 3.20: Toom's rule for photonic-ising model [101]

As an example consider the figure 3.20. Here the cavity in the middle is ancilla. At first, we applied $CNOT$ gates with $c_0$ being control and $c_1$ & $c_2$ being targets. Then we applied Toffoli gate with $c_1$ & $c_2$ being control while ancillary cavity as target. We then apply $CNOT$s again with control $c_0$ and $c_1$ & $c_2$ targets to invert followed by ancillary cavity being control and $c_0$ being target. Finally couple the ancilla with transmon to reset. So, we achieved fault-tolerance.

# CHAPTER 4

# MACHINE LEARNING ASSISTED QEC

In this chapter, we look for some methods that uses machine learning to optimize QECCs. The methods that we look at are related to the surface codes and quantum memory codes that we discussed before. Some details of the methods are provided in the Appendix to keep discussion organized.

## 4.1 Variational QEC

This method is developed for *noisy intermediate scale quantum (NISQ)* devices which are noisy, therefore require shallow circuits. In this method, cost function is created following Knill-Laflamme conditions. This method provides basis state for *variational quantum circuits* (VQC) with given noise channel and graph connectivity and it can find code for any error model. Note that we follow [105] for this method.

### 4.1.1 Cost Function Definition

Machine learning algorithms use cost functions to optimize what is necessary for algorithm to learn. For this end, let $\mathcal{E} = \{E_i\}$ be some error set. For a quantum code having parametrized orthogonal basis state, set $\{|\psi_1(\theta)\rangle, |\psi_2(\theta)\rangle, \cdots, |\psi_K(\theta)\rangle\}$ we can define cost functions:

- with $l_1$-norm:

$$\mathcal{C}^{l_1}_{n,K,\mathcal{E}}(\boldsymbol{\theta}) \equiv \sum_{E_i \in \mathcal{E}} \left( \sum_{1 \leq p \leq q \leq K} |\langle \psi_p | E_i | \psi_q \rangle| + \frac{1}{2} \sum_{q=1}^{K} \left| \langle \psi_q | E_i | \psi_q \rangle - \overline{\langle E_i \rangle} \right| \right) \tag{4.1}$$

- with $l_2$-norm:

$$\mathcal{C}^{l_2}_{n,K,\mathcal{E}}(\boldsymbol{\theta}) \equiv \sum_{E_i \in \mathcal{E}} \left( \sum_{1 \leq p \leq q \leq K} |\langle \psi_p | E_i | \psi_q \rangle|^2 + \frac{1}{4} \sum_{q=1}^{K} \left| \langle \psi_q | E_i | \psi_q \rangle - \overline{\langle E_i \rangle} \right|^2 \right) \tag{4.2}$$

where $\overline{\langle E_i \rangle} = \frac{1}{K} \sum_{q=1}^{K} \langle \psi_q | E_i | \psi_q \rangle$

Note that the zero points of those cost functions are the same and they are positive. If $\mathcal{C}^{l_1}_{n,K,\mathcal{E}} \leq 1$ then $\mathcal{C}^{l_2}_{n,K,\mathcal{E}} \leq (\mathcal{C}^{l_1}_{n,K,\mathcal{E}})^2$ and if $\mathcal{C}^{l_1}_{n,K,\mathcal{E}} = 0$ then we can detect $\mathcal{E}$ perfectly.

### 4.1.2 Algorithm

In this method, we take output states to be the QECC basis states and the circuit to be the encoder.

If there is a NISQ device having connectivity graph $\mathcal{G}$ of the hardware where vertices and edges corresponding to qubits and neighbouring qubit couples respectively single-qubit operators and two-qubit operators can be applied on qubits and neighbouring qubits respectively. With learning what we try to find is encoding circuit with lowest possible depth and a QECC of $K$-dimensions capable of detecting $\mathcal{E} = \{E_i\}$ error set.

For algorithm design, let $L$ and $L_{max}$ indicate VQC layer number and maximum allowed layer number respectively. VQC evolves with $U(\boldsymbol{\theta})$ and $\boldsymbol{\theta}$ is the circuit parameter to be updated. Beginning with $L = 1$ and random $\boldsymbol{\theta}$ value and choosing physical qubits $k = \lceil (\log_2(K)) \rceil$[1] used to logical data preparation. We want those chosen ones to scatter so that the edges connecting other qubits to chosen qubits are very few.

The algorithm that we see in fig.**??** is as follows: we begin with initializing the chosen qubits to one of the member of the set $\{|\mathbf{0}\rangle, |\mathbf{1}\rangle, \cdots, |\mathbf{K-1}\rangle\}$ which has $K$ number of binary strings, and rest to $|0\rangle^{\otimes(n-k)}$. Then input codespace $C_{in}$ which has distance $d = 1$ is spanned by the resulting states such that

$$C_{in} = span\{|\mathbf{0}\rangle |0\rangle^{\otimes(n-k)}, \cdots, |\mathbf{K-1}\rangle |0\rangle^{\otimes(n-k)}\} \tag{4.3}$$

[1] $\lceil \bullet \rceil$ is ceiling function that rounds its input into integer bigger than the input.

To estimate cost functions, we estimate $\langle\psi_q|E_i|\psi_q\rangle$ & $|\langle\psi_p|E_i|\psi_q\rangle|$ and then do measurements in computational basis. For $\langle\psi_q|E_i|\psi_q\rangle$, initial state $|\mathbf{q}-\mathbf{1}\rangle|0\rangle^{\otimes(n-k)}$ is prepared, system is evolved with $U(\boldsymbol{\theta})$ and local observable $E_i$ is measured. For $\langle\psi_p|E_i|\psi_q\rangle$, again state $|\mathbf{q}-\mathbf{1}\rangle|0\rangle^{\otimes(n-k)}$ is initiazed, system is evolved using $U(\boldsymbol{\theta})$, $E_i$ and $U^\dagger(\boldsymbol{\theta})$ in this order and measurement is done. As far as measurement is concerned, $n-k$ ancillary qubits are measured and then the rest is measured if the first measurement outcome is $|0\rangle^{\otimes(n-k)}$. Let $\mathcal{P}_{pq}$ be the binary string $|\mathbf{p}-\mathbf{1}\rangle|0\rangle^{\otimes(n-k)}$ outcome probability, then $|\langle\psi_p|E_i|\psi_q\rangle| = \sqrt{\mathcal{P}_{pq}}$. We can also get $\langle\psi_q|E_i|\psi_q\rangle$ with this step but due to the fact that VarQEC is for NISQ devices, it would be wise to use shallow circuits to estimate cost functions.

Note that though $\mathcal{E}$ can contain non-Hermitian and non-unitary terms, we assumed that it only have Pauli errors, but this can be solved by rotating Pauli terms or with additional ancillas.

### 4.1.3 $\theta$ Optimization

At first step, previously sampled $\boldsymbol{\theta}$ is used in mini-batch gradient descent for $\mathcal{C}^{l_2}_{n,K,\mathcal{E}}$ minimizing. During an iteration, subset $\mathcal{E}_s \subset \mathcal{E}$ is sampled, and we make estimation of the associated part of $l_2$-norm cost function and the gradient which are:

$$\mathcal{C}^{l_2}_{n,K,\mathcal{E}}(\boldsymbol{\theta}) \equiv \sum_{E_i\in\mathcal{E}_s}\left(\sum_{1\le p\le q\le K}|\langle\psi_p|E_i|\psi_q\rangle|^2 + \frac{1}{4}\sum_{q=1}^{K}\left|\langle\psi_q|E_i|\psi_q\rangle - \overline{\langle E_i\rangle}\right|^2\right) \tag{4.4}$$

and

$$\nabla\boldsymbol{\theta} = \frac{\partial\mathcal{C}^{l_2}_{n,K,\mathcal{E}}(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}} \tag{4.5}$$

Following this, we carry one step of gradient descent having $\eta_l$ learning rate with measurement:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_l\nabla\boldsymbol{\theta} \tag{4.6}$$

We can estimate $\mathcal{C}^{l_2}_{n,K,\mathcal{E}}(\boldsymbol{\theta})$ with $\mathcal{O}(\frac{K^2|\mathcal{E}|^2}{\epsilon^2})$ measurements up to some estimation error $\epsilon$ and gradient with finite-difference method or a combination of chain and parameter shift rules (See appendix). Using mini-batch gradient descent is advantageous because it converges more vigorously while preventing being stuck in local minima. Before converging, we keep sampling and gradient descent. Note that we begin minimizing with $\mathcal{C}^{l_2}_{n,K,\mathcal{E}}(\boldsymbol{\theta})$ because it is differentiable while $\mathcal{C}^{l_1}_{n,K,\mathcal{E}}(\boldsymbol{\theta})$ is not and $\mathcal{C}^{l_1}_{n,K,\mathcal{E}}(\boldsymbol{\theta})$

is much slower in converging.
In the case that $\mathcal{E}$ is an error set that is too large, we can use classical shadow technique on all $|\psi_q\rangle$ basis states (see appendix). Classical shadows allow us to make cost function estimation classically and optimization of a large-batch requiring less measurements.

We can start fine-tuning $\boldsymbol{\theta}$ and estimating $\mathcal{C}^{l_1}_{n,K,\mathcal{E}}(\boldsymbol{\theta})$ after a suitable mini-batch learning which estimates $\mathcal{C}^{l_2}_{n,K,\mathcal{E}}(\boldsymbol{\theta})$ sufficiently small ($< 0.001$ for example) [105]

In [105] authors used Powell's method (see appendix) as fine tuning. When we halt the optimization depends on $\mathcal{C}^{l_1}_{n,K,\mathcal{E}}(\boldsymbol{\theta})$ such that

$$\mathcal{C}^{l_1}_{n,K,\mathcal{E}}(\boldsymbol{\theta}) < \mathcal{C}^{l_1}_{tol} \tag{4.7}$$

where $\mathcal{C}^{l_1}_{tol}$ is a tolerance value for cost function which we set (in [105], this value is set $\mathcal{C}^{l_1}_{tol} = 10^{-6}$) as a hyperparameter. Before satisfying equation 4.7, we continue by increasing the layer number $L$ followed by optimizations until we reach $L = L_{max}$ case at the end of which we find that there is no such code. In the case when we satisfy equation 4.7, halting the optimization gives us the fine-tuned parameters $\boldsymbol{\theta}_{opt}$ such that ideally are:

$$\boldsymbol{\theta}_{opt} = \arg(\min_{\boldsymbol{\theta}}(\mathcal{C}^{l_1}_{n,K,\mathcal{E}}(\boldsymbol{\theta}))) \tag{4.8}$$

So, the resulting approximate QECC is:

$$\begin{aligned} C_{out}(\boldsymbol{\theta}_{out}) = span\{\, |\psi_0\rangle = U(\boldsymbol{\theta}_{opt})\,|\mathbf{1}\rangle\,|0\rangle^{\otimes(n-k)}\,, |\psi_1\rangle = U(\boldsymbol{\theta}_{opt})\,|\mathbf{2}\rangle\,|0\rangle^{\otimes(n-k)}\,, \\ \cdots, |\psi_K\rangle = U(\boldsymbol{\theta}_{opt})\,|\mathbf{K-1}\rangle\,|0\rangle^{\otimes(n-k)}\} \end{aligned} \tag{4.9}$$

with sufficiently small $\epsilon$. Here we can consider $U(\boldsymbol{\theta}_{opt})$ as encoding circuit and if there are redundant gates in them, we can erase them.

