Cost of Universality: A Comparative Study of the Overhead of State Distillation and Code Switching with Color Codes

Michael E. Beverland[1,*] Aleksander Kubica,[2,3,4,5] and Krysta M. Svore[1]

[1] *Microsoft Quantum and Microsoft Research, Redmond, Washington 98052, USA*
[2] *Perimeter Institute for Theoretical Physics, Waterloo, Ontario N2L 2Y5, Canada*
[3] *Institute for Quantum Computing, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada*
[4] *AWS Center for Quantum Computing, Pasadena, California 91125, USA*
[5] *California Institute of Technology, Pasadena, California 91125, USA*

(Received 2 January 2021; accepted 21 April 2021; published 15 June 2021)

Estimating and reducing the overhead of fault-tolerance (FT) schemes is a crucial step toward realizing scalable quantum computers. Of particular interest are schemes based on two-dimensional (2D) topological codes such as the surface and color codes that have high thresholds but lack a natural implementation of a non-Clifford gate. In this work, we directly compare two leading FT implementations of the T gate in 2D color codes under circuit noise across a wide range of parameters in regimes of practical interest. We report that implementing the T gate via code switching to a three-dimensional (3D) color code does not offer substantial savings over state distillation in terms of either space or space-time overhead. We find a circuit-noise threshold of 0.07(1)% for the T gate via code switching, almost an order of magnitude below that achievable by state distillation in the same setting. To arrive at these results, we provide and simulate an optimized code-switching procedure, and bound the effect of various conceivable improvements. Many intermediate results in our analysis may be of independent interest. For example, we optimize the 2D color code for circuit noise yielding its largest threshold to date 0.37(1)%, and adapt and optimize the restriction decoder finding a threshold of 0.80(5)% for the 3D color code with perfect measurements under Z noise. Our work provides a much-needed direct comparison of the overhead of state distillation and code switching, and sheds light on the choice of future FT schemes and hardware designs.

DOI: 10.1103/PRXQuantum.2.020341

Recent progress in demonstrating operational quantum devices [1–5] has brought the noisy intermediate-scale quantum era [6], where low-depth algorithms are run on small numbers of qubits. However, to handle the cumulative effects of noise and faults as these quantum 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 implement logical operations on the encoded information, all while continually diagnosing and repairing faults. This requires additional resources, and much of the current research in quantum error correction (QEC) is dedicated toward developing FT schemes with low overhead.

The choice of FT scheme to realize universal quantum computing has important ramifications. Good schemes can significantly enhance the functionality and lifetime of a given quantum computer. Moreover, FT schemes vary in their sensitivity to the hardware architecture and design, such as qubit quality [16], connectivity, and operation speed. The understanding and choice of FT scheme will therefore influence the system design, from hardware to software, and developing an early understanding of the trade-offs is critical in our path to a scalable quantum computer.

At the base of most FT schemes is a QEC code, which (given the capabilities and limitations of a particular hardware platform) 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 that are known to achieve as many of these points as possible 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.

[*] mbev50@gmail.com

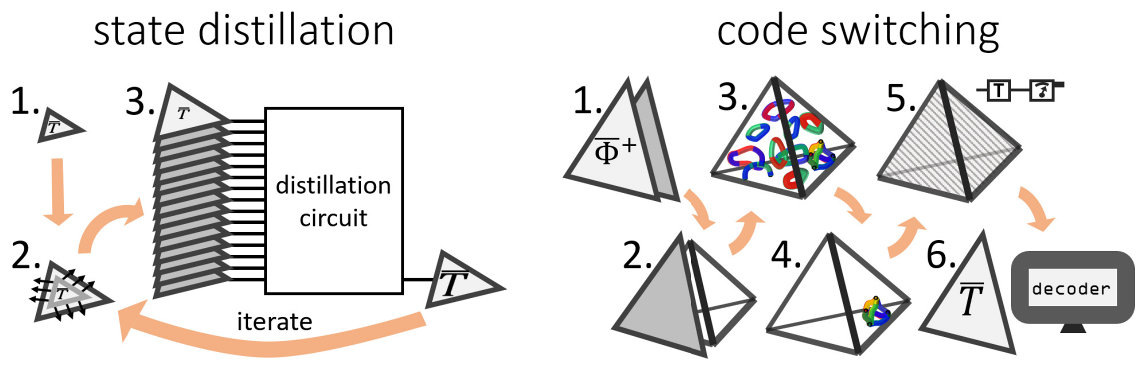


FIG. 1. Two methods of preparing high-fidelity T states encoded in the 2D color code: state distillation and code switching. Both approaches are implemented using quantum-local operations in 3D, i.e., noisy quantum operations are geometrically local, whereas ideal classical operations can be performed globally. In state distillation, many noisy encoded T states are produced and fed into a Clifford distillation circuit. In code switching, one switches to the 3D color code, where the transversal T gate is implemented.

[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 neighboring 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. These gates are automatically FT since they do not grow the support of errors. Unfortunately a QEC code that 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 [43–46]. There are, however, many 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 *state distillation* [50–52]. It relies on first producing many noisy encoded copies of a T *state* $|\overline{T}\rangle = (|\overline{0}\rangle + e^{i\pi/4}|\overline{1}\rangle)/\sqrt{2}$, also known as a *magic state*, and then processing them using Clifford operations to output a high-fidelity encoded version of the state. The high-fidelity T state can then be used to implement the non-Clifford $T = \mathrm{diag}(1, e^{i\pi/4})$ gate. Despite significant recent improvements, the overhead of state distillation is expected to be large in practice [35,53,54]. A compelling alternative is *code switching* via gauge fixing [55–58] to a three-dimensional (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 consideration in a hardware design, a detailed study is required.

In our work, we estimate the resources needed to prepare high-fidelity T states encoded in the 2D color code, via either state distillation or code switching. We assume that both approaches are implemented using quantum-local operations [59] in three dimensions, i.e., quantum operations are noisy and geometrically local, whereas classical operations can be performed globally and perfectly (although they must be computationally efficient). In particular, we simulate 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 state distillation, this 3D setting allows a stack of 2D color-code patches, whereas for code switching it allows implementation of 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 state distillation or code switching?*

Our main finding is that code switching does not offer substantial savings over state distillation in terms of both *space overhead*, i.e., the number of physical qubits required, and *space-time overhead*, i.e., the space overhead multiplied by the number of physical time units required; see Fig. 2. State distillation significantly outperforms code switching over most of the circuit-noise error rates $10^{-4} \le p \le 10^{-3}$ and target T state infidelities $10^{-20} \le p_{\mathrm{fin}} \le 10^{-4}$, except for the smallest values of p, where code switching slightly outperforms state distillation. In our analysis 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. On the other hand, we consider only a standard state-distillation scheme, and using more optimized schemes such as Refs. [60–62] would give further advantage to a state-distillation approach. We also find asymptotic expressions that support our finding that state distillation requires lower overhead than code switching

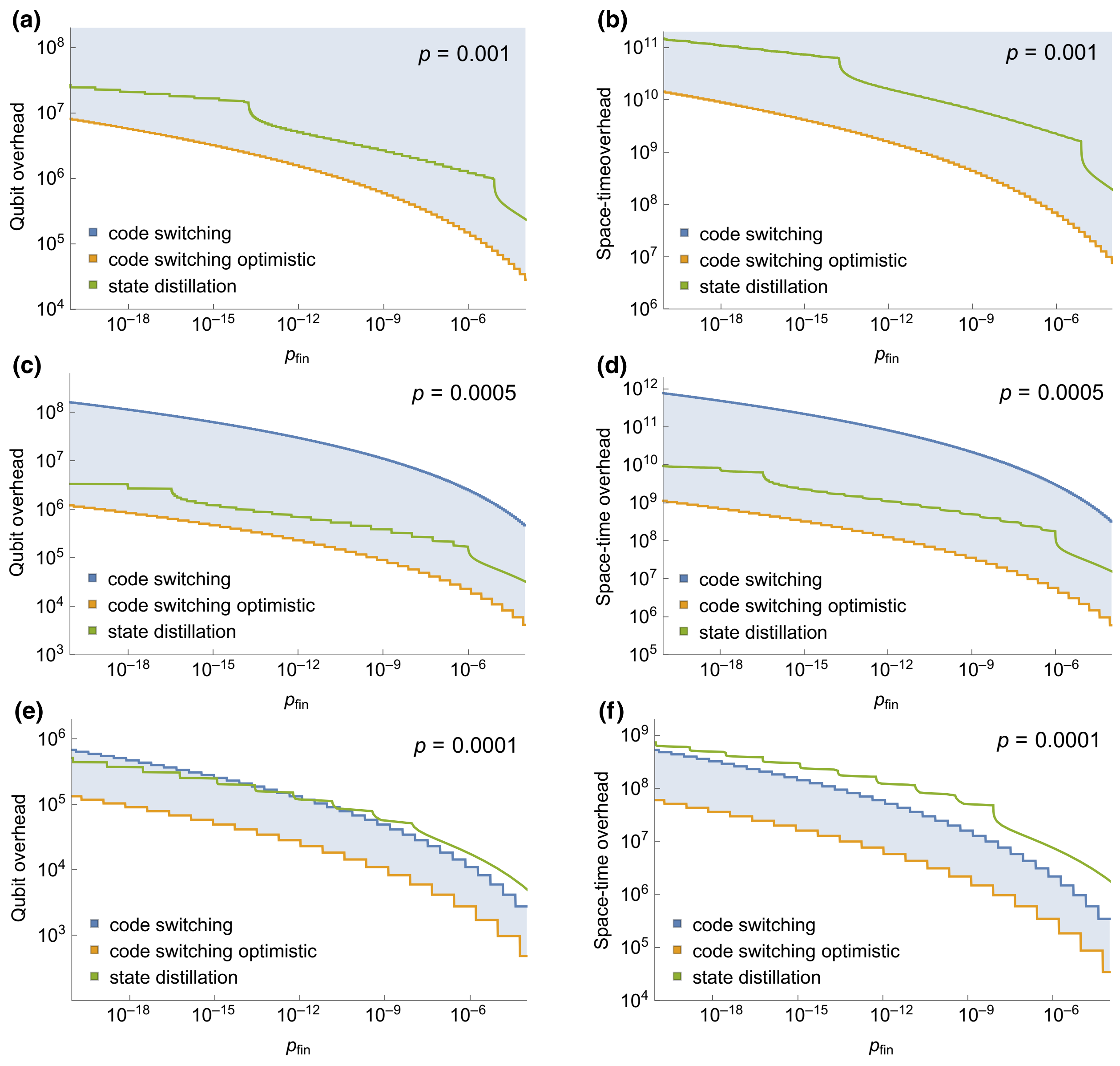


FIG. 2. Comparison of (a),(c),(e) the qubit and (b),(d),(f) space-time overhead as a function of the infidelity p_{fin} of the output T state for state distillation and code switching. Possible future improvements of any steps of our code-switching protocol would be included within the shaded region. Note there is no code-switching curve for $p = 0.001$ without assuming optimistic improvements to the protocol as this is higher than the observed threshold for code switching.

for $p \ll 1$ and $\log p_{\text{fin}}/\log p \gg 1$. In particular, the space and space-time overhead scale as $(\log p_{\text{fin}}/\log p)^{\Gamma_*}$ and $(\log p_{\text{fin}}/\log p)^{\Gamma_*+1}$, respectively, where $\Gamma_{\text{CS}} = 3$ for code switching and $\Gamma_{\text{SD}} = \log_3 15 = 2.46\ldots$ for the distillation scheme we implement.

To arrive at our main simulation results, we accomplish the intermediate goals below.

2D color-code optimization and analysis.—In Sec. II, 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 0.37(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 state-distillation analysis.—Using the effective logical noise model, we carefully analyse the overhead of state distillation in Sec. III. We strengthen the bounds on failure and rejection rate by explicitly calculating the effect of faults at each location in the Clifford state-distillation circuits rather than simply counting the total number of locations [35,53,63–65]. We remark that we stack 2D color codes in the third dimension to implement logical operations such as the controlled NOT (CNOT) gate in constant time, whereas strictly 2D approaches such as lattice surgery would require a time proportional to code distance. The circuit-noise threshold for this state-distillation scheme with the 2D color code is equal to the error-correction threshold of 0.37(1)%.

Further insights into 3D color codes.—In Sec. IV, we provide a surprisingly direct way to switch between the 2D color code and the 3D subsystem color code. Our method exploits a particular gauge fixing of the 3D subsystem color code for which 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 0.80(5)% in the setting of perfect measurements and a better performance for small system sizes.

End-to-end code-switching simulation.—Sec. V is the culmination of our work, where building upon results from the previous sections we provide a simplified recipe for code switching, detailing each step, and specifying important optimizations. 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(1)%. 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.

This work, which builds upon early approximate estimation work in Refs. [66] and [67], provides a much-needed comparative study of the overhead of state distillation and code switching, and enables a deeper understanding of these two approaches to FT universal quantum computation. More generally, 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 expect our main finding, i.e., that code switching does not significantly outperform state distillation, to hold for other topological codes such as the toric code as considered in Ref. [49]. Furthermore, our intuition suggests that state distillation will not be outperformed by code switching exploiting either 2D subsystem codes [68–70] or emulation of a 3D system with a dynamical 2D system [71–74] since these schemes are even more constrained than when 3D quantum-local operations are allowed. We remark that there are other known FT techniques for implementing a universal gate set [12,75–79], 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.

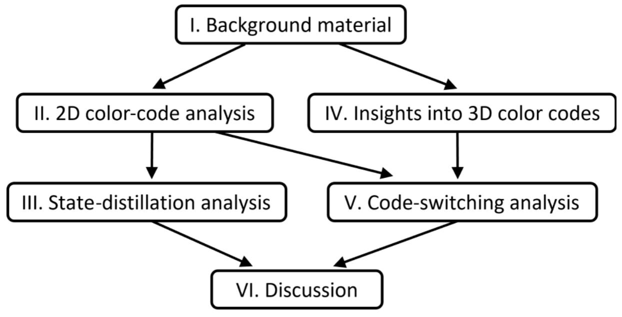

$$\begin{tikzcd}
 & \boxed{\text{I. Background material}} \arrow[dl] \arrow[dr] & \\
\boxed{\text{II. 2D color-code analysis}} \arrow[d] \arrow[drr] & & \boxed{\text{IV. Insights into 3D color codes}} \arrow[d] \\
\boxed{\text{III. State-distillation analysis}} \arrow[dr] & & \boxed{\text{V. Code-switching analysis}} \arrow[dl] \\
 & \boxed{\text{VI. Discussion}} &
\end{tikzcd}$$

FIG. 3. Flow diagram illustrating the organization and dependencies between different sections of the paper.

The structure of the paper is summarized in Fig. 3.

I. BACKGROUND MATERIAL

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

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

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

where the parameter p can be interpreted as an error probability. The depolarizing channel leaves a single-qubit state unaffected with probability $1-p$ and applies an error X, Y, or Z, each with probability $p/3$. Similarly, the depolarizing

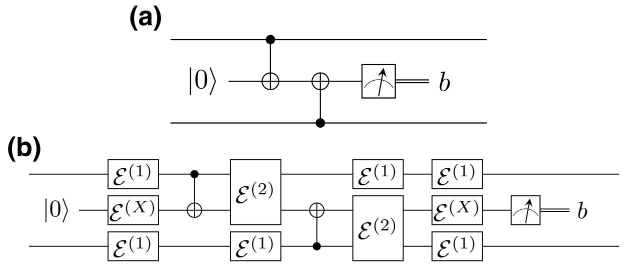


FIG. 4. (a) Example of an ideal circuit to measure weight-two operator ZZ. We assume that every gate, as well as preparation and single-qubit measurements take one time unit. (b) The noisy circuit is modeled with ideal gates followed by the depolarizing channel of strength p; ideal preparation can fail 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.

channel leaves a two-qubit state unaffected with probability $1-p$ and with probability $p/15$ applies a nontrivial Pauli error $P_1 \otimes P_2$.

We consider three standard scenarios.

1. *Depolarizing noise.*—Error correction is implemented with perfect measurements following a single 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; see Fig. 4(a). One- and two-qubit unitary operations experience depolarizing noise of strength p. One-qubit preparations and measurements fail with probability p by producing an orthogonal state or flipping the outcome; see Fig. 4(b).

In circuit noise, we approximate every noisy gate, i.e., Pauli X, Y, and Z operators, the Hadamard gate H, the phase gate T, the CNOT gate, and the idle gate I, by an ideal gate followed by the depolarizing channel on qubits acted on by the gate; see Fig. 4. 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 use *error rate* and *noise strength* interchangeably to describe the single parameter p.

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

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

written in the orthonormal basis $\{|T\rangle, |T^{\perp}\rangle = Z|T\rangle\}$. Now consider a "twirling operation" consisting of randomly applying the Clifford $XS^{\dagger} \propto |T\rangle\langle T| - |T^{\perp}\rangle\langle T^{\perp}|$, with probability $1/2$. This single-qubit Clifford gate can be implemented instantaneously and perfectly by a frame update [80]. The state is transformed as follows:

$$\rho \mapsto \frac{1}{2}\rho + \frac{1}{2}(XS^{\dagger})\rho(XS^{\dagger})^{\dagger} = \rho_{00}|T\rangle\langle T| + \rho_{11}|T^{\perp}\rangle\langle T^{\perp}|. \quad (4)$$

We therefore assume that the noisy T state is of the form $\rho = (1-q)|T\rangle\langle T| + q|T^{\perp}\rangle\langle T^{\perp}|$, or equivalently that each T state is afflicted by a Z error with probability q. Due to this simplified form of the noise, we use *infidelity*, which is defined by $1 - \langle T|\rho|T\rangle$, interchangeably with the noise rate and noise strength to refer to the single parameter q when describing errors on T states.

For the purpose of defining pseudothresholds 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 nontrivial operator applied to it over t time units under this noise model. It can be calculated as follows:

$$p_{\text{phy}}(t) = 1 - \sum_{P_1P_2\cdots P_t = I} \Pr(P_1)\Pr(P_2)\cdots\Pr(P_t), \quad (5)$$

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, 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 the 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 $i = 1, \ldots, b$ 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

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

Note that when reporting estimated values, we 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.

We often consider the failure probability $\overline{p}_{\text{task}}$ of various tasks using a distance d code and noise strength p. We use the following ansatz that characterizes the generic feature that $\overline{p}_{\text{task}}$ decreases exponentially with d for any p below the threshold value p^*, i.e.,

$$\overline{p}_{\text{task}}(p, d) = \alpha(p)\beta(p)^d, \tag{7}$$

where $\alpha(p)$ and $\beta(p)$ are functions of p alone. In particular, $\beta(p)$ is smaller than one for $p < p^*$. This can be considered a generalization of the heuristic behavior of error-correction failure rate $(p/p^*)^{(d+1)/2}$ in topological codes for error rate p in the vicinity of their threshold p^* [35,81,82].

B. Basics of 2D and 3D color codes

Here we briefly review some important features of color codes, focusing on 2D and 3D. 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. [57,66,83].

Color codes are topological QEC codes, which can be defined on any D-dimensional lattice composed of D-simplices with $(D+1)$-colorable vertices, where $D \geq 2$. Recall that i-simplices for $i = 0, 1, 2, 3$ 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 $x + z \leq D - 2$. 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 2D triangular lattice with a triangular boundary $\mathcal{L}_{2\text{D}}$, and a 3D bcc lattice with a tetrahedral boundary $\mathcal{L}_{3\text{D}}$; see Fig. 5. We occasionally refer to $\mathcal{L}_{2\text{D}}$ as the *triangular lattice* and $\mathcal{L}_{3\text{D}}$ 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 [83]; see Fig. 5(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 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

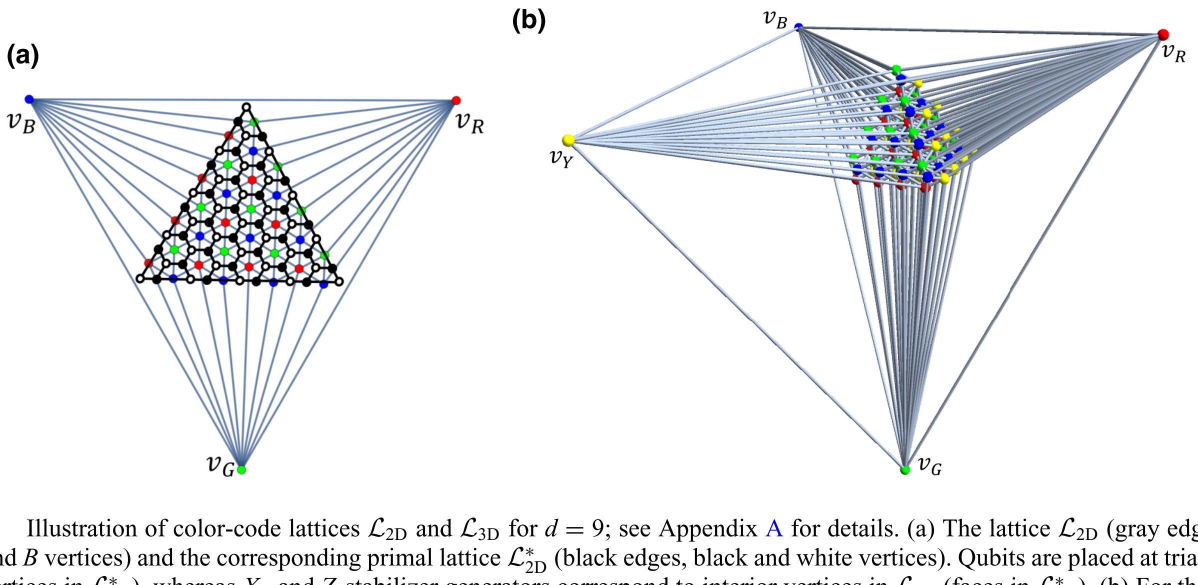


FIG. 5. Illustration of color-code lattices $\mathcal{L}_{2\text{D}}$ and $\mathcal{L}_{3\text{D}}$ for $d = 9$; see Appendix A for details. (a) The lattice $\mathcal{L}_{2\text{D}}$ (gray edges with R, G, and B vertices) and the corresponding primal lattice $\mathcal{L}^*_{2\text{D}}$ (black edges, black and white vertices). Qubits are placed at triangles in $\mathcal{L}_{2\text{D}}$ (vertices in $\mathcal{L}^*_{2\text{D}}$), whereas X- and Z-stabilizer generators correspond to interior vertices in $\mathcal{L}_{2\text{D}}$ (faces in $\mathcal{L}^*_{2\text{D}}$). (b) For the lattice $\mathcal{L}_{3\text{D}}$, qubits are identified with tetrahedra, whereas X- and Z-stabilizer generators correspond to interior vertices and interior edges, respectively. Note that $\mathcal{L}_{2\text{D}}$ can be obtained from $\mathcal{L}_{3\text{D}}$ by retaining only those vertices connected to the boundary vertex v_Y along with the edges and faces containing only those 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. For any subset of edges $\gamma \subseteq \Delta_1(\mathcal{L})$ and any vertex $v \in \Delta_0(\mathcal{L})$ we define a restriction $\gamma|_v$ as the subset of edges of γ incident to v. We separate vertices in $\mathcal{L}$ into two types: *boundary vertices* [the three and four outermost vertices in Figs. 5(a) and 5(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 Figs. 5(a) and 5(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 2D 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 stringlike logical operators. One can implement the full Clifford group transversally on the 2D color code.

On the 3D lattice, we can have either a stabilizer color code [84] (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 stringlike, respectively. One can implement the full Clifford group transversally 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 sheetlike, 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:

$$\partial_{a,b}\alpha = \begin{cases} \sum_{\beta \in \Delta_b(\mathcal{L}):\beta \supseteq \alpha} \beta & \text{if } a \leq b, \\ \sum_{\beta \in \Delta_b(\mathcal{L}):\beta \subset \alpha} \beta & \text{if } a > b. \end{cases} \tag{8}$$

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 Calderbank Shor Steane (CSS) [85,86], we can cast the decoding problem in terms of chain complexes, and treat X and Z errors and correction independently [30,31]. 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 pointlike 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 looplike and pointlike 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 Sec. IV B we discuss in detail the structure of the looplike gauge measurement outcomes for the 3D subsystem color code.

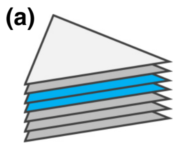
C. Fault-tolerant computation with 2D color codes

Here we briefly review an approach to implement quantum computation with a 3D stack of the 2D color codes as in Fig. 6(a). Note that alternatively we could lay out the 2D color codes on a 2D plane and use lattice surgery [87], although many of the elementary logical operations, e.g., CNOT gates, would be slower than for a stack.

Error correction and decoding.—On each patch of the 2D color code, QEC is continuously performed. We use the term *QEC cycle* to refer to a full cycle 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, a *perfect measurement decoder* is used to infer a correction C for any error E given the input $\sigma(E)$. The correction C will return the system to the code space if applied. The perfect measurement decoder fails if and only if CE is a nontrivial logical operator.

In a scenario in which measurements are not perfect, we consider a sequence of QEC cycles $t = 1, \ldots n_{\text{cyc}}$, with



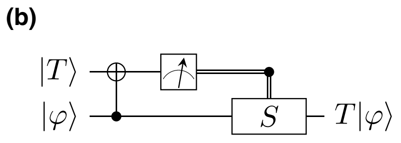


FIG. 6. (a) By stacking 2D color codes, one can implement transversal CNOT and SWAP operations between adjacent patches (colored in blue) with geometrically local gates. (b) A T state can be used to implement a T gate via gate teleportation using only Clifford operations.

error E_t introduced during each cycle, and observed syndrome σ_t for each. For simplicity we assume that the first and last cycle have no additional error and that the syndrome is measured perfectly. Then a *faulty measurement decoder* is a decoder with takes the full history of syndromes $\sigma_1, \sigma_2, \ldots \sigma_{n_{\text{cyc}}}$ and outputs a correction C such that $CE_1 \cdots E_{n_{\text{cyc}}}$ has trivial syndrome. The faulty measurement decoder fails if and only if $CE_1 \cdots E_{n_{\text{cyc}}}$ is a nontrivial logical operator. We discuss decoders for the 2D color code in detail in Sec. II.

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

(a) *State preparation.*—To prepare the $|\overline{0}\rangle$ state each data qubit is prepared in $|0\rangle$, then d QEC cycles are performed, and the X-type syndrome outcomes σ at the first cycle are inferred (d cycles are needed to do so fault tolerantly). A Z-type fixing operator with the syndrome σ is applied. The $|\overline{+}\rangle$ state is prepared analogously.
(b) *Idle operation.*—A single QEC cycle is performed.
(c) *Single-qubit Clifford gates.*—As the gates are transversal, these are done in software by Pauli frame updates so are instantaneous and perfect.
(d) CNOT *gate.*—This is implemented by the application of a transversal CNOT between adjacent patches in the stack, and followed by $n_{\text{cyc}}^{\text{after}}$ QEC cycles; see Sec. II D for details.
(e) SWAP *gate.*—This is implemented by swapping the data qubits on adjacent patches, and followed by a single QEC cycle.
(f) *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 parity of the corrected 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.

(a) *T gate*. It is implemented using gate teleportation of an encoded T state $|\overline{T}\rangle = (|\overline{0}\rangle + e^{i\pi/4}|\overline{1}\rangle)/\sqrt{2}$ as shown in Fig. 6(b). We consider the production of the encoded T state by both state distillation and code switching.

D. State distillation

Here we briefly review state distillation [50,51], in which Clifford operations are used with postselection to convert noisy resource states into fewer but crucially less-noisy resource states. In our analysis of the overhead of state distillation in Sec. III, we consider only the standard 15-to-1 state-distillation protocol. However, given the wide range of state-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 state-distillation protocols, 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 [62], for which the 15-to-1 protocol that we consider is a special instance. In our discussion here we assume Clifford operations are perfect, although we relax that assumption in Sec. III.

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 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. 6(b). The three classes of state-distillation protocols are then summarized as follows (and depicted Fig. 7).

(a) *Code projector then resource rotator.*—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. See schemes in Refs. [50,60,62,88,89].
(b) *Resource stabilizer then code projector.*—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 postselection is less than one even for perfect resource states. See schemes in [50,90].
(c) *Code projector then resource stabilizer.*—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 [91]. 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 postselect on the $+1$ outcome. To measure $\overline{C_R}$ we use

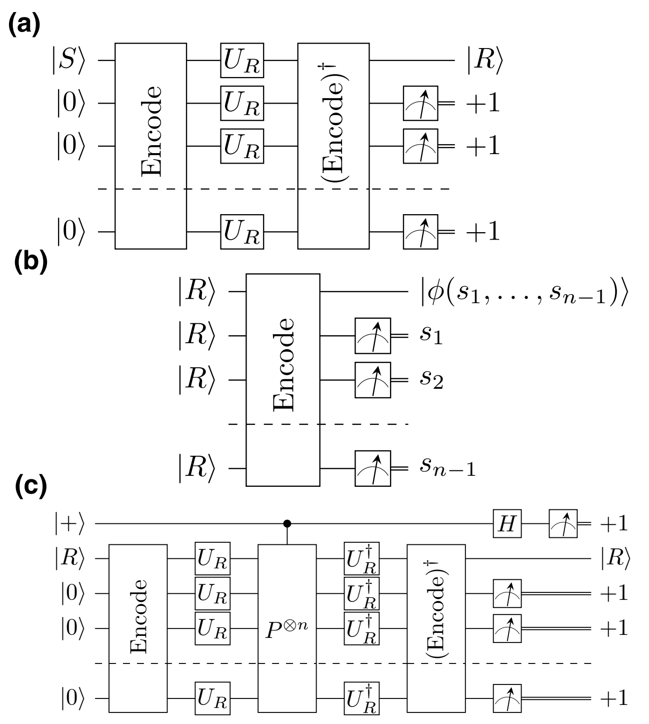


FIG. 7. The three known types of state-distillation schemes for a resource state $|R\rangle$. Each wire represents a logical qubit of a QEC scheme that fault tolerantly performs all Clifford gates. (a) We need a code with a transversal non-Clifford gate U_R, such that $U_R|S\rangle = |R\rangle$ for some 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 postselected 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.

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. See schemes in Refs. [51,52,92–96].

Historically, the first state-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 classification, but which has the same parameters as Knill's (later the two schemes were shown to be mathematically equivalent [97]). The second scheme in Ref. [50] is of type b and has less favorable parameters than the 15-to-1 scheme, but a higher state distillation threshold. Another type-b scheme was found by Reichardt [90], which could successfully distill states arbitrarily close to the stabilizer polytope. In Ref. [60], type-a schemes outputting multiple resource states were proposed. Multiple outputs were also achieved for type-b schemes in Refs. [93,96]. 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 state-distillation protocols of type a [62]. These are based on puncturing quantum Reed-Muller codes and have both interesting asymptotic properties [89] and appear to be practically favorable [62]. We review this family here, which includes the 15-to-1 scheme that we analyze in detail in Sec. III.

Haah-Hastings state-distillation protocols.—This family of state-distillation protocols [62] involves first producing what we call a *Reed-Muller state* $|Q_r\rangle$ 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. 8. 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

$$|Q_r\rangle = \prod_{i=1}^{k} \left(\frac{|0\rangle_i|\overline{0}\rangle_i + |1\rangle_i|\overline{1}\rangle_i}{\sqrt{2}} \right), \tag{9}$$

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 and that transversal T implements a logical $\overline{T}$ on each logical qubit of the code. Let the Z distance d_Z be the weight of the smallest nontrivial Z-type logical operator of this code. 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 $|Q_r\rangle$ state (which, for example, could be specified by propagating the initial single-qubit stabilizers through the CNOT circuit in Fig. 8). Choosing a stabilizer-generator set for $|Q_r\rangle$ in such a way that only one X- and one Z-type generator have nontrivial 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 nonpunctured qubit,

$$T^{\otimes(n-k)}|Q_r\rangle = \prod_{i=1}^{k} \left(\frac{T|+\rangle_i \otimes |\overline{+}\rangle_i + T|-\rangle_i \otimes |\overline{-}\rangle_i}{\sqrt{2}} \right). \tag{10}$$

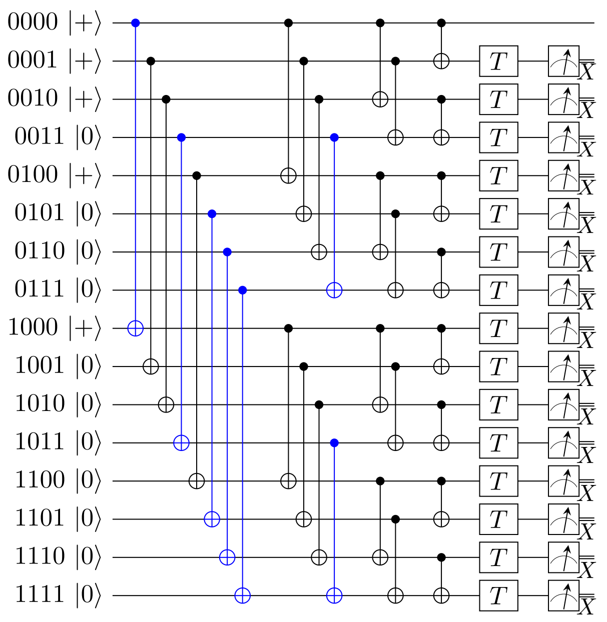


FIG. 8. A 15-to-1 state-distillation circuit as the $r = 1$ instance of the Haah-Hastings construction. Each qubit is labeled by a string of $3r + 1$ bits, and if the bit string has weight at most r, then that qubit is prepared in the $|+\rangle$ state, otherwise in the $|0\rangle$ state. In the jth 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 $|Q_1\rangle \propto |+\rangle|\overline{+}\rangle + |-\rangle|\overline{-}\rangle$, 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 nonpunctured 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 remaining unpunctured qubit is in the state $T^{3-2m_{\overline{X}}}|+\rangle$.

To see this, note that $(I \otimes \overline{T})(|0\overline{0}\rangle + |1\overline{1}\rangle/\sqrt{2}) = (T \otimes I)(|0\overline{0}\rangle + |1\overline{1}\rangle/\sqrt{2}) = (T \otimes I)(|+\overline{+}\rangle + |-\overline{-}\rangle/\sqrt{2})$. 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 $n-k$ nonpunctured qubits in the X basis, and postprocessing. Note that postprocessing 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 $n-k$ encoded T states of infidelity q, and outputs k encoded T states of infidelity Aq^{d_Z}, where A is the number of nontrivial Z-type logical operators of minimal weight d_Z.

In Fig. 8 we show an instance of the Haah-Hastings protocol for $r = 1$, which we analyze more explicitly in Sec. II. 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. 7(a). Some larger instances of this family have very good properties for state distillation, although we do not analyze them in detail here.

II. 2D COLOR CODE ANALYSIS

In this section, we describe an efficient and optimized implementation of the 2D color code under circuit noise. First, in Sec. II 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 Sec. II B. We optimize the stabilizer extraction circuits in Sec. II C, finding 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 Ref. [37], and brings it closer to the toric code threshold of near one percent [98]. 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 Refs. [82,99], and by optimizing extraction circuits and removing the restriction on the qubit connectivity in the case of Ref. [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 $\mathcal{L}_{2D}$, which has boundaries. Our adaptation is essentially the same as that presented in Ref. [99], 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.

We consider the 2D color code on the lattice $\mathcal{L} = \mathcal{L}_{2D}$ from Sec. I 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. 9) can then be described as follows.

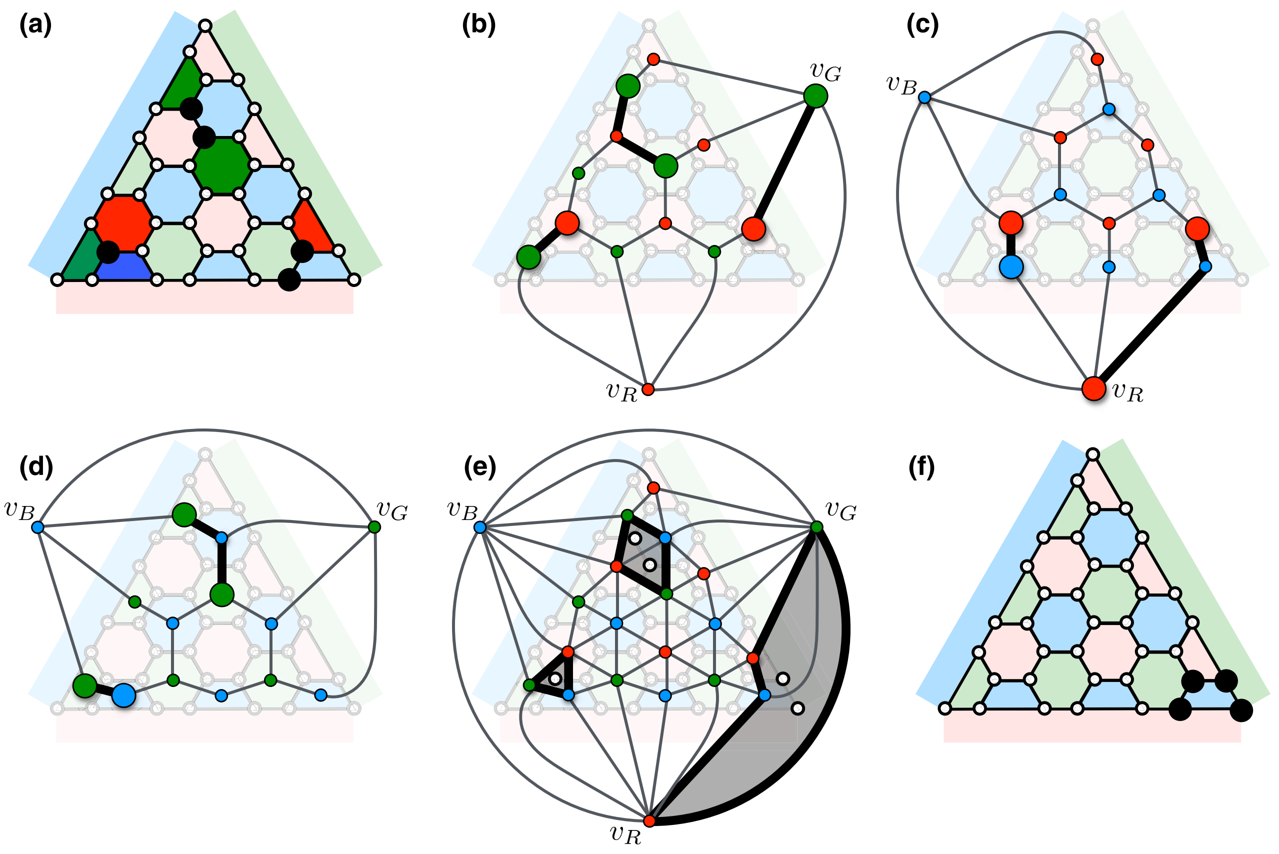


FIG. 9. The projection decoder for the 2D color code of distance $d = 7$ on the lattice $\mathcal{L}_{2D}$. (a) The primal lattice with errors (black dots) and the corresponding syndrome (highlighted faces). (b)–(d) We find pairings E^{RG}, E^{RB}, and E^{GB} (thick lines) of highlighted vertices V^{RG}, V^{RB}, and V^{GB} within the restricted lattices $\mathcal{L}^{RG}$, $\mathcal{L}^{RB}$, and $\mathcal{L}^{GB}$, respectively. Note that in (b),(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).

1. For each pair of colors $\mathcal{K} \in \{RG, RB, GB\}$ we use the minimum weight perfect matching (MWPM) 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 restricted 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 $\Delta \subseteq \Delta_2(\mathcal{L})$ and $\Delta^c = \Delta_2(\mathcal{L}) \setminus \Delta$ and we choose the correction $\hat{\epsilon}$ to be the smaller of the regions Δ and Δ^c.

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 MWPM algorithm we choose 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. 10(a) we present the failure probability of this decoder under depolarizing noise and perfect measurements. In Fig. 10(b) we consider two distance-dependent quantities that should both converge to the threshold as $1/d$ approaches zero. The first is the pseudothreshold $p^*_{\text{dep}}(d)$, defined as a solution to $p_{\text{fail}}(p, d) = p$. The pseudothreshold 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 $(d + 1)/2$ for $d \equiv 1 \mod 4$. Note that we choose to find the crossing of curves corresponding to distances d and $(d + 1)/2$ because their slopes differ quite significantly near their crossing and thus it is relatively easy to identify its location; the resulting threshold, which is an asymptotic value, should not depend on that choice. From a linear extrapolation of the data we find intercepts 13.16(4)% for $p^\times_{\text{dep}}(d)$ and 12.08(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

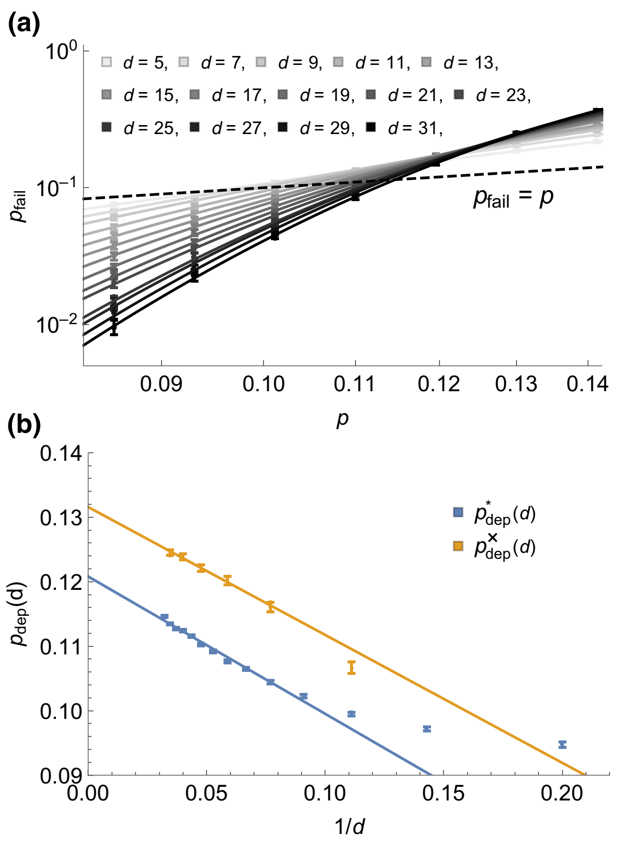


FIG. 10. (a) Performance of the projection decoder for the 2D color code of distance d under depolarizing noise and perfect measurements. (b) The pseudothresholds $p^*_{\text{dep}}(d)$ and the crossings $p^{\times}_{\text{dep}}(d)$ between pairs of curves corresponding to distances d and $(d+1)/2$ for various d. From a linear extrapolation of the data for $d \geq 11$ we obtain intercepts of 13.16(4)% for $p^{\times}_{\text{dep}}(d)$ and 12.08(4)% for $p^*_{\text{dep}}(d)$.

$d = 29$ as a conservative estimate of the threshold under depolarizing noise, i.e., $p^*_{\text{dep}} = 12.45(4)\%$ in accordance with previous threshold estimates in Refs. [28,37].

This analysis of the pseudothresholds and failure-curve crossings highlights two general points. Firstly, the threshold may not be a good proxy to the finite-size performance. For example, we see in Fig. 10(b) that our estimate of the threshold is considerably above the observed pseudothresholds. Secondly, in order to find the threshold from the data for small systems one has to exert caution, as extrapolating different quantities (which nevertheless should recover the same threshold value in the limit of infinite d) can result in inconsistent estimates.

Although we do not proved it, we suspect that our adaptation of the projection decoder can correct all the errors of weight at most $d/3$ (up to an additive constant). Examples of the smallest weight errors, which lead to a logical failure, are the same as for the adapted restriction decoder in Ref. [37] and can be found in Appendix A thereof.

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 QEC 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. 11. We use a shorthand notation $\Lambda_{[t_1,t_2]}$ to denote the part of Λ within QEC cycles t_1 and t_2. By a QEC cycle, we simply mean a full cycle of stabilizer measurements. Temporal edges, which vertically connect corresponding vertices in two copies of $\mathcal{L}$ at QEC cycles t and $t+1$, correspond to stabilizer measurements at QEC cycle t.

We use the same concepts and nomenclature for the lattice Λ as for $\mathcal{L}$ in Sec. I B. For instance, we say a vertex (v, t) of Λ is a boundary vertex if and only if v is a boundary vertex of $\mathcal{L}$; otherwise, it is an interior vertex. An edge of Λ is a boundary edge if and only if 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 restricted lattice 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 history of the syndrome σ consisting of the subset of temporal edges corresponding to -1 stabilizer measurement outcomes. We define the set of syndrome

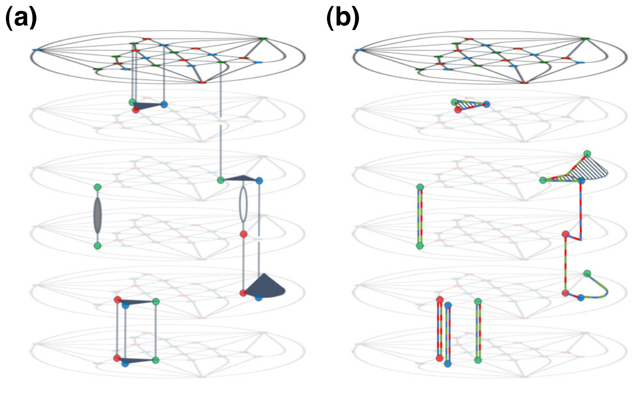


FIG. 11. The noisy-syndrome projection decoder on the (2+1)D color-code lattice Λ. (a) During each QEC cycle, errors can appear on data qubits (shaded triangles), followed by stabilizer measurements, which when incorrect are drawn as ellipses. The observed syndrome σ (dark temporal edges and filled ellipses) 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 restricted lattice Λ. Then, for every "flattened" connected component of $E^{RG} + E^{RB} + E^{GB}$ we find a correction $\epsilon(t)$ (hatched regions).

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 $t \in [n_{\mathrm{cyc}}]$ initialize $F(t) = \emptyset$.
2. For every $\mathcal{K} \in \{RG, RB, GB\}$ use the MWPM algorithm to find the pairing $E^{\mathcal{K}}$ of syndrome flips $V^{\mathcal{K}}$ within the restricted lattice $\Lambda^{\mathcal{K}}$.
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 QEC 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 QEC cycle $t_i^{(2)}$.
5. For each QEC 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 the restricted lattice $\Lambda^{\mathcal{K}}$ 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 QEC cycles, followed by a single cycle 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 QEC cycles with lattice surgery or braiding. Moreover, one would expect the effects of the artificially perfect preparation and final measurement cycle to be negligible over d cycles 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. 12(a) we find the failure probability after d time units of phenomenological noise. In Fig. 12(d) we show crossings $p_{\mathrm{phe}}^{\times}(d)$ 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.

The notion of a pseudothreshold must be revisited in the setting of faulty measurements and we cannot extract a meaningful pseudothreshold directly from these curves as we did for the perfect measurement case in Fig. 10. Consider the scenario in which we assume a perfect initial code state and a perfect final measurement cycle, but consider the performance over a varying number n_{cyc} of noisy QEC cycles; see Fig. 12(b). For each d and n_{cyc}, we define the time-dependent pseudothreshold $p_{\mathrm{phe}}^{*}(d, n_{\mathrm{cyc}})$ as the error rate at which the encoded failure probability after n_{cyc} QEC 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 pseudothreshold $p_{\mathrm{phe}}^{*}(d)$, as can be seen in Fig. 12(c). The following ansatz

$$p^*(d, n_{\mathrm{cyc}}) = p^*(d)\left(1 - \left[1 - \frac{p^*(d,1)}{p^*(d)}\right] n_{\mathrm{cyc}}^{-\gamma}\right). \tag{11}$$

fits the data well, and allows us to extract an estimate of the long-time pseudothreshold. We remark that our approach of finding long-time pseudothresholds is similar in spirit, but not exactly the same as the one used to find the "sustainable threshold" [100,101]. Also, the ansatz in Eq. (11) was used in Ref. [102] to analyze thresholds of cellular-automata decoders for topological codes.

The long-time pseudothresholds $p_{\mathrm{phe}}^{*}(d)$, as well as the failure curve crossings $p_{\mathrm{phe}}^{\times}(d)$ 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 4.38(3)% for $p_{\mathrm{phe}}^{*}(d)$ and 3.67(3)% for $p_{\mathrm{phe}}^{\times}(d)$; see Fig. 12(d). Note that $p_{\mathrm{phe}}^{*}(d)$ appears to converge more quickly than $p_{\mathrm{phe}}^{\times}(d)$. 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., $p_{\mathrm{phe}}^{*} = 4.19(4)\%$.

We remark that the modification of the projection decoder that we present here to handle noisy syndrome measurements differs from that discussed by Stephens [99] 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 Ref. [99] 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

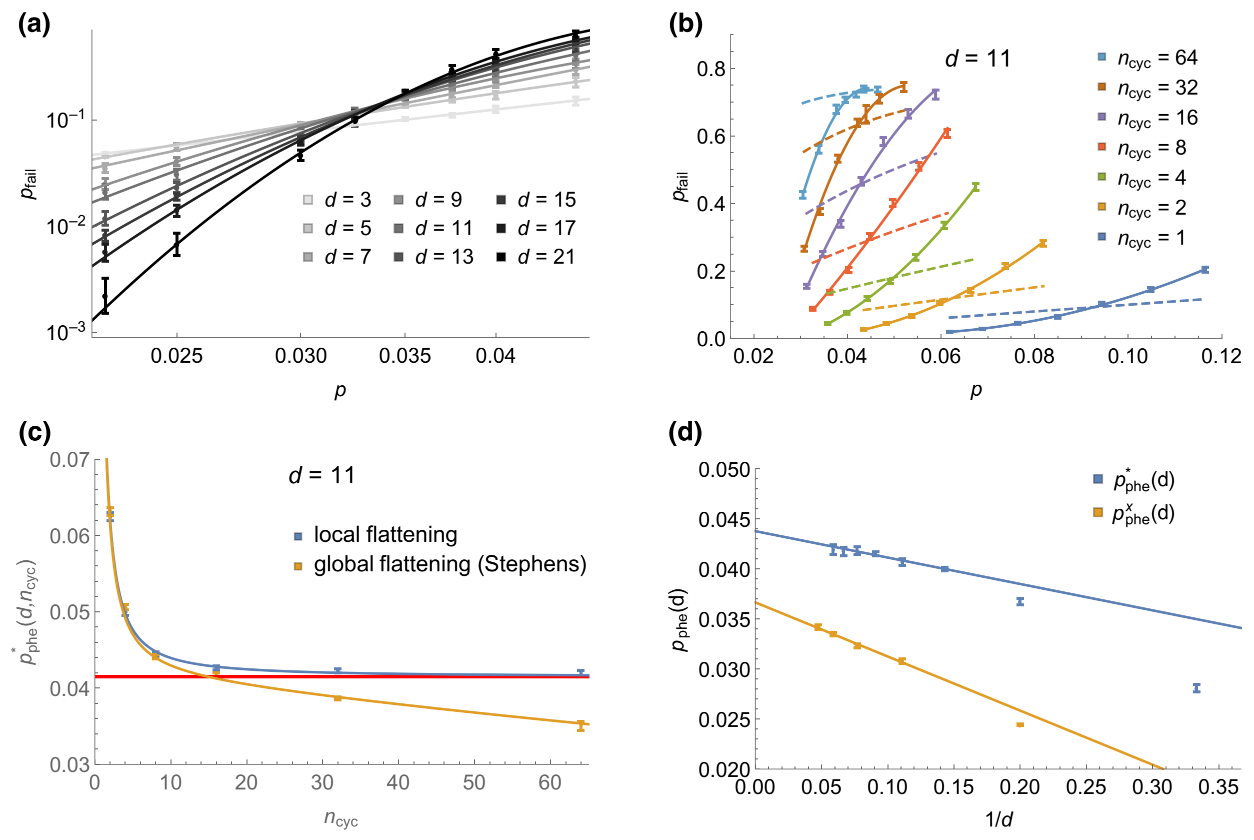


FIG. 12. (a) Failure probability $p_{\mathrm{fail}}(d, n_{\mathrm{cyc}})$ of the noisy-syndrome projection decoder for $n_{\mathrm{cyc}} = d$ time units of phenomenological noise and various distances d. (b) We estimate the time-dependent pseudothreshold $p^*_{\mathrm{phe}}(d, n_{\mathrm{cyc}})$ from 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 $d = 11$. See Appendix B for the analysis of other distances. (c) We estimate the long-time pseudothreshold $p^*_{\mathrm{phe}}(d) = 4.15(2)\%$ for $d = 11$ (red) by fitting $p^*_{\mathrm{phe}}(d, n_{\mathrm{cyc}})$ with the numerical ansatz in Eq. (11). Note that Stephens' adaptation of the decoder (yellow), which uses a global flattening, fails to stabilize. (d) The long-time pseudothresholds $p^*_{\mathrm{phe}}(d)$ (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 4.38(3)% for $p^*_{\mathrm{phe}}(d)$ and 3.67(3)% for $p^\times_{\mathrm{phe}}(d)$.

decoder, we verify that the time-dependent pseudothreshold for this decoder fails to stabilize for large n_{cyc}; see Fig. 12(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 assume that there is a separate ancilla qubit per stabilizer generator, such that there are two ancillas per face of the lattice $\mathcal{L}_{2D}$, and do not worry about the precise connectivity details, requiring only that coupled qubits are nearby. The total number of qubits required for our implementation of the distance-d 2D color code is therefore

$$N_{2D}(d) = (3d^2 - 1)/2. \tag{12}$$

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 generator, 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 [82] must be satisfied.

(a) At each time unit at most one operation can be applied to any given qubit.
(b) The measurement circuit preserves the group generated by the elements of the stabilizer group and Pauli-X or -Z operators stabilizing the ancilla qubits [103].

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

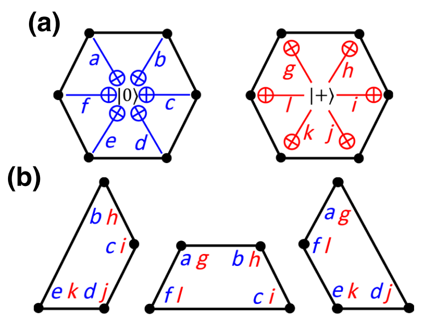


FIG. 13. (a) The CNOT schedule $\{a, b, c, d, e, f\,; g, h, i, j\,, k, l\}$ is a list specifying the time units when each CNOT gate is applied. At the end, the ancillas, which are initially prepared in $|0\rangle$ and $|+\rangle$, are measured to extract the Z- (blue) and X- (red) stabilizer measurement outcomes, respectively. (b) The CNOT schedules for stabilizers on faces along the boundary can be viewed as restrictions of the CNOT schedule in (a).

CNOT schedule for stabilizer generators along the boundary of the lattice are specified by restricting the schedule for those in the bulk; see Fig. 13.

The CNOT schedule includes 12 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 12 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 [82].
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.

   (a) 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 $(j\,-f\,)(i-a) > 0$.

   (b) 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 $(a-g)(d-j\,)(e-k)(f\,-l) > 0$.

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.

We remove an ordering from the list of valid orderings if it is equivalent to another ordering in the list up to a symmetry of the lattice $\mathcal{L}_{2\mathrm{D}}$. No schedules with 6 time units satisfy all these conditions. However, we find that there are 234, 4854, and 39160 valid orderings for 7, 8, and 9 time units, respectively.

To select a good CNOT schedule among the large number of valid ones, we focus on the 234 shortest schedules, because fewer time units in an error-correction cycle tends to translate into fewer possible faults and better performance. We test each using a $d = 7$ code for d QEC cycles with circuit noise of strength $p = 0.0035$ and estimate the failure probability by sampling. This value of p is chosen because preliminary studies indicated it was close to the threshold, and distance $d = 7$ is chosen as a compromise between reducing the simulation run time and limiting the impact of boundary effects. The limited sampling resources are focused on identifying the best CNOT schedules; see Fig. 14(a). We find that up to sampling error, $\{4, 1, 2, 3, 6, 5; 3, 2, 5, 6, 7, 4\}$ is the best-performing schedule, with a logical failure probability of 12.8(1)%. For comparison, the worst-performing length-7 schedule is $\{4, 1, 2, 7, 6, 3; 1, 6, 7, 4, 5, 2\}$ and results in substantially worse logical failure probability of 21.1(1)%. Note that the QEC cycle for the best schedule requires only 8 time units to implement by preparing and measuring the X ancilla at time units 0 and 7, respectively, and preparing and measuring the Z ancilla at time units 1 and 8, respectively.

Now we focus on the best-performing CNOT schedule under circuit noise and find the long-time pseudothreshold for a range of distances in Fig. 14(b) using the same approach as in Sec. II B. Unlike in the cases for depolarizing and phenomenological noise, the data for circuit noise appears to be in the regime where both $p^{*}_{\mathrm{cir}}(d)$ and $p^{\times}_{\mathrm{cir}}(d)$ can be fitted with a linear fit and their intercepts agree to within error. We take their shared intercept as an estimate of the threshold under circuit noise $p^{*}_{\mathrm{cir}}(d) = 0.37(1)\%$.

To estimate the impact of this kind of circuit optimization, we compare the best- and worst-performing length-7

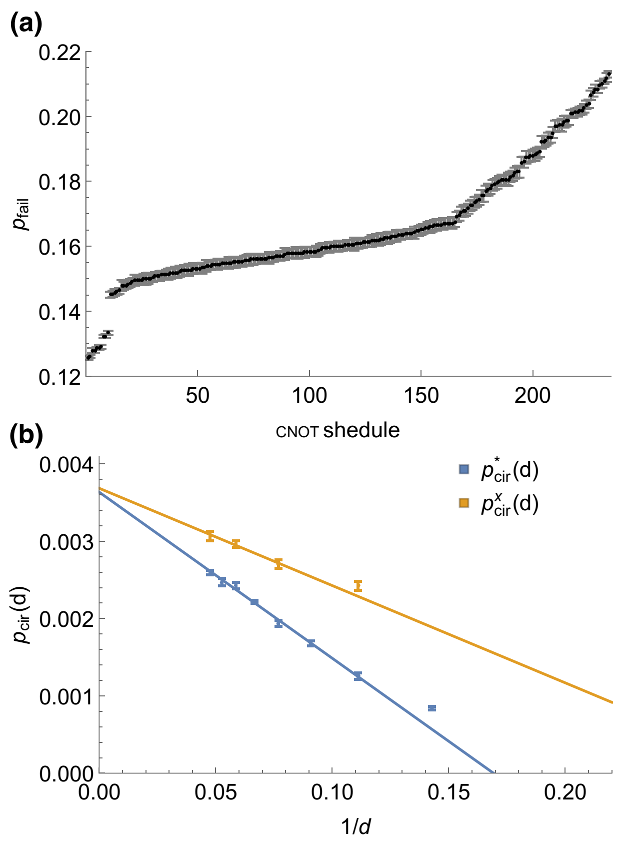


FIG. 14. (a) Logical failure probability over $n_{\text{cyc}} = 7$ cycles for distance $d = 7$ and depolarizing circuit noise of strength $p = 0.0035$ for each of the 234 CNOT schedules, sorted by their failure rate. (b) The long-time pseudothresholds $p^*_{\text{cir}}(d)$ and crossings $p^\times_{\text{cir}}(d)$ for the best-performing CNOT schedule $\{4, 1, 2, 3, 6, 5; 3, 2, 5, 6, 7, 4\}$ under circuit noise; see Appendix B for further details. From a linear extrapolation of the data for $d \geq 11$ we obtain intercepts of 0.366(6)% for $p^*_{\text{cir}}(d)$ and 0.37(1)% for $p^\times_{\text{cir}}(d)$.

CNOT schedules and find that the long-time pseudothreshold differs by a factor of almost 2 for distance $d = 13$; see Appendix B for more details.

D. Modeling noise in logical operations

Now we describe an effective noise model for logical operations in the optimized 2D color code under circuit noise, which we later use to estimate the performance of state-distillation circuits. For circuit-noise strength p and code distance d, the effective noise model is specified in terms of the overall failure probability $\overline{p}_{\text{oper}}(p, d)$ of each logical operation oper = prep, idle, CNOT, which we estimate numerically and record in Table I in the form of the ansatz in Eq. (7).

In our simulations each operation is followed by a full decoding. This results in the application of a logical Pauli operator, which if nontrivial is interpreted as a failure. Our

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 report only the measurement failure probability for $p = 10^{-3}$.

Logical operation	Failure probability $\overline{p}_{\text{oper}}(p, d)$	
	0.0151×0.812^d	for $p = 10^{-3}$
Prep	0.0074×0.719^d	for $p = 5 \times 10^{-4}$
	0.0041×0.527^d	for $p = 10^{-4}$
	0.0124×0.721^d	for $p = 10^{-3}$
Idle	0.0092×0.618^d	for $p = 5 \times 10^{-4}$
	0.0018×0.484^d	for $p = 10^{-4}$
	0.0972×0.894^d	for $p = 10^{-3}$
CNOT	0.0603×0.761^d	for $p = 5 \times 10^{-4}$
	0.0078×0.607^d	for $p = 10^{-4}$
Meas	0.0011×0.690^d	for $p = 10^{-3}$

effective noise model is designed to overestimate the probability of each nontrivial logical Pauli, and assumes each operation fails independently of others. Specifically,

(a) for state preparation noise is modeled by preparing an orthogonal logical state to that intended with probability $\overline{p}_{\text{prep}}(p, d)$,
(b) for an idle operation, noise is modeled by applying $\overline{X}$ or $\overline{Z}$ each with probability $\overline{p}_{\text{idle}}(p, d)/2$ or $\overline{Y}$ with probability $\overline{p}_{\text{idle}}(p, d)/20$,
(c) single-qubit Clifford gates are done in software by Pauli-frame updates so are noise-free,
(d) for SWAP, noise is modeled 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 $\overline{p}_{\text{idle}}(p, d)/20$,
(e) for CNOT noise is modeled 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 nontrivial Pauli each with probability $\overline{p}_{\text{CNOT}}(p, d)/20$,
(f) measurements in the logical Pauli bases are assumed to be perfect.

Let us make a number of remarks about this noise model. Firstly, each failure mode for a given operation occurs independently in the model for simplicity. Secondly, the conservative estimates of different types of failure are quite loose for large distances, and could be made tighter by fine tuning the noise model. Thirdly, one may wonder why we treat the SWAP operation as a pair of idle qubits, whereas we treat the CNOT operation separately and find it has a considerable failure probability. The reason is that 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

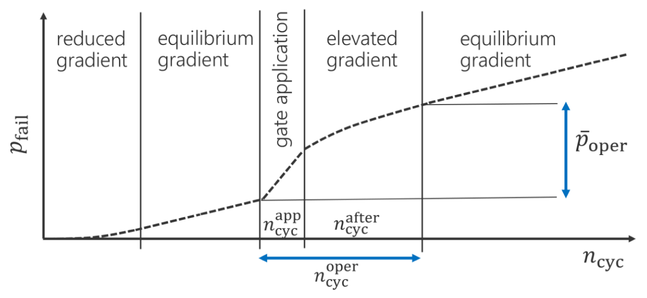

reduced gradient
equilibrium gradient
gate application
elevated gradient
equilibrium gradient
p_{fail}
$\bar{p}_{\mathrm{oper}}$
$n_{\mathrm{cyc}}^{\mathrm{app}}$
$n_{\mathrm{cyc}}^{\mathrm{after}}$
$n_{\mathrm{cyc}}^{\mathrm{oper}}$
n_{cyc}

FIG. 15. Sketch of the cumulative failure probability before, during, and after a logical operation. Logical preparations and measurements comprise only some parts of this sketch. Before we apply the operation, we first make sure that the system reaches QEC equilibrium. The logical operation, which would take $n_{\mathrm{cyc}}^{\mathrm{app}}$ cycles by itself, is augmented with $n_{\mathrm{cyc}}^{\mathrm{after}}$ idle cycles to ensure the system has returned to QEC equilibrium afterwards. The logical operation failure rate $\bar{p}_{\mathrm{oper}}$ is the increment of p_{fail} over the $n_{\mathrm{cyc}}^{\mathrm{oper}} = n_{\mathrm{cyc}}^{\mathrm{app}} + n_{\mathrm{cyc}}^{\mathrm{after}}$ QEC cycles.

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 QEC equilibrium both before and after the logical operation (or just before or after for measurement or preparation, respectively); see Fig. 15. We therefore implement d QEC cycles on the patches prior to the operation and, when necessary, augment the logical operation by including additional idle QEC cycles at the end of the operation, checking that the residual noise has stabilized before the perfect measurement is implemented. By QEC equilibrium, we mean a regime in which the failure probability increases linearly with the number of QEC cycles. Throughout our analysis we use the best-performing CNOT schedule from Sec. II C, which uses one ancilla per stabilizer generator. This results in $(3d^2 - 1)/2$ qubits being needed for a distance-d patch.

Logical state preparation.—Recall that $|\overline{0}\rangle$ is prepared by initializing data qubits in $|0\rangle$ and then measuring the stabilizers for d QEC cycles under circuit noise of strength p, and applying a Z-type operator intended to fix the X stabilizers, and standard error correction to correct X errors. 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. 16 and fitted with the ansatz in Eq. (7) using the same parameters. This fit provides the entry for $\bar{p}_{\mathrm{prep}}(p,d)$ in Table I.

Idle logical qubit.—We extract $\bar{p}_{\mathrm{idle}}(p,d)$, the failure probability of an idle logical qubit for a single QEC cycle, by considering the failure probability of a single patch of distance-d 2D color code over n_{cyc} QEC cycles. We use the noisy measurement decoder to decode the full history of the n_{cyc} QEC cycles given a perfect initial state and a perfect additional final QEC cycle. After a few initial

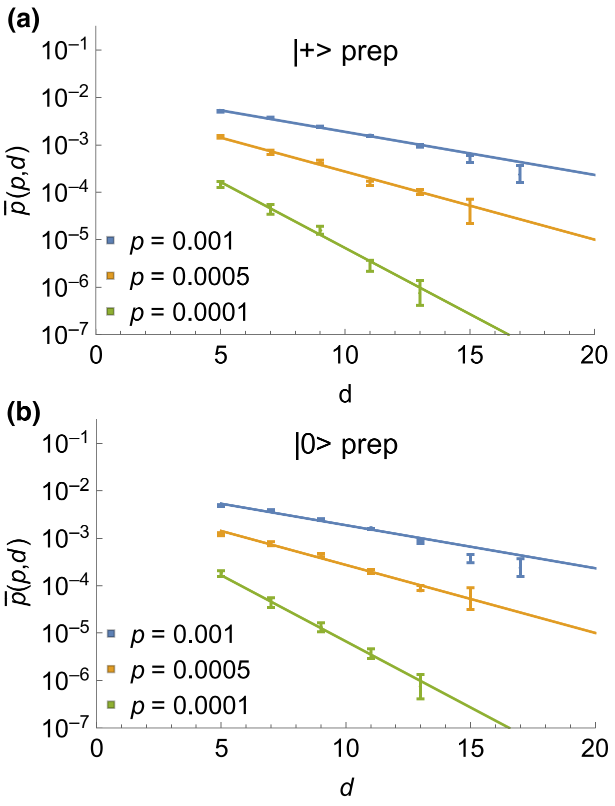


FIG. 16. Logical failure probability for preparing the $|\overline{+}\rangle$ and $|\overline{0}\rangle$ states. The ansatz in Eq. (7) is fitted, giving $\bar{p}_{\mathrm{prep}}(p,d)$ in Table I. Both plots can be fitted well using the same parameters, which justifies treating both $|\overline{+}\rangle$ and $|\overline{0}\rangle$ preparation identically in the noise model.

QEC cycles, 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 $p_{\mathrm{fail}}(d,n_{\mathrm{cyc}})$. We can use linear growth of $p_{\mathrm{fail}}(d,n_{\mathrm{cyc}})$ with time as a hallmark of the system being in QEC equilibrium. To estimate $\bar{p}_{\mathrm{idle}}(p,d)$ for fixed p and d, we fit the failure probability $p_{\mathrm{fail}}(d,n_{\mathrm{cyc}})$ data, such as that presented in Fig. 17(a) for $p = 10^{-3}$, with the following linear function of n_{cyc}, i.e.,

$$p_{\mathrm{fail}}(d,n_{\mathrm{cyc}}) = \bar{p}_{\mathrm{idle}}(p,d) \times n_{\mathrm{cyc}} + c(p,d), \tag{13}$$

where $c(p,d)$ is a constant. We plot this and the fit in Fig. 17(a) for a particular value of p, and use the gradient to estimate $\bar{p}_{\mathrm{idle}}(p,d)$ for various p and d in Fig. 17(b). We fit Eq. (7) to the data, giving the entry in Table I. Note that in Fig. 17(a) the y intercept is negative since the simulation begins with a perfect code state and so the probability of failure during the early QEC cycles is artificially reduced. For later logical operations we assume that d QEC cycles prior to the logical operation is sufficient to ensure the system has reached QEC equilibrium.

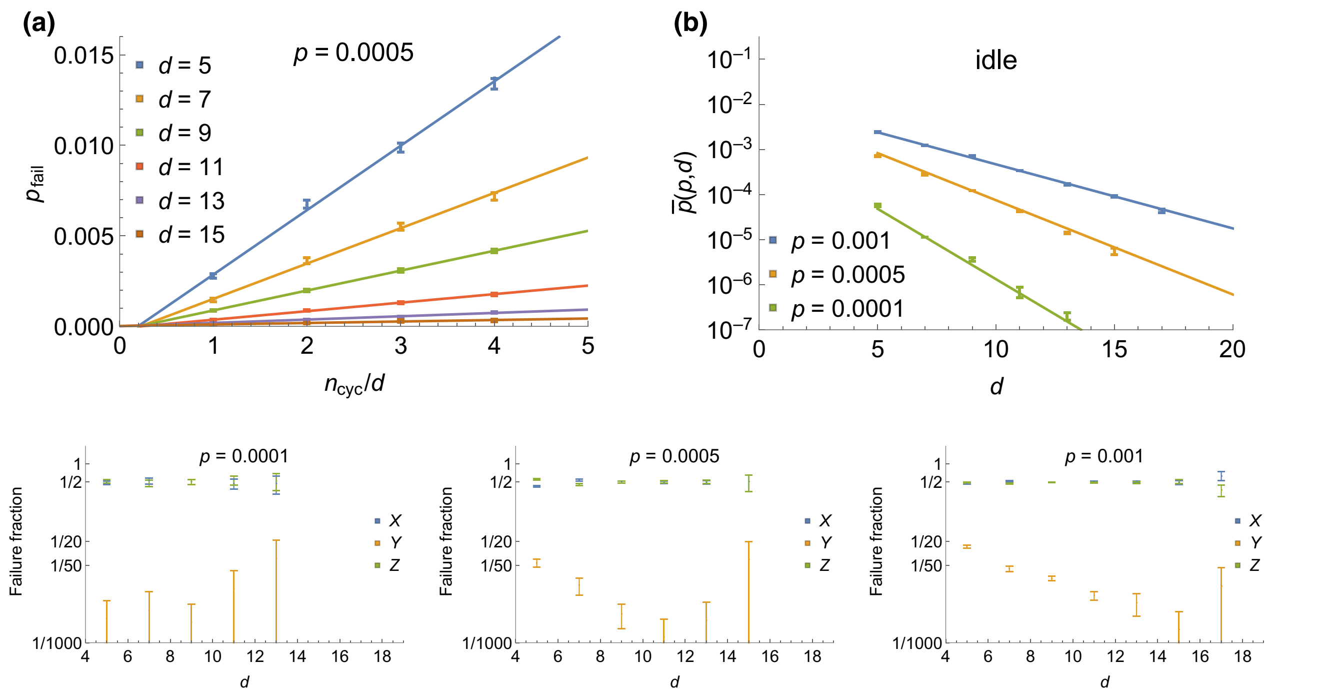


FIG. 17. Analysis of the logical idle operation. (a) The failure probability using the best-performing CNOT schedule as a function of n_{cyc}/d for $p = 5 \times 10^{-4}$. We use the ansatz in Eq. (13) to find the logical failure rate $\overline{p}(p,d)$. We observe that QEC equilibration occurs by the dth cycle; see Appendix 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. (7) 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.

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 the bottom panels of Fig. 17, 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.

Transversal logical CNOT.—The logical CNOT is implemented transversally between a pair of 2D color-code patches. We assume the system is in QEC equilibrium before the gate, which is ensured in the simulation by running d QEC cycles 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 QEC equilibrium, we include $n_{\text{cyc}}^{\text{after}}$ QEC cycles after the gate is applied in the logical CNOT operation. We conclude from Fig. 18(a) that $n_{\text{cyc}}^{\text{after}} = 2$ is sufficient, which we then incorporate into the logical CNOT operation. We analyze the overall failure probability of the logical CNOT in Fig. 18(b), and the fraction of failures resulting in each logical Pauli in the panels at the bottom of Fig. 18. Note that in contrast with the phenomenon observed in Fig. 17(a) in which the initial system equilibrated from a state of lower noise, here the system equilibrates from a state of higher noise.

To decode each patch, we use 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 QEC cycles 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 QEC cycles by applying a perfect-measurement decoding on the system immediately after the d QEC cycles, and propagate that through the logical CNOT gate [104].

Logical readout.—To fault tolerantly read off $\overline{Z}$, one measures each data qubit in 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 QEC cycles to ensure equilibration 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

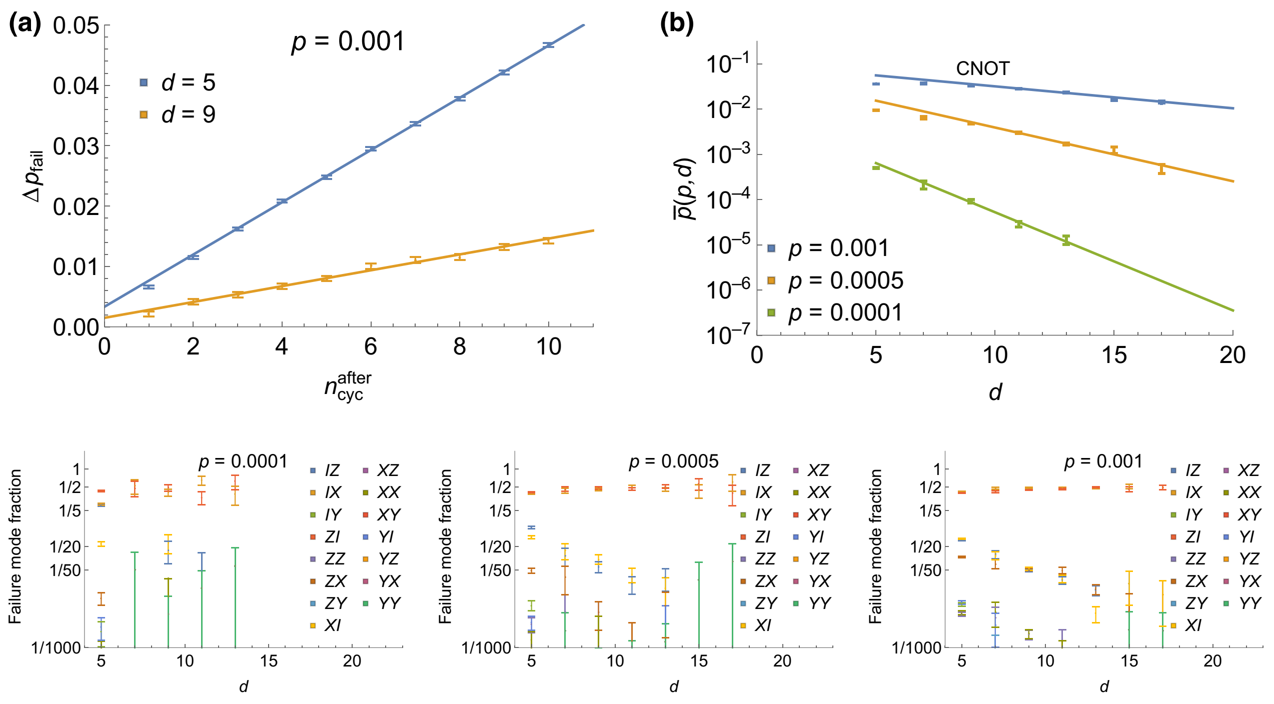


FIG. 18. Analysis of the logical CNOT operation. (a) The change in failure probability between the time immediately after the transversal application of CNOT gates and $n_{\mathrm{cyc}}^{\mathrm{after}}$ QEC cycles later. We observe that QEC equilibration occurs after two QEC cycles. (b) For each value of p, we use the ansatz in Eq. (7) to extract the parameters for $\overline{p}_{\mathrm{CNOT}}(p,d)$ in Table I. (Bottom panels) The fraction of overall failures corresponding to different logical Pauli errors. The dominant errors are $\overline{ZI}$ and $\overline{IX}$, since the logical CNOT propagates X errors onto the second patch and Z errors onto the first patch. Other noticeable errors include $\overline{IZ}$ and $\overline{XI}$, which correspond to the failure over two QEC cycles, but become negligible for large d. The points whose data range falls below the horizontal axis correspond to no observed failures.

logical measurement alone, we find and remove the contribution to the logical operator from the initial d QEC cycles by applying a perfect-measurement decoding on the system immediately after the d QEC cycles. We estimate the failure probability and mode fraction for logical measurement of $\overline{X}$ and $\overline{Z}$ in Fig. 19. We fit Eq. (7) to the data for both $\overline{X}$ and $\overline{Z}$ measurements, which agree with one another and this fit provides the entry for $\overline{p}_{\mathrm{meas}}(p,d)$ in Table I.

Note that the noise contributed by readout is orders of magnitude below the other logical operations, and we take only data for $p = 10^{-3}$ since a prohibitively large number of samples would be required for $p = 10^{-4}$ and

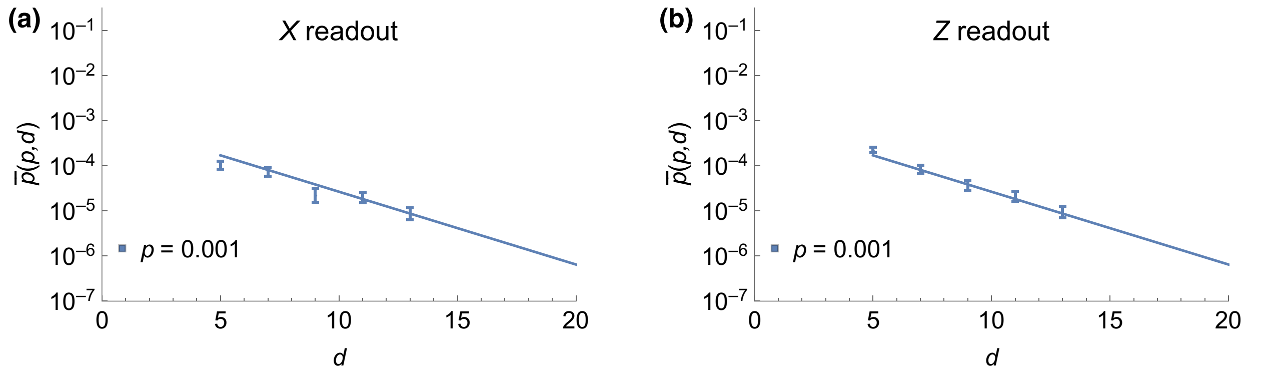


FIG. 19. Readout failure probability for $\overline{X}$ and $\overline{Z}$. Note the value is orders of magnitude lower than for the other logical operations and that we show only that for $p = 0.001$, since the smaller values of p are so low that they are difficult to observe using Monte Carlo. We use the ansatz in Eq. (7) to extract the parameters for $\overline{p}_{\mathrm{meas}}(p,d)$ in Table I, although in our noise model we neglect noise in the measurement.

$p = 5 \times 10^{-4}$. This justifies that we neglect contributions from readout in our effective noise model.

III. STATE-DISTILLATION ANALYSIS

In this section we carefully analyze the performance and estimate the overhead of state distillation of T states using the standard 15-to-1 scheme.

A. Creating the T state via state distillation in three steps

A protocol to produce an encoded T state using state distillation, which we illustrate in Fig. 20, consists of the following steps.

1. *T-state initialization.*—We initialize encoded T states in small-distance 2D color-code patches, which later serve as the input to the first round of state distillation.
2. *Expansion and movement of patches.*—The code distance in consecutive rounds of state distillation is required to increase to protect the produced T states of improving fidelity. We must therefore increase the size of a patch, which is output from one round, and move it to the desired location where it becomes an input for the next round.
3. *State-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 3D stack. The input consists of 15 encoded T states, and the output is one higher fidelity encoded T state.

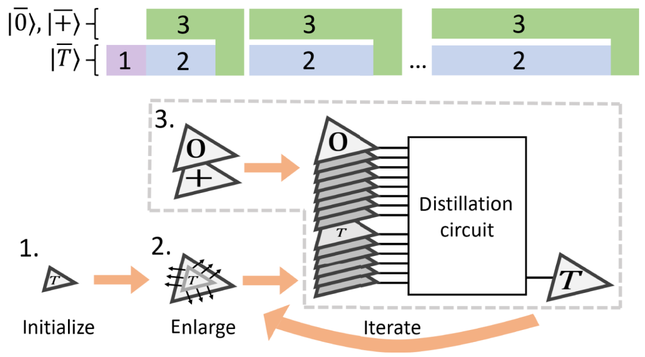


FIG. 20. Protocol to create an encoded T state via state distillation, with a qualitative timeline of each step. In step 1, noisy encoded T states are prepared non-fault-tolerantly in a small 2D color code. In step 2, each code patch is expanded to increase its distance, and we assume that the logical infidelity of the encoded T state does not change. In step 3, 16 logical states, either $|\overline{0}\rangle$ or $|\overline{+}\rangle$, are prepared and then the state-distillation circuit is run on them and 15 $|\overline{T}\rangle$'s. Given successful postselection, the output is a single encoded T state of higher fidelity (indicated by a larger font size). Steps 2 and 3 are iterated until a state of the desired infidelity is produced.

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 a T 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 Sec. II D to analyze the performance of logical-level circuits implemented with the 2D color code.

1. *T-state initialization*

The first step of any state-distillation protocol is to produce the initial encoded T states. The initialization protocol to do this can be crucial since the results of state distillation depend strongly on the quality of the initial T states. For example, taking the 15-to-1 scheme with perfect Clifford operations, if the starting infidelity is decreased by a factor of 2 (which, as we see, can be achieved by varying the CNOT order in the initialization protocol), the output infidelity is reduced by about 1, 3, and 8 orders of magnitude over one, two, and three state-distillation rounds, respectively. Although abstract state-distillation protocols have received a lot of attention, there is surprisingly little research on the initialization of T states as inputs for state distillation despite the enormous potential impact. For the surface code, a non-fault-tolerant scheme with the logical error of the final encoded T state comparable with that of raw state was proposed by Li [105]. More recent works [78,106] present fault-tolerant approaches to initialize encoded T states and can, in some regimes, achieve higher fidelity encoded T states with low overhead, but are somewhat more challenging to implement.

Our strategy of initializing T states for the 2D color code can be viewed as a generalization of the approach in Ref. [105], which consists of two main steps: (i) produce an encoded T 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 than the noise from step (i). We produce an encoded T 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 $|0\rangle$; see Fig. 21(a).
2. Measure each stabilizer twice; see Fig. 21(b). If the observed syndrome is not the same or the syndrome

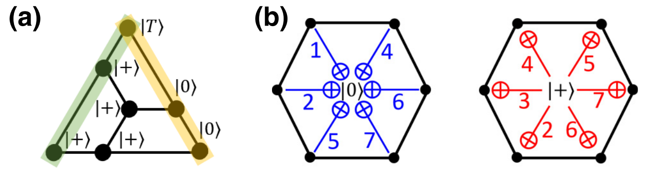


FIG. 21. T-state initialization for the 2D color code with distance $d = 3$. (a) In step 1, the top qubit is prepared in $|T\rangle$, and the remaining data qubits are prepared in either $|+\rangle$ or $|0\rangle$. We depict the support of logical X and Z representatives as shaded yellow and green strips. (b) In step 2, all the stabilizers are measured using the depicted CNOT order (or its restriction).

could not have arisen without fault, the initialization procedure is restarted.

3. Apply a Pauli operator fixing the observed syndrome.

In our simulation, we say the procedure has succeeded in creating an encoded T state if upon a single additional fault-free QEC cycle, one obtains the encoded T 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.

Lastly, we optimize the initialization protocol by varying the order in which the CNOT gates are applied during the two QEC rounds in step 2. We consider a range of system sizes, and again assume that there is a separate ancilla qubit per stabilizer generator and consider the 234 valid schedules consisting of 7 CNOT time units as in Sec. II C. We find that the $d = 3$ size resulted in the best parameters, and that the worst schedule (which is not the same as that used for standard error correction) results in more than twice the lowest-order failure probability of the best schedule; see Appendix C for more details.

The protocol takes

$$\tau^{\text{init}} = 19 \tag{14}$$

time units: one time unit to prepare the qubits in $|0\rangle$, $|+\rangle$ and $|T\rangle$, and two QEC 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

$$p_{\text{fail}}^{\text{init}} = 6.07p, \quad p_{\text{rej}}^{\text{init}} = 126p. \tag{15}$$

2. Expansion and movement of patches

We neglect any error introduced by expansion of the encoded T 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 patches are moved into the starting location for the $i+1$ round. 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 $d^{(i)}$ QEC rounds, and the swaps needed to move the outputs each take just one QEC cycle, which we expect can be done during the $d^{(i+1)}$ QEC rounds needed to prepare the $|\overline{0}\rangle$ and $|\overline{+}\rangle$ states at the start of the state-distillation circuit; see Fig. 22.

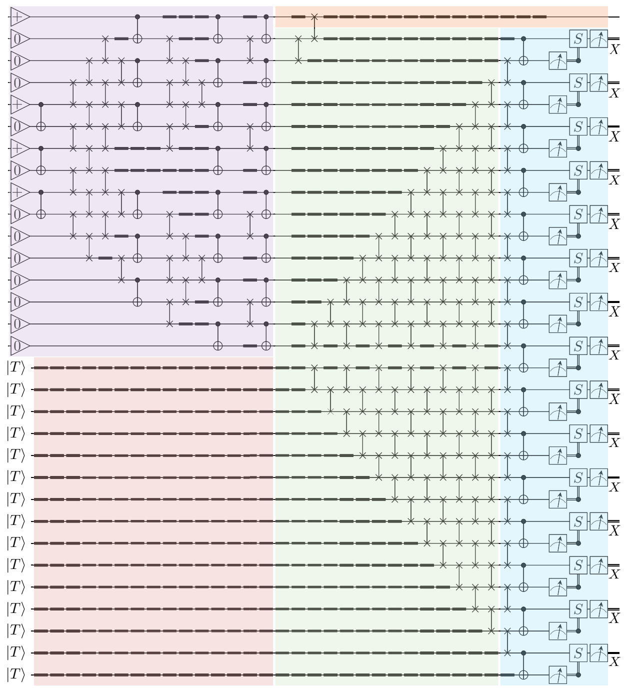


FIG. 22. The 15-to-1 state distillation protocol implemented on the logical level using nearest-neighbor operations in a stack of 2D color codes. The 31 logical qubits are encoded in distance-d 2D color codes, which are stacked on top of one another. Logical idle locations (thick wire segments), SWAPs and CNOTs last one, one, and two QEC cycles, respectively. The 15 input encoded T states remain idle for $d + 16$ QEC cycles (pink) while the Reed-Muller state is prepared (lilac). We assume the logical $|0\rangle$ and $|+\rangle$ states are only prepared (indicated by a triangle) when they are needed, taking d QEC cycles. A shuffle circuit (green), which lasts 14 QEC cycles interleaves the encoded T states with the qubits in the Reed-Muller state, where they undergo gate teleportation and measurement (blue) in two QEC cycles. The distilled encoded T state is in the top wire (orange).

3. 15-to-1 state 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 3D stack. The logical circuits are analyzed using the effective noise model in Sec. II D, which takes two parameters: the distance d of the 2D color codes used, and the strength p of the underlying circuit noise. Since we allow Clifford operations only between adjacent patches in the stack, we have to appropriately modify the state distillation circuit; see Fig. 22. In our analysis, we keep track only of errors up to order q^3 and $\overline{p}$, which are of similar order of magnitude in the regime of interest, where q is the infidelity of the input encoded T states, and $\overline{p}$ is the noise strength in the effective noise model. This splits the analysis into looking at error either in the encoded T states alone, or in the Clifford operations alone.

Noisy T states.—First, we briefly review the effect of noise on T states [50]. As described in Sec. I A, we simplify the form of noise assumed on a T state by twirling, which in this case corresponds to randomly applying the Clifford $XS^\dagger \propto |T\rangle\langle T| - |T^\perp\rangle\langle T^\perp|$ with probability 1/2. Recall that single-qubit Clifford gates can be done instantaneously and perfectly by frame tracking in the 2D color code as described in Sec. I C. This twirling forces the noisy T 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 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 if and only if E is a nontrivial 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

$$p^T_{\text{rej}} = 15q, \quad p^T_{\text{fail}} = 35q^3. \tag{16}$$

Noisy Clifford operations.—Now we consider the effect of noise in the Clifford operations [63,64] in the state-distillation circuit. First, we analyze the faults that occur during the Reed-Muller state preparation (lilac in Fig. 22), which takes 16 QEC 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

$$\begin{aligned} p^{\text{RM}}_{\text{rej}} &= 12.3\overline{p}_{\text{prep}} + 73\overline{p}_{\text{idle}} + 38.2\overline{p}_{\text{CNOT}}, \\ p^{\text{RM}}_{\text{fail}} &= 0.875\overline{p}_{\text{idle}} + 1.93\overline{p}_{\text{CNOT}}. \end{aligned} \tag{17}$$

Next we consider faults in the 16 idle QEC 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 simulate all 16 choices, and select the third qubit as the output since it has the lowest contribution from $\overline{p}_{\text{CNOT}}$; see Appendix C for further details. Any failure in any of these locations will result in an undetected failure

$$p^{\text{out}}_{\text{rej}} = 0, \quad p^{\text{out}}_{\text{fail}} = 16\overline{p}_{\text{idle}}. \tag{18}$$

By explicit calculation, we find the exact contribution to p_{rej} and p_{fail} of every Clifford fault location in Fig. 22 according to our effective noise model.

Lastly we analyze the remaining fault locations. These consist of the $15(d+16)$ idle locations involving qubits holding the T 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. 22. 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 only on one pair of qubits as shown, 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. 23. At this point, all other measurements are completed, and if the Pauli operator is 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 two-qubit circuit with its pure initial state and for the Pauli operator find the probability $p_{\text{flip}}(x_1, x_2, x_3, x_4)$ that the outcome is flipped, namely

$$p_{\text{flip}} = \begin{cases} 0 & \text{if } x_1 = x_2 \text{ and } z_2 = 0, \\ 1/2 & \text{if } x_1 \neq x_2 \text{ and } z_2 = 0, \\ 1 & \text{if } x_1 = x_2 \text{ and } z_2 = 1, \\ 1/2 & \text{if } x_1 \neq x_2 \text{ and } z_2 = 1. \end{cases} \tag{19}$$

Then all that remains is to count the contribution to each of these Paulis from the aforementioned locations according

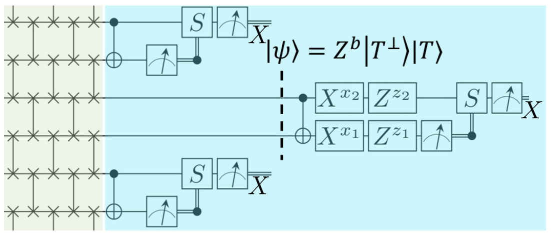


FIG. 23. A part of the state-distillation circuit with the inclusion of a Pauli error $X_1^{x_1}Z_1^{z_1}X_2^{x_2}Z_2^{z_2}$ on a pair of qubits. Such an error can appear as a result of previous faults being propagated through the circuit. Without loss of generality we assume that all the other measurements are 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 two-qubit circuit can then be analyzed for all 16 cases of the error $X_1^{x_1}Z_1^{z_1}X_2^{x_2}Z_2^{z_2}$.

to the effective noise model, which yields

$$p_{\rm rej}^{\rm rem} = (392 + 4.13d)\overline{p}_{\rm idle} + 13.5\overline{p}_{\rm CNOT}, \\ p_{\rm fail}^{\rm rem} = 0. \tag{20}$$

The contributions from Eqs. (16)–(18) and (20) combine to give the rejection and failure probability of the state-distillation step

$$p_{\rm rej}^{\rm dist} = 15q + 12.3\overline{p}_{\rm prep} + (466 + 4.13d)\overline{p}_{\rm idle} \\ + 51.7\overline{p}_{\rm CNOT}, \tag{21}$$

$$p_{\rm fail}^{\rm dist} = 35q^3 + 16.9\,\overline{p}_{\rm idle} + 1.93\overline{p}_{\rm CNOT}. \tag{22}$$

The number of time units required to implement the state-distillation circuit can be straightforwardly identified from Fig. 22 and is equal to

$$\tau^{\rm dist}(d) = 8\,(d + 32)\,. \tag{23}$$

B. State-distillation overhead

Here we estimate the overhead required for a k-round state-distillation protocol using distances $\{d^{(1)}, d^{(2)}, \ldots, d^{(k)}\}$ under circuit noise of strength p, 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 T state produced by the overall protocol.

Recall that the first step of the state-distillation protocol is to initialize encoded T states in distance-5 2D color codes; see Fig. 20 and Sec. III A 1. Writing the infidelity of the state after initialization as $q^{(0)} = p_{\rm fail}^{\rm init}(p)$, the remaining infidelities are then calculated iteratively according to

$$q^{(i)} = p_{\rm fail}^{\rm dist}(q^{(i-1)}, p, d^{(i)}). \tag{24}$$

To estimate the overhead, it is useful to streamline our notation. Recall from Eq. (12) that $N_{\rm 2D}(d) = (3d^2 - 1)/2$ qubits are used to implement the distance-d 2D color code. For each state distillation round $i \in \{1, 2, \ldots k\}$, let $r^{(i)} = \left[1 - p_{\rm rej}^{\rm dist}(p, q^{(i-1)}, d^{(i)})\right]$ 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 state-distillation circuit. To account for the initialization step, we also set $d^{(0)} = 3$, $r^{(0)} = \left[1 - p_{\rm rej}^{\rm init}(p)\right]$, $R^{(0)} = 1$, 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 T states, and thus we can talk about the average overhead [107]. Since the state-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 $\alpha^{(k)} N_{\rm 2D}(d^{(k)}) R^{(k)}/r^{(k)}$ 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 requires $\alpha^{(k)} N_{\rm 2D}(d^{(k)}) R^{(k-1)} R^{(k)}/(r^{(k-1)} r^{(k)})$ physical qubits, and so on. The qubit overhead $N_{\rm SD}$ of the state-distillation protocol can then be found as the number of qubits needed in the most qubit-expensive state-distillation round, namely

$$N_{\rm SD} = \max_{i=1,\ldots,k} \left[\alpha^{(i)} N_{\rm 2D}\left(d^{(i)}\right) \prod_{j=i}^{k} \frac{R^{(j)}}{r^{(j)}}\right]. \tag{25}$$

The time required for state distillation is

$$\tau_{\rm SD} = \tau^{\rm init} + \sum_{i=1}^{k} \tau^{\rm dist}(d^{(i)}). \tag{26}$$

The space-time overhead is then simply $N_{\rm SD}\tau_{\rm SD}$. For various values of p, we run a simple search over a number of rounds $k \in \{1, 2, 3\}$ and distances $\{d^{(1)}, d^{(2)}, \ldots, d^{(k)}\}$ to distill a target infidelity $p_{\rm fin}$ for a low space or space-time overhead; see Fig. 24.

Let us briefly remark on the threshold of this 15-to-1 state distillation using the 2D color code. This threshold is the noise strength above which arbitrarily low target infidelity cannot be achieved, irrespective of the number of state-distillation rounds. There are two main features

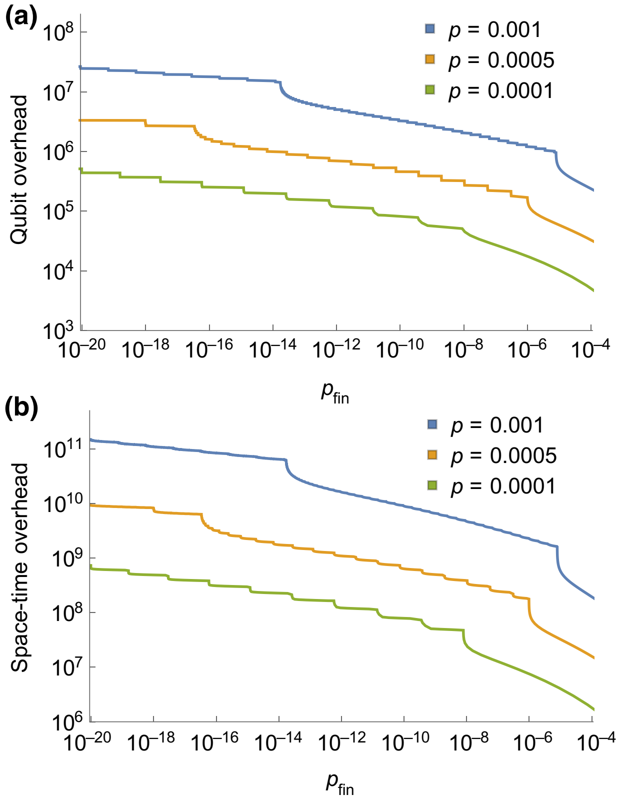


FIG. 24. (a) The qubit and (b) space-time overhead of state distillation as a function of the infidelity $p_{\rm fin}$ of the output T state.

that could limit the state-distillation threshold. Firstly, error correction in the 2D color code could be the limiting factor. Namely, if p is above the circuit-noise threshold of 0.37(1)%, one will be unable to achieve logical states with arbitrarily low infidelity. Secondly, the first round of state distillation can be a bottleneck if the infidelity of the initial encoded T state cannot be improved by it. For simplicity, we assume that the encoded Clifford operations execute perfectly, and solve for the critical infidelity q_c satisfying $35q_c^3 = q_c$. Using Eq. (15), we obtain a corresponding critical error rate of $1/(6.07 \times \sqrt{35}) \approx 3\%$. We conclude that error correction is the bottleneck, so we estimate the threshold for state distillation with this scheme to be 0.37(1)%.

IV. INSIGHTS INTO 3D COLOR CODES

In this section we present some insights into 3D color codes that are useful for code switching. In Sec. IV A, we describe a simple approach to switch between the 2D and 3D color codes. At the core of this approach is the fact that in a particular state of the gauge qubits, the 3D subsystem color code can be viewed as a collection of 2D color codes. In Sec. IV 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 transversal $\overline{T}$ gate. Our discussion closely follows material in Ref. [108]. In Sec. IV C, we generalize the restriction decoder to correct Z errors in the 3D color code with boundaries, which are later 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 code space 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 anticommutes 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. 25. The gauge and stabilizer groups $\mathcal{G}_{\mathrm{sub}}$ and $\mathcal{S}_{\mathrm{sub}}$ for the 3D subsystem color code, and the stabilizer group $\mathcal{S}_{\mathrm{3D}}$ for the 3D stabilizer color code are

$$\mathcal{G}_{\mathrm{sub}} = \langle X(e), Z(e) \mid \forall e \in \Delta_1'(\mathcal{L}_{\mathrm{3D}}) \rangle, \tag{27}$$

$$\mathcal{S}_{\mathrm{sub}} = \langle X(v), Z(v) \mid \forall v \in \Delta_0'(\mathcal{L}_{\mathrm{3D}}) \rangle, \tag{28}$$

$$\mathcal{S}_{\mathrm{3D}} = \langle X(v), Z(e) \mid \forall v \in \Delta_0'(\mathcal{L}_{\mathrm{3D}}), e \in \Delta_1'(\mathcal{L}_{\mathrm{3D}}) \rangle. \tag{29}$$

We can define the stabilizer group $\mathcal{S}_{\mathrm{2D}}$ of the 2D color code within the 3D lattice $\mathcal{L}_{\mathrm{3D}}$ since $\mathcal{L}_{\mathrm{2D}}$ is "contained" in $\mathcal{L}_{\mathrm{3D}}$; see Fig. 25(b). Let $\mathcal{S}_{\mathrm{int}}$ be the stabilizer group for the interior, i.e., the qubits that are not near the Y-boundary

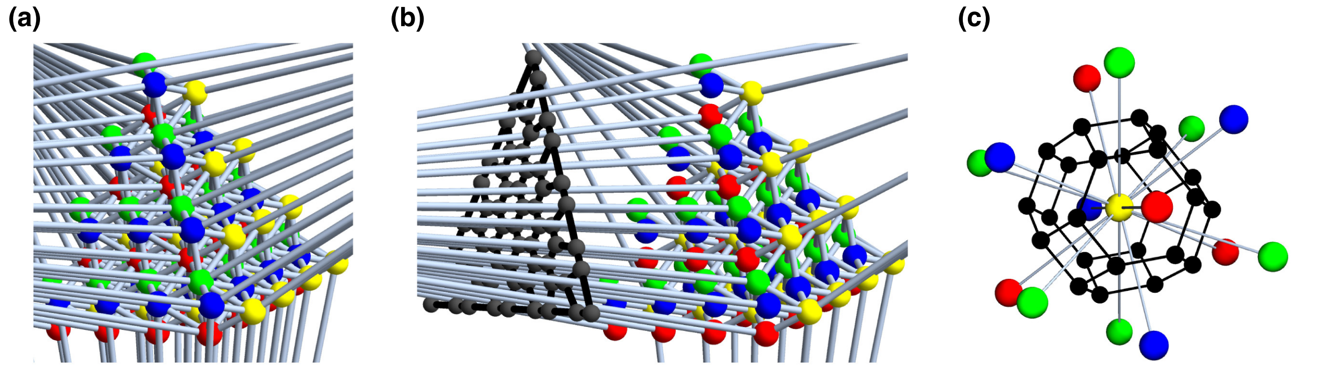


FIG. 25. (a) In one gauge fixing of the 3D subsystem color code $\mathcal{G}_{\mathrm{sub}}$ on the lattice $\mathcal{L}_{\mathrm{3D}}$ [shown in Fig. 5(b)], all Z edges (depicted by light struts) are satisfied. This corresponds to the 3D stabilizer color code $\mathcal{S}_{\mathrm{3D}}$. (b) In another gauge fixing of $\mathcal{G}_{\mathrm{sub}}$, all X and Z edges incident to Y vertices in $\Delta_0(\mathcal{L}_{\mathrm{3D}})$ (depicted by light struts) are satisfied. This corresponds to the 2D color code $\mathcal{S}_{\mathrm{2D}}$ on the qubits near the boundary vertex v_Y and the 2D spherical color codes $\mathcal{S}_{\mathrm{int}}$ on other qubits. We depict the primal lattice of the 2D color code on the lattice $\mathcal{L}_{\mathrm{2D}}$ [shown in Fig. 5(a)] in black. (c) Around each Y vertex in $\Delta_0'(\mathcal{L}_{\mathrm{3D}})$, there is the 2D spherical color code, whose primal lattice we depict in black.

vertex v_Y. Then,

$$\mathcal{S}_{\mathrm{2D}} = \langle X(e), Z(e) \mid \forall e \in \Delta_1'(\mathcal{L}_{\mathrm{3D}}) \text{ incident to } v_Y \rangle, \quad (30)$$

$$\mathcal{S}_{\mathrm{int}} = \langle X(e), Z(e) \mid \forall e \in \Delta_1'(\mathcal{L}_{\mathrm{3D}}) \text{ incident to any interior } Y \text{ vertex} \rangle, \quad (31)$$

where v_Y is the Y-boundary vertex. We can think of $\mathcal{S}_{\mathrm{int}}$ as the group generated by the stabilizers of the 2D spherical color codes centered around every Y interior vertex of $\mathcal{L}_{\mathrm{3D}}$; see Fig. 25(c). Note that every 2D spherical color code encodes zero logical qubits.

It is straightforward to see that $\mathcal{G}_{\mathrm{sub}}$ contains both $\mathcal{S}_{\mathrm{3D}}$ and $\langle \mathcal{S}_{\mathrm{2D}}, \mathcal{S}_{\mathrm{int}} \rangle$. Moreover, $\mathcal{S}_{\mathrm{sub}} \subseteq \mathcal{S}_{\mathrm{3D}}$ and $\mathcal{S}_{\mathrm{sub}} \subseteq \langle \mathcal{S}_{\mathrm{2D}}, \mathcal{S}_{\mathrm{int}} \rangle$ 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 $\mathcal{L}_{\mathrm{2D}}$). Therefore, to move from the subsystem code $\mathcal{G}_{\mathrm{sub}}$ to either of the stabilizer codes, i.e., $\mathcal{S}_{\mathrm{3D}}$ or $\langle \mathcal{S}_{\mathrm{2D}}, \mathcal{S}_{\mathrm{int}} \rangle$, 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 $\mathcal{S}_{\mathrm{3D}}$ or $\langle \mathcal{S}_{\mathrm{2D}}, \mathcal{S}_{\mathrm{int}} \rangle$, must also be a $(+1)$ eigenstate of every element in $\mathcal{S}_{\mathrm{sub}}$.

This example of gauge fixing is sometimes referred to as a *dimensional jump* because the logical information is moved between codes defined on 2D and 3D lattices [58].

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. Our discussion closely follows material in Ref. [108]. 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

$$\gamma = \sum_{K \in \mathcal{K}} \gamma^K. \quad (32)$$

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. 26. 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 $v \in \Delta_0'(\mathcal{L})$ of color K_1, 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.,

$$\left|\gamma^{K_1K_2}|_v + \gamma^{K_1K_3}|_v\right| \equiv 0 \mod 2. \quad (33)$$

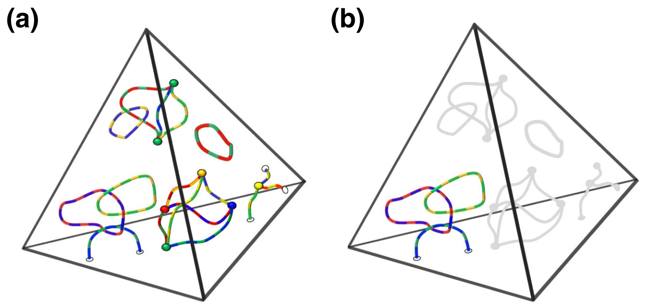


FIG. 26. Schematic representation of the Z-gauge flux γ in the bulk (enclosed by the black tetrahedron) of the 3D subsystem color code on the lattice $\mathcal{L}_{\mathrm{3D}}$ in Fig. 5(b). (a) The flux γ consists of strings of six different colors. There are seven branching points of γ (depicted as red, green, blue, and yellow vertices). The flux γ has to satisfy the Gauss law in the bulk, and can terminate at any of the boundary vertices of $\mathcal{L}_{\mathrm{3D}}$. (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 parity of the number of -1 measurement outcomes among those gauge generators in both cases is the same, we recover Eq. (33).

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 incident to 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, which anticommutes with precisely those Z gauge generators in γ.

If an edge set satisfies the Gauss law, we say that it is *valid*. An edge set that does not satisfy the Gauss law is said to be *invalid*, and we refer to all vertices that violate the Gauss law as *violation 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. 27.

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.,

$$\gamma = \sum_{j=1}^{a} \gamma_j. \quad (34)$$

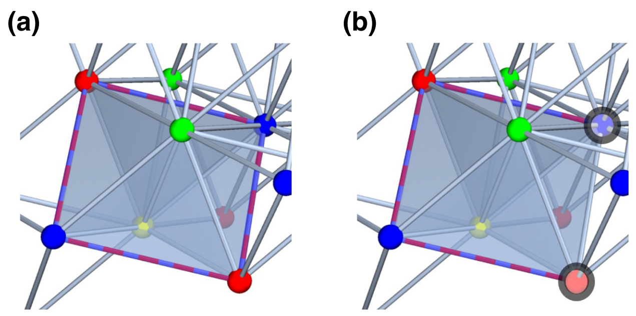


FIG. 27. (a) The flux γ (highlighted *RB* edges) due to a single *GY*-edge generator (shaded four tetrahedra) contains only *RB* edges. This set of edges γ satisfies the Gauss law. (b) A noisy measurement of the flux may not be valid, meaning it does not satisfy the Gauss law. Here $\widetilde{\gamma}$ has two violation points (highlighted *R* and *B* vertices) at which the Gauss law is not satisfied.

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. 26(b). Finally, we can decompose γ as a disjoint union of its linked components

$$\gamma = \sum_{i=1}^{b} \lambda_i. \tag{35}$$

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 γ 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 collection of excitation configurations* $\Sigma(\lambda_i)$ for λ_i with the neutral total charge as follows:

$$\Sigma(\lambda_i) = \left\{ \sigma \subseteq \Delta_0(\lambda_i) \,\middle|\, \sum_{v\in\sigma} \mathrm{col}(v) = (0,0,0) \right\}. \tag{36}$$

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*

$$\Sigma'(\lambda_i) = \Sigma(\gamma_{i_1}) \times \Sigma(\gamma_{i_2}) \times \cdots \times \Sigma(\gamma_{i_k}). \tag{37}$$

We remark that the linking charge is a charge that can be transferred between two connected components γ_i and γ_j, which are linked; see [108]. Note that $\Sigma'(\lambda_i)$ is contained in $\Sigma(\lambda_i)$, i.e., $\Sigma'(\lambda_i) \subseteq \Sigma(\lambda_i)$, and they coincide whenever the linked component λ_i consists of a single connected component.

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 from Ref. [31], which was originally defined for lattices with no boundary, to the tetrahedral lattice $\mathcal{L}_{3D}$. We also apply additional minor modifications to improve the performance. In what follows we briefly review the restriction decoder and describe our modifications.

Let $\sigma \subseteq \Delta_0'(\mathcal{L}_{3D})$ be the syndrome of the 3D stabilizer code on the tetrahedral lattice $\mathcal{L}_{3D}$. We pick one color, say Y, and for each color $K \in \{R, G, B\}$ we separately analyze the restricted syndrome $\sigma^{KY} \subseteq \Delta_0'(\mathcal{L}_{3D}^{KY})$ within the restricted lattice $\mathcal{L}_{3D}^{KY}$. 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 $E^{KY} \subseteq \Delta_1(\mathcal{L}_{3D}^{KY})$, which provides a pairing of vertices of σ^{KY} within the restricted lattice $\mathcal{L}_{3D}^{RY}$. Note that we can find a pairing of minimal weight by using the MWPM 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 from Ref. [31]. Since $\mathcal{L}_{2D}$ is a sublattice of $\mathcal{L}_{3D}$ [see Fig. 25(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 Sec. II 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 3D is analogous to an adaption presented in Ref. [37] for the 2D case.

In Fig. 28(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(5)% for independent and identically distributed (IID) phase-flip Z noise. This value can be contrasted with the restriction decoder threshold of 0.77% for the color code on the three-torus reported in Ref. [31]. Also, a similar adaption of the

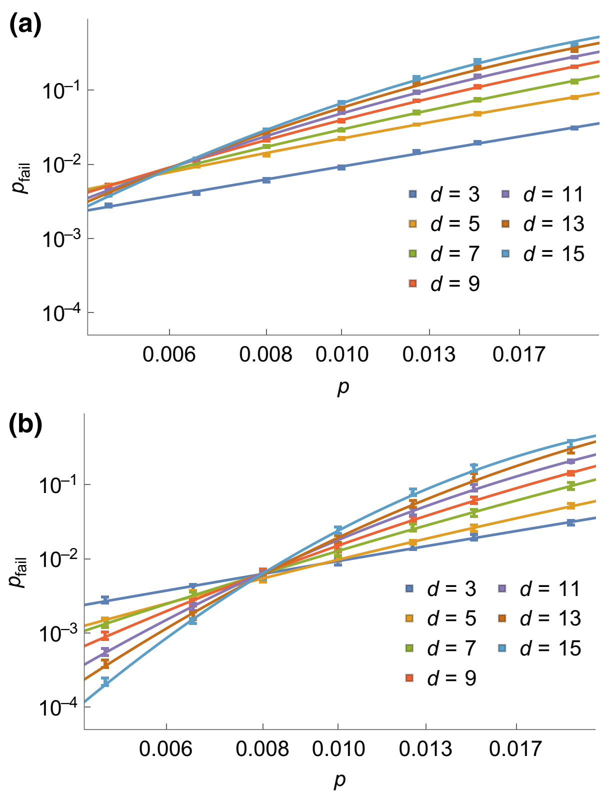


FIG. 28. Performance of the restriction decoder for the 3D color code adapted to the tetrahedral lattice $\mathcal{L}_{3D}$ with IID Z noise. Using the decoder (a) once for color Y, and (b) once each for R, G, B, and Y and then selecting the lowest weight output. This improves the performance and raises the observed threshold from 0.55(5)% to 0.80(5)%.

restriction decoder to the 3D color code with a boundary was recently presented in Ref. [109], and although the initial posting of their paper had low reported threshold values of 0.1%–0.2%, a later posting reported values consistent with those we find.

This adapted restriction decoder on moderate system sizes is far from optimal. For example, some weight-2 errors can cause failure for any $d \leq 9$. This phenomenon is not solely due to the presence of the boundaries. Namely, we find that there are some weight-2 errors in the color code of distance $d = 6$ on the three-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 3 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 that corrects all weight-2 errors is now $d = 7$ compared to $d = 11$, which is needed without the modification, and the threshold is increased from 0.55(5)% to 0.80(5)%; see Fig. 28.

V. CODE-SWITCHING 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 analyze the overhead of this process.

A. Creating the *T* state via code switching in six steps

Here we outline the protocol to produce an encoded T state using code switching in the presence of circuit noise. 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 transversally. Then, by measuring $\overline{X}$ for the 3D code, the encoded T state is effectively teleported to the remaining 2D code (up to a known logical Pauli correction). This approach avoids switching from the 3D code back to the 2D code, which we believe considerably reduces the amount of extra noise and simplifies our simulation. More explicitly, the protocol consists of the following steps (which are illustrated in Fig. 29).

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 near the yellow vertex of the 3D lattice as in Fig. 25(b).
2. *Prepare the 3D interior.*—The remaining qubits in $\mathcal{L}_{3D}$, i.e., those that 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 is a Bell pair between a 2D color code and the 3D subsystem color code $(|\overline{0}\rangle_{2D}|\overline{0}\rangle_{\text{sub}} + |\overline{1}\rangle_{2D}|\overline{1}\rangle_{\text{sub}})/\sqrt{2}$.
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 $(|\overline{0}\rangle_{2D}|\overline{0}\rangle_{3D} + |\overline{1}\rangle_{2D}|\overline{1}\rangle_{3D})/\sqrt{2}$.
5. *Apply $\widetilde{T}$, and measure.*—We apply $\widetilde{T}$ to every data qubit. This, in turn, implements a logical $\overline{T}$ gate, yielding the state $(|\overline{0}\rangle_{2D}|\overline{0}\rangle_{3D} +$

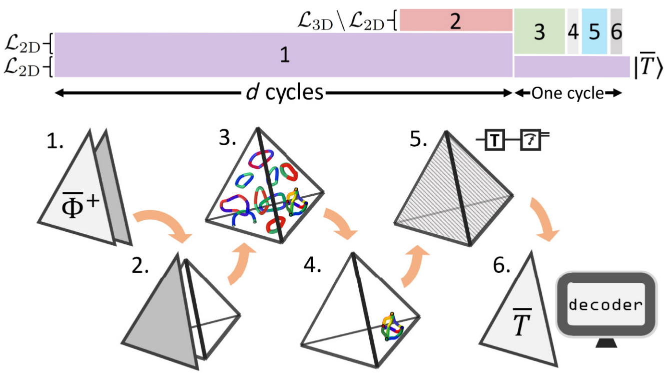


FIG. 29. Protocol to create an encoded T state via code switching, with the timeline of each step in units of QEC cycles for the 2D color code. In steps 1 and 2 we simultaneously prepare the Bell state $|\overline{\Phi}^+\rangle$ in a pair of 2D codes, and the interior of the 3D subsystem color code. 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 $\overline{Z}$.

$e^{i\pi/4}|\overline{1}\rangle_{2D}|\overline{1}\rangle_{3D})/\sqrt{2}$. 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 $(|\overline{0}\rangle_{2D} + me^{i\pi/4}|\overline{1}\rangle_{2D})/\sqrt{2}$, which for $m = -1$ needs to be fixed to the encoded 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.

(a) 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.
(b) 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.
(c) We choose the fault-tolerant error correction for the 2D color code to be the same optimized configuration we assume for state distillation (see Sec. II D) to allow for a fair comparison between code switching and state distillation.
(d) We assume a single ancilla qubit per gauge operator in the 3D color code interior.

In Fig. 30(a) we present our findings by showing the overall probability of failure of code switching using our simulations. In Figs. 30(b)–30(f) we indicate the impact of optimistic 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 $(|\overline{0}\rangle_{2D}|\overline{0}\rangle_{2D} + |\overline{1}\rangle_{2D}|\overline{1}\rangle_{2D})/\sqrt{2}$ 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 transversally 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 QEC cycles and fixing the inferred initial syndrome of the Z stabilizers. Preparation of $|\overline{0}\rangle_{2D}$ similarly involves initialization of all data qubits in $|0\rangle$, followed by d QEC cycles and fixing the inferred initial syndrome of the X stabilizers. Since each QEC cycle for the 2D color code requires 8 time units, the preparation of the encoded Bell state takes $8d$ time units. 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 Appendix 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. 25(b). Implementation details and optimizations of this step are given in Appendix D, 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 2D color code around a yellow vertex 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 randomly selected outcome is flipped, producing a valid syndrome, which is then fixed. The interior preparation step is started sufficiently early so that any repeated preparations have finished by the time the two 2D color-code patches complete. 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 a number of QEC cycles proportional to d.

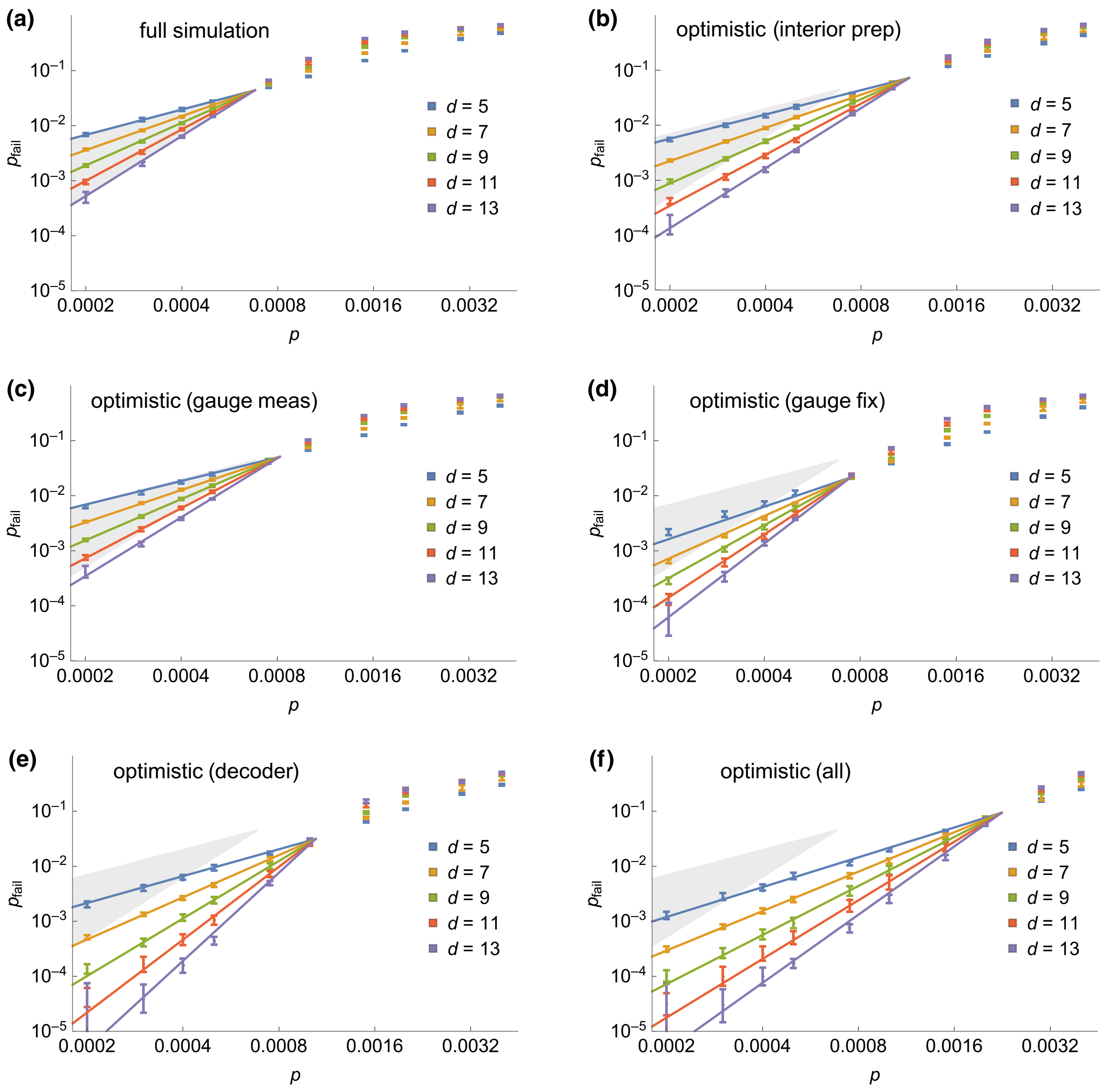


FIG. 30. (a) Failure probability of the code-switching protocol under circuit noise, for which we observe a T gate threshold of 0.07(1)%. We also estimate the code-switching failure probability when one step, i.e., (b) interior preparation, (c) gauge measurement, (d) gauge fixing, and (e) the decoder, is replaced with an optimistic version. In (f), all four of these steps are replaced by their optimistic versions. We use a special form of the ansatz in Eq. (7), i.e., $p_{\text{fail}} = A\left(p/p^*_{\text{CS}}\right)^{(Cd+D)}$ to fit the data up to the crossing. The gray region between the fit lines from (a) is superimposed onto (b)–(f) to guide the eye. We observe that improving the interior preparation and developing a better decoder have the most potential to improve the performance of code switching. If all the potential improvements are achievable, the threshold could be as large as 0.22(5)%.

Optimistic improvements. One may hope to improve the preparation of the 2D spherical color codes. Although we find 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 exploit the unused ancilla associated with the RG, RB, and GB edges and to consider collective rather than independent preparations of the 2D color codes. No matter how good such a preparation protocol is, it would require at least 3 time units as each qubit appears in three Z-type stabilizer generators. Therefore, we use 3 idle time units with circuit noise of strength p to bound potential improvements of this step in Fig. 30(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. Implementation details and optimizations of this step are given in Appendix 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 units, although the preparation can be done during the last time unit of the interior preparation step.

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

Optimistic 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 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 3 time units would be required to implement the measurement. Therefore, we use 3 idle time units with circuit noise of strength p in our optimistic simulation in Fig. 30(c).

4. Gauge fixing

This step corresponds to a classical algorithm which takes as its input the restricted noisy gauge flux $\tilde{\gamma}^{\mathcal{K}} \subseteq \Delta_1'(\mathcal{L}_{3D}^{\mathcal{K}})$ 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 $\tilde{\gamma}^{K}$ by matching its violation 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 violation points of $\tilde{\gamma}^{K}$. Then, $\hat{\gamma} = \sum_{K\in\mathcal{K}}(\tilde{\gamma}^K + \lambda^K)$ 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 efficiently found by, e.g., Gaussian elimination.

In this step, we are treating the restricted noisy gauge flux $\tilde{\gamma}^{\mathcal{K}}$ as arising from a gauge flux $\hat{\gamma}$ with no branching points along with some measurement errors; see Fig. 31. In other words, we want to find a gauge flux $\hat{\gamma}$ corresponding to some X-gauge operator just by looking at its restricted noisy gauge flux $\tilde{\gamma}^{\mathcal{K}}$. However, the gauge flux can in fact have branching points due to X errors in the system. In

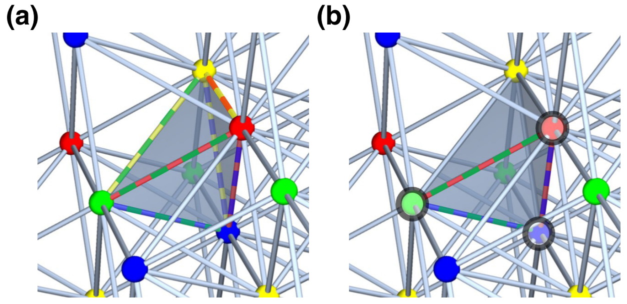


FIG. 31. (a) A Z-gauge flux due to an isolated X error (shaded tetrahedron) consists of six edges of different colors. (b) The restricted gauge flux, which is measured (assuming no errors occur during the measurement), excludes edges incident to Y vertices. The gauge-fixing algorithm in this scenario would connect the marked vertices in each subgraph 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.

such a case, the restricted noisy gauge flux does not even satisfy the Gauss law in the absence of measurement errors as we omit Z-type operators associated with RY, BY, and GY edges. For every $K \in \mathcal{K}$ we choose to independently pair up all the violation points of $\tilde{\gamma}^{K}$ within the restricted lattice $\mathcal{L}^{K}$, which in turn allows us to find the desired $\hat{\gamma}$.

Optimistic improvements. 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. 31. To bound the impact of 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. 30(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 1 time unit. Despite being one of the simplest code-switching steps to implement, it is quite challenging to simulate efficiently due to the large number of T gates. 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 $\rho_X \subseteq \Delta_3'(\mathcal{L}_{3D})$; similarly, let ρ_Z denote 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 residual X error ρ_X. This is made precise in the following lemma.

Lemma V.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 a decomposition of γ into its linked components λ_j's. If $\widetilde{T}$ is applied to every data qubit and the X stabilizers are perfectly measured, then one obtains a syndrome*

$$\sigma = \partial_{3,1}\rho_Z + \sum_{j=1}^{b}\sigma_j, \tag{38}$$

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

In the above, we use notation introduced in Sec. IV B. To simulate this step, one can generate Pauli operators resulting in the same distribution of syndromes. Note that there is a possibility of introducing a logical Z operator at this step, which we ignore in our simulation, leading to an underestimate of the failure of code switching. For more details and a justification of this lemma, see Ref. [108].

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

$$\tau_Z = \sum_{v\in\Delta_0'(\gamma)} \tau(\gamma|_v), \tag{39}$$

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.,

$$[\partial_{3,1}\tau(\gamma|_v)]|_v = \sum_{K\in\mathcal{K}'} E^{AK}. \tag{40}$$

We remark that in step (iii) one can always find an appropriate subset of tetrahedra $\tau(\gamma|_v)$. Namely, since γ satisfies the Gauss law, all $\gamma^{AK}|_v$'s have the same parity, and subsequently all $E^{AK}|_v$'s have the same parity. Thus, the endpoints of the $E^{AK}|_v$'s excluding the vertex v can be viewed as a valid syndrome of the 2D color code on a sphere around v, i.e., the sphere bounding the ball comprising the tetrahedra in $\partial_{0,3}v$. Decoding that syndrome configuration will give us some subset χ of triangular faces around v. Subsequently, we can choose $\tau(\gamma|_v)$ to be a subset of tetrahedra in $\partial_{0,3}v$, which are spanned by faces in χ and the vertex v.

We now explain why this algorithm produces $\tau_Z \subseteq \Delta_3'(\mathcal{L}_{3\mathrm{D}})$ 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. (40). 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 $(|\overline{0}\rangle_{2\mathrm{D}} + me^{i\pi/4}|\overline{1}\rangle_{2\mathrm{D}})/\sqrt{2}$, which is the encoded T state, up to the multiplication by $\overline{Z}$ for $m = -1$.

We implement a decoder based on the restriction decoder from Ref. [31], but modified in two ways. The first modification is to adapt the decoder to the 3D stabilizer color code on the lattice $\mathcal{L}_{3\mathrm{D}}$, which has boundaries. 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 Sec. IV C.

Optimistic improvements. A different decoder may 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 phase-flip Z noise is 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. Choose uniformly at random a real number from 0 to 1, and if it is smaller than $p^{(1)}_{\mathrm{3DCC}}\left((w/n)/p^{(1)}_{\mathrm{3DCC}}\right)^{(d+1)/2}$, then the correction fails; otherwise, the correction succeeds.

Note that $p^{(1)}_{\mathrm{3DCC}} \simeq 1.9\%$ is the known threshold of the optimal decoder for the 3D stabilizer color code under IID Z noise [102]. We provide more detailed justifications for this estimate in Appendix G. The impact of potential improvements of this step on code switching is shown in Fig. 30(e).

B. Code-switching overhead

To calculate the space and time overhead required to produce an encoded T state with failure probability p_{fin}, we first find the minimum distance $d(p_{\mathrm{fin}})$, which achieves this by extrapolating the data in Fig. 30(a). The time to implement the code-switching protocol at distance d is simply $8(d+1)$, 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

$$N_{\mathrm{CS}}(d) = (26d^3 + 51d^2 + 22d - 51)/24. \tag{41}$$

This is comprised of two 2D color-code patches [each of which has $N_{\mathrm{2D}}(d) = (3d^2-1)/2$ qubits from Eq. (12)] 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 Appendix A.

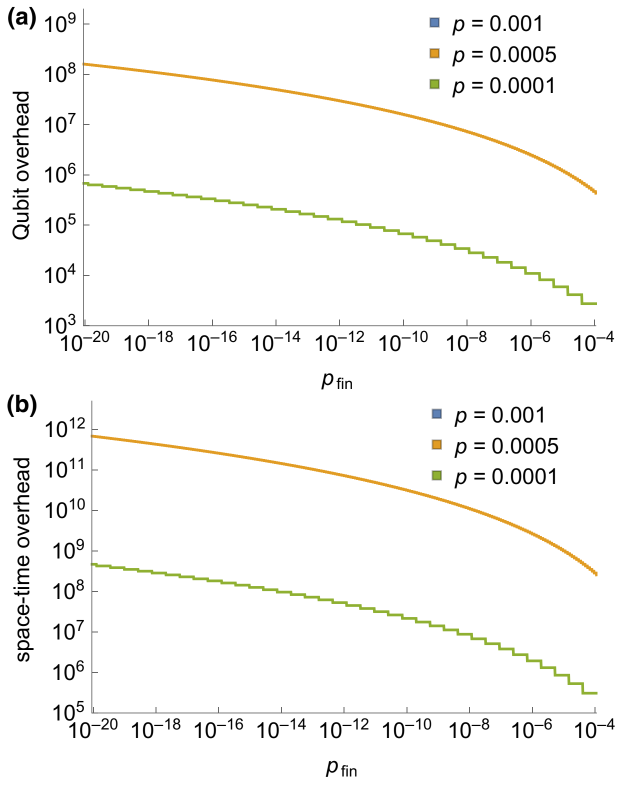


FIG. 32. (a) The qubit and (b) space-time overhead of code switching as a function of the infidelity p_{fin} of the output T state. Note there is no curve for $p = 0.001$ as this is higher than the observed threshold for code switching.

In Fig. 30 we fit a special form of the ansatz in Eq. (7), i.e., $p_{\mathrm{fail}} = A\left(p/p^*_{\mathrm{CS}}\right)^{(Cd+D)}$ to fit the data up to the crossing. Setting the failure probability to be at most the target infidelity, i.e., $p_{\mathrm{fail}} \le p_{\mathrm{fin}}$, and solving for d we obtain

$$d = \frac{\log(p_{\mathrm{fin}}/A)}{C\log\left(p/p^*_{\mathrm{CS}}\right)} - D/C, \tag{42}$$

where we round the right-hand side up to the closest odd integer. Finally, in Fig. 32 we substitute this into $N_{\mathrm{CS}}(d)$ and plot the qubit overhead and the space-time overhead as a function of p_{fin} for various values of p for two cases: assuming no improvements, and assuming all optimistic improvements can be achieved.

VI. DISCUSSION

In this work, we simulate concrete realizations of state distillation and code switching under circuit noise and compare the overhead that each requires to produce high-fidelity T states encoded in 2D color codes. We focus on regimes of practical interest, with physical error rates from 10^{-4} to 10^{-3} and target logical error rates from 10^{-20} to 10^{-4}; see Fig. 2. For error rates of 10^{-3} and above, our implementation of code switching is not viable since its observed threshold is 0.07(1)%, considerably lower than that of 0.37(1)% for state distillation. For error rates near 5×10^{-4}, our implementation of code switching becomes viable, but requires considerably more overhead than state distillation. However, for error rates around 10^{-4}, our code-switching implementation begins to slightly outperform that of state distillation. Lastly, we see in Fig. 2 that a highly optimistic implementation of code switching (which may be very difficult if not impossible to achieve) would not provide substantial savings over our simple implementation of state distillation in the studied regimes.

We now discuss the scaling of the space overhead O_*^{S} and space-time overhead O_*^{ST} of code switching and state distillation for small physical error rate, i.e., $p \ll 1$. We further assume that $\log p_{\mathrm{fin}}/\log p \gg 1$. As we derive in Appendix H, we have

$$O_*^{\mathrm{S}} \sim c_*^{\mathrm{S}} \left(\frac{\log p_{\mathrm{fin}}}{\log p}\right)^{\Gamma_*}, \quad O_*^{\mathrm{ST}} \sim c_*^{\mathrm{ST}} \left(\frac{\log p_{\mathrm{fin}}}{\log p}\right)^{\Gamma_*+1}, \tag{43}$$

where c_*^{S} and c_*^{ST} are some constants, $\Gamma_{\mathrm{CS}} = 3$ for code switching and $\Gamma_{\mathrm{SD}} = \max(2, \log_F R)$ for state distillation. Here, R is the ratio of the number of input to output magic states and F is the order of error suppression in a single distillation round. The change in the value of Γ_{SD} can be understood as follows—if $\log_F R > 2$, then the overhead is dominated by the initial distillation round; otherwise, it is dominated by the final round. For the 15-to-1 distillation scheme we have $R = 15$ and $F = 3$, leading to $\log_F R \approx 2.46$ and $\Gamma_{\mathrm{SD}} < \Gamma_{\mathrm{CS}}$. From Eq. (43) we thus conclude that for $\log p_{\mathrm{fin}}/\log p \gg 1$ state distillation will always outperform code switching. On the other hand, if we are willing to extrapolate Eq. (43) to more modest values of $\log p_{\mathrm{fin}}/\log p$, the overhead of code switching is predicted to drop below that of state distillation. The precise value of the crossover is highly sensitive to the implementation details, but this could explain why we observe code switching outperforming state distillation for low p and modest p_{fin} in Fig. 2.

Our findings point toward a number of future research directions. First, it could be fruitful to explore the potential of code switching in the regime of very low error rates, which may be achievable in trapped ion qubits [110] and topological qubits [38]. Second, improving the 2D color-code state-preparation procedure, syndrome extraction, and decoding algorithms would further reduce the overhead of both code switching and state distillation. We expect that code switching may also greatly benefit from better 3D color-code decoders. Third, our code-switching protocol produces the T state, but it is possible to directly apply the T gate rather than injecting it, which may allow one to implement code switching in constant time rather than a time that scales with the code distance d. It is, however, far from obvious whether there exists a regime where this would lead to an improvement. On the one hand, the time cost would be improved; on the other hand, the performance might heavily degrade due to replacing the destructive measurements and subsequent perfect-measurement decoding of the standard 3D color code with a more complex decoding problem. Fourth, we see that in contradiction with the commonly held belief that a distillation scheme overhead is dominated by that of the first round approximately $(\log 1/p_{\mathrm{fin}})^{\log_F R}$, the last round overhead is approximately $(\log 1/p_{\mathrm{fin}})^2$ when implemented with 2D codes, and dominates when $\log_F R < 2$; similar observations were also made in Refs. [14,111]. This suggests that searches and optimizations of new distillation schemes should not be restricted to those with very low $\log_F R$. Lastly, it is interesting to consider scenarios in which our main conclusions may not hold. For instance, if the physical error rate continues to drop significantly, it seems possible that the cost of code switching would continue to drop relative to distillation. Also, in a setting of qubit abundance, the somewhat lower time cost of code switching could make it favorable compared to distillation.

ACKNOWLEDGMENTS

M.E.B. acknowledges relevant scientific discussions with many researchers from the QEC community, including Thomas Bohdanowicz, Héctor Bombín Ben Brown, Earl Campbell, Nicolas Delfosse, Austin Fowler, Jeongwan Haah, Tomas Jochym O'Connor, Cody Jones, Andrew Landahl, Ying Li, Michael Newman, Adam Paetznick, John Preskill, and Marcus P da Silva. He is especially grateful for the encouragement for this project from David Poulin. M.E.B. also wishes to think Achiamar Lee Rivera for her unwavering support throughout this research. 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.

APPENDIX A: LATTICE PARAMETERS AND SPECIFICATIONS

Here we provide parameters of the lattices that are used throughout the paper. Tables II and III show parameters of the direct lattice and the dual lattice used to define the 2D and 3D color codes of distance d. In general, these lattices can be obtained by tessellating a D sphere with D-simplices and then removing one vertex from that tessellation, as well as all the simplices containing that vertex [57,83].

TABLE II. The lattice $\mathcal{L}_{2D}$, which we use to define the 2D stabilizer color code of distance d, consists of $3(d^2+7)/8$ vertices, $3(3d^2+5)/8$ edges, and $(1+3d^2)/4$ faces. We group vertices of $\mathcal{L}_{2D}$ into three different categories depending on the number of faces incident to them. Note that qubits are placed on faces, whereas stabilizers are identified with vertices of $\mathcal{L}_{2D}$.

No. of vertices	No. of incident faces
$3(d-1)/2$	4
$3(d-3)(d-1)/8$	6
3	d

APPENDIX B: SUPPLEMENTARY DETAILS FOR 2D COLOR-CODE SIMULATIONS

Here we provide additional 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 Sec. II in the main text. In Fig. 33 we show the performance under phenomenological noise, which we use to produce the long-time pseudothreshold plot in Fig. 12(d) in the main text. Similarly, in Fig. 34

TABLE III. The lattice $\mathcal{L}_{3D}$, which we use to define the 3D subsystem and stabilizer color codes of distance d, consists of $(d-1)(d+1)(d+3)/12+4$ vertices, $(d-1)(7d^2+10d+15)/12+6$ edges, d^3+d+2 faces, and $(d^3+d)/2$ tetrahedra. We group vertices and edges of $\mathcal{L}_{3D}$ into, respectively, five and three different categories depending on the number of tetrahedra incident to them. Note that qubits are placed on tetrahedra, whereas stabilizer and gauge generators are identified with vertices and edges of $\mathcal{L}_{3D}$.

No. of vertices	No. of incident tetrahedra	No. of edges	No. of incident tetrahedra
4	$\frac{1+3d^2}{4}$	6	d
$2(d-1)$	8	$\frac{d^3+3d^2+11d-15}{4}$	4
$\frac{(d-3)(d-1)}{2}$	12	$\frac{(d-3)(d-1)(2d+5)}{6}$	6
$\frac{(d-3)(d-1)}{2}$	18		
$\frac{(d-5)(d-3)(d-1)}{12}$	24		

we show the performance under phenomenological noise, which we use to produce the long-time pseudothreshold in Fig. 14(b) in the main text. We point out that the decay

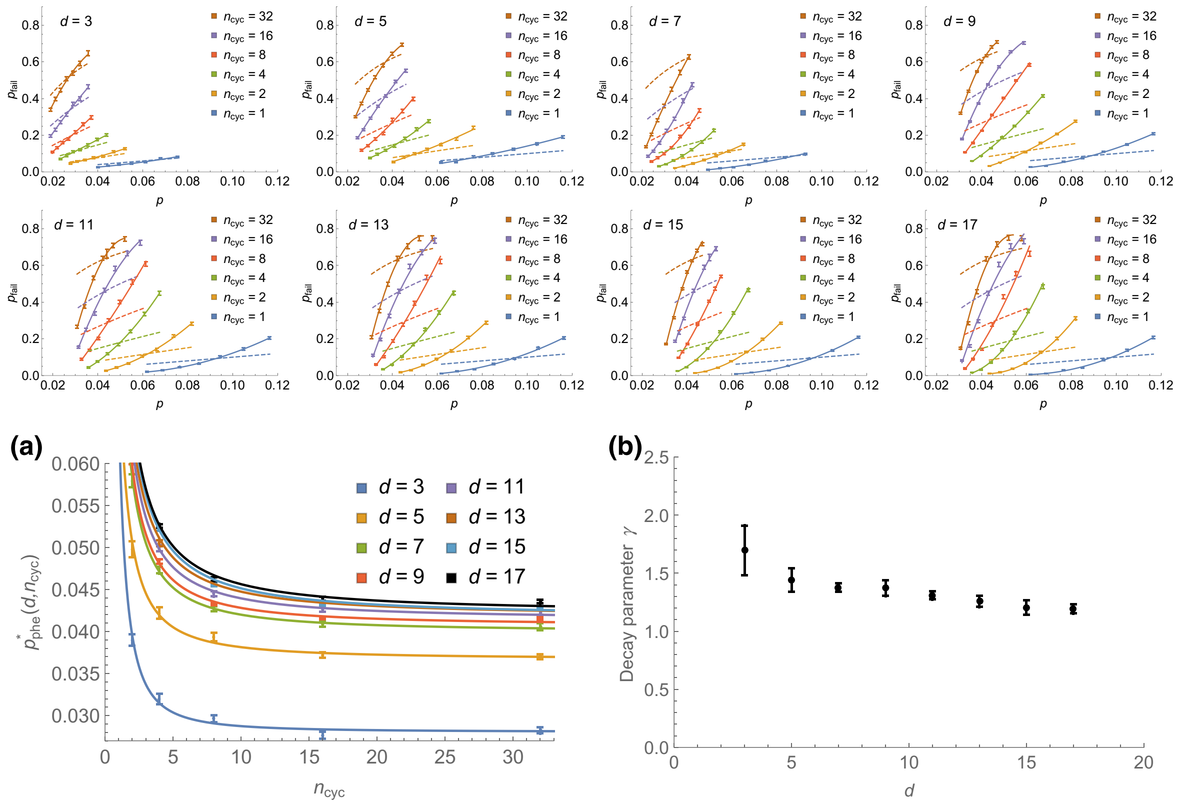
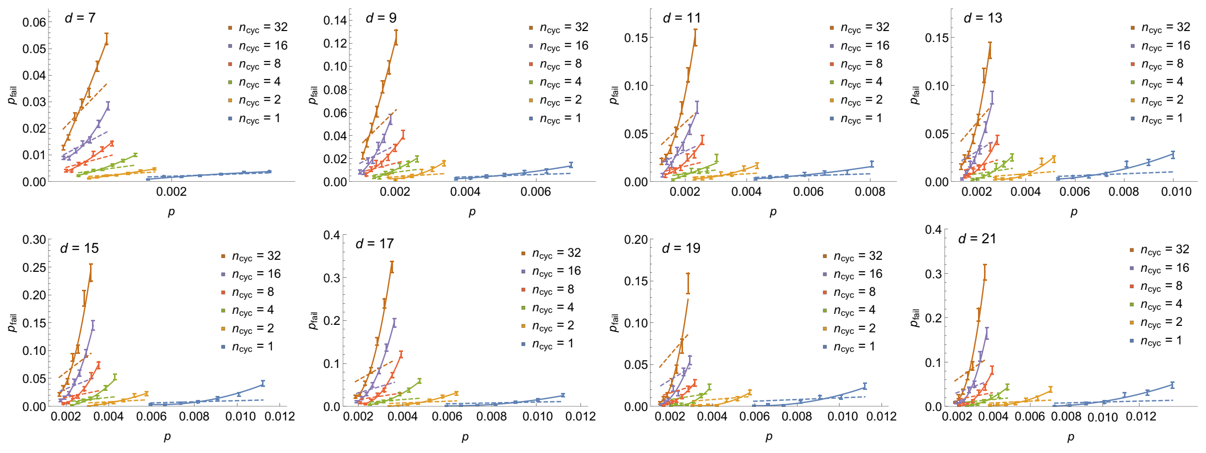


FIG. 33. (Top two rows) Failure probability of n_{cyc} QEC cycles for phenomenological noise of strength p given a perfect initial state and final QEC cycle, for various distances d. We estimate the time-dependent pseudothreshold $p^*_{phe}(d, n_{cyc})$ as an intersection of quadratic fits (solid lines) with the corresponding physical qubit error probability $p_{phy}(n_{cyc})$ as defined in Eq. (5) in the main text (dashed curves). (a) The time-dependent pseudothreshold $p^*_{phe}(d, n_{cyc})$ as a function of n_{cyc}. We estimate the long-time pseudothresholds $p^*_{phe}(d)$ by fitting $p^*_{phe}(d, n_{cyc})$ with the ansatz in Eq. (11) in the main text 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 that is independent of system size.



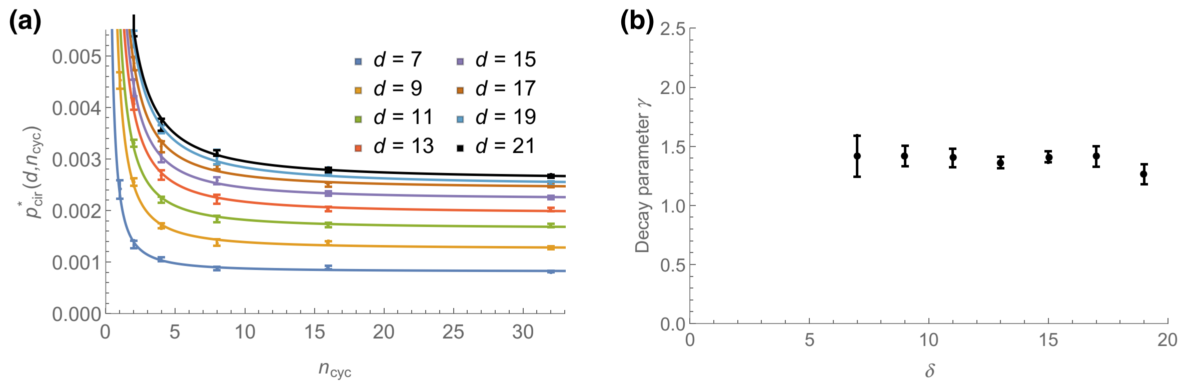


FIG. 34. (Top two rows) Failure probability of n_{cyc} QEC cycles for circuit noise of strength p given a perfect initial state and final QEC cycle, for various distances d. We estimate the time-dependent pseudothreshold $p^*_{\mathrm{cir}}(d, n_{\mathrm{cyc}})$ as an intersection of quadratic fits (solid lines) with the corresponding physical qubit error probability $p_{\mathrm{phy}}(n_{\mathrm{cyc}})$ as defined in Eq. (5) in the main text (dashed curves). (a) The time-dependent pseudothreshold $p^*_{\mathrm{cir}}(d, n_{\mathrm{cyc}})$ as a function of n_{cyc}. We estimate the long-time pseudothresholds $p^*_{\mathrm{cir}}(d)$ by fitting $p^*_{\mathrm{cir}}(d, n_{\mathrm{cyc}})$ with the ansatz in Eq. (11) in the main text 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.

parameter in the fit of the time-dependent pseudothreshold in Figs. 33(b) and 34(b) shows that convergence to the long-time pseudothreshold seems not to depend significantly on the system size. In Fig. 35 we provide the data used to produce the crossings in Fig. 14(b) in the main text. In Fig. 36 we show the impact of optimizing the stabilizer extraction circuit on the performance of error correction under circuit noise by comparing the performance of the best and worst ranked schedues from Fig. 14(a) in the main text. In Fig. 37, we show additional data from which we extract the logical failure rates for $p = 0.001, 0.0001$ in Fig. 17(b) in the main text.

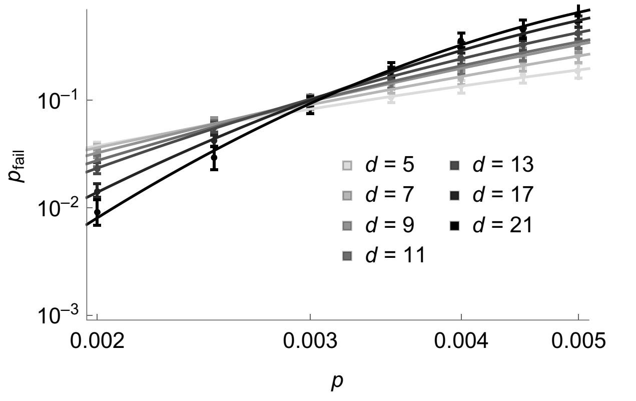


FIG. 35. Failure probability p_{fail} of the noisy-syndrome projection decoder for $n_{\mathrm{cyc}} = d$ rounds of circuit noise and various distances d.

APPENDIX C: SUPPLEMENTARY DETAILS FOR DISTILLATION ANALYSIS

Here we provide more details of the distillation analysis in Sec. III in the main text. In Table IV we consider variants of the distillation circuit formed by puncturing different qubits of the quantum Reed-Muller state. This helps us identify that puncturing qubit number three is best with the noise model we assume in this paper since it results in the smallest leading-order failure probability given that $\overline{p}_{\mathrm{CNOT}} \geq \overline{p}_{\mathrm{prep}} \geq \overline{p}_{\mathrm{idle}}$.

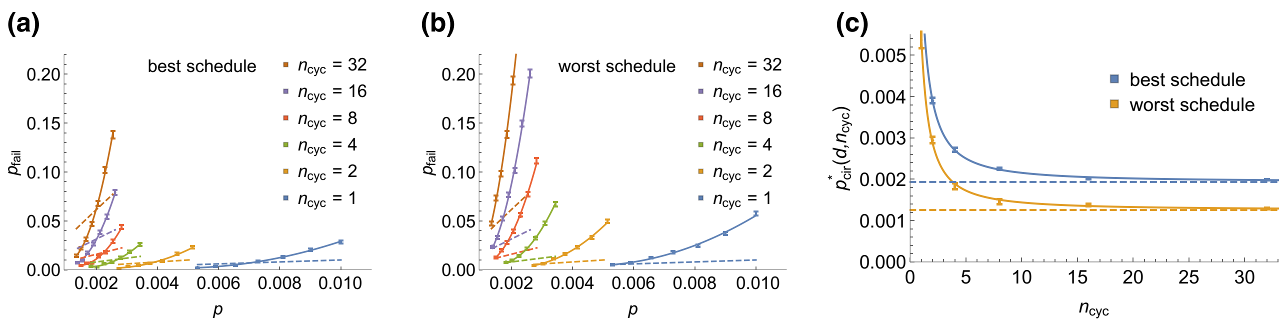


FIG. 36. Performance comparison of (a) the best ranked, and (b) the worst ranked length-7 CNOT schedules, namely $\{4,1,2,3,6,5;3,2,5,6,7,4\}$ and $\{4,1,2,7,6,3;1,6,7,4,5,2\}$. We numerically estimate the failure probability of n_{cyc} QEC cycles for circuit noise of strength p for $d=13$ given a perfect initial state and final QEC cycle. 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). We estimate the pseudothreshold $p^*_{\text{cir}}(d,n_{\text{cyc}})$, which we plot in subfigure (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 pseudothreshold for the good circuit is 0.193(2)% (blue, dashed), considerably larger than that of 0.126(3)% for the bad circuit (yellow, dashed).

Now we elaborate on the procedure to initialize a T state in a distance d color code described in Sec. III A 1 in the main text, and explain some structure that we exploit to simplify the noise analysis. First note that even in the absence of noise, the stabilizers, which are measured in the initialization protocol, have uncertain outcomes because the state is not a code state of the 2D color code at the start of the protocol. Let A and B be the two sets of qubits prepared in $|0\rangle$ and $|+\rangle$, respectively; see Fig. 21(a). The protocol involves taking the observed syndrome σ, and producing a Pauli operator $P(\sigma)=P^XP^Z$, where P^X and P^Z are Pauli X and Z operators supported on qubits in B and A, respectively. The protocol rejects if no such fix exists or if the syndromes from the consecutive rounds of QEC differ. Note that the fix is unique up to a stabilizer since neither of the disjoint sets A and B support a logical operator. This implies that for any two syndromes σ and σ' the Pauli operators $P(\sigma)P(\sigma')$ and $P(\sigma+\sigma')$ are equivalent up to a stabilizer.

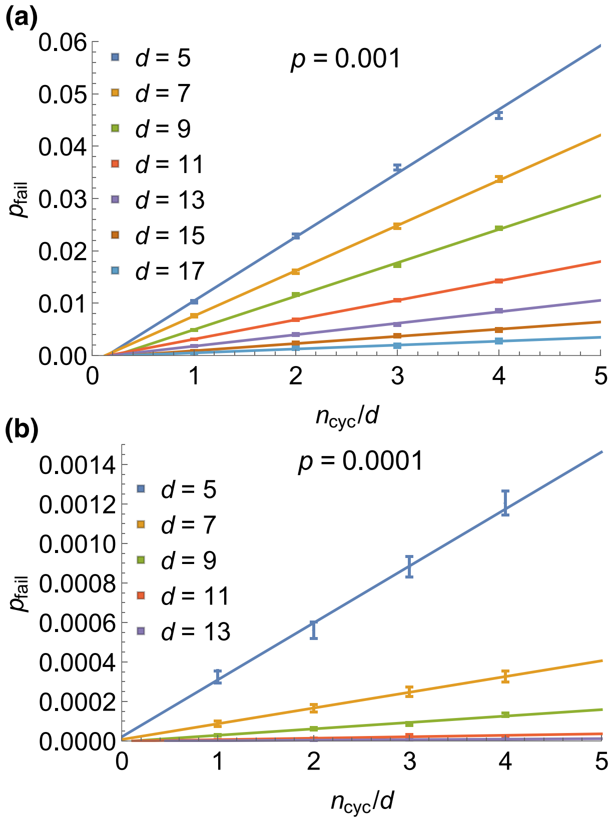


FIG. 37. Failure probability of the logical idle operation using the best-performing CNOT schedule as a function of n_{cyc}/d for (a) $p=10^{-3}$ and (b) $p=10^{-4}$. We use the ansatz in Eq. (13) in the main text to find the logical failure rate $\overline{p}(p,d)$. We observe that QEC equilibration occurs by the dth QEC cycle.

To faithfully simulate this protocol in the presence of noise, one can consider the scenario in which a complete set of perfect stabilizer measurements is performed (and their outcomes discarded) before the QEC circuits are applied. This allows us to begin the simulation with a state, which is an eigenstate of the stabilizer group. The simulation can therefore be implemented as follows.

1. Start with a perfect code state.
2. With some probability $\Pr(\sigma)$, apply a Pauli operator $P(\sigma)=P^X(\sigma)P^Z(\sigma)$, where $P^X(\sigma)$ and $P^Z(\sigma)$ are Pauli X and Z operators supported within B and A, respectively.
3. Run the noisy circuits explicitly as described in Sec. I A in the main text. A set of faults will propagate to some Pauli error E on the data qubits at the end of the protocol. The faults could result in the syndromes from the two QEC cycles disagreeing, in which case the protocol is rejected, or both QEC rounds having the same observed syndrome $\widetilde{\sigma}$, possibly different from the syndrome σ of P. Note that

TABLE IV. Leading-order contributions to the probability of failure $p_{\text{fail}}^{\text{RM}}$ and rejection $p_{\text{rej}}^{\text{RM}}$ of the 15-to-1 distillation scheme from the production of the Reed-Muller state for each choice of punctured qubit.

Punctured qubit	Failure probability $p_{\text{fail}}^{\text{RM}}$	Rejection probability $p_{\text{rej}}^{\text{RM}}$
1	$1.98\overline{p}_{\text{CNOT}} + 0.875\overline{p}_{\text{idle}} + \overline{p}_{\text{prep}}$	$38\overline{p}_{\text{CNOT}} + 75.6\overline{p}_{\text{idle}} + 11.3\overline{p}_{\text{prep}}$
2	$2.2\overline{p}_{\text{CNOT}} + 0.85\overline{p}_{\text{idle}}$	$38.2\overline{p}_{\text{CNOT}} + 73.3\overline{p}_{\text{idle}} + 12.3\overline{p}_{\text{prep}}$
3	$1.93\overline{p}_{\text{CNOT}} + 0.875\overline{p}_{\text{idle}}$	$38.2\overline{p}_{\text{CNOT}} + 73\overline{p}_{\text{idle}} + 12.3\overline{p}_{\text{prep}}$
4	$2.45\overline{p}_{\text{CNOT}} + 2.31\overline{p}_{\text{idle}} + \overline{p}_{\text{prep}}$	$38.1\overline{p}_{\text{CNOT}} + 77.1\overline{p}_{\text{idle}} + 11.8\overline{p}_{\text{prep}}$
5	$1.98\overline{p}_{\text{CNOT}} + 0.875\overline{p}_{\text{idle}}$	$38\overline{p}_{\text{CNOT}} + 73\overline{p}_{\text{idle}} + 12.3\overline{p}_{\text{prep}}$
6	$2.2\overline{p}_{\text{CNOT}} + 0.85\overline{p}_{\text{idle}} + 0.5\overline{p}_{\text{prep}}$	$38.2\overline{p}_{\text{CNOT}} + 75.5\overline{p}_{\text{idle}} + 12\overline{p}_{\text{prep}}$
7	$1.93\overline{p}_{\text{CNOT}} + 0.875\overline{p}_{\text{idle}} + 0.5\overline{p}_{\text{prep}}$	$38.2\overline{p}_{\text{CNOT}} + 75.3\overline{p}_{\text{idle}} + 12\overline{p}_{\text{prep}}$
8	$2.45\overline{p}_{\text{CNOT}} + 3.69\overline{p}_{\text{idle}} + 2\overline{p}_{\text{prep}}$	$38.1\overline{p}_{\text{CNOT}} + 76.4\overline{p}_{\text{idle}} + 11.3\overline{p}_{\text{prep}}$
9	$1.98\overline{p}_{\text{CNOT}} + 0.875\overline{p}_{\text{idle}} + \overline{p}_{\text{prep}}$	$38\overline{p}_{\text{CNOT}} + 74.8\overline{p}_{\text{idle}} + 11.3\overline{p}_{\text{prep}}$
10	$2.2\overline{p}_{\text{CNOT}} + 0.85\overline{p}_{\text{idle}}$	$38.2\overline{p}_{\text{CNOT}} + 73.3\overline{p}_{\text{idle}} + 12.3\overline{p}_{\text{prep}}$
11	$1.93\overline{p}_{\text{CNOT}} + 0.875\overline{p}_{\text{idle}}$	$38.2\overline{p}_{\text{CNOT}} + 73\overline{p}_{\text{idle}} + 12.3\overline{p}_{\text{prep}}$
12	$2.45\overline{p}_{\text{CNOT}} + 2.31\overline{p}_{\text{idle}} + \overline{p}_{\text{prep}}$	$38.1\overline{p}_{\text{CNOT}} + 77.1\overline{p}_{\text{idle}} + 11.8\overline{p}_{\text{prep}}$
13	$1.98\overline{p}_{\text{CNOT}} + 0.875\overline{p}_{\text{idle}}$	$38\overline{p}_{\text{CNOT}} + 73\overline{p}_{\text{idle}} + 12.3\overline{p}_{\text{prep}}$
14	$2.2\overline{p}_{\text{CNOT}} + 0.85\overline{p}_{\text{idle}} + 0.5\overline{p}_{\text{prep}}$	$38.2\overline{p}_{\text{CNOT}} + 75.9\overline{p}_{\text{idle}} + 12\overline{p}_{\text{prep}}$
15	$1.93\overline{p}_{\text{CNOT}} + 0.875\overline{p}_{\text{idle}} + 0.5\overline{p}_{\text{prep}}$	$38.2\overline{p}_{\text{CNOT}} + 75.5\overline{p}_{\text{idle}} + 12\overline{p}_{\text{prep}}$
16	$2.45\overline{p}_{\text{CNOT}} + 2.59\overline{p}_{\text{idle}} + 2\overline{p}_{\text{prep}}$	$38.1\overline{p}_{\text{CNOT}} + 77\overline{p}_{\text{idle}} + 11.3\overline{p}_{\text{prep}}$

$\sigma + \widetilde{\sigma}$ can only depend on the set of faults, and is independent of σ.

4. Apply a Pauli operator $P(\widetilde{\sigma}) = P^X(\widetilde{\sigma})P^Z(\widetilde{\sigma})$ with the syndrome $\widetilde{\sigma}$, where $P^X(\widetilde{\sigma})$ and $P^Z(\widetilde{\sigma})$ are Pauli X and Z operators supported on qubits in B and A, or reject if such an operator cannot be found.
5. If the net operator $EP(\sigma)P(\widetilde{\sigma})$ is correctable, then we say that the protocol succeeds, and otherwise we say it fails.

However, note that the net operator $EP(\sigma)P(\widetilde{\sigma})$ must be independent of σ, since $P(\widetilde{\sigma})$ and $P(\sigma)P(\sigma + \widetilde{\sigma})$ are equivalent up to a stabilizer. In our simulation we therefore set σ to be trivial for simplicity without loss of generality.

Using this simulation approach, we analyze all 234 valid CNOT schedules for a variety of distances, and find that $\{1, 4, 6, 7, 5, 2; 4, 5, 7, 6, 2, 3\}$, as shown in Fig. 21, has the lowest failure pro ability, namely $p_{\text{fail}}^{\text{init}} = 6.07p$. For comparison, the worst schedule using $d = 3$ is $\{1, 4, 6, 7, 5, 2; 4, 5, 7, 6, 2, 3\}$ and results in a $p_{\text{fail}}^{\text{init}}$ of $12.7p$.

APPENDIX D: PREPARING THE 3D INTERIOR FOR CODE SWITCHING

Here we consider the 3D interior of the initial state for code switching discussed in Sec. V A 2 in the main text, which consists of a (trivial) 2D color-code state on the qubits surrounding each Y interior vertex. This object can be viewed as a 2D sphere; see Fig. 38. 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 in a single QEC cycle, followed by

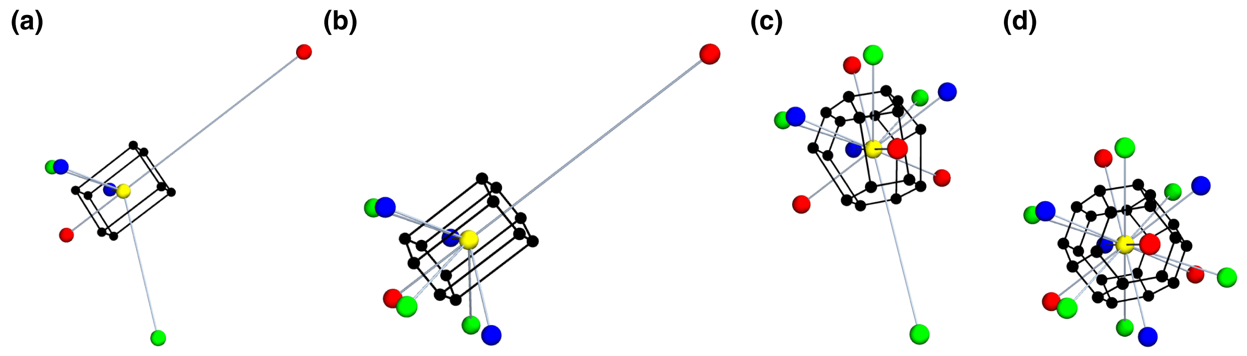


FIG. 38. Dual and primal lattices for the 2D spherical color codes, which appear in the preparation of interior of the 3D color code. Qubits are black dots, and stabilizers are gray edges with colored vertices. For each, the subset of vertices from v_{list} in Eq. (D1) are (a) $\{1, 2, 3, 4, 8, 15, 17\}$, (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) $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16\}$.

TABLE V. Shortest CNOT sequences to measure each Z stabilizer with a single ancilla for the 8-, 12-, 18-, and 24-qubit spherical color codes in Fig. 38. 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.

Sphere size	Edge	Time unit 1	Time unit 2	Time unit 3	Time unit 4	Time unit 5	Time unit 6
	(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	(8,3)	(3,8,15,17)	(2,3,8,15)	(1,2,3,8)	(1,3,8,17)		
qubits	(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)		
	(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)
12	(8,4)			(1,2,4,8)	(2,4,7,8)	(1,4,8,17)	(4,7,8,17)
qubits	(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)
	(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)
18	(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)
qubits	(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)
	(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)
24	(8,7)			(2,4,7,8)	(2,7,8,11)	(4,7,8,13)	(7,8,11,13)
qubits	(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)

the application of an appropriate X Pauli to fix incorrect stabilizers.

CNOT *circuits.*—For each of the four types of spheres, we find an ordering of CNOTs, which allows for a minimal-length circuit to measure the stabilizers; see Table V. These sequences are 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 than one gate per time unit) minus the average time unit when each gate occurs. The greedy algorithm terminates when all constraints are satisfied and when the cost function cannot be further reduced. We fix the total number of time units to be minimal (four CNOT time units for the eight-qubit sphere, which involves only weight-4 stabilizers, and six CNOT time units for the 12-, 18-, and 24-qubit spheres, 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 find are optimal with regard to this cost function: all constraints are 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 that are optimal in this sense, but we did not attempt to explore among those for schedules that performed better.

To make the representation of the schedules slightly more compact, we specify 17 vertex locations in a list v_{list} in lexicographical order as follows:

$$v_{\text{list}} = \Big\{ (-1,\ 0,\ 0), \left(-\tfrac{1}{2}, -\tfrac{1}{2}, -\tfrac{1}{2}\right), \left(-\tfrac{1}{2}, -\tfrac{1}{2}, \tfrac{1}{2}\right), \left(-\tfrac{1}{2}, \tfrac{1}{2}, -\tfrac{1}{2}\right), \left(-\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2}\right), (0, -1,\ 0), (0,\ 0, -1), (0,\ 0,\ 0), (0,\ 0,\ 1), (0,\ 1,\ 0), \left(\tfrac{1}{2}, -\tfrac{1}{2}, -\tfrac{1}{2}\right), \left(\tfrac{1}{2}, -\tfrac{1}{2}, \tfrac{1}{2}\right), \left(\tfrac{1}{2}, \tfrac{1}{2}, -\tfrac{1}{2}\right), \left(\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2}\right), (1, -1, -1), (1,\ 0,\ 0), \left(\tfrac{3}{2}, \tfrac{3}{2}, \tfrac{3}{2}\right) \Big\}, \tag{D1}$$

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, 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 V, 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 coordinates in v_{list}, but can be identified and shifted to the standard coordinate 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 random outcomes) determines the X-type gauge 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 render the syndrome invalid, in the sense that there is no gauge operator with the observed syndrome. 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 nine code in Fig. 39. 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

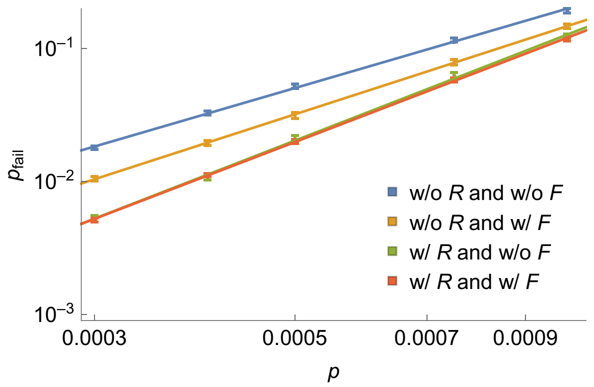


FIG. 39. Overall code-switching performance of four syndrome fixing approaches for the preparation of 2D spherical color codes. Each approach either uses or does not use the repeat (R) and flip (F) strategies for distance $d = 9$. In the approach, which uses both strategies, we use the repeat strategy, 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. Both the repeat and flip strategies improve performance.

be applied on spheres that failed the first round. 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.

APPENDIX E: CHOICES OF BASIS FOR CODE-SWITCHING PREPARATION

Here we analyze the effect of the choice of basis in the first two steps of the code switching described in Secs. V A 1 and V A 2 in the main text. 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 Sec. V A 1 in the main text. We fault tolerantly prepare the first patch in $|\overline{+}\rangle$, and the second in $|\overline{0}\rangle$ before applying a transversal 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 2D *patch* $|+\rangle$ and 2D *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 Sec. V A 2 in the main text. We can do this by either preparing each data qubit in $|0\rangle$ and then measuring the X stabilizers, or by preparing each data qubit in $|+\rangle$ and then measuring the Z stabilizes. We refer to these cases as 3D *interior* $|0\rangle$ and 3D *interior*

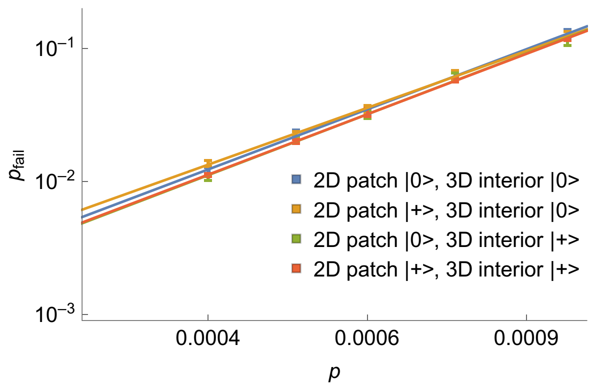


FIG. 40. The impact on the failure probability of the code-switching protocol from different choices of basis for first two steps of the protocol.

$|+\rangle$, respectively, and compare their impact along with the choice for the patch preparation in Fig. 40. The data presented there is for distance $d = 9$, and has all other steps as described in Sec. V in the main text for the optimized code-switching protocol. We find that there is very little difference in the performance of each of these choices, and we select 2D *patch* $|+\rangle$ and 3D *interior* $|+\rangle$ for use in the optimized version of the protocol.

APPENDIX F: GAUGE MEASUREMENT CIRCUITS FOR CODE SWITCHING

Here we discuss the circuits used to measure the Z-type gauge generators corresponding to RG, RB, and GB edges used in Sec. V A 3 in the main text 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 20 time units is constructed by initially preparing ancillas in the first time unit, sequentially applying CNOTs associated with RG, RB, and GB edges in time units 2–7, 8–13, 14–19, respectively, and finally measuring ancillas in time unit 20.

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 labeled locally, and are all consistent. There edges containing vertices in the boundary are treated separately; see Fig. 41 where we show parts of the full lattice from Fig. 5(b) in the main text.

For completeness, we now specify our conventions to label the circuits. To label tetrahedra around an edge with two interior vertices, we specify a local coordinate system. The z-basis vector for an RG, GB, or RB edge is the normalized vector v_{edge} from the first to the second vertex. 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 labeled in sequence according to the azimuthal angle to their midpoint in this coordinate system. For edges that contain a boundary vertex, the same applies, but the vector v_{edge} is specified according to Table VI. In the bulk, the tetrahedra are positioned regular 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 VI is found using a greedy algorithm starting 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

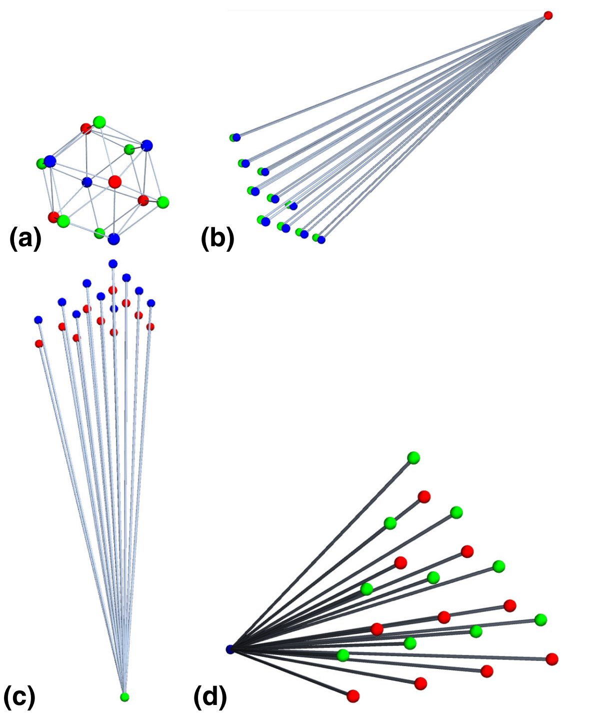


FIG. 41. Z-type gauge operators that are measured in the protocol to implement code switching are associated with RG, RB, and GB edges in the lattice, illustrated for the distance $d = 9$ lattice. We separate the edges into 20 cases. (a) Edges connecting interior vertices are classified by color and 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.

TABLE VI. Measurement circuit specification for RG, RB, and GB gauge operators. Edges connecting interior vertices (first 14 rows) are labeled by the vertex colors (first column) and the vector connecting them (second column). Edges connecting a boundary vertex are labeled by that boundary vertex v_R, v_G, or v_B and the color of the interior vertex. Qubits around an edge are labeled 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 measurement qubit for the edge.

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

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 edge types of weight 4.

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 anticommute with g_X.

APPENDIX G: OPTIMISTIC IMPROVEMENTS FOR THE 3D COLOR CODE DECODER

Here we justify our estimate of the impact on code-switching performance due to any potential improvements on the 3D color-code decoder as used in Sec. V A 6 in the main text. 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 its 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 failure probability satisfying

$$p_{\text{fail}}^{\text{IID}}(w,d) > p_{\text{3DCC}}^{(1)}\left(\frac{w/n}{p_{\text{3DCC}}^{(1)}}\right)^{(d+1)/2}, \qquad \text{(G1)}$$

where n is the number of data qubits, and $p_{\text{3DCC}}^{(1)} \simeq 1.9\%$ is the known optimal threshold of the the 3D stabilizer color-code decoder for IID Z noise [102].

We use the stabilizer-reduced weight rather than the actual weight of an error as a proxy for how hard it is to correct. It is important to do this since errors are equivalent up to the application of a stabilizer when applied to code states. Moreover, single faults in a stabilizer measurement circuit can propagate to a high-weight error, which is equivalent to a low-weight error up to the stabilizer being measured by the circuit. 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 Sec. IV C in the main text seeks to output a low-weight correction.

The second assumption is somewhat crude, since decoders that take correlations into account can outperform those that do not; see, e.g., Ref. [28]. However, we suspect that the assumption holds for typical error patterns, since the decoder will 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 gain further confidence in the first two assumptions by numerically testing the decoder on correlated and uncorrelated noise; see Fig. 42. There, we verify that the performance of the modified restriction decoder for noise produced by the code-switching protocol with stabilizer-reduced weight w (estimated using the first assumption) is comparable to its performance for 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^* [35,81,82], i.e., that $p_{\text{fail}}(p,d) \simeq (p/p^*)^{(d+1)/2}$, which is a special case of the ansatz in Eq. (7). To form an inequality in terms of the error weight w, we first take w as a proxy for the 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

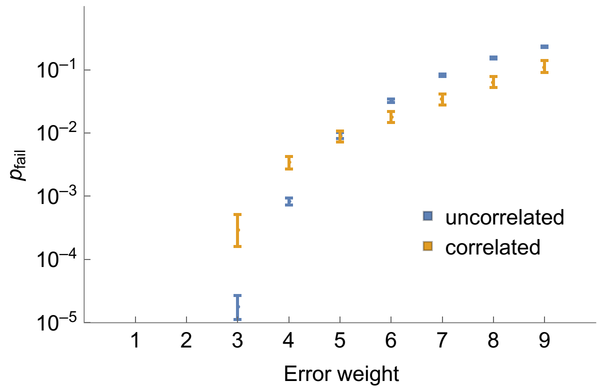


FIG. 42. Performance of the modified restriction decoder for the 3D color code for distance $d = 9$ given various error weights. Uncorrelated Z noise (blue) of weight w is generated by randomly selecting w of the n qubits and applying a Z error. Correlated Z noise (yellow) is generated by running the 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.

would correspond to the pseudothreshold being independent of system size, thereby overestimating error rate for most topological codes.

APPENDIX H: ASYMPTOTIC OVERHEAD ANALYSIS

Here we derive the scaling of the space and space-time overhead of code switching between 2D and 3D codes and state distillation using 2D codes in Eq. (43) in Sec. VI in the main text. We consider the regime with $p \ll 1$ and $\log p_{\text{fin}}/\log p \gg 1$, which allows us to assume the code distances and the number of distillation rounds are large.

First, we consider code switching. We assume that the failure probability obeys

$$\overline{p}_{\text{CS}}(p,d) = A_{\text{CS}}\left(\frac{p}{p^*_{\text{CS}}}\right)^{C_{\text{CS}}d}. \tag{H1}$$

In order to implement code switching we need (to leading order in d) $N_{3\text{D}}d^3$ qubits and $\tau_{3\text{D}}d$ time units. In the implementation presented in Sec. V in the main text, $N_{3\text{D}} = 13/12$ and $\tau_{3\text{D}} = 8$. To produce a state of infidelity p_{fin}, we find the smallest d for which $\overline{p}_{\text{CS}}(p,d) < p_{\text{fin}}$, arriving at the expressions

$$O^{\text{S}}_{\text{CS}} \sim \frac{N_{3\text{D}}}{C^3_{\text{CS}}}\left(\frac{\log p_{\text{fin}} - \log A_{\text{CS}}}{\log p - \log p^*_{\text{CS}}}\right)^3, \tag{H2}$$

$$O^{\text{ST}}_{\text{CS}} \sim \frac{N_{3\text{D}}\tau_{3\text{D}}}{C^4_{\text{CS}}}\left(\frac{\log p_{\text{fin}} - \log A_{\text{CS}}}{\log p - \log p^*_{\text{CS}}}\right)^4. \tag{H3}$$

Now we consider distillation. The starting point is the simplification that the failure probability of logical operations in the 2D color code is approximately captured by the phenomenological form

$$\overline{p}_{2\text{D}}(p,d) = A_{2\text{D}}\left(\frac{p}{p^*_{2\text{D}}}\right)^{C_{2\text{D}}d}. \tag{H4}$$

We assume that the iterative application of an R-to-1 distillation scheme, which (with perfect Clifford operations) reduces the error in a T state from q to Eq^F using a Clifford circuit containing L locations and R_{anc} encoded ancilla qubits. In the case of the 15-to-1 scheme presented in Sec. III in the main text, $R = 15$, $E = 35$, and $F = 3$, and L is of the order of 10^3, depending on the precise details of the implementation and the effective noise model. We assume that the input to the first round is an encoded magic state in a code of fixed size d_0 with infidelity $q_0 = Gp^H$, where the initialization scheme presented in Sec. III A 1 in the main text has $G = 6.07$ and $H = 1$.

We consider k rounds of distillation, with distances $\{d_1, d_2, \ldots, d_k\}$, and with outputs of each round having infidelity $\{q_1, q_2, \ldots, q_k\}$ satisfying $q_{i+1} = Eq_i^F$ in the absence of Clifford noise. In this analysis, we assume that for each round the distance d_i is chosen such that $L\overline{p}_{2\text{D}}(p, d_i) \sim q_i$, and neglect the effect of Clifford error. We can use the fact that the final round should output the target infidelity, i.e., $q_k = p_{\text{fin}}$, to relate q_i to p_{fin} as follows:

$$\begin{aligned} q_i &= \frac{p_{\text{fin}}^{\alpha(k-i)}}{E^{\beta(k-i)}}, \\ \alpha(j) &= 1/F^j, \\ \beta(j) &= \sum_{l=1}^{j} F^{-l} = \frac{1-F^{-j}}{F-1}. \end{aligned} \tag{H5}$$

We solve for the number of rounds k in $q_0 = Gp^H = p_{\text{fin}}^{\alpha(k)}/E^{\beta(k)}$, which yields

$$k = \log_F\left(\frac{\log E + (F-1)\log p_{\text{fin}}}{\log E + (F-1)\log(Gp^H)}\right). \tag{H6}$$

The ratio of the number of qubits needed in round $i+1$ and round i is

$$\begin{aligned} \frac{d_{i+1}^2}{Rd_i^2} &= \frac{[\log q_{i+1} - \log(A_{2\text{D}}L)]^2}{R\,[\log q_i - \log(A_{2\text{D}}L)]^2}, \\ &= \frac{\left[\log(Eq_i^F) - \log(A_{2\text{D}}L)\right]^2}{R\,[\log q_i - \log(A_{2\text{D}}L)]^2} \xrightarrow[p\to 0]{} \frac{F^2}{R}, \end{aligned} \tag{H7}$$

as $p \to 0$ implies $q_i \to 0$. Note that this ratio is independent of the round number i, implying that the overhead changes monotonically with i. Therefore, if $\log_F R > 2$, the first round dominates, whereas if $\log_F R < 2$ then the last round dominates. Similar observations were also made in Refs. [14,111]. We can therefore conclude that the space overhead is dominated by either the first or last round of distillation. The distances d_1 and d_k for the first and last rounds are

$$d_1 \sim \frac{HF}{C_{2\text{D}}}\left(\frac{\log p + \frac{1}{HF}\log\left(\frac{EG^F}{A_{2\text{D}}L}\right)}{\log p - \log p_{2\text{D}}^*}\right), \tag{H8}$$

$$d_k \sim \frac{1}{C_{2\text{D}}}\left(\frac{\log p_{\text{fin}} - \log(A_{2\text{D}}L)}{\log p - \log p_{2\text{D}}^*}\right). \tag{H9}$$

There are $[(R + R_{\text{anc}})/R]\,R^k$ logical qubits needed in the first rounds, and $R + R_{\text{anc}}$ for the last. To leading order in d, there are $N_{2\text{D}}d^2$ qubits needed for a distance d 2D color code, and $\tau_{2\text{D}}d$ time units, where $N_{2\text{D}} = 3/2$ and $\tau_{2\text{D}} = 8$ in our implementation in Eq. (12) in the main text. The time cost scales as the sum of the distances, which is between $\tau_{2\text{D}}d_k$ and $k\tau_{2\text{D}}d_k$ since d_k is the largest distance; see Eq. (23) in the main text. However, we just take the time cost to be $\tau_{2\text{D}}d_k$ (neglecting the prefactor k) since k depends doubly logarithmically on both p and p_{fin}. The space and space-time overhead are then

$$O_{\text{SD}}^{\text{S}} \sim N_{2\text{D}}\left(1 + R_{\text{anc}/R}\right)\max\left(R^k d_1^2, Rd_k^2\right). \tag{H10}$$

$$O_{\text{SD}}^{\text{ST}} \sim \tau_{2\text{D}}O_{\text{SD}}^{\text{S}}d_k. \tag{H11}$$

Assuming $p \ll 1$ and $\log p_{\text{fin}}/\log p \gg 1$ then yields Eq. (43) in the main text with

$$\Gamma_{\text{CS}} = 3, \tag{H12}$$

$$\Gamma_{\text{SD}} = \max(2, \log_F R), \tag{H13}$$

$$c_{\text{CS}}^{\text{S}} \sim \frac{N_{3\text{D}}}{C_{\text{CS}}^3}, \tag{H14}$$

$$c_{\text{CS}}^{\text{ST}} \sim \frac{\tau_{3\text{D}}c_{\text{CS}}^{\text{S}}}{C_{\text{CS}}}, \tag{H15}$$

$$c_{\text{SD}}^{\text{S}} \sim \frac{N_{2\text{D}}\left(1 + \frac{R_{\text{anc}}}{R}\right)}{C_{2\text{D}}^2}\begin{cases} F^2H^{2-\log_F R} & \text{if } \log_F R > 2 \\ R & \text{otherwise} \end{cases}, \tag{H16}$$

$$c_{\text{SD}}^{\text{ST}} \sim \frac{\tau_{2\text{D}}c_{\text{SD}}^{\text{S}}}{C_{2\text{D}}}. \tag{H17}$$

We expect the asymptotic expressions derived here to apply quite generally for code switching from a 2D to a 3D code, and for state distillation using 2D codes. Note that a similar analysis was carried out for the case of distillation in Ref. [14].

[1] D. Nigg, M. Müller, E. A. Martinez, P. Schindler, M. Hennrich, T. Monz, M. A. Martin-Delgado, and R. Blatt, Quantum computations on a topologically encoded qubit, 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, Superconducting quantum circuits at the surface code threshold for fault tolerance, 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, Process verification of two-qubit quantum gates by randomized benchmarking, Phys. Rev. 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, Extending the lifetime of a quantum bit with error correction in superconducting circuits, Nature **536**, 441 (2016).

[5] F. Arute *et al.*, Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).
[6] J. Preskill, Quantum computing in the NISQ era and beyond, Quantum **2**, 79 (2018).
[7] P. Shor, in *Proceedings of 37th Conference on Foundations of Computer Science* (IEEE Comput. Soc. Press, Burlington, VT, USA, 1996), p. 56.
[8] E. Knill, Scalable quantum computing in the presence of large detected-error rates, Phys. Rev. A **71**, 042322 (2005).
[9] A. M. Steane, Active Stabilization, Quantum Computation, and Quantum State Synthesis, 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, El Paso, Texas, 1997), p. 176.
[11] J. Preskill, Reliable quantum computers, Proc. Roy. Soc. Lond. **454**, 385 (1998).
[12] P. Aliferis, D. Gottesman, and J. Preskill, Quantum accuracy threshold for concatenated distance-3 codes, Quantum Inf. Comput. **6**, 097 (2005).
[13] R. Raussendorf and J. Harrington, Fault-Tolerant Quantum Computation with High Threshold in two Dimensions, Phys. Rev. Lett. **98**, 190504 (2007).
[14] R. Raussendorf, J. Harrington, and K. Goyal, Topological fault-tolerance in cluster state quantum computation, New J. Phys. **9**, 199 (2007).
[15] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, J. Math. Phys. **43**, 4452 (2002).
[16] A. W. Cross, D. P. Divincenzo, and B. M. Terhal, A comparative code study for quantum fault tolerance, Quantum Info. Comput. **9**, 541 (2009).
[17] B. J. Brown, D. Loss, J. K. Pachos, C. N. Self, and J. R. Wootton, Quantum memories at finite temperature, Rev. Mod. Phys. **88**, 045005 (2016).
[18] G. Duclos-Cianci and D. Poulin, Fast Decoders for Topological Quantum Codes, Phys. Rev. Lett. **104**, 050504 (2010).
[19] H. Anwar, B. J. Brown, E. T. Campbell, and D. E. Browne, Fast decoders for qudit topological codes, New J. Phys. **16**, 063038 (2014).
[20] G. Duclos-Cianci and D. Poulin, Kitaev's Z_{d}-code threshold estimates, Phys. Rev. A **87**, 062338 (2013).
[21] S. Bravyi and J. Haah, Quantum Self-Correction in the 3D Cubic Code Model, Phys. Rev. Lett. **111**, 200501 (2013).
[22] K. Duivenvoorden, N. P. Breuckmann, and B. M. Terhal, Renormalization group decoder for a four-dimensional toric code, IEEE Trans. Inf. Theory **65**, 2545 (2019).
[23] A. Kubica and J. Preskill, Cellular-Automaton Decoders with Provable Thresholds for Topological Codes, Phys. Rev. Lett. **123**, 020501 (2019).
[24] M. Vasmer, D. E. Browne, and A. Kubica, Cellular automaton decoders for topological quantum codes with noisy measurements and beyond, Sci. Rep. **11**, 2027 (2021).
[25] S. Bravyi, M. Suchara, and A. Vargo, Efficient algorithms for maximum likelihood decoding in the surface code, Phys. Rev. A – At., Mol., Opt. Phys. **90**, 032326 (2014).
[26] A. S. Darmawan and D. Poulin, Linear-time general decoding algorithm for the surface code, Phys. Rev. E **97**, 051302 (2018).
[27] N. H. Nickerson and B. J. Brown, Analysing correlated noise on the surface code using adaptive decoding algorithms, Quantum **3**, 131 (2019).
[28] N. Maskara, A. Kubica, and T. Jochym-O'Connor, Advantages of versatile neural-network decoding for topological codes, Phys. Rev. A **99**, 052351 (2019).
[29] C. Chamberland and P. Ronagh, Deep neural decoders for near term fault-tolerant experiments, Quantum Sci. Technol. **3**, 044002 (2018).
[30] N. Delfosse, Decoding color codes by projection onto surface codes, Phys. Rev. A **89**, 012317 (2014).
[31] A. Kubica and N. Delfosse, Efficient color code decoders in $d \geq 2$ dimensions from toric code decoders, arXiv:1905.07393 (2019).
[32] A. Y. Kitaev, Quantum computations: Algorithms and error correction, Russ. Math. Surv. **52**, 1191 (1997).
[33] S. Bravyi and A. Y. Kitaev, Quantum codes on a lattice with boundary, arXiv:9811052 (1998).
[34] H. Bombin and M. Martin-Delgado, Topological Quantum Distillation, Phys. Rev. Lett. **97**, 180501 (2006).
[35] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Surface codes: Towards practical large-scale quantum computation, Phys. Rev. A **86**, 032324 (2012).
[36] C. Chamberland, G. Zhu, T. J. Yoder, J. B. Hertzberg, and A. W. Cross, Topological and Subsystem Codes on Low-Degree Graphs with Flag Qubits, Phys. Rev. X **10**, 011022 (2020).
[37] C. Chamberland, A. Kubica, T. J. Yoder, and G. Zhu, Triangular color codes on trivalent graphs with flag qubits, New J. Phys. **22**, 023019 (2020).
[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, Scalable designs for quasiparticle-poisoning-protected topological quantum computation with majorana zero modes, Phys. Rev. B **95**, 235305 (2017).
[39] R. Chao, M. E. Beverland, N. Delfosse, and J. Haah, Optimization of the surface code design for majorana-based qubits, Quantum **4**, 352 (2020).
[40] B. Eastin and E. Knill, Restrictions on Transversal Encoded Quantum Gate Sets, Phys. Rev. Lett. **102**, 110502 (2009).
[41] B. Zeng, A. Cross, and I. L. Chuang, Transversality versus universality for additive quantum codes, IEEE Trans. Inf. Theory **57**, 6272 (2011).
[42] T. Jochym-O'Connor, A. Kubica, and T. J. Yoder, Disjointness of Stabilizer Codes and Limitations on Fault-Tolerant Logical Gates, Phys. Rev. X **8**, 21047 (2018).
[43] S. Bravyi and R. König, Classification of Topologically Protected Gates for Local Stabilizer Codes, Phys. Rev. Lett. **110**, 170503 (2013).
[44] F. Pastawski and B. Yoshida, Fault-tolerant logical gates in quantum error-correcting codes, arXiv:1408.1720 (2014).

[45] M. E. Beverland, O. Buerschaper, R. Koenig, F. Pastawski, J. Preskill, and S. Sijher, Protected gates for topological quantum field theories, J. Math. Phys. **57,** 022201 (2016).

[46] P. Webster, M. Vasmer, T. R. Scruby, and S. D. Bartlett, Universal fault-tolerant quantum computing with stabiliser codes, arXiv:2012.05260 (2020).

[47] H. Bombin and M. Martin-Delgado, Exact topological quantum order in $d = 3$ and beyond: Branyons and brane-net condensates, Phys. Rev. B **75**, 075103 (2007).

[48] A. Kubica, B. Yoshida, and F. Pastawski, Unfolding the color code, New J. Phys. **17**, 083026 (2015).

[49] M. Vasmer and D. E. Browne, Three-dimensional surface codes: Transversal gates and fault-tolerant architectures, Phys. Rev. A. **100**, 012312 (2019).

[50] S. Bravyi and A. Kitaev, Universal quantum computation with ideal clifford gates and noisy ancillas, Phys. Rev. A **71**, 022316 (2005).

[51] E. Knill, Fault-tolerant postselected quantum computation: Threshold analysis, arXiv:0404104 (2004).

[52] E. Knill, Fault-tolerant postselected quantum computation: Schemes, arXiv:0402171 (2004).

[53] D. Litinski, Magic state distillation: Not as costly as you think, Quantum **3**, 205 (2019).

[54] M. Beverland, E. Campbell, M. Howard, and V. Kliuchnikov, Lower bounds on the non-clifford resources for quantum computations, Quantum Sci. Technol. **5**, 035009 (2020).

[55] A. Paetznick and B. W. Reichardt, Universal Fault-Tolerant Quantum Computation with Only Transversal Gates and Error Correction, Phys. Rev. Lett. **111**, 090505 (2013).

[56] J. T. Anderson, G. Duclos-Cianci, and D. Poulin, Fault-Tolerant Conversion between the Steane and Reed-Muller Quantum Codes, Phys. Rev. Lett. **113**, 080501 (2014).

[57] H. Bombin, Gauge color codes: Optimal transversal gates and gauge fixing in topological stabilizer codes, New J. Phys. **17**, 083002 (2015).

[58] H. Bombín, Dimensional jump in quantum error correction, New J. Phys. **18**, 043038 (2016).

[59] H. Bombin, Single-Shot Fault-Tolerant Quantum Error Correction, Phys. Rev. X **5**, 031043 (2015).

[60] S. Bravyi and J. Haah, Magic-state distillation with low overhead, Phys. Rev. A **86**, 052329 (2012).

[61] J. Haah, M. B. Hastings, D. Poulin, and D. Wecker, Magic state distillation with low space overhead and optimal asymptotic input count, Quantum **1**, 31 (2017).

[62] J. Haah and M. B. Hastings, Codes and protocols for distilling T, controlled-S, and toffoli gates, Quantum **2**, 71 (2018).

[63] P. Brooks, Ph.D thesis, Caltech, 2013, https://resolver.caltech.edu/CaltechTHESIS:05302013-143644943.

[64] T. Jochym-O'Connor, Y. Yu, B. Helou, and R. Laflamme, The robustness of magic state distillation against errors in clifford gates, Quantum Inf. Comput. **13**, 0361 (2013).

[65] N. C. Jones, Logic synthesis for fault-tolerant quantum computers, PhD Thesis (Stanford):1310.7290 [quant-ph] (2013).

[66] A. Kubica, Ph.D. thesis, Caltech, 2018, https://resolver.caltech.edu/CaltechTHESIS:05282018-173928314.

[67] M. Beverland, Ph.D. thesis, Caltech, 2016, https://resolver.caltech.edu/CaltechTHESIS:06072016-162802972.

[68] S. Bravyi and A. Cross, Doubled Color Codes, arXiv:1509.03239 (2015).

[69] T. Jochym-O'Connor and S. D. Bartlett, Stacked codes: Universal fault-tolerant quantum computation in a two-dimensional layout, Phys. Rev. A **93**, 022323 (2016).

[70] C. Jones, P. Brooks, and J. Harrington, Gauge color codes in two dimensions, Phys. Rev. A **93**, 052332 (2016).

[71] H. Bombin, 2D quantum computation with 3D topological codes, arXiv:1810.09571 (2018).

[72] B. J. Brown, A fault-tolerant non-clifford gate for the surface code in two dimensions, Sci. Adv. **6** (2020).

[73] T. R. Scruby, D. E. Browne, P. Webster, and M. Vasmer, Numerical implementation of just-in-time decoding in novel lattice slices through the three-dimensional surface code, arXiv:2012.08536 (2020).

[74] J. Iverson and A. Kubica (to be published).

[75] C. D. Hill, A. G. Fowler, D. S. Wang, and L. C. L. Hollenberg, Fault-tolerant quantum error correction code conversion, Quantum Info. Comput. **13**, 439 (2013).

[76] T. J. Yoder, R. Takagi, and I. L. Chuang, Universal Fault-Tolerant Gates on Concatenated Stabilizer Codes, Phys. Rev. X **6**, 031039 (2016).

[77] T. Jochym-O'Connor and R. Laflamme, Using Concatenated Quantum Codes for Universal Fault-Tolerant Quantum Gates, Phys. Rev. Lett. **112**, 010505 (2014).

[78] C. Chamberland and A. W. Cross, Fault-tolerant magic state preparation with flag qubits, Quantum **3**, 143 (2019).

[79] C. Chamberland and K. Noh, Very low overhead fault-tolerant magic state preparation using redundant ancilla encoding and flag qubits, npj Quantum Inf. **6**, 91 (2020).

[80] Since we assume throughout that arbitrary one- and two-qubit physical 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 later see that logical Clifford operations can be done offline in 2D color codes such that the logical noise on encoded T states can also be twirled offline.

[81] A. G. Fowler, Analytic asymptotic performance of topological codes, Phys. Rev. A **87**, 040301 (2013).

[82] A. J. Landahl, J. T. Anderson, and P. R. Rice, Fault-tolerant quantum computing with color codes, arXiv:1108.5738 (2011).

[83] A. Kubica and M. Beverland, Universal transversal gates with color codes: A simplified approach, Phys. Rev. A **91**, 032330 (2015).

[84] There is another stabilizer code with parameters $x = 1, z = 0$ but we work only with the $x = 0, z = 1$ version here.

[85] A. R. Calderbank and P. W. Shor, Phys. Rev. A **54**, 1098 (1996).

[86] A. M. Steane, Phys. Rev. A **54**, 4741 (1996).

[87] A. J. Landahl and C. Ryan-Anderson, Quantum computing by color-code lattice surgery, arXiv:1407.5103 [quant-ph] (2014).

[88] A. J. Landahl and C. Cesare, Complex instruction set computing architecture for performing accurate quantum Z rotations with less magic, arXiv:1302.3240 (2013).
[89] M. B. Hastings and J. Haah, Distillation with Sublogarithmic Overhead, Phys. Rev. Lett. **120**, 050504 (2018).
[90] B. W. Reichardt, Quantum universality from magic states distillation applied to CSS codes, Quantum Inf. Process. **4**, 251 (2005).
[91] 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, state distillation for higher-order schemes should be possible.
[92] A. M. Meier, B. Eastin, and E. Knill, Magic-state distillation with the four-qubit code, Quantum Inf. Comput. **13**, 0195 (2013).
[93] C. Jones, Multilevel distillation of magic states for quantum computing, Phys. Rev. A **87**, 042305 (2013).
[94] G. Duclos-Cianci and D. Poulin, Reducing the quantum-computing overhead with complex gate distillation, Phys. Rev. A **91**, 042315 (2015).
[95] E. T. Campbell and J. O'Gorman, An efficient magic state approach to small angle rotations, Quantum Sci. Technol. **1**, 015007 (2016).
[96] J. Haah, M. B. Hastings, D. Poulin, and D. Wecker, Magic state distillation at intermediate size, arXiv:1709.02789 (2017).
[97] J. Haah, Towers of generalized divisible quantum codes, Phys. Rev. A **97**, 042327 (2018).
[98] In Ref. [112], a threshold above 1% was reported, however follow-up work [37] failed to reproduce this result and reported 0.7%.
[99] A. M. Stephens, Efficient fault-tolerant decoding of topological color codes, arXiv:1402.3037 (2014).
[100] B. J. Brown, N. H. Nickerson, and D. E. Browne, Fault-tolerant error correction with the gauge color code, Nat. Commun. **7**, 4 (2015).
[101] B. M. Terhal, Quantum error correction for quantum memories, Rev. Mod. Phys. **87**, 307 (2015).
[102] A. Kubica, M. E. Beverland, F. Brandão, J. Preskill, and K. M. Svore, Three-Dimensional Color Code Thresholds via Statistical-Mechanical Mapping, Phys. Rev. Lett. **120**, 180501 (2018).
[103] We say that an ancilla prepared in $|+\rangle$ or $|0\rangle$ state is stabilized by a Pauli-X or -Z operator, respectively.
[104] Note that it is possible to considerably improve this implementation of the CNOT gate. Namely, one first decodes 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 applies the correction to the destination of the copied error before correcting the residual error there [113].
[105] Y. Li, A magic state's fidelity can be superior to the operations that created it, New J. Phys. **17**, 023037 (2015).
[106] C. Chamberland and K. Noh, Very low overhead fault-tolerant magic state preparation using redundant ancilla encoding and flag qubits, arXiv:2003.03049 (2020).
[107] If we were to produce just one output T state, then the overhead would slightly increase as we would need to guarantee that with high probability there are sufficiently many T states at each level of the state-distillation protocol.
[108] H. Bombin, Transversal gates and error propagation in 3d topological codes, arXiv:1810.09575 (2018).
[109] S. Turner, J. Hanish, E. Blanchard, N. Davis, and B. L. Cour, A decoder for the color code with boundaries, arXiv:2003.11602 (2020).
[110] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, Trapped-ion quantum computing: Progress and challenges, Appl. Phys. Rev. **6**, 021314 (2019).
[111] J. O'Gorman and E. T. Campbell, Quantum computation with realistic magic-state factories, Phys. Rev. A **95**, 032338 (2017).
[112] D. S. Wang, A. G. Fowler, and L. C. L. Hollenberg, Surface code quantum computing with error rates over 1%, Phys. Rev. A **83**, 020302 (2011).
[113] A. Fowler (private communication).