博士論文

Theory of quantum error correction in near-term quantum devices

(近未来で実現される量子デバイスにおける量子誤り訂正の理論)

鈴木　泰成

Abstract

In this thesis, we investigated a theory of quantum error correction (QEC) in near-term quantum devices. Due to the recent development of quantum technologies, QEC with several tens of qubits is expected to be demonstrated in the next few years. On the other hand, since such a large quantum system cannot be tracked with current classical computers, several problems about practical QEC experiments emerge as quantum devices are scaled up. We provided solutions to the following two essential problems among them.

First, we extended a simulatability of quantum error correcting code under coherent noise using matchgate circuits. This extension enables an efficient and accurate evaluation of an effective physical error probability suppressed with QEC under practical errors. Since this probability should be known before we design quantum devices for QEC, this result provides vital information for experiments.

Second, we proposed a general framework to use machine learning as a part of decoding algorithm in QEC. QEC requires a fast and high-performance classical decoder for its implementation. Using machine learning for decoding is one of the most promising solutions to this. However, what part of a decoding algorithm should be delegated to machine learning in order to achieve the near-optimal performance has not been discussed. We constructed a general framework to discuss it, and showed fundamental relations between the decoding problem and the task of machine learning. Furthermore, we proposed a criterion for achieving high performance, and showed specific formulation of tasks of machine learning. Then, we confirmed that the performance of the proposed machine-learning-based decoder is superior to the known decoders in various situations.

Acknowledgement

This thesis is written based on various supports and advice from many people. I would like to thank Prof. Masato Koashi, who has totally supervised my research during undergraduate and graduate school. His thoughtful advice and questions are helpful for me to obtain clear viewpoint of my research. I am frequently impressed by his outstanding perspective of quantum information science, and the discussions with him strongly encourage my interest in physics.

I also would like to thank Assoc. Prof. Keisuke Fujii, who supervised both of the two main results of the thesis. My knowledge about quantum computation is supported by his deep understanding of it. I am very motivated by his strong enthusiasm for realizing quantum computer.

Mr. Amarsanaa Davaasuren is a collaborator of the result about quantum error correction using machine learning discussed in Chapter 4 of this thesis. His many creative ideas about both machine learning and quantum information have amazed me. The result of Chapter 4 is largely supported by his collaboration.

Assist. Prof. Takanori Sugiyama gave me lectures about the latest and essential problems in near-term quantum devices. In particular, his lectures were vital for us to find the research topic of an efficient simulation of quantum circuits of quantum error correction under practical noise models, which is discussed in Chapter 3 of this thesis.

I have been financially supported by Advanced Leading Graduate Course for Photon Science (ALPS). During this program, I have had an opportunity to discuss with Prof. Akira Furusawa. His advice from the standpoint of an experimental researcher helps my practical understanding of experimental systems. His philosophy as a professional researcher shapes my attitude to research.

I also would like to thank the members and alumni of Koashi's group. In particular, I thank Assoc. Prof. Haruka Tanji-Suzuki and Assist. Prof. Toshihiko Sasaki for advice about from fundamental physics to presentation skills. My skills as a researcher are trained through discussions with them. I also thank Dr. Yoshifumi Nakata and Dr. Shun Kawakami for helpful discussions about fundamental quantum information. I also thank Mr. Takaya Matsuura for giving me valuable advice about quantum field theory.

Contents

Chapter 1

Introduction

1.1 Overview

Quantum computation is a framework to utilize properties of quantum mechanics for computation [1]. Quantum computer is believed to be capable of faster processing than classical ones in a variety of tasks, such as performing integer factoring [2], searching in unstructured database [3], solving a linear system [4], simulating a quantum dynamics [5, 6], studying quantum chemistry with a variational scheme [7], solving semi-definite programming [8], analyzing a large data set [9, 10], and accelerating machine learning [11]. Furthermore, scalable quantum technologies are essential for extending applicable scopes of other quantum applications, such as quantum cryptography [12] and quantum metrology [13]. When we try to build quantum computer in a scalable manner, the most difficult obstacle is unavoidable physical errors caused by noise from environment and by imperfection in experimental controls. If we use noisy quantum circuits without any technique of an error suppression, an applicable size of computational tasks is limited by an achievable physical error probability. Quantum error correction (QEC) [14] provides essential solutions to this problem.

We can construct clean logical qubits from several noisy physical qubits with QEC. A typical scenario of QEC under a probabilistic Pauli noise model is as follows. We use a 2^k-dimensional subspace in a noisy n-qubit system as a space of k virtual qubits. This subspace is called logical space, and virtual k qubits in the logical space are called logical qubits. We encode a k-qubit state in the 2^k-dimensional logical space. While we keep the k-qubit state in the 2^k-dimensional logical space of the n-qubit system, physical errors occur on each qubit with a probability p. In order to check where physical errors occurred, we measure an observable called a syndrome, in such a way that the state of the k logical qubits should not be destroyed. Then, we decode the k-qubit state by performing a recovery operation, which is determined with the obtained syndrome. When we can assume a probabilistic Pauli noise model, correct recovery operations recover an original quantum state of the k qubits, and incorrect ones lead to an orthogonal

quantum state to the original state. We call a failure probability of the recovery as a logical error probability p_{L}. In order to build scalable quantum computer, the logical error probability p_{L} should be decreased to an arbitrary small value by increasing the number of the physical qubits n. It is known that there is a value of the physical error probability called an error threshold p_{th}. If the physical error probability p is smaller than the threshold value p_{th}, we can achieve an arbitrary small logical error probability p_{L} by increasing the number of the qubits n [15–17]. Thus, in order to demonstrate QEC, it is required to construct quantum devices with their physical error probability smaller than p_{th}. At the beginning of the theory of QEC, known threshold values were about 10^{-5} [1]. On the other hand, an experimentally achievable physical error probability was far worse than the threshold value, and the number of implementable qubits was not enough for QEC. There was a large gap between theoretical requirements for QEC and achievable quantum technologies. Thanks to recent theoretical and experimental developments, this gap has become smaller, and it is expected that quantum devices with several tens of qubits with a physical error probability smaller than the threshold value are achieved in the next few years. One of the most vital progresses in theoretical QEC is the discovery of topological stabilizer codes [18]. Topological stabilizer codes are a family of quantum error correcting codes, and have a high threshold value about 10^{-2} [19]. Furthermore, topological stabilizer codes have several preferable properties for realizing QEC in experiments [18, 20, 21]. In the experimental aspect, various quantum devices have been scaled up with keeping long coherence time and high controllability [22–25]. Thus, QEC is expected to be experimentally demonstrated below the threshold value in the next few years with these near-term quantum devices.

On the other hand, when we start to build quantum devices with more than about 50 qubits, we encounter several new problems. In particular, there are two essential problems in QEC experiments with near-term quantum devices. One is computational hardness of accurate evaluation of topological stabilizer codes under practical noise models. Though the performance of the topological stabilizer codes such as the logical error probability and the threshold value under a practical noise model is essential for designing quantum devices, we cannot efficiently and accurately know these values with classical computation in general. The other problem is computational hardness of optimal decoding in QEC. Since we cannot perform optimal decoding except few specific cases [26, 27], we should construct a fast, near-optimal, and versatile decoder for QEC. In order to demonstrate QEC in experiments, these problems should be solved.

In this thesis, we focus on both of the two above problems. For the problem of the performance evaluation, since the performances of topological stabilizer codes are essential, they have been evaluated in various settings [28–45]. The performances of topological stabilizer codes can be accurately evaluated if we can simulate quantum circuits of quantum error correcting codes under a practical noise model with classical computers. On the other hand, when the number of the

physical qubits n is larger than about 50, we cannot store a 2^n-dimensional complex vector of an n qubit state with any existing classical computer. Therefore, we cannot simulate the whole quantum system with a straightforward method. According to the Gottesman-Knill theorem [46], any quantum circuit with n qubits which consists of Clifford operations and Pauli measurements can be simulated with a computational cost that increases polynomially to n. In the case of topological stabilizer codes, measurements of a syndrome can be represented with Pauli measurements. Various practical noise models such as dephasing, depolarizing, and amplitude damping noise models can be represented or approximated with Clifford operations. Thus, we can evaluate the performance of topological stabilizer codes under these noise models with a practical computational cost of classical computers. However, though coherent noise is unavoidably caused in experiments, for example, by over-rotation due to imperfect calibration of quantum controls, it cannot be approximated with Clifford operations and Pauli measurements. Therefore, novel schemes for evaluating topological stabilizer codes under coherent noise models are demanded. In this thesis, we propose an efficient scheme for simulating quantum error correcting codes under coherent noise. This scheme enables an efficient evaluation of vital properties of quantum error correcting codes, such as the logical error probability and the threshold value. The key idea of this scheme is the use of an efficiently simulatable class of quantum circuits called matchgate circuits [47–50]. An advantage of the framework of matchgate circuits compared with that of Clifford circuits is that a gate set of matchgate circuits contains continuously parametrized gate operations, such as the Pauli-X rotation with an arbitrary angle. We expect this property is used for simulating coherent noise. On the other hand, a drawback of the framework of matchgate circuits is that this framework cannot treat even single Pauli-Z operations and single qubit measurements with Pauli-Z basis. Usually, quantum circuits of quantum error correcting codes are represented with Clifford operations and Pauli measurements. Therefore, it is nontrivial whether we can represent quantum circuits of stabilizer codes only with an allowed gate set of matchgate circuits. By solving this discrepancy, we will eventually show that the surface code, which is one of the most promising topological stabilizer codes, under coherent noise can be represented as a matchgate circuit. This representation enables an efficient and accurate evaluation of the property of the surface code under coherent noise, which is immediately helpful for designs of quantum devices. Since we cannot define an error probability of coherent noise, we characterize a coherent noise map with two parameters, a degree of coherence $c \in [0, 1]$ and an effective physical error probability p, so that p coincides with the physical error probability when the noise is incoherent, namely, $c = 0$. Then, we investigate the dependency of the properties of quantum error correcting codes such as the logical error probability p_{L} and the threshold value p_{th} on the degree of coherence c. We will find that the increase of coherence c worsens these properties. We will also discuss the mechanism of this degradation by comparing the accurate results with a simple approximate model.

As for the problem of decoding algorithm, since the performance of quantum error correcting codes depends on the performance of a chosen decoder for determining a recovery operation from an obtained syndrome, various near-optimal decoders for specific topological stabilizer codes and noise models have been proposed [21, 27, 51–53]. However, a decoder which is fast, achieves a small logical error probability, and is applicable to various topological stabilizer codes under various noise models is still lacking. In this thesis, we propose a general framework for constructing such a fast, reliable, and versatile decoder based on machine learning. Machine learning provides generic instructions to construct a function which predicts behavior of an unknown system using a previously known data set. This process for constructing a prediction function is called training. Machine learning can be used in various situations, and the constructed prediction function can be computed fast. Therefore, by using machine learning as a part of decoding algorithm, we expect that a fast and versatile machine-learning-based decoder can be constructed. When we want a machine-learning-based decoder to be reliable, there are several important factors which determine the performance of the decoder. The most important factor among them is what part of decoding algorithm to be delegated to machine learning. Though several constructions of machine-learning-based decoders were proposed [54–58], this point has not been studied. In order to clarify this point, we propose a framework for constructing a near-optimal decoder using machine learning as a part. In this framework, we can treat all the existing decoding methods based on machine learning as specific cases. Based on this framework, we will clarify what part of the decoding algorithm should be delegated to the machine learning in order to guarantee that the whole decoding algorithm becomes optimal in any noise model when a training is ideally performed. Furthermore, we propose a criterion which should be optimized to construct a near-optimal decoding algorithm when we use machine learning as a part. We also show specific constructions for various topological codes which optimize the criterion. We will numerically show that the performance of the proposed machine-learning-based decoder is superior to all the existing machine-learning-based decoders. We also confirm that the performance of the proposed decoder is close to the optimal performance not only in the surface code but also in color codes, which are another family of topological stabilizer codes.

1.2 Construction of the thesis

In Chapter 2, we provide a preliminary of this thesis. We first clarify the notations of this thesis. Then, we review theories of quantum error correction, simulatability of quantum circuits, and supervised machine learning. In Chapter 3, we discuss the first topic, an efficient simulation of quantum error correcting codes under coherent noise. By using matchgate circuits, we show that the surface code under coherent noise can be efficiently simulated. Then, we show numerical results, and we discuss

the effect of the coherence in noise on the threshold value. In Chapter 4, we discuss the second topic, a general framework for constructing decoding algorithm using machine learning as a part. We discuss fundamental relations between the decoding problems in QEC and the framework of supervised machine learning. We propose a criterion to optimize for constructing a machine-learning-based decoder. Then, we provide numerical results to show that the proposed decoder is superior to the known decoders. In Chapter 5, we summarize the thesis, and describe an outlook.

Chapter 2

Preliminary

2.1 Notations

In this section, we introduce notations used in the thesis to clarify the definitions of symbols.

2.1.1 Basic notations

Landau notation

We use Landau notation O, Ω, and Θ for representing an order of a function to variables. These notations represent upper-bound, lower-bound, and exact order of a function, respectively. Let n be a variable. For given positive functions $f(n)$ and $g(n)$, if there are constants ϵ and N such that $\frac{f(n)}{g(n)} < \epsilon$ for an arbitrary $n > N$, we say $f(n)$ is $O(g(n))$. For given positive functions $f(n)$ and $g(n)$, if there are constants ϵ and N such that $\frac{f(n)}{g(n)} > \epsilon$ for an arbitrary $n > N$, we say $f(n)$ is $\Omega(g(n))$. When $f(n)$ is both $O(g(n))$ and $\Omega(g(n))$, we say $f(n)$ is $\Theta(g(n))$.

In particular, at several evaluations in this thesis, we are interested in whether there exists a polynomial function which can upper-bound a growing speed of a given function in terms of a variable. This is because a scheme which requires an exponential amount of time or memory to a problem size is not practical. We say a function $f(n_0, n_1, \cdots)$ is $\mathrm{poly}(n_0, n_1, \cdots)$ to mean that there exists a polynomial function $g(n_0, n_1, \cdots)$ such that $f(n_0, n_1, \cdots)$ is $O(g(n_0, n_1, \cdots))$. When we evaluate a computational cost of an algorithm, we say an algorithm is efficient if it requires at most polynomial time.

Indexing and counting

When we mention about an indexed object, such as a vector, an ordered set, and a matrix, we always use 0-indexed representation. For example, n qubits are counted from the 0-th qubit to the $(n-1)$-th qubit.

We represent a vector as a bold lowercase character such as $\boldsymbol{v}$. When we introduce $\boldsymbol{v}$ as a vector, we assume that the i-th element of $\boldsymbol{v}$ is implicitly defined as v_i. We represent a matrix as an uppercase character such as M. We denote the element at the i-th row and the j-th column as M_{ij} We denote the i-th row vector of a matrix M as $(M)_i$. We denote the length of a vector $\boldsymbol{v}$ as $|\boldsymbol{v}|$. We denote the number of elements in a set $\mathcal{A}$ as $|\mathcal{A}|$.

2.1.2 Quantum states and operations

Qubits and quantum states

We use a two-level quantum system as an element of quantum computation called a qubit. Any quantum state of n qubits is represented as a linear operator ρ acting on a tensor product of n two-dimensional Hilbert spaces $\mathcal{H}^{\otimes n}$ such that ρ satisfies $\mathrm{Tr}(\rho) = 1$ and $\rho \geq 0$. This linear operator ρ is called a density operator. If the rank of a density operator ρ is unity, the corresponding quantum state is called pure. Any pure quantum state ρ can be denoted with a 2^n-dimensional vector $|\psi\rangle \in \mathcal{H}^{\otimes n}$ such that $\rho = |\psi\rangle\langle\psi|$, where $|\psi\rangle$ is an eigenvector corresponding to the non-zero eigenvalue of ρ, and $\langle\psi|$ is the Hermitian conjugate of $|\psi\rangle$. Note that though any density operator satisfies the normalization condition $\mathrm{Tr}(\rho) = 1$, we sometimes use unnormalized ($\mathrm{Tr}(\rho) < 1$) density operators for theoretical treatments in this thesis.

Pauli operators and computational basis

We denote the identity operator as I, and the Pauli operators as X, Y, and Z, where

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{2.1}$$

We denote a group which consists of tensor products of n Pauli operators with coefficients $\{\pm 1, \pm i\}$ as

$$\mathcal{P}_n := \{\pm 1, \pm i\} \times \{I, X, Y, Z\}^{\otimes n}. \tag{2.2}$$

Since these Pauli operators are considered as operations on an n-qubit systems, we call $\mathcal{P}_n$ as an n-qubit Pauli group, and call elements in $\mathcal{P}_n$ as n-qubit Pauli operators. We define $P_i = I^{\otimes i} \otimes P \otimes I^{\otimes(n-i-1)}$ ($P \in \{X, Y, Z\}$) as an n-qubit Pauli operator which acts on the i-th qubit as P, acts on the other qubits as an identity operator, and has $+1$ coefficient.

The 2^n simultaneous eigenstates of n Pauli operators $\{Z_0, Z_1, \cdots, Z_{n-1}\}$ are a complete orthonormal set of the 2^n-dimensional Hilbert space of the n-qubit system. We call this set as computational basis. We denote an eigenstate which has an eigenvalue $z_i \in \{\pm 1\}$ for Z_i ($i = 0, \ldots, n-1$) as $|\boldsymbol{x}\rangle$, where $\boldsymbol{x} \in \{0, 1\}^n$ is the binary vector defined as $x_i = (1 + z_i)/2$. We also denote the state $|\boldsymbol{x}\rangle$ as $|x\rangle$, where $x = \sum_{i=0}^{n-1} 2^i x_i$.

Operations

An arbitrary physical operation on a quantum system can be represented as a completely-positive trace-preserving (CPTP) map. We say a linear map $\mathcal{E}$ is completely-positive (CP) if $\mathcal{E}$ satisfies

$$(\mathrm{Id} \otimes \mathcal{E})(\rho) \geq 0 \tag{2.3}$$

for an arbitrary density operator ρ, where Id is an identity channel. We say a map $\mathcal{E}$ is trace-preserving (TP) if it satisfies

$$\mathrm{Tr}(\mathcal{E}(\rho)) = \mathrm{Tr}(\rho) \tag{2.4}$$

for an arbitrary density operator ρ. When a given map $\mathcal{E}$ is CP and TP, we say $\mathcal{E}$ is CPTP. It is known that any CP map on an n-qubit space has a representation of

$$\mathcal{E}(\rho) = \sum_{i=0}^{L-1} K_i \rho K_i^\dagger, \tag{2.5}$$

where $1 \leq L \leq 4^n$, and $\sum_{i=0}^{L-1} K_i^\dagger K_i \leq I$. This form is called Kraus representation of $\mathcal{E}$, and an operator K_i is called a Kraus operator. We call the minimum number of Kraus operators L as the Kraus rank. Any CP map with unit Kraus rank takes an arbitrary pure state to another pure state. An operator K which satisfies $K^\dagger K = I$ is called a unitary operator. Note that all the Pauli operators are unitary.

The information of a CP map $\mathcal{E}$ can be stored as a channel state. A channel state is defined as follows. We consider a doubled Hilbert space $\mathcal{H}^{\otimes n} \otimes \mathcal{H}^{\otimes n}$, where we denote the first Hilbert space as $\mathcal{H}_A$, and the other as $\mathcal{H}_B$. We define a $2n$-qubit maximally entangled state $|\psi_M\rangle$ in the space $\mathcal{H}_A \otimes \mathcal{H}_B$ as

$$|\psi_M\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle_A \otimes |i\rangle_B \,, \tag{2.6}$$

where $\{|i\rangle_A\}$ and $\{|i\rangle_A\}$ are the computational basis of the spaces of $\mathcal{H}_A$ and $\mathcal{H}_B$, respectively. The channel state $\rho_\mathcal{E}$ of a CP map $\mathcal{E}$ is given by

$$\rho_\mathcal{E} = (\mathcal{E} \otimes \mathrm{Id})(\rho_M), \tag{2.7}$$

where $\rho_M = |\psi_M\rangle \langle\psi_M|$. Using the channel state, for an arbitrary given quantum state ρ, we can obtain $\mathcal{E}(\rho)$ as

$$\mathcal{E}(\rho) = 2^n \mathrm{Tr}_B((I \otimes \rho^\mathrm{T})\rho_\mathcal{E}), \tag{2.8}$$

where Tr_B is a partial trace of the space of $\mathcal{H}_B$.

Measurements

An arbitrary measurement on a quantum system can be formulated as an instrument. Instrument is described with a set of indexed CP maps $\{\mathcal{E}_i\}$ which satisfies $\sum_i \mathrm{Tr}(\mathcal{E}_i(\rho)) = 1$ for an arbitrary normalized density operator ρ. When we measure a quantum state ρ with an instrument $\{\mathcal{E}_i\}$, we obtain a classical output i with the probability $\mathrm{Tr}(\mathcal{E}_i(\rho))$, and the state after the measurement is $\frac{\mathcal{E}_i(\rho)}{\mathrm{Tr}(\mathcal{E}_i(\rho))}$.

An operator K which satisfies $K = K^\dagger$ and $K^2 = K$ is called a projection operator. When every CP map $\mathcal{E}_i$ of an instrument has unit Kraus rank and its Kraus operator is a projection operator, the measurement with the instruments is called a projection measurement.

Properties of quantum operations

We define a weight $w(P)$ of an n-qubit unitary operator P as a number of qubits to which P applies non-trivially. Equivalently, we also say that a unitary operator P with a weight $w(P)$ as a $w(P)$-local operator.

Any pair of Pauli operators either commute or anti-commute. We define a commutation relation function c for two Pauli operators as follows.

$$c(P, P') = \begin{cases} 0 & PP' = P'P \\ 1 & PP' = -P'P \end{cases} \tag{2.9}$$

Fidelity of states and maps

Fidelity is measure of a distance between two normalized quantum states ρ and ρ', which is defined as follows.

$$F(\rho, \rho') = \left(\mathrm{Tr}\left(\sqrt{\sqrt{\rho}\rho'\sqrt{\rho}}\right)\right)^2 \tag{2.10}$$

The fidelity becomes unity if two states are equivalent, and becomes zero if two states are orthogonal.

In the context of quantum error correction, we are interested in how close the resultant state of quantum error correction is to an input state. We measure it with an entanglement fidelity defined as follows. Let $\mathcal{E}$ be a CPTP map which takes a n-qubit input state to another n-qubit output state. The entanglement fidelity of the CPTP map $\mathcal{E}$ is defined by the fidelity between the channel state of $\mathcal{E}$ and the channel state of the identity map, i.e.

$$F(\mathcal{E}) := F(\rho_M, (\mathrm{Id} \otimes \mathcal{E})(\rho_M)), \tag{2.11}$$

where ρ_M is a $2n$-qubit maximally entangled state defined in Eq. (2.6).

2.1.3 Binary variables, operations, and representations

Binary variables and operators

The number of one-elements in a binary vector $\boldsymbol{v}$ and a matrix M is represented as $h(\boldsymbol{v})$ or $h(M)$, respectively, which is called a hamming weight.

In the coding theory, we frequently use calculus in GF(2). In GF(2), the addition of binary values v and u is performed with modulo 2, i.e.,

$$v \oplus u := v + u \mod 2. \tag{2.12}$$

The calculation of binary vectors and binary matrices are defined in GF(2).

The symplectic product $\odot$ is defined for two vectors which have the same and even length. Let $\boldsymbol{v}$ and $\boldsymbol{u}$ be binary vectors such that $|\boldsymbol{v}| = |\boldsymbol{u}| = 2n$. The symplectic product is given by

$$\boldsymbol{v} \odot \boldsymbol{u} := \oplus_{i=0}^{n-1}(v_i u_{i+n} \oplus v_{i+n} u_i). \tag{2.13}$$

Suppose that $\boldsymbol{v}$ and $\boldsymbol{u}$ are column vectors. It is convenient to denote the symplectic product as

$$\boldsymbol{v} \odot \boldsymbol{u} = \boldsymbol{v}^{\mathrm{T}} \Lambda \boldsymbol{u}, \tag{2.14}$$

where Λ is a $2n \times 2n$ matrix defined as

$$\Lambda = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix}. \tag{2.15}$$

Note that 0 and I are $n \times n$ submatrices.

In this thesis, a set of vectors denoted by $\{0,1\}^n$ represents a set of row vectors. We perform operations such as additions and multiplications between real-valued object and binary object. In such a case, we assume that a binary object is implicitly interpreted as a real-valued object, i.e., $\boldsymbol{v} + \boldsymbol{v}' \in \mathbb{R}^n$ if $\boldsymbol{v} \in \mathbb{R}^n$ and $\boldsymbol{v}' \in \{0,1\}^n$.

Masked binary vector

When we consider a marginal probability, it is convenient to define a masked binary vector $\tilde{\boldsymbol{v}} \in \{0,1,*\}^n$. The character $*$ means a wild card, i.e., $*$ may be either 0 or 1. For a given binary vector $\boldsymbol{v} = (v_0, \ldots, v_{n-1}) \in \{0,1\}^n$ and a masked binary vector $\tilde{\boldsymbol{v}} = (\tilde{v}_0, \ldots, \tilde{v}_{n-1}) \in \{0,1,*\}^n$, we say $\boldsymbol{v} \sim \tilde{\boldsymbol{v}}$ if and only if $v_i = \tilde{v}_i$ is satisfied for all the indices i such that $\tilde{v}_i \in \{0,1\}$.

Binary representation of Pauli operators

When we are not interested in the global phase of Pauli operators, it is convenient to represent calculation between Pauli operators with the calculation of binary vectors in GF(2). We represent Pauli operators on the i-th qubit as follows.

$$I_i \mapsto \sigma_{00}^{(i)}, X_i \mapsto \sigma_{01}^{(i)}, Y_i \mapsto \sigma_{10}^{(i)}, Z_i \mapsto \sigma_{11}^{(i)} \tag{2.16}$$

Then, we define a map $b : \mathcal{P}_n \to \{0,1\}^{2n}$ and a map $B : \{0,1\}^{2n} \to \{I, X, Y, Z\}^{\otimes n}$ as

$$b(\alpha \otimes_{i=0}^{n-1} \sigma_{v_i v_{i+n}}^{(i)}) = \boldsymbol{v}, \tag{2.17}$$

$$B(\boldsymbol{v}) = \otimes_{i=0}^{n-1} \sigma_{v_i v_{i+n}}^{(i)}, \tag{2.18}$$

where $\alpha \in \{\pm 1, \pm i\}$.

We can represent calculations of Pauli operators as calculations in GF(2) as follows. Let P and P' be n-qubit Pauli operators. We see P is equivalent to P' up to a global phase if and only if $b(P) = b(P')$. The products of Pauli operators can be represented as

$$b(PP') = b(P) \oplus b(P'). \tag{2.19}$$

The commutation relation function $c(P, P')$ can be represented as a symplectic product of binary vectors, i.e.,

$$c(P, P') = b(P) \odot b(P'). \tag{2.20}$$

A weight of a Pauli operator P can be also defined as a function of a binary vector $\boldsymbol{v} = b(P)$ as

$$w(\boldsymbol{v}) = \sum_{i=0}^{n-1} (v_i \oplus v_{i+n} \oplus v_i v_{i+n}). \tag{2.21}$$

Note that the summation is performed without modulo 2.

2.2 Quantum error correction

When we experimentally perform a computational task, there are unavoidable errors on physical qubits due to experimental imperfections and environmental noises. We can construct logical qubits with a small effective error probability from a larger number of physical qubits. This process is called quantum error correction (QEC). QEC is a vital technique to construct a fault-tolerant quantum computer. In this section, we review the known facts about the theory of QEC.

2.2.1 Quantum error correction with stabilizer code

QEC provides a method to construct a logical qubit with an arbitrary small effective error probability using a number of physical qubits with a sufficiently small error probability [15–17]. When we use n physical qubits for constructing k ($k \leq n$) logical qubits, we achieve this by encoding k logical qubits into a 2^k-dimensional subspace in the 2^n-dimensional Hilbert space of the n qubits. We call this encoded 2^k subspace as the logical space, and a quantum state in the logical space as a logical state.

Stabilizer formalism

Stabilizer formalism [59] provides efficient representation of a 2^k-dimensional subspace. A quantum error correcting code which is constructed with stabilizer formalism is called a stabilizer code. In QEC with a stabilizer code, the logical space is specified using $(n-k)$ of n-qubit Pauli operators, which is called a stabilizer generator.

We first define a stabilizer generator as follows.

Definition 2.2.1. (Stabilizer generator) We say a set $\mathcal{S} \subset \mathcal{P}_n$ is a stabilizer generator if $\mathcal{S}$ satisfies all of the following properties.

- The number of elements in the group generated from $\mathcal{S}$ is $2^{|\mathcal{S}|}$.
- All the elements in $\mathcal{S}$ commute.
- The negative identity $-I$ is not in the group generated from $\mathcal{S}$.

A given stabilizer generator represents a $2^{n-|\mathcal{S}|}$-dimensional subspace spanned by a set of quantum states $\{|\psi\rangle\}$, where $S|\psi\rangle = |\psi\rangle$ for all $S \in \mathcal{S}$. We call a subgroup of the n-qubit Pauli group generated from $\mathcal{S}$ as a stabilizer group $\langle\mathcal{S}\rangle$, and call an element of the stabilizer group as a stabilizer operator. When we specify a 2^k-dimensional subspace, we choose a stabilizer generator $\mathcal{S}$ such that $|\mathcal{S}| = n-k$. In this case, there are 2^{n-k} stabilizer operators.

Quantum error correction with stabilizer codes

Stabilizer code is a family of quantum error correcting codes which specifies a logical space with stabilizer formalism. For a given stabilizer generator $\mathcal{S} = \{S_0, \ldots, S_{n-k-1}\}$, we consider a set of projection operators

$$\Pi_{\mathcal{S}}(\boldsymbol{s}) \;:=\; \prod_{i=0}^{n-k-1} \frac{I + (-1)^{s_i} S_i}{2}, \tag{2.22}$$

where $\boldsymbol{s}^{\mathrm{T}} \in \{0,1\}^{n-k}$. The operator $\Pi_{\mathcal{S}}(0)$ is a projection operator to the logical space, and $\Pi_{\mathcal{S}}(\boldsymbol{s})$ with $\boldsymbol{s} \neq 0$ is that to the subspace orthogonal to the logical space. A set of projection maps $\{\mathcal{E}_{\boldsymbol{s}}\}$ such that

$$\mathcal{E}_{\boldsymbol{s}}(\rho) = \Pi_{\mathcal{S}}(\boldsymbol{s}) \rho \Pi_{\mathcal{S}}(\boldsymbol{s}), \tag{2.23}$$

is considered as a measurement, and a Kraus rank of each map is unity. We call this measurement as a stabilizer measurement, and call the $(n-k)$-bit outcome of the stabilizer measurement $\boldsymbol{s}$ as a syndrome. A stabilizer measurement is a non-destructive measurement for any quantum state in the logical space. After the stabilizer measurement, we choose a CPTP map $\mathcal{R}_{\boldsymbol{s}}$ as a recovery operation,

based on the obtained syndrome $\boldsymbol{s}$. We call an estimator of a recovery operator from $\boldsymbol{s}$ as a decoder.

We call a CPTP map on a k-qubit space from encoding to decoding in QEC as a logical channel. We use an entanglement fildelity of the logical channel as performance of QEC. To define a logical channel, we show a typical scenario of QEC. We consider a case when we encode information of k-qubit state $\rho_{\rm init}^k$ into n physical qubits.

1. We choose stabilizer generator $\mathcal{S}$ such that $|\mathcal{S}| = n-k$. We choose a decoder to determine a recovery operation $\mathcal{R}_{\boldsymbol{s}}$ based on a syndrome $\boldsymbol{s}$.

2. We encode the state $\rho_{\rm init}^k$ to the 2^k-dimensional logical space specified by $\mathcal{S}$ in the n physical qubits. Then we obtain a quantum state of the n physical qubits $\rho_{\rm init}$.

3. We keep the quantum state $\rho_{\rm init}$ in a while.

4. We perform a syndrome measurement, and we obtain an $(n-k)$-bit syndrome $\boldsymbol{s}$. We apply a recovery operation $\mathcal{R}_{\boldsymbol{s}}$ determined with a decoder from $\boldsymbol{s}$ on the physical qubits. Then, we obtain a resultant state $\rho_{\rm final}$.

We consider the whole above process, state preparation, repetitive syndrome measurements, and a recovery operation, as a CPTP map $\mathcal{C}$ from the initial state $\rho_{\rm init}^k$ to the logical space of the final state $\rho_{\rm final}$. We call this CPTP map $\mathcal{C}$ as the logical channel. Since we expect that QEC enables us to store any initial logical state, we use an entanglement fidelity of the logical channel $\mathcal{C}$ as a performance of QEC. If the entanglement fidelity of the logical channel is unity, the logical channel is equivalent to the identity channel.

Quantum error correction with stabilizer code under Pauli noise

In the discussion below, we employ the following assumptions.

- State preparation, stabilizer measurements, and a recovery operation are performed without noise.

- We assume that a Pauli error occurs on each physical qubit independently. Thus, a noise map $\mathcal{E}$ can be represented as $\mathcal{E}_{\rm noise}(\rho) = \sum_{P\in\mathcal{P}_n} p_P P\rho P^\dagger$ with a probability distribution $\{p_P\}$. We also assume that we know the probability distribution $\{p_P\}$.

- We do not care about time efficiency in determining a recovery operation from $\boldsymbol{s}$.

- The recovery operation is a Pauli operation, i.e., $\mathcal{R}_{\boldsymbol{s}}(\rho) = R(\boldsymbol{s})\rho R(\boldsymbol{s})^\dagger$ for $R(\boldsymbol{s}) \in \mathcal{P}_n$.

We further discuss properties of QEC with these assumptions.

We first consider detectable and undetectable Pauli errors. We say an error $P \in \mathcal{P}_n$ is detectable if we obtain a non-zero syndrome with the stabilizer measurement. We see a Pauli error is undetectable if and only if for any $S \in \mathcal{S}$, there exists a Pauli operator $S' \in \langle \mathcal{S} \rangle$ such that

$$PS = S'P. \tag{2.24}$$

Thus, we obtain a set of undetectable operations as follows.

$$\mathcal{N}(\mathcal{S}) := \{P | P \in \mathcal{P}_n, \forall S \in \langle \mathcal{S} \rangle, PSP^\dagger \in \langle \mathcal{S} \rangle\}, \tag{2.25}$$

which is called the normalizer of $\langle \mathcal{S} \rangle$. Since any Pauli operator commutes or anti-commutes with another Pauli operator, a set of undetectable Pauli errors can be rewritten as

$$\mathcal{C}(\mathcal{S}) := \{P | P \in \mathcal{P}_n, \forall S \in \langle \mathcal{S} \rangle, PSP^\dagger = S\}, \tag{2.26}$$

which is called the centralizer of $\langle \mathcal{S} \rangle$. Among undetectable Pauli operators, operators in

$$\mathcal{L}_I := \{\pm 1, \pm i\} \times \langle S \rangle \tag{2.27}$$

do not change any logical state since the logical state is an eigenstate of the operators. A set of undetectable and non-trivial operators on the logical space is then given by

$$\mathcal{L}_P := \mathcal{N}(\mathcal{S}) \setminus \mathcal{L}_I. \tag{2.28}$$

When we can assume that noise occurs on each qubit independently, the probability with which a Pauli error P occurs on the quantum system is decreased exponentially to its weight $w(P)$. In order to minimize the probability with which one of the most probable physical errors in $\mathcal{L}_P$ occurs, the smallest weight of undetectable and non-trivial Pauli operators

$$d := \min_{P \in \mathcal{L}_P} w(P) \tag{2.29}$$

should be large. The value d is called the distance of the stabilizer code. We denote a quantum code which encodes k logical qubits into n physical qubits with a distance d as $[[n, k, d]]$ code.

Next, we discuss the definition of a logical error probability p_{L}. Suppose that a Pauli error $P \in \mathcal{P}_n$ occurred on the physical qubits, and then the syndrome measurement was performed. We denote each element of $\mathcal{S}$ as

$$\mathcal{S} \;=\; \{S_0, \ldots S_{n-k-1}\}. \tag{2.30}$$

The outcome $\boldsymbol{s}$ can be represented as a binary vector

$$\boldsymbol{s}(P) := (c(P, S_0), \ldots, c(P, S_{n-k-1}))^{\mathrm{T}}, \tag{2.31}$$

since

$$\Pi_{\mathcal{S}}(\boldsymbol{s})P \;\; = \;\; P\Pi_{\mathcal{S}}(\boldsymbol{s}(P) \oplus \boldsymbol{s}) \tag{2.32}$$

for $\boldsymbol{s}^{\mathrm{T}} \in \{0,1\}^{n-k}$, and thus we have

$$\mathrm{Tr}(\mathcal{E}_{\boldsymbol{s}(P)}(P\rho_{\mathrm{init}}P)) \;\; = \;\; 1. \tag{2.33}$$

This means we obtain the syndrome $\boldsymbol{s}(P)$ with the unity probability. The final state is written as

$$\rho_{\mathrm{final}} = R(\boldsymbol{s}(P))P\rho_{\mathrm{init}}PR(\boldsymbol{s}(P)), \tag{2.34}$$

where $R(\boldsymbol{s}(P)) \in \mathcal{P}_n$ is a recovery Pauli operation. Obviously, any correct recovery operator should satisfy $\boldsymbol{s}(R(\boldsymbol{s}(P))P) = 0$. Thus, we only consider a decoder which outputs $R(\boldsymbol{s}(P))$ satisfying

$$\boldsymbol{s}(R(\boldsymbol{s}(P))P) = 0. \tag{2.35}$$

In this case, the resultant state ρ_{final} is at least in the logical space, namely, $R(\boldsymbol{s}(P))P \in \mathcal{N}(\mathcal{S})$. If a recovery operator $R(\boldsymbol{s}(P))$ such that $R(\boldsymbol{s}(P))P \in \mathcal{L}_P$ is chosen, the entanglement fidelity of the logical channel $\mathcal{C}$ is zero, and the entanglement fidelity becomes unity if $R(\boldsymbol{s}(P))P \in \mathcal{L}_I$. The logical error probability p_{L} is defined by the probability with which a recovery operator such that $R(\boldsymbol{s}(P))P \in \mathcal{L}_P$ is chosen, which can be written as

$$p_{\mathrm{L}} := 1 - F(\mathcal{C}) = \mathrm{Pr}_{P\sim\{p_P\}}\left[R(\boldsymbol{s}(P))P \notin \mathcal{L}_I\right]. \tag{2.36}$$

Binary representation of quantum error correction

We can simplify the calculation of QEC by using a binary representation of Pauli operators. Let a row vector $\boldsymbol{e} := b(P)$ be the binary representation of a Pauli error P. We define a matrix

$$H_c \;\; := \;\; \begin{pmatrix} b(S_0) \\ \vdots \\ b(S_{n-k-1}) \end{pmatrix}, \tag{2.37}$$

which is called a check matrix. We see a syndrome $\boldsymbol{s}(P)$ is given by

$$\boldsymbol{s}(P) = H_c\Lambda\boldsymbol{e}^{\mathrm{T}}. \tag{2.38}$$

We denote $\boldsymbol{s}(\boldsymbol{e}) = \boldsymbol{s}(P)$ with $\boldsymbol{e} = b(P)$ in the discussion below.