In [105] authors demonstrated the capability of the algorithm by making to find $((6,2,3))_2$ and $((7,2,3))_2$ QECCs but also showing that $((7,3,3))_2$ codes are not possible. It found channel adaptive codes for nearest-neighbour correlated error models.

If the cost function in the final is low enough, we have approximate quantum codes with lower bound inaccuracy.

## 4.2 Optimizing QECCs with Reinforced Learning

Surface codes as we discussed in chapter 2 are used in this method with the help of *reinforcement learning (RL)* and *transfer learning* techniques to modify logical error rate to a desired value. We follow [106] through this section.

As a summary, we feed the agent with recent status of the code structure. What agent has to do is to modify the code and take the logical error rate below a desired threshold value. Agent modifies the code via fault-tolerant deformations locally.[107] Reward is given to the agent if the logical qubits are protected. This type of reward-punishment mechanism is called reinforced learning.

One upside about this technique is that the agent don't have the information that it is in whether in experiment or a simulation in other words, it works in black-box environment. As a result, it can be used for different platforms such as ion traps or superconducting qubits etc. Agent modifies surface codes by adding new qubits and connections. To estimate logical error rate, a benchmarking algorithm called *SQUAB* is used. (see appendix) As a learning algorithm, projective simulation is used. After training the agent in one case, transfer learning is used to use it in other case. This paper shows that transfer learning can be used to optimize algorithms in simulations and then use them in experiments, therefore with low costs.

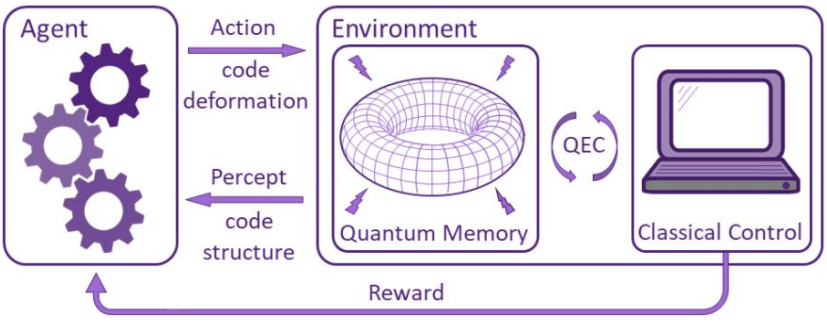



Figure 4.1: Reinforced learning framework [106]

Note that the framework in fig.4.1 can be used with different codes, decoders, error

models or RL methods with promising potential.

### 4.2.1 Reinforcement Learning and Projective Simulation

Generally in a RL setting, agent and environment interact with the following way: Perceptual input (*percepts*) are given to the agent at each step, percepts contain information about the environment. Then agent triggers the reward mechanism with its actions. Here, agent tries to maximize its rewards per time step. In [106] no assumption about the environment are made so it seemed as if it is a block box, and environment here is the surface code memory and a control mechanism that estimates the logical error rates. Agent don't have the information about the error model. To have information about error model, process tomography is needed which is an energy intensive process, and we expect agent to suppress error causes instead of estimating its model.

#### 4.2.1.1 Projective Simulation Model (PSM)

Learning algorithm we use is developed for physics simulations.[109] This network contains episodic memory units called *clips* and those clips are consist of *percept* and *action* clips. Let $P$ and $A$ be disjoint sets of percepts and actions respectively. Interactive RL has agents that know the environment via percept clip (which can be considered as the possible environment states) $s_i \in P \; i = 1, \cdots, N^{(t)}$ where superscript of $N^{(t)}$ indicates the time step and $N$ is the percept number, and then agent reacts with an action clip (which can be considered as an operator acting on the environment) $a_j \in A \; j = 1, \cdots, M_i$ where $M_i$ gives the available action number for a given percept $s_i$.

In fig.4.2, we see a clip network having two layers. Notice that this bipartite graph is directed which makes it different from Tanner graphs. This is because in this case, graph shows the connection between percept and action occurs with possibility $\mathcal{P}_{ij} \coloneqq p(a_j|s_i)$ (meaning probability of action $a_j$ in the case of $s_i$) with some directed edge $(i, j)$.
Agent decides how will it act next according to the transition probability in the

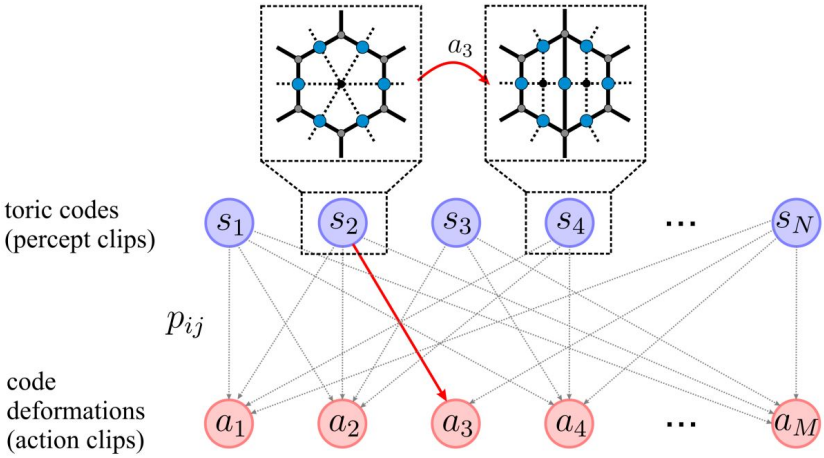



Figure 4.2: Bipartite graph of PSM [106]

episodic memory. Newly formed clips updates network as a result updates the transition probabilities as well. The learning is occurred in this way. New percept at every step comes out and added to the $P$ set.

Now let's define $h$-value $h_{ij}^{(t)}$ which is some time-dependent weight related to an edge $(i, j)$. Then probability of percept $s_i$ transitioning to action $a_j$ is:

$$\mathcal{P}_{ij}^{(t)} = \frac{e^{\beta h_{ij}^{(t)}}}{\sum_{k=1}^{M_i} e^{\beta h_{ij}^{(t)}}} \tag{4.10}$$

called *softmax function* [110] where $\beta > 0$ is *softmax parameter*.
At the beginning, agent acts randomly which corresponds to uniform transition probabilities in which $h$-values are all 1. Then non-negative rewards $\lambda^{(t)}$ will be given to the agent by the environment to reinforce it adjusting transition probabilities in which agent tries to find the action $a_j$ on given percept $s_i$ maximizing the reward. Note that agent is looking forward to the reward received in later iterations. So, matrix elements $h_{ij}$ of $h$-matrix are updated due to environment feedbacks $\lambda^{(t)}$. The update of $h$-values are done with following update rule:

$$h^{(t+1)} = h^{(t)} + \lambda^{(t)} g^{(t)} + \gamma(1 - h^{(t)}) \tag{4.11}$$

which describes the step from time $t$ to $t + 1$. Here $g$ is the *glow matrix* and $\gamma \in [0, 1]$

which serves as a forgetting mechanism such that at each update, it decreases $h$-values. Note that it doesn't make $h$-values less than 1.

And the glow matrix basically is the long-term memory. What glow matrix does is making $\lambda^{(t)}$ in the further past decrease (or remembered less strongly) so that past experiences rarely affect future attitutes. At this point, one may ask, if we want past to influence less as the time goes, why do we keep it at the first place? The reason is that not all actions end up with a reward. More generally, when a reward is delayed, agent uses its past experiences. At the beginning, glow matrix is a zero matrix and updated parallel to $h$-matrix. We set $g_{ij} = M_i/M_0$ where $M_i$ and $M_0$ are action numbers at step $i^{th}$ and initial steps recpectively, whenever some edge (i,j) is crisscrossed while making decision. Note that glow matrix accounts how much past a reward belongs to by glow parameter $\eta \in [0, 1]$ with following update rule:

$$g^{(t+1)} = (1 - \eta)g^{(t)} \tag{4.12}$$

at each iteration.

Moreover, we have an erasing mechanism in order to save memory in which percept clips that are not used are erased. For instance we delete $h$-values if their average is below $1 + \delta$ value. Note that to begin erasing procedure, agent has to wait until rewarded iteration number $\tau$ (*immunity time*) to pass some value. In addition, if in an iteration no rewards are given, all the clips are removed.

Notice that $\beta, \gamma, \delta, \eta$ and $\tau$ are hyperparameters meaning that they have to be adjusted by hand. Although PSM can learn them, it takes so much time.[111] At the end, optimized transition probabilities which is the clip network serves as the memory of the agent governing the decision making.

#### 4.2.1.2 Transfer Learning

In transfer learning, agent can apply its past experiences in a situation to another new situation that is related. Best side of this technique is that we don't need to start over whenever environment or the task changed a little. Agent can use its experiences which are gained from the simulation on experiments. The success of this technique depends on how much the error model in the simulation is similar to the real life version. To see how powerful transfer learning in quantum memories is, we first train our

agent in some error model $\mathcal{E}$ then we give it another, more realistic error model $\tilde{\mathcal{E}}$. The change in the error may caused by the change in the environment, instrument failure, malicious attack or going from simulation to the experiment. Since the agent has no information about the environment, it can solve all of them without even knowing the problem.

