Hardness of Decoding Quantum Stabilizer Codes

Pavithran Iyer and David Poulin

***Abstract*— In this paper, we address the computational hardness of optimally decoding a quantum stabilizer code. Much like classical linear codes, errors are detected by measuring certain check operators which yield an error syndrome, and the decoding problem consists of determining the most likely recovery given the syndrome. The corresponding classical problem is known to be NP-complete, and are appropriate a similar decoding problem for quantum codes is also known to be NP-complete. However, this decoding strategy is not optimal in the quantum setting as it does not consider error degeneracy, which causes distinct errors to have the same effect on the code. Here, we show that optimal decoding of stabilizer codes (previously known to be NP-hard) is in fact computationally much harder than optimal decoding of classical linear codes, it is #P-complete.**

***Index Terms*— Stabilizer codes, degenerate errors, maximum likelihood decoding, counting complexity.**

I. Introduction

IN HIS seminal papers that gave birth to the field of information theory, Shannon showed that the capacity of a channel could be achieved using codes whose codewords are random bit strings [5]. Despite this optimality, random codes have no practical use because we do not know how to decode them efficiently, i.e. in a time polynomial in the number of encoded bits. In 1978, Berlekamp *et al.* [1] (see also [2]) showed that decoding a classical linear code is an NP-Complete problem, which strongly indicates that no efficient algorithm will ever be found to decode generic classical codes. A central problem in coding theory therefore consists in designing codes that retain the essential features of random codes, but yet have enough structure to be efficiently (approximately) decoded.

Quantum information science poses additional challenges to coding theory. While the stabilizer formalism establishes many key parallels between classical and quantum coding [6], [7], important distinctions remain. On the one hand, quantum code design is obstructed by the additional burden that check operators must mutually commute. For that reason, it has proven difficult to quantize some of the best families of classical codes, such as low density parity check (LDPC) codes [8]–[11] and turbo codes [12]. On the other hand, quantum codes can be *degenerate*, which means that distinct errors can have the same effect on all codewords. This changes the nature of the decoding problem [13]–[16], and our goal here is to explore how degeneracy impacts the computational complexity of the decoding problem.

The conceptually simplest method to decode a stabilizer code is to ignore error degeneracy and to proceed as with classical linear codes: amongst all errors consistent with the error syndrome, find the one with the highest probability. We call this decoding method *Quantum Maximum Likelihood Decoding* (QMLD). It was shown in [3], [4], and [17] that QMLD is NP-Complete.

In the presence of degeneracy however, errors naturally fall into equivalence classes, with all errors in the same class having the same effect on all codewords. The optimal decoding method searches over all equivalence classes of errors that are consistent with the error syndrome, the one with the largest probability. The probability of a class of error is simply the sum of the probabilities of all errors it contains. We call this decoding method *Degenerate Quantum Maximum Likelihood Decoding* (DQMLD). Our main result is the following theorem.

Thm. (2) (Informally) DQMLD $\in$ #P-Complete *Up to Polynomial-Time Turing Reduction:* We need Turing reduction since decoding is not a counting problem, while problems in #P consist in counting the number of solutions to a decision problem in NP. Our result can be understood intuitively from the fact that in order to compute the probability associated to an equivalence class, DQMLD must determine how many errors of a given weight belong to an equivalence class, hence the need to count. Our proof uses a reduction from the problem of evaluating the *weight enumerator* of a classical (binary) linear code, which was shown by Vyalyi to be #P-Complete [18], [19].

The rest of this paper is organized as follows. For self-containment, the next two sections provide elementary introductions to computational complexity and to the stabilizer formalism. Sec. IV formally defines the decoding problem with a particular emphasis on the role of degeneracy. This section also contains an informal discussion on the importance of error degeneracy and how it impacts the decoding problem in specific settings. Sec. V presents the main result, which is proved in Sec. VI; the expert reader can jump directly to these two sections. The conclusion proposes possible extensions of the present work.

II. Computational Complexity

Two key resources required to solve any problem in computer science are space and time. One way of classifying problems is based on the runtime of a corresponding algorithm for that problem. This time is expected to depend on the size

Manuscript received November 13, 2013; revised May 18, 2014; accepted January 28, 2015. Date of publication April 28, 2015; date of current version August 14, 2015. This work was supported in part by NSERC and in part by Lockheed Martin Corporation.

The authors are with the Département de Physique, Université de Sherbrooke, Sherbrooke, QC J1K 2R1, Canada (e-mail: pavithran.iyer.sridharan@usherbrooke.ca; david.poulin@usherbrooke.ca).

Communicated by A. Ashikhmin, Associate Editor for Coding Techniques.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2015.2422294

of the input, n, to the problem. Instead of precisely denoting the runtime as a function $f(n)$, which would depend on the underlying hardware of the solver, it is largely classified by its *limiting behavior* as: $\mathcal{O}(\log_2 n), \mathcal{O}(n^k), \mathcal{O}(2^n)$ and so on. Consequently, the class of problems for which there exists an algorithm which runs in time $\mathcal{O}(n^k)$, on an input of size n, k being a constant independent of n, is called P. Note that any problem in P is a decision problem, i.e, one for which the solution space is binary. If the problem is formulated to produce an output string, then the existence of a polynomial time algorithm classifies this problem in the class FP. There are problems to which any witness or certificate for a solution can be verified in polynomial time. These fall into the class called NP and clearly, $\mathsf{P} \in \mathsf{NP}$.

A problem P_1 is *at least as hard as* P_2 if we could use the solver for P_1 to solve P_2. This is formalized by the notion of a *reduction*, which enables the classification of problems that are harder than a particular class, thereby introducing $\mathsf{NP\text{-}Hard}$ and $\mathsf{NP\text{-}Complete}$, the latter being a class of hardest problems in NP.

Besides decision problems, a category of problems involve enumerating the elements of a set. When this set is in NP, the corresponding enumeration problem is classified as $\#\mathsf{P}$ [20], [21].

Definition 1 (Counting Class): $\#\mathsf{P}$: A counting function $f : \{0,1\}^* \to \mathbb{N}$ is in $\#\mathsf{P}$ if $\exists$ a problem $P_1 \in \mathsf{NP}$ such that $f(x) = \left|\{y \in \{0,1\}^{\mathsf{poly}(|x|)} : P_1(x,y) = 1\}\right|$, $\forall x \in \{0,1\}^*$.

Hence, a function in $\#\mathsf{P}$ counts the number of solutions to a corresponding decision problem in NP. One can also compare the relative hardness of two counting problems as is done for the NP case, using the notion of counting reductions.

Definition 2 (Counting Reduction): Let f, g be counting functions. The function f is Turing reducible to g, if $\forall x \in \{0,1\}^*$, $f(x) \in \mathsf{FP}^g$.

That is, $f(x)$ can be computed in polynomial-time with polynomially many queries to an oracle for $g(x)$, for any x. Subsequently, analogous to $\mathsf{NP\text{-}Complete}$, a notion of reduction for counting functions defines the hardest problems in $\#\mathsf{P}$, as the class $\#\mathsf{P\text{-}Complete}$ [21].

Definition 3 (Counting Complete): $\#\mathsf{P\text{-}Complete}$: A counting function f is in $\#\mathsf{P\text{-}Complete}$ iff f is in $\#\mathsf{P}$ and $g \in \mathsf{FP}^f$, $\forall g \in \#\mathsf{P}$.

The last criterion can also be replaced with: $g \in \mathsf{FP}^f$, for some $g \in \#\mathsf{P\text{-}Complete}$. We will prove $\#\mathsf{P\text{-}Complete}$-ness of the problem of our concern, by adhering to such a recursive definition.

The level of hardness of a $\#\mathsf{P\text{-}Complete}$ problem can be appreciated by a consequence of a result shown by Arora and Barak [21] and Toda [22], stating that a polynomial time algorithm that is allowed a single access to a $\#\mathsf{P}$ oracle, can solve any problem in PH, i.e, $\mathsf{P}^{\#\mathsf{P}} = \mathsf{NP} \cup \mathsf{NP}^{\mathsf{NP}} \cup \mathsf{NP}^{\mathsf{NP}^{\mathsf{NP}}} \ldots$.

A particular example of a counting function that will be of interest in the coming sections is the *weight enumerator* function for a linear code.

Definition 4 (Weight Enumerator Problem): WE: Given a (n,k) linear code $\mathcal{C}$ and a positive integer i, compute $\mathsf{WE}_i(\mathcal{C}) = |\{c \in \mathcal{C} : |c| = i\}|$, the number of codewords of Hamming-weight $|c| = i$.

The corresponding decision problem, which is to determine if there exists a codeword in $\mathcal{C}$ of weight i, is known to be in $\mathsf{NP\text{-}Complete}$ [1], [23]. This immediately implies that $\mathsf{WE}_i(\mathcal{C}) \in \#\mathsf{P}$, and furthermore, it is known to be complete for this class [18], [19].

Theorem 1: Hardness of WE: For a linear code $\mathcal{C}$ and $i = \Omega(\mathsf{polylog}(n))$, $\mathsf{WE}_i(\mathcal{C}) \in \#\mathsf{P\text{-}Complete}$.

Hence to show that a problem of computing f is in $\#\mathsf{P\text{-}Complete}$, in addition to its membership in $\#\mathsf{P}$, it suffices to show that for any (n,k) linear code $\mathcal{C}$, $\mathsf{WE}_i(\mathcal{C})$, $i \in o(\mathsf{polylog}(n))$ can be computed in polynomial-time by allowing at most polynomially many queries to an oracle for computing f.

III. Stabilizer Codes

In this section, we will provide the necessary background material on stabilizer codes, see [6], [24] for more complete introductions. In the setting of quantum coding, information is encoded into n-qubit states in $\mathcal{H}_2^n = (\mathbb{C}^2)^{\otimes n}$. Errors are operators acting on this space, modeled as elements of the *Pauli group* $\mathcal{G}_n$.

Definition 5 (Pauli Group): The Pauli group on 1-qubit, denoted by $\mathcal{G}_1$ is a matrix group over $\mathbb{C}^2$, generated by the Pauli matrices; $\mathcal{G}_1 = \{\pm X, \pm Y, \pm Z, \pm iX, \pm iY, \pm iZ, \pm \mathbb{1}\} = \langle i, X, Y, Z\rangle$. The Pauli group on n-qubits is the group generated by the set of Pauli matrices acting on n-qubits, denoted by $\mathcal{G}_n = \mathcal{G}_1^{\otimes n} = \{\epsilon P_1 \otimes \cdots \otimes P_n | \epsilon \in \{\pm 1, \pm i\}, P_i \in \{\mathbb{1}, X, Y, Z\}\}$. Though the definition of the Pauli group in Def. 5 contains the scalars $\{\pm 1, \pm i\}$, they are often considered unimportant for error detection or correction as they do not affect the error syndrome nor the error correction. This enables us to define the effective Pauli group by identifying operators that are related by a multiplicative constant in $\{\pm 1, \pm i\}$, denoted by $\overline{\mathcal{G}}_n = \mathcal{G}_n / \{\pm 1, \pm i\}$. The number of qubits affected by an error $E \in \overline{\mathcal{G}}_n$ is the number of non-identity components in the tensor product form of E, and is called the *weight* of the error, denoted by $\mathsf{wt}(E)$.

A quantum stabilizer code is defined by a set of check operators, which are also elements of the Pauli group.

Definition 6 (Stabilizer Codes): A stabilizer code is a subspace of the n-qubit vector space $\mathcal{H}_2^n$, described as the common $+1$ eigenspace of an Abelian and Hermitian subgroup of $\mathcal{G}_n$, called the stabilizer subgroup. Hence, for a stabilizer code $\mathcal{Q}$, the stabilizers are defined by:

$$\mathcal{S} = \{S \in \overline{\mathcal{G}}^n : S|\psi\rangle = |\psi\rangle, \quad \forall |\psi\rangle \in \mathcal{Q}\}. \tag{1}$$

Equivalently, we can define the code $\mathcal{Q}$ in terms of its stabilizers

$$\mathcal{Q} = \{|\psi\rangle \in \mathcal{H}_2^n : S|\psi\rangle = |\psi\rangle, \quad \forall S \in \mathcal{S}\}. \tag{2}$$

Stabilizer codes are by far the most widely studied codes in the quantum setting.