We define a binary representation of the normalizer of the stabilizer group as

$$\mathcal{L} := b(\mathcal{N}(\mathcal{S})) = b(\mathcal{C}(\mathcal{S})) = \{\boldsymbol{v} | \boldsymbol{v} \in \{0,1\}^{2n}, \forall \boldsymbol{v}' \in b(\mathcal{S}), \boldsymbol{v} \odot \boldsymbol{v}' = 0\}. \tag{2.39}$$

Note that $|\mathcal{L}| = 2^{n+k}$. Since the group $\mathcal{N}(\mathcal{S})$ contains the stabilizer group, there exists a set of $(n+k)$ Pauli operators $\mathcal{S}_N$ which satisfies $\mathcal{L} = b(\langle \mathcal{S}_N \rangle)$ and includes $\mathcal{S}$ as a subset. We denote added elements as

$$\mathcal{S}_N \setminus \mathcal{S} = \{L_0, \ldots, L_{2k-1}\}. \tag{2.40}$$

Then, we define a logical generator matrix as

$$G := \begin{pmatrix} b(L_0) \\ \vdots \\ b(L_{2k-1}) \end{pmatrix}. \tag{2.41}$$

We define cosets $\mathcal{L}_{\boldsymbol{w}}$ with $\boldsymbol{w} \in \{0,1\}^{2k}$ which are defined by

$$\mathcal{L}_{\boldsymbol{w}} = \{\boldsymbol{l} \oplus \boldsymbol{w}G | \boldsymbol{l} \in b(\langle \mathcal{S} \rangle)\}. \tag{2.42}$$

We introduce a pure error $\boldsymbol{t}(\boldsymbol{s})$, which maps a syndrome value $\boldsymbol{s}^{\mathrm{T}} \in \{0,1\}^{n-k}$ to a binary representation of a Pauli operator $\boldsymbol{t}(\boldsymbol{s}) \in \{0,1\}^{2n}$ such that

$$\boldsymbol{t}(\boldsymbol{s}(\boldsymbol{e})) \oplus \boldsymbol{e} \in \mathcal{L}, \tag{2.43}$$

for all $\boldsymbol{e} \in \{0,1\}^{2n}$. Given a generator matrix G and a pure error $\boldsymbol{t}(\boldsymbol{s})$, we can uniquely represent the binary representation of an arbitrary physical error $\boldsymbol{e}$ as

$$\boldsymbol{e} = \boldsymbol{l}(\boldsymbol{e}) \oplus \boldsymbol{w}(\boldsymbol{e})G \oplus \boldsymbol{t}(\boldsymbol{s}(\boldsymbol{e})), \tag{2.44}$$

where $\boldsymbol{l}(\boldsymbol{e}) \in \mathcal{L}_0$ and $\boldsymbol{w}(\boldsymbol{e}) \in \{0,1\}^{2k}$. As for the representation in Eq. (2.44), we can easily confirm the following relations:

$$b(\mathcal{N}(\mathcal{S})) = \mathcal{L} = \cup_{\boldsymbol{w} \in \{0,1\}^{2k}} \mathcal{L}_{\boldsymbol{w}} \tag{2.45}$$

$$b(\langle \mathcal{S} \rangle) = \mathcal{L}_0 \tag{2.46}$$

$$\boldsymbol{e} \in \mathcal{L}, \boldsymbol{e}' \in \mathcal{L}_0 \rightarrow \boldsymbol{e} \odot \boldsymbol{e}' = 0 \tag{2.47}$$

$$\boldsymbol{e} \in \mathcal{L} \leftrightarrow \boldsymbol{s}(\boldsymbol{e}) = 0 \tag{2.48}$$

$$\boldsymbol{e} \in \mathcal{L}_0 \leftrightarrow \boldsymbol{s}(\boldsymbol{e}) = 0, \boldsymbol{w}(\boldsymbol{e}) = 0 \tag{2.49}$$

We denote the binary representation of a recovery Pauli operator by $\boldsymbol{r}(\boldsymbol{s}) := b(R(\boldsymbol{s}))$ for $\boldsymbol{s}^{\mathrm{T}} \in \{0,1\}^{n-k}$. Since the global phase of the occurred Pauli error P does not matter the decoding result, the probability distribution of Pauli errors $\{p_P\}$ can be interpreted as that of the binary representation of Pauli errors $\{p_{\boldsymbol{e}}\}$ written by

$$p_{\boldsymbol{e}} = \sum_{P = \{\pm 1, \pm i\} \times B(\boldsymbol{e})} p_P. \tag{2.50}$$

Given an error distribution $\{p_{\boldsymbol{e}}\}$ and a decoder $\boldsymbol{r}(\boldsymbol{s})$, the logical error probability is written as

$$p_{\mathrm{L}} = \Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}\left[\boldsymbol{e}\oplus\boldsymbol{r}(\boldsymbol{s}(\boldsymbol{e}))\notin\mathcal{L}_0\right] \tag{2.51}$$

Since we consider decoders satisfying Eq. (2.35), the decoder $\boldsymbol{r}(\boldsymbol{s})$ satisfies

$$\boldsymbol{e}\oplus\boldsymbol{r}(\boldsymbol{s}(\boldsymbol{e}))\in\mathcal{L} \tag{2.52}$$

for an arbitrary physical error $\boldsymbol{e}$. The logical error probability is then written as

$$p_{\mathrm{L}} = \Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}\left[\boldsymbol{w}(\boldsymbol{e})\neq\boldsymbol{w}(\boldsymbol{r}(\boldsymbol{s}(\boldsymbol{e})))\right]. \tag{2.53}$$

We see estimating a correct recovery Pauli operator $\boldsymbol{r}(\boldsymbol{s}(\boldsymbol{e}))$ from $\boldsymbol{s}(\boldsymbol{e})$ is equivalent to estimating a correct binary vector $\boldsymbol{w}(\boldsymbol{e})$ from $\boldsymbol{s}(\boldsymbol{e})$.

Optimal decoder

An optimal decoder is defined as a decoder which minimizes the logical error probability for a given probability distribution $\{p_{\boldsymbol{e}}\}$. Such a minimization is achieved by choosing the most probable $\boldsymbol{w}$ given the observed syndrome $\boldsymbol{s}$. We call a binary vector $\boldsymbol{w}$ as a class, and call $\boldsymbol{w}(\boldsymbol{e})$ as a class of $\boldsymbol{e}$. We define the probability of class $\boldsymbol{w}$ conditioned on a syndrome $\boldsymbol{s}$ as

$$q_{\boldsymbol{s}}(\boldsymbol{w}) = \Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}\left[\boldsymbol{w}(\boldsymbol{e})=\boldsymbol{w}|\boldsymbol{s}(\boldsymbol{e})=\boldsymbol{s}\right]. \tag{2.54}$$

We define the maximum probability among the classes as

$$q^*_{\boldsymbol{s}} = \max_{\boldsymbol{w}\in\{0,1\}^{2k}} q_{\boldsymbol{s}}(\boldsymbol{w}). \tag{2.55}$$

The optimal decoder outputs a binary representation of a recovery operation $\boldsymbol{r}(\boldsymbol{s})$ such that

$$q_{\boldsymbol{s}}(\boldsymbol{w}(\boldsymbol{r}(\boldsymbol{s}))) = q^*_{\boldsymbol{s}} \tag{2.56}$$

for any $\boldsymbol{s}$ with $\Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}\left[\boldsymbol{s}(\boldsymbol{e})=\boldsymbol{s}\right]>0$.

2.2.2 Topological stabilizer codes

Properties of topological codes

The topological stabilizer code [18] is a family of stabilizer codes. It is known that the topological stabilizer code has the following two properties which are desirable for implementing QEC in experiments.

First, topological codes are known to have high threshold values p_{th} for various noise models. If an error probability per qubit p is smaller than the threshold

value $p_{\rm th}$, the logical error probability $p_{\rm L}$ is improved by increasing the distance d of topological codes. It is also known that if p is sufficiently small compared to $p_{\rm th}$, the logical error probability $p_{\rm L}$ exponentially decreases as the distance d increases.

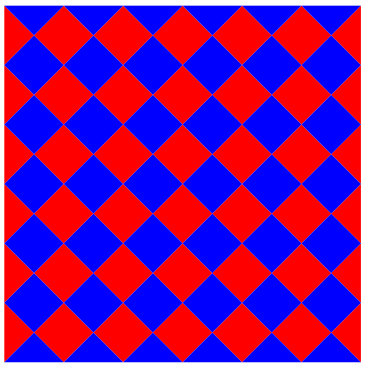
The other property is implementation-friendly arrangements of physical qubits. In the case of the topological code, there is a stabilizer generator set such that each Pauli operator in the generator set is a spatially local Pauli operation in two- or three-dimensional array of physical qubits. In any practical quantum system, qubits are allocated as at most a three-dimensional array, and interaction between qubits happens in a spatially local region. Thus, it is preferred that a stabilizer measurement can be performed only with a small number of spatially local operations on physical qubits which are allocated in experimentally feasible manner.

From these reasons, the topological code is considered as the most promising candidate of quantum error correcting codes. Therefore, current experiments toward a scalable quantum computer pursue realization of the topological stabilizer code.

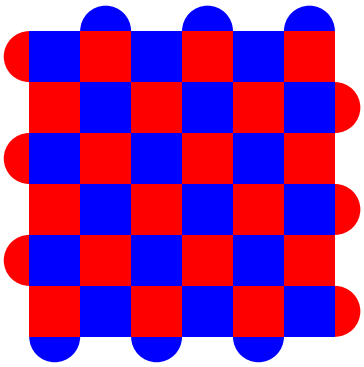
Specific qubit allocations of topological codes

Among the topological stabilizer codes, two specific constructions called surface codes [18, 20, 21] and color codes [60] are considered to be easy to implement in experiments. We show two specific constructions each for surface codes and color codes. The qubit allocation of the surface code is shown in Fig. 2.1. The $[[2d^2-2d+1, 1, d]]$ code and the $[[d^2, 1, d]]$ code are shown in Fig. 2.1 (a) and (b), respectively. In both figures, the physical qubits are located on the vertices of the colored faces. Each red face represents a stabilizer operator which is a product of Pauli-Z operators on the physical qubits of its vertices. Each blue face represents one with Pauli-X operators. We see every physical qubit is measured by two Pauli-Z stabilizer operators (red faces) and two Pauli-X stabilizer operators (blue faces).

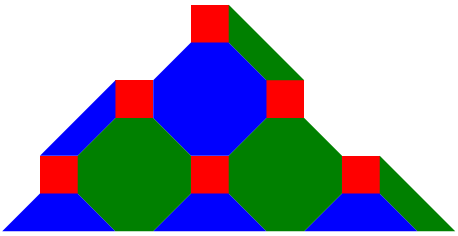
As for the color codes, two types of codes, the [4,8,8]-color code and the [6,6,6]-color code, are shown in Fig. 2.1 (c) and (d), respectively. The physical qubits are also located on each vertex of the faces. Each colored face represents two stabilizer operators, a stabilizer operator which is a product of Pauli-Z operators on the physical qubits of its vertices, and that of Pauli-X operators. Note that the color of the faces is irrelevant to the definition of the corresponding stabilizer operators. The $[4, 8, 8]$-color code is a $[[\frac{1}{2}d^2 + d - \frac{1}{2}, 1, d]]$ code, and the $[6, 6, 6]$-color code is a $[[\frac{3}{4}d^2 + \frac{1}{4}, 1, d]]$ code. We see every physical qubit except that on the boundary is measured by three Pauli-Z operators of three faces of the different colors, and is measured by three Pauli-X operators of three colored faces.



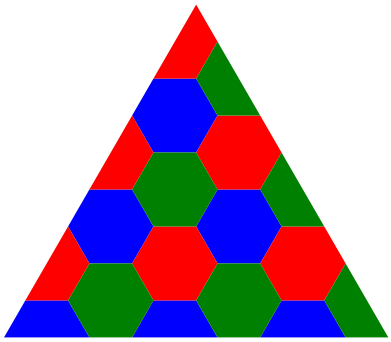
(a) $[[2d^2 - 2d + 1, 1, d]]$ surface code



(b) $[[d^2, 1, d]]$ surface code



(c) [4,8,8]-color code.



(d) [6,6,6]-color code.

Figure 2.1: The qubit allocations of surface codes and color codes are shown. Figure (a) and (b) show the qubit allocation of the surface codes with the $[[2d^2 - 2d + 1, 1, d]]$ code and the $[[d^2, 1, d]]$ code, respectively. Each vertex corresponds to a physical qubit. Red and blue faces correspond to stabilizer measurements with X and Z Pauli operator, respectively. Figure (c) and (d) shows the qubit allocation of the [4,8,8]-color code and the [6,6,6]-color code, respectively. Each vertex corresponds to a physical qubit. Each face corresponds to two stabilizer operators, Pauli X on its vertices and Pauli Z on its vertices. The color of the faces is irrelevant to the definition of the corresponding stabilizer operators.

Decoder based on minimum-weight perfect matching

It is known that we cannot efficiently perform the optimal decoding except few specific cases [27]. Since time for decoding is limited [32, 61], we cannot take long time for decoding process. This means that we cannot perform the optimal decoding in practice. Therefore, fast decoding algorithms which achieve a small logical error probability have been studied.

One popular solution to the problem in the case of the surface codes is to use one of the most probable physical errors $\boldsymbol{e}^*(\boldsymbol{s})$ conditioned on the observed syndrome $\boldsymbol{s}$ as a recovery operator, instead of finding the most probable class. This scheme is called a minimum-distance (MD) decoder. Though the class of the most probable physical errors is not necessarily the most probable, it is numerically shown that the MD decoder achieves a near-optimal performance in various codes [31, 62]. It is known that we can perform MD decoding in the surface codes if we can assume that bit-flip and phase-flip errors occur independently on each physical qubit [32]. This is because we can reduce the task of finding one of the most probable physical errors from an observed syndrome to an instance of minimum-weight perfect matching (MWPM), which is a problem of graph theory known to be efficiently solved with blossom algorithm [63, 64]. The problem of MWPM is defined as follows.

Definition 2.2.2. (Minimum-weight perfect matching) We consider a weighted graph with $2n$ nodes $G = (V, E)$, where $V = (v_0, \ldots, v_{2n-1})$ is a set of nodes and $E : V \times V \to \mathbb{R}$ is a set of weighted edges. We call $M_i = \{m_{i,0}, m_{i,1}\}$, where $0 \leq m_{i,0}, m_{i,1} \leq (2n-1)$ and $m_{i,0} \neq m_{i,1}$, as a pair. We also call a set of pairs $\mathcal{M} = \{M_0, \ldots M_{|\mathcal{M}|-1}\}$ as a matching. We say a matching $\mathcal{M}$ is perfect if every index $0 \leq m \leq (2n-1)$ is contained in exactly one pair among $\mathcal{M}$. A weight of a perfect matching $\mathcal{M}$ is defined by

$$\sum_{i=0}^{n-1} E(v_{m_{i,0}}, v_{m_{i,1}}). \tag{2.57}$$

The problem of minimum-weight perfect matching is to find a perfect matching with the minimum weight.

The blossom algorithm is an efficient algorithm to solve MWPM.

Theorem 2.2.1. (Blossom algorithm) There is an efficient algorithm to solve minimum-weight perfect matching [63], which is called blossom algorithm. The best known variation of blossom algorithm solves minimum-weight perfect matching with $O(\sqrt{|V|}|E|)$ steps of classical computation [64], where $|V|$ is the number of nodes, and $|E|$ is the number of edges.

Here we give a brief explanation of the reduction from the task of MD decoding in the $[[2d^2-2d+1, 1, d]]$ surface code under an independent bit-flip noise model to

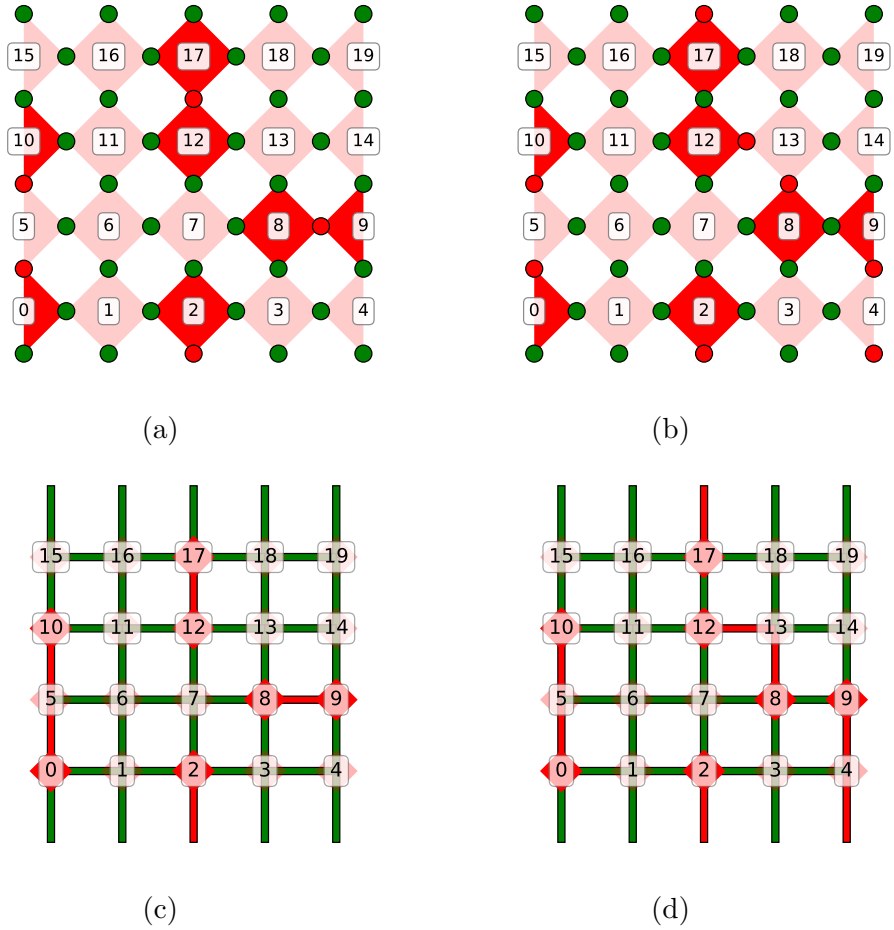


Figure 2.2: Examples of an obtained syndrome and occurred bit-flip errors in the $[[2d^2 - 2d + 1, 1, d]]$ surface code at $d = 5$. Figure (a) and (b) have different physical errors, but show the same syndrome pattern. Figure (c) and (d) are another representation of Figure (a) and (b), respectively.

an instance of MWPM. Before starting mathematical explanation, we show why the decoding problem corresponds to the matching problem with the illustrations shown in Fig. 2.2. Since we assume that there are only bit-flip (Pauli-X) errors, we ignore blue faces of the surface code, which monitor phase-flip (Pauli-Z) errors with Pauli-X stabilizer operator. The green and red circles at the vertices of the red faces correspond to physical qubits. If there is a bit-flip error on a physical qubit, qubit is colored with red. Otherwise, it is colored with green. The i-th red square corresponds to the i-th element of the stabilizer generator S_i, where S_i is a product of Pauli-Z operators on its vertices. The i-th red square is colored deep red if S_i anti-commutes with an occurred error, which means that the i-th stabilizer generator detects a bit-flip error. The i-th red square is colored with pale red otherwise.

We see both Fig. 2.2 (a) and (b) show the same syndrome pattern, but occurred physical errors are different. The situations of Fig. 2.2 (a) and (b) are shown with another representation in Fig. 2.2 (c) and (d), respectively. In Fig. 2.2 (c) and (d), each deep red face is represented as a deep red square, and each pale one as a pale red square. Each physical error is represented as an edge connecting two squares, where the color of each physical error is not changed. In these figures, we see any red square is paired with another red square or boundary through red edges. Since the number of red edges is the same as the weight of a bit-flip error, finding the most probable bit-flip error is related to finding a perfect matching of red squares with the minimum number of red edges. Actually, we can construct an instance of MWPM such that we can construct one of the most probable physical errors from the answer of the instance. We construct such an instance as follows.

As seen in Fig. 2.2, there are $d(d-1) = \frac{n-1}{2}$ relevant stabilizer operators (red faces) for detecting bit-flip errors. We denote the corresponding syndrome bits by $s_0, \ldots, s_{\frac{n-1}{2}-1}$. We also define the parity of these bits as

$$s_{\frac{n-1}{2}} := \bigoplus_{i=0}^{\frac{n-1}{2}-1} s_i, \tag{2.58}$$

and construct an extended syndrome binary vector

$$\boldsymbol{s}_{\text{node}} = (s_0, \ldots, s_{\frac{n-1}{2}}). \tag{2.59}$$

We denote the set of the indices of the syndrome bits of which the value is unity as

$$V = \{m | 0 \leq m \leq \frac{n-1}{2}, (\boldsymbol{s}_{\text{node}})_m = 1\}. \tag{2.60}$$

In the case of the surface code, any bit-flip error on a physical qubit changes exactly two elements of $\boldsymbol{s}_{\text{node}}$. Thus, the number of elements in $\boldsymbol{m}_{\text{node}}$ is always even, which we denote by $2n$. For a given pair $M = \{m, m'\}$, we denote one of the most probable physical errors which generate a syndrome such that $(\boldsymbol{s}_{\text{node}})_m = (\boldsymbol{s}_{\text{node}})_{m'} = 1$ and $(\boldsymbol{s}_{\text{node}})_j = 0$ for all $j \neq m, m'$ as $\boldsymbol{e}_M$. We denote the probability with which the physical error $\boldsymbol{e}_M$ happens as p_M. Note that p_M and $\boldsymbol{e}_M$ can be efficiently calculated in this setting. We consider a perfect matching $\mathcal{M} = \{M_0, \ldots, M_{n-1}\}$ of $\boldsymbol{m}_{\text{node}}$. The probability $p_{\mathcal{M}}$ of the perfect matching $\mathcal{M}$ is defined by the product of the probabilities of each pair p_{M_i}, i.e.,

$$p_{\mathcal{M}} = \prod_{i=0}^{n-1} p_{M_i}. \tag{2.61}$$

By taking a logarithm of Eq. (2.61) and inverting a sign, we obtain

$$-\ln p_{\mathcal{M}} = \sum_{i=0}^{n-1} (-\ln p_{M_i}). \tag{2.62}$$

The minimization of the right-hand side of Eq. (2.62) in terms of $\mathcal{M}$ is equivalent to the maximization of $p_{\mathcal{M}}$. We construct an instance of MWPM with a graph $G = (V, E)$, where

$$E(v, v') \quad = \quad -\ln p_{\{v,v'\}}. \tag{2.63}$$

According to Theorem 2.2.1, we can efficiently obtain a perfect matching $\mathcal{M}^*$ with the minimum weight, which maximizes $p_{\mathcal{M}}$. Then, the physical error

$$\boldsymbol{r}^*(\boldsymbol{s}) = \bigoplus_{M_i \in \mathcal{M}^*} \boldsymbol{e}_{M_i} \tag{2.64}$$

is one of the most probable physical errors conditioned on $\boldsymbol{s}$. Since a graph G is fully connected, this procedure takes $O(n^{5/2}) = O(d^5)$ steps in the worst case.

We briefly explain why $\mathcal{M}^*$ should be correspond to one of the most probable physical errors. We see any physical error $\boldsymbol{e}$ leading to the given syndrome can be related to at least one perfect matching $\mathcal{M}$ as shown in Fig. 2.2. Then, we have $p_{\boldsymbol{e}} \leq p_{\mathcal{M}}$ since $p_{\mathcal{M}}$ is defined as the maximal probability among the errors consistent with $\mathcal{M}$. On the other hand, since $\mathcal{M}^*$ is the solution of the MWPM instance, we have $p_{\mathcal{M}} \leq p_{\mathcal{M}^*} = p_{\boldsymbol{r}^*(\boldsymbol{s})}$. Hence we have $p_{\boldsymbol{e}} \leq p_{\boldsymbol{r}^*(\boldsymbol{s})}$.

In the case of Fig. 2.2 (c), it corresponds to a perfect matching

$$\{(0, 10), (2, 20), (8, 9), (12, 17)\}, \tag{2.65}$$

and Fig. 2.2 (d) corresponds to

$$\{(0, 10), (2, 17), (8, 12), (9, 20)\}, \tag{2.66}$$

where "20" corresponds to the parity node. Note that a pair $(2, 17)$ in the latter case is considered to be connected through the parity node s_{20}, and thus $(2, 17)$ is considered to be paired through 2 edges. When a noise occurs uniformly, the logarithm of the probability $(-p_{\{m,m'\}})$ is proportional to the Manhattan distance between two nodes s_m and $s_{m'}$ (the minimum number of edges to connect two nodes). Under a uniform error probability, Fig. 2.2 (c) achieves MWPM.

Suppose that the actual physical error is Fig. 2.3 (a). If we use Fig. 2.2 (c) as a recovery operation, the resultant Pauli operator $R(\boldsymbol{s})P$ is as shown in Fig. 2.3 (b). Since this Pauli operator is in the stabilizer group, we see QEC is a success. On the other hand, if we use Fig. 2.2 (d) as a recovery operation, the resultant Pauli operator becomes as shown in Fig. 2.3 (c), which is not included in the stabilizer group. Thus, the resultant operation non-trivially applies to the logical space, and thus the logical state changes in general.

When bit- and phase-flip errors happen independently, we can independently perform this process for each type of errors. On the other hand, when bit-flip and phase-flip errors does not happen independently, we cannot perform MD decoding with this approach. We say such a noise model as a correlated noise model. A

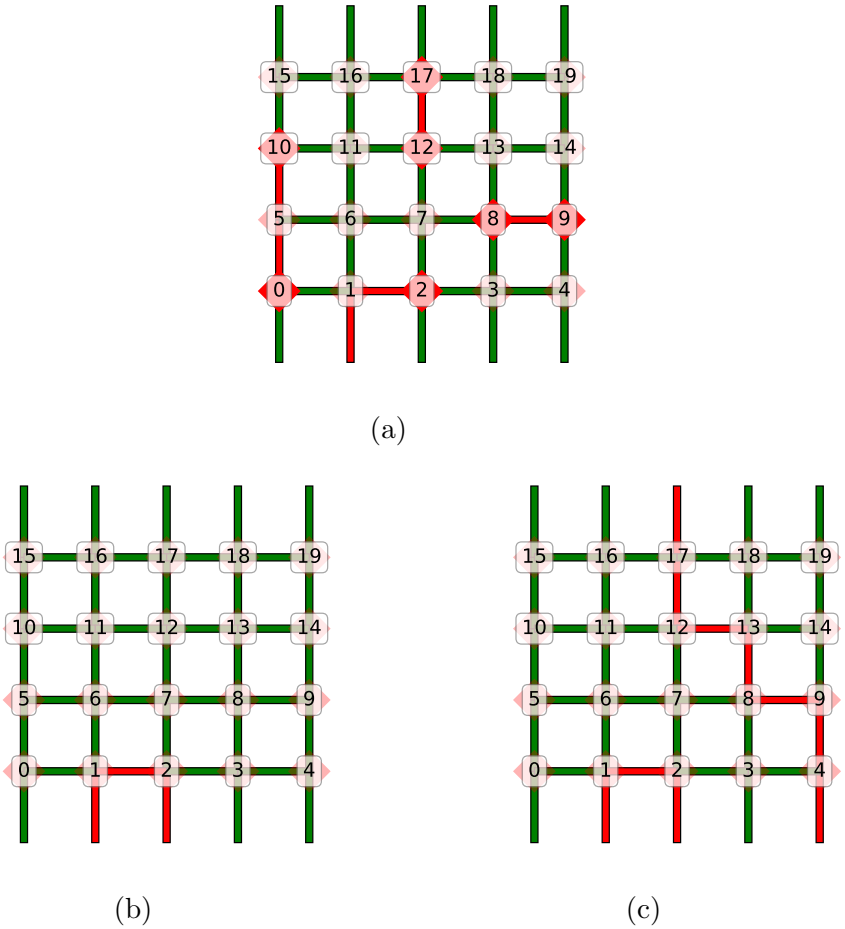


Figure 2.3: Examples of an obtained syndrome and occurred bit-flip errors in the $[[2d^2 - 2d + 1, 1, d]]$ surface code at $d = 5$. Figure (a) represents an actual bit-flip error. Figure (b) and (c) represent the resultant operation when we use bit-flip errors in Fig. 2.2 (c) and (d) as a recovery operation, respectively.

practical example of a correlated noise model is depolarizing noise model, which assumes that Pauli X-, Y-, and Z-operators occur on each physical qubit independently with the same probability.

In the reduction, we used a property of the surface code that any physical qubit is measured by at most two stabilizer operators. Since this does not hold in the case of the color code, we cannot apply this approach straightforwardly to the color code, even when we can assume that there are only bit-flip errors. Unfortunately, it is known that the MD decoder is computationally hard to construct in general [26]. Thus, another approach is required for performing near-optimal decoding under correlated noise models, or for performing it in topological stabilizer codes other than the surface code. The main topic of the Chapter 4 of the thesis is to construct a near-optimal decoder which is fast, high-performance, and applicable to an arbitrary noise model and an arbitrary topological stabilizer code, by allowing massive computation before we use decoding algorithm in experiments.

Threshold value and dropping rate

We show typical behavior of the logical error probability in topological stabilizer codes. We numerically calculated logical error probabilities in the $[[2d^2 - 2d + 1, 1, d]]$ surface code under a uniform bit-flip (Pauli X) noise model using the MD decoder with $d = 3$ to $d = 15$. We calculated 10^6 samples per plot of whether QEC succeeds or not. The logical error probability $p_{\rm L}$ to the physical error probability p is shown in Fig 2.4.

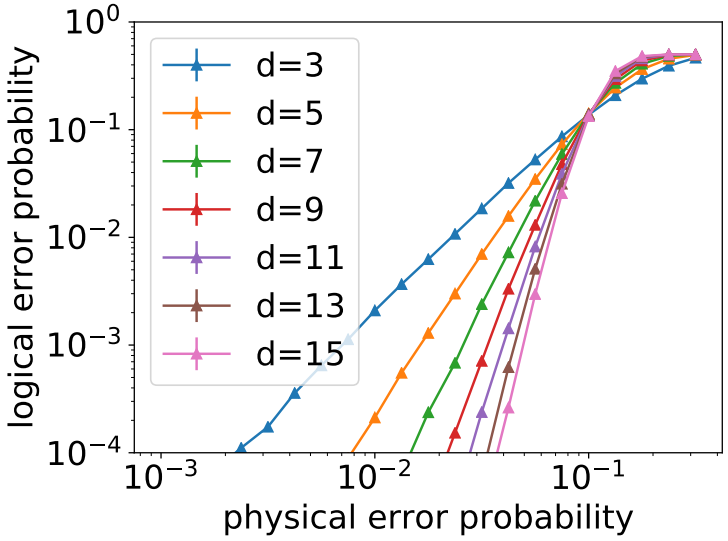
Fig 2.4 (a) shows logical error probabilities against small p to large p. The logical error probability is exponentially small to the distance d. We see that there is a threshold value of physical error probability, such that the logical probability improves by increasing the distance if the physical error probability is smaller than a threshold value. This value of the physical error probability is called the error threshold. Fig 2.4 (b) shows logical error probabilities around the threshold value. According to a scaling ansatz [30, 33], the behavior of a logical error probability around the threshold value is written by

$$p_{\rm L}(p, d) = a + b(p - p_{\rm th})d^{1/c}, \tag{2.67}$$

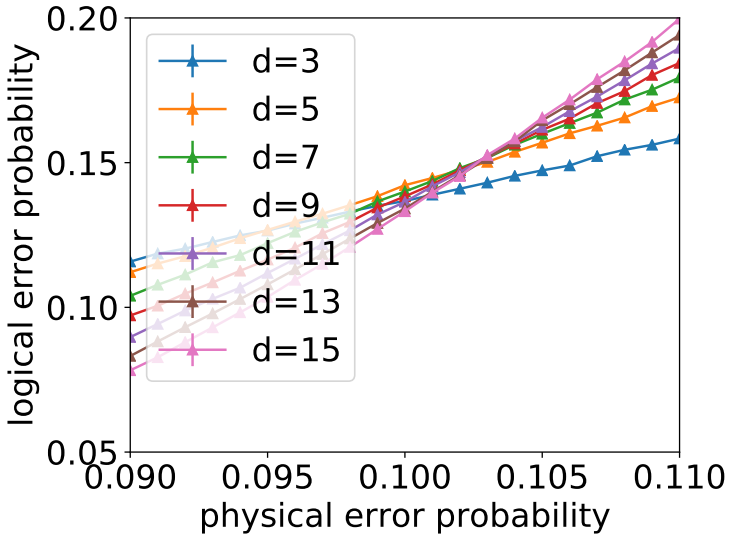
where $p_{\rm th}$ is a threshold value, and a, b, and c are fitting parameters. We obtained $p_{\rm th} = 0.1029(1)$ by fitting the plots with the above ansatz. This value is consistent with reported values 0.1030(1) [30, 33]. Note that we use plots with $d \geq 7$ for avoiding finite size effects.

In a practical task of quantum computation, it is expected that QEC is used with a physical error probability far smaller than the threshold value. At a region of $p \ll p_{\rm th}$, a logical error probability is expected to obey the following equation [61, 65]

$$p_{\rm L}(p, d) = a\left(b\frac{p}{p_{\rm th}}\right)^{(d+1)/2}, \tag{2.68}$$



(a)



(b)

Figure 2.4: Logical error probabilities according to physical error probability per qubit are plotted. Two figures are plotted in different range of physical error probabilities.

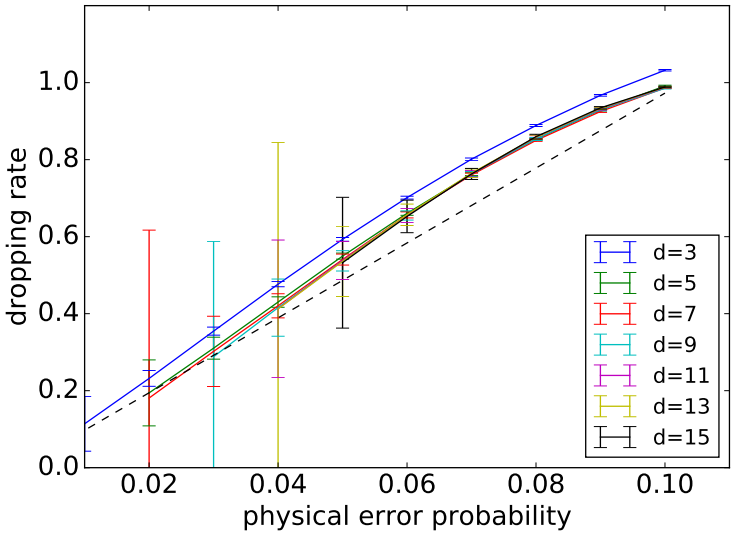


Figure 2.5: Dropping rate of logical error probabilities $\lambda(p, d)$ is plotted to the physical error probability p. The black dashed line corresponds to $\lambda = \frac{p}{p_{\rm th}}$ with $p_{\rm th}$ determined from the threshold behavior.

where a and b are fitting parameters, and $p_{\rm th}$ is the threshold value. According to the existing work [61], b is expected to be close to 1. We defined a dropping rate $\lambda(p, d)$ as

$$\lambda(p, d) = \frac{p_{\rm L}(p, d+2)}{p_{\rm L}(p, d)}. \tag{2.69}$$

The calculated dropping rates λ is shown in Fig 2.5. We calculated at least 5×10^7 samples per plot, and showed plots of which the error bar is smaller than 0.5. Note that the error bar is calculated only up to the first order of the statistical deviation. The dropping rate is expected to be close to $\frac{p}{p_{\rm th}}$ if $b \sim 1$. The numerical results agree with the expectation. Since the dropping rate is slightly larger than $\frac{p}{p_{\rm th}}$, we see that b is slightly larger than 1.

In this subsection, we showed two values, the threshold value $p_{\rm th}$ and the dropping rate $\lambda(p, d)$, which are calculated from the logical error probability $p_{\rm L}$ as a function of the physical error probability p. The threshold value represents a requirement on the imperfection in quantum devices. If the physical error probability is larger than the threshold value, the logical error probability is not improved by QEC. The dropping rate determines how many qubits we need to achieve a required logical error probability. Thus, these two values are essential when we design quantum experiments for QEC.

2.2.3 Summary and discussion

We showed a framework of QEC with stabilizer codes. Stabilizer formalism provides a scheme to specify a subspace of a quantum system. Stabilizer codes are a family of quantum error correcting codes which specify its logical space with stabilizer formalism.

The entanglement fidelity of a logical channel under practical noise models is the most important property of stabilizer codes. When we can assume probabilistic Pauli noise models, the value of the entanglement fidelity is formulated as a logical error probability p_{L}. The logical error probability is expected to become small exponentially to the distance d, if noise in a quantum system is sufficiently small.

We reviewed topological stabilizer codes. Topological stabilizer codes are high-performance quantum error correcting codes with implementation-friendly properties for experiments. We showed specific qubit allocations for two types of topological stabilizer codes, the surface code and the color code. These topological stabilizer codes are expected to be the most promising candidates of the quantum error correcting codes for demonstrating QEC in experiments.

When we perform decoding in experiments, we should use a high-performance decoder since the performance of the stabilizer code such as the logical error probability p_{L} depends on the performance of a chosen decoder. Since the optimal decoding algorithm is believed to take a time exponential to the number of the physical qubits n in general, the optimal decoding algorithm is not practical except few cases. We introduced a minimum-distance (MD) decoder, which is known to be near-optimal. We showed that an efficient MD decoder can be constructed for the surface code under uncorrelated bit- and phase-flip errors by reducing the decoding problem to minimum-weight perfect matching (MWPM), and by solving it with the blossom algorithm.

We numerically calculated the logical error probability p_{L} in terms of an error probability per qubit p for the $[[2d^2 - 2d + 1, 1, d]]$ surface code under the bit-flip noise model with the MD decoder. We saw that there is a threshold value p_{th} such that the logical error probability p_{L} decreases as the increase of the number of the physical qubits n if the physical error probability p is smaller than p_{th}. We can say that the threshold value p_{th} represents a requirement on the imperfection in quantum devices in experiments. We confirmed that the threshold value in the above setting is about 10%. We are also interested in how much the logical error probability p_{L} improves according to the increase of the distance d when the physical error probability p is sufficiently smaller than p_{th}. The dropping rate $\lambda(p, d)$ is defined as the decreasing rate of the logical error probability p_{L} in terms of the distance d. It determines how many qubits n are required for achieving a required logical error probability p_{L}. As in Eq. (2.68), it is conjectured that the dropping rate $\lambda(p, d)$ is almost independent of the distance d, and its value can be estimated from the ratio between the threshold value p_{th} and the physical error probability p. We numerically confirmed that the dropping rate $\lambda(p, d)$ shows

the behavior expected from this conjecture. These parameters, the logical error probability p_{L}, the threshold value p_{th}, and the dropping rate $\lambda(p, d)$ are essential properties of quantum error correcting codes for designing quantum devices for QEC.

2.3 Efficiently simulatable class of quantum circuits

2.3.1 Overview

In this section, we discuss the simulatability of a given quantum circuit. In order to evaluate performance of a quantum error correcting code for a noise model, whether quantum circuits of the quantum error correcting code under the noise model can be efficiently simulated or not is essential. The most straightforward method of the simulation is to store and update a quantum state vector, i.e., 2^n amplitudes in computational basis of an n-qubit system. This method takes memory and time exponential to the number of qubit n, and this approach is exhausted at around 50 qubits with the existing classical computers [66, 67]. Thus, there have been massive efforts to reveal what class of quantum circuits can be efficiently simulated with classical computer.

In this section, we clarify three definitions of simulatability of quantum circuits: strong simulation, weak simulation, and adaptive simulation. The adaptive simulatability is the most essential property in the context of QEC. Then, we show two important efficiently simulatable classes of quantum circuits: the Clifford circuit and the matchgate circuit. Clifford circuits can be understood as an efficiently traceable dynamics in stabilizer formalism. The simulatability of matchgate circuits is based on an efficiently traceable dynamics of free fermions.

2.3.2 Definitions of simulatabilities

We first clarify the definitions of simulatability [68]. We show three types of simulatability: strong simulation, weak simulation, and adaptive simulation.

In this thesis, we consider an n-qubit quantum circuit C_n with the following properties. Every qubit is finally measured or discarded. The outcome of the final measurements for qubits is binary. Other intermediate operations in C_n may include unitary operations, probabilistic operations such as noise, intermediate measurements such as stabilizer measurements, and unitary operations based on the previous outcomes of intermediate measurements such as a recovery operation.

For a given quantum circuit C_n, we denote the number of qubits which subject to the final measurements as $m(C_n)$. We denote the probability with which we obtain $\boldsymbol{v} \in \{0,1\}^{m(C_n)}$ as $p_{C_n}(\boldsymbol{v})$, where $v_i \in \{0,1\}$ is the outcome of the final measurement at the i-th measured qubits. We also define a marginal probability

$p_{C_n}(\tilde{\boldsymbol{v}})$ with $\tilde{\boldsymbol{v}} \in \{0,1,*\}^{m(C_n)}$ as

$$p_{C_n}(\tilde{\boldsymbol{v}}) := \sum_{\boldsymbol{v} \sim \tilde{\boldsymbol{v}}} p_{C_n}(\boldsymbol{v}). \tag{2.70}$$

The first definition is strong simulation, which is defined as follows.

Definition 2.3.1. (Strong simulation) We say an n-qubit quantum circuit C_n is strongly simulatable if the probability $p_{C_n}(\tilde{\boldsymbol{v}})$ for an arbitrary masked binary vector $\tilde{\boldsymbol{v}} \in \{0,1,*\}^{m(C_n)}$ can be calculated with poly(n) steps of classical computation.

The second is weak simulation, which is defined as follows.

