The cost of universality: A comparative study of the overhead of state distillation and code switching with color codes

Michael E. Beverland,[1] Aleksander Kubica,[2,3] and Krysta M. Svore[1]

[1] *Microsoft Quantum and Microsoft Research, Redmond, WA 98052, USA*
[2] *Perimeter Institute for Theoretical Physics, Waterloo, ON N2L 2Y5, Canada*
[3] *Institute for Quantum Computing, University of Waterloo, Waterloo, ON N2L 3G1, Canada*

(Dated: November 16, 2020)

Optimizing and estimating the overhead of existing fault tolerance schemes is a crucial step toward realizing scalable quantum computers. Many of the most promising schemes are based upon two-dimensional (2D) topological codes such as the surface and color codes, with a universal gate set consisting of readily implementable Cliffords along with the much more costly T-gate. In our work, we compare the cost of fault-tolerantly implementing the T-gate in color codes using two leading approaches: state distillation and code switching. We report that despite being treated less favorably in our analysis, state distillation is more resource-efficient than code switching, in terms of both qubit overhead and space time overhead. In particular, we find a T-gate threshold via code switching of 0.07% under circuit noise, almost an order of magnitude below that for distillation. To arrive at this result, we implement an end-to-end code switching simulation and accomplish many intermediate goals. For instance: (i) we optimize the 2D color code for circuit noise yielding it's largest threshold to date 0.037(1)% and (ii) we adapt and optimize the restriction decoder and find a threshold of 0.8% for the 3D color code with perfect measurements under Z noise. We not only find numerical estimates of the overhead of the explicit code switching protocol, but also lower bound the overhead with various conceivable improvements.

CONTENTS

I. MOTIVATION AND SUMMARY OF RESULTS

Recent experimental progress in demonstrating operational quantum devices [1–5] has brought us into the noisy intermediate-scale quantum era [6], where low-depth algorithms can be run on a small number of qubits. However, to handle the cumulative effects of noise and faults as these systems are scaled, fault-tolerant (FT) schemes [7–15] will be needed to reliably implement universal quantum computation. FT schemes encode logical information into many physical qubits, and run gadgets to continually diagnose faults and implement logical operations on the encoded information. FT requires additional resources, and much of the current quantum error correction (QEC) research is dedicated toward developing FT schemes with low overhead.

The choice of FT scheme used to realize universal quantum computing has important ramifications. Good schemes can significantly enhance the power of the resulting quantum computer with given quantum hardware. Moreover, as different FT schemes can vary in their sensitivity to hardware features such as qubit quality [16] and connectivity, the choice of FT scheme will influence many hardware decisions which could be costly to alter. At the base of most FT schemes is a QEC code which (given the capabilities and limitations of a given hardware) should: (i) tolerate realistic noise (ii) have an efficient classical decoding algorithm to correct faults, and (iii) admit a FT universal gate set. In the search of good FT schemes we focus our attention on QEC codes which are known to achieve most of these points with low overhead.

Topological codes are particularly compelling as they typically exhibit high accuracy thresholds with QEC protocols involving geometrically local quantum operations and efficient decoders; see e.g. Refs. [15, 17–31]. Two-dimensional (2D) topological codes such as the toric code [32, 33] and the color code [34] are particularly appealing for superconducting [35–37] and Majorana [38, 39] hardware, where qubits are laid out on a plane and quantum operations are limited to those involving nearby qubits.

The FT implementation of logical gates with 2D topological codes poses some challenges. The simplest FT logical gates are applied transversally, i.e. by independently addressing individual physical qubits. This is automatically FT since they do not grow the support of errors. Sadly, finding a QEC code which admits a universal set of transversal logical gates is ruled out by the Eastin-Knill theorem [40–42]. Furthermore, in 2D topological codes such gates can only perform Clifford operations [43–45], which are insufficient for universal quantum computation [46]. There are, however, many other innovative approaches to achieve universality, which typically focus on implementing non-Clifford logical gates [47–49], which achieve universality when combined with the Clifford gates.

The standard approach to achieve universality with 2D topological codes is known as *magic state distillation* [50–52]. It relies on first producing many noisy encoded copies of a *magic state* $|\overline{T}\rangle = (|\overline{0}\rangle + e^{i\pi/4}|\overline{1}\rangle)/\sqrt{2}$, and then processing them using Clifford operations to output a high fidelity encoded version of the state. The high-fidelity magic T state can then be used to implement the non-Clifford $\overline{T} = \mathrm{diag}(1, e^{i\pi/4})$ gate. Despite significant recent improvements, the overhead of distillation is expected to be large in practice [35, 53]. An exciting alternative that has been recently proposed is *code switching* via gauge fixing [54–57] to a 3D topological code which has a transversal T gate. The experimental difficulty of moving to 3D architectures could potentially be justified if it significantly reduces the overhead compared to state distillation. To compare these two approaches and find which is most practical for implementing in hardware, a detailed study is required.

In our work, we estimate the resources needed to prepare high-fidelity magic T states encoded in the 2D color code, via either magic state distillation or code switching. We assume that both approaches are implemented using quantum-local operations [58] in 3D, i.e., quantum operations are noisy and geometrically local, whereas classical operations can be performed globally and

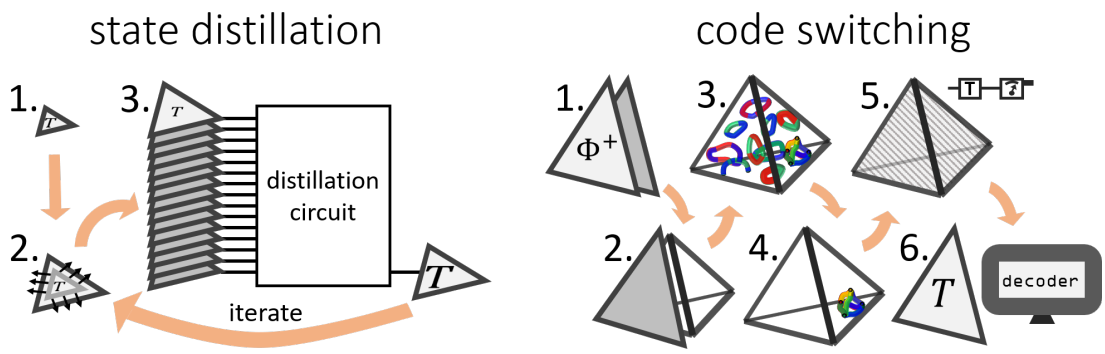


FIG. 1. Two methods of preparing high-fidelity magic T states encoded in the 2D color code:state distillation and code switching. Both approaches are implemented using quantum-local operations in 3D, i.e., noisy quantum operations are geometrically local, whereas ideal classical operations can be performed globally. In distillation, we produce many noisy encoded T states, and then run a Clifford distillation circuit. In code switching, we switch from 2D to 3D, and then implement the transversal T gate in the 3D color code.

perfectly (although they must be computationally efficient). In particular, we perform end-to-end simulations of these two approaches by implementing them with noisy circuits built from single-qubit state preparations, unitaries and measurements, and two-qubit unitaries between nearby qubits. For distillation, this 3D setting allows a stack of 2D color code patches, whereas for code switching it allows to implement the 3D color code; see Fig. 1. We then seek to answer the following question: *to prepare T states of a given fidelity, are fewer resources required for magic state distillation or code switching?*

Our main finding is that magic state distillation is more resource-efficient than code switching, in terms of both qubit overhead and space time overhead; see Fig. 2. To strengthen this separation we treat distillation much less favorably than code-switching in our analysis. More specifically, we consider only the standard 15-to-1 distillation scheme without much careful optimization. On the other hand we carefully optimize each step of code switching, and also investigate the effects of replacing each step by an optimal version to account for potential improvements. Even in the highly-optimistic scenario in which all the potential improvements can be realized, the overhead of code switching is still greater than that of distillation. To arrive at the main result, we accomplish many intermediate goals, which we elaborate on below.

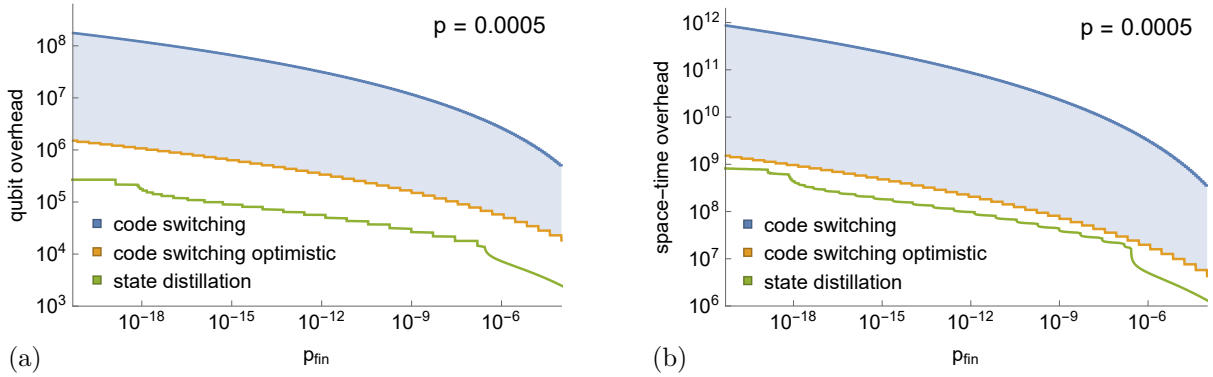


FIG. 2. The comparison of (a) the qubit and (b) space-time overhead as a function of the infidelity p_{fin} of the output T state for state distillation and code switching. Possible future improvements of any steps of our code switching protocol would be included within the shaded region. For other values of the circuit noise strength p the results are qualitatively the same; see Fig. 31.

2D color code performance.—In Section III, we first adapt the projection decoder [30] to the setting where the 2D color code has a boundary and syndrome extraction is imperfect, as well as optimize the stabilizer extraction circuits. We find a circuit noise threshold greater than 0.037(1)%, which is the highest to date for the 2D color code, narrowing the gap to that of the surface code. We also analyze the noise equilibration process during logical operations in the 2D color code and provide an effective logical noise model.

Noisy distillation analysis.—Using the effective logical noise model, we carefully analyse the overhead of distillation in Section IV. We strengthen the performance bounds by explicitly calculating the effect of failures at each location in the logical Clifford distillation circuits rather than simply counting the total number of locations [35, 53, 59–61]. We remark that we make use of the stack of 2D color codes to implement some logical operations faster than can be done strictly 2D approaches such as lattice surgery. The circuit-level threshold for distillation is equal to that of the 2D color code, which is 0.037(1)%.

Further insights into 3D color codes.—In Section V, we provide a surprisingly direct way to switch between the 2D color code and the 3D subsystem color code. Our method exploits the fact that for a particular gauge-fixing of the 3D subsystem color code, the code state admits a local tensor product structure in the bulk and can therefore be prepared in constant time. We also adapt the restriction decoder [31] to the setting where the 3D color code has a boundary and optimize it, which results in a threshold of 0.8% and a better performance for small system sizes.

End-to-end code switching simulation.—Section VI is the culmination of our work, where building upon results from the previous sections we provide a simplified and detailed description of every step of the code switching protocol. In our simulation, we exploit the special structure of the 3D subsystem color code to develop a method of propagating noise through the T gates in the system, despite the believed computational hardness of simulating general circuits with many qubits and T gates. We numerically find the failure probability of implementing the T gate with code switching as a function of the code distance and the circuit noise strength, which, in turn, allows us to estimate the T gate threshold to be 0.07%. We not only find numerical estimates of the overhead of the fully specified protocol, but also bound the minimal overhead of a code switching protocol with various conceivable improvements, such as using optimal measurement circuits, and optimal classical algorithms for decoding and gauge fixing of the 3D color code.

We feel this work constitutes a pioneering comparative study of the overhead of state distillation and code switching. More generally, we feel that careful end-to-end analyses with this level of detail will become increasingly important to identify the most resource-efficient FT schemes and, in turn, to influence the evolution of quantum hardware. Although our study focuses on color codes, we see no reason that our main finding, i.e., that state distillation is more resource-efficient than code switching, would not hold true for other topological codes such as the toric code [49]. Furthermore, we believe that state distillation will not be outperformed by code switching exploiting either 2D subsystem codes [62–64] or emulation of a 3D system with a dynamical 2D system [65–67] since these schemes must operate under even more constraints than when 3D quantum-local operations are allowed. We remark that there are other known fault-tolerant techniques for implementing a universal gate set [12, 68–72], however they are not immediately applicable to large-scale topological codes. Nevertheless, we are hopeful that there are still new and ingenious FT schemes to be discovered that could dramatically reduce the overhead and hardware requirements for scalable quantum computing.

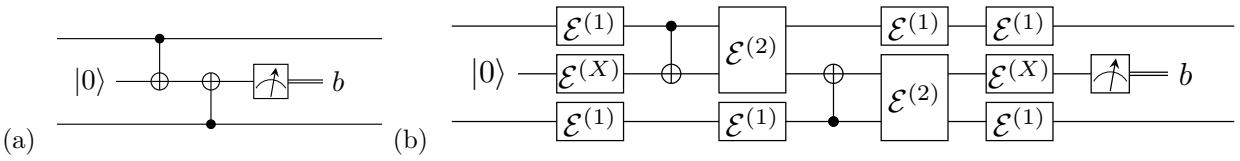


FIG. 3. (a) An example of an ideal circuit to measure weight-two operator ZZ. We assume that every gate, as well as preparation and single-qubit measurements take one time unit. (b) The noisy circuit is modeled with ideal gates followed by the depolarizing channel of strength p, ideal preparation can fail and produce a state orthogonal to the desired one with probability p, and the outcome of the ideal measurement can be flipped with probability p.

II. BACKGROUND AND NOTATION

In this section we review some relatively standard but important background material that we will refer to throughout the paper. In Section II A we describe the noise models and simulation approaches that we use to analyze and simulate distillation and code switching. In Section II B we provide some basic information about the color codes. In Section II C we review how to implement logical Clifford operations using 2D color codes. Finally in Section II C we review magic state distillation.

A. Noise and simulation

The noise model we use throughout the paper is the *depolarizing channel*, which on single- and two-qubit density matrices $\rho^{(1)}$ and $\rho^{(2)}$ has the action

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

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

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

We consider three standard scenarios,

1. **Depolarizing noise.**—Error correction is implemented with perfect measurements following each time-unit, during which single-qubit depolarizing noise of strength p acts. This is often referred to in the literature as the code capacity setting.

2. **Phenomenological noise.**—Error correction is implemented with perfect measurements following each time-unit, during which single-qubit depolarizing noise of strength p acts. However the measurement outcome bits are flipped with probability p.

3. **Circuit noise.**—Measurements are implemented with the aid of ancilla qubits and a sequence of one- and two-qubit operations as depicted in Fig. 3(a). Each operation experiences depolarizing noise of strength p.

In circuit noise, we approximate every noisy gate, e.g., Pauli X, Y and Z operators, the Hadamard gate H, the phase gate T, the controlled-not gate CNOT, and the idle gate I, by an ideal gate followed by the depolarizing channel on qubits acted on by the gate see Fig. 3. Preparations of the state orthogonal to that intended occur with probability p, and measurement outcome bits are flipped with probability p. We assume that all the elementary operations take the same time, which we refer to as one *time unit*.

For each of the three noise models, we will use *error rate* and *noise strength* interchangeably to describe the single parameter p.

We assume a special form of noise on magic states, which is justified as follows. Consider an arbitrary single-qubit state

$$\rho = \rho_{00}|T\rangle\langle T| + \rho_{01}|T\rangle\langle T^\perp| + \rho_{10}|T^\perp\rangle\langle T| + \rho_{11}|T^\perp\rangle\langle T^\perp|, \tag{3}$$

written in the orthonormal basis $\{|T\rangle, |T^\perp\rangle = Z|T\rangle\}$. Now consider a 'twirling operation' consisting of randomly applying one of the Cliffords A, A^2, A^3 and $A^4 = I$, each with probability 1/4, where $A \propto e^{i\pi/2}|T\rangle\langle T| + e^{-i\pi/2}|T^\perp\rangle\langle T^\perp|$. These single-qubit Clifford gates can be implemented instantaneously and perfectly by a frame update*. The state is transformed as follows

$$\rho \mapsto \frac{1}{4}\sum_{k=1}^{4} A^k \rho (A^k)^\dagger = \rho_{00}|T\rangle\langle T| + \rho_{11}|T^\perp\rangle\langle T^\perp|. \tag{4}$$

This can be used to justify our assumption that the noisy $|T\rangle$ state is of the form $\rho = (1-q)|T\rangle\langle T| + q|T^\perp\rangle\langle T^\perp|$, or equivalently that each $|T\rangle$ state is afflicted by a Z error with probability q. Due to this simplified form of the noise, we use *infidelity* interchangeably with the noise rate and noise strength to refer to the single parameter q when describing errors on magic states, since the infidelity of a noisy state σ with respect to a pure state $|\psi\rangle$ is $1 - \langle\psi|\sigma|\psi\rangle$.

For the purpose of defining pseudo-thresholds later, we find it useful to define the physical error probability $p_{\text{phy}}(t)$ for a time t as the probability that a single physical qubit will have a non-trivial operator applied to it over t time units under this noise model. It can be calculated as follows

$$p_{\text{phy}}(t) = 1 - \sum_{P_1 P_2 \ldots P_t = I} \Pr(P_1)\Pr(P_2)\ldots\Pr(P_t), \tag{5}$$

where $\Pr(I) = 1-p$ and $\Pr(X) = \Pr(Y) = \Pr(Z) = p/3$. For example $p_{\text{phy}}(1) = p$, $p_{\text{phy}}(2) = 2p(1-p) + 2p^2/3$, etc. Note that $\lim_{t\to\infty} p_{\text{phy}}(t) = 3/4$.

To simulate noise in our simulations, for Clifford circuits, we track the net Pauli operator which has been applied to the system by noisy operations using the standard binary symplectic representation. When non-Clifford operations are involved, we use modified techniques which are explained throughout the text.

To estimate statistical uncertainty of any quantity of interest ξ we use the bootstrap technique, i.e., we repeat sampling from the existing data set $\mathcal{I} = \{I_1, \ldots, I_A\}$ to evaluate $\xi = \xi(\mathcal{I})$. In particular, for $i = 1, \ldots, A$ we (i) randomly choose A data points $I_{i(j)}$ from the data set $\mathcal{I}$, where $i(j) \in \{1, \ldots, A\}$, (ii) evaluate the quantity $\xi_i = \xi(\mathcal{I}_i)$ using the data set $\mathcal{I}_i = \{I_{i(1)}, \ldots, I_{i(A)}\}$. We remark that the same data point can be chosen multiple times in step (i). We then estimate the quantity of interest to be

$$\xi = \hat{\xi} \pm \sqrt{\sum_{i=1}^{A} \frac{(\hat{\xi} - \xi_i)^2}{A-1}}, \qquad \hat{\xi} = \frac{1}{A}\sum_{i=1}^{A} \xi_i. \tag{6}$$

* Since we assume throughout that arbitrary one- and two-qubit pysical operations are allowed, single-qubit Clifford physical operations can actually be done 'offline' by tracking the basis of each physical qubit, and modifying future operations to be applied to that qubit accordingly. They are therefore perfect and instantaneous, as they only involve classical processing. Moreover, we will later see that logical Clifford operations can be done offline in 2D color codes such that the logical noise on encoded magic states can also be twirled offline.

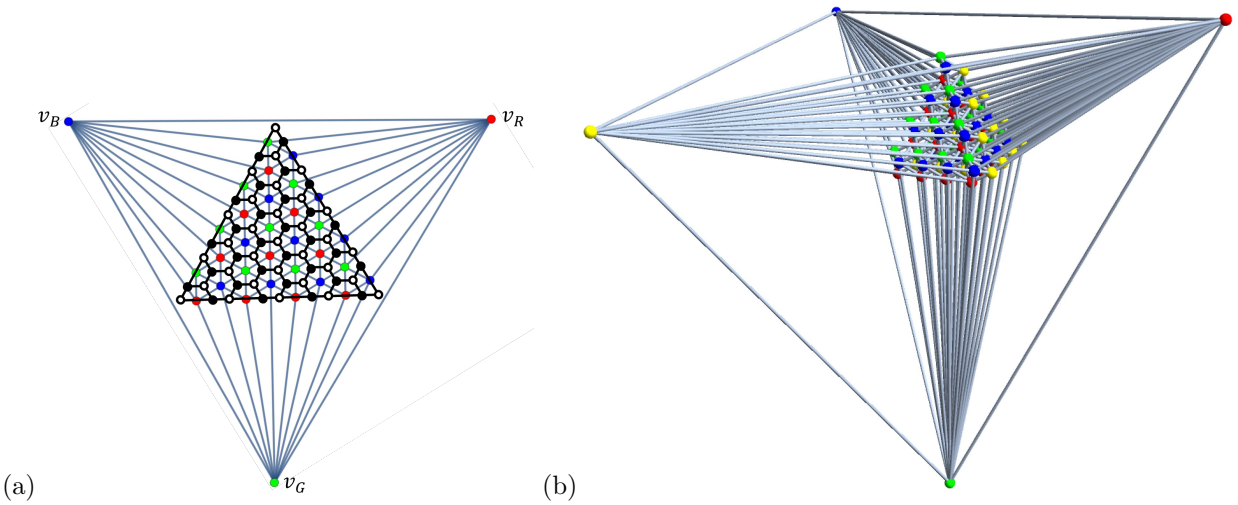


FIG. 4. An illustration of color code lattices $\mathcal{L}_{2D}$ and $\mathcal{L}_{3D}$ for $d = 9$; see Section VI A for details. (a) The lattice $\mathcal{L}_{2D}$ (gray edges with R, G and B vertices) and the corresponding primal lattice $\mathcal{L}^*_{2D}$ (black edges, black and white vertices). Qubits are places at triangles in $\mathcal{L}_{2D}$ (vertices in $\mathcal{L}^*_{2D}$), whereas X- and Z-stabilizer generators correspond to interior vertices in $\mathcal{L}_{2D}$ (faces in $\mathcal{L}^*_{2D}$). (b) For the lattice $\mathcal{L}_{3D}$, qubits are identified with tetrahedra, whereas X- and Z-stabilizer generators correspond to interior vertices and interior edges, respectively. Note that $\mathcal{L}_{2D}$ can be obtained from $\mathcal{L}_{3D}$ by retaining only those vertices connected to the boundary vertex v_Y along with the edges and faces only containing those vertices.

Note that when reporting estimated values, we will use a digit in parenthesis to indicate the standard deviation of the preceding digit. For example, we write $0.021(3)$ in place of 0.021 ± 0.003.

B. Basics of 2D and 3D color codes

Here we briefly review some important features of color codes, focusing on two and three dimensions. We also specify the lattices and some notation we use throughout the paper. For a more complete review of the topics covered in this subsection, see Refs. [56, 73, 74].

Color codes are very general families of topological quantum error-correcting codes defined on D-dimensional lattices composed of D-simplices with $(D+1)$-colorable vertices, where $D \geq 2$. Recall that $0, 1, 2, 3$-simplices are vertices, edges, triangles and tetrahedra, respectively. Qubits are placed on D-simplices of the lattice; X- and Z-type gauge generators are on $(D-2-z)$- and $(D-2-x)$-simplices, and X- and Z-type stabilizer generators are on x- and z-simplices, where $x, z \geq 0$ and $x + z \geq D - 2$. We say an operator is on a k-simplex when its support comprises all the D-simplices containing that k-simplex.

In this paper, we focus on color codes on two particular (families of) lattices: a two-dimensional triangular lattice with a triangular boundary $\mathcal{L}_{2D}$, and a three-dimensional bcc lattice with a tetrahedral boundary $\mathcal{L}_{3D}$; see Fig. 4. We will often refer to $\mathcal{L}_{2D}$ as the *triangular lattice* and $\mathcal{L}_{3D}$ as the *tetrahedral lattice*. Most of the time we rely on context and drop the subscript, simply writing $\mathcal{L}$. Members of these lattice families are parameterized by the code distance $d = 3, 5, 7, \ldots$ of color codes defined on them. We typically work with the dual lattice $\mathcal{L}$, but occasionally make use of the primal lattice $\mathcal{L}^*$. We remark that the graph constructed from the vertices and edges of the primal lattice $\mathcal{L}^*$ is bipartite [73]; see Fig. 4(a).

Given a lattice $\mathcal{L}$, it is useful to define $\Delta_k(\mathcal{L})$ as the set of all k-simplices in $\mathcal{L}$. We will abuse notation and write $\beta \in \Delta_b(\alpha)$ (or equivalently $\beta \subseteq \alpha$) to denote that a b-simplex β belongs to an a-simplex α, where $0 \leq b \leq a \leq D$. It is also useful to construct the color-restricted lattice $\mathcal{L}^{\mathcal{K}}$,

where $\mathcal{K}$ is a set of colors, by removing from $\mathcal{L}$ all the simplices, whose vertices have colors not only in $\mathcal{K}$. For example, the restricted lattice $\mathcal{L}^{RG}$ is obtained by keeping all the vertices of color R or G, as well as all the edges connecting them. We refer to the edges in $\mathcal{L}^{RG}$ as RG edges; similarly for other colors. We separate vertices in $\mathcal{L}$ into two types: *boundary vertices* (the three and four outermost vertices in Fig. 4(a) and (b), respectively), and all others which we call *interior vertices*. We call edges connecting two boundary vertices *boundary edges* (there are three and six boundary edges in Fig. 4(a) and (b), respectively), and call all other edges *interior edges*. More generally, we denote the sets of interior objects by $\Delta'_k(\mathcal{L})$, which is the set of all k-simplices in $\mathcal{L}$ containing at least one interior vertex

On the two-dimensional lattice, we define the stabilizer *2D color code* as follows. Qubits are on triangles, and both X- and Z-stabilizer generators are on interior vertices $\Delta'_0(\mathcal{L})$. This code has one logical qubit and string-like logical operators. One can implement the full Clifford group transversely on the 2D color code.

On the three-dimensional lattice, we can have either a stabilizer color code (with $x = 0$ and $z = 1$) or a subsystem color code (with $x = z = 0$). In both cases, there is one logical qubit and the physical qubits are on tetrahedra. For the *3D subsystem color code*, X- and Z-stabilizer generators are on interior vertices $\Delta'_0(\mathcal{L})$, while X- and Z-gauge generators are on interior edges $\Delta'_1(\mathcal{L})$. Recall that for subsystem codes, logical Pauli operators come in two flavors: *bare logical operators* which commute with the gauge generators, and *dressed logical operators* which commute with the stabilizer generators. In the 3D subsystem color code, bare and dressed logical operators are sheet- and string-like, respectively. One can implement the full Clifford group transversely on the 3D subsystem color code.

For the *3D stabilizer color code*, X- and Z-stabilizer generators are on interior vertices $\Delta'_0(\mathcal{L})$ and interior edges $\Delta'_1(\mathcal{L})$ respectively. The logical Pauli Z and X operators are string- and sheet-like, respectively. Crucially, the T gate is a transversal logical operator. To implement it, we split the n qubits in $\mathcal{L}$ into two groups, $(n+1)/2$ white tetrahedra and $(n-1)/2$ black tetrahedra, such that no two tetrahedra of the same color share a face. Applying T to white qubits and T^{-1} to black qubits implements the logical $\overline{T}$ gate. For notational convenience we write $\widetilde{T} = T^{\pm 1}$, determined by the color of the qubit.

Lastly, it is useful to introduce the notions of vector spaces associated with the constituents of the lattice $\mathcal{L}$ and linear maps between them. We define C_i to be a vector space over $\mathbb{F}_2$ with the set of i-simplices $\Delta_i(\mathcal{L})$ as its basis. Note that the elements of C_i are binary vectors and we can identify them with the subsets of $\Delta_i(\mathcal{L})$. For any $a, b \in [D] = \{0, 1, \ldots, D\}$ we define a generalized boundary operator $\partial_{a,b} : C_a \to C_b$, which is an $\mathbb{F}_2$-linear map specified on the basis element $\alpha \in \Delta_a(\mathcal{L})$ as follows

$$\partial_{a,b}\alpha = \begin{cases} \sum_{\beta \in \Delta_b(\mathcal{L}):\beta \supseteq \alpha} \beta & \text{if } a \leq b, \\ \sum_{\beta \in \Delta_b(\mathcal{L}):\beta \subset \alpha} \beta & \text{if } a > b. \end{cases} \tag{7}$$

We remark that the standard boundary operator $\partial_i : C_i \to C_{i-1}$ is a special case of the generalized boundary operator $\partial_{a,b}$ above if we choose $a = b + 1 = i$. These boundary maps are helpful in discussing error correction with the color code. In particular, since the color code is a CSS code, we can cast the decoding problem in terms of chain complexes, and treat X and Z errors and correction independently [30]. For the 2D color code, the boundary map $\partial_{2,0}$ allows us to find for any error configuration $\epsilon \subseteq \Delta_2(\mathcal{L}_{2D})$ its point-like syndrome via $\partial_{2,0}\epsilon$, where ϵ is the support of either X or Z error. For the 3D stabilizer color code, the syndromes of X and Z errors correspond to loop-like and point-like objects and can be found as $\partial_{3,1}\epsilon$ and $\partial_{3,0}\epsilon$, where $\epsilon \subseteq \Delta_3(\mathcal{L}_{3D})$ denotes the support of X and Z error, respectively. In Section V B we discuss in detail the structure of the loop-like gauge measurement outcomes for the 3D subsystem color code.

C. Fault-tolerant computation with 2D color codes

Here we briefly review an approach to implement quantum computation with a three-dimensional stack of the 2D color codes as in Fig. 5(a). Note that alternatively we could lay out the 2D color codes on a two-dimensional plane and use lattice surgery [75], although many of the elementary logical operations, e.g. CNOT gates, would be slower than for a stack.

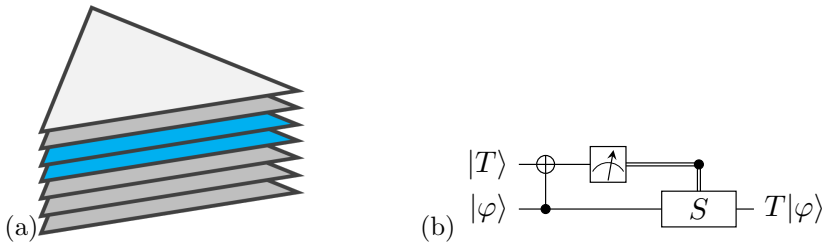


FIG. 5. (a) By stacking 2D color codes, one can implement transversal CNOT and SWAP operations between adjacent patches (colored in blue) with geometrically-local gates. (b) A T state can be used to implement a T gate via gate teleportation using only Clifford operations.

Error correction and decoding.—On each patch of the 2D color code, error correction (EC) is continuously performed. We use the term *EC cycle* to refer to a full round of stabilizer extraction circuits producing a syndrome σ, i.e., the set of measured stabilizer generators with outcome -1.

In a scenario in which measurements are performed perfectly given some noise E on the system, a *perfect measurement decoder* is used to infer a correction C given the input $\sigma(E)$ which will return the system to the code space if applied. The perfect measurement decoder fails if and only if CE is a non-trivial logical operator.

In a scenario in which measurements are not perfect, we consider a sequence of EC cycles $t = 1, \dots n_{\text{cyc}}$, with error E_t applied after each, and observed syndrome σ_t for each. For simplicity we assume that the first and last round have no additional error and that the syndrome is measured perfectly. Then a *perfect measurement decoder* is a decoder with takes the full history of syndromes $\sigma_1, \sigma_2, \dots \sigma_{n_{\text{cyc}}}$ and outputs a correction C such that $CE_1 \dots E_{n_{\text{cyc}}}$ has trivial syndrome. The faulty measurement decoder fails if and only if $CE_1 \dots E_{n_{\text{cyc}}}$ is a non-trivial logical operator. We discuss decoders for the 2D color code in detail in Section III.

Logical operations.—The elementary logical operations for the 2D color codes are implemented as follows.

- *State preparation.* To prepare the $|\overline{0}\rangle$ state each data qubit is prepared in $|0\rangle$, then d rounds of EC are performed, and the X-type syndrome outcomes σ at the first round are inferred (d rounds are needed to do so fault-tolerantly). A Z-type fixing operator with the syndrome σ is applied. The $|\overline{+}\rangle$ state is prepared analagously.

- *Idle operation.* A single EC cycle is performed.

- *Single-qubit Clifford gates.* As the gates are transversal, these are done in software by Pauli frame updates so are instantaneous and perfect.

- *CNOT.* This is implemented by the application of a transversal CNOT between adjacent patches in the the stack, and followed by $n_{\text{cyc}}^{\text{after}}$ EC cycles; see Section III D for details.

- *SWAP.* This is implemented by the consecutive application of three transversal CNOTs with alternating control and target qubits on adjacent patches, and followed by a single EC cycle.

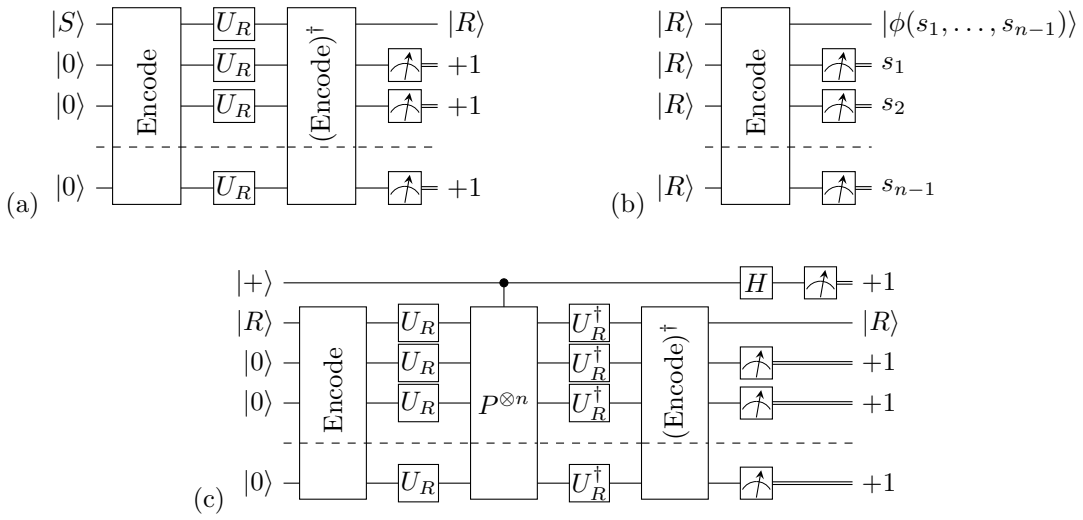


FIG. 6. The three known types of distillation schemes for a resource state $|R\rangle$. Each wire represents a logical qubit of a QEC scheme that fault-tolerantly performs all Clifford gates. (a) We need a code with a transversal non-Clifford gate U_R, such that $U_R|S\rangle = |R\rangle$ for some stabilizer state $|S\rangle$. (b) We need a code with a transversal gate C_R, such that $C_R|R\rangle = |R\rangle$. The measurement of the stabilizers of the code will be probabilistic even when $|R\rangle$ input is noiseless, with post-selected state $|\phi(+1,\ldots,+1)\rangle = |R\rangle$. (c) We need a code with a transversal gate C_R, such that $C_R|R\rangle = |R\rangle$ and $C_R = U_R P U_R^\dagger$ for some Pauli operator P.

- *Measurement.* The readout of single-qubit measurements in the $\overline{X}$ and $\overline{Z}$ basis is implemented by measuring each data qubit in the X and Z basis. The output bit string is then processed in two stages: (1) a perfect measurement decoder is run to correct the bit string such that it satisfies all stabilizers, and then (2) the outcome is read off from the corrected bit string by from the parity of the bit string restricted to the support of any representative of the logical operator.

The above list consists of Clifford operations, which are not by themselves universal for quantum computation. In addition, we consider the following non-Clifford gate.

- *T gate.* It is implemented using gate teleportation of an encoded magic state $|\overline{T}\rangle = (|\overline{0}\rangle + e^{i\pi/4}|\overline{1}\rangle)/\sqrt{2}$ as shown in Fig. 5(b). We will consider the production of the encoded magic state $|\overline{T}\rangle$ by both distillation and code switching.

D. Magic State Distillation

Here we briefly review magic state distillation [50, 51], in which Clifford operations are used with post selection to convert noisy resource states into fewer but crucially less noisy resource states. In our analysis of the overhead of magic state distillation in Section IV, we consider only the standard 15-to-1 distillation protocol. However, given the wide range of distillation protocols that have been proposed in recent years, we take the opportunity to attempt to consolidate the concepts behind them here. In particular, we outline three classes of distillation protocol, which to our knowledge include all known proposals up to small variations. We then go on to describe a family of protocols recently introduced by Haah and Hastings [76], for which the 15-to-1 protocol that we consider is an special instance. In our discussion here we will assume Clifford operations are perfect, although we will relax that assumption in Section IV.

Let C_R be an operator stabilizing a resource state $|R\rangle$, i.e., $C_R|R\rangle = |R\rangle$, and $U_R^\dagger$ be a non-Clifford unitary transforming $|R\rangle$ into some stabilizer state $|S\rangle$, i.e., $U_R^\dagger|R\rangle = |S\rangle$. We will refer to C_R and U_R as the *resource stabilizer* and *resource rotator*, respectively, and throughout when we write that U_R should be applied, it can be implemented using $|R\rangle$ by gate teleportation as in Fig. 5(b). The three classes of distillation protocols are then summarized as follows (and depicted Fig. 6).

(a) *Code projector then resource rotator* [50, 76–79]. We use a code with a transversal logical gate $\overline{U_R} = U_R \otimes U_R \otimes \cdots \otimes U_R$. We prepare the (noisy) logical state $|\overline{R}\rangle$ by first encoding the logical stabilizer state $|\overline{S}\rangle$ and then implementing the transversal gate $\overline{U_R}$ (using n copies of the noisy resource state $|R\rangle$). Unencoding $|\overline{R}\rangle$ and postselecting on the $+1$ measurement outcomes of the stabilizers of the code gives a distilled resource state with infidelity reduced from q to $\mathcal{O}(q^d)$ for code distance d.

(b) *Resource stabilizer then code projector* [50, 80]. We use a code with a transversal logical gate $\overline{C_R} = C_R \otimes C_R \otimes \cdots \otimes C_R$. We start with n noisy resource states $|R\rangle^{\otimes n}$, which by definition satisfy $\overline{C_R}|R\rangle^{\otimes n} = |R\rangle^{\otimes n}$, then measure the stabilizers of the code. If all measurement outcomes are $+1$, then the state $|\overline{R}\rangle$ has been prepared, which can then be further decoded to yield a distilled resource state $|R\rangle$ with infidelity suppressed from q to $\mathcal{O}(q^d)$. Note that this approach seems less promising than the other two since the probability of successful post-selection is less than one even for perfect resource states.

(c) *Code projector then resource stabilizer* [51, 52, 81–85]. We use a code with a transversal logical gate $\overline{C_R} = C_R \otimes C_R \otimes \cdots \otimes C_R$ and assume that $U_R C_R U_R^\dagger = P$ is a Pauli operator.[†] First, we encode a resource state $|R\rangle$ with infidelity q_1 in the code, giving $|\overline{R}\rangle$, and then measure the logical operator $\overline{C_R}$ and post-select on the $+1$ outcome. To measure $\overline{C_R}$ we use a measurement gadget consisting of the following three steps. First, apply $U_R \otimes U_R \otimes \cdots \otimes U_R$ using n noisy $|R\rangle$ states, each with infidelity q_2. Second, apply the Clifford gate control-$(P \otimes P \otimes \cdots \otimes P)$, controlled by an ancilla state $|+\rangle$. Third, apply $U_R^\dagger \otimes U_R^\dagger \otimes \cdots \otimes U_R^\dagger$ using another n noisy $|R\rangle$ states, each with infidelity q_2. After this gadget, the stabilizers are checked, and if all are satisfied, then the encoded state is decoded and kept as a distilled resource state. The output has infidelity suppressed to $\mathcal{O}(q_1^2) + \mathcal{O}(q_2^d)$, but the suppression with respect to q_1 can be boosted by, for example, repeating the measurement gadget.

Historically, the first distillation protocol was proposed by Knill [51] (type c), which takes 15 input T states of infidelity q to produce an output of infidelity $35q^3$, with acceptance probability $1 - 15q$. Shortly after, Bravyi and Kitaev [50] proposed two schemes, the first of which is type a in our classificiation, but which has the same parameters as Knill's (later the two schemes were shown to be mathematically equivalent [86]). The second scheme in Ref. [50] is of type b and has less favorable parameters than the 15-to-1 scheme, but a higher distillation threshold. Another type b scheme was found by Reichardt [80], which could successfully distill states arbitrarily close to the stabilizer polytope. In Ref. [78], type a schemes outputting multiple resource states were proposed. Multiple outputs were also achieved for type b schemes in Refs. [82, 85]. These 'high rate k/n' schemes have promising parameters and may perform well in certain regimes. However, they also tend to have large Clifford circuits likely inhibiting their practicality when including the effect of realistically noisy Clifford operations. Recently, Haah and Hastings introduced yet another family of distillation protocols of type (a). These are based on puncturing quantum Reed-Muller

[†] If the unitary U_R is in the third level of the Clifford hierarchy, then C_R is a Clifford operator, but by using this approach with a code which implements a non-Clifford transversal gate C_R, distillation for higher-order schemes should be possible.

codes and have both interesting asymptotic properties [79] and appear to be practically favorable [76]. We review this family here, which includes the 15-to-1 scheme that we analyze in detail in Section IV.

Haah-Hastings distillation protocols.— This family of distillation protocols involves first producing what we will call a *Reed-Muller state* $|\mathrm{QRM}_r\rangle$ on $n = 2^{3r+1}$ qubits, where $r = 1, 2, \ldots$, with a Clifford circuit of depth $3r + 1$ using $(3r + 1)2^{3r}$ CNOTs; see Fig. 7. This state has the property that for any subset of k qubits, the state can be decomposed as a set of k Bell pairs between those 'punctured' qubits and those which remain, namely

$$|\mathrm{QRM}_r\rangle = \prod_{i=1}^{k} \left(\frac{|0\rangle_i|\overline{0}\rangle_i + |1\rangle_i|\overline{1}\rangle_i}{\sqrt{2}} \right), \tag{8}$$

where the states of the k punctured qubits have no bar, and where $\{|\overline{0}\rangle_i, |\overline{1}\rangle_i\}_{i=1,\ldots,k}$ are logical basis states for a $[[n-k, k, d_Z]]$ CSS code which is triply-even (therefore transversal T implements logical T on each logical qubit of the code), with Z-distance d_Z (i.e. the weight of the smallest non-trivial Z-type logical operator). To specify the CSS code, which we call the punctured Reed-Muller code, one can start with the X- and Z-type stabilizer generators of the $|\mathrm{QRM}_r\rangle$ state (which, for example, could be specified by propagating the initial single-qubit stabilizers through the CNOT circuit in Fig. 7). Choosing a stabilizer generator set for $|\mathrm{QRM}_r\rangle$ in such a way that only one X- and one Z-type generator has non-trivial support on each punctured qubit, these form logical operators and the remaining generators with no support on any punctured qubits form the stabilizer generators of the punctured Reed-Muller code.

The next step of the protocol is to apply a (noisy) T gate to each non-punctured qubit,

$$T^{\otimes(n-k)}|\mathrm{QRM}(r)\rangle = \prod_{i=1}^{k} \left(\frac{T|+\rangle_i \otimes |\overline{+}\rangle_i + T|-\rangle_i \otimes |\overline{-}\rangle_i}{\sqrt{2}} \right). \tag{9}$$

To see this, note that $(I \otimes \overline{T})\frac{|0\overline{0}\rangle+|1\overline{1}\rangle}{\sqrt{2}} = (T \otimes I)\frac{|0\overline{0}\rangle+|1\overline{1}\rangle}{\sqrt{2}} = (T \otimes I)\frac{|+\overline{+}\rangle+|-\overline{-}\rangle}{\sqrt{2}}$. Finally, one measures the logical $\overline{X}_i$ operators of the CSS code. If the measurement outcome of $\overline{X}_i$ is $+1$, then the ith punctured qubit is left in the state $T|+\rangle$; otherwise, the ith punctured qubit is in the state $T|-\rangle = ZT|+\rangle$, requiring a simple Pauli Z fix. In practice this is achieved by measuring all $n - k$ non-punctured qubits in the X-basis, and post-processing. Note that post-processing the measurement outcomes also tells us what the X-stabilizer generators for the CSS code are, which should all be $+1$ in the absence of error. This, in turn provides, a postselection condition of the protocol. The scheme takes $n-k$ magic states of infidelity q, and ouptuts k magic states of infidelity Aq^{d_Z}, where A is the number of non-trivial Z-type logical operators of minimal weight d_Z.

In Fig. 7 we show an instance of the Haah-Hastings protocol for $r = 1$, which we analyze more explicitly in Section III. To our knowledge, this instance was first shown in Ref. [35], and is essentially a modification of the 15-to-1 Bravyi-Kitaev protocol, which avoids the need of the unencoding part of the circuit in Fig. 6(a). Some larger instances of this family have very good properties for distillation, although we do not analyze them in detail here.

III. PERFORMANCE OF 2D COLOR CODES

In this section, we find an efficient and optimized implementation of the 2D color code under circuit noise. First, in Section III A we adapt Delfosse's color code projection decoder [30] to allow for boundaries in the lattice. Then, we further adapt the decoder to accommodate faulty measurements in Section III B, optimize the stabilizer extraction circuits in Section III C and find

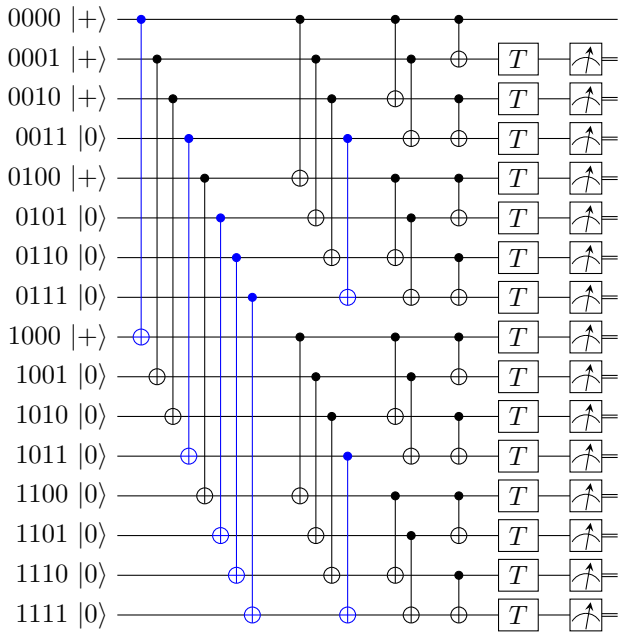


FIG. 7. A 15-to-1 distillation circuit as the $r = 1$ instance of the Haah-Hastings construction. Each qubit is labeled by a string of $3r+1$ bits and if the bit string has weight at most r, then that qubit is prepared in the $|+\rangle$ state, otherwise in the $|0\rangle$ state. In the jth round of CNOT gates, we apply CNOTs between the pairs of qubits with bit string differing only at the jth position. Note that the blue CNOTs are redundant as their control or target qubits are $|0\rangle$ or $|+\rangle$, respectively. The resulting state is $|\mathrm{QRM}_1\rangle \propto |+\rangle|\overline{+}\rangle + |-\rangle|\overline{-}\rangle$, where in our case $|\overline{+}\rangle$ and $|\overline{-}\rangle$ are code states of the 15-qubit Reed-Muller code. We then apply the transversal $\overline{T}$ gate and measure every non-punctured qubit in the X basis to infer the measurement outcomes m_i's for X-stabilizers and $m_{\overline{X}}$ for the logical $\overline{X}$. The protocol rejects if $m_i = -1$ for some i. If there are no faults, then the the remaining unpunctured qubit is in the state $T^{3-2m_{\overline{X}}}|+\rangle$.

a circuit noise threshold of 0.37(1)%. This represents a significant improvement over the previous highest threshold value for the color code of 0.2% in [37], and brings it closer to the toric code threshold of near[‡] 1%. However, we stress that we are primarily interested in the finite-size rather than asymptotic performance, and therefore take care to accurately account for effects that are often ignored when focusing on threshold alone, such as residual error. We believe the performance improvements we see over previous studies of the color code under circuit noise are due to our use of the hexagonal rather than the square-octagon primal lattice in the case of [88, 89], and by optimizing extraction circuits and removing the restriction on the qubit connectivity in the case of [37].

A. Projection decoder with boundaries

In this subsection, we briefly describe our adaptation of Delfosse's color code projection decoder [30] to the lattice $\mathcal{L}_{2\mathrm{D}}$, which has boundaries. Our adaptation is essentially the same as that presented in [89], but for the hexagonal rather than the square-octagon primal lattice. This decoder assumes that the stabilizer measurements are perfect, and we analyse its performance under depolarizing noise.

[‡] In Ref. [87], a threshold above 1% was reported, however follow-up work [37] failed to reproduce this result and reported 0.7%.

We consider the 2D color code on the lattice $\mathcal{L} = \mathcal{L}_{2D}$ from Section II B. Since the 2D color code is a self-dual CSS code, we can decode X- and Z-errors separately and describe here the correction of X-errors; the correction of Z-errors is identical. We denote the support of the X errors by $\epsilon \subseteq \Delta_2(\mathcal{L})$, the Z-type syndrome by $\sigma \subseteq \Delta_0'(\mathcal{L})$ and the resulting correction by $\hat{\epsilon} \subseteq \Delta_2(\mathcal{L})$. For each pair of colors $\mathcal{K} \in \{RG, RB, GB\}$ we define the set of highlighted vertices $V^{\mathcal{K}}$ to contain the subset $\sigma^{\mathcal{K}} \subseteq \sigma$ of all the vertices of color in $\mathcal{K}$ and we also include in $V^{\mathcal{K}}$ a boundary vertex v_K for only one color $K \in \mathcal{K}$ whenever $|\sigma^{\mathcal{K}}|$ is odd, i.e., $V^{\mathcal{K}} = \sigma^{\mathcal{K}}$ or $V^{\mathcal{K}} = \sigma^{\mathcal{K}} + v_K$. Note that by definition $|V^{\mathcal{K}}|$ is even.

The *Projection Decoder* (see Fig. 8) can then be described as follows.

1. For each pair of colors $\mathcal{K} \in \{RG, RB, GB\}$ we use the Minimum Weight Perfect Matching algorithm to find a subset of edges $E^{\mathcal{K}} \subseteq \Delta_1(\mathcal{L}^{\mathcal{K}})$ which connect pairs of highlighted vertices in $V^{\mathcal{K}}$ within the projected lattice $\mathcal{L}^{\mathcal{K}}$.

2. The combined edge set $E = E^{RG} + E^{RB} + E^{GB}$ separates the lattice $\mathcal{L}$ into two complementary regions $R_1 \subseteq \Delta_2(\mathcal{L})$ and $R_2 = \Delta_2(\mathcal{L}) \setminus R_1$ and we choose the correction $\hat{\epsilon}$ to be the smaller of the regions R_1 and R_2.

Note that step 1 can be viewed as the problem of decoding the toric code defined on the lattice $\mathcal{L}^{\mathcal{K}}$, and thus one could use any toric code decoder to find a pairing $E^{\mathcal{K}} \subseteq \Delta_1(\mathcal{L}^{\mathcal{K}})$. The Minimum-Weight Perfect Matching algorithm we chose for this step is computationally efficient. The boundary edges are permitted in $E^{\mathcal{K}}$, but their edge weight is set to zero for matching.

In Fig. 9(a) we present the failure probability of this decoder under depolarizing noise and perfect measurements. In Fig. 9(b) we consider two distance-dependent quantities that should both converge to the threshold as $1/d$ approaches zero. The first is the pseudo-threshold $p^*_{\text{dep}}(d)$, defined as a solution to $p_{\text{fail}}(p, d) = p$. The pseudo-threshold is a good proxy to identify the regime in which the code of distance d is useful. The second is the crossing $p^{\times}_{\text{dep}}(d)$ between pairs of failure curves of distances d and $(d+1)/2$. From a linear extrapolation of the data we find intercepts $0.1316(4)$ for $p^{\times}_{\text{dep}}(d)$ and $0.1208(4)$ for $p^*_{\text{dep}}(d)$.

Note that the discrepancy in the intercept values suggests that the systems we consider are too small for a naive linear extrapolation to work reliably. Assuming that both $p^*_{\text{dep}}(d)$ and $p^{\times}_{\text{dep}}(d)$ continue to change monotonically with d we then take the maximum observed value of $p^{\times}_{\text{dep}}(d)$ at $d = 29$ as a conservative estimate of the threshold under depolarizing noise, i.e., $p^*_{\text{dep}} = 0.1245(4)$, however we believe the true threshold value is likely to be higher.

This analysis of the pseudo-thresholds and failure-curve crossings highlight two general points. Firstly, the threshold may not be a good proxy to the finite-size performance. For example, we see in Fig. 9(b) that our estimate of the threshold is considerably above the observed pseudo-thresholds. Secondly, in order to find the threshold from the data for small systems one has to exert caution, as extrapolating different quantities (which nevertheless recover the same threshold value in the limit of infinite d) can result in inconsistent estimates.

B. Noisy-syndrome projection decoder with boundaries

In this subsection we further generalize the projection decoder to handle phenomenological noise. To reliably extract the syndrome and perform error correction one can repeat the stabilizer measurements multiple times. The input of the decoder then consists of stabilizer measurement outcomes (possibly incorrect) at EC cycles labeled by integers and can be visualized as a (2+1)-dimensional lattice $\Lambda = \mathcal{L} \times [n_{\text{cyc}}]$, where $[n_{\text{cyc}}] = \{0, 1, \ldots, n_{\text{cyc}}\}$ and the extra dimension represents time; see Fig. 10. We use a shorthand notation $\Lambda_{[t_1,t_2]}$ to denote the part of Λ within EC cycles

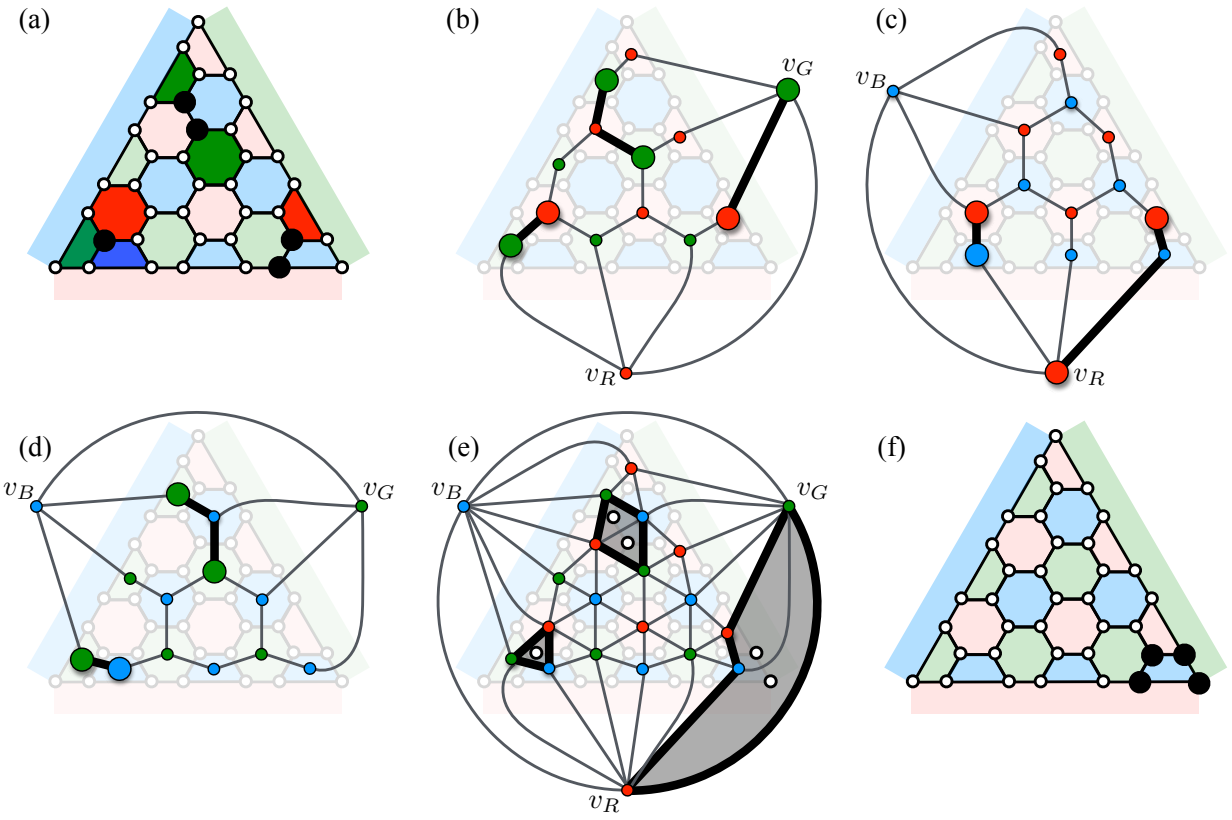


FIG. 8. The projection decoder for the 2D color code of distance $d = 7$ on the lattice $\mathcal{L}_{2D}$. (a) The primal lattice with errors (black dots) and the corresponding syndrome (highlighted faces). (b)-(d) We find pairings E^{RG}, E^{RB} and E^{GB} (thick lines) of highlighted vertices V^{RG}, V^{RB} and V^{GB} within the sublattices $\mathcal{L}^{RG}$, $\mathcal{L}^{RB}$ and $\mathcal{L}^{GB}$, respectively. Note that in (b) and (c) boundary vertices v_G and v_R are included as highlighted vertices. (e) The error estimate is the region (shaded) with boundary $E^{RG} + E^{RB} + E^{GB}$, which contains the minimal number of qubits (white dots). (f) The decoding succeeds since the initial error and the estimate differ by a stabilizer (black dots in the primal lattice).

t_1 and t_2. By an EC cycle, we simply mean a full cycle of stabilizer measurements. Temporal edges, which vertically connect corresponding vertices in two copies of $\mathcal{L}$ at EC cycles t and $t+1$, correspond to stabilizer measurements at EC cycle t.

We use the same concepts and nomenclature for the lattice Λ as for $\mathcal{L}$ in Section II B. For instance, we say a vertex (v, t) of Λ is a boundary vertex iff v is a boundary vertex of $\mathcal{L}$; otherwise, it is an interior vertex. An edge of Λ is a boundary edge iff it connects two boundary vertices. The sets of interior vertices and edges of Λ are denoted $\Delta_0'(\Lambda)$ and $\Delta_1'(\Lambda)$, respectively. Furthermore, we denote by $\Lambda^{\mathcal{K}}$ the sublattice of Λ consisting of vertices of color in $\mathcal{K}$, and the edges connecting them.

The input of the *noisy-syndrome projection decoder* is an observed syndrome σ consisting of the subset of temporal edges corresponding to -1 stabilizer measurement outcomes. We define the set of syndrome flips $V \subseteq \Delta_0'(\Lambda)$ to be the set of all the vertices, which are incident to an odd number of edges in σ. The decoder is then implemented using the following steps.

1. For each $t \in [n_{\text{cyc}}]$ initialize $F(t) = \emptyset$.

2. For every $\mathcal{K} \in \{RG, RB, GB\}$ use Minimum-Weight Perfect Matching to find the pairing $E^{\mathcal{K}}$ of syndrome flips $V^{\mathcal{K}}$ within sublattice $\Lambda^{\mathcal{K}}$.

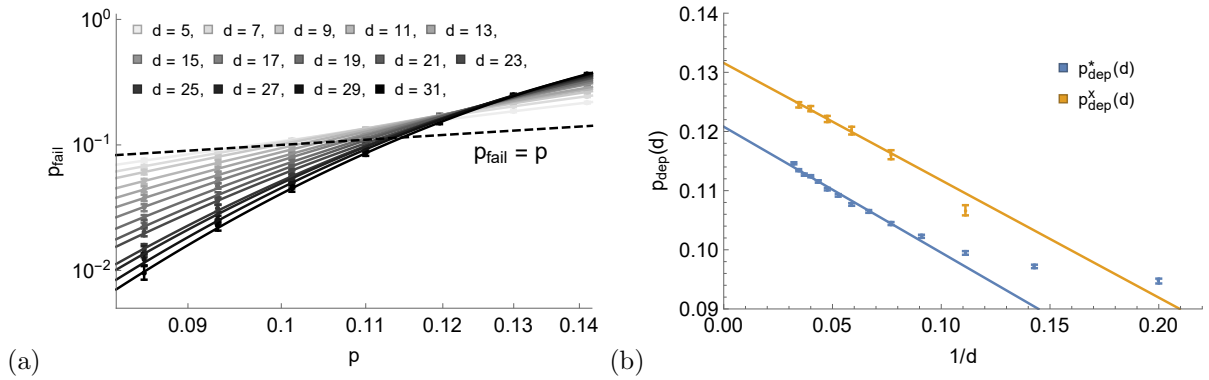


FIG. 9. (a) Performance of the projection decoder for the 2D color code of distance d under depolarizing noise and perfect measurements. (b) The pseudo-thresholds $p^*_{\rm dep}(d)$ and the crossings $p^\times_{\rm dep}(d)$ between pairs of curves corresponding to distances d and $(d+1)/2$ for various d. From a linear extrapolation of the data for $d \geq 11$ we obtain intercepts of 0.1316(4) for $p^\times_{\rm dep}(d)$ and 0.1208(4) for $p^*_{\rm dep}(d)$.

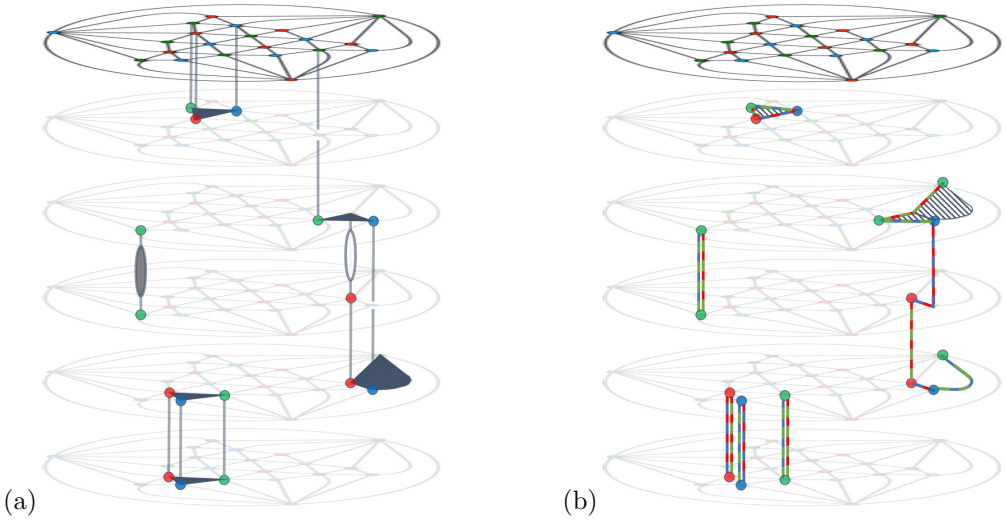


FIG. 10. The noisy-syndrome projection decoder on the (2+1)D color code lattice Λ. (a) During each EC cycle, errors can appear on data qubits (shaded triangles), followed by stabilizer measurements which when incorrect are drawn as ellipses. The observed syndrome σ (dark temporal edges and filled ellipses) allows us to find the set of syndrome flips V (highlighted vertices). (b) For $\mathcal{K} \in \{RG, RB, GB\}$ we find the pairing $E^{\mathcal{K}}$ (colored edges) of the syndrome flips $V^{\mathcal{K}}$ within the sublattice Λ. Then, for every "flattened" connected component of $E^{RG} + E^{RB} + E^{GB}$ we find a correction $\epsilon(t)$ (hatched regions).

3. Combine the obtained pairings, $E = E^{RG} + E^{RB} + E^{GB}$, and decompose E as a disjoint sum of maximal connected components, $E = \sum_i E_i$.

4. For every connected component E_i:

   (a) Find the minimal window of EC cycles $\tau_i = \left[t_i^{(1)}, t_i^{(2)}\right]$ enclosing E_i, i.e. $E_i \subset \Delta_1(\Lambda_{\tau_i})$.

   (b) Project E_i onto $\mathcal{L}$ in order to obtain the "flattened pairing" $F_i = \pi(E_i)$, where $\pi : \Delta_1(\Lambda) \to \Delta_1(\mathcal{L})$ removes temporal edges and adds horizontal ones modulo two.

   (c) Add the edges of F_i modulo two to the edge set $F\left(t_i^{(2)}\right)$ for EC cycle $t_i^{(2)}$.

5. For each EC cycle t, find a correction $\epsilon(t) \subseteq \Delta_2(\mathcal{L})$ as the minimal region enclosed by $F(t)$.

We make some additional technical remarks about the noisy-syndrome projection decoder. In step 2, the boundary edges of sublattice $\Lambda^{\mathcal{K}}$ are assigned zero weight when used for pairing. A boundary vertex of a color in $\mathcal{K}$ (it does not matter which) is added to $V^{\mathcal{K}}$ whenever $|V^{\mathcal{K}}|$ is odd. In step 3, we remove weight-zero edges when establishing connected components of E.

To analyze error correction thresholds in a faulty-measurement setting, it is common to study the somewhat contrived scenario of an initially perfect code state undergoing d rounds of error correction, followed by a single round of perfect measurements. The justification for this is underpinned by the fact that in the fault-tolerant setting, the logical clock cycle (the time required to implement logical gates) requires approximately d rounds of error correction with lattice surgery or braiding. Moreover, one would expect the effects of the artificially perfect preparation and final measurement round to be negligible over d rounds when d is sufficiently large, making this scenario appropriate for estimating the threshold value (but not for estimating the actual performance of finite sizes). In Fig. 11(a) we find the failure probability after d rounds of phenomenological noise. In Fig. 11(d) we show crossings $p^{\times}_{\mathrm{phe}}(d)$ between pairs of these failure curves corresponding to distances d and $(d+1)/2$, which should converge to the threshold p^{*}_{phe} as $1/d$ approaches to zero.

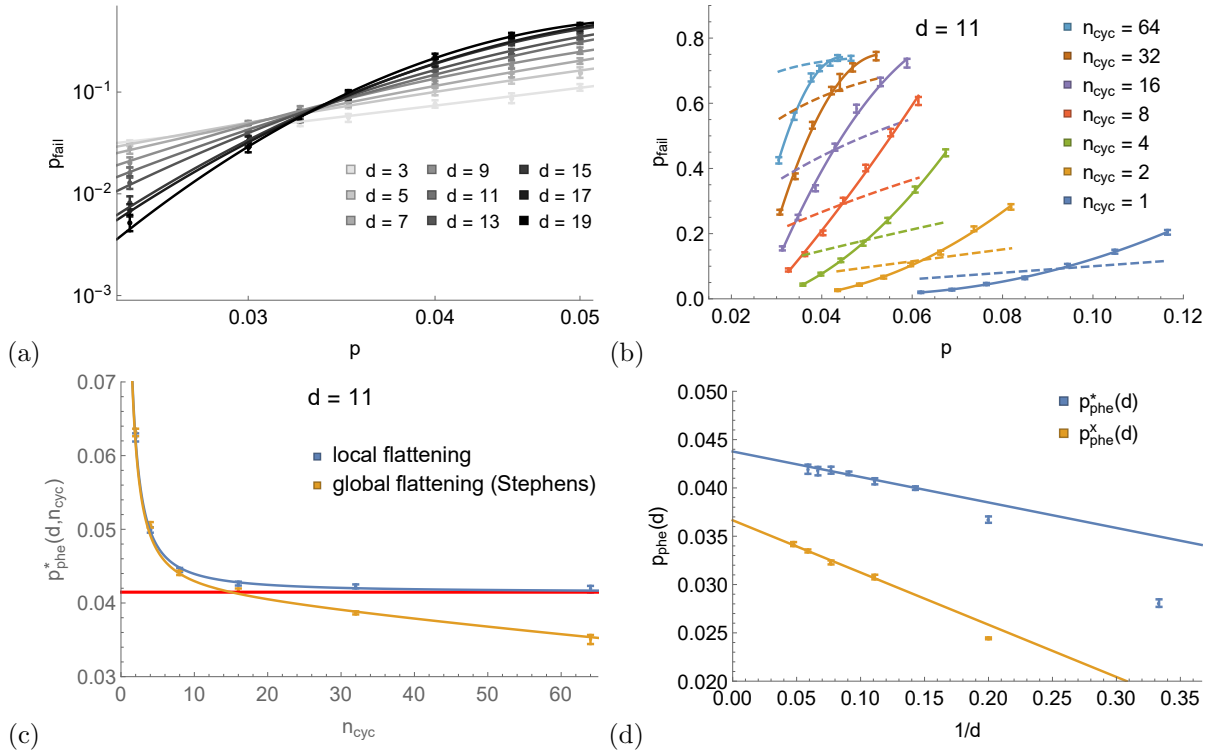


FIG. 11. (a) The failure probability $p_{\mathrm{fail}}(d, n_{\mathrm{cyc}})$ of the noisy-syndrome projection decoder for $n_{\mathrm{cyc}} = d$ rounds of phenomenological noise and various distances d. (b) We estimate the time-dependent pseudo-threshold $p^{*}_{\mathrm{phe}}(d, n_{\mathrm{cyc}})$ from the intersection of $p_{\mathrm{fail}}(d, n_{\mathrm{cyc}})$ with the error probability for the unencoded qubit (dashed) after n_{cyc} time units for various n_{cyc} and $d = 11$. (c) We estimate the long-time pseudo-threshold $p^{*}_{\mathrm{phe}}(d) = 0.0415(2)$ for $d = 11$ (red) by fitting $p^{*}_{\mathrm{phe}}(d, n_{\mathrm{cyc}})$ with the numerical ansatz in Eq. (10). Note that the Stehpens' adaptation of the decoder (yellow), which uses a global flattening, fails to stabilize. (d) The long-time pseudo-thresholds $p^{*}_{\mathrm{phe}}(d)$ (blue) and the crossings $p^{\times}_{\mathrm{phe}}(d)$ (yellow) between pairs of curves corresponding to distances d and $(d+1)/2$ for various d. From a linear extrapolation of the data for $d \geq 7$ we obtain intercepts of 0.0438(3) for $p^{*}_{\mathrm{phe}}(d)$ and 0.0367(3) for $p^{\times}_{\mathrm{phe}}(d)$.

The notion of a pseudo-threshold must be revisited in the setting of faulty measurements and we cannot extract a meaningful pseudo-threshold directly from these curves as we did for the perfect measurement case in Fig. 9. Consider the scenario in which we assume a perfect initial

code state and a perfect final measurement round, but consider the performance over a varying number n_{cyc} of noisy EC cycles; see Fig. 11(b). For each d and n_{cyc}, we define the time-dependent pseudo-threshold $p^*_{\mathrm{phe}}(d, n_{\mathrm{cyc}})$ as the error rate at which the encoded failure probability after n_{cyc} EC cycles matches $p_{\mathrm{phy}}(n_{\mathrm{cyc}})$ the unencoded failure probability for n_{cyc} time units, as defined in Eq. (5). As n_{cyc} increases, $p^*_{\mathrm{phe}}(d, n_{\mathrm{cyc}})$ is expected to decrease due to the buildup of residual error. However, for sufficiently large n_{cyc} the time-dependent pseudo-threshold $p^*_{\mathrm{phe}}(d, n_{\mathrm{cyc}})$ should eventually stabilize to the long-time pseudo-threshold $p^*_{\mathrm{phe}}(d)$, as can be seen in Fig. 11(c). The following ansatz

$$p^*(d, n_{\mathrm{cyc}}) = p^*(d) \left(1 - \left[1 - \frac{p^*(d,1)}{p^*(d)} \right] n_{\mathrm{cyc}}^{-\gamma} \right). \tag{10}$$

fits the data well, and allows us to extract an estimate of the long-time pseudo-threshold. We remark that our approach of finding long-time pseudo-thresholds is similar in spirit, but not exactly the same as the one used to find the "sustainable threshold" [90, 91]. Also, the ansatz in Eq. (10) was used in [92] to analyze thresholds of cellular-automata decoders for topological codes.

The long-time pseudo-thresholds $p^*_{\mathrm{phe}}(d)$, as well as the failure curve crossings $p^{\times}_{\mathrm{phe}}(d)$ should converge in the limit of infinite d to the threshold under phenomenological noise. From a linear extrapolation of the data we obtain intercepts of 0.0438(3) for $p^*_{\mathrm{phe}}(d)$ and 0.0367(3) for $p^{\times}_{\mathrm{phe}}(d)$; see Fig. 11(d). Note that $p^*_{\mathrm{phe}}(d)$ appears to converge more quickly than $p^{\times}_{\mathrm{phe}}(d)$. Assuming that both quantities continue to monotonically change with d we take the maximum observed value of $p^*_{\mathrm{phe}}(d)$ at $d = 17$ as a conservative estimate of the threshold under phenomenological noise, i.e., $p^*_{\mathrm{phe}} = 0.0419(4)$.

We remark that the modification of the projection decoder that we have presented here to handle noisy syndrome measurements differs from that discussed by Stephens [89] in an important detail. Namely, our "flattening" of pairings is local, since it occurs separately on each connected component after the matching in the (2+1)-dimensional graphs. In contrast, Stephens' flattening is global—all pairings are flattened together and a global correction is produced. For any finite noise strength and for a sufficiently large number of cycles (which does not need to grow with code distance), the success probability of this global flattening will be vanishingly small. Therefore the adaption proposed by Stephens does not have a finite error-correction threshold, and is not fault tolerant. This did not contribute noticeably to the numerical results presented in [89] since the total cycle number was fixed to d, for data in ranges satisfying $d < 1/p$. However, by looking at the performance over longer times of Stephens' adaption of the projection decoder, we verify that the time-dependent pseudo-threshold for this decoder fails to stabilize for large n_{cyc}; see Fig. 11(c).

C. Optimizing stabilizer extraction and circuit noise analysis

There is significant freedom in precisely which circuits are used to extract the syndrome for error correction. We will assume that there is a separate ancilla qubit per stabilizer generator, such that there are two ancillas per face of the lattice $\mathcal{L}_{2\mathrm{D}}$, and will not worry about the precise connectivity details, requiring only that coupled qubits are nearby. Each circuit starts by preparing an ancilla qubit in either $|+\rangle$ or $|0\rangle$ state, followed by applying CNOT gates between the ancilla qubit and all the qubits of the stabilizer, and finishes with measuring the ancilla qubit in the corresponding X- or Z-basis. During each time unit new errors can appear in the system and thus it can be beneficial to parallelize as much as possible the circuits used for stabilizer measurement. When circuits for measuring different stabilizers are interleaved, not all schedules of CNOT gates will work. The following conditions [88] must be satisfied:

- At each time unit at most one operation can be applied to any given qubit.

- The measurement circuit preserves the group generated by the elements of the stabilizer group and Pauli X or Z operators stabilizing the ancilla qubits.[§]

In our optimization we assume that it suffices to specify the CNOT ordering for a single X and Z stabilizer generator in the bulk, as the code is translation invariant. Moreover, the CNOT schedule for stabilizer generators along the boundary of the lattice are specified by restricting the schedule for those in the bulk; see Fig. 12.

The CNOT schedule includes twelve CNOT gates to extract both Z and X stabilizers. Each CNOT gate is applied at some time unit, and thus the CNOT schedule is specified by a list of twelve non-negative integers $\mathcal{A} = \{a, b, c, d, e, f; g, h, i, j, k, l\}$, possibly with repetitions. We are interested in CNOT schedules which satisfy the following condition:

1. **As short as possible.**— To ensure there is no time unit in which both ancillas in a face are idle $\mathcal{A} = \{a, b, c, d, e, f, g, h, i, j, k, l\}$ contains all numbers from 1 to $\max \mathcal{A}$.

Note that this implies that $\max \mathcal{A} \leq 12$, and thus there are at most 12^{12} CNOT schedules. However, most of them are invalid as they do not satisfy one of the following necessary conditions [88]:

2. **One operation per qubit at a time.**— The integers a, b, c, d, e, f must be all distinct, as well as g, h, i, j, k, l, and d, j, f, l, b, h, and e, k, a, g, c, i.

3. **Correct syndrome extraction.**— To ensure the ancilla measurement after each CNOT sequence extracts the stabilizer measurement outcome, the following inequalities must hold:

   – For the stabilizers in the bulk: $(a-g)(b-h)(c-i)(d-j)(e-k)(f-l) > 0$, $(e-g)(d-h) > 0$, $(k-a)(j-b) > 0$, $(f-h)(e-i) > 0$, $(l-b)(k-c) > 0$, $(d-l)(c-g) > 0$, and $(j-f)(i-a) > 0$.

   – For the stabilizers along the boundary: $(a-g)(b-h)(c-i)(f-l) > 0$, $(b-h)(c-i)(d-j)(e-k) > 0$ and $(a-g)(b-h)(c-i)(d-j) > 0$.

To illustrate how we obtain the inequalities in the last condition, let us analyze how the Pauli X operator stabilizing the X syndrome extraction ancilla on a given face is spread by the CNOT schedule. It propagates to all the data qubits on that face. From each data qubit it may further propagate to the Z syndrome extraction ancilla on the same face, and this is determined by the relative order of the CNOT gates used for the X and Z syndrome extraction. We need to ensure that at the end of the CNOT schedule the Pauli X operator on the X syndrome extraction ancilla has not propagated to the Z syndrome extraction ancilla, which is equivalent to the inequality $(a-g)(b-h)(c-i)(d-j)(e-k)(f-l) > 0$ being true. The other inequalities are derived similarly.

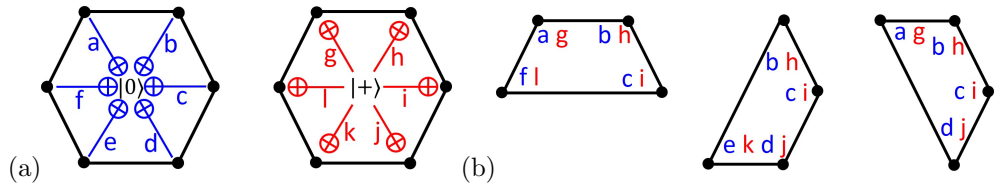


FIG. 12. (a) The CNOT schedule $\{a, b, c, d, e, f; g, h, i, j, k, l\}$ is a list specifying the time units when each CNOT gate is applied. At the end, the ancillas which are initially prepared in $|0\rangle$ and $|+\rangle$ are measured to extract the Z (blue) and X (red) stabilizer measurement outcomes, respectively. (b) The CNOT schedules for stabilizers on faces along the boundary can be viewed as restrictions of the CNOT schedule in (a).

[§] We say that an ancilla prepared in $|+\rangle$ or $|0\rangle$ state is stabilized by a Pauli X or Z operator, respectively.

We remove an ordering from the list of valid orderings if it equals another ordering in the list up to a rotation or a reflection. No schedules with six time units satisfy all these conditions. However, we find that there are 116, 2340, 18024 and 73136 valid orderings for 7, 8, 9 and 10 time units, respectively.

To select a good CNOT schedule among the large number of valid ones, we focus on the 116 shortest schedules, because fewer time units in an error correction round tends to translate into fewer possible faults and better performance. We tested each using a $d = 7$ code for d rounds of error correction with circuit noise of strength $p = 0.0035$ and estimated the failure probability by sampling. This value of p was chosen because preliminary studies indicated it was close to the threshold, and distance $d = 7$ was chosen as a compromise between reducing the simulation run time and limiting the impact of boundary effects. The limited sampling resources were focused on identifying the best CNOT schedules, see Fig. 13(a). It is difficult to definitively find the best length-7 CNOT schedule, but we found that $\{1, 4, 5, 6, 3, 2; 2, 3, 4, 7, 6, 5\}$ had logical error rate 0.128(1), which we expect to be at least close to optimal performance since no schedule was found to have logical error rate below 0.125 to within three standard deviations. For comparison, the worst-performing length-7 schedule was $\{1, 4, 3, 6, 7, 2; 6, 1, 2, 5, 4, 7\}$ and resulted in substantially worse logical failure probability 0.211(1).

Now we focus on the best-performing CNOT schedule under circuit noise and find the long-time pseudo-threshold for a range of distances in Fig. 13(b) using the same approach as in Section III B. Unlike in the cases for depolarizing and phenomenological noise, the data for circuit noise appears to be in the regime where it is well fit linearly and both $p^*_{\rm cir}(d)$ and $p^\times_{\rm cir}(d)$, and their intercepts agree to within error. We take their shared intercept as an estimate of the threshold under circuit noise $p^*_{\rm cir}(d) = 0.0037(1)$.

To estimate the impact of this kind of circuit optimization, we compare the best and worst performing length-7 CNOT schedules and found that the long-time pseudo-threshold differs by a factor of almost two for distance $d = 13$; see Section VI B for more details.

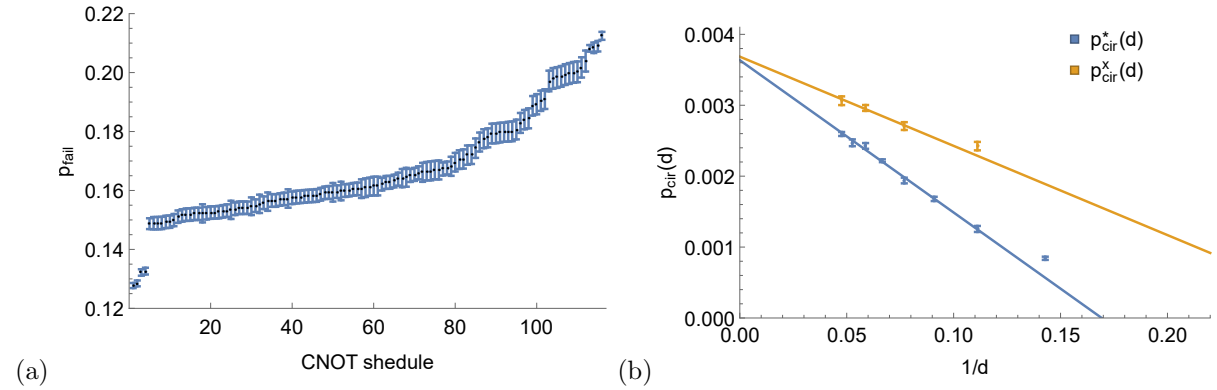


FIG. 13. (a) The logical failure probability over $n_{\rm cyc} = 7$ cycles for distance $d = 7$ and depolarizing circuit noise of strength $p = 0.0035$ for each of the 116 CNOT schedules, sorted by their failure rate. (b) The long-time pseudo-thresholds $p^*_{\rm cir}(d)$ and crossings $p^\times_{\rm cir}(d)$ for the best-performing CNOT schedule $\{1, 4, 5, 6, 3, 2; 2, 3, 4, 7, 6, 5\}$ under circuit noise; see Section VI B for further details. From a linear extrapolation of the data for $d \geq 11$ we obtain intercepts of 0.00366(6) for $p^*_{\rm cir}(d)$ and 0.0037(1) for $p^\times_{\rm cir}(d)$.

D. Modelling noise in logical operations

Now we describe an effective noise model for logical operations in the optimized 2D color code under circuit noise, which will later use to estimate the performance of distillation circuits. An

logical operation	failure probability $\overline{p}_{\text{oper}}(p,d)$
prep	$0.0151 \cdot 0.816^d$ for $p = 10^{-3}$
	$0.0074 \cdot 0.719^d$ for $p = 5 \cdot 10^{-4}$
	$0.0041 \cdot 0.527^d$ for $p = 10^{-4}$
idle	$0.0124 \cdot 0.721^d$ for $p = 10^{-3}$
	$0.0092 \cdot 0.618^d$ for $p = 5 \cdot 10^{-4}$
	$0.0018 \cdot 0.487^d$ for $p = 10^{-4}$
CNOT	$0.0972 \cdot 0.894^d$ for $p = 10^{-3}$
	$0.0603 \cdot 0.761^d$ for $p = 5 \cdot 10^{-4}$
	$0.0078 \cdot 0.607^d$ for $p = 10^{-4}$
meas	$0.0011 \cdot 0.690^d$ for $p = 10^{-3}$.

TABLE I. Failure probabilities of logical operations for the 2D color code as a function of the distance d. Since the failure probability for logical Pauli measurements is negligible, we assume measurements to be perfect in the effective noise model, and only report the measurement failure probability for $p = 10^{-3}$.

input to the effective noise model is the overall failure probability $\overline{p}_{\text{oper}}(p,d)$ of each logical operation oper $= \text{prep}, \text{idle}, \text{CNOT}$, which we estimate numerically and record in Table I in the form of the following ansatz

$$\overline{p}_{\text{oper}}(p,d) = \alpha(p)\beta(p)^d. \tag{11}$$

Note that Eq. (11) is a generalization of the standard ansatz $\alpha\,(p/p^*)^{(d+1)/2}$, where α and p^* are allowed to depend on p. In our simulations each operation is followed by a full decoding. This results in the application of a logical Pauli operator, which if non-trivial is interpreted as a failure. Our effective noise model is designed to overestimate the probability of each non-trivial logical Pauli, and assumes each operation fails independently of others. Specifically,

- for state preparation noise is modelled by preparing an orthogonal logical state to that intended with probability $\overline{p}_{\text{prep}}(p,d)$,
- for an idle operation, noise is modelled by applying $\overline{X}$ or $\overline{Z}$ each with probability $\overline{p}_{\text{idle}}(p,d)/2$ or $\overline{Y}$ with probability $\overline{p}_{\text{idle}}(p,d)/20$,
- single-qubit Cliffords are done in software by Pauli-frame updates so are noise-free,
- for SWAP, noise is modelled by applying to each of the two qubits $\overline{X}$ or $\overline{Z}$ each with probability $\overline{p}_{\text{idle}}(p,d)/2$, or $\overline{Y}$ with probability $\overline{p}_{\text{idle}}(p,d)/20$,
- for CNOT noise is modelled by applying $\overline{IX}$ or $\overline{ZI}$ each with probability $\overline{p}_{\text{CNOT}}(p,d)/2$, or $\overline{XI}$ or $\overline{IZ}$ each with probability $\overline{p}_{\text{CNOT}}(p,d)/4$, or other non-trivial Pauli each with probability $\overline{p}_{\text{CNOT}}(p,d)/20$,
- measurements in the logical Pauli bases are assumed to be perfect.

Let us make a number of remarks about this noise model. Firstly, the conservative estimates of different types of failure are very loose for large distances, and could be made tighter by fine-tuning the error model. Secondly, one may wonder why we treat the SWAP operation, which is built from three transverse CNOTs as a pair of idle qubits, whereas the CNOT alone has a considerable failure

probability. The reason is that the noise introduced by the physical CNOT gates themselves is negligible, but the propagation of previously existing error (which is exchanged between patches by SWAP, but added across patches by CNOT) can be very significant.

In the remainder of this section, we describe how we estimate the logical failure probabilities using distance-d patches of 2D color code under circuit noise of strength p followed by a perfect-measurement decoding. To avoid the influence of boundary effects in these simulations, we ensure that the patches are close to EC equilibrium both before and after the logical operation by implementing d rounds of EC on the patches prior to the operation and checking that the residual noise has stabilized before the perfect measurement is implemented. By EC equilibrium, we mean a regime in which the failure probability linearly with the number of EC cycles, as can be seen in Fig. 16. Throughout our analysis we use the best-performing CNOT schedule $\{1, 4, 5, 6, 3, 2; 2, 3, 4, 7, 6, 5\}$, which uses one ancilla per stabilizer generator. This results in $(3d^2 - 1)/2$ qubits being needed for a distance-d patch.

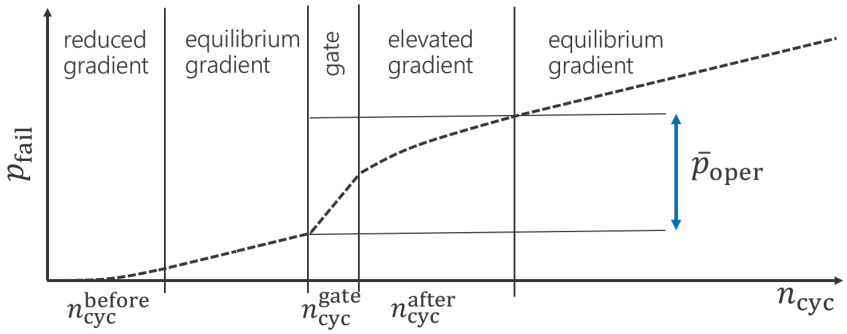


FIG. 14.

Logical state preparation.— Recall that $|\overline{0}\rangle$ is prepared by initializing data qubits in $|0\rangle$ and then measuring the stabilizers for d rounds under circuit noise of strength p, and applying a Z-type operator intended to fix the X stabilizers. We simulate this followed by the perfect-measurement decoder and estimate the failure probability from the proportion of trials in which the a logical $\overline{X}$ is applied. The analogous procedure is used to identify the failure probability for preparing $|\overline{0}\rangle$, The data is presented in Fig. 15(a) and (b) with a fit to the ansatz in Eq. (11) using the same parameters. This fit provides the entry for $\overline{p}_{\text{prep}}(p, d)$ in Table I.

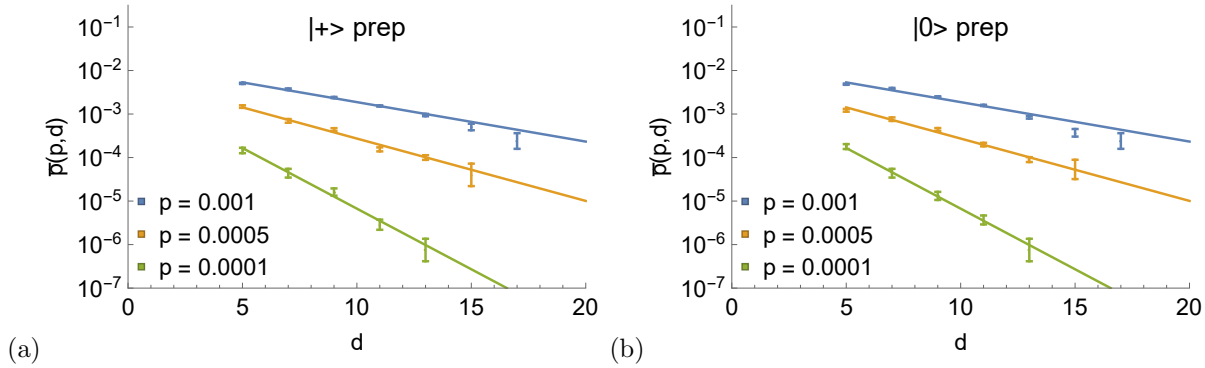


FIG. 15. The logical failure probability for preparing the $|\overline{+}\rangle$ and $|\overline{0}\rangle$ states. The ansatz in Eq. (11) is fitted, giving $\overline{p}_{\text{prep}}(p, d)$ in Table I. Both plots are fit well using the same parameters, which justifies treating both $|\overline{+}\rangle$ and $|\overline{0}\rangle$ preparation identically in the noise model.

Idle logical qubit.— We extract $\overline{p}_{\text{idle}}(p, d)$, the failure probability of an idle logical qubit for

a single EC round, by considering the failure probability of a single patch of distance-d 2D color code over n_{cyc} rounds of EC, given a perfect initial state and a perfect final round of stabilizer extraction. After a few initial rounds of error correction, we expect the residual error in the system to reach an equilibrium, and that thereafter the failure rate should increase linearly with n_{cyc} in the regime of small $p_{\text{fail}}(d, n_{\text{cyc}})$. We can use linear growth of $p_{\text{fail}}(d, n_{\text{cyc}})$ with time as a hallmark of the system being in EC equilibrium. To estimate $\overline{p}_{\text{idle}}(p, d)$ for fixed p and d, we fit the failure probability $p_{\text{fail}}(d, n_{\text{cyc}})$ data, such as that presented in Fig. 16(a) for $p = 0.001$, with the following linear function of n_{cyc}, i.e.,

$$p_{\text{fail}}(d, n_{\text{cyc}}) = \overline{p}_{\text{idle}}(p, d) \cdot n_{\text{cyc}} + c(p, d), \tag{12}$$

where $c(p, d)$ is a constant. We plot this and the fit in Fig. 16(a) for a particular value of p, and use the gradient to estimate $\overline{p}_{\text{idle}}(p, d)$ for various p and d in Fig. 16(b). We fit Eq. (11) to the data, giving the entry in Table I. Note that in Fig. 16(a) the y-intercept is negative since the simulation begins with a perfect code state and so the probability of failure during the early rounds is artificially reduced. For later logical operations we will assume that d rounds of EC prior to the logical operation is sufficient to ensure the system has reached EC equilibrium.

In our noise model, we claim that $\overline{p}_{\text{idle}}(p, d)/2$, $\overline{p}_{\text{idle}}(p, d)/2$ and $\overline{p}_{\text{idle}}(p, d)/20$ are conservative estimates of the probability of $\overline{X}$, $\overline{Z}$ and $\overline{Y}$ respectively. In Fig. 16(c) and (d), we justify this claim by finding the fraction of failures which occur for each Pauli operator. Despite the symmetry of the depolarizing noise, we observe that $\overline{Y}$ failures are much less frequent than $\overline{X}$ or $\overline{Z}$ failures due to the independent detection and decoding of X and Z errors.

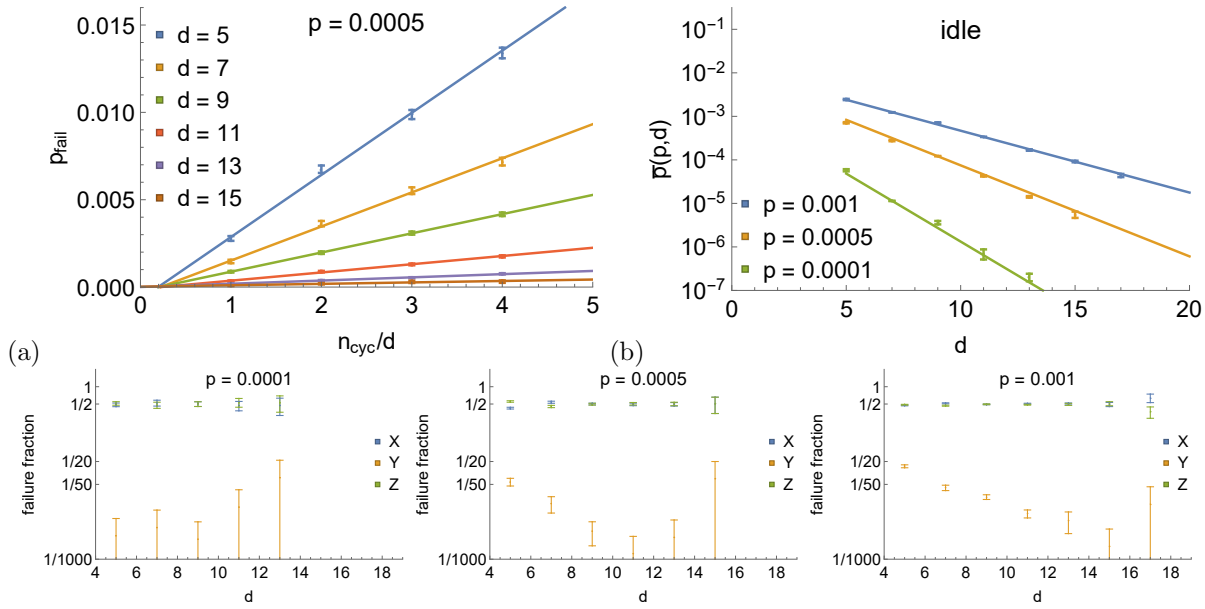


FIG. 16. Analysis of the logical idle operation. (a) The failure probability using the best-performing CNOT schedule as a function of n_{cyc}/d for $p = 0.0005$. We use the ansatz in Eq. (12) to find the logical failure rate $\overline{p}(p, d)$. We observe that EC equilibration occurs by the dth round; see Section VI B for additional data. Not discarding the initial equilibration period would result in an underestimate of the failure probability. (b) For each value of p, we use the ansatz in Eq. (11) to extract the parameters for $\overline{p}_{\text{idle}}(p, d)$ in Table I. (bottom panels) The fraction of overall failures corresponding to different logical Pauli errors. The points whose data range falls below the horizontal axis correspond to no observed failures.

Transverse logical CNOT.— The logical CNOT is implemented transversely between a pair of 2D color code patches. We assume the system is in EC equilibrium before the gate,

which is ensured in the simulation by running d rounds of EC on an initially error-free state. Immediately after the CNOT gates are applied, the system's residual noise is elevated since the CNOT propagates X errors from the control to the target and Z errors from the target to the control. To allow the system to return to EC equilibrium, we include $n_{\text{cyc}}^{\text{after}}$ rounds of EC after the gate is applied in the logical CNOT operation. We conclude from Fig. 17(a) that $n_{\text{cyc}}^{\text{after}} = 2$ is sufficient, which we then incorporate into the logical CNOT operation. We analyse the overall failure probability of the logical CNOT in Fig. 17(b), and the fraction of failures resulting in each logical Pauli in the panels at the bottom of Fig. 17. Note that in contrast with the phenomenon observed in Fig. 16(a) in which the initial system equilibrated from a state of lower noise, here the system equilibrates from a state of higher noise.

Note that it is important that decoding for each patch is implemented with the combined syndrome history—since the CNOT propagates X errors from the first patch to the second one, we add the Z-type stabilizer history from the first d rounds of EC from the first patch to that of the second before decoding X errors in the second patch, and similarly for Z errors in the first patch. To isolate the contribution to the logical operator from the CNOT alone, we find and remove the contribution to the logical operator from the initial d rounds of EC by applying a perfect-measurement decoding on the system immediately after the d rounds, and propagate that through the logical CNOT gate[¶].

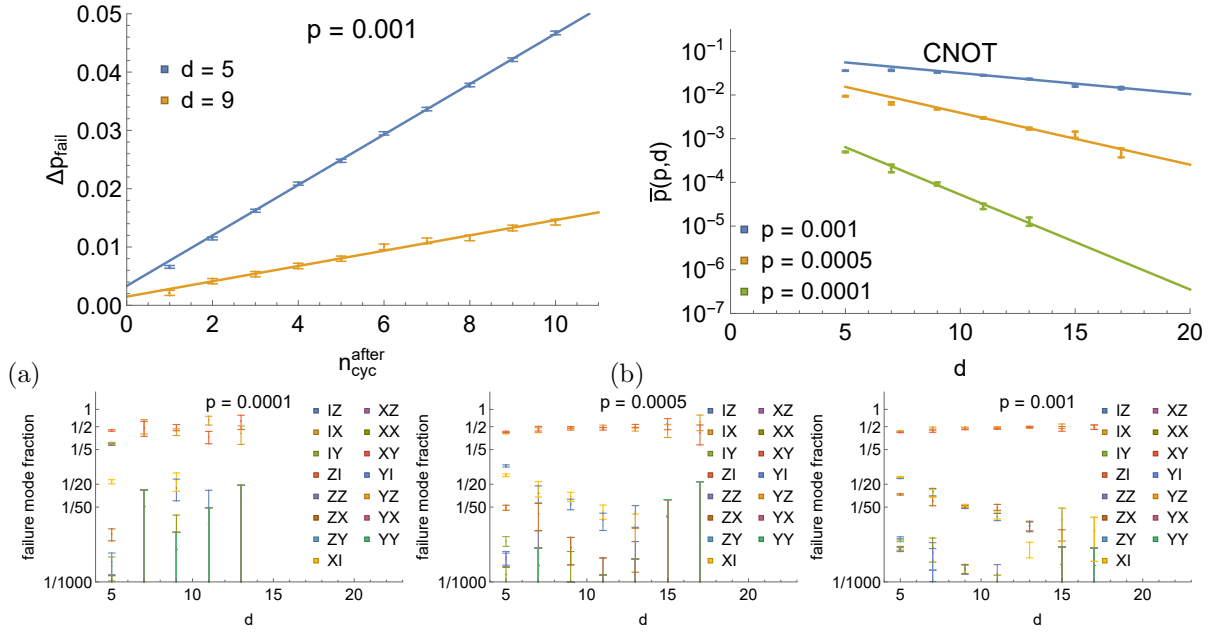


FIG. 17. Analysis of the logical CNOT operation. (a) The change in failure probability between the time immediately after the transverse application of CNOT gates and $n_{\text{cyc}}^{\text{after}}$ EC rounds later. We observe that EC equilibration occurs after two EC cycles. (b) For each value of p, we use the ansatz in Eq. (11) to extract the parameters for $\overline{p}_{\text{CNOT}}(p, d)$ in Table I. (bottom panels) The fraction of overall failures corresponding to different logical Pauli errors. The dominant errors are ZI and IX, since the logical CNOT propagates X errors onto the second patch and Z errors onto the first patch. Other noticeable errors include IZ and XI, which correspond to the failure over two rounds of EC, but become negligible for large d. The points whose data range falls below the horizontal axis correspond to no observed failures.

Logical readout.— Recall that to fault-tolerantly read off $\overline{Z}$, one measures each data qubit in

[¶] Note that it is possible to improve this implementation of the CNOT gate considerably as follows. One can first decode the error separately for the patch which is the source of the copied error (i.e. the control patch for X error and the target patch for the Z error) and then apply the correction to the destination of the copied error before correcting the residual error there [93].

the Z basis. The resulting bit string can be fixed using a perfect-measurement decoder so that it satisfies all Z-type stabilizers, and then the outcome of the $\overline{Z}$ logical operator can be read off from any representative. One can fault-tolerantly measure $\overline{X}$ similarly by measuring each data qubit in the X basis.

To simulate the logical $\overline{X}$ readout we first run the system for d rounds of EC to ensure thermalization has occurred, then measure each qubit in the X basis under circuit noise of strength p, followed by perfect-measurement decoding. To isolate the contribution to the logical operator from the logical measurement alone, we find and remove the contribution to the logical operator from the initial d rounds of EC by applying a perfect-measurement decoding on the system immediately after the d rounds. We estimate the failure probability and mode fraction for logical measurement of $\overline{X}$ and $\overline{Z}$ in Fig. 18. We fit Eq. (11) to the data for both $\overline{X}$ and $\overline{Z}$ measurement, which agree with one another and this fit provides the entry for $\overline{p}_{\text{meas}}(p, d)$ in Table I.

Note that the noise contributed by readout is orders of magnitude below the other logical operations, and we only take data for $p = 0.001$ since a prohibitively large number of samples would be required for $p = 0.0005$ and $p = 0.0001$. This justifies us to neglect contributions from readout in our effective noise model.

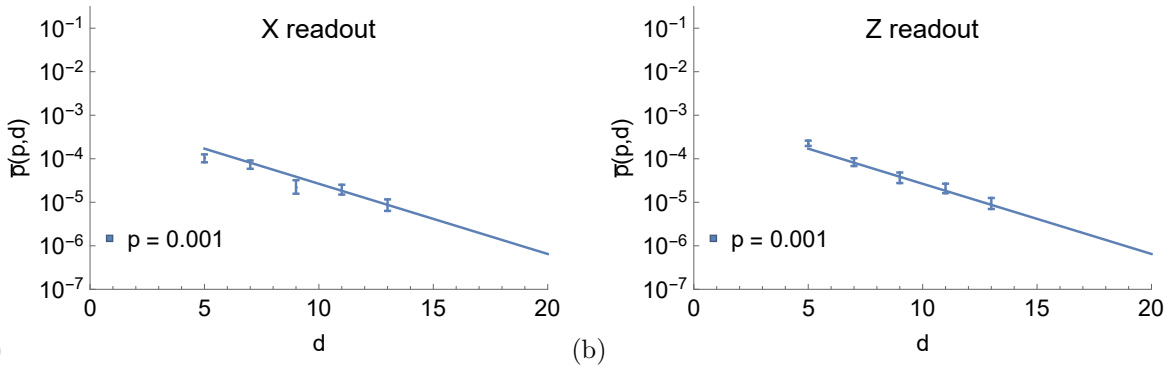


FIG. 18. The readout failure probability for $\overline{X}$ and $\overline{Z}$. Note the value is orders of magnitude lower than for the other logical operations and that we only show that for $p = 0.001$, since the smaller values of p are so low that they are difficult to observe using Monte Carlo. We use the ansatz in Eq. (11) to extract the parameters for $\overline{p}_{\text{meas}}(p, d)$ in Table I, although in our noise model we neglect noise in the measurement.

IV. STATE DISTILLATION OVERHEAD ANALYSIS

In this section we carefully analyze the performance and estimate the overhead of state distillation of $|T\rangle$ states using the standard 15-to-1 scheme. As we will see, this simple distillation protocol already outperforms code switching in the regime of interest.

A. Creating the magic state via distillation in 3 steps

A protocol to produce an encoded magic state using distillation, which we illustrate in Fig. 19, consists of the following steps.

1. **Magic state initialization.**—We initialize encoded magic states in small-distance 2D color code patches, which later serve as the input to the first round of distillation.

2. **Expansion and movement of patches.**—The code distance in consecutive rounds of distillation is required to increase to protect the produced magic states of improving fidelity. We must therefore increase the size of a patch output from one round, and move it to the desired location where it becomes an input for the next round.

3. **Distillation circuit.**—This is a circuit with every qubit encoded in a distance-d 2D color code, consisting of nearest-neighbor logical Clifford operations on patches arranged in a three-dimensional stack. The input consists of 15 encoded $|\overline{T}\rangle$'s, and one higher fidelity encoded $|\overline{T}\rangle$ output.

The second and third steps are repeated (on multiple copies of the procedure in parallel) with increasing code distances chosen to minimize the overhead until magic state of the desired quality is produced. In the following subsections, we go through each of the three steps, elaborating on the implementation and simulation details. In our analysis, we assume circuit noise of strength p, and use the effective noise model presented in Section III D to analyze the performance of logical-level circuits implemented with the 2D color code.

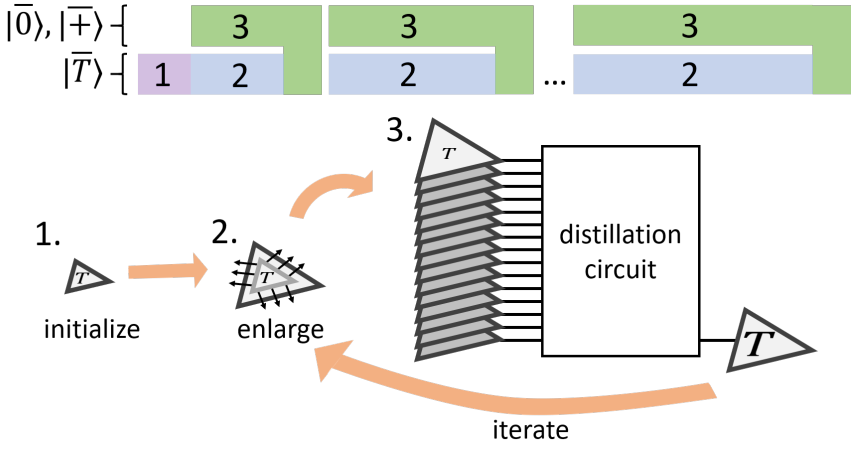


FIG. 19. The protocol to create an encoded magic T state via state distillation, with a qualitative timeline of each step. In step 1, noisy encoded magic states are prepared non-fault-tolerantly in a small 2D color code. In step 2, each code patch is expanded to increase its distance, and we assume that the logical infidelity of the encoded magic state does not change. In step 3, the distillation circuit is run on 15 magic states, and, given successful post-selection, outputs a single magic state of higher fidelity (indicated by a larger T). Steps 2 and 3 are iterated until a state of the desired infidelity is produced.

1. *Magic state initialization*

The first step of any magic state distillation protocol is to produce the initial encoded magic states. The initialization protocol to do this is crucial since the results of distillation depend strongly on the quality of the initial magic states. For example, taking the 15-to-1 scheme with perfect Clifford operations, if the starting infidelity is decreased by a factor of 2.24 (which, as we will see later, can be achieved by varying the CNOT order), the output infidelity is reduced by about 10, 10^3 and 10^9 over one, two and three distillation rounds, respectively. Although abstract distillation protocols have received a lot of attention, there is surprisingly little research on the initialization of magic states as inputs for distillation despite the enormous potential impact. For the surface code,

a non fault-tolerant scheme with the logical error of the final encoded magic state comparable with that of raw state was proposed by Li in Ref. [94]. More recent works [71, 95] present fault-tolerant approaches to initialize encoded magic states and can, in some regimes, achieve higher fidelity encoded magic states with low overhead, but are somewhat more challenging to implement.

Our strategy of initializing magic states for the 2D color code can be viewed as a generalization of the approach in [94], which consists of two main steps: (i) produce an encoded magic state in a distance d_1 code (with $d_1 < d$), then (ii) enlarge the code from d_1 to d. For judiciously chosen d_1, the noise added during step (ii) can be neglected because it is much less significant that the noise from step (i). We produce an encoded magic state in the following steps.

1. Choose representatives of the logical X and Z operators which intersect on a single qubit, and prepare that qubit in $|T\rangle$. Prepare the remaining data qubits along the support of the logical X and Z in $|+\rangle$ and $|0\rangle$, respectively. Other data qubits are prepared in either $|+\rangle$ or $|0\rangle$ to maximize the number of satisfied stabilizers; see Fig. 20(a).

2. Measure each stabilizer twice; see Fig. 20(b). If the observed syndrome is not the same or the syndrome could not have arisen without fault, it is rejected and discarded.

3. Apply a Pauli operator fixing the observed syndrome.

In our simulation, we say the procedure has succeeded in creating an encoded magic state if upon a single additional fault-free round of error correction, one obtains the encoded magic state. We remark that the state from step 1. is not an eigenstate of all the stabilizers measured in step 2., and thus, even in the absence of faults, we may need to apply a nontrivial Pauli operator in step 3. to ensure that all the stabilizers are satisfied. We present additional details about how to improve the efficiency of a naive simulation of steps 1.-3. in Section VI C.

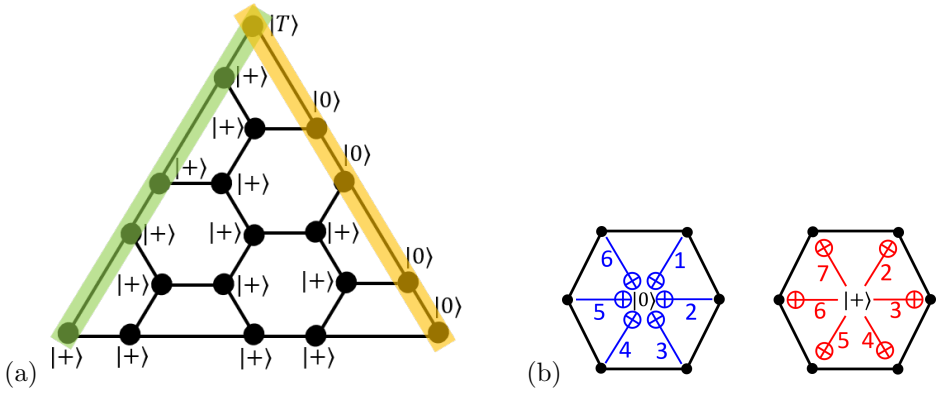


FIG. 20. Magic state initialization for the 2D color code with distance $d = 5$. (a) In step 1., the top qubit is prepared in the $|T\rangle$ state, and the remaining data qubits are prepared in either $|+\rangle$ or $|0\rangle$. We depict the support of logical X and Z representatives as shaded yellow and green strips. (b) In step 2., all the stabilizers are measured using the depicted CNOT order (or its restriction).

.

Lastly, we optimize the initialization protocol by varying the order in which the CNOT gates are applied during the two EC rounds in step 2. We again assume that there is a separate ancilla qubit per stabilizer generator and consider the 144 valid sequences consisting of 7 CNOT steps as in Section III C. The parameters for the best circuit (which is not the same as that used for standard error correction) are more than twice as favorable as for the worst circuit; see Section VI C for more details and a similar analysis for the rotated surface code.

The protocol takes

$$\tau^{\text{init}} = 19 \tag{13}$$

time units: one time unit to prepare the qubits in $|0\rangle$, $|+\rangle$ and $|T\rangle$ states, and two EC cycles each lasting 9 time units. We find that under circuit noise of strength p the lowest order contributions to the output infidelity $p_{\text{fail}}^{\text{init}}$ and rejection probability $p_{\text{rej}}^{\text{init}}$ are

$$p_{\text{fail}}^{\text{init}} = 3.93p, \quad p_{\text{rej}}^{\text{init}} = 379p. \tag{14}$$

2. Expansion and movement of patches

We neglect any error introduced by expansion of the $|\overline{T}\rangle$ states, i.e., while enlarging the distance of the base code from $d^{(i)}$ to $d^{(i+1)}$ between rounds i and $i+1$, and while these are moved. This is justified since errors are suppressed throughout the expansion similarly as during error correction of the distance-$d^{(i)}$ code. We also neglect any additional time overhead introduced by the expansion and movement of patches. This is justified since the expansion can be done within the $d^{(i)}$ EC rounds, and the swaps needed to move the outputs each take just one EC round, which we expect can be done during the $d^{(i+1)}$ EC rounds needed to prepare the $|\overline{0}\rangle$ and $|\overline{+}\rangle$ states at the start of the distillation circuit; see Fig. 21.

3. 15-to-1 distillation circuit

Here, we analyze the 15-to-1 scheme run on logical qubits encoded in patches of 2D color code with distance d arranged in a three-dimensional stack. The logical circuits are analysed using the effective noise model in Section III D, which takes two parameters: the distance d of the 2D color codes used, and the strength p of the underlying circuit noise. We assume noisy initial T states of the form $\rho = (1-q)|T\rangle\langle T| + q|T^\perp\rangle\langle T^\perp|$. Since we allow Clifford operations only between adjacent patches in the stack, we have to appropriately modify the distillation circuit; see Fig. 21. In our analysis, we only keep track of errors up to order q^3 and $\overline{p}$, which are of similar order of magnitude in the regime of interest. This splits the analysis into looking at error either in the magic states alone, or in the Clifford operations alone.

Noisy magic states.— First we briefly review the effect of noise on T states [50]. As described in Section II A, we simplify the analysis by twirling by randomly applying one of the Cliffords A, A^2, A^3 and $A^4 = I$, each with probability 1/4, where $A \propto e^{i\pi/2}|T\rangle\langle T| + e^{-i\pi/2}|T^\perp\rangle\langle T^\perp|$. Recall that single-qubit Cliffords can be done instantaneously and perfectly by frame tracking in the 2D color code as described in Section II C. This twirling forces the noisy $|T\rangle$ state to be of the form $\rho = (1-q)|T\rangle\langle T| + q|T^\perp\rangle\langle T^\perp|$, or equivalently that each $|T\rangle$ state is afflicted by a Z error with probability q. The noise on the set of 15 input noisy T states is therefore represented as a Z-type Pauli error E occurring with probability $q^{|E|}(1-q)^{15-|E|}$. The protocol will reject if E is a detectable error for the punctured Reed-Muller code, which has distance $d = 3$ for Z-type operators. The protocol results in a failure iff E is a non-trivial logical operator. Explicit enumeration shows that there are 35 weight-3 Z-type logical operators, such that the contributions p_{rej} and p_{fail} due to T errors are

$$p_{\text{rej}}^{T} = 15q, \quad p_{\text{fail}}^{T} = 35q^3. \tag{15}$$

Noisy Clifford operations.—Now we consider the effect of noise in the Clifford operations [59, 60] in the distillation circuit. First we analyze the faults that occur during the Reed-Muller

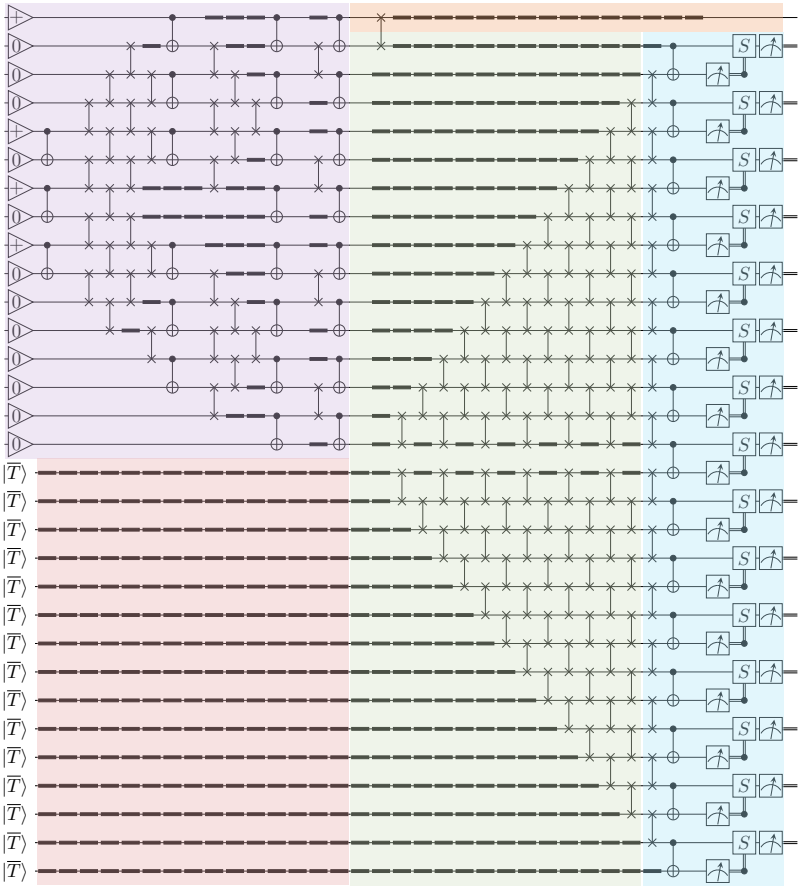


FIG. 21. The 15-to-1 distillation protocol using nearest-neighbor operations in a stack of 2D color codes. The 31 logical qubits are encoded in distance-d 2D color codes which are stacked on top of one another. Idle locations (thick wire segments), SWAPs and CNOTs last 1,1, and 2 EC cycles respectively. The 15 input $|\overline{T}\rangle$ states remain idle for $d+16$ EC cycles (pink) while the Reed-Muller state is prepared (lilac). We assume the $|\overline{0}\rangle$ and $|\overline{+}\rangle$ states are only prepared (indicated by a triangle) when they are needed, taking d EC cycles. A shuffle circuit (green) which lasts 14 EC cycles interleaves the $|\overline{T}\rangle$ states with the qubits in the Reed-Muller state, where they undergo gate teleportation and measurement (blue) in 2 EC cycles. The distilled magic state is in the top wire (orange).

state preparation (lilac in Fig. 21), which takes 16 EC cycles to complete. We propagate each fault as a logical Pauli operator through the circuit, and assume that every other operation acts perfectly, including the $\overline{T}$ gates. By explicitly representing the state and operations as the vector and matrices of dimension 2^{16} and $2^{16}\times 2^{16}$, respectively, we find that the contributions are

$$p_{\rm rej}^{\rm RM} = 12.3\ \overline{p}_{\rm prep} + 155\ \overline{p}_{\rm idle} + 38.1\ \overline{p}_{\rm CNOT}, \quad p_{\rm fail}^{\rm RM} = 1.9\ \overline{p}_{\rm idle} + 1.5\ \overline{p}_{\rm CNOT}. \tag{16}$$

Next we consider faults in the 16 idle EC cycles of the output qubit. Note that there is a choice of which of the 16 qubits of the Reed-Muller state to puncture in the 15-to-1 protocol. We simulated all 16 choices, and selected the second qubit as the output since it had the lowest contribution from $\overline{p}_{\rm CNOT}$. Any failure in any of these locations will result in an undetected failure

$$p_{\rm rej}^{\rm out} = 0, \quad p_{\rm fail}^{\rm out} = 16\ \overline{p}_{\rm idle}. \tag{17}$$

By explicit calculation, we find the exact contribution to $p_{\rm rej}$ and $p_{\rm fail}$ of every Clifford fault location in Fig. 21 according to our effective noise model.

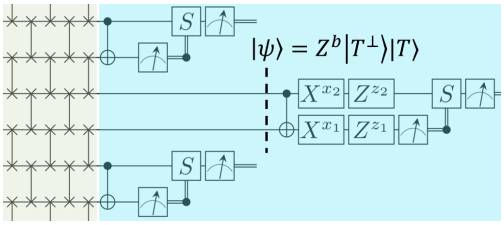


FIG. 22. A part of the distillation circuit with the inclusion of a Pauli error $X_1^{x_1} Z_1^{z_1} X_2^{x_2} Z_2^{z_2}$ on a pair of qubits. Such an error can appear as a result of previous faults being propagated through the circuit. Without loss of generality we assume that all the other measurements have been completed before this error, which ensures that the pair of qubits is decoupled from the rest of the system. If the error-free outcome is b, then the error-free state must be $|\psi\rangle = (Z^b|T^\perp\rangle) \otimes |T\rangle$. This simple 2-qubit circuit can then be analyzed for all 16 cases of the error $X_1^{x_1} Z_1^{z_1} X_2^{x_2} Z_2^{z_2}$.

Lastly we analyze the remaining fault locations. These consist of the $15(d+16)$ idle locations involving qubits holding the $|\overline{T}\rangle$ states which remain idle during the production of the RM state (pink), the 420 idle and SWAP locations in the shuffle circuit (green), and the 15 CNOTs used to implement gate teleportation (blue) in Fig. 21. A single fault in any of these locations will propagate to a Pauli operator $X_1^{x_1} Z_1^{z_1} X_2^{x_2} Z_2^{z_2}$ acting on the pair of qubits in a gate teleportation circuit between the CNOT and the measurements, where x_1, z_1, x_2 and z_2 each take values $0, 1$ and the first qubit holds the $|\overline{T}\rangle$, and the second is from the Reed-Muller state. To analyze the effect of such a Pauli operator, we imagine delaying the measurements on the affected pair of qubits until after the completion of the rest of the circuit; see Fig. 22. At this point, all other measurements are completed, and if the Pauli operator were trivial, i.e., if $x_1 = z_1 = x_2 = z_2 = 0$, the outcome $b \in \{0, 1\}$ of the X measurement would be determined by the previous outcomes since the X-stabilizers must be satisfied. Therefore, the pair of qubits must be completely unentangled with the rest of the system. This tells us that none of these fault locations can result in a failure, but result in rejection if and only if the outcome of the X measurement is modified by the Pauli operator $X_1^{x_1} Z_1^{z_1} X_2^{x_2} Z_2^{z_2}$. We straightforwardly analyze this 2-qubit circuit with its pure initial state and for Pauli operator find the probability $p_{\text{flip}}(x_1, x_2, x_3, x_4)$ that the outcome is flipped, namely

$$p_{\text{flip}}(x_1, x_2, x_3, x_4) = \begin{cases} 0 & \text{if } x_1 = x_2 \text{ and } z_2 = 0, \\ 1/2 & \text{if } x_1 \neq x_2 \text{ and } z_2 = 0, \\ 1 & \text{if } x_1 = x_2 \text{ and } z_2 = 1, \\ 1/2 & \text{if } x_1 \neq x_2 \text{ and } z_2 = 1. \end{cases} \tag{18}$$

Then all that remains is to count the contribution to each of these Paulis from the aforementioned locations according to the effective error model, which yields

$$p_{\text{rej}}^{\text{rem}} = (392 + 4.13d)\ \overline{p}_{\text{idle}} + 13.5\ \overline{p}_{\text{CNOT}}, \quad p_{\text{fail}}^{\text{rem}} = 0. \tag{19}$$

The contributions from Eq. (15), Eq. (16), Eq. (17) and Eq. (19) combine to give the rejection and failure probability of the distillation step

$$p_{\text{rej}}^{\text{dist}} = 15q + 12.3\ \overline{p}_{\text{prep}} + 524\ \overline{p}_{\text{idle}} + 50.8\ \overline{p}_{\text{CNOT}}, \tag{20}$$

$$p_{\text{fail}}^{\text{dist}} = 35q^3 + 17.9\ \overline{p}_{\text{idle}} + 1.52\ \overline{p}_{\text{CNOT}}. \tag{21}$$

The number of time units required to implement the distillation circuit can be straightforwardly identified from Fig. 21 and is equal to

$$\tau^{\text{dist}}(d) = 9\,(d + 32)\,. \tag{22}$$

B. Distillation overhead

Here we estimate the overhead required for a k-round distillation protocol using distances $\{d^{(1)}, d^{(2)}, \ldots, d^{(k)}\}$ under strength-p circuit noise, as well as the infidelities output by each round $\{q^{(1)}, q^{(2)}, \ldots, q^{(k)}\}$. Note that $q^{(k)}$ is the infidelity of the encoded magic state produced by the overall protocol.

Recall that the first step of the distillation protocol is to initialize encoded magic states in distance-5 2D color codes; see Fig. 19 and Section IV A 1. Writing the infidelity of the state after initialization as $q^{(0)} = p_{\text{fail}}^{\text{init}}(p)$, the remaining infidelities are then calculated iteratively according to

$$q^{(i)} = p_{\text{fail}}^{\text{dist}}(q^{(i-1)}, p, d^{(i)}). \tag{23}$$

To estimate the overhead, it is useful to streamline our notation. Let $N_{2D}(d) = (3d^2 - 1)/2$ be the number of qubits used to implement the distance-d 2D color code. For each distillation round $i \in \{1, 2, \ldots k\}$, let $r^{(i)} = \left[1 - p_{\text{rej}}^{\text{dist}}(p, q^{(i-1)}, d^{(i)})\right]$ be the acceptance probability, $R^{(i)} = 15$ be the number of $|\overline{T}\rangle$s required assuming acceptance, and $\alpha^{(i)} = 31/15$ be the number of logical qubits needed per input $|\overline{T}\rangle$ in the distillation circuit. To account for the initialization step, we also set $d^{(0)} = 5$, $r^{(0)} = \left[1 - p_{\text{rej}}^{\text{init}}(p)\right]$, $R^{(0)} = 1$ and $\alpha^{(0)} = 1$.

We can calculate the expected number of physical qubits required in each round. We imagine preparing a large number of output magic states, and thus we can talk about the average overhead.** Since the distillation protocol in the last round succeeds with probability $r^{(k)}$, on average it needs $R^{(k)}/r^{(k)}$ input $|\overline{T}\rangle$s. We therefore require on average $\alpha^{(k)} N_{2D}(d^{(k)}) R^{(k)}/r^{(k)}$ qubits for the last round. To supply the kth round, the $(k-1)$th round must therefore output $R^{(k)}/r^{(k)}$ $|\overline{T}\rangle$s on average, which $R^{(k)}/(r^{(k-1)} r^{(k)})$ input $|\overline{T}\rangle$s, which requires $\alpha^{(k)} N_{2D}(d^{(k)}) R^{(k-1)} R^{(k)}/(r^{(k-1)} r^{(k)})$ physical qubits, and so on. The qubit overhead N_{SD} of the distillation protocol can then be found as the number of qubits needed in the most qubit-expensive distillation round, namely

$$N_{\text{SD}} = \max_{i=1,\ldots,k} \left[\alpha^{(i)} N_{2D}(d^{(i)}) \prod_{j=i}^{k} \frac{R^{(j)}}{r^{(j)}} \right]. \tag{24}$$

The time required for distillation is

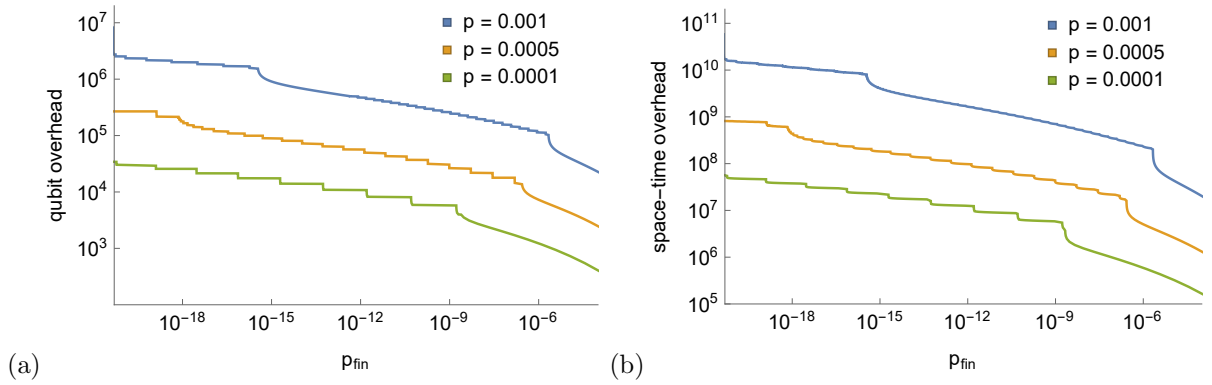
$$\tau_{\text{SD}} = \tau^{\text{init}} + \sum_{i=1}^{k} \tau^{\text{dist}}(d^{(i)}). \tag{25}$$

The space-time overhead is then simply $N_{\text{SD}} \tau_{\text{SD}}$. For various values of p, we run a simple search over number of rounds $k \in \{1, 2, 3\}$ and distances $\{d^{(1)}, d^{(2)}, \ldots, d^{(k)}\}$ to distill a target infidelity p_{fin} for a low space or space-time overhead; see Fig. 23.

V. FURTHER INSIGHTS INTO 3D COLOR CODES

In this section we present some insights into 3D color codes that are useful for code switching. In Section V A, we describe a simple approache to switch between the 2D and 3D color codes. At the core of this approach is the fact that a gauge-fixing of the 3D subsystem color code can be

** If we were to produce just one output magic state, then the overhead would slightly increase as we would need to guarantee that with high probability there are sufficiently many magic states at each level of the distillation protocol.



(a) (b)

FIG. 23. (a) The qubit and (b) space-time overhead of state distillation as a function of the infidelity p_{fin} of the output T state.

viewed as a collection of 2D color codes. In Section V B, we describe some relevant features of the gauge operators of the 3D color code which will be relevant for gauge-fixing and also for the simulation of noise upon the application of the transverse $\overline{T}$ gate. In Section V C, we generalize the restriction decoder to correct Z-errors in the 3D color code with boundaries, which will later be used in our code switching protocol.

A. A simple way to switch between 2D and 3D color codes

First we recall the general procedure for *gauge fixing* from a subsystem code with gauge group $\mathcal{G}$ to a stabilizer code with stabilizer group $\mathcal{S}' \subseteq \mathcal{G}$, where both codes share a set of bare logical operators. We require that the stabilizer group $\mathcal{S}$ of the subsystem code, which is the centralizer of $\mathcal{G}$ in the Pauli group intersected with $\mathcal{G}$ modulo the phase, i.e., $\mathcal{S} = (\mathcal{Z}(\mathcal{G}) \cap \mathcal{G})/\langle iI \rangle$, is contained in $\mathcal{S}'$, namely $\mathcal{S} \subseteq \mathcal{S}'$. Consider any state $|\psi\rangle$ in the codespace of the subsystem code, which by definition is a $(+1)$-eigenstate of all elements of $\mathcal{S}$. To switch from $\mathcal{G}$ to $\mathcal{S}'$, we first measure a generating set of $\mathcal{S}' \setminus \mathcal{S}$. A subset of those measured generators may have -1 outcomes, but there must exist an element $g \in \mathcal{G}$ which anti-commutes with precisely that subset of generators. Hence after applying g, all the stabilizers of $\mathcal{S}'$ will be satisfied, and since g commutes with the bare logical operators, the logical state is unaffected, which completes the transfer from $\mathcal{G}$ to $\mathcal{S}'$. This procedure is named gauge fixing since it involves measuring some (initially unsatisfied) gauge operators, and fixing them to be $+1$.

Central to switching between color codes is the fact that both the 3D stabilizer color code and the 2D color code can be viewed as gauge fixings of the 3D subsystem color code; see Fig. 24. The gauge and stabilizer groups $\mathcal{G}_{\text{sub}}$ and $\mathcal{S}_{\text{sub}}$ for the 3D subsystem color code, and the stabilizer group $\mathcal{S}_{\text{3D}}$ for the 3D stabilizer color code are:

$$\mathcal{G}_{\text{sub}} = \langle X(e), Z(e) \mid \forall e \in \Delta_1'(\mathcal{L}_{\text{3D}}) \rangle, \quad \mathcal{S}_{\text{sub}} = \langle X(v), Z(v) \mid \forall v \in \Delta_0'(\mathcal{L}_{\text{3D}}) \rangle, \tag{26}$$

$$\mathcal{S}_{\text{3D}} = \langle X(v), Z(e) \mid \forall v \in \Delta_0'(\mathcal{L}_{\text{3D}}), e \in \Delta_1'(\mathcal{L}_{\text{3D}}) \rangle. \tag{27}$$

We can define the stabilizer group $\mathcal{S}_{\text{2D}}$ of the 2D color code within the 3D lattice $\mathcal{L}_{\text{3D}}$ since $\mathcal{L}_{\text{2D}}$ is 'contained' in $\mathcal{L}_{\text{3D}}$; see Fig. 24(b). Namely,

$$\mathcal{S}_{\text{2D}} = \langle X(e), Z(e) \mid \forall e \in \Delta_1'(\mathcal{L}_{\text{3D}}) \text{: } e \text{ is incident to } v_Y \rangle, \tag{28}$$

where v_Y is the Y boundary vertex. Let us define the stabilizer group $\mathcal{S}_{\text{int}}$ supported on the qubits which are not near v_Y as follows

$$\mathcal{S}_{\text{int}} = \langle X(e), Z(e) \mid \forall e \in \Delta_1'(\mathcal{L}_{\text{3D}}) \text{: } e \text{ is incident to any interior } Y \text{ vertex} \rangle. \tag{29}$$

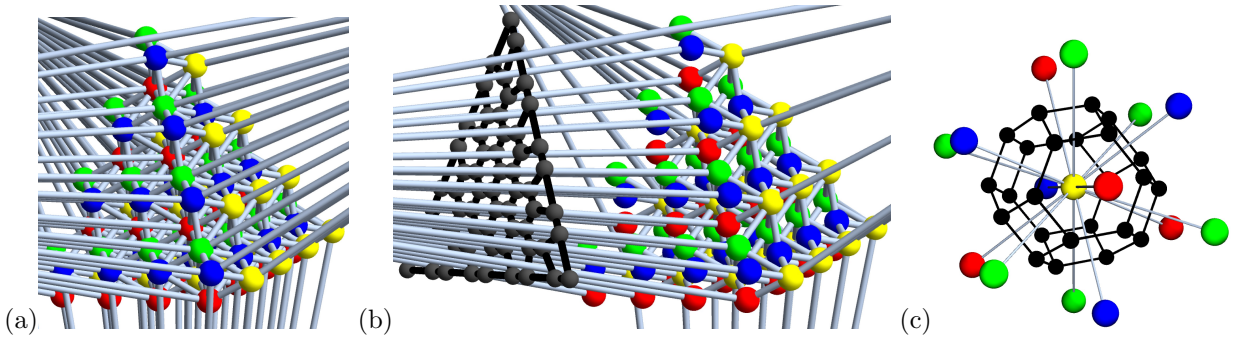


FIG. 24. (a) In one gauge fixing of the 3D subsystem color code $\mathcal{G}_{\rm sub}$ on the lattice $\mathcal{L}_{\rm 3D}$ (shown in Fig. 4b), all Z-edges (depicted by light struts) are satisfied. This corresponds to the 3D stabilizer color code $\mathcal{S}_{\rm 3D}$. (b) In another gauge fixing of $\mathcal{G}_{\rm sub}$, all X- and Z-edges incident to Y vertices in $\Delta_0(\mathcal{L}_{\rm 3D})$ (depicted by light struts) are satisfied. This corresponds to the 2D color code $\mathcal{S}_{\rm 2D}$ on the qubits near the boundary vertex v_Y and the 2D spherical color codes $\mathcal{S}_{\rm int}$ on other qubits. We depict the primal lattice of the 2D color code on the lattice $\mathcal{L}_{\rm 2D}$ (shown in Fig. 4a) in black. (c) Around each Y vertex in $\Delta_0'(\mathcal{L}_{\rm 3D})$, there is the 2D spherical color code, whose primal lattice we depict in black.

We can think of $\mathcal{S}_{\rm int}$ as the group generated by the stabilizers of the 2D spherical color codes centered around every Y interior vertex of $\mathcal{L}_{\rm 3D}$; see Fig. 24(c). Note that every 2D spherical color code encodes zero logical qubits.

It is straightforward to see that $\mathcal{G}_{\rm sub}$ contains both $\mathcal{S}_{\rm 3D}$ and $\langle \mathcal{S}_{\rm 2D}, \mathcal{S}_{\rm int}\rangle$. Moreover, $\mathcal{S}_{\rm sub} \subseteq \mathcal{S}_{\rm 3D}$ and $\mathcal{S}_{\rm sub} \subseteq \langle \mathcal{S}_{\rm 2D}, \mathcal{S}_{\rm int}\rangle$ since vertex operators $X(v)$ and $Z(v)$ can be formed by multiplying operators $X(e)$ and $Z(e)$ on all the edges of the same color incident to v. Also note that there is a shared representation of bare logical operators for all three codes (for example X and Z applied to every qubit in $\mathcal{L}_{\rm 2D}$). Therefore, to move from the subsystem code $\mathcal{G}_{\rm sub}$ to either of the stabilizer codes, i.e., $\mathcal{S}_{\rm 3D}$ or $\langle \mathcal{S}_{\rm 2D}, \mathcal{S}_{\rm int}\rangle$, gauge switching can be used. Switching from either of the stabilizer codes to the subsystem code requires no action, since any state which is a $(+1)$-eigenstate of every element of $\mathcal{S}_{\rm 3D}$ or $\langle \mathcal{S}_{\rm 2D}, \mathcal{S}_{\rm int}\rangle$ must also be a $(+1)$-eigenstate of every element in $\mathcal{S}_{\rm sub}$.

This example of gauge fixing is sometimes referred to as a *dimensional jump* because the logical information is moved between codes defined on two- and three-dimensional lattices [57].

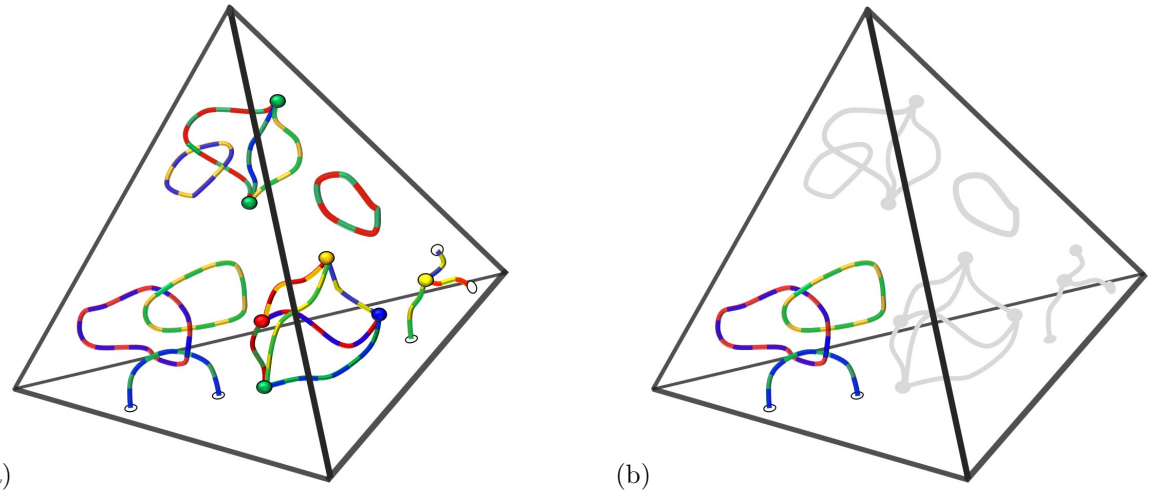
B. Physics of the gauge flux in 3D color codes

Here we consider general features of the gauge operators of the 3D subsystem color code. Suppose there is some X error $\epsilon \subseteq \Delta_3(\mathcal{L})$ in the system. We define the Z-type *gauge flux* $\gamma \subseteq \Delta_1'(\mathcal{L})$ to be the subset of interior edges which would return -1 outcomes if all Z edges in the system were measured perfectly. Since each edge has one color from $\mathcal{K} = \{RG, RB, RY, GB, GY, BY\}$, we distinguish six types of the flux and write

$$\gamma = \sum_{K\in\mathcal{K}} \gamma^K. \tag{30}$$

Although the flux γ can be random, it has to form a collection of strings, which may branch and can terminate only at the boundary of the lattice; see Fig. 25. A local constraint capturing this behavior, which we call the *Gauss law*, can be stated as follows. Let $K_1, K_2, K_3, K_4 \in \{R, G, B, Y\}$ be four different colors. Then, for any vertex $v \in \Delta_0'^{K_1}(\mathcal{L})$, the number of edges of γ of color K_1K_2 or K_1K_3 and incident to v has to be even, i.e.,

$$\left|\gamma^{K_1K_2}|_v + \gamma^{K_1K_3}|_v\right| \equiv 0 \mod 2 \tag{31}$$



(a) (b)

FIG. 25. A schematic representation of the Z-gauge flux γ in the bulk (enclosed by the black tetrahedron) of the 3D subsystem color code on the lattice $\mathcal{L}_{3D}$ in Fig. 4(b). (a) The flux γ consists of strings of six different colors. There are seven branching points of γ (depicted as red, green, blue and yellow vertices). The flux γ has to satisfy the Gauss law in the bulk, and can terminate at the boundary of $\mathcal{L}_{3D}$. (b) We highlight one linked component of γ, which contains three connected components of γ that are linked.

This local constraint arises from the redundancies among gauge generators. Namely, in order to form a stabilizer generator $Z(v)$ identified with the vertex v, we can multiply all the gauge generators on edges of color K_1K_2 incident to v. Alternatively, we can obtain $Z(v)$ as the product of all gauge generators on edges of color K_1K_3 incident to v. Since the number of -1 measurement outcomes among those gauge generators in both cases must be equal, we recover Eq. (31).

Note that when the stabilizer $Z(v)$ is violated (indicating the presence of some X error), then the number of -1 outcomes for gauge generators on edges touching v of color K_1K_2 has to be odd, i.e., $\left|\gamma^{K_1K_2}|_v\right| \equiv 1 \mod 2$. In such a case, we call the vertex v a branching point of γ, as three different flux types $\gamma^{K_1K_2}$, $\gamma^{K_1K_3}$ and $\gamma^{K_1K_4}$ meet at v. We remark that to perform error correction with the 3D subsystem color code, one can use the information about the branching points of the flux γ. If the flux γ has no branching points at any interior vertex, then all Z stabilizers are satisfied and there always exists an X-type gauge operator supported on edges $f(\gamma) \subseteq \Delta_1(\mathcal{L}_{3D})$ which anti-commutes with precisely those Z gauge generators in γ.

If an edge set satisfies the Guass law, we say that it is *valid*. An edge set which does not satisfy the Gauss law is said to be *invalid*, and we refer to all vertices which violate the Gauss law as *termination points*. When the gauge flux is measured with noisy circuits, there can be errors in the reported *noisy gauge flux* causing it to be invalid; see Fig. 26.

Now, we discuss the structure of the flux γ which satisfies the Gauss law. First, we can find a decomposition of γ in terms of its connected components, i.e.,

$$\gamma = \sum_{j=1}^{a} \gamma_j. \tag{32}$$

By definition, different connected components are disjoint, i.e., $\gamma_j \cap \gamma_k = \emptyset$ for $j \neq k$.

Then, we define a linked component of γ as a subset of all connected components of γ, which are linked (in the sense of knot theory); see Fig. 25(b). Finally, we can decompose γ as a disjoint union of its linked components

$$\gamma = \sum_{i=1}^{b} \lambda_i. \tag{33}$$

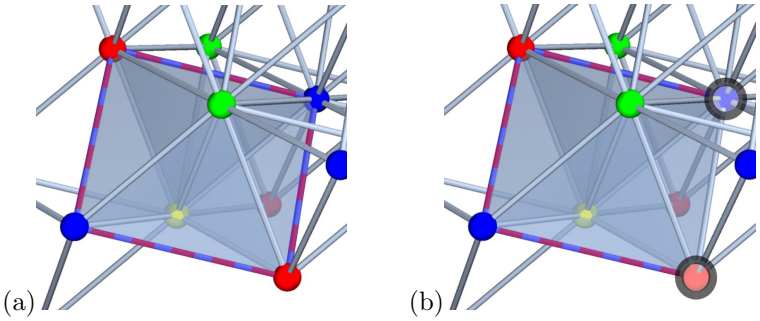


FIG. 26. (a) The flux γ (highlighted RB edges) due to a single GY-edge generator (shaded four tetrahedra) contains only RB edges. This set of edges γ satisfies the Gauss law. (b) A noisy measurement of the flux may not be valid, meaning it does not satisfy the Gauss law. Here $\widetilde{\gamma}$ has two termination points (highlighted R and B vertices) at which the Gauss law is violated.

Note that each λ_i is the sum of some γ_j's.

For convenience, we introduce a function $\mathrm{col} : \Delta_0(\mathcal{L}) \to \mathbb{Z}_2^3$, which for each vertex v returns its color, where we set $R = (1,0,0)$, $G = (0,1,0)$, $B = (0,0,1)$ and $Y = (1,1,1)$.

For any linked component λ_i of we refer to a subset $\sigma \subseteq \Delta_0(\lambda_i)$ as an excitation configuration for λ_i and call $\sum_{v\in\sigma} \mathrm{col}(v)$ the total charge of σ. We denote the *the collection of excitation configurations* $\Sigma(\lambda_i)$ for λ_i with the neutral total charge as follows

$$\Sigma(\lambda_i) = \left\{ \sigma \subseteq \Delta_0(\lambda_i) \;\middle|\; \sum_{v\in\sigma} \mathrm{col}(v) = (0,0,0) \right\}. \tag{34}$$

Writing the linked component λ_i in terms of its connected components $\lambda_i = \gamma_{i_1} + \gamma_{i_2} + \cdots + \gamma_{i_k}$, we also introduce *the collection of excitation configurations without the linking charge*

$$\Sigma'(\lambda_i) = \Sigma(\gamma_{i_1}) \times \Sigma(\gamma_{i_2}) \times \cdots \times \Sigma(\gamma_{i_k}). \tag{35}$$

We remark that the linking charge is a charge which can be transferred between two connected components γ_i and γ_j, which are linked; see [96]. Note that $\Sigma'(\lambda_i)$ is contained in $\Sigma(\lambda_i)$ and that when the linked component λ_i consists of a single connected component, they coincide $\Sigma'(\lambda_i) = \Sigma(\lambda_i)$.

C. Restriction decoder for 3D color codes with boundaries

Here we provide details on correcting Z type errors in the 3D stabilizer color code with perfect measurements, which is needed for the final step of the code switching protocol. We seek an efficient decoder for the 3D color code with good performance. Our approach is to adapt the restriction decoder, which was originally defined for lattices with no boundary from [31], to the tetrahedral lattice of interest here $\mathcal{L}_{3\mathrm{D}}$. We also apply additional minor modifications to improve the performance. In what follows we briefly review the restriction decoder and describe our modifications.

Let $\sigma \subseteq \Delta_0'(\mathcal{L}_{3\mathrm{D}})$ be the syndrome of the 3D stabilizer code on the tetrahedral lattice $\mathcal{L}_{3\mathrm{D}}$. We pick one color, say Y, and for each color $K \in \{R, G, B\}$ we separately analyze the restricted syndrome $\sigma^{KY} \subseteq \Delta_0'(\mathcal{L}_{3\mathrm{D}}^{KY})$ within the restricted lattice $\mathcal{L}_{3\mathrm{D}}^{KY}$. If $|\sigma^{KY}| \equiv 1 \mod 2$, then we add the boundary vertex v_Y to σ^{KY}. Next, we find a subset of edges $E^{KY} \subseteq \Delta_1(\mathcal{L}_{3\mathrm{D}}^{KY})$, which provides a pairing of vertices of σ^{KY} within the restricted lattice $\mathcal{L}_{3\mathrm{D}}^{RY}$. Note that we can find a pairing of minimal weight by using the minimum-weight perfect matching algorithm. Also, the weight of the edge connecting the boundary vertices v_Y and v_K is set to zero.

After finding the pairing $E = E^{RY} + E^{GY} + E^{BY}$ we apply a local lifting procedure to every Y vertex in the interior of $\mathcal{L}_{3D}$. Namely, for every Y vertex $v \in \Delta_0'(\mathcal{L}_{3D})$ we find any subset of tetrahedra $\tau(v) \subseteq \partial_{0,3} v$ in the neighborhood of v, whose 1-boundary locally matches E, i.e., $(\partial_{3,1}\tau(v))|_v = E|_v$. We emphasize that all such choices of $\tau(v)$ result in operators $Z(\tau(v))$, which may differ only by a stabilizer operator.

To lift the boundary vertex v_Y, we need to adapt the original restriction decoder in [31]. Since $\mathcal{L}_{2D}$ is a sublattice of $\mathcal{L}_{3D}$ (see Fig. 24(b)), one can show that the problem of finding $\tau(v_Y) \subseteq \partial_{0,3} v_Y$, which satisfies $(\partial_{3,1}\tau(v_Y))|_{v_Y} = E|_{v_Y}$, is equivalent to the problem of decoding the 2D color code defined on the facet of the tetrahedral lattice $\mathcal{L}_{3D}$ near v_Y. We remark that different choices of $\tau(v_Y)$ may lead to operators $Z(\tau(v_Y)$ differing by a logical operator. We use the projection decoder for the 2D color code as described in Section III A. Finally, the correction operator is found as $\prod_{v \in \Delta_0(\mathcal{L}_{3D})} Z(\tau(v))$, where the product is over all Y vertices, including the boundary vertex v_Y. Note that this adaption to accommodate a lattice boundary in three dimensions is analogous to an adaption presented in [37] for the two-dimensional case.

In Fig. 27(a) we show the performance of the restriction decoder adapted to the 3D stabilizer color code on the tetrahedral lattice $\mathcal{L}_{3D}$, finding a threshold of 0.55% for the iid phase-flip Z noise. This value can be contrasted with the threshold of 0.77% the restriction decoder for the color code on the 3-torus reported in [31]. Also, a similar adaption of the restriction decoder to the 3D color code with a boundary was recently presented in [97], however the reported values of 0.1–0.2% are surprisingly low.

This adapted restriction decoder on moderate system sizes is far from optimal. For example, some weight-2 errors cause failure for all $d \leq 9$. This phenomenon is not solely due to the presence of the boundaries. Namely, we found that there are some weight-2 errors in the color code of distance $d = 6$ on the 3-torus, which cause the restriction decoder to introduce a logical error.

To improve the performance, we consider a very simple additional modification of the restriction decoder to improve its performance for moderate system sizes. In addition to selecting a small weight set $\tau(v_Y)$ for the boundary vertex v_Y, we choose as $\tau(v)$ the set of minimal weight for every Y vertex $v \in \Delta_0'(\mathcal{L}_{3D})$. As mentioned above, this can only change the decoder output by a stabilizer, but the explicit representation will typically be of lower weight. Then, we simply rerun the same decoder three times by picking other colors, i.e., R, G and B instead of Y, and finally select the correction which has the lowest total weight among the four for colors Y, R, G, and B. This simple modification yields significant improvements: the lowest distance which corrects all weight-2 errors is now $d = 7$ compared to $d = 11$, and the threshold is increased from 0.55% to 0.8%; see Fig. 27.

VI. CODE SWITCHING OVERHEAD ANALYSIS

In this section we describe how code switching between 2D and 3D color codes can be used to fault-tolerantly produce an encoded T state in the 2D color code, and analyse the error and overhead of the process.

A. Creating the magic state via code switching in 6 steps

Here we outline the protocol to produce an encoded magic state using code switching. The main idea is to first produce a Bell-state across a pair of 2D color codes, and switch one of the two into a 3D color code where the logical $\overline{T}$ is applied transversely. Then by measuring $\overline{X}$ for the 3D code, the $\overline{T}$ state is effectively teleported to the remaining 2D code (up to a known Pauli correction). This approach avoids switching from the 3D code back to the 2D code, which we believe allows us

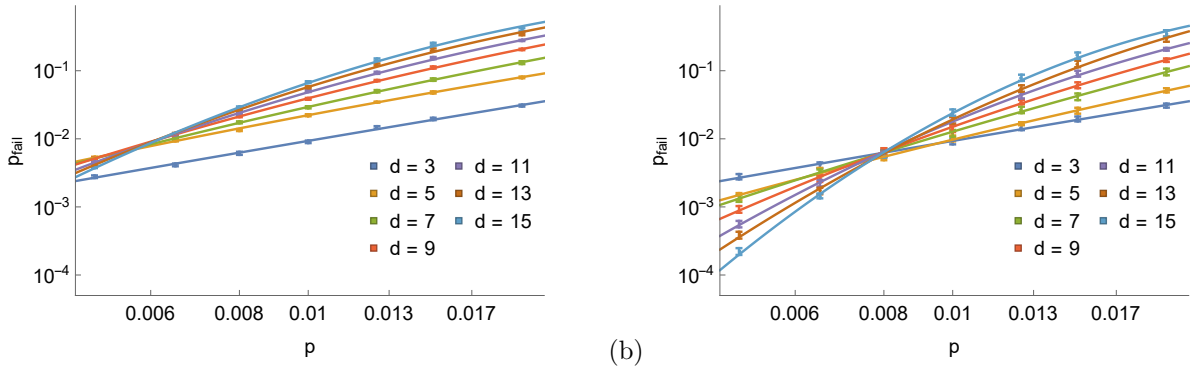


FIG. 27. Performance of the restriction decoder for the 3D color code adapted to the tetrahedral lattice $\mathcal{L}_{3D}$ with iid Z noise. Using the decoder (a) once for color Y, and (b) once each for R, G, B, and Y and then selecting the lowest weight output. This improves the performance and raises the observed threshold from 0.0055 to 0.008.

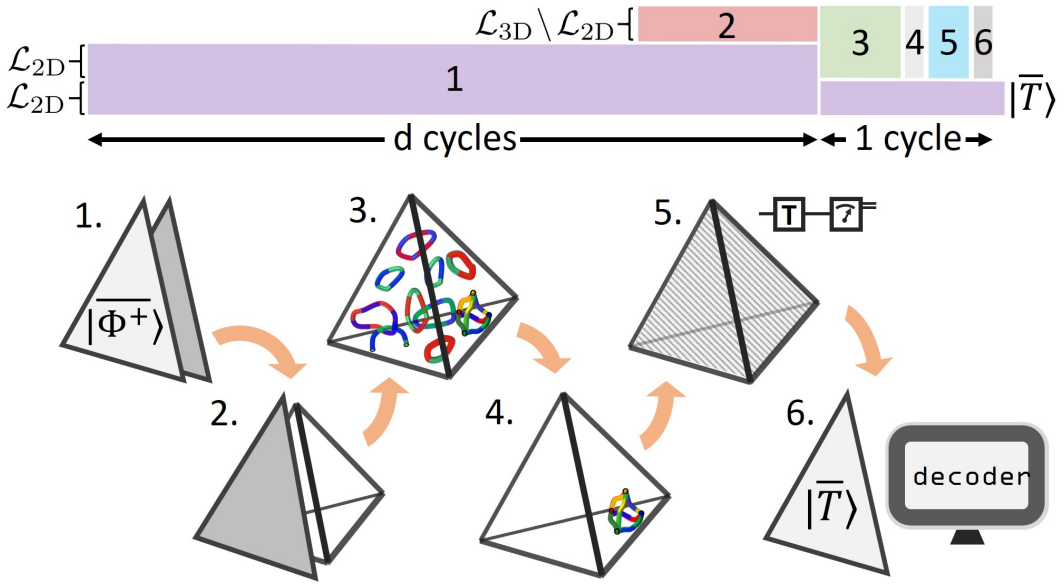


FIG. 28. The protocol to create an encoded magic T state via code switching, with the timeline of each step in units of error correction cycles for the 2D color code. In steps 1 and 2 we simultaneously prepare the Bell state $|\Phi^+\rangle$ in a pair of 2D codes, and the interior of the 3D subsystem color code. In step 3, we measure gauge operators, before fixing them in step 4 to end up in the 3D stabilizer color code. In step 5, $\overline{T}$ is applied and all the qubits are measured. In step 6, a decoder is used to infer the outcome of the logical $\overline{X}$ of the 3D stabilizer color code, and the state of the remaining 2D patch is $|\overline{T}\rangle$ up to a logical $\overline{Z}$.

to avoid a considerable amount of extra noise and simplifies our simulation. More explicitly, the protocol consists of the following steps (which are illustrated in Fig. 28):

1. **Prepare Bell state in 2D codes.**— The encoded Bell state $(|\overline{0}\rangle_{2D}|\overline{0}\rangle_{2D} + |\overline{1}\rangle_{2D}|\overline{1}\rangle_{2D})/\sqrt{2}$ state is fault-tolerantly prepared in a pair of 2D color codes, defined on two copies of the lattice $\mathcal{L}_{2D}$. The second of these 2D color codes should be seen as the code defined along the 2D boundary of the yellow vertex of the 3D lattice as in Fig. 24(b).

2. **Prepare the 3D interior.**— The remaining qubits in $\mathcal{L}_{3D}$, i.e., those which are not near the boundary vertex v_Y, are prepared as a tensor product of unique spherical 2D color code

states. The logical state of the system can then be thought of as a Bell pair between a 2D color code and the 3D subsystem color code $(|\overline{0}\rangle_{2D}|\overline{0}\rangle_{\text{sub}} + |\overline{1}\rangle_{2D}|\overline{1}\rangle_{\text{sub}})/\sqrt{2}$, plus residual error produced during the preparation.

3. **Measure gauge operators.**— All Z-edge operators for all edges not incident to any Y vertex are measured, yielding the subset of edges $\tilde{\gamma}$ corresponding to -1 outcomes.

4. **Gauge fix.**— We find and apply an X-gauge operator which seeks to fix all Z-edge operators to have $+1$ outcomes. The logical state is now a Bell state encoded into the 2D color code and the 3D stabilizer color code $(|\overline{0}\rangle_{2D}|\overline{0}\rangle_{3D} + |\overline{1}\rangle_{2D}|\overline{1}\rangle_{3D})/\sqrt{2}$, plus some residual error.

5. **Apply $\overline{T}$, and measure.**— We apply $\widetilde{T}$ to every data qubit. This, in turn, implements a logical $\overline{T}$ gate, yielding the state $(|\overline{0}\rangle_{2D}|\overline{0}\rangle_{3D} + e^{i\pi/4}|\overline{1}\rangle_{2D}|\overline{1}\rangle_{3D})/\sqrt{2}$, plus some residual error. Then we measure each individual data qubit in the 3D code in the X basis.

6. **Decode Z errors in 3D code.**— We use the single-qubit X-basis measurements to first decode Z-errors, and then infer the outcome $m = \pm 1$ of the logical $\overline{X}$. Then, the state encoded in the 2D color code is $(|\overline{0}\rangle_{2D} + me^{i\pi/4}|\overline{1}\rangle_{2D})/\sqrt{2}$ (plus some residual error), which for $m = -1$ needs to be fixed to the encoded $\overline{T}$ state by application of logical $\overline{Z}$.

In the following subsections, we go through each of the six steps, elaborating on the implementation and simulation details. In our analysis, we adhere to the following guiding principles.

- For each step, we use Monte Carlo simulations under circuit noise to estimate the performance. We select the best error correction techniques, fault-tolerant gadgets and efficient decoding algorithms that we are aware of, and optimize measurement circuits where possible.

- It is possible that some steps will benefit from future improvements in error correction techniques and decoders. We estimate the impact these could have on the performance of this code switching protocol by replacing those steps by a justified estimate of the best improvement one could hope for.

- We choose the fault-tolerant error correction for the 2D color code to be the same optimized configuration we assumed for distillation (see Section III D) to allow for a fair comparison between code switching and distillation.

- We assume a single ancilla qubit per gauge operator in the 3D color code interior.

In Fig. 29(a) we present our findings by showing the overall probability of failure of code switching using our simulations. In Fig. 29(b-f) we indicate the impact of potential improvements of various steps in the protocol on the overall probability of failure of code switching.

1. Preparing the Bell state in 2D codes

To fault-tolerantly prepare the encoded Bell state $(|\overline{0}\rangle_{2D}|\overline{0}\rangle_{2D} + |1\rangle_{2D}|1\rangle_{2D})/\sqrt{2}$ in a pair of 2D color codes of distance d, we first fault-tolerantly prepare them in $|\overline{+}\rangle_{2D}$ and $|\overline{0}\rangle_{2D}$ respectively, and then transversely apply a CNOT from the first to the second. Preparation of $|\overline{+}\rangle_{2D}$ is carried out by initializing all data qubits in the 2D color code in $|+\rangle$, and then performing d rounds of standard error correction (and fixing the inferred initial syndrome of the Z stabilizers). Preparation of $|\overline{0}\rangle_{2D}$ similarly involves preparation in $|0\rangle$, followed by d rounds of standard error correction (and fixing the inferred initial syndrome of the X stabilizers). Since each round of error correction for the 2D

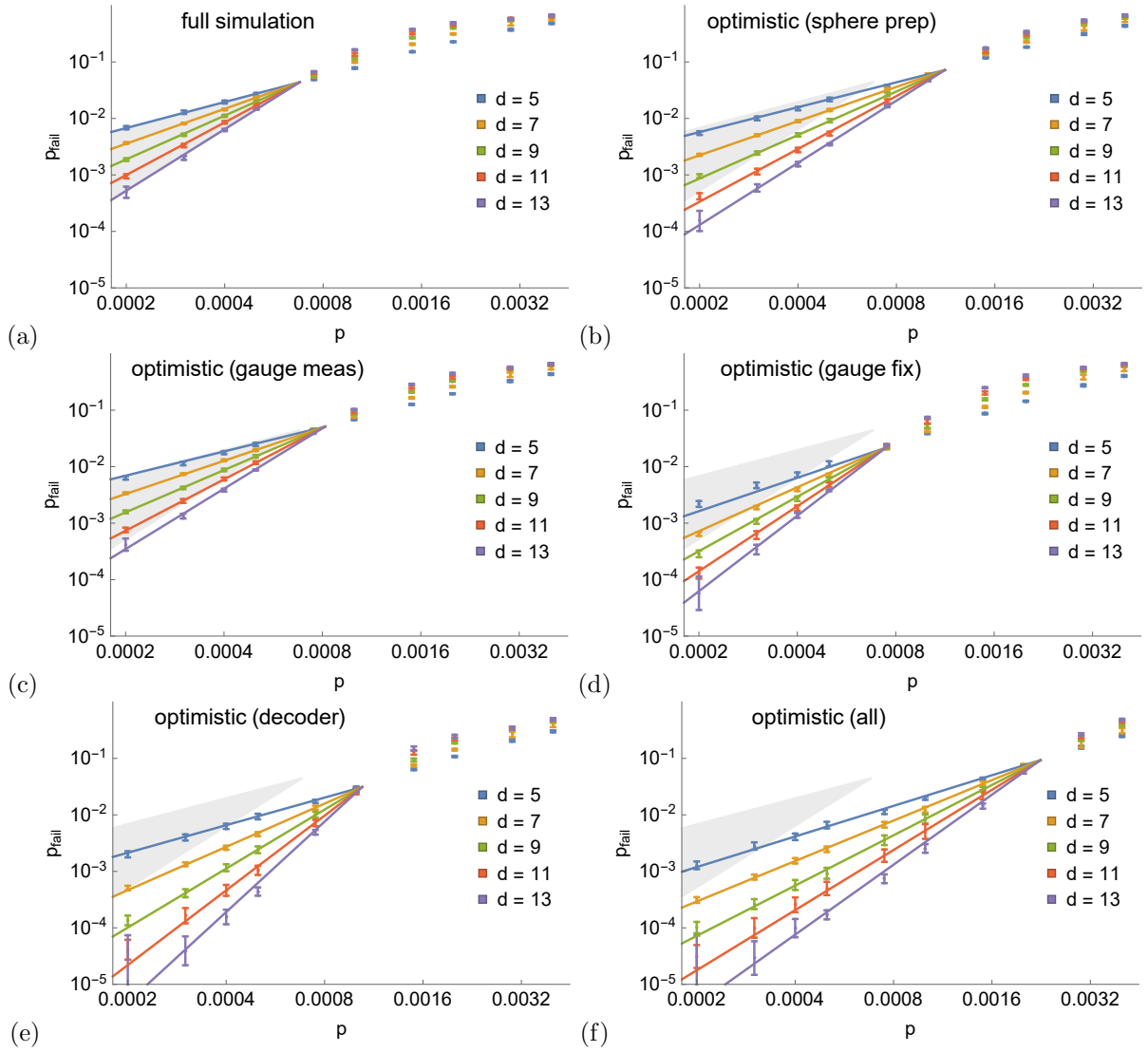


FIG. 29. The failure probability of the code-switching protocol as a function of physical error rate. In (a) every step in the simulation is implemented explicitly under circuit noise with specified circuits and efficient decoders. This therefore represents our circuit level performance analysis for code switching. The fit lines for this case are repeated in each of the remaining sub-figures for comparison. In the next four sub figures we replace just one step by an estimate of the best improvement one could hope for in that step. In particular (b), (c), (d) and (e) show this for sphere preparation, gauge measurement, gauge fixing and the decoder respectively. In (f), all four of these steps are replaced by the estimate of the best improvement one could hope for. For all sub figures, we fit the data to $p_{\text{fail}} = A\,(p/p^*_{\text{CS}})^{(Cd+D)}$ up to the crossing, and include the shaded region between these lines from (a) to guide the eye.

color code requires 9 time steps, the preparation of the encoded Bell state takes $9d$ time steps. After this point the second of the 2D codes undergoes a single additional QEC cycle while code switching occurs for the first 2D code patch. Although it is arbitrary which of the two patches is used for code switching, in practice there is an asymmetry in the X and Z noise for the patches. We find a marginal benefit from feeding the first patch (initialized in $|+\rangle$ prior to the CNOT) to be used for code switching. For details, see Section VI E.

2. Preparing the 3D interior

There are 2D spherical color code states to be prepared around each yellow vertex as that shown in Fig. 24(b). Detailed explanations of this step's implementation and its optimization are given in Section VI D and Section VI E, but we provide the key features here.

Data qubits are prepared in $|+\rangle$, then Z stabilizers are measured with the shortest circuit that uses a single ancilla. If the syndrome extracted for a sphere is *valid*, meaning there is an X operator that will set all stabilizers to have $+1$ outcomes, then such a fix is applied, otherwise the preparation is repeated. If the second iteration also yields an invalid syndrome, one outcome is flipped, producing a valid syndrome which is then fixed. The sphere preparation step is started sufficiently early so that any repeated preparations have finished by the time the two 2D color code patches completes. Note that since the 2D spherical color code encodes no logical qubits, it is not necessary to repeat measurements, in contrast to preparation of a logical state in the 2D color code patch, which requires $O(d)$ cycles.

Potential improvements. One may hope to improve the preparation of the 2D spherical color codes. Although we found a schedule of shortest length to measure the stabilizers, there could be many others of the same length which could potentially lead to fewer errors. Moreover, it may be possible to use the neighboring qubits associated with the RG, RB and GB edges in the lattice, or to share ancilla qubits for preparations of different spheres. No matter how good such a preparation protocol is, it would require at least three time steps as each qubit appears in three Z-type stabilizer generators. Therefore, we use three rounds of iid noise with strength p to bound potential improvements of this step in Fig. 29(b).

3. Measuring gauge operators

To change gauge to the 3D stabilizer code, the Z-edges of color in $\mathcal{K} = \{RG, RB, GB\}$ are measured. A detailed explanation of this step's implementation and its optimization is given in Section VI F. We find a minimal length circuit to measure the gauge operators in parallel with a single ancilla qubit per edge. Including ancilla preparation and measurement, this circuit requires 8 time steps, although the preparation can be started during the last time step of the preparation round.

Since we choose to measure only Z-edge operators of color in $\mathcal{K}$, not all Z-edge operators, we only learn the restricted noisy gauge flux $\widetilde{\gamma}^{\mathcal{K}}$. We emphasize that due to faults in the measurement process, $\widetilde{\gamma}^{\mathcal{K}}$ is likely to violate the Gauss law and differ from the restricted gauge flux $\gamma^{\mathcal{K}}$ in the system.

Potential improvements. It seems difficult to reduce the errors introduced by this step without increasing the space or time overhead significantly, and for example to use verified cat states to measure each edge operator. We will neglect any space or time overhead in our estimate so that we bound the effect of potential improvements. As each data qubit is in three measurements, a minimum of three time-steps would be required to implement the measurement. Therefore, we use three rounds of iid noise with strength p in our optimistic simulation in Fig. 29(c).

4. Gauge fixing

This step corresponds to a classical algorithm which takes as its input the noisy gauge flux $\tilde{\gamma}^{\mathcal{K}} \subseteq \Delta_1'(\mathcal{L}_{3\mathrm{D}}^{\mathcal{K}})$ corresponding to some subset of edges of color in $\mathcal{K} = \{RG, RB, GB\}$, and outputs some X-type gauge operator aiming to fix $\tilde{\gamma}^{\mathcal{K}}$. The algorithm proceeds as follows.

(i) Validation of the noisy flux: for any color $K \in \mathcal{K}$ we validate the noisy restricted flux $\widetilde{\gamma}^K$ by matching its termination points within the restricted lattice $\mathcal{L}_{3D}^{RG}$. We denote by $\lambda^K \subseteq \Delta_1'(\mathcal{L}_{3D}^K)$ the set of edges used in the pairing of the termination points of $\widetilde{\gamma}^K$. Then, $\hat{\gamma} = \sum_{K\in\mathcal{K}}(\widetilde{\gamma}^K + \lambda^K)$ is the validated flux. Note that $\hat{\gamma} \subseteq \Delta_1'(\mathcal{L}_{3D}^{\mathcal{K}})$.

(ii) Gauge-fixing from the validated flux: we find an operator $\prod_{e\in f(\hat{\gamma})} X(e)$ consistent with $\hat{\gamma}$ and apply it. Such an X-gauge operator can be found by e.g. Gaussian elimination.

In this step, we are treating the measured gauge flux as if it has no branching points; see Fig. 30. In other words, we think of the flux as if it was due to some X-gauge operator. However the gauge flux can in fact have branching points due to X errors in the system. Moreover the measured gauge flux may not even satisfy the Gauss law due to the noisy implementation of the measurements, and the omission of Z edge operators incident to Y vertices which are not measured. Thus, in order to convert the measured flux into one which satisfies the Gauss law and has no branching points, we pair up break points for the flux of the same color independently for each color RG, RB and GB.

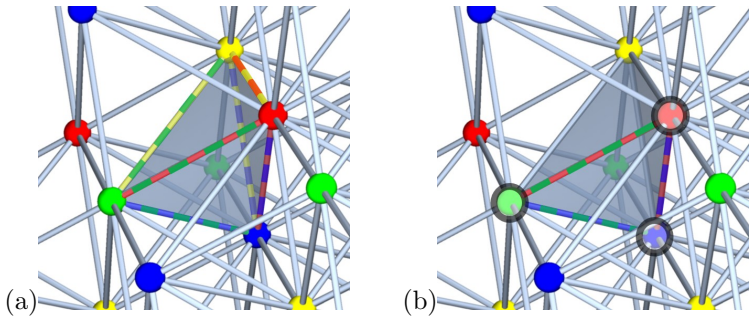


FIG. 30. (a) A Z-gauge flux due to an isolated X error (shaded tetrahedron) consists of six edges of different colors. (b) The restricted gauge flux which is measured (assuming no errors occur during the measurement) excludes edges incident to Y vertices. The gauge fixing algorithm in this scenario would connect the marked vertices in each sub-graph thereby removing the RG, RB and BG edges in the measured restricted gauge flux as if they were each erroneous, leaving the X error present in the system following gauge fixing. An improved algorithm could potentially correct this error.

Potential improvements. No additional noise is introduced in this step due to the depolarizing channel, but one can ask how close to optimal the performance of the algorithm is. For instance, it may be possible to not only estimate a gauge fix from the noisy Z-flux measurement, but also to correct X errors in the system; see Fig. 30. To bound any potential improvements of this step in our simulation, we assume that all X errors in the system prior to the gauge measurements are corrected, and that the gauge itself is identified exactly; see Fig. 29(d). The only remaining X error that is retained in this bound is therefore that which is introduced by the noisy gauge measurement circuits themselves.

5. *Applying T and measuring the 3D code's data qubits*

In this step, we first apply the $\widetilde{T}$ gate to each data qubit, and then measure it in the X basis. These can be combined into a single operation requiring one time unit. Despite being one of the simplest code switching steps to implement, it is quite challenging to simulate efficiently, because the non-Clifford $\tilde{T}$ gates propagate errors in an intricate way. Faithfully simulating general non-Clifford circuits is not believed to be efficient on classical computers. In what follows we describe how we exploit some additional structure and assumptions that render the simulation tractable.

Let the X-type residual error before the application of the logical $\overline{T}$ gate be supported on $\rho_X \subseteq \Delta_3'(\mathcal{L}_{3\mathrm{D}})$; similarly, ρ_Z denotes the Z-type residual error. Note that if we could perfectly fix the gauge in the previous step and there were no additional X-errors, then there would be no residual X-error. Generically, there is some nontrivial residual X-error, which unlike the Z-error does not commute with the logical $\overline{T}$ gate. The effect of the residual X-error on the measurement outcomes amounts to, roughly speaking, flipping some outcomes along the Z-gauge flux $\gamma = \partial_{3,1}\rho_X$ of the the residual X-error ρ_X. This is made precise in the following lemma.

Lemma VI.1. *Let $\rho_X, \rho_Z \subseteq \Delta_3'(\mathcal{L}_{3D})$ be residual X- and Z-errors. Let $\gamma = \partial_{3,1}\rho_X$ be the Z-gauge flux and $\gamma = \sum_{j=1}^{b} \lambda_j$ be its decomposition into its linked components. If we apply $\widetilde{T}$ to every data qubit and then measure it in the X basis, the outcomes will form an X-type syndrome σ satisfying*

$$\sigma = \partial_{3,1}\rho_Z + \sum_{j=1}^{b} \sigma_j, \tag{36}$$

where the excitation configuration $\sigma_j \in \Sigma(\lambda_j)$ is chosen uniformly at random from $\Sigma(\lambda_j)$.

Here we have used notation introduced from Section V B, and the justification for this lemma can be found in [96].

To simulate this step in code switching, we aim to generate a Z-type Pauli operator τ_Z which will result in the same success probability as the true system when used in the following step. Recall that any Pauli operator E can be uniquely decomposed as $E = RSL$, where R is a fixed correction for the syndrome ∂E, S is some stabilizer operator, and L is some representative of the logical operator. For the simulation to faithfully describe what happens in the system we require the Z-type operator generated in the simulation τ_Z and that present in the system ρ_Z have matching syndromes and logical components.

In our simulation we make the additional simplifying assumption to sample σ_j uniformly from within the collection of excitation configurations without the linking charge $\Sigma'(\lambda_j)$ rather than the collection of excitation configurations $\Sigma(\lambda_j)$. Recall that $\Sigma'(\lambda_j) \subseteq \Sigma(\lambda_j)$. This assumption amounts to ignoring the linking charge, and we do not expect it to substantially effect the performance. We generate the random Z-error $\tau_Z \subseteq \Delta_3'(\mathcal{L}_{3\mathrm{D}})$ as

$$\tau_Z = \sum_{v \in \Delta_0(\gamma)} \tau(\gamma|_v), \tag{37}$$

where $\tau(\gamma|_v)$ is a local sampling procedure, which we now describe in detail. Let $v \in \Delta_0'(\gamma)$ be a vertex of color A, which is incident to the flux $\gamma = \partial_{3,1}\rho_X$, and $\gamma|_v$ be the restriction of γ to the edges incident to v. Let $\mathcal{K}' = \{R, G, B, Y\} \setminus \{A\}$ denote the set of three different colors. We find a subset $\tau(\gamma|_v) \subseteq \partial_{0,3} v$ of tetrahedra containing v as follows.

(i) With probability $1/2$ set $\Xi = 0$; otherwise $\Xi = 1$.

(ii) For each $K \in \mathcal{K}'$, if $\Xi \prod_{K \in \mathcal{K}} \left|\gamma|_v^{AK}\right| = 0$, then choose uniformly at random a subset $E^{AK} \subseteq \gamma|_v^{AK}$ of even cardinality; otherwise choose uniformly at random a subset $E^{AK} \subseteq \gamma|_v^{AK}$ of odd cardinality.

(iii) Find a subset of tetrahedra $\tau(\gamma|_v)$, whose 1-boundary locally matches $\sum_{K \in \mathcal{K}'} E^{AK}$, i.e.,

$$(\partial_{3,1}\tau(\gamma|_v))|_v = \sum_{K \in \mathcal{K}'} E^{AK}. \tag{38}$$

We now explain why this algorithm produces $\tau_Z \subseteq \Delta_3'(\mathcal{L}_{3D})$ with the correct syndrome distribution. First, any excitation configuration $\sigma_j \in \Sigma'(\lambda_j)$ can be viewed as a sum of local excitation configurations with neutral total charge. If the vertex v is in $\Delta_0'(\lambda_j)$, then step (ii) is equivalent to randomly selecting a local excitation configuration for $\lambda_j|_v$, which is created by operators supported within the neighborhood $\partial_{0,3}v$ of v and with the neutral total charge. Since each local excitation configuration is equally likely selected, thus the resulting excitation configuration σ_j is chosen uniformly at random from $\Sigma'(\lambda_j)$. Also note that the search in (iii) can be implemented by exhaustively checking which of the possible subsets of tetrahedra $\partial_{0,3}v$ containing v satisfies Eq. Eq. (38). This naive implementation is nevertheless efficient since $|\partial_{0,3}v|$ is bounded. Lastly, we remark that the residual Z-error present in the system right before the measurement in the X basis is $\rho_Z + \tau_Z$. We use this Z-error in the following code switching step.

6. Decoding Z errors in the 3D color code

In the final step of the protocol, a classical decoding algorithm is run to correct Z errors for the 3D stabilizer color code. The input to the decoder is the set of X measurement outcomes for each data qubit from the previous step. The output of the decoder will be a Z-type Pauli correction which, if applied, would flip all the syndromes computed given the single-qubit X outcomes. At that point one can reliably read off the logical $\overline{X}$ measurement $m = \pm 1$. Then, the state encoded in the remaining 2D color code from step 1 is $(|\overline{0}\rangle_{2D} + me^{i\pi/4}|\overline{1}\rangle_{2D})/\sqrt{2}$ (plus some residual error), which is the $\overline{T}$ state, up to the multiplication of $\overline{Z}$ for $m = -1$.

We implement a decoder based on the restriction decoder of Kubica and Delfosse [31], but modified in two ways. The first modification is to adapt the decoder, which was originally defined for the 3D stabilizer color code on the 3-torus, to the 3D stabilizer color code on the lattice $\mathcal{L}_{3D}$ which has boundaries that we consider in this paper. The second modification is to improve performance. We do this by favoring low-weight corrections (which differ by a stabilizer from the output of the original decoder). Running four independent versions of this decoder in parallel, we then simply select the lowest-weight correction from the four candidates. We describe and analyze the performance of this modified restriction decoder in Section V C.

Potential improvements. It is possible that an improved decoder could be developed which would improve the performance of code switching. It is difficult to give as rigorous bounds on the performance of this step as we were able to give for the previous steps. However, by assuming that the best decoder performs as if the noise was iid, we estimate the effect of any potential performance improvements using the following steps:

(i) Run the modified restriction decoder, and if it succeeds then we assume the improved version would also succeed, if it fails then continue to the next step.

(ii) Let w be the minimum of the weight of the error, and the weight of the correction produced by the modified restriction decoder. Let n be the number of data qubits in the distance-d 3D stabilizer color code. Uniformly choose a real number $0 \leq q \leq 1$.

$$\text{If } q < p^{(1)}_{3\text{DCC}} \left(\frac{w/n}{p^{(1)}_{3\text{DCC}}} \right)^{\frac{d+1}{2}}, \quad \text{then the correction fails, otherwise it succeeds.}$$

Note that $p^{(1)}_{3\text{DCC}} \simeq 0.019$ is the known threshold of the optimal decoder for the 3D stabilizer color code under iid Z noise [92]. We provide more detailed justifications for this estimate in Section VI G. The impact of potential improvements of this step on code switching is shown in Fig. 29(e).

B. Code switching overhead

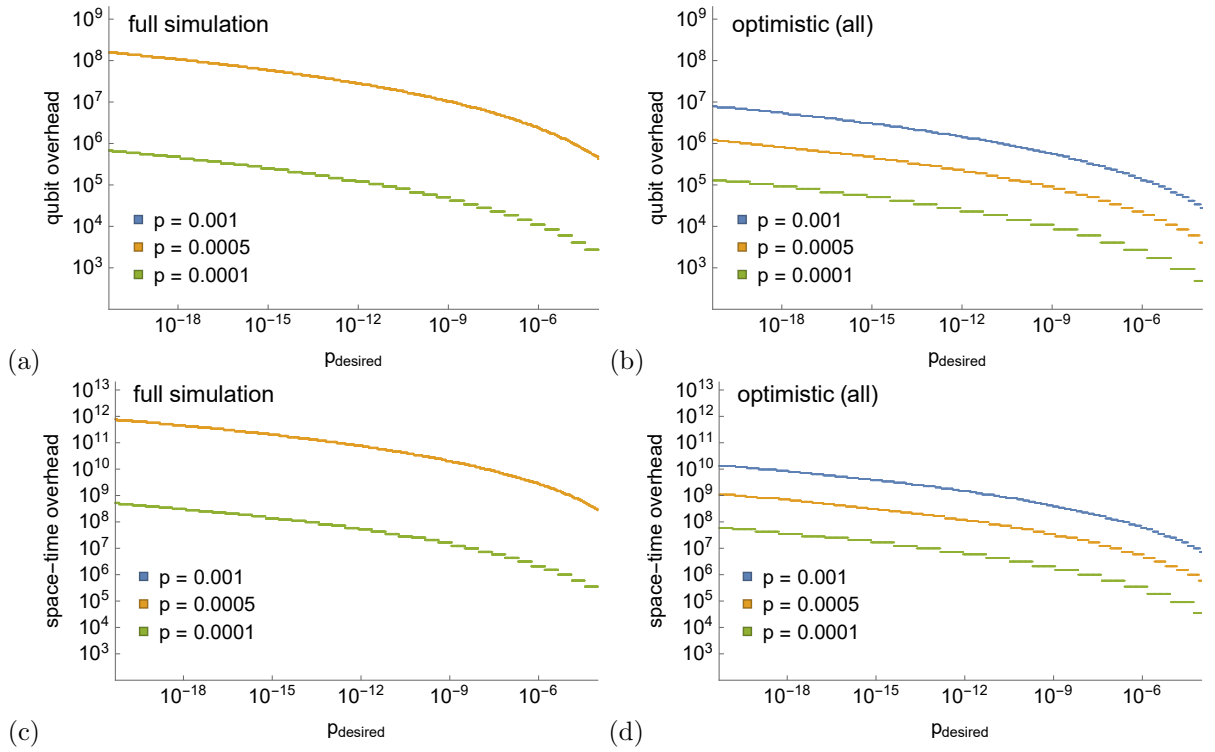


FIG. 31. The overhead of code switching to produce an encoded magic state of a desired quality. (a) Space overhead. (b) Space overhead assuming all optimistic improvements can be realized. (c) Space-time overhead. (d) Space-time overhead assuming all optimistic improvements can be realized. There is no curve for $p = 0.001$ without assuming optimistic improvements to the protocol as this is higher than the observed threshold for code switching.

To calculate the space and time overhead required to produce a $\overline{T}$ state with failure probability at most p_{target}, we first find the minimum distance $d(p_{\text{target}})$ which achieves this by extrapolating the data in Fig. 29(a). The time to implement the code switching protocol at distance d is simply $9(d+1)$, which is the time for $(d+1)$ QEC cycles of the 2D color code to complete. The number of qubits to implement the code switching protocol at distance d is

$$N_{\text{CS}}(d) = (26d^3 + 51d^2 + 22d - 51)/24, \tag{39}$$

which is comprised of two 2D color code patches (each of which has $(1+3d^2)/4$ data qubits and $3(d+1)(d-1)/4$ measurement qubits), and the 3D interior (which has $d(d^2+1)/2 - (1+3d^2)/4$ data qubits and $(d-1)(7d^2+10d+15)/12 - 3(d+1)(d-1)/8$ measurement qubits). Details of the lattices which make clear where these numbers originate from can be found in Section VI A.

We fit the code switching data in Fig. 29 using the numerical ansatz

$$p_{\text{fail}} = A\left(\frac{p}{p^*_{\text{CS}}}\right)^{(Cd+D)}. \tag{40}$$

If we solve for d as a function of p and p_{fail}, then we obtain

$$d = \frac{\log(p_{\text{fail}}/A)}{C\log\left(p/p^*_{\text{CS}}\right)} - D/C, \tag{41}$$

where we round the right hand side up to the closest odd integer. Finally, in Fig. 31 we substitute this into $N_{\mathrm{CS}}(d)$ and plot the qubit overhead and the space-time overhead as a function of p_{fail} for various values of p for two cases: assuming no potential improvements, and assuming all potential improvements can be achieved.

ACKNOWLEDGMENTS

A.K. is deeply indebted to Héctor Bombín for many colorful discussions. A.K. acknowledges funding provided by the Simons Foundation through the "It from Qubit" Collaboration. Research at Perimeter Institute is supported in part by the Government of Canada through the Department of Innovation, Science and Economic Development Canada and by the Province of Ontario through the Ministry of Colleges and Universities. This work was completed prior to A.K. joining AWS Center for Quantum Computing.

# vertices in face	# faces
4	$3(d-1)/2$
6	$3(d-3)(d-1)/8$
Total:	$3(d+1)(d-1)/8$

TABLE II. Parameters of the direct lattice used to define the 2D stabilizer color code for distance d. The faces are 3-colorable, such that no two faces of the same color share an edge. There are a total of $(1+3d^2)/4$ vertices in the lattice, and $9(d+1)(d-1)/8$ edges.

# tetrahedra containing vertex	# vertices	# tetrahedra containing edge	# edges
$(1+3d^2)/4$	4	d	6
8	$2(d-1)$	4	$(d^3+3d^2+11d-15)/4$
12	$(d-3)(d-1)/2$	6	$(d-3)(d-1)(2d+5)/6$
18	$(d-3)(d-1)/2$		
24	$(d-5)(d-3)(d-1)/12$		
Total:	$(d-1)(d+1)(d+3)/12+4$	Total:	$(d-1)(7d^2+10d+15)/12+6$

TABLE III. Parameters of the dual lattice used to define the 3D subsystem and stabilizer color codes for distance d. The vertices are 4-colored , such that no two vertices of the same color share an edge. There are a total of $d(d^2+1)/2$ tetrahedra in the lattice, each of which contains precisely 4 vertices, and 6 edges all of distinct colorings. Note that there are no stabilizer or gauge generators associated with the 4 vertices of weight-$(1+3d^2)/4$ or the 6 edges of weight-d for either of the 3D stabilizer or subsystem color codes.

APPENDICES

A. Lattice parameters and specifications

Here provide parameters of the lattices which are used to define the 2D and 3D color codes. Table II provides parameters of the direct lattice used to define the 2D stabilizer color code for distance d, which is the family of lattices constructed from hexgonal tilings with triangular boundary [34]. Table III provides parameters of the dual lattice used to define the 3D subsystem and stabilizer color codes for distance d. This is constructed from a pair of BCC lattices with diagonal edges added and tetrahedra defined that form a 4-colorable lattice, with a tetrahedral boundary [56].

B. Supplementary details for 2D color code simulations

Here we provide additional supporting data for the analysis of the performance of the 2D color code using the faulty-measurement projection decoder under phenomenological noise and circuit noise. In Fig. 32 we show the performance under phenomenological noise, which we use to produce the long-time pseudo-threshold in Fig. 11(d) in the main text. Similarly, in Fig. 33 we show the performance under phenomenological noise, which we use to produce the long-time pseudo-threshold in Fig. 13(b) in the main text. We point out that the decay parameter in the fit of the time-dependent pseudo-threshold in Fig. 32(b) and Fig. 33(b) shows that stabilization toward

the long-time pseudo-threshold seems not to depend significantly on the system size. In Fig. 34 we provide the data used to produce the crossings in Fig. 13(b). In Fig. 35 we show the impact of optimizing the stabilizer extraction circuit on the performance of error correction under circuit noise.

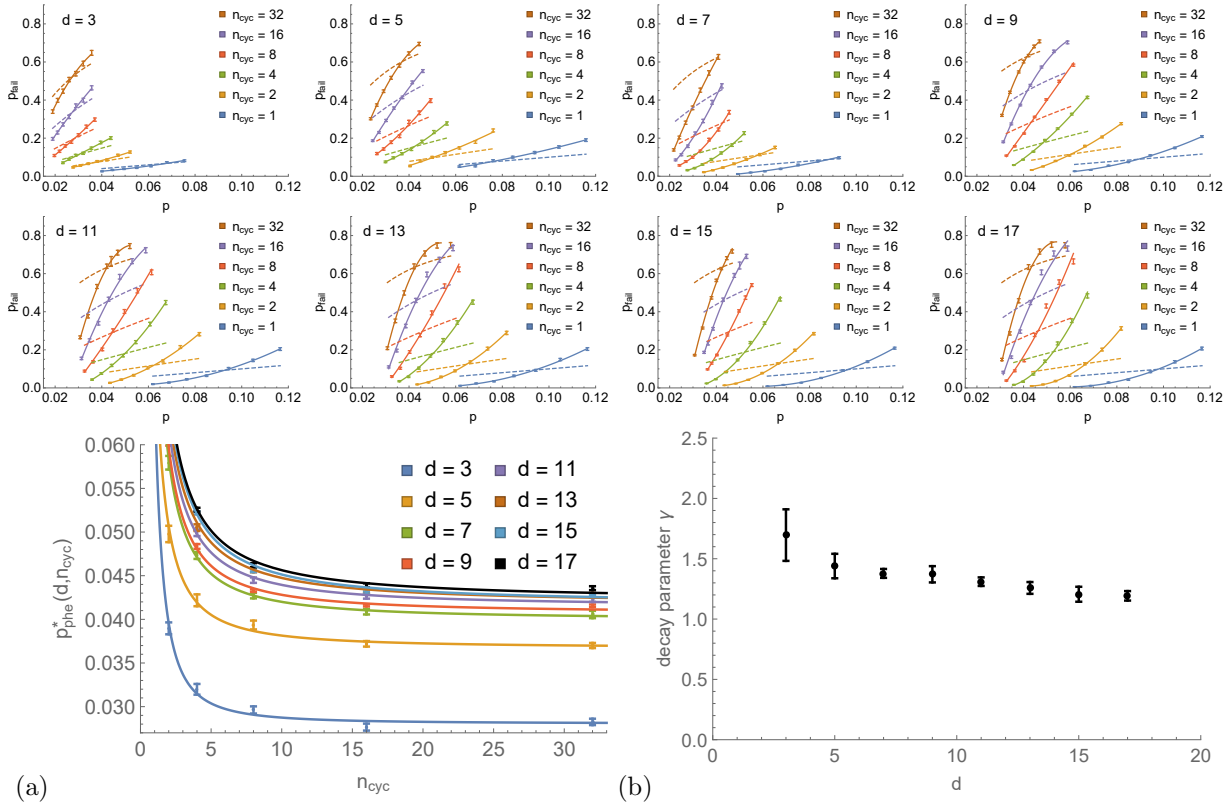


FIG. 32. (Top two rows) The failure probability of n_{cyc} EC cycles for phenomenological noise of strength p given a perfect initial state and final measurement round, for various distances d. We estimate the time-dependent pseudo-threshold $p^*_{\mathrm{phe}}(d, n_{\mathrm{cyc}})$ as an intersection of quadratic fits (solid lines) with the corresponding physical qubit error probability $p_{\mathrm{phy}}(n_{\mathrm{cyc}})$ (dashed curves). (a) The time-dependent pseudo-threshold $p^*_{\mathrm{phe}}(d, n_{\mathrm{cyc}})$ as a function of n_{cyc}. We estimate the long-time pseudo-thresholds $p^*_{\mathrm{phe}}(d)$ by fitting $p^*_{\mathrm{phe}}(d, n_{\mathrm{cyc}})$ with the ansatz in Eq. Eq. (10) and in (b) we plot the decay parameter γ. We observe that γ stabilizes rapidly with increasing d, confirming that the residual noise reaches equilibrium over a time which is independent of system size.

Here we provide additional supporting data for the analysis of the performance of the logical operations of the 2D color code under circuit noise. In Fig. 36, we show additional data from which we extract the logical failure rates for $p = 0.001, 0.0001$ in Fig. 16(b) in the main text.

C. Supplementary details for distillation analysis

Here we more details for the distillation analysis in Section IV.

The lowest order failure probability of the encoded state (given accepted syndrome measurements) was of the form $q + c_1 p_1 + c_2 p_2$, where q is the probability that $|T\rangle$ is replaced by $Z|T\rangle$ of error for the T gate, p_p for preparation, p_m for measurement, p_1 for idle qubits, and p_2 for CNOT gates. For all minimum-length sequences, $\frac{7}{3} \leq c_1 \leq 8$ and $\frac{14}{15} \leq c_2 \leq \frac{62}{15}$. The sequence given in the main text minimizes both c_1 and c_2. To highlight the value of this optimization, we point out that

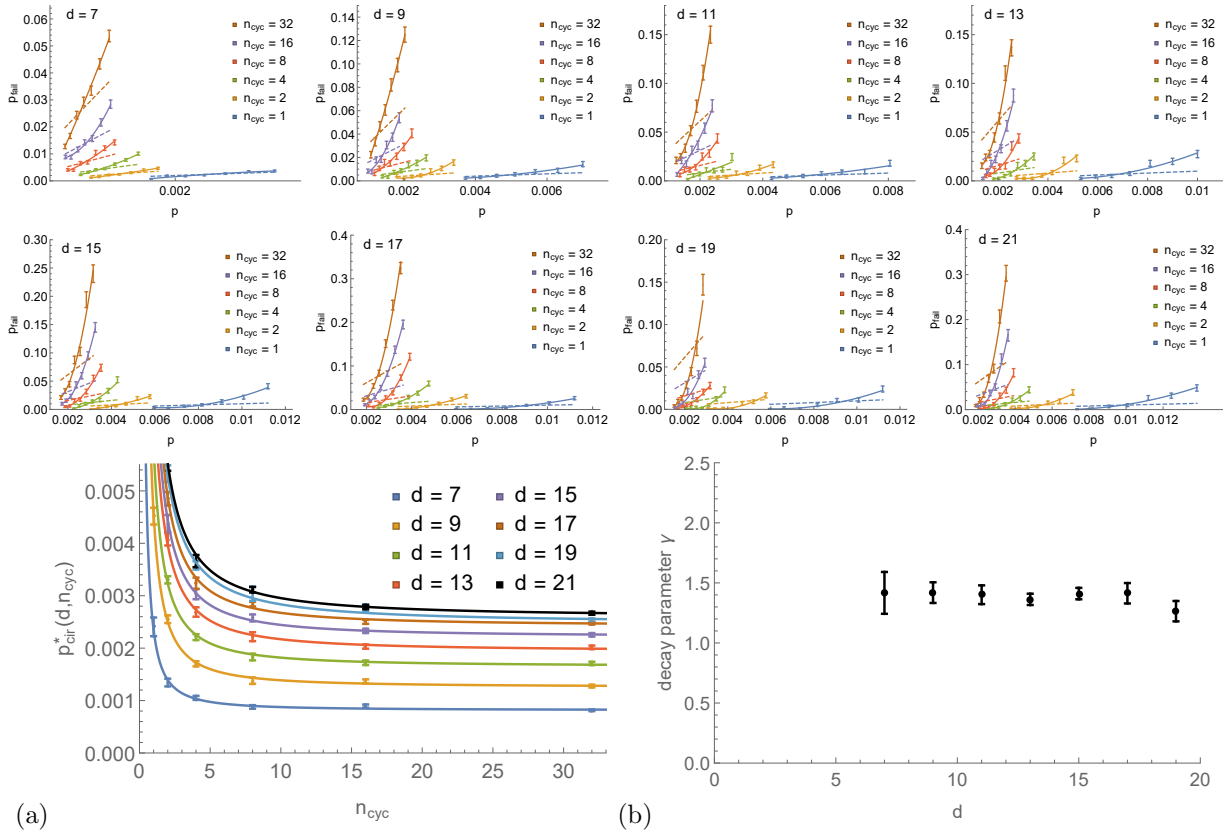


FIG. 33. (Top two rows) The failure probability of n_{cyc} EC cycles for circuit noise of strength p given a perfect initial state and final measurement round, for various distances d. We estimate the time-dependent pseudo-threshold $p^*_{\text{cir}}(d, n_{\text{cyc}})$ as an intersection of quadratic fits (solid lines) with the corresponding physical qubit error probability $p_{\text{phy}}(n_{\text{cyc}})$ (dashed curves). (a) The time-dependent pseudo-threshold $p^*_{\text{cir}}(d, n_{\text{cyc}})$ as a function of n_{cyc}. We estimate the long-time pseudo-thresholds $p^*_{\text{cir}}(d)$ by fitting $p^*_{\text{cir}}(d, n_{\text{cyc}})$ with the ansatz in Eq. Eq. (10) and in (b) we plot the decay parameter γ. We observe that γ stabilizes rapidly with increasing d, confirming that the residual noise reaches equilibrium over a time which is independent of system size.

the worst performing minimum-length sequence, $\{6, 7, 4, 3, 2, 5; 5, 6, 3, 2, 1, 4\}$, had a failure rate of more than twice that of the best performing sequence.

D. Preparing the 3D top for code switching

Here we consider the 3D top of the initial state for code switching discussed in Section VI A 2, which consists of a (trivial) 2D color code state on the sphere surrounding each Y interior vertex (see Fig. 37). There are 4 sizes of sphere consisting of 8, 12, 18 and 24 qubits respectively.

This state of each sphere is formed by preparing all data qubits in the $|+\rangle$ state, then measuring the Z stabilizers, each with an ancilla measurement qubit using a single in a single round of error correction, followed by the application of an appropriate X Pauli to fix any stabilizers which are incorrect.

CNOT circuits.— For each of the four types of sphere, we find an ordering of CNOTs which allows for a minimal-length circuit to measure the stabilizers (see Table V and Table VI). These sequences were found using a greedy algorithm with a random initial sequence and minimizing the cost function of the number of unsatisfied constraints (we require that no qubit is involved in more

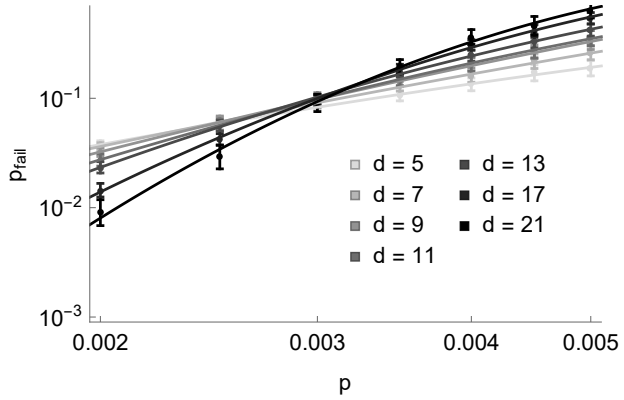


FIG. 34. The failure probability $p_{\text{fail}}(d, n_{\text{cyc}})$ of the noisy-syndrome projection decoder for $n_{\text{cyc}} = d$ rounds of circuit noise and various distances d. We use this data to find the failure curve crossings $p^{\times}_{\text{cir}}(d)$, which we plot in Fig. 13(b).

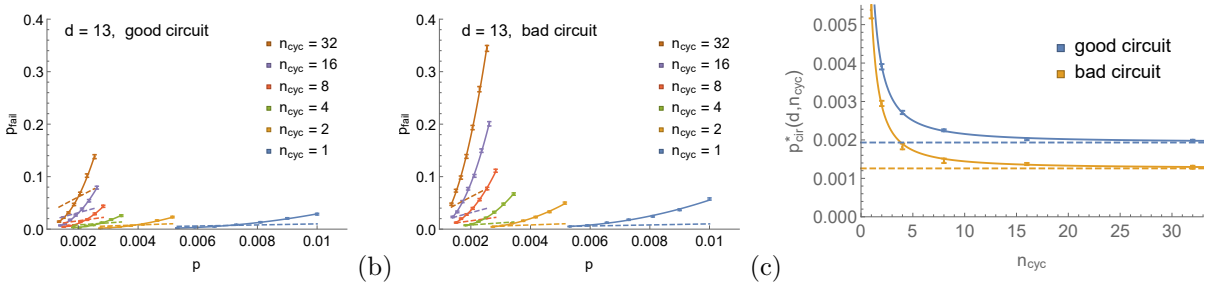


FIG. 35. Performance comparison of a good and a bad length-7 CNOT extraction circuit, which are the top and bottom ranked circuits from Fig. 13(a), namely $\{1, 4, 5, 6, 3, 2; 2, 3, 4, 7, 6, 5\}$ and $\{1, 4, 3, 6, 7, 2; 6, 1, 2, 5, 4, 7\}$. For the good circuit in sub-figure (a) and for the bad circuit in sub-figure (b), we numerically estimate the failure probability of n_{cyc} EC cycles for circuit noise of strength p given a perfect initial state and final measurement round, for $d = 13$. We fit this data with quadratic functions (solid lines), and also show the physical qubit error probability $p_{\text{phy}}(n_{\text{cyc}})$ (dashed curves), and we estimate the pseudo-threshold $p^*_{\text{cir}}(d, n_{\text{cyc}})$ which we plot in sub-figure (c) by identifying the intersection of the solid and dashed curves for a given d, p and n_{cyc}. We see that the long-time pseudo-threshold for the good circuit is 0.00193 ± 0.00002 (blue, dashed), considerably larger than that of 0.00126 ± 0.00003 for the bad circuit (yellow, dashed).

than one gate per time unit) minus the average time unit when each gate occurs. We fixed the total number of time units to be minimal (4 CNOT time units for the 8-qubit sphere in Table V which involves only weight-4 stabilizers, and 6 CNOT time units for the 12-, 18- and 24-qubit spheres in Table VI which involve weight-6 stabilizers). We maximize the average time unit when each gate occurs since this removes idle time units by not preparing ancillas for lower-weight stabilizer measurements until they are needed. The sequences we found are optimal with regard to this cost function: all constraints were satisfied using the minimal number of time units, and all weight-r stabilizer generators have CNOTs in the last r time units of each schedule. There could be many schedules which are optimal in this sense, but we did not attempt to explore among those for schedules which performed better.

To make the representation of the schedules slightly more compact, we specify 17 vertex locations

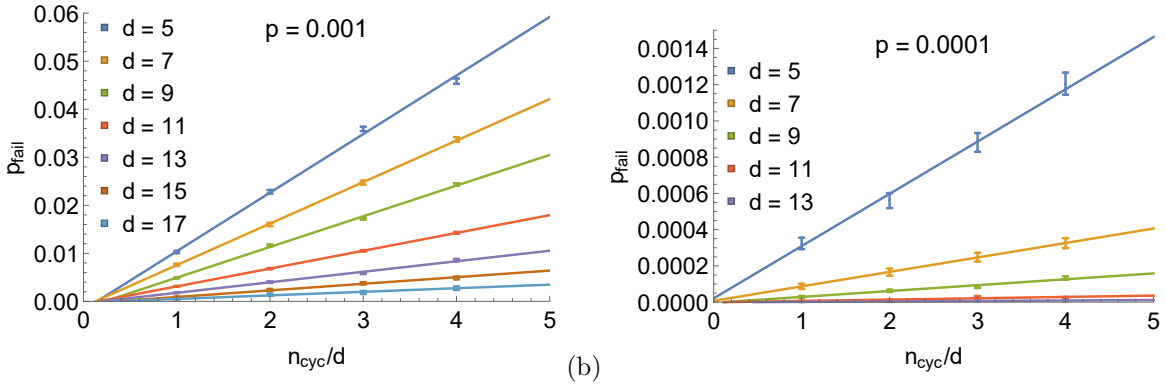


FIG. 36. Failure probability as a function of n_{cyc}/d for a variety of distances d. Each plot is for a fixed value of p as specified.

Punctured qubit	Failure Probability	Rejection Probability
1	$14p_2/15 + p_p \ \ (= 1.93p)$	$589p_2/30 + 45p_p/4 \ \ (= 30.88p)$
2	$13p_2/15 \ \ (= 0.87p)$	$119p_2/6 + 49p_p/4 \ \ (= 32.08p)$
3	$5p_2/6 \ \ (= 0.83p)$	$593p_2/30 + 49p_p/4 \ \ (= 32.02p)$
4	$9p_2/10 + p_p \ \ (= 1.90p)$	$119p_2/6 + 47p_p/4 \ \ (= 31.58p)$
5	$14p_2/15 \ \ (= 0.93p)$	$589p_2/30 + 49p_p/4 \ \ (= 31.88p)$
6	$13p_2/15 + p_p/2 \ \ (= 1.37p)$	$119p_2/6 + 12p_p \ \ (= 31.83p)$
7	$5p_2/6 + p_p/2 \ \ (= 1.33p)$	$593p_2/30 + 12p_p \ \ (= 31.77p)$
8	$9p_2/10 + 2p_p \ \ (= 2.90p)$	$119p_2/6 + 45p_p/4 \ \ (= 31.08p)$
9	$14p_2/15 + p_p \ \ (= 1.93p)$	$589p_2/30 + 45p_p/4 \ \ (= 30.88p)$
10	$13p_2/15 \ \ (= 0.87p)$	$119p_2/6 + 49p_p/4 \ \ (= 32.08p)$
11	$5p_2/6 \ \ (= 0.83p)$	$593p_2/30 + 49p_p/4 \ \ (= 32.02p)$
12	$9p_2/10 + p_p \ \ (= 1.90p)$	$119p_2/6 + 47p_p/4 \ \ (= 31.58p)$
13	$14p_2/15 \ \ (= 0.93p)$	$589p_2/30 + 49p_p/4 \ \ (= 31.83p)$
14	$13p_2/15 + p_p/2 \ \ (= 1.37p)$	$119p_2/6 + 12p_p \ \ (= 31.83p)$
15	$5p_2/6 + p_p/2 \ \ (= 1.33p)$	$593p_2/30 + 12p_p \ \ (= 31.77p)$
16	$9p_2/10 + 2p_p \ \ (= 2.90p)$	$119p_2/6 + 45p_p/4 \ \ (= 31.08p)$

TABLE IV. 15-to-1 scheme analysis for different choices of punctured qubit. Only faults that occur in the preparation or CNOT circuit are included here (as the other fault locations are independent of the choice of punctured qubit). The parameters p_1, p_2, p_p, p_m and q are the probabilities of single-qubit gate, two-qubit gate, preparation, measurement and magic state errors respectively. We also set $p_1 = p_2 = p_p = p_m = p$ to allow for an easy comparison.

in a list v_{list} in lexicographical order:

$$\begin{aligned} v_{\text{list}} = \Big\{ & (-1,\ 0,\ 0)\,, \left(-\tfrac{1}{2}, -\tfrac{1}{2}, -\tfrac{1}{2}\right), \left(-\tfrac{1}{2}, -\tfrac{1}{2}, \tfrac{1}{2}\right), \left(-\tfrac{1}{2},\ \tfrac{1}{2}, -\tfrac{1}{2}\right), \left(-\tfrac{1}{2},\ \tfrac{1}{2},\ \tfrac{1}{2}\right), \\ & (\ 0, -1,\ 0)\,, (\ 0,\ 0, -1)\,, (\ 0,\ 0,\ 0)\,, (\ 0,\ 0,\ 1)\,, (\ 0,\ 1,\ 0)\,, \left(\ \tfrac{1}{2}, -\tfrac{1}{2}, -\tfrac{1}{2}\right), \\ & \left(\ \tfrac{1}{2}, -\tfrac{1}{2}, \tfrac{1}{2}\right), \left(\ \tfrac{1}{2},\ \tfrac{1}{2}, -\tfrac{1}{2}\right), \left(\ \tfrac{1}{2},\ \tfrac{1}{2},\ \tfrac{1}{2}\right), (\ 1, -1, -1)\,, (\ 1,\ 0,\ \ 0)\,, \left(\ \tfrac{3}{2},\ \tfrac{3}{2},\ \tfrac{3}{2}\right) \Big\}, \end{aligned} \tag{42}$$

where the coordinates $(\ 1, -1, -1)$ and $\left(\ \frac{3}{2},\ \frac{3}{2},\ \frac{3}{2}\right)$ are for placing boundary vertices v_G and v_R, while the others describe the interior vertices. The colors of the vertices in v_{list} in order are G,

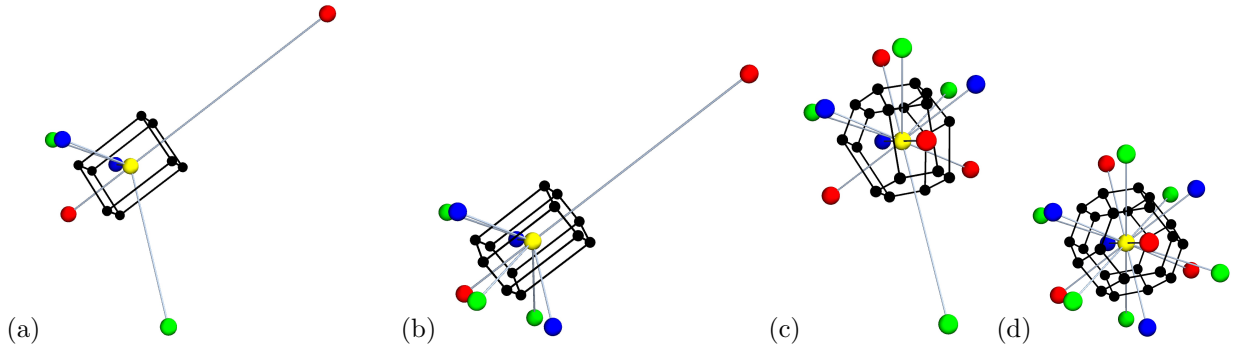


FIG. 37. Dual and primal lattices for the 2D spherical color codes which appear in the preparation of interior of the 3D color code. Qubits are black dots, and stabilizers are grey edges with colored vertices. For each, the subset of vertices from v_{list} in Eq. (42) are: (a) $\{1, 2, 3, 4, 8, 15, 17\}$, (b) $\{1, 2, 3, 4, 6, 7, 8, 11, 17\}$, (c) $\{1, 2, 3, 4, 5, 8, 9, 10, 12, 13, 14, 15\}$, and (d) $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16\}$.

Edge	time unit 1	time unit 2	time unit 3	time unit 4
(8,1)	(1,2,4,8)	(1,2,3,8)	(1,3,8,17)	(1,4,8,17)
(8,2)	(1,2,3,8)	(1,2,4,8)	(2,3,8,15)	(2,4,8,15)
(8,3)	(3,8,15,17)	(2,3,8,15)	(1,2,3,8)	(1,3,8,17)
(8,4)	(2,4,8,15)	(4,8,15,17)	(1,4,8,17)	(1,2,4,8)
(8,15)	(2,3,8,15)	(2,4,8,15)	(4,8,15,17)	(3,8,15,17)
(8,17)	(1,3,8,17)	(1,4,8,17)	(3,8,15,17)	(4,8,15,17)

TABLE V. Shortest CNOT sequence to measure each Z stabilizer with a single ancilla for the 8-qubit spherical color code in Fig. 37(a). Sets of two and four vertices from v_{list} are used to label edges (measurement qubits) and tetrahedra (data qubits), which form the target and control of each CNOT which is applied in the specified time unit. Note that each qubit is involved in at most one gate per time unit.

R, B, B, R, G, G, Y, G, G, B, R, R, B, G, G, R. The central Y vertex is placed at the origin $(0, 0, 0)$ and is the eighth entry in v_{list}. To apply the CNOT schedules we specify in Table VI, one must map the neighborhood of every vertex to one of these four standard configurations. All interior vertices lie precisely on these standard coordinates relative to their central yellow vertex. Boundary vertices connected to their central yellow vertex will not be on the standard co-ordinates in v_{list}, but can be identified and shifted to the standard co-ordinate uniquely since there is at most one boundary vertex of each color included in the neighborhood of any yellow vertex.

Syndrome fixing.— In the absence of error, the syndrome readout of the Z-stabilizer measurements for each spherical 2D color code (which will have randomized outcomes) infers an X-type operator to 'fix' the outcomes to be $+1$. Since the spherical 2D color code encodes no logical qubits, any X-type operator with the correct syndrome will suffice, and can be found with Gaussian elimination. However, faults in the measurement circuits can cause the stabilizer outcomes to have no valid fixing operator. We consider a number of strategies to deal with this. Firstly, we consider the *repeat* strategy: if the observed syndrome is invalid, we repeat the preparation once. Secondly, we consider the *flip* strategy: if the observed syndrome is invalid, we flip a bit of the syndrome to produce a valid syndrome, and then apply a fixing operator. We test the impact of these approaches on the failure probability of code switching using a distance 9 code in Fig. 38. Note that when using the repeat strategy, if the first round is successful, the qubits are left idle in the prepared state to allow for a second round to be applied on spheres that failed the first round.

Edge	time unit 1	time unit 2	time unit 3	time unit 4	time unit 5	time unit 6
	2D color code on 12 qubit sphere					
(8,1)			(1,3,8,17)	(1,2,3,8)	(1,2,4,8)	(1,4,8,17)
(8,2)	(1,2,3,8)	(1,2,4,8)	(2,3,6,8)	(2,6,8,11)	(2,4,7,8)	(2,7,8,11)
(8,3)			(1,2,3,8)	(2,3,6,8)	(1,3,8,17)	(3,6,8,17)
(8,4)			(1,2,4,8)	(2,4,7,8)	(1,4,8,17)	(4,7,8,17)
(8,6)			(6,8,11,17)	(3,6,8,17)	(2,3,6,8)	(2,6,8,11)
(8,7)			(2,7,8,11)	(7,8,11,17)	(4,7,8,17)	(2,4,7,8)
(8,11)			(2,6,8,11)	(2,7,8,11)	(6,8,11,17)	(7,8,11,17)
(8,17)	(1,3,8,17)	(1,4,8,17)	(3,6,8,17)	(4,7,8,17)	(7,8,11,17)	(6,8,11,17)
	2D color code on 18 qubit sphere					
(8,1)			(1,2,3,8)	(1,2,4,8)	(1,3,5,8)	(1,4,5,8)
(8,2)			(1,2,4,8)	(1,2,3,8)	(2,3,8,15)	(2,4,8,15)
(8,3)	(1,2,3,8)	(1,3,5,8)	(3,5,8,9)	(2,3,8,15)	(3,8,9,12)	(3,8,12,15)
(8,4)	(1,2,4,8)	(1,4,5,8)	(4,5,8,10)	(2,4,8,15)	(4,8,10,13)	(4,8,13,15)
(8,5)	(1,3,5,8)	(3,5,8,9)	(1,4,5,8)	(4,5,8,10)	(5,8,9,14)	(5,8,10,14)
(8,9)			(5,8,9,14)	(3,8,9,12)	(3,5,8,9)	(8,9,12,14)
(8,10)			(4,8,10,13)	(5,8,10,14)	(4,5,8,10)	(8,10,13,14)
(8,12)			(3,8,9,12)	(3,8,12,15)	(8,9,12,14)	(8,12,14,15)
(8,13)			(8,13,14,15)	(4,8,13,15)	(8,10,13,14)	(4,8,10,13)
(8,14)	(5,8,9,14)	(5,8,10,14)	(8,9,12,14)	(8,10,13,14)	(8,12,14,15)	(8,13,14,15)
(8,15)	(2,4,8,15)	(3,8,12,15)	(4,8,13,15)	(8,12,14,15)	(8,13,14,15)	(2,3,8,15)
	2D color code on 24 qubit sphere					
(8,1)			(1,2,3,8)	(1,2,4,8)	(1,3,5,8)	(1,4,5,8)
(8,2)	(1,2,3,8)	(1,2,4,8)	(2,6,8,11)	(2,3,6,8)	(2,4,7,8)	(2,7,8,11)
(8,3)	(2,3,6,8)	(1,2,3,8)	(1,3,5,8)	(3,5,8,9)	(3,8,9,12)	(3,6,8,12)
(8,4)	(1,2,4,8)	(2,4,7,8)	(1,4,5,8)	(4,7,8,13)	(4,5,8,10)	(4,8,10,13)
(8,5)	(1,3,5,8)	(1,4,5,8)	(3,5,8,9)	(4,5,8,10)	(5,8,9,14)	(5,8,10,14)
(8,6)			(2,3,6,8)	(2,6,8,11)	(3,6,8,12)	(6,8,11,12)
(8,7)			(2,4,7,8)	(2,7,8,11)	(4,7,8,13)	(7,8,11,13)
(8,9)			(3,8,9,12)	(5,8,9,14)	(3,5,8,9)	(8,9,12,14)
(8,10)			(4,5,8,10)	(4,8,10,13)	(5,8,10,14)	(8,10,13,14)
(8,11)	(2,6,8,11)	(2,7,8,11)	(8,11,12,16)	(6,8,11,12)	(7,8,11,13)	(8,11,13,16)
(8,12)	(3,6,8,12)	(3,8,9,12)	(6,8,11,12)	(8,9,12,14)	(8,11,12,16)	(8,12,14,16)
(8,13)	(4,7,8,13)	(4,8,10,13)	(7,8,11,13)	(8,10,13,14)	(8,11,13,16)	(8,13,14,16)
(8,14)	(5,8,10,14)	(8,9,12,14)	(8,10,13,14)	(8,13,14,16)	(8,12,14,16)	(5,8,9,14)
(8,16)			(8,11,13,16)	(8,12,14,16)	(8,13,14,16)	(8,11,12,16)

TABLE VI. Shortest CNOT sequences to measure each Z stabilizer with a single ancilla for the 12-, 18- and 24-qubit spherical color codes in Fig. 37(b-d). Sets of two and four vertices from v_{list} are used to label edges (measurement qubits) and tetrahedra (data qubits), which form the target and control of each CNOT which is applied in the specified time unit. Note that weight-4 stabilizers have their CNOTs applied in the last 4 time units, avoiding unnecessary idle times in the measurement circuits.

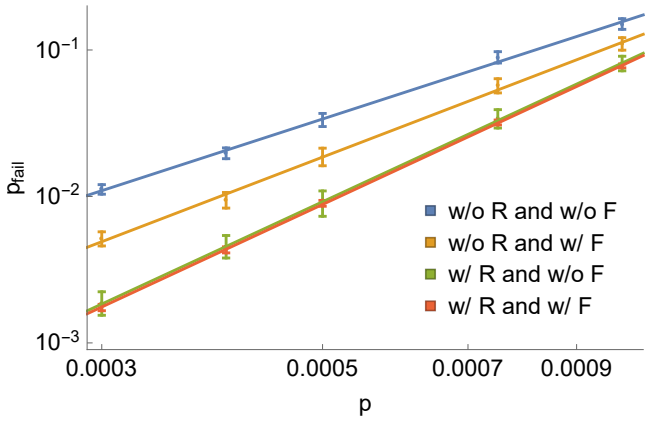


FIG. 38. Overall code switching performance of four syndrome fixing approaches for the preparation of 2D spherical color codes. Each approach either uses or does not use the repeat (R) and flip (F) strategies for distance $d = 9$. In the approach which uses both strategies, we use the repeat strategy, and the flip strategy if the repeat also yields an invalid syndrome. In each case, the stabilizers are measured with the CNOT circuits described in Table V and Table VI. Both the repeat and flip strategies improve performance.

In practice we find that these idle rounds do not have a lot of impact on the performance in the regime of interest. We see that both the repeat and flip strategies improve performance, and so we incorporate both into the optimized code switching protocol.

E. Choices of basis for code switching preparation

Here we analyze the effect of the choice of basis in first two steps of the code switching described in Section VI A 1 and Section VI A 2. The performance of the code may depend on this choice as it could lead to an asymmetry in the X and Z type noise and later steps of the protocol would not treat X and Z error in the same way.

The first choice is for the preparation of the Bell state in the 2D color code patches in Section VI A 1. We fault-tolerantly prepare the first patch in $|\overline{+}\rangle$, and the second in $|\overline{0}\rangle$ before applying a transverse CNOT from the first to the second. Since the CNOT copies X noise from the first to the second patch, and Z from the second to the first, we can expect that the first patch has more Z noise and the second patch has more X after a faulty preparation. We choose whether to feed the first or the second patch to be involved in code switching, which we refer to as *patch* $|+\rangle$ and *patch* $|0\rangle$ respectively.

The second choice is for the preparation of the 2D spherical color code in its unique code state around each interior yellow vertex as described in Section VI A 2. We can do this by either preparing each data quibt in $|0\rangle$ and then measuring the X stabilizers, or by preparing each data quibt in $|+\rangle$ and then measuring the Z stabilizes. We refer to these cases as *sphere* $|0\rangle$ and *sphere* $|+\rangle$ respectively, and compare their impact along with the choice for the patch preparation in Fig. 39. We find that somewhat surprisingly, there is very little difference in the performance of each of these choices, and we select *patch* $|+\rangle$ and *sphere* $|+\rangle$ for use in the optimized version of the protocol.

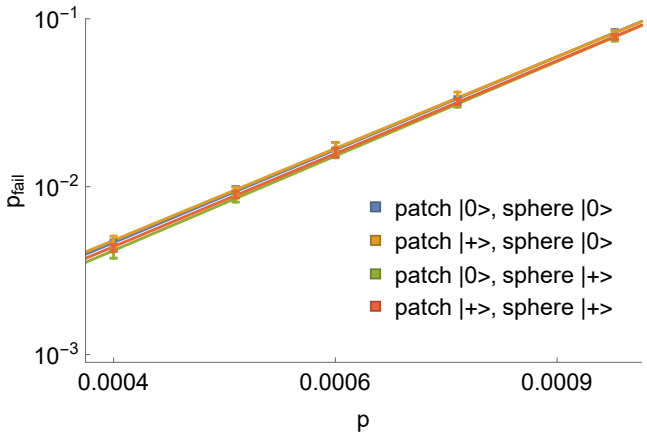


FIG. 39. The impact on the failure probability of the code switching protocol from different choices of basis for first two steps of the protocol. The data is for distance $d = 9$, and has all other steps as described in Section VI for the optimized code switching protocol.

F. Gauge measurement circuits for code switching

Here we discuss the circuits used to measure the Z-type gauge corresponding to RG, RB and GB edges used in Section VI A 3 for code switching. We find a minimum-length circuit with a single ancilla, which needs 8 time units (including preparation and measurement). A more naive circuit of 18 time units could be constructed by measuring edges of the same color in parallel, since no two edges of the same color share a qubit.

To specify the CNOT sequences, we separate edges (measurement qubits) into 20 types by color and orientation and identify the tetrahedra containing them (data qubits) in a systematic way in terms of their spatial position relative to the edge. The CNOT sequence is then specified for each edge type. To specify the neighborhood of each edge we label every qubit in its support with some number from 1 to 6 in such a way that all edges of the same type have qubits labelled locally, and are all consistent. There are some technical issues that arise due to the boundary, see Fig. 40. Firstly, some edges may contain a boundary vertex—such edges form their own type labeled by the boundary vertex and the color of the interior vertex in the edge. Secondly, some vertices in the tetrahedra around an edge may be boundary vertices, causing the precise shape of the neighborhood of the edge to deviate from the typical shape. Thirdly, some edge types which typically have 6 qubits in their support can have a reduced support since they are essentially 'cut' by the boundary.

To label tetrahedra around an edge with two interior vertices, we specify a local coordinate system. The z basis vector is the normalized vector v_{edge} from the vertex of lower to higher color in the ordering R, G, B, Y. The x basis vector is the normalized vector parallel to $v_{\text{irr}} = (1, \pi, \sqrt{2})$, but perpendicular to v_{edge}, which is guaranteed not to be parallel to any edges. The y basis vector is then specified uniquely to form a right hand coordinate system (x, y, z). The tetrahedra around the edge are then labelled in sequence according to the azimuthal angle to their mid-point in this co-ordinate system. For edges which contain a boundary vertex, the same applies, but the vector v_{edge} is specified according to Table VII. In the bulk, the tetrahedra are positioned at precise standard azimuthal angles for each edge type, but at the boundary we identify the tetrahedra by whichever standard angle it is closest to. When tetrahedra are 'missing' around an edge due to cuts along the boundary, they are simply skipped in the CNOT sequence, with no CNOTs applied during the assigned time unit for that edge.

The sequence in Table VII was found using a greedy algorithm starting with a random initial

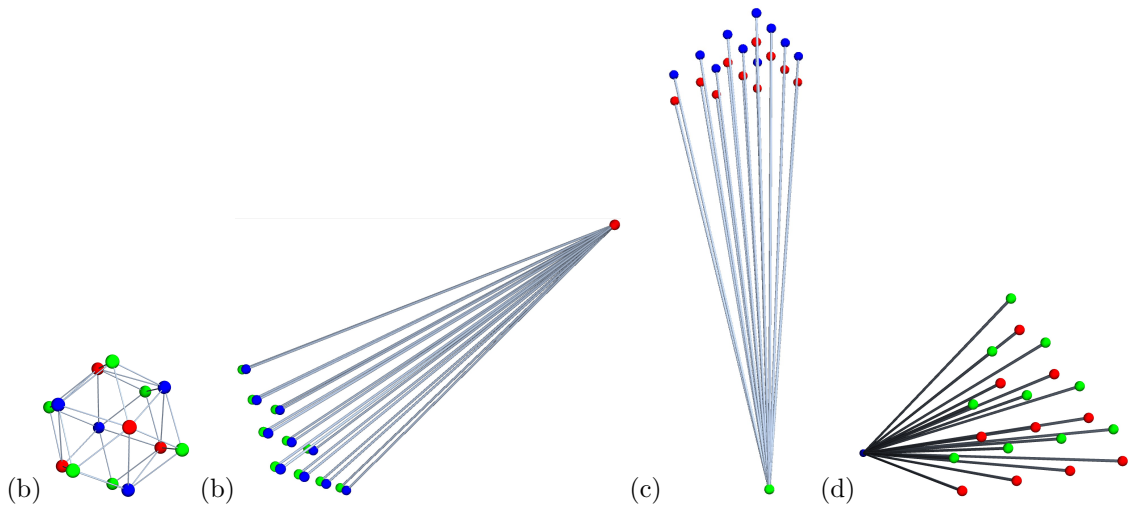


FIG. 40. The Z-type gauge operators that are measured in the protocol to implement code switching are associated with RG, RB and GB edges in the lattice, illustrated for the distance $d = 9$ lattice. These are parts of the full lattice in Fig. 4(a). We separate the edges into 20 cases. (a) Edges connecting interior vertices are classified by color and the vector from one vertex to the other. (b)-(d) Other edges are classified by the boundary vertex and the color of the interior vertex they are incident to.

sequence and minimizing the cost function of the number of unsatisfied constraints (we require that no qubit is involved in more than one gate per time step) minus the average step time for each gate. The result is the shortest possible measurement circuit using a single measurement qubit per edge, and which involves no idle time units for edges types of weight four.

To simulate the gauge measurements, we first apply an X type gauge operator g_X supported on randomly selected RB, GB and RG edges. Then the specified circuits are applied and produce the outcomes $\tilde{\gamma}$, which in the absence of error would correspond to precisely the Z-edges which anti-commute with g_X.

G. Potential improvements on the 3D color code decoder

Here we justify of our estimate of the potential impact on code switching performance due to improvements on the 3D color code decoder. For any error E we define its *stabilizer-reduced weight* to be the minimum of the weight of sE for any stabilizer s. The estimate rests on the following assumptions.

1. We assume that the stabilizer-reduced weight of errors generated by code switching can be approximated by the minimum of the weight of either the error itself or it's correction produced by the modified restriction decoder.

2. The decoder does not take into account any correlations in the noise produced during the various steps of code switching and its failure probability for errors with stabilizer-reduced weight w produced by code switching is no lower than that of the optimal decoder for iid Z errors of weight w.

3. The optimal decoder for iid Z noise, when applied to uniformly drawn weight-w errors has

vertex types	edge vector	time unit 1	time unit 2	time unit 3	time unit 4	time unit 5	time unit 6
R-G	$(+\frac{1}{2},-\frac{1}{2},-\frac{1}{2})$	1	3	2	4	5	6
R-G	$(+\frac{1}{2},+\frac{1}{2},+\frac{1}{2})$	1	2	3	4	5	6
R-G	$(-\frac{1}{2},+\frac{1}{2},-\frac{1}{2})$	1	2	3	6	5	4
R-G	$(-\frac{1}{2},-\frac{1}{2},+\frac{1}{2})$	1	2	4	3	5	6
G-B	$(+\frac{1}{2},-\frac{1}{2},-\frac{1}{2})$	1	2	3	6	4	5
G-B	$(+\frac{1}{2},+\frac{1}{2},+\frac{1}{2})$	1	2	3	6	5	4
G-B	$(-\frac{1}{2},+\frac{1}{2},-\frac{1}{2})$	3	1	2	4	5	6
G-B	$(-\frac{1}{2},-\frac{1}{2},+\frac{1}{2})$	1	2	3	6	5	4
R-B	$(+0,+0,-1)$			1	2	3	4
R-B	$(+0,+1,+0)$			1	2	3	4
R-B	$(+1,+0,+0)$			1	2	3	4
R-B	$(-1,+0,+0)$			1	2	3	4
R-B	$(+0,-1,+0)$			1	2	4	3
R-B	$(+0,+0,+1)$			1	2	3	4
v_R-G	$(-1,-1,-1)$	1	2	3	4	5	6
R-v_G	$(+1,-1,-1)$	1	2	3	4	5	6
v_G-B	$(-1,+1,+1)$	1	2	3	5	6	4
G-v_B	$(-1,+1,-1)$	2	1	3	4	5	6
v_R-B	$(-1,-1,-1)$	1	2	5	3	4	6
R-v_B	$(-1,+1,-1)$	1	2	3	4	6	5

TABLE VII. The measurement circuit specification for RG, RB and GB gauge operators. Edges connecting interior vertices (first 14 rows) are labelled by the vertex colors (first column) and the vector connecting them (second column). Edges connecting a boundary vertex are labelled by that boundary vertex v_R, v_G or v_B and the color of the interior vertex. Qubits around an edge are labelled geometrically in order of increasing azimuthal angle, and the column in which each data qubit appears specifies which time unit a CNOT occurs from that data qubit to the corresponding mesurement qubit for the edge.

failure probability satisfying

$$p_{\text{fail}}^{\text{iid}}(w,d) > p_{\text{3DCC}}^{(1)} \left(\frac{w/n}{p_{\text{3DCC}}^{(1)}} \right)^{\frac{d+1}{2}}, \tag{43}$$

where n is the number of data qubits, and $p_{\text{3DCC}}^{(1)} \simeq 0.019$ is the known optimal threshold of the the 3D stabilizer color code decoder for iid Z noise [92].

It is clearly important to account for stabilizers contributing to the error, since a single fault in stabilizer measurement circuits often results in a large fraction of a full stabilizer being included in the error. These seemingly high-weight errors are equivalent (up to the stabilizer whose support they almost contain) to low-weight errors, and are therefore relatively harmless. This motivates us to use the stabilizer-reduced weight rather than the actual weight of an error as a proxy for how hard it is to correct. The first assumption is then made to avoid needing to multiply the error by an exponential number of stabilizers to find its minimum-weight representative. This assumption is somewhat justified by the fact that the modified restriction decoder in Section V C seeks to output a low-weight correction.

The second assumption is somewhat crude, since decoders which take correlations into account can outperform those which do not [**?**]. However we suspect it is close to the truth, since

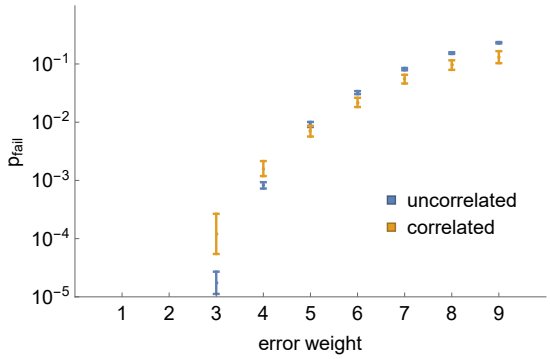


FIG. 41. Performance of the modified restriction decoder for the 3D color code for distance $d = 9$ given various error weights. Uncorrelated Z noise (blue) of weight w is generated by randomly selecting w of the n qubits and applying a Z error. Correlated Z noise (yellow) is generated by running the decoder for error rate $p = 0.0003$, and for each error generated, estimating its stabilizer-reduced weight w by taking the smaller of the weight of the error itself and its correction. The correlated and uncorrelated cases are not identical but are very similar, justifying our assumptions.

in this case the decoder will also have to correct uncorrelated weight w errors in addition to correlated errors, therefore we believe it is unlikely that one can design a decoder which significantly outperforms the optimal decoder for iid Z noise in this setting. We build further confidence in the first two assumptions numerically in Fig. 41. There, we verify that the performance of the modified restriction decoder on noise produced by the code switching protocol with stabilizer-reduced weight w (estimated using the first assumption) is comparable to it's performance on iid Z noise with weight w.

The third assumption is based on the heuristic behavior of error correction failure rate p_{fail} in topological codes for error rate p in the vicinity of their threshold p^* [35, 88, 98], i.e. that $p_{\text{fail}}(p, d) \sim (p/p^*)^{(d+1)/2}$. To form an inequality in terms of the error weight w, we first take w as a proxy for the typical typical error $\langle w \rangle$, which for iid Z noise on n qubits satisfies $\langle w \rangle = pn$. We then note that taking p^* as a prefactor would correspond to the pseudo-threshold being independent of system size, thereby overestimating error rate for most topological codes.

[1] D. Nigg, M. Müller, E. A. Martinez, P. Schindler, M. Hennrich, T. Monz, M. A. Martin-Delgado, and R. Blatt, Science **345**, 302 (2014).

[2] R. Barends, J. Kelly, A. Megrant, A. Veitia, D. Sank, E. Jeffrey, T. C. White, J. Mutus, A. G. Fowler, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, C. Neill, P. O'Malley, P. Roushan, A. Vainsencher, J. Wenner, A. N. Korotkov, A. N. Cleland, and J. M. Martinis, Nature **508**, 500 (2014).

[3] A. D. Córcoles, J. M. Gambetta, J. M. Chow, J. A. Smolin, M. Ware, J. Strand, B. L. T. Plourde, and M. Steffen, Physical Review A **87**, 030301 (2013).

[4] N. Ofek, A. Petrenko, R. Heeres, P. Reinhold, Z. Leghtas, B. Vlastakis, Y. Liu, L. Frunzio, S. M. Girvin, L. Jiang, M. Mirrahimi, M. H. Devoret, and R. J. Schoelkopf, Nature **536**, 441 (2016).

[5] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis, Nature **574**, 505 (2019).

[6] J. Preskill, Quantum **2**, 79 (2018).

[7] P. Shor, in *Proceedings of 37th Conference on Foundations of Computer Science* (IEEE Comput. Soc. Press, 1996) pp. 56–65.

[8] E. Knill, Phys. Rev. A **71**, 042322 (2005).

[9] A. M. Steane, Phys. Rev. Lett. **78**, 2252 (1997).

[10] D. Aharonov and M. Ben-Or, in *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing* (ACM, 1997) pp. 176–188.

[11] J. Preskill, Proc. Roy. Soc. Lond. **454**, 385 (1998).

[12] P. Aliferis, D. Gottesman, and J. Preskill, Quantum Information and Computation **6**, 097 (2005).

[13] R. Raussendorf and J. Harrington, Physical Review Letters **98** (2007), 10.1103/PhysRevLett.98.190504.

[14] R. Raussendorf, J. Harrington, and K. Goyal, New Journal of Physics **9** (2007), 10.1088/1367-2630/9/6/199.

[15] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, J. Math. Phys. **43**, 4452 (2002).

[16] A. W. Cross, D. P. Divincenzo, and B. M. Terhal, Quantum Info. Comput. **9**, 541572 (2009).

[17] B. J. Brown, D. Loss, J. K. Pachos, C. N. Self, and J. R. Wootton, Reviews of Modern Physics **88**, 045005 (2016).

[18] G. Duclos-Cianci and D. Poulin, Physical Review Letters **104**, 050504 (2010).

[19] H. Anwar, B. J. Brown, E. T. Campbell, and D. E. Browne, New Journal of Physics **16**, 063038 (2014).

[20] G. Duclos-Cianci and D. Poulin, Physical Review A **87**, 062338 (2013).

[21] S. Bravyi and J. Haah, Physical Review Letters **111**, 200501 (2013).

[22] K. Duivenvoorden, N. P. Breuckmann, and B. M. Terhal, IEEE Transactions on Information Theory **65**, 2545 (2019).

[23] A. Kubica and J. Preskill, Physical Review Letters **123** (2019), 10.1103/PhysRevLett.123.020501.

[24] M. Vasmer, D. E. Browne, and A. Kubica, arXiv:2004.07247 (2020).

[25] S. Bravyi, M. Suchara, and A. Vargo, Physical Review A - Atomic, Molecular, and Optical Physics **90** (2014), 10.1103/PhysRevA.90.032326.

[26] A. S. Darmawan and D. Poulin, Physical Review E **97** (2018), 10.1103/PhysRevE.97.051302.

[27] N. H. Nickerson and B. J. Brown, Quantum **3**, 131 (2019).

[28] N. Maskara, A. Kubica, and T. Jochym-O'Connor, Physical Review A **99** (2019), 10.1103/PhysRevA.99.052351.

[29] C. Chamberland and P. Ronagh, Quantum Science and Technology **3** (2018), 10.1088/2058-9565/aad1f7.

[30] N. Delfosse, Phys. Rev. A **89**, 012317 (2014).

[31] A. Kubica and N. Delfosse, arXiv (2019), arXiv:1905.07393.
[32] A. Y. Kitaev, Russ. Math. Surv. **52**, 1191 (1997).
[33] S. Bravyi and A. Y. Kitaev, (1998), arXiv:quant-ph/9811052.
[34] H. Bombin and M. Martin-Delgado, Physical Review Letters **97**, 180501 (2006).
[35] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Physical Review A **86**, 032324 (2012).
[36] C. Chamberland, G. Zhu, T. J. Yoder, J. B. Hertzberg, and A. W. Cross, Physical Review X **10** (2020), 10.1103/PhysRevX.10.011022.
[37] C. Chamberland, A. Kubica, T. J. Yoder, and G. Zhu, New Journal of Physics **22** (2020), 10.1088/1367-2630/ab68fd.
[38] T. Karzig, C. Knapp, R. M. Lutchyn, P. Bonderson, M. B. Hastings, C. Nayak, J. Alicea, K. Flensberg, S. Plugge, Y. Oreg, C. M. Marcus, and M. H. Freedman, Phys. Rev. B **95**, 235305 (2017).
[39] R. Chao, M. E. Beverland, N. Delfosse, and J. Haah, Quantum **4**, 352 (2020).
[40] B. Eastin and E. Knill, Phys. Rev. Lett. **102**, 110502 (2009).
[41] B. Zeng, A. Cross, and I. L. Chuang, IEEE Transactions on Information Theory **57**, 6272 (2011).
[42] T. Jochym-O'Connor, A. Kubica, and T. J. Yoder, Physical Review X **8**, 21047 (2018).
[43] S. Bravyi and R. König, Phys. Rev. Lett. **110**, 170503 (2013).
[44] F. Pastawski and B. Yoshida, "Fault-tolerant logical gates in quantum error-correcting codes," (2014), arXiv:1408.1720.
[45] M. E. Beverland, O. Buerschaper, R. Koenig, F. Pastawski, J. Preskill, and S. Sijher, Journal of Mathematical Physics **57**, 022201 (2016).
[46] D. Gottesman, (1998), arXiv:quant-ph/9807006.
[47] H. Bombin and M. Martin-Delgado, Physical Review B **75**, 075103 (2007).
[48] A. Kubica, B. Yoshida, and F. Pastawski, New Journal of Physics **17**, 083026 (2015).
[49] M. Vasmer and D. E. Browne, Physical Review A **100**, 012312 (2019).
[50] S. Bravyi and A. Kitaev, Phys. Rev. A **71**, 022316 (2005).
[51] E. Knill, arXiv preprint arXiv:0404104 (2004).
[52] E. Knill, arXiv preprint arXiv:0402171 (2004).
[53] D. Litinski, Quantum **3**, 205 (2019).
[54] A. Paetznick and B. W. Reichardt, Phys. Rev. Lett. **111**, 090505 (2013).
[55] J. T. Anderson, G. Duclos-Cianci, and D. Poulin, Phys. Rev. Lett. **113**, 080501 (2014).
[56] H. Bombin, New Journal of Physics **17**, 083002 (2015).
[57] H. Bombín, New Journal of Physics **18**, 043038 (2016).
[58] H. Bombin, Physical Review X **5**, 031043 (2015).
[59] P. Brooks, Caltech Ph.D Thesis (2013).
[60] T. Jochym-O'Connor, Y. Yu, B. Helou, and R. Laflamme, Quantum Information and Computation **13**, 0361 (2013).
[61] N. C. Jones, "Logic synthesis for fault-tolerant quantum computers," (2013), PhD Thesis (Stanford):1310.7290 [quant-ph].
[62] S. Bravyi and A. Cross, arXiv:1509.03239 (2015).
[63] T. Jochym-O'Connor and S. D. Bartlett, Physical Review A **93** (2016), 10.1103/PhysRevA.93.022323.
[64] C. Jones, P. Brooks, and J. Harrington, Physical Review A **93** (2016), 10.1103/PhysRevA.93.052332.
[65] H. Bombin, arXiv:1810.09571 (2018).
[66] B. J. Brown, Science Advances **6** (2020), 10.1126/sciadv.aay4929.
[67] J. Iverson and A. Kubica, in preparation (2020).
[68] C. D. Hill, A. G. Fowler, D. S. Wang, and L. C. L. Hollenberg, Quantum Info. Comput. **13**, 439451 (2013).
[69] T. J. Yoder, R. Takagi, and I. L. Chuang, Physical Review X **6**, 031039 (2016).
[70] T. Jochym-O'Connor and R. Laflamme, Physical Review Letters **112**, 010505 (2014).
[71] C. Chamberland and A. W. Cross, Quantum **3**, 143 (2019).
[72] C. Chamberland and K. Noh, npj Quantum Information **6**, 91 (2020).
[73] A. Kubica and M. Beverland, Physical Review A **91**, 032330 (2015).
[74] A. Kubica, *The ABCs of the color code: A study of topological quantum codes as toy models for fault-tolerant quantum computation and quantum phases of matter*, Ph.D. thesis (2018).
[75] A. J. Landahl and C. Ryan-Anderson, "Quantum computing by color-code lattice surgery," (2014), arXiv:1407.5103 [quant-ph].

[76] J. Haah and M. B. Hastings, Quantum **2**, 71 (2018).
[77] A. J. Landahl and C. Cesare, arXiv (2013), arXiv:1302.3240.
[78] S. Bravyi and J. Haah, Physical Review A **86**, 052329 (2012).
[79] M. B. Hastings and J. Haah, Phys. Rev. Lett. **120**, 050504 (2018).
[80] B. W. Reichardt, Quantum Information Processing **4**, 251 (2005).
[81] A. M. Meier, B. Eastin, and E. Knill, Quantum Information and Computation **13**, 0195 (2013).
[82] C. Jones, Physical Review A **87**, 042305 (2013).
[83] G. Duclos-Cianci and D. Poulin, Physical Review A **91**, 042315 (2015).
[84] E. T. Campbell and J. OGorman, Quantum Science and Technology **1**, 015007 (2016).
[85] J. Haah, M. B. Hastings, D. Poulin, and D. Wecker, arXiv preprint arXiv:1709.02789 (2017).
[86] J. Haah, Phys. Rev. A **97**, 042327 (2018).
[87] D. S. Wang, A. G. Fowler, and L. C. L. Hollenberg, Phys. Rev. A **83**, 020302 (2011).
[88] A. J. Landahl, J. T. Anderson, and P. R. Rice, arXiv preprint arXiv:1108.5738 (2011).
[89] A. M. Stephens, arXiv preprint arXiv:1402.3037 (2014).
[90] B. J. Brown, N. H. Nickerson, and D. E. Browne, Nature Communications **7**, 4 (2015), arXiv:1503.08217.
[91] B. M. Terhal, Rev. Mod. Phys. **87**, 307 (2015).
[92] A. Kubica, M. E. Beverland, F. Brandão, J. Preskill, and K. M. Svore, Phys. Rev. Lett. **120**, 180501 (2018).
[93] A. Fowler, private communication (2020).
[94] Y. Li, New Journal of Physics **17**, 023037 (2015).
[95] C. Chamberland and K. Noh, (2020), arXiv:2003.03049.
[96] H. Bombin, arXiv (2018), arXiv:1810.09575.
[97] S. Turner, J. Hanish, E. Blanchard, N. Davis, and B. L. Cour, arXiv (2020), arXiv:2003.11602.
[98] A. G. Fowler, Phys. Rev. A **87**, 040301 (2013).