A. Operator Basis

When $\mathcal{S}$ is generated by $n-k$ independent *stabilizer generators* $\{S_j\}_{j=1\ldots n-k}$, the subspace $\mathcal{Q}$ has dimension 2^k,

so it encodes k logical qubits, and we denote its parameters by $[[n, k]]$. All the elements in $\overline{\mathcal{G}}_n$ that leave the code $\mathcal{Q}$ globally invariant belong to the *normalizer* of $\mathcal{S}$ in $\overline{\mathcal{G}}$, denoted by $\mathcal{N}(\mathcal{S})$. The normalizer $\mathcal{N}(\mathcal{S})$ forms a group generated by $n+k$ generators and $\mathcal{S} \subset \mathcal{N}(\mathcal{S})$. However, not all operators in $\mathcal{N}(\mathcal{S})$ necessarily fix every state in $\mathcal{Q}$. Since the action of $\mathcal{S}$ is trivial on all code states, we define a subset of operators in $\mathcal{N}(\mathcal{S})$ called the *logical operators* that represent the quotient space $\mathcal{L} = \mathcal{N}(\mathcal{S})/\mathcal{S}$, each of which have a distinct action on individual states in $\mathcal{Q}$. For a shorthand notation, we will identify these representative with elements of $\mathcal{L}$. This group has $2k$ *canonical logical generators* $\{\overline{X}_j, \overline{Z}_j\}_{j=1\ldots k}$, such that all generators mutually commute except the pairs $\overline{X}_j$ and $\overline{Z}_j$ that anti-commute. The smallest weight of a nontrivial logical operation is the distance of the code,

$$d = \min_{E \in \mathcal{N}(\mathcal{S}) \setminus \mathcal{S}} \mathsf{wt}(E). \tag{3}$$

Note that the operators defined so far, $\{S_j\}_{j=1\ldots n-k}$ and $\{\overline{X}_j, \overline{Z}_j\}_{j=1\ldots k}$ do not generate $\overline{\mathcal{G}}_n$. To complete the basis, we need to define the group of *pure errors* $\mathcal{T} = \mathcal{N}(\mathcal{L})/\mathcal{S}$, which is also Abelian. We can always find a set of *canonical pure error generators* $\{T_j\}_{j=1\ldots n-k}$ such that T_j commutes with all other pure error generators, all logical generators, and all stabilizer generators except S_j with which it anti-commutes. To summarize, we have the canonical basis of the Pauli group $\{S_i, T_i, \overline{X}_j, \overline{Z}_j\}_{i=1\ldots n-k, j=1\ldots k}$ with all commutation relations trivial, except

$$T_i S_i = -S_i T_i \quad \text{and} \quad \overline{X}_j \overline{Z}_j = -\overline{Z}_j \overline{X}_j. \tag{4}$$

Any Pauli operator $E \in \bar{\mathcal{G}}_n$ can be expressed as a product of elements in these respective groups:

$$E = T \cdot L \cdot S \qquad (\text{where } T \in \mathcal{T},\ L \in \mathcal{L} \text{ and } S \in \mathcal{S}). \tag{5}$$

Decomposition in this basis will be particularly useful to formulate the decoding problem.

B. Degenerate and Non-Degenerate Errors

Two errors E and E' are called *degenerate* if they have an identical effect on all code states, i.e. $E|\psi\rangle = E'|\psi\rangle$, $\forall |\psi\rangle \in \mathcal{Q}$. Given the decomposition in (5) and the definition of the code in (1), we see that this is only possible if the two errors are related by an element of $\mathcal{S}$, i.e., $E' = E \cdot S$ for some $S \in \mathcal{S}$. This naturally leads to an equivalence relation between errors, with two errors belonging to the same equivalence class if they are related by an element of $\mathcal{S}$. In other words, the set of equivalence classes is the quotient space $\overline{\mathcal{G}}_n/\mathcal{S} \sim \mathcal{L} \cdot \mathcal{T}$. We can thus label the equivalence classes by L, T with $L \in \mathcal{L}$ and $T \in \mathcal{T}$. For a fixed T, the different classes labelled by $L \in \mathcal{L}$ are referred to as *logical classes* and each class is of size 2^{n-k}. The class labelled by $L = \mathbb{1}$ and $T = \mathbb{1}$ is $\mathcal{S}$ itself. Any other logical class can be expressed as a coset of $\mathcal{S}$, but however they are not groups by themselves.

Note that degeneracy is unique to the quantum error correction setting. When a bit flip pattern e is applied to a classical bit string x, the resulting string $y = x + e$ always differs from the original one, except for the trivial error $e = 0^n$. Consequently, each logical class contains only one element. Our main result in this paper shows that accounting for degenerate errors in decoding stabilizer codes greatly increases its computational complexity.

C. Symplectic Representation

There is a one-to-one correspondence between *classical symplectic linear codes* in $\mathbb{Z}_2^{2n}$ and stabilizer codes [6], [7], which follows from a mapping η of pauli operators in $\overline{\mathcal{G}}_n$ into binary strings of length $2n$. The mapping is performed by first expressing every $\mathsf{M} \in \overline{\mathcal{G}}_n$ in the form:

$$\mathsf{M} = (\mathsf{A}_1 \otimes \cdots \otimes \mathsf{A}_n) \cdot (\mathsf{B}_1 \otimes \cdots \otimes \mathsf{B}_n), \quad \mathsf{A}_i \in \{Z, I\}, \quad \mathsf{B}_j \in \{X, I\}, \quad 1 \leq i, j \leq n \tag{6}$$

and substituting X and Z by '1' and $\mathbb{1}$ by '0'. For instance, $\eta(X \otimes Z \otimes I \otimes Y) = 01011001$. This maps $\overline{\mathcal{G}}^n$ into its *symplectic representation* in $\mathbb{Z}_2^{2n}$. Moreover, any two mutually commuting operators $P, Q \in \mathcal{G}_n$ are mapped into binary strings $x, y \in \mathbb{Z}_2^{2n}$ that are orthogonal under the *symplectic product*, defined by $(x, y) = x \Lambda y^{\mathrm{T}}$ with $\Lambda = I_n \otimes X$. This immediately implies that $\eta(\mathcal{N}(\mathcal{S}))$ is a vector space (code) which is the kernel of $\eta(\mathcal{S})$ under the symplectic product. The parity check matrix for this classical code consists of row vectors, each of which are given by the symplectic encoding of a unique stabilizer generator, i.e, the rows of the parity check matrix are just $\eta(\{S_i\}_{i=1}^{n-k})$. The corresponding generator matrix is now a $k \times 2n$ matrix each of whose rows are given by the symplectic encoding of a unique logical generator, i.e, the rows of the generator matrix are $\eta(\{\overline{X}_j, \overline{Z}_j\}_{j=1}^{k}, \{S_i\}_{i=1}^{n-k})$. Strictly speaking, the ordering of rows within the parity check matrix or within the generator matrix does not change the stabilizer code and hence is irrelevant for error correction purposes. For a $2n$-bit string $b = (z|x)$ expressed as the concatenation of two n-bit stings, we denote the inverse mapping $\eta^{-1}(b) = Z^z \cdot X^x$ where we use the shorthand notation $Q^x = Q^{x_1} \otimes Q^{x_2} \otimes \cdots \otimes Q^{x_n}$.

The mapping η implicitly indicates that we consider a Y-type Pauli operation as a Z-type operation followed by a X-type operation, in other words, as an error of weight two. Consequently, we define the *symplectic-weight* of the Pauli error as $\mathsf{wt}_2(E) = |\eta(E)|$ with $|\cdot|$ denoting the usual Hamming weight. Clearly, $2\,\mathsf{wt}(E) \geq \mathsf{wt}_2(E) \geq \mathsf{wt}(E)$.

This correspondence with classical linear codes is the key to most of the complexity results [3], [4], [17], [18] involving stabilizer codes, including ours. In many cases it can be used to build parallels to already known results for classical (linear) symplectic codes.

D. Error Model

An error model assigns probabilities to various errors, which are then used by the decoder to statistically infer what recovery is most likely given the error syndrome. We restrict ourselves to errors from $\overline{\mathcal{G}}_n$, so the corresponding error model is referred to as a *Pauli channel*. There are multiple type of Pauli channels which are often used in studying error correcting codes [24], [25]. Here, we will further assume that errors act independently on each qubit.

Definition 7 (Memoryless Pauli Channel): On a memoryless Pauli channel, the probability of error $E = \bigotimes_{i=1}^{n} E_i \in \bar{\mathcal{G}}_n$,

with $E_i \in \bar{\mathcal{G}}_1$, is $\mathsf{Pr}(E) = \prod_{i=1}^{n} q_{i,E_i}$ where the q_i are properly normalized probability vectors on $\{I, X, Y, Z\}$.

In general, the probabilities q_{i,E_i} can be different for all qubits. One important feature of memoryless Pauli channels is that they can be efficiently specified (to finite accuracy), i.e. using $\mathcal{O}(n)$ bits of information. An obvious simplification of the above channel is made by supposing that the noise rates are the same for all X, Y, Z type errors on each qubit. This specifies to a *depolarizing channel*, and can be expressed as

$$\mathsf{Pr}(E) = \left(\frac{p}{3}\right)^{\mathsf{wt}(E)} \times (1-p)^{n-\mathsf{wt}(E)}. \tag{7}$$

Alternatively, one can assume that each qubit is first affected by X-type errors, and a then by Z-type errors, and moreover, these errors are independent of each other. A Y-type error occurs only when both a Z-type as well as an X-type error affect a qubit. This specifies a *independent X-Z channel*, which we will use to prove our main result.

Definition 8 (Independent X-Z Channel): An independent X-Z channel is a memoryless Pauli channel defined by $q_X = q_Z = p/2\,(1 - p/2)$, $q_Y = p^2/4$, and $q_I = (1 - p/2)^2$ for each qubit. The probability of an error $E \in \bar{\mathcal{G}}_n$ can thus be written as

$$\mathsf{Pr}(E) = \left(\frac{p}{2}\right)^{\mathsf{wt}_2(E)} \times \left(1 - \frac{p}{2}\right)^{2n-\mathsf{wt}_2(E)} \tag{8}$$

where $\mathsf{wt}_2(E)$ is the symplectic weight of E, see Sec. III-C.

One key feature of both the depolarizing channel and the independent X-Z channel is that the probability of an error depends only on its weight (either wt or wt_2). As a consequence, evaluating the probability of a logical class can be done by counting the number of its elements of a given weight, which puts the problem in #P. Notice also that $\mathsf{Pr}(E)$ is monotonically decreasing with its weight for $p \in [0, 1/2]$, implying that in such a range, a minimum weight error has the maximum probability. We will often refer to p to as the error-rate per qubit or the physical noise-rate.

IV. The Decoding Problem

In this section we define the decoding problem more formally, and explain how it is affected by the existence of degenerate errors as defined in Sec. III-B.

The qubits are prepared in a code state $|\psi\rangle \in \mathcal{Q}$ and are subject to the memoryless Pauli channel (Def. 7). The received state is $|\phi\rangle = E|\psi\rangle$ where $E \in \overline{\mathcal{G}}_n$ is an unknown error chosen from the distribution $\mathsf{Pr}(E)$. *Decoding* refers to the operation performed by the receiver in recovering the state $|\psi\rangle$ from the state $|\phi\rangle = E|\psi\rangle$. Since all Pauli operators square to the identity, it suffices for the receiver to determine E and apply it to the system to recover the original state, i.e. $E|\phi\rangle = E^2|\psi\rangle = |\psi\rangle$.

The error, being an element of the Pauli group, can either commute or anti-commute with each of the stabilizer generators. Thus, upon measurement of each stabilizer generator, we obtain an eigenvalue $+1(-1)$ indicating that E commutes (anti-commutes) with the stabilizer generator:

$$S_j|\phi\rangle = S_j \cdot E|\psi\rangle = \begin{cases} +|\phi\rangle, & \text{if } [E, S_j] = 0 \quad (9a) \\ -|\phi\rangle, & \text{if } \{E, S_j\} = 0 \quad (9b) \end{cases}$$

Each ± 1 measurement outcome m_j is encoded into a bit s_j such that $m_j = (-1)^{s_j}$. The outcomes of measuring all the check operators is encoded as a $(n-k)$ bit vector s called the *error syndrome*.