Definition 2.3.2. (Weak simulation) We say an n-qubit quantum circuit C_n is weakly simulatable if we can sample a binary vector $\boldsymbol{v} \in \{0,1\}^{m(C_n)}$ from the probability distribution $\{p_{C_n}(\boldsymbol{v})\}$ with poly(n) steps of classical computation.

We finally introduce the definition of adaptive simulation, which is a hybrid of strong one and weak one, in the following sense. We represent all of the intermediate components in C_n such as unitary operations, probabilistic operations, intermediate measurements, and unitary operations based on the previous outcomes of intermediate measurements, as a large instrument I_n. We denote the outcome of I_n as $\boldsymbol{v}_{I_n}$. We also denote the probability with which we obtain $\boldsymbol{v}_{I_n}$ as $p_{I_n}(\boldsymbol{v}_{I_n})$, and the probability with which we obtain $\tilde{\boldsymbol{v}} \in \{0,1,*\}^{m(C_n)}$ at the final measurement conditioned on the outcome $\boldsymbol{v}_{I_n}$ as $p_{C_n}(\tilde{\boldsymbol{v}}|\boldsymbol{v}_{I_n})$. Then, adaptive simulatability is defined as follows.

Definition 2.3.3. (Adaptive simulation) We say an n-qubit quantum circuit C_n with an intermediate outcome $\boldsymbol{v}_{I_n}$ is simulatable in adaptive sense if we can sample $\boldsymbol{v}_{I_n}$ from the probability distribution $\{p_{I_n}(\boldsymbol{v}_{I_n})\}$ with poly(n) steps of classical computation, and if the probability $p_{C_n}(\tilde{\boldsymbol{v}}|\boldsymbol{v}_{I_n})$ can be calculated for an arbitrary masked binary vector $\tilde{\boldsymbol{v}} \in \{0,1,*\}^{m(C_n)}$ and for an arbitrary outcome $\boldsymbol{v}_{I_n}$ of I_n with poly(n) steps of classical computation.

When we can simulate a quantum circuit of quantum error correcting codes in adaptive sense, we can sample syndrome values, and can calculate the success probability of decoding conditioned on the observed syndrome efficiently and accurately. By repeating the sampling of the success probabilities, we can estimate the logical error probability within a statistical error. Thus, adaptive simulatability of a given quantum error correcting code under a noise model is a sufficient condition for evaluating the performance of the quantum error correcting code accurately and efficiently.

In this section, we show two known simulatable classes of quantum circuits in adaptive sense, the Clifford circuit and the matchgate circuit.

2.3.3 Clifford circuit

The Clifford circuit is the most popular efficiently simulatable class of quantum circuits, which is studied by Gottesman [46]. The simulatability of Clifford circuits is based on the stabilizer formalism described in Sec 2.2.1. Since stabilizer formalism specifies a 2^k-dimensional subspace with $(n-k)$ Pauli operators, we can specify a quantum state with n Pauli operators. For example, a quantum state $|0\rangle^{\otimes n}$ can be stabilized with a set of n Pauli operators $\{Z_0, \ldots, Z_{n-1}\}$.

Let us consider an n-qubit quantum circuit C_n with the following properties. An initial state is $|0\rangle^{\otimes n}$. Every intermediate quantum operation is a unitary operation. We perform a projection measurement with projection operators $\frac{I \pm P_i}{2}$, where $P_i \in \{X_i, Y_i, Z_i\}$, as the final measurements on each qubit, which we call measurements with Pauli basis. We see that the initial state is characterized with a generator set $\{Z_0, \ldots, Z_{n-1}\}$. The action of the circuit C_n can be regarded as sequential updating of the quantum state via the intermediate operations. Suppose that a current quantum state $|\psi\rangle$ which is specified with a set of n Pauli operators $\mathcal{S}$ is updated with a unitary operator U contained in the circuit C_n. Since

$$U|\psi\rangle = US|\psi\rangle = (USU^\dagger)U|\psi\rangle \tag{2.71}$$

for any $S \in \mathcal{S}$, the updated quantum state $U|\psi\rangle$ is specified as the $+1$ eigenstate of a set of operators $\mathcal{S}' = \{USU^\dagger | S \in \mathcal{S}\}$. If the new set of operators $\mathcal{S}'$ consists only of Pauli operators, we can specify the new quantum state with the stabilizer formalism. The following class of operators is thus defined.

Definition 2.3.4. (Clifford operator) An operator C is called a Clifford operator if and only if C satisfies

$$CPC^\dagger \in \mathcal{P}_n \tag{2.72}$$

for an arbitrary Pauli operator $P \in \mathcal{P}_n$.

If every intermediate unitary operator in C_n is a Clifford operator, we can specify quantum states at any step in C_n using a set of n Pauli operators. Since any set of n Pauli operators can be represented by $2n(n+1)$ classical bits, we can efficiently track a quantum circuit C_n. It is also known that we can efficiently calculate the probability $p(s)$, where s is an outcome of a Pauli measurement, and can efficiently calculate the updated stabilizer generator of the post-measurement state [1]. As for the final measurements, we can calculate the marginal probability $p_{C_n}(\tilde{\boldsymbol{v}})$ for an arbitrary masked vector $\tilde{\boldsymbol{v}}$ as follows. We set $i = 0$ and $p = 1$. If the i-th element of $\tilde{\boldsymbol{v}}$ is $*$, we set $i \leftarrow i + 1$. If not, we calculate the probability $p(\tilde{v}_i)$ for the measurement of the i-th qubit, update $p \leftarrow pp(\tilde{v}_i)$, update the stabilizer generator by assuming we obtain the outcome $\tilde{v}_i$, and set $i \leftarrow i+1$. We repeat them from $i = 0$ to $i = (m(C_n) - 1)$. Then, the resultant p is the value of $p_{C_n}(\tilde{\boldsymbol{v}})$. When there is an intermediate measurement in a quantum circuit C_n, we can simulate it

in weak sense as follows. Let s be the outcome of the intermediate measurement. We calculate the probabilities $p(s)$ for all possible values of s, randomly choose a specific outcome s with the probability $p(s)$, and construct the stabilizer generator specifying the state after the measurement. We can obviously treat probabilistic Clifford operations and Clifford operations based on the previous outcomes of intermediate measurements in weak sense. Thus, we reach the following theorem.

Theorem 2.3.1. (Gottesman-Knill theorem [46]) Let C_n be an arbitrary quantum circuit which satisfies the following. The quantum circuit C_n consists of unitary Clifford operations, probabilistic Clifford operations, intermediate Pauli measurements, and unitary Clifford operations based on the outcomes of the intermediate Pauli measurements. The initial state of C_n is $|0\rangle^{\otimes n}$, and each qubit is eventually measured by Pauli basis or discarded. Then, we can simulate C_n in adaptive sense.

2.3.4 Matchgate circuit

Matchgate circuits are another efficiently simulatable class of quantum circuits, which is suggested by Valiant [47]. The class of matchgate circuits is later understood to be equivalent to efficiently simulatable dynamics of free fermions [48, 49]. The applicable scope of matchgate circuits is widened from unitary dynamics to non-unitary dynamics by Knill and Bravyi [48, 50].

Free-fermionic dynamics

We consider a quantum system with n fermionic modes. Let a_i and $a_i^\dagger$ with $i = 0, \ldots n-1$ be the annihilation operator and the creation operator of the i-th fermionic mode, respectively. These operators satisfy the following commutation relations,

$$\{a_i, a_j\} = \{a_i^\dagger, a_j^\dagger\} = 0, \tag{2.73}$$

$$\{a_i, a_j^\dagger\} = \delta_{ij}, \tag{2.74}$$

where $\{a, b\} = ab + ba$.

Let $|\mathrm{vac}\rangle_f$ be the vacuum state. We denote 2^n Fock basis of the n fermionic modes as $\{|\boldsymbol{f}\rangle_f\}$, where $\boldsymbol{f} \in \{0,1\}^n$ and

$$|\boldsymbol{f}\rangle_f := (a_0^\dagger)^{f_0} \cdots (a_{n-1}^\dagger)^{f_{n-1}} |\mathrm{vac}\rangle_f . \tag{2.75}$$

We define $2n$ Majorana fermionic operators $\{c_i\}$ with $i = 0, \ldots, 2n-1$ as

$$c_{2i} := a_i + a_i^\dagger, \tag{2.76}$$

$$c_{2i+1} := (-i)(a_i - a_i^\dagger). \tag{2.77}$$

The following relations are derived from those of the fermionic operators Eqs. (2.73) and (2.74):

$$c_i = c_i^\dagger, \tag{2.78}$$
$$c_i^2 = I, \tag{2.79}$$
$$\{c_i, c_j\} = 2\delta_{ij}. \tag{2.80}$$

We define a vector of Majorana fermionic operators as $\boldsymbol{c} = (c_0, \ldots, c_{2n-1})^{\mathrm{T}}$.

The essence of efficient simulatability of matchgate circuits is based on a property of fermionic Gaussian states and fermionic Gaussian operations. We define fermionic Gaussian states and fermionic Gaussian operations as follows.

Definition 2.3.5. (fermionic Gaussian state) For given $2n$ Majorana fermionic operators $\boldsymbol{c}$, we say a quantum state ρ is Gaussian if it has a form of

$$\rho = C \exp\left(\frac{i}{2}\boldsymbol{c}^{\mathrm{T}} G \boldsymbol{c}\right), \tag{2.81}$$

where C is a normalization factor, and G is a $2n \times 2n$ anti-symmetric real-valued matrix.

Definition 2.3.6. (fermionic Gaussian operation) For given $2n$ Majorana fermionic operators $\boldsymbol{c}$, we say an operator K is Gaussian if it has a form of

$$K = C \exp\left(\frac{1}{2}\boldsymbol{c}^{\mathrm{T}} A \boldsymbol{c}\right), \tag{2.82}$$

where C is a normalization factor, and G is a $2n \times 2n$ anti-symmetric complex matrix.

Note that fermionic Gaussian operations are not necessarily unitary. Let ρ be an arbitrary fermionic Gaussian state. The state ρ can be perfectly characterized by its covariance matrix M and its norm $\Gamma := \mathrm{Tr}(\rho)$, where the covariance matrix M is defined by

$$M_{ij} = \frac{i}{2}\frac{\mathrm{Tr}(\rho[c_i, c_j])}{\mathrm{Tr}(\rho)}, \tag{2.83}$$

where $[a, b] = ab - ba$. Consider a CP map $\mathcal{E}$ defined by $\mathcal{E}(\rho) = K\rho K^\dagger$, where K is a fermionic Gaussian operator. We call such a map a Gaussian CP map. It is known that a Gaussian CP map takes any fermionic Gaussian state to another fermionic Gaussian state [48, 50], and thus the resultant state $\mathcal{E}(\rho)$ can also be characterized by its covariance matrix and norm. The covariance matrix and norm of the resultant state can be obtained as follows. We define a maximally entangled

fermionic state ρ_{FM} in the space of $2n$ fermionic modes (corresponding to $4n$ Majorana fermionic operators) as follows.

$$\rho_{FM} = \frac{1}{2^{2n}} \prod_{i=0}^{2n-1} (I + ic_i c_{i+2n}). \tag{2.84}$$

The channel state of Gaussian CP map $\mathcal{E}$ is defined as

$$\rho_{\mathcal{E}} := (\mathcal{E} \otimes I)(\rho_{FM}). \tag{2.85}$$

Its $4n \times 4n$ covariance matrix $M_{\mathcal{E}}$ is given by

$$(M_{\mathcal{E}})_{ij} = \frac{i}{2} \frac{\mathrm{Tr}(\rho_{\mathcal{E}} [c_i, c_j])}{\mathrm{Tr}(\rho_{\mathcal{E}})}. \tag{2.86}$$

Let write $M_{\mathcal{E}}$ in the following form

$$M_{\mathcal{E}} = \begin{pmatrix} A & B \\ -B^{\mathrm{T}} & D \end{pmatrix}, \tag{2.87}$$

where A, B, and D are $2n \times 2n$ submatrices. We denote the norm of the channel state $\rho_{\mathcal{E}}$ as $\Gamma_{\mathcal{E}} := \mathrm{Tr}(\rho_{\mathcal{E}})$. Let (M, Γ) be the covariance matrix and the norm of an input state ρ, and (M', Γ') be those of the output state $\mathcal{E}(\rho)$. Then, (M', Γ') can be obtained as

$$M' = A - B(M - D)^{-1} B^{\mathrm{T}} \tag{2.88}$$

and

$$\Gamma' = \Gamma \Gamma_{\mathcal{E}} \mathrm{Pf}(M - D), \tag{2.89}$$

where $\mathrm{Pf}(\cdot)$ is the Pffafian of a matrix. Note that the Pffafian of a matrix M can be calculated as a square root of a determinant of M, i.e., $\mathrm{Pf}(M)^2 = \det(M)$. Eqs. (2.88) and (2.89) can be derived with Gaussian integration in Grassman algebra. See Ref. [50] for the derivation. The calculation of a determinant and a matrix multiplication takes at most $O(n^3)$ computational steps. Thus, we can efficiently obtain (M', Γ') from (M, Γ), norm $\Gamma_{\mathcal{E}}$, and submatrices A, B, and D. Thus, the following theorem holds.

Theorem 2.3.2. (simulatability of fermionic Gaussian dynamics) Consider a system of n fermionic modes. Let $\{\mathcal{E}_i\}$ be an arbitrary instrument that consists solely of fermionic Gaussian CP maps. Let ρ be an arbitrary fermionic Gaussian state. We denote the channel state of $\mathcal{E}_i$ as $\rho_{\mathcal{E}_i}$. If the covariance matrices and the norms of ρ and $\rho_{\mathcal{E}_i}$ are given, we can efficiently calculate the probability of obtaining outcome i when we measure the state ρ with the instrument $\{\mathcal{E}_i\}$. We can calculate the covariance matrix and the norm of the post-measurement quantum state conditioned on outcome i with $\mathrm{poly}(n)$ steps of classical computation.

Matchgate circuit

We map the efficiently simulatable framework of fermionic Gaussian dynamics to that of quantum circuits, and construct a class of simulatable quantum circuits in adaptive sense, which is called matchgate circuits.

We achieve this by relating the 2^n Fock basis $\{|\boldsymbol{f}\rangle_f\}$ to the 2^n computational basis $\{|\boldsymbol{v}\rangle\}$. There are several schemes to relate the Fock basis to the computational basis [69, 70]. The simplest and the most intuitive relation is the Jordan-Wigner transformation, which relates Fock basis to computational basis as $|\boldsymbol{f}\rangle_f = |\boldsymbol{f}\rangle$. In this thesis, to be compatible with the notation in Chapter 3, we choose the following relation.

$$|\boldsymbol{f}\rangle_f \leftrightarrow H^{\otimes n} |\boldsymbol{f}\rangle , \tag{2.90}$$

where H is a Hadamard operator

$$H = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{2.91}$$

Using the Hadamard-conjugated Jordan-Wigner transformation, Pauli operators are identified with $2n$ Majorana fermionic operators as follows.

$$c_{2i} = \prod_{j=0}^{j=i-1} X_j Z_i \tag{2.92}$$

$$c_{2i+1} = \prod_{j=0}^{j=i-1} X_j Y_i. \tag{2.93}$$

Thus, a quadratic term of the fermionic operators ic_ic_j takes one of the following forms:

$$\begin{aligned} \mathcal{Q} := & \{X_i | 0 \le i \le n-1\} \\ & \cup \{Z_i X_{i+1} \cdots X_{j-1} Z_j | 0 \le i < j \le n-1\} \\ & \cup \{Z_i X_{i+1} \cdots X_{j-1} Y_j | 0 \le i < j \le n-1\} \\ & \cup \{Y_i X_{i+1} \cdots X_{j-1} Z_j | 0 \le i < j \le n-1\} \\ & \cup \{Y_i X_{i+1} \cdots X_{j-1} Y_j | 0 \le i < j \le n-1\}. \end{aligned} \tag{2.94}$$

Applying this relation to the Theorem 2.3.2, we can map efficiently simulatable fermionic dynamics to an efficiently simulatable class of quantum circuits. We redefine the fermionic Gaussian operation in the picture of quantum circuits as follows.

Definition 2.3.7. (complete Gaussian instruments) We say an instrument $\{\mathcal{E}_i\}$ is complete if every indexed CP map has unit Kraus rank. We say a complete instrument $\{\mathcal{E}_i\}$ is complete Gaussian if every Kraus operator K_i of each CP map

$$\mathcal{E}_i(\rho) = K_i \rho K_i^\dagger \tag{2.95}$$

has a form of

$$K_i = C \exp\left(\sum_j \alpha_j Q_j\right), \tag{2.96}$$

where α_j is a complex value and $Q_j \in \mathcal{Q}$.

We show that we can efficiently simulate quantum circuits which consist of complete Gaussian instruments $\{\mathcal{E}_i\}$ in adaptive sense as follows. Let ρ be a current state, and (M, Γ) be the covariance matrix and the norm of ρ. We define p_i as the probability with which we obtain i as a classical outcome. The probability p_i is given by

$$p_i := \frac{\mathrm{Tr}(\mathcal{E}_i(\rho))}{\mathrm{Tr}(\rho)}. \tag{2.97}$$

We can efficiently calculate this using Eq. (2.89). This means we can efficiently calculate the probability distribution of the classical outcome, and we can sample an outcome accurately. The covariance matrix of the state after measurement is obtained using Eq. (2.88). We can also treat complete Gaussian instruments of which the description depends on the previously obtained outcomes in the same way. We call such a complete Gaussian instruments is adaptive. As for final measurements, if all the measurements are described as complete Gaussian instruments, we can efficiently calculate the probability distribution of the classical outcomes, including marginal ones. Thus, we can say that if the initial state is fermionic Gaussian, and every operation including final measurements in quantum circuits is a complete Gaussian instrument, we can efficiently simulate the quantum circuits in adaptive sense. Thus, the following theorem holds.

Theorem 2.3.3. (Adaptive simulatability of matchgate circuit) Let C_n be an arbitrary quantum circuit which satisfies the followings. The circuit C_n consists of complete Gaussian instruments, which may include adaptive ones. The initial state of C_n is a fermionic Gaussian state, and qubits are eventually measured with fermionic Gaussian instruments or discarded. Then, we can simulate quantum circuits C_n in adaptive sense.

When we are given quantum states and CP maps which are not guaranteed to be Gaussian, we often need to check whether the given ones are Gaussian or not. To this end, the following two properties are useful. As for a state, it is known that a state ρ is a fermionic Gaussian pure state if and only if $M^{\mathrm{T}}M = I$, where M is a covariance matrix of ρ. As for a CP map, it is known that a CP map $\mathcal{E}$ is Gaussian if it has a form of $\mathcal{E}(\rho) = K\rho K^\dagger$ with

$$K = \cos\theta I + \sin\theta c_i c_j \tag{2.98}$$

or

$$K = \frac{1}{2}(I + e^{i\theta} i c_i c_j), \tag{2.99}$$

where $\theta \in \mathbb{R}$. See Appendix A for the proof of these properties.

2.3.5 Summary and discussion

We introduced three types of simulatabilities: strong, weak, and adaptive simulation. Since a recovery operation based on outcomes of stabilizer measurements is an adaptive operation, it is essential to know what class of quantum circuits can be efficiently simulated in adaptive sense.

We showed two types of efficiently simulatable families of quantum circuits in adaptive sense: Clifford circuits and matchgate circuits. The simulatability of Clifford circuits is based on stabilizer formalism. The simulatability of matchgate circuits is based on Gaussian dynamics of fermions.

Since any stabilizer measurement without noise can be written as Pauli measurements, any noiseless quantum circuit of an arbitrary stabilizer code can be written as a Clifford circuits containing adaptive operations. Thus, as far as noise can be written as a probabilistic Clifford operation based on Pauli measurements, we can simulate noisy quantum circuits in adaptive sense using Gottesman-Knill theorem. This implies that the Gottesman-Knill theorem is a powerful tool for simulation and evaluation of stabilizer codes.

A property of the framework of matchgate circuits distinguished from Clifford circuits is that it contains quantum operations such as the Pauli-X rotation $e^{i\theta X_i}$ with an arbitrary angle $\theta \in \mathbb{R}$. We expect that coherent noise, which is not accurately tractable in the Gottesman-Knill theorem, can be treated with matchgate circuits. On the other hand, a drawback of the framework of matchgate circuits is that it cannot treat even single Pauli operators, such as Z_i and Y_i. It has not been known whether there is a non-trivial set of an error correcting code and a noise model which can be represented as a matchgate circuit. This is why matchgate circuits have not been used for evaluating quantum error correcting codes. The main result of Chapter 3 is that the surface code under coherent noise can be represented as a matchgate circuit.

2.4 Supervised machine learning

Machine learning is a framework of tasks such as predicting behavior of an unknown system only with previously known information. While such a general task cannot always be solved perfectly, there are several state-of-the-art methods which are expected to achieve it with a high accuracy. A framework of machine learning has been applied to many practical problems in both theoretical and experimental research in physics, such as the classification of the readout signals of experiments [71], simulation of quantum systems [72], classification of the phase of matter [73], data compression of the quantum state [74], and decoding in QEC [54–58].

In this thesis, we focus on a framework called supervised machine learning. Since a part of decoding problems can be formulated as a task of supervised machine learning, we expect that a decoder which is applicable to general settings can be constructed by using state-of-the-art methods in supervised machine learning.

In this section, we show a specific formalism of supervised machine learning for constructing a machine-learning-based decoder.

2.4.1 Framework of supervised machine learning

Though there are several specific formalisms of supervised machine leaning, we consider the following formalism in this thesis. We consider two finite and discrete variable spaces, feature space $\mathcal{X}$ and label space $\mathcal{Y}$. Let $\mathcal{D}$ be a joint probability distribution of a feature space $\mathcal{X}$ and a label space $\mathcal{Y}$. We sample $|T|$ pairs of features and labels $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}$ independently with the probability $\mathcal{D}(\boldsymbol{x}, \boldsymbol{y})$. A set of $|T|$ pairs $T := \{(x_i, y_i)\}_i$ is called a training data set. Our purpose is to construct a function $F : \mathcal{X} \to \mathcal{Y}$ using the training data set T, where F can predict $\boldsymbol{y}$ only from $\boldsymbol{x}$ with a high probability. The function F is called a prediction function. The accuracy of the prediction function F is given by

$$p_{\rm acc}(F) := \Pr_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\left[\boldsymbol{y} = F(\boldsymbol{x})\right]. \tag{2.100}$$

Since sampled $\boldsymbol{x}$ and $\boldsymbol{y}$ are not perfectly correlated in general, the maximal prediction accuracy is not necessarily unity. A prediction model $F_{\rm opt}$ is optimal if it satisfies

$$\Pr_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\left[\boldsymbol{y} = F_{\rm opt}(\boldsymbol{x})|\boldsymbol{x} = \boldsymbol{x}'\right] = \max_{\boldsymbol{y}'\in\mathcal{Y}} \Pr_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\left[\boldsymbol{y} = \boldsymbol{y}'|\boldsymbol{x} = \boldsymbol{x}'\right] \tag{2.101}$$

for an arbitrary feature $\boldsymbol{x}' \in \mathcal{X}$ such that $\Pr_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\left[\boldsymbol{x} = \boldsymbol{x}'\right] > 0$. The maximal accuracy is then given by

$$p_{\rm opt} := p_{\rm acc}(F_{\rm opt}). \tag{2.102}$$

2.4.2 Training process

The most vital and non-trivial factor of supervised machine learning is how to choose an accurate prediction function using the training data set. It is empirically known that the following process called *training* generates a high-performance prediction function in various cases.

We redefine the discrete label space $\mathcal{Y}$ as a set of $|\mathcal{Y}|$ discrete points in a continuous variable space $\mathcal{Y}'$. We call $\mathcal{Y}'$ as a prediction space. We choose a prediction model $\hat{M} : \mathcal{X} \times \Theta \to \mathcal{Y}'$, where Θ is a space of a set of parameters. An element $\boldsymbol{\theta} \in \Theta$ is called model parameters, which represent a configuration of the prediction model. We choose a loss function $L : \mathcal{Y}' \times \mathcal{Y} \to \mathbb{R}$, which represents how a predicted output $\boldsymbol{y}' \in \mathcal{Y}'$ is deviated from the correct label $\boldsymbol{y} \in \mathcal{Y}$. Thus, usually it is minimized if and only if $\boldsymbol{y} = \boldsymbol{y}'$. We choose an interpreting function $\xi : \mathcal{Y}' \to \mathcal{Y}$ to interpret a prediction $\boldsymbol{y}' \in \mathcal{Y}'$ as an element of $\mathcal{Y}$. We choose an optimizer $\hat{O}$ which suggests a preferable set of model parameters as a result of optimization using a training data set T and a loss function L, i.e., $\hat{O}(\hat{M}, L, T) \in \Theta$.

When a training data set T is given, we find a preferable set of model parameters $\boldsymbol{\theta}^*$ using the optimizer as $\boldsymbol{\theta}^* = \hat{O}(\hat{M}, L, T)$. This process is called training. We choose $M_{\boldsymbol{\theta}^*} : \mathcal{X} \to \mathcal{Y}$, where $M_{\boldsymbol{\theta}^*}(\boldsymbol{x}) := \xi(\hat{M}(\boldsymbol{x}, \boldsymbol{\theta}^*))$, as a prediction function. The accuracy of the prediction function $M_{\boldsymbol{\theta}^*}$ is given by

$$p_{\mathrm{acc}}(M_{\boldsymbol{\theta}^*}) = \Pr_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}[\boldsymbol{y} = M_{\theta^*}(\boldsymbol{x})]. \tag{2.103}$$

In principle, an optimizer $\hat{O}$ tries to minimize a total loss for a training data set, which is defined by

$$L(\boldsymbol{\theta}) := \frac{1}{|T|} \sum_{i=0}^{|T|-1} L(\hat{M}(\boldsymbol{x}_i, \boldsymbol{\theta}), \boldsymbol{y}_i). \tag{2.104}$$

The optimizer usually uses gradients of the total loss function according to a set of model parameters $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$ in order to find a set of model parameters $\boldsymbol{\theta}^*$ which minimizes the total loss function. For a stable training, it is required that $L(\boldsymbol{\theta})$ has smooth and computable gradient for model parameters at almost all the points. This is why the prediction space $\mathcal{Y}'$ should be continuous though what we want to predict is a discrete element of $\mathcal{Y}$.

We show a simple example. Suppose $\mathcal{X}$ and $\mathcal{Y}$ be discrete variable spaces $\mathcal{X} = \{0, 1\}^3$ and $\mathcal{Y} = \{0, 1, 2, 3\}^2$, respectively. We choose a linear prediction model such that $\hat{M}(\boldsymbol{x}, (\boldsymbol{a}, A)) = \boldsymbol{a} + A\boldsymbol{x}$, where $\boldsymbol{a} \in \mathbb{R}^3$ and A is a 2×3 real-valued matrix. It is equivalent to choosing a set of model parameters $\boldsymbol{\theta}$ as $\boldsymbol{\theta} = (a_0, a_1, a_2, A_{0,0}, A_{0,1}, \ldots, A_{1,2})$, Θ as a space where $(\boldsymbol{a}, A)$ spans, and $\mathcal{Y}' = \mathbb{R}^3$. We choose a rounding function $\xi(\boldsymbol{x})_i = \min(\max(\lfloor x_i + 0.5 \rfloor, 0), 3)$ as an interpreting function. We choose squared L2 distance between a prediction vector and a correct label vector $L(\boldsymbol{y}', \boldsymbol{y}) = ||\boldsymbol{y}' - \boldsymbol{y}||_2^2$ as a loss function. With these choices, we expect that a relation between $\boldsymbol{x}$ and $\boldsymbol{y}$, which are sampled from the distribution $\mathcal{D}$, is linearly fitted.

2.4.3 Vital factors in training process

Though we can achieve the optimal accuracy p_{opt} in ideal situations, there are several causes of degradation of accuracy in practice. In this subsection, we describe vital factors which determine the performance of the resultant prediction function.

The first factor is the choice of a prediction model $\hat{M}$. When a chosen prediction model $\hat{M}$ cannot represent any optimal prediction function by tuning $\boldsymbol{\theta}$, there is a gap between an achievable accuracy and the optimal one. This means that we can achieve the optimal accuracy only when any optimal prediction function is in $\{M_{\boldsymbol{\theta}} | \boldsymbol{\theta} \in \Theta\}$. How many functions can be represented with a chosen prediction model by tuning a set of parameters $\boldsymbol{\theta}$ is called a representation power of a prediction model. We should choose a prediction model with a sufficiently large representation power to achieve a required accuracy. Though we can enlarge

a representation power of the prediction model by increasing its degrees of freedom, i.e., the number of model parameters, this leads to an increase in the time spent for each prediction. Though it is hard to find a prediction model which can achieve both of a short prediction time and a large representation power, several state-of-the-art prediction models are known which are effective for a variety of tasks.

The second factor is the choice of a loss function L, an optimizer $\hat{O}$, and a time for the training process. The minimization of the total loss is usually formulated as an instance of non-linear optimization. Since non-linear optimization is known to be computationally hard in general, there is a trade-off relation between an achievable loss and the time for training. Though this minimization may take an exponential time to the number of model parameters, it is required only once before many runs of prediction. A required accuracy is achieved when we choose appropriate loss function and optimization algorithm, and pay a sufficient time for optimization.

The third is the size $|T|$ of the training data set. When the data set T consists of randomly drawn finite samples, the result of the minimization of the total loss depends on statistical fluctuation of samples in the training data set. When a given training data set is too small, a distribution expected from the training data set is deviated from the actual distribution $\mathcal{D}$, which leads to a degradation of the resultant accuracy. Thus, it is basically preferable to use as large a training data set as we can. If a task allows us to sample a training data set with any size, we should use a sufficiently large training data set so that we can achieve a required accuracy.

The final one is the design of a feature space $\mathcal{X}$, a label space $\mathcal{Y}$, a prediction space $\mathcal{Y}'$, and an interpreting function ξ. Since a raw input and output of the task does not necessarily coincide with the feature and label, these spaces and function should be chosen so that the resultant accuracy becomes high. Allowed and preferable choices of these spaces and the function are dependent on the task. Thus, understanding the properties of the task is essential to choose appropriate spaces and function. Since the degradation of the accuracy due to wrong choices of them cannot be retrieved by increasing an amount of computational resource, these choices should be carefully examined according to the task.

The achievable accuracy is mainly determined from these four factors. Though we cannot perfectly satisfy all of them in practical cases, we should examine these factors to construct a fast and reliable prediction function for a given task.

2.4.4 Summary and discussion

The framework of supervised machine learning provides instructions to construct a prediction function for estimating behavior of a given system using a previously known training data set. In the framework, we construct a prediction function by choosing a prediction model which is characterized by a set of model parameters,

and by choosing model parameters so that a total loss for the training data set is minimized. This process of minimization is called training. Since this framework is general, supervised machine learning has been widely applied to various problems including problems in the field of quantum information.

We reviewed four essential factors which determine the accuracy of the resultant prediction function in supervised machine learning: 1) the choice of a prediction model $\hat{M}$, 2) the choice of a loss function L, an optimizer $\hat{O}$, and a time for the loss minimization, 3) the preparation of the training data set of a size $|T|$, and 4) the choice of a feature space $\mathcal{X}$, a label space $\mathcal{Y}$, a prediction space $\mathcal{Y}'$, and an interpreting function ξ.

As detailed in Chapter 4, we discuss a general framework for constructing a decoder based on machine learning. In that discussion, we mainly focus on the last factor, the choices of the spaces and the interpreting function.

2.5 List of notations

Many of the symbols and definitions introduced in this chapter are used throughout this thesis. Here, we briefly summarize them for the reference. The notations about quantum states and operations are shown in Table 2.1. The notations about quantum error correction are shown in Table 2.2. The notations about efficiently simulatable classes of quantum circuits are shown in Table 2.3. The notations about supervised machine learning are shown in Table 2.4.

Table 2.1: The notations about quantum states and operations.

Symbol	Description
$\mathcal{P}_n$	n-qubit Pauli group ($\{\pm, \pm i\} \times \{I, X, Y, Z\}^{\otimes n}$).
ρ_M	Maximally entangled state ($\rho_M = \|\psi_M\rangle\langle\psi_M\|$ and $\|\psi_M\rangle = \frac{1}{\sqrt{2^n}} \sum_i \|i\rangle \otimes \|i\rangle$).
$(\mathcal{E} \otimes \mathrm{Id})(\rho_M)$	Channel state of a CP map $\mathcal{E}$, which gives a representation of CP maps by quantum states in doubled Hilbert space. See explanations around Eq. (2.7).
$\mathcal{F}(\mathcal{E})$	Entanglement fidelity of a CP map $\mathcal{E}$. See Eq. (2.11) for the definition.
$b(P)$	Binary representation of Pauli operator ($b : \mathcal{P}_n \to \{0,1\}^{2n}$). See Eq. (2.17) for the definition.
$c(P, P')$	Commutation relation of Pauli operators (0:commute, 1:anti-commute).
Λ	$2n \times 2n$ matrix for a symplectic product. The commutation relation can be represented as $c(P, P') = b(P)\Lambda b(P')^{\mathrm{T}}$.
$w(P)$	Weight of Pauli operators. The number of qubits to which P applies non-trivially.

Table 2.2: The notations about quantum error correction.

Symbol	Description
$\mathcal{S}$	Stabilizer generator set. The stabilizer formalism specifies a subspace spanned by $\{\|\psi\rangle\}$ such that $S\|\psi\rangle = \|\psi\rangle$ for all $S \in \mathcal{S}$. See Definition. 2.2.1.
$\mathcal{C}$	Logical channel. The CPTP map from initial k-qubit state to the logical space of recovered quantum state. We measure performance of quantum error correcting codes using the entanglement fidelity of logical channels.
$\boldsymbol{s}$	Syndrome ($\boldsymbol{s} \in \{0,1\}^{n-k}$). The outcome of the stabilizer measurements.
H_c	Check matrix ($(n-k) \times 2n$ binary matrix). When the error is represented by a Pauli operator P, the syndrome is given by $\boldsymbol{s}(\boldsymbol{e}) = H_c \Lambda \boldsymbol{e}^{\mathrm{T}}$, where $\boldsymbol{e} = b(P)$.
$[[n,k,d]]$	A code with n physical qubits, k logical qubits, and distance d.
G	Logical generator matrix ($2k \times 2n$ binary matrix). Its $2k$ rows correspond to binary representations of $2k$ independent logical operators.
$\boldsymbol{t}(\boldsymbol{s})$	Pure error ($\boldsymbol{t} : \{0,1\}^{n-k} \to \{0,1\}^{2n}$). A pure error satisfies $\boldsymbol{t}(\boldsymbol{s}(\boldsymbol{e})) \oplus \boldsymbol{e} \in \mathcal{L}$, where $\boldsymbol{e}$ is a binary representation of an arbitrary Pauli error.
$\boldsymbol{w}(\boldsymbol{e})$	Class of a physical error $\boldsymbol{e}$. Using a syndrome, a class, a logical generator, and a pure error, a binary representation of a Pauli error can be uniquely decomposed as in Eq. (2.44).
$\mathcal{L}$	Binary representations of all the undetectable Pauli operators.
$\mathcal{L}_{\boldsymbol{w}}$	A set of binary representations of physical errors which have a class $\boldsymbol{w}$ in the decomposition in Eq. (2.44).
$q_{\boldsymbol{s}}(\boldsymbol{w})$	The probability with which an occurred physical error has a class $\boldsymbol{w}$ conditioned on an observed syndrome $\boldsymbol{s}$.
$\boldsymbol{r}(\boldsymbol{s})$	Recovery operator for an observed syndrome $\boldsymbol{s}$. A decoder is an algorithm determining a recovery operator from a syndrome.
p_{L}	Logical error probability. In the case of a probabilistic Pauli noise model, see Eq. (2.36) for the definition. An optimal decoder satisfies Eq. (2.55).
p_{th}	Threshold value. If a physical error probability per qubit is smaller than p_{th}, we can achieve an arbitrary small logical error probability p_{L} by increasing the number of physical qubit.
$\lambda(p,d)$	Dropping rate. See Eq. (2.69) for its definition.

Table 2.3: The notations about Clifford circuits and matchgate circuits.

Symbol	Description
$\{C \mid \forall P \in \mathcal{P}_n, CPC^\dagger \in \mathcal{P}_n\}$	A set of Clifford operations. Clifford circuits are efficiently simulatable.
$a_i, a_i^\dagger$	Fermionic annihilation and creation operators.
$c_{2i} = a_i + a_i^\dagger, c_{2i+1} = (-i)(a_i - a_i^\dagger)$	Majorana fermionic operators. These operators are related to Pauli operators as in Eqs. (2.92) and (2.93).
$\boldsymbol{c} = (c_0, \ldots, c_{2n-1})^{\mathrm{T}}$	Array of Majorana fermionic operators
$Ce^{\boldsymbol{c}^{\mathrm{T}} A \boldsymbol{c}}$	Fermionic Gaussian state. A is a $2n \times 2n$ real-valued matrix, and C is a normalization factor.
$Ce^{\boldsymbol{c}^{\mathrm{T}} G \boldsymbol{c}}$	Fermionic Gaussian operation. G is a $2n \times 2n$ complex matrix, and C is a normalization factor.
M $(M_{ij} = \frac{i}{2}\mathrm{Tr}(\rho[c_i, c_j])/\mathrm{Tr}(\rho))$	Covariance matrix of a state ρ. Any density matrix of fermionic Gaussian state ρ is characterized by its covariance matrix M and norm $\mathrm{Tr}(\rho)$.
ρ_{FM}	Fermionic maximally entangled state. See Eq. (2.84) for the definition. Using the channel state of a fermionic Gaussian operation, a resultant state after applying the fermionic Gaussian operation to a fermionic Gaussian state can be efficiently computed. See Eqs. (2.88) and (2.89) for the scheme.

Table 2.4: The notations about supervised machine learning.

Symbol	Description
$\mathcal{X}$,$\mathcal{Y}$	Feature space and label space.
$\mathcal{D}$	Joint probability distribution of $\mathcal{X}$ and $\mathcal{Y}$. Our purpose in a task of supervised machine learning is to predict a label $\boldsymbol{y} \in \mathcal{Y}$ from a feature $\boldsymbol{x} \in \mathcal{X}$, where a pair $(\boldsymbol{x}, \boldsymbol{y})$ is sampled from $\mathcal{D}$.
T	Training data set ($T = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_i$). The training data set is a set of $\lvert T \rvert$ pairs of features and labels independently sampled from $\mathcal{D}$.
$\mathcal{Y}'$	Prediction space. We require the prediction space should be continuous. Elements in the label space $\mathcal{Y}$ are considered as discrete points in the prediction space $\mathcal{Y}'$.
Θ	A space of model parameters.
$\hat{M}$	Prediction model ($\hat{M} : \mathcal{X} \times \Theta \to \mathcal{Y}'$). A set of model parameters $\boldsymbol{\theta} \in \Theta$ represents a configuration of a prediction model.
ξ	Interpreting function ($\xi : \mathcal{Y}' \to \mathcal{Y}$).
L	Loss function ($L : \mathcal{Y}' \times \mathcal{Y} \to \mathbb{R}$). A measure of distance from a predicted value $\boldsymbol{y}' \in \mathcal{Y}'$ to the correct label $\boldsymbol{y} \in \mathcal{Y}$.
$\hat{O}$	Optimizer. An algorithm determining a preferable set of model parameters $\boldsymbol{\theta} \in \Theta$ using the training data set T, the loss function L, and the prediction model $\hat{M}$.
$M_{\boldsymbol{\theta}}$	Resultant prediction function ($M_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$). The function $M_{\boldsymbol{\theta}}$ is defined by $M_{\boldsymbol{\theta}}(\boldsymbol{x}) := \xi(\hat{M}(\boldsymbol{x}, \boldsymbol{\theta}))$, where $\boldsymbol{\theta} = \hat{O}(\hat{M}, L, T)$.

Chapter 3

Efficient simulation of quantum error correction under coherent error using matchgate

3.1 Background

Properties of quantum error correcting codes under a noise model, such as the logical error probabilities, the threshold values, and the dropping rates, are vital information when we design quantum devices. Therefore, performances of various QEC schemes have been evaluated under various assumptions of the noise models and degrees of rigor [28–45]. If we can simulate a quantum circuit of a quantum error correcting code under a given noise model in adaptive sense, we can sample a syndrome from accurate distribution, and can calculate the probability with which QEC succeeds conditioned on the sampled syndrome. Then, we can calculate the logical error probability for a given physical error probability within statistical error. The threshold value and dropping rate can be obtained from the behavior of the logical error probability as shown in Sec 2.2.2. Our goal is to represent noisy quantum circuits of a given quantum error correcting code under a given noise model as simulatable quantum circuits in adaptive sense.

