IEEE Transactions on Quantum Engineering

Received 15 January 2025; revised 18 July 2025; accepted 29 July 2025; date of publication 4 August 2025; date of current version 2 September 2025.

Digital Object Identifier 10.1109/TQE.2025.3595778

Fault-Tolerant Noise Guessing Decoding of Quantum Random Codes

DIOGO CRUZ[1,2], FRANCISCO A. MONTEIRO[1,3] (Member, IEEE), ANDRÉ ROQUE[1,2], AND BRUNO C. COUTINHO[1,4]

[1]Instituto de Telecomunicações, 1049-001 Lisbon, Portugal
[2]Instituto Superior Técnico, Universidade de Lisboa, 1049-001 Lisbon, Portugal
[3]ISCTE—Instituto Universitário de Lisboa, 1649-026 Lisbon, Portugal
[4]Institute of Communications and Navigation, German Aerospace Center (DLR), 82234 Weßling, Germany

Corresponding author: Diogo Cruz (e-mail: diogo.cruz@lx.it.pt).

This work was supported in part by the European Union's Horizon 2020 Research and Innovation Program through the Project Quantum Internet Alliance (QIA) under Grant 820445, in part by the FCT/MECI through National Funds and when applicable co-funded EU funds under Grant UID/50008: Instituto de Telecomunicações, and in part by the FCT/MECI through project reference QuNetMed under Grant 2022.05558.PTDC, with DOI identifier 10.54499/2022.05558.PTDC. The work of Diogo Cruz was supported by FCT scholarship UI/BD/152301/2021.

ABSTRACT This work addresses the open question of implementing fault-tolerant quantum random linear codes (QRLCs) with feasible computational overhead. We present a new decoder for QRLCs capable of dealing with imperfect decoding operations. A first approach, introduced by Cruz et al. (2023), only considered channel errors and perfect gates at the decoder. Here, we analyze the fault-tolerant characteristics of QRLCs with a new noise guessing decoding technique, when considering preparation, measurement, and gate errors in the syndrome extraction procedure, while also accounting for error degeneracy. Our findings indicate a threshold error rate ($p_{\text{threshold}}$) of approximately 2×10^{-5} in the asymptotic limit, while considering realistic noise levels in the mentioned physical procedures.

INDEX TERMS Fault-tolerance, noise guessing decoding, quantum error correction, quantum random linear codes (QRLCs), syndrome extraction (SE).

I. INTRODUCTION

It is known that classical random linear codes (RLCs) are capacity-achieving [1]. However, until the advent of guessing random additive noise decoding (GRAND), their decoding was not practical, except for some decoders based on trellises (as pointed out in [2]). GRAND has been proposed with the aim of reducing end-to-end latency in coded wireless systems, which has been a drawback for a long time. The rationale in the original proposal of GRAND was that by using short codewords, the so-called interleavers, which are used to make the errors independent and identically distributed, would no longer be required [3]. Using short blocks in wireless systems also helps to better adapt to the channel variations when applying precoding techniques [4], [5], [6]. In the quantum realm, due to technical limitations in manipulating qubits, short block codes appear as natural candidates for quantum error correction codes (QECCs) [7], [8], [9], [10], [11]. These limitations also necessitate the development of fault-tolerant techniques to handle noise and errors in quantum operations [12].

Like classical RLCs, quantum RLCs (QRLCs) attain the capacity of the quantum channels, but no practical decoder existed for them until the advent of quantum GRAND (QGRAND), which allowed to numerically assess their performance for the first time [2]. A recent work also used a GRAND-like approach to decode several families of structured quantum codes that are based on stabilizer codes [13]. QGRAND has also been applied to the purification of quantum links, taking advantage of the connection between purification and error correction [14], which will have great implications on the way routing is implemented in quantum networks [15], [16], [17], [18].

QRLCs are a much more flexible solution than other structured quantum codes for QECCs, with advantages in respect to the state-of-the-art solutions designed to detect and correct errors in quantum setups [19], [20]. In contrast to structured codes, which may only exist for a very limited number of code rates and codeword lengths [21], [22], QRLCs can exist for a wide range of coding rates and codeword lengths that may better fit some particular applications. A method to generate QRLCs efficiently was proposed in [23]. However,

almost no practical method existed until recently to decode them until the proposal in [2].

The channel model used in that work was a Shannon-like channel, where errors occur only in the channel and all the decoding process is perfect. However, in all current technologies implementing qubits, the errors that take place in the quantum gates of the decoding circuit cannot be ignored. Hence, a practical challenge remained after [2]: can a QRLC-QGRAND system be made practical in the presence of the extra errors coming from the quantum gates, enabling fault-tolerant QECCs based on QRLCs?

This article shows that, surprisingly, due to the particular way that the syndrome extraction (SE) takes place in codes based on stabilizers, some heavy reduction of the effects of those errors takes place, making the whole system viable. Building on previous work [2], [14], we present a comprehensive analysis of fault-tolerant QRLCs, incorporating the effects of preparation, measurement, and gate errors. Our results show that QRLCs, decoded with the proposed method, exhibit robust error correction capabilities with a threshold error rate $p_{\text{threshold}}$ of approximately 2×10^{-5}. This advancement paves the way for practical implementations of QRLCs in quantum error correction, contributing to the development of scalable and resilient quantum systems.

While recent work by Nelson et al. [24] has addressed fault-tolerant quantum error correction using low-depth random circuit codes, our work focuses on QRLCs decoded with the QGRAND technique. Both approaches work with stabilizer codes generated through random constructions, but differ in their specific code constructions, decoding methods, and target applications. Their work addresses fault-tolerant state preparation and distillation protocols for quantum memory applications, while our approach focuses on fault-tolerant syndrome decoding for error correction.

Although our results suggest that QGRAND could in theory enable a fault-tolerant implementation of QRLCs, some challenges remain that limit its usefulness in that regime. QGRAND is most suitable for situations where the noise entropy is relatively low, in which case decoding becomes computationally efficient. However, in the fault-tolerant regime where n may be considered to be considerably large or it is necessary to iteratively apply error correction to suppress errors, the noise entropy can be considerably high. In this regime, the optimal procedure described in this article becomes infeasible, and suboptimal heuristics would have to be introduced. Nonetheless, this work paves the way for applications of QGRAND whenever the considered noise types all have low entropy, which encompasses setups with realistic noise conditions.

The rest of this article is organized as follows. In Section II, we introduce the setup considered in the analysis, and in particular its noise model. In Section III, we define some useful error notation terms and set the notation used throughout this article. Section IV presents the decoding method, extended form [2] to account for degenerate errors. In Section V, we present an analysis of the codes' performance for various qubit counts. Finally, Section VI concludes this article. To help the reader, a notation summary is listed in Table 1.

II. SETUP AND NOISE MODEL

We use the same setup as in [2], but consider the fault-tolerant regime, where the constituent quantum gates in the circuit may be affected by error. We consider an initial k-qubit $|\psi\rangle$ quantum state, to be encoded into $n > k$ qubits. Brown and Fawzi [23] presented a method of generating a random qubit encoding, which we use in this work. One starts by randomly selecting Clifford unitaries from the $\mathcal{C}_2$ group (i.e., Clifford unitaries for 2 qubits). There are $|\mathcal{C}_2| = 11\,520$ such unitaries, and all of them can be built by simple combinations of the Hadamard (H), phase ($\sqrt{Z}$), and CNOT gates, which have efficient physical implementations in virtually any quantum setting [25]. In matrix form, these are defined as

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad \sqrt{Z} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \quad \text{CNOT} = \begin{pmatrix} I & 0 \\ 0 & X \end{pmatrix} \tag{1}$$

with X, Y, and Z the Pauli matrices, and I the 2×2 identity matrix.

After selecting these random unitaries from $\mathcal{C}_2$, one successively applies each of them to a random pair of qubits, taken from the set of n qubits.

This process leads to an encoding unitary for our stabilizer code which, when applied to the initial k qubits and $(n-k)$ extra $|0\rangle$ qubits added, returns a n-qubit encoded quantum state. As shown in [23], as long as $\mathcal{O}(n \log^2 n)$ gates are used, with a circuit depth of $\mathcal{O}(\log^3 n)$, the construction leads to a highly performant (n, k) code, and from [26] it is already known that these complexity orders can be further lowered.

We use these QRLCs to construct stabilizer codes. Compared to the approach in [2], in this work, we consider a noise model that is more realistic by also including preparation, measurement, and gate errors. Given that, in practical applications, the error of 2-qubit entangling gates generally dominates over single-qubit gate errors [10], we focus on the former type of error. We further analyze the appropriateness of our model in Appendix L. We assume that every gate in both the encoding and SE steps is decomposed into the Clifford gates $\{\text{CNOT}, H, \sqrt{Z}\}$.

For the noise statistics, we consider the model similar to the one in [10], but without single-qubit gate errors (see Fig. 1).

1) CNOT *gate errors:* After the ideal implementation of the CNOT(a, b) gate, with qubit a controlling b, it is assumed that one of the 15 errors of the form

$$O_a O_b \tag{2}$$

IEEE Transactions on Quantum Engineering

TABLE 1. Notation Summary

Variable	Description	Relationships
U	Unitary encoding circuit for a quantum error-correcting code	
S_i	The i^{th} minimal stabilizer of the code ($1 \leq i \leq n-k$)	$S_i = UZ_{i+k}U^\dagger$
$\bar{X}_j$	Logical X operator on the j^{th} encoded qubit ($1 \leq j \leq k$)	$\bar{X}_j = UX_jU^\dagger$
$\bar{Z}_j$	Logical Z operator on the j^{th} encoded qubit ($1 \leq j \leq k$)	$\bar{Z}_j = UZ_jU^\dagger$
$\mathcal{N}$	List of all possible errors E_i in the noise model, with associated probabilities p_i	
E_i^B	A base error (error affecting a single qubit or gate)	
ω	The number of base errors that compose a compound error	$E_j = E_{i_1}^B \cdots E_{i_\omega}^B$
e_i^l	Local error pattern corresponding to a base error E_i^B	
C, B	Unitary components of a SE circuit, applied before/after an error E_i^B occurs	$V = CB$
V_E	SE circuit affected by error E_i^B	$V_E = Ce_i^lB$
e_i	Propagated error pattern on the main qubits after E_i^B and subsequent circuit operations	$e_i = (Ce_i^lC^\dagger)_m$
e_i^s	Propagated error pattern on the ancilla qubits after E_i^B and subsequent circuit operations	$e_i^s = (Ce_i^lC^\dagger)_a$
$\mathbf{A}$	Quantum check matrix of the code (binary representation in $[X\|Z]$ format)	
$\mathbf{e}_i$	Binary representation of the main error pattern e_i (in $[Z\|X]$ format)	
$\mathcal{L}$	Logical error group generated by $\bar{X}_j$ and $\bar{Z}_j$	
$\mathtt{L}_i$	One of the 2^{2k} logical error patterns in $\mathcal{L}$	$e_i = \mathtt{E}_i\mathtt{S}_i\mathtt{L}_i$
$\mathcal{S}$	Stabilizer group generated by S_j	
$\mathtt{S}_i$	One of the 2^{n-k} stabilizer patterns in $\mathcal{S}$	$e_i = \mathtt{E}_i\mathtt{S}_i\mathtt{L}_i$
$\mathtt{E}_i$	Error pattern with the same syndrome as e_i	$e_i = \mathtt{E}_i\mathtt{S}_i\mathtt{L}_i$
$\mathbf{0}$	Vector representing zero syndrome ($n-k$ zero bits)	
$\hat{\mathbf{s}}_i$	Syndrome associated with a (propagated) error pattern e_i	$\hat{\mathbf{s}}_i = \mathbf{e}_i\mathbf{A}^T$
$\mathcal{D}$	A degenerate set: A set of error patterns with the same syndrome that can be corrected similarly	
e_i^d	Representative of the errors in a degenerate set $\mathcal{D}$	$e_i^d = \mathtt{E}_i\mathtt{L}_i$
g	Index of the SE where an error E_i^B occurred	
$\tilde{\mathbf{s}}_i$	Syndrome acquired in the same extraction as error E_i^B	$\tilde{\mathbf{s}}_i = \text{comp}_X(\text{bin}(e_i^s))$
$\hat{\mathbf{s}}_i$	Syndrome acquired in a subsequent extraction after E_i^B occurred	$\hat{\mathbf{s}}_i = \text{comp}_X(\text{bin}((Ve_iV^\dagger)_a))$
$\mathbf{s}_i$	Measured syndrome in a particular SE	
s	List of all acquired syndromes over multiple extractions: $\{\mathbf{s}^1, \ldots, \mathbf{s}^q\}$	
$\mathcal{Q}(E_j)$	Syndrome sequence expected for a compound error E_j	$\mathcal{Q}(E_j) = s_{i_1} \oplus \cdots \oplus s_{i_\omega}$

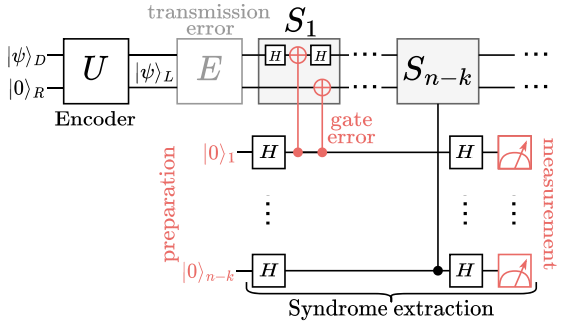


FIGURE 1. Noise model considered.

with $O_a, O_b \in \{I, X, Y, Z\}$ and excluding $O_aO_b \neq I_aI_b$, occurs with probability $p/15$. Here, I is the identity gate, and X, Y, and Z are the Pauli matrices.

2) *Preparation errors:* While setting the $(n-k)$ ancilla qubits (for each SE) to $|0\rangle$, each qubit has (independently) a probability p of being prepared in the state $|1\rangle$ instead.

3) *Measurement errors:* While measuring each ancilla qubit to extract the syndrome, each measurement bit has a probability p of being misread, so that a zero bit is read as a 1, and vice-versa.

Unlike the model in [2], to demonstrate the fault-tolerant properties of this model, we exclude a source of error between the encoding and SE sections (i.e., the "transmission error" in Fig. 1), and instead focus on the case where the CNOT gate error stemming from the SE dominates the noise statistics of the circuit. This simpler model facilitates the study of the QGRAND decoding approach in the fault-tolerant regime, which is the focus of this work. While possible (see Appendix A), we make no further modifications to the circuit implementation.

III. ERROR CORRECTION OVERVIEW

In the fault-tolerant regime with a noisy gate model, degenerate errors play a significant role in the error correction capabilities of the code [27]. As a result, the approximation made in [2], where codes were approximated

to be nondegenerate, is no longer accurate, as it would significantly underestimate the code's capabilities.

In addition, in the fault-tolerant regime, we must consider an iterated application of the SE procedure, instead of a single application, in order to capably detect the errors being introduced by the SE procedure itself. This is a common approach [19], [28] to quantum error correction when gate and measurement errors are nonnegligible, and the decoding procedure has into account not just the syndrome from one extraction process, but the whole history of syndrome measurements.

As a result of this added complexity, in this section we clarify the notation we use in this work. We use $\bar{X}_j, \bar{Z}_j$ to represent the logical operators corresponding to the unencoded operators X_j, Z_j, respectively. Given an encoding U (see Fig. 1), the choice of minimal stabilizers S_i and logical operators is not unique. Without loss of generality (W.l.o.g.), we consider the minimal stabilizer S_i (for $1 \leq i \leq n-k$) and logical operators $\bar{X}_j, \bar{Z}_j$ (for $1 \leq j \leq k$) to be given by

$$S_i = UZ_{i+k}U^\dagger \tag{3}$$

$$\bar{X}_j = UX_jU^\dagger \tag{4}$$

$$\bar{Z}_j = UZ_jU^\dagger. \tag{5}$$

Following Section II, the noise model enables us to create a list $\mathcal{N} = \{(p_0, E_0), (p_1, E_1), \ldots\}$ of all the errors E_i that the encoded quantum state may be subjected to, along with its, respectively, probability p_i of occurring. An error E_i refers to the qualitative process that occurred physically, such as "a X_2Z_3 error occurred in the CNOT(2,3) gate, and no other errors," for example.

An error E_i that corresponds to either only one wrongly prepared qubit, or one wrongly measured qubit, or one noisy CNOT gate, is called a *base error*, and may be explicitly labeled as E_i^B. Every other error in the noise model of Section II can be described as a combination of base errors.

We consider the errors E_i to be disjunctive, so only one error in $\mathcal{N}$ may occur, and their probability sums to 1. When using the base error notation E_i^B, we implicitly refer only to the specific base error that occurred, without making claims about the occurrence of other base errors. For example, using the noise model in Section II, the base error $E^B =$"X_2Z_3 error in the CNOT(2,3) gate" would have a probability of occurring of $p/15$, while the corresponding error $E =$"X_2Z_3 error in the CNOT(2,3) gate, and no other errors" would have a probability of $(p/15) \times P$(no other base error occurs), which would possibly be much lower. We may use the shorthand notation $\hat{E}^B := E$ for errors where only one base error occurs.

Compound errors may be represented by their own symbol or as the product of errors that compose it. That is, for simplicity, given base errors E_i^B and E_j^B, we also have the compound error notation

$$E_r = E_i^B \cap E_j^B \cap \left(\bigcap_{m \neq i,j} \overline{E_m^B} \right) =: E_i^B E_j^B. \tag{6}$$

An error E_j is said to be of order ω if ω base errors suffice to describe it, so that $E_j = E_{i_1}^B \cdots E_{i_\omega}^B$. Note that the constituting base errors may stem from different SE.

Given a base error, let e^l be the local error pattern corresponding to the error that occurred locally. In the example above, we would have $e_i^l = X_2Z_3$. In general, for gate errors affecting only one CNOT gate, we would have an error pattern from (2). For preparation and measurement errors, e^l would be represented by X operators in the appropriate ancilla qubits.

Unless the error E_i^B occurs at the end of a SE process, it will propagate through the rest of the quantum circuit, possibly impacting other qubits. Let V be the unitary corresponding to one noiseless SE process, minus the final measurement step of the ancilla qubits. We may partition V into the unitaries B and C, corresponding to the portion of the SE circuit that occurs before and after the error E_i^B, respectively. If the SE affected by E_i^B is given by the unitary V_E, we have

$$V = CB \tag{7}$$

$$V_E = Ce_i^lB = (e_i^s \otimes e_i)V. \tag{8}$$

The resulting (propagated) error pattern may affect nontrivially both the main n qubits (main *error pattern* e_i) and the $(n-k)$ ancilla qubits (*ancilla error pattern* e_i^s).

A Pauli string of n qubits is an operator that is the product of the Pauli operators X, Y, and Z on those qubits. It has the form

$$e^{i\frac{\pi}{2}\phi} O_1 O_2 \cdots O_n$$
$$\text{with } O_j \in \{X, Y, Z, I\} \text{ and } \phi \in \{0, 1, 2, 3\}. \tag{9}$$

As e_i^l is a Pauli string, and C is a Clifford unitary, then both e_i and e_i^s are Pauli strings. Similarly, following (3) to (5), the minimal stabilizers and logical operators are also Pauli strings, since U is a Clifford unitary as well. A Pauli string is said to have weight t if it acts on t qubits, that is, if its Pauli string contains t Pauli operators (excluding the identity).