A. Non-Degenerate Decoding

The decoding problem consists in identifying E conditioned on knowledge of the error syndrome. As in the classical case [1], [23], [26], the conceptually simplest strategy is to choose E that has the highest probability amongst all errors consistent with the measured syndrome. This is called *Quantum Maximum Likelihood Decoding* (QMLD) [3], [4], [17], and can be formulated mathematically as:

$$E_{\mathsf{QMLD}}(s) = \mathsf{ArgMax}_{E \in \bar{\mathcal{G}}_n} \mathsf{Pr}(E|s) \tag{10}$$

Using the decomposition (5), we can view the error probability $\mathsf{Pr}(E)$ as a joint probability over the group $\mathcal{T}$, $\mathcal{L}$, and $\mathcal{S}$ in a natural way:

$$\begin{aligned} &\mathsf{Pr}(T, L, S) = \mathsf{Pr}(E = T \cdot L \cdot S), \\ \text{with} \quad &T \in \mathcal{T}, \quad L \in \mathcal{L}, \quad \text{and S} \in \mathcal{S}. \end{aligned} \tag{11}$$

Using the commutation relations given at (4), it follows that the knowledge of s is equivalent to knowledge of T, since T_j is the only element of this basis that anti-commutes with S_j. Thus, we have $T_s = \prod_j T_j^{s_j}$, and all errors in $T_s \cdot \mathcal{N}(\mathcal{S})$ are consistent with the syndrome s. Hence, at this stage a *best guess* for the elements in $\mathcal{S}$ and $\mathcal{L}$ needs to be employed. This involves finding $L \cdot S$ that *maximizes the likelihood* of $E = T_s \cdot L \cdot S$, implying an equivalent definition of QMLD:

$$E_{\mathsf{QMLD}}(s) = T_s \cdot \mathsf{ArgMax}_{L \in \mathcal{L}, S \in \mathcal{S}} \mathsf{Pr}(L, S|T_s) \tag{12}$$

where the conditional probability is given by Bayes' rule $\mathsf{Pr}(L, S|T_s) = \mathsf{Pr}(L, S, T_s)/\mathsf{Pr}(T_s)$ with the marginal defined as usual $\mathsf{Pr}(T_s) = \sum_{L,S} \mathsf{Pr}(L, S, T_s)$.

Informally speaking, QMLD addresses the problem of determining the element of $\mathcal{L} \cdot \mathcal{S}$, whose probability is maximum, given an error rate and a syndrome. For the special cases of the depolarizing channel (7) and the independent X-Z channel Def. 8, the search for an operator with maximum probability is synonymous to the search for an operator of minimum weight (with two possible notions of weight, one over the $\overline{\mathcal{G}}_n$ and the other over $\mathbb{Z}_2^{2n}$). Consequently, QMLD is also known as a *minimum-weight decoder*.

One subtlety arises in case where the maximum probability is a close tie, as we cannot expect a decoder to discriminate probabilities within an arbitrary accuracy. Thus, we can define QMLD as the problem of identifying the optimal couple L, S, but we tolerate that it fails when more than one choice have probabilities that are within a small distance Δ from the optimal. The standard way of formalizing this notion is with a *promise gap*, where we just assume in the definition of the problem that there is no close tie.

Definition 9 [(Quantum Maximum Likelihood Decoding) QMLD]:

Input: An n-qubit quantum code with stabilizer group $\mathcal{S}$ specified by $n - k$ independent generators, a memoryless

Pauli channel $\mathsf{Pr}(E)$, an error syndrome $s \in \mathbb{Z}_2^{n-k}$, and a promise gap Δ.

Promise: There exists a couple $S^\star \in \mathcal{S}$ and $L^\star \in \mathcal{N}(\mathcal{S})/\mathcal{S}$ such that

$$\mathsf{Pr}(L^\star, S^\star|T_s) - \mathsf{Pr}(L, S|T_s) \geq \Delta\, \mathsf{Pr}(L^\star, S^\star|T_s), \quad \forall (L, S) \neq (L^\star, S^\star). \tag{13}$$

Output: $L^\star S^\star$.

As mentioned above, for the depolarizing channel and the independent X-Z channel QMLD is formally equivalent to a minimum-weight decoder, in which case the promise gap is irrelevant and can be set to 0.

B. Degenerate Decoding

We will now explain how degeneracy changes the decoding strategy. As explained in Sec. III-B, errors can be classified into equivalence classes labelled by L, T, with all errors within a class having the same effect on the code and therefore all being correctable by the same operation. As a consequence, we see that QMLD is a suboptimal decoding strategy—in the sense that it does not reach the maximum probability of correctly decoding—because it fails to recognize the equivalence between degenerate errors. Instead of searching the most likely error, the optimal decoder seeks for the most likely equivalence class of errors, with the probability of a class of errors equal to the sum of the probability of the errors it contains. Since all errors in an equivalence class are related by an element of $\mathcal{S}$ and their $\mathcal{T}$ component is fixed by the syndrome, we can write the probability of a class conditioned on syndrome s as

$$\mathsf{Pr}(L|s) = \sum_{S \in \mathcal{S}} \mathsf{Pr}(L, S|T_s), \tag{14}$$

where we use standard Bayesian calculus as above. The *Degenerate Maximum Likelihood Decoding* (DQMLD) problem can be formulated mathematically as determining an error in the most probable logical class, for a given syndrome, (14).

Definition 10: [(Degenerate Maximum Likelihood Decoding) DQMLD]:

Input: An n-qubit quantum code with stabilizer group $\mathcal{S}$ specified by $n-k$ independent generators, a memoryless Pauli channel $\mathsf{Pr}(E)$, an error syndrome $s \in \mathbb{Z}_2^{n-k}$, and a promise gap Δ.

Promise: There exists an $L^\star \in \mathcal{N}(\mathcal{S})/\mathcal{S}$ such that

$$\mathsf{Pr}(L^*|T_s) - \mathsf{Pr}(L|T_s) \geq \Delta\, \mathsf{Pr}(L^*|T_s), \quad \forall L \neq L^*. \tag{15}$$

Output: L^*.

Note that the promise gap can be expressed as a relative gap or an additive gap, the former being the right one in our setting. This is because the relative promise gap can be related to the failure probability of the code under optimal decoding. Consider a large promise gap $\Delta = 1 - 4^{-k}\epsilon$. Rewriting the promise as $P(L|T_s) \leq 4^{-k}\epsilon P(L^*|T_s)$ and summing over all $L \neq L^*$ (of which there are $4^k - 1$), we arrive at $P(L^*|T_s) \geq 1-2\epsilon$, which simply says that the probability that the error that occurred is not equivalent to L^*—and hence that the decoder fails—is at most 2ϵ.

Note also that for a fixed T, the probabilities $\mathsf{Pr}(L, S|T)$ and $\mathsf{Pr}(L, S, T)$ differ only by a constant, so we can perform the optimization in Def. 10 on the joint probability instead of the conditional probability. The sum appearing in the this probability (14), being over 2^{n-k} terms, forbids a polynomial-time direct computation of its value. However, for i.i.d. Pauli channels such as the depolarizing channel and the independent X-Z channel, by grouping terms of equal weight in the sum, we can express the sum in (14) more succinctly. For the case of the independent X-Z channel Def. 8, the above joint probability can be expressed as

$$\mathsf{Pr}(T_s, L, S) = \left(1 - \frac{p}{2}\right)^{2n} \sum_{S \in \mathcal{S}} \tilde{p}^{\mathsf{wt}_2(T_s \cdot L \cdot S)} \tag{16}$$

where $\tilde{p} = p/(2-p)$. By grouping terms of equal weight in the sum, we arrive at a sum involving only $n+1$ terms

$$\mathsf{Pr}(T_s, L, S) = \left(1 - \frac{p}{2}\right)^{2n} \sum_{i=0}^{n} A_i(s, L)\tilde{p}^i \tag{17}$$

with

$$A_i(s, L) = |\{S \in \mathcal{S} : \mathsf{wt}_2(E = T_s \cdot L \cdot S) = i\}| \tag{18}$$

and

$$\sum_{i=0}^{n} A_i(s, L) = 2^{n-k}. \tag{19}$$

The coefficients $\{A_i(s, L)\}_{i=0}^n$ are called the *weight enumerators of the coset* associated to s and L. Note that the coset weight enumerators play a very important role in estimating the decoder performances of both QMLD as well as DQMLD.

The sum in (17) is now over polynomially many terms unlike its previous form (16). Computing such a sum for each logical operator and subsequently optimizing over their values would solve DQMLD. An $[[n, k]]$ stabilizer code has $|\mathcal{L}| = 4^k$, implying that even if the weight enumerators can be computed efficiently, a polynomial-time optimization cannot be performed over the different cosets labeled by L, unless $k \in \mathcal{O}(\log(n))$, which is the regime that we are interested in. Furthermore, we believe DQMLD is at least as hard otherwise, i.e for $k \in \Omega(\log(n))$.

At this stage, we like to remark that though QMLD and DQMLD are stated as decision problems in [3] and [4], the problem of practical interest is one of determining a Pauli operator that maximizes the respective probabilities in Defs. 9 and 10. The standard method to formulate a decision problem that is computationally equivalent to function problem (a problem which has an output that is not necessarily one of 0 or 1) is by introducing an extra parameter. To be more precise let us consider the case of QMLD on a $[[n, k]]$ stabilizer code, in which case t could be the bounded distance of the code, an integer between 0 and $n/2$. Now, a decision problem which takes the same input as QMLD according to Def. 9 as well as the integer t will answer whether or not there is any error of weight less that t, that is consistent with a given syndrome. This is exactly what the authors in [3] and [4] have adopted. Likewise, a decision version of DQMLD will only answer whether or not there is any logical operator $L \in \mathcal{L}$

whose probability according to (16) is at least c, where c here is a real number between 0 and 1.

In both of the above cases, it may seem that the decision version of the decoding problems is computationally easier than the real decoding problems themselves. However, this is not true. This decision version of the decoders can be used to reveal more information about the underlying stabilizer code than what is really expected from of DQMLD or QMLD in practice. In particular, by varying this constant c in DQMLD, along with the input in Def. 10, one could use the oracle to not only learn the optimal correction but also the probability of its equivalence class. The latter is not necessary to perform error correction and thus gives more power to the decoder oracle than it should. This is why we formulated DQMLD as a function problem that does not explicitly reveal the probability of the equivalence class of the optimal correction, and therefore we consider it to be closer to the real world decoding problem.

C. Importance of Degeneracy

Before addressing the computational complexity of degenerate decoding, we close this section with a discussion of its practical relevance. The two decoders QMLD and DQMLD will provide different answers whenever the most likely equivalence class does not contain the error with the largest probability. Consider the hypothetical scenario where the class of error L_1 contains a single error of low weight a and $2^{n-k}-1$ errors of high weight $b \gg a$, and that the class L_2 contains 2^{n-k} errors of intermediate weight c, $a \ll c \ll b$. The QMLD would choose the error from the class L_1, because it is the most likely error. On the other hand, the probabilities of these two classes are given by

$$\mathsf{Pr}(L_1) \propto \tilde{p}^a + (2^{n-k}-1)\tilde{p}^b \approx \tilde{p}^a \tag{20}$$

and

$$P(L_2) \propto 2^{n-k}\tilde{p}^c. \tag{21}$$

Thus, we see that DQMLD would provide a different answer if $P(L_1) < P(L_2)$, or equivalently $\tilde{p} > 2^{-\frac{n-k}{c-a}}$, and that the two decoders would agree otherwise. Thus, for sufficiently low error rates, and in particular in the extreme limit $p \leq 2^{k-n}$, degeneracy does not affect the decoding problem.

More importantly, degeneracy becomes unimportant when the *failure rate* of the code is very low, which doesn't necessarily require a low physical noise rate. Remember Sec. IV-B that the failure rate 2ϵ of the code is related to the DQMLD promise gap $\Delta = 1 - 4^{-k}\epsilon$. The following lemma, whose proof is presented in App. B), shows that for $\epsilon = 2^{k-n}$, QMLD provides the same answer as DQMLD, which in turn implies that DQMLD is in NP with such a large gap.

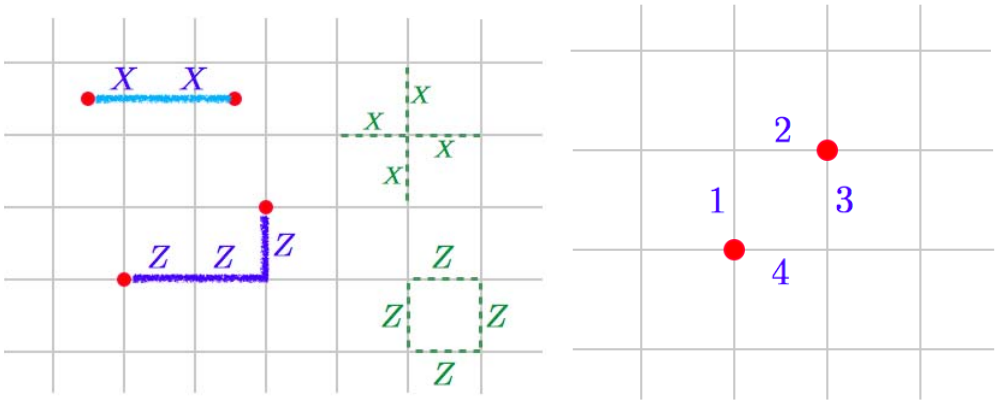
Lemma 1: With a promise gap $\Delta = 1 - 2^{-n-k}$ and on an independent X-Z channel, DQMLD and QMLD are equivalent.