By virtue of the Gottesman-Knill theorem [46, 75], shown in Theorem 2.3.1, we can efficiently simulate any Clifford circuit in adaptive sense. Since noiseless stabilizer measurements can always be represented as Pauli measurements, any noiseless quantum circuit of stabilizer codes can be represented as a Clifford circuit. When we can assume a noise model which only consists of probabilistic Clifford operations and Pauli measurements, the logical error probability can be efficiently and accurately estimated numerically [30–33]. The bit-flip noise and the depolarizing noise are examples of probabilistic Clifford noise. On the other hand, non-Clifford noise is also unavoidable in practical experiments [22–24]. Quantum error correcting codes under such a non-Clifford noise cannot be treated with the Gottesman-Knill theorem. Specifically, it is theoretically predicted that coher-

ent noise, which is non-Clifford and is caused, for example, by over rotation, can have negative effects on quantum error correction [76]. Therefore, massive effort has been made for evaluating the effect of noise coherence on the performance of quantum error correcting codes.

When the number of qubits is so small that we can fully simulate the whole state vector in classical computer, we can evaluate the performances using brute-force simulation. It was reported that evaluation of the performance of the $[[d^2, 1, d]]$ surface code at $d = 3$ can be achieved with thousands of hours with this method [43]. The computational cost of such a brute-force simulation can be relaxed using tensor-network [44, 45]. By contracting the network appropriately, the computational cost can be reduced to some extent. Furthermore, this method becomes efficient if we allow approximations. In the case of the concatenated codes, which is not a family of topological codes, there is an efficient method to analytically estimate the error threshold under non-Clifford noise [41]. We may approximate non-Clifford noise using Clifford operations and Pauli measurements for an efficient simulation [34–40]. Several non-Clifford noise models such as amplitude damping noise can be approximated with Clifford circuits with a certain accuracy. However, all of these methods sacrifice either accuracy, efficiency, or applicability to topological stabilizer codes. An efficient and accurate scheme to simulate quantum circuits of topological stabilizer codes under a non-Clifford noise is still lacking.

In this chapter, we show that the surface code under coherent noise can be efficiently and accurately simulated. The key idea in our scheme is the use of matchgate circuits for simulating quantum circuits of topological stabilizer codes. As we explained in Chapter 2, a gate set of the matchgate circuits contains coherent operations such as $e^{i\theta X}$ for $\theta \in \mathbb{R}$. We can expect that quantum circuits under coherent Pauli-X noise may be treated with matchgate circuits efficiently and accurately. On the other hand, the same gate set of the matchgate circuits does not contain even basic Pauli operations and measurements. For example, we cannot perform Pauli-Z operations on a single qubit, and cannot perform measurements on a qubit with Pauli Z-basis. Since stabilizer topological codes are usually described with Clifford circuits, it has been non-trivial whether we can describe quantum circuits of quantum error correcting codes with matchgate circuits.

First, in the next section, we show that one-dimensional (1D) repetition code, which is a 1D line of the surface code, under coherent noise can be represented as a matchgate circuit. Though the 1D repetition code is a classical code, it is still able to capture a necessary ingredient for fault-tolerant QEC, and hence was experimentally demonstrated as a building block for scalable fault-tolerant quantum computation [22]. We focus on the simulatability of the 1D repetition code since we use a similar technique when we will show that quantum circuits of the surface code under coherent errors can be represented as a matchgate circuit. As a noise model, we assume that coherent error occurs on both of the physical qubits and measurement processes. Using the above result, we numerically calculate logical error probabilities for various degrees of coherence of noise in the 1D repetition

code. We evaluate the effect of the coherence in noise on the threshold value and the dropping rate. As compared to the probabilistic noise model, we will find that the error threshold of the 1D repetition code becomes about one third when noise is fully coherent. We also confirm that the dropping rate becomes large when the threshold value decreases due to the noise coherence as expected from Eq. (2.68). The dependence of the error threshold on noise coherence is explained by using a leading-order analysis with respect to coherence terms of noise map.

We then proceeds to show, in Sec 3.3, that quantum circuits of the surface code under coherent errors can also be represented with a matchgate circuit. We can allow a noise model in which there are coherent X-type errors on the physical qubits, and probabilistic Pauli Z, Y and X errors on the physical qubits and the measurement process.

3.2 Efficient simulation of one-dimensional repetition code under coherent error

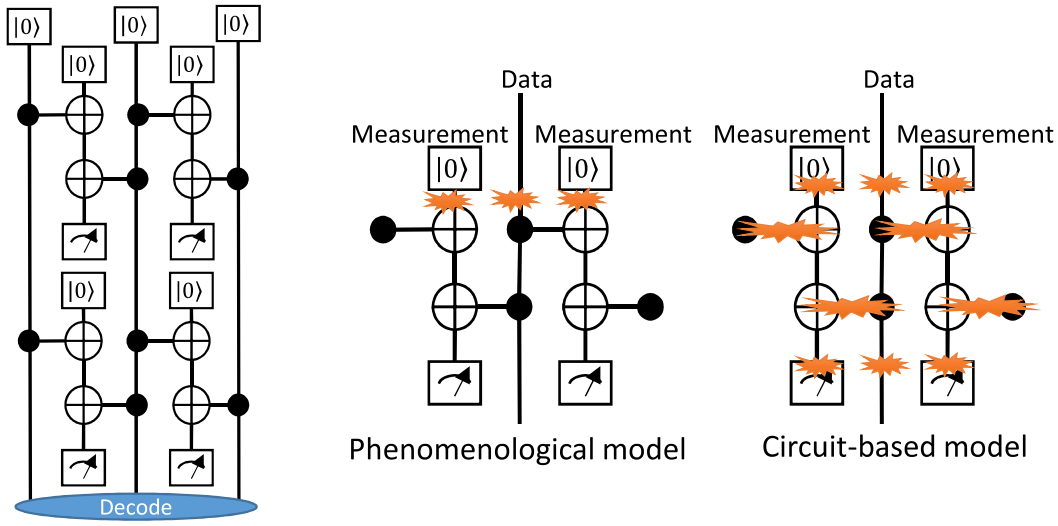
Before representing the surface code using matchgate circuits, we show a matchgate representation of the 1D repetition code.

3.2.1 Problem settings

One-dimensional repetition code

The 1D repetition code is a $[[d, 1, d]]$ classical error correcting code, which is capable of correcting only bit-flip noise. In the 1D repetition code, a classical bit $a_{\rm in}$ is encoded into physical $n(= d)$ qubits $\{|0\rangle^{\otimes n}, |1\rangle^{\otimes n}\}$, which are stabilized by $(n-1)$ operators $\mathcal{S} = \{Z_i Z_{i+1}\}_{i=0}^{n-2}$. Since we later consider a noise model which assumes errors also on the measurement processes, we explicitly consider auxiliary $(n-1)$ qubits for measurements, in addition to the n qubits forming the code. We call them as measurements qubits and data qubits, respectively. The stabilizer measurements are achieved using the $(n-1)$ measurement qubits, each of which monitors the parities of the neighboring data qubits, namely, the i-th measurement qubit is used for the stabilizer measurement with Pauli operator $Z_i Z_{i+1}$. In the 1D repetition code with repetitive parity measurement, the data qubits are repetitively measured for r cycle, and we obtain syndromes with $r(n-1)$ bits. The encoded bit is finally decoded using the final state of the n data qubits and $r(n-1)$ syndromes. We denote the decoded classical bit as $a_{\rm out} \in \{0, 1\}$. The quantum circuit of the 1D repetition code with repetitive parity measurements is shown in Fig. 3.1(a). Since this is a classical error correction, the logical error probability can be simply considered as the probability with which $a_{\rm out}$ is flipped from $a_{\rm in}$, i.e.

$$p_{\rm L} = {\rm Prob}(a_{\rm in} \neq a_{\rm out}). \tag{3.1}$$



(a) (b)

Figure 3.1: (a) The QEC circuit of the 1D repetition code with $n = 3$. (b) The error allocation of the phenomenological (left) and circuit-based (right) models.

Noise model and allocations

We consider a noise model such that coherent X-type errors occur on both of the data and measurement qubits. Since the 1D repetition code is capable of correcting only X-type error, we consider a CPTP map of a general single-qubit X-type noise, which is regarded as a mixture of the X-type unitary (fully-coherent) and stochastic (incoherent) noise:

$$\begin{aligned} \mathcal{E}(\rho, X) &= ce^{i\theta X}\rho e^{-i\theta X} + (1-c)\left((1-p)\rho + pX\rho X\right) \\ &= \frac{1+c}{2}e^{i\theta X}\rho e^{-i\theta X} + \frac{1-c}{2}e^{-i\theta X}\rho e^{i\theta X}, \end{aligned} \tag{3.2}$$

where θ is defined by $\cos\theta = \sqrt{1-p}$ and $\sin\theta = \sqrt{p}$. The parameter c $(0 \leq c \leq 1)$, which we call noise coherence, is a measure of coherence in the noise. We call the parameter p $(0 \leq p \leq 1)$ as the physical error probability since it can be understood as the probability with which the input state $|0\rangle$ is measured as the output state $|1\rangle$. When $c = 0$, the noise becomes a probabilistic bit-flip with the probability p. When $c = 1$, it becomes a unitary X-type operation.

We consider two types of noise allocation models [62] as shown in Fig. 3.1(b). In the case of the phenomenological model, a noise map $\mathcal{E}(\rho, X)$ is located on each of the data and measurement qubits at the beginning of each cycle. In the case of the circuit-based model, the noise map is located at each time step of preparation, gate operation, and measurement, on every qubit including the one that is idle at the time step. There, we assume that two-qubit noise map

$$\mathcal{E}_{\text{2qubit}}(\rho, X) := p_{XI}\mathcal{E}(\rho, X \otimes I) + p_{IX}\mathcal{E}(\rho, I \otimes X) + p_{XX}\mathcal{E}(\rho, X \otimes X) \tag{3.3}$$

acts on the output qubits after each controlled-Not (CNOT) operation. Since our noise models are symmetric over the bit values, we may choose $a_{\rm in} = 0$ to evaluate the logical error probability as $p_{\rm L} = \Pr(a_{\rm out} = 1)$.

Decoder

In order to calculate the logical error probability, we should choose a decoding algorithm. It is known that we can perform MD decoding for the 1D repetition code under probabilistic bit-flip noise ($c = 0$) both on the data and measurement qubits. This is done by constructing an instance of MWPM as we explained in Sec 2.2.2. This reduction is detailed in Ref. [22]. Here we give a brief explanation.

We define $s_{x,y} \in \{0, 1\}$ as an outcome of the syndrome measurement with the stabilizer operator $Z_x Z_{x+1}$ ($x = 0, \ldots n-2$) at the y-th ($y = 0, \ldots r-1$) cycle. We decode the final bit by directly measuring all the data qubits with noisy Pauli-Z basis. We denote the measurement outcome of the x-th data qubit ($x = 0, \ldots n-1$) as $d_x \in \{0, 1\}$. We define

$$s_{x,r} = d_x \oplus d_{x+1} \tag{3.4}$$

for $x = 0, \ldots n-2$. We also define parity bits $m_{x,y}$ as

$$m_{x,y} \quad = \quad s_{x,y-1} \oplus s_{x,y} \tag{3.5}$$

for $0 \leq x \leq n-2$ and $0 \leq y \leq r$, where syndrome values with undefined indices are assumed to be zero. We also define the parity of all $m_{x,y}$ as

$$m_{\rm parity} := \oplus_{x,y} m_{x,y}. \tag{3.6}$$

Then, we construct a binary vector with $((n-1)(r+1)+1)$ values as

$$s_{\rm node} := (m_{0,0}, m_{0,1}, \ldots, m_{n-2,r}, m_{\rm parity}). \tag{3.7}$$

When the noise is incoherent ($c = 0$) and we choose the phenomenological noise model, a bit-flip error occurs independently on each data and measurement qubit in each cycle. When a bit-flip occurs on a data qubit on the boundary, one bit of $m_{x,y}$ and a parity bit $m_{\rm parity}$ are flipped. When a bit-flip occurs on other data qubits, two neighboring bits $m_{x,y}$ and $m_{x+1,y}$ are flipped. When it occurs on a measurement qubit, two bits of $m_{x,y}$ and $m_{x,y+1}$ are flipped. We see an arbitrary single bit-flip occurs on data or measurement qubit at any cycle flips exactly two bits of $s_{\rm node}$. When we choose repetitive count as $r = n-1$, the situation is perfectly equivalent to that of the $[[2d_s^2 - 2d_s + 1, 1, d_s]]$ surface code at $d_s = n$ under bit-flip noise only on data qubits with uniform error probability p, which we demonstrated in Sec 2.2.2. Thus, we can calculate the most probable physical error by using the blossom algorithm. Let $\boldsymbol{r}(\boldsymbol{s}_{\rm all}) \in \{0, 1\}^n$ be the recovery operation

determined with this decoding scheme. Since we can assume that the recovered state $\boldsymbol{r}(\boldsymbol{s}_{\text{all}}) \oplus (d_0, \ldots, d_{n-1})$ is at least in the logical space,

$$\boldsymbol{r}(\boldsymbol{s}_{\text{all}}) \oplus (d_0, \ldots, d_{n-1}) \in \{0^n, 1^n\} \tag{3.8}$$

is guaranteed.

In this section, we use this decoder for all the settings. Since this decoder is not the MD decoder in other settings, we call this decoder as an MWPM decoder in order to distinguish it from the MD decoder. Eq. (3.8) is always guaranteed, regardless of noise models. Note that when we choose $c = 0$ and choose the circuit-based noise model, we can construct the MD decoder by setting appropriate non-uniform weights. This construction is discussed later.

3.2.2 Reduction to matchgate circuit

Interpreting noise maps as a sequence of complete Gaussian instruments

According to Theorem 2.3.3, we can simulate quantum circuits which consist of complete Gaussian instruments and a fermionic Gaussian initial state in adaptive sense. Thus, our purpose is to represent quantum circuits of the 1D repetition code under coherent noise as a sequence of complete Gaussian instruments. To this end, we start from showing that noise maps and noisy measurements can be represented as indexed CP maps on the data qubits, each of which has unit Kraus rank. Note that though we cannot obtain the index of the chosen probabilistic noise map in actual experiments, we can always assume that the corresponding measurement was virtually performed to obtain the index, for the purpose of simulation.

There are three types of operations on the data qubits in the quantum circuits of the 1D repetition code: single qubit noise on a data qubit, noisy measurements on data qubits, and decoding process. We identify them with indexed Kraus operators $K_\alpha^{(\beta)}$ corresponding to each CP map with unit Kraus rank, where α represents the description of the Kraus operator, and β is an outcome of its CP map. For clarity, we describe the case of the phenomenological model in the discussion below.

The first type is the single-qubit noise $\mathcal{E}$ given in Eq. (3.2). Its operation on the i-th qubits is equivalently described by Kraus operators

$$K_{\text{noise},i}^{(\phi)} = \sqrt{p(\phi)} e^{i\phi X_i}, \tag{3.9}$$

where $\phi \in \{\pm\theta\}$ and $p(\pm\theta) := \frac{1\pm c}{2}$.

The second type is the parity measurement on the i-th and $(i+1)$-th data qubits, which composed of a measurement qubit and two CNOT gates. Treating the noise map $\mathcal{E}$ on the measurement qubit as above, it can be represented by

$$K_{\text{parity},i}^{(s,\phi)} = \sqrt{p(\phi)} \frac{1}{2} (I + (-1)^s e^{-2i\phi} Z_i Z_{i+1}), \tag{3.10}$$

where $s \in \{0, 1\}$ is the output of the parity measurement.

The third type appears in an alternate description of the decoding process. As we described in Eq. (3.4), in the decoding process, we perform single-qubit noisy measurements on each data qubit, calculate the parities of the outcomes $d_x \oplus d_{x+1}$ for the MWPM decoder, and the decoded bit $a_{\rm out}$ is determined. Instead of describing these noisy direct measurements and classical computations, we use the following equivalent process. We apply map $\mathcal{E}$ on each data qubit. We perform ideal parity measurements on neighboring data qubits, whose Kraus operator is given by

$$K^{(s)}_{{\rm parity},i} = \frac{1}{2}(I + (-1)^s Z_i Z_{i+1}). \tag{3.11}$$

Note that these outcomes are equivalent to the parities $d_x \oplus d_{x+1}$ for all x. In the whole circuit of the 1D repetition code including the above ideal measurements, we denote the outcome of the k-th instrument as t_k, and a corresponding CP map as $\mathcal{E}^{(t_k)}_k(\rho) = K^{(t_k)}_k \rho (K^{(t_k)}_k)^\dagger$. Kraus operator $K^{(t_k)}_k$ which corresponds to the k-th CP map with outcome t_k should be written as one of the Kraus operators enumerated above, $K^{(\phi)}_{{\rm noise},i}$, $K^{(s,\phi)}_{{\rm parity},i}$, and $K^{(s)}_{{\rm parity},i}$. The probability of a sequence of outcomes $\boldsymbol{t}_k := t_k ... t_0$ is given by

$$\Pr(\boldsymbol{t}_k) = \Gamma(\boldsymbol{t}_k) := \langle 0^{\otimes n} | (\boldsymbol{K}^{(\boldsymbol{t}_k)})^\dagger \boldsymbol{K}^{(\boldsymbol{t}_k)} | 0^{\otimes n} \rangle \,, \tag{3.12}$$

where $\boldsymbol{K}^{(\boldsymbol{t}_k)} := K^{(t_k)}_k K^{(t_{k-1})}_{k-1} \dots K^{(t_0)}_0$. Let k_f be the index of the last ideal parity measurement, and define $\boldsymbol{t} := \boldsymbol{t}_{k_f}$. All the $r(n-1)$ bits of syndrome values and $(n-1)$ bits of parities are contained in $\boldsymbol{t}$. We denote these bits required for MWPM decoding as $\boldsymbol{s}_{\rm all}$. Based on the obtained syndrome $\boldsymbol{s}_{\rm all}$, we choose a recovery operation

$$R(\boldsymbol{s}_{\rm all}) = \prod_{i=1}^{n} X_i^{r_i(\boldsymbol{s}_{\rm all})}, \tag{3.13}$$

where $\boldsymbol{r}(\boldsymbol{s}_{\rm all}) = (r_0(\boldsymbol{s}_{\rm all}), \dots, r_{n-1}(\boldsymbol{s}_{\rm all}))$ is determined using the MWPM decoder. The recovered state $R(\boldsymbol{s}_{\rm all})\boldsymbol{K}^{(\boldsymbol{t})} |0\rangle^{\otimes n}$ is in the code space of the 1D repetition code, which should be written in the form

$$R(\boldsymbol{s}_{\rm all})\boldsymbol{K}^{(\boldsymbol{t})} |0\rangle^{\otimes n} = \alpha |0\rangle^{\otimes n} + \beta |1\rangle^{\otimes n} \,, \tag{3.14}$$

where $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = \Gamma(\boldsymbol{t})$. The decoded bit $a_{\rm out}$ is thus obtained by, for example, measuring the $(n-1)$-th qubit with Z-basis. The probability of a decoding failure conditioned on $\boldsymbol{t}$ is given by

$$\Pr(a_{\rm out} = 1, \boldsymbol{t}) = \Gamma_{\rm L}(\boldsymbol{t}) := \langle 0^{\otimes n} | (R(\boldsymbol{t})\boldsymbol{K}^{(\boldsymbol{t})})^\dagger \frac{I - Z_{n-1}}{2} R(\boldsymbol{t})\boldsymbol{K}^{(\boldsymbol{t})} | 0^{\otimes n} \rangle \tag{3.15}$$

$$= \langle 0^{\otimes n} | (\boldsymbol{K}^{(\boldsymbol{t})})^\dagger \frac{I - (-1)^{r_{n-1}(\boldsymbol{s}_{\rm all})} Z_{n-1}}{2} \boldsymbol{K}^{(\boldsymbol{t})} | 0^{\otimes n} \rangle \,, \tag{3.16}$$

$$\Pr(a_{\rm out} = 1 | \boldsymbol{t}) = \frac{\Gamma_{\rm L}(\boldsymbol{t})}{\Gamma(\boldsymbol{t})}. \tag{3.17}$$

By taking the average of $\Pr(a_{\rm out}=1|\boldsymbol{t})$ about the probability distribution of $\boldsymbol{t}$, we have

$$p_{\rm L}=\Pr(a_{\rm out}=1)=\langle \Pr(a_{\rm out}=1|\boldsymbol{t})\rangle_{\boldsymbol{t}}\,. \tag{3.18}$$

Since all the Kraus operators $K^{(\phi)}_{{\rm noise},i}$, $K^{(s,\phi)}_{{\rm parity},i}$, and $K^{(s)}_{{\rm parity},i}$ have a form of Eq. (2.98) or Eq. (2.99), we see these Kraus operators are fermionic Gaussian operators. If the initial state $|0\rangle^{\otimes n}$ were a fermionic Gaussian state, we could efficiently sample $\boldsymbol{t}$ with probability $\Gamma(\boldsymbol{t})$. If the operator $\frac{I-(-1)^{r_{n-1}(\boldsymbol{s}_{\rm all})}Z_{n-1}}{2}$ were also a fermionic Gaussian operation, we could calculate the conditioned probability $\Pr(a_{\rm out}=1|\boldsymbol{t})=\Gamma_{\rm L}(\boldsymbol{t})/\Gamma(\boldsymbol{t})$ efficiently. However, the initial state $|0\rangle^{\otimes n}$ and the final measurement operator $\frac{I-(-1)^{r_{n-1}(\boldsymbol{s}_{\rm all})}Z_{n-1}}{2}$ are not a fermionic Gaussian state and a fermionic Gaussian operation, respectively. For an efficient simulation, we need a further trick as follows. Note that while similar tricks in Refs. [68, 77, 78] might be employed, the following construction is much simpler and more efficient for our purpose. We add one ancillary qubit, indexed as the n-th, and corresponding Majorana fermionic operators c_{2n}, c_{2n+1}. We consider a quantum state

$$|\tilde{\psi}\rangle := (|0\rangle^{\otimes(n+1)}+|1\rangle^{\otimes(n+1)})/\sqrt{2}. \tag{3.19}$$

We can verify that $|\tilde{\psi}\rangle$ is a fermionic Gaussian state since its covariance matrix $\tilde{M}$ satisfies $\tilde{M}\tilde{M}^{\rm T}=I$. We suppose the case when we use the state $|\tilde{\psi}\rangle$ as an initial state instead of $|0\rangle^{\otimes n}$. Using the fact that $K^{(\boldsymbol{t}_k)}$ commutes with $\prod_{j=0}^{n}X_j$, we obtain

$$\begin{aligned}
\langle\tilde{\psi}|(\boldsymbol{K}^{(\boldsymbol{t}_k)})^\dagger\boldsymbol{K}^{(\boldsymbol{t}_k)}\otimes I_n|\tilde{\psi}\rangle &= \frac{1}{2}\sum_{i,j=0,1}\langle i|^{\otimes(n+1)}(\boldsymbol{K}^{(\boldsymbol{t}_k)})^\dagger\boldsymbol{K}^{(\boldsymbol{t}_k)}\otimes I_n\,|j\rangle^{\otimes(n+1)}\\
&= \sum_{j=0,1}\langle 0|^{\otimes(n+1)}(\boldsymbol{K}^{(\boldsymbol{t}_k)})^\dagger\boldsymbol{K}^{(\boldsymbol{t}_k)}\otimes I_n\,|j\rangle^{\otimes(n+1)}\\
&= \langle 0|^{\otimes n}(\boldsymbol{K}^{(\boldsymbol{t}_k)})^\dagger\boldsymbol{K}^{(\boldsymbol{t}_k)}\,|0\rangle^{\otimes n}\\
&= \Gamma(\boldsymbol{t}_k).
\end{aligned} \tag{3.20}$$

Since $\frac{I-(-1)^{r_{n-1}(\boldsymbol{s}_{\rm all})}Z_{n-1}Z_n}{2}$ also commutes with $\prod_{j=0}^{n}X_j$, we obtain

$$\langle\tilde{\psi}|(\boldsymbol{K}^{(\boldsymbol{t})})^\dagger\frac{I-(-1)^{r_{n-1}(\boldsymbol{s}_{\rm all})}Z_{n-1}Z_n}{2}\boldsymbol{K}^{(\boldsymbol{t})}|\tilde{\psi}\rangle = \Gamma_{\rm L}(\boldsymbol{t}). \tag{3.21}$$

We see $\frac{I-(-1)^{r_{n-1}(\boldsymbol{s}_{\rm all})}Z_{n-1}Z_n}{2}$ is a fermionic Gaussian operation for both $r_{n-1}(\boldsymbol{s}_{\rm all})\in\{0,1\}$. Thus, we see that the initial state is Gaussian, and all the CP maps in the quantum circuits are complete Gaussian instruments. This means we can represent any quantum circuit of the 1D repetition code under coherent noise as a matchgate circuit.

Interpretation in the case of the circuit-based noise model

In the case of the circuit-based model, the two-qubit noise map assumed after each CNOT gate contains a non-local X-type noise term $e^{iX_{\text{control}}X_{\text{target}}}$. We can convert this non-local noise to local noise by replacing it with a local noise preceding the CNOT gate U_{CNOT} since

$$\mathcal{E}(U_{\text{CNOT}}\rho U_{\text{CNOT}}^{\dagger}, X_{\text{control}}X_{\text{target}}) = U_{\text{CNOT}}\mathcal{E}(\rho, X_{\text{control}}I_{\text{target}})U_{\text{CNOT}}^{\dagger}. \tag{3.22}$$

Thus, to simulate the two-qubit noise map faithfully, we probabilistically place a single-qubit noise map $\mathcal{E}$ at

- the target qubit after the CNOT gate
- the control qubit after the CNOT gate
- the control qubit before the CNOT gate

with the probabilities p_{XI}, p_{IX} and p_{XX}, respectively. The single-qubit noises associated with state preparations and measurements for the measurement qubits are placed deterministically. The ones for the data qubits at the beginning of the decoding process are also placed deterministically.

With these replacements, we can consider a noise allocation of the circuit-based noise model as that of the phenomenological noise model, where there are additional one-qubit noise map $\mathcal{E}(\rho, X)$ on the data and measurement qubits. We can treat additional noise maps on the data qubits by increasing the length of the sequence of CP maps. When there are additional noise maps on the measurement qubits, we are allowed to use the same form of $K_{\text{parity},i}^{(s,\phi)}$ except that the probability mass function $p(\phi)$ is replaced by

$$p(k\theta) = \binom{N}{(N-k)/2}\left(\frac{1+c}{2}\right)^{(N-k)/2}\left(\frac{1-c}{2}\right)^{(N+k)/2} \tag{3.23}$$

for $k = \{-N, -N+2, ..., N-2, N\}$, where N is a number of noise maps on the measurement qubit. Thus, the sequence of CP maps in the case of circuit-based noise model can be treated as a variation of the phenomenological noise model.

3.2.3 Sampling scheme

Covariance matrix of initial state and channel states

For the numerical calculation, we need the covariance matrix $\tilde{M}$ of the initial states $|\tilde{\psi}\rangle$. The covariance matrix of the inital state $|\tilde{\psi}\rangle$ is obtained as follows.

$$\tilde{M} = \begin{pmatrix} 0 & 0 & & & & & & -1 \\ 0 & 0 & -1 & & & & & \\ & 1 & 0 & 0 & & & & \\ & & 0 & 0 & -1 & & & \\ & & & 1 & 0 & & & \\ & & & & & \ddots & & \\ & & & & & & 0 & -1 \\ & & & & & & 1 & 0 \\ 1 & & & & & & & 0 \end{pmatrix} \begin{matrix} & & & & & & 0 \\ & & & & & & 0 \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & 0 \\ & & & & & & 0 \\ & & & & & & 0 \end{matrix} \tag{3.24}$$

We also need the description of the channel state of the Gaussian CP maps. We calculated $2n \times 2n$ submatrices A, B and D of the $4n \times 4n$ covariance matrix of the channel state as described in Eq. (2.87), and calculated the norm of the channel state Γ_G for each CP map. For $K^{(\phi)}_{\text{noise},i} = \sqrt{p(\phi)}e^{i\phi X_i}$ with $p(\pm\theta) = \frac{1\pm c}{2}$, which is a noise operation on a data qubit, we have $\Gamma_G^{(t_k)} = p(\phi)$ and $A^{(t_k)} = D^{(t_k)} = 0$. The submatrix $B^{(t_k)}$ is calculated as $B^{(t_k)} = I + B'$, where B' has nonzero elements only for

$$\begin{pmatrix} B'_{2i-1,2i-1} & B'_{2i-1,2i} \\ B'_{2i,2i-1} & B'_{2i,2i} \end{pmatrix} = \begin{pmatrix} -1+\cos 2\phi & -\sin 2\phi \\ \sin 2\phi & -1+\cos 2\phi \end{pmatrix}. \tag{3.25}$$

For $K^{(s,\phi)}_{\text{parity},i} = \sqrt{p(\phi)}\frac{1}{2}(I + (-1)^s e^{-2i\phi} Z_i Z_{i+1})$, which represents a parity measurement on two qubits, we have $\Gamma_G^{(t_k)} = p(\phi)/2$. The matrix $A^{(t_k)}$ has nonzero elements only for

$$A_{2i,2i+1} = -A_{2i+1,2i} = -(-1)^s \cos 2\phi, \tag{3.26}$$

and $D^{(t_k)} = -A^{(t_k)}$. The matrix $B^{(t_k)}$ is calculated as $B^{(t_k)} = I + B''$, where B'' has nonzero elements only for

$$\begin{pmatrix} B''_{2i,2i} & B''_{2i,2i+1} \\ B''_{2i+1,2i} & B''_{2i+1,2i+1} \end{pmatrix} = \begin{pmatrix} -1 & (-1)^s \sin 2\phi \\ -(-1)^s \sin 2\phi & -1 \end{pmatrix}. \tag{3.27}$$

The form of $p(\phi)$ depends on the noise allocation model.

For $K^{(s)}_{\text{parity},i}$, we have $\Gamma_G^{(t_k)} = \frac{1}{2}$, and $A^{(t_k)}$, $B^{(t_k)}$ and $D^{(t_k)}$ are equivalent to those for the $K^{(s,\phi)}_{\text{parity},i}$ with $\phi = 0$.

For the final measurement $\frac{I-(-1)^{r_{n-1}(\boldsymbol{s}_{\text{all}})}Z_{n-1}Z_n}{2}$, we also have $\Gamma_G^{(t_k)} = \frac{1}{2}$, and the submatrices are equivalent to those for $K^{(s,\phi)}_{\text{parity},i}$ with $\phi = 0$ and $s = 1 - r(\boldsymbol{s}_{\text{all}})_{n-1}$.

Sampling process

We describe the scheme of sampling $\boldsymbol{t}$ and computing $\Gamma_{\rm L}(t)/\Gamma(t)$. The simulation can be divided into three processes.

Process 1 — Allocations of noise maps
We allocate local noise maps according to a given assumed noise allocation model. Then, we obtain a sequence of CP maps characterized by Kraus operators $\{K_k^{(t_k)}\}$.

Process 2 — Simulation of the circuit
We start the simulation from $(\tilde{M}, 1)$, which is formally denoted by $(M(\boldsymbol{t}_0), \Gamma(\boldsymbol{t}_0))$. Given $(M(\boldsymbol{t}_{k-1}), \Gamma(\boldsymbol{t}_{k-1}))$, the value of t_k is sampled from the probability $\Gamma(\boldsymbol{t}_k)/\Gamma(\boldsymbol{t}_{k-1})$. Then the updated pair $(M(\boldsymbol{t}_k), \Gamma(\boldsymbol{t}_k))$ is calculated by

$$M(\boldsymbol{t}_k) = A^{(t_k)} - B^{(t_k)}(M(\boldsymbol{t}_{k-1}) - D^{(t_k)})^{-1}(B^{(t_k)})^T, \tag{3.28}$$

$$\Gamma(\boldsymbol{t}_k) = \Gamma_G^{(t_k)}\Gamma(\boldsymbol{t}_{k-1})\sqrt{\det(M(\boldsymbol{t}_{k-1}) - D^{(t_k)})}, \tag{3.29}$$

where $A^{(t_k)}, B^{(t_k)}, D^{(t_k)}$ and $\Gamma_G^{(t_k)}$ are associated with the fermionic Gaussian operator $K_k^{(t_k)}$.

After repeating the $r(n-1)$ syndrome measurements, we perform noiseless parity measurement for each neighboring data qubits. As a result, we obtain $(r+1)(n-1)$ outputs of the parity measurements $\boldsymbol{s}_{\rm all}$ included in $\boldsymbol{t}$, and the final state of data qubits $(M(\boldsymbol{t}), \Gamma(\boldsymbol{t})) = (M(\boldsymbol{t}_{k_f}), \Gamma(\boldsymbol{t}_{k_f}))$.

Process 3 — Decoding
We determine the recovery operation $R(\boldsymbol{t}) = \prod_{i=0}^{n-1} X_i^{r_i(\boldsymbol{s}_{\rm all})}$ by using MWPM decoder. We then calculate $\Gamma_{\rm L}(\boldsymbol{t})/\Gamma(\boldsymbol{t})$ from Eqs. (3.20) and (3.21) as

$$\frac{\Gamma_{\rm L}(\boldsymbol{t})}{\Gamma(\boldsymbol{t})} = \frac{1 - (-1)^{r_{n-1}(\boldsymbol{s}_{\rm all})}M(\boldsymbol{t})_{2n,2n-1}}{2}. \tag{3.30}$$

Time efficiency and implementations

There are $O(nr)$ noise maps and syndrome measurements. Each of them takes at most $O(n^3)$ steps. Therefore, this scheme requires $O(n^4 r)$ computational time except that for decoding. The decoding time with the MWPM decoder is $O((nr)^{5/2})$. When we choose $r = O(n)$, both of them are $O(n^5)$. When we choose $r = n - 1$, the time for simulation per sample with single thread of Intel Core i7 6700 takes about 20 ms with the parameter $(n, p, c) = (15, 0.03, 0)$ in the circuit-based model. See Appendix D for the detailed implementation of the MWPM decoder.

3.2.4 Numerical results

The effect of coherence in noise on threshold value

We show the logical error probability $p_{\rm L}$ as a function of the physical error probability p under incoherent noise ($c = 0$) and fully coherent noise ($c = 1$) in Fig. 3.2.

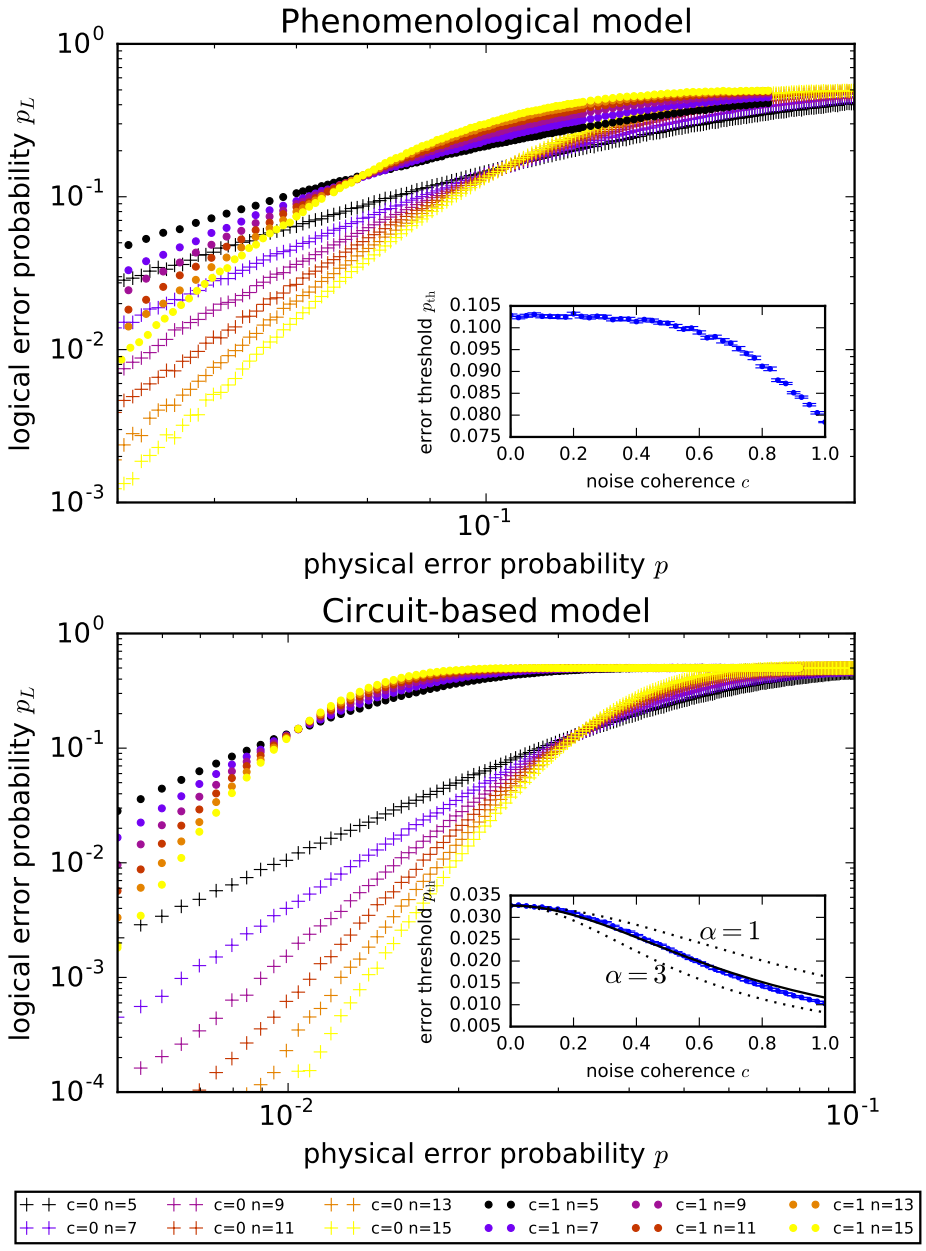


Figure 3.2: The logical error probability $p_{\rm L}$ is plotted as a function of the physical error probability p for the two patterns of noise allocation. Insets show the error threshold $p_{\rm th}$ as a function of the amount of the coherence c. The blue dots are numerical results. The solid black curve in the circuit-based model is estimated behavior from the simulation of small-size QEC circuits. The dotted curves are drawn as references.

We calculated 50,000 samples of decoding results for each physical error probability p, noise coherence c, distance n, and noise model. We have varied the number r of cycles according to n as $r = n - 1$. We assumed uniform error probability for two-qubit noise, i.e., $p_{XI} = p_{IX} = p_{XX} = \frac{1}{3}$, in the circuit-based model.

The logical error probability p_{L} is expected to be exponentially small in the number of the data qubits n as far as the physical error probability p is below a certain value p_{th}. According to the scaling ansatz described in Sec 2.2.2, we obtained the threshold values $p_{\mathrm{th}} = 10.34(1)\%$ for $c = 0$ and $7.87(2)\%$ for $c = 1$ in the phenomenological model, and $3.243(6)\%$ for $c = 0$ and $1.040(5)\%$ for $c = 1$ in the circuit-based model. Dependence of the error threshold p_{th} on the noise coherence c is shown in the insets of Fig. 3.2. We see that the error threshold p_{th} decreases as the noise coherence c increases. In particular, when we choose the circuit-based model as the noise model, the threshold value under fully coherent noise ($c = 1$) becomes about one-third compared with that of the incoherent limit ($c = 0$).

We also confirmed exponential decay of logical error probability p_{L} with code distance d below the threshold value. Recall that approximated equation is

$$p_{\mathrm{L}}(p, d) = a \left(b \frac{p}{p_{\mathrm{th}}} \right)^{(d+1)/2} \tag{3.31}$$

as in Eq. (2.68). We numerically investigate whether this approximation is still valid for the coherent noises by calculating the dropping rate $\lambda(p, d)$ defined by

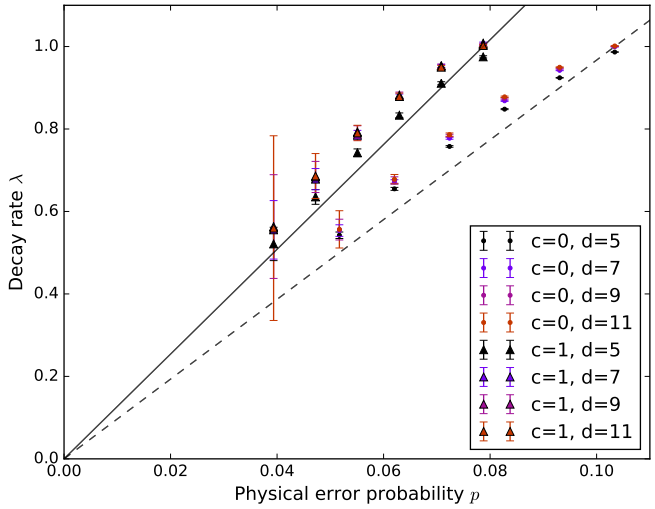
$$\lambda(p, d) := \frac{p_L(p, d+2)}{p_{\mathrm{L}}(p, d)} \sim \frac{p}{p_{\mathrm{th}}}. \tag{3.32}$$

The result is shown in Fig. 3.3 for the case of the coherent noise ($c = 1$) and the incoherent noise ($c = 0$). In both the phenomenological and the circuit-based model, Eq. (3.31) is satisfied with the same level of approximation regardless of the degree of coherence in the noises. We can conclude that the dropping rate is also worsened when the noise becomes fully coherent, and that the conjectured universal relation between the dropping rate and the threshold value [Eq. (2.68)] also holds true for coherent noise models.

