

Planar Quantum Low-Density Parity-Check Codes with Open Boundaries

Zijian Liang (梁子健)¹, Jens Niklas Eberhardt^{2,*} and Yu-An Chen (陳昱安)^{1,†}

¹International Center for Quantum Materials, School of Physics, [Peking University](#), Beijing 100871, China

²Institute of Mathematics, [Johannes Gutenberg-Universität Mainz](#), 55128 Mainz, Germany

 (Received 4 July 2025; revised 21 August 2025; accepted 7 October 2025; published 12 November 2025)

Although high-threshold and low-overhead quantum low-density parity-check (qLDPC) codes, such as bivariate bicycle (BB) codes, can reduce the physical-qubit cost by an order of magnitude compared to the Kitaev toric code, their torus layout remains difficult for physical implementation. In this work, we introduce the first systematic procedure to convert BB codes into fully planar, open-boundary qLDPC codes, preserving their performance. We present planar code families with logical dimensions $6 \leq k \leq 13$, e.g., $[[78, 6, 6]]$, $[[107, 7, 7]]$, $[[268, 8, 12]]$, $[[405, 9, 15]]$, $[[348, 10, 13]]$, $[[450, 11, 15]]$, $[[386, 12, 12]]$, and $[[362, 13, 11]]$, all with geometrically local weight-6 stabilizers. Allowing weight-8 stabilizers produces a $[[282, 12, 14]]$ code, exhibiting an efficiency metric (kd^2/n) an order of magnitude higher than the surface code. The construction combines boundary-anyon condensation with the “lattice-grafting” optimization, yielding high-performance qLDPC codes natively compatible with planar hardware architectures. It also uncovers Sierpinski-type fractal logical operators the distance of which scales with the fractal area on finite lattices. These planar qLDPC codes provide an implementable route to resource-efficient, high-threshold fault tolerance and a flexible framework for future code design on realistic two-dimensional hardware.

DOI: [10.1103/qv65-vmzr](https://doi.org/10.1103/qv65-vmzr)

I. INTRODUCTION

The realization of large-scale fault-tolerant quantum computers relies on quantum error correction (QEC) to protect fragile quantum information from noise [1–5]. While the surface code has emerged as a main candidate due to its planar geometry and local stabilizer checks [6–17], its resource overhead remains a significant challenge, requiring potentially millions of physical qubits for practical algorithms [18,19]. For the rotated surface code [20], the scaling is limited by the ratio $kd^2/n \approx 1$, where n , k , and d denote the numbers of physical qubits, logical qubits, and the code distance, respectively. This motivates the search for alternative quantum codes that are also implementable with local interactions on a two-dimensional (2D) planar architecture but offer better resource efficiency.

While general quantum low-density parity-check (qLDPC) codes can achieve linear scaling of both the number of logical qubits k and the code distance d with the number of physical qubits n [21–26], 2D local qLDPC

codes are restricted by the Bravyi-Poulin-Terhal bound [27,28]:

$$\frac{kd^2}{n} \in \mathcal{O}(1). \quad (1)$$

Within this constraint, we aim to significantly improve the *constant factor* over the surface code. In this context, bivariate bicycle (BB) codes have recently attracted considerable attention. With a fixed parity-check weight of 6, they can achieve a ratio of $kd^2/n \approx 10$ on a torus, thereby representing an order-of-magnitude improvement over the toric code [29–46]. However, implementing torus geometry on planar hardware (e.g., superconducting architectures) is challenging, as it requires nonlocal stabilizer checks that undermine the geometric locality advantage of the surface code.

To overcome the geometric constraints of BB codes, we will use insights from condensed-matter physics, specifically the theory of topological order. In this framework, 2D translation-invariant stabilizer codes, including BB codes, can be understood as equivalent (via finite-depth local quantum circuits) to multiple copies of the Kitaev toric code—the standard $\mathbb{Z}_2$ lattice gauge theory [47–51]. This allows us to employ topological quantum field theory (TQFT) concepts—such as partition functions (capturing logical dimensions), anyon Wilson lines (representing logical operators), and, most importantly, anyon condensation of a Lagrangian subgroup at gapped boundaries (defining

*Contact author: mail@jenseberhardt.com

†Contact author: yuanchen@pku.edu.cn

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.

the boundary-stabilizer Hamiltonian) [52–91]. We utilize the TQFT perspective to develop a systematic approach for adapting BB codes defined on a torus to planar geometries with open boundaries. Our approach ensures that the resulting codes achieve the maximal logical dimensions predicted by TQFT, preserve the locality and weight of stabilizers, and exhibit code distances that scale with the length of the system. Moreover, we introduce a “lattice-grafting” technique that reduces the number of physical qubits by removing boundary edges while preserving all other code properties.

We explicitly demonstrate our approach by constructing families of planar qLDPC codes derived from BB codes. Notably, all stabilizer checks in these planar codes remain local, with check weights not exceeding 6, and the ratio kd^2/n outperforms the surface code by an order of magnitude. Moreover, the resulting codes offer more flexible lattice sizes than the BB codes designed on a torus. This represents the first construction, to our knowledge, of 2D local qLDPC codes exhibiting such a substantial advantage over the surface code [92]. We remark that our $[[288, 8, 12]]$ code has inspired the work of Steffan *et al.* [93], which has extended our construction with higher-weight stabilizers to obtain different code parameters.

Furthermore, our analysis reveals intriguing properties of the logical operators in these planar qLDPC codes. In finite-sized systems relevant for near-term implementations (e.g., 100 ~ 500 qubits), we observe that the minimal-weight logical operators exhibit fractal structures, reminiscent of Sierpinski triangles embedded within the finite lattice. This fractal regime, where the code distance is proportional to the fractal area, precedes the expected emergence of stringlike logical operators in the thermodynamic limit [94–98].

In the future, the open boundaries inherent to our construction will not only facilitate physical implementation but potentially enable advanced QEC techniques such as lattice surgery [19,99,100] and magic state distillation [101,102]. Moreover, investigating the transition between the fractal and stringlike regimes could provide helpful intuition for understanding the code distances of these 2D qLDPC codes [103]. Overall, our work provides a strategy for designing high-performance, hardware-compatible quantum error-correcting codes, paving the way toward more resource-efficient fault-tolerant quantum computation on realistic boundary-constrained architectures.

A. Summary of results

We present explicit families of high-performance planar qLDPC codes with open boundaries that outperform the surface code in resource efficiency (measured by kd^2/n) while preserving geometric locality suitable for 2D architectures. Our approach systematically adapts BB codes

from the torus to planar geometries via boundary-anyon condensation motivated by topological order, and further refines them through a “lattice-grafting” optimization.

As a primary example, Sec. II constructs the $[[288, 8, 12]]$ code on a 12×12 open square lattice (Fig. 1), and explicitly identifies its lowest-weight logical operators, which follow a Sierpinski-triangle pattern.

Section III outlines the construction, derives logical operators from the anyon viewpoint, compares our codes with generalized toric codes, describes the code search procedure, and introduces the “lattice-grafting” optimization. For small fixed logical dimension k and code distance d , Table I reports the minimal n and the corresponding stabilizers realizing $[[n, k, d]]$ planar codes produced by our method (prior to grafting).

Section IV analyzes planar code families with $6 \leq k \leq 9$, and Appendix D covers families with $10 \leq k \leq 13$. Focusing on stabilizers with weight ≤ 6 , representative planar codes include:

- (a) $k = 6$ (Fig. 8): $[[54, 6, 4]]$, $[[88, 6, 6]]$, $[[247, 6, 12]]$.
- (b) $k = 7$ (Fig. 7): $[[131, 7, 7]]$, $[[205, 7, 10]]$, $[[337, 7, 14]]$.
- (c) $k = 8$ (Fig. 6): $[[188, 8, 9]]$, $[[326, 8, 13]]$, $[[368, 8, 15]]$.
- (d) $k = 9$ (Figs. 14 and 15): $[[225, 9, 9]]$, $[[265, 9, 10]]$, $[[432, 9, 14]]$, $[[441, 9, 15]]$.
- (e) $k = 10$ (Figs. 16 and 17): $[[226, 10, 9]]$, $[[324, 10, 11]]$, $[[381, 10, 13]]$, $[[490, 10, 15]]$.
- (f) $k = 11$ (Fig. 18): $[[248, 11, 9]]$, $[[494, 11, 15]]$.
- (g) $k = 12$ (Figs. 19 and 20): $[[279, 12, 9]]$, $[[432, 12, 12]]$, $[[486, 12, 13]]$, $[[588, 12, 15]]$.
- (h) $k = 13$ (Figs. 21 and 22): $[[286, 13, 9]]$, $[[392, 13, 11]]$, $[[495, 13, 13]]$, $[[648, 13, 16]]$.

Sec. IV D additionally introduces a $k = 12$ code family with weight-8 stabilizers (Fig. 9), including $[[240, 12, 12]]$, $[[292, 12, 14]]$, and $[[399, 12, 18]]$.

Applying lattice grafting (Sec. III E) preserves locality and stabilizer weight while reducing n . For the examples above, the grafted parameters are summarized below (see Appendix E for detailed constructions):

$$\begin{aligned}
 [[54, 6, 4]] &\longrightarrow [[44, 6, 4]] : \frac{kd^2}{n} = 2.18, \\
 [[88, 6, 6]] &\longrightarrow [[78, 6, 6]] : \frac{kd^2}{n} = 2.77, \\
 [[131, 7, 7]] &\longrightarrow [[107, 7, 7]] : \frac{kd^2}{n} = 3.21, \\
 [[188, 8, 9]] &\longrightarrow [[173, 8, 9]] : \frac{kd^2}{n} = 3.75, \\
 [[288, 8, 12]] &\longrightarrow [[268, 8, 12]] : \frac{kd^2}{n} = 4.30, \\
 [[441, 9, 15]] &\longrightarrow [[405, 9, 15]] : \frac{kd^2}{n} = 5.00,
 \end{aligned}$$