In the light of these observations, one might imagine that in general, DQMLD can at most offer a marginal improvement over QMLD, i.e. that for a given code and noise rate, the decoding error probability of QMLD is upper bounded by a function of the decoding error probability of DQMLD. There is strong evidence that this is not the case however. Monte Carlo simulations [13], [27], [28] have shown that QMLD and DQMLD achieve different error thresholds with Kitaev's topological code. (This statement is equivalent to the fact that the critical disorder strength separating the ordered from the disordered phase in the random bound Ising model decreases below the Nishimori temperature.) Thus, for noise rates p falling between these two thresholds, the failure probability of DQMLD tends to 0 as the number of qubit n increases, while the failure probability of QMLD tends to $1 - 4^{-k}$ (the failure probability of a random guess), so the performances of both decoders can be significantly different.

Degeneracy can also severely impact the performances of certain decoding algorithms. Due to degeneracy of quantum errors, the ability to correct errors does not imply that the a posteriori marginal error probability over individual qubits is sharply peaked, in contrast to the classical setting. This is the case in particular for low density parity check (LDPC) codes. These codes have the property of admitting a set of stabilizer generators S_j of weight bounded by a constant. As a consequence, the weight of equivalent errors E and E' related by a stabilizer generator $E' = ES_j$ will differ at most by a small constant. Thus, we expect degeneracy to play an important role: each equivalence class contains many errors of roughly the same weights. As discussed in [29], conventional decoding algorithms for LDPC codes (belief propagation [30]) are *marginal decoders* in the sense that they optimize the probability of error independently for each qubit. But in the presence of degeneracy, we can have a probability sharply peaked over a single equivalence class of errors—ensuring the success of DQMLD—but yet have a very broad marginal distribution over individual qubits—leading to a failure of a marginal decoder. So the existence of a good, general purpose decoder for quantum LDPC codes, playing a role analogous to belief propagation in the classical setting, remains an outstanding open question.

This situation is best illustrated with Kitaev's topological code [31]. In this code, errors correspond to strings on a regular square lattice, and error syndromes are located at the endpoints of the strings. The weight of an error is equal to the length of the corresponding string. Lastly, the different equivalence classes of errors correspond to the homology classes of the lattice. For a syndrome configuration shown at Fig. 1, all the short paths have the same homology, so the probability is sharply peaked over one equivalence class. But there are several distinct strings of the same length compatible with this syndrome, so the marginal error probability over individual qubits is very broad.

Lastly, we note that to achieve the true capacity of certain quantum channels (as opposed to the single-shot capacity), it is necessary to encode the information in a degenerate code [32]. In other words, there are certain channels that could not be used to send any quantum information if a non-degenerate code was used, but can reliably do so at a finite rate with degenerate codes. We do not know however if degeneracy also needs to be taken into account during the decoding process to realize this. In particular, the example in [32] uses a generalization of Shor's code [33], for which



(a) (b)

Fig. 1. The toric code has two type of stabilizer generators, shown here in green Fig. 1a, centered around each face and each vertex of the lattice. Strings of Z errors on the lattice generate vertex syndromes at their endpoints, while strings of X errors on the dual lattice generate face syndromes at their endpoints. Strings with same holonomy represent equivalent errors. In Fig. 1b the qubits are labelled $1-4$ for easy reference. There are many errors consistent with this syndrome, the lowest weight of which are Z_1Z_2 and Z_3Z_4, and by symmetry $\mathsf{Pr}(Z_1Z_2) = \mathsf{Pr}(Z_3Z_4) = \max_{E\in\overline{\mathcal{G}}_n} \mathsf{Pr}(E|s)$. Applying either of these operators serves as a valid correction since they have the same holonomy. Errors with a different holonomy have a probability that decreases exponentially with the lattice size, so DQMLD would successfully decode. But marginally, qubits 1, 2, 3, and 4 all have identical probabilities, so a marginal decoder would either correct with $Z_1Z_2Z_3Z_4$ or the identity, none of which are acceptable corrections.

QMLD and DQMLD always yield the same output. We know of only a few examples of codes for which DQMLD can be computed efficiently, namely concatenated codes [14], convolutional codes [16], and Bacon-Shor codes [34]. There exist heuristic methods to take degeneracy into account in topological codes [15] and turbo codes [12].

V. Complexity of the Decoding Problem

The one-to-one correspondence between $[n,k]$ stabilizer codes and $(2n,k)$ symplectic linear codes [6] is used in [3] and [4] to show that a solution for QMLD can be used to decide an NP-Complete problem in polynomial time. Consequently, QMLD $\in$ NP-Complete.

DQMLD was shown to be NP-Hard for the case of an independent X-Z channel in [3] and depolarizing channel in [4], using a reduction from a NP-Complete problem pertaining to classical linear code. We now state our main result, which establishes that DQMLD is in fact much harder than what there previous results anticipated.

Theorem 2: DQMLD (c.f. Def. 10) of an $[[n,k=1]]$ stabilizer code on an independent X-Z channel and with a promise gap $\Delta \leq 2[2+n^{\lambda}]^{-1}$, with $\lambda = \Omega(\mathsf{polylog}(n))$, is in #P-Complete.

In the following sections, we will show that for a classical binary linear code $\mathcal{C}$ and $\lambda \in [0,n]$, the problem of computing $\mathsf{WE}_i(\mathcal{C})$ for $i = 0,1,2,\ldots,\lambda$ is polynomial time Turing reducible to DQMLD on an independent X-Z channel, for $\Delta \leq 2[2+n^{\lambda}]^{-1}$.

In the case of a general memoryless Pauli channel, it is not always possible to express the probability of a logical class as a weight enumerator sum (17) with polynomially many terms in the sum. Hence, in such cases, the containment of DQMLD in #P is not known and we can only claim that it is #P-Hard. However, whenever one can express $\mathsf{Pr}(L|s)$ as a sum with polynomially many terms, each of which is a #P function, then DQMLD can be put into #P.

For the independent X-Z channel, the containment in #P is straightforward.

Lemma 2: DQMLD in Def. 10 on a $[[n,k]]$ stabilizer code with $k = \mathcal{O}(\log_2 n)$ on an independent X-Z channel (8) or a depolarizing channel, is in #P.

Proof: From the definition of DQMLD, one can subdivide it into two problems. The first step consists of computing the probabilities of all the logical classes, corresponding to the given syndrome, followed next by an optimization over the probabilities, thereby choosing an error from the logical class with the largest probability. Since $k = \mathcal{O}(\log_2 n)$, there are at most polynomially many logical classes and hence a naive search for the maximum amongst the set of probabilities of all the logical classes can be achieved in polynomial-time.

It now remains to show that for each logical class, its probability (19) for a general noise rate p, is a #P function. This immediately follows from the polynomial form in p, taken by the logical class probabilities, (17) and the observation that the the coefficients of the polynomial are weight enumerator coefficients of a suitable linear code Sec. III-C.

The promise gap Δ is irrelevant to the containment of DQMLD in #P unless $\lambda = \Omega(2^{-\mathsf{poly}(n)})$. A gap of $2^{-\mathsf{poly}(n)}$ is essential since the probabilities of various logical classes are represented as bit-strings for performing arithmetic and they must differ in the first polynomially many bits. ■

VI. Reduction

In this section, we present a polynomial time algorithm that accepts as input a classical linear code $\mathcal{C}$ and outputs $\{\mathsf{WE}_i(\mathcal{C})\}_{i=0}^{n}$ as in Def. 4 by querying a DQMLD oracle with an independent X-Z noise model and a promise gap Δ which is $1/\mathsf{quasi\text{-}polynomial}(n)$.

A. Reduction Overview

The correspondence between symplectic linear codes and stabilizer codes discussed in Sec. III-C implies that with an

independent X-Z channel, the probability of a logical class is related to the weight enumerator polynomial of a classical linear code. In particular, with the trivial syndrome $s = 0$ and at very low noise rate $p \ll 1/2$, the most likely equivalence class is always the trivial one $L = \mathbb{1}$. Hence, in this setting, the probability of the trivial logical class for a quantum code with stabilizers $\mathcal{S}$ can be expressed, similar to (17), in terms of a corresponding classical code $\mathcal{C}_\mathcal{S}$ as follows

$$\mathsf{Pr}(\mathbb{1}|0) = \left(1 - \frac{p}{2}\right)^{2n} \sum_{i=0}^{2n} \mathsf{WE}_i(\mathcal{C}_\mathcal{S}) \tilde{p}^i \tag{22}$$

Since this probability is a polynomial of degree $2n$, determining its value for $2n+1$ distinct values of the physical noise rate p would enable us to determine the weight enumerator of the corresponding classical code. There are two caveats to this approach. First, this approach only works for classical codes $\mathcal{C}_\mathcal{S}$ whose generator matrix corresponds to the symplectic representation of a quantum stabilizer code. Second, this approach requires knowledge of the probability of a certain equivalence class, while DQMLD only outputs the equivalence class with the largest probability; it does not reveal the value of the corresponding probability.

The first caveat can easily be circumvented, for instance by padding the n-bit classical code with n additional 0s, thus obtaining a valid symplectic representation of a n-qubit quantum code (one whose stabilizer generators contain only Z operators). Consequently, all stabilizers have weights between 0 and n and the probability of the trivial logical class can be expressed as (22), where the range of sum is up to n. To circumvent the second caveat, we need to use the DQMLD oracle to obtain equality constraints on the weight enumerator of $\mathcal{C}$. This is done by varying the physical noise rate p, always keeping the syndrome trivial. As mentioned above, at very low noise rate the optimal logical class is $\mathbb{I}$. Increasing the noise rate p, we will reach a *crossing point* p_1 where the DQMLD output changes from $\mathbb{1}$ to $L^\star \neq \mathbb{1}$. At this point, the promise gap condition is violated, i.e, $|\mathsf{Pr}(\mathbb{1}|s) - \mathsf{Pr}(L^\star|s)| \leq \Delta\, \mathsf{Pr}(\mathbb{1}|s)$, which can be expressed in terms of weight enumerators as:

$$\sum_{i=0}^{n} \mathsf{WE}_i(\mathcal{C}) \tilde{p}_1^i - \sum_{i=0}^{n} B_i \tilde{p}_1^i \leq \Delta \sum_{i=0}^{n} \mathsf{WE}_i(\mathcal{C}) \tilde{p}_1^i \tag{23}$$

where B_i are the weight enumerators of an affine code. In the case where $\Delta = 0$, this crossing point provides an equality condition between two polynomials, which is what we are seeking. But since these are polynomials with integer coefficients, knowing the location of a crossing point within a finite accuracy, which translate into a finite promise gap Δ, is enough to determine the exact crossing point, see Lemma. 3. As we will show, there exists a fixed range of p that provably contains a unique crossing point, enabling a polynomial-time accurate determination of the crossing point using a binary search procedure.

This gives us one potential equality, but introduces more unknown coefficients B_i. To get additional linear constraints, we modify the code in a very special way, described in Sec. VI-B. This modification requires adding one *tunable* qubit and one stabilizer generator. By varying the noise rate on the tunable qubit over a range of values, we can change the location of the crossing point, and thus obtain new linear constraints relating $\{\mathsf{WE}_i(\mathcal{C})\}_{i=0}^n$ and the $\{B_i\}_{i=0}^n$. Repeating this procedure $2n + 2$ times and making sure that all the linear constraints are linearly independent Sec. VI-D enable us to determine the weight enumerator coefficients. While the ability to change the noise rate of the tunable qubit gives us more linear constraints, it breaks the requirement that the noise model be the independent X-Z channel with the same strength on all qubits. We will fix this problem in App. A by showing that the required channel can be simulated by concatenating the code with a Shor code. In fact, we will use this technique repeatedly in our proof.

B. Stabilizer Code Construction

Let G be the $k \times n$ generator matrix of an (n, k) classical linear code $\mathcal{C} = \{x \in \mathbb{Z}_2^n : x = yG,\ y \in \mathbb{Z}_2^k\}$. Denote $\{g_i\}_{i=1,\ldots k}$ the rows of G and let $\{g_i\}_{i=k+1\ldots n}$ be a generating set of the complement of the row space of G, i.e. in such a way that $\{g_i\}_{i=1\ldots n}$ span $\mathbb{Z}_2^n$. Construct a matrix $\tilde{G}$ with rows$\{g_i\}_{i=1\ldots n}$. This matrix is full rank, and therefore has an inverse H that can be computed efficiently, and obeys $\tilde{G}H^T = \mathbb{I}$. Denote the rows of H by $\{h_i\}_{i=1}^n$.