For the previous example with E_i^B, we would have $e_i^s \otimes e_i = C(X_2Z_3)C^\dagger$. While the effect of e_i^s is removed by the syndrome measurement, the same cannot be said of e_i. As E_i^B propagates through the circuit, the pattern e_i is picked up by subsequent SE, and is ultimately the error pattern that our correction process needs to consider to undo the effect of E_i^B on the main n qubits. See Fig. 2 for an example.

In Fig. 2, we showcase a simple example, with two SE, where there is only one minimal stabilizer $S_1 = X_1X_2$. An error occurs in the first CNOT gate, so $E =$"Z_1I_2 error in first CNOT gate, and no other errors." The error is not detected by the first SE process, so $\mathbf{s} = \tilde{\mathbf{s}} = 0$. By the end of the first SE, the evolved uncorrected error is $e = Z_1I_2$. It is now detected by the second extraction, so $\mathbf{s} = \hat{\mathbf{s}} = 1$. If it is not corrected, subsequent extractions will behave similarly to the second one, returning the syndrome 1.

IEEE Transactions on Quantum Engineering

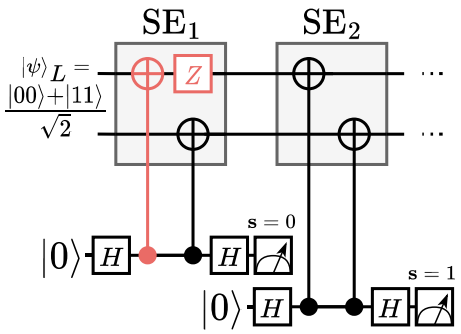


FIGURE 2. Simple example, with two SE. The error is not detected by the first SE process, but it is detected by the second extraction.

Since $Y = iXZ$, any Pauli string of n qubits can also be written in the form

$$e^{i\frac{\pi}{2}\varphi}\left(O_1^X \cdots O_n^X\right)\left(O_1^Z \cdots O_n^Z\right) \tag{10}$$

with $O_j^X \in \{X, I\}$, $O_j^Z \in \{Z, I\}$ and $\varphi \in \{0, 1, 2, 3\}$.

The Pauli string may then be encoded as a binary row vector. In $[X|Z]$ format, it takes the form

$$\left[b_1^X\ b_2^X\ \cdots\ b_n^X\ b_1^Z\ b_2^Z\ \cdots\ b_n^Z\right] \tag{11}$$

$$\text{with } b_j^P = \begin{cases} 1, & \text{if } O_j = P \\ 0, & \text{if } O_j = I \end{cases} \text{ and } P \in \{X, Z\}. \tag{12}$$

In $[Z|X]$ format, the b_j^X entries are swapped with b_j^Z. By default, binary vectors and matrices are represented in bold. We may use the functions

$$\text{bin}(e) := \mathbf{e}, \qquad \text{op}(\mathbf{e}) := e \tag{13}$$

$$\text{comp}_P(\mathbf{e}) := \left[b_1^P \cdots b_c^P\right] \tag{14}$$

with $P \in \{X, Z\}$ and $c = |\mathbf{e}|/2$, to indicate the conversion to and from binary representation, and to refer to a particular component of $\mathbf{e}$, respectively. Calculations using binary are performed in $\mathbb{F}_2$, that is, using modular arithmetic mod 2. The functions bin and op stand for the transformations of Pauli operators from and to, respectively, binary arrays.

Let $\mathbf{A}$ be the quantum check matrix [29], a $(n-k) \times 2n$ binary matrix (in $[X|Z]$ format) where each row j encodes the minimal stabilizer S_j of the code. This is a compact way of representing the encoding used. Let $\mathbf{e}_i$ be the binary representation of the error pattern e_i as a $2n$-sized row vector, in $[Z|X]$ format. Any evolved error pattern e_i can be written as

$$e_i = \mathtt{E}_i\mathtt{S}_i\mathtt{L}_i \tag{15}$$

where $\mathtt{L}_i$ is one of the 2^{2k} logical error patterns in the logical error group $\mathcal{L}$ generated by the logical operators $\bar{X}_j, \bar{Z}_j$, $1 \le j \le k$; $\mathtt{S}_i$ is one of the 2^{n-k} stabilizers in the stabilizer group $\mathcal{S}$ generated by the minimal stabilizers S_j, $1 \le j \le n-k$; and $\mathtt{E}_i$ is some error pattern with the same syndrome $\hat{\mathbf{s}} = \mathbf{e}_i\mathbf{A}^T$ as e_i [29]. We use $\mathbf{0}$ to represent the syndrome with all entries equal to zero. W.l.o.g., for the decomposition, we assume the phase factor φ [see (10)] to be zero, since neglecting it adds at most a global phase to the encoded quantum state, which can be disregarded. As a result, we consider each error pattern to equal its inverse.

The decomposition in (15) is not unique, and is dependant on the choice made for the particular logical operators, minimal stabilizers, and $\mathtt{E}_i$ patterns to use. For the sake of simplicity in the notation, in this work, it is assumed that such a decomposition is the unique one obtained deterministically by following the procedure described in Section IV. Consequently, we assume that, associated with each error pattern e_i, there is a unique set of operators $\mathtt{E}_i$, $\mathtt{S}_i$, and $\mathtt{L}_i$. In particular, for patterns with $\hat{\mathbf{s}} = \mathbf{0}$, the operator $\mathtt{E}_i$ is the identity. Since all error patterns with the same syndrome will have the same error component $\mathtt{E}$, we use $\mathtt{E}_{\hat{\mathbf{s}}}$ to indicate the error component of the error patterns with syndrome $\hat{\mathbf{s}}$.

Compound error patterns, such as $e_r = e_ie_j$, may be easily encoded in binary form by using the modular sum (i.e., XOR), so that $\mathbf{e}_r = \mathbf{e}_i \oplus \mathbf{e}_j$ and $\hat{\mathbf{s}}_r = \hat{\mathbf{s}}_i \oplus \hat{\mathbf{s}}_j$. Given two error patterns e_i, e_j with the same syndrome $\hat{\mathbf{s}}$, their product e_r has syndrome $\hat{\mathbf{s}}_k = \hat{\mathbf{s}} \oplus \hat{\mathbf{s}} = \mathbf{0}$, so $\mathtt{E}_r$ is the identity operator. Therefore, there is a unique $\mathtt{S} \in \mathcal{S}$, $\mathtt{L} \in \mathcal{L}$ such that $e_i = e_j\mathtt{S}\mathtt{L}$, with $\mathtt{S} = \mathtt{S}_i\mathtt{S}_j$ and $\mathtt{L} = \mathtt{L}_i\mathtt{L}_j$.

A degenerate set $\mathcal{D}$ is a set of evolved error patterns that can be treated similarly, for correction purposes. This set depends on $\hat{\mathbf{s}}$ and $\mathtt{L}$. Although all error patterns with the same syndrome $\hat{\mathbf{s}}$ have the same representative error pattern $\mathtt{E}_{\hat{\mathbf{s}}}$, not all can be corrected similarly. For that to be the case, their logical error component $\mathtt{L}$ must be the same. Since we know that two error patterns are degenerate if $e_ie_j \in \mathcal{S}$, we may verify this by computing $e_r = e_ie_j = (\mathtt{E}_i\mathtt{S}_i\mathtt{L}_i)(\mathtt{E}_j\mathtt{S}_j\mathtt{L}_j) = (\mathtt{E}_i\mathtt{E}_j)(\mathtt{S}_i\mathtt{S}_j)(\mathtt{L}_i\mathtt{L}_j) = \mathtt{E}_r\mathtt{S}_r\mathtt{L}_r$. For e_r to be in $\mathcal{S}$, we must have $\mathtt{E}_r = \mathtt{L}_r = I$, which is only the case if $\mathtt{E}_i = \mathtt{E}_j$ and $\mathtt{L}_i = \mathtt{L}_j$. The former is true if e_i and e_j have the same syndrome $\hat{\mathbf{s}}$, while the latter is more complicated to verify, but we know that there are only 2^{2k} possibilities in $\mathcal{L}$ to consider. Ultimately, we may index the degenerate sets based on their syndrome and logical error component, both represented by the tuple $(\hat{\mathbf{s}}, \mathtt{L})$. We use $e_i^d = \mathtt{E}_i\mathtt{L}_i$ to refer to the actual representative of e_i (and its degenerate equivalents) during the correction process, since we know that if we can correct e_i^d by applying the unitary $(e_i^d)^\dagger$ to the circuit, so can we indirectly also correct e_i, since $(e_i^d)^\dagger e_i = \mathtt{S}_i$, and stabilizers act as the identity on the encoded quantum state.

For an error E_i arising from SE g, there are two associated syndromes of interest, instead of one. The syndrome

$$\hat{\mathbf{s}}_i = \mathbf{e}_i\mathbf{A}^T \tag{16}$$

corresponds to the syndrome obtained from a subsequent noiseless SE after extraction g, that is, after the error has occurred. In this case, we can consider the error model to be similar to the one used in [2], where the (propagated) error pattern e_i is present before any stabilizer is applied for the SE process. Instead of using (16), we may alternatively compute

$\hat{\mathbf{s}}_i$ by first computing the ancilla error pattern. If we think of e_i as a different local error pattern e_j^l, then, following (8), we must have

$$Ve_iV^\dagger = Ve_j^lV^\dagger = e_j^s \otimes e_i \tag{17}$$

and $\hat{\mathbf{s}}_i$ is given by the X component [i.e., second half in $[Z|X]$ format; see (11)] of $\mathbf{e}_j^s$. This type of syndrome is always zero for preparation and measurement errors, since measurement errors do not affect subsequent extractions. Under this latter formalism, we may observe this by noting that if $e_i = I$, then necessarily $e_j^s = I$ and $\hat{\mathbf{s}}_i = \mathbf{0}$.

Beyond this typical syndrome, we also have the syndrome $\tilde{\mathbf{s}}_i$ obtained from the same extraction g where the error E_i^B occurred. For instance, if the error occurs at the last implemented CNOT gate, it is likely that $\tilde{\mathbf{s}}_i = 0$. Unlike $\hat{\mathbf{s}}_i$, this syndrome is nonzero for simple measurement errors. In general, this syndrome contains less information than $\hat{\mathbf{s}}_i$, since, for errors later in the extraction, many of the syndrome bits will be zero, as the stabilizers were applied before the error occurred. Following the second approach previously presented to compute $\hat{\mathbf{s}}_i$ [see (17)], we have that $\tilde{\mathbf{s}}_i$ is given by the X component of the original ancilla qubit pattern $\mathbf{e}_i^s$ (in binary form).

Both $\tilde{\mathbf{s}}$ and $\hat{\mathbf{s}}$ refer to a $(n-k)$ bit string, corresponding to the syndrome that could be obtained from a single SE. As both these syndromes can be deterministically obtained from the error E of interest, we use the simple notation $\mathbf{s}$ to specifically refer to the syndrome that is measured during the SE. As previously stated, for errors stemming from only one extraction (labeled g), we have either $\mathbf{s} = \tilde{\mathbf{s}}$ (if the measured syndrome comes from the noisy extraction), $\mathbf{s} = \mathbf{0}$ (if it comes from a previous extraction) or $\mathbf{s} = \hat{\mathbf{s}}$ (if from a subsequent extraction).

In general, compound errors may stem from multiple SE. We use superscript notation to indicate the extraction index, in order to distinguish it from the error index (which is a subscript). When there are q SE, we refer to the total list of measured syndromes by $s := \{\mathbf{s}^1, \ldots, \mathbf{s}^q\}$. If some base error $E_i^{B,g}$ occurs at extraction g, we expect to measure the syndrome sequence given by

$$s_i = \left\{\ldots, \mathbf{0}^{g-1}, \tilde{\mathbf{s}}_i^g, \hat{\mathbf{s}}_i^{g+1}, \ldots, \hat{\mathbf{s}}_i^q\right\}. \tag{18}$$

Note that, for compound errors, the syndrome sequences of the constituting errors may be combined. If $E_r = E_iE_j$, then $s_r = s_i \oplus s_j$, where the modular sum operation is applied element-wise, to all q syndromes. Then, for $E_j = E_{i_1}^B \cdots E_{i_\omega}^B$, we have

$$\mathcal{Q}(E_j) := s_j = s_{i_1} \oplus \cdots \oplus s_{i_\omega}. \tag{19}$$

In particular, if errors E_i and E_j occur at extractions g and h $(h > g)$, respectively, and no other errors occur, then we would expect to measure the syndrome sequence

$$s = \left\{\ldots, \mathbf{0}^{g-1}, \tilde{\mathbf{s}}_i^g, \hat{\mathbf{s}}_i^{g+1}, \ldots,\right.$$

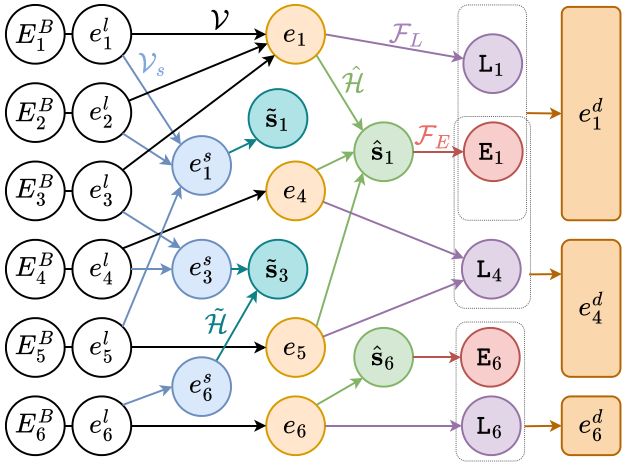


FIGURE 3. Relations between the different quantities of interest. For simplicity, every error represented is assumed to be a base error. The notation is explained in the last paragraph of Section III.

$$\hat{\mathbf{s}}_i^{h-1}, \hat{\mathbf{s}}_i^h \oplus \tilde{\mathbf{s}}_j^h, \hat{\mathbf{s}}_i^{h+1} \oplus \hat{\mathbf{s}}_j^{h+1}, \ldots\Big\}. \tag{20}$$

As a result, we observe that $\tilde{\mathbf{s}}_i, \tilde{\mathbf{s}}_j, \hat{\mathbf{s}}_i,$ and $\hat{\mathbf{s}}_j$ do not directly provide the full information necessary to identify the compound error $E_i^g E_j^h$ that occurred. In the general case where there are q SE, we may require all q measured syndromes to optimally correct errors.

For compound errors E_r stemming from multiple SE, the syndrome $\tilde{\mathbf{s}}_r$ is undefined, but $\hat{\mathbf{s}}_r$ may still be defined as

$$\hat{\mathbf{s}}_r = \bigoplus_{j=1}^{\omega} \hat{\mathbf{s}}_{i_j} \tag{21}$$

where $\hat{\mathbf{s}}_{i_j}$ are the syndromes of the constituting base errors. Similarly, $e_r = e_{i_1} \cdots e_{i_\omega}$. The decomposition in (15) and subsequent analysis is also applicable.

The notation is summarized in Fig. 3, with an example given in Fig. 2. The mappings $\mathcal{V}_s$, $\mathcal{V}$, $\hat{\mathcal{H}}$, $\tilde{\mathcal{H}}$, $\mathcal{F}_E$, and $\mathcal{F}_L$ are mostly independent, and generally noninjective. $\mathcal{V}$ and $\mathcal{V}_s$ stem from (8), and provide the main and ancilla error patterns e_i and e_i^s. Concretely, we have $e_i = \mathcal{V}(e_i^l) = (Ce_i^lC^\dagger)_m$ and $e_i^s = \mathcal{V}_s(e_i^l) = (Ce_i^lC^\dagger)_a$, where m and a stand for the main and ancilla subspaces, respectively. Using (17) yields $\tilde{\mathcal{H}}$ and $\hat{\mathcal{H}}$. The latter can also be implemented by using (16). Concretely, we have $\hat{\mathbf{s}}_i = \hat{\mathcal{H}}(e_i) = \text{bin}(e_i)\mathbf{A}^T = \text{comp}_X(\text{bin}((Ve_iV^\dagger)_a))$ and $\tilde{\mathbf{s}}_i = \tilde{\mathcal{H}}(e_i^s) = \text{comp}_X(\text{bin}(e_i^s))$ [see (13) and (14)]. The $\hat{\mathbf{s}}_i$ and $\mathbf{s}_i$ obtained by the SE process can then be used to try and determine e_i^d, the representative pattern of the degenerate set to which e_i belongs. By applying e_i^d to the noisy quantum state, we correct the effect of the error E_i. The mappings $\mathcal{F}_E$ and $\mathcal{F}_L$ are quite involved, so their description is delegated to Section IV.

Algorithm 1: Optimal decoding.

Require: $\mathcal{N}$
Ensure: A decoding table T
Initialize empty decoding table T
$D \leftarrow \textsc{Data}(\mathcal{N})$
for all entry s in D **do**
    Set $T[s]$ as the pattern e^d with highest p in $D[s]$
end for

Algorithm 2: Error processing.

Require: $\mathcal{N}$
Ensure: A data table D
Initialize empty data table D
for all $(p_i, E_i) \in \mathcal{N}$ **do**
    Compute e_i, $\hat{\mathbf{s}}_i$, and s_i
    Compute $\mathtt{L}_i$ associated with e_i
    Compute e_i^d
    if e_i^d not in any entry in $D[s_i]$ **then**
        Store (e_i^d, p_i) in $D[s_i]$
    else
        Add p_i to p in entry (e_i^d, p)
    end if
end for

IV. DECODING

The error pattern statistics given by the noise model of Section II lead to a very high number of degenerate error patterns (see Section B for examples). As a result, the approximation made in [2], where codes were approximated to be non-degenerate, is no longer accurate, as it would significantly underestimate the code's capabilities. In this section, we modify the decoding procedure in [2] to account for error degeneracy. The modified procedure is optimal in principle. It has previously been shown that such optimal procedures must be #P-complete in general [30], [31], [32]. Since we are applying the decoding procedure to random codes with no exploitable structure, our decoding procedure has poor scaling capabilities for high entropy noise and large code sizes. Nonetheless, following [2], we hope to show it to be of interest in regimes of small code size or low entropy, so it is still worth exploring the decoding properties of this optimal procedure. It is also possible (though not covered in this work) for the decoding complexity to be greatly improved with simpler approximations and heuristics to the optimal approach. The optimal decoding procedure is summarily presented in Algorithm 1.

When considering this optimal decoding procedure, we note that, while we focus on the noise model in Fig. 1, the decoding procedure is naturally applicable to models where there are additional sources of error independent of the SE themselves. For that case, the noise statistics would simply include those additional errors.

Moreover, the decoding procedure presented in this section does incorporate any assumptions about the underlying nature of the noise, as it meant to be a fully general procedure. In particular, we do not wish to assume that higher order errors are less likely than lower order ones, as there may be practical regimes where particular high order errors dominate the noise statistics (such as burst errors). In Appendix H, we adapt the general decoding procedure to the particular noise model described in Section II.

For the decoding, we require a procedure that, given a syndrome sequence s, outputs the error pattern e^d that needs to be applied to the circuit to correct the most likely source of error. Since we are interested in the analysis of the decoding procedure in the optimal case, and less concerned about practical limitations, we assume that such a procedure corresponds to a decoding table T, storing the s, e^d pairs.

The decoding table T may be obtained as follows (see Algorithms 1 and 2 for pseudocode, and Fig. 8 in Appendix E for an example).