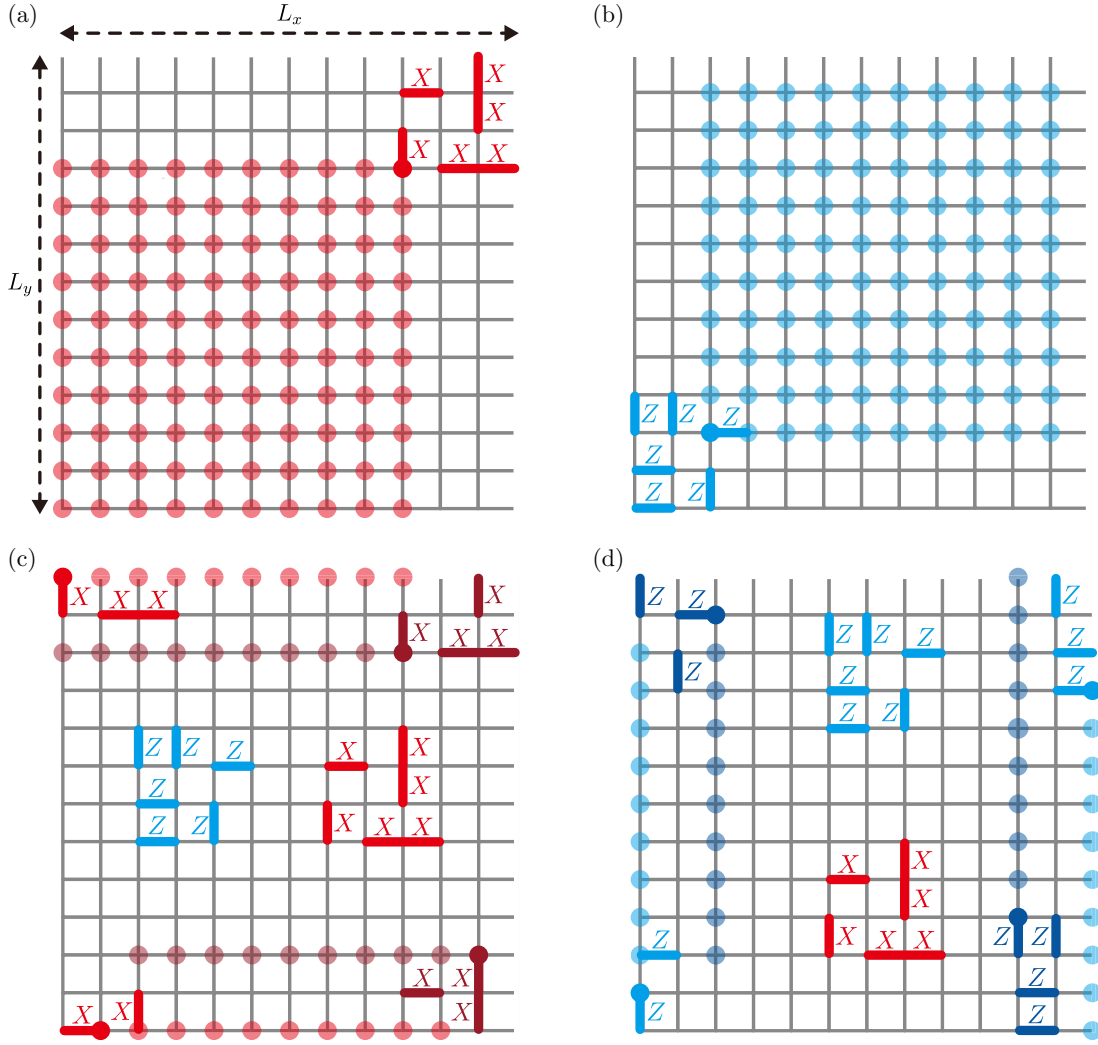


FIG. 1. The $[[288, 8, 12]]$ code on an open square lattice. Bulk stabilizers are specified by the polynomials $f(x, y) = x(1 + x + x^{-1}y^2)$ and $g(x, y) = x^2y(1 + y + x^{-2}y^{-1})$ (see Appendix A for a review of the polynomial formalism). (a) The bulk X -stabilizers. Each red dot represents a weight-6 Pauli operator. With translational symmetry, there are 100 distinct bulk X -stabilizers. (b) 100 bulk Z -stabilizers arranged similarly. (c) Boundary X -stabilizers near the top and bottom boundaries. Two distinct types are present at each boundary, exhibiting translational symmetry (ten stabilizers each), resulting in 40 boundary X -stabilizers. (d) 40 boundary Z -stabilizers near the left and right boundaries. The independence of these bulk and boundary stabilizers yields a total logical dimension of $k = n - \# \text{ of stabilizers} = 288 - 2 \times 100 - 2 \times 40 = 8$. The lattice sizes L_x and L_y are flexible, generating a code family with fixed logical dimension $k = 8$: $[[2L_x L_y, 8, d(L_x, L_y)]]$. The code distances $d(L_x, L_y)$ are listed in Table V in Appendix D, and are computed exactly using the integer-programming approach [31,104].

$$[[381, 10, 13]] \longrightarrow [[348, 10, 13]] : \frac{kd^2}{n} = 4.86,$$

$$[[494, 11, 15]] \longrightarrow [[450, 11, 15]] : \frac{kd^2}{n} = 5.50,$$

$$[[432, 12, 12]] \longrightarrow [[386, 12, 12]] : \frac{kd^2}{n} = 4.48,$$

$$[[392, 13, 11]] \longrightarrow [[362, 13, 11]] : \frac{kd^2}{n} = 4.35,$$

$$[[292, 12, 14]] \longrightarrow [[282, 12, 14]] : \frac{kd^2}{n} = 8.34.$$

We conclude with a discussion in Sec. V and provide tables of code parameters in the appendices.

II. $[[288, 8, 12]]$ PLANAR CODE

In this section, we begin with the construction of the $[[288, 8, 12]]$ code, illustrated in Fig. 1. We consider a truncated square lattice with a single qubit at each edge. Both bulk and boundary stabilizers are explicitly depicted. These stabilizers exhibit translational symmetry: bulk stabilizers are uniformly distributed across the 2D plane,

TABLE I. For each fixed logical dimension k and code distance d , we enumerate all families of bulk stabilizers of the form $f(x, y) \propto 1 + x + x^a y^b$ and $g(x, y) \propto 1 + y + x^c y^d$, with $-5 \leq a, b, c, d \leq 6$ (see Appendix A for stabilizer conventions), to identify quantum $[[n, k, d]]$ codes with the minimal number of physical qubits n . Details of the boundary and corner stabilizer construction are provided in Sec. III D. Each entry lists the minimal n , followed by the corresponding bulk stabilizers; the second and third lines indicate the terms $x^a y^b$ and $x^c y^d$, respectively. Entries sharing the same color correspond to the same bulk stabilizers. In each family, the logical dimension k is fixed, while the code distance $d(L_x, L_y)$ depends on the system size. The exact code distances are computed using the integer-programming approach [104]. Our results show that, for each k , one or two dominant families provide the best performance. These optimal families, spanning $k = 6$ to $k = 13$, are presented in Sec. IV and Appendix D.

$k \backslash d$	4	5	6	7	8	9	10	11	12	13	14	15	16
6	54 $x^{-1}y^{-2}$ xy^{-1}	77 $x^{-1}y^{-2}$ xy^{-1}	88 $x^{-1}y^{-2}$ xy^{-1}	117 $x^{-1}y^{-2}$ xy^{-1}	150 $x^{-1}y^{-2}$ xy^{-1}	165 $x^{-1}y^{-2}$ xy^{-1}	204 $x^{-1}y^{-2}$ xy^{-1}	228 $x^{-1}y^{-2}$ xy^{-1}	247 $x^{-1}y^{-2}$ xy^{-1}	294 $x^{-1}y^{-2}$ xy^{-1}	345 $x^{-1}y^{-2}$ xy^{-1}	400 $x^{-1}y^{-2}$ xy^{-1}	425 $x^{-1}y^{-2}$ xy^{-1}
7	55 $x^{-1}y$ xy^3	89 $x^{-1}y$ xy^3	118 $x^{-1}y$ xy^3	131 $x^{-1}y$ xy^3	166 $x^{-1}y$ xy^3	202 $x^{-2}y^{-1}$ $x^{-1}y^2$	205 $x^{-1}y$ xy^3	248 $x^{-1}y$ xy^3	267 $x^{-1}y$ xy^3	316 $x^{-1}y$ xy^3	337 $x^{-1}y$ xy^3	392 $x^{-1}y$ xy^3	451 $x^{-1}y$ xy^3
8	59 $x^{-1}y^2$ xy^3	94 $x^{-1}y^2$ xy^3	105 $x^{-1}y^2$ xy^3	137 $x^{-1}y^2$ xy^3	173 $x^{-1}y^2$ xy^3	188 $x^{-1}y^2$ xy^3	230 $x^{-1}y^2$ xy^3	276 $x^{-1}y^2$ xy^3	288 $x^{-1}y^2$ $x^{-2}y^{-1}$	326 $x^{-1}y^2$ xy^3	347 $x^{-1}y^2$ xy^3	368 $x^{-1}y^2$ xy^3	426 $x^{-1}y^2$ xy^3
9	78 $x^{-3}y^{-1}$ xy^{-1}	110 $x^{-1}y^3$ xy^3	132 $x^{-1}y^3$ xy^3	164 $x^{-3}y^{-1}$ xy^{-1}	204 $x^{-3}y^{-1}$ xy^{-1}	225 $x^{-1}y^3$ xy^3	265 $x^{-3}y^{-1}$ xy^{-1}	334 $x^{-3}y^{-1}$ xy^{-1}	353 $x^{-3}y^{-1}$ xy^{-1}	411 $x^{-3}y^{-1}$ xy^{-1}	432 $x^{-3}y^{-1}$ xy^{-1}	441 $x^{-1}y^3$ xy^3	506 $x^{-1}y^3$ xy^3
10	78 $x^{-2}y^{-2}$ $x^{-2}y^2$	119 $x^{-2}y^{-2}$ $x^{-2}y^2$	133 $x^{-1}y^2$ xy^4	170 $x^{-1}y^2$ xy^4	211 $x^{-1}y^2$ xy^4	226 $x^{-1}y^2$ xy^4	273 $x^{-1}y^2$ xy^4	324 $x^{-2}y^{-2}$ $x^{-2}y^2$	362 $x^{-1}y^2$ xy^4	381 $x^{-1}y^2$ xy^4	442 $x^{-1}y^2$ xy^4	490 $x^{-2}y^{-2}$ $x^{-2}y^2$	530 $x^{-1}y^2$ xy^4
11	79 $x^{-2}y^2$ xy^3	127 $x^{-1}y^{-3}$ xy^{-3}	138 $x^{-1}y^{-3}$ xy^{-3}	176 $x^{-1}y^{-3}$ xy^{-3}	233 $x^{-1}y^{-3}$ xy^{-3}	248 $x^{-1}y^{-3}$ xy^{-3}	298 $x^{-1}y^{-3}$ xy^{-3}	352 $x^{-1}y^{-3}$ xy^{-3}	371 $x^{-1}y^{-3}$ xy^{-3}	431 $x^{-1}y^{-3}$ xy^{-3}	473 $x^{-1}y^{-3}$ xy^{-3}	494 $x^{-1}y^{-3}$ xy^{-3}	564 $x^{-1}y^{-3}$ xy^{-3}
12	105 $x^{-1}y^2$ xy^5	140 $x^{-1}y^3$ x^2y^3	172 $x^{-1}y^2$ xy^5	216 $x^{-1}y^2$ xy^5	264 $x^{-1}y^2$ xy^5	279 $x^{-1}y^2$ xy^5	330 $x^{-1}y^3$ x^2y^3	384 $x^{-1}y^3$ x^2y^3	432 $x^{-1}y^3$ x^2y^3	486 $x^{-1}y^2$ xy^5	560 $x^{-1}y^3$ x^2y^3	588 $x^{-1}y^3$ x^2y^3	660 $x^{-1}y^3$ x^2y^3
13	98 $x^{-2}y^2$ x^2y^3	145 $x^{-2}y^2$ x^2y^3	177 x^5y x^3y^{-1}	222 x^5y x^3y^{-1}	271 x^5y x^3y^{-1}	286 x^5y x^3y^{-1}	341 x^5y x^3y^{-1}	392 $x^{-2}y^2$ x^2y^3	476 x^5y x^3y^{-1}	495 x^5y x^3y^{-1}	578 $x^{-2}y^2$ x^2y^3	610 x^5y x^3y^{-1}	648 $x^{-2}y^2$ x^2y^3

while boundary stabilizers can shift along directions parallel to the boundary, provided that they remain fully supported within the finite lattice. Specifically, the top and bottom boundaries contain Pauli- X operators, and the left and right boundaries host Pauli- Z operators. The boundary stabilizers are carefully designed to ensure that they commute with each other around the four corners. These stabilizers are independent; hence, the logical dimension k is equal to the number of physical qubits n minus the number of stabilizers: $k = 288 - 2 \times 100 - 2 \times 40 = 8$, as illustrated in Fig. 1.