We define a $[[2n-k-1, 1]]$ quantum code with stabilizer generators and logical operators given by

$$S_i = Z^{g_i} : i = 1, \ldots, k \tag{24}$$

$$S_{k+i} = Z^{g_{k+i}} \otimes Z_{n+i} : i = 1, \ldots, n-k-1 \tag{25}$$

$$S_{n-1+i} = X^{h_{k+i}} \otimes X_{n+i} : i = 1, \ldots, n-k-1 \tag{26}$$

$$\overline{Z} = Z^{g_n} \tag{27}$$

$$\overline{X} = X^{h_n}, \tag{28}$$

where it is implicitly assumed that operators are padded to the right by identities to be elements of $\mathcal{G}_{n+k-1}$. The validity of the resulting code can be verified from the fact that

1) There are in total $2n - k - 1$ qubits.
2) The $2n - k - 2$ stabilizer generators are independent. This follows from the linear independence of the h_i and the linear independence of the g_i, together with the fact that X-type operators are linearly independent of Z type generators.
3) The stabilizer generators mutually commute. This is trivial among the first $n - 1$ generators as they contain only Z operators and similarly among the last $n - k - 1$ last generators. Between these two sets, the commutation follows from the fact that $h_i \cdot g_j = 0$ except when $i = j$, in which case the presence of additional X and Z on the $n + i$th qubit ensures commutation.
4) The logical operators commute with the stabilizer generators. This follows from the fact that $h_i \cdot g_j = 0$ for $i \neq j$, and the fact that X-type operators commute among themselves and similarly for Z-type operators.

As discussed in Sec. IV-B, the probability of the trivial logical class $\mathbb{I}$ given a trivial syndrome $s = 0$ is simply the sum of the probabilities of all stabilizer group elements. Suppose now that the last $n - k - 1$ qubits are error-free, while the

other qubits are subject to an independent X-Z channel. Then, the probability of an element of $\mathcal{S}$ is zero if it contains a generator S_i from the above list with $i > k$. Otherwise, this element of $\mathcal{S}$ can be written as $S = Z^x \otimes \mathbb{I}$ for some $x \in \mathbb{Z}_2^n$ and its probability is $(p/2)^{|x|}(1 - p/2)^{2n-|x|}$. We conclude that the probability of the trivial logical class is given by (22) with $\mathcal{C}$ the classical code defined by the generating matrix G.

Constraints on the weight enumerator polynomial will be obtained by finding crossing points where $\mathsf{Pr}(\mathbb{I}|0) \approx \mathsf{Pr}(L|0)$ with $L \neq \mathbb{I}$. For technical reasons, we would like to be able to choose which L will be the one realizing the crossing. This is because we want to force the crossing to happen with the same L every time. To do this, we will modify the stabilizer by adding an extra qubit and making the transformations

$$S_i \to S_i \otimes \mathbb{I} \tag{29}$$

$$\overline{Z} \to \overline{Z} \otimes \mathbb{I} \tag{30}$$

$$\overline{X} \to \overline{X} \otimes X \tag{31}$$

to the stabilizers and logical operators, and adding the following stabilizer generator

$$S_{2n-k} = \overline{Z} \otimes Z. \tag{32}$$

This defines an $[[2n-k, 1]]$ stabilizer code, and its validity can easily be verified given the commutation relations worked out above. Moreover, if we assume that the added $(2n-k)$th qubit is also error-free, then the probability of the trivial logical class $\mathbb{I}$ given a trivial syndrome $s = 0$ is unchanged, and moreover the only other logical class with non-zero probability is the one associated to $\overline{Z}$, i.e. $\mathsf{Pr}(\overline{X}|0) = \mathsf{Pr}(\overline{Y}|0) = 0$.

We need to perform one last modification to the code in order to be able to tune the crossing point, and hence obtain linearly independent equalities between $\mathsf{Pr}(\mathbb{I}|0)$ and $\mathsf{Pr}(\overline{Z}|0)$. This transformation is quite similar to the previous one, and given by

$$S_i \to S_i \otimes \mathbb{I} \tag{33}$$

$$\overline{Z} \to \overline{Z} \otimes Z \tag{34}$$

$$\overline{X} \to \overline{X} \otimes \mathbb{I} \tag{35}$$

and adding the following stabilizer generator

$$S_{2n-k+1} = \overline{X} \otimes X. \tag{36}$$

For this last qubit, we will assume a noise model where $p_X = p_Y = 0$, $p_Z = q$, and $p_{\mathbb{I}} = 1 - q$ with q being a tunable parameter. With this last choice, the only two non-zero probabilities of logical class conditioned on the trivial syndrome are given by

$$\mathsf{Pr}(\mathbb{I}|0) = \frac{1}{\mathcal{Z}}(1-q)\sum_{i=0}^{n} \mathsf{WE}_i(\mathcal{C})\,\tilde{p}_1^i \tag{37}$$

$$\mathsf{Pr}(\overline{Z}|0) = \frac{1}{\mathcal{Z}} q \sum_{i=0}^{n} B_i\,\tilde{p}_1^i, \tag{38}$$

where $\tilde{p} = p/(2-p)$ as above, $\mathcal{Z}$ is a suitable normalization factor, and B_i are the weight enumerators of the affine code associated to the $\overline{Z}$ logical class

$$B_i = \big|\{x \in \mathcal{C} + g_n : |x| = i\}\big|. \tag{39}$$

A crossing point is observed when

$$\upsilon \sum_{i=0}^{n} \mathsf{WE}_i(\mathcal{C}) \left(\frac{\tilde{p}_1}{2}\right)^i - \sum_{i=0}^{n} B_i \left(\frac{\tilde{p}_1}{2}\right)^i \leq \Delta\upsilon \sum_{i=0}^{n} \mathsf{WE}_i(\mathcal{C}) \left(\frac{\tilde{p}_1}{2}\right)^i \tag{40}$$

where $\upsilon = (1-q)/q$ is a tunable parameter over the positive reals. Changing the value of υ will change the crossing point between these two logical classes, and provide linear constraints between two degree n polynomials. If we can identify $2n+1$ such crossing points, it would provide enough information to retrieve the two polynomials, and hence solve the weight enumerator problem.

C. Finding Crossing Points

At this point, we have a deterministic procedure that, given any classical linear code $\mathcal{C}$, can be used to generate linear constrains on its weight enumerator coefficients. Clearly, the overhead in the runtime of this procedure is the time required to spot a crossing point. A crossing point can potentially be observed at a physical noise rate p anywhere between 0 and 1. One obvious indication of a crossing point is the switch in the output of the DQMLD oracle as we move p across the crossing point. However if we move p across two crossing points, we will not notice any net switch in the outputs of the DQMLD oracle. For this reason, we now want to restrict the values of p to a range where we can prove that there is at most one crossing point. This will be possible by restricting the tunable parameter υ to a small interval near 0.

Lemma 3: Given the stabilizer code defined above with a tunable parameter $\upsilon \leq (1-\Delta)n^{-d}$, where $1 \leq d \leq |g_n| \leq n$ is the distance between $\mathcal{C}$ and $\mathcal{C} + g_n$, then there exists exactly one crossing point between the pair of logical classes $\mathbb{1}$ and $\overline{Z}$ in the interval $0 \leq p \leq 1/n$.

Proof: First, note that for sufficiently small p and for any value of υ, DQMLD will output the identity class, simply because its probability is a polynomial in p with a constant term, and the probabilities of all other logical classes contain no constant term. Furthermore, we claim that the probability of the trivial logical class is a strictly decreasing function of the noise rate p, when $0 \leq p \leq 1/n$. To justify this, it suffices to show that the derivative of $\mathsf{Pr}(\mathbb{1}|0)$ is strictly negative in the prescribed range of noise rates. Recalling the weight enumerator polynomial for $\mathsf{Pr}(\mathbb{1}|0)$ and computing the derivative w.r.t p, we find:

$$\frac{\partial\,\mathsf{Pr}(\mathbb{1}|0)}{\partial p} = \frac{1}{2}\sum_{i=0}^{n} \mathsf{WE}_i(\mathcal{C}) \left(\frac{p}{2}\right)^i \left(1-\frac{p}{2}\right)^{2n-i} \times \left[\frac{i}{p/2} - \frac{2n-i}{1-p/2}\right] \tag{41}$$

$$= \sum_{i=1}^{n} \mathsf{WE}_i(\mathcal{C}) \left(\frac{p}{2}\right)^i \left(1-\frac{p}{2}\right)^{2n-i} \left[\frac{i-np}{p(1-p/2)}\right] - n\left(1-\frac{p}{2}\right)^{2n-1} \tag{42}$$

where the last term is added because we have changed the range of summation. It only remains to show that the expression above is strictly negative. Clearly, this cannot be true for all $p \in [0, 1]$. However, when $p \leq 1/n$, we know that the first term in (42) is non-negative. Indeed, all of its terms are positive by definition, except the one in square bracket which is non-negative when $p \leq 1/n$. We claim now that when $p \leq 1/n$, the second term of (42) is negative and greater in norm than the first one, so the entire expression is strictly negative. This can be observed by the following inequality:

$$\begin{aligned}\sum_{i=1}^{n} \mathsf{WE}_i(\mathcal{C}) \left(\frac{p}{2}\right)^i \left(1-\frac{p}{2}\right)^{2n-i} \left[\frac{i-np}{p(1-p/2)}\right] \\ < \sum_{i=1}^{2n} \binom{2n}{i} \left(\frac{p}{2}\right)^i \left(1-\frac{p}{2}\right)^{2n-i} \left[\frac{i-np}{p(1-p/2)}\right] \\ = \sum_{i=0}^{2n} \binom{2n}{i} \left(\frac{p}{2}\right)^i \left(1-\frac{p}{2}\right)^{2n-i} \left[\frac{i-np}{p(1-p/2)}\right] \\ + n\left(1-\frac{p}{2}\right)^{2n-1} = n\left(1-\frac{p}{2}\right)^{2n-1},\end{aligned}$$

where the last equality is a consequence of the fact that the mean of a binomial distribution with parameters $2n$ and $p/2$ is np. Hence, we see that indeed the probability of the trivial logical class is strictly decreasing when $0 \leq p \leq 1/n$. Since there are only two logical classes in our setting, it is clear that the probability associated to the other class is increasing in that interval.

We have identified an interval where the probabilities are monotonic, and what remains to be shown is that there is indeed a crossing point inside this interval when the parameter υ is chosen carefully. Intuitively, we can see that decreasing the value of υ (and hence of the trivial logical class) will decrease the value of p where the first crossing point occurs. We will now set an upper bound $p_{\max}$ on the value of p where the first crossing point occurs. The first crossing point will occur at the latest when $\upsilon \,\mathsf{Pr}(\mathbb{1}|0) - \mathsf{Pr}(\overline{Z}|0) = \Delta\,\mathsf{Pr}(\mathbb{1}|0)$, so equivalently $\upsilon(1-\Delta) = \mathsf{Pr}(\overline{Z}|0)/\mathsf{Pr}(\mathbb{1}|0)$. On the other hand, the ratio $\mathsf{Pr}(\overline{Z}|0)/\mathsf{Pr}(\mathbb{1}|0)$ is lower-bounded by $[p/(2-p)]^d$ since each word in $x \in \mathcal{C}$ is mapped onto a word $y = x + g_n$ of weight at most $|x|+|g_n|$ in $\mathcal{C}+g_n$. Hence, the first crossing occurs for a value of p lower or equal to the point where

$$\upsilon = (1-\Delta)^{-1}\left[\frac{p}{2-p}\right]^d, \tag{43}$$

$$\text{or equivalently } p_{\max} = \frac{2}{1+(1-\Delta)^{\frac{1}{d}}\upsilon^{-\frac{1}{d}}}. \tag{44}$$

By choosing

$$p_{\max} \leq \frac{1}{n}, \quad \text{or equivalently } \upsilon \leq (1-\Delta)^{-1} n^{-d}, \tag{45}$$

we are sure that the first crossing point occurs in the monotonic region, and hence that the interval $0 \leq p \leq 1/n$ contains a single crossing point. ∎

The existence of a unique crossing point in the interval $p \in (0, 1/n]$ enables a *binary-search like* procedure to narrow down on the possible location of a crossing point. If DQMLD produces the same output for a pair of p's in that interval, it implies that no such crossing point exists between them, and furthermore, that a crossing point lies outside of this interval. This halves the size of the interval between the next pair of points to be queried with. Hence the location of the crossing point can be obtained to an accuracy of 2^{-j} with at most j queries to DQMLD.