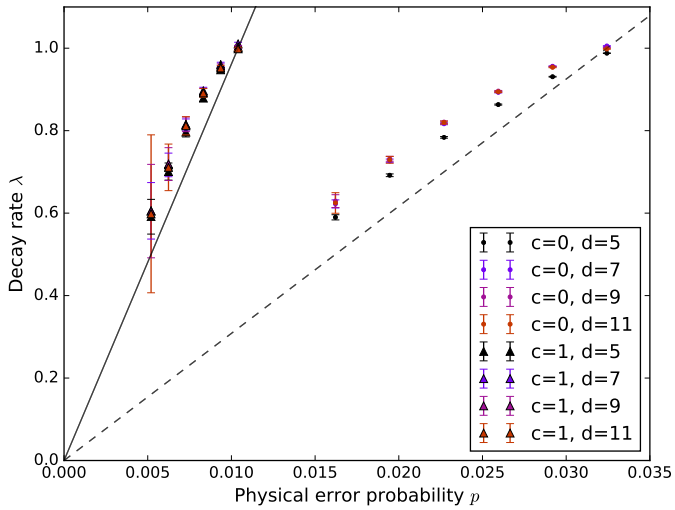
Improvement with non-uniform weighting in MWPM decoder

We applied the MWPM decoder assuming uniformly weighted edges for all of the settings. Though it works as the MD decoder when we choose the phenomenological noise model under incoherent noise ($c = 0$), that is not the case for the other settings. It is possible to find a weighted graph that faithfully reproduces the MD decoder in the case of the circuit-based model under incoherent noise ($c = 0$).

We constructed the weighted graph for the circuit-based model under incoherent noise. Note that each value of weights is approximated with its leading



(a) Phenomenological noise model



(b) Circuit-based noise model

Figure 3.3: The figures show the decay rate parameter $\lambda(p,d) := \frac{p_{\mathrm{L}}(p,d+2)}{p_L(p,d)}$ in terms of the physical error probability p, (a) for the phenomenological model and (b) for the circuit-based model. The cases with the incoherent noise ($c = 0$) and those with the fully coherent noise ($c = 1$) are marked with circles and triangles, respectively. The dashed and solid lines in each figure correspond to $\lambda = \frac{p}{p_{\mathrm{th}}}$ with p_{th} determined from the threshold behavior for $c = 0$ and 1, respectively. The black symbols are for $d = 5$, and the red ones are for $d = 11$.

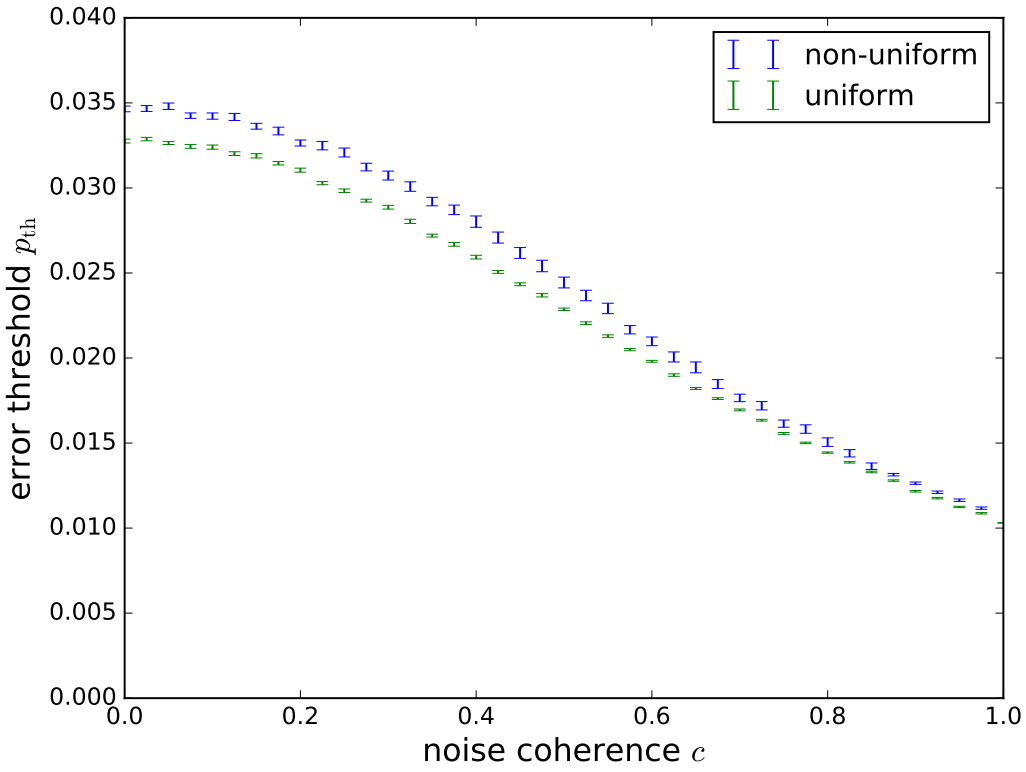


Figure 3.4: The error threshold $p_{\rm th}$ versus coherence c in the circuit-based model. Two plots correspond to uniformly weighted decoder and optimally weighted decoder.

$O(p)$ term for simplicity. These non-uniform weights were efficiently calculated using Warshall-Floyd algorithm [79]. We applied the decoder based on the constructed graph to the circuit-based model with various values of coherence c. The obtained error thresholds are shown in Fig. 3.4, together with the thresholds for the uniform-weight decoder. Compared to the error threshold using the uniform weight, the error threshold is improved for arbitrary values of c, but the amount of the improvement is small and the dependence of the error threshold on the noise coherence c is also similar.

3.2.5 Leading-order analysis of threshold value based on an ansatz

The dependence of the logical error probability and the threshold value on c can be explained with a leading-order analysis as follows. The noise map of Eq. (3.2)

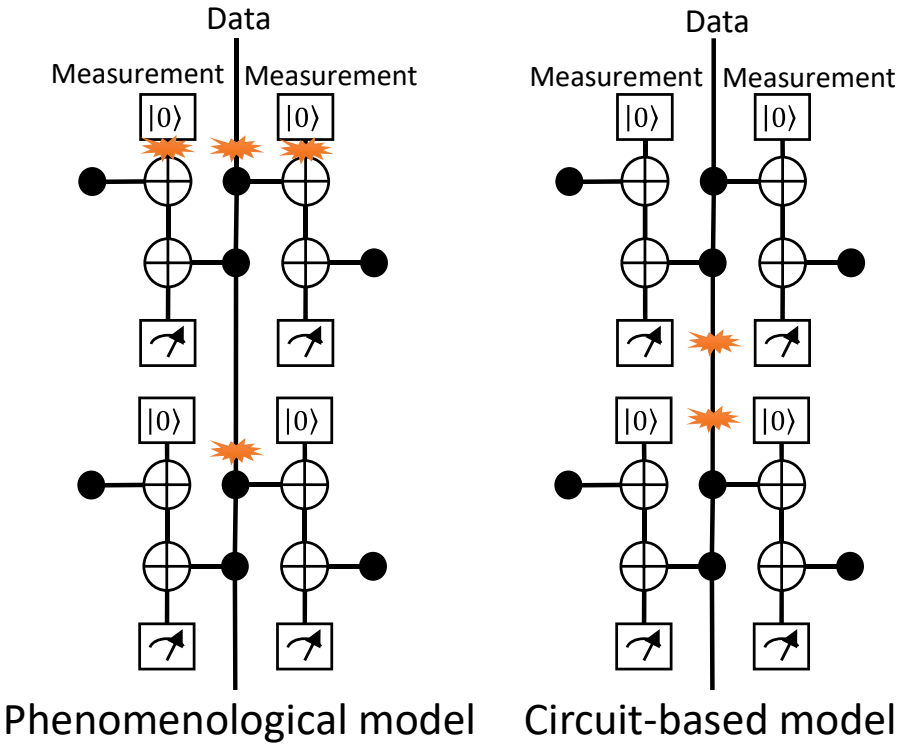


Figure 3.5: The sets of coherent error allocations which contribute to the probability distribution of the syndrome measurements.

can be rewritten as

$$\mathcal{E}(\rho, X) = (1-p)\rho + ic\sqrt{(1-p)p}(X\rho - \rho X) + pX\rho X. \tag{3.33}$$

We call the second term $ic\sqrt{(1-p)p}(X\rho - \rho X)$ as the coherence term. This term contributes to diagonal terms of the density matrix only through a concatenation of multiple noise maps. The correction to the diagonal terms after several cycles is written as even-order terms in $c\sqrt{(1-p)p}$. For $p \ll 1$, the leading order of the correction is $O(p)$ in the circuit-based model, while it is $O(p^2)$ in the phenomenological model since an error on a data qubit spreads to two measurement qubits before the next noise map is applied on the data qubit. For example, the product of the coherence terms of noise maps located in the positions shown in Fig. 3.5 contributes the correction. In the case of the phenomenological model, the leading term is proportional to c^4p^2, and its sign depends on the results of previous syndrome measurements. Such a noise leads to space-time correlations in the syndrome measurements. Since the decoder is not adapted to such correlations, the existence of coherence in noise is expected to result in a worse logical error probability. On the other hand, in the case of the circuit-based model, the leading term is proportional to c^2p, and it always increases the error probability. This

directly worsens the logical error probability and the error threshold.

In the case of the circuit-based model, we seek a more quantitative explanation of the behavior by proposing a heuristic ansatz as follows. We define an effective physical error probability of a data qubit per cycle $p_{\text{eff}}(p, c)$ as a marginal probability with which results of two measurement qubits neighboring a data qubit are flipped at a certain cycle from the results of the previous cycle. More precisely, we define $p_{\text{eff}}(p, c)$ as the marginal probability of $m_{x,y} = m_{x+1,y} = 1$, using the notation introduced in Sec 3.2.1. While $p_{\text{eff}}(p, c)$ may depend on the values of x and y, its leading term for small p is independent of x and y (except $y = 0$) and of the system size n.

The probability $p_{\text{eff}}(p, c)$ should be expanded for small p as

$$p_{\text{eff}}(p, c) = \beta(1 + \alpha c^2)p + O(p^2), \tag{3.34}$$

where α is constant and is independent of the system size n. This leading term can be analytically obtained as $\frac{8}{3}(1 + \frac{11}{6}c^2)p$, and thus $\alpha = 11/6$.

We assume that the logical error probability $p_{\text{L}}(p, c)$ can be well explained by the local increase of noise, i.e.,

$$p_{\text{L}}(p, c) = p_{\text{L}}((1 + \alpha c^2)p, 0). \tag{3.35}$$

Based on this ansatz, the error threshold under coherent noise $p_{\text{th}}(c)$ can be written as

$$p_{\text{th}}(c) \sim \frac{p_{\text{th}}(0)}{1 + \alpha c^2} \tag{3.36}$$

if $p_{\text{th}} \ll 1$. By using the analytically obtained value $\alpha = 11/6$, this ansatz gives the solid curve in the inset of Fig. 3.2, which is in good agreement with the accurate numerical results. We may expect that a similar leading-order ansatz also holds for the surface code, since it is a two-dimensional extension of the 1D repetition code. The factor α is also easily obtained by analytically calculating the effective bit-flip probability, and $p_{\text{th}}(0)$ for incoherent noises can be efficiently computed since only a few noise maps and qubits are relevant to the leading term. Therefore, the error threshold of the surface code under coherent noise will be estimated by the same approach.

3.3 Efficient simulation of surface code under coherent error

3.3.1 Problem settings

Here we extend our scheme of the efficient classical simulation using matchgate circuits to the surface code with a specific noise model including coherent errors.

We consider the $[[d^2, 1, d]]$ surface code for an odd number d. An example with $d = 5$ is shown in Fig 3.6.

The data qubits are located on the d^2 vertices of the colored faces. Each colored face corresponds to a measurement qubit. The measurement operator of each face is the product of Y or Z Pauli operators on the data qubits on its vertices. Note that assignments of Pauli operators to the vertices are modified from the code explained in Sec 2.2.2 for the compatibility to the modified Jordan-Wigner transformation of Eq. (2.90). We denote the stabilizer operator of the blue face at the r-th row as S_r^{b} $(r = 0, \ldots, d-2)$. We denote the stabilizer operators of the red faces in the c-th column $(c = 0, \ldots, d-2)$ as $\{S_i^{\mathrm{r},c}\}$ $(i = 0, \ldots, d-2)$, and that of the green face as $S^{\mathrm{g},c}$. The rules for assigning the index i will be explained later. For each cycle of syndrome measurement, the stabilizer operators as observables are measured via controlled gates and Z-basis measurement on the measurement qubits.

We consider an error model in which the code truly works as a fully quantum code, namely, including both X- and Z-type errors. More specifically, we assume the following phenomenological noise model. For each cycle, one of the following three types of errors occurs on each data qubit probabilistically, the Pauli Y error, the Pauli Z error, and an X-type coherent error as in Eq. (3.2). We assume that the measurement qubits also suffer from the three types of errors probabilistically, the Pauli X, Y, and Z errors. We also assume, for simplicity, that the syndrome measurement in the final cycle is error-free. This noise model can be considered as the phenomenological model with added Pauli Y and Z noises on both types of qubits, while limiting the X-type coherent errors to the data qubits. Apparently, such an error model including the coherent noise cannot be treated with the method based on the Gottesman-Knill theorem.

In contrast to the 1D repetition code in Sec 3.2 where the logical error probability is the only parameter of interest, we evaluate the performance of the quantum code with the entanglement fidelity of the logical channel. Since we assumed that the final syndrome is correct, the whole circuit including a recovery operation can be viewed as a one-qubit logical channel $\mathcal{C}$ on the logical qubit space spanned by $\{|0_L\rangle, |1_L\rangle\}$. Thus, it is enough for evaluating the performance of the code to compute the channel state $\rho(\mathcal{C})$ of the logical channel.

3.3.2 Reduction to matchgate circuit

Since the fermionic representation depends on the order of the data qubits, we assign numbers $0, 1, \ldots, d^2 - 1$, where $n = d^2$, to the data qubits in the order as shown in the gray boxes in Fig 3.6. We also assume that the ancillary qubit is the

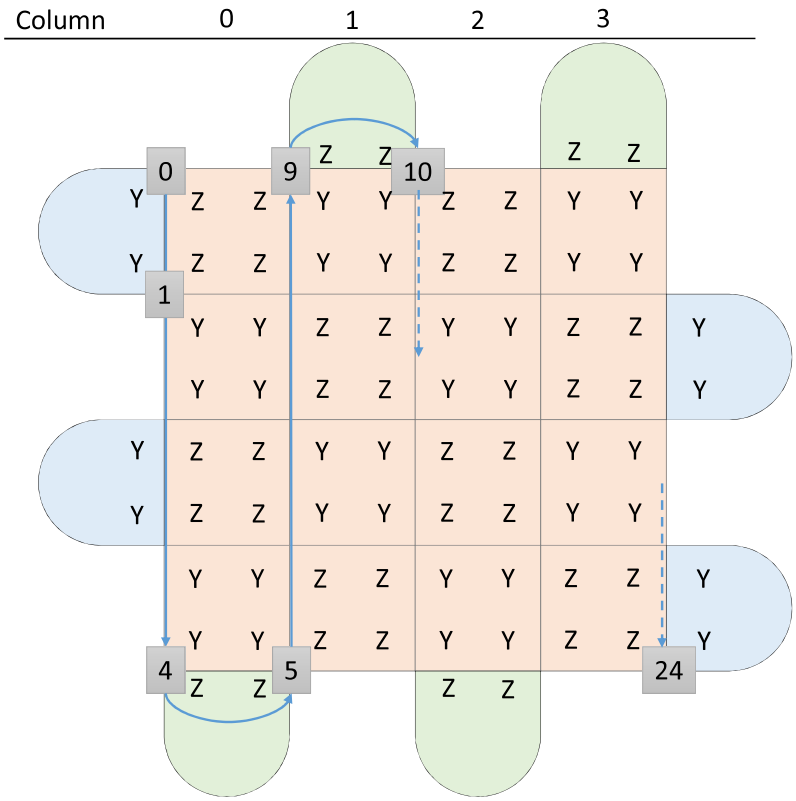


Figure 3.6: The figure shows the qubit allocation of the surface code in the distance $d = 5$. This architecture consists of d^2 data qubits and $d^2 - 1$ measurement qubits.

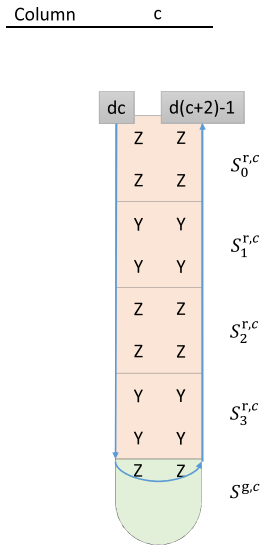


Figure 3.7: The figure shows a line of the surface code.

n-th data qubit. Recall that the Majorana fermionic operators can be chosen as

$$c_{2i} = \left(\prod_{j=0}^{i-1} X_j\right) Z_i \tag{3.37}$$

$$c_{2i+1} = \left(\prod_{j=0}^{i-1} X_j\right) Y_i, \tag{3.38}$$

where $i = 0, \ldots, n-1$. While the stabilizer operators for the blue and green faces are quadratic, those for the red faces are not. Our idea is to introduce a new equivalent set of stabilizer operators which are all quadratic. Let us consider the stabilizer operators in the c-th column, $\{S_0^{r,c}, \ldots, S_{d-2}^{r,c}, S^{g,c}\}$. With an appropriate rotation, each column can be considered as a part shown in Fig 3.7. We assign the index of the stabilizer operators of the red faces in the column as shown in the figure. Let us introduce new red stabilizer operators defined from $\{S_0^{r,c}, \ldots, S_{d-2}^{r,c}, S^{g,c}\}$ as

$$\tilde{S}_i^{r,c} = (\prod_{j=i}^{d-2} S_j^{r,c}) S^{g,c} \tag{3.39}$$

for $i = 0, \ldots, d-2$. Since they are explicitly written in the form

$$W_{dc+j} X_{dc+j+1} \cdots X_{d(c+2)-2-j} W_{d(c+2)-1-j}, \tag{3.40}$$

where W is either Z or Y, they are all quadratic in Majorana fermionic operators. The set $\{S_0^{r,c}, \ldots, S_{d-2}^{r,c}, S^{g,c}\}$ can be conversely expressed in terms of

$\{\tilde{S}_0^{\mathrm{r},c}, \ldots, \tilde{S}_{d-2}^{\mathrm{r},c}, S^{\mathrm{g},c}\}$ as

$$S_i^{\mathrm{r},c} = \begin{cases} \tilde{S}_i^{\mathrm{r},c} \tilde{S}_{i+1}^{\mathrm{r},c} & \text{if } i < d-2 \\ \tilde{S}_{d-2}^{\mathrm{r},c} S^{\mathrm{g},c} & \text{if } i = d-2 \end{cases}. \tag{3.41}$$

This indicates that $\{S_0^{\mathrm{r},c}, \ldots, S_{d-2}^{\mathrm{r},c}, S^{\mathrm{g},c}\}$ and $\{\tilde{S}_0^{\mathrm{r},c}, \ldots, \tilde{S}_{d-2}^{\mathrm{r},c}, S^{\mathrm{g},c}\}$ are equivalent sets as stabilizer generators. To be precise, let us introduce the projector onto the eigenspace of a stabilizer operator S associated with syndrome bit s by

$$P(S, s) := \frac{1}{2}(I + (-1)^s S). \tag{3.42}$$

Then, it follows that

$$P(S^{\mathrm{g},c}, s^{\mathrm{g}}) \prod_{i=0}^{d-2} P(S_i^{\mathrm{g},c}, s_i^{\mathrm{r}}) = P(S^{\mathrm{g},c}, s^{\mathrm{g}}) \prod_{i=0}^{d-2} P(\tilde{S}_i^{\mathrm{g},c}, \tilde{s}_i^{\mathrm{r}}) \tag{3.43}$$

if

$$s_i = \begin{cases} (\tilde{s}_i^{\mathrm{r}} \oplus \tilde{s}_{i+1}^{\mathrm{r}}) & \text{if } j < d-2 \\ (\tilde{s}_{d-2}^{\mathrm{r}} \oplus s^{\mathrm{g}}) & \text{if } j = d-2 \end{cases}. \tag{3.44}$$

This implies that the syndrome measurement of $\{S_0^{\mathrm{r},c}, \ldots, S_{d-2}^{\mathrm{r},c}, S^{\mathrm{g},c}\}$ can be equivalently done by those of $\{\tilde{S}_0^{\mathrm{r},c}, \ldots, \tilde{S}_{d-2}^{\mathrm{r},c}, S^{\mathrm{g},c}\}$ followed by calculation according to Eq. (3.44).

Now we will show that the quantum error correction circuit can be efficiently simulated to compute $\rho(\mathcal{C})$. First, we show that the initial state $|\phi_{\mathrm{init}}\rangle$ is a fermionic Gaussian state. The projection operator to the code space is given by

$$\begin{aligned} \mathcal{P}_{\mathrm{C}} &= \left(\prod_{S \in \{S_i^{\mathrm{b}}\} \cup \{S_i^{\mathrm{r},c}\} \cup \{S^{\mathrm{g},c}\}} P(S, 0) \right) \\ &= \left(\prod_{S \in \{S_i^{\mathrm{b}}\} \cup \{\tilde{S}_i^{\mathrm{r},c}\} \cup \{S^{\mathrm{g},c}\}} P(S, 0) \right), \end{aligned} \tag{3.45}$$

which is a fermionic Gaussian operator. We consider the following logical Z and Y operators,

$$L_Z = Z_{n-d} \left(\prod_{i=n-d+1}^{n-1} X_i \right), \tag{3.46}$$

$$L_Y = Y_{d-1} \left(\prod_{i=d}^{n-1} X_i \right), \tag{3.47}$$

and choose $|0_L\rangle$ and $|1_L\rangle$ to satisfy $L_Z|0_L\rangle = |0_L\rangle$, $L_Z|1_L\rangle = -|1_L\rangle$, and $L_Y|0_L\rangle = i|1_L\rangle$. Since $|\phi_{\text{init}}\rangle$ is stabilized by $L_Z Z_n$, $-L_Y Y_n$, and the d^2-1 stabilizer operators, we have

$$|\phi_{\text{init}}\rangle\langle\phi_{\text{init}}| = \mathcal{P}_{\text{C}} P(L_Z Z_n, 0) P(-L_Y Y_n, 0). \tag{3.48}$$

Since $L_Z Z_n$ and $L_Y Y_n$ are quadratic, $|\phi_{\text{init}}\rangle\langle\phi_{\text{init}}|$ is a fermionic Gaussian operator, and hence $|\phi_{\text{init}}\rangle$ is a fermionic Gaussian state.

Simulation of the execution of the circuit is done as follows. The error on each data qubit is sampled, and if it is X-type, the state can be directly updated to a new fermionic Gaussian state since the X-type error operator is a fermionic Gaussian operator. Pauli Z and Y operator is not a fermionic Gaussian operator, and hence it cannot be directly applied. When we implement Z error on the i-th qubit, we apply $Z_i Z_n$ instead, which is a fermionic Gaussian operator, and the updated state can be calculated. We apply $Y_i Z_n$ for Y error. The appearance of Z_n should be recorded, and will be compensated in the final stage. A syndrome measurement of a stabilizer operator S on a current fermionic Gaussian state $|\phi\rangle$ without any error is done by calculating

$$p_{s'} := \langle\phi|P(S,s')|\phi\rangle \tag{3.49}$$

for $s' \in \{0,1\}$, and sampling s' accordingly, and then calculating the updated state

$$\frac{P(S,s')|\phi\rangle}{\sqrt{\langle\phi|P(S,s')|\phi\rangle}}. \tag{3.50}$$

As we have seen, we are allowed to use $\tilde{S}_i^{\text{r},c}$ instead of $S_i^{\text{r},c}$, and to compute the syndrome for $\{S_i^{\text{r},c}\}$ through Eq. (3.44), which assures that $P(S,s')$ is always a fermionic Gaussian operator. As for the Pauli error on the measurement qubit, it can be equivalently translated to a composition of the following operations: a bit-flip of the computed syndrome (for X and Y), and Pauli errors on the data qubit (for Y and Z). After all the cycles are executed, the recovery operator R is calculated using the sampled syndrome values. R can be written as a product of Pauli operators and hence computed by possible inclusion of Z_n.

After a single run of simulation, we obtain a final fermionic Gaussian state $|\phi_{\text{final}}\rangle$ and the record of the number w of applications of Z_n. If w is even, the fermionic Gaussian state is an accurate sample of the desired state $\rho(\mathcal{C})$. If w is odd, $Z_n|\phi_{\text{final}}\rangle$ is an accurate sample of $\rho(\mathcal{C})$. Hence, all we need is to convert the description of $|\phi_{\text{final}}\rangle$ as an $(n+1)$-qubit fermionic Gaussian state to that as a two-qubit density operator. We choose the Pauli operators for the logical qubit as L_Z and L_Y defined in Eqs. (3.46) and (3.47), $L_I := I^{\otimes n}$, and

$$L_X = -iL_Y L_Z = -Y_{d-1}\left(\prod_{i=d}^{n-d-1} X_i\right) Y_{n-d}. \tag{3.51}$$

The density operator is then decomposed as

$$|\phi_{\text{final}}\rangle\langle\phi_{\text{final}}| = \sum_{W,W'} \frac{1}{4} A_{W,W'} L_W W'_n \tag{3.52}$$

$$A_{W,W'} := \langle\phi_{\text{final}}|L_W W'_n|\phi_{\text{final}}\rangle, \tag{3.53}$$

where $W, W' \in \{I, X, Y, Z\}$ and $A_{I,I} = 1$. The coefficient $A_{X,X}$ can be obtained as follows. The final state $|\phi_{\text{final}}\rangle$ is a $+1$ eigenstate of all the stabilizer operators. The operator L_X can be transformed to $X^{\otimes n}$ by multiplying all the $(d-1)$ blue stabilizers, all the $(d-1)$ green stabilizers, and the $(d-1)^2/2$ red stabilizers composed of Pauli Zs. We thus have $L_X X_n |\phi_{\text{final}}\rangle = X^{\otimes(n+1)} |\phi_{\text{final}}\rangle$. Since $X^{\otimes(n+1)}$ commutes with all the fermionic Gaussian operators and $|\phi_{\text{init}}\rangle$ is a $+1$ eigenstate of $X^{\otimes(n+1)}$, we have $X^{\otimes(n+1)} |\phi_{\text{final}}\rangle = |\phi_{\text{final}}\rangle$, and hence $A_{X,X} = 1$. This also implies $A_{W,W'} = 0$ if $L_W W'_n$ anti-commutes with $L_X X_n$. The remaining 6 non-trivial coefficients are expectation values of $L_X, X_n, L_Y Y_n, L_Y Z_n, L_Z Y_n$, and $L_Z Z_n$ which are all expectation values of fermionic Gaussian operators. These values can be calculated from the fermionic Gaussian state description of $|\phi_{\text{final}}\rangle$. We thus obtain the density operator $|\phi_{\text{final}}\rangle\langle\phi_{\text{final}}|$.

An accurate sample of $\rho(\mathcal{C})$ is then given by applying the correction Z_n^w. The $\rho(\mathcal{C})$ is calculated as

$$\rho(\mathcal{C}) = \text{ave}(Z_n^w |\phi_{\text{final}}\rangle\langle\phi_{\text{final}}| Z_n^w), \tag{3.54}$$

where ave$(\cdot)$ represents averaging function over the samples. The entanglement fidelity $F(\mathcal{C})$ is then computed as

$$\begin{aligned} F(\mathcal{C}) &:= \langle\phi_{\text{init}}|\rho(\mathcal{C})|\phi_{\text{init}}\rangle \\ &= \frac{1}{4}(1 + \text{tr}(\rho(\mathcal{C})L_X X_n) - \text{tr}(\rho(\mathcal{C})L_Y Y_n) + \text{tr}(\rho(\mathcal{C})L_Z Z_n)) \\ &= \frac{1}{4}(1 + \text{ave}((-1)^w(1 - A_{Y,Y}) + A_{Z,Z})). \end{aligned} \tag{3.55}$$

3.3.3 Extension of simulatable noise model

We would like to introduce another fully quantum error correcting circuit which is efficiently simulated by our method. The circuit is a modification of the previous circuit. It uses the same surface code, but each of the new red syndromes $\{\tilde{S}_i^{\text{r},c}\}$ is *actually* measured through a measurement qubit, instead of $\{S_i^{\text{r},c}\}$. For this circuit, we can allow X-type coherent errors on measurement qubits, just as in the phenomenological noise model in the case of the 1D repetition code. In this case, the X-type coherent error can be absorbed by replacing the measurement operator $P(S, s)$ with

$$P'(S, s, \theta) := \frac{1}{2}(I + (-1)^s e^{-2i\theta} S), \tag{3.56}$$

where θ is a rotation angle dictated by the X-type coherent error. Since all the stabilizer operators are quadratic terms of Majorana fermionic operators in the modified circuit, all of these operators are fermionic Gaussian operators.

The second example, i.e., the noise model described above, shows that the 1D repetition code in Sec 3.2 can be extended to a fully quantum code with no compromise on the noise model. Hence the applicability of our method relies neither on the simple structure of the repetition code nor on the absence of Z and Y errors. On the other hand, the comparison between the two examples of the surface code reveals an interesting trade-off. In the second example, the new syndrome measurements are non-local in the column direction, and only local in the row direction. In other words, it may be regarded as a 1D circuit with local stabilizer measurements. The first example, the noise model employed in Sec 3.3.1, is a true 2D circuit, but the efficient simulation seems to be possible only when the measurement qubits suffer no coherent errors. It is an open problem whether we can achieve both of them, efficient simulation of a truly 2D quantum error correction circuit with local syndrome measurement under coherent errors on both the data and measurement qubits.

3.4 Conclusion

We constructed an efficient and accurate scheme for calculating a logical error probability of the 1D repetition code and the surface code under coherent noise. We used matchgate circuits for simulating the noisy quantum circuits of these codes.

We have calculated the error threshold under coherent noise in terms of the physical error probability p and the noise coherence c. When noise becomes fully coherent, we found that the threshold value of the 1D repetition code becomes one-third compared with that in the case of the incoherent noise. We also numerically showed that the dropping rate of the logical error probability shows the same behavior. This implies that the conjectured relation between the dropping rate and the threshold value [Eq. (2.68)] holds also for coherent noise models.

Then, we showed that the surface code under coherent noise can be also represented as a matchgate circuit. This is the first example of efficiently and accurately simulatable quantum circuits of topological stabilizer codes under a noise model which assumes all X-, Y- and Z-Pauli errors and a part of them is a continuously parametrized coherent error.

We have also proposed a leading-order ansatz for the estimation of the error threshold under coherent noise, and found that it gives good approximation of the accurate numerical results in the case of the 1D repetition code. This suggests that the effect of any type of coherent noise on the surface code will be assessed by an analogous ansatz, which can be calculated easily.

Our results imply that several future studies are possible. The proposed simu-

lation scheme is the first example which provides efficient and accurate evaluation of the topological stabilizer codes except those based on the Gottesman-Knill theorem. In this chapter, we have shown two important examples, but there may still be more examples which are efficiently simulated using matchgate circuits. For example, recently, another scheme to simulate the surface code under coherent noise using matchgate circuits was proposed [80]. In this work, simulation of the surface code under different types of coherent noise is enabled by representing each physical qubit with four Majorana fermionic operators. More generally, it is also important to find general conditions that stabilizer codes and noise models can be represented as matchgate circuits.

Our results are also useful for evaluating applicability of other efficient but not accurate simulation schemes based on approximations. The obtained accurate error thresholds will serve as a reference to test the accuracy of approximated or heuristic schemes for simulating non-Clifford noise. For example, we have used the accurate results as a benchmark for the leading-order ansatz.

It is also interesting to investigate what types of noise can be well approximated with matchgate circuits. As many types of noises are approximated with Clifford circuits, there may be practical noise models which can be efficiently approximated with matchgate circuits, but not with Clifford circuits.

In the case of the Clifford circuit, it is known that quantum circuits dominated by Clifford gates but contain a few T gates, which is not a Clifford gate, can be simulated with a cost increase exponential to the number of the T gates [81]. Though the time for simulation grows exponentially, the cost is expected to be much smaller than the full simulation of the general quantum circuits. We may find a matchgate analog of this simulation. If there is such a framework, we can simulate quantum circuits under noise dominated by matchgate operations with a far smaller cost than that of the full simulation. This enables us to simulate a noisy quantum circuit including a few non-matchgate operations with a practical computational cost.

An architecture of topological fault-tolerant quantum computation [82] explicitly utilizes Majorana fermions as elements of computation. A fermionic Gaussian operation is understood as a dynamics with Hamiltonian which consists only of quadratic terms of Majorana fermionic operators. In such a system, dynamics with such a Hamiltonian with quadratic terms is expected to be a dominant factor of the physical error. Thus, using the framework of matchgate circuits may be the best way to evaluate the performances of quantum error correcting codes for topological fault-tolerant quantum computation under a practical noise.

Chapter 4

General framework for constructing near-optimal machine-learning-based decoder of the topological stabilizer codes

4.1 Background

The performances of quantum error correcting codes, such as the logical error probability, the threshold value, and the dropping rate, depend on the performance of a chosen decoder. It is known that carrying out the optimal decoding takes a time that grows exponentially to the number of physical qubits n except few cases [27]. According to a proposed architecture of fault-tolerant quantum computation, time for decoding is limited [32, 61]. Therefore, we cannot use the optimal decoding in general. As we saw in an example in Sec 2.2.2, another method is to use one of the most likely physical errors that are consistent with the observed syndrome as a recovery operation. This method is called the minimum-distance (MD) decoder. Though the MD decoder is known to be near-optimal, construction of an efficient MD decoder is known only for limited cases of quantum codes and noise models [21]. In the case of the surface code under correlated noise models, by ignoring the correlation of bit-flip and phase-flip errors, we can use a decoder based on minimum-weight perfect matching (MWPM) as an efficient decoder, while its performance is low. This approach is called the MWPM decoder. As for the other topological codes, we cannot even use the MWPM decoder straightforwardly. Thus, massive efforts have been paid for developing efficient and near-optimal decoders which is applicable to various topological stabilizer codes and various noise models.

A method to use the MWPM decoder also for the color code [60] by projecting a color code to a surface code is proposed [51]. Another approach is to use renormalization group method [52] for decoding, which is applicable to any topological

code including the surface code and the color code. A method proposed in Ref. [53] uses union-find data-structure for decoding, which is fast and applicable to both of the surface and color codes, but has performance lower than that of the MWPM decoder. Therefore, the existing methods sacrifice either efficiency, near-optimal performance, or applicability to various topological codes and various noise models. For the first experimental realization of QEC on near-term devices, faster, more versatile, and near-optimal decoders are demanded.

In this chapter, we discuss a general construction of machine-learning-based decoders. The framework of supervised machine learning provides a state-of-the-art method to construct a prediction function which can estimate the correct label from a given feature with a high probability if the prediction model is appropriately trained. By delegating a part of decoding algorithm to machine learning, we can expect that a prediction function which estimates one of the correct recovery operations from an observed syndrome with a high probability is constructed. There are several advantages of such a machine-learning-based decoder compared with the previously known approaches. First, since the framework of machine learning is general, we expect that machine-learning-based decoders can be used for an arbitrary quantum error correcting code under an arbitrary noise model. Second, if we can train the prediction model appropriately, the trained model is expected to achieve a near-optimal accuracy in principle. Finally, though the training process may take a long time, it is required only at once before experiments, and each prediction for a given feature (syndrome) is expected to be fast. From the three points, we expect that machine-learning-based decoders can be used as a fast and high-performance decoder which is applicable to various topological codes and noise models.

Very recently, several machine-learning-based decoders have been proposed [54–58]. The first proposal of the machine-learning-based decoder was given by Torlai *et al.* [54]. They used a restricted Boltzmann machine as a prediction model, and numerically showed that a high-performance machine-learning-based decoder can be constructed. They named machine-learning-based decoders which use neural networks as a prediction model as *neural decoders*. Varsamopoulos *et al.* [55] and Krastanov *et al.* [57] studied what type of neural networks shows high performance as a prediction model of the neural decoder. They proposed neural decoders using neural networks called multi-layer perceptron models. Baireuther *et al.* [56] and Breuckmann *et al.* [58] extended the applicability of machine-learning-based decoders to the cases where the length of the syndrome data is not fixed, which often occurs when noisy syndrome measurements are repeated. All these existing studies numerically showed that the performance of the neural decoder is comparable with or superior to the known efficient decoders for the surface code when sufficiently large amount of the training data set is supplied. On the other hand, the applicability of these neural decoders appears to be limited to a very small distance. The neural decoders proposed in Refs. [55, 56] showed high performance only at distances no larger than 7. The neural decoders proposed in Refs. [54, 57, 58] re-

quire a post-processing which is expected to take a time that grows exponentially to the number of the physical qubits n. In other words, the interpreting function ξ introduced in Sec 2.4.2 cannot be computed efficiently, which slows down each prediction. Furthermore, the applicability of all the existing neural decoders was shown only for the surface code. In order to circumvent these drawbacks and to construct high-performance machine-learning-based decoders, it should be carefully studied how the decoding problem should be translated to the task of machine learning.

We explained in Sec 2.4.3 that there are four essential factors which determine the performance of the prediction function. The first is the choice of the prediction model, which was studied in Refs. [55, 57]. The second is the choice of the loss function and the optimizer, and a time for training. Since time for training process is not basically limited, we can use a time as long as accessible computational resource allows. Refs. [56, 58] improved the re-usability of the trained model, which is helpful when the training process takes a lot of time. The third factor is to prepare a sufficiently large training data set. In all the existing studies, the prediction model is trained with as large a training data set as we can prepare with the practical computational time. Compared to these three factors, little or no attention has been paid to the fourth factor, the choices of the feature space $\mathcal{X}$, the label space $\mathcal{Y}$, the prediction space $\mathcal{Y}'$, and the interpreting function ξ. So far, each of the previous studies introduces its own heuristic construction of the data set with little consideration on this point. We believe it is the reason why the existing neural decoders are only applicable to codes of a very small size. In this chapter, we focus on how we should choose these spaces and the function based on the properties of decoding problems in QEC.

As for the feature space $\mathcal{X}$, it is natural to choose a binary space of a syndrome $\{0,1\}^{n-k}$ as a feature space $\mathcal{X}$, since we can only observe a syndrome in experiments. On the other hand, there are various possible choices about the label space $\mathcal{Y}$. Recall that the binary representation of a physical error $\boldsymbol{e} \in \{0,1\}^{2n}$ is uniquely decomposed as

$$\boldsymbol{e} = \boldsymbol{l}_0(\boldsymbol{e}) \oplus \boldsymbol{w}(\boldsymbol{e})G \oplus \boldsymbol{t}(\boldsymbol{s}(\boldsymbol{e})), \tag{4.1}$$

as shown in Eq. (2.44). The optimal decoder outputs one of the most probable $\boldsymbol{w}$ conditioned on $\boldsymbol{s}$. The MD decoder outputs one of the most probable $\boldsymbol{e}$ conditioned on $\boldsymbol{s}$. Intuitively, possible choices are to use a binary space of $\boldsymbol{w}$ as the label space $\mathcal{Y}$, or to use a binary space of $\boldsymbol{e}$ as the label space. However, there is no guarantee that we can achieve near-optimal performance with these choices. Actually, we later prove that we may not achieve the optimal performance with these two choices even if we can assume that a prediction model has an infinite representation power, we use an unlimited training data set, and the total loss is perfectly minimized. Though all the existing studies avoid this problem using heuristic methods, it is not understood why such a heuristic is required. This implies that an analytical study about the mechanism of machine-learning-based

decoder is essential for constructing near-optimal neural decoders.