The 12×12 square lattice can be readily generalized to any $L_x \times L_y$ rectangle by inserting additional rows or columns, yielding a family of qLDPC codes with $n = 2L_x L_y$ physical qubits and a fixed logical dimension of $k = 8$. Representative examples of this family on small

lattices include:

- (1) $L_x = L_y = 6$: $[[72, 8, 4]]$, $kd^2/n = 1.78$.
- (2) $L_x = L_y = 8$: $[[128, 8, 6]]$, $kd^2/n = 2.25$.
- (3) $L_x = L_y = 10$: $[[200, 8, 9]]$, $kd^2/n = 3.24$.
- (4) $L_x = L_y = 12$: $[[288, 8, 12]]$, $kd^2/n = 4$.
- (5) $L_x = L_y = 13$: $[[338, 8, 13]]$, $kd^2/n = 4$.
- (6) $L_x = L_y = 14$: $[[392, 8, 15]]$, $kd^2/n = 4.59$.

A comprehensive table detailing the code distances d for each combination of L_x and L_y is provided in Appendix D. The exact code distances have been computed using the integer-programming approach [104].

For the $[[288, 8, 12]]$ code, in Fig. 2 we depict the corresponding nontrivial logical operators with the lowest weight. Notably, rather than exhibiting the string operator

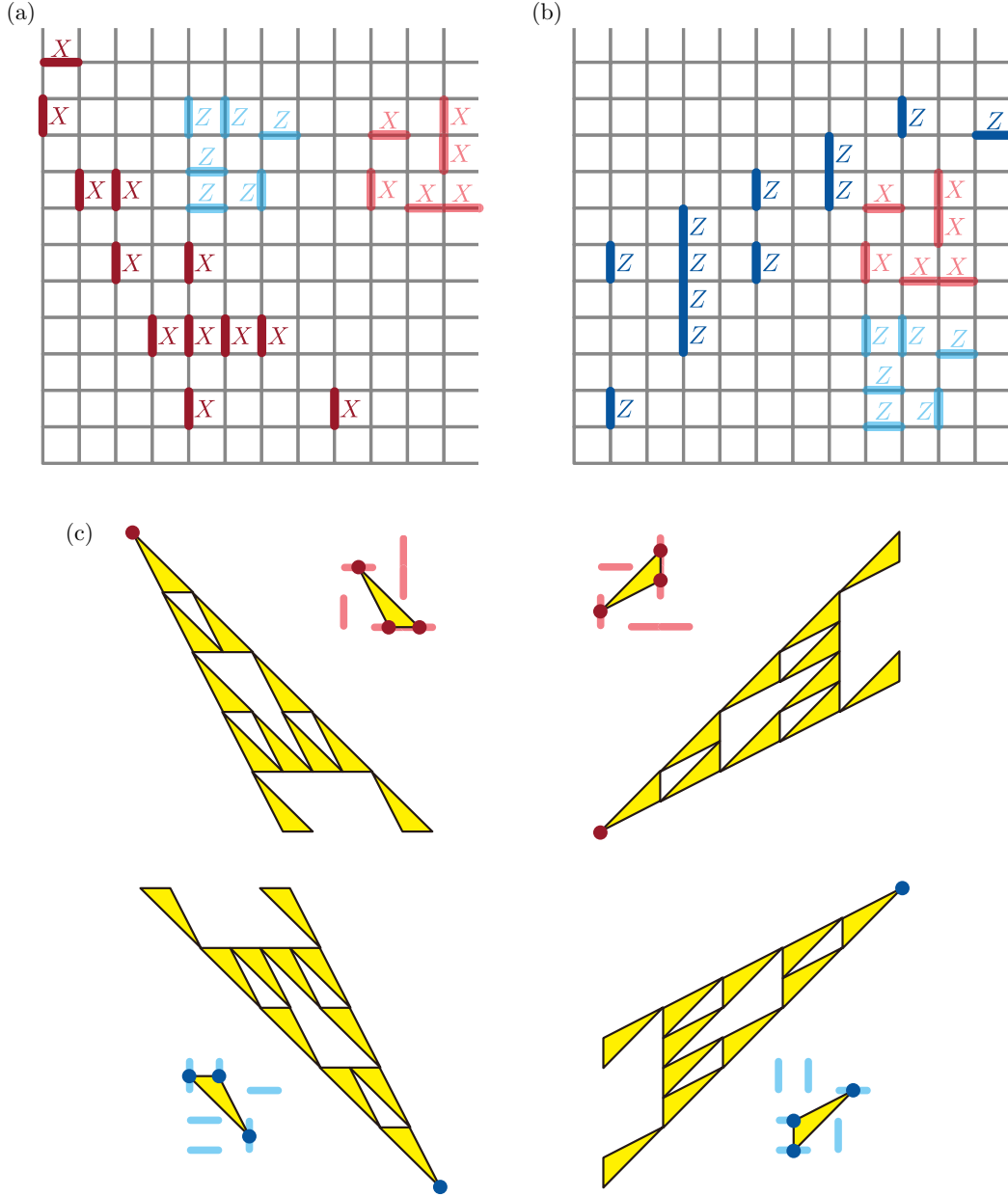


FIG. 2. Illustrations of the lowest-weight logical operators and the emergence of the fractal structure in the $[[288, 8, 12]]$ code. (a) The lowest-weight nontrivial logical X operator (deep red). (b) The lowest-weight nontrivial logical Z operator (deep blue). (c) The truncated Sierpinski triangle corresponding to the bulk stabilizers.

anticipated from the anyon theory (unitary modular tensor category [105–108]), the logical operators manifest as fractal patterns. More precisely, they follow the Sierpinski triangle that has emerged from the stabilizers. Although for a large system size L , the weight of a string operator, $\Omega(L)$, is asymptotically more favorable than that of a Sierpinski-triangle operator with the weight $\Omega(L^{1.59})$ [109], in our finite system the Sierpinski-triangle operators have lower weight compared to the string operator.

We verify that the numerically obtained code distances match the number of small triangles of the Sierpinski

fractal embedded in the open square lattice. More precisely, each single Pauli operator violates three stabilizers, corresponding to the vertices of a small yellow triangle in Fig. 2(c). A natural way to cancel these violations is to construct a semi-infinite Sierpinski triangle, which leaves only a single violation at its apex. By combining two such Sierpinski triangles, the remaining violations at their apexes cancel, as illustrated in Fig. 3. When the combined operator is embedded into the finite lattice such that it intersects the boundary only along the top and bottom edges, the resulting truncated operator commutes with all stabilizers

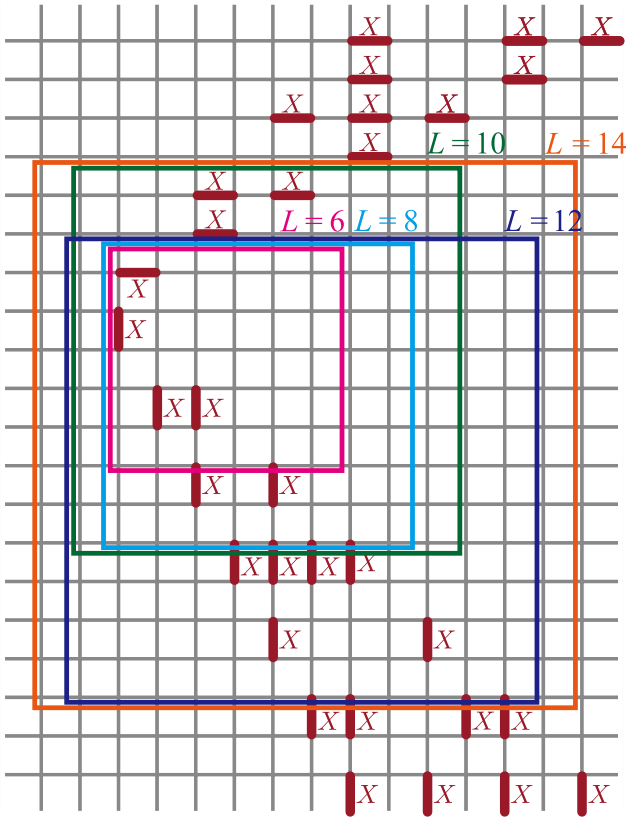


FIG. 3. The red edges depict a fractal operator that transports an anyon across the infinite plane. On finite lattices, logical operators arise from truncating this infinite fractal operator. The minimal number of X -edges (with positions defined at their mid-points) enclosed within $L \times L$ squares for $L = 6, 8, 10, 12$, and 14 are $4, 6, 9, 12$, and 15 , respectively, as shown in the figure (with the constraint that the fractal operator may exit the finite square only through its top and bottom boundaries). These values exactly match the code distances $d(L_x, L_y)$ of the $[[288, 8, 12]]$ code family listed in Table V.

and thus forms a logical operator. Because this operator transports an anyon (i.e., the violation), it is nontrivial, and its weight provides an upper bound for the code distance. Remarkably, for the small lattices listed in Table V, this bound is tight; the code distance exactly equals the number of small triangles from the combined Sierpinski triangles contained within the finite lattice.

As shown in Fig. 3, the minimal number of small triangles enclosed by an $L \times L$ square for $L = 6, 8, 10, 12$, and 14 is $4, 6, 9, 12$, and 15 , respectively. These values precisely match the code distances of the representative examples discussed earlier. This correspondence demonstrates that these planar codes, constructed on small open square lattices, remain in the “fractal regime” and have not yet reached the thermodynamic limit characterized by string operators. Such Sierpinski-type fractal logical operators commonly arise whenever the stabilizers are weight-6, with three Pauli operators acting on horizontal edges and

three on vertical edges. For the bulk stabilizers of the $[[288, 8, 12]]$ code, string operators are stable at $L_x = L_y = 217$ (the $(-2, 2)$ -BB code in Ref. [98]) [110].

Finally, we emphasize that these codes can be further optimized using “lattice grafting,” a technique that removes qubits near the boundary, which will be discussed in Sec. III E. Our numerical experiments indicate that this approach can reduce the total number of qubits n by $O(L_x + L_y)$. For example, by applying lattice grafting to the $[[288, 8, 12]]$ code to remove boundary qubits, we obtain the $[[268, 8, 12]]$ code, as illustrated in Fig. 26 in Appendix E.