D. Independent Constraints and the Promise Gap

We have described a polynomial-time method to estimate the location of crossing points within exponential accuracy, i.e. values of p where (40) is fulfilled. It remains to show that a polynomial number of such linear constraints are sufficient to determine the weight enumerators of the linear code. Every crossing point provides a linear constraint on $\{\mathsf{WE}_i(\mathcal{C}), B_i\}_{i=0}^n$. However these conditions are inequalities. In the following lemma, we establish that for Δ sufficiently small, the system of inequalities provides the same integer solutions as the system of equalities (i.e., obtained with $\Delta = 0$).

Lemma 4: Provided that $\Delta \leq 2[2+n^\lambda]^{-1}$, the first λ weight enumerator coefficients $\{\mathsf{WE}_i(\mathcal{C})\}_{i=0}^{\lambda}$ can be extracted efficiently from the value of $2n+1$ crossing points p_k that produce linearly independent inequalities (40).

Proof: Given a crossing point p_k, we can rewrite the inequality (40) as an equality

$$\begin{aligned}-\upsilon_k \sum_{i=0}^{n} \mathsf{WE}_i(\mathcal{C})\,\tilde{p_k}^i + \sum_{i=0}^{n} B_i\,\tilde{p_k}^i \\ + \delta_k \Delta\left[\upsilon_k \sum_{i=0}^{n} \mathsf{WE}_i(\mathcal{C})\,\tilde{p_k}^i\right] = 0\end{aligned}$$

by introducing an unknown parameter $0 \leq \delta_k \leq 1$. Given $2n+1$ distinct crossing points p_k, we obtain a linear system $\mathsf{M}\cdot\omega - \Delta(\mathcal{J}\cdot\mathsf{M})\cdot\omega = 0$, where $||\mathcal{J}||_\infty \leq 1$ and ω is the vector containing the weight enumerator coefficients. Since the location of a crossing point is invariant under multiplying both weight enumerators $\{\mathsf{WE}(\mathcal{C})_i, B_i\}_{i=0}^n$ by a constant, it is clear that $2n+1$ linear independent constraints alone would result in the trivial solution. However, the normalization condition

$$\sum_{i=0}^{n} (\mathsf{WE}(\mathcal{C})_i + B_i) = 2^n \tag{46}$$

along with the $2n+1$ linearly independent constraints obtained from the location of crossing points is sufficient to determine the weight enumerators coefficients in ω,

$$\omega = \left[[\mathbb{1} + \Delta\mathcal{J}]\cdot\mathsf{M}\right]^{-1}\cdot b \tag{47}$$

where we have defined

$$\mathsf{M} = \begin{bmatrix} 1\ldots & \tilde{p}_1^n & -\upsilon_1 & \ldots & -\upsilon_1\tilde{p}_1^n \\ 1\ldots & \tilde{p}_2^n & -\upsilon_2 & \ldots & -\upsilon_2\tilde{p}_2^n \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 1\ldots & \tilde{p}_{2n+1}^n & -\upsilon_{2n+1} & \ldots & -\upsilon_{2n+1}\tilde{p}_{2n+1}^n \\ 1\ldots & 1 & 1 & \ldots & 1 \end{bmatrix} \tag{48}$$

$$b = \left(0 \cdots 02^n\right)^{\mathsf{T}} \tag{49}$$

$$\omega = \left(B_0 \cdots B_n \mathsf{WE}_0(\mathcal{C}) \cdots \mathsf{WE}_n(\mathcal{C})\right)^{\mathsf{T}} \tag{50}$$

$$\mathcal{J} = \left[\begin{array}{ccc|ccc} 0 & & & & & \\ & \ddots & & & \mathbf{0} & \\ & & 0 & & & \\ \hline & & & -\delta_1 & & \\ & \mathbf{0} & & & \ddots & \\ & & & & & -\delta_{n+1} \\ \hline 0 & \ldots & 0 & 0 & \ldots & 1 \end{array}\right] \tag{51}$$

$$\text{with} \qquad 0 \le \delta_i \le 1, \quad \forall 1 \le i \le n+1, \tag{52}$$

$$\text{and} \qquad \upsilon_k = \frac{\sum_{i=0}^{n} B_i \tilde{p}_k^i}{\sum_{i=0}^{n} \mathsf{WE}(\mathcal{C})_i \tilde{p}_k^i}. \tag{53}$$

Denote ω_0 the solution of this linear system in the case where $\Delta = 0$, i.e., $\omega_0 = \mathsf{M}^{-1} \cdot b$. This solution is not the weight enumerator, because we omitted the $\Delta\mathcal{J}$ term. However, since the weight enumerator coefficients are integers, we expect that for Δ sufficiently small, rounding up each component of ω_0 to the nearest integer will give the correct answer ω. For this approach to succeed, it suffices to ensure that $|\omega_i - \omega_{0,i}| \le 1/2$. Using (47) for ω and a similar one for ω_0 (with $\Delta = 0$), we can rewrite this condition in terms of the constraint matrix in (48) as:

$$|\omega_i - \omega_{0,i}| = \left| \left(\left[[\mathbb{1} + \Delta\mathcal{J}] \cdot \mathsf{M} \right]_{i,n}^{-1} - [\mathsf{M}^{-1}]_{i,n} \right) 2^n \right| \le \frac{1}{2} \tag{54}$$

where we have exploited the special structure of b which ensures that only the elements in the last column of the inverse matrix have a non-trivial contributions to the components of ω.

Using properties of matrix multiplication and the ∞-norms of matrices in (54), we obtain

$$\left| \left[[\mathbb{1} + \Delta\mathcal{J}] \cdot \mathsf{M} \right]^{-1} - \mathsf{M}^{-1} \right|_{i,n} \le \left(\left|\left| [\mathbb{1} + \Delta\mathcal{J}]^{-1} \right|\right|_{\infty} - 1 \right) \left| \left[\mathsf{M}^{-1} \right]_{i,n} \right|. \tag{55}$$

The explicit form of the matrix in (52) can be used to verify that $\left|\left|(\mathbb{1} + \Delta\mathcal{J})^{-1}\right|\right|_{\infty} \le (1 - \Delta)^{-1}$ and also $[\mathsf{M}^{-1}]_{i,n} 2^n = \mathsf{WE}_i(\mathcal{C}) \le n^i, \forall 1 \le i \le n+1$. Hence we can now state a sufficient condition for (54) to hold:

$$\left[\frac{1}{1 - \Delta} - 1 \right] n^i \le \frac{1}{2}, \quad \forall 1 \le i \le n+1 \tag{56}$$

The lemma follows by imposing this condition for all $i \le \lambda$, yielding

$$\left[\frac{1}{1 - \Delta} - 1 \right] n^{\lambda} \le \frac{1}{2} \Rightarrow \Delta \le \frac{2}{2 + n^{\lambda}}. \tag{57}$$

■

Though it is clear that we must have $2n + 1$ linearly independent constraints on $\{\mathsf{WE}_i(\mathcal{C}), B_i\}_{i=0}^n$ along with a promise gap of $\Delta \le 2[2 + n^{\lambda}]^{-1}$, it is not immediately clear how many distinct crossing points need to be located to obtain these constraints, i.e. some of the crossing points associated to distinct pairs (υ_l, p_l) may result in linearly dependent constraints (40). The following lemma ensures that we can always efficiently find linearly independent constraints.

Lemma 5: The identification of at most $8n^2$ distinct crossing points (υ_l, p_l) of (40) will produce $2n + 1$ linearly independent constraints.

Proof: The $(2n + 2) \times (2n + 2)$ matrix in (48) can be represented as $\mathsf{M} = [\mathsf{V}|\mathsf{D} \cdot \mathsf{V}]$ where $\mathsf{V}_{i,j} = \tilde{p}_i^j$ is a $(2n+2) \times (n+1)$ *Vandermonde matrix* and $\mathsf{D}_{i,j} = -\upsilon_i \delta_{i,j}$ is a diagonal square matrix. We shall prescribe a polynomial-time method to construct a full-rank matrix of this form.

Let T_l represent the sub-matrix of M formed by taking the first l column entries of the first l rows. Hence T_l is a square matrix. Moreover, notice that a sufficient condition for the matrix in (48) to describe l independent constraints is that T_l must have a non-vanishing determinant. Hence, it suffices to show that each new constraint can be generated such that the corresponding square matrix T_l satisfy $\det(\mathsf{T}_l) \neq 0$.

Consider the simple case when $l \le n + 1$. In this case, T_l is always given by a *Vandermonde matrix*, which is full rank if and only if all p_i are distinct [35]. Hence, this implies that $n+1$ distinct values of υ_i suffice to generate $n+1$ constraints that are necessarily independent, since different υ's necessarily produce different p's (see Lemma. 6).

For $l > n + 1$, it is not immediately clear if the procedure prescribed in Sec. VI-B will provide linearly independent constraints even if we choose all υ_i to be distinct. However, a sufficient condition for independency is again the non-vanishing of $\det(\mathsf{T}_l)$. Assume that the choices of $\{\upsilon_1, \ldots, \upsilon_{l-1}\}$ are such that $\det(\mathsf{T}_{l-1}) \neq 0$. The Laplace-expansion of the determinant suggests that $\det(\mathsf{T}_l) = 0$ when υ_l is given by:

$$\upsilon_l = \frac{\sum_{i=0}^{n} E_i \tilde{p}_l^i}{\tilde{p}_{n+t}^{t-1} \det(T_{l-1}) + \sum_{j=0}^{t-2} \tilde{p}_{n+j}^j C_j} \tag{58}$$

where $l > n + 1$ and E_i, C_i are constants (independent of p_{n+i}; they are minors of p^i and υp^i respectively).

The above equation classifies the particular values of υ_l that fail to generate l linearly independent constraints on $\{\mathsf{WE}_i(\mathcal{C}), B_i\}_{i=0}^n$. Note that (υ_l, p_l) are related by (53). Hence, equating this with (58) results in a rational polynomial in p_l, which has at most $2n$ roots, unless the expressions in Eqs. (53) and (58) are identical. Assume for now that these expressions are not identical. Then we can choose to sample $2n + 1$ different values of υ_l according to (53), and for at least one of them, the resulting p_l will violate (58), so the pair (υ_l, p_l) will produces a linearly independent constraint. On the other hand, if the two functions in Eqs. (53) and (58) are identical, then we have that $\forall p$:

$$\frac{\sum_{i=0}^{n} \mathsf{WE}_i(\mathcal{C}) \tilde{p}^i}{\sum_{i=0}^{n} B_i \tilde{p}^i} = \frac{\sum_{i=0}^{n} E_i \tilde{p}^i}{\tilde{p}^{k-1} \det(T_{n+k-1}) + \sum_{j=0}^{k-2} \tilde{p}^j C_j}. \tag{59}$$

The above relation immediately fixes the coefficients $\{\mathsf{WE}_i(\mathcal{C}), B_i\}_{i=0}^n$ by just comparing the powers of $\tilde{p}$.

Since we must sample at most $2n + 1$ crossing points to obtain a linearly independent constraint on $\{\mathsf{WE}_i(\mathcal{C}), B_i\}_{i=0}^n$, we require at most $(2n + 1)(2n + 1)$ or (for expressional convenience) $8n^2$ distinct crossing points to construct the a full rank matrix of the form in (48). ■

The last ingredient we need is a bound on the distance between crossing points. Remember that we are only able to locate the value of a crossing probability p_k to exponential accuracy. Thus, it is necessary that changing the tunable parameter υ has a significant effect on the value of the crossing point in order to generate linearly independent constraints (with significantly different values of p_k). Combining the restriction on the values of the tunable parameter in (45) with Lemma. 5 immediately tells that the smallest change in the tunable parameter will be at least $(1-\Delta)n^{-n}8^{-1}n^{-2}$. This naturally implies a minimum separation between two crossing points, as the lemma below addresses:

Lemma 6: Let $\{\upsilon_k, p_k\}_{k=1}^{2n+1}$ denote crossing points of a stabilizer code constructed as above with $|\upsilon_k - \upsilon_l| \geq [8n^2(1-\Delta)n^n]^{-1}$. Then $|p_i - p_j| \geq 4^{-n\log_2 n}$.