1) For each error E_i (with corresponding probability p_i), we compute its syndrome sequence s_i, and also e_i and $\hat{\mathbf{s}}_i$ (corresponding to the mapping $\mathcal{Q}$ in (19), and the mappings $\mathcal{V}$ and $\hat{\mathcal{H}} \circ \mathcal{V}$, respectively, in Fig. 3, for base errors). Since the circuit is a stabilizer circuit, it can be efficiently simulated [33], and these quantities efficiently computed.
2) We compute $\mathtt{E}_i = \mathcal{F}_E(\hat{\mathbf{s}}_i)$ and $\mathtt{L}_i = \mathcal{F}_L(e_i)$. As we already know $\hat{\mathbf{s}}_i$, we end up with the degenerate set $(\hat{\mathbf{s}}_i, \mathtt{L}_i)$ (and its representative $e_i^d = \mathtt{E}_i\mathtt{L}_i$), to which e_i belongs.
3) We repeat steps 1 and 2 for all errors in $\mathcal{N}$. Starting with an empty data table D, for each error E_i, we add (e_i^d, p_i) to the entry $D(s_i)$. If an entry with e_i^d already exists, we add p_i to the entry's probability. This procedure results in the data table D.
4) For each syndrome sequence $s = \{\mathbf{s}^1, \ldots, \mathbf{s}^q\}$ in D, we choose the degenerate set with the highest associated probability as the actual coset leader, that is, the one that is corrected if s is measured.
5) The resulting syndrome table T then acts as our decoding method.

This decoding is optimal because, for any given syndrome, there is no other way for the decoding to be more successful than as described here, since this method already picks the most probable degenerate set $(\hat{\mathbf{s}}, \mathtt{L})$, given the only information available a priori, which is $\mathcal{N}$ and the observed syndrome $\mathbf{s}$. It is optimal under the reasoning that we consider any unsuccessful correction to be a complete failure, with no possible partial success.

While the procedure as described is done in series, iteratively traversing the E_i errors, it can be trivially parallel, by splitting the error list across multiple parallel workers. (See Appendix E for a full description, including the pseudocode for the parallel implementation, in Algorithm 4.)

To implement the decoding procedure (in particular step 2), a priori, we require the efficient implementation of two functions as follows.

1) A function $\mathcal{F}_E : \hat{\mathbf{s}} \mapsto \mathtt{E}_{\hat{\mathbf{s}}}$ that, for a given code, and taking a syndrome $\hat{\mathbf{s}}$ as input, outputs a deterministic error pattern $\mathtt{E}_{\hat{\mathbf{s}}}$ that can act as a coset leader for the syndrome $\hat{\mathbf{s}}$. That is, any error pattern e_i with syndrome $\hat{\mathbf{s}}$ can be decomposed [following (15)] using the error component $\mathtt{E}_{\hat{\mathbf{s}}}$. Having access to this function considerably reduces the required serialized processing for the decoding, and the required memory, as we do not need to keep track of tentative coset leaders as we iterate through the errors E_i, and we can be sure that different parallel workers have the coset leader in the same degenerate set (in fact, we can be sure that they are equal).
2) A function $\mathcal{F}_L : e_i \mapsto \mathtt{L}_i$ that, for a given code, and taking an error pattern e_i as input, deterministically outputs the logical component $\mathtt{L}_i$ of the degenerate set to which this error pattern belongs to.

These functions are described in detail in Sections IV-A and IV-B.

A. FUNCTION $\mathcal{F}_E$

When analyzing the code, instead of working with the $(n-k)$ minimal stabilizers $\{S_i\}$ we extracted from the encoding U [see (3)], we work with a different set $\{S_{i,\text{rre}}\}$. Each stabilizer in this new set can be thought of as some combination of the stabilizers in $\{S_i\}$. To be more specific, considering that S_i corresponds to row i of the quantum check matrix $\mathbf{A}$ (with size $(n-k)\times 2n$), $\{S_{i,\text{rre}}\}$ corresponds to row i of $\mathbf{A}$ in reduced row echelon form (also known as canonical form). We can convert $\mathbf{A}$ to reduced row echelon form because products of stabilizers are still stabilizers. Since we are in $\mathbb{F}_2$, adding or subtracting rows of $\mathbf{A}$ is equivalent to multiplying stabilizers. As long as the resulting matrix is full rank (which is always the case, since the procedure for converting to reduced row echelon form preserves rank), the resulting new matrix $\mathbf{A}_{\text{rre}}$ encodes a new set of minimal stabilizers, $\{S_{i,\text{rre}}\}$, in its rows.

In practice, we can imagine that the measured syndrome $\mathbf{s}$ gets converted to the "reduced row echelon" syndrome $\mathbf{s}_{\text{rre}}$, which can be done with a $(n-k)\times(n-k)$ matrix that encodes the steps needed to convert $\mathbf{A}$ to reduced row echelon form. Let this matrix be $\mathbf{J}$. We have $\mathbf{A}_{\text{rre}} = \mathbf{J}\mathbf{A}$ and $\mathbf{s}_{\text{rre}} = \mathbf{J}\mathbf{s}$.

Working with $\mathbf{A}_{\text{rre}}$, let h_i be the index of the pivot of row i (it is not guaranteed that $h_i = i$, since the pivots may not all be along the main diagonal of $\mathbf{A}_{\text{rre}}$). Since $\{S_{i,\text{rre}}\}$ comes from reduced row echelon form, the stabilizer $S_{i,\text{rre}}$ will be the only minimal stabilizer with a nonzero entry at index h_i. Then, if $1 \le h_i \le n$, the error Z_{h_i} necessarily yields the syndrome bit 1 for $S_{i,\text{rre}}$ and zero for all other minimal stabilizers in $\{S_{i,\text{rre}}\}$. If $n+1 \le h_i \le 2n$, the error X_{h_i-n} necessarily yields the syndrome bit 1 for $S_{i,\text{rre}}$ and zero for all other minimal stabilizers. Consequently, we can use these $(n-k)$ errors as a basis to construct a deterministic error pattern $\mathtt{E}_{\hat{\mathbf{s}}}$ for every syndrome $\hat{\mathbf{s}}$. Let $\mathtt{E}_i$ be the error associated with $S_{i,\text{rre}}$ ($\mathtt{E}_i$ equals Z_{h_i} or X_{h_i-n}, as described). Since the code is linear, for any syndrome $\mathbf{s}_{\text{rre}}$, if $i_1, \ldots, i_k$ are the indices where the syndrome $\mathbf{s}_{\text{rre}}$ is 1, then the compound error $e = \mathtt{E}_{i_1} \ldots \mathtt{E}_{i_k}$ must necessarily have syndrome $\mathbf{s}_{\text{rre}}$. In other words, $\mathbf{s}_{\text{rre}} = \mathbf{e}\mathbf{A}_{\text{rre}}{}^T$.

Since $e = \mathtt{E}_{\mathbf{s}_{\text{rre}}}$ when using the minimal stabilizers $\{S_{i,\text{rre}}\}$, and $\mathbf{J}$ is a linear transformation, then we must have $e = \mathtt{E}_{\hat{\mathbf{s}}}$ when using the minimal stabilizers $\{S_i\}$. Therefore, if the code has stabilizers $\{S_i\}$, then $\mathcal{F}_E$ implements the procedure

$$\hat{\mathbf{s}} \Rightarrow \mathbf{s}_{\text{rre}} \Rightarrow e = \mathtt{E}_{\hat{\mathbf{s}}}. \tag{22}$$

Because of the construction of e, for all error patterns with the same syndrome, the same error component is computed. The error $\mathtt{E}_{\hat{\mathbf{s}}}$ then acts as the error component for all error patterns with syndrome $\hat{\mathbf{s}}$, for the decoding.

B. FUNCTION $\mathcal{F}_L$

For the function $\mathcal{F}_L$, we need to consider the different degenerate sets. We know that each error e_i can be decomposed in terms of the coset leader $\mathtt{E}_i$, some stabilizer $\mathtt{S}_i$, and some logical operator $\mathtt{L}_i$ [see (15)]. There are 4^k logical operators (including the identity) and each identifies one of the 4^k degenerate sets associated with each syndrome $\hat{\mathbf{s}}$.

Consequently, we can create a one-to-one mapping between the logical operators and the degenerate sets. Since we can determine the tentative coset leader $\mathtt{E}_i$ with $\mathcal{F}_E$, we only need to determine the logical component $\mathtt{L}_i$ that composes our input error pattern.

With this in mind, we continue the approach from Section IV-A. We use the row echelon form of $\mathbf{A}$, that is, $\mathbf{A}_{\text{rre}}$, so we work with $\{S_{i,\text{rre}}\}$ instead. We perform the same procedure to the logical operators. Let $\bar{\mathbf{X}}_i$ and $\bar{\mathbf{Z}}_i$ be the binary row vector representations of $\bar{X}_i$ and $\bar{Z}_i$, respectively [see (4) and (5)], in $[X|Z]$ format. Let $\mathbf{L}$ be the $2k\times 2n$ binary matrix that encodes the original $2k$ minimal logical operators (see (4) and (5)), that act as generators to the 4^k total logical operators. Row i of $\mathbf{L}$ is given either by $\bar{\mathbf{X}}_i$, if $1 \le i \le k$, or by $\bar{\mathbf{Z}}_{i-k}$, if $k+1 \le i \le 2k$.

Just as with the stabilizers, we know that products of logical operators are also logical operators. Moreover, products of a logical operator with stabilizers correspond to the same logical operator. Since we are working in $\mathbb{F}_2$, adding or subtracting rows of $\mathbf{A}_{\text{rre}}$ or $\mathbf{L}$ is equivalent to multiplying stabilizers and operators in Pauli string form. Considering the augmented $(n+k)\times 2n$ matrix $\begin{bmatrix}\mathbf{A}_{\text{rre}}\\ \mathbf{L}\end{bmatrix}$, by using the rows of $\mathbf{A}_{\text{rre}}$ and $\mathbf{L}$, we may put the $\mathbf{L}$ component in its row echelon form, $\mathbf{L}_{\text{re}}$. We can then put $\mathbf{L}_{\text{re}}$ in reduced row echelon form using only the rows of $\mathbf{L}_{\text{re}}$, yielding $\mathbf{L}_{\text{rre}}$. Note that we cannot use the rows in $\mathbf{L}_{\text{re}}$ to further simplify $\mathbf{A}_{\text{rre}}$, as the resulting rows would no longer correspond to stabilizers. The procedure may be represented as

$$\begin{bmatrix}\mathbf{A}_{\text{rre}}\\ \mathbf{L}_{\text{rre}}\end{bmatrix} = \begin{bmatrix}I_{n-k} & 0\\ 0 & \mathbf{J}_L\end{bmatrix}\begin{bmatrix}\mathbf{A}_{\text{rre}}\\ \mathbf{L}_{\text{re}}\end{bmatrix} \tag{23}$$

$$= \begin{bmatrix}I_{n-k} & 0\\ 0 & \mathbf{J}_L\end{bmatrix}\begin{bmatrix}I_{n-k} & 0\\ \mathbf{J}_A & I_{2k}\end{bmatrix}\begin{bmatrix}\mathbf{A}_{\text{rre}}\\ \mathbf{L}\end{bmatrix} \tag{24}$$

with I_p a $p \times p$ identity matrix. The matrices $\mathbf{J}_A$ and $\mathbf{J}_L$ are of size $2k \times (n-k)$ and $2k \times 2k$, respectively, and just like $\mathbf{J}$ in Section IV-A, they represent the linear transformation required to put the matrix in reduced row echelon form.

The resulting rows of $\mathbf{L}_{\text{rre}}$ correspond to $2k$ (possibly different) generators for the logical operators. These generators may no longer satisfy the anti-commutation relations expected of $\bar{X}$ and $\bar{Z}$, but they are not required to. The final $\mathbf{L}_{\text{rre}}$ matrix is such that the columns with the same index as the pivots of $\mathbf{A}_{\text{rre}}$ are zero, and the columns with the pivots of $\mathbf{L}_{\text{rre}}$ have only one nonzero element, its pivot.

For every error pattern e_i with syndrome $\mathbf{s}_i$, we know that its error component $\mathtt{E}_i$ (given by $\mathcal{F}_E$) is such that $e_i\mathtt{E}_i = \mathtt{S}_i\mathtt{L}_i =: e'_i$, for some unknown $\mathtt{S}_i$ and $\mathtt{L}_i$. Consequently, to determine the degenerate set to which e_i belongs, we only need to decompose e'_i into its $\mathtt{S}_i$ and $\mathtt{L}_i$ components. Concretely, we are looking for the unique row vectors $\mathbf{u}_A$ (of size $n-k$) and $\mathbf{u}_L$ (of size $2k$) such that

$$\mathbf{e}'_{i,[X|Z]} = \begin{bmatrix}\mathbf{u}_A & \mathbf{u}_L\end{bmatrix}\begin{bmatrix}\mathbf{A}_{\text{rre}} \\ \mathbf{L}_{\text{rre}}\end{bmatrix} \tag{25}$$

where $\mathbf{e}'_i$ is exceptionally in $[X|Z]$ format. Since $\mathbf{A}_{\text{rre}}$ and $\mathbf{L}_{\text{rre}}$ are already in reduced row echelon form, finding the two vectors is straightforward. The procedure is described in Algorithm 3. Once the $\mathbf{u}_A$ and $\mathbf{u}_L$ row vectors are determined, the $\mathtt{S}_i$ and $\mathtt{L}_i$ components are simply given by [see (13)]

$$\mathtt{S}_i = \text{op}(\mathbf{u}_A\mathbf{A}_{\text{rre}}) \tag{26}$$

$$\mathtt{L}_i = \text{op}(\mathbf{u}_L\mathbf{L}_{\text{rre}}). \tag{27}$$

Alternatively, we may simply skip the computation of $\mathbf{u}_L$ in Algorithm 3. Let $\mathbf{v}'$ equal the computed row vector $\mathbf{v}$ just after $\mathbf{u}_A$ is computed, but before the iteration through the pivots of $\mathbf{L}_{\text{rre}}$. Then, we equivalently have $\mathtt{L}_i = \text{op}(\mathbf{v}')$.

The full procedure

$$e_i \Rightarrow e_i\mathtt{E}_i \Rightarrow \mathbf{u}_L \text{ (or } \mathbf{v}') \Rightarrow \mathtt{L}_i \tag{28}$$

corresponds to the function $\mathcal{F}_L$.

Algorithm 3: Finding logical component.

Require: $\boldsymbol{e}'_i, \boldsymbol{A}_{\text{rre}}, \boldsymbol{L}_{\text{rre}}$
Ensure: Vectors $\boldsymbol{u_A}, \boldsymbol{u_L}, \boldsymbol{v}'$
Initialize $\boldsymbol{u_A}, \boldsymbol{u_L}$ to zero vectors
Let $\boldsymbol{v} \leftarrow \boldsymbol{e}'_i$
Let H_A (respectively, H_L) be an ordered list of pivot positions of $\boldsymbol{A}_{\text{rre}}$ (respectively, $\boldsymbol{L}_{\text{rre}}$)
for all h_i in H_A **do**
  if entry h_i of $\boldsymbol{v}$ is 1 **then**
    Subtract row i of $\boldsymbol{A}_{\text{rre}}$ from $\boldsymbol{v}$ (mod 2)
    $[\boldsymbol{u_A}]_i \leftarrow 1$
  end if
end for
Let $\boldsymbol{v}' \leftarrow \boldsymbol{v}$
for all l_i in H_L **do**
  if entry l_i of $\boldsymbol{v}$ is 1 **then**
    Subtract row i of $\boldsymbol{L}_{\text{rre}}$ from $\boldsymbol{v}$ (mod 2)
    $[\boldsymbol{u_L}]_i \leftarrow 1$
  end if
end for
return $\boldsymbol{u_A}, \boldsymbol{u_L}$
(if the computation was correct, then $\boldsymbol{v}$ should equal $\mathbf{0}$)

V. ASYMPTOTIC REGIME

We can estimate the optimal performance we can obtain from the decoding procedure by looking at how it performs as the number of extractions considered is increased. We are interested in computing the limit where we have infinite extractions, where the decoding would be optimal. Although this regime is impossible to attain in practice, we expect that, as we increase the number of extractions, the decoding dynamics should converge to the asymptotic corresponding to that optimal case, allowing us to estimate the code's performance in that regime.

We consider the total probability of correction failure P_{total} to correspond to the probability that an error is not completely corrected. That is, the correction chosen does not correspond to the right degenerate set. Note that this definition provides a lower bound on the fidelity F of the resulting quantum state, given by

$$F \geq 1 - P_{\text{total}} \tag{29}$$

since it effective treats any unsuccessful correction as producing a state with zero fidelity, whereas in practice the uncorrected error may not produce an orthogonal quantum state. Nonetheless, it is a useful lower bound often used in the literature [2], [10], and that we choose to use here as well.

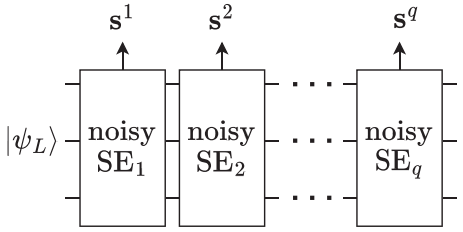


FIGURE 4. Model considered to compute the asymptotic regime, where $q \to \infty$.

Let P_∞ be the asymptotic limit of P_{total} when the number of SE q goes to infinity (see Fig. 4). Since there are $L = 2^{2k}$ degenerate sets associated to each syndrome sequence s, then, regardless of the encoding used, for a given s, the probability p of an error having occurred that is in the most likely degenerate set satisfies $p \geq 1/L$. If e_i^c is the error pattern representative of the most likely degenerate set for s_i, then we must have

$$P_\infty = 1 - \sum_i p(e_i^c e_i \in \mathcal{S}|E_i)p(E_i) \tag{30}$$

$$\leq 1 - \sum_i (1/L) p(E_i) \tag{31}$$

$$= 1 - 1/L \tag{32}$$

$$= 1 - 2^{-2k} \tag{33}$$

with equality achieved for the case of maximum noise entropy.

When comparing two setups, A and B, with q_A and $q_B = q_A + h$ ($h > 0$) SE, respectively, we necessarily have

$$P_{\text{total}}(A) \leq P_{\text{total}}(B) \tag{34}$$

since introducing h additional noisy syndromes extractions introduces additional errors in the model, which may or may not be correctable.

We are interested in determining the failure probability P_{failure} induced by a single additional SE, preceded and succeeded by an arbitrarily high number of extractions. In the context of fault-tolerance, we wish to determine if, for a given p, P_{failure} increases or decreases with increasing qubit count n. We expect, for low (respectively, high) values of p, P_{failure} decreases (respectively, increases) with n, with a phase transition at some $p_{\text{threshold}}$, to be determined.

Since we do not have direct access to P_{failure}, it must be computed from the measured value of P_{total}. We develop an effective model to quantitatively relate the two quantities.

Consider a variant of the B setup above, labeled B', where the first q_A extractions are solely used to identify and correct errors stemming from implementing those extractions, and similarly, the last h extractions are solely used to deal with errors resulting from the h extractions, for a total of $q_B = q_A + h$, as before. In other words, in the B' setup, the procedure is partitioned into to separate and independent decoding processes.

We expect

$$P_{\text{total}}(B) \leq P_{\text{total}}(B'). \tag{35}$$

Since, in the B setup, the last h extractions also provide information about errors in the previous extractions, which can lead to a more successful decoding. The B' setup does not use this information.