Here, we propose a general framework for constructing a neural decoder, *linear prediction framework*, to elucidate the factors that determine the performance of the decoders. In particular, we show conditions required for achieving the optimal performance when we can assume an unlimited training data set and the perfect loss minimization. We also propose a criterion called *normalized sensitivity* which should be optimized for constructing a near-optimal machine-learning-based decoder. Then, we propose specific constructions of a training data set for surface codes and color codes which satisfy the above condition and optimize the normalized sensitivity. We call these constructions as *uniform data construction*. We show the neural decoder with the uniform data construction shows performance higher than the existing neural decoders for surface codes. By comparing our results with that of the MD decoder, which is known to be near-optimal, we check that the performance of our construction achieves near-optimal performance. We also calculate the performance of the neural decoder for the color codes, and showed our construction and criterion is valid also for the color codes.

4.2 Linear prediction framework

In this section, we discuss how the decoding problem should be formulated as a task of machine learning in order to achieve near-optimal performance. To this end, we propose a general framework, which we call *linear prediction framework*. In this framework, we can analytically study the behavior of the neural decoder, and can discuss requirements for achieving near-optimal performance.

In order to discuss the behavior of the neural decoder in a unified view, we consider a neural decoder with the following two specifications. First, the neural decoder uses the syndrome vector $\boldsymbol{s}$ as the feature data to be fed to the trainable model, namely, $\mathcal{X} = \{0,1\}^{n-k}$. Second, the label space $\mathcal{Y}$ is a set of binary vectors, and the prediction space $\mathcal{Y}'$ is a real-valued vector space which has the same dimension as the binary vectors. Each label $\boldsymbol{y} \in \mathcal{Y}$ is linearly generated from the physical error vector $\boldsymbol{e}$ in GF(2). We call a linearly generated correct label vector $\boldsymbol{g}$ as a *diagnosis*, and a matrix H_g which generates the diagnosis $\boldsymbol{g} := H_g \Lambda \boldsymbol{e}^{\mathrm{T}}$ as a diagnosis matrix. We denote the length of the diagnosis $\boldsymbol{g}$ by L_g. Therefore, the label space $\mathcal{Y}$ is $\{H_g \Lambda \boldsymbol{e}^{\mathrm{T}} | \boldsymbol{e} \in \{0,1\}^{2n}\} \subseteq \{0,1\}^{L_g}$, and the prediction space $\mathcal{Y}'$ is $\mathbb{R}^{L_g}$. The recovery operator $\boldsymbol{r}$ is calculated from the predicted diagnosis $\boldsymbol{g}$ and the syndrome $\boldsymbol{s}$. We use an assumed physical error distribution $\{p_{\boldsymbol{e}}\}$ only for generating a training data set $\{(\boldsymbol{s}_i, \boldsymbol{g}_i)\}$, and do not use it for constructing H_g or in the calculation of the recovery operator $\boldsymbol{r}$ from $\boldsymbol{g}$ and $\boldsymbol{s}$. Though this framework restricts the label to be linearly generated from the physical error, this is general enough to formulate all the constructions described in the existing methods as special cases with small technical exceptions.

Since the actual performance of the neural decoder depends on many factors

such as configurations of the training process, the size of the training data set, and details of the chosen prediction model, we start with considering the problem under an ideal limit. We first consider the problem under the simple 0-1 loss function with an unlimited size of the training data set. Then, we relax these impractical assumptions to practical ones. Though we numerically investigate the case of a single logical qubit ($k = 1$) later, we present the formalism for a general value of k.

4.2.1 The neural decoder with the 0-1 loss function and an unlimited training data set

We first consider a hypothetical decoder that can minimize any loss function with an unlimited number of the training data set. Though such an assumption is not practical, it is convenient to reveal the conditions for performing optimal decoding with machine learning in the ideal limit. We choose the 0-1 delta function $\delta(\boldsymbol{g}, \boldsymbol{g}')$ as the loss function, which is zero if the predicted and the correct diagnosis are the same, and unity otherwise. Let us consider the portion of training data set with a specific value of $\boldsymbol{s}$ with $\Pr_{\boldsymbol{e}\sim\{p_e\}}[\boldsymbol{s}(\boldsymbol{e}) = \boldsymbol{s}] > 0$. If the neural decoder returns diagnosis $\boldsymbol{g}$ for the input $\boldsymbol{s}$, the total loss for this portion is proportional to the following value,

$$\begin{aligned} L_{\boldsymbol{s}}^{(\delta)}(\boldsymbol{g}) &:= \mathbb{E}_{\boldsymbol{e}\sim\{p_e\}}\left[\delta(\boldsymbol{g}, H_g\Lambda\boldsymbol{e}^{\mathrm{T}})\middle|\boldsymbol{s}(\boldsymbol{e}) = \boldsymbol{s}\right] \\ &= 1 - \Pr_{\boldsymbol{e}\sim\{p_e\}}\left[H_g\Lambda\boldsymbol{e}^{\mathrm{T}} = \boldsymbol{g}\middle|\boldsymbol{s}(\boldsymbol{e}) = \boldsymbol{s}\right]. \end{aligned} \tag{4.2}$$

Let $\boldsymbol{g}^{(\delta)}(\boldsymbol{s})$ be the output of the ideally trained neural decoder. Since it should minimize the total loss for every $\boldsymbol{s}$, it satisfies

$$L_{\boldsymbol{s}}^{(\delta)}(\boldsymbol{g}^{(\delta)}(\boldsymbol{s})) = \min_{\boldsymbol{g}} L_{\boldsymbol{s}}^{(\delta)}(\boldsymbol{g}). \tag{4.3}$$

We call this ideal decoder a *delta diagnosis decoder* and $\boldsymbol{g}^{(\delta)}(\boldsymbol{s})$ a *delta diagnosis vector*.

We show the condition for a diagnosis matrix H_g to guarantee that we can perform the optimal decoding with the delta diagnosis decoder. To this end, we define a property of the diagnosis matrix and introduce a set of diagnosis vectors as follows.

Definition 4.2.1. *faithful diagnosis matrix* — Given a check matrix H_c, we say diagnosis matrix H_g is *faithful* if

$$\mathrm{span}(\{(H_{cg})_i\}) = \mathcal{L}, \tag{4.4}$$

or equivalently,

$$H_{cg}\Lambda\boldsymbol{e}^{\mathrm{T}} = 0 \leftrightarrow \boldsymbol{e} \in \mathcal{L}_0, \tag{4.5}$$

where

$$H_{cg} := \begin{pmatrix} H_c \\ H_g \end{pmatrix}. \tag{4.6}$$

Definition 4.2.2. *faithful diagnosis vectors* — Given a check matrix H_c, a pure error $\boldsymbol{t}(\boldsymbol{s})$, and a faithful diagnosis matrix H_g, we define 2^{2k} faithful diagnosis vectors $\{\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w})\}$ ($\boldsymbol{w} \in \{0,1\}^{2k}$) associated with a syndrome vector $\boldsymbol{s}$ by

$$\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}) := H_g \Lambda (\boldsymbol{w}G \oplus \boldsymbol{t}(\boldsymbol{s}))^{\mathrm{T}}. \tag{4.7}$$

Note that the faithful condition of H_g implies that

$$\boldsymbol{w} \mapsto \boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}) \tag{4.8}$$

is injective and

$$H_g \Lambda \boldsymbol{e}^{\mathrm{T}} = \boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}(\boldsymbol{e})), \tag{4.9}$$

with $\boldsymbol{s} = H_c \Lambda \boldsymbol{e}^{\mathrm{T}}$. As a result, when H_g is faithful, we have

$$1 - L_{\boldsymbol{s}}^{(\delta)}(\boldsymbol{g}) = \Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}} [\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}(\boldsymbol{e})) = \boldsymbol{g} | \boldsymbol{s}(\boldsymbol{e}) = \boldsymbol{s}] \tag{4.10}$$

from Eqs. (4.2) and (4.9). Then the injective property of $\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w})$ leads to

$$1 - L_{\boldsymbol{s}}^{(\delta)}(\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w})) = q_{\boldsymbol{s}}(\boldsymbol{w}), \tag{4.11}$$

where $q_{\boldsymbol{s}}(\boldsymbol{w})$ is defined in Eq. (2.54).

When the diagnosis matrix is faithful, we can construct an optimal decoder as follows. From Eqs. (4.3) and (4.11), we see that the delta diagnosis vector $\boldsymbol{g}^{(\delta)}(\boldsymbol{s})$ is one of the faithful diagnosis vectors. We can thus write it in the form

$$\boldsymbol{g}^{(\delta)}(\boldsymbol{s}) = \boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}^*(\boldsymbol{s})). \tag{4.12}$$

Eqs. (4.3), (4.11), and (4.12) imply that

$$\begin{aligned} 1 - q_{\boldsymbol{s}}(\boldsymbol{w}^*(\boldsymbol{s})) &= L_{\boldsymbol{s}}^{(\delta)}(\boldsymbol{g}^{(\delta)}(\boldsymbol{s})) \\ &= \min_{\boldsymbol{w}\in\{0,1\}^{2k}} (1 - q_{\boldsymbol{s}}(\boldsymbol{w})) \\ &= 1 - \max_{\boldsymbol{w}\in\{0,1\}^{2k}} q_{\boldsymbol{s}}(\boldsymbol{w}). \end{aligned} \tag{4.13}$$

Since $\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w})$ is injective, one can calculate $\boldsymbol{w}^*(\boldsymbol{s})$ from the diagnosis $\boldsymbol{g}^{(\delta)}(\boldsymbol{s})$ and syndrome $\boldsymbol{s}$. The recovery operator is then chosen as

$$\boldsymbol{r}(\boldsymbol{s}) = \boldsymbol{w}^*(\boldsymbol{s})G \oplus \boldsymbol{t}(\boldsymbol{s}). \tag{4.14}$$

For the optimality, we have

$$\begin{aligned} \Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}} [\boldsymbol{e} \oplus \boldsymbol{r}(\boldsymbol{s}) \in \mathcal{L}_0 | \boldsymbol{s}(\boldsymbol{e}) = \boldsymbol{s}] &= q_{\boldsymbol{s}}(\boldsymbol{w}^*(\boldsymbol{s})) \\ &= \max q_{\boldsymbol{s}}(\boldsymbol{w}) \end{aligned} \tag{4.15}$$

for any $\boldsymbol{s}$ with $\Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}} [\boldsymbol{s}(\boldsymbol{e}) = \boldsymbol{s}] > 0$, which satisfies Eq. (2.55).

We can also prove a converse statement for the cases where H_g is not faithful (see Appendix B), arriving at the following lemma.

Lemma 4.2.1. If the diagnosis matrix H_g is faithful, there exists a map $\boldsymbol{r}^*(\boldsymbol{g}, \boldsymbol{s})$ such that the decoder with $\boldsymbol{r}(\boldsymbol{s}) = \boldsymbol{r}^*(\boldsymbol{g}^{(\delta)}(\boldsymbol{s}), \boldsymbol{s})$ is optimal for arbitrary distribution $\{p_{\boldsymbol{e}}\}$. If the diagnosis matrix H_g is not faithful, no such map exists.

This lemma implies that we can perform optimal decoding with the delta diagnosis decoder only when the diagnosis matrix H_g is faithful. Note that the set of the faithful vectors $\{\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w})|\boldsymbol{w} \in \{0,1\}^{2k}\}$ is independent of the choice of the generator G and the pure error $\boldsymbol{t}(\boldsymbol{s})$. Whether we can perform the optimal decoding or not is dependent only on the construction of H_g.

4.2.2 The neural decoder with the L2 loss function and an unlimited training data set

In this subsection, we replace the 0-1 loss function with a more practical one, which is the squared L2 distance. We still consider the limit of an infinite size of the training data set and the perfect loss minimization. In this case, the total loss for a fixed $\boldsymbol{s}$ under an unlimited training data set is proportional to the following value.

$$L_{\boldsymbol{s}}^{(\mathrm{L2})}(\boldsymbol{g}) = \mathbb{E}_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}\left[||\boldsymbol{g} - H_g\Lambda\boldsymbol{e}^{\mathrm{T}}||_2^2 \middle| \boldsymbol{s}(\boldsymbol{e}) = \boldsymbol{s}\right] \tag{4.16}$$

We define a decoder which is ideally trained with the L2 loss function as an *L2 diagnosis decoder*. We also call the output of the L2 diagnosis decoder as an *L2 diagnosis vector* $\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s})$. The L2 diagnosis vector satisfies the following equation.

$$L_{\boldsymbol{s}}^{(L2)}(\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s})) \quad = \quad \min_{\boldsymbol{g}\in\{0,1\}^{L_g}} L_{\boldsymbol{s}}^{(\mathrm{L2})}(\boldsymbol{g}). \tag{4.17}$$

When the chosen diagnosis matrix is faithful, the L2 diagnosis vector can be written as follows.

$$\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s}) \quad := \quad \sum_{\boldsymbol{w}\in\{0,1\}^{2k}} q_{\boldsymbol{s}}(\boldsymbol{w})\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}) \tag{4.18}$$

Let us define a column vector of order 2^{2k} as

$$\boldsymbol{q}_{\boldsymbol{s}} := (q_{\boldsymbol{s}}(0^{2k}), \dots q_{\boldsymbol{s}}(1^{2k}))^{\mathrm{T}}. \tag{4.19}$$

It satisfies the following matrix equation:

$$\begin{pmatrix} \hat{\boldsymbol{g}}^{(\mathrm{L2})}(\boldsymbol{s}) \\ 1 \end{pmatrix} = D_{\boldsymbol{s}}\boldsymbol{q}_{\boldsymbol{s}}, \tag{4.20}$$

where

$$D_{\boldsymbol{s}} = \begin{pmatrix} \boldsymbol{g}_{\boldsymbol{s}}(0^{2k}) & \cdots & \boldsymbol{g}_{\boldsymbol{s}}(1^{2k}) \\ 1 & \cdots & 1 \end{pmatrix}. \tag{4.21}$$

We can solve it for $\boldsymbol{q}_{\boldsymbol{s}}$ if $D_{\boldsymbol{s}}$ has a left inverse $D_{\boldsymbol{s}}^{-1}$ such that $D_{\boldsymbol{s}}^{-1}D_{\boldsymbol{s}} = I$ in the real-valued calculation, namely, if the rank of $D_{\boldsymbol{s}}$ as a real-valued matrix is 2^{2k}. If the rank is smaller, solution $\boldsymbol{q}_{\boldsymbol{s}}$ is not unique, and hence it is not always possible to determine $\boldsymbol{w}$ that maximizes $q_{\boldsymbol{s}}(\boldsymbol{w})$, which implies we cannot perform the optimal decoding.

Though the rank condition depends apparently on the syndrome $\boldsymbol{s}$, we can formulate it as a condition which is independent of $\boldsymbol{s}$. Any faithful diagnosis $\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w})$ can be written as

$$\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}) = H_g\Lambda(\boldsymbol{w}G)^{\mathrm{T}} \oplus \boldsymbol{\delta}(\boldsymbol{s}) \tag{4.22}$$

with

$$\boldsymbol{\delta}(\boldsymbol{s}) := H_g\Lambda\boldsymbol{t}(\boldsymbol{s})^{\mathrm{T}} \in \left(\{0,1\}^{L_g}\right)^{\mathrm{T}}. \tag{4.23}$$

We define a transformation $\sigma_{\boldsymbol{\delta}}$ by

$$(\sigma_{\boldsymbol{\delta}}(\boldsymbol{v}))_i \;:=\; \delta_i + (-1)^{\delta_i}v_i \tag{4.24}$$

for $\boldsymbol{\delta} \in \{0,1\}^{2k}$ and $\boldsymbol{v} \in \mathbb{R}^{2k}$. It is affine, isometric, and involutory. Since $\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}) = \sigma_{\boldsymbol{\delta}(\boldsymbol{s})}(H_g\Lambda(\boldsymbol{w}G)^{\mathrm{T}})$, we have

$$D_{\boldsymbol{s}} \;=\; \begin{pmatrix} \sigma_{\boldsymbol{\delta}(\boldsymbol{s})}(H_g\Lambda((0^{2k})G)^{\mathrm{T}}) & \cdots & \sigma_{\boldsymbol{\delta}(\boldsymbol{s})}(H_g\Lambda((1^{2k})G)^{\mathrm{T}}) \\ 1 & \cdots & 1 \end{pmatrix}. \tag{4.25}$$

We see that a transformation $\sigma_{\boldsymbol{\delta}}$ is an affine transformation, and this transformation satisfies

$$\sigma_{\boldsymbol{\delta}}(\sigma_{\boldsymbol{\delta}}(\boldsymbol{v})) \;=\; \boldsymbol{v} \tag{4.26}$$

$$\sigma_0(\boldsymbol{v}) \;=\; \boldsymbol{v}. \tag{4.27}$$

When we apply the transformation $\sigma_{\boldsymbol{\delta}(\boldsymbol{s})}$ to Eq. (4.20), we obtain

$$\begin{pmatrix} \sigma_{\boldsymbol{\delta}}(\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s})) \\ 1 \end{pmatrix} = D\boldsymbol{q}_{\boldsymbol{s}}, \tag{4.28}$$

where

$$D \;:=\; \begin{pmatrix} H_g\Lambda((0,\ldots,0)G)^{\mathrm{T}} & \cdots & H_g\Lambda((1,\ldots,1)G)^{\mathrm{T}} \\ 1 & \cdots & 1 \end{pmatrix}, \tag{4.29}$$

Thus, we can uniquely calculate $\boldsymbol{q}_{\boldsymbol{s}}$ for an arbitrary $\boldsymbol{s}$ if a matrix D has a left inverse, which is equivalent to the condition that $\{H_g\Lambda(\boldsymbol{w}G)^{\mathrm{T}} | \boldsymbol{w} \in \{0,1\}^{2k}\}$ is affinely independent. We will call a diagnosis matrix satisfying this condition to be decomposable:

Definition 4.2.3. *decomposable diagnosis matrix* — Given a generator matrix G, we say a diagnosis matrix H_g is decomposable if a set of real vectors $\{H_g\Lambda(\boldsymbol{w}G)^{\mathrm{T}}|\boldsymbol{w} \in \{0,1\}^{2k}\}$ is affinely independent, namely, the rank of a matrix D defined in Eq. (4.29) is 2^{2k} when we consider D as a real-valued matrix.

When H_g is faithful, the above definition is independent of G, because the set $\{H_g\Lambda(\boldsymbol{w}G)^{\mathrm{T}}|\boldsymbol{w} \in \{0,1\}^{2k}\}$ is independent of G then.

We show a scheme to perform the optimal decoding using L2 diagnosis decoder when a diagnosis matrix is faithful and decomposable. When H_g is decomposable, there exists a left inverse D^{-1} such that $D^{-1}D = I$ in real vector space. When we observe a syndrome vector $\boldsymbol{s}$, we obtain the L2 diagnosis $\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s})$ using the trained L2 diagnosis decoder, and calculate $\boldsymbol{\delta}(\boldsymbol{s}) = H_g\Lambda\boldsymbol{t}(\boldsymbol{s})$. Since the diagnosis matrix is faithful, the probabilities of the faithful diagnosis vectors are given by

$$\boldsymbol{q}_{\boldsymbol{s}} = D^{-1}\begin{pmatrix}\sigma_{\boldsymbol{\delta}(\boldsymbol{s})}(\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s}))\\ 1\end{pmatrix}. \tag{4.30}$$

Then, we construct a recovery operator as

$$\boldsymbol{r}(\boldsymbol{s}) = \boldsymbol{w}^*(\boldsymbol{s})G \oplus \boldsymbol{t}(\boldsymbol{s}), \tag{4.31}$$

where $\boldsymbol{w}^*(\boldsymbol{s})$ satisfies

$$q_{\boldsymbol{s}}(\boldsymbol{w}^*(\boldsymbol{s})) = \max_{\boldsymbol{w}} q_{\boldsymbol{s}}(\boldsymbol{w}). \tag{4.32}$$

With this recovery operator, we obtain

$$\mathrm{Pr}_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}\left[\boldsymbol{e} \oplus \boldsymbol{r}(\boldsymbol{s}) \in \mathcal{L}_0|\boldsymbol{s}(\boldsymbol{e}) = \boldsymbol{s}\right] = q_{\boldsymbol{s}}(\boldsymbol{w}^*(\boldsymbol{s})), \tag{4.33}$$

and thus this decoder satisfies Eq. (2.55).

When the diagnosis matrix H_g is faithful, we can also prove a converse statement for the cases where a faithful diagnosis matrix H_g is not decomposable (see Appendix B), arriving at the following lemma.

Lemma 4.2.2. If the diagnosis matrix H_g is faithful and decomposable, there exists a map $\boldsymbol{r}^*(\boldsymbol{g},\boldsymbol{s})$ such that the decoder with $\boldsymbol{r}(\boldsymbol{s}) = \boldsymbol{r}^*(\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s}),\boldsymbol{s})$ is optimal for arbitrary distribution $\{p_{\boldsymbol{e}}\}$. If the diagnosis matrix H_g is faithful but not decomposable, no such map exists.

We show a simple example of a faithful and decomposable matrix H_g in the case of $k = 1$. We choose vectors $\boldsymbol{l}_{01}$, $\boldsymbol{l}_{10}$, and $\boldsymbol{l}_{11}$ from $\mathcal{L}_{01}$, $\mathcal{L}_{10}$, and $\mathcal{L}_{11}$, respectively. We construct H_g and generator G as

$$H_g = \begin{pmatrix}\boldsymbol{l}_{01}\\ \boldsymbol{l}_{10}\\ \boldsymbol{l}_{11}\end{pmatrix}, \tag{4.34}$$

$$G = \begin{pmatrix}\boldsymbol{l}_{01}\\ \boldsymbol{l}_{10}\end{pmatrix}. \tag{4.35}$$

We see that $\mathrm{span}(\{(H_g)_i\}) = \mathcal{L}$, and thus H_g is faithful. A set $\{H_g\Lambda(\boldsymbol{w}G)^{\mathrm{T}}|\boldsymbol{w} \in \{00, 01, 10, 11\}\}$ is

$$\{(0,0,0)^{\mathrm{T}}, (0,1,1)^{\mathrm{T}}, (1,0,1)^{\mathrm{T}}, (1,1,0)^{\mathrm{T}}\}, \tag{4.36}$$

which is affinely independent, and thus H_g is decomposable. We can verify the same by checking the rank of

$$\begin{aligned} D &= \begin{pmatrix} \boldsymbol{g}(00) & \boldsymbol{g}(01) & \boldsymbol{g}(10) & \boldsymbol{g}(11) \\ 1 & 1 & 1 & 1 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} \end{aligned} \tag{4.37}$$

to be 4 in real vector space.

4.2.3 The neural decoder with the L2 loss function under a finite training data size

In practical cases, the size of the training data set is limited, and hence the loss is not perfectly minimized. This implies that the output diagnosis from the model deviates from the L2 diagnosis vector. In such a case, it is desirable to construct a decoder such that its prediction is as robust against the deviations as possible. We introduce a slight modification to the optimum decoding scheme in the last subsection, so that it should applicable to an output diagnosis deviated from the L2 diagnosis vector.

We denote the predicted diagnosis as $\boldsymbol{g}^{\mathrm{P}}(\boldsymbol{s}) \in \mathbb{R}^{L_g}$, which deviates from the L2 diagnosis vector. Note that $\boldsymbol{g}^{\mathrm{P}}(\boldsymbol{s})$ cannot be represented as a linear combination of the faithful diagnosis vectors in general. In order to construct a decoding scheme which is robust to a small deviation, it is natural to extend the scheme employed in Sec. 4.2.2 such that we project $\boldsymbol{g}^{\mathrm{P}}(\boldsymbol{s})$ to the hyper-plane formed by affine combinations of the faithful diagnosis vectors, and then extract the coefficients $\boldsymbol{q}_{\boldsymbol{s}}^{\mathrm{P}}$ from the projected point. This projection and extraction is achieved as follows. We perform QR decomposition for D, and obtain $D = QR$, where Q is an orthogonal matrix, and R is an upper-triangular matrix. We construct $D^{-1} = R^{-1}Q^{\mathrm{T}}$, which satisfies $D^{-1}D = I$. Then, we obtain a predicted vector $\boldsymbol{q}_{\boldsymbol{s}}^{\mathrm{P}}$ as

$$\boldsymbol{q}_{\boldsymbol{s}}^{\mathrm{P}} = D^{-1} \begin{pmatrix} \sigma_{\boldsymbol{\delta}(\boldsymbol{s})}(\boldsymbol{g}^{\mathrm{P}}(\boldsymbol{s})) \\ 1 \end{pmatrix}, \tag{4.38}$$

where $\boldsymbol{\delta}(\boldsymbol{s}) = H_g\Lambda\boldsymbol{t}(\boldsymbol{s})$. We construct a recovery operator as

$$\boldsymbol{r}(\boldsymbol{s}) = \boldsymbol{w}^*(\boldsymbol{s})G \oplus \boldsymbol{t}(\boldsymbol{s}), \tag{4.39}$$

where $\boldsymbol{w}^*(\boldsymbol{s})$ satisfies

$$q^{\mathrm{P}}_{\boldsymbol{s}}(\boldsymbol{w}^*(\boldsymbol{s})) = \max_{\boldsymbol{w}} q^{\mathrm{P}}_{\boldsymbol{s}}(\boldsymbol{w}). \tag{4.40}$$

Note that though elements of $\boldsymbol{q}^{\mathrm{P}}_{\boldsymbol{s}}$ may be out of $[0, 1]$, the above procedure is still well-defined.

4.3 Criterion for diagnosis matrix and specific constructions

In the last section, we showed that we can achieve the optimal performance with the ideal training when a diagnosis matrix is faithful and decomposable. There are several constructions of faithful and decomposable diagnosis matrices. In this section, we show a criterion called *normalized sensitivity*, which should be optimized for performing near-optimal decoding when a size of a training data set is limited. Then, we show specific constructions of diagnosis matrices, which are faithful and decomposable, and have the optimal order of the normalized sensitivity in terms of the distance. We call these specific constructions as *uniform data constructions*.

4.3.1 Normalized sensitivity

In practice, the number of the training data set is far smaller than the total variation of syndrome vectors $\boldsymbol{s}$ when distance d is larger than about 7. For example, according to the existing methods [54–57], the size of the training data set is at most 10^9. On the other hand, the number of variations in the syndrome, 2^{n-k}, becomes larger than 10^9 at the distance $d = 7$ for the $[[d^2, 1, d]]$ surface code. This implies that almost all the patterns of the syndrome vector $\boldsymbol{s}$ given in experiments are not found in the training data set. The model should infer the label $\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s})$ of $\boldsymbol{s}$ where $\boldsymbol{s}$ is not included in the training data set. The aim of this subsection is to propose a criterion for H_g which we believe to reflect the robustness of the prediction when we use such a sparsely sampled training data set.

Since the problem is to estimate the vector-valued function $\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s})$ from a sparsely sampled set of values, its difficulty should depend on how rapidly the function changes its output value as the input value $\boldsymbol{s}$ varies. From Eqs. (4.9) and (4.18), we see that the function is written as

$$\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s}) = \mathbb{E}_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}[H_g \Lambda \boldsymbol{e}^{\mathrm{T}} | \boldsymbol{s}(\boldsymbol{e}) = \boldsymbol{s}], \tag{4.41}$$

which shows that $\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s})$ is implicitly determined from the two functions of errors, $\boldsymbol{g}(\boldsymbol{e}) = H_g \Lambda \boldsymbol{e}^{\mathrm{T}}$ and $\boldsymbol{s}(\boldsymbol{e}) = H_c \Lambda \boldsymbol{e}^{\mathrm{T}}$. In order to quantify how rapidly these function

change, let us introduce a sensitivity $m(H)$ of a binary matrix H as

$$\begin{aligned} m(H) &:= \max_{\substack{\boldsymbol{e},\boldsymbol{e}'\in\{0,1\}^{2n} \\ h(\boldsymbol{e}\oplus\boldsymbol{e}')=1}} ||H\Lambda\boldsymbol{e}^{\mathrm{T}} - H\Lambda\boldsymbol{e}'^{\mathrm{T}}||_2^2 \\ &= \max_{\substack{\boldsymbol{e}\in\{0,1\}^{2n} \\ h(\boldsymbol{e})=1}} h(H\Lambda\boldsymbol{e}^{\mathrm{T}}). \end{aligned} \tag{4.42}$$

Using the sensitivity, the variation of $\boldsymbol{s}(\boldsymbol{e})$ is bounded as

$$||\boldsymbol{s}(\boldsymbol{e}) - \boldsymbol{s}(\boldsymbol{e}')||_2^2 \leq m(H_c)h(\boldsymbol{e}\oplus\boldsymbol{e}'). \tag{4.43}$$

In the case of topological codes, $m(H_c)$ is a small constant. This is because each physical qubit is monitored by at most constant number of the stabilizer operators.

Suppose that $\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s})$ is close to one of the faithful diagnosis $\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}^*)$, and let $S(\boldsymbol{s},\boldsymbol{w}^*;0)$ be the set of errors $\boldsymbol{e}$ satisfying $\boldsymbol{w}(\boldsymbol{e}) = \boldsymbol{w}^*$ and $\boldsymbol{s}(\boldsymbol{e}) = \boldsymbol{s}$. We further define a set

$$S(\boldsymbol{s},\boldsymbol{w}^*;h) := \{\boldsymbol{e}|\exists\boldsymbol{e}' \text{ s.t. } \boldsymbol{e}' \in S(\boldsymbol{s},\boldsymbol{w}^*;0), h(\boldsymbol{e}\oplus\boldsymbol{e}') \leq h\} \tag{4.44}$$

We see that any $\boldsymbol{e} \in S(\boldsymbol{s},\boldsymbol{w}^*;h)$ produces a training data $(\boldsymbol{s}',\boldsymbol{g}')$ such that

$$||\boldsymbol{s}' - \boldsymbol{s}||_2^2 \leq m(H_c)h \tag{4.45}$$

$$||\boldsymbol{g}' - \boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}^*)||_2^2 \leq m(H_g)h. \tag{4.46}$$

The choice of H_g also affects how precisely $\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s})$ should be estimated in order to determine $\boldsymbol{w}^*$ correctly. To quantify this, we consider how far $\boldsymbol{g}^{\mathrm{P}}(\boldsymbol{s})$ can be deviated from a faithful diagnosis $\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w})$ without affecting the decoding method of Eqs. (4.38) and (4.39). When the decoding result changes from $\boldsymbol{w}^* = \boldsymbol{w}$ to $\boldsymbol{w}^* = \boldsymbol{w}'$, the solution of Eq. (4.38) should satisfy $q^{\mathrm{P}}_{\boldsymbol{s}}(\boldsymbol{w}) = q^{\mathrm{P}}_{\boldsymbol{s}}(\boldsymbol{w}')$, namely, $\boldsymbol{g}^{\mathrm{P}}(\boldsymbol{s})$ should be written in the form

$$\boldsymbol{g}^{\mathrm{P}}(\boldsymbol{s}) = \alpha(\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}) + \boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}')) + \sum_{\boldsymbol{w}''\neq\boldsymbol{w},\boldsymbol{w}'} \beta_{\boldsymbol{w}''}\boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w}''). \tag{4.47}$$

We define the minimum boundary distance $M(H_g)$ so as to assure that $\boldsymbol{w}^* = \boldsymbol{w}$ as long as $|\boldsymbol{g}^{\mathrm{P}}(\boldsymbol{s}) - \boldsymbol{g}_{\boldsymbol{s}}(\boldsymbol{w})| \leq M(H_g)$. Hence $M(H_g)$ can be explicitly defined as

$$M(H_g) := \min_{\boldsymbol{w},\boldsymbol{w}',\alpha,\{\beta_{\boldsymbol{w}''}\}} ||(1-\alpha)\boldsymbol{g}(\boldsymbol{w}) - \alpha\boldsymbol{g}(\boldsymbol{w}') - \sum_{\boldsymbol{w}''\neq\boldsymbol{w},\boldsymbol{w}'} \beta_{\boldsymbol{w}''}\boldsymbol{g}(\boldsymbol{w}'')||_2^2. \tag{4.48}$$

Note that the above definition is independent of $\boldsymbol{s}$, since the affine transformation $\sigma_{\boldsymbol{\delta}(\boldsymbol{s})}$ is isometric. $M(H_g)$ is nonzero if and only if H_g is decomposable.

Regarding $M(H_g)$ as the relevant length scale, we define the following quantity to be used as a criterion for a better construction of H_g.

Definition 4.3.1. *Normalized sensitivity* — We define normalized sensitivity $N(H_g)$ of a faithful and decomposable matrix H_g as

$$N(H_g) := \frac{m(H_g)}{M(H_g)}, \tag{4.49}$$

where $m(H_g)$ is a sensitivity of H_g defined in Eq. (4.42), and $M(H_g)$ is a minimum boundary distance of H_g defined in Eq. (4.48).

Eqs. (4.46) and (4.48) implies that an error belonging to $S(\boldsymbol{s}, \boldsymbol{w}^*; h)$ with $h \sim (m(H_g)/M(H_g))^{-1}$ leads to a training data useful for estimation of $\boldsymbol{g}^{(\mathrm{L2})}(\boldsymbol{s})$. We thus expect that the use of a diagnosis matrix H_g with a small normalized sensitivity $N(H_g)$ enables high-performance prediction with a small training data set.

4.3.2 Uniform data construction

We propose specific constructions which minimize the normalized sensitivity up to the order of d in the case of $k = 1$. We first consider a lower-bound of the normalized sensitivity. When a diagnosis matrix H_g is faithful, each row vector of H_g corresponds to a logical operator or a stabilizer operator. We denote the number of the logical operators in the rows of H_g as n_L. The minimum boundary distance $M(H_g)$ is upper-bounded by

$$M(H_g) \leq \frac{n_L}{4} \tag{4.50}$$

Since any logical operator has at least d of one-elements in its binary representation, there are at least dn_L of one-elements in the diagnosis matrix. By denoting the number of the one-elements in the diagnosis matrix H_g as $\chi(H_g)$, we have

$$dn_L \leq \chi(H_g). \tag{4.51}$$

Since there are $2n$ columns in H_g, we also have

$$\chi(H_g) \leq 2n \max_i h((H_g^{\mathrm{T}})_i). \tag{4.52}$$

The sensitivity $m(H_g)$ is equal to the maximum hamming weight of the column vectors of the diagnosis matrix, namely,

$$\max_i h((H_g)_i^{\mathrm{T}}) = m(H_g). \tag{4.53}$$

From Eqs. (4.50) - (4.53), we obtain

$$N(H_g) \geq \frac{2d}{n} \tag{4.54}$$

In particular, when we focus on the two-dimensional topological codes such that $n = \Theta(d^2)$, the order of the normalized sensitivity is lower-bounded as

$$N(H_g) = \Omega(d^{-1}). \tag{4.55}$$

For surface codes and color codes with the single logical qubit, we found specific constructions of H_g such that $N(H_g)$ scales as $\Theta(d^{-1})$. See appendix C for the specific constructions. We named these constructions as *uniform data construction* of the data set, since logical operators corresponding to the rows of H_g are chosen uniformly to cover all the physical qubits.

4.4 Construction of data set and example

Let us summarize the discussion in Sec 4.2 and Sec 4.3. Given the check matrix H_c of the code and the error model $\{p_{\boldsymbol{e}}\}$, the whole protocol can be described as follows.

- **Preparation:** We construct a faithful and decomposable diagnosis matrix H_g with a small normalized sensitivity, possibly $N(H_g) = \Theta(d/n)$. We choose a pure error $\boldsymbol{t}(\boldsymbol{s})$ and a generator matrix G. We perform QR decomposition to a matrix

$$D := \begin{pmatrix} \boldsymbol{g}(0^{2k}) & \cdots & \boldsymbol{g}(1^{2k}) \\ 1 & \cdots & 1 \end{pmatrix}, \tag{4.56}$$

  where $\boldsymbol{g}(\boldsymbol{w}) = H_g \Lambda (\boldsymbol{w} G)^{\mathrm{T}}$, and obtain Q and R. We calculate the left inverse matrix D^{-1} as

$$D^{-1} := R^{-1} Q^{\mathrm{T}}. \tag{4.57}$$

- **Data generation:** We generate a set of physical errors $\{\boldsymbol{e}_0, \boldsymbol{e}_1, \ldots\}$ with the probability distribution $\{p_{\boldsymbol{e}}\}$, and generate data set $\{(\boldsymbol{s}_0, \boldsymbol{g}_0), (\boldsymbol{s}_1, \boldsymbol{g}_1), \ldots\}$ from it, where $\boldsymbol{s}_i := H_c \Lambda \boldsymbol{e}_i^{\mathrm{T}}$ and $\boldsymbol{g}_i := H_g \Lambda \boldsymbol{e}_i^{\mathrm{T}}$.

- **Training:** The model is trained so that it can predict $\boldsymbol{g}$ from $\boldsymbol{s}$. The loss of the prediction is defined as the L2 distance between $\boldsymbol{g}$ and $\boldsymbol{g}^{\mathrm{P}}$, where $\boldsymbol{g}^{\mathrm{P}}$ is a real-valued output vector of the model.

- **Prediction:** When an observed syndrome $\boldsymbol{s}$ is given to the trained model, it predicts $\boldsymbol{g}^{\mathrm{P}}(\boldsymbol{s})$. In parallel, we calculate $\boldsymbol{\delta}(\boldsymbol{s})$ given by

$$\boldsymbol{\delta}(\boldsymbol{s}) = H_g \Lambda \boldsymbol{t}(\boldsymbol{s})^{\mathrm{T}}. \tag{4.58}$$

  We calculate vector $\boldsymbol{q}_{\boldsymbol{s}}$ defined in Eq. (4.19) as

$$\boldsymbol{q}_{\boldsymbol{s}}^{\mathrm{P}} = D^{-1} \begin{pmatrix} \sigma_{\boldsymbol{\delta}(\boldsymbol{s})}(\boldsymbol{g}^{\mathrm{P}}(\boldsymbol{s})) \\ 1 \end{pmatrix}, \tag{4.59}$$

where $\sigma_{\boldsymbol{\delta}(\boldsymbol{s})}$ is an affine transformation such that

$$(\sigma_{\boldsymbol{\delta}(\boldsymbol{s})}(\boldsymbol{v}))_i = \delta_i + (-1)^{\delta_i} v_i. \tag{4.60}$$

We choose $\boldsymbol{w}^{\mathrm{P}}$ that satisfies

$$q^{\mathrm{P}}_{\boldsymbol{s}}(\boldsymbol{w}^{\mathrm{P}}) = \max_{\boldsymbol{w}\in\{0,1\}^{2k}} q^{\mathrm{P}}_{\boldsymbol{s}}(\boldsymbol{w}), \tag{4.61}$$

where $\{q^{\mathrm{P}}_{\boldsymbol{s}}(\boldsymbol{w})\}$ is the elements of $\boldsymbol{q}^{\mathrm{P}}_{\boldsymbol{s}}$. Then, we obtain an estimated recovery operator

$$\boldsymbol{r}(\boldsymbol{s}) = \boldsymbol{w}^{\mathrm{P}} G \oplus \boldsymbol{t}(\boldsymbol{s}). \tag{4.62}$$

We emphasize that the choice of $\boldsymbol{t}(\boldsymbol{s})$ and G dose not affect the performance of the decoder, since the success of the estimation is independent of them. Only the construction of H_g affects the performance of the decoder.

We show a specific example of the decoding scheme. For simplicity, we consider the case where there is only bit-flip errors in the $[[2d^2 - 2d + 1, 1, d]]$ surface code. In this case, it is enough for QEC to consider the stabilizer operator with Pauli Z operators. The simplified picture of the code is shown in Fig. 4.1.

In this picture, a bit-flip error on a physical qubit is represented by the color of the corresponding edge (green: no error, red: error), and the syndrome value is represented by the color of the circle (green: undetected, red: detected). As shown in Fig. 4.1(a), The matrix H_g is constructed with logical operators each of which is the product of the Pauli Z operators on the edges crossing the dotted line. In this case, we see $M(H_g) = O(d)$, $m(H_g) = O(1)$, and $\frac{m(H_g)}{M(H_g)} = O(d^{-1})$.