III. PLANAR CODES FROM TOPOLOGICAL ORDER

In this section, we introduce our general approach to constructing planar qLDPC codes applicable to arbitrary planar bulk stabilizers from a topological-order viewpoint. We begin in Sec. III A by reviewing the concepts of boundary gauge operators and boundary anyons, which allow us to achieve anyon condensation on the boundary and ensure that the topological-order (TO) condition is satisfied. In Sec. III B, we detail the layout of interleaving $\{m_i\}$ -condensed and $\{e_i\}$ -condensed boundaries, and examine the geometric structure of logical operators associated with these boundaries. Section III C provides a comparative analysis between open and periodic boundary conditions, and discusses their implications for designing high-performance qLDPC codes. Lastly, in Sec. III E, we introduce lattice grafting, which enables us to further reduce the number of physical qubits in the examples presented in the previous section without altering any other parameters.

A. Boundary gauge operators and anyons

This section reviews the essential terminology from Ref. [98] that underpins our construction of planar qLDPC codes. We begin by defining the notion of the topological-order condition.

Definition 1. The *topological-order (TO) condition* [111] for a stabilizer code requires that any local operator $\mathcal{O}$ that commutes with all stabilizers must itself be a local product of stabilizers. That is, if $[\mathcal{O}, S_i] = 0$ for all stabilizers S_i , then there exists a finite subset A of stabilizers such that

$$\mathcal{O} = \prod_{i \in A} S_i. \quad (2)$$

Starting with a qLDPC code on a torus, one can truncate it to an open lattice by retaining only those bulk stabilizers that are fully supported within the finite lattice. Near the boundaries, however, the TO condition is violated, meaning that there exist local operators commuting with all bulk

stabilizers that are not products of these stabilizers. These operators are referred to as *boundary gauge operators*. More precisely, the set of local operators that commute with the bulk stabilizers, modulo the bulk stabilizers, forms the group $\mathcal{G}$ of boundary gauge operators. Note that these operators may not necessarily commute with each other [112]. Therefore, to ensure that the boundary is gapped, we need to incorporate a *subset* of boundary gauge operators into the Hamiltonian such that they mutually commute and restore the TO condition near the boundary. In the language of subsystem codes, this step is called “gauge fixing” [113–115]. What we seek is an effective way to perform this gauge fixing so that the resulting stabilizer codes have favorable code parameters.

In this framework, bulk and boundary anyons can be defined as follows [45,51,97,98,116].

Definition 2. A *bulk anyon* is defined as a finite violation of stabilizers on the lattice. It is a homomorphism ϕ from the stabilizer group $\mathcal{S}$ to $\mathbb{Z}_2$:

$$\phi : \mathcal{S} \rightarrow \mathbb{Z}_2 = \{1, -1\}, \quad (3)$$

such that $\phi(S_i) = 1$ for all but a finite number of $S_i \in \mathcal{S}$.

Definition 3. A *boundary anyon* is defined as the local “syndrome pattern” of boundary gauge operators that indicates how boundary gauge operators are violated:

$$\varphi : \mathcal{G} \rightarrow \mathbb{Z}_2 = \{1, -1\}, \quad (4)$$

such that the homomorphism $\varphi(G_i) = 1$ for all but a finite number of $G_i \in \mathcal{G}$.

By quotienting out trivial anyons—those created by local operators—the resulting equivalence classes form a finite Abelian fusion group. Nontrivial anyons can be realized at the end points of string operators, corresponding to twisted sectors under the 1-form symmetries generated by closed anyon world lines. In particular, boundary anyons arise at the end points of a *boundary string operator*—a product of boundary gauge operators along a boundary segment. There exists a one-to-one correspondence between bulk and boundary anyons, allowing boundary anyons to move into the bulk and vice versa [98].

Each gapped boundary in an Abelian topological order (i.e., anyon theory $\mathcal{A}$) is associated with a *Lagrangian subgroup* $\mathcal{L} \subset \mathcal{A}$, defined as the maximal set of bosonic anyons that braid trivially with one another [117–120]. The condensation of the bosons in $\mathcal{L}$ on the boundary allows their string operators to terminate at the boundary without any energy cost. Constructing the boundary-stabilizer Hamiltonian involves two steps:

- (1) *Boundary-anyon condensation.* Incorporate mutually commuting short boundary string operators

corresponding to the bosons in the Lagrangian subgroup into the Hamiltonian.

- (2) *Topological-order (TO) completion.* Introduce extra local boundary gauge operators into the Hamiltonian to ensure that the topological-order condition near the boundary is fully satisfied.

Although the second step is overlooked in the literature, it is critical for constructing a lattice Hamiltonian with gapped boundaries, particularly because not all boundary gauge operators arise from truncating bulk stabilizers. For a detailed discussion of these two steps and a comprehensive algorithm outlining the procedure, see Ref. [98].

B. Planar code construction and logical operators

Any 2D translation-invariant stabilizer code satisfying the topological-order condition is equivalent to a direct sum of Kitaev toric codes via a finite-depth Clifford circuit [47–51,121]. Consequently, the corresponding topological order (anyon theory) is characterized by the fusion group $A = \mathbb{Z}_2^q \times \mathbb{Z}_2^q$, generated by the anyons $\{e_i\}$ and $\{m_i\}$ for $1 \leq i \leq q$. These anyons are bosons with the braiding statistics

$$B(e_i, m_i) = -1, \quad \forall 1 \leq i \leq q, \quad (5)$$

while $B(a, b) = 1$ for all other pairs.

Although the general procedure applies to arbitrary stabilizer codes, we focus on Calderbank-Shor-Steane (CSS) codes for simplicity. The planar code layout in Fig. 4 depicts an open square with two distinct Lagrangian subgroups arranged on opposite edges: $\mathcal{L}_1 = \{e_i\}$ (violations of X -stabilizers) and $\mathcal{L}_2 = \{m_i\}$ (violations of Z -stabilizers). More intricate geometries can be considered, such as puncturing additional holes into the open square or considering polygons with interleaving Lagrangian subgroups (e.g., $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$, etc.) [20]. These approaches are standard techniques for increasing the logical dimension in surface codes, but the code performance ratio kd^2/n usually decreases. Therefore, we focus on the simplest layout in Fig. 4.

According to the procedure outlined in the previous section, one should first identify the boundary anyons $\{e_i\}$ or $\{m_i\}$, incorporate their corresponding short boundary operators as stabilizers, and then perform topological-order completion to eliminate any local logical operators. However, for an arbitrary CSS code as depicted in Fig. 4, the boundary construction algorithm can be simplified. Specifically, along the left and right boundaries, all $\{e_i\}$ -anyons are generated exclusively by Z -type operators, allowing the direct imposition of all Z -type boundary gauge operators as stabilizers. Similarly, along the top and bottom boundaries, the $\{m_i\}$ -anyons are created solely by X -type operators, enabling the direct inclusion of all X -type boundary gauge operators as stabilizers. Consequently,

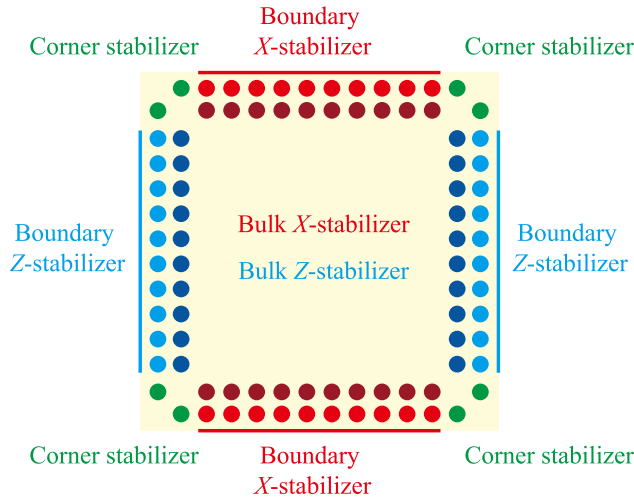


FIG. 4. Planar Calderbank-Shor-Steane (CSS) code construction. The $\{e_i\}$ -anyons are condensed along the left and right boundaries, while the $\{m_i\}$ -anyons are condensed along the top and bottom boundaries. In addition to incorporating the short boundary string operators of these bosons into the stabilizer group, extra local boundary gauge operators are introduced near the boundaries and corners to remove any local logical operators. This crucial “topological-order completion” step ensures that the code distance scales with the system size. The $\{m_i\}$ -anyon strings can terminate on the top and bottom boundaries without any energy cost, and similarly, the $\{e_i\}$ -anyon bulk string operators can terminate on the left and right boundaries. Vertical $\{m_i\}$ -strings and horizontal $\{e_i\}$ -strings serve as the logical X and Z operators, respectively, so that the logical dimension k equals the number of $\{e_i\}$ -anyons.

topological-order completion is required only in the vicinity of the four corners to resolve the anticommutation between the X and Z boundary gauge operators.

In this setting, the logical X operators correspond to $\{m_i\}$ -string operators, with one end point on the top boundary and the other on the bottom boundary. In practice, we first solve the anyon equation [97] to obtain the vertical X -type anyon string operators (which transport $\{m_i\}$ -anyons) on an infinite plane. These operators are then embedded into the finite lattice by truncating the portions that lie outside it. This embedding yields logical operators because the truncation does not affect the bulk stabilizers, ensuring that the truncated string operators continue to commute with all bulk stabilizers on the finite lattice. Moreover, the top and bottom boundaries consist solely of boundary X -stabilizers, which naturally commute with the truncated X -string operators. A similar argument applies to the horizontal $\{e_i\}$ -string operators truncated by the left and right boundaries; therefore, logical dimension k equals the number of independent $\{e_i\}$ -anyons. Algebraically, the number of $\{e_i\}$ -anyons can be computed as [45]

$$k_{\text{open}} = \dim \left(\frac{\mathbb{Z}_2[x, y, x^{-1}, y^{-1}]}{\langle f(x, y), g(x, y) \rangle} \right), \quad (6)$$

where $f(x, y)$ and $g(x, y)$ are the defining polynomials of the bulk stabilizers, labeling the positions of the horizontal and vertical edges of each stabilizer [97]. Intuitively, a single Pauli operator can violate the stabilizers at the locations specified by $f(x, y)$ or $g(x, y)$, creating a trivial anyon. Consequently, an equivalence class of anyons corresponds to the violation pattern in Eq. (3), represented as a polynomial in $\mathbb{Z}_2[x, y, x^{-1}, y^{-1}]$, modulo the trivial anyon configurations defined by $f(x, y)$ and $g(x, y)$. Notably, the logical dimension k is independent of the system size as long as the top and bottom boundaries condense $\{m_i\}$ -anyons and the left and right boundaries condense $\{e_i\}$ -anyons.