In fact, if an error occurs at extraction q_A, we expect that a lower number h of subsequent extractions will result in higher failure rates (per extraction), since the decoding process has less information to correctly identify the error. Since we are interested in the limiting case where the number of subsequent extractions is infinite (for any given extraction where an error can occur), we wish to discount the effect of limited syndrome information from the calculated probability of failure P_{failure}. We label the errors that could have been successfully corrected with $h \to \infty$ but were not with low h escaped errors.

For low q, we expect (see [28]) that escaped errors (in particular errors with $\tilde{\mathbf{s}} = 0$ but $\hat{\mathbf{s}} \neq 0$) can be identified with additional syndromes from more subsequent extractions, so that each new extraction reduces the number of escaped errors by a factor δ. Of course, new extractions also introduce new escaped errors, and errors at the end extractions are more likely to escape correction.

Consider a setup with q extractions as a setup with $q-1$ extractions preceded by one additional extraction. The errors of this additional extraction will be detected by the $q-1$ subsequent extractions, so that only a factor δ^{q-1} escape through the whole setup incorrectly identified. Therefore, applying the reasoning recursively for q extractions, we expect the probability of an error passing through the extractions undetected to be given by

$$P_u(p, q) \simeq P_u(p, q-1) + \delta^{q-1} P_u(p, 1) \tag{36}$$

$$\simeq P_u(p, 1) \frac{1 - \delta^q}{1 - \delta}. \tag{37}$$

For $q \gg 1$, we expect that the probability of successful correction with q extractions $P_{\text{succ}}(p, q)$ will be approximately the probability of success with $q-1$ extractions times the per-extraction probability of success (which accounts for the additional extraction). We have

$$P_{\text{succ}}(p, q) \simeq P_{\text{succ}}(p, q-1)(1 - P_{\text{failure}}(p)) \tag{38}$$

$$= (1 - P_{\text{failure}}(p))^q \tag{39}$$

$$\text{with } P_{\text{succ}}(p, 0) = 1. \tag{40}$$

However, since P_{failure} does not incorporate escaped errors, the true probability of success P'_{succ} is given by

$$P'_{\text{succ}}(p, q) = P_{\text{succ}}(p, q)(1 - P_u(p, q)) \tag{41}$$

yielding

$$P_{\text{total}}(p, q) \simeq 1 - (1 - P_{\text{failure}}(p))^q (1 - P_u(p, q)). \tag{42}$$

If $\delta \ll 1$, which is expected, then, when using $q \gg 1$ total extractions, we may approximate $P_u(p, q) \simeq P_u(p, \infty)$, leading to

$$P_{\text{total}}(p, q) \simeq 1 - C(p)(1 - P_{\text{failure}}(p))^q \tag{43}$$

$$\text{with } C(p) := 1 - P_u(p, \infty) \tag{44}$$

where C incorporates the escaped errors. See Fig. 5 for a numerical verification of this model. Once more, note that we discount these errors, and their probability P_u, from the failure probability P_{failure}, since we are interested in the per-extraction failure probability, and P_u constitutes a global effect. As previously indicated, if a SE is followed by $h \to \infty$ extractions, then the probability that an error stemming from that extraction goes completely undetected is $\delta^h P_u(p, 1) \to 0$.

P_{failure} is the asymptotic contribution of each SE to the probability of failure. We have that, by fitting, for each p

$$Y = mX + b \tag{45}$$

$$\text{with } Y := \log(1 - P_{\text{total}}(p, q)), \quad X := q. \tag{46}$$

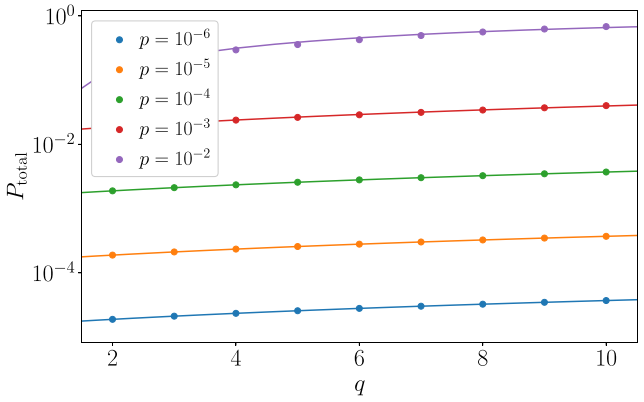


FIGURE 5. Experimental P_{total}, and resulting fit by (43). We observe the experimental data matching the qualitative description of the theoretical analysis, with $R^2 > 0.999$ for all cases.

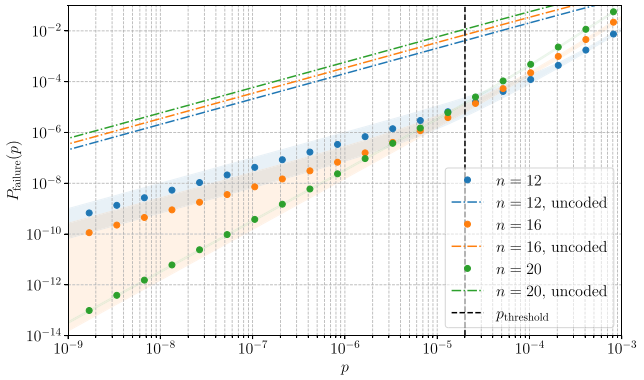


FIGURE 6. Performance of QRLCs with $k = 1$, using fault-tolerant QGRAND. We observe that the asymptotic version presents a threshold around $p \simeq 2 \times 10^{-5}$. The shaded regions denote a 60% confidence interval (from 50 samples), and the dots are the median performance observed. The vertical line indicates the $p_{\text{threshold}}$ value. The dashed lines at the top correspond to the uncoded case, which is plotted for reference.

Then, from (43)

$$P_{\text{failure}}(p) = 1 - e^m \tag{47}$$

$$C(p) = e^b \tag{48}$$

for each p considered.

The results are are presented in Fig. 6. We observe $p_{\text{threshold}} \simeq 2 \times 10^{-5}$, suggesting the QGRAND technique can be used in the fault-tolerant regime. Nonetheless, we note that this optimal decoding procedure is not scalable for high entropy noise models, such as those given by $p \gtrsim p_{\text{threshold}}$. Therefore, to use these techniques, we are forced to either simplify our model, turning the decoding procedure suboptimal, or to focus on a regime where it can be applied in practice.

For reference, we also estimate the performance of the uncoded case. Considering the number of noisy CNOT gates present in a circuit with q iterations to be $N_{\text{CNOT}}(q)$, we get

$$P_{\text{failure, uncoded}}(p) \simeq 1 - (1-p)^{N_{\text{CNOT}}(1)} \tag{49}$$

where N_{CNOT} does not include preparation and measurement errors, as these do not propagate throughout the circuit.

The simulation is performed for $\omega < \omega_{\max}$, and approximated for the higher ω values. For high ω, we use the reasonably accurate assumption that error patterns are uniformly assigned to the syndrome sequences. Therefore, before computing P_{failure}, we modify P_{total} with the correction

$$P'_{\text{total}} = P_{\text{total}} - \frac{P(\omega > \omega_{\max})}{2^{n+k}} \tag{50}$$

where 2^{n+k} is the number of degenerate sets associated with each syndrome sequence. The correction is negligible for high n, but may play a noticeable role for $p \simeq 1$. Based on our analysis in Appendix K, we find that for the n values considered, the minimum error order that nontrivially contributes to the failure probability never exceeds 2. This justifies using $\omega_{\max} = 2$ for exact simulation in the parameter regimes studied. The effect of higher orders can be reasonably accounted for using (50).

VI. DISCUSSION AND CONCLUSION

In this work, we extend the decoding procedure for QRLCs introduced in [2] to explicitly account for error degeneracy. Consequently, our technique constitutes a maximum likelihood decoding procedure, which is guaranteed to be optimal. We analyze the fault-tolerant characteristics of QRLCs with the presented decoding technique, by accounting for preparation, measurement, and gate errors in the SE procedure itself, and observe a $p_{\text{threshold}} \simeq 2 \times 10^{-5}$ in the asymptotic limit. To the best of our knowledge, this work presents the first fault-tolerant decoding technique specifically applying QGRAND to QRLCs in the presence of preparation, measurement, and gate errors during SE.

We note that this decoding procedure is not equivalent to finding the lowest weight error pattern associated with each syndrome, as might be done by more standard algorithms, since a faulty CNOT gate error effect can propagate considerably through the circuit before being possible to detect it, so that, by the time it is detected, its error pattern is no longer low weight.

In this work, we have removed the channel errors present in [2] and considered only the main error sources associated with the SE process. In particular, we considered preparation and measurement errors in the ancilla qubits, and two-qubit gate errors. Although this is a common approach to take when studying the fault-tolerance capabilities of different codes [10], it leads to unrealistic results for high code rates. In the limit when $R \to 1$, the SE process has a negligible number of minimal stabilizers, and as a result negligible error sources, under this noise model. Consequently, in this regime, higher code rates lead to lower P_{failure}, not because of better correction capabilities, but because error sources

decrease faster than the correction capabilities do, as the code rate increases.

Moreover, we have analyzed the asymptotic regime of infinite SE. Although impractical, these asymptotic results enable us to study the behavior of the optimal decoding procedure, as previously described. Nonetheless, practical limitations might impose suboptimal steps in the decoding process, and obviously a finite number of SE.

To account for these limitations, we must consider that, in practice, there are nontrivial computing steps performed between SE (such as logical gates for quantum computing, and Bell-pair creation for quantum communication) that introduce their own errors independently from the SE steps. When accounting for this additional error source, we expect the pathological behavior for high code rates to disappear. In future work, we intend to study these more practical regimes. Furthermore, we assumed that all-to-all connectivity (between any of the n qubits) is possible in practice. This assumption is required for the scaling results in [23], and is used in this work. Nonetheless, it may be dropped for practical reasons, as the more recent results in [26] suggest.

As previously mentioned, the noise guessing decoding procedure is expected to be viable only in situations of low noise entropy and low n. Even disregarding the limitations imposed by the asymptotically large number of SE, it is also the case that the noise entropy increases rapidly as $n \to \infty$ and $k = 1$, as considered for the fault-tolerance analysis. This is a known limitation of the decoding procedure. For this reason, and for the fact that better known codes, such as surface codes, have higher $p_{\text{threshold}}$ values, we do not expect the decoding procedure described in this work to be competitive in those regimes. Although it remains to be confirmed in future work, we conjecture that, given the optimal decoding properties of the described procedure, it may be worthwhile to employ in scenarios where code versatility is needed, the noise statistics are not approximately fixed, and the code rate is desired to be very high. In those cases, we expect the method to have similar use cases to those previously described in [2], as the additional gates used by the SE process would not have a strong impact on decoding performance.

Beyond the straightforward approach described in Algorithm 1, we may also wish to sacrifice the decoding optimality for the sake of decoding throughput, or lower hardware requirements, rendering the decoding process more easily scalable. This can be achieved with either known techniques, such as compressive sensing or deep learning, or with more straightforward approaches, such as greedy variants of the method. For instance, for the Bernoulli noise model in this work, if we take the coset leader to be the first error pattern associated with a syndrome, we will end up with a suboptimal version of Algorithm 1, equivalent to [2], but one which reduces the memory requirements by as much as a factor of 4^k. There are also specific simplifications that can be used to implement the decoding procedure faster when the noise model has some exploitable structure, as is the case with the noise model considered here. We plan to cover some of these approaches in future work.

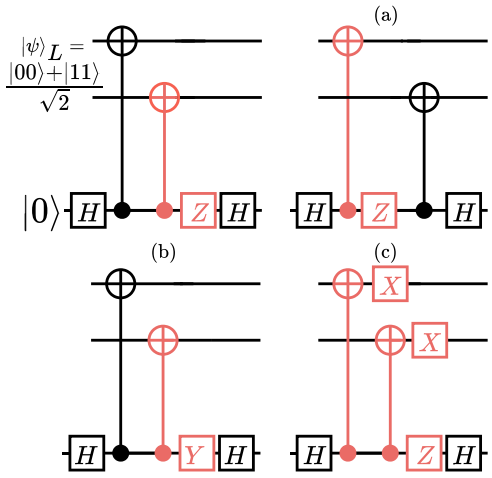


FIGURE 7. For the example code in Fig. 2, we may observe the three forms of degenerate errors. Considering E_i as the error showcased in the top left circuit, the identical, pseudoidentical, and nonidentical errors can be seen in (a), (b), and (c), respectively.

APPENDIX A
REDUCING STABILIZER WEIGHT

Although we are working with QRLCs, which have little exploitable structure a priori, we note that the minimal stabilizers can be efficiently chosen to have weight lower than the average of $3n/4$. To do so, we may take the original minimal stabilizer arising from the technique described in [2], represent them with the parity check matrix, and put the matrix in canonical form, which is equivalent to reduced row echelon form. The new simpler minimal stabilizers correspond to the rows of the resulting matrix.

If the pivots of the matrix in reduced row echelon form are all in the first n columns, then this technique reduces the weight of the non-Z components of each minimal stabilizer to at most $1 + k$, and $1 + k/2$ on average. If k is low and n is large, this technique can result in a considerable reduction in the weight of the minimal stabilizers, as their average weight goes from $3n/4$ to $3k/4 + (n - k - 1)/2 + 1 = (2n + k + 2)/4$. Instead of each term being equally distributed between I, X, Y, and Z as before, here only the indices greater than $n - k$ maintain that distribution, and we have, for S_i, index i equally distributed between X and Y, and index $j \leq n - k$, $j \neq i$ equally distributed between I and Z.

This structure simplifies the application of the SE process, as it reduces the number of CNOT gates from $\sim 3n(n - k)/4$ to $(2n + k + 2)(n - k)/4$, and similarly reduces the number of 1-qubit gates. Despite these benefits, in our numerical analysis we have not assumed such an approach was taken in the circuit implementation, in order not to introduce unwarranted structure in the noise statistics, as we are interested in analyzing the more general scenario.

Nonetheless, as explained in Sections IV-A and IV-B, we have used this simplification in our decoding implementation, when given the noise statistics associated with the unsimplified stabilizers. As they generate the same stabilizer group, they are mathematically equivalent for the same given noise statistics, and we expect either approach to lead to similar numerical results.

APPENDIX B
DEGENERACY ANALYSIS

Regarding degeneracy, following the notation introduced in Section III, we consider errors E_i and E_j to be degenerate if and only if $e_i e_j \in \mathcal{S}$. Nonetheless, we may describe three types of degenerate errors, all prevalent in the noise model considered in this work.

a) Identical errors: These are errors such that $e_i = e_j$ and $e_i^s = e_j^s$. For example, since we are considering the model implementation where the CNOT gates of the conditional stabilizer are implemented in succession, with the control always being the ancilla qubit (see Fig. 1), then any error of the form $Z_c I_t$ (with c and t the control and target qubit indices, respectively) will commute with subsequent CNOT gates controlled by the qubit c. Therefore, this component does not add any error terms to the main n qubits, and instead simply negates the measured ancilla qubit. As a result, for any error term without this component, there is an error term with this component where the error pattern in the main n qubits is the same, and the ancilla qubit is simplify negated. Given this degeneracy, the problem reduces to two scenarios: one where there is an even number of such errors, where the syndrome is unaffected, and one with an odd number of such errors, where the ancilla bit is negated. Among these two classes, all errors are not only degenerate, but identical.

b) Pseudoidentical errors: Besides the identical errors, we also observe cases where $e_i = e_j$ but $e_i^s \neq e_j^s$ (with $\mathrm{comp}_X(e_i^s e_j^s) = \mathbf{0}$, otherwise the errors would have different syndromes).

c) Nonidentical errors: We also have the more general case where $e_i \neq e_j$ (with either $e_i^s = e_j^s$ or $e_i^s \neq e_j^s$), while still retaining $e_i e_j \in \mathcal{S}$.

See Fig. 7 for an example. There, the code's sole nontrivial stabilizer is X_1X_2. We have $e_i^s = e_a^s = a_c^s = X$, $e_b^s = Y$ (ignoring the phase), $e_i = e_a = a_b = I_1I_2$, and $e_c = X_1X_2$.

APPENDIX C
ANALYTICAL THRESHOLD WITHOUT DEGENERACY

For this analysis, we disregard preparation and measurement errors, as the derivation can be readily extended to the complete model.

Keeping k constant, the value of n will determine the number of CNOT gates (N_{CNOT}) in the circuit, and the code's correction capabilities. We may estimate its performance by considering the approximation given by random ideal codes, as in [2].

For $S =: 2^{n-k} \gg 1$ and N the number of distinct errors, the equation

$$f = \frac{S}{N+1}\left[1 - \left(1 - \frac{1}{S}\right)^{N+1}\right] \tag{51}$$

may be approximated by

$$f \simeq \frac{1 - e^{-r}}{r} \qquad r := \frac{N+1}{S} \tag{52}$$

$$\Rightarrow f \simeq \frac{1 - (1 - r + r^2/2)}{r} = 1 - \frac{r}{2} \text{ for } r \ll 1. \tag{53}$$

Since, from (51), we have

$$f = \frac{1}{r}\left[1 - \left[\left(1 - \frac{1}{S}\right)^S\right]^r\right] \tag{54}$$

$$\simeq \frac{1}{r}\left[1 - \left(e^{-1}\right)^r\right]. \tag{55}$$

Since $S = 2^{n-k}$, this indicates that

$$f \geq 1 - \epsilon \tag{56}$$

$$\iff N \lesssim 2^{n-k+1}\epsilon. \tag{57}$$

W.l.o.g., consider that each CNOT gate can only suffer from a specific error, instead of 15. For the Bernoulli noise model we are considering, the error order ω is given by the binomial distribution

$$\omega \sim \mathcal{B}(N_{\mathrm{CNOT}}, p). \tag{58}$$

For p fixed, and as $N_{\mathrm{CNOT}} \to \infty$, this distribution can be approximated by

$$\mathcal{N}(N_{\mathrm{CNOT}}p, N_{\mathrm{CNOT}}p(1-p)) \tag{59}$$

using the De Moivre–Laplace theorem.