In general, if the error channel is changing little, learning experince in some channel should be useful in another related channel. So, RL optimizes in a method and adjusts it according to a new environment, in this way strength of RL increases since long term memory is required between tasks. Agent has global information about the environment in its memory if the agent explored the search space with some reward mechanism. In short, if an unoptimal QECC is given to the agent, it knows what to do.

In [106] for a quantum memory, agent is subjected to error channel:

$$\mathcal{E}(\rho) = pZ\rho Z + (1-p)\rho \tag{4.13}$$

with $p = 0.14$ and its task was to decrease logical error rate lower than $P_L^{rew} = 0.001$ logical rate for reward. After agent discovered a strategy to find some good QEC, agent is expected to perform under $p = 0.16$ case. In fig.4.3(a) we see how agent used its experiences on $p = 0.14$ case to protect qubits in $p = 0.16$ case. Note that what agent achieved in $p = 0.16$ case is not possible if it would start from scratch within 6000 trials.

For more realistic cases, 60 agents trained on error model in equation 4.13 with $p = 0.1$. After enough iterations for agents to find good strategies, one agent from bests is chosen at random to encounter a realistic model with:

$$\tilde{\mathcal{E}}_{1,k}(\rho) = p_{x,k}X\rho X + p_{z,k}Z\rho Z + (1 - p_{x,k} - p_{z,k})\rho \tag{4.14}$$

where $k$ indicates which qubit is subjected to error. This model contains Pauli $X$- and$Z$-errors. Error rates for all qubits are set to $p_{x,k} = 0.02$ & $p_{z,k} = 0.14$ and to make it even more realistic, spatial correlations (which may occur due to manifacturing error) modeled by adding 0.15 to $p_{x,k}$ if that qubit $k$ is in the neighbourhood of some plaquette.

To push the limits of the algorithm, second stage contains trial number of the second

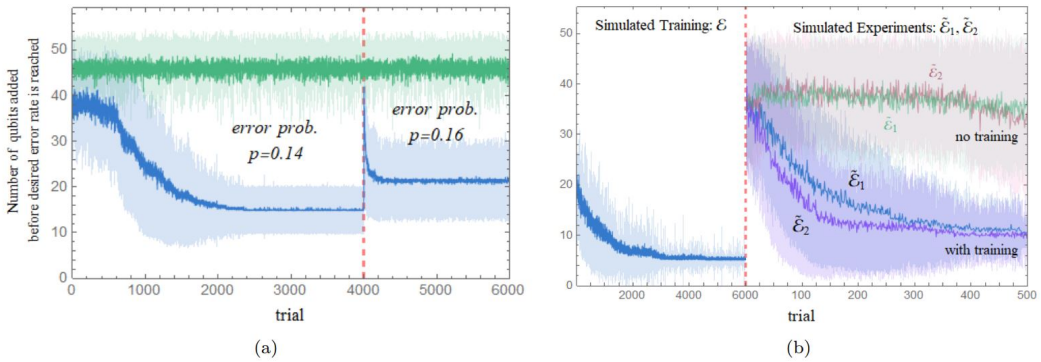



Figure 4.3: (a) Transfer learning in error model $\mathcal{E}$(b) Comparison of pre-trained (with error model $\mathcal{E}$) and unexperienced agents in more realistic error models $\tilde{\mathcal{E}}_1$ and $\tilde{\mathcal{E}}_2$.[106]

stage cannot exceed 10% of the first one. This is because in real life, doing an experiment is more resource intensive compared to a simulation.
In fig.4.3(b) we see that agents can use their knowledge to achieve good QECCs with less qubits added as drawn in blue. On the other hand, drawn in green, agents without previous experience failed to decrease required number of added qubits within allowed trial number.

Now to see if learning transfer works, we consider another error model in which $p_x$ is doubled and spatial correlations don't exist as:

$$\tilde{\mathcal{E}}_2(\rho) = p_x X\rho X + p_z Z\rho Z + (1 - p_x - p_z)\rho \tag{4.15}$$

where $p_z = 0.14$ and $p_X = 0.04$. Again taking some agents with experience and transfer them to new setup, we see (in fig.4.3(b)) how pre-trained and unpre-trained agents differ (drawn with purple and pale rose respectively) as well as success of pre-trained agents on $\tilde{\mathcal{E}}_1$ and $\tilde{\mathcal{E}}_2$.
Therefore we conclude that transfer learning is useful for quantum memories of near future with limited resource. We see that agent trained on a simulation can transfer its experience onto experiment.

## 4.3 Approximate Autonomous QEC with Reinforced Learning

Autonomous QEC doesn't contain repetitive feedback cycles of measurement with continuous dissipation by engineering the reservoir as we discussed in chapter 3. In this section, we follow [112]. Knill-Laflamme conditions dictates requirement for challengingly realized intricate and delicate superpositions of Fock states. Due to this, we needed distance $d \geq 2$. In [112], Knill-Laflamme conditions are untightened resulting possibility of lowering the distance, meaning that, we now won't getting exact QEC. Therefore we apply approximate AQEC (AAQEC) which we try to optimize via RL and consider successful when found AQEC exceeds break-even point. Turns out that in optimal AQEC we use $|2\rangle$ and $|4\rangle$ Fock states for codewords, in this way RL exceeds break-even point decreases infidelity more than 80%. In the meantime lowers distance to $d = 1$. (before this study, we had $d_g = 2$ at best) RL based AAQEC can be achieved by implementing encoded bosonic mode having two-level lossy ancilla using already existing techniques.

### 4.3.1 Approximate AQEC (AAQEC)

In AAQEC, we engineer dissipation to prevent natural decay processes. Once we intensify the effects of engineered Lindblad operators $\sum_j \mathcal{D}(L_{eng,j})$, infidelity (defined between some arbitrary state $\rho(\theta, \phi, t_0) = |\psi_{\theta,\phi}\rangle\langle\psi_{\theta,\phi}|$ and state after evolution $\rho(\theta, \phi, t) = \mathcal{M}(\rho(\theta, \phi, t_0)))$ growth is lowered. Here dissipative channel $\mathcal{M}(\bullet)$ contains natural Lindblad operators $\sum_j \mathcal{D}(L_{nat,j})$ (where $L_{nat} \in \{\mathbb{I}, a\}$) and responsible for minimization of infidelity due to engineered Lindblad operators. Note that $\theta$ and $\phi$ define the quantum state in Bloch sphere and

$$\mathcal{D}(x) = 2xx^\dagger - x^\dagger x\rho - \rho x^\dagger x \tag{4.16}$$

If we pick logical codewords $\left|\overline{0}\right\rangle$ and $\left|\overline{1}\right\rangle$, which decides the mean fidelity $\overline{\mathcal{F}}(\left|\overline{0}\right\rangle, \left|\overline{1}\right\rangle)$, in such a way that Knill-Laflamme conditions are satisfied which can be difficult task for bosonic AQECCs, we can reach exact QEC. In addition, if the system needs many nonlinear interactions with high order and numerous control fields, implementation is challenging. So instead we use AAQEC and introduce a loosening in Knill-Laflamme conditions by excluding $\left\langle\overline{1}\right|L_{nat,j}^\dagger L_{nat,j}\left|\overline{1}\right\rangle = \left\langle\overline{0}\right|L_{nat,j}^\dagger L_{nat,j}\left|\overline{0}\right\rangle$ constraint. Then

codewords satisfying remaining conditions are:

$$\begin{aligned} \left|\overline{0}\right\rangle = \sum_{i=0} c_i^{(0)} \left|4i\right\rangle \quad \& \quad \left|\overline{1}\right\rangle \sum_{i=0} c_i^{(1)} \left|4i+2\right\rangle \\ \ni \sum_i \left|c_i^{(j)}\right|^2 = 1 \ \& \ c_i^{(j)} \in \mathbb{R} \ \forall j \in \{0,1\} \end{aligned} \tag{4.17}$$

where $c_i^{(j)}$ coefficients are left unknown and will be used for optimizing AQEC. We have one jump operator:

$$L_{eng} = \frac{L_0}{\sqrt{\mathrm{Tr}\left(L_0^\dagger L_0\right)}} \quad \ni L_0 = \left|\overline{0}\right\rangle \left\langle 0_{err}\right| + \left|\overline{1}\right\rangle \left\langle 1_{err}\right| \tag{4.18}$$

where $\left|i_{err}\right\rangle = \frac{a\left|\bar{i}\right\rangle}{\sqrt{\left\langle\bar{i}\right|a^\dagger a\left|\bar{i}\right\rangle}}$ $\forall i \in \{0,1\}$ are states in the error space. An encoded state in error space is carried by $L_{eng}$ to codespace. We can expand $L_{eng}$ as:

$$L_{eng} = \sum_{|d'|\leq d} \sum_i \lambda_{id'} \left|i\right\rangle \left\langle i+d'\right| \tag{4.19}$$

When we add some lossy ancilla, coupling Hamiltonian $H_{eff} = g(L_{eng}\sigma^+ + L_{eng}^+\sigma^-)$ can be used to describe on engineered dissipation. Tracing out the qubit gives Lindblad superoperator $\mathcal{D}(L_{eng})$ and system is evolved by:

$$\frac{d\rho}{dt} = -i[H_{eff}, \rho] + \frac{\gamma_a}{2}\mathcal{D}(a) + \frac{\gamma_b}{2}\mathcal{D}(\sigma^-) \tag{4.20}$$

where $\gamma_a$ is the rate of single-photon loss of mode $a$ and $\gamma_b$ is the decay rate of ancilla. Note that we assumed $\gamma_a \ll g \ll \gamma_b$. Recovery process has two possible QEC cycles:

$$\begin{aligned} &\bullet \left|\psi_{\theta,\phi,0}\right\rangle \xrightarrow{\gamma_a} \left|\psi_{\theta,\phi,0}^{err}\right\rangle \xrightarrow{g} \left|\psi_{\theta,\phi,1}\right\rangle \xrightarrow{\gamma_b} \left|\psi_{\theta,\phi,0}\right\rangle \\ &\bullet \left|\psi_{\theta,\phi,0}\right\rangle \xrightarrow{\gamma_a} \left|\psi_{\theta,\phi,0}^{err}\right\rangle \xrightarrow{g} \left|\psi_{\theta,\phi,1}\right\rangle \xrightarrow{\gamma_a} \left|\psi_{\theta,\phi,1}^{err}\right\rangle \xrightarrow{\gamma_b} \left|\psi_{\theta,\phi,0}^{err}\right\rangle \xrightarrow{g} \left|\psi_{\theta,\phi,1}\right\rangle \xrightarrow{\gamma_b} \left|\psi_{\theta,\phi,0}\right\rangle \end{aligned} \tag{4.21}$$