Proof: We need to relate the difference in υ to a difference in p. For this, let us recall that if γ is a small change in υ and δ a small change in p, then $\gamma = \delta(d\upsilon/dp)$, or equivalently: $\gamma\,(d\upsilon/dp)^{-1} = \delta$. Computing the derivative using the expression for υ in (53), we have:

$$\delta \geq \gamma \cdot \left(\frac{(1-p)^2}{\sum_{i,j=0}^{n} \left[i\, B_i\, \mathsf{WE}_j(\mathcal{C}) \left(\frac{p}{1-p}\right)^j \left(\frac{p}{1-p}\right)^i \right]} \right)$$
$$\geq \gamma \cdot \frac{(1-p)^{2n+2}}{n} \Rightarrow \delta \geq \gamma \cdot \frac{4^{1-n}}{n} \tag{60}$$

Since $\gamma \geq (1-\Delta)n^{-n}8^{-1}n^{-2}$, we find: $\delta \geq (1-\Delta)n^{-n}8^{-1}n^{-3}4^{1-n} \geq 4^{-n\log_2 n}$. ∎

Hence it suffices to estimate the location of each crossing point to within an accuracy of $4^{-n\log_2 n}$, implying that we must run at most $2n\log_2 n$ iterations of the above binary-search like procedure, mentioned at the end of Sec. VI-C, to locate the crossing point. Assuming that each query to a DQMLD oracle is answered in constant time, the above procedure takes $\mathcal{O}(n\log_2 n)$ time.

In appendix App. A we construct a repetition code with polynomially many qubits to encode a single qubit, such that the resulting noise on the encoding qubit obeys $p_Z = q$, $p_I = 1-q$, and $p_Z = p_Y = 0$ with an exponential accuracy (the error-free model is a special case $q = 0$). This implies that the tunable parameters υ_k used to find crossing points can be set to exponential accuracy.

With these, we are in a position to prove our main result.

Proof of Thm. (2): Given an input classical linear code $\mathcal{C}$, we built a generating set for a stabilizer code, with two logicals, that satisfies (22). Appending additional qubits to this code and choosing a specific channel Sec. VI-B, on those qubits, enabled the introduction of a tunable parameter in the logical class probabilities Eqs. (37) and (38). Varying this tunable parameter as per Lemma. 3, with a polynomial number of queries to a DQMLD oracle, we can estimate the location of a crossing point to within exponential accuracy, thereby providing a linear constraint on the weight enumerator coefficients. Repeating this procedure $2n+1$ times, we showed in Lemma. 4 that, as long as the promise gap is $2[2+n^{\lambda}]^{-1}$, the system of inequalities yield the same solution to the first λ weight enumerator coefficients of $\mathcal{C}$, up to an integer approximation, as the system of equalities considered without any promise gap. Since $\lambda = \mathsf{polylog}(n)$ in Thm. 2, it suffices to have a promise gap in DQMLD that is $1/\mathsf{quasi\text{-}polynomial(n)}$. It followed from Lemma. 5 that at most $8n^2$ crossing points are sufficient to generate the necessary linearly independent constraints on the weight enumerator coefficients and from Lemma. 6 that the accuracy on their location, achievable in polynomial time, is sufficient. Lastly, we showed in App. A that it suffices to direct all queries to a DQMLD oracle on a independent X-Z channel. Thus we prove Thm. 2. □

VII. Conclusion

We will close with mentioning a few open problems which we were not able to address in this paper. In the course of this paper we have addressed the optimal decoding problem on an independent X-Z channel. However, the same can be done for a depolarizing channel by introducing the notion of a generalized weight described in [17]. Hence, Sec. VI-B of the paper will undergo certain modifications when choosing to address a depolarizing channel.

The key problem turns out to be the classification of the *parametrized* complexity of the decoding with the promise gap parameter Def. 10, denoted by Δ. We know the complexities for two extreme cases, namely DQMLD is #P-Complete when $\Delta = 1/\mathsf{quasi\text{-}polynomial}(n)$, c.f. Thm. 2 and in NP when $\Delta = 1 - 2^{-n-k}$, c.f. Lemma. 1. However, for a vast intermediate range of Δ, complexity of DQMLD remains open. As described in Sec. IV-B, for a code encoding a single qubit, the promise gap Δ is related to the decoding failure probability ϵ as $\Delta = 1-\epsilon$. Thus, the two extreme cases considered above correspond respectively to optimal decoding in a very noisy regime (failure probability approaching unity) and optimal decoding with an exponentially small failure probability. Unfortunately, the case of practical interest falls somewhere in between.

The complexity of any problem is only a highlight of the runtime of any algorithm on the worst case instance of the problem. Hence it is of practical interest to know the runtime of the algorithm for any *typical* instance. This could refer for instance to a typical syndrome or a random code. However—for the same reason that random classical codes don't have typical low-weight codewords—, random codes are non-degenerate [6], [36], so the decoding is not expected to be affected by the degeneracy in errors, so our result is probably not relevant in this setting. However, for the practically relevant class of *sparse codes*, the complexity of optimal decoding strategy remains an important open question.

Lastly, our analysis has focused on stabilizer codes over Pauli channels. This is particularly convenient due to the discrete nature of the resulting decoding problem. This setting could be generalized in two obvious ways. First, we could consider codes that are not stabilizer codes, defined from a set of commuting projectors. There exists a growing interest for those codes, particularly in the setting of topological quantum order [31], [37]–[39]. In this setting, we could study for instance the decoding problem for systems that support non-Abelian anyons [40]. Second, we could consider errors that are

not described by Pauli operators. This problem is of practical importance because no real-world device undergoes a Pauli channel; for instance the physical process of relaxation is not described by a Pauli channel.

APPENDIX A
SIMULATING PAULI CHANNELS WITH AN INDEPENDENT X-Z CHANNEL

In our reduction, we have used two features of the general memoryless Pauli channel, that are not intrinsic to an independent X-Z channel. They involve the freedom of assigning unequal noise rates for $\mathbb{1}, X, Y$ and Z type errors on some qubits of the code. However, when a qubit of the code is encoded into an auxiliary code, the the optimal decoder on this concatenated code will replace the physical noise rates for the qubit with the logical noise rates (logical class probabilities) of the auxiliary code [14]. Therefore, the physical noise rates for that qubit can be controlled by controlling the logical class probabilities of the auxiliary code. The latter can be achieved by varying the syndrome on the auxiliary code.

In our case, the auxiliary code is the Shor code [33], [34] on a $n_1 n_2$ qubits, with $n_1, n_2 \sim \mathsf{poly}(n)$. The stabilizers of this code can be represented graphically on a $n_1 \times n_2$ lattice with a qubits at each vertex, see Fig. 2a. Each horizontal link between a pair of vertices represents a X-type stabilizer on the qubits corresponding to the vertices. A pair of rows represents a Z-type stabilizer on the qubits corresponding to the vertices on those rows.

Any Z-type error chain will turn on a syndrome, represented by a pair of points on two ends of the chain, c.f. Fig. 2b. Let us denote the smallest number of links between the points as ℓ. We will restrict to the case where there is a single continuous error chain, which is confined to a row of the lattice. Hence ℓ completely defines the syndrome. For the syndrome in Fig. 2b, one can immediately write the probabilities for the four logical classes $\overline{\mathbb{1}}$, $\overline{X}$, $\overline{Z}$ and $\overline{Y}$ as:

$$\mathsf{Pr}(\overline{\mathbb{1}}|s)=\frac{\left[\frac{p}{2}\left(1-\frac{p}{2}\right)\right]^{\ell}\left[\left(1-\frac{p}{2}\right)\left(1-\frac{p}{2}\right)\right]^{n_2-\ell}}{\tilde{P}(s)} \tag{61}$$

$$\mathsf{Pr}(\overline{Z}|s)=\frac{\left[\frac{p}{2}\left(1-\frac{p}{2}\right)\right]^{n_2-\ell}\left[\left(1-\frac{p}{2}\right)\left(1-\frac{p}{2}\right)\right]^{\ell}}{\tilde{P}(s)} \tag{62}$$

$$\mathsf{Pr}(\overline{X}|s)=\mathsf{Pr}(\overline{\mathbb{1}}|s)\sum_{i=1}^{n_2}\binom{n_2}{i}\left(\frac{p}{2}\right)^{n_1 i}\left(1-\frac{p}{2}\right)^{2n_1 n_2-n_1 i} \tag{63}$$

$$\mathsf{Pr}(\overline{Y}|s)=\mathsf{Pr}(\overline{Z}|s)\sum_{i=1}^{n_2}\binom{n_2}{i}\left(\frac{p}{2}\right)^{n_1 i}\left(1-\frac{p}{2}\right)^{2n_1 n_2-n_1 i} \tag{64}$$

where:

$$\tilde{P}(s)=\left[\left(\frac{p}{2}\right)^{\ell}\left(1-\frac{p}{2}\right)^{2n_2-\ell}+\left(\frac{p}{2}\right)^{n_2-\ell}\left(1-\frac{p}{2}\right)^{2\ell}\right]\times\sum_{i=0}^{n_2}\binom{n_2}{i}\left(\frac{p}{2}\right)^{n_1 i}\left(1-\frac{p}{2}\right)^{2n_1 n_2-n_1 i} \tag{65}$$

Given a constant υ, we wish to simulate a channel, on the encoded qubit, with $p_I = 1-q, p_Z = q$ such that $(1-q)/q = \upsilon$. This implies a ratio between $\mathsf{Pr}(\overline{\mathbb{1}}|s)$ and $\mathsf{Pr}(\overline{Z}|s)$ equal to υ, and ℓ given by

$$\upsilon := \frac{\mathsf{Pr}(\overline{\mathbb{1}}|s)}{\mathsf{Pr}(\overline{Z}|s)} = \left[\frac{p}{2(1-p)}\right]^{2\ell}\left[\frac{2(1-p)}{p}\right]^{n_2} \tag{66}$$

$$\Rightarrow \ell = \frac{1}{2}\left(n_2 + \frac{|\log_2 \upsilon|}{1+\log_2 \frac{1-p}{p}}\right) \tag{67}$$

where in the last simplification, we have assumed $\upsilon \leq 1$. As a consequence of the upper bound on υ in (45) it suffices to choose $n_2 = 2n$. We will fix the number of columns in the auxiliary code Fig. 2a to $2n$.

In addition to the ratio between $p_{\mathbb{1}}$ and p_Z, the channel on the encoded qubit also requires that $p_X = p_Y = 0$. To achieve this, we must choose a value of n_1 such that the expressions in Eqs. (63) and (64) are vanishingly small. Note that the ratio of the combinatorial sums in the the expressions for $\mathsf{Pr}(\overline{Y}|s)$ and $\tilde{P}(s)$ can be bounded from above as:

$$\frac{\sum_{i=1}^{n_2}\binom{n_2}{i}\left(\frac{p}{2}\right)^{n_1 i}\left(1-\frac{p}{2}\right)^{2n_1 n_2-n_1 i}}{\sum_{i=0}^{n_2}\binom{n_2}{i}\left(\frac{p}{2}\right)^{n_1 i}\left(1-\frac{p}{2}\right)^{2n_1 n_2-n_1 i}} \leq \left(\frac{p}{2-p}\right)^{n_1}+\frac{n_2^2\left(\frac{p}{2-p}\right)^{2n_1}}{1-n_2\left(\frac{p}{2-p}\right)^{n_1}} \tag{68}$$

$$\leq (2n-1)^{-n_1}+\frac{n_2^2(2n-1)^{-2n_1}}{1-n_2(2n-1)^{-n_1}} \tag{69}$$

Hence it suffices to take $n_1 = 2n$ for the above ratio to be bounded above by a vanishingly small number. Moreover, as the ratio is an upper bound for $\mathsf{Pr}(\overline{X}|s)$ and $\mathsf{Pr}(\overline{Y}|s)$, we will fix the number of rows in the auxiliary code Fig. 2 to $2n$.

Summarizing, the qubit is encoded into a Shor code on a $2n \times 2n$ lattice Fig. 2a and the syndrome is chosen as indicated in Fig. 2b with ℓ specified by (67). As a result we have a channel on the qubit with $p_{\mathbb{1}} = 1-q$, $p_Z = q$, $p_X = p_Y = 0$, to within exponential error bars. Note that an error free channel is a special case of the above channel, with $q = 0$. Hence, we repeat the same encoding but choose the syndrome on the Shor code to be trivial.

APPENDIX B
PROOF OF LEMMA. 1

In this appendix, we present a proof of Lemma. 1 which states that with a promise gap $\Delta \geq 1-2^{-n-k}$ and on an independent X-Z channel, the outputs of QMLD and DQMLD are equivalent.