Suppose we start by correcting the lowest order errors (which are more likely to occur), and we wish to correct the errors up to order $\omega_{\max}$ such that we have the probability $(1-\epsilon)$ of correcting an error we observe. For a normal distribution, this is given by the quantile function

$$Q(p) = \mu + \sigma\sqrt{2}\mathrm{erf}^{-1}(2p-1). \tag{60}$$

The error function erf cannot be easily approximately. Nonetheless, we may observe that it can approximated by

$$\mathrm{erf}(x) \simeq 1 - \alpha\exp(-(x-\beta)^2), \text{ for } x \gg 1 \tag{61}$$

leading to

$$\mathrm{erf}^{-1}(x) \simeq \beta + \sqrt{\log\left(\frac{\alpha}{1-x}\right)} \text{ for } 1 - x \ll 1. \tag{62}$$

For simplicity, we consider $\alpha = 1, \beta = 0$, which does not meaningfully affect our conclusions here. The quantile function then becomes

$$Q(1-\epsilon) \simeq \mu + \sigma\sqrt{2}\left(\beta + \sqrt{\log\left(\frac{\alpha}{2\epsilon}\right)}\right) \tag{63}$$

$$\simeq \mu + \sigma\sqrt{2\log\left(\frac{1}{2\epsilon}\right)} \tag{64}$$

with

$$\mu = N_{\text{CNOT}}p \tag{65}$$

$$\sigma = \sqrt{N_{\text{CNOT}}p(1-p)} \tag{66}$$

$$\simeq \sqrt{N_{\text{CNOT}}p}. \tag{67}$$

Now that we have $\omega_{\max} = Q(1-\epsilon)$, we need to estimate the number of errors $\tilde{N}$ up to order $\omega_{\max}$, given by

$$\tilde{N} = \sum_{j=0}^{\omega_{\max}} \binom{N_{\text{CNOT}}}{j}. \tag{68}$$

Unfortunately, there is no closed form expression for this value. However, for $\omega_{\max} \ll N_{\text{CNOT}}$, this is roughly equal to

$$\tilde{N} \simeq \binom{N_{\text{CNOT}}}{\omega_{\max}} \tag{69}$$

$$\simeq \frac{{N_{\text{CNOT}}}^{N_{\text{CNOT}}}}{{\omega_{\max}}^{\omega_{\max}}(N_{\text{CNOT}}-\omega_{\max})^{N_{\text{CNOT}}-\omega_{\max}}} \times \sqrt{\frac{N_{\text{CNOT}}}{2\pi\omega_{\max}(N_{\text{CNOT}}-\omega_{\max})}}. \tag{70}$$

This may be simplified down to

$$\tilde{N} = 2^{N_{\text{CNOT}}h_2(\omega_{\max}/N_{\text{CNOT}})}\sqrt{\frac{1}{2\pi\omega_{\max}}} \tag{71}$$

where h_2 is the Shannon entropy.

From (57) and (71), we therefore conclude that

$$\log N \sim \tilde{O}(n) \tag{72}$$

$$\log \tilde{N} \sim \tilde{O}(N_{\text{CNOT}}) \tag{73}$$

where $\tilde{O}$ denotes big-O notation up to log factors. Since N indicates the code's correction capabilities, while $\tilde{N}$ indicates the necessary number of errors that the code needs to correct to preserve P_{failure}, then we must have $N \gtrsim \tilde{N}$ and consequently $N_{\text{CNOT}} \sim \mathcal{O}(n)$. This is verified for some common codes, such as surface codes, but for our implementation we have

$$N_{\text{CNOT}} = \frac{3}{4}n(n-k) \sim \mathcal{O}\left(n^2\right) \tag{74}$$

so we conclude that, if all errors are nondegenerate, the code does not have a visible $p_{\text{threshold}}$, since an increase in n increases P_{failure}, regardless of p.

However, we actually observe a threshold for QRLCs. This is thanks to the fact that the noise statistics given by the noise model of Section II actually lead to a very high number of degenerate errors. Therefore, in practice, the number of distinct errors grows with $\mathcal{O}(n)$ and not $\mathcal{O}(n^2)$ as indicated by the analysis above. We later confirm this scaling for escaped errors, in Appendix K.

Algorithm 4: Parallel decoding.

Require: $\mathcal{N}, W$
Ensure: A decoding table T
Initialize empty data table D and decoding table T
Split $\mathcal{N}$ into W sets (almost) equal in size, labeled $\mathcal{N}_w$ $(1 \le w \le W)$
for all w parallel workers **do**
    Initialize empty data table D_w
    $D_w \leftarrow \textsc{Data}(\mathcal{N}_w)$
end for
$D \leftarrow \bigcup_w D_w$
for all entry s in D **do**
    Set $T[s]$ as the pattern e^d with highest p in $D[s]$
end for
return T

With this insight in mind, we modify the QGRAND algorithm in [2] to account for the possibility of degenerate errors. The modified algorithm is presented in Section IV.

APPENDIX D
APPLYING ERROR CORRECTION

Certain QECCs, such as surface codes, are designed so that a physical error correction is actually unnecessary to implement, as all changes can be made in software, classically [10]. If the whole quantum circuit is unitary, then this procedure can actually be implemented in general: instead of correcting the error, we leave the affected state as is and simply XOR any subsequent syndrome $\mathbf{s}$ with the identified error syndrome $\mathbf{e}$, that is, $\mathbf{s} \mapsto \mathbf{s} \oplus \mathbf{e}$. As a result of this, we only need to keep track of these detected errors classically, in order to correct the subsequent syndromes.

In any case, since the correction portion is always single-qubit gates, we assume that their contribution to the total error is negligible, so we can disregard this trick for now, for the sake of simplicity. As a result, the procedure to apply the error correction is the same as in [2].

APPENDIX E
PARALLEL DECODING

The decoding procedure described in Section IV can be performed in parallel. If there are W parallel workers available, we start by splitting the entries in $\mathcal{N}$ into W parts of equal size, labeling each $\mathcal{N}_w$, $1 \le w \le W$. Each set $\mathcal{N}_w$ is then processed independently by an individual procedure according to the procedure described in the Section IV, thereby yielding the data table D_w.

All the W data tables D_w may then be merged to generate the full data table D, from which the decoding table T can be straightforwardly computed. See Algorithm 4 for a description of the parallel decoding procedure.

IEEE Transactions on Quantum Engineering

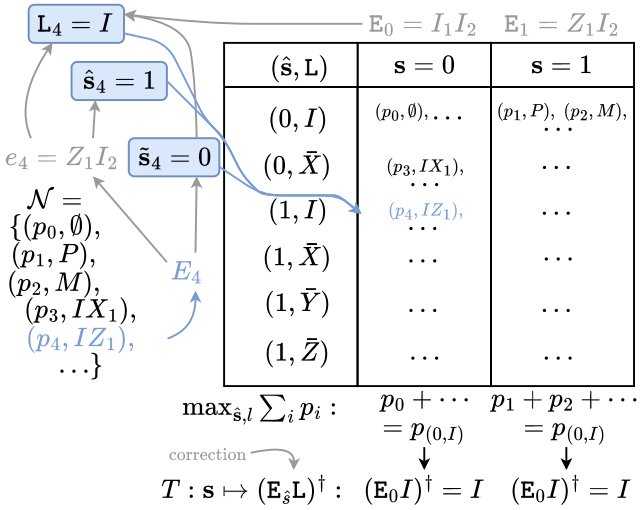


FIGURE 8. Optimal decoding of the code in Fig. 2 (but with only one SE), as described in Algorithm 1.

APPENDIX F
FULL DECODING EXAMPLE

For the example in Fig. 8, we consider the encoding gate to be $U = \text{CNOT}(2, 1)H_2$, with the starting qubit at index 1. P and M stand for preparation and measurement error in the only ancilla qubit a, respectively, AB_j) is the error corresponding to "error A_aB_j occurred at CNOT gate j." Following Algorithm 1, we iterate through the errors in $\mathcal{N}$ and, using the functions $\mathcal{F}_E$ and $\mathcal{F}_L$, determine where to put them in the table D. Once we have iterated through all errors, we compute the optimal $(\hat{\mathbf{s}}, \mathtt{L})$ entry, or alternatively, e^d representative, and delete the remaining entries, yielding the decoding table T.

Given U, and following (3) to (5), the minimal stabilizer is $S_1 = X_1X_2$, and the logical operators are $\bar{X} = I_1X_2$ and $\bar{Z} = Z_1Z_2$. This choice of encoding leads to minimal stabilizers and logical operators such that the augmented matrix is

$$\begin{bmatrix} \mathbf{A} \\ \mathbf{L} \end{bmatrix} = \begin{bmatrix} \mathbf{A} \\ \bar{\mathbf{X}} \\ \bar{\mathbf{Z}} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{\text{rre}} \\ \mathbf{L}_{\text{rre}} \end{bmatrix}. \tag{75}$$

Since, in this simple example, we already have $\mathbf{A} = \mathbf{A}_{\text{rre}}$ and $\mathbf{L} = \mathbf{L}_{\text{rre}}$, so $\mathbf{J} = I$. Here, $\mathbf{A}$, $\bar{\mathbf{X}}$, and $\bar{\mathbf{Z}}$ are the binary representations of the minimal stabilizers and logical operators, in $[\,X|Z\,]$ format.

Following the procedure in Section IV-A, we have

$$\mathtt{E}_0 = I_1I_2 \qquad \mathtt{E}_1 = Z_1I_2. \tag{76}$$

Let us consider the full procedure in Section IV applied to the specific error in Fig. 2, such as

$$E = \text{"Error } I_aZ_1 \text{ in CNOT gate 1."} \tag{77}$$

Associated to this error, we have the quantities

$$\tilde{\mathbf{s}} = 0 \tag{78}$$

TABLE 2. Decoding for the Code in Fig. 2

$(\hat{\mathbf{s}}_{\text{rre}}, \mathtt{L})$	e^d	$\mathbf{s} = 0$	$\mathbf{s} = 1$
$(0, I)$	I_1I_2	$\emptyset, XI_2, XX_1$	$P, M, ZI_2, ZI_1, YX_1, YI_2$
$(0, \bar{X})$	I_1X_2	IX_1, XI_1, IX_2, XX_2	ZX_1, YI_1, ZX_2, YX_2
$(0, \bar{Y})$	Z_1Y_2		
$(0, \bar{Z})$	Z_1Z_2		
$(1, I)$	Z_1I_2	IZ_1, XY_1	ZZ_1, YY_1
$(1, \bar{X})$	Z_1X_2	XZ_1, IY_1	YZ_1, ZY_1
$(1, \bar{Y})$	Z_1Y_2	IY_2, XY_2	ZY_2, YY_2
$(1, \bar{Z})$	I_1Z_2	IZ_2, XZ_2	ZZ_2, YZ_2

TABLE 3. Final Decoding Table for Fig. 2

$\mathbf{s} = 0$	$\mathbf{s} = 1$
$(0, \bar{X})$	$(0, I)$
$\downarrow$	$\downarrow$
$\mathtt{E}_0\bar{X} = I_1X_2$	$\mathtt{E}_0 I = I_1I_2$

$$e = Z_1I_2 \tag{79}$$

$$\mathbf{e} = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \tag{80}$$

$$\hat{\mathbf{s}} = \mathbf{e}\mathbf{A}^T = 1 \tag{81}$$

$$\hat{\mathbf{s}}_{\text{rre}} = \hat{\mathbf{s}} = 1 \tag{82}$$

$$e\mathtt{E}_{\hat{\mathbf{s}}} = I_1I_2 = \mathtt{S}\mathtt{L} \tag{83}$$

$$\Rightarrow \mathtt{L} = I_1I_2 = I. \tag{84}$$

Applying the procedure in Section IV and Algorithm 1, we obtain the tentative decoding table in Table 2. In the table, for the sake of simplicity, we represent preparation and measurement errors by P and M, respectively. AB_i represents the CNOT gate error where error A_aB_i occurs just after the noiseless CNOT gate. For simplicity, compound error are not shown.

The preparation and measurement errors have probability p of occurring, and the gate errors have probability $p/15$. If we assume that it is very unlikely that no error has occurred (i.e., p is high), then the final decoding table (when only considering errors of order 1) is given by Table 3. In the table, if $\mathbf{s} = 0$ is measured, the most likely degenerate set to have occurred is $(0, \bar{X})$, with probability $4p/15$ (see Table II). We assume that p is such that the no-error case is unlikely, otherwise $(0, \bar{I})$ would be the most likely degenerate set. If $\mathbf{s} = 1$ is measured, the most likely degenerate set to have occurred is $(0, \bar{I})$, with probability $34p/15$. For each case, the error pattern that should be applied to the quantum state to correct the error is given by $e^d = \mathtt{E}_{\hat{\mathbf{s}}}\mathtt{L}$.

APPENDIX G
SIMPLIFIED NOISE STATISTICS

When considering a Bernoulli noise model, such as in Appendix H, including preparation and measurement errors into the noise model breaks some of the structure of the noise statistics, since not all base errors will be equally likely. It also makes the decoding process harder to simulate. In the simpler setup where only gate errors occur, if we have g

CNOT gates affected with an error, with each error having probability $p_{\text{CNOT}}/15$ of occurring, then the number of errors and their individual probability would be given by

$$N_E(g) = 15^g \binom{N_{\text{CNOT}}}{g} \tag{85}$$

$$P(g) = \left(\frac{p_{\text{CNOT}}}{15}\right)^g (1 - p_{\text{CNOT}})^{N_{\text{CNOT}}-g}. \tag{86}$$

These formulas stem from the fact that each CNOT gate has 15 associated errors, and only one of these may occur at a time. For preparation and measurement errors, there is only one error pattern per ancilla qubit: either there is a bit flip, or there is not. If there are A ancilla qubits (generally, $A = n - k$), and a preparation or measurement error occurs with probability p_M, these same quantities are given by

$$N_E(\omega) = \sum_{g=0}^{\omega} 15^g \binom{N_{\text{CNOT}}}{g} \binom{2A}{\omega - g} \tag{87}$$

$$P(\omega, g) = \left(\frac{p_{\text{CNOT}}}{15}\right)^g (1 - p_{\text{CNOT}})^{N_{\text{CNOT}}-g} \tag{88}$$

$$\times\, p_M^{\omega-g}(1 - p_M)^{2A-\omega+g} \tag{89}$$

where ω indicates the total error order and g indicates the error order when ignoring preparation and measurement errors. Note that these expressions are somewhat more complicated. In particular, for errors of order ω, we now need to keep track of the distinct number of CNOT (g) and preparation/measurement ($\omega - g$) base errors that occurred, instead of just one parameter.

Fortunately, if we are using optimal decoding (that properly accounts for degenerate errors), there is a quick-and-dirty way to mimic the simpler noise statistics associated with only having gate errors. If we use $p_M = p_{\text{CNOT}} =: p$ (which is a relatively common choice in the literature, see [10], and the one used in Section II), then we can consider that each preparation and measurement error is a CNOT gate error. Instead of there being only one error per qubit (corresponding to the possible bit flip), we consider that there are 15, all equal in nature, and each occurring with probability $p_M/15$. These cloned errors will be degenerate among themselves, so the optimal decoding procedure will analyze this setup correctly. The total number of error patterns $N_E(\omega)$ will be overcounted, but we never actually use it directly for the decoding procedure, so the overcounting does not constitute an issue. In this formulation, we may pretend that we have no preparation and measurement errors, and that we have instead

$$\tilde{N}_{\text{CNOT}} := N_{\text{CNOT}} + 2A \tag{90}$$

CNOT gates in the SE circuit. The probability associated with each error will be correct, yielding

$$P(\omega) = \left(\frac{p}{15}\right)^{\omega} (1 - p)^{\tilde{N}_{\text{CNOT}}-\omega}. \tag{91}$$

APPENDIX H
BERNOULLI NOISE MODEL IMPROVEMENTS

For the special case where each CNOT gate has the same probability p of suffering an error, as described in Section II, the decoding procedure can be made much more efficient. This procedure may be adapted to other Bernoulli-like noise statistics, but here we focus on this error model. We can optimize this decoding process in order to avoid having to iterate through all compound errors. Since the technique relies on the inherent structure of errors with the same probability, here we employ the reformulation detailed in Appendix G to treat preparation and measurement errors as additional gate errors.

The procedure for the list of base errors (with $\omega = 1$) is similar to the one described in the beginning of Section IV. Instead of using the full noise statistics $\mathcal{N}$, instead we consider a list of base errors

$$\mathcal{B} := \{E_1^B, E_2^B, \ldots\}. \tag{92}$$

Given the Bernoulli noise model, all base errors have the same associated probability of occurring, given by $p/15$, so it does not need to be stored in $\mathcal{B}$ (we exclude the no error case). Once we have the data table D_1 for $\omega = 1$, we can start to optimize the analysis for compound errors. Instead of iterating through compound errors individually, we iterate through the degenerate sets obtained in the $\omega = 1$ step. We also preserve the probability associated with observing an error pattern from each degenerate set in the table. For $\omega = 1$, and given the reformulation of Appendix G, all errors $\hat{E}_i^B$ may be considered to have a probability $P(1)$ of occurring [see (91)], so the probability associated with each degenerate set in the data table is given by

$$p_i := N_i P(1) \tag{93}$$

where N_i is the number of errors E_i that can be corrected by applying the coset leader e_i^d associated with the degenerate set of $\hat{E}_i^B$. In summary, we may restructure the data table D_1 obtained with Algorithm 1 (before the final degenerate set selection)

$$D_1 = \{s_1 : \{(e_{11}^d, p_{11}), (e_{12}^d, p_{12}), \ldots\},$$
$$s_2 : \{(e_{21}^d, p_{21}), (e_{22}^d, p_{22}), \ldots\}, \ldots\} \tag{94}$$

to encode the count N_i instead of p_i, and to store a list of error counts for different orders, yielding

$$\tilde{D}_1 = \{s_1 : \{e_{11}^d : \mathbf{n}_{11}, e_{12}^d : \mathbf{n}_{12}, \ldots\},$$
$$s_2 : \{e_{21}^d : \mathbf{n}_{21}, \ldots\}, \ldots\} \tag{95}$$

with

$$\mathbf{n}_i := (0, N_i, 0, \ldots) \tag{96}$$

a list of size $(\omega + 1)$, where only the second entry of the list (corresponding to $\omega = 1$) starts with nonzero entries. The first entry is only nonzero for the $E_0 = I$ case, when $\mathbf{s} = \hat{\mathbf{s}} = 0$ and $\mathrm{L} = I$, corresponding to the case where no error occurs. We represent the entry of order g by $\mathbf{n}_i(g)$.

IEEE Transactions on Quantum Engineering

Under the formulation of (95), instead of iterating through the combinations $\{E_1^B E_2^B, E_1^B E_3^B, \ldots, E_2^B E_3^B, \ldots\}$ as we could do with the naive implementation of Algorithm 1, we iterate through the degenerate sets in $\tilde{D}_1$ as a whole.

Consider the data table $\tilde{D}_{\omega-1}$ that includes the errors up to order $\omega-1$. To obtain the table for errors up to order ω, we iterate through the combination of the degenerate sets in $\tilde{D}_{\omega-1}$ and $\tilde{D}_1$. If the noise statistics are highly degenerate (which is generally the case following the noise model in Section II), we can have considerable computational savings, since we only need to perform $|\tilde{D}_{\omega-1}||\tilde{D}_1|$ computations instead of $15^{\omega}\binom{N_{\text{CNOT}}+2(n-k)}{\omega}$ (see Appendix G). While we expect the latter to grow quickly with $\mathcal{O}(n^{2\omega})$, the former approach should grow, at worst, with $\mathcal{O}(n^{\omega})$, and it may grow more slowly in practice.

With this approach we generally overcount the number of errors N_i associated with each degenerate set. There are the following three types of overcounting.

1) Counting permuted copies: Consider an order-$(\omega-1)$ error $E_{i_1}^B E_{i_2}^B \ldots E_{i_{\omega-1}}^B$ (with $i_1 < i_2 < i_3 < \ldots$), coming from $\tilde{D}_{\omega-1}$, and the error $\hat{E}_j^B$, coming from $\tilde{D}_1$. W.l.o.g., suppose $j < i_1$. Then, for $\tilde{D}_{\omega}$, we will not only count the error $E_j^B E_{i_1}^B E_{i_2}^B \ldots E_{i_{\omega-1}}^B$, but also that same error coming from the combination of the errors $E_j^B E_{i_1}^B E_{i_2}^B \ldots E_{i_{\omega-1}}^B \backslash E_{i_k}^B$ and $E_{i_k}^B$. In total, we overcount each order-ω error ω times.
2) Recounting lower order errors: for the error $E_{i_1}^B E_{i_2}^B \ldots E_{i_{\omega-1}}^B$, composing with any $E_{i_k}^B$ $(1 \le k \le \omega-1)$ reduces the error to one of order $\omega-2$, which was previously counted. Each order-$(\omega-2)$ error we counted before will be recounted $\zeta_{\omega-1,1}$ times, where ζ is given in (100).
3) Counting two errors occurring in the same CNOT gate: An error of order ω stems from base errors that occurred on ω CNOT gates. When composing this error with another, the resulting compound error may have more than one base error occurring at one or more CNOT gates. As this compound error is impossible, it should be discounted.