Suppose that bit-flip errors occur on a set of the physical qubits as shown in Fig. 4.1(b). The physical error is detected with the syndrome values as shown in the same figure. The diagnosis vector is calculated as the commutation relation of the chosen logical operators and the physical error. We show the calculated diagnosis on the right side of the lattice. In the training phase, the model learns the relation between the positions of the red circles and the values of the diagnosis vector. In the prediction phase, only the positions of the red circles are given. The trained neural network outputs a real-valued prediction of the diagnosis vector as shown in Fig. 4.1(c), for example. From this information, we extract the probabilities of the faithful diagnosis, and we choose the faithful diagnosis which is expected to be the most probable, as shown in Fig. 4.1(d). Since the chosen diagnosis vector is equivalent to the diagnosis vector generated by the actual physical error, this decoding trial is a success.

4.5 Relation to the existing methods

In this section, we explain how the existing methods [54–57] can be treated in the linear prediction framework. The method proposed by Varsamopoulos *et al.* [55]

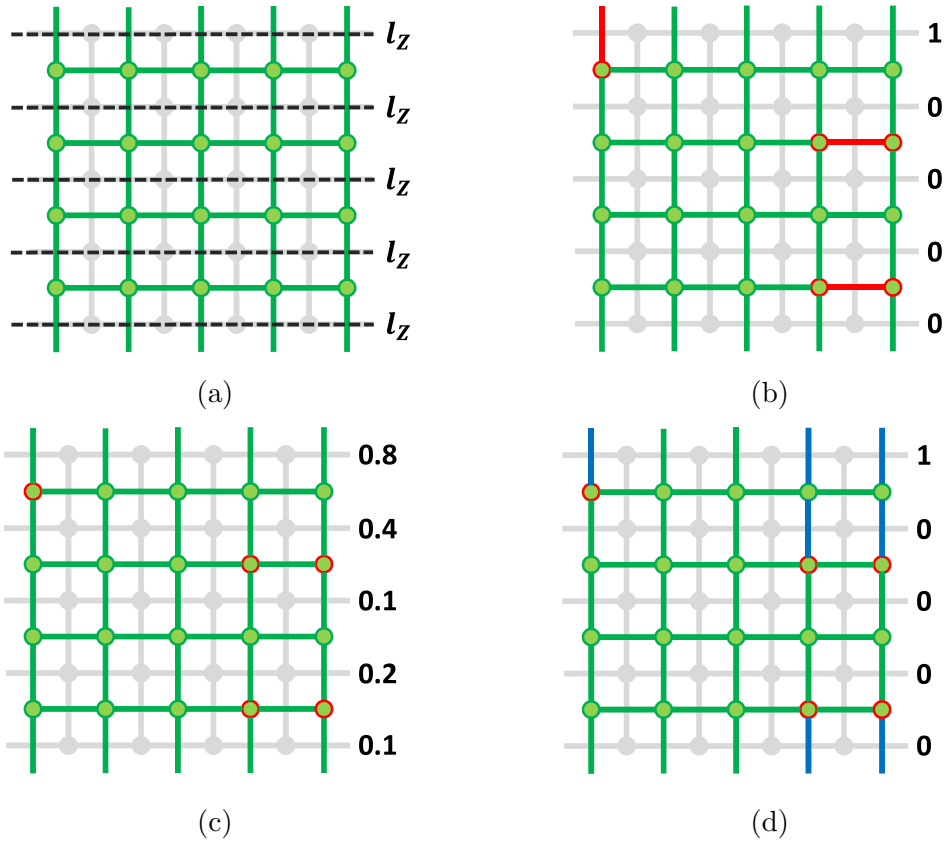


Figure 4.1: The figures show the decoding process based on proposed scheme. Each picture shows only Z lattice, of which the edge corresponds to whether there is a bit-flip error on the physical qubit or not, and the circle shows whether an error is detected through the syndrome measurement. (a) Five logical Z operators which minimize the normalized sensitivity $\frac{m(H_g)}{M(H_g)}$. (b) The actual physical error is drawn as red edges, and the detected syndromes as red circles. The binary numbers shown to the right is the diagnosis vector of the physical errors. The neural network learns the relation between the location of the detected syndromes and the diagnosis vector. (c) The real-valued diagnosis vector is predicted by the neural decoder. (d) With the syndrome pattern, faithful diagnosis vector is either 10000 or 01111. The chosen faithful diagnosis vector is 10000. Accordingly, we choose the recovery operator shown in the figure. In this case, the decoding succeeds.

used an approach similar to the example shown in Sec 4.2.2 in the case of $k = 1$. In this method, a linear map is used for the pure error, which is called a simple decoder. The pure error is then written in the form $\boldsymbol{t}(\boldsymbol{s})^{\mathrm{T}} = T\boldsymbol{s}$, where T is a $2n \times (n-k)$ matrix satisfying $H_c \Lambda T = I$. The label vector used in this method can essentially be regarded as being generated by a diagnosis matrix defined by

$$H_g = \begin{pmatrix} \boldsymbol{l}_{01} \\ \boldsymbol{l}_{10} \\ \boldsymbol{l}_{11} \end{pmatrix} (I \oplus \Lambda T H_c). \tag{4.63}$$

We see this is faithful and decomposable constructions. Let a generator matrix G be

$$G = \begin{pmatrix} \boldsymbol{l}_{01} \\ \boldsymbol{l}_{10} \end{pmatrix}. \tag{4.64}$$

Then, a diagnosis generated from the diagnosis matrix is

$$\boldsymbol{g} = H_g \Lambda \boldsymbol{e} = \begin{pmatrix} w(\boldsymbol{e})_1 \\ w(\boldsymbol{e})_0 \\ w(\boldsymbol{e})_0 \oplus w(\boldsymbol{e})_1 \end{pmatrix}, \tag{4.65}$$

where $\boldsymbol{w}(\boldsymbol{e}) = (w(\boldsymbol{e})_0, w(\boldsymbol{e})_1)$. The method in Ref. [55] uses a different set of label vectors $\boldsymbol{g}'$ called one-hot representation, which has a one-to-one correspondence with $\boldsymbol{g}$ as

$$\boldsymbol{g} = (0,0,0)^{\mathrm{T}} \mapsto \boldsymbol{g}' = (1,0,0,0)^{\mathrm{T}} \tag{4.66}$$
$$\boldsymbol{g} = (0,1,1)^{\mathrm{T}} \mapsto \boldsymbol{g}' = (0,1,0,0)^{\mathrm{T}} \tag{4.67}$$
$$\boldsymbol{g} = (1,0,1)^{\mathrm{T}} \mapsto \boldsymbol{g}' = (0,0,1,0)^{\mathrm{T}} \tag{4.68}$$
$$\boldsymbol{g} = (1,1,0)^{\mathrm{T}} \mapsto \boldsymbol{g}' = (0,0,0,1)^{\mathrm{T}}. \tag{4.69}$$

The above relation as real vectors can be written as

$$\boldsymbol{g}' = \frac{1}{2} \begin{pmatrix} -1 & -1 & -1 & 1 \\ -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{g} \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{4.70}$$

Since it is an isometric affine transformation, we expect that this transformation has little effect on the performance of the supervised machine learning. The matrix H_g is faithful and decomposable, but its normalized sensitivity is $O(1)$. We thus expect that this decoder becomes near-optimal when the training is ideally performed, but the prediction is not robust when the size of the training data set is small.

The method proposed by Baireuther *et al.* [56] mainly focuses on a model applicable to quantum error correction when we perform various counts of repetitive stabilizer measurements by utilizing recurrent neural network. They use the

commutation relation between the physical error and a logical Z operator as the label, since they only concerned about the logical bit-flip probability with the fixed initial state in the logical space. We can thus consider this method as a case of the linear prediction framework.

Torlai *et al.* [54], Krastanov *et al.* [57], and Breuckmann *et al.* [58] took a different approach from the above two [55, 56]. They used the binary representation of the physical error as the label vector. In the linear prediction framework, it corresponds to a choice of $H_g = \Lambda$ leading to

$$\boldsymbol{g} = H_g \Lambda \boldsymbol{e}^{\mathrm{T}} = \boldsymbol{e}^{\mathrm{T}} \tag{4.71}$$

Since H_g is not faithful, it cannot constitute an optimal decoder even with the delta diagnosis decoder. Interestingly, the delta diagnosis decoder with this choice of H_g works as an MD decoder, which can be shown by the following lemma.

Lemma 4.5.1. If the matrix H_{cg} has rank $2n$ in GF(2), there exists a map $\boldsymbol{r}^*(\boldsymbol{g}, \boldsymbol{s})$ such that the decoder with $\boldsymbol{r}(\boldsymbol{s}) = \boldsymbol{r}^*(\boldsymbol{g}^{(\delta)}(\boldsymbol{s}), \boldsymbol{s})$ works as an MD decoder for arbitrary distribution $\{p_{\boldsymbol{e}}\}$. If H_{cg} does not have rank $2n$, no such map exists.

Proof. If H_{cg} has rank $2n$, there exists a left inverse binary matrix H_{cg}^{-1} such that $H_{cg}^{-1} H_{cg} = I$. Then, we can obtain the physical error $\boldsymbol{e}$ as

$$\Lambda H_{cg}^{-1} \begin{pmatrix} \boldsymbol{s} \\ \boldsymbol{g} \end{pmatrix} = \boldsymbol{e}^{T}. \tag{4.72}$$

Thus, we can obtain the most probable physical error $\boldsymbol{e}^*(\boldsymbol{s})$ from the most probable diagnosis.

If H_{cg} does not have rank $2n$ in GF(2), there exist two physical errors which generate the same pair of syndrome and diagnosis. We cannot determine which is more probable. Thus, we cannot perform MD decoding when H_{cg} does not have rank $2n$. □

A drawback in this approach is difficulty arising when we replace a loss function with a practical one such as L2 distance. In order to satisfy a decomposable property in MD decoding, the length of the diagnosis must be no shorter than 2^{n+k} since there are 2^{n+k} possible candidates of the most probable physical error. This is not practical when the distance is large, and thus it requires heuristics such as repetitive sampling.

4.6 Numerical Result

We numerically show that the uniform data construction improves the performance of the neural decoder in the case of $k = 1$. We trained a prediction model called multi-layer perceptron (MLP) with the uniform data construction, and compare

it with other data constructions of the neural decoders with the MLP model. See Appendix D for the definition of the multi-layer perceptron. We also make a comparison with known decoders such as the MD decoder and the MWPM decoder. We choose the $[[d^2, 1, d]]$ surface code for the comparison, since most of the existing methods were benchmarked with this code. We calculated the performance for two types of error models, the bit-flip noise and depolarizing noise. The probability distribution of the bit-flip noise is described as follows.

$$p_{\boldsymbol{e}} = \begin{cases} p^{w(\boldsymbol{e})}(1-p)^{n-w(\boldsymbol{e})} & \forall i > n, e_i = 0 \\ 0 & \text{otherwise} \end{cases}, \tag{4.73}$$

where p is an error probability per physical qubit, and $w(\boldsymbol{e})$ is a weight of physical error $\boldsymbol{e}$ defined in Eq. (2.21). The probability distribution of the depolarizing noise is described as follows.

$$p_{\boldsymbol{e}} = (p/3)^{w(\boldsymbol{e})}(1-p)^{n-w(\boldsymbol{e})}. \tag{4.74}$$

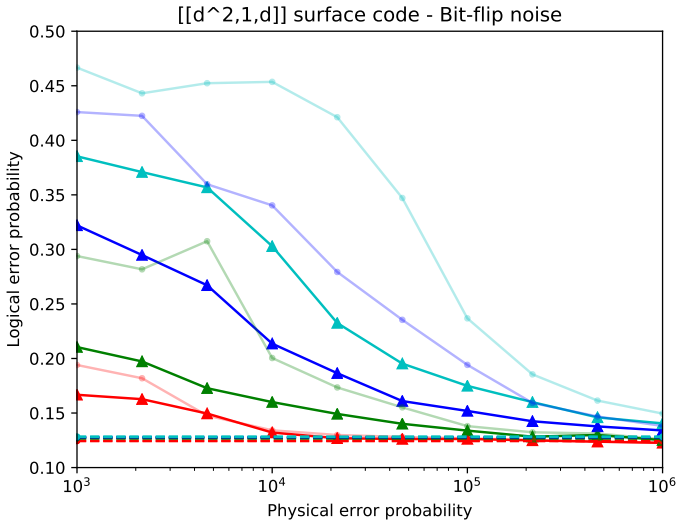
Note that the occurrences of the bit-flip and phase-flip errors are correlated in the depolarizing noise.

We first calculated the performance when the physical error probability is around the error threshold, namely, $p = 0.1$ for the bit-flip noise and $p = 0.15$ for the depolarizing noise. The specific construction of the MLP model is optimized for each noise model and for each size of the training data set. See Appendix D for the details of the parameter optimization and implementation.

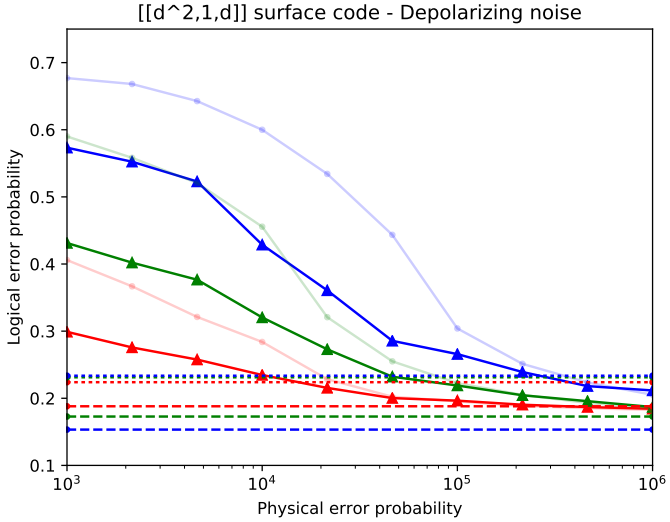
The performance of the neural decoder under the bit-flip noise is shown in Fig. 4.2(a). The solid lines are the performance of the neural decoder with the uniform data construction. The bottom dashed lines represent the logical error probability achievable with the MD decoder. The colors red, green, blue, and cyan correspond to distances 5, 7, 9, 11, respectively.

Comparing these two types of decoders, we see that the logical error probability of the neural decoder is near-optimal with 10^6 data set at distance 11. On the other hand, there are gaps between the converged logical error probabilities of the neural decoder and that of the MD decoder when the distance is large. These gaps can be improved by explicitly utilizing the spatial information of the topological codes. This point is discussed in Ref. [83].

We also implemented the neural decoder with short diagnosis, i.e., the construction with $N_{01} = N_{10} = N_{11} = 1$, where $N_{\boldsymbol{w}}$ is a number of logical operators in the rows of H_g corresponding to the class $\boldsymbol{w}$. This is equivalent to the construction which we showed as an example in Sec 4.2.2. We call this construction, with the normalized sensitivity of $O(1)$, as short diagnosis construction, which is shown as the pale plots in Fig. 4.2(a). Note that the performance of this decoder depends on the choice of the logical operators. We have tried this construction with various choice of the logical operators. The plotted data is the best among our trials. Although both constructions become near-optimal in the limit of large

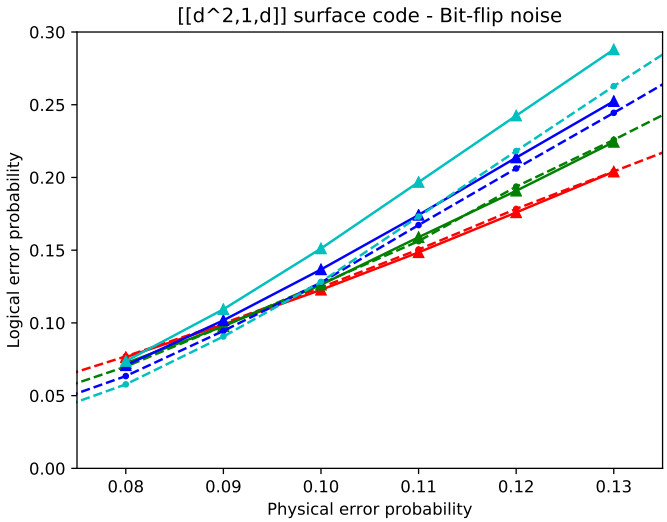


(a)

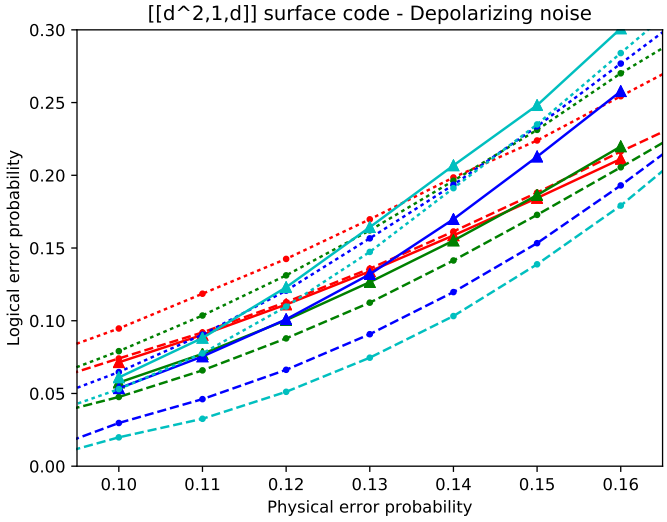


(b)

Figure 4.2: The performance comparison between the neural decoder with the uniform construction (solid lines) and that with short diagnosis construction (pale lines), the MD decoder (dashed lines), and the MWPM decoder (dotted lines) in the case of the $[[d^2, 1, d]]$ surface code. The logical error probabilities are plotted against of the sizes of the training data set with the fixed physical error probability p. We calculated the performance for distances $d = 5$ (red), 7 (green), 9 (blue), and 11 (cyan). (a) The case for the bit-flip noise with $p = 0.1$. Note that there are no lines of MWPM decoder since the MWPM decoder is equivalent to the MD decoder in this setting. (b) The case for the depolarizing noise with $p = 0.15$.

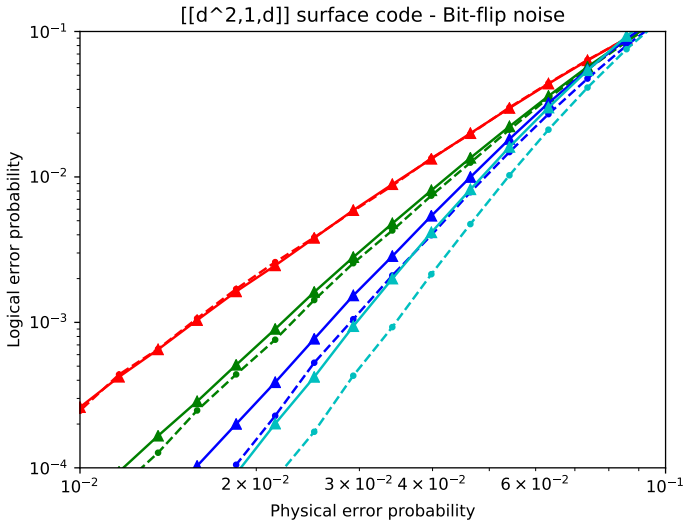


(a)

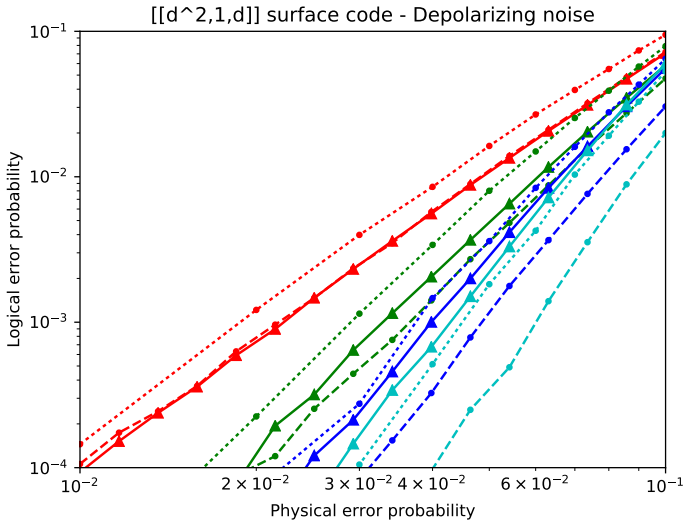


(b)

Figure 4.3: The performance comparison between the neural decoder with the uniform construction (solid lines), the MD decoder (dashed lines), and the MWPM decoder (dotted lines) in the case of the $[[d^2, 1, d]]$ surface code. We calculated the performance for distances $d = 5$ (red), 7 (green), 9 (blue), and 11 (cyan) with the same 10^6 training data set. (a) The case of the bit-flip noise. (b) The case of the depolarizing noise.



(a)



(b)

Figure 4.4: The performance comparison between the neural decoder with the uniform construction (solid lines), the MD decoder (dashed lines), and the MWPM decoder (dotted lines) in the case of the $[[d^2, 1, d]]$ surface code. The neural decoder is trained with the 10^6 training data set. We calculated the performance for distances $d = 5$ (red), 7 (green), 9 (blue), and 11 (cyan). (a) The case of the bit-flip noise. The training data set is generated at the physical error probability $p = 0.08$. (b) The case of the depolarizing noise. The training data set is generated at the physical error probability $p = 0.11$.

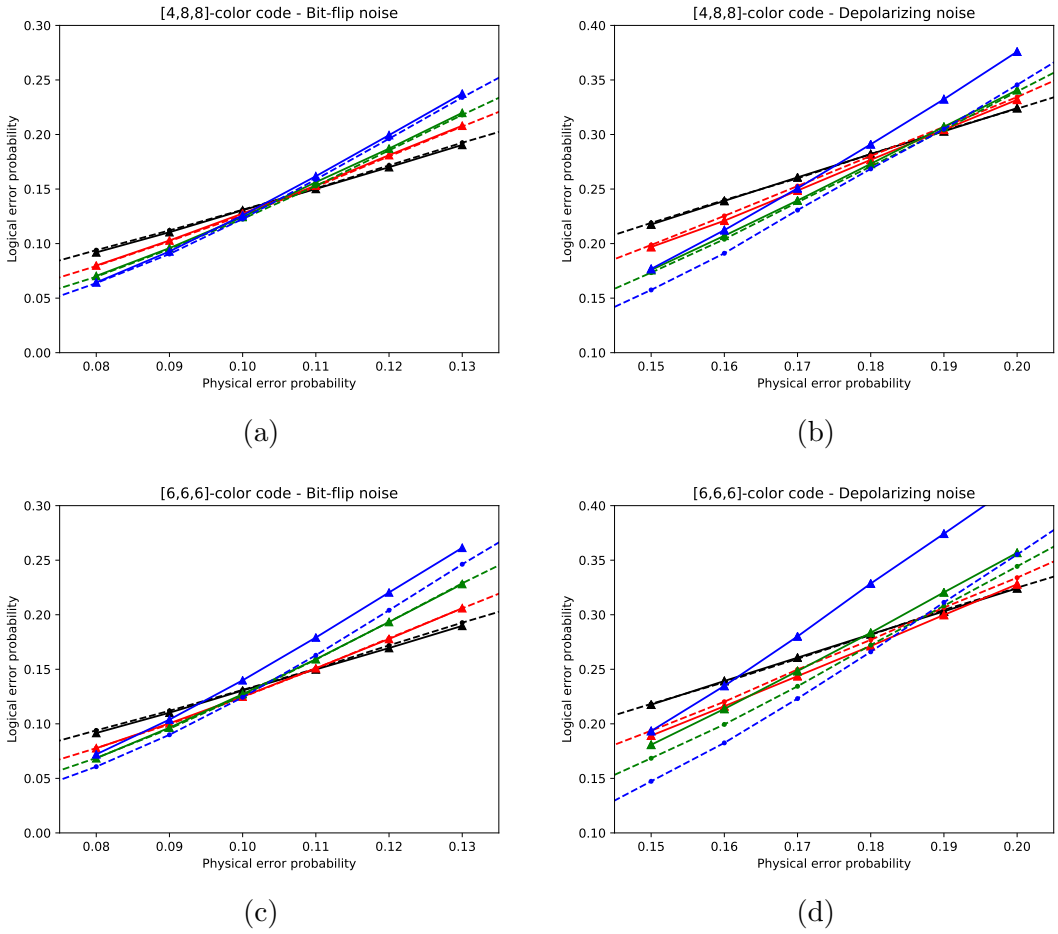


Figure 4.5: The performance comparison between the neural decoder with the uniform construction (solid lines), and the MD decoder (dashed lines) in the color codes. We calculated the performance for distances $d = 3$ (black), 5 (red), 7 (green), and 9 (blue) with the 10^6 training data set. (a) The case of the bit-flip noise in the [4,8,8]-color code. (b) The case of the depolarizing noise in the [4,8,8]-color code. (c) The case of the bit-flip noise in the [6,6,6]-color code. (d) The case of the depolarizing noise in the [6,6,6]-color code.

training data size, we see that the performance with the uniform data construction achieves smaller logical error probability than that with the short diagnosis construction for any size of the training data set. We have also confirmed that the performance of the neural decoder degrades when the row vectors of H_g consist of the same $O(d)$ logical operators of X, Y and Z. In this case, while the number of the rows in H_g is the same as that of the uniform data construction, the sensitivity $m(H_g)$ becomes $O(d)$, which makes the normalized sensitivity $\frac{m(H_g)}{M(H_g)}$ to be $O(1)$. Though these results are not plotted, the performance of this construction is almost the same as the short diagnosis construction. These results support our argument that it is essential for the performance of the neural decoder to minimize the normalized sensitivity.

The results with the depolarizing noise are shown in Fig. 4.2(b). Note that for the surface code under correlated noise such as the depolarizing noise, it is not known how an efficient MD decoder can be constructed. We see that the performance of the neural decoder becomes near-optimal, and is superior to that of the MWPM decoder with 10^6 training samples at $d = 5, 7, 9$.

We also calculated the logical error probability in terms of the physical error probability. The plots in the vicinity of the threshold value are shown in Fig. 4.3. We chose the size of the training data set as 10^6, and calculated the performance for the distance $d = 5, 7, 9, 11$ and for the bit-flip and depolarizing noises. Note that as for not trainable parameters about the MLP model, such as the number of model parameters, we used the best configurations in the calculation of Fig. 4.2 when the size of the training data set is 10^6. See Appendix D for these untrainable parameters called hyper-parameter. For both of the noise models, the performance is near-optimal when the distance is small. On the other hand, when the distance becomes large, the logical error probability becomes larger than that of the MWPM decoder. The error threshold is usually estimated with the cross point of the performance in terms of the distance. We see that the error threshold based on the distance is worse than that of the MWPM decoder, though the logical error probability is smaller than that of the MWPM decoder.

The actual experiment is expected to be performed with a physical error probability sufficiently smaller than the threshold value. Therefore, we calculated the performance of the decoder with a small physical error probability. The numerical results are shown in Fig. 4.4. Since the training data set generated with a small value of p is highly imbalanced, we trained the model with $p = 0.08$ for the bit-flip noise model, and with $p = 0.11$ for the depolarizing noise model. Then, we tested the trained models with the data set generated with $p \leq 0.1$. We see that the logical error probability is smaller than the MWPM decoder in this region, for all the distances except $d = 11$.

We also calculated the performance of the neural decoder for two types of color codes. We chose the size of training data set as 10^6, and calculated the logical error probability for the distance $d = 3, 5, 7, 9$. Note that we cannot construct an efficient MD decoder in the color code even under independent bit-flip and

phase-flip noise. The plots of the logical error probability to the physical error probability p are shown in Fig. 4.5. The configurations of the plots and lines are the same as that for the surface code. In the case of the bit-flip noise, the near-optimal performance is achieved. The performance is also near-optimal in the case of the depolarizing noise at distances except $d = 9$. We also see that the performance of the [4,8,8]-color code is better than that of [6,6,6]-color code. We speculate that this is because the number of the physical qubits required in the [4,8,8]-color codes is smaller than that of the [6,6,6]-color code at the same distance. These results suggest that the neural decoder with the uniform data construction is effective also for the color codes.

4.7 Conclusion

In this chapter, we theoretically analyzed the mechanism of machine-learning-based decoders for QEC, and proposed a general direction to construct the tasks of machine learning. Then, we have numerically shown that our direction is effective compared with the previously known approaches.

Since the formalism of the machine learning is flexible, there are many possible ways to reduce the decoding problem in QEC to the task of the machine learning. In order to clarify what is the best way of reduction, we introduced the linear prediction framework. This framework essentially includes the existing methods as specific cases, and enables us to discuss conditions for satisfying natural requirements for a good decoder for QEC. In particular, we have derived the condition to perform the optimal decoding in the limit of a large training data size. We also introduced a measure, normalized sensitivity, which represents a properly-scaled bound on the deviation in the prediction target resulting from a small change in the physical error pattern. We proposed to use this measure as a criterion for constructing a better decoder. We then proposed specific constructions of the data set, uniform data construction, for surface codes and color codes. We numerically confirmed that the performance of the neural decoder is improved with the uniform data construction. Our decoder was found to be superior to known efficient decoders, such as the known neural decoders and the decoder based on the reduction to minimum-weight perfect matching. We also confirmed that the performance of our neural decoder is near-optimal in various situations by comparing it with the minimum-distance decoder, which is known to be near-optimal but not efficient in general. We also confirmed that the neural decoder can achieve near-optimal performance not only for surface codes but also for color codes.

Since using machine learning for QEC is an emergent field, there are still many possible extensions and directions of the neural decoders. As we detailed in Appendix D, the prediction time of the neural decoders is smaller than that of the MD decoder, but larger than that of the MWPM decoder in our desktop PC. Since the prediction of the neural decoders can be done with simple matrix multiplica-

tions and non-linear activations, the time for prediction can be further made short by using an optimized hardware, such as field-programmable gate array (FPGA). While we have discussed only a label linearly generated in GF(2), the performance may be improved further by allowing labels nonlinearly generated from the physical error. For example, the relation between the syndrome values and the weight of the physical error, which cannot be generated linearly in GF(2), can be trained and predicted independently with a neural network. Then, the recovery map can be predicted with the syndrome values and the predicted weight with another neural network. The linear prediction framework also limits the sample in the training data set to that is sampled from the assumed physical error distribution. However, the distribution which is the best for the training is not necessarily the same as the actual distribution. There can be a more artificial way to construct the training data set to achieve the performance with a smaller size of the training data set. In the numerical investigation, we observed that the required amount of the data set becomes exponentially large in terms of the distance. This may be suppressed by using more structured neural networks, such as convolutional neural networks, as a prediction model. When the stabilizer measurements themselves suffer from noise, stabilizer measurements are often repetitively performed during QEC. In such a case, the length of the syndrome data is not fixed. In our construction, we need to train the neural network again whenever the length of the syndrome data changes. The studies of Refs. [56, 58] focused on removing this drawback by utilizing recurrent neural network and convolutional neural network. Using the technique proposed in Refs. [56, 58], our neural decoder may be also applicable to the cases when we perform repetitive stabilizer measurements.

Chapter 5

Summary and outlook

5.1 Summary

In this thesis, we investigated a theory of QEC for near-term quantum devices. In order to build quantum computer in a scalable manner, QEC is a vital technique. Topological stabilizer codes are considered as the promising candidates of quantum error correcting codes, since they have preferable properties for experimental implementation. Therefore, current experiments towards scalable quantum computer pursue realization of QEC using topological stabilizer codes as a milestone. On the other hand, when we experimentally try to demonstrate QEC with more than about 50 qubits, we encounter several practical problems about QEC. We focused on two essential problems among them in this thesis.

One is computational hardness of accurate and efficient evaluation of topological stabilizer codes under practical noise models. When we design quantum devices for QEC, the performance of quantum error correcting codes under practical noise models is essential. Quantum circuits of any stabilizer code are described as Clifford circuits, and we can simulate any Clifford circuit efficiently using the Gottesman-Knill theorem. Therefore, if we can assume noise models which consist only of Clifford operations, we can efficiently simulate quantum circuits of quantum error correcting codes, and we can evaluate the performance of them. On the other hand, though coherent noise, which is practically caused by over-rotation due to experimental imperfection, is unavoidable in experiments, we cannot efficiently evaluate the performance of topological stabilizer codes under coherent noise models since coherent noise cannot be represented or approximated with Clifford operations. Thus, a novel scheme for evaluating the performance of topological stabilizer codes under coherent noise models is demanded. In Chapter 3, we proposed an efficient and accurate scheme for simulating quantum circuits of the surface code under coherent noise. Using this scheme, we can efficiently evaluate the performances of the surface code under coherent error, such as the logical error probability, the threshold value, and the dropping rate. Since coherent noise is unavoidable in experiments, our results are immediately useful for evaluating

the effect of the coherence in noise on the performance of QEC.

The key idea of our scheme is the use of matchgate circuits, which is an efficiently simulatable class of quantum circuits. Though a gate set of matchgate circuits cannot treat even single Pauli-Z operations, we showed that quantum circuits of the surface code under coherent noise can be represented as a matchgate circuit. We first constructed such a representation for the one-dimensional repetition code, which is a one-dimensional line of the surface code. We then numerically investigated the performance of the one-dimensional repetition code. We found that when the noise becomes fully coherent, the threshold value becomes one-third compared with the case of the incoherent noise. We observed that the dropping rate is also worsened when the noise becomes fully coherent, and confirmed that the conjectured universal relation between the dropping rate and the threshold value [Eq. (2.68)] also holds true for coherent noise models.

By extending the scheme for the one-dimensional repetition code, we showed that quantum circuits of the surface code under coherent noise can also be represented with a matchgate circuit. In the simulation of the surface code, we assumed a noise model such that there are all X-, Y-, and Z-type Pauli errors and the X-type errors can be coherent. Since this noise model contains coherent errors and contains both of bit- and phase-flip errors, our scheme enables the simulation of quantum circuits of the surface code which suffers from quantum and coherent errors.

Though our scheme covers a variety of coherent noise models, it does not support the simulation of quantum circuits of the surface code under an arbitrary coherent noise model. It is also not obvious whether our idea is applicable to other topological stabilizer codes. In order to obtain deeper understanding on how the coherence in errors reduces the threshold value, we considered an approximation based on a leading-order ansatz. We investigated the effect of the coherence in noise on the performance of QEC using a leading-order ansatz. The ansatz is based on a conjecture that the effect of coherence in noise can be attributed to an effective increase of a physical error probability under an incoherent noise model. As long as the latter model is efficiently simulatable, this approximation allows us to calculate the logical error probability efficiently. We confirmed that the approximated values in the one-dimensional repetition code agree well with the accurately estimated values. Since this approximation is expected to be used for an arbitrary coherent noise model and an arbitrary topological stabilizer code, we may efficiently evaluate the property of topological stabilizer codes under an arbitrary coherent noise model using the ansatz.

The other problem is the lack of a fast, reliable, and versatile decoder in QEC. The performance of a quantum error correcting code depends on the performance of a chosen decoder. Since we cannot perform the optimal decoding efficiently in general, a fast decoder which is applicable to various topological stabilizer codes under various noise models, and achieves a small logical error probability is demanded. Supervised machine learning provides generic instructions to construct a

fast and accurate prediction function through the training process using a training data set. Thus, we expect that a decoder which has preferable properties can be constructed using machine learning. When we construct a machine-learning-based decoder, one of the most essential factors that determine its performance is to choose which part of decoding algorithm to be delegated to machine learning. In Chapter 4, we focused on this problem. In order to understand this issue comprehensively, we proposed a general framework of the machine-learning-based decoder called linear prediction framework. In this general framework, we showed the diagnosis matrix, which characterize the label space, must be faithful and decomposable in order to achieve the optimal performance in the limit of a large training data set. We also proposed the criterion called normalized sensitivity which should be minimized for achieving near-optimal performance with a training data set of a practical size. We proposed uniform data constructions for the $[[2d^2 - 2d + 1, 1, d]]$ and $[[d^2, 1, d]]$ surface codes and the [6,6,6]- and [4,8,8]-color codes, which satisfy the faithful and decomposable conditions, and optimize the normalized sensitivity. We numerically showed that the machine-learning-based decoder constructed with the uniform data construction is superior to the previously known machine-learning-based decoders in the surface code under the bit-flip noise and the depolarizing noise. We also numerically showed that the performance of the proposed machine-learning-based decoder is near-optimal by comparing it with the minimum-distance decoder, which is known to be inefficient but near-optimal. We also numerically showed that the performance is also near-optimal for the color codes.

5.2 Outlook

In this thesis, we focused on the problems about QEC for near-term quantum devices, and proposed two solutions to the problems. We used two theories, matchgate circuits and machine learning, as essential tools for solving the problems. We believe that these tools are potentially useful for tackling problems other than QEC.

One example is a technique of calibrating experimental quantum gates. Though we assumed that the noise model is given with problems in Chapter 3 and 4, it is non-trivial how to characterize an actual behavior of a given quantum system. Obviously, finding an appropriate model is essential not only for evaluation in QEC, but also for calibrating and improving quantum controls. Even when the number of qubits is small, it is still hard to accurately estimate properties of the actual noise since we are only allowed to use noisy instruments for measuring properties of the noise in practice. For example, it is hard to distinguish a small physical error probability per quantum gate from noise due to state preparations and measurements. A method called randomized benchmarking [84, 85] enables us to accurately evaluate several properties of the actual noise such as a

very small physical error probability and a degree of coherence in noise. In the randomized benchmarking, the properties are extracted using a quantum circuit which consists of a sequence of randomly chosen Clifford operations. Thanks to a theoretically tractable property of random Clifford operations, we can characterize these properties with a practical number of experiments. By revealing properties of a sequence of operations randomly chosen from a gate set allowed in matchgate circuits, more helpful properties of the actual noise may be characterized efficiently. When the number of qubits becomes large, the actual noise behavior becomes complex due to cross-talks between physical qubits and those between quantum controls. A method to obtain a measure for precise calibration in such a large quantum system is proposed in Refs. [66, 86]. This method is also based on a property of random quantum circuits which consist of quantum gates chosen from a universal quantum gate set. Using this method, we can estimate an effective amount of very small noise in a large quantum system. On the other hand, since this method requires strong simulation of random universal quantum circuits with classical computer, this method is not applicable to a system with more than about 50 qubits. Thus, when properties of random circuits of efficiently simulatable classes are further studied, such a complex property of noise for a large quantum system may be characterized efficiently.

Though various tasks about large quantum systems may be solved with efficiently tractable theories such as Clifford circuits and matchgate circuits, there will always be other important tasks with no such efficient solution, since simulating generic quantum circuits is believed to be hard. In such a case, using heuristic methods is one of the best practical solutions, and machine learning will be a convenient way to find a heuristic function which can be computed fast and is expected to be reliable for various purposes. Supervised machine learning has provided practical solutions to a variety of tasks about quantum computation other than QEC, such as simulation of quantum systems [72] and calibration of quantum controls [71]. It is a definite merit of the machine learning that the same methodology is applicable to a wide variety of problems. Nonetheless, we have learned in Chapter 4 that in order to construct a prediction function to be reliable, it must be carefully considered how we should use machine learning for solving the tasks. In the case of decoding in QEC, we showed that a naive use of machine learning leads to degraded performance of machine-learning-based decoders even if we can assume an unlimited training data set. More generally saying, when we try to map our problem to a task of machine learning, deep understanding about what we must essentially extract from quantum systems is required. Therefore, we believe that understanding of fundamental relations between the problems of quantum computation and the framework of machine learning makes a technique of machine learning a more powerful and versatile tool for practical treatments of near-term quantum devices.

Appendix A

Supplemental materials about matchgate

A.1 Sufficient condition for fermionic Gaussian state

We show a fermionic Gaussian state ρ is pure if and only if its covariance matrix M satisfies $MM^{\mathrm{T}} = I$. We can transform any fermionic Gaussian state to a canonical form as follows. It is known that any anti-symmetric real-valued matrix G has an orthogonal matrix W such that

$$WGW^{\mathrm{T}} = \bigoplus_{i=0}^{n-1} \begin{pmatrix} 0 & \lambda_i \\ -\lambda_i & 0 \end{pmatrix}, \tag{A.1}$$

where $\lambda_i \in \mathbb{R}$. Thus, we can expand the form of a Gaussian state as

$$\begin{aligned} \rho &= C \exp\left(\frac{i}{2}\boldsymbol{c}^{\mathrm{T}} G \boldsymbol{c}\right) && \text{(A.2)} \\ &= C \prod_{i=0}^{n-1} (\cosh \lambda_i I + \sinh \lambda_i i d_{2i} d_{2i+1}), && \text{(A.3)} \end{aligned}$$

where

$$\boldsymbol{d} = (d_0, \ldots, d_{n-1})^{\mathrm{T}} := W^{\mathrm{T}} \boldsymbol{c}. \tag{A.4}$$

We say the form of Eq (A.3) as a canonical form of a fermionic Gaussian state ρ. Since $\mathrm{Tr}(\rho^2) = \mathrm{Tr}(\rho)$ if and only if $\tanh \lambda_i = \pm 1$ for all $i = 0, \ldots, n-1$, any pure fermionic Gaussian state is represented as a limit of $\lambda_i \to \pm\infty$.

For a pure Gaussian state ρ, using the canonical form of ρ, we can calculate

the element of covariance matrix $M_{ij}(\rho)$ as follows. We see

$$\frac{i}{2}\mathrm{Tr}(\rho\,[d_i, d_j]) = \begin{cases} \tanh\lambda_k & 2k = i, 2k+1 = j \\ -\tanh\lambda_k & 2k = j, 2k+1 = i \\ 0 & \text{otherwise} \end{cases}. \tag{A.5}$$

The covariance matrix M is obtained as

$$M = W^{\mathrm{T}} \bigoplus_{i=0}^{n-1} \begin{pmatrix} 0 & \tanh\lambda_i \\ -\tanh\lambda_i & 0 \end{pmatrix} W. \tag{A.6}$$

Since $WW^{\mathrm{T}} = I$, we see $MM^{\mathrm{T}} = I$ if and only if $\tanh\lambda_i = \pm 1$, which implies ρ is a pure Gaussian state.

A.2 Sufficient condition for fermionic Gaussian operation

In Chapter 3, we use the fact that a Kraus operator K which has the form of

$$K = \cos\theta I + \sin\theta c_i c_j \tag{A.7}$$

or

$$K = \frac{1}{2}(I + e^{i\theta} c_i c_j) \tag{A.8}$$

is Gaussian for arbitrary indices i, j and for an arbitrary real value $\theta \in \mathbb{R}$. The first one is trivial since

$$K = e^{-\theta c_i c_j} = \cos\theta I + \sin\theta c_i c_j. \tag{A.9}$$

The other is derived as

$$K = \frac{1}{\sqrt{2}(\cosh\phi - i\sinh\phi)} e^{\pm(\frac{\pi}{4} + i\phi)c_i c_j} \tag{A.10}$$

$$= \frac{1}{2}\left(I \pm \frac{1 + i\tanh\phi}{1 - i\tanh\phi} c_i c_j\right) \tag{A.11}$$

This operator covers $0 \le \theta < 2\pi$ of Eq. (A.8). Note that $\theta = \frac{\pi}{2}, \frac{3}{2}\pi$ is obtained as a limit of $\phi \to \pm\infty$.

Appendix B

Proof of the lemmas in Chapter 4

B.1 Proof of the converse part in Lemma 4.2.1

Here we prove the last statement of Lemma 4.2.1. When Eq. (4.4) does not hold, either (i) there exists $\boldsymbol{e}_1$ such that

$$\begin{aligned} \boldsymbol{e}_1 &\notin \mathcal{L}_0, & \text{(B.1)} \\ H_{cg}\Lambda \boldsymbol{e}_1^{\mathrm{T}} &= 0, & \text{(B.2)} \end{aligned}$$

or (ii) there exists $\boldsymbol{e}_1$ such that

$$\begin{aligned} \boldsymbol{e}_1 &\in \mathcal{L}_0, & \text{(B.3)} \\ H_{cg}\Lambda \boldsymbol{e}_1^{\mathrm{T}} &\neq 0. & \text{(B.4)} \end{aligned}$$

For (i), consider two probability distributions $\{p_{\boldsymbol{e}}\}$ and $\{p'_{\boldsymbol{e}}\}$ such that

$$\begin{aligned} \Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}[\boldsymbol{e} = 0 | \boldsymbol{s}(\boldsymbol{e}) = 0] &= 0.75, & \text{(B.5)} \\ \Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}[\boldsymbol{e} = \boldsymbol{e}_1 | \boldsymbol{s}(\boldsymbol{e}) = 0] &= 0.25, & \text{(B.6)} \end{aligned}$$

and

$$\begin{aligned} \Pr_{\boldsymbol{e}\sim\{p'_{\boldsymbol{e}}\}}[\boldsymbol{e} = 0 | \boldsymbol{s}(\boldsymbol{e}) = 0] &= 0.25, & \text{(B.7)} \\ \Pr_{\boldsymbol{e}\sim\{p'_{\boldsymbol{e}}\}}[\boldsymbol{e} = \boldsymbol{e}_1 | \boldsymbol{s}(\boldsymbol{e}) = 0] &= 0.75. & \text{(B.8)} \end{aligned}$$

An optimal decoder for each case succeeds with probability 0.75 given $\boldsymbol{s} = 0$. On the other hand, since $\boldsymbol{g}^{(\delta)}(0) = 0$ in both cases, only the value of $\boldsymbol{r}^*(0,0)$ is relevant. Since $\boldsymbol{w}(0) \neq \boldsymbol{w}(\boldsymbol{e}_1)$, any choice of $\boldsymbol{r}^*(0,0)$ leads to a success probability no greater than 0.25 for at least one of the cases.

For (ii), choose $\boldsymbol{w} \neq 0$, and if $H_g\Lambda(\boldsymbol{w}G)^{\mathrm{T}} \neq 0$, define

$$\boldsymbol{e}_2 := \boldsymbol{w}G. \quad \text{(B.9)}$$

Otherwise, define

$$\boldsymbol{e}_2 := \boldsymbol{e}_1 \oplus \boldsymbol{w}G. \tag{B.10}$$

It ensures that $\boldsymbol{s}(\boldsymbol{e}_2) = 0$ and $\boldsymbol{g}_2 := H_g \Lambda \boldsymbol{e}_2 \neq 0$. Consider two probability distributions $\{p_{\boldsymbol{e}}\}$ and $\{p'_{\boldsymbol{e}}\}$ such that

$$\begin{aligned}
\Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}[\boldsymbol{e} = 0 | \boldsymbol{s}(\boldsymbol{e}) = 0] &= 0.4, & \text{(B.11)}\\
\Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}[\boldsymbol{e} = \boldsymbol{e}_1 | \boldsymbol{s}(\boldsymbol{e}) = 0] &= 0.0, & \text{(B.12)}\\
\Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}}[\boldsymbol{e} = \boldsymbol{e}_2 | \boldsymbol{s}(\boldsymbol{e}) = 0] &= 0.6, & \text{(B.13)}
\end{aligned}$$

and

$$\begin{aligned}
\Pr_{\boldsymbol{e}\sim\{p'_{\boldsymbol{e}}\}}[\boldsymbol{e} = 0 | \boldsymbol{s}(\boldsymbol{e}) = 0] &= 0.3, & \text{(B.14)}\\
\Pr_{\boldsymbol{e}\sim\{p'_{\boldsymbol{e}}\}}[\boldsymbol{e} = \boldsymbol{e}_1 | \boldsymbol{s}(\boldsymbol{e}) = 0] &= 0.3, & \text{(B.15)}\\
\Pr_{\boldsymbol{e}\sim\{p'_{\boldsymbol{e}}\}}[\boldsymbol{e} = \boldsymbol{e}_2 | \boldsymbol{s}(\boldsymbol{e}) = 0] &= 0.4. & \text{(B.16)}
\end{aligned}$$

An optimal decoder for each case succeeds with probability 0.6 given $\boldsymbol{s} = 0$. On the other hand, since $\boldsymbol{g}^{(\delta)}(0) = \boldsymbol{g}_2$ in both cases, only the value of $\boldsymbol{r}^*(\boldsymbol{g}_2, 0)$ is relevant. Since $\boldsymbol{w}(0) \neq \boldsymbol{w}(\boldsymbol{e}_2)$, any choice of $\boldsymbol{r}^*(\boldsymbol{g}_2, 0)$ leads to a success probability no greater than 0.4 for at least one of the cases.

B.2 Proof of the converse part in Lemma 4.2.2

When the diagnosis matrix is not decomposable, there exists a non-empty subset $\mathcal{W} \subset \{0,1\}^{2k}$ such that

$$\sum_{\boldsymbol{w}\in\mathcal{W}} \alpha_{\boldsymbol{w}} \boldsymbol{g}_0(\boldsymbol{w}) = \sum_{\boldsymbol{w}\in\{0,1\}^{2k}\setminus\mathcal{W}} \beta_{\boldsymbol{w}} \boldsymbol{g}_0(\boldsymbol{w}), \tag{B.17}$$

where $\alpha_{\boldsymbol{w}}, \beta_{\boldsymbol{w}} \geq 0$ and

$$\Gamma := \sum_{\boldsymbol{w}\in\mathcal{W}} \alpha_{\boldsymbol{w}} = \sum_{\boldsymbol{w}\in\{0,1\}^{2k}\setminus\mathcal{W}} \beta_{\boldsymbol{w}} > 0. \tag{B.18}$$

Consider two probability distributions $\{p_{\boldsymbol{e}}\}$ and $\{p'_{\boldsymbol{e}}\}$ such that

$$\Pr_{\boldsymbol{e}\sim\{p_{\boldsymbol{e}}\}_A}[\boldsymbol{w}(\boldsymbol{e}) = \boldsymbol{w}, \boldsymbol{l}(\boldsymbol{e}) = \boldsymbol{l} | \boldsymbol{s}(\boldsymbol{e}) = 0] = \begin{cases} \alpha_{\boldsymbol{w}}/\Gamma & \boldsymbol{w} \in \mathcal{W}, \boldsymbol{l} = 0 \\ 0 & \text{otherwise} \end{cases}, \tag{B.19}$$

$$\Pr_{\boldsymbol{e}\sim\{p'_{\boldsymbol{e}}\}_B}[\boldsymbol{w}(\boldsymbol{e}) = \boldsymbol{w}, \boldsymbol{l}(\boldsymbol{e}) = \boldsymbol{l} | \boldsymbol{s}(\boldsymbol{e}) = 0] = \begin{cases} \beta_{\boldsymbol{w}}/\Gamma & \boldsymbol{w} \notin \mathcal{W}, \boldsymbol{l} = 0 \\ 0 & \text{otherwise} \end{cases}. \tag{B.20}$$

From Eq. (B.17), the L2 diagnosis vector $\boldsymbol{g}^{(\mathrm{L2})}(0)$ is identical for the two distributions. On the other hand, the most probable class $\boldsymbol{w}$ is different for the two probability distributions. This means that a single decoder cannot perform the optimal decoding for both of the two distributions.

Appendix C

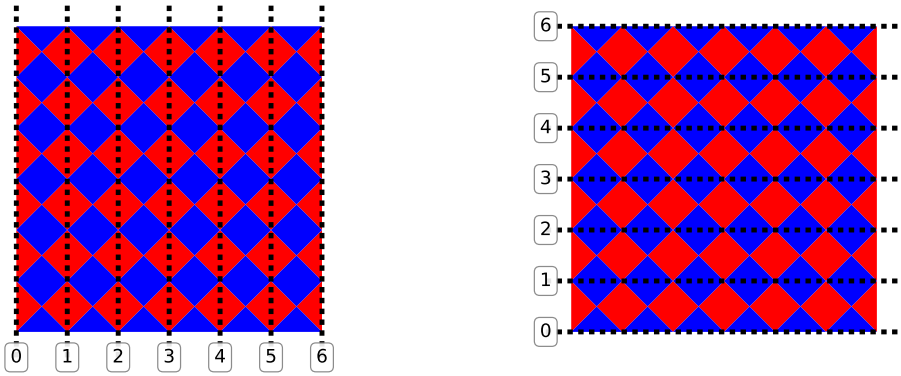
Uniform data construction for surface codes and color codes

We have introduced the uniform data construction in Chapter 4. In this appendix, we show specific uniform data construction for the surface and color codes.

We choose $3d$ logical operators for the $[[2d^2-2d+1, 1, d]]$ surface code by using two patterns as shown in Fig. C.1. For pattern 1, each dotted line corresponds to a logical X operator, which is the product of the Pauli Z operators on the vertices on the line. For pattern 2, each dotted line corresponds to a logical Z operator, which is the product of the Pauli X operators on the vertices on the line. We choose d logical Y operators written as the product of the i-th logical X operator and the i-th logical Z operator for $i = 0, \ldots, d-1$. We choose $3d$ logical operators for the $[[d^2, 1, d]]$ surface code with two patterns as shown in Fig. C.2. The rule of choice is the same as that of the $[[d^2, 1, d]]$ surface code.

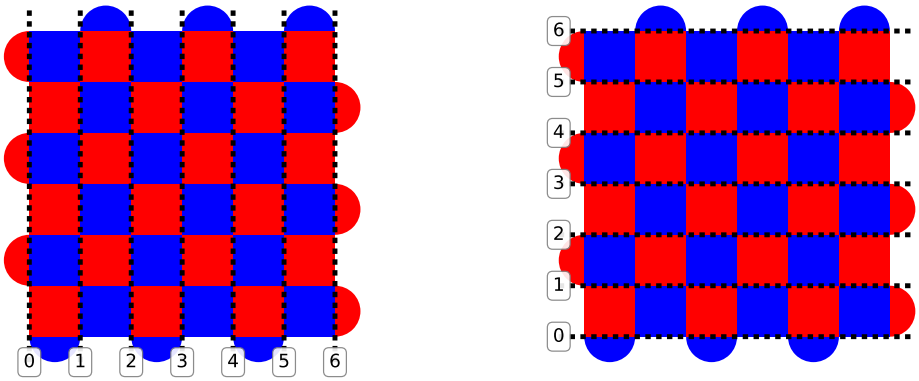
We choose $\frac{9}{2}(d+1)$ logical operators for the [6,6,6]-color code as shown in Fig. C.3. There are $\frac{1}{2}(d+1)$ lines for each pattern. In all of the three patterns, each line corresponds to the logical X-, Z-, and Y-Pauli operators on the physical qubits on the line. We choose $6(d+1)$ logical operators for the [4,8,8]-color code as shown in Fig. C.4. There are $\frac{1}{2}(d+1)$ lines for each pattern. The choice of the logical operators is the same as that of the [6,6,6]-color codes.

In all the patterned choice of the logical operators, we can verify that the sensitivity is constant, since every physical qubit is measured by at most constant number of logical operators. On the other hand, the minimum boundary distance is scaled as $O(d)$, since the same number $O(d)$ of logical X-, Y-, and Z-operators are used. Thus, the normalized sensitivity is scaled as $O(d^{-1})$ with these choices.



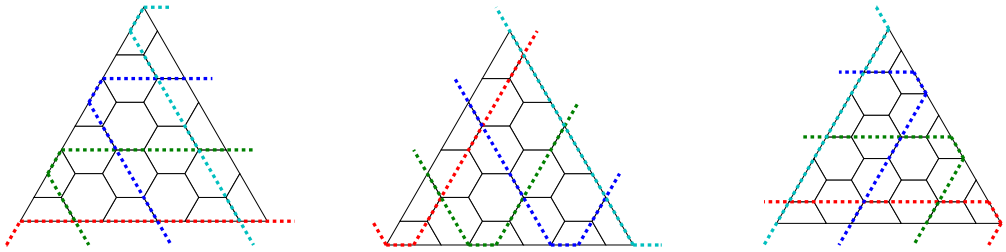
(a) Pattern 1 (b) Pattern 2

Figure C.1: Logical operators used for the construction of a diagnosis matrix for the surface codes $[[2d^2 - 2d + 1, 1, d]]$. Each dotted black line corresponds to a chosen logical operator.



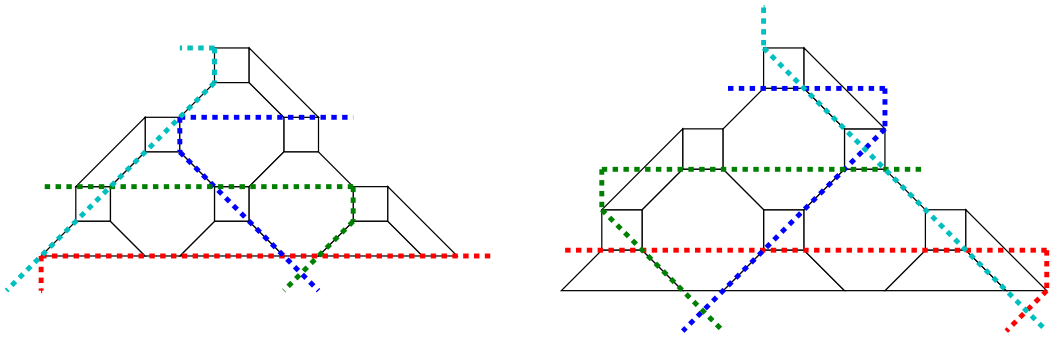
(a) Pattern 1 (b) Pattern 2

Figure C.2: Logical operators used for the construction of a diagnosis matrix for the surface codes $[[d^2, 1, d]]$. Each dotted black line corresponds to a chosen logical operator.



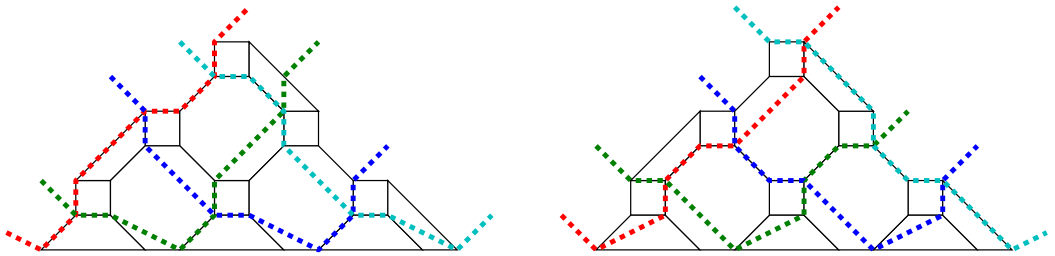
(a) Pattern 1 (b) Pattern 2 (c) Pattern 3

Figure C.3: Logical operators used for the construction of a diagnosis matrix for the [6,6,6]-color codes. Each colored line corresponds to chosen logical operators. The lines are colored only for visibility, and are not related to the colors of color codes.



(a) Pattern 1

(b) Pattern 2



(c) Pattern 3

(d) Pattern 4

Figure C.4: Logical operators used for the construction of a diagnosis matrix for the [4,8,8]-color codes. Each colored line corresponds to chosen logical operators. The lines are colored only for visibility, and are not related to the colors of color codes.

Appendix D

Implementations of decoders

D.1 Neural decoder

D.1.1 Definition of multi-layer perceptron

We used a multi-layer perceptron as a prediction model in Chapter 4. A multi-layer perceptron model is defined as follows. Suppose the feature space is $\mathcal{X} = \{0,1\}^{n_0}$ and the prediction space is $\mathcal{Y}' = \mathbb{R}^{n_l}$, where l is a positive integer, and $\{n_i\}$ $(0 \leq i \leq l)$ is a set of integers. We consider a function $\hat{M}$ such that

$$\hat{M}(\boldsymbol{x}, \boldsymbol{\theta}) = \boldsymbol{h}_l(\boldsymbol{x}, \boldsymbol{\theta}), \tag{D.1}$$

$$\boldsymbol{h}_i(\boldsymbol{x}, \boldsymbol{\theta}) = \begin{cases} \boldsymbol{x} & i = 0 \\ \phi_{i-1}(A_{i-1}\boldsymbol{h}_{i-1}(\boldsymbol{x}, \boldsymbol{\theta}) + \boldsymbol{b}_{i-1}) & l \geq i \geq 1 \end{cases}, \tag{D.2}$$

where $\{\phi_i\}$ is a set of element-wise non-linear functions for a real-valued vector, $\{A_i\}$ is a set of $n_{i+1} \times n_i$ real-valued matrices, $\{\boldsymbol{b}_i\}$ is a set of real-valued vectors such that $|\boldsymbol{b}_i| = n_{i+1}$, and a set of model parameters $\boldsymbol{\theta}$ consists of all the elements of $\{A_i\}$ and $\{\boldsymbol{b}_i\}$.

Each vector $\boldsymbol{h}_i$ $(0 \leq i \leq l)$ is called a layer, and each real-valued element of layers is called a neuron. The layers except $\boldsymbol{h}_0$ and $\boldsymbol{h}_l$ are called hidden layers. The whole function $\hat{M}$ is called a multi-layer perceptron (MLP). According to the universal approximation theorem [87], when chosen non-linear functions $\{\phi_0, \ldots, \phi_{l-1}\}$ are non-constant, bounded, and monotonically-increasing continuous functions, for an arbitrary real value $\epsilon > 0$ and for an arbitrary function $F : \mathcal{X} \to [0,1]^{n_l}$, there exists a set of finite integers $\{n_1, \ldots, n_{l-1}\}$ and model parameters $\boldsymbol{\theta}$ such that $|\hat{M}(\boldsymbol{x}, \boldsymbol{\theta}) - F(\boldsymbol{x})| < \epsilon$ for all $\boldsymbol{x}$ if $l \geq 2$. Therefore, we can expect that the MLP model with $l \geq 2$ has a sufficient representation power for our purpose by constructing the model with a set of sufficiently large integers $\{n_1, \ldots, n_{l-1}\}$.

D.1.2 Detailed settings of multi-layer perceptron

In this subsection, we describe the detailed implementation of the MLP model and training process. We chose a rectified linear unit function ($\mathrm{ReLU}(\boldsymbol{x})_i = \max(0, x_i)$) as the element-wise non-linear function for $\phi_0, \ldots \phi_{l-2}$, and a sigmoid function ($\mathrm{Sigmoid}(\boldsymbol{x})_i = 1/(1 + e^{-x_i})$) for ϕ_{l-1}. We assume that all the hidden layers have the same number of neurons h, i.e., $n_i = h$ for all i except 0 and l. As a loss optimization algorithm $\hat{O}$, we used the following process. We shuffled the order of the training data set T, and split it to exclusive subsets of the same size. Each subset is called a batch, and the number of samples per batch $|T|_b$ is called a batch size. We can extract $\lfloor |T|/|T|_b \rfloor$ batches from a training data set T. We computed the loss for a batch, and computed its gradients to the model parameters $\boldsymbol{\theta}$. Then, we updated $\boldsymbol{\theta}$ with Adam optimization method [88] using the computed gradients. We controlled the speed of optimization with a coefficient λ, which is called a learning rate. We sequentially performed the above update of $\boldsymbol{\theta}$ for every batch. This set of the $\lfloor |T|/|T|_b \rfloor$ updates of $\boldsymbol{\theta}$ is called an epoch. We repeated the above process until the logical error probability for $\boldsymbol{\theta}$ converges to a certain value. In order to avoid over-fitting of the model to a small training data set, we used a technique called L2 regularization, i.e., we added L2 norms of the model parameters as an extra loss in the training process with the coefficient β, which is called a regularization parameter. In order to accelerate the training, we also used a technique called batch normalization.

There are tunable parameters which are important but not included in the model parameters: the number of hidden layers $(l-1)$, the number of neurons per layer h, the batch size $|T|_b$, the regularization parameter for the L2 regularization β, and the schedule of the learning rate λ. These parameters are called hyper-parameters. As for the learning rate λ, we started the training with $\lambda = 10^{-3}$, and it was decreased to $\lambda = 10^{-5}$ according to the count of epochs. As for the other hyper-parameters, we calculated the logical error probability for all the combination of $(l-1) \in \{2, 3, 4\}$, $h \in \{d^2, d^3, d^4\}$, $|T|_b \in \{100, 500\}$, and $\beta \in \{0, 0.01, 0.1\}$. Note that in the case of $d = 11$, we tuned h by hand since we cannot choose $h = d^4$ due to the memory limit of devices. Then, we chose the combination of the hyper-parameters which shows the best logical error probability. When we compared the performance in terms of the combination of the hyper parameters, we calculated the logical error probability with an independently generated data set of the size 10^5, which is called a validation data set. We optimized these hyper-parameters for each construction of the diagnosis matrix, distance, physical error probability, error model, and size of the training data set in the calculation for Fig. 4.2 and Fig. 4.5. The optimized hyper-parameters are re-used in the calculation for Fig. 4.3 and Fig. 4.4. Since the optimized set of the hyper-parameters is over-fitted to the validation data set, the plotted logical error probability in the main text was calculated using another independently generated data set of the size 10^6, which is called a test data set.

D.2 Other decoders

D.2.1 Minimum-weight perfect matching decoder

The minimum-distance decoder of the surface code under uncorrelated noise models such as bit-flip noise model can be implemented by reducing the problem into minimum-weight perfect matching. See Sec 2.2.2 for the reduction to minimum-weight perfect matching.

D.2.2 Inefficient minimum-distance decoder

As far as we can assume probabilistic Pauli noise, we can reduce any decoding problem of the minimum-distance decoding to an instance of binary programming. Though binary programming is known to be computationally hard, we can solve it when the distance is small. In the case of the stabilizer codes, we can formulate the problem as

$$\text{Minimize } w(\boldsymbol{e}) \text{ s.t. } H_c \Lambda \boldsymbol{e}^{\mathrm{T}} = s. \tag{D.3}$$

We solved this problem for each sample using an integer-programming solver. In the numerical results in the main text, we obtained at least 10^5 samples for each plot. In all the cases, the solver reached the optimal solution.

D.3 Time for single prediction and environment

We measured the time for single decoding on the $[[2d^2 - 2d + 1, 1, d]]$ surface code with $d = 11$ and $p = 0.15$ under the depolarizing noise for the MD decoder, MWPM decoder, and the proposed neural decoders with the MLP models. Note that the times of the MD decoder and the MWPM decoder depend on the physical error probability.

We used a software IBM ILOG CPLEX via python-wrapper for constructing the MD decoder. The program was executed on Intel Xeon E5-2687W v4 with default settings. The MD decoder took about 330 milliseconds per decoding. Note that the time may be improved by optimizing the settings of CPLEX. The Kolmogorov's implementation of blossom algorithm [63, 89] was used for the MWPM decoder. The codes were compiled with Microsoft Visual C++ 2015. The program was executed on Intel Core i7-6700 without parallelization. The MWPM decoder took about 56 microseconds per decoding. The proposed neural decoders were implemented with python. We used library tensorflow to construct a neural network and perform training process. We measured the time for single prediction by setting batch size as 1, the number of hidden layers as 2, and the number of units per layer as 7000. The computation was performed using Intel Core i7-6700 and GeForce GTX 1060 6GB with CUDA 8.0. The proposed neural decoders with

the MLP model took 4 milliseconds for feed-forwarding the input data and finding the most probable class. Since the prediction of the neural decoders can be done with simple matrix multiplications, we expect that the time for single prediction of the neural decoder can be made shortened by using an optimized hardware such as FPGA, for example.

Bibliography

[1] Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.

[2] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[3] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219. ACM, 1996.

[4] Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15):150502, 2009.

[5] Richard P Feynman. Simulating physics with computers. *International journal of theoretical physics*, 21(6):467–488, 1982.

[6] Daniel S Abrams and Seth Lloyd. Simulation of many-body fermi systems on a universal quantum computer. *Physical Review Letters*, 79(13):2586, 1997.

[7] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L O'brien. A variational eigenvalue solver on a photonic quantum processor. *Nature communications*, 5, 2014.

[8] Fernando GSL Brandao and Krysta Svore. Quantum speed-ups for semidefinite programming. *arXiv preprint arXiv:1609.05537*, 2016.

[9] Seth Lloyd, Silvano Garnerone, and Paolo Zanardi. Quantum algorithms for topological and geometric analysis of data. *Nature communications*, 7:10138, 2016.

[10] Iordanis Kerenidis and Anupam Prakash. Quantum recommendation systems. *arXiv preprint arXiv:1603.08675*, 2016.

[11] Srinivasan Arunachalam and Ronald de Wolf. A survey of quantum learning theory. *arXiv preprint arXiv:1701.06806*, 2017.

[12] Charles H Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. *Theoretical computer science*, 560:7–11, 2014.

[13] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Quantum-enhanced measurements: beating the standard quantum limit. *Science*, 306(5700):1330–1336, 2004.

[14] Peter W Shor. Scheme for reducing decoherence in quantum computer memory. *Physical review A*, 52(4):R2493, 1995.

[15] A Yu Kitaev. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys*, 52(6):1191–1249, 1997.

[16] Emanuel Knill, Raymond Laflamme, and Wojciech H Zurek. Resilient quantum computation: error models and thresholds. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 454, pages 365–384. The Royal Society, 1998.

[17] Dorit Aharonov and Michael Ben-Or. Fault-tolerant quantum computation with constant error. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 176–188. ACM, 1997.

[18] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43(9):4452–4505, 2002.

[19] Emanuel Knill. Quantum computing with realistically noisy devices. *Nature*, 434(7029):39–44, 2005.

[20] Sergey B Bravyi and A Yu Kitaev. Quantum codes on a lattice with boundary. *arXiv preprint quant-ph/9811052*, 1998.

[21] Austin G Fowler, Matteo Mariantoni, John M Martinis, and Andrew N Cleland. Surface codes: Towards practical large-scale quantum computation. *Physical Review A*, 86(3):032324, 2012.

[22] J Kelly, R Barends, AG Fowler, A Megrant, E Jeffrey, TC White, D Sank, JY Mutus, B Campbell, Yu Chen, et al. State preservation by repetitive error detection in a superconducting quantum circuit. *Nature*, 519(7541):66–69, 2015.

[23] AD Córcoles, Easwar Magesan, Srikanth J Srinivasan, Andrew W Cross, M Steffen, Jay M Gambetta, and Jerry M Chow. Demonstration of a quantum error detection code using a square lattice of four superconducting qubits. *Nature communications*, 6, 2015.

[24] Diego Ristè, S Poletto, M-Z Huang, A Bruno, V Vesterinen, O-P Saira, and L DiCarlo. Detecting bit-flip errors in a logical qubit using stabilizer measurements. *Nature communications*, 6, 2015.

[25] Chao Song, Kai Xu, Wuxin Liu, Chui-ping Yang, Shi-Biao Zheng, Hui Deng, Qiwei Xie, Keqiang Huang, Qiujiang Guo, Libo Zhang, et al. 10-qubit entanglement and parallel logic operations with a superconducting circuit. *Physical review letters*, 119(18):180511, 2017.

[26] Min-Hsiu Hsieh and François Le Gall. Np-hardness of decoding quantum error-correction codes. *Physical Review A*, 83(5):052331, 2011.

[27] Sergey Bravyi, Martin Suchara, and Alexander Vargo. Efficient algorithms for maximum likelihood decoding in the surface code. *Physical Review A*, 90(3):032326, 2014.

[28] Jesse Fern, Julia Kempe, Slobodan Simic, and Shankar Sastry. Generalized performance of concatenated quantum codes–a dynamical systems approach. In *IEEE Trans. on Automatic Control*, volume 51, pages 448–459, 2006.

[29] Daniel Greenbaum and Zachary Dutton. Modeling coherent errors in quantum error correction. *arXiv preprint arXiv:1612.03908*, 2016.

[30] Chenyang Wang, Jim Harrington, and John Preskill. Confinement-higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory. *Annals of Physics*, 303(1):31–58, 2003.

[31] David S Wang, Austin G Fowler, and Lloyd C. L. Hollenberg. Surface code quantum computing with error rates over 1%. *Physical Review A*, 83(2):020302, 2011.

[32] Austin G. Fowler, Adam C. Whiteside, and Lloyd C. L. Hollenberg. Towards practical classical processing for the surface code. *Phys. Rev. Lett.*, 108:180501, May 2012.

[33] Ashley M Stephens. Fault-tolerant thresholds for quantum error correction with the surface code. *Physical Review A*, 89(2):022321, 2014.

[34] Joydip Ghosh, Austin G Fowler, and Michael R Geller. Surface code with decoherence: An analysis of three superconducting architectures. *Physical Review A*, 86(6):062318, 2012.

[35] Michael R Geller and Zhongyuan Zhou. Efficient error models for fault-tolerant architectures and the pauli twirling approximation. *Physical Review A*, 88(1):012314, 2013.

[36] Mauricio Gutiérrez, Lukas Svec, Alexander Vargo, and Kenneth R Brown. Approximation of realistic errors by clifford channels and pauli measurements. *Physical Review A*, 87(3):030302, 2013.

[37] Mauricio Gutiérrez and Kenneth R Brown. Comparison of a quantum error-correction threshold for exact and approximate errors. *Physical Review A*, 91(2):022335, 2015.

[38] Easwar Magesan, Daniel Puzzuoli, Christopher E Granade, and David G Cory. Modeling quantum noise for efficient testing of fault-tolerant circuits. *Physical Review A*, 87(1):012324, 2013.

[39] Daniel Puzzuoli, Christopher Granade, Holger Haas, Ben Criger, Easwar Magesan, and David G Cory. Tractable simulation of error correction with honest approximations to realistic fault models. *Physical Review A*, 89(2):022306, 2014.

[40] Mauricio Gutiérrez, Conor Smith, Livia Lulushi, Smitha Janardan, and Kenneth R Brown. Errors and pseudo-thresholds for incoherent and coherent noise. *Physical Review A*, 94:042338, 2016.

[41] Benjamin Rahn, Andrew C Doherty, and Hideo Mabuchi. Exact performance of concatenated quantum codes. *Physical Review A*, 66(3):032304, 2002.

[42] Christopher Chamberland, Joel J Wallman, Stefanie Beale, and Raymond Laflamme. Hard decoding algorithm for optimizing thresholds under general markovian noise. *arXiv preprint arXiv:1612.02830*, 2016.

[43] Yu Tomita and Krysta M Svore. Low-distance surface codes under realistic quantum noise. *Physical Review A*, 90(6):062320, 2014.

[44] Andrew J. Ferris and David Poulin. Tensor networks and quantum error correction. *Phys. Rev. Lett.*, 113:030501, Jul 2014.

[45] Andrew S Darmawan and David Poulin. Tensor-network simulations of the surface code under realistic noise. *arXiv preprint arXiv:1607.06460*, 2016.

[46] Daniel Gottesman. The heisenberg representation of quantum computers. *arXiv preprint quant-ph/9807006*, 1998.

[47] Leslie G Valiant. Quantum circuits that can be simulated classically in polynomial time. *SIAM Journal on Computing*, 31(4):1229–1254, 2002.

[48] Emanuel Knill. Fermionic linear optics and matchgates. *arXiv preprint quant-ph/0108033*, 2001.

[49] Barbara M Terhal and David P DiVincenzo. Classical simulation of noninteracting-fermion quantum circuits. *Physical Review A*, 65(3):032325, 2002.

[50] Sergey Bravyi. Lagrangian representation for fermionic linear optics. In *Quantum Inf. and Comp.*, volume 5, pages 216–238, 2005.

[51] Nicolas Delfosse. Decoding color codes by projection onto surface codes. *Physical Review A*, 89(1):012317, 2014.

[52] Guillaume Duclos-Cianci and David Poulin. Fast decoders for topological quantum codes. *Physical review letters*, 104(5):050504, 2010.

[53] Nicolas Delfosse and Naomi H Nickerson. Almost-linear time decoding algorithm for topological codes. *arXiv preprint arXiv:1709.06218*, 2017.

[54] Giacomo Torlai and Roger G Melko. Neural decoder for topological codes. *Physical Review Letters*, 119(3):030501, 2017.

[55] Savvas Varsamopoulos, Ben Criger, and Koen Bertels. Decoding small surface codes with feedforward neural networks. *arXiv preprint arXiv:1705.00857*, 2017.

[56] P Baireuther, TE O'Brien, B Tarasinski, and CWJ Beenakker. Machine-learning-assisted correction of correlated qubit errors in a topological code. *arXiv preprint arXiv:1705.07855*, 2017.

[57] Stefan Krastanov and Liang Jiang. Deep neural network probabilistic decoder for stabilizer codes. *arXiv preprint arXiv:1705.09334*, 2017.

[58] Nikolas P Breuckmann and Xiaotong Ni. Scalable neural network decoders for higher dimensional quantum codes. *arXiv preprint arXiv:1710.09489*, 2017.

[59] Daniel Gottesman. Stabilizer codes and quantum error correction. *arXiv preprint quant-ph/9705052*, 1997.

[60] Hector Bombin and Miguel A Martin-Delgado. Homological error correction: Classical and quantum codes. *Journal of mathematical physics*, 48(5):052105, 2007.

[61] N Cody Jones, Rodney Van Meter, Austin G Fowler, Peter L McMahon, Jungsang Kim, Thaddeus D Ladd, and Yoshihisa Yamamoto. Layered architecture for quantum computing. *Physical Review X*, 2(3):031007, 2012.

[62] Andrew J Landahl, Jonas T Anderson, and Patrick R Rice. Fault-tolerant quantum computing with color codes. *arXiv preprint arXiv:1108.5738*, 2011.

[63] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of mathematics*, 17(3):449–467, 1965.

[64] Silvio Micali and Vijay V Vazirani. An $O(\sqrt{V}E)$ algoithm for finding maximum matching in general graphs. In *Foundations of Computer Science, 1980., 21st Annual Symposium on*, pages 17–27. IEEE, 1980.

[65] Robert Raussendorf, Jim Harrington, and Kovid Goyal. Topological fault-tolerance in cluster state quantum computation. *New Journal of Physics*, 9(6):199, 2007.

[66] Sergio Boixo, Sergei V Isakov, Vadim N Smelyanskiy, Ryan Babbush, Nan Ding, Zhang Jiang, John M Martinis, and Hartmut Neven. Characterizing quantum supremacy in near-term devices. *arXiv preprint arXiv:1608.00263*, 2016.

[67] Erich Strohmaier, Jack Dongarra, Horst Simon, Martin Meuer, and Hans Meuer. The top 500 list, 2015.

[68] Daniel J Brod. Efficient classical simulation of matchgate circuits with generalized inputs and measurements. *Physical Review A*, 93:062332, 2016.

[69] Jacob T Seeley, Martin J Richard, and Peter J Love. The bravyi-kitaev transformation for quantum computation of electronic structure. *The Journal of chemical physics*, 137(22):224109, 2012.

[70] Sergey B Bravyi and Alexei Yu Kitaev. Fermionic quantum computation. *Annals of Physics*, 298(1):210–226, 2002.

[71] Easwar Magesan, Jay M Gambetta, AD Córcoles, and Jerry M Chow. Machine learning for discriminating quantum measurement trajectories and improving readout. *Physical review letters*, 114(20):200501, 2015.

[72] Giuseppe Carleo and Matthias Troyer. Solving the quantum many-body problem with artificial neural networks. *Science*, 355(6325):602–606, 2017.

[73] Juan Carrasquilla and Roger G Melko. Machine learning phases of matter. *Nature Physics*, 13(5):431–434, 2017.

[74] Jonathan Romero, Jonathan Olson, and Alan Aspuru-Guzik. Quantum autoencoders for efficient compression of quantum data. *arXiv preprint arXiv:1612.02806*, 2016.

[75] Scott Aaronson and Daniel Gottesman. Improved simulation of stabilizer circuits. *Physical Review A*, 70(5):052328, 2004.

[76] Richard Kueng, David M. Long, Andrew C. Doherty, and Steven T. Flammia. Comparing experiments to the fault-tolerance threshold. *Phys. Rev. Lett.*, 117:170502, Oct 2016.

[77] Sergey Bravyi. Classical capacity of fermionic product channels. *arXiv preprint quant-ph/0507282*, 2005.

[78] Richard Jozsa, Akimasa Miyake, and Sergii Strelchuk. Jordan-wigner formalism for arbitrary 2-input 2-output matchgates and their classical simulation. In *Quantum Inf. and Comp.*, volume 15, pages 541–556, 2015.

[79] Thomas H Cormen. *Introduction to algorithms*. MIT press, 2009.

[80] Sergey Bravyi, Matthias Englbrecht, Robert Koenig, and Nolan Peard. Correcting coherent errors with surface codes. *arXiv preprint arXiv:1710.02270*, 2017.

[81] Sergey Bravyi and David Gosset. Improved classical simulation of quantum circuits dominated by clifford gates. *Physical review letters*, 116(25):250501, 2016.

[82] Sergey Bravyi. Universal quantum computation with the $\nu= 5/ 2$ fractional quantum hall state. *Physical Review A*, 73(4):042313, 2006.

[83] Amarsanaa Davaasuren, Yasunari Suzuki, Keisuke Fujii, and Masato Koashi. General framework for constructing fast and near-optimal machine-learning-based decoder of the topological stabilizer codes. (To be submitted).

[84] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland. Randomized benchmarking of quantum gates. *Phys. Rev. A*, 77:012307, Jan 2008.

[85] Joel Wallman, Chris Granade, Robin Harper, and Steven T Flammia. Estimating the coherence of noise. *New Journal of Physics*, 17(11):113020, 2015.

[86] C Neill, P Roushan, K Kechedzhi, S Boixo, SV Isakov, V Smelyanskiy, R Barends, B Burkett, Y Chen, Z Chen, et al. A blueprint for demonstrating quantum supremacy with superconducting qubits. *arXiv preprint arXiv:1709.06678*, 2017.

[87] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[88] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[89] Vladimir Kolmogorov. Blossom v: a new implementation of a minimum cost perfect matching algorithm. *Mathematical Programming Computation*, 1(1):43–67, 2009.