### 4.3.2 Optimal Codespace

To find optimum values of $c_i^{(0)}$ and $c_i^{(1)}$ for maximizing mean fidelity $\overline{\mathcal{F}}$ by fixing time as a reference, we use RL. At the beginning, we divide every agent-environment interaction (called *episode*) into $k = [1, K]$ steps. For all steps, in step $k$ for example, agent

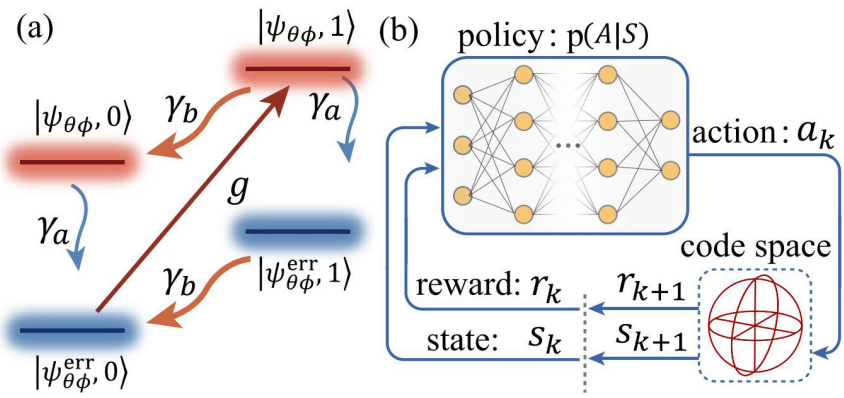



Figure 4.4: (a) AAQEC transitions between energy levels. (b) Schematics of AAQEC.[112]

looks the state $s_k \in S$, that is the recent state at that step, and according to $p(A|S)$ it decides to make action $a_k \in A$ where $p(A|S)$ is the *policy function*.responsible for choosing an action using current state and reward. We describe action with $[c_i^{(0)}, c_i^{(1)}]$ coefficient vector and fidelity $\mathcal{F}(\theta, \phi, t_0) = \mathrm{Tr}(\rho(\theta, \phi, t_0)\rho(\theta, \phi, t))\ \forall\theta \in \{0, \pi/2, \pi\}, \phi \in \{0, \pi/2, \pi, 3\pi/2\}$ corresponds to the state. After acting on the system, action gives reward $r_{k+1}$ to the agent and takes state to $s_{k+1}$. Here what reward try to maximize is break-even point and mean fidelity difference.

In the study [112], state and reward are solved by simulating the equation 4.20 via QuTiP library till $\gamma_a t$ becomes a specific value that is set by hand. Besides $\gamma_a t$, $g/\gamma_a$ and $\gamma_b/\gamma_a$ are also hyperparameters. When the algorithm takes state and reward values during sampling, it inputs them to proximal policy optimization (PPO) (see appendix) and updates $p(A|S)$ accordingly.

In [112] RL algorithm found a code with $\overline{\mathcal{F}} \approx 0.95$ at $\gamma_a t = 0.6$ with Fock states: $|\overline{0}\rangle \approx |4\rangle$ and $|\overline{1}\rangle \approx |2\rangle$. This result is experimentally achievable with current techniques and $L_{eng} \propto |2\rangle\langle 1| + |4\rangle\langle 3|$ has distance $d = 1$ meaning that we don't need many control fields or nonlinear interactions to implement Hamiltonian. They also found that fidelity $\mathcal{F} \in [0.93, 1]$ which is above the break-even threshold which is 0.84 implying that every quantum state within the codespace of RL are protectec by AQEC.

### 4.3.3 Coupling Engineering

To achieve a distance-1 Hamiltonian, one possible construction includes encoding mode $a$ coupled with dissipative mod $c$ via intermediate qubit (see fig.4.5) with associated Hamiltonian:

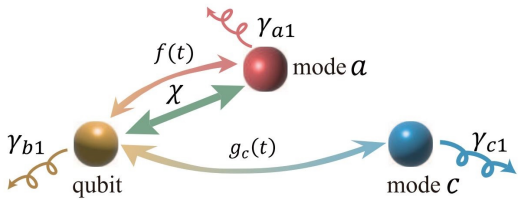



Figure 4.5: System-environment coupling for AAQEC [112]

$$H = \omega_a a^\dagger a + \omega_c c^\dagger c + \left(f(t)(a + a^\dagger) + g_c(t)(c^\dagger + c)\right) X + \left(\frac{\omega_b}{2} + \frac{\chi}{2} a^\dagger a\right) Z \quad (4.22)$$

where $f(t)$ and $g_c(t)$ are control fields such that:

$$f(t) = \frac{2\alpha_0}{\sqrt{2}} \cos\left((\omega_s + \frac{3\chi}{2})t\right) + \frac{2\alpha_0}{2} \cos\left((\omega_s + \frac{7\chi}{2})t\right) \quad (4.23)$$

$$g_c(t) = 2\alpha_1 \cos(2\chi t) + 2\alpha_1 \cos(4\chi t) \quad (4.24)$$

Here $\omega_s = \omega_a + \omega_b$ is blue sideband transition resonant frequency, $\alpha_j$ $(j = 0, 1)$ control field strengths, $\chi$ is coefficient that is nonlienar and $\omega_i$ $\forall i \in \{a, b, c\}$ are mode and qubit resonant frequencies.

One upside about this technique is that it decreases the distance at minimum possible while keeping the fidelity above the break-even threshold value. Why this is important comes from the fact that high distance means high number of many-body interactions and control fields.

# CHAPTER 5

# CONCLUSIONS

We have seen that quantum systems can be protected against various quantum errors using QEC codes. We can derive such codes from classical error correction codes or designed solely for quantum systems. Such codes can be constructed with product of two different codes. This type of methods will provide us with higher distance and the properties of the ingredient codes. Besides that, we can exploit the geometrical structures to construct QECCs thanks to homology. Identifying $i$-cells of the homology, we can achieve desired quantum codes. Moreover, with the help of machine learning methods as we discussed in chapter 4, we can optimize such geometric codes with less resources but more logical qubits, as a result with better encoding rates.

Machine learning in both classical and NISQ devices may help us to construct not just only surface codes but also may help us to make better simulations to carry more precise experiments. Thanks to transfer learning, we can make experiments with low costs. Machine learning techniques also allow us to construct QECCs with low distance values while keeping the fidelity values above the break-even threshold. Due to the fact that current quantum systems have few qubits, we always loose some resource to the encoding giving us less logical qubits compared to physical qubits. However, codes with high encoding rate can be used for future quantum systems with high number of qubits. Such codes with large sizes can be achieved by machine learning.

One important problem of scalable quantum computation is the quantum memories. We also discussed various candidates. Most of them require macroscopic energy gap between ground and excited states. One downside of such codes is that they require

more than 2 dimensions. We can construct quantum codes in higher dimensions like 4D toric code or Haah cubic code. We also provide a new approach to Haah cubic codes. In this new approach, instead of naming vertices of the cubes, we can use carthesian coordinates. Therefore, we managed reducing the number of the equations for commutation relations of the corner operators and the conditions that tell whether an operator is logical operator or not. Besides higher dimensional structures, mechanisms like introducing a defect grid and causing energy penalty for excitations might be a solution for constructing 2D quantum memories. Or we can try to achieve structures that haven't been tested before using machine learning.

For future work, using the VarQEC method with deeper networks as the NISQ devices become more robust against noise, we can try to implement transfer learning technique to achieve passive protection of quantum information.

# APPENDIX A

# COMPLEXITY NOTATION

## A.1 Bachman-Landau Asymptotic Notation

The notation used to describe how a function is bounded which is used in complexity theory. We use this notation mostly for distances.

i) $f = \mathcal{O}(g)$ if $\exists\ c\ \&\ x_0 \ni f(x) \leq cg(x)\ \forall x > x_0$

ii) $f = \Omega(g)$ if $\exists\ c\ \&\ x_0 \ni f(x) \geq cg(x)\ \forall x > x_0$

iii) $f = \Theta(g)$ if $f = \mathcal{O}(g)$ & $f = \Omega(g)$ (implying $f(x) = cg(x)$)

iv) $f = o(g)$ if $\lim_{x\to\infty}(f(x)/g(x)) = 0$

where $c$ is a constant. [56]

# APPENDIX B

# QLDPC TOOLS

## B.1 Homological Algebra

We can use homological algebra to generate QECCs. In this section, following [64], we give a brief overview on chain complexes.

To begin with, let $\partial_i$ be linear maps called *boundary operators*, $\mathtt{C}_i$ are $\mathbb{F}_2$ vector spaces and chain complex $\mathtt{C}$ with length $n+1$:

$$\mathtt{C} = (\mathtt{C}_n \xrightarrow{\partial_n} \cdots \xrightarrow{\partial_2} \mathtt{C}_1 \xrightarrow{\partial_1} \mathtt{C}_0) \tag{B.1}$$

satisfying

$$\partial_i \partial_{i+1} = 0 \tag{B.2}$$

And to describe $\partial_i$ as matrices, assume that all vector spaces $\mathtt{C}_i$ have a basis allowing it. Since chain complexes are from algebraic topology, we call basis vectors of $\mathtt{C}_i$ as $i$-cells and elements $i$-chains ($i$-cells span $i$-chains). To make it more concrete, the $i$-cells are elements of subdivision, i.e., a 0-cell is a vertex, a 1-cell is an edge, a 2-cell is a face.

An $i$-cycle is an element in $Ker(\partial_i)$ since it is an $i$-chain having trivial boundary and $i$-boundary is an element in $Im(\partial_{i+1})$ since it is an $i$-chain in the image of the boundary operator.