We can extend this approach further. Instead of constructing the data table in one order increments, if we already have $\tilde{D}_{\omega}$, we may combine it with itself to obtain $\tilde{D}_{2\omega}$, thereby requiring exponentially fewer iterations, as ω increases, as long as ω is such that the codes capabilities are not yet saturated, i.e., not all syndromes are assigned to a degenerate set.

In general, if we compute the noise statistics of errors of order a and b to compute those of order $\omega = a+b$, we have

$$\mathbf{n}_i(\omega) = \frac{1}{R_{a,b}}\left[\tilde{\mathbf{n}}_i(\omega) - \sum_{\substack{0<k+r\le b\\ k,r\ge 0}} \zeta_{a,b}(k,r)\mathbf{n}_i(c)\right] \tag{97}$$

Algorithm 5: Data table $\tilde{D}_1$ for Bernoulli noise.

Require: $\mathcal{B}$
Ensure: A data table $\tilde{D}_1$
Initialize empty data table $\tilde{D}_1$
$\boldsymbol{n_0} \leftarrow (1,0)$
Store $\{I : \boldsymbol{n_0}\}$ in $\tilde{D}_1[\{\mathbf{0},\ldots,\mathbf{0}\}]$
for all E_i^B in $\mathcal{B}$ **do**
  Compute e_i, $\hat{s}_i$, and s_i
  Compute $\mathcal{L}_i$ associated with e_i
  Compute e_i^d
  if e_i^d not in $\tilde{D}_1[s_i]$ **then**
    $\boldsymbol{n_i} \leftarrow (0,0)$
    $\boldsymbol{n_i}(1) \leftarrow 1$
    Store $\{e_i^d : \boldsymbol{n_i}\}$ in $\tilde{D}_1[s_i]$
  else
    $\boldsymbol{n_i}(1) \leftarrow \boldsymbol{n_i}(1) + 1$
    Update $\{e_i^d : \boldsymbol{n_i}\}$ in $\tilde{D}_1[s_i]$
  end if
end for
return $\tilde{D}_1$

Algorithm 6: Data table $\tilde{D}_{a+b}$.

Require: $\tilde{D}_a, \tilde{D}_b$
Ensure: A data table $\tilde{D}_{a+b}$
Initialize empty data table $\tilde{D}_{a+b}$
for all $(s_i, e_i^d, \boldsymbol{n_i})$ in $\tilde{D}_a$ **do**
  for all $(s_j, e_j^d, \boldsymbol{n_j})$ in $\tilde{D}_b$ **do**
    $\tilde{\boldsymbol{n}}_{ij} \leftarrow \boldsymbol{n_i} * \boldsymbol{n_j}$ *(convolution)*
    $\boldsymbol{n_{ij}} \leftarrow \tilde{\boldsymbol{n}}_{ij}$ with overcounting correction
    if $e_i^d e_j^d$ not in $\tilde{D}_{a+b}[s_i \oplus s_j]$ **then**
      Store $\{e_i^d e_j^d : \tilde{\boldsymbol{n}}_{ij}\}$ in $\tilde{D}_{a+b}[s_i \oplus s_j]$
    else
      Add $\tilde{\boldsymbol{n}}_{ij}$ to vector in $e_i^d e_j^d$ entry in $\tilde{D}_{a+b}[s_i \oplus s_j]$
    end if
  end for
end for
for all $(s_r, e_r^d, \boldsymbol{n_r})$ in $\tilde{D}_{a+b}$ **do**
  $\boldsymbol{n_r} \leftarrow \tilde{\boldsymbol{n}}_{\boldsymbol{r}}$ with overcounting correction
  Store $\{e_r^d : \boldsymbol{n_r}\}$ in $\tilde{D}_{a+b}[s_r]$
end for
return $\tilde{D}_{a+b}$

$$\text{with } c := a+b-2k-r \tag{98}$$

where $\tilde{\mathbf{n}}_i(\omega)$ is the count obtained for order ω before the overcounting correction. The auxiliary functions are given by

$$R_{a,b} := \binom{a+b}{a} \tag{99}$$

$$\zeta_{a,b}(k,r) := K^k(K-1)^r \xi_{a,b}(k,r) \tag{100}$$

Algorithm 7: Decoding table T_{a+b}.

Require: $\tilde{D}_\omega$
Ensure: A decoding table T_ω
1: Initialize empty decoding table T_ω
2: **for all** s in $\tilde{D}_\omega$ **do**
3: $(j, p) \leftarrow (-1, 0)$
4: **for all** $(e_i^d, \boldsymbol{n_i})$ in $\tilde{D}_\omega[s]$ **do**
5: Using $\boldsymbol{n_i}$, compute p_i ((102))
6: $(j, p) \leftarrow (i, p_i)$ if $p_i > p$
7: **end for**
8: Set $T[s]$ as the pattern e_j^d, which has the highest p in $D[s]$
9: **end for**
10: **return** T_ω

$$\xi_{a,b}(k, r) := \binom{\tilde{N}_{\text{CNOT}} - c}{k} \binom{c}{r, a-k-r, b-k-r} \tag{101}$$

where K is the number of distinct errors per CNOT gate (in our case, always 15). Note the use of binomial and multinomial coefficients. To incorporate the effect of preparation and measurement errors, we use $\tilde{N}_{\text{CNOT}} = N_{\text{CNOT}} + 2(n-k)$ and not N_{CNOT}, as explained in Appendix G. See Appendix I for a derivation of these expressions.

The probability associated with the degenerate set with list $\mathbf{n}_i$ is given by

$$p_i = \sum_{j=0}^{\omega} \mathbf{n}_i(j) P(j). \tag{102}$$

Given this procedure to obtain $\tilde{D}_\omega$, we may obtain the decoding table T_ω by following Algorithm 5 and 7.

APPENDIX I
DERIVATION OF DECODING FORMULAS FOR BERNOULLI NOISE

As stated in Appendix H, a straightforward implementation of the procedure described will overcount the number of errors associated to any given syndrome sequence. There are three types of overcounting, which we may analyze separately.

A. RECOUNTING LOWER ORDER ERRORS

Suppose we have already computed the correct error count $\mathbf{n}_i(j)$ for $0 \le j \le \omega - 1$ (for all degenerate sets), and we are currently trying the determine $\mathbf{n}_i(\omega)$.

For the error $E_{i_1}^B E_{i_2}^B \dots E_{i_{\omega-1}}^B$, composing with any $E_{i_k}^B$ $(1 \le k \le \omega - 1)$ reduces the error to one of order $\omega - 2$, which was previously counted. To determine how many errors stem from this dynamic, we may note that any fake compound error of order ω has a corresponding error of order $\omega - 2$, which has already been counted in $\mathbf{n}_i(\omega - 2)$. Similarly, any error E_i counted in $\mathbf{n}_i(\omega - 2)$ has a corresponding set of fake compound errors that appear in $n_i(\omega)$. As E_i stems from $\omega - 2$ base errors, each affecting a different CNOT gate, these fake compound errors must correspond to an error of the form $E_i E_j^B E_j^B$, where E_j^B is a base error from a CNOT gate not present in E_i. For each CNOT gate there are $K = 15$ associated base errors, so there are a total of

$$\zeta_{\omega-1,1}(1, 0) = K(\tilde{N}_{\text{CNOT}} - (\omega - 2)) \tag{103}$$

fake error compounds that associated with the error E_i. Alternatively, defining $\omega = a + k$, with $k = 1$, we may also write the total as

$$\zeta_{a,1}(1, 0) = K(\tilde{N}_{\text{CNOT}} - (a - k)). \tag{104}$$

The same principle applies to higher order combinations. If a order-a error E_i is composed with a order-k error E_j $(a \ge k)$, and the base errors composing E_j all stem from CNOT gates whose same base errors already compose E_i, then the resulting compound error will have order $a - k$, instead of $a + k$. The total number of errors is now given by

$$\zeta_{a,k}(k, 0) = \frac{1}{k!} \prod_{j=0}^{k-1} K(\tilde{N}_{\text{CNOT}} - (a - k) - j) \tag{105}$$

$$= K^k \binom{\tilde{N}_{\text{CNOT}} - (a - k)}{k}. \tag{106}$$

Note, however, that, when composing a order-a error E_i with a order-b error E_j $(a \ge b \ge k)$, it may be the case that only some, and not all, of the base errors composing E_j appear in E_i. In general, there will only be k base errors in common, for all $1 \le k \le b$.

In this case, to cover all possible fake errors, we must choose not k CNOT gates out of the $\tilde{N}_{\text{CNOT}} - (a - k)$ gates not related to the order-$(a - k)$ error (as in (106)), but instead choose k CNOT gates out of the gates not related to both the order-$(a - k)$ error, but also the $(b - k)$ base errors in E_j that are valid. Therefore, there are a total of $\tilde{N}_{\text{CNOT}} - (a - k) - (b - k)$ CNOT gates from which we must consider k invalid base errors.

Moreover, associated with the order-$(a - k)$ error from E_i, there are several possible valid $(b - k)$ base errors stemming from E_j. The total number of fake errors is then given by

$$\zeta_{a,b}(k, 0) = K^k \binom{\tilde{N}_{\text{CNOT}} - (a - k) - (b - k)}{k} \tag{107}$$

$$\times \binom{(a - k) + (b - k)}{b - k} \tag{108}$$

$$= K^k \binom{\tilde{N}_{\text{CNOT}} - c}{k} \binom{c}{b - k} \tag{109}$$

$$\text{with } c := a + b - 2k. \tag{110}$$

Note that, for $b = k$, we have $c = a - k$, so that (109) trivially reduces to (106). The resulting lower order error will have order c. We would need to discount its affect on $\mathbf{n}_i(\omega)$ to obtain the correct count. Unfortunately, it would be difficult to determine the original syndromes of the errors that combined to result in the impossible error, as they may have different

origins. As an approximation, we use $\mathbf{n}_i(c)$ to estimate the error count. The resulting correction is achieved by subtracting $\mathbf{n}_i(c)$ times ζ from $\mathbf{n}_i(\omega)$.

B. COUNTING IMPOSSIBLE ERRORS

If a order-a error E_i is composed with a order-r error E_j ($a \geq r$), and the base errors composing E_j all stem from CNOT gates already associated with the base errors that compose E_i, then the resulting compound error will be impossible, since it will contain at least two different base errors associated with the same CNOT gate (one from E_i and one from E_j).

For $r = 1$, each error in $\mathbf{n}_i(a)$ will have

$$\zeta_{a,1}(0,1) = (K-1)a \tag{111}$$

associated impossible order-$(a+1)$ errors, since E_i is composed of a base errors, and for each base error, there are $(K-1)$ different base errors associated to the same CNOT gate.

For a general r, we have, for each order-a error

$$\zeta_{a,r}(0,r) = (K-1)^r \binom{a}{r} \tag{112}$$

associated impossible errors.

When composing a order-a error E_i with a order-b error E_j ($a \geq b$), it may be the case that only $r < b$ base errors composing E_j are impossible, with the remaining $(b-r)$ base errors stemming from CNOT gates not related to E_i.

To estimate the number of impossible errors, we may look at $\mathbf{n}_i(a+b-r)$. As before, we must choose r CNOT gates out of the a gates related to E_i to count the number of impossible errors. But, as seen in the previous section, to must also count the possible $(b-r)$ base errors in E_j that are valid. These factors result in

$$\zeta_{a,b}(0,r) = (K-1)^r \binom{a}{r}\binom{a+b-r}{b-r} \tag{113}$$

$$= (K-1)^r \binom{c}{r, a-r, b-r} \tag{114}$$

$$\text{with } c := a+b-r \tag{115}$$

$$\text{and } \binom{x+y+z}{x,y,z} := \frac{(x+y+z)!}{x!y!z!}. \tag{116}$$

The multinomial coefficient. Again, note that (114) trivially reduces to (112) when $b = r$. The resulting lower order error will have order c, so, for this case, we would also need to discount its affect on $\mathbf{n}_i(\omega)$ by subtracting $\mathbf{n}_i(c)$ times ζ.

C. COUNTING PERMUTED COPIES

Consider an order-$(\omega-1)$ error $E^B_{i_1}E^B_{i_2}\ldots E^B_{i_{\omega-1}}$ (with $i_1 < i_2 < i_3 < \ldots$), coming from $\tilde{D}_{\omega-1}$, and the error $\hat{E}^B_j$, coming from $\tilde{D}_1$. W.l.o.g., suppose $j < i_1$. Then, for $\tilde{D}_\omega$, we will not only count the error $E^B_j E^B_{i_1}E^B_{i_2}\ldots E^B_{i_{\omega-1}}$, but also that same error coming from the combination of the errors $E^B_j E^B_{i_1}E^B_{i_2}\ldots E^B_{i_{\omega-1}}\backslash E^B_{i_k}$ and $E^B_{i_k}$. In total, we overcount each order-ω error ω times.

In general, for every order-ω error, any possible combination of order-a errors and order-b errors that can generate it (with $\omega = a+b$) will appear in the counting. Since there are

$$R_{a,b} := \binom{\omega}{a} = \binom{\omega}{b} \tag{117}$$

ways for order-a and order-b errors to generate an order-ω error, the final counting (after discounting the previous overcounting cases) should be reduced by a factor of $R_{a,b}$.

D. FULL EXPRESSION

In general, the erroneous errors that the decoding procedure may containing not only repeated base errors ($k > 0$), but also base errors stemming from the same CNOT gate ($r > 0$). Therefore, these two factors need to be considered together.

Combining the analyses of the previous sections, we conclude that, when composing a order-a error E_i with a order-b error E_j ($a \geq b$), we can have k base errors E_j already appearing in E_i, and r base errors in E_j sharing the same origin CNOT gate as a base error in E_i, with $k + r \leq b$.

To count all these errors, we may look that the errors with order $c = (a+b-2k-r)$, from which we can generate all the invalid order-ω errors. As before, the k repeated base errors are chosen from those associated with CNOT gates that are not related to a valid base error in the compound error. There are $\tilde{N}_{\text{CNOT}} - c$ such gates, and each one has K associated base errors.

Moreover, the from the c base errors, we may consider that $a-k-r$ (respectively, $b-k-r$) correspond to the base errors in E_i (respectively, E_j) that raise no issue, with the remaining r errors corresponding to base errors stemming from CNOT gates that also originated invalid base errors in E_j.

Grouping all three overcounting issues, we end up with

$$\zeta_{a,b}(k,r) = K^k(K-1)^r \xi_{a,b}(k,r) \tag{118}$$

with

$$\xi_{a,b}(k,r) := \binom{\tilde{N}_{\text{CNOT}} - c}{k}\binom{c}{r, a-k-r, b-k-r} \tag{119}$$

$$\text{and } c := a+b-2k-r \tag{120}$$

which generalizes (109) and (114).

The corrected count is consequently given by

$$\mathbf{n}_i(\omega) = \frac{1}{R_{a,b}}\left[\tilde{\mathbf{n}}_i(\omega) - \sum_{\substack{0<k+r\leq b\\ k,r\geq 0}} \zeta_{a,b}(k,r)\mathbf{n}_i(c)\right] \tag{121}$$

$$\text{with } c := a+b-2k-r \tag{122}$$

as indicated in (97), with $\tilde{\mathbf{n}}_i(\omega)$ being the original count from the optimized decoding process. As previously indicated, since the estimate of the impossible errors is not exact, this formula is approximate.

APPENDIX J
ALTERNATIVE DEFINITIONS OF $\mathcal{F}_E$ AND $\mathcal{F}_L$

Instead of using the formulation described in Sections IV-A and IV-B, we may consider an alternative formulation that, while less computationally efficient, is conceptually more straightforward. Under this formalism, the components $\mathtt{E}_i$ and $\mathtt{L}_i$ can be computed at once from e_i, so there is less of an need to separate the two processes.

For a given error E_i, we compute e_i. Taking the encoding U, we compute

$$e_i^u := U^\dagger e_i U. \tag{123}$$

The unencoded error pattern e_i^u corresponds to the effect of e_i on the quantum state if it were decoded. We may decompose it into

$$e_i^u := \mathtt{E}_i^u \mathtt{S}_i^u \mathtt{L}_i^u \tag{124}$$

with

$$\mathtt{E}_i^u = U^\dagger \mathtt{E}_i U \tag{125}$$

$$\mathtt{S}_i^u = U^\dagger \mathtt{S}_i U \tag{126}$$

$$\mathtt{L}_i^u = U^\dagger \mathtt{L}_i U. \tag{127}$$

We may also decompose it into the Pauli string for the first k data qubits (e_i^D) and the additional $(n-k)$ redundancy qubits (e_i^R)

$$e_i^u =: e_i^D \otimes e_i^R. \tag{128}$$

From (3) to (5), we have that

$$Z_{i+k} = U^\dagger S_i U \tag{129}$$

$$X_j = U^\dagger \bar{X}_j U \tag{130}$$

$$Z_j = U^\dagger \bar{Z}_j U. \tag{131}$$

Therefore, decoding e_i into e_i^u cleanly separates the different components. $\mathtt{L}_i^u$ corresponds to the components of e_i^u in the first k qubits. We have

$$\mathtt{L}_i^u := e_i^D \otimes I_{n-k} \tag{132}$$

$$= \left(\prod_{j=1}^{k} X_j^{b_j^X}\right)\left(\prod_{j=1}^{k} Z_j^{b_j^Z}\right) \tag{133}$$

$$\Rightarrow \mathtt{L}_i = U(e_i^D \otimes I_{n-k})U^\dagger \tag{134}$$

$$= \left(\prod_{j=1}^{k} \bar{X}_j^{b_j^X}\right)\left(\}\prod_{j=1}^{k} \bar{Z}_j^{b_j^Z}\right) \tag{135}$$

where b_j^X and b_j^Z are the X and Z components of $\mathbf{e}_i^D$, respectively.

Let $e_i^{R,X}$ and $e_i^{R,Z}$ be the X and Z components of e_i^R, respectively [where the Y components have been decomposed into X and Z, as in (10)], so that $e_i^R = e_i^{R,X} e_i^{R,Z}$ (disregarding the phase factor). We have

$$\mathtt{E}_i^u := I_k \otimes e_i^{R,X} \tag{136}$$

$$= \prod_{i=k+1}^{n} X_i^{b_{i-k}^X} \tag{137}$$

$$\Rightarrow \mathtt{E}_i = U(I_k \otimes e_i^{R,X})U^\dagger \tag{138}$$

$$= \prod_{i=k+1}^{n} (U X_i U^\dagger)^{b_{i-k}^X} \tag{139}$$

and

$$\mathtt{S}_i^u := I_k \otimes e_i^{R,Z} \tag{140}$$

$$= \prod_{i=k+1}^{n} Z_i^{b_{i-k}^Z} \tag{141}$$

$$\Rightarrow \mathtt{S}_i = U(I_k \otimes e_i^{R,Z})U^\dagger \tag{142}$$

$$= \prod_{i=k+1}^{n} (U Z_i U^\dagger)^{b_{i-k}^Z} \tag{143}$$

where b_{i-k}^X and b_{i-k}^Z are the X and Z components of $\mathbf{e}_i^{R,X}$ and $\mathbf{e}_i^{R,Z}$, respectively. As this procedure is deterministic, we obtain unique components $\mathtt{E}_i$, $\mathtt{S}_i$, and $\mathtt{L}_i$ associated with the error pattern e_i.