Finally, we emphasize that the term “string operators” above refers to operators that move anyons. It has been shown that, in 2D systems, every particle excitation can be moved by a string operator [51]. Thus, our construction of planar qLDPC codes can be scaled to arbitrarily large systems, with the code distance growing proportionally to the system size in the thermodynamic limit. However, in small lattices, the “string operators” may not exhibit a conventional stringlike form. For instance, in Fig. 2(c) we illustrate a fractal pattern for moving an anyon, demonstrating the emergence of fractal operators.

C. Comparison between open and periodic boundary conditions

The same stabilizer code can exhibit markedly different behavior depending on whether open or periodic boundary conditions are imposed. The code parameters $[[n, k, d]]$ can vary significantly:

- (1) *Logical dimension k .* For open boundary conditions, all anyon string operators are explicitly realized on a finite lattice, regardless of the shape or size of the lattice (provided that the lattice is larger than the support of the stabilizers). This is characterized by Eq. (6). In contrast, under periodic boundary conditions, the logical dimension is given by [45, 48]

$$k_{\text{torus}} = 2 \dim \left(\frac{\mathbb{Z}_2[x, y, x^{-1}, y^{-1}]}{\langle f(x, y), g(x, y), x^{L_x} - 1, y^{L_y} - 1 \rangle} \right), \quad (7)$$

where $f(x, y)$ and $g(x, y)$ are the polynomials of the bulk stabilizers implemented on an $L_x \times L_y$ torus. Two key distinctions emerge:

- (a) The factor of 2 reflects the fact that on a torus, vertical logical operators can be realized as both e -string and m -string operators, whereas open boundaries permit only one type of logical operator.

- (b) On a torus, sites are identified modulo $(L_x, 0)$ and $(0, L_y)$, requiring the quotient of the Laurent polynomial ring by $x^{L_x} - 1$ and $y^{L_y} - 1$. Consequently, achieving the maximal logical dimension requires a torus of an appropriate size.

For example, our planar $[[288, 8, 12]]$ code shares the same bulk stabilizers as the $(-2, 2)$ -BB code in Ref. [98]. On a torus, achieving the maximal $k = 16$ would require a lattice of size $L_x = L_y = 217$ [122]. Thus, while toric codes can support a larger logical dimension, they impose stringent requirements on lattice size. In contrast, open lattices, though limited to half the maximal logical dimension, do not require compatibility between the bulk stabilizers and the system size, offering a more practical alternative. This trade-off illustrates that, despite their reduced logical dimension, planar qLDPC codes on open lattices offer greater flexibility in lattice size.

- (2) *Code distance d .* On an open lattice, anyons can be pumped from one side to the other, and the corresponding operator embedded in the finite lattice serves as a logical operator of the planar qLDPC code. As shown in Fig. 2, the lowest-weight logical operators in small lattices take the form of a Sierpinski triangle, with the code distance corresponding to the number of such triangles embedded in the lattice. If the range of the stabilizers is reduced (resulting in smaller triangles), more triangles can be accommodated within the finite lattice, leading to larger code distances for a fixed lattice size.

In contrast, the code distances of toric codes are determined by the string operators that transport anyons vertically and horizontally [123]. Consequently, the sparsity or density of Pauli operators in these string operators is the key factor in determining the code distance. This criterion differs from planar qLDPC codes with open boundaries, implying that the optimal families for planar and toric codes are distinct. For example, the well-known toric code, the $[[144, 12, 12]]$ code from Ref. [31], does not yield a favorable family of planar codes.

- (3) *Physical qubits n .* For open boundaries, it is necessary to complete the stabilizer Hamiltonian at the corners to ensure the topological-order condition. Multiple options exist for these corner terms, some of which may involve only a single Pauli operator. Although selecting such single-qubit corner operators can reduce the number of physical qubits, it may also decrease the code distance. To balance these trade-offs, we prefer higher-weight corner terms to preserve the code distance. Next, we apply lattice grafting (see Sec. III E) to further reduce the number of physical qubits while maintaining the code

distance. This approach is feasible because most boundary edges participate in only a few stabilizers, so their removal and the subsequent rearrangement of stabilizers do not degrade the code distance or the locality of the stabilizers. Consequently, the number of physical qubits, n , is flexible and can often be significantly reduced. For example, our $[[288, 8, 12]]$ planar code can be reduced to a $[[268, 8, 12]]$ code without affecting the locality or the maximal weight of the stabilizers. Under periodic boundary conditions, however, the number of physical qubits is fixed by the lattice dimensions as $n = 2L_xL_y$, leaving no room for further optimization.

D. Enumerating planar qLDPC codes on open square lattices

In this section, we describe our strategy for identifying promising families of planar qLDPC codes. In the toric code case, previous works [45, 123] have focused on a specific form—referred to as the generalized toric code—defined by

$$\begin{aligned} f(x, y) &= 1 + x + x^a y^b, \\ g(x, y) &= 1 + y + x^c y^d. \end{aligned} \quad (8)$$

This form of stabilizer can be viewed as the Kitaev toric code augmented with two extra edges, and it has been shown to yield optimal toric codes on twisted tori for $n \leq 400$. For simplicity, we focus on this particular family of bulk stabilizers, from which we construct the corresponding boundary Hamiltonian and evaluate their performance.

The procedure is summarized as follows:

- (1) *Bulk stabilizer.* Choose a bulk stabilizer with fixed parameters $-5 \leq a, b, c, d \leq 6$. We multiply by appropriate monomials in $f(x, y)$ and $g(x, y)$ to eliminate any negative powers of x or y . For instance, the $[[288, 8, 12]]$ code defined in Sec. II uses the following normalization:

$$\begin{aligned} 1 + x + x^{-1}y^2 &\rightarrow x(1 + x + x^{-1}y^2), \\ 1 + y + x^{-2}y^{-1} &\rightarrow x^2y(1 + y + x^{-2}y^{-1}). \end{aligned} \quad (9)$$

This renders the bulk stabilizers more geometrically local, which in turn facilitates the subsequent boundary construction. Note that the logical dimension k is determined from Eq. (6).

- (2) *Boundary-anyon condensation.* Consider a finite $L_x \times L_y$ square lattice. Away from the corners, at the left and right boundaries, we determine all Z boundary gauge operators and incorporate them into the Hamiltonian in a vertically translation-invariant fashion. Similarly, at the top and bottom boundaries

away from the corners, we include all X boundary gauge operators with horizontal translation symmetry as stabilizers [124].

- (3) *Corner topological-order completion.* The exclusion of boundary terms near the corners causes the presence of local nontrivial logical operators. We remedy this issue by promoting these local operators to corner stabilizers, using the topological-order completion algorithm described in Ref. [98]. There is some flexibility in this step: given local logical X and Z operators, one may choose either to serve as the stabilizer. Based on our experiments, we prefer to select the operator with larger weight as the stabilizer, as it is more likely to enhance the code distance (choosing the lower-weight operator might allow its combination with other logical operators, thereby reducing their effective weights and the code distance). This choice typically affects the code distance d only by $O(1)$.
- (4) *Removing single-qubit stabilizers.* In the process described above, some boundary and corner stabilizers may act on a single qubit. Since these can be disentangled from the system, we remove them from the lattice, thereby reducing the total number of physical qubits $n(L_x, L_y)$.
- (5) *Code-distance table $d(L_x, L_y)$.* For a fixed bulk stabilizer (i.e., fixed a , b , c , and d), the previous steps yield a family of planar qLDPC codes parametrized by the lattice dimensions, due to translation symmetry (one may insert an arbitrary number of rows or columns). We compute the code distances for various values of L_x and L_y , collecting the data in the form $\llbracket n(L_x, L_y), k, d(L_x, L_y) \rrbracket$.
- (6) *Optimal n for fixed k and d .* After enumerating the parameters a , b , c , and d over the finite range, we obtain a large set of $\llbracket n, k, d \rrbracket$ data. We then address the following question: for fixed k and d , which family of bulk stabilizers minimizes the number of physical qubits n ? The optimal values of n for each small k and d and the corresponding bulk stabilizers are listed in Table I.

Caveat. The boundary and corner stabilizers obtained from our computation differ slightly from the examples presented in previous sections. The lattice derived from the computation is often more irregular, as the topological-order-completion algorithm at the corners does not preserve translational symmetry along the boundary, and some single-qubit stabilizers may be removed. For clarity and convenience, we manually reconstruct this family to ensure that the lattice corners have regular shapes and that all corner stabilizers arise from the translation of boundary stabilizers. In particular, all boundary and corner stabilizers are obtained by truncating the bulk stabilizers, simplifying the presentation of our planar codes. Although

the details around the corners differ, the overall performance remains essentially the same, particularly after the “lattice-grafting” step described in the next section.

E. Lattice grafting

We now explain how to further optimize the planar qLDPC with open boundary as considered in Secs. IV and III. For this we present an algorithm that successively removes qubits from the boundary of the codes while recombining the stabilizers supported on these qubits to ensure that they still commute. The algorithm of recombining the stabilizers is reminiscent of grafting a plant—hence the name lattice grafting.

In detail, the algorithm takes as an input planar code with open boundaries and performs the following steps:

- (1) *Removing a physical qubit.* Choose a physical qubit A in the boundary of the lattice which is contained in a minimal number of stabilizers. Then remove the qubit from the code.
- (2) *Grafting.* Choose a Pauli label $O \in \{X, Z\}$. Choose an enumeration $[S_1, S_2, S_3, \dots, S_r]$ of all O -stabilizers which were supported on the qubit A . Replace these O -stabilizers in the code by the stabilizers $[S_1 S_2, S_2 S_3, \dots, S_{r-1} S_r]$.
- (3) *Checking parameters.* Compute the weight of the new stabilizers and the distance of the code. If the weight increased or the distance decreased, undo the last two steps.
- (4) *Iterating.* Repeat with step 1 until all qubits have been checked.

Pruning ensures that the stabilizers in the code constructed in each iteration still commute and that the number of stabilizers is decreased by one. The number of logicals stays constant since in each iteration the numbers of qubits and stabilizers are each decreased by one.

Lattice pruning is an iterative and nondeterministic algorithm. It involves, in every step, the choice of a qubit, a Pauli label, and the enumeration of stabilizers. Each choice potentially affects the successive steps. Iterating over all possibilities is not feasible. Hence, in practice, we perform the algorithm many times with randomized choices and then pick the best result.