The $i^{th}$ homology $H_i(\mathtt{C})$ is the vector space of $i$-cycles (mod($i$-boundaries)) where

$$H_i(\mathtt{C}) = \frac{Ker(\partial_i)}{Im(\partial_{i+1})} \tag{B.3}$$

and dually, we define $i$-cohomology

$$H^i(\mathtt{C}') = \frac{Ker(\partial_i^T)}{Im(\partial_{i+1}^T)} \tag{B.4}$$

$i$-cocycles and $i$-coboundaries where $\mathtt{C}'$-operators are related to the cohomology group. Note that homologically speaking, a classical code is a chain complex of length-2 in which boundary operator defines parity checks and quantum code is length-3 with 2 boundary operators corresponding to PCMs $H_x$ & $H_z$.

$$\mathtt{C} = (\mathtt{C}_2 \xrightarrow{\partial_2 = H_z^T} \mathtt{C}_1 \xrightarrow{\partial_1 = H_x} \mathtt{C}_0) \tag{B.5}$$

As an example, consider a surface. Here, when we apply $\partial_2$ to a face, we get 4 edges, applying $\partial_1$ to an edge gives two vertices which they are the boundaries of the initial $i$-cell and so on.
By this way, logical $Z$-operators are similar to the homology group
$H_1(\mathtt{C}) = Ker(\partial_1)/Im(\partial_2)$ while logical $X$s are similar to the cohomology group $H^1(\mathtt{C}) = Ker(\partial_2^T)/Im(\partial_1^T)$, logical qubits $k = \dim(H_1(\mathtt{C}))/\dim(H^1(\mathtt{C}))$. The distance $d = \min(d_z, d_x)$ where $d_x$ & $d_z$ are respectively minimum Hamming weight of all non-trivial cohomology and homology classes. On the other hand with two consecutive boundary operators, single chain complex can give many CSS codes.

Quantum codes can be constructed with the tessellation of manifolds or surfaces. Toric code is one of the most famous examples which is a torus with square tiles where faces are $Z$-checks, vertices are $X$-checks and edges are the physical qubits. In general, for a tessellation of a $D$ dimensional manifold, we see qubits as $i$-cells where $0 < i < D$, $Z$-checks as $(i+1)$-cells and $(i-1)$-cells as $X$-checks.
Note that $i$-cell subsets with no boundaries and not being the boundary of $(i+1)$-cell subsets correspond to non-trivial logical $Z$-operators. Moreover, vector spaces spanned by $i$-cells and the boundary operator form cellular chain complex which makes a bridge between homological and geometrical perspectives.
We also can consider Hasse diagram of tessellation to connect to Tanner graphs. Same manner is applied to $X$-logicals by either taking the dual tessellation or taking cohomology classes into account.

In a higher dimensional manifold $\mathcal{M}$ with genus $g$ (the holes on the manifold), the length, area or the volume of the smallest non-contractible $i$-dimensional submanifolds are called $i$-systole (denoted by $sys_i(\mathcal{M})$) and it is related to the distance as:

$$d \sim \min_{[\gamma] \in H_i} vol_i(\gamma) \tag{B.6}$$

where $\gamma$ is essential cycle. In other words, minimum weights of logical operators are related to the minimum volumes of essential cycles. This gives us families of codes and for suitable tessellations they are LDPC codes.

## B.2 Tanner Graphs

A bipartite graph with partitions associated with bits and checks is called a Tanner graph and we can use it to describe any linear code $C$. In fig.B.1, we see the Tanner graph of [7,4,3] Hamming code where squares are representing the physical bits while squares are representing checks.

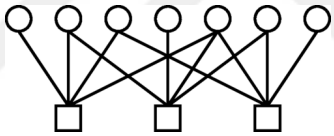


Figure B.1: Tanner graph of [7,4,3] Hamming code [64]

In CSS codes we add another layer since we have both $X$ & $Z$ checks. The fig.B.2 shows the Shor code Tanner graph where circles are qubits this time, top squares are $Z$-checks and bottom squares are $X$-checks. Note that to create geometric interpretation, we choose a linearly dependent set of stabilizer checks. Note that incident matrices between the layers are PCMs $H_x$ & $H_z$.

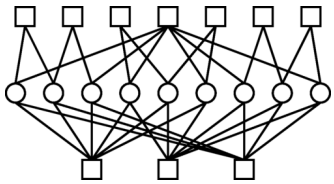


Figure B.2: Tanner graph of Shor's code [64]

### B.3 Distance Balancing

A technique suggested by Evra-Kaufman-Zemor in [55] allows us to construct a quantum code with nearly equal distances of $X$ and $Z$. Note that for this topic we follow [55]. Consider an abstract chain complex $U = (U_0, U_1, U_2)$ where $U_0$ & $U_2$ index the row set of low-density PCMs $H_x$ & $H_z$ of quantum code $Q(U)$ while $U_1$ indexes the column set of both $H_x$ & $H_z$.

Now consider a 1D chain complex $W = (P, R)$ where the sets $P$ & $R$ index the columns and rows of the PCM $H$ of some classical LDPC code $C(W)$. And note that $H$ doesn't have any redundant rows.

Then 2D chain complex $\Gamma = (\gamma_0, \gamma_1, \gamma_2)$ allows us to construct new quantum code $Q(\Gamma)$ such that

$$\begin{aligned} \gamma_0 &= (U_0 \times P) \cup (U_1 \times R) \\ \gamma_1 &= (U_1 \times P) \cup (U_2 \times R) \\ \gamma_2 &= U_2 \times P \end{aligned} \tag{B.7}$$

Now we define the incidences between $\gamma_i$ elements in $U$ & $W$:

- Between $\gamma_0$ and $\gamma_1$:
  - in $U$:
    $(u_1, p)$ incident to $(u_0, p)$ where $(u_1, p) \in U_1 \times P$
    for all $u_0$ incident to $u_1$ where $u_0 \in U_0$
  - in $W$:
    $(u_1, p)$ incident to $(u_1, r)$
    for all $r$ incident to $p$ where $r \in R$
- Between $\gamma_1$ and $\gamma_2$:
  - in $U$:
    $(u_2, p)$ incident to $(u_1, p)$ where $(u_2, p) \in U_2 \times P$
    for all $u_1$ incident to $u_2$ where $u_1 \in U_1$
  - in $W$:
    $(u_2, p)$ incident to $(u_2, r)$
    for all $r$ incident to $p$ where $r \in R$

So, the quantum code $Q(\Gamma)$ has length, dimension and distances of $X$ & $Z$:

$$\begin{aligned} N &= |U_1||P| + |U_2||R| \\ K &= \dim(Q)\dim(C) \\ D_x &= d_x(Q)d(C) \\ D_z &= d_z(Q) \end{aligned} \tag{B.8}$$

Meaning that for a quantum code with $d_z \gg d_x$, $\Gamma$ construction via a classical code with $d \approx d_z/d_x$ can be used to generate a new quantum code having $D_x \approx D_z$.

# APPENDIX C

# MACHINE LEARNING TOOLS

## C.1 SQUAB algorithm

This algorithm allows us to benchmark a surface code with linear complexity.[108] The main routine of the algorithm takes a tiling $G$, its dual $G^*$ and an erasure channel $\mathcal{E}$ and decides if error $\mathcal{E}$ can be corrected by given surface code with tiling $G$.
For homology group $H_1(G_\mathcal{E})$ which is generated by the edge of $\mathcal{E}$, denote

$$h_1(G, \mathcal{E}) \equiv \dim(H_1(G_\mathcal{E})) \tag{C.1}$$

After computing $h_1(G, \mathcal{E})$ & $h_1(G^*, \mathcal{E}^*)$, we check the following:

$$h_1(G, \mathcal{E}) + h_1(G^*, \mathcal{E}^*) \leq 0 \tag{C.2}$$

If equation C.2 is satisfied, it is correctable, otherwise it is uncorrectable. The software is available at [108]

## C.2 Finite Difference Method

The technique allowing us to approximate $n^{th}$ order derivation $\frac{d^n f(x)}{dx^n}$ of a given function $f(x)$ at a point $x = h$ which is based on Taylor series expansions. Beginning with [114]

$$f(x+h) = f(x) + h\frac{f(x)}{dx} + \frac{h^2}{2!}\frac{d^2 f(x)}{dx^2} + \frac{h^3}{3!}\frac{d^3 f(x)}{dx^4} + \frac{h^4}{4!}\frac{d^4 f(x)}{dx^4} + \cdots \tag{C.3}$$

$$f(x-h) = f(x) - h\frac{f(x)}{dx} + \frac{h^2}{2!}\frac{d^2 f(x)}{dx^2} - \frac{h^3}{3!}\frac{d^3 f(x)}{dx^4} + \frac{h^4}{4!}\frac{d^4 f(x)}{dx^4} + \cdots \tag{C.4}$$

then

$$\frac{df(x)}{dx} = \frac{f(x+h) - f(x-h)}{2h} + \mathcal{O}(h^2) \tag{C.5}$$

called $1^{st}$ *central difference approximation of* $f'(x)$. When we need to solve $f'(x)$ for right or left of $x$, we use forward or backward *finite difference approximation (FDA)*. To get forward FDA, we solve equation C.3 for $f'(x)$ and the result is the $1^{st}$ *forward difference approximation*:

$$\frac{df(x)}{dx} = \frac{f(x+h) - f(x)}{h} + \mathcal{O}(h) \tag{C.6}$$

Similarly, equation C.4 for $f'(x)$ yields $1^{st}$ *backward difference method*:

$$\frac{df(x)}{dx} = \frac{f(x) - f(x-h)}{h} + \mathcal{O}(h) \tag{C.7}$$

Note that we only considered $1^{st}$ differences but one may look for higher orders in [114]

## C.3 Parameter Shift Rule

Since we can express a variational quantum circuit or a unitary gate as a quantum function, say $f(\mathbf{x})$ where $\mathbf{x} = \{x_1, x_2, \cdots, x_N\}$, we can write its partial derivative as:[115]

$$\frac{\partial f}{\partial x} = c\left(f(x+\phi) - f(x-\phi)\right) \tag{C.8}$$

where $f = f(x)$, $c$ is some constant and $\phi$ is the shift.

Note that any gate in the form $U(\mathbf{x}) = e^{-i\mu O}$ where $\mu$ is the gate parameter and $O$ is some Hermitian operator with two or less eigenvalues have the following property: [116]

$$U(\mathbf{x}) = U_N(x_N)U_{N-1}(x_{N-1})\cdots U_1(x_1)U_0(x_0) \tag{C.9}$$

therefore we can use this rule on the gate.

## C.4 Powell's Method

In [117] Powell proposed a method for finding local minima without necessity of derivatives. And we will use this technique for fine tuning of $\boldsymbol{\theta}$ in VarQEC.
Suppose we find the minimum of function $f$. For this method, we need $n$ linearly independent directions $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n$ and a guess for minimum $P_0$. We then repeat the following routine:

- find $\lambda_i$ making $f(P_{i-1} + \lambda_i \mathbf{v}_i)$ minimum
- define new point $P_i = P_{i-1} + \lambda_i \mathbf{v_i}\ \forall i \in \{1, 2, \cdots, n\}$
- replace $\mathbf{v}_i$ with $\mathbf{v}_{i+1}\ \forall i \{1, \cdots, n-1\}$
- set $\mathbf{v}_i$ as $(P_n - P_0)$
- find $\lambda$ making $f(P_n + \lambda(P_n - P_0))$ minimum
- set $P_0 + \lambda(P_n - P_0)$ as new $P_0$

We keep the routine repeating and halt when the function doesn't decrease anymore.

## C.5 Proximal Policy Optimization (PPO)

This method is useful for hyperparameter optimization, or policy optimization. Before we define the method, let's introduce some functions:[118]

- State-value function of policy $\mathbf{p}$:

$$V^{\mathbf{p}}(s) = \mathbb{E}_{\mathbf{p}}\left(\sum_{j=0}^{\infty} \gamma^j r_{t+j+1} | s_t = s\right) \tag{C.10}$$

  where $\gamma \in [0, 1]$ is the discount factor and $\mathbb{E}_{\mathbf{p}}$ is the expectation over policy $\mathbf{p}$

- Action-value function of policy $\mathbf{p}$:

$$Q^{\mathbf{p}}(s, a) = \mathbb{E}_{\mathbf{p}}\left[\sum_{j=0}^{\infty} \gamma^j r_{t+j+1} | s_t = s, a_t = a\right] \tag{C.11}$$

- Probability ratio of new and old policies at time $t$, $r_t$, such that:

$$r_t(\boldsymbol{\theta}) = \frac{\mathbf{p}_{\boldsymbol{\theta}}(a_t|s_t)}{\mathbf{p}_{\boldsymbol{\theta}_{old}(a_t|s_t)}} \tag{C.12}$$

  where $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_{old}$ are the policy parameter vector at time $t$ and before the update respectively. Notice that these policies are stochastic (in the form of $\mathbf{p}(a|s)$ instead of $a = \mathbf{p}(s)$).

- Advantage function $A(s, a)$ at time $t$:

$$A(s, a) = Q(s, a) - V(s) \tag{C.13}$$

Then the function we want to optimize is:[118]

$$L^{CLIP}(\boldsymbol{\theta}) = \hat{\mathbb{E}}_t[\min(r_t(\boldsymbol{\theta}))\hat{A}_k, clip(r_t(\boldsymbol{\theta}), 1-\epsilon, 1+\epsilon)\hat{A}_k] \tag{C.14}$$

where $\hat{A}_k$ is the esimator of advantage function and $\epsilon$ is the hyperparameter governing the size of policy update. Note that we don't want this $\epsilon$ to be large.[118]

## C.6 Classical Shadow Approximation

Instead of trying to get full description of some quantum state, we can approximate its necessary information using few measurements. The resulting classical information is called *classical shadow*. This method can be used for fidelity estimation, entanglement verification, prediction of local Hamiltonian with many-body interactions etc. [119]

Suppose we have a quantum state $\rho$ having $n$-qubits and we have no prior information about this state. To get information out of it, we need to rotate and measure the resulting state again and again. To do so, we apply some random unitary operator $U$ to map $\rho$ to $U\rho U^\dagger$, and then measure it in the computational basis yielding classical result in a bit string $\left|\tilde{b}\right\rangle \in \{0,1\}^n$. We store *classical snapshot* $U^\dagger \left|\tilde{b}\right\rangle\left\langle\tilde{b}\right| U$ in the memory. We can consider the average of this mapping as some quantum channel $\mathcal{M}$ such as:[119]

$$\mathbb{E}[U^\dagger \left|\tilde{b}\right\rangle\left\langle\tilde{b}\right| U] = \mathcal{M}(\rho) \tag{C.15}$$

yielding:

$$\rho = \mathbb{E}[\mathcal{M}^{-1}(U^\dagger \left|\tilde{b}\right\rangle\left\langle\tilde{b}\right| U)] \tag{C.16}$$

where $\mathbb{E}$ is average over measurement. Note that $\mathcal{M}^{-1}$ is not physical since it is not completely positive, but since it is acting on classical information, we can use it. This classical snapshot $\tilde{\rho} = \mathcal{M}^{-1}(U^\dagger \left|\tilde{b}\right\rangle\left\langle\tilde{b}\right| U)$ gives the original state exactly if we take its expectation: $\mathbb{E}(\tilde{\rho}) = \rho$. [119]

Repeating this procedure $N$ times gives us an array of classical snapshots which we call classical shadow:

$$S(\rho; N) = \{\tilde{\rho_1} = \mathcal{M}^{-1}(U_1^\dagger \left|\tilde{b}_1\right\rangle\left\langle\tilde{b}_1\right| U_1), \cdots, \tilde{\rho_N} = \mathcal{M}^{-1}(U_N^\dagger \left|\tilde{b}_N\right\rangle\left\langle\tilde{b}_N\right| U_N)\} \tag{C.17}$$

Here $S(\rho; N)$ is the classical shadow of $\rho$ with length $N$. Technique for using this shadow differs based on what we're looking for. So, we leave it as further reading to the original paper: [119]

# REFERENCES


[1] Knill, E., Laflamme, R. & Milburn, G. *A scheme for efficient quantum computation with linear optics*. Nature 409, 46–52 (2001). https://doi.org/10.1038/35051009

[2] Knill, E., Laflamme, R. & Milburn, G. *Thresholds for Linear Optics Quantum Computation*, 2000. arXiv:quant-ph/0006120

[3] Rozgonyi, Áron & Széchenyi, Gábor, *Break-even point of the quantum repetition code*, 2023. arXiv:2303.17810 [quant-ph]

[4] Ralph, T. C. and Hayes, A. J. F. and Gilchrist, Alexei, *Loss-Tolerant Optical Qubits*, 2005. Phys. Rev. Lett. 95, 100501 (2005). DOI: https://doi.org/10.1103/PhysRevLett.95.100501

[5] Hayes, A. J. F., Gilchrist, A., Myers, C. R., and Ralph, T. C., *Utilizing encoding in scalable linear optics quantum computing*, 2004. Journal of Optics B: Quantum and Semiclassical Optics 6 533, DOI 10.1088/1464-4266/6/12/008

[6] Shor, Peter W., *Scheme for reducing decoherence in quantum computer memory*, 1995. Phys. Rev. A 52, R2493(R). DOI: https://doi.org/10.1103/PhysRevA.52.R2493

[7] Keane, Kyle and Korotkov, Alexander N., *Simplified quantum error detection and correction for superconducting qubits*, 2012. Phys. Rev. A 86, 012333, DOI:https://doi.org/10.1103/PhysRevA.86.012333

[8] Knill, E., Laflamme, R. & Milburn, G. *A scheme for efficient quantum computation with linear optics*. Nature 409, 46–52 (2001). https://doi.org/10.1038/35051009

[9] Nielsen, M., Chuang, I., *Quantum Computation and Quantum Information*, 2010.

[10] Nakahara, M. & Ohmi, T., *Quantum Computing From Linear Algebra to Physical Realizations*, 2008.

[11] Lidar, D.A. & Brun, T.A., *Quantum Error Correction*, 2013.

[12] Hivadi, M. *On quantum SPC product codes*. Quantum Inf Process 17, 324 (2018). DOI: https://doi.org/10.1007/s11128-018-2095-3

[13] Tonchev, Vladimir D. , *Error-correcting codes from graphs*, 2002. Discrete Mathematics, Vol. 257. DOI: https://doi.org/10.1016/S0012-365X(02)00513-7

[14] Wolf,J. , *On codes derivable from the tensor product of check matrices*. IEEE Transactions on Information Theory, vol. 11, no. 2, pp. 281-284, April 1965, doi: 10.1109/TIT.1965.1053771.

[15] Ostrev, D., et.al., *Classical product code constructions for quantum Calderbank-Shor-Steane codes*, 2022. arXiv:2209.13474 [quant-ph]

[16] Gottesman, Daniel, *Class of quantum error-correcting codes saturating the quantum Hamming bound*, 1996. Phys. Rev. A 54, 1862. DOI: https://doi.org/10.1103/PhysRevA.54.1862

[17] MacWilliams, F.J., Sloane, N.J., *The theory of error correcting codes*, 1977

[18] Steane, A. M., *Quantum Reed-Muller codes*,1996. in IEEE Transactions on Information Theory, vol. 45, no. 5, pp. 1701-1703, July 1999, doi: 10.1109/18.771249.