It suffices to demonstrate that, for any syndrome s, the logical class containing the minimum weight error $L_{\min}$ must satisfy (15) with $L^{\star} = L_{\min}$. This will be true whenever $1-\mathsf{Pr}(L_{\min}|s) \leq \Delta = 1-2^{-n-k}$, or in other words

$$\frac{\mathsf{Pr}(L_{\min}, s)}{\mathsf{Pr}(s)} \geq 2^{-n-k}. \tag{70}$$

Suppose that $a(s)$ is the minimum weight of an error, consistent with the syndrome s. Let us derive lower and upper bounds separately for the numerator and the denominator of (70), respectively. Every element in the logical class of $L_{\min}$

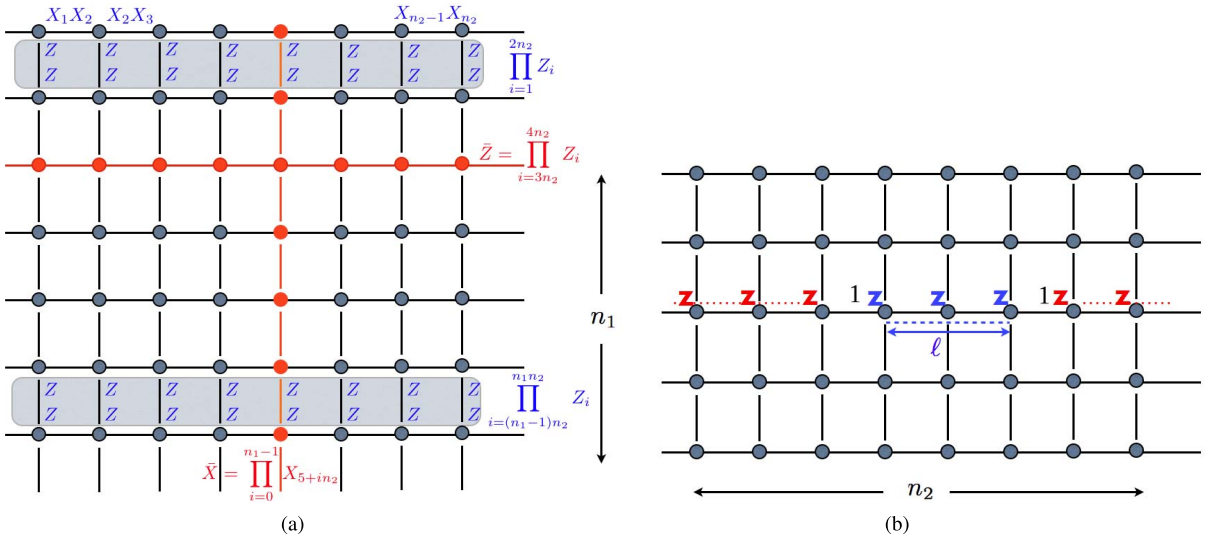


Fig. 2. (a) Stabilizers and logicals of the Shor code on a $n_1 \times n_2$ lattice. (b) Syndrome corresponding to a Z-type error on qubit along a row.

is constructed by taking the product of the minimum weight error with every stabilizer in $\mathcal{S}$. To find the lowest possible value of the probability of the product, we suppose that each product results in an element whose weight is the sum of $a(s)$ and that of the corresponding stabilizer. Hence we have:

$$\mathsf{Pr}(L_{\min}, s) \geq \left[1 - \frac{p}{2}\right]^{2n} \sum_{i=0}^{n} B_i \tilde{p}^{i+a(s)} \tag{71}$$

where $\tilde{p} = p/(2-p)$ and $B_i = |\{g \in \mathcal{S} : \mathsf{wt}_2(g) = i\}|$ is a weight enumerator coefficient of $\mathcal{S}$.

Similarly, all the errors consistent with the syndrome s can be constructed by taking the error of weight $a(s)$ and taking its product with every element of the normalizer $\mathcal{N}(\mathcal{S})$. For an upper bound to the probability of this product, let us suppose that each product has a weight equal to the difference between $a(s)$ and the weight of the corresponding stabilizer. As the weight of the product cannot be lower than $a(s)$ itself, this gives:

$$\mathsf{Pr}(s) \leq \left[1 - \frac{p}{2}\right]^{2n} \left[\sum_{i=0}^{2a(s)} B_i^{\perp} \tilde{p}^{a(s)} + \sum_{i=2a(s)+1}^{n} B_i^{\perp} \tilde{p}^{i-a(s)}\right] \tag{72}$$

where $B_i^{\perp} = |\{g \in \mathcal{N}(\mathcal{S}) : \mathsf{wt}_2(g) = i\}|$ is a weight enumerator coefficient of $\mathcal{N}(\mathcal{S})$ [6].

The ratio of expressions in Eqs. (71) and (72), bounds the quantity in (70) from above, by:

$$\frac{\mathsf{Pr}(L_{\min}, s)}{\mathsf{Pr}(s)} \geq \frac{1}{\sum_{i=0}^{2a(s)} B_i^{\perp} + \tilde{p} \sum_{i=2a(s)+1}^{n} B_i^{\perp}} \tag{73}$$

$$\geq \frac{1}{\sum_{i=0}^{n} B_i^{\perp}} = 2^{-n-k} \tag{74}$$

thereby proving the lemma. □

ACKNOWLEDGEMENTS

We thank Daniel Gottesman for stimulating discussions and Guillaume Duclos-Cianci for careful reading of this manuscript.

REFERENCES

[1] E. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg, "On the inherent intractability of certain coding problems (Corresp.)," *IEEE Trans. Inf. Theory*, vol. 24, no. 3, pp. 384–386, May 1978.

[2] A. Vardy, "The intractability of computing the minimum distance of a code," *IEEE Trans. Inf. Theory*, vol. 43, no. 6, pp. 1757–1766, Nov. 1997.

[3] M.-H. Hsieh and F. L. Gall, "NP-hardness of decoding quantum error-correction codes," *Phys. Rev. A*, vol. 83, p. 052331, May 2011.

[4] K.-Y. Kuo and C.-C. Lu, "On the hardness of decoding quantum stabilizer codes under the depolarizing channel," in *Proc. Int. Symp. Inf. Theory Appl. (ISITA)*, Oct. 2012, pp. 208–211.

[5] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948.

[6] D. Gottesman, "Stabilizer codes and quantum error correction," Ph.D. dissertation, Dept. Phys., Math. Astronomy, California Inst. Technol., Pasadena, CA, USA, 1997.

[7] A. R. Calderbank, E. M. Rains, P. W. Shor, and N. J. A. Sloane, "Quantum error correction and orthogonal geometry," *Phys. Rev. Lett.*, vol. 78, pp. 405–408, Jan. 1997.

[8] M. H. Freedman and D. A. Meyer, "Projective plane and planar quantum codes," *Found. Comput. Math.*, vol. 1, no. 3, pp. 325–332, Jul. 2001.

[9] D. J. C. MacKay, G. Mitchison, and P. L. McFadden, "Sparse-graph codes for quantum error correction," *IEEE Trans. Inf. Theory*, vol. 50, no. 10, pp. 2315–2330, Oct. 2004.

[10] T. Camara, H. Ollivier, and J.-P. Tillich, "A class of quantum LDPC codes: Construction and performances under iterative decoding," in *Proc. ISIT*, Jun. 2007, pp. 811–815.

[11] J. Tillich and G. Zemor, "Quantum LDPC codes with positive rate and minimum distance proportional to $n^{1/2}$," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun./Jul. 2009, pp. 799–803.

[12] D. Poulin, J. Tillich, and H. Ollivier, "Quantum serial turbo codes," *IEEE Trans. Inf. Theory*, vol. 55, no. 6, pp. 2776–2798, Jun. 2009.

[13] C. Wang, J. Harrington, and J. Preskill, "Confinement-Higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory," *Ann. Phys.*, vol. 303, no. 1, pp. 31–58, Jan. 2003.

[14] D. Poulin, "Optimal and efficient decoding of concatenated quantum block codes," *Phys. Rev. A*, vol. 74, no. 5, p. 052333, Nov. 2006.

[15] G. Duclos-Cianci and D. Poulin, "Fast decoders for topological quantum codes," *Phys. Rev. Lett.*, vol. 104, p. 050504, Feb. 2010.

[16] E. Pelchat and D. Poulin, "Degenerate Viterbi decoding," *IEEE Trans. Inf. Theory*, vol. 59, no. 6, pp. 3915–3921, Jun. 2013.

[17] K.-Y. Kuo and C.-C. Lu. (2013). "On the hardnesses of several quantum decoding problems." [Online]. Available: http://arxiv.org/abs/1306.5173
[18] M. N. Vyalyi. (2003). "Hardness of approximating the weight enumerator of a binary linear code." [Online]. Available: http://arxiv.org/abs/cs/0304044
[19] I. Briquel, "Complexity issues in counting, polynomial evaluation and zero finding," Ph.D. dissertation, Dept. Math., Univ. Hong Kong, Hong Kong, 2011.
[20] L. G. Valiant, "The complexity of enumeration and reliability problems," *SIAM J. Comput.*, vol. 8, no. 3, pp. 410–421, 1979.
[21] S. Arora and B. Barak, *Computational Complexity: A Modern Approach*. Cambridge, U.K.: Cambridge Univ. Press, 2009.
[22] S. Toda, "PP is as hard as the polynomial-time hierarchy," *SIAM J. Comput.*, vol. 20, no. 5, pp. 865–877, Oct. 1991.
[23] A. Barg, "Handbook of coding theory," in *Complexity Issues in Coding Theory*, vol. 1. Amsterdam, The Netherlands: Elsevier, 1997, pp. 649–754.
[24] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge Series on Information and the Natural Sciences). Cambridge, U.K.: Cambridge Univ. Press, 2000.
[25] J. Preskill, "Quantum error correction," in *Lecture Notes for Physics 229: Quantum Information and Computation*, 1998. [Online]. Available: http://www.theory.caltech.edu/people/preskill/ph229/
[26] E. R. Berlekamp, *Algebraic Coding Theory*. New York, NY, USA: McGraw-Hill, 1984.
[27] A. Honecker, M. Picco, and P. Pujol, "Universality class of the Nishimori point in the 2D±J random-bond Ising model," *Phys. Rev. Lett.*, vol. 87, p. 047201, Jul. 2001.
[28] H. Bombin, R. S. Andrist, M. Ohzeki, H. G. Katzgraber, and M. A. Martin-Delgado, "Strong resilience of topological codes to depolarization," *Phys. Rev. X*, vol. 2, p. 021004, Apr. 2012.
[29] D. Poulin and Y. Chung, "On the iterative decoding of sparse quantum codes," *Quantum Inf. Comput.*, vol. 8, no. 10, pp. 987–1000, Nov. 2008.
[30] S. M. Aji and R. J. McEliece, "The generalized distributive law," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 325–343, Mar. 2000.
[31] A. Y. Kitaev, "Fault-tolerant quantum computation by anyons," *Ann. Phys.*, vol. 303, no. 1, pp. 2–30, Jan. 2003.
[32] G. Smith and J. A. Smolin, "Degenerate quantum codes for Pauli channels," *Phys. Rev. Lett.*, vol. 98, p. 030501, Jan. 2007.
[33] P. W. Shor, "Scheme for reducing decoherence in quantum computer memory," *Phys. Rev. A*, vol. 52, no. 4, pp. R2493–R2496, Oct. 1995.
[34] J. Napp and J. Preskill, "Optimal Bacon-Shor codes," *Quantum Inf. Comput.*, vol. 13, nos. 5–6, pp. 490–510, May 2013.
[35] K. B. Petersen and M. S. Pedersen, *The Matrix Cookbook*. Kongens Lyngby, Denmark: Technical Univ. Denmark, 2012.
[36] A. E. Ashikhmin, A. M. Barg, E. Knill, and S. N. Litsyn, "Quantum error detection II: Bounds," *IEEE Trans. Inf. Theory*, vol. 46, no. 3, pp. 789–800, May 2000.
[37] J. Haah and J. Preskill, "Logical-operator tradeoff for local quantum codes," *Phys. Rev. A*, vol. 86, p. 032308, Sep. 2012.
[38] O. Landon-Cardinal and D. Poulin, "Local topological order inhibits thermal stability in 2D," *Phys. Rev. Lett.*, vol. 110, p. 090502, Feb. 2013.
[39] N. E. Bonesteel and D. P. DiVincenzo, "Quantum circuits for measuring Levin-Wen operators," *Phys. Rev. B*, vol. 86, p. 165113, Oct. 2012.
[40] C. Brell, S. Burton, G. Dauphinais, S. Flammia, and D. Poulin, "Thermalization, error correction, and memory lifetime for ising anyon systems," *Phys. Rev. X*, vol. 4, no. 3, pp. 031058-1–031058-21, Sep. 2014. [Online]. Available: http://link.aps.org/doi/10.1103/PhysRevX.4.031058

Pavithran Iyer, biography not available at the time of publication.

David Poulin, biography not available at the time of publication.