The general lattice-grafting procedure can affect the locality and weight of stabilizers because it generates new stabilizers as products of the original ones (e.g., $S_1 S_2$). Consequently, for bivariate bicycle codes defined on a torus, lattice grafting reduces the number of physical qubits but sacrifices the locality and LDPC properties of the stabilizers. This approach is thus not suitable for (generalized) toric codes. In contrast, for our planar qLDPC codes with open boundaries, many qubits near the boundary participate in only one X - or Z -stabilizer ($r = 1$ in step 2 above). In such cases, we can simply remove the corresponding

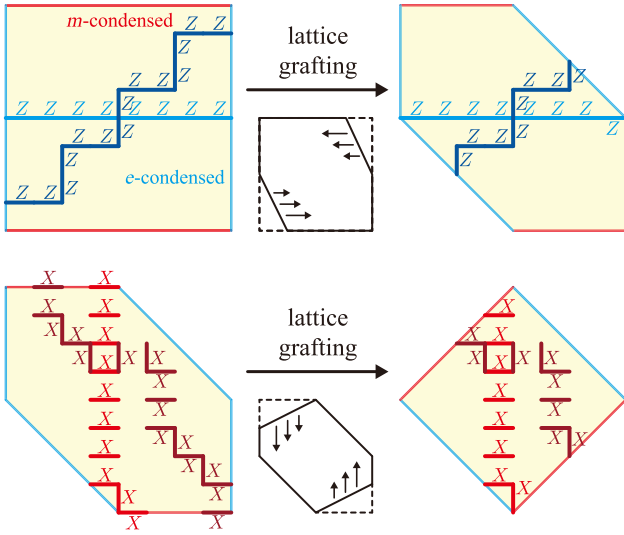


FIG. 5. Lattice-grafting construction of the rotated surface code. Starting from the standard surface code—with e -condensed left or right boundaries and m -condensed top or bottom boundaries—the logical Z operator can be realized either as a straight e -string (blue) or as a zig-zag operator (deep blue) with a higher weight. By iteratively removing qubits from the top-right and bottom-left corners, we shrink the zig-zag logical operator until its weight matches that of the straight string, without altering the code distance. Applying the same removal process to the top-left and bottom-right corners for the logical X operator, which could be a (dual) straight string (red) or a zig-zag path (deep red), produces the grafted lattice that is exactly the rotated surface code.

stabilizer without introducing any new ones, thereby preserving both locality and the LDPC structure (although we must verify that the code distance d remains unaffected). We restrict ourselves to this scenario when optimizing the previously presented examples of planar qLDPC codes. This strategy is feasible only under open boundary conditions, where the boundary qubits are sufficiently “loose” to allow their removal without impacting the parameters k or d , locality, or the overall LDPC properties. Explicit examples of grafted qLDPC codes are provided in Appendix E. Each code has been subjected to 1000 runs of the lattice grafting optimization, and we report the grafted code with the minimal n obtained.

Our lattice-grafting procedure already covers the “rotation” trick for the surface code: the rotated surface code follows directly from the standard surface code via the grafting steps illustrated in Fig. 5. In that setting, one can manually “cut” the lattice due to the simple geometry of the logical operators, while preserving the code distance. By contrast, the logical operators in our planar qLDPC constructions, derived from BB codes, have far more intricate support. As a result, we rely on an automated lattice-grafting algorithm to remove redundant qubits and verify that the code distance remains unchanged in the pruned lattice.

IV. PLANAR qLDPC CODES

In this section, we explicitly construct three families of planar qLDPC codes, with examples including the $[[188, 8, 9]]$ code in Sec. IV A, the $[[131, 7, 7]]$ code in Sec. IV B, and the $[[88, 6, 6]]$ code in Sec. IV C. Additional planar qLDPC code families, such as the $[[381, 10, 13]]$, $[[494, 11, 15]]$, $[[432, 12, 12]]$, and $[[392, 13, 11]]$ codes, are presented in Appendix D. All of these code families feature weight-6 stabilizers. Finally, we show that by allowing weight-8 stabilizers, the code parameters can be further improved, yielding a $[[292, 12, 14]]$ code, as discussed in Sec. IV D.

We consider a translation-invariant $\mathbb{Z}_2$ CSS code defined on the square lattice, with bulk stabilizer generators

$$A_v = \begin{bmatrix} f(x, y) \\ g(x, y) \\ 0 \\ 0 \end{bmatrix}, \quad B_p = \begin{bmatrix} 0 \\ 0 \\ \frac{g(x, y)}{f(x, y)} \\ f(x, y) \end{bmatrix}, \quad (10)$$

using the notation from Ref. [97]. In these examples, the boundary stabilizers are obtained by truncating the bulk stabilizer generators at the lattice boundary.

A. $[[188, 8, 9]]$ planar code

The $[[188, 8, 9]]$ code is implemented on an 8×13 square lattice, as illustrated in Fig. 6. In our construction, the lattice sizes L_x and L_y are adjustable. Note that the lattice deviates from a perfect square lattice, as certain qubits along the right and bottom boundaries have been omitted. Consequently, the number of physical qubits is given by

$$n = 2L_x L_y - L_x - L_y + 1. \quad (11)$$

The logical dimension remains fixed at $k = 8$, determined by subtracting the number of independent stabilizers from the total number of physical qubits. A table of the code distances $d(L_x, L_y)$ is provided in Table IV in Appendix D, and a few representative cases are listed below:

- (1) $L_x = 6, L_y = 10$: $[[105, 8, 6]]$, $kd^2/n = 2.74$.
- (2) $L_x = 7, L_y = 11$: $[[137, 8, 7]]$, $kd^2/n = 2.86$.
- (3) $L_x = 8, L_y = 12$: $[[173, 8, 8]]$, $kd^2/n = 2.96$.
- (4) $L_x = 8, L_y = 13$: $[[188, 8, 9]]$, $kd^2/n = 3.45$.
- (5) $L_x = 9, L_y = 14$: $[[230, 8, 10]]$, $kd^2/n = 3.48$.
- (6) $L_x = 10, L_y = 15$: $[[276, 8, 11]]$, $kd^2/n = 3.51$.
- (7) $L_x = 10, L_y = 16$: $[[295, 8, 12]]$, $kd^2/n = 3.91$.
- (8) $L_x = 11, L_y = 16$: $[[326, 8, 13]]$, $kd^2/n = 4.15$.
- (9) $L_x = 11, L_y = 17$: $[[347, 8, 14]]$, $kd^2/n = 4.52$.
- (10) $L_x = 11, L_y = 18$: $[[368, 8, 15]]$, $kd^2/n = 4.89$.

This family of planar qLDPC codes has the same logical dimension $k = 8$ as the previously discussed $[[288, 8, 12]]$ code in Sec. II, and generally exhibits

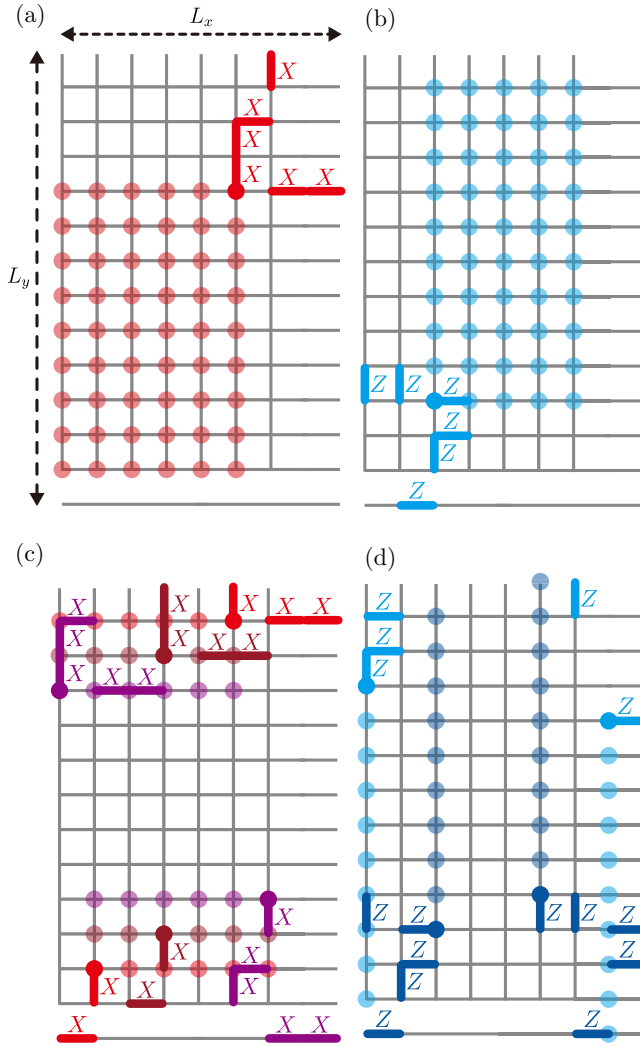


FIG. 6. The $[[188, 8, 9]]$ code on an open square lattice. The vertical edges of the rightmost column and the bottommost row have been removed. The bulk stabilizers are specified by $f(x, y) = x(1 + x + x^{-1}y^2)$ and $g(x, y) = 1 + y + xy^3$. (a) The bulk X -stabilizers consist of 54 Pauli operators that are translations of one another. (b) 50 bulk Z -stabilizers. (c) 36 boundary X -stabilizers near the top and bottom boundaries. (d) 40 boundary Z -stabilizers appear near the left and right boundaries. The logical dimension is $k = 188 - 54 - 50 - 36 - 40 = 8$. This generates another code family of $k = 8$: $[[2L_x L_y - L_x - L_y + 1, 8, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table IV in Appendix D.

improved performance, except in the $d = 12$ case, as summarized in Table I.

B. $[[131, 7, 7]]$ planar code

The $[[131, 7, 7]]$ code is implemented on a 7×11 square lattice, as shown in Fig. 7. In this construction, the lattice dimensions L_x and L_y are adaptable. Note that some qubits along the rightmost column and the two bottom rows are

removed, resulting in a total of

$$n = 2L_x L_y - 2L_x - L_y + 2, \quad (12)$$

physical qubits, while the logical dimension remains fixed at $k = 7$. The table of code distances $d(L_x, L_y)$ is provided in Table III in Appendix D, with a few representative cases listed below:

- (1) $L_x = 7, L_y = 11$: $[[131, 7, 7]]$, $kd^2/n = 2.62$.
- (2) $L_x = 8, L_y = 12$: $[[166, 7, 8]]$, $kd^2/n = 2.70$.
- (3) $L_x = 9, L_y = 13$: $[[205, 7, 10]]$, $kd^2/n = 3.41$.
- (4) $L_x = 10, L_y = 14$: $[[248, 7, 11]]$, $kd^2/n = 3.42$.
- (5) $L_x = 10, L_y = 15$: $[[267, 7, 12]]$, $kd^2/n = 3.78$.
- (6) $L_x = 11, L_y = 16$: $[[316, 7, 13]]$, $kd^2/n = 3.74$.
- (7) $L_x = 11, L_y = 17$: $[[337, 7, 14]]$, $kd^2/n = 4.07$.

For toric codes, the logical dimension k must be an even integer [45]; therefore, constructing a planar code family with odd k is noteworthy.

C. $[[88, 6, 6]]$ planar code

In Fig. 8, we illustrate the $[[88, 6, 6]]$ planar code implemented on a 6×8 square lattice. Since the qubits along the right boundary are decoupled, the number of physical qubits is given by

$$n = 2L_x L_y - L_y. \quad (13)$$

Within this family, the logical dimension remains fixed at $k = 6$. Table II in Appendix D details the distances $d(L_x, L_y)$, with several representative cases listed below:

- (1) $L_x = 6, L_y = 8$: $[[88, 6, 6]]$, $kd^2/n = 2.45$.
- (2) $L_x = 8, L_y = 10$: $[[150, 6, 8]]$, $kd^2/n = 2.56$.
- (3) $L_x = 8, L_y = 11$: $[[165, 6, 9]]$, $kd^2/n = 2.95$.
- (4) $L_x = 10, L_y = 12$: $[[228, 6, 11]]$, $kd^2/n = 3.18$.
- (5) $L_x = 10, L_y = 13$: $[[247, 6, 12]]$, $kd^2/n = 3.50$.