[19] Shrinivas Kudekar, Santhosh Kumar, Marco Mondelli, Henry D. Pfister, Eren Şaşoğlu, and Rüdiger Urbanke, *Reed-Muller codes achieve capacity on erasure channels*, 2016. In Proceedings of the forty-eighth annual ACM symposium on Theory of Computing (STOC '16). Association for Computing Machinery, New York, NY, USA, 658–669. https://doi.org/10.1145/2897518.2897584

[20] Campbell, E.T., Hussain, A. and Browne, D.E., *Magic-State Distillation in All Prime Dimensions Using Quantum Reed-Muller Codes*, 2012. Phys. Rev. X 2, 041021, Vol. 2, Iss. 4. DOI:https://doi.org/10.1103/PhysRevX.2.041021

[21] den Nest, M. V., Dehaene, J., De Moor, B., *Local unitary versus local Clifford equivalence of stabilizer states*, 2005. Phys. Rev. A 71, 062323 (2005). DOI: https://doi.org/10.1103/PhysRevA.71.062323

[22] den Nest, M. V., Dehaene, J., De Moor, B., *Graphical description of the action of local Clifford transformations on graph states*, 2004. Phys. Rev. A 69, 022316. DOI: https://doi.org/10.1103/PhysRevA.69.022316

[23] Zeng, B., Chung, H., Cross, A. W. and Chuang, I. L., *Local unitary versus local Clifford equivalence of stabilizer and graph states*, 2007. Phys. Rev. A 75, 032325. DOI: https://doi.org/10.1103/PhysRevA.75.032325

[24] Rains, Eric M., *Quantum codes of minimum distance two*, 1997. DOI: https://doi.org/10.48550/arXiv.quant-ph/9704043

[25] Hein, M., Eisert, J. and Briegel H. J., *Multiparty entanglement in graph states*, 2004. Phys. Rev. A 69, 062311. DOI: https://doi.org/10.1103/PhysRevA.69.062311

[26] `https://errorcorrectionzoo.org/c/quantum_hamming_css`

[27] Chao, R. and Reichnardt, B.W., *Quantum error correction with only two extra qubits*, 2017. Phys. Rev. Lett. 121, 050502 (2018). DOI: https://doi.org/10.1103/PhysRevLett.121.050502

[28] Bravyi, S. and Haah, J., *Magic state distillation with low overhead*, 2012. Phys. Rev. A 86, 052329 (2012). DOI: https://doi.org/10.1103/PhysRevA.86.052329

[29] Nezami, S. and Haah, J., *Classification of small triorthogonal codes*, 2022. Phys. Rev. A 106, 012437. DOI: https://doi.org/10.1103/PhysRevA.106.012437

[30] Jones, Cody, *Multilevel distillation of magic states for quantum computing*, 2013. Phys. Rev. A 87, 042305. DOI: https://doi.org/10.1103/PhysRevA.87.042305

[31] Gottesman, Daniel, *A Theory of Fault-Tolerant Quantum Computation*, 1997. Phys.Rev.A57:127,1998. DOI: https://doi.org/10.1103/PhysRevA.57.127

[32] Vaidman, L., Goldenberg, L. and Wiesner, S., *Error prevention scheme with four particles*, 1996. Phys. Rev. A 54, R1745(R). DOI:https://doi.org/10.1103/PhysRevA.54.R1745

[33] Linke, N. M., Gutierrez, M., Landsman, K. A., Figgatt, C., Debnath, S., Brown, K. R. and Monroe, C., *Fault-tolerant quantum error detection*, 2017. Sci. Adv. 2017. DOI: https://doi.org/10.1126/sciadv.1701074

[34] Grassl, M., Thomas Beth, T. and Pellizzari, T., *Codes for the Quantum Erasure Channel*, 1997. Phys. Rev. A, vol. 56, no. 1, July 1997, pp. 33-38. DOI: https://doi.org/10.1103/PhysRevA.56.33

[35] Leung,D. W., Nielsen, M. A., Chuang, I. L. and Yamamoto Y., *Approximate quantum error correction can lead to better codes*, 1997. Phys.Rev.A56:2567-2573,1997. DOI: https://doi.org/10.1103/PhysRevA.56.2567

[36] Church A., *A Set of Postulates for the Foundation of Logic*, 1932. Annals of Mathematics 33.2

[37] Deutsch, David, *Quantum theory, the Church–Turing principle and the universal quantum computer*, 1985. Proceedings of the Royal Society of London A, 400, 1985 pp. 97–117. https://doi.org/10.1098/rspa.1985.0070

[38] https://newsroom.ibm.com/2022-11-09-IBM-Unveils-400-Qubit-Plus-Quantum-Processor-and-Next-Generation-IBM-Quantum-System-Two

[39] Wootters, W., Zurek, W. *A Single Quantum Cannot be Cloned*, 1982. Nature. 299 (5886): 802–803. DOI:10.1038/299802a0.

[40] Dieks, Dennis *Communication by EPR devices*, 1982. Physics Letters A. 92 (6): 271–272. DOI:10.1016/0375-9601(82)90084-6

[41] Schumacher,B. 1995, *Quantum Coding* in Phys. Rev. A 51, 2738 – Published 1 April 1995.
DOI: https://doi.org/10.1103/PhysRevA.51.2738

[42] Liang, Y-C. et al., 2018, *Quantum fidelity measures for mixed states* in Rep. Prog. Phys. 82, 076001 (2019)
DOI: https://doi.org/10.1088/1361-6633/ab1ca4

[43] https://errorcorrectionzoo.org/c/commuting_projector reached at 21.06.2023

[44] Hu, J., Liang, Q. and Calderbank, R., *Divisible Codes for Quantum Computation*, 2022. DOI: https://doi.org/10.48550/arXiv.2204.13176

[45] Hu, J., Liang, Q. and Calderbank, R., *Designing the Quantum Channels Induced by Diagonal Gates*, 2021. Quantum 6, 802 (2022) DOI: https://doi.org/10.22331/q-2022-09-08-802

[46] Ward, H. N., *Divisible codes – a survey*, 2001. Serdica Math. J., vol. 27 (4), pp. 263–278, 2001.

[47] Ax, J., *Zeros of polynomials over finite fields*. Amer.J.Math. 86 (1964), 255-261; MR 28#3986

[48] Lindblad, G., 1976. *On the Generators of Quantum Dynamical Semigroups* in Commun. math. Phys. 48, 119–130 (1976)

[49] Ardeh, H.A., Allen, M.S. (2013). Investigating Cases of Jump Phenomenon in a Nonlinear Oscillatory System. In: Kerschen, G., Adams, D., Carrella, A. (eds) Topics in Nonlinear Dynamics, Volume 1. Conference Proceedings of the Society for Experimental Mechanics Series, vol 35. Springer, New York, NY. https://doi.org/10.1007/978-1-4614-6570-6_28

[50] Hastings, M.B., Haah, J., O'Donnell R. (2020). *Fiber Bundle Codes: Breaking the $N^{1/2}polylog(N)$ Barrier for Quantum LDPC Codes*. arXiv:2009.03921 [quant-ph]

[51] https://indico.physik.uni-muenchen.de/event/84/attachments/246/488/S3.HastingsHaah.slides.pdf reached at October 2023

[52] Panteleev, P. and Kalachev, G., 2021. *Quantum LDPC Codes with Almost Linear Minimum Distance*, https://doi.org/10.1109/TIT.2021.3119384

[53] Tillich, J.P. and Zemor, G., 2009. *Quantum LDPC codes with positive rate and minimum distance proportional to $n^{1/2}$* in IEEE Trans. Inform. Theory. Vol. 60, No 2, pp. 1193–1202. 2014

[54] Breuckmann, N. P. and Eberhardt, J. N., 2020. *Balanced Product Quantum Codes* in IEEE Transactions on Information Theory 2021.

[55] Evra, S., Kaufman, T., Zémor G., 2020. *Decodable quantum LDPC codes beyond the $\sqrt{n}$ distance barrier using high dimensional expanders*, arXiv:2004.07935v1.

[56] Fetaya, E., 2011. *Homological error correcting codes and systolic geometry*, Master's thesis, The Hebrew University of Jerusalem (2011), arXiv:1108.2886 [math.DG]

[57] Kitaev, A.Yu., 1997 Russ. Math. Surv. 52 1191
DOI: 10.1070/RM1997v052n06ABEH002155

[58] Pastawski, F., Kay, A., Schuch, N., Cirac I., 2009, *Limitations of Passive Protection of Quantum Information*, arXiv:0911.3843v1 [quant-ph]

[59] Browne, D., *Lectures on Topological Codes and Quantum Computation*, 2014. https://drive.google.com/file/d/1AB0Kd5JV5TzqwvR4VysrJtF3GVin2S4/view reached at 01.10.2023

[60] Freeman, C.D., Herdman, C. M., Gorman, D.J., Whaley, K. B., 2014, *Relaxation dynamics of the toric code in contact with a thermal reservoir: Finite-size scaling in a low temperature regime*, Phys. Rev. B 90, 134302 (2014)

[61] Preskill, J., 1997. *Fault-tolerant quantum computation*, arXiv:quant-ph/9712048

[62] Freedman, M.H., Meyer, D.A., 1998. *Projective plane and planar quantum codes*,
arXiv:quant-ph/9810055v1

[63] Kitaev, A.Yu., 1997. *Fault-tolerant quantum computation by anyons*.
Annals Phys. 303 (2003) 2-30. arXiv:quant-ph/9707021

[64] Breuckmann, N.P., Eberhardt, J.N., 2021. *Quantum Low-Density Parity-Check Codes*, PRX Quantum 2 (4), 040101, 2021
arXiv:2103.06309v2

[65] Bravyi, S., Poulin, D., and Terhal, B., 2010. *Tradeoffs for Reliable Quantum Information Storage in 2D Systems*, Phys. Rev. Lett. 104, 050503 – Published 5 February 2010 DOI:https://doi.org/10.1103/PhysRevLett.104.050503

[66] Breuckmann, N.P., Terhal,B.M., 2015. *Constructions and Noise Threshold of Hyperbolic Surface Codes* in IEEE Transactions on Information Theory, vol. 62, no. 6, pp. 3731-3744, June 2016, doi: 10.1109/TIT.2016.2555700.

[67] N. Delfosse, N., 2013. *Tradeoffs for reliable quantum information storage in surface codes and color codes* IEEE International Symposium on Information Theory - ISIT 2013, pp. 917-921. arXiv:1301.6588v1

[68] Tillich, J.P., Zemor, G., 2009. *Quantum LDPC codes with positive rate and minimum distance proportional to* $n^{1/2}$, IEEE Trans. Inform. Theory. Vol. 60, No 2, pp. 1193–1202. 2014 arXiv:0903.0566v2

[69] Freedman, M.H., Meyer, D.A., and Luo, F., 2002. *Z2-systolic freedom and quantum codes*. Mathematics of quantum computation, Chapman & Hall/CRC, pp. 287–320

[70] Ratcliffe, J.G., 2006, *Foundations of Hyperbolic Manifolds*. Springer New York, no. v. 10.

[71] Macaj, M., Siran, J. and Ipolyiova, M., 2007 *Planar width of regular maps*. in Electronic Notes in Discrete Mathematics, vol. 28, pp. 477–484

[72] Moran, J.F., 1997, *The growth rate and balance of homogeneous tilings in the hyperbolic plane*. in Discrete Mathematics, vol. 173, no. 1, pp. 151–186

[73] Conder, M. and Dobcsanyi, P., 2005. *Applications and adaptations of the low index subgroups procedure*. in Mathematics of computation, vol. 74, no. 249, pp. 485–497, 2005.

[74] Dietze, A. and Schaps, M., 1974, *Determining Subgroups of a Given Finite Index in a Finitely Presented Group* in Canadian Journal of Mathematics , Volume 26 , Issue 4 , August 1974 , pp. 769 - 782 DOI: https://doi.org/10.4153/CJM-1974-072-0

[75] Freedman, M.H., Meyer, D.A. and Luo, F., 2002. *Mathematics of quantum computation*, Chapman Hall/CRC , 287

[76] Fowler, A. G., Mariantoni, M., Martinis, J. M. and Cleland, A. N., 2012. *Surface codes: Towards practical large-scale quantum computation* in Phys. Rev. A 86, 032324 (2012)
DOI: https://doi.org/10.1103/PhysRevA.86.032324

[77] Babar, Z., Botsinis, P., Alanis, D., Ng, S. X. and Hanzo, L., 2015. *Fifteen Years of Quantum LDPC Coding and Improved Decoding Strategies* in IEEE Access, vol. 3, pp. 2492-2519, 2015, doi: 10.1109/ACCESS.2015.2503267.

[78] Gottesman, D., 1996. *A Class of Quantum Error-Correcting Codes Saturating the Quantum Hamming Bound* in Phys.Rev. A54 (1996) 1862
DOI: https://doi.org/10.1103/PhysRevA.54.1862

[79] Steane, A., 1996. *Simple Quantum Error Correcting Codes* in Phys.Rev.A54:4741,1996.
DOI: https://doi.org/10.1103/PhysRevA.54.4741

[80] Calderbank, A. R., Rains, E. M., Shor, P. W. and Sloane, N. J. A., 1996. *Quantum Error Correction and Orthogonal Geometry* in Phys.Rev.Lett.78:405-408,1997.
DOI: https://doi.org/10.1103/PhysRevLett.78.405

[81] Hastings, M. B., Watson, G. H. and Melko, R. G., 2013. *Self-Correcting Quantum Memories Beyond the Percolation Threshold* in Phys. Rev. Lett. 112, 070501 (2014) arXiv:1309.2680 [cond-mat.str-el]

[82] Brown, B. J., Loss, D., Pachos, J. K., Self, C. N., and Wootton, J. R., 2014. *Quantum memories at finite temperature* in Reviews of Modern Physics, vol. 88, no. 4, 2016. DOI:10.1103/RevModPhys.88.045005.

[83] Kossakowski, A., A. Frigerio, V. Gorini, and M. Verri, 1977. *Quantum detailed balance and KMS condition* in Commun. Math. Phys. 57, 97. DOI: https://doi.org/10.1007/BF01625769

[84] Peierls, R. 1936. *On Ising's model of ferromagnetism* in Mathematical Proceedings of the Cambridge Philosophical Society, 32(3), 477-481. doi:10.1017/S0305004100019174

[85] Alicki, R., M. Horodecki, P. Horodecki, and R. Horodecki, 2010. *On thermal stability of topological qubit in Kitaev's 4D model* in Open Syst. Inf. Dyn. 17, 1 https://doi.org/10.48550/arXiv.0811.0033

[86] Jeongwan Haah, *Local stabilizer codes in three dimensions without string logical operators*, Phys. Rev. A 83, 042330 (2011) arXiv:1101.1962 [quant-ph]

[87] Brown, B. J., A. Al-Shimary, and J. K. Pachos, 2014. *Entropic Barriers for Two-Dimensional Quantum Memories* in Phys. Rev. Lett. 112, 120503 (2014)

[88] Landon-Cardinal, O., and D. Poulin, 2013. *Local Topological Order Inhibits Thermal Stability in 2D* in Phys. Rev. Lett. 110, 090502 (2013)

[89] Bravyi, S., and B. Terhal, 2009. *A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes* in New J. Phys. 11, 043029 (2009).
DOI: 10.1088/1367-2630/11/4/043029

[90] Kitaev, A. Y., 2003. *Fault-tolerant quantum computation by anyons* in Ann. Phys. 303, 2.

[91] Pastawski, F., Clemente, L., and Cirac, J.I., 2010. *Quantum memories based on engineered dissipation* in Phys. Rev. A 83, 012304 (2011)
https://doi.org/10.1103/PhysRevA.83.012304

[92] Brell, C. G., 2014. *A proposal for self-correcting stabilizer quantum memories in 3 dimensions (or slightly less)* in New J. Phys. 18, 013050 (2016) arXiv:1411.7046

[93] Chiaverini, J., Leibfried, D., Schaetz, T. et al. *Realization of quantum error correction* in Nature 432, 602–605 (2004). https://doi.org/10.1038/nature03074

[94] Leghtas, Z., Kirchmair, G., Vlastakis, B., Schoelkopf, R., Devoret, M. and Mirrahimi M., 2012. *Hardware-efficient autonomous quantum error correction* in

Phys. Rev. Lett. 111, 120501 (1013)
https://doi.org/10.1103/PhysRevLett.111.120501

[95] Supplamentary material for [94] reached at 24.11.2023: http://link.aps.org/supplemental/10.1103/PhysRevLett.111.120501

[96] Chesi, S., Loss, D., Bravyi, S. and Terhal, B.M., 2009. *Thermodynamic stability criteria for a quantum memory based on stabilizer and subsystem codes* in New J. Phys. 12, 025013 (2010).
https://doi.org/10.1088/1367-2630/12/2/025013

[97] Nussinov, Z. and Ortiz, G., *Autocorrelations and thermal fragility of anyonic loops in topologically quantum ordered systems* in Phys. Rev. B 77, 064302 (2008).
https://doi.org/10.1103/PhysRevB.77.064302

[98] Chesi, S., Röthlisberger, B. and Loss D., 2009. *Self-Correcting Quantum Memory in a Thermal Environment* in Phys. Rev. A 82, 022305 (2010)
https://doi.org/10.1103/PhysRevA.82.022305

[99] Davies, E.B. *Markovian master equations* in Comm. Math. Phys. 39, 91 (1974)

[100] Cohen-Tannoudji, C., Dupont-Roc, J. and Grynberg, G. *Atom-Photon Interactions: Basic Processes and Applications* (Wiley-Interscience, 1992), 1st ed, p356

[101] Lieu, S., Liu, Y.-J. and Gorshkov A.V., 2022. *Candidate for a passively protected quantum memory in two dimensions*. arXiv:2205.09767v3

[102] Lieu, S., Belyansky, R., Young, J.T., Lundgren, R., Albert, V.V. and Gorshkov, A.V., 2020. *Symmetry breaking and error correction in open quantum systems* in Phys. Rev. Lett. 125, 240405 (2020)
https://doi.org/10.1103/PhysRevLett.125.240405

[103] Buca, B. and Prosen, T., 2012. *A note on symmetry reductions of the Lindblad equation: transport in constrained open spin chains* in New Journal of Physics 14 (2012) 073007 https://doi.org/10.1088/1367-2630/14/7/073007

[104] Campagne-Ibarcq, P., Eickbusch, A., Touzard, S. et al. *Quantum error correction of a qubit encoded in grid states of an oscillator* in Nature 584, 368–372 (2020).
https://doi.org/10.1038/s41586-020-2603-3

[105] Cao, C., Zhang, C., Wu, Z., Grassl, M. and Zeng, B., 2022. *Quantum variational learning for quantum error-correcting codes* in Quantum 6, 828 (2022)
https://doi.org/10.22331/q-2022-10-06-828

[106] Nautrup, H.P., Delfosse, N., Dunjko, V., Briegel, H.J. and Friis, N., 2018. *Optimizing Quantum Error Correction Codes with Reinforcement Learning* in Quantum 3, 215 (2019)
https://doi.org/10.22331/q-2019-12-16-215

[107] Bombin, H. and Martin-Delgado M.A., 2009. *Quantum measurements and gates by code deformation* in J. Phys. A: Math. Theor. 42 (2009) 095302
DOI 10.1088/1751-8113/42/9/095302

[108] Delfosse, N., Iyer, P. and Poulin, D., 2016. *A linear-time benchmarking tool for generalized surface codes*.
https://doi.org/10.48550/arXiv.1611.04256
http://quantum-squab.com/ (reached at 21.11.2023)

[109] Briegel, H.J. and De las Cuevas G., 2011. *Projective simulation for artificial intelligence* in Sci. Rep. 2, 400 (2012)
https://doi.org/10.48550/arXiv.1104.3787

[110] Sutton, R.S. and Barto, A.G. *Reinforcement Learning: An Introduction* (MIT press, Cambridge, 1998).

[111] Makmal, A., Melnikov, A.A., Dunjko, V. and Briegel, H.J. *Meta-learning within Projective Simulation* in IEEE Access 4, 2110 (2016), arXiv:1602.08017.

[112] Zeng, Y., Zhou, Z.-Y., Rinaldi, E., Gneiting, C. and Nori F., 2022. *Approximate Autonomous Quantum Error Correction with Reinforcement Learning* in Phys. Rev. Lett. 131, 050601 (2023)
https://doi.org/10.1103/PhysRevLett.131.050601

[113] Knill, E. and Laflamme R., 1996. *A Theory of Quantum Error-Correcting Codes* in Phys.Rev.Lett.84:2525-2528,2000 https://doi.org/10.1103/PhysRevLett.84.2525

[114] Kiusalaas, J. (2005). *Numerical Methods in Engineering with Python*. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511812217

[115] Mitarai, K., Negoro, M., Kitagawa, M. and Fujii, K., 2018. *Quantum Circuit Learning* in Phys. Rev. A 98, 032309 (2018) https://doi.org/10.1103/PhysRevA.98.032309

[116] Schuld, M., Bergholm, V., Gogolin, C., Izaac, J. and Killoran, N., 2018. *Evaluating analytic gradients on quantum hardware* in Phys. Rev. A 99, 032331 (2019) https://doi.org/10.1103/PhysRevA.99.032331

[117] Powell, M. J. D. *An efficient method for finding the minimum of a function of several variables without calculating derivatives* in The Computer Journal 7, 155–162 (1964) https://doi.org/10.1093/comjnl/7.2.155

[118] Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Klimov O., 2017. *Proximal Policy Optimization Algorithms* https://doi.org/10.48550/arXiv.1707.06347

[119] Huang, HY., Kueng, R. and Preskill, J. *Predicting many properties of a quantum system from very few measurements* in Nat. Phys. 16, 1050–1057 (2020). https://doi.org/10.1038/s41567-020-0932-7