Regarding runtime complexity, the method presented in Section IV-A requires just $\mathcal{O}(n-k)$ steps per error E_i, in order to assemble $\mathtt{E}_i$ from the precomputed Z_{h_i} and X_{h_i-n} terms. The method presented in Section IV-B is more involved. Computing e_i' requires $2n$ steps. Determining its stabilizer component $\mathtt{S}_i$ requires identifying the pivots in $\mathbf{e}_i'$ ($\mathcal{O}(n)$ steps) and then multiplying the constituting stabilizers by $\mathbf{e}_i'$. As it is constituted by $\mathcal{O}(n-k)$ stabilizers, and accounting for each takes at most $(2n-(n-k-1)) = n+k+1$ steps, the whole stabilizer part takes $\mathcal{O}((n-k)(n+k+1))$ steps. For the logical component, there are $\mathcal{O}(2k)$ components in $\mathbf{e}_i'$, and the whole each operator takes $(2n-(n-k)-2k+1) = n-k+1$ steps, for a total of $\mathcal{O}((2k)(n+k-1))$ steps. The full procedure requires

$$\mathcal{O}\left(n + (n-k)(n+k+1) + (n-k+1)(2k)\right) \tag{144}$$

$$\sim \mathcal{O}\left(n^2, k^2\right) \sim \mathcal{O}\left(n^2\right) \tag{145}$$

steps per error E_i to compute $\mathtt{L}_i$. The computation for N_E errors requires

$$\mathcal{O}\left(N_E n^2\right) \tag{146}$$

steps.

For this simpler technique, the greatest computational expense comes from computing e_i^u for each error E_i, as the remaining steps can be precomputed and subsequently applied to all errors.

To facilitate the calculation, we may precompute the e^u patterns associated with all base errors, and use those to compute the pattern e_i^u for each error E_i.

From [33], simulating a stabilizer circuit (i.e. U) with N gates takes $\mathcal{O}(n^2 N)$ steps. Since U is built from $\mathcal{O}(n \log^2 n)$

Clifford gates, we have that the full precomputation associated with the base errors scales as

$$\mathcal{O}\left(N_{\text{CNOT}} n^3 \log^2 n\right). \tag{147}$$

The cost of computing e_i^u for each E_i then scales as

$$\mathcal{O}(n\omega) \tag{148}$$

where ω is the order of the error. The full computation for N_E errors requires

$$\mathcal{O}\left(N_E n\omega + N_{\text{CNOT}} n^3 \log^2 n\right) \tag{149}$$

steps. For cases where $N_E \gg N_{\text{CNOT}}$, the simpler approach may lead to a faster implementation, while, for smaller cases, the main approach is faster, as it does not require precomputation.

APPENDIX K
ANALYSIS OF NUMERICAL RESULTS

To get a good understanding of the performance of the decoding method, we consider an equidistant range for h, and we sample p using the expression

$$p = 10^h. \tag{150}$$

For that reason, most of the fits in this section are performed after applying a logarithmic transformation to both the dependent and independent variables. That is, we prefer to work with $\log(p)$ than p directly, as it is more numerically stable.

In this section, we verify that $C(p)$ and P_{failure} scale as

$$C(p) \simeq e^{-\gamma N_{\text{CNOT}} p} \tag{151}$$

$$\simeq 1 - \gamma N_{\text{CNOT}} p \tag{152}$$

$$P_{\text{failure}}(p) \simeq 1 - e^{-\mu p^\eta}, \text{ for } \eta \in \mathbb{N} \tag{153}$$

$$\simeq \mu p^\eta \tag{154}$$

for $p \ll 1$. μ and γ are positive real parameters, and η is the lowest order that the code cannot fully correct. We analyze these expressions separately in the next sections.

A. ESCAPED ERRORS

We start by considering $P_u(p, 1)$. For the noise model in Section II, we expect that the probability of finding escaped errors will be given by

$$P_u(p,1) \simeq \sum_{i=1}^{N_{\text{CNOT}}} \binom{N_{\text{CNOT}}}{i} (\gamma p)^i (1-\gamma p)^{N_{\text{CNOT}}-i} \tag{155}$$

$$= \sum_{i=1}^{N_{\text{CNOT}}} (-1)^{i+1} \binom{N_{\text{CNOT}}}{i} (\gamma p)^i \tag{156}$$

$$= 1 - \sum_{i=0}^{N_{\text{CNOT}}} \binom{N_{\text{CNOT}}}{i} (-\gamma p)^i \tag{157}$$

where γ reflects the fraction of errors that can escape. For $p \ll 1$, the lower order terms dominate, so we may use the approximation

$$\binom{N_{\text{CNOT}}}{i} \simeq \frac{{N_{\text{CNOT}}}^i}{i!}, \text{ for } N_{\text{CNOT}} \gg i \tag{158}$$

and we have

$$P_u(p,1) \simeq 1 - \sum_{i=0}^{N_{\text{CNOT}}} \frac{(-N_{\text{CNOT}}\gamma p)^i}{i!} \tag{159}$$

$$\simeq 1 - \sum_{i=0}^{\infty} \frac{(-N_{\text{CNOT}}\gamma p)^i}{i!} \tag{160}$$

$$= 1 - e^{-\gamma N_{\text{CNOT}} p}. \tag{161}$$

Given (37) and (44), we end up with (151)

$$C(p) \simeq e^{-\gamma N_{\text{CNOT}} p} \tag{162}$$

where γ is an unknown parameter. Since we are considering $p \ll 1$, this expression may be further simplified into

$$C(p) \simeq 1 - \gamma N_{\text{CNOT}} p. \tag{163}$$

In order to perform a fit, we consider a more general version of this expression, given by

$$C(p) \simeq \exp(-\gamma N_{\text{CNOT}} p^{\eta_C}) \tag{164}$$

and we fit the function

$$\log_{10}(-\log(C(p))) = \eta_C \log_{10}(p) + \log_{10}(\gamma N_{\text{CNOT}}) \tag{165}$$

$$\Rightarrow \log_{10}(-b) = \eta_C \log_{10}(p) + \beta \tag{166}$$

where $\beta = \log_{10}(\gamma N_{\text{CNOT}})$ and b stems from (48). We use $\log_{10}$ to keep the figures more legible.

We may now study the behavior of η_C and β for different n. See Fig. 9 for the results. As expected, we observe that $\eta_C \simeq 1$, regardless of n.

From Appendix C, we have that $N_{\text{CNOT}} \sim \mathcal{O}(n^2)$. If we assume that $\gamma \sim \mathcal{O}(\text{poly}(n))$, then β should scale as

$$\beta = b_1 \log(n - n_\beta) + b_2. \tag{167}$$

with $\eta_0 = 1$. We assume this expression for its fit. The fitted parameters are given in Table 4.

TABLE 4. Fitted Parameters for β

Variable	Value
b_1	0.4494
n_β	3.6453
b_2	0.7505

We may alternatively write β as

$$\beta \simeq \log_{10}(5.630 \cdot (n - 3.6453)^{1.035}). \tag{168}$$

Given the dependence between β and N_{CNOT}, we may conclude that

$$\gamma N_{\text{CNOT}} \sim \mathcal{O}\left(n^{1.035}\right) \simeq \mathcal{O}(n) \tag{169}$$

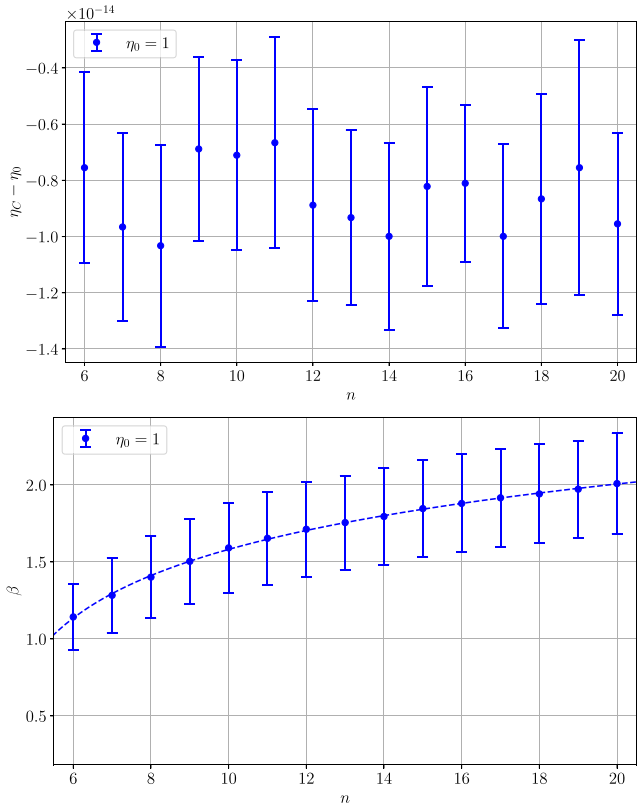


FIGURE 9. Fit of the various parameters, averaged over 50 random codes for each n.

despite the actual N_{CNOT} count scaling with $\mathcal{O}(n^2)$. These results are in line with the expectations from the theoretical analysis of Appendix C, indicating that we may observe a visible $p_{\text{threshold}}$.

B. DIRECT P_{FAILURE} EXTRAPOLATION

We may apply a similar procedure to P_{failure}. We empirically observe that (153) holds. However, unlike the previous section, it is no longer the case that $\eta_C \simeq 1$.

If a code is able to correct all errors of order $\omega < \eta$, then we expect P_{failure} to be given by

$$P_{\text{failure}} = \sum_{i=1}^{N_{\text{CNOT}}} f_i(p, N_{\text{CNOT}}) \binom{N_{\text{CNOT}}}{i} p^i (1-p)^{N_{\text{CNOT}}-i} \tag{170}$$

$$\simeq \sum_{i=\eta}^{N_{\text{CNOT}}} \tilde{f}_i \binom{N_{\text{CNOT}}}{i} p^i (1-p)^{N_{\text{CNOT}}-i}, \text{ for } p \ll 1 \tag{171}$$

$$\simeq \tilde{f}_\eta \binom{N_{\text{CNOT}}}{\eta} p^\eta (1-p)^{N_{\text{CNOT}}-\eta}, \text{ for } p \ll 1 \tag{172}$$

$$= \mu p^\eta, \text{ for } p \ll 1 \tag{173}$$

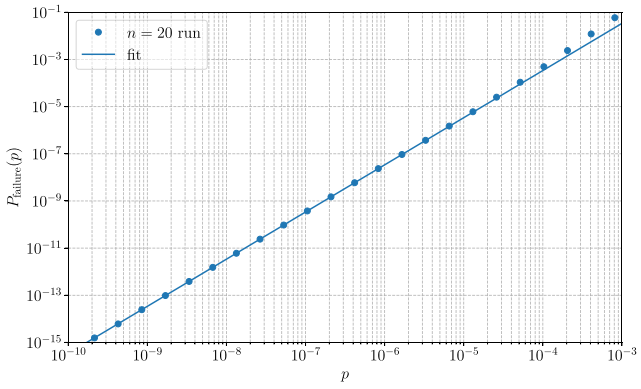


FIGURE 10. Fit of (166), for a random example with $n = 20$. We fit only using values below 10^{-6}, where it is safe to assume $p \ll 1$. As expected, the fit worsens for higher p.

TABLE 5. Fitted Parameters for α

Variable	$\eta_0 = 1$	$\eta_0 = 2$
a_1	−0.0074	0.0838
a_2	−0.8170	2.8530

for some unknown μ, where $\tilde{f}_i$ is an approximation of $f_i(p, N_{\text{CNOT}})$, which is the unknown function. In the approximation in (171), we consider $\tilde{f}_i$ as a scalar.

In order to perform a fit, we consider an equivalent version of this expression for $p \ll 1$, given by

$$P_{\text{failure}}(p) \simeq 1 - \exp(-\mu p^\eta) \tag{174}$$

and we fit the function

$$\log_{10}(-\log(1 - P_{\text{failure}})) = \eta \log_{10}(p) + \log_{10}(\mu) \tag{175}$$

$$= \eta \log_{10}(p) + \alpha \tag{176}$$

where $\alpha = \log_{10}(\mu)$. As before, we use $\log_{10}$ to keep the figures more legible. See Fig. 10 for an example.

We may now study the behavior of η and α for different n. See Fig. 11 for the results. In this case, we have different η values, and the values cluster around integers. Since the n values we tested are relatively low, we only observe η equal to either 1 or 2. As these cases display notably different behavior, we separate their data before fitting.

As before, we fit η to an expression of the form

$$\eta = \eta_0 + d_1 \log(1 + \exp(d_2 \cdot (n - n_P))). \tag{177}$$

We empirically observe that the α variable is better fitted by a simple linear expression

$$\alpha = a_1 n + a_2. \tag{178}$$

The fitted parameters are given in Table 5.

We may also have a look at the probability of having different η values. The empirical results can be seen in Fig. 12 and Table 6. We observe that, for low n values, the fraction of random codes with $\eta = 1$ and $\eta = 2$ is roughly constant, since the code's capabilities are not large enough to generally

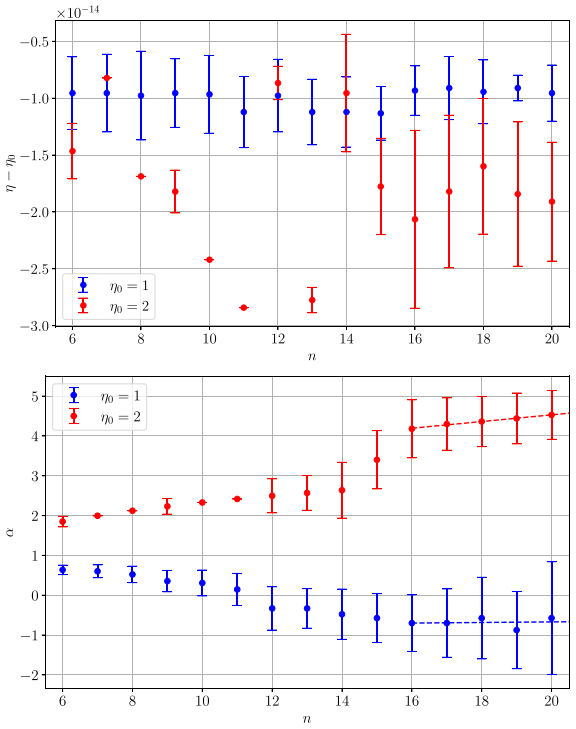


FIGURE 11. Plot of η and α. We fit a linear function to α for the higher n values.

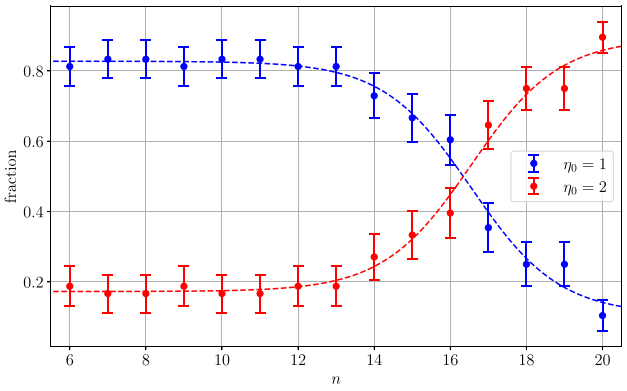


FIGURE 12. Fraction of the simulations with different η_0 values.

TABLE 6. Fitted Parameters

Variable	$\eta_0 = 1$	$\eta_0 = 2$
s_1	-0.7206	0.7206
s_2	0.8665	0.8665
n_f	16.5506	16.5506
s_3	0.4673	0.5327

correct all $\omega = 1$ errors. As n increases, the probability that the code corrects all order-1 errors also increases, and we so the fraction of $\eta = 2$ cases increases. Its behavior follows a sigmoid-like function, of the form:

$$s_1 \left(\frac{1}{1 + e^{-s_2(n-n_f)}} - \frac{1}{2} \right) + s_3. \tag{179}$$

The errorbars indicate the standard deviation, which stems from the finite number (48) of samples taken for each n value. Also, note the symmetry in the parameters in Table 6, reflecting the fact that the fractions must add up to one.

For very low $n \leq 12$, we observe that about 20% of the codes have $\eta = 2$. On further inspection, their seemingly high performance does not stem from strong correction capabilities, but from the fact that, due to random chance, the stabilizers of these codes have relatively low weight, leading to a very low number of CNOT gates in the SE circuit, and consequently fewer errors needed to consider. In a more complex setting where there are additional sources of error, we would expect these codes in particular to perform poorly.

C. EXTRAPOLATING BEHAVIOR FOR LARGER n

We may take the results above and use them to extrapolate the performance for larger n values. We also incorporate the uncertainty observed in the numerical data by using the standard deviation observed for the sampled n to estimate the deviation for larger n.

APPENDIX L
SINGLE-QUBIT GATE ERROR MODEL CONSIDERATIONS

Standard circuit-level noise models typically include single-qubit gate errors, particularly identity gate errors from qubit idling [34]. The exclusion of these errors in the present work represents a limitation of the current theoretical analysis. Including comprehensive single-qubit gate errors would likely reduce our reported threshold values, and we consider it as important future work.

We deliberately use a simple, unoptimized stabilizer implementation to avoid confounding our theoretical results with circuit-specific optimizations, as indicated in Appendix A. While gate count and idling could be reduced through circuit optimization techniques, we chose not to incorporate such optimizations to maintain the generality of our theoretical analysis.

Our model would be more realistic for trapped-ion systems where idling errors (due to relaxation or decoherence) are significantly lower ($\sim 10^{-6}$) compared to superconducting qubits ($\sim 10^{-4}$). In trapped-ion platforms, coherence times are typically much longer, making idling errors less significant compared to active gate operations. This platform-dependent behavior suggests that our simplified model captures the dominant error sources for certain quantum computing architectures.

Nonetheless, in the rest of this section we show that many of the single-qubit gate errors can be absorbed into our model through mathematical equivalences.

A. HADAMARD GATE ERRORS

The circuit contains, for each ancilla qubit, a Hadamard gate after preparation and before measurement. If we model this gate as suffering an error corresponding to an additional Pauli gate, then it can be shown that the error can be absorbed into preparation and measurement errors, respectively.

For the Hadamard gate after preparation, a Z-type error is equivalent to an X error before the Hadamard gate (since $HX = ZH$), which is equivalent to a flip preparation bit. A X-type error similarly corresponds to a Z-type error before the Hadamard gate, i.e., right after preparation. Since the prepared qubit is initially either in the state $|0\rangle$ or $|1\rangle$, a Z-type error does not introduce additional errors, since it acts as the identity.

For the Hadamard gate before measurement, a Z-type error is similarly irrelevant, since the measurement is done in the computational basis. A X-type error is equivalent to a measurement error, where the measurement bit flips.

For both cases, a Y-type error is a combination of both error types. Therefore, if the probability of a preparation/measurement error is p_M, and the probability of a specific Hadamard gate error is $p_H/3$, then we can model Hadamard gate errors by simply using the equivalent probability $p'_M = p_M(1 - \frac{2p_H}{3}) + \frac{2p_H}{3}(1 - p_M) \simeq p_M + \frac{2p_H}{3}$. Since, in practice, $p_M \gg p_H$, disregarding the Hadamard gate errors does not significantly affect the analysis.

B. 1-QUBIT GATE ERRORS

For the 1-qubit gates we consider in our model, possibly appearing before and after the target of a CNOT gate, we can similarly map them so that the errors actually arise from the associated CNOT gate, while assuming that the 1-qubit gates are perfect. If their error rate is p_G, then the equivalent error rate for the CNOT gates incorporating the 1-qubit Pauli gate errors is $p' \simeq p + p_G$. Similarly to before, $p \gg p_G$, so their exclusion from our model does not significantly affect the analysis.

C. IDLING ERRORS

For comprehensive circuit-based error models, it is standard to also include errors coming from idle qubits changing over time, either due to relaxation, decoherence, or more complex dynamics (e.g., crosstalk). This scenario can be modeled by assuming that identity gates are applied to the circuit, and that these gates get converted to a Pauli gate when an error occurs.

Unlike the previous scenarios, it is harder to map these errors to equivalent errors in the CNOT gates. We can evolve the idling errors forward in the circuit up to the first CNOT error they encounter, where they can potentially increase in weight. In our formulation, if we assume that each identity gate with error p_I takes as long as a CNOT gate application, then the errors can compound quickly. Since the stabilizer application is not optimized, if we apply the $n - k$ stabilizers in series, and each stabilizer affects $w := \lambda n$ qubits (where $\lambda = 3/4$ in our implementation, and $\simeq 1/2$ in the optimized version in Appendix A), then for each main qubit there are on average

$$\simeq \lambda n \left[1 + 2(1-\lambda) + 3(1-\lambda)^2 + \cdots\right] \tag{180}$$

$$= \frac{n}{\lambda} =: m \tag{181}$$

identity gates between CNOT gates.

During every idle slot the qubit is acted on by a singlequbit depolarizing channel $\mathcal{D}_{p_I}(\rho) = \left(1 - \frac{3p_I}{4}\right)\rho + \frac{p_I}{4}\sum_{\alpha\in\{X,Y,Z\}}\alpha\rho\alpha$.

In the Pauli frame each gate independently applies

$$E_j = \begin{cases} I, & \text{with probability } 1-q \\ X, Y \text{ or } Z, & \text{each with probability } q/3 \end{cases} \tag{182}$$

$$q := \tfrac{3p_I}{4}. \tag{183}$$

Because the nontrivial Pauli operators modulo a phase form the abelian group $V_4 = \{I, X, Y, Z\}$, the net error after the m idle slots is the group product $E := E_m \cdots E_1$. A standard character argument for random walks on V_4 gives

$$\Pr[E = I] = \tfrac{1}{4}\left[1 + 3(1-q)^m\right]. \tag{184}$$

Hence, the equivalent identity error is

$$p'_I = 1 - \Pr[E = I] \tag{185}$$

$$= \frac{3}{4}\left[1 - (1-q)^m\right] \tag{186}$$

$$= \frac{3}{4}\left(1 - \left(1 - \frac{3p_I}{4}\right)^{n/\lambda}\right) \tag{187}$$

$$\simeq \frac{9}{16} n p_I/\lambda, \quad \text{for } np_I/\lambda \ll 1. \tag{188}$$

Since we expect p_I to be about one order of magnitude lower than p (except for trapped-ion platforms, where p_I is much lower), even relatively low values of n make idling nonnegligible. Nonetheless, we can show that, in a more realistic setting, where the gate implementation is optimized, idling can be largely removed, thereby making this error source negligible.

In an optimized setting, besides implementing the stabilizer optimizations in Appendix A, we would also parallelize the CNOT gates associated with the different stabilizers as much as possible. In general, it would be possible to parallelize CNOT gates associated with different stabilizers whenever their target qubits are different.

To quantify how much circuit parallelization is possible we model the scheduling of the CNOT gates associated with the $t := (n - k) \simeq n$ stabilizers. Write $A = \{a_1, \ldots, a_t\}$ for the ancilla (control) qubits and $B = \{b_1, \ldots, b_n\}$ for the data (target) qubits. Every CNOT is an $edgee = (a_i, b_j)$ in the bipartite graph $G = (A \cup B, E)$. For the code families considered here the following:

1) each ancilla participates in $w = \lambda n$ edges ($\lambda = \frac{1}{2}$ or $\frac{3}{4}$);
2) for a fixed data qubit b_j the number of incident edges X_j is a random variable $X_j \sim \text{Binomial}\left(t, \frac{w}{n}\right) = \text{Binomial}\,(n, \lambda)$ with mean $\mu = \mathbb{E}[X_j] = \lambda n = w$.

IEEE Transactions on Quantum Engineering

A single time step ("layer") of the circuit is a matching of G: a set of edges with no common end–points, so that no qubit is touched twice in the same layer. Partitioning E into the minimum number of such matchings is the edge–coloring problem; for every bipartite graph, Kőnig's theorem gives

$$\chi'(G) = \Delta(G) \tag{189}$$

where $\Delta(G)$ is the maximum vertex degree. Consequently

$$D = \Delta(G) = \max\Big\{ w,\ \max_{1\le j\le n} X_j \Big\} \tag{190}$$

is the optimal CNOT depth.

Let $M_n := \max_j X_j$. A standard Chernoff bound gives, for any $a > 0$

$$\Pr[X_j \ge \mu + a] \le \exp\left(-\tfrac{a^2}{2(\mu+a/3)}\right). \tag{191}$$

Choosing $a = \sqrt{2\mu \ln n}$ and unionbounding over the n data qubits

$$\Pr\left[M_n \ge \mu + \sqrt{2\mu \ln n}\right] \xrightarrow[n\to\infty]{} 0 \tag{192}$$

so that, with high probability

$$M_n \ \le\ w + \sqrt{2\, w \ln n}.$$

Because $M_n \ge w$ always, we may replace the order symbol in (190) by its leading constant and write

$$\mathbb{E}[D] = \ w + \sqrt{2\, w \ln n} \ + \ o\left(\sqrt{w \ln n}\right) \tag{193}$$

$$\simeq\ w + \sqrt{2\, w \ln n} \tag{194}$$

$$=\ \lambda\, n + \sqrt{2\,\lambda\, n \ln n}. \tag{195}$$

Along any ancilla line exactly w CNOTs are executed in D layers, so the number of idle layers on that qubit is $D - w$. Averaging the gap between successive CNOTs over the $w - 1$ internal intervals gives

$$N_I = \frac{D - w}{w - 1} \tag{196}$$

$$\simeq \frac{\sqrt{2\, w \ln n}}{w} \tag{197}$$

$$= \sqrt{\frac{2 \ln n}{w}} = \sqrt{\frac{2}{\lambda}\,\frac{\ln n}{n}}. \tag{198}$$

Hence, keeping only the firstorder term,

$$N_I \ \simeq\ \sqrt{\frac{2}{\lambda}\,\frac{\ln n}{n}}. \tag{199}$$

This value is lower than 1 and quickly converges to 0 for high n. Therefore, similarly to before, we could consider an equivalent $p' \simeq p + N_I p_I$ to model idling errors.

D. EQUIVALENT THRESHOLD

Accounting for these errors, and considering that they are at least one order of magnitude lower than CNOT, preparation and measurements, leads to an equivalent error rate that is around 10% higher. Consequently, in this more complex model, with the necessary circuit optimizations, we would get a $p_{\text{threshold}}$ that is 10% lower. We note, however, that this setup would not be sensible, since our empirical threshold stems from an purposefully unoptimized stabilizer implementation. It is possible that an implementation with optimized stabilizers (which is outside the scope of this work) would lead to a naturally higher threshold, regardless of the 1-qubit gate and idling errors.

ACKNOWLEDGMENT

The authors thank Dr. Bill Munro (NTT Basic Research Labs, Japan) and Prof. Kae Nemoto (National Institute of Informatics, Japan) for insightful discussions about QGRAND. The author Francisco Monteiro thanks Prof. Frank Kschischang (University of Toronto) for discussions on noise guessing decoding, and Dr. Ioannis Chatzigeorgiou (Lancaster University) for discussions on RLCs and noise guessing decoding.

REFERENCES

[1] K. R. Duffy, J. Li, and M. Médard, "Capacity-achieving guessing random additive noise decoding," *IEEE Trans. Inf. Theory*, vol. 65, no. 7, pp. 4023–4040, Jul. 2019, doi: 10.1109/TIT.2019.2896110.

[2] D. Cruz, F. A. Monteiro, and B. C. Coutinho, "Quantum error correction via noise guessing decoding," *IEEE Access*, vol. 11, pp. 19446–119461, 2023, doi: 10.1109/ACCESS.2023.3327214.

[3] W. An, M. Médard, and K. R. Duffy, "Keep the bursts and ditch the interleavers," *IEEE Trans. Commun.*, vol. 70, no. 6, pp. 3655–3667, Jun. 2022, doi: 10.1109/TCOMM.2022.3171798.

[4] M. Shirvanimoghaddam et al., "Short block-length codes for ultra-reliable low latency communications," *IEEE Commun. Mag.*, vol. 57, no. 2, pp. 130–137, Feb. 2019, doi: 10.1109/MCOM.2018.1800181.

[5] R. Alberto and F. A. Monteiro, "Downlink MIMO-NOMA with and without CSI: A short survey and comparison," in *Proc. 12th Int. Symp. Commun. Syst., Netw. Digit. Signal Process.*, Porto, Portugal, Jun. 2020, pp. 1–6, doi: 10.1109/CSNDSP49049.2020.9249527.

[6] F. A. Monteiro and I. J. Wassell, "Recovery of a lattice generator matrix from its gram matrix for feedback and precoding in MIMO," in *Proc. 4th Int. Symp. Commun., Control Signal Process.*, Limassol, Cyprus, 2010, pp. 1–6, doi: 10.1109/ISCCSP.2010.5463408.

[7] M. Chiani and L. Valentini, "Short codes for quantum channels with one prevalent pauli error type," *IEEE J. Sel. Areas Inf. Theory*, vol. 1, no. 2, pp. 480–486, Aug. 2020, doi: 10.1109/JSAIT.2020.3012827.

[8] P. W. Shor, "Scheme for reducing decoherence in quantum computer memory," *Phys. Rev. A*, vol. 52, pp. R2493–R2496, 1995, doi: 10.1103/PhysRevA.52.R2493.

[9] A. M. Steane, "Error correcting codes in quantum theory," *Phys. Rev. Lett.*, vol. 77, pp. 793–797, 1996, doi: 10.1103/PhysRevLett.77.793.

[10] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Phys. Rev. A*, vol. 86, no. 3, Sep. 2012, Art. no. 032324, doi: 10.1103/PhysRevA.86.032324.

[11] A. R. Calderbank and P. W. Shor, "Good quantum error-correcting codes exist," *Phys. Rev. A*, vol. 54, no. 2, pp. 1098–1105, Aug. 1996, doi: 10.1103/PhysRevA.54.1098.

[12] D. Gottesman, "Stabilizer codes and quantum error correction," Ph.D. dissertation, California Inst. of Technol., Pasadena, CA, USA, May 1997, doi: 10.48550/arXiv.quant-ph/9705052

[13] D. Chandra, Z. B. K. Egilmez, Y. Xiong, S. X. Ng, R. G. Maunder, and L. Hanzo, "Universal decoding of quantum stabilizer codes via classical guesswork," *IEEE Access*, vol. 11, pp. 19059–19072, 2023, doi: 10.1109/ACCESS.2023.3247966.

[14] A. Roque, D. Cruz, F. A. Monteiro, and B. C. Coutinho, "Efficient entanglement purification based on noise guessing decoding," *Quantum*, vol. 8, Sep. 2024, Art. no. 1476, doi: 10.22331/q-2024-09-19-1476.

[15] S. Santos, F. A. Monteiro, B. C. Coutinho, and Y. Omar, "Shortest path finding in quantum networks with quasi-linear complexity," *IEEE Access*, vol. 11, pp. 7180–7194, 2023, doi: 10.1109/ACCESS.2023.3237997.

[16] L. Bugalho, B. C. Coutinho, F. A. Monteiro, and Y. Omar, "Distributing multipartite entanglement over noisy quantum networks," *Quantum*, vol. 7, p. 920, Feb. 2023, doi: 10.22331/q-2023-02-09-920.

[17] B. C. Coutinho, W. J. Munro, K. Nemoto, and Y. Omar, "Robustness of noisy quantum networks," *Commun. Phys.*, vol. 5, no. 1, pp. 1–9, Apr. 2022, doi: 10.1038/s42005-022-00866-7.

[18] A.-K. Wu, L. Tian, B. C. Coutinho, Y. Omar, and Y.-Y. Liu, "Structural vulnerability of quantum networks," *Phys. Rev. A*, vol. 101, May 2020, Art. no. 052315, doi: 10.1103/PhysRevA.101.052315.

[19] G. Q. AI, "Suppressing quantum errors by scaling a surface code logical qubit," *Nature*, vol. 614, no. 7949, pp. 676–681, Feb. 2023, doi: 10.5281/zenodo.6804040.

[20] D. F. Locher, L. Cardarelli, and M. Müller, "Quantum error correction with quantum autoencoders," *Quantum*, vol. 7, p. 942, Mar. 2023, doi: 10.22331/q-2023-03-09-942.

[21] J. Roffe, "Quantum error correction: An introductory guide," *Contemporary Phys.*, vol. 60, no. 3, pp. 226–245, Jul. 2019, doi: 10.1080/00107514.2019.1667078.

[22] Z. Babar et al., "Duality of quantum and classical error correction codes: Design principles and examples," *IEEE Commun. Surv. Tut.*, vol. 21, no. 1, pp. 970–1010, Firstquarter 2019, doi: 10.1109/COMST.2018.2861361.

[23] W. Brown and O. Fawzi, "Short random circuits define good quantum error correcting codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Istanbul, Turkey, Jul. 2013, pp. 346–350, doi: 10.1109/ISIT.2013.6620245.

[24] J. Nelson, G. Bentsen, S. T. Flammia, and M. J. Gullans, "Fault-tolerant quantum memory using low-depth random circuit codes," *Phys. Rev. Res.*, vol. 7, Jan. 2025, Art. no. 013040, doi: 10.1103/PhysRevResearch.7.013040.

[25] A. K. Fedorov, N. Gisin, S. M. Beloussov, and A. I. Lvovsky, "Quantum computing at the quantum advantage threshold: A down-to-business review," Mar. 2022, *arXiv:2203.17181*, doi: 10.48550/arXiv.2203.17181.

[26] M. J. Gullans, S. Krastanov, D. A. Huse, L. Jiang, and S. T. Flammia, "Quantum coding with low-depth random circuits," *Phys. Rev. X*, vol. 11, no. 3, Sep. 2021, Art. no. 031066, doi: 10.1103/PhysRevX.11.031066.

[27] G. Smith and J. A. Smolin, "Degenerate quantum codes for pauli channels," *Phys. Rev. Lett.*, vol. 98, Jan. 2007, Art. no. 030501, doi: 10.1103/PhysRevLett.98.030501.

[28] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, "Topological quantum memory," *J. Math. Phys.*, vol. 43, no. 9, pp. 4452–4505, Sep. 2002, doi: 10.1063/1.1499754.

[29] I. B. Djordjevic, *Quantum Information Processing, Quantum Computing, and Quantum Error Correction: An Engineering Approach*, 2nd ed. Oxford, U.K.: Academic Press, 2021, doi: 10.1016/C2019-0-04873-X.

[30] P. Iyer and D. Poulin, "Hardness of decoding quantum stabilizer codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 9, pp. 5209–5223, Sep. 2015, doi: 10.1109/TIT.2015.2422294.

[31] M.-H. Hsieh and F. Le Gall, "NP-hardness of decoding quantum error-correction codes," *Phys. Rev. A*, vol. 83, May 2011, Art. no. 052331, doi: 10.1103/PhysRevA.83.052331.

[32] 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 its Appl.*, Honolulu, HI, USA, 2012, pp. 208–211. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6400919

[33] S. Aaronson and D. Gottesman, "Improved simulation of stabilizer circuits," *Phys. Rev. A*, vol. 70, no. 5, Nov. 2004, Art. no. 052328, doi: 10.1103/PhysRevA.70.052328.

[34] P. Das, S. Tannu, S. Dangwal, and M. Qureshi, "Adapt: Mitigating idling errors in qubits via adaptive dynamical decoupling," in *Proc. 54th Annu. IEEE/ACM Int. Symp. Microarchitecture*, New York, NY, USA, 2021, pp. 950–962, doi: 10.1145/3466752.3480059.

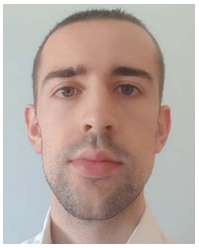
Diogo Cruz received the B.Sc. and M.Sc. degrees in physics engineering from Instituto Superior Técnico, University of Lisbon, Lisbon, Portugal, in 2019, where he is currently working toward the Ph.D. degree in physics.

He is a Researcher with Instituto de Telecomunicações, Lisbon. He was a Calouste Gulbenkian Scholar in 2018/2019.

Francisco A. Monteiro (Member, IEEE) received the Licenciatura and M.Sc. degrees in electrical and computer engineering from Instituto Superior Técnico (IST), University of Lisbon, Lisbon, Portugal, and the Ph.D. degree from the University of Cambridge, Cambridge, U.K.

He is currently an Associate Professor with the Department of Information Science and Technology, ISCTE—University Institute of Lisbon, Lisbon, and a Researcher with Instituto de Telecomunicações, Lisbon. He was a Teaching Assistant with the IST, University of Lisbon. He held visiting research positions with the University of Toronto, Toronto, ON, Canada, University of Lancaster, Lancaster, U.K., University of Oulu, Oulu, Finland, and University of Pompeu Fabra, Barcelona, Spain.

Dr. Monteiro was the recipient of the two best paper prizes awards at IEEE conferences (2004 and 2007), a Young Engineer Prize (3rd place) from the Portuguese Engineers Institution (Ordem dos Engenheiros) in 2002, and for two years in a row Exemplary Reviewer Awards from the IEEE Wireless Communications Letters (in 2014 and in 2015). He co-edited the book "*MIMO Processing for 4G and Beyond: Fundamentals and Evolution*," published by CRC Press in 2014. In 2016 he was the Lead Guest Editor of a special issue on Network Coding of the *EURASIP Journal on Advances in Signal Processing*. He was a General Chair of ISWCS 2018 - The 15th International Symposium on Wireless Communication Systems, an IEEE major conference in wireless communications.



André Roque received the B.Sc. and M.Sc. degrees in applied mathematics and computation from Instituto Superior Técnico, University of Lisbon, Lisbon, Portugal, in 2023.

Since 2023, he has been a Research Assistant with the Physics of Information and Quantum Technologies Group, Instituto de Telecomunicações, Lisbon.

Bruno C. Coutinho received the B.Sc. and M.Sc. in physics, from the University of Aveiro, Aveiro, Portugal, in 2009 and 2011, respectively, and the Ph.D. degree in physics from Northeastern University, Boston, MA, USA, in 2016.

Since 2017, he has been with the Physics of Information and Quantum Technologies Group, Instituto de Telecomunicações, Lisbon, Portugal, initially as a Postdoctoral, and later as a Research Fellow. In 2025, he joined the Institute of Communications and Navigation, German Aerospace Center, Weßling, Germany.