Compared to the $[[188, 8, 9]]$ code family, for a fixed distance d , the number of physical qubits n is slightly lower in this family; however, the logical dimension decreases from 8 to 6. Consequently, except for the $[[88, 6, 6]]$ code itself, the remaining members of this family exhibit suboptimal performance metrics kd^2/n .

D. $[[292, 12, 14]]$ planar code (weight-8 stabilizers)

Previously, we have focused on weight-6 stabilizers and characterized broad families of planar qLDPC codes. The same methodology extends to weight-8 stabilizers and can yield planar qLDPC codes with improved performance. However, the search space grows exponentially in stabilizer weight, rendering an exhaustive examination for weight-8 codes impractical. To manage the complexity,

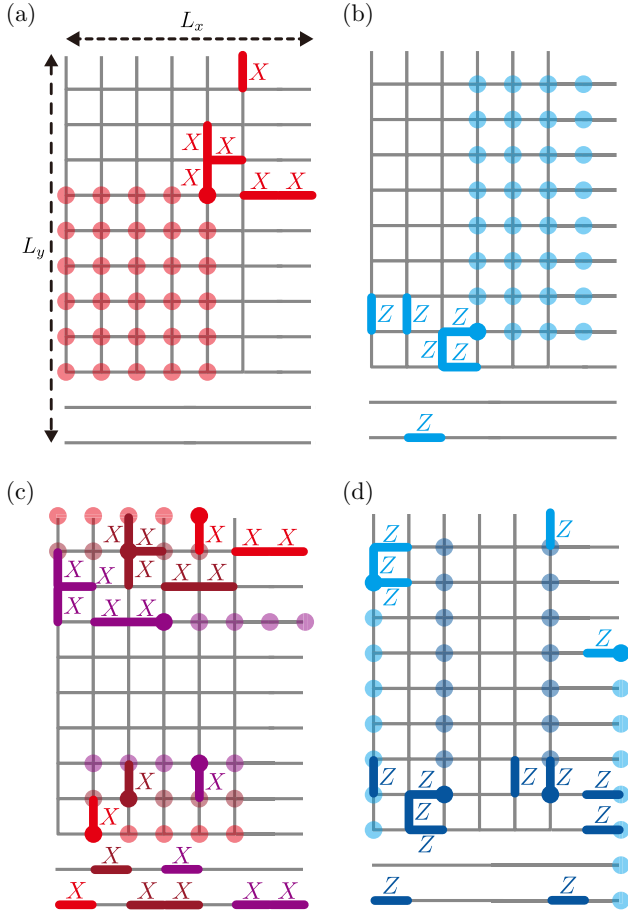


FIG. 7. The $[[131, 7, 7]]$ code on an open square lattice. The bulk stabilizers correspond to $f(x, y) = x(1 + x + x^{-1}y)$ and $g(x, y) = 1 + y + xy^3$. (a) 30 bulk X -stabilizers. (b) 32 bulk Z -stabilizers. (c) 30 boundary X -stabilizers near the top and bottom boundaries. (d) 32 boundary Z -stabilizers appear near the left and right boundaries. The logical dimension is $k = 131 - 30 - 32 - 30 - 32 = 7$. This generates a code family of $k = 7$: $[[2L_xL_y - 2L_x - L_y + 2, 6, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table III in Appendix D.

we therefore constrain our search to codes with fixed logical dimension $k = 12$, which leads to the discovery of a notable code family (see Fig. 9). Importantly, the boundary stabilizers in this family are not mere truncations of bulk operators: the green terms in Fig. 9(e) are secondary boundary gauge operators [98].

A representative member of this family is the $[[292, 12, 14]]$ code on an 8×20 square lattice. In the lattice configuration, the rightmost column of qubits decouples, so the total number of physical qubits is

$$n = 2L_xL_y - L_y. \quad (14)$$

Across this family, the logical dimension remains fixed at $k = 12$.

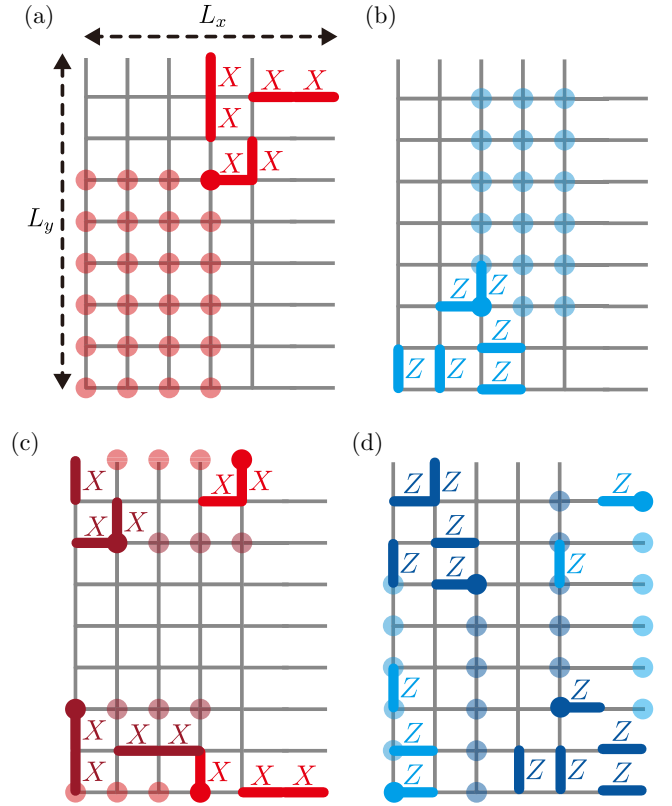


FIG. 8. The $[[88, 6, 6]]$ code on an open square lattice. The bulk stabilizers correspond to $f(x, y) = xy^2(1 + x + x^{-1}y^{-2})$ and $g(x, y) = y(1 + y + xy^{-1})$. (a) 24 bulk X -stabilizers. (b) 18 bulk Z -stabilizers. (c) 16 boundary X -stabilizers near the top and bottom boundaries. (d) 24 boundary Z -stabilizers appear near the left and right boundaries. The logical dimension is $k = 88 - 24 - 18 - 16 - 24 = 6$. This generates a code family of $k = 6$: $[[2L_xL_y - L_y, 6, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table II in Appendix D.

Table VI in Appendix D details the distances $d(L_x, L_y)$, with several representative cases listed below:

- (1) $L_x = 7, L_y = 18$: $[[227, 12, 11]]$, $kd^2/n = 6.40$.
- (2) $L_x = 7, L_y = 19$: $[[240, 12, 12]]$, $kd^2/n = 7.20$.
- (3) $L_x = 8, L_y = 20$: $[[292, 12, 14]]$, $kd^2/n = 8.05$.
- (4) $L_x = 9, L_y = 22$: $[[365, 12, 16]]$, $kd^2/n = 8.42$.
- (5) $L_x = 9, L_y = 23$: $[[382, 12, 17]]$, $kd^2/n = 9.08$.
- (6) $L_x = 9, L_y = 24$: $[[399, 12, 18]]$, $kd^2/n = 9.74$.

V. DISCUSSION AND CONCLUSIONS

In this section, we explore several intriguing properties and aspects of our construction that are worth further investigation.

A. Bulk stabilizers of optimal code families

We have presented the optimal families with logical dimensions ranging from $k = 6$ to $k = 13$. Each family is

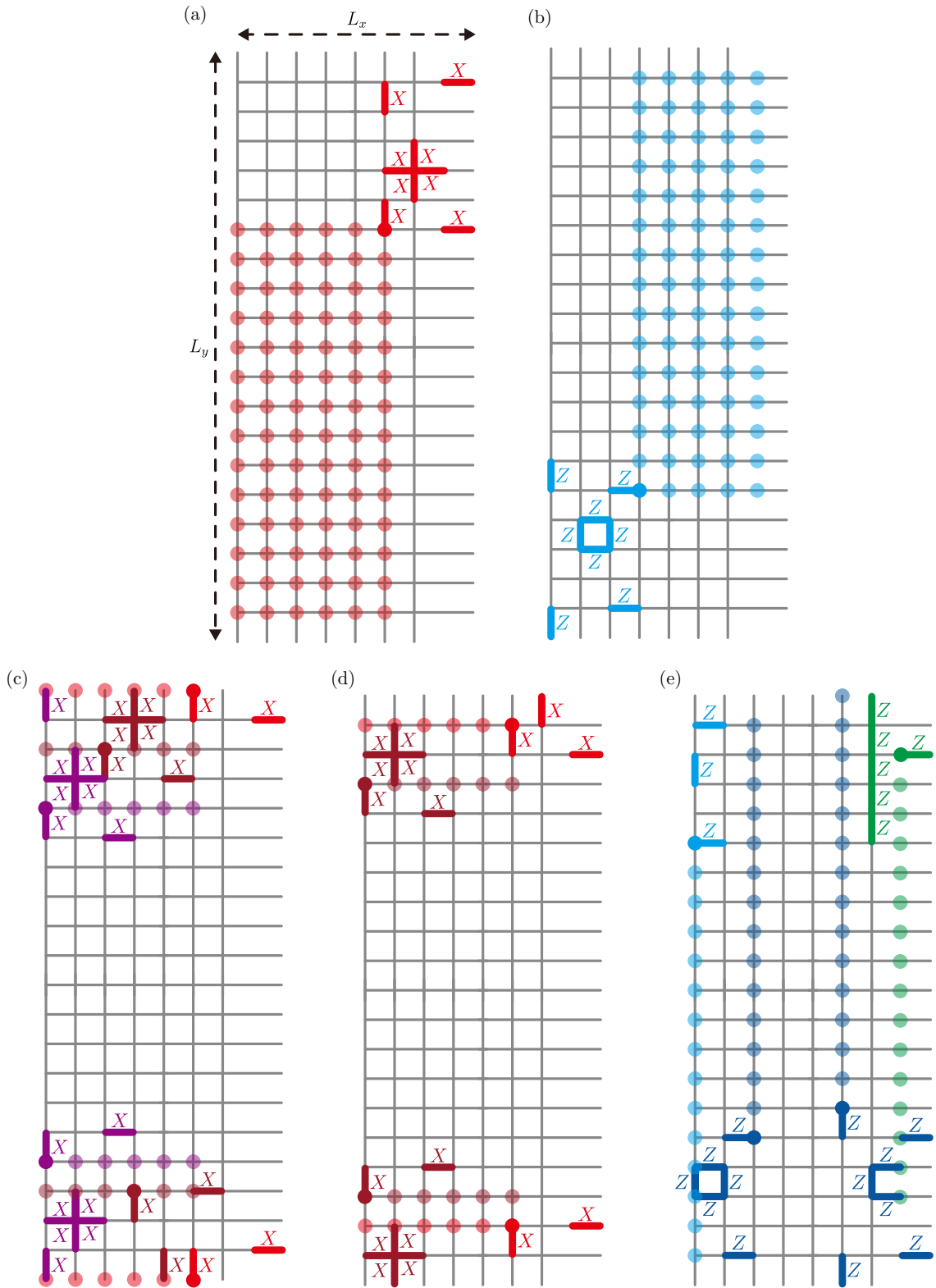


FIG. 9. The $[[292, 12, 14]]$ code on an open square lattice. The bulk stabilizers correspond to $f(x, y) = y^2(1 + x + x^2y^3 + x^2y^{-2})$ and $g(x, y) = xy(1 + y + x^{-1}y^{-1} + x^{-1}y^3)$. (a) 84 bulk X -stabilizers. (b) 75 bulk Z -stabilizers. (c) 36 boundary X -stabilizers near the top and bottom boundaries. (d) 24 boundary X -stabilizers near the top and bottom boundaries. (e) 61 boundary Z -stabilizers appear near the left and right boundaries; notably, the green terms cannot be obtained by truncating any bulk stabilizer. The logical dimension is $k = 292 - 84 - 75 - 36 - 24 - 61 = 12$. This generates a code family of $k = 12$: $[[2L_x L_y - L_y, 12, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table VI in Appendix D.

determined independently since the logical dimension is defined solely by the bulk stabilizers, and families with different k values are not directly compared in Table I. Consequently, we did not expect any relation among the bulk stabilizers of different code families. However, our numerical results reveal that the optimal families for different k share similar stabilizers. For instance, the $k = 6$ family corresponds to the bulk stabilizers

$$S_1^{k=6} = \begin{array}{|c|c|c|} \hline X & & \\ \hline X & -X- & -X- \\ \hline & X & \\ \hline -X & & \\ \hline \end{array}, \quad S_2^{k=6} = \begin{array}{|c|c|c|} \hline & Z & \\ \hline & Z & -Z- \\ \hline & Z & \\ \hline & Z & -Z- \\ \hline \end{array},$$

while the $k = 11$ code family is described by

$$S_1^{k=11} = \begin{array}{|c|c|c|} \hline X & & \\ \hline X & -X- & -X- \\ \hline & & \\ \hline & X & \\ \hline -X & & \\ \hline \end{array}, \quad S_2^{k=11} = \begin{array}{|c|c|c|} \hline & Z & \\ \hline & Z & -Z- \\ \hline & & \\ \hline & Z & -Z- \\ \hline & & \\ \hline \end{array}.$$

Remarkably, $S_1^{k=11}$ and $S_2^{k=11}$ can be interpreted as a vertical extension of $S_1^{k=6}$ and $S_2^{k=6}$, where each part of the stabilizer is shifted farther apart.

Furthermore, we compare the bulk stabilizers for the $k = 7$, $k = 8$ ($[[188, 8, 9]]$), $k = 9$ ($[[441, 9, 15]]$), $k = 10$ ($[[381, 10, 13]]$), $k = 12$ ($[[432, 12, 12]]$), and $k = 13$ ($[[392, 13, 11]]$) families:

$$S_1^{k=7} = \begin{array}{|c|c|c|} \hline X & & \\ \hline X & -X- & -X- \\ \hline & X & \\ \hline -X & & \\ \hline \end{array}, \quad S_2^{k=7} = \begin{array}{|c|c|c|} \hline & Z & \\ \hline & Z & -Z- \\ \hline & & \\ \hline & Z & -Z- \\ \hline \end{array},$$

$$S_1^{k=8} = \begin{array}{|c|c|c|} \hline X & & \\ \hline X & -X- & -X- \\ \hline & X & \\ \hline -X & & \\ \hline \end{array}, \quad S_2^{k=8} = \begin{array}{|c|c|c|} \hline & Z & \\ \hline & Z & -Z- \\ \hline & & \\ \hline & Z & -Z- \\ \hline \end{array},$$

$$S_1^{k=9} = \begin{array}{|c|c|c|} \hline X & & \\ \hline X & -X- & -X- \\ \hline & X & \\ \hline -X & & \\ \hline \end{array}, \quad S_2^{k=9} = \begin{array}{|c|c|c|} \hline & Z & \\ \hline & Z & -Z- \\ \hline & & \\ \hline & Z & -Z- \\ \hline \end{array},$$

$$S_1^{k=10} = \begin{array}{|c|c|c|} \hline X & & \\ \hline X & -X- & -X- \\ \hline & X & \\ \hline -X & & \\ \hline \end{array}, \quad S_2^{k=10} = \begin{array}{|c|c|c|} \hline & Z & \\ \hline & Z & -Z- \\ \hline & & \\ \hline & Z & -Z- \\ \hline \end{array},$$

$$S_1^{k=12} = \begin{array}{|c|c|c|} \hline X & & \\ \hline X & -X- & -X- \\ \hline & X & \\ \hline -X & & \\ \hline \end{array}, \quad S_2^{k=12} = \begin{array}{|c|c|c|} \hline & Z & \\ \hline & Z & -Z- \\ \hline & & \\ \hline & Z & -Z- \\ \hline \end{array},$$

$$S_1^{k=13} = \begin{array}{|c|c|c|} \hline X & & \\ \hline X & -X- & -X- \\ \hline & X & \\ \hline -X & & \\ \hline \end{array}, \quad S_2^{k=13} = \begin{array}{|c|c|c|} \hline & Z & \\ \hline & Z & -Z- \\ \hline & & \\ \hline & Z & -Z- \\ \hline \end{array}.$$

Notably, the $k = 10$ family arises from a vertical extension of the $k = 8$ stabilizers, while the $k = 13$ family results from a horizontal extension of the $k = 8$ stabilizers. Moreover, the stabilizers of the $k = 8$ family are derived from the $k = 7$ family by shifting one edge; similarly, the $k = 9$ stabilizers are formed by shifting one edge of the $k = 8$ stabilizers, and the $k = 12$ stabilizers are generated by an additional one-edge shift of the $k = 9$ stabilizers.

Overall, these observations suggest that the bulk stabilizers are closely related. It would be of great interest to develop an analytical method to obtain these families directly rather than relying solely on numerical search, as this could provide valuable guidelines for designing high-performance families of planar qLDPC codes with larger logical dimensions k . We conjecture that the code families described above achieve an optimal balance between k and the code distance d : the logical dimension k scales with the range of the stabilizers (determined by the independent monomials after quotienting out $f(x, y)$ and $g(x, y)$), while the fractal operators favor smaller stabilizers in order to accommodate more small triangles. In other words, the bulk stabilizers presented above may be arranged to maximize the area of the Newton polytope associated with $f(x, y)$ and $g(x, y)$ [123], while remaining compact enough to allow the embedding of numerous small Sierpinski triangles within finite lattices, thereby achieving larger code distances. It would be worthwhile to formalize this trade-off more rigorously.

B. Comparison with surface code and implementation

Our planar qLDPC codes offer advantage over the surface code in theoretical resource efficiency, with a kd^2/n

ratio an order of magnitude higher. However, their implementation presents different challenges. While the surface code has a planar Tanner graph allowing single-layer 2D hardware implementation with only nearest-neighbor interactions, our codes, despite being 2D local, possess nonplanar Tanner graphs. Still, it seems feasible to implement our codes on a two-layer architecture (e.g., via flip-chip bonding) while maintaining geometric locality within and between the layers. This is more achievable than BB (toric) codes, which require long-range connections due to their periodic boundary conditions [31]. Therefore, our planar construction shifts the implementation challenge primarily to the development of practical two-layer quantum processor architectures. Evaluating the precise layout requirements and feasibility on specific multilayer architectures is an important next step.

C. Extending the numerical search for bulk stabilizers

The numerical search for optimal bulk stabilizers presented in this work has focused on a specific, relatively simple class of polynomials: $f(x, y) \propto 1 + x + x^a y^b$ and $g(x, y) \propto 1 + y + x^c y^d$, with exponents (a, b, c, d) within a limited range. It is natural to ask if further improvements are possible by broadening the search space. Future numerical efforts could explore the following:

- (1) Searching over larger values of a, b, c , and d .
- (2) Investigating polynomials with different structures, such as

$$\begin{aligned} f(x, y) &= 1 + x^{a_1} y^{b_1} + x^{a_2} y^{b_2}, \\ g(x, y) &= 1 + x^{c_1} y^{d_1} + x^{c_2} y^{d_2}, \end{aligned} \quad (15)$$

which corresponds to the same stabilizer weight (i.e., weight-6) but different geometries.

- (3) Exploring stabilizers with weights greater than 6, such as weight-8 operators, while maintaining geometrical locality. This approach could examine the trade-off between stabilizer weight and code distance.

Expanding the search in these directions could lead to the discovery of new code families with even better $[[n, k, d]]$ parameters or improved asymptotic scaling within the $\mathcal{O}(1)$ bound for kd^2/n . Moreover, the relationships between the bulk stabilizers observed in Sec. V A may offer valuable insights for guiding such extended searches.

D. Exploring alternative boundary geometries

Our construction methodology, based on anyon condensation at gapped boundaries, provides a systematic way to define boundaries for planar codes derived from translation-invariant systems. The examples presented

here use standard e - and m -condensed boundaries on a rectangular grid. However, the flexibility of the TQFT framework invites exploration of alternative boundary geometries. For example, one could explore boundaries rotated relative to the lattice axes, implement intricate interleaving of distinct Lagrangian subgroups (e.g., $\mathcal{L}_1, \mathcal{L}_2$, and $\mathcal{L}_3$) along the boundary, or even introduce internal boundaries by puncturing holes within the lattice. Systematically studying these variations could reveal new code families, potentially offer advantages in connectivity or robustness, or enable different types of logical operations. In the future, one could investigate how to tailor code properties through boundary engineering.

E. Performance simulation and threshold estimation

A crucial next step is to evaluate the performance of our codes using numerical simulations, such as belief propagation combined with ordered statistic decoding (BP+OSD). The primary goal of these simulations would be to estimate the pseudocode threshold of our planar code families against relevant noise models (e.g., circuit-level noise). Since our construction preserves both geometric locality and the LDPC structure, we conjecture that the (pseudo)threshold will be comparable to that of the surface code and BB codes, the thresholds of which are around the range of 0.7% to 1.1% [31].

ACKNOWLEDGMENTS

We would like to thank Dave Aasen, Shin Ho Choe, Arpit Dua, Tyler D. Ellison, Yingfei Gu, Francisco Revson F. Pereira, Hao Song, Vincent Steffan, Nathanan Tanti-vasadakarn, Yi-Fei Wang, Dominic J. Williamson, and Bowen Yang for their valuable discussions. Y.-A.C. is supported by the National Natural Science Foundation of China (Grant No. 12474491) and the Fundamental Research Funds for the Central Universities, Peking University. J.N.E. is supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), project number 45744154, “Equivariant K-motives and Koszul duality.”

APPENDIX A: REVIEW OF THE LAURENT POLYNOMIAL FORMALISