Toward Self-Correcting Quantum Codes for Neutral Atom Arrays

Jinkang Guo,[1,2] Yifan Hong,[1,2,3] Adam Kaufman,[1,4] and Andrew Lucas[1,2,*]

[1] *Department of Physics, University of Colorado, Boulder, Colorado 80309, USA*

[2] *Center for Theory of Quantum Matter, University of Colorado, Boulder, Colorado 80309, USA*

[3] *Joint Quantum Institute & Joint Center for Quantum Information and Computer Science, NIST/University of Maryland, College Park, Maryland 20742, USA*

[4] *JILA, University of Colorado and National Institute of Standards and Technology, Boulder, Colorado 80309, USA*

(Received 11 August 2025; revised 31 October 2025; accepted 26 November 2025; published 2 January 2026)

Discovering low-overhead quantum error-correcting codes is of significant interest for fault-tolerant quantum computation. For hardware capable of long-range connectivity, the bivariate bicycle codes offer significant overhead reduction compared to surface codes with similar performance. In this work, we present "Z semidirect Z (ZSZ) codes," a simple nonabelian generalization of the bivariate bicycle codes based on the group $\mathbb{Z}_\ell \rtimes \mathbb{Z}_m$. We numerically demonstrate that certain instances of this code family achieve competitive performance with the bivariate bicycle codes under circuit-level depolarizing noise using a belief-propagation and ordered-statistics decoder, with an observed threshold around 0.5%. We also benchmark the performance of this code family under local "self-correcting" decoders, where we observe significant improvements over the bivariate bicycle codes, including evidence of a sustainable threshold around 0.095%, which is higher than the 0.06% that we estimate for the four-dimensional toric code under the same noise model. These results suggest that ZSZ codes are promising candidates for scalable self-correcting quantum memories. Finally, we describe how ZSZ codes can be realized with neutral atoms trapped in movable tweezer arrays, where a complete round of syndrome extraction can be achieved using simple global motions of the atomic arrays.

DOI: 10.1103/mfmt-fwkg

I. INTRODUCTION

Recent advances in experimental quantum processors have resulted in numerous demonstrations of small-scale quantum error correction (QEC) [1–12], a key component of fault-tolerant quantum computation. The sizes of these experiments range from tens to hundreds of physical qubits with logical qubits in the range of a few to a few dozen.

With larger-scale quantum processors come the prospects of increasingly sophisticated quantum error-correcting codes. A significant milestone was the discovery of quantum low-density parity-check (LDPC) codes with asymptotically optimal (good) $[\![n, k = \Theta(n), d = \Theta(n)]\!]$ parameters [13–15]. Many other exciting advances include the discovery of asymptotically good non-LDPC quantum codes capable of magic state distillation with constant overhead [16], quantum LDPC codes with high rate and transversal non-Clifford gates [17–19], non-LDPC quantum codes with addressable non-Clifford gates [20,21], and end-to-end fault tolerance with constant spatial and low temporal overheads with concatenated codes [22] and LDPC codes [23,24]. There has furthermore been progress in other aspects of fault tolerance such as lowering the overhead for addressable logical Pauli measurements [25–29], a sufficient ingredient for addressable logical Clifford gates and the Pauli-based computational model [30].

For the near-term, the asymptotically good codes might not be practical due to their large connectivity requirements. Still, one may hope that the mathematical techniques used to discover these codes can give rise to simple small instances that nevertheless have favorable properties. One such example is arguably a family of quantum LDPC codes called bivariate bicycle (BB) codes, which achieve competitive memory performance with the surface code while occupying considerably fewer physical qubits, at the cost of nonlocal connectivity [31]. It was later shown that a fault-tolerant architecture consisting of BB codes

*Contact author: andrew.j.lucas@colorado.edu

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

and surface codes as memory and computational blocks respectively achieved a lower footprint than surface codes on a wide range of practical algorithms [29,32,33]. However, all BB codes are local (with finite-range interactions) in some finite-dimensional Euclidean space [34], which constrains the ultimate scaling of the code parameters with system size [35,36]. These bounds motivate the study of ways to minimally circumvent them while hopefully retaining their favorable error-correcting properties.

Given that neutral atom platforms have the potential ability to realize the nonlocal interactions required of very high-dimensional codes, this paper asks the question of whether—with a similar code size and implementation overhead—there are "better" potential codes than BB codes. In this paper, we will focus on the question of finding quantum memories, including those which may exhibit passive decoding, where errors are locally corrected with only local syndrome information. Locality here is defined with respect to the Tanner graph of the code; i.e., how qubits and checks are connected. Passive QEC offers a few alluring properties over traditional QEC, typically at the expense of a lower threshold. First, it enables a new paradigm of error correction that forgoes mid-circuit measurements and feedback, also known as measurement-free quantum error correction (MFQEC) [37–39]. Second, each decoding "cycle" in a passive memory is typically a constant-depth circuit that does not depend on any information from previous cycles, analogous to single-shot error correction [40,41], and so constant-depth logical operations (e.g., transversal) can be performed between each cycle. In other words, a logical cycle of a passive quantum memory takes $O(1)$ time, in units of syndrome extraction cycles. In contrast, the typical logical cycle for surface codes and BB codes requires $\Theta(d)$ time [31,37]; logical operations can be performed at a faster rate but at the cost of increased decoding complexity [42,43].

In this paper, we introduce the **Z semidirect Z (ZSZ) codes**, a family of quantum LDPC codes where each parity check involves six qubits, and each qubit participates in six parity checks split evenly between three X-type and three Z-type checks. These codes are a nonabelian generalization of the BB codes where we "twist" the associated product involved in the construction. This twist, otherwise known as a semidirect product, involves a relatively simple adjustment to the microscopic rules for how qubits and checks are connected; see Fig. 1 for an illustration. Nonetheless, we demonstrate that this modification can lead to *global* changes in code properties, the most drastic of which is the possibility for passive error correction. Under our experimentally inspired noise model, we observe a threshold around 0.5% for ZSZ codes under d rounds of syndrome extraction and global decoding, which is close to the estimated 0.8% threshold for the surface code under the same noise model. Using a passive "self-correcting" decoder, we observe a sustainable threshold around 0.095%, which is higher than the estimated 0.06% threshold for the four-dimensional toric code. To the best of our knowledge at the time of writing, 0.095% is the highest observed sustainable threshold for passive decoding of any known quantum LDPC code under similar circuit-level noise. We note that this threshold can potentially increase if measurements are utilized and decoding is performed "offline" on a noiseless classical computer; when running the decoder in the measurement-free setting,

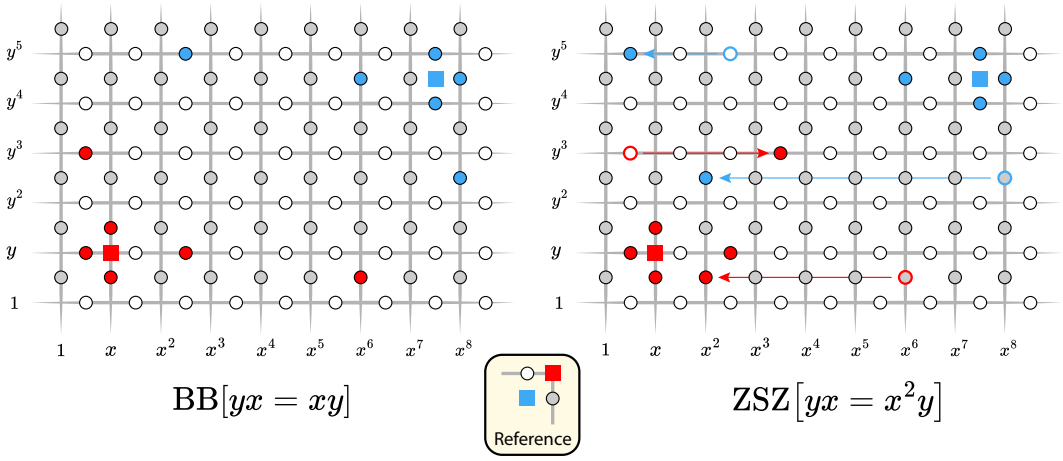


FIG. 1. The physical layouts of a bivariate bicycle and a ZSZ code with shared polynomials $a = 1 + x^2 + y^2$ and $b = 1 + x^5 + y$ are depicted. Qubits (circles) live on the horizontal and vertical links of a 9×6 grid with periodic boundaries. The red (blue) square denotes an X-check (Z-check) and the red-shaded (blue-shaded) circles label its support, which can be computed through a and b. The arrows on the right side show how the check support changes upon applying the twist $yx = x^2y$, which converts the bivariate bicycle code to the ZSZ code. In contrast to bivariate bicycle codes, the other checks of ZSZ codes cannot generically be obtained upon translation.

one needs to also take into account possible faults in its implementation. We also note that these thresholds assume access to an initially noiseless logical state and may be affected when accounting for noisy state preparation, for which we do not currently have a passive scheme. We finally study the implementation of ZSZ codes as a memory in neutral atom arrays and describe the necessary optical-tweezer movements required to perform syndrome extraction. Similar to the BB codes, we provide a two-dimensional rectangular embedding of ZSZ codes. With respect to this embedding, we then construct a routing protocol for syndrome extraction whose complexity is logarithmic in the horizontal dimension and linear in the vertical dimension. Although our ZSZ routing complexity increases with system size, unlike that of toric and BB codes with constant complexity, the single-shot property of ZSZ codes allows for fewer rounds of syndrome extraction within a logical cycle. A more detailed study of this trade-off would be important when deciding between ZSZ codes or BB codes for the neutral-atom architecture.

The paper is organized as follows. In Sec. II, we review the construction of quantum stabilizer codes from two-block matrices and focus our attention on those based on group algebras. In Sec. III, we introduce the ZSZ codes as a special family of these two-block quantum codes and present some geometrical and algebraical arguments relevant to their understanding. In Sec. IV, we discuss our noise model and present the results of our numerical simulations. In Sec. V, we discuss the implementation of ZSZ codes in the neutral-atom architecture and present a routing protocol to realize their long-range connectivity. Finally in Sec. VI, we close with some open questions and concluding remarks. The Appendixes contain technical details for the arguments in the main text.

A. Related works

Paz-Silva *et al.* [44] design a MFQEC protocol for the 9-qubit Bacon-Shor code and demonstrate a pseudothreshold around 4×10^{-5} for preparation and gate errors. Heußen *et al.* [45] design a flagged-based MFQEC protocol for the 7-qubit Steane code and achieve a pseudothreshold around 6×10^{-5} with a single-parameter noise model and a two-qubit gate decomposition as well as 6×10^{-4} with a multiparameter noise model including native multiqubit gates such as CCZ. Our noise model is relatively close to theirs, but we note that their simulations involve the full state vector and so can account for coherent errors from non-Clifford gates. More recently, Butt *et al.* [46] experimentally demonstrate a universal set of measurement-free fault-tolerant gadgets using a combination of the $[\![4,2,2]\!]$ "Iceberg" code and the $[\![8,3,2]\!]$ color code.

The above works focus on specific small-instance codes, and scalable fault tolerance is achieved through concatenation. In contrast, our proposed scheme relies on constant-depth decoding, and scalable fault tolerance is achieved through a growing family of LDPC codes. Recently, Park *et al.* [47] study this problem for the 2D repetition and 4D toric codes and employ classical reinforcement learning to achieve MFQEC circuits with multiqubit gates that outperform conventional local decoders based on Toom's (sweep) rule in the subthreshold regime. Their tools are general-purpose, and it would be interesting to see if and by how much they can improve passive thresholds when applied to ZSZ codes.

II. REVIEW OF TWO-BLOCK CODES

In this section, we review some of the concepts used in our code construction, starting from an abstract prescription of two-block quantum codes [48] and then focusing on specific two-block codes built with the help of a group algebra [49,50].

A. Two-block quantum CSS codes

A Calderbank-Shor-Steane (CSS) code [51,52] is a specific type of quantum stabilizer code [53] whose Pauli checks are strictly X-type or Z-type. The Pauli checks can be neatly packaged into the rows of two binary parity-check matrices H_X and H_Z, and the stabilizer commutativity condition becomes the orthogonality condition $H_X H_Z^{\mathsf{T}} = 0$, where addition is performed modulo 2. Such a code is an LDPC code if the rows and columns of H_X and H_Z have an O(1) number of nonzero entries.

A two-block CSS code [48] builds the above parity-check matrices with the help of two $n \times n$ commuting square matrices A and B,

$$H_X = \left(A \mid B\right), \tag{1a}$$

$$H_Z = \left(B^{\mathsf{T}} \mid A^{\mathsf{T}}\right). \tag{1b}$$

Since $[A,B] = 0$, we have $H_X H_Z^{\mathsf{T}} = AB + BA = [A,B] = 0$ [54] and thus H_X and H_Z define a valid CSS code. Since A and B are $n \times n$ matrices, there are $2n$ physical qubits and n Pauli checks of each type. Because of the two-block structure above, we also have that

$$\ker A, \ker B \subset \ker H_X, \tag{2a}$$

$$\ker A^{\mathsf{T}}, \ker B^{\mathsf{T}} \subset \ker H_Z, \tag{2b}$$

where by slight abuse of notation, we write $\ker A$ to mean $\ker A$ on the left block and zeros on the right block, and vice versa for $\ker B$. The task is now to construct the commuting matrices A and B, which we can view as two classical linear codes. Ideally, we would like A and B to have large code distances in the hopes that their two-block CSS code will as well. In addition, we would also like A and B to be sparse so that the resulting CSS code is LDPC and hence has inherent fault-tolerant properties [55].

B. Two-block group-algebra (2BGA) codes

One method of constructing two sparse, commuting matrices A and B is with the help of a group algebra [49,50]. Suppose we have a finite group $\mathcal{G}$ of order n. A group algebra $K[\mathcal{G}]$, where K is a field, is an object that marries the additive properties of fields and the multiplicative properties of groups. Since we are interested in qubit CSS codes constructed from binary matrices, we will choose $K = \mathbb{F}_2$ as our field. A generic element a of the group algebra $\mathbb{F}_2[\mathcal{G}]$ can be written as a linear combination of group elements with binary coefficients,

$$a = \sum_{i=1}^{n} c_i g_i. \tag{3}$$

Addition and multiplication are done in a natural way. For example, $(g_1 + g_2) + (g_3 + g_4) = g_1 + g_2 + g_3 + g_4$, while $(g_1 + g_2)(g_3 + g_4) = g_1g_3 + g_1g_4 + g_2g_3 + g_2g_4$ and $g_1 + g_1 = 0$.

For a 2BGA code, we will choose two elements $a, b \in \mathbb{F}_2[\mathcal{G}]$. Because we are ultimately constructing LDPC codes, we will choose a and b to have a finite number of terms, which we will see results in sparse parity-check matrices. Our binary matrices A and B are given by a binary matrix representation of our group-algebra elements a and b,

$$A = \mathbb{B}[a], \quad B = \mathbb{B}[b], \tag{4}$$

where $\mathbb{B}[a]$ denotes some binary matrix representation of $a \in \mathbb{F}_2[\mathcal{G}]$, which is usually taken as the (binary) regular representation. In the regular representation, a group element maps to a unique basis vector over $\mathbb{F}_2^n$, and its group action becomes an $n \times n$ permutation matrix which encodes the group's multiplication (Cayley) table. For nonabelian groups, we may choose this regular representation to correspond to either left ($L[\cdot]$) or right ($R[\cdot]$) multiplication [56], a fact which will prove useful shortly. It follows that A and B are sums of permutation matrices. For example, using the group $\mathcal{G} = \mathbb{Z}_3$ with polynomial $a = 1 + x$, we get

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}, \tag{5}$$

which is a classical parity-check matrix for the 3-bit repetition code. A benefit of using the regular representation is that the row and column weights of A and B are immediately bounded by the number of nonzero terms of their corresponding group-algebra elements a and b. Choosing only a small constant number of terms in Eq. (3) results in a sparse A and B. Commutativity between A and B follows from that of a and b, which always holds if $\mathcal{G}$ is abelian. As an example, BB codes [31] are 2BGA codes over the abelian group $\mathbb{Z}_\ell \times \mathbb{Z}_m$ for integers $\ell, m > 0$.

When $\mathcal{G}$ is nonabelian, A and B are no longer guaranteed to commute since a and b do not necessarily commute anymore. However, recall that for a nonabelian group, we have the choice of choosing either the left-regular or right-regular representations for A and B. As such, we choose $A = L[a]$ as the left-regular representation and $B = R[b]$ as the right-regular representation. The associativity of group multiplication then ensures that a and b, and thereby A and B, commute.

As an example, the symmetric group S_3 on three symbols is the smallest nonabelian group. In cycle notation, all six of its elements are

$$S_3 = \{(1), (1,2), (1,3), (2,3), (1,2,3), (1,3,2)\}. \tag{6}$$

Since $|S_3| = 6$, its regular representation will be six-dimensional, and each of the above group actions will map to some 6×6 permutation matrix. Take the transposition $(1,3)$ for instance. Using the ordering of Eq. (6), it will map to the basis vector $(0,0,1,0,0,0) \in \mathbb{F}_2^6$. Now, acting $(1,3)$ to the left of Eq. (6) gives us $(1,3) \cdot S_3 = \{(1,3), (1,2,3), (1), (1,3,2), (1,2), (2,3)\}$, and acting on the right gives us $S_3 \cdot (1,3) = \{(1,3), (1,3,2), (1), (1,2,3), (2,3), (1,2)\}$. Comparing with the ordering of Eq. (6), the left-regular and right-regular representations of $(1,3)$ are the permutation matrices

$$L[(1,3)] = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix},$$

$$R[(1,3)] = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{7}$$

One can quickly verify that $L[(1,3)]$ and $R[(1,3)]$ commute, as is guaranteed by the associativity of group multiplication.

III. ZSZ CODES

We are now ready to define our ZSZ codes. Let ℓ, m, q be positive integers satisfying

$$q^m \equiv 1 \pmod{\ell}. \tag{8}$$

We define the ZSZ group of order ℓm according to the presentation

$$\mathbb{Z}_\ell \rtimes_q \mathbb{Z}_m := \langle x, y \,|\, x^\ell = y^m = yxy^{-1}x^{-q} = 1 \rangle. \tag{9}$$

A generic group element can be written as some word made up of the symbols x, y, e.g., x^2yxy^2. The relations in Eq. (9) provide us a way to compute equivalences between different words. For both $\mathbb{Z}_\ell \times \mathbb{Z}_m$ and $\mathbb{Z}_\ell \rtimes_q \mathbb{Z}_m$, we can always reduce all words to a canonical lexicographical form $x^i y^j$, where $i = 0, \ldots, \ell - 1$ and $j = 0, \ldots, m - 1$. The final relation $yxy^{-1} = x^q$ defines the twist that the y terms apply to the x terms upon conjugation and distinguishes between the abelian direct product ($q = 1$) and nonabelian semidirect product ($q > 1$). In the language of group theory, $\mathbb{Z}_\ell := \langle x \,|\, x^\ell = 1 \rangle$ is a normal subgroup of Eq. (9), and $y \in \mathbb{Z}_m$ acts on this subgroup via conjugation, i.e., the automorphism $\varphi_y(x) = x^q$ of $\mathbb{Z}_\ell$, the validity of which is ensured by Eq. (8). We define a **ZSZ code** to be a 2BGA code where the binary matrix $A(B)$ in (1) is the left(right)-regular representation of a $\mathbb{F}_2$-group-algebra element $a(b)$ with respect to a ZSZ group; i.e., $a, b \in \mathbb{F}_2[\mathbb{Z}_\ell \rtimes_q \mathbb{Z}_m]$. We review some basic group theories relevant to this code in Appendix A. For brevity, we denote $\mathrm{ZSZ}(\ell, m, q; a, b)$ to be the ZSZ code with group (9) and polynomials $a, b \in \mathbb{F}_2[\mathbb{Z}_\ell \rtimes_q \mathbb{Z}_m]$.

A geometrical object called a Cayley graph will be useful to describe how parity checks and qubits are connected in a ZSZ code. For any finite group G equipped with a set of generators $S = \{s_1, s_2, \ldots\}$ that does not include the identity, the (directed) Cayley graph $\mathrm{Cay}(G, S)$ is a simple graph where vertices are labeled by group elements, and two vertices share an edge if and only if their corresponding group elements are related by a generator in S; e.g., the vertices corresponding to g_1 and g_2 are connected by an edge if $g_1 = sg_2$ or $g_2 = sg_1$ for some $s \in S$. When G is nonabelian, the order of multiplication matters, which motivates the distinction between left and right Cayley graphs. Edges on the left Cayley graph represent left-multiplication by a generator, and edges on the right Cayley graph represent right-multiplication. For example, $\mathrm{Cay}(\mathbb{Z}_\ell, \{x\})$ is a ring graph of length ℓ, and $\mathrm{Cay}(\mathbb{Z}_\ell \times \mathbb{Z}_m, \{x, y\})$ is an $\ell \times m$ rectangular lattice with periodic boundaries.

Our lexicographical ordering $x^i y^j$ of the ZSZ group elements motivates a two-dimensional rectangular layout where one axis labels the exponent of x and the other y. Because we have a two-block code, we will also have two copies of this group, for a total of $2\ell m$ data qubits, which can be arranged as the links (edges) of a rectangular lattice with periodic boundaries: horizontal and vertical links comprise the qubits in the A and B blocks of Eq. (1) respectively (see Fig. 1).

For a BB code, which we remind the reader corresponds to the abelian case $q = 1$, it is relatively straightforward to determine the support of the parity checks from the polynomials a and b. For an X-check at coordinate (i, j) corresponding to the term $x^i y^j$, we examine the monomials in a and b and add their respective x and y-exponents to i and j. This simple addition is possible because we can commute the x and y terms through each other and combine like terms together. One can also envision these rules as traversing a path on the rectangular lattice: a generic monomial such as $x^\alpha y^\beta$ corresponds to moving α units along the x direction and β units along the y direction. Each monomial in a goes to a horizontal qubit, and each monomial in b goes to a vertical qubit. For example, in Fig. 1 with polynomials $a = 1 + x^2 + y^2$ and $b = 1 + x^5 + y$, the X-check at position xy connects to horizontal qubits at positions given by $a(xy) = xy + x^3y + xy^3$ and vertical qubits at positions given by $(xy)b = xy + x^6y + xy^2$.

For a ZSZ code, the simple addition rules above do not work because when we move x terms through y terms, the exponents on the x terms will change according to the twist q in Eq. (9). Instead, the rules are given by the following "push-through" relations:

$$(x^i y^j) x^\alpha = x^{i + q^j \alpha} y^j, \tag{10a}$$

$$y^\beta (x^i y^j) = x^{q^\beta i} y^{j+\beta}. \tag{10b}$$

Note that these push-through relations are inherently encoded in the ZSZ group's left and right Cayley graphs, and so we can recover an analogous geometrical path picture as in the abelian case upon replacing the rectangular lattice with the Cayley graphs of the corresponding ZSZ group. This prescription also encompasses the BB codes since their Cayley graphs are precisely the rectangular lattices used in their analyses. Now, when we see a monomial in a like $x^\alpha y^\beta$, we will read off from right to left [57]: we first traverse β steps along y-edges and then α steps along x-edges in the left Cayley graph. When we see a monomial in b like $x^\alpha y^\beta$, we will read off from left to right: we first traverse α steps along x-edges and then β steps along y-edges in the right Cayley graph. See Fig. 2 for an explicit example of a ZSZ code with both its underlying left and right Cayley graphs drawn out. In our rectangular layout, we see that the left Cayley graph consists of m copies of $\mathbb{Z}_\ell$ arranged in rows, with neighboring rows related by the group automorphism $\varphi_y(x) = x^q$. The right Cayley graph also has m rows of $\mathbb{Z}_\ell$, but φ_y now acts independently within each row rather than between rows: the row associated with y^j transforms according to $\varphi_y^j(x) = x^{qj}$.

The advantage of the nontrivial push-through relations is as follows. Let $B_r(h)$ be the set of vertices within distance r of vertex h in the Cayley graph of the underlying group. If $q = 1$, i.e., we have a BB code with p distinct monomials $g_1, \ldots, g_p$ in a and b, then

$$|B_r(h)| = O(r^p). \tag{11}$$

This bound is very loose, but it is easy to motivate: after r multiplications by p elements, the number of distinct group elements we can reach (up to the left or right separation of

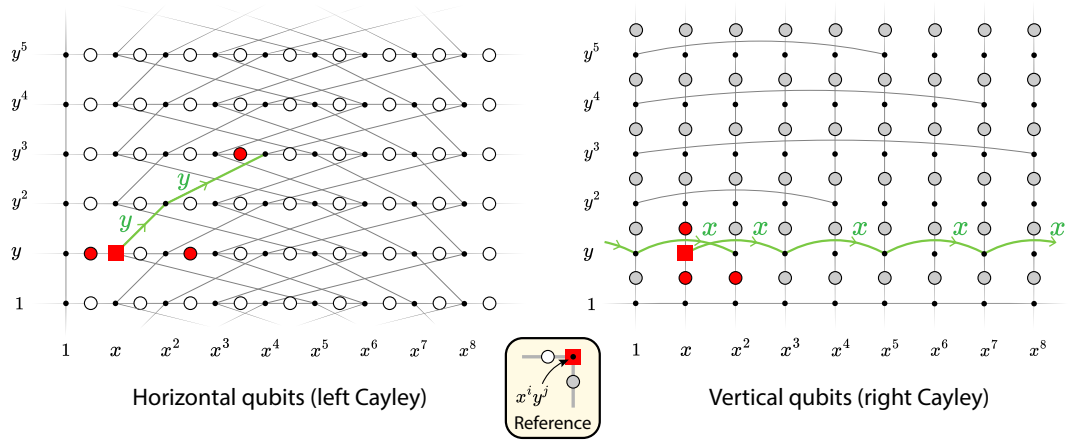


FIG. 2. The underlying structure of ZSZ$(9, 6, 2; a, b)$ with polynomials $a = 1 + x^2 + y^2$ and $b = 1 + x^5 + y$ is depicted. Horizontal qubits (white circles) are indexed by the coordinates of the black dots to their right, while vertical qubits (gray circles) are indexed by the coordinates of the black dots above them. **Left:** The horizontal support (red circles) of an X-check (red square) at location xy is determined by walks on the left Cayley graph (gray lines) according to the monomials in a; the path associated with y^2 is explicitly drawn in green. **Right:** The vertical support follows similarly but according to the right Cayley graph and the monomials in b. A single horizontal edge is drawn within each row, and the other edges are given by its translations. For Z-checks (not shown), the roles of horizontal and vertical qubits are exchanged.

the qubits) is $g_1^{r_1} \cdots g_p^{r_p} h$ with $r = r_1 + \cdots + r_p$. The number of choices of $(r_1, \ldots, r_p)$ scales as Eq. (11). In contrast, for a ZSZ code, we can have

$$|B_r(h)| = \exp[\Omega(r)]. \tag{12}$$

This can be seen by explicit construction with $q = 2$. Given generators x and y alone and any integer $J = \sum_\nu J_\nu 2^\nu$ where $J_\nu \in \{0, 1\}$ denotes the νth digit of J in its binary representation, we can express

$$x^J = x^{J_0} y x^{J_1} \cdots x^{J_{R-1}} y x^{J_R} y^{-R}, \tag{13}$$

where $R = O(\log J) \leq r/3$ is the number of binary digits that we needed. Since for any given R there are 2^R distinct values for J, clearly we can reach the number of group elements given by Eq. (12) in the desired number of steps.

Equation (12) is desirable because it suggests that the Cayley graph exhibits small-set expansion—the number of vertices at the boundary of small subsets is proportional to the volume. Codes whose Cayley graphs are expanding may exhibit linear confinement, which is sufficient to realize single-shot [58] and passive error correction [59,60]. Unfortunately, we show in Appendix B 1 that ZSZ Cayley graphs have a constant girth, in contrast to Ramanujan graphs [61] that have logarithmic girth. However, girth only tells us about the maximal degree of confinement, and linear confinement typically persists well beyond the girth [62,63]. The parametric improvement of Eq. (12) over Eq. (11) suggests that ZSZ codes may have much better performance under single-shot or autonomous decoders, relative to BB codes, a key feature which we will confirm to be the case in extensive simulations in Sec. IV.

For LDPC codes lacking an extensive number of redundant parity checks, like BB and ZSZ codes, we note that all known examples with self-correction exhibit small-set expansion in their Tanner graphs [59,64–68].

We probabilistically search through 3-term polynomials in $\mathbb{F}_2[\mathbb{Z}_\ell \rtimes_q \mathbb{Z}_m]$ to construct both A and B according to the 2BGA prescription described in Sec. II B. The resulting ZSZ codes will hence have $n = 2\ell m$ data qubits with ℓm X-checks and ℓm Z-checks. Since the total number of CSS parity checks is n, the existence of logical qubits will be based on linear dependencies among the parity checks; for any CSS code, we generically have $k = n - \text{rank}(H_X) - \text{rank}(H_Z)$. We estimate the minimum distance using the QDistRnd package in GAP (Groups, Algorithms, and Programming) [69], which probabilistically searches for low-weight logical operators. The most promising codes that we found from our computer search are listed in Table I. In this numerical search, we did not constrain the power q to be small. As we will discuss in Sec. V, this will make it more difficult to realize the optimized code in experiments.

IV. NUMERICAL SIMULATIONS

In this section, we present several instances of ZSZ codes and numerically simulate quantum memory experiments under circuit-level depolarizing noise with three types of decoders. For all the numerical simulations, we execute the following protocol:

(1) Noiseless preparation of data qubits in $|0\rangle^{\otimes n}$.
(2) Multiple rounds of syndrome extraction, alternating between X-type and Z-type checks.

TABLE I. Candidate ZSZ codes and their parameters and displayed, sorted accordingly to numerical benchmarking against various error models and decoders. Code distances are numerically estimated using the GAP package QDISTRND [69].

Decoding	Name	$[\![n,k,d]\!]$	ℓ,m,q	A	B
d rounds	ZSZ80	$[\![80,2,\leq 10]\!]$	5,8,2	$1+x^4y^4+x^4y$	$1+x^3+x^2y^7$
	ZSZ108	$[\![108,2,\leq 12]\!]$	3,18,2	$1+x+y^3$	$1+xy+x^2y^{12}$
	ZSZ160	$[\![160,2,\leq 16]\!]$	5,16,2	$1+y^9+x^4y^{14}$	$1+xy^{13}+x^2y^{13}$
	ZSZ180	$[\![180,2,\leq 18]\!]$	3,30,2	$1+x^2+y^3$	$1+xy^{18}+xy^{19}$
	ZSZ162	$[\![162,8,\leq 10]\!]$	27,3,10	$x^8+x^{18}y+x^{25}y^2$	$x^{21}+x^{14}y+x^{16}y^2$
	ZSZ288-1	$[\![288,12,\leq 16]\!]$	24,6,5	$xy+x^3y^2+x^5y^3$	$x^{10}y^3+x^{23}y^4+x^{21}y^4$
	ZSZ360-1	$[\![360,16,\leq 20]\!]$	30,6,19	$1+x^8y^2+x^{26}y^4$	$1+x^{21}y^4+x^5y^5$
	ZSZ360-2	$[\![360,20,\leq 20]\!]$	30,6,19	$1+x^9y+x^{26}y^4$	$1+x^4y^5+x^{16}y^3$
Passive	ZSZ144-3	$[\![144,12,\leq 8]\!]$	12,6,5	$y+x^2y^2+x^8y^3$	$x^4+x^6y^4+xy^5$
	ZSZ288-2	$[\![288,12,\leq 8]\!]$	24,6,5	$1+x^{14}+x^3y^3$	$1+y+x^8y^5$
	ZSZ360-3	$[\![360,12,\leq 20]\!]$	30,6,11	$x^{12}y+x^{21}y^4+x^4y^5$	$x^4y^3+x^{20}y^3+x^{22}y^4$
	ZSZ540	$[\![540,16,\leq 12]\!]$	45,6,19	$1+x^{34}+x^4y^1$	$1+x^{38}+x^2y^5$
	ZSZ756	$[\![756,24,\leq 18]\!]$	42,9,25	$1+x^4y^4+x^6y^3$	$1+x^{36}y^5+x^{21}y^7$

(3) Transversal measurement of all data qubits in the Z basis.

For the purposes of our memory simulations, step 1 can also be interpreted as starting in the logical $|\overline{0}\rangle^{\otimes k}$ state. The number of syndrome extraction cycles in step 2 is variable and will depend on the type of memory experiment that we are trying to simulate. For each syndrome extraction cycle, we sequentially extract first the X-syndrome and then the Z-syndrome; specially scheduled syndrome extraction circuits may further reduce logical error rates than what we report. For step 3, a final noiseless Z-syndrome can be inferred from the data qubit measurement outcomes that will be used to return the system exactly to the codespace. Logical $\overline{Z}$ measurement outcomes on all logical qubits can then be read off from the corrected data qubit measurement outcomes. We declare success if all logical $\overline{Z}$ measurement outcomes are $+1$ and failure otherwise.

Since ZSZ codes are LDPC, they are amenable to single-ancilla syndrome extraction: we initialize an ancilla "syndrome" qubit for each parity check and interact the syndrome qubit with its corresponding data qubits according to the circuits in Fig. 3. While simple, the one drawback

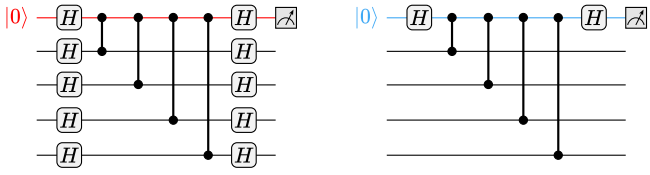


FIG. 3. Single-ancilla syndrome extraction circuits for four-qubit X-check (left) and Z-check (right) operators are shown. The top qubit denotes the syndrome qubit that is to be initialized and measured to obtain the parity of the associated check operator.

to this approach is the potential introduction of ancillary hook errors, where a single fault on the ancilla in the middle of the circuit can propagate to multiple faults on its data qubits. However, since our ZSZ codes are LDPC with check weight 6, these circuits have depth 6 and so any hook error has support on at most 3 data qubits. More precisely, all hook errors are stabilizer-equivalent to an error with weight at most 3; note that a hook error occurring at the beginning of the circuit enacts the check operator itself which is trivial since it belongs to the stabilizer group. Thus, up to constant-weight hook errors, this syndrome extraction strategy is inherently fault-tolerant for ZSZ codes. To further mitigate the effect of ancillary hook errors, one can either schedule the CZ gates in a particular order or employ flagged syndrome extraction [70] to detect the presence of hook errors at the cost of additional ancilla qubits.

For neutral atoms, the Hadamard gate can be realized by single-qubit laser pulses, and a high-fidelity CZ gate between two atoms can be realized by a blockade interaction [71,72]. A full round of single-ancilla syndrome extraction can then be performed in two stages as follows. In the first stage, we will extract the X-syndrome. Initialize $\ell m = n/2$ syndrome qubits all in $|0\rangle$ and apply a Hadamard gate to all syndrome and data qubits. Using the acousto-optic deflector (AOD) optical tweezers, move each syndrome qubit to its corresponding data qubits and apply the necessary CZ gates; the details of the AOD movements are reserved for later in Sec. V. Finally, apply Hadamard to all syndrome and data qubits. Up to physical noise, the syndrome qubits now "store" the classical X-syndrome corresponding to Z errors. To extract the Z-syndrome in the second stage, we perform a nearly equivalent procedure except that we do not apply the Hadamard gates to the data qubits.

A. Noise models and decoders

Our circuit-level noise model is parameterized by a single physical noise strength p. For simulation efficiency, we restrict to local depolarizing noise, which is a standard benchmark for preliminary analyses of new codes. Any qubit undergoing a single-qubit gate, including idling, experiences single-qubit depolarizing noise with probability $p/10$: one of the three Paulis is applied randomly. Two qubits undergoing a two-qubit gate experience two-qubit depolarization with probability p: one of the 15 two-qubit Paulis is applied randomly. Ancilla qubits are incorrectly measured and incorrectly reset with probability p. This noise model closely resembles the actual physical noise observed in recent hardware experiments involving trapped ions [73] and neutral atoms [74]. We use the Python package STIM [75] to perform all circuit-level noise simulations in this work.

The first simulation that we consider involves d syndrome extraction cycles, followed by global decoding of all $d+1$ measured Z-syndromes, including the final noiseless syndrome. We employ a belief-propagation and ordered-statistics decoder (BP+OSD) [76], which performs local message-passing on a decoding (hyper)graph before inferring a correction [77]. BP+OSD has been previously demonstrated to achieve good performance across a breadth of qLDPC codes [31,78,79], and as such is widely expected to be a general-purpose decoder for all qLDPC codes. We construct our decoding graph by laying out $d+1$ copies of the code's Z-Tanner graph and inserting "detector" qubits between equivalent syndrome nodes, which can also be interpreted as taking a homological product with a length-$(d+1)$ repetition code. For simulation efficiency, we use this "phenomenological" decoding graph rather than the larger circuit-level detector graph outputted by STIM, in essence trading some accuracy for speed. For decoding, we configure BP+OSD with 1000 maximum iterations of "min-sum" BP followed by "combination-sweep" OSD with search depth 5.

The second simulation that we consider involves ≥ 100 syndrome extraction cycles, using a single-shot, local "greedy" decoder inspired by Glauber dynamics for self-correcting memories. In the classical coding literature, this greedy decoder is more commonly known as the "flip" decoder [62,80]: local X-corrections are greedily applied one qubit at a time to lower the Z-syndrome weight. From the physics perspective, the greedy decoder can be viewed as a zero-temperature Gibbs sampler that locally tries to minimize the energy (syndrome weight). Noise that corrupts the output of the greedy decoder, such as an incorrect input due to syndrome measurement errors or the decoding algorithm being imperfect itself, can all be accounted for by "raising the temperature"; see Appendix C 1 for details. For certain classical and quantum codes, one can leverage the slow mixing time of low-temperature Gibbs sampling [67,68,81] to bound the performance of the greedy decoder [37,59,82,83]. In addition, if the code is LDPC, one can "sweep" through all data qubits in constant time. After every syndrome extraction cycle, we sweep through all data qubits once and apply the greedy decoder to the Z-syndromes for each qubit, corrupting its output with probability p: if the decoder outputs an X-correction, we do not apply it with probability p and vice versa. This last error is to approximate the situation where we perform "passive" (measurement-free) error correction by implementing one sweep of the greedy decoder as a constant-depth noisy circuit within the quantum computer itself [84]; note that we still include circuit-level noise during the syndrome extraction circuit as this step is unchanged from before. Details regarding passive error correction and its implementation can be found in Appendix D. After the final transversal Z-measurement, we decode the final syndrome with belief-propagation and localized statistics decoding (BP+LSD), a variant of BP+OSD that reduces the runtime of OSD for large LDPC codes by leveraging the percolation structure of local errors [85]. We configure BP+LSD with 1000 maximum iterations of "min-sum" BP followed by order-5 "combination-sweep" LSD.

Both simulations do not make use of the possibility for erasure checks—the position-resolved detection of qubit leakage due to physical errors—as has been demonstrated in superconducting qubits and neutral-atom arrays [12,86–93]. Numerical studies have affirmed that the information garnered from erasures can significantly improve thresholds, particularly when detected mid-circuit and even as a "delayed" erasure [86,94–96]. While the ZSZ code already localizes its information to a degree to enable passive decoding, we expect that further gains in performance should be achievable with the use of erasures. We leave these numerical investigations to future work.

B. Performance

The simulation results for d rounds of syndrome extractions and global decoding are displayed in Fig. 4. We plot both the logical block error rate (BLER), in other words the probability that any logical qubit is measured incorrectly, as well as the logical error rate for any single logical qubit. For a code block encoding k logical qubits with BLER $\overline{p}$, we estimate the single-logical-qubit error rate as $\overline{p}_1 = 1-(1-\overline{p})^{1/k}$. We also simulate the performance of the $[\![144,12,12]\!]$ (BB144) and $[\![288,12,18]\!]$ (BB288) BB codes from Ref. [31] under the same BP+OSD decoder as well as several (unrotated) 2D surface codes decoded with minimum-weight perfect-matching (MWPM), using the PYMATCHING package in Python.

We observe an intersection of the ZSZ BLER curves around $p \approx 0.5\%$, below which the BLER falls exponentially with the code size. This exponential decay is indicative of subthreshold behavior where we would generically expect $\overline{p} \sim p^{-\Theta(d)}$. Thus, we optimistically declare

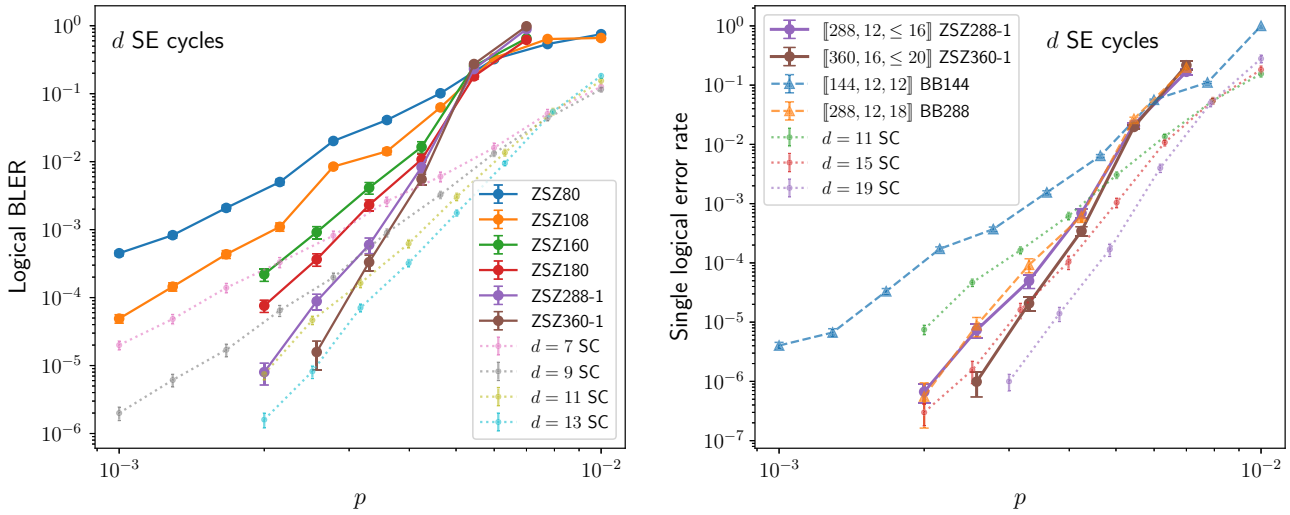


FIG. 4. Numerical simulation results are displayed for global decoding over d syndrome extraction cycles. **Left:** The logical block error rate is plotted as a function of the physical noise strength p for a handful of ZSZ codes in Table I under "d rounds" decoding. For comparison, we also plot the performance of the (unrotated) surface code (dotted curves). **Right:** We normalize the logical block error rate by the number of logical qubits to estimate the logical error rate of a single logical qubit, which is plotted as a function of p for two high-rate ZSZ codes. For comparison, we also plot the performance of the surface code (dotted) as well as two bivariate bicycle codes (dashed) from Ref. [31]. Uncertainties are given by the standard error.

$p_{\text{th}} \approx 0.5\%$ as the threshold of ZSZ codes under our choice of decoder and circuit-level noise model. For comparison, the threshold of the 2D surface code with MWPM decoding under the same noise model is observed to be approximately 0.8% [97]. Interestingly enough, we also see that some higher-distance ZSZ codes, such as ZSZ180 with estimated code distance 18, have a worse BLER performance than lower-distance ZSZ codes, such as ZSZ288-1 with estimated code distance 16. We attribute this discrepancy to ancillary hook errors during syndrome extraction: a single fault on the ancilla can propagate to multiple data qubits and lower the total *fault distance* of the code. For LDPC codes, this propagation is a constant and so is unimportant from an asymptotic standpoint, but for small-to-moderate length codes this constant can matter. Our ZSZ codes have check weight 6, and so the weight of any hook error is at most 3. Note that (unrotated) surface codes do not suffer from this issue [98]. When we normalize by the number of logical qubits and plot the single logical error rate, we notice that the gap between the slopes of the LER curves of the high-rate ZSZ codes (ZSZ288-1 and ZSZ360-1) and the surface codes closes. In particular, the subthreshold LER slope of ZSZ288-1 closely follows that of BB288 as well as the distance-15 surface code, and the LER slope of ZSZ360-1 closely follows that of the distance-19 surface code. Although there remains a factor of 5–10× difference between ZSZ codes and surface codes with similar distances, ZSZ codes achieve significantly lower overhead compared to surface codes for comparable logical error performance. For instance, at physical error rate $p \approx 3 \times 10^{-3}$, the curves of ZSZ360-1 and distance-15 surface code intersect. Since the overheads of ZSZ360-1 and distance-15 surface code are 22.5 and 225 respectively, there is roughly a 10× saving in qubit overhead for ZSZ codes.

The simulation results for ≥ 100 rounds of syndrome extraction and passive decoding are displayed in Fig. 5. Similar to the previous situation with d rounds of decoding, we observe subthreshold behavior below a noise strength $p \approx 0.1\%$ when decoding a cumulative of 100 syndrome extraction cycles. A more detailed numerical analysis near the observed crossing point can be found in Appendix E 1. In order to rule out finite-size effects, we also fix the noise strength p and vary the number of syndrome extraction cycles. We see that the normalized logical error rates *per cycle* for our ZSZ codes stabilize at $\gtrsim 100$ syndrome extraction cycles, suggesting that our numerics at 100 cycles is a good indicator of the asymptotic behavior at long times. As such, we optimistically declare $p_{\text{th}} \approx 0.1\%$ as the *sustainable* threshold of ZSZ codes under our model of passive decoding. For comparison, we perform the same simulations for the 4D toric code and estimate its sustainable threshold under passive decoding to be approximately 0.06%; see Appendix E 1 for details. In particular, we see that ZSZ756 outperforms four copies of the $L = 4$ 4D toric code ($[\![6144, 24, 16]\!]$) while using only $\approx 12\%$ of the number of data qubits for the same number of logical qubits. We also simulate several instances of BB codes and do not observe any evidence of a sustainable threshold for passive decoding, suggesting that ZSZ codes and BB codes

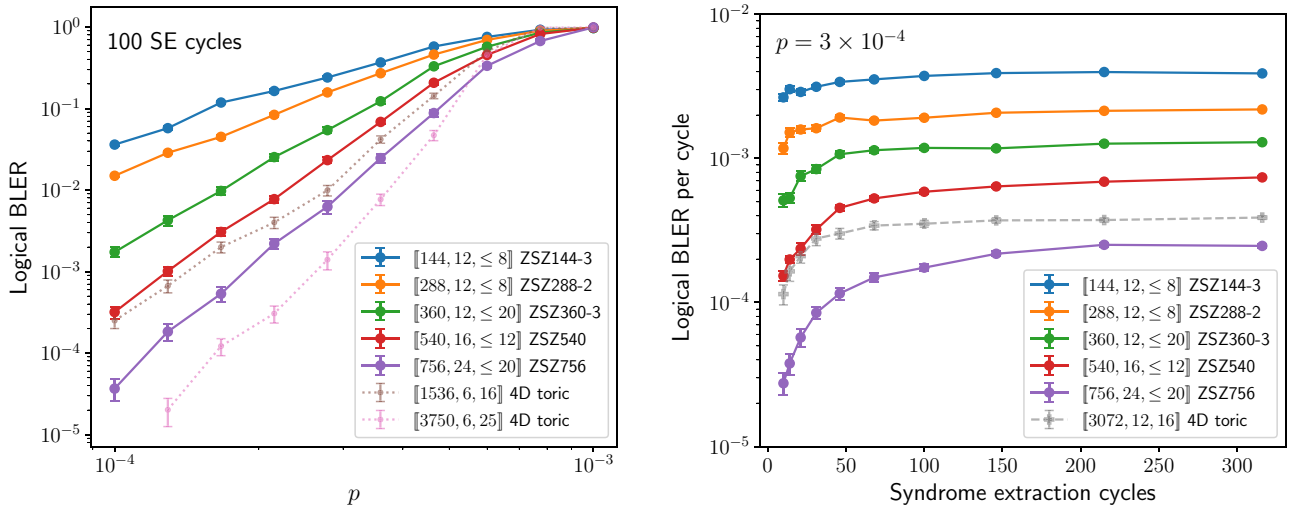


FIG. 5. Numerical simulation results are displayed for noisy local (passive) decoding over many syndrome extraction cycles. **Left:** The logical block error rate over 100 syndrome extraction cycles is plotted as a function of the physical noise strength p for the ZSZ codes in Table I under passive decoding. **Right:** The logical block error rate per syndrome extraction cycle with fixed $p = 3 \times 10^{-4}$ is plotted as a function of the total number of syndrome extraction cycles. For comparison, we also plot the performance of the 4D toric code with linear sizes $L = 4, 5$. Uncertainties are given by the standard error.

can have drastically different qualitative behaviors at moderate code lengths; see Appendix E 2 for details on the BB code simulations.

V. IMPLEMENTATION IN NEUTRAL ATOM ARRAYS

In this section we discuss the implementation of ZSZ codes on the neutral atom platform. In this platform, each physical qubit is encoded in two long-lived states of a neutral atom [99–104]. The atoms are optically trapped by a spatial light modulator (SLM), and their positions can be rearranged using AOD optical tweezers [9,103,105]. Typically, an SLM is used to generate a static two-dimensional array of traps, and AODs are used to transfer atoms between different SLM traps to facilitate long-range gates [9]. A typical AOD reconfiguration step, or "grid transfer," consists of picking up a subgrid of qubits from static SLM traps onto dynamical AOD traps, moving the AOD traps to unoccupied SLM traps and then dropping off the atoms onto these SLM traps. Importantly, parallel AOD rows and columns should not intersect to minimize disturbance of static qubits; in other words, the ordering of atoms participating in an AOD move should remain unchanged when moves are constrained to a line. For example, Fig. 6 illustrates how a "riffle shuffle" permutation of atoms arranged in a 1D line can be executed with a constant number of AOD moves and auxiliary SLM traps. This riffle shuffle is

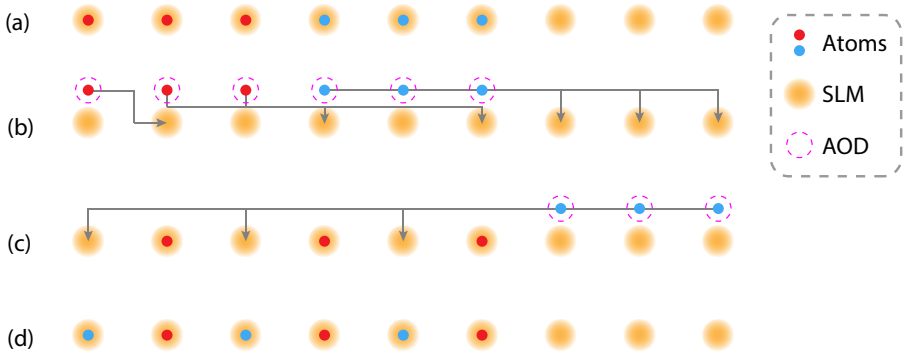


FIG. 6. A step-by-step visualization of a 1D riffle shuffle of N atoms (6 shown) using $\lfloor N/2 \rfloor$ auxiliary spatial light modulator traps. (a) The initial configuration of atoms. (b) A subset of atoms (blue) is transferred by acousto-optic deflectors to the auxiliary spatial light modulator traps, and the remaining atoms (red) are moved to some order-preserving subset of the original N sites, in this case the even sites. (c) The blue atoms are then transferred back to the remaining unoccupied spatial light modulator traps of the initial N sites. (d) The final configuration of the atoms is displayed.

ALGORITHM 1. ZSZ left-action routing with grid transfers.

Input : ZSZ parameters (ℓ, m, q) and a polynomial $x^\alpha y^\beta$
Return : A permutation of an $\ell \times m$ rectangular lattice of sites
// y^β part, requiring $\lfloor m/2 \rfloor$ additional rows above and $\lfloor \ell/2 \rfloor$ columns on the right for scratch space
$\sigma = [q^\beta i \,\%\, \ell$ **for** $i = 0$ **to** $\ell - 1]$
Apply a suitable 1D permutation algorithm (e.g. Algorithm 1 of [101]) to implement σ on all ℓ columns, ordered from left to right.
$\beta' = \min(\beta, m - \beta)$
if $\beta' == \beta$ **then**
  Shift the entire lattice upwards by β' units.
  Pick up the top β' rows of atoms and shift them m units downwards to the bottom.
else
  Pick up the bottom β' rows of atoms and shift them m units upwards to the top.
  Shift the entire lattice downwards by β' units.
end
// x^α part, requiring $\lfloor \ell/2 \rfloor$ additional columns for scratch space
$\alpha' = \min(\alpha, \ell - \alpha)$
if $\alpha' == \alpha$ **then**
  Shift the entire lattice to the right by α' units.
  Pick up the rightmost α' columns of atoms and shift them ℓ units to the left side.
else
  Pick up the leftmost α' columns of atoms and shift them ℓ units to the right side.
  Shift the entire lattice to the left by α' units.
end

a key subroutine of the 1D permutation algorithm by Xu *et al.* [106], which can implement any 1D permutation of N atoms using $\log_2 N$ riffle shuffles and $\lfloor N/2 \rfloor$ additional SLMs for scratch space.

Recall that the semidirect product structure of a ZSZ code naturally gives rise to an $\ell \times m$ rectangular embedding (Fig. 2) where the horizontal axis labels the group elements of the normal subgroup $\mathbb{Z}_\ell$ and the vertical axis those of $\mathbb{Z}_m$. We propose a single-ancilla syndrome extraction strategy, where we initialize a syndrome qubit for every parity check and couple the syndrome qubit to its corresponding data qubits. After data-syndrome coupling, we can then either measure the syndrome qubits to extract the classical syndrome bitstring or further process the syndrome qubits for passive decoding. We will now show that a complete round of X-syndrome extraction can be performed using $O(\log \ell)$ AOD moves for the horizontal sector and $O(m)$ moves for the vertical sector. The Z-syndrome extraction proceeds analogously but using the transpose matrices (i.e., negating the exponents) as well as interchanging the roles of the left and right Cayley graphs, with the same routing complexity.

To extract the X-syndrome, we proceed in two steps. First, we couple the syndrome and horizontal data qubits given by the polynomial a, which corresponds to routing on the ZSZ group's left Cayley graph. For each monomial $x^\alpha y^\beta$ in a, we can implement the y^β part using a combination of a vertical cyclic shift of all rows and a 1D horizontal "grid-type" permutation of all columns. The number of AOD moves required for this part is generically $O(\log \ell)$, stemming from the 1D permutation complexity. The x^α part can be implemented with a simple horizontal cyclic shift; see Algorithm 1 for the explicit steps and Fig. 7(a) for a visualization. Hence, the total routing complexity to couple the horizontal data qubits is $O(\log \ell)$. Second, we couple the syndrome and vertical data qubits given by the polynomial b. For each monomial $x^\alpha y^\beta$ in b, we can implement the x^α part using horizontal cyclic shifts among the rows; each row will correspond to a distinct cyclic shift given by Eq. (10a). Using grid transfers and n additional SLM traps as scratch space on the right side, we can perform these cyclic shifts as follows. Using AODs, pick up the entire grid of n syndrome qubits and translate it horizontally, dropping off each row into SLM traps according to their respective shift amount. Then, pick up all the atoms in the scratch space and merge it on the left side of the remaining atoms using additional grid transfers. The number of AOD moves required for this part is $O(m)$, since the rows needs to be sequentially dropped off. Finally, we can implement the y^β part of b with a global vertical cyclic shift. See Algorithm 2 for the explicit steps and Fig. 7(b) for a visualization.

To summarize, for syndrome extraction of a ZSZ code with $n = 2\ell m$ data qubits, we require $O(\log \ell)$ grid transfers for routing on the left Cayley graph and $O(m)$ grid transfers on the right Cayley graph. For the ZSZ codes with logarithmic diameter, their dimensions satisfy $m = O(\log \ell)$ which gives a routing complexity of $O(\log \ell)$ for the vertical sector, the same as that of the horizontal sector. Note that all the candidate ZSZ codes for passive decoding in Table I have small m relative to ℓ. For comparison, a complete round of syndrome extraction has an

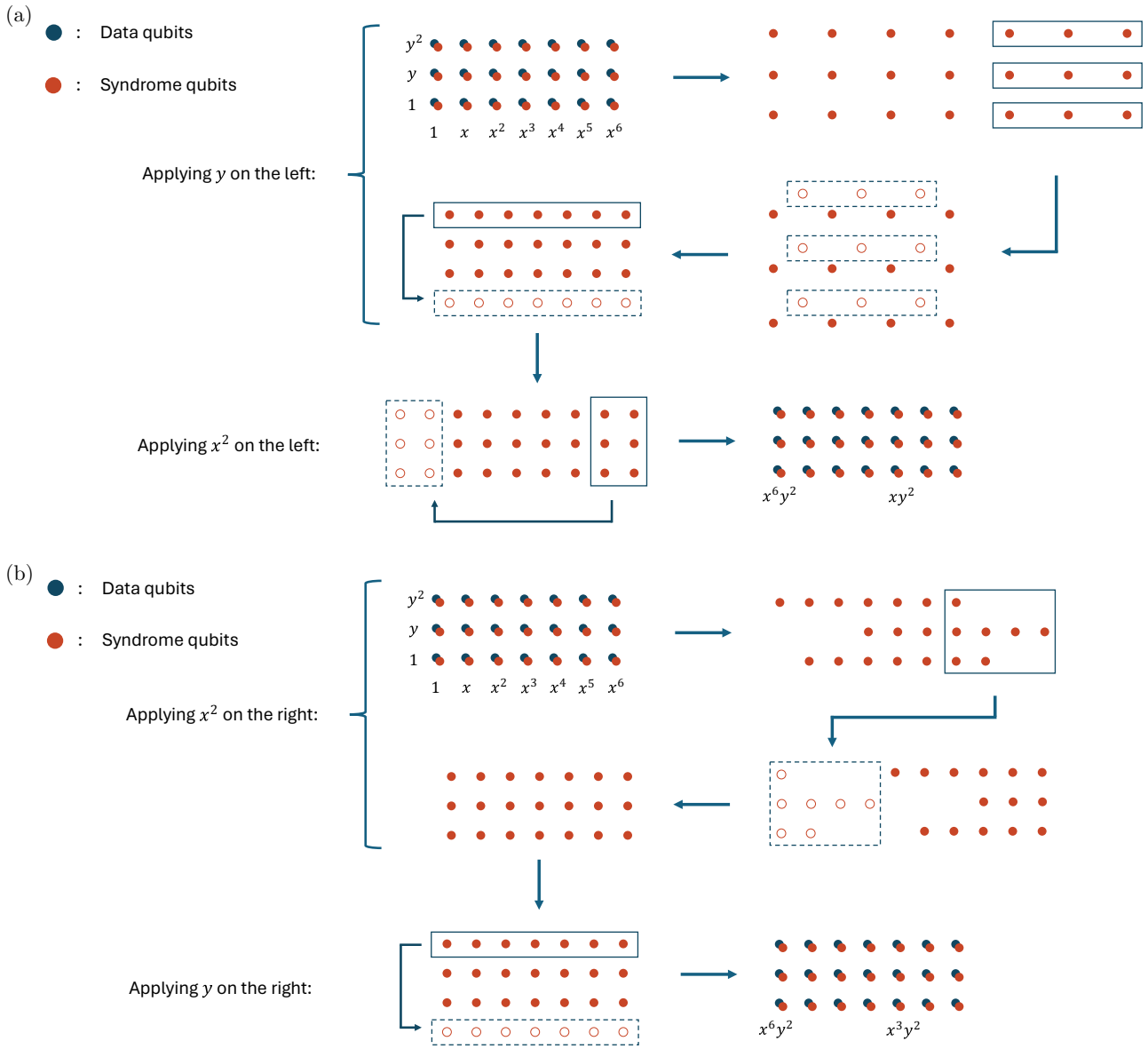


FIG. 7. The acousto-optic deflector moves required for implementing the monomial x^2y on the horizontal data qubits for the group $\mathbb{Z}_7 \rtimes_2 \mathbb{Z}_3$ are depicted. The blue dots represent the horizontal data qubits, and the orange dots represent the syndrome qubits. (a) Applying $x^2y \in a$ on the left. The first and second steps illustrate the grid-type permutations (riffle shuffles) required for implementing y, and the later steps illustrate the vertical and horizontal cyclic shifts for the rest. In the last frame, the previous locations of two ancilla qubits are labeled. Applying x^2y to their previous locations gives the present locations. (b) Applying $x^2y \in b$ on the right. The first and second steps illustrate the distance shift for each row for implementing x^2. The last row is the horizontal cyclic shift for implementing y. The previous locations of two ancilla qubits are labeled.

(AOD) routing complexity of $O(1)$ for BB codes since all the required couplings can be realized in parallel with cyclic shifts [32], similar to the case with the surface code. For generic hypergraph product codes, a complete round of syndrome extraction has $O(\log n)$ routing complexity [106], coming from the 1D permutation complexity of rearranging rows and columns. So for ZSZ codes with $m = O(\log \ell)$, their routing complexity for syndrome extraction is the same as that for hypergraph product codes, but not as fast as that of BB and surface codes.

Constantinides *et al.* [107] proposed a hardware upgrade that allows for "selective transfers": within a subgrid formed by AODs, we can select an arbitrary subset of atoms to be left behind during the grid transfer. The process involves deepening the SLM trap potential on the selected sites so that the atoms are not picked up by the AOD

ALGORITHM 2. ZSZ right-action routing with grid transfers.

Input : ZSZ parameters (ℓ, m, q) and a polynomial $x^\alpha y^\beta$
Return : A permutation of an $\ell \times m$ rectangular lattice of sites
// x^α part, requiring ℓ additional columns on the right for scratch space
$S = [q^j \alpha \,\%\, \ell$ **for** $j = 0$ **to** $m - 1]$
$S', I = \mathrm{Sort}(S)$ // Sorted with index set I
$\Lambda \leftarrow$ the entire lattice
for $i = 1$ **to** $m - 1$ **do**
  Shift Λ to the right by $S'[i]$ units.
  Drop off row $I[i]$.
  $\Lambda \leftarrow \Lambda -$ row $I[i]$
end
Pick up the ℓ columns of scratch space and shift them ℓ units to the left side.
// y^β part, requiring $\lfloor m/2 \rfloor$ additional rows above for scratch space.
$\beta' = \min(\beta, m - \beta)$
if $\beta' == \beta$ **then**
  Shift the entire lattice upwards by β' units.
  Pick up the top β' rows of atoms and shift them m units downwards to the bottom.
else
  Pick up the bottom β' rows of atoms and shift them m units upwards to the top.
  Shift the entire lattice downwards by β' units.
end

tweezers. They show that, using these selective transfers, arbitrary permutations of the sites on a 2D lattice have routing complexity $O(\log n)$. Since their result encompasses arbitrary permutations of the atoms, it applies to the ZSZ codes, and so the routing complexity of syndrome extraction with these selective transfers becomes $O(\log n)$ irrespective of ℓ and m.

We note that in our numerical simulations in Sec. IV, we did not consider sources of error during the rearrangement of the atoms, where heating, atom loss, and positional disorder in the AOD tweezer traps can also increase overall error rates. These errors become increasingly relevant when the system size grows as the rearrangement process becomes more complex and involves longer transport distances and more number of atom moves. Including this movement time would cause our syndrome extraction circuits to no longer be constant depth, and if errors are introduced during movement, would lead to a loss of a threshold in the thermodynamic limit $L \to \infty$ [106]. We leave a detailed study of trade-offs in move distance and code distance to future investigations.

VI. DISCUSSION AND OUTLOOK

In this work, we introduced ZSZ codes as a nonabelian generalization of bivariate bicycle codes. We proposed a two-dimensional rectangular layout that facilitates the movement of ancilla qubits for syndrome extraction in neutral atom platforms. We then simulated memory experiments under a circuit-level depolarizing noise model using both conventional decoding with mid-circuit measurements as well as a local measurement-free scheme inspired by self-correcting memories. We observed numerical evidence of a sustainable threshold to the measurement-free scheme and found that it is higher than that of the 4D toric code under the same noise model and decoder. Finally, we described how to implement a complete round of syndrome extraction in neutral atom platforms with optical-tweezer arrays.

There are several directions for future work that could improve both the ZSZ codes' decoding performance as well as their implementation in a fault-tolerant architecture. To start, it would be immediately beneficial if one can parallelize the X-syndrome and Z-syndrome extraction circuits to reduce the overall depth of syndrome extraction. A lower syndrome extraction depth would mean less noise accumulation for the data qubits due to idling as well as a faster QEC cycle time. It would also be practically relevant to optimize the CZ scheduling in order to mitigate the effects of hook errors, as has been done for the bivariate bicycle codes [31]. In addition, for the measurement-free simulations, we only used a phenomenological noise model for the local decoding circuit. It would be more accurate to account for circuit-level noise in the local decoder, and this would most likely involve state-vector simulations due to the decoding circuit involving non-Clifford gates; one can also include noise during the atom routing process such as from heating, as has been done in Ref. [106]. Alongside circuit-level optimizations, it would also be interesting to explore other potential local decoding strategies, such as if local cellular automaton decoders for topological codes [108,109] can be adapted to ZSZ codes. For the 4D toric code, it has been numerically observed that a local sweep rule outperforms the local majority vote in terms of both a higher threshold and lower logical error rates [110].

Although we have benchmarked the memory performance of ZSZ codes, there are several considerations that we have glanced over, which would be crucial to performing encoded logic. First, in our numerical simulations, we have assumed that the logical codespace has already been prepared fault-tolerantly. In an actual platform, we will typically need to prepare the codespace starting from a product state. The usual method for fault-tolerant state preparation of CSS codes works here: to initialize the logical $|\bar{0}\rangle/|\overline{+}\rangle$ state, we initialize all data qubits in $|0\rangle$ and perform d rounds of syndrome measurement. Using spacetime mappings, one can also reduce this depth to $O(1)$ at the cost of $O(d)$ additional ancilla qubits [111–113] per data qubit. For some single-shot codes such as the 4D toric code, the local dependencies among the check operators enable the constant depth without the additional ancillas [114,115]. Since we discuss measurement-free error correction, it would be practical to also have a measurement-free version of state preparation. However, we are not yet aware of such a protocol, and progress along this direction would enable the full memory experiment to be measurement-free, other than the final readout measurement of all data qubits. Slow state preparation is okay if the purpose of the ZSZ code block is solely for memory and not computation, as we would in principle only need to prepare the codespace once at the beginning. However, if we are using state preparation as a subroutine for logical computation (e.g., in Ref. [115]), then its speed may bottleneck the otherwise fast (single-shot) QEC cycles. In addition to logical state preparation, we would also require fault-tolerant gadgets to perform logical computation. Like for the BB codes, we can construct ancillary systems [25,27–29,31] that can measure arbitrary combinations of logical Pauli operators within a code block or between different code blocks to realize the full logical Clifford group [30]. Note that, unlike the BB codes, we cannot leverage translational symmetries (shift automorphisms) to reduce the size of this ancillary system [29]. Furthermore, these "lattice surgery" schemes generically require $\Theta(d)$ rounds of measurements and feedback for fault tolerance [116], and it is not yet known whether a measurement-free version exists.

ACKNOWLEDGMENTS

We thank Ali Fahimniya for enlightening discussions on neutral atom routing. This work was supported by the Air Force Office of Scientific Research under Grant No. FA9550-24-1-0120 (J.G., A.L.), the Office of Naval Research under Grant No. N00014-23-1-2533 (J.G., A.K., A.L.), the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR00112490357 (Y.H.), and the DoE ASCR Quantum Testbed Pathfinder program under Award Nos. DE-SC0019040 and DE-SC0024220 (Y.H.).

DATA AVAILABILITY

The data that support the findings of this article are openly available [117].

APPENDIX A: CLASSICAL GROUP CODES

A classical linear code $\mathcal{C}$ encoding k logical bits among n physical bits is described by a k-dimensional subspace of $\mathbb{F}_2^n$, the vector space of all length-n binary bitstrings. The 2^k bitstrings in this subspace are called logical codewords, and the code distance d is defined as the minimum Hamming weight (i.e., number of ones) among all $2^k - 1$ nonzero codewords. These three parameters are often packaged using the notation $[n, k, d]$. Because $\mathcal{C}$ is linear, we can choose k codewords to form a basis for this vector subspace, which we can organize into a generator matrix $G \in \mathbb{F}_2^{k\times n}$ that succinctly describes $\mathcal{C}$. Using the dual vector space, we can equivalently describe $\mathcal{C}$ with a parity-check matrix $H \in \mathbb{F}^{m\times n}$ whose rows annihilate those of G, and the code is thereby defined as $\mathcal{C} = \ker H$. If the row and column weights of H are bounded by constants independent of n, then we say that H (and likewise $\mathcal{C}$) is a low-density parity-check (LDPC) code.

1. Group theory

We begin by reviewing some relevant group theory knowledge, the details of which can be found in textbooks [118].

Definition A1 (Cyclic group). The cyclic group $\mathbb{Z}_n$ obeys the presentation $\mathbb{Z}_n = \langle x | x^n = 1\rangle$. It is an abelian group and, if n is prime, has no nontrivial subgroups.

Proposition A1. The automorphism group of $\mathbb{Z}_n$, denoted $\mathrm{Aut}(\mathbb{Z}_n)$, which corresponds to the group of all isomorphisms from $\mathbb{Z}_n$ into itself, is isomorphic to $\mathbb{Z}_n^\times$, which is the multiplication group of nonzero integers mod n, restricted to the integers coprime to n.

Definition A2 (Regular representation). Given a finite group G of order $|G| = n$, its *left-regular representation* is the set of $n \times n$ permutation matrices $\{L_i\}, i = 1, \ldots, n$ where the basis runs over the group elements, and $(L_i)_{jk} = 1$ if and only if $g_j = g_i g_k$. The right-regular representation $(R_i)_{jk}$ is defined analogously using the right-multiplication rule $g_j = g_k g_i$.

As an example, the regular representation of the cyclic group $\mathbb{Z}_n$ consists of circulant matrices $(L_i)_{jk} = \delta_{j,i+k}$.

Definition A3 (Commutator subgroup). For a group G, its *commutator subgroup* or *derived subgroup* is defined as

$$[G, G] := \langle [g, h] = g^{-1}h^{-1}gh \mid g, h \in G\rangle. \tag{A1}$$

Definition A4 (Derived series). For a group G, its *derived series* is a sequence of groups

$$G = G^{(0)} \triangleright G^{(1)} \triangleright G^{(2)}, \triangleright \cdots, \tag{A2}$$

where $G^{(i)} := [G^{(i-1)}, G^{(i-1)}]$ is normal in $G^{(i-1)}$. The smallest l such that $G^{(l)} = 1$ is trivial is known as the *derived length* of G.

As a simple example, the derived length for any abelian group is 1 since its commutator subgroup is trivial. The notion of derived length will later prove useful when we analyze the expansion of Cayley graphs.

2. Cayley graphs and group-algebra codes

In this Appendix, we present a geometrical perspective on classical group-algebra codes, which will aid in later arguments.

Definition A5 (Cayley graph). Given a group G and a set of generators $S \subset G$, the *left Cayley graph* $\mathcal{G}(G,S) = (V,E)$ is a directed graph whose vertices $v_i \in V$ correspond to group elements $g_i \in G$, and $(v_i, v_j) \in E$ if and only if $g_j = sg_i$ for some $s \in S$. The *right Cayley graph* is similarly defined but with the condition $g_j = g_i s$ instead.

Note that if $\mathbb{1} \in S$, then G contains a self-loop at each vertex.

Definition A6 (Graph double cover). Given a graph $\mathcal{G} = (V,E)$, the *double cover* of $\mathcal{G}$ is the (possibly directed) bipartite graph $\bar{\mathcal{G}} = (V_1 \cup V_2, E_1 \cup E_2) = \mathcal{G} \times K_2$, where K_2 is the complete graph on two vertices. Specifically, V_1 and V_2 are copies of V, $(a_i \in V_1, b_j \in V_2) \in E_1$ if and only if $(v_i \in V, v_j \in V) \in E$, and $(b_j \in V_2, a_i \in V_1) \in E_2$ if and only if $(v_j, v_i) \in E$.

In other words, V_1 and V_2 partition the left and right vertices respectively. Similarly, E_1 and E_2 partition the left-emanating and right-emanating edges. If the underlying base graph $\mathcal{G}$ is undirected, then $E_1 = E_2$. Since $|V_1| = |V_2| = |V|$, the double cover $\bar{\mathcal{G}}$ is a balanced graph.

Definition A7 (Group-algebra code; geometric interpretation). Given a finite group G with $|G| = n$ and a set of generators S, the parity-check matrix of the *left group-algebra code* is defined as the biadjacency matrix of $\bar{\mathcal{G}}_2 = (V_1 \cup V_2, E_2) \subset \bar{\mathcal{G}}$, the double cover of the left Cayley graph $\mathcal{G}(G,S)$ with only right-emanating edges. Left and right vertices are mapped to bits and checks respectively. The parity-check matrix $H \in \mathbb{F}_2^{n\times n}$ is defined by

$$H = \sum_{s\in S} L[s], \tag{A3}$$

where $L[s] \in \mathbb{F}_2^{n\times n}$ denotes the left-regular representation of s. The right group-algebra code is defined analogously using right actions.

Note that any parity-check matrix associated with, say the left-regular representation of, an element of $\mathbb{F}_2[G]$ can be put in the form $H = L'\left(\mathbb{1} + \sum_i L_i\right)$, which is equivalent to $H' = \mathbb{1} + \sum_i L_i$ up to the global permutation (automorphism) L'. H' is now the parity-check matrix of the Cayley graph code of $\mathcal{G}(G, \mathbb{1} \cup \{S_i\})$ per the above definitions.

We now review some basic properties of a group-algebra code. Since the left and right degrees of a Cayley graph's double cover are both equal to the number of generators $|S|$, if $|S|$ is constant, then the associated group-algebra code is LDPC. Since the double cover has an equal number of bits and checks, the parity-check matrix of our Cayley graph code is square. Hence, the logical code dimension will be determined by the rank deficiency corresponding to linear dependencies among the parity checks.

3. Classical ZSZ codes

Definition A8 (Semidirect product of two cyclic groups). Let ℓ, m be coprime integers and $q^m = 1 \pmod{\ell}$. The semidirect product of two cyclic groups can be presented as

$$\mathbb{Z}_\ell \rtimes_q \mathbb{Z}_m := \langle x, y | x^\ell = y^m = yxy^{-1}x^{-q} = 1 \rangle. \tag{A4}$$

Definition A9 (Classical ZSZ code). We define a *classical ZSZ code* to be a group-algebra code (Definition A7) formed from the group $\mathbb{Z}_\ell \rtimes_q \mathbb{Z}_m$ with parity-check matrix described by

$$H = \sum_{i=1}^{q} \mathbb{B}\left[x^{a_i} y^{b_i}\right], \tag{A5}$$

where $\mathbb{B}[\cdot]$ can denote either the left-regular or right-regular representation.

APPENDIX B: QUANTUM 2BGA CODES

This appendix contains the relevant details for 2BGA code properties mentioned in the main text.

Definition B1 (2BGA code). Let $A, B \in \mathbb{F}_2^{n\times n}$ be the parity-check matrices of a left and a right group-algebra code (Definition A7) respectively based on the same group

G of order n. Then the CSS parity-check matrices of the quantum 2BGA code are given by

$$H_X = \left(A \mid B\right), \tag{B1a}$$

$$H_Z = \left(B^\mathsf{T} \mid A^\mathsf{T}\right). \tag{B1b}$$

From Eq. (B1), it is clear that the check weights of a quantum 2BGA code are simply the sum of the check weights of the two component classical codes. The orthogonality of H_X and H_Z follows from the commutativity of A and B, which follows from the associativity of group multiplication. Equation (B1) can also be interpreted as a tensor (hypergraph) product between A and B followed by factoring out the "diagonal" action $G \times G \to G$ [119].

1. Girth

In this Appendix, we provide upper bounds on the girths of the Tanner graphs of abelian and nonabelian 2BGA codes. Recall that a parity-check matrix H generates the Tanner graph of the corresponding code. The Tanner graph is a type of bipartite factor graph that depicts how qubit nodes and check nodes are connected. Define the qubit X-adjacency (Z-adjacency) graphs as a n-vertex graph where vertex i and j are connected by an edge if and only if qubits i and j lie in the support of an X-check (Z-check).

Definition B2 (Graph girth). Given a simple graph $G = (V, E)$, let $\partial_{ve} \in \mathbb{F}_2^{|V|\times|E|}$ be the vertex-edge incidence matrix such that $\ker \partial_{ve}$ is the space of closed cycles or loops on G. The girth of G is then defined as

$$\mathrm{girth}(G) = \min_{c \in \ker \partial_{ve}} |c|. \tag{B2}$$

For a classical linear code $\mathcal{C}$ with parity-check matrix H, we define its code girth $\mathrm{girth}(H)$ as the girth of the bit adjacency graph corresponding to H. For a quantum CSS code, we define its girth to be the minimum girth between the qubit X-adjacency and Z-adjacency graphs.

Let us decompose the 2BGA block matrices A and B in terms of their components $A = \sum_i A_i$ and $B = \sum_i B_i$. Now let us consider the girth of the qubit X-adjacency graph induced by $H_X = \left(A \mid B\right)$. H_X naturally divides the vertices into left and right sectors. If we start from a given left vertex, corresponding to some group element, all its left neighbors can be obtained by examining the associated column in $A_i^\mathsf{T} A_j$ $(i \neq j)$ that corresponds to that group element; the right neighbors are obtained similarly but using $B_i^\mathsf{T} A_j$. Similarly, for right vertices, we examine $A_i^\mathsf{T} B_j$ and $B_i^\mathsf{T} B_j$ to get their left and right neighbors respectively.

Theorem B1 (Girth upper bound for abelian 2BGA codes). For an abelian 2BGA code with left and right sector weights at least 2, its code girth is at most 3.

Proof. For an abelian 2BGA code, recall that all component matrices A_i and B_i commute with each other. Consider the following path of length 3 on the X-type qubit adjacency graph, starting from a left vertex,

$$P = \left(A_i^\mathsf{T} B_k\right)\left(B_k^\mathsf{T} A_j\right)\left(A_j^\mathsf{T} A_i\right), \quad i \neq j. \tag{B3}$$

We read the path P above from right to left, and importantly the matrix type (A vs B) must be the same when we go between the tuples in the parentheses, corresponding to arriving at a left or right vertex and subsequently leaving from that vertex. Since all the As and Bs commute, we can rearrange the above expression to get $P = \left(B_k^\mathsf{T} B_k\right)\left(A_j^\mathsf{T} A_j\right)\left(A_i^\mathsf{T} A_i\right) = \mathbb{1}$, using the fact that each A and B is an orthogonal matrix. Thus P is a loop of length 3, and so the girth is at most 3. ■

Theorem B2 (Girth upper bound for generic 2BGA codes). For any 2BGA code with left and right sector weights at least 2, its girth is at most 4.

Proof. Consider the following path of length 4 on the X-type qubit adjacency graph, starting from a right vertex,

$$P = \left(B_l^\mathsf{T} A_j\right)\left(A_i^\mathsf{T} B_l\right)\left(B_k^\mathsf{T} A_i\right)\left(A_j^\mathsf{T} B_k\right), \quad i \neq j,\ l \neq k. \tag{B4}$$

Rearranging the parentheses in the above expression, we get $P = B_l^\mathsf{T}\left(A_j A_i^\mathsf{T}\right)\left(B_l B_k^\mathsf{T}\right)\left(A_i A_j^\mathsf{T}\right) B_k$. Now, since $A_j A_i^\mathsf{T}$ acts nontrivially solely on the left sector and likewise $B_l B_k^\mathsf{T}$ on the right sector, they commute. Hence, upon rearranging, we have $P = B_l^\mathsf{T}\left(A_j A_i^\mathsf{T}\right)\left(A_i A_j^\mathsf{T}\right)\left(B_l B_k^\mathsf{T}\right) B_k = B_l^\mathsf{T}\left(A_j A_i^\mathsf{T} A_i A_j^\mathsf{T}\right)\left(B_l B_k^\mathsf{T}\right) B_k = B_l^\mathsf{T} B_l B_k^\mathsf{T} B_k = \mathbb{1}$. Thus P is a loop of length 4, and so the girth is at most 4. ■

2. Diameter

In this Appendix, we show that the diameters of the Tanner graphs can be quite different between ZSZ codes and abelian 2BGA codes. We prove that if we fix the weight of parity checks w, we can construct families of ZSZ codes with diameter $O(\log n)$, where the diameter is with respect to the qubit adjacency graphs. In contrast, for any BB code, the diameter scales as $\Omega(n^{1/(w-1)})$.

Theorem B3 (ZSZ Tanner graphs with small diameter). Suppose the block matrices A and B have weight 3. If $\ell = q^m - 1$ and the component group elements g_1, g_2, g_3 obey $g_2 g_1^{-1} g_3 \neq g_3 g_1^{-1} g_2$, then the corresponding ZSZ code has diameter $O(\log n)$.

Proof. We will analyze the case where the Tanner graphs of A and B are connected and deal with

disconnected components afterward. In the connected case, the diameter of H_X is less than the sum of the diameters of the graphs of A and B. Our goal is to show that the diameters of A and B can be $O(\log n)$. Without loss of generality, we will only discuss the diameter of the graph of A.

Every vertex (qubit) in the left sector can be represented as $x^i y^j$, and all vertices are equivalent to one another due to the transitivity of the group. Let us start with the simplest case where $s = \{1, x, y\}$. We will show that all vertices can be reached from the vertex $x^0 y^0$ in $O(m) = O(\log n)$ steps. In each step, the vertices that can be reached from any initial vertex can be obtained by applying $\{x, x^{-1}, y, y^{-1}, x^{-1}y, y^{-1}x\}$. For $q > 1$, any positive integer $i < q^m$ can be decomposed as $i = \sum_{a=0}^{m-1} i_a q^a$, where $0 \leq i_j < q$. Since $yx = x^q y$, we have $x^i y^j = \prod_{a=0}^{m-1} (x^{i_a} y)\, y^{j+1}$, which means that the vertex $x^i y^j$ can be reached in $\sum_a i_a + m + j < ((q+1)/2 + 2)\, m$ steps, and hence the diameter is bounded by $((q+1)/2 + 2)\, m = O(\log n)$.

Now consider $s = \{1, x^u, x^{u'} y^v\}$ and $\gcd(u, \ell) = \gcd(v, m) = 1$. The latter condition is to ensure that the graph is connected. We then have $(x^{u'} y^v) x^u = x^{uq^v} (x^{u'} y^v)$, which says that this graph is isomorphic to one generated by $s' = \{1, x, y\}$ if we suitably modify $q \to q' = q^v$. For s' and q', we can decompose $i < q^m$ as $i = \sum_{a=0}^{m-1} i_a q^{va \bmod m} = \sum_{a=0}^{m-1} i_a q'^a$, and hence the diameter has the same upper bound $((q+1)/2 + 2)\, m = O(\log n)$.

For a more general set of generators $s = \{s_1, s_2, s_3\}$, we can decompose s as $s = s_1 s' = s_1 \{1, s_1^{-1} s_2, s_1^{-1} s_3\} = s_1 \{1, s_2', s_3'\}$. Since the adjacency matrix is generated by $s^{-1} s_j \backslash \{1\} = s_i'^{-1} s_j' \backslash \{1\}$, to $x^{i'} y^{j'}$, the graph generated by s is isomorphic to that of s'. The interpretation of the condition that the generators have to satisfy becomes clear now: $s_2 s_1^{-1} s_3 \neq s_3 s_1^{-1} s_2$ means s_2' and s_3' do not commute, so we can get $x^u = s_2' s_3' s_2'^{-1} s_3'^{-1}$ with a nonzero u. The above condition also implies that s_2' or s_3' has a nonzero exponent of y. If we assume $s_2' = x^{u'} y^v$ with $\gcd(v, m) = 1$ and $\gcd(u, n) = 1$, the situation becomes similar to the previous case and we can get a looser upper bound of the diameter $(2(q+1) + 2)\, m = O(\log n)$.

So far, we assumed that the graph of A is connected, which corresponds to the condition $\gcd(u, \ell) = \gcd(v, m) = 1$. If the graph is disconnected, we start with the fact that the diameter of H_X is less than the sum of every disjoint subgraph of A and B. Consider the worst case, where $\gcd(u, \ell) = u$ and $\gcd(v, m) = v$. The graph can be separated into uv disconnected subgraphs. It can be checked that each subgraph is equivalent to the graph of $s = \{1, x, y\}$ with $q \to q^v$, $s \to s/u$ and $m \to m/v$. Therefore, the diameter of each connected graph is still $O(m)$ and the sum of the diameters is $O(uvm)$. As a result, for any nonzero u and v, the diameter of H_X is bounded by $O(\log n)$. ■

Theorem B4 (Diameter of abelian 2BGA Tanner graphs). Suppose we have a 2BGA code with check weight w_c, i.e., there are w_c total group generators in the specifications of the block matrices A and B. Then the diameter of the X-type and Z-type Tanner graphs is at least $\Omega\left(n^{1/(w_c-1)}\right)$.

Proof. For abelian 2BGA codes, the diameter of the Tanner graph corresponding to H_X or H_Z is larger than that corresponding to $A + B$. Hence, a lower bound on the diameter of $A + B$ will also be a lower bound on the diameters of H_X and H_Z. Recall that we can apply a change of basis to all generators to normalize one of them to be 1. We can then regard the $w_c - 1$ nontrivial generators as unit vectors that span a vector space where data qubits are associated with distinct vectors. Since we have $w_c - 1$ unit vectors, this vector space is at most $(w_c - 1)$-dimensional, which means that the diameter of the Tanner graph is at least $\Omega\left(n^{1/(w_c-1)}\right)$. ■

APPENDIX C: SELF-CORRECTION, CONFINEMENT, AND EXPANSION

This appendix contains a brief primer on self-correcting memories and expander graphs. Given a classical linear or quantum stabilizer code, we can define a code Hamiltonian that is a (negative) sum of all parity checks. This Hamiltonian is fully commuting and has an integer spectrum labeled by distinct error syndromes whose ground state subspace is the codespace. Loosely speaking, we say that a code is self-correcting if we allow it to interact with a heat bath, according to its code Hamiltonian, at some constant nonzero temperature and can successfully recover the encoded information with high probability after this interaction for a time diverging with system size. Typically, one also assumes that the system-bath interaction is local and obeys detailed balance, so that the steady state is given by the Gibbs distribution; examples of such interactions include classical Markov-chain Monte Carlo algorithms [120–122] as well as their quantum generalizations [123–126]. Successful final recovery is dependent on a specific decoder, which is usually assumed to be a theoretically tractable decoder such as minimum-weight or maximum-likelihood. Formally, let $\rho_{\mathcal{C}}$ denote an arbitrary state in the codespace, $\mathcal{B}_{T,\tau}$ the CPTP map for the interaction with the bath at temperature T for time τ, and $\mathcal{R}$ the CPTP map for the final recovery operation. We say that a code is self-correcting if for $T < T_c = \Omega(1)$ and $\tau = \omega_n(1)$ (typically exponential),

$$\mathcal{R} \circ \mathcal{B}_{T,\tau}[\rho_{\mathcal{C}}] \propto \rho_{\mathcal{C}}, \tag{C1}$$

with probability $\mathbf{P} = 1 - o_n(1)$. In other words, the Knill-Laflamme QEC conditions [127] are satisfied for the error channel corresponding to $\mathcal{B}_{T,\tau}$. The word "self-correcting"

originates from the interpretation that the bath simultaneously corrects as well as produces errors, and below the critical temperature there is a strong bias toward correction. There will always be some residual error at the end, and so the final recovery $\mathcal{R}$ is more or less just a proxy for $\mathcal{B}_{T,\tau}$ according to Eq. (C1).

An important quantity in the analysis of a self-correcting memory is the free energy $F = E - TS = -T\log Z$, where S is the entropy and Z is the canonical partition function. The free energy characterizes the competition between entropy (related to errors) and energy (related to correction). Intuitively, for transitions between different states given by the system-bath interaction, the probability is proportional to $e^{-\beta\Delta E}$, and the number of transitions that are accessible is proportional to $e^{\Delta S}$, and so transitions that increase the free energy are unfavored whereas those that decrease the free energy are favored in typical trajectories of the system under random dynamics. Hence, if we desire a lower bound on the self-correcting memory time τ, then it suffices to obtain a lower bound on the free energy cost of incurring a logical error: in other words, to argue that there is a low free energy bottleneck around each codeword that must be reached to incur a logical error [59].

Confinement is a code property which, loosely speaking, asserts that increasingly larger errors produce increasingly larger syndromes (energy cost), up to a cutoff known as the energy barrier of the code. Beyond this energy barrier, a suitably large error may actually produce a small syndrome [59,128]. The energy cost as a function of the error weight is known as the confinement function. For an LDPC code, since each qubit only participates in a constant number of parity checks, the confinement function can at most be linear. It was recently shown that a linear confinement function is sufficient to overcome any effects of entropy and result in a macroscopic free energy barrier, thus leading to self-correction [59,60,67,68].

Theorem C1 (Small-set linear confinement implies self--correction (informal) [59]). Suppose we have a quantum LDPC stabilizer code of length n such that the energy cost $\mathcal{E}(P)$ of Pauli error P satisfies

$$\mathcal{E}(P) \geq \alpha|P|, \quad \forall\, |P| \leq \gamma n^b \tag{C2}$$

for constants $\alpha, \gamma, b > 0$ independent of n. Then there exists a critical temperature, below which the self-correcting memory time τ diverges as

$$\tau = e^{\Omega(n^b)}. \tag{C3}$$

Note that self-correcting memories exist without the strict requirement of linear confinement (C2) such as toric codes in $D \geq 4$ spatial dimensions [83] and color codes in $D \geq 6$ spatial dimensions [129]. However, in both of these examples, the parity checks form an extensively overcomplete set, and the special redundant structure restricts entropic effects. Since our ZSZ codes do not possess such an extensive amount of redundant parity checks, we speculate that our numerical observations of self-correction are attributed to strong confinement in typical error clusters. From Eq. (C2), it is clear that linear confinement requires the size of the neighborhood or boundary of small vertex sets to be proportional to their volume. This boundary $\propto$ volume correspondence is a defining trait of a special class of sparse graphs known as expander graphs.

Definition C1 (Graph expansion). For a graph $\mathcal{G} = (V, E)$, its *edge expansion* or *Cheeger constant* $h(\mathcal{G})$ is defined as

$$h(\mathcal{G}) = \min_{S\subset V:\, |S|\leq |V|/2} \frac{|\partial S|}{|S|}, \tag{C4}$$

where $\partial S := \{\{u, v\} \in E : u \in S,\, v \notin S\} \subset E$ is the edge neighborhood of S. We say a family of graphs $\{\mathcal{G}_i\}$ of increasing sizes is *expanding* if $h(\mathcal{G}_i) = \Omega(1)$.

Intuitively, the edge expansion $h(\mathcal{G})$ tells us how many edges we need to cut in order to disconnect the graph into disjoint components. Note that the condition $h(\mathcal{G}) = \Omega(1)$ implies that $\mathrm{diam}(\mathcal{G}) = O(\log n)$, since the individual neighborhoods of any two vertices grow exponentially and eventually encompass at least half of all vertices. By the pigeonhole principle, there must then exist a path of length $O(\log n)$ between any two vertices. However, the converse is not true. A regular tree has logarithmic diameter but poor expansion because cutting any edge disconnects the entire branch attached to it, which may have extensive size.

Typically, one can derive linear confinement bounds from the underlying (bipartite) expansion of the code's Tanner graph. If the expansion coefficient of a subset is larger than half the vertex degree, then there must exist a fraction of parity checks in the neighborhood of that subset that are only connected by a single edge, i.e., a "unique neighbor," and hence correspond to unsatisfied checks if the subset is the support of an error. For random graphs, one can probabilistically demonstrate this unique-neighbor expansion up to a constant fraction of vertices using combinatorial arguments [80,130]. Explicit constructions typically revolve around local modifications of expanding Cayley graphs, e.g., [61,131], in order to upgrade them to unique-neighbor expanders [62,132,133]. We now recite a no-go theorem regarding the expansion of Cayley graphs.

Theorem C2 (Nonexpanding Cayley graphs; Corollary 3.3 of [134]). Let $\{G_i\}$ be a family of finite groups, each with derived length $l = O(1)$ and generating set S_i with

$|S_i| = O(1)$. Then the family of Cayley graphs $\mathcal{G}(G_i, S_i)$ is not expanding.

We will not formally rederive this established result, but we can explain the intuition behind it. Loosely speaking, a group with a constant derived length, also known as a solvable group, has an "almost abelian" structure in the sense that it can be viewed as finite extensions of abelian groups. We can see this "almost abelian" structure inside $\mathbb{Z}_\ell \rtimes_q \mathbb{Z}_m$ as follows. Its commutator subgroup is precisely the normal subgroup $\mathbb{Z}_\ell$. When we quotient out this normal subgroup, we obtain $\mathbb{Z}_m$. Geometrically (recalling the rectangular layout of Fig. 2), we are treating each row [copy of $\mathrm{Cay}(\mathbb{Z}_\ell, x)$] as a conglomerate object, with the automorphism $\varphi(x) = x^q$ acting as a coarse-grained "edge" between neighboring rows; this is the structure of $\mathrm{Cay}(\mathbb{Z}_m, y)$, which is not an expander. In particular, the size of the neighborhood of a collection of adjacent rows remains constant irrespective of the number of included rows.

Corollary C1 (ZSZ derived length). The derived length of the group $\mathbb{Z}_\ell \rtimes_q \mathbb{Z}_m$ is 2, and thus a ZSZ Cayley graph is not an expander according to Definition C1.

Proof. Without loss of generality, we label all elements of $\mathbb{Z}_\ell \rtimes_q \mathbb{Z}_m$ as $x^i y^j$ for $i = 1, \ldots, \ell$ and $j = 1, \ldots, m$. Since both $\mathbb{Z}_\ell$ and $\mathbb{Z}_m$ are abelian, the only nontrivial elements in the commutator subgroup of their semidirect product come from terms between the two groups. For generators $x \in \mathbb{Z}_\ell$ and $y \in \mathbb{Z}_m$,

$$[x, y] = x^{-1} y^{-1} x y = x^{-1} x^{q^{m-1}} y^{-1} y = x^{q^{m-1}-1}, \tag{C5}$$

which generates an abelian subgroup of $\mathbb{Z}_\ell$. Since the commutator subgroup of an abelian group is trivial, the derived series of $\mathbb{Z}_\ell \rtimes_q \mathbb{Z}_m$ terminates after two iterations. Theorem C2 then asserts that this family of Cayley graphs cannot be expanding. ■

Notice, however, that Definition C1 for an expander graph is a *global* one: the boundary $\propto$ volume scaling must persist to half the size of the graph. As is clear from Theorem C1, quantum self-correction will follow from any LDPC code with sufficient expansion on small sets, whose maximum size can be subextensive $o(n)$. Due to both the strong numerical evidence of self-correction as well as the lack of local metachecks, we speculate:

Conjecture 1 (Free energy barrier in ZSZ codes). There exists a particular sequence of ZSZ codes of increasing length n, which exhibit a free energy barrier that grows with n around codewords, and as such possess a self-correction threshold.

1. Relation to noisy greedy decoding

In this Appendix, we briefly review the connection between local greedy decoding and a particular (classical) Gibbs sampler known as Glauber dynamics. For a quantum CSS code with code Hamiltonian $\mathcal{H} = \mathcal{H}_X + \mathcal{H}_Z$, classical Glauber dynamics on independent X and Z errors is sufficient to sample the entire spectrum of the code Hamiltonian and hence is a valid quantum Gibbs sampler. Without loss of generality, we focus our attention on X errors. In Glauber dynamics, we select a data qubit (labeled by j) of the code at random and measure its connected Z-checks to obtain a local error syndrome. We then apply the local Pauli X_j operator with probability

$$\mathbf{P}\left(|\psi\rangle \to X_j |\psi\rangle\right) = \frac{1}{1 + \mathrm{e}^{\beta \Delta E}}, \tag{C6}$$

where ΔE is the change in energy or syndrome weight upon applying X_j, and β is the inverse temperature. One can quickly verify that Eq. (C6) along with its Z-error version satisfies the detailed balance condition $\rho(|\psi'\rangle)\mathbf{P}(|\psi'\rangle \to |\psi\rangle) = \rho(|\psi\rangle)\mathbf{P}(|\psi\rangle \to |\psi'\rangle)$ when $\rho \propto \mathrm{e}^{-\beta\mathcal{H}}$ is the equilibrium Gibbs state. Since every data qubit has an equal probability of being chosen, every eigenstate of $\mathcal{H}$ has a nonzero probability of being reached (ergodicity). Thus, the Gibbs state is the unique steady state of Glauber dynamics (C6).

To understand the connection of Eq. (C6) to greedy decoding, let us first examine the zero-temperature $\beta \to \infty$ limit. In this limit, Eq. (C6) becomes a step function: local Paulis that lower the energy ($\Delta E > 0$) are always applied, those that increase the energy ($\Delta E < 0$) are never applied, and ties ($\Delta E = 0$) are handled according to a 50% coin toss. We can interpret this limit as a greedy decoder that tries to always locally lower the energy according to a majority vote. When β is finite, the greedy decoder now applies the local majority "correctly" with probability p (C6) and "incorrectly" with probability

$$q(\Delta E) = 1 - p(\Delta E) = \frac{1}{1 + \mathrm{e}^{\beta |\Delta E|}}. \tag{C7}$$

Equation (C7) may seem a bit odd at first since it requires very specific failure probabilities for different inputs to the majority vote. It could be the case that we have a uniform noise model where the greedy decoder outputs the wrong answer with a fixed probability for all inputs. In this case, we cannot exactly map it to Glauber dynamics, but we can provide an upper bound and say that it fails *less often* than Glauber dynamics at some finite temperature. Since our code is LDPC, each qubit participates in at most $w_q = O(1)$ Z-checks. Notice that at fixed β, Eq. (C7) is minimized when $|\Delta E|$ is maximized, which occurs when

either all or none of the parity checks are satisfied:

$$q_{\min} = \frac{1}{1 + e^{\beta w_q}}. \tag{C8}$$

So given some failure probability q, we can provide an upper-bound temperature β_{upper} according to Eq. (C8). If $\beta_{\text{upper}} > \beta_c$ is in the self-correcting phase, then we can expect the performance of the noisy greedy decoder to be *no worse* than that of Glauber dynamics at temperature β_{upper}. Note that we can always deliberately add in our own "errors" to exactly match Eq. (C7) if desired.

APPENDIX D: SINGLE-SHOT GREEDY DECODER IMPLEMENTATION

At the heart of our passive memory lies a single-shot implementation of the local greedy decoder. A decoder is single-shot if it can reliably operate using information from only $O(1)$ syndrome extraction cycles [41]. Roughly speaking, the goal of a single-shot decoder is not to eliminate *all* errors, but only to reduce enough errors at each cycle such that the encoded information can *eventually* be recoverable. For scalable fault-tolerant MFQEC, it is crucial that this decoder be not only single-shot but also implementable with a constant-depth circuit. So in addition to requiring $O(1)$ syndrome extraction cycles, we also demand $O(1)$ classical time complexity, so that qubit idling times remain finite even as $n \to \infty$. Note that typical single-shot decoders only require the first condition but not the second; e.g., MWPM for fixing broken loop excitations in the 3D toric code has an $O(n^3)$ classical time complexity.

It remains to show that we can implement a full sweep (all sites acted on once) of the greedy decoder in constant depth. A naive scheduling of the sweep such as "typewriter" order may require linear depth since the majority vote on the next data qubit may require the updated syndrome information from the majority vote on the previous qubit. Fortunately, because the ZSZ codes are LDPC, there exist "nonoverlapping" sets of data qubits that we can address in parallel. Define the qubit X-adjacency graph as the simple graph where vertices denote data qubits, and two vertices are connected by an edge if and only if their associated data qubits share an X-check; the qubit Z-adjacency graph follows suit but with respect to the Z-checks. Since each data qubit participates in 3 X-checks, and each X-check involves 6 data qubits, the maximum degree of the qubit X-adjacency graph is $3(6-1) = 15$. Brooks's theorem [135] then tells us that we can partition all the vertices into at most 15 nonoverlapping subsets from which we can apply our greedy decoder in parallel. We note that the vertex degree is only an upper bound, and for some graphs the minimum partition, or chromatic number, can be smaller. Using "sequential" and "independent-set" greedy graph coloring algorithms in NETWORKX, we found colorings between 7 and 10 for all of our ZSZ codes in Table I. In general, it is NP-hard to compute the chromatic number χ of an arbitrary graph. Nonetheless, Brooks's theorem provides us with an upper bound $\chi = O(1)$ for LDPC codes. Perhaps better approximate colorings can be obtained by leveraging the underlying Cayley graph structure of the qubit adjacency graphs.

Suppose we have a χ-coloring of our qubit adjacency graph with $\chi = O(1)$. Within each subset of fixed color,

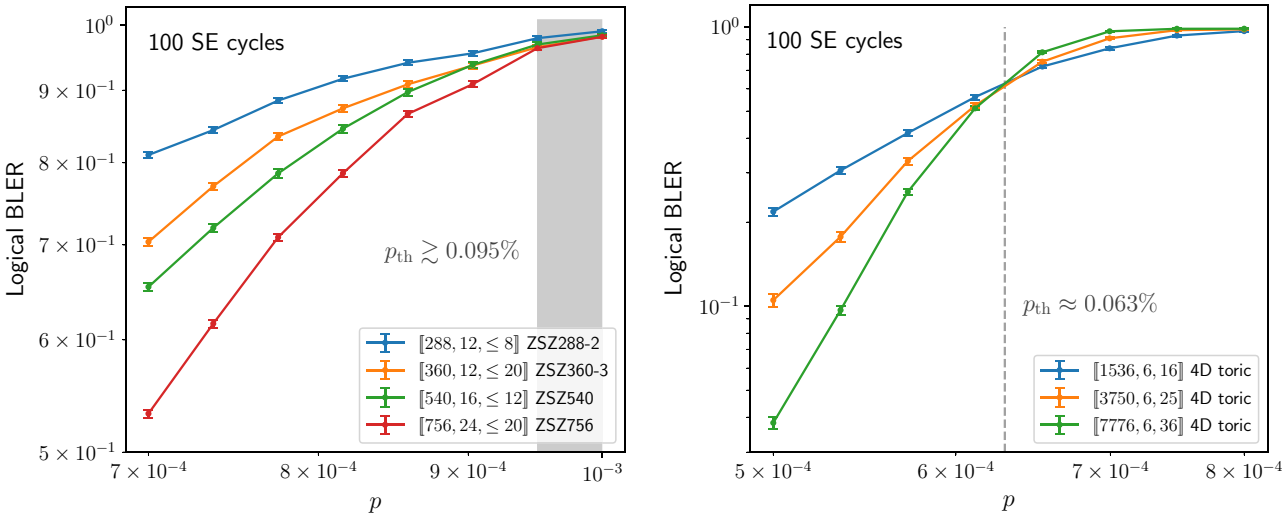


FIG. 8. The logical block error rate as a function of the physical noise strength p is plotted near the observed sustainable thresholds for both the ZSZ codes as well as 4D toric codes with linear sizes $L = 4, 5, 6$ under passive decoding. The curves of the 4D toric codes display a clear intersection around $p \approx 0.063\%$. The curves of the ZSZ codes do not show an obvious intersection, but we observe subthreshold behavior (decreasing logical block error rate with increasing n) below the region highlighted by the shaded rectangle, and so we estimate a lower bound $p \gtrsim 0.095\%$ on the sustainable threshold.

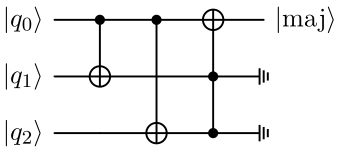
TABLE II. Bivariate bicycle codes and their parameters for the simulations in Fig. 9 are displayed. Code distances are numerically estimated using the GAP package QDISTRND [69].

Decoding	Name	$[\![n,k,d]\!]$	ℓ,m	A	B
Passive	BB144-2	$[\![144,12,\leq 12]\!]$	12,6	$1+x^{11}y+x^4y^5$	$x^6+x^7+x^8y^3$
	BB360-2	$[\![360,12,\leq 12]\!]$	30,6	$x^9y^2+x^3y^3+x^2y^5$	$x^{25}+x^{21}+x^{18}y^5$
	BB756-1	$[\![756,16,\leq 20]\!]$	21,18	$1+x^{20}y^3+x^4y^6$	$x^8y^9+x^{20}y^{17}+y^{17}$

we can apply the greedy decoder to all qubits in parallel. The full decoding procedure then proceeds as follows:

(1) Perform a round of (X or Z) syndrome extraction with $n/2$ syndrome qubits.
(2) Choose a nonoverlapping subset of data qubits according to the χ-coloring of the qubit adjacency graph and apply the greedy decoder to all subset data qubits in parallel. Update the relevant bits in the error syndrome.
(3) Iterate step 2 (χ total iterations) until all data qubits have been addressed by the greedy decoder.

In a classical computer, steps 2 and 3 can be compiled into simple Boolean arithmetic in a straightforward manner. For a MFQEC implementation, we will need to compile these instructions into a quantum circuit. For step 2, we can borrow the reversible circuit of [136] to implement the majority vote on three inputs,



(D1)

The above majority circuit does not preserve the inputs, but one can simply copy their classical values prior to the circuit using fresh ancillas in $|0\rangle$ and CNOT gates. After the majority qubit is obtained, we apply a controlled gate from it to the data qubit, with the target rotation being X or Z depending on which Pauli error we are correcting. The syndrome qubits can also be updated similarly using CNOT gates from the majority qubit. These updated syndrome qubits can then be used for step 3. After we perform the full sweep of the greedy decoder on all data qubits, we can discard all ancilla qubits and repeat the process, starting with a new round of syndrome extraction.

APPENDIX E: ADDITIONAL NUMERICAL SIMULATIONS

1. Sustainable threshold estimation

We perform additional numerical simulations of passive error correction near the region ($p \approx 10^{-3}$) where the curves intersect in Fig. 5. Figure 8 presents these numerical results for both the ZSZ codes and the 4D toric code family. From the plots, we estimate the sustainable threshold of ZSZ codes to be $p_{\mathrm{th}}^{\mathrm{ZSZ}} \approx 0.095\%$; similarly, we estimate a sustainable threshold of $p_{\mathrm{th}}^{\mathrm{4D}} \approx 0.063\%$ for the 4D toric code.

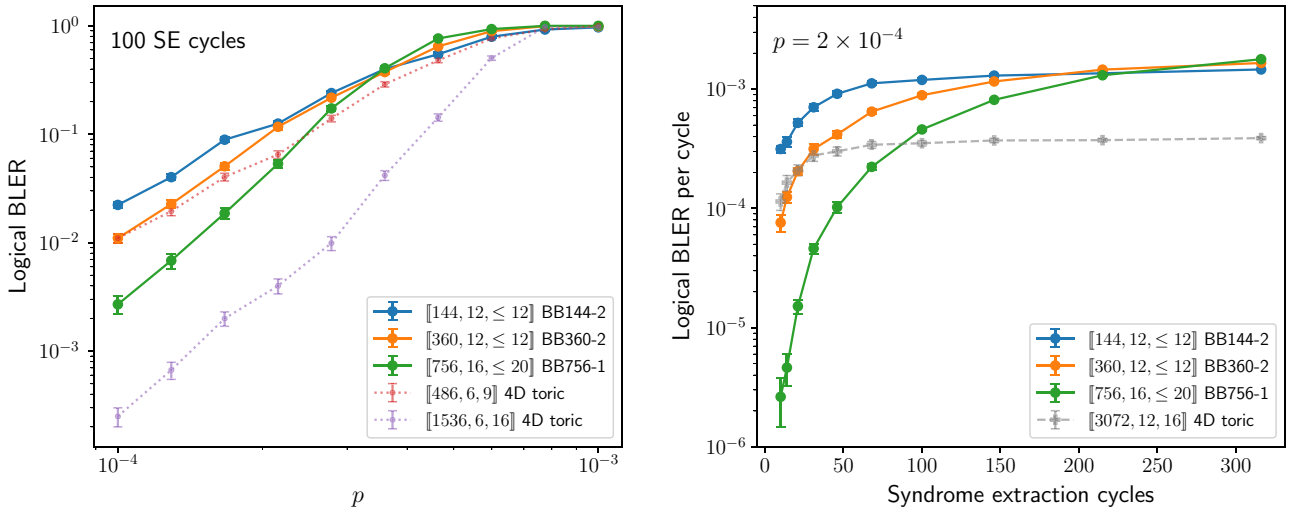


FIG. 9. The analog of Fig. 5 for the bivariate bicycle codes is displayed. In contrast to ZSZ codes, we do not observe any evidence of a sustainable threshold for bivariate bicycle codes under passive decoding.

2. Passive error correction of bivariate bicycle codes

We perform an equivalent numerical search and noise simulation for BB codes as for our ZSZ codes under passive decoding. See Table II for the parameters of the BB codes we have found that achieved the best observed performance under passive decoding. The numerical simulation results are presented in Fig. 9. Unlike the case for ZSZ codes, we do not see reasonable evidence of a sustainable threshold. The left plot seems to suggest a transition near $p \approx 0.04\%$, but upon examination of the right plot, we see that this behavior is simply a finite-size effect: the logical error rates of the larger BB codes do not stabilize with increasing syndrome extraction cycles and eventually surpass those of the smaller BB codes.

[1] Rajeev Acharya *et al.*, Suppressing quantum errors by scaling a surface code logical qubit, Nature **614**, 676 (2023).

[2] Rajeev Acharya *et al.*, Quantum error correction below the surface code threshold, Nature **638**, 920 (2024).

[3] A. Paetznick *et al.*, Demonstration of logical qubits and repeated error correction with better-than-physical error rates, arXiv:2404.02280.

[4] Yifan Hong, Elijah Durso-Sabina, David Hayes, and Andrew Lucas, Entangling four logical qubits beyond break-even in a nonlocal code, Phys. Rev. Lett. **133**, 180601 (2024).

[5] Noah Berthusen, Joan Dreiling, Cameron Foltz, John P. Gaebler, Thomas M. Gatterman, Dan Gresh, Nathan Hewitt, Michael Mills, Steven A. Moses, Brian Neyenhuis, Peter Siegfried, and David Hayes, Experiments with the four-dimensional surface code on a quantum charge-coupled device quantum computer, Phys. Rev. A **110**, 062413 (2024).

[6] Ben W. Reichardt, David Aasen, Rui Chao, Alex Chernoguzov, Wim van Dam, John P. Gaebler, Dan Gresh, Dominic Lucchetti, Michael Mills, Steven A. Moses, Brian Neyenhuis, Adam Paetznick, Andres Paz, Peter E. Siegfried, Marcus P. da Silva, Krysta M. Svore, Zhenghan Wang, and Matt Zanner, Demonstration of quantum computation and error correction with a tesseract code, arXiv:2409.04628.

[7] Lucas Daguerre, Robin Blume-Kohout, Natalie C. Brown, David Hayes, and Isaac H. Kim, Experimental demonstration of high-fidelity logical magic states from code switching, Phys. Rev. X **15**, 041008 (2025).

[8] Shival Dasu, Simon Burton, Karl Mayer, David Amaro, Justin A. Gerber, Kevin Gilmore, Dan Gresh, Davide Del-Vento, Andrew C. Potter, and David Hayes, Breaking even with magic: demonstration of a high-fidelity logical non-Clifford gate, arXiv:2506.14688.

[9] Dolev Bluvstein, Simon J. Evered, Alexandra A. Geim, Sophie H. Li, Hengyun Zhou, Tom Manovitz, Sepehr Ebadi, Madelyn Cain, Marcin Kalinowski, Dominik Hangleiter *et al.*, Logical quantum processor based on reconfigurable atom arrays, Nature **626**, 58 (2024).

[10] Pedro Sales Rodriguez *et al.*, Experimental demonstration of logical magic state distillation, Nature **645**, 620 (2025).

[11] Ben W. Reichardt *et al.*, Fault-tolerant quantum computation with a neutral atom processor, arXiv:2411.11822.

[12] Dolev Bluvstein *et al.*, Architectural mechanisms of a universal fault-tolerant quantum computer, arXiv:2506.20661.

[13] Pavel Panteleev and Gleb Kalachev, in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022 (Association for Computing Machinery, New York, NY, USA, 2022), p. 375.

[14] Anthony Leverrier and Gilles Zemor, in *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)* (IEEE Computer Society, Los Alamitos, CA, USA, 2022), p. 872.

[15] Irit Dinur, Min-Hsiu Hsieh, Ting-Chun Lin, and Thomas Vidick, in *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023 (Association for Computing Machinery, New York, NY, USA, 2023), p. 905.

[16] Adam Wills, Min-Hsiu Hsieh, and Hayata Yamasaki, Constant-overhead magic state distillation, arXiv:2408.07764.

[17] Ting-Chun Lin, Transversal non-Clifford gates for quantum LDPC codes on sheaves, arXiv:2410.14631.

[18] Louis Golowich and Ting-Chun Lin, Quantum LDPC Codes with Transversal Non-Clifford Gates via Products of Algebraic Codes, in *Proceedings of the 57th Annual ACM Symposium on Theory of Computing (STOC 2025)* (ACM, Prague, Czechia, 2025), pp. 689–696.

[19] Nikolas P. Breuckmann, Margarita Davydova, Jens N. Eberhardt, and Nathanan Tantivasadakarn, Cups and gates I: Cohomology invariants and logical quantum operations, arXiv:2410.16250.

[20] Zhiyang He, Vinod Vaikuntanathan, Adam Wills, and Rachel Yun Zhang, Quantum codes with addressable and transversal non-Clifford gates, arXiv:2502.01864.

[21] Zhiyang He, Vinod Vaikuntanathan, Adam Wills, and Rachel Yun Zhang, Asymptotically good quantum codes with addressable and transversal non-Clifford gates, arXiv:2507.05392.

[22] Hayata Yamasaki and Masato Koashi, Time-efficient constant-space-overhead fault-tolerant quantum computation, Nat. Phys. **20**, 247 (2024).

[23] Quynh T. Nguyen and Christopher A. Pattison, Quantum Fault Tolerance with Constant-Space and Logarithmic-Time Overheads, in *Proceedings of the 57th Annual ACM Symposium on Theory of Computing (STOC 2025)* (ACM, Prague, Czechia, 2025), pp. 730–737.

[24] Shiro Tamiya, Masato Koashi, and Hayata Yamasaki, Polylog-time- and constant-space-overhead fault-tolerant quantum computation with quantum low-density parity-check codes, arXiv:2411.03683.

[25] Dominic J. Williamson and Theodore J. Yoder, Low-overhead fault-tolerant quantum computation by gauging logical operators, arXiv:2410.02213.

[26] Alexander Cowtan, SSIP: Automated surgery with quantum LDPC codes, arXiv:2407.09423.

[27] Esha Swaroop, Tomas Jochym-O'Connor, and Theodore J. Yoder, Universal adapters between quantum LDPC codes, arXiv:2410.03628.

[28] Zhiyang He, Alexander Cowtan, Dominic J. Williamson, and Theodore J. Yoder, Extractors: QLDPC architectures for efficient Pauli-based computation, arXiv:2503.10390.

[29] Theodore J. Yoder, Eddie Schoute, Patrick Rall, Emily Pritchett, Jay M. Gambetta, Andrew W. Cross, Malcolm Carroll, and Michael E. Beverland, Tour de gross: A modular quantum computer based on bivariate bicycle codes, arXiv:2506.03094.

[30] Daniel Litinski, A game of surface codes: Large-scale quantum computing with lattice surgery, Quantum **3**, 128 (2019).

[31] Sergey Bravyi, Andrew W. Cross, Jay M. Gambetta, Dmitri Maslov, Patrick Rall, and Theodore J. Yoder, High-threshold and low-overhead fault-tolerant quantum memory, Nature **627**, 778 (2024).

[32] Joshua Viszlai, Willers Yang, Sophia Fuhui Lin, Junyu Liu, Natalia Nottingham, Jonathan M. Baker, and Frederic T. Chong, Matching generalized-bicycle codes to neutral atoms for low-overhead fault-tolerance, arXiv:2311.16980.

[33] Samuel Stein, Shifan Xu, Andrew W. Cross, Theodore J. Yoder, Ali Javadi-Abhari, Chenxu Liu, Kun Liu, Zeyuan Zhou, Charles Guinn, Yufei Ding, Yongshan Ding, and Ang Li, Architectures for heterogeneous quantum error correction codes, arXiv:2411.03202.

[34] François Arnault, Philippe Gaborit, Wouter Rozendaal, Nicolas Saussay, and Gilles Zémor, A variant of the Bravyi-Terhal bound for arbitrary boundary conditions, arXiv:2502.04995.

[35] Sergey Bravyi and Barbara Terhal, A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes, New J. Phys. **11**, 043029 (2009).

[36] Sergey Bravyi, David Poulin, and Barbara Terhal, Tradeoffs for reliable quantum information storage in 2D systems, Phys. Rev. Lett. **104**, 050503 (2010).

[37] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill, Topological quantum memory, J. Math. Phys. **43**, 4452 (2002).

[38] Charlene Ahn, Andrew C. Doherty, and Andrew J. Landahl, Continuous quantum error correction via quantum feedback control, Phys. Rev. A **65**, 042301 (2002).

[39] Mohan Sarovar and G. J. Milburn, Continuous quantum error correction by cooling, Phys. Rev. A **72**, 012306 (2005).

[40] D. A. Spielman, Linear-time encodable and decodable error-correcting codes, IEEE Trans. Inf. Theory **42**, 1723 (1996).

[41] Héctor Bombín, Single-shot fault-tolerant quantum error correction, Phys. Rev. X **5**, 031043 (2015).

[42] Madelyn Cain, Chen Zhao, Hengyun Zhou, Nadine Meister, J. Pablo Bonilla Ataides, Arthur Jaffe, Dolev Bluvstein, and Mikhail D. Lukin, Correlated decoding of logical algorithms with transversal gates, arXiv:2403.03272.

[43] Hengyun Zhou, Chen Zhao, Madelyn Cain, Dolev Bluvstein, Casey Duckering, Hong-Ye Hu, Sheng-Tao Wang, Aleksander Kubica, and Mikhail D. Lukin, Low-overhead transversal fault tolerance for universal quantum computation, Nature **646**, 303 (2025).

[44] Gerardo A. Paz-Silva, Gavin K. Brennen, and Jason Twamley, Fault tolerance with noisy and slow measurements and preparation, Phys. Rev. Lett. **105**, 100501 (2010).

[45] Sascha Heußen, David F. Locher, and Markus Müller, Measurement-free fault-tolerant quantum error correction in near-term devices, PRX Quantum **5**, 010333 (2024).

[46] Friederike Butt, Ivan Pogorelov, Robert Freund, Alex Steiner, Marcel Meyer, Thomas Monz, and Markus Müller, Demonstration of measurement-free universal fault-tolerant quantum computation, arXiv:2506.22600.

[47] Mincheol Park, Nishad Maskara, Marcin Kalinowski, and Mikhail D. Lukin, Enhancing quantum memory lifetime with measurement-free local error correction and reinforcement learning, Phys. Rev. A **111**, 012419 (2025).

[48] Alexey A. Kovalev and Leonid P. Pryadko, Quantum Kronecker sum-product low-density parity-check codes with finite rate, Phys. Rev. A **88**, 012311 (2013).

[49] Renyu Wang, Hsiang-Ku Lin, and Leonid P. Pryadko, Abelian and non-abelian quantum two-block codes, arXiv:2305.06890.

[50] Hsiang-Ku Lin and Leonid P. Pryadko, Quantum two-block group algebra codes, Phys. Rev. A **109**, 022407 (2024).

[51] A. R. Calderbank and P. W. Shor, Good quantum error-correcting codes exist, Phys. Rev. A **54**, 1098 (1996).

[52] Andrew Steane, Multiple-particle interference and quantum error correction, Proc. R. Soc. London. Series A: Math. Phys. Eng. Sci. **452**, 2551 (1997).

[53] Daniel Gottesman, Stabilizer codes and quantum error correction, arXiv:quant-ph/9705052.

[54] $+$ and $-$ are equivalent modulo 2.

[55] Nikolas P. Breuckmann and Jens Niklas Eberhardt, Quantum low-density parity-check codes, PRX Quantum **2**, 040101 (2021).

[56] Note that here we are using a slight abuse of notation. Usually the right-regular representation is defined as $R[a] \equiv a^{-1}$ so that the representation condition $R[a]R[b] = R[ab] \equiv b^{-1}a^{-1}$ is satisfied. So when we write $R[a]$ to denote right-multiplication by a, we really mean $R[a^{-1}]$ by the standard convention.

[57] This particular order follows the multiplication order of multiplying $x^\alpha y^\beta$ on the left: y^β is acted first followed by x^α.

[58] Armanda O. Quintavalle, Michael Vasmer, Joschka Roffe, and Earl T. Campbell, Single-shot error correction of three-dimensional homological product codes, PRX Quantum **2**, 020340 (2021).

[59] Yifan Hong, Jinkang Guo, and Andrew Lucas, Quantum memory at nonzero temperature in a thermodynamically trivial system, Nat. Commun. **16**, 316 (2025).

[60] Benedikt Placke, Tibor Rakovszky, Nikolas P. Breuckmann, and Vedika Khemani, Topological Quantum Spin Glass Order and its realization in qLDPC codes, arXiv:2412.13248.

[61] A. Lubotzky, R. Phillips, and P. Sarnak, Ramanujan graphs, Combinatorica **8**, 261 (1988).

[62] M. Sipser and D. A. Spielman, Expander codes, IEEE Trans. Inf. Theory **42**, 1710 (1996).

[63] Theo McKenzie and Sidhanth Mohanty, in *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*, Leibniz International Proceedings in Informatics (LIPIcs), Vol. 198, edited by Nikhil Bansal, Emanuela Merelli, and James Worrell (Schloss Dagstuhl—Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2021), p. 96:1.

[64] P. M. Bleher, J. Ruiz, and V. A. Zagrebnov, On the purity of the limiting Gibbs state for the Ising model on the Bethe lattice, J. Stat. Phys. **79**, 473 (1995).

[65] C. Kenyon, E. Mossel, and Y. Peres, Glauber Dynamics on Trees and Hyperbolic Graphs, in *42nd Annual Symposium on Foundations of Computer Science (FOCS 2001)* (IEEE Computer Society, Las Vegas, Nevada, USA, 2001), pp. 568–578.

[66] Andrea Montanari and Guilhem Semerjian, On the dynamics of the glass transition on Bethe lattices, J. Stat. Phys. **124**, 103 (2006).

[67] David Gamarnik, Bobak T. Kiani, and Alexander Zlokapa, Slow mixing of quantum Gibbs samplers, arXiv:2411.04300.

[68] Tibor Rakovszky, Benedikt Placke, Nikolas P. Breuckmann, and Vedika Khemani, Bottlenecks in quantum channels and finite temperature phases of matter, arXiv:2412.09598.

[69] L. P. Pryadko, V. A. Shabashov, and V. K. Kozin, QDistRnd, calculate the distance of a q−ary quantum stabilizer code, Version 0.9.5, 2024, see https://QEC-pages.github.io/QDistRnd, GAP package.

[70] Rui Chao and Ben W. Reichardt, Quantum error correction with only two extra qubits, Phys. Rev. Lett. **121**, 050502 (2018).

[71] D. Jaksch, J. I. Cirac, P. Zoller, S. L. Rolston, R. Côté, and M. D. Lukin, Fast quantum gates for neutral atoms, Phys. Rev. Lett. **85**, 2208 (2000).

[72] M. D. Lukin, M. Fleischhauer, R. Cote, L. M. Duan, D. Jaksch, J. I. Cirac, and P. Zoller, Dipole blockade and quantum information processing in mesoscopic atomic ensembles, Phys. Rev. Lett. **87**, 037901 (2001).

[73] Matthew DeCross *et al.*, The computational power of random quantum circuits in arbitrary geometries, Phys. Rev. X **15**, 021052 (2025).

[74] Simon J. Evered, Dolev Bluvstein, Marcin Kalinowski, Sepehr Ebadi, Tom Manovitz, Hengyun Zhou, Sophie H. Li, Alexandra A. Geim, Tout T. Wang, Nishad Maskara, Harry Levine, Giulia Semeghini, Markus Greiner, Vladan Vuletić, and Mikhail D. Lukin, High-fidelity parallel entangling gates on a neutral-atom quantum computer, Nature **622**, 268 (2023).

[75] Craig Gidney, Stim: A fast stabilizer circuit simulator, Quantum **5**, 497 (2021).

[76] Joschka Roffe, LDPC: Python tools for low density parity check codes (2022), see https://pypi.org/project/ldpc/.

[77] Pavel Panteleev and Gleb Kalachev, Degenerate quantum LDPC codes with good finite length performance, Quantum **5**, 585 (2021).

[78] Joschka Roffe, David R. White, Simon Burton, and Earl Campbell, Decoding across the quantum low-density parity-check code landscape, Phys. Rev. Res. **2**, 043423 (2020).

[79] Oscar Higgott and Nikolas P. Breuckmann, Improved single-shot decoding of higher-dimensional hypergraph-product codes, PRX Quantum **4**, 020332 (2023).

[80] R. Gallager, Low-density parity-check codes, IRE Trans. Inf. Theory **8**, 21 (1962).

[81] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov Chains and Mixing Times* (American Mathematical Society, Providence, RI, 2009).

[82] Lawrence E. Thomas, Bound on the mass gap for finite volume stochastic Ising models at low temperature, Commun. Math. Phys. **126**, 1 (1989).

[83] R. Alicki, M. Horodecki, P. Horodecki, and R. Horodecki, On thermal stability of topological qubit in Kitaev's 4D model, Open Syst. Inf. Dyn. **17**, 1 (2010).

[84] The greedy decoder involves a majority function that cannot be implemented by a Clifford circuit and so falls outside of the simulation regime of STIM. Thus, we stick to this simpler "phenomenological" error model for the passive decoding step.

[85] Timo Hillmann, Lucas Berent, Armanda O. Quintavalle, Jens Eisert, Robert Wille, and Joschka Roffe, Localized statistics decoding: A parallel decoding algorithm for quantum low-density parity-check codes, Nat. Commun. **16**, 8214 (2025).

[86] Yue Wu, Shimon Kolkowitz, Shruti Puri, and Jeff D. Thompson, Erasure conversion for fault-tolerant quantum computing in alkaline earth Rydberg atom arrays, Nat. Commun. **13**, 4657 (2022).

[87] James D. Teoh, Patrick Winkel, Harshvardhan K. Babla, Benjamin J. Chapman, Jahan Claes, Stijn J. de Graaf, John W. O. Garmon, William D. Kalfus, Yao Lu, Aniket Maiti *et al.*, Dual-rail encoding with superconducting cavities, Proc. Natl. Acad. Sci. U.S.A. **120**, e2221736120 (2023).

[88] Aleksander Kubica, Arbel Haim, Yotam Vaknin, Harry Levine, Fernando Brandão, and Alex Retzker, Erasure qubits: Overcoming the T1 limit in superconducting circuits, Phys. Rev. X **13**, 041022 (2023).

[89] Shuo Ma, Genyue Liu, Pai Peng, Bichen Zhang, Sven Jandura, Jahan Claes, Alex P. Burgers, Guido Pupillo, Shruti Puri, and Jeff D. Thompson, High-fidelity gates and mid-circuit erasure conversion in an atomic qubit, Nature **622**, 279 (2023).

[90] Pascal Scholl, Adam L. Shaw, Richard Bing-Shiun Tsai, Ran Finkelstein, Joonhee Choi, and Manuel Endres, Erasure conversion in a high-fidelity Rydberg quantum simulator, Nature **622**, 273 (2023).

[91] Harry Levine, Arbel Haim, Jimmy S. C. Hung, Nasser Alidoust, Mahmoud Kalaee, Laura DeLorenzo, E. Alex Wollack, Patricio Arrangoiz-Arriola, Amirhossein Khalajhedayati, Rohan Sanil *et al.*, Demonstrating a long-coherence dual-rail erasure qubit using tunable transmons, Phys. Rev. X **14**, 011051 (2024).

[92] Aruku Senoo, Alexander Baumgärtner, Joanna W. Lis, Gaurav M. Vaidya, Zhongda Zeng, Giuliano Giudici, Hannes Pichler, and Adam M. Kaufman, High-fidelity entanglement and coherent multi-qubit mapping in an atom array, arXiv:2506.13632.

[93] Bichen Zhang, Genyue Liu, Guillaume Bornet, Sebastian P. Horvath, Pai Peng, Shuo Ma, Shilin Huang, Shruti Puri,

and Jeff D. Thompson, Leveraging erasure errors in logical qubits with metastable ^{171}Yb atoms, arXiv:2506.13724.
[94] Markus Grassl, Th. Beth, and Thomas Pellizzari, Codes for the quantum erasure channel, Phys. Rev. A **56**, 33 (1997).
[95] Kaavya Sahay, Junlan Jin, Jahan Claes, Jeff D. Thompson, and Shruti Puri, High-threshold codes for neutral-atom qubits with biased erasure errors, Phys. Rev. X **13**, 041013 (2023).
[96] Gefen Baranes, Madelyn Cain, J. Ataides, Dolev Bluvstein, Josiah Sinclair, Vladan Vuletic, Hengyun Zhou, and Mikhail D. Lukin, Leveraging atom loss errors in fault tolerant quantum algorithms, arXiv:2502.20558.
[97] We note that these thresholds can be slightly improved by using depth-optimized syndrome extraction circuits (we sequentially extract the X-syndrome and then the Z-syndrome).
[98] Austin G. Fowler, Matteo Mariantoni, John M. Martinis, and Andrew N. Cleland, Surface codes: Towards practical large-scale quantum computation, Phys. Rev. A **86**, 032324 (2012).
[99] M. Saffman, T. G. Walker, and K. Mølmer, Quantum information with Rydberg atoms, Rev. Mod. Phys. **82**, 2313 (2010).
[100] Adam M. Kaufman and Kang-Kuen Ni, Quantum science with optical tweezer arrays of ultracold atoms and molecules, Nat. Phys. **17**, 1324 (2021).
[101] D. Bluvstein, A. Omran, H. Levine, A. Keesling, G. Semeghini, S. Ebadi, T. T. Wang, A. A. Michailidis, N. Maskara, W. W. Ho, S. Choi, M. Serbyn, M. Greiner, V. Vuletić, and M. D. Lukin, Controlling quantum many-body dynamics in driven Rydberg atom arrays, Science **371**, 1355 (2021).
[102] Iris Cong, Harry Levine, Alexander Keesling, Dolev Bluvstein, Sheng-Tao Wang, and Mikhail D Lukin, Hardware-efficient, fault-tolerant quantum computation with Rydberg atoms, Phys. Rev. X **12**, 021049 (2022).
[103] Dolev Bluvstein, Harry Levine, Giulia Semeghini, Tout T. Wang, Sepehr Ebadi, Marcin Kalinowski, Alexander Keesling, Nishad Maskara, Hannes Pichler, Markus Greiner *et al.*, A quantum processor based on coherent transport of entangled atom arrays, Nature **604**, 451 (2022).
[104] Alec Jenkins, Joanna W. Lis, Aruku Senoo, William F. McGrew, and Adam M. Kaufman, Ytterbium nuclear-spin qubits in an optical tweezer array, Phys. Rev. X **12**, 021027 (2022).
[105] Jérôme Beugnon, Charles Tuchendler, Harold Marion, Alpha Gaëtan, Yevhen Miroshnychenko, Yvan R. P. Sortais, Andrew M. Lance, Matthew P. A. Jones, Gaétan Messin, Antoine Browaeys, and Philippe Grangier, Two-dimensional transport and transfer of a single atomic qubit in optical tweezers, Nat. Phys. **3**, 696 (2007).
[106] Qian Xu, J. Pablo Bonilla Ataides, Christopher A. Pattison, Nithin Raveendran, Dolev Bluvstein, Jonathan Wurtz, Bane Vasić, Mikhail D. Lukin, Liang Jiang, and Hengyun Zhou, Constant-overhead fault-tolerant quantum computation with reconfigurable atom arrays, Nat. Phys. **20**, 1084 (2024).
[107] Nathan Constantinides, Ali Fahimniya, Dhruv Devulapalli, Dolev Bluvstein, Michael J. Gullans, J. V. Porto, Andrew M. Childs, and Alexey V. Gorshkov, Optimal routing protocols for reconfigurable atom arrays, arXiv:2411.05061.
[108] Aleksander Kubica and John Preskill, Cellular-automaton decoders with provable thresholds for topological codes, Phys. Rev. Lett. **123**, 020501 (2019).
[109] Shankar Balasubramanian, Margarita Davydova, and Ethan Lake, A local automaton for the 2D toric code, arXiv:2412.19803.
[110] Nikolas P. Breuckmann, Kasper Duivenvoorden, Dominik Michels, and Barbara M. Terhal, Local decoders for the 2D and 4D toric code, Quantum Inf. Comput. **17**, 181 (2017).
[111] Thiago Bergamaschi and Yunchao Liu, On Fault Tolerant Single-Shot Logical State Preparation and Robust Long-Range Entanglement, in *16th Innovations in Theoretical Computer Science Conference (ITCS 2025)*, edited by Raghu Meka, Leibniz International Proceedings in Informatics (LIPIcs) (Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2025), Vol. 325, pp. 16:1–16:9.
[112] Yifan Hong, Single-shot preparation of hypergraph product codes via dimension jump, Quantum **9**, 1879 (2025).
[113] Timo Hillmann, Guillaume Dauphinais, Ilan Tzitrin, and Michael Vasmer, Single-shot and measurement-based quantum error correction via fault complexes, Phys. Rev. A **112**, L040401 (2025).
[114] Earl T. Campbell, A theory of single-shot error correction for adversarial noise, Quantum Sci. Technol. **4**, 025006 (2019).
[115] Qian Xu, Hengyun Zhou, Guo Zheng, Dolev Bluvstein, J. P. Bonilla Ataides, Mikhail D. Lukin, and Liang Jiang, Fast and parallelizable logical computation with homological product codes, Phys. Rev. X **15**, 021065 (2025).
[116] See [113,137,138] for recent progress on single-shot lattice surgery for topological codes.
[117] Jinkang Guo, Yifan Hong, Adam Kaufman, and Andrew Lucas, Towards self-correcting quantum codes for neutral atom arrays, 2025, https://github.com/yifanhong/ZSZ-codes-numerics.
[118] D. S. Dummit and R. M. Foote, *Abstract Algebra* (Wiley, Hoboken, NJ, 2004), 3rd ed.
[119] Nikolas P. Breuckmann and Jens N. Eberhardt, Balanced product quantum codes, IEEE Trans. Inf. Theory **67**, 6653 (2021).
[120] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller, Equation of state calculations by fast computing machines, J. Chem. Phys. **21**, 1087 (1953),
[121] W. K. Hastings, Monte Carlo sampling methods using Markov chains and their applications, Biometrika **57**, 97 (1970).
[122] Roy J. Glauber, Time-dependent statistics of the Ising model, J. Math. Phys. **4**, 294 (1963),
[123] K. Temme, T. J. Osborne, K. G. Vollbrecht, D. Poulin, and F. Verstraete, Quantum Metropolis sampling, Nature **471**, 87 (2011).
[124] Chi-Fang Chen, Michael J. Kastoryano, and András Gilyén, An efficient and exact noncommutative quantum Gibbs sampler, arXiv:2311.09207.

[125] Zhiyan Ding, Bowen Li, and Lin Lin, Efficient Quantum Gibbs Samplers with Kubo–Martin–Schwinger Detailed Balance Condition, Commun. Math. Phys. **406**, 67 (2025).

[126] Jiaqing Jiang and Sandy Irani, Quantum metropolis sampling via weak measurement, arXiv:2406.16023.

[127] Emanuel Knill, Raymond Laflamme, and Lorenza Viola, Theory of quantum error correction for general noise, Phys. Rev. Lett. **84**, 2525 (2000).

[128] Tibor Rakovszky and Vedika Khemani, The physics of (good) LDPC codes II. Product constructions, arXiv:2402.16831.

[129] H. Bombin, R. W. Chhajlany, M. Horodecki, and M. A. Martin-Delgado, Self-correcting quantum computers, New J. Phys. **15**, 055023 (2013).

[130] Tom Richardson and Ruediger Urbanke, *Modern Coding Theory* (Cambridge University Press, 2008).

[131] G. A. Margulis, Explicit group-theoretical constructions of combinatorial schemes and their application to the design of expanders and concentrators, Problemy Peredachi Informatsii **24**, 51 (1988).

[132] R. Tanner, A recursive approach to low complexity codes, IEEE Trans. Inf. Theory **27**, 533 (1981).

[133] Michael Capalbo, Omer Reingold, Salil Vadhan, and Avi Wigderson, in *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '02 (Association for Computing Machinery, New York, NY, USA, 2002), p. 659.

[134] A. Lubotzky and B. Weiss, Groups and expanders, in *Expanding graphs (Princeton, NJ, 1992)*, edited by J. Friedman, DIMACS Ser. Discrete Math. Theoret. Comput. Sci. (American Mathematical Society, Providence, RI, 1993), Vol. 10, pp. 95–109.

[135] R. L. Brooks, On colouring the nodes of a network, Math. Proc. Cambr. Philos. Soc. **37**, 194 (1941).

[136] P. O. Boykin and V. P. Roychowdhury, Reversible Fault-Tolerant Logic, in *Proc. 2005 Int. Conf. Dependable Systems and Networks (DSN 2005/DSN'05)* (Yokohama, Japan, 2005), pp. 444–453.

[137] David Aasen, Jeongwan Haah, Matthew B. Hastings, and Zhenghan Wang, Geometrically enhanced topological quantum codes, arXiv:2505.10403.

[138] David Aasen, Matthew B. Hastings, Vadym Kliuchnikov, Juan M. Bello-Rivas, Adam Paetznick, Rui Chao, Ben W. Reichardt, Matt Zanner, Marcus P. da Silva, Zhenghan Wang, and Krysta M. Svore, A topologically fault-tolerant quantum computer with four dimensional geometric codes, arXiv:2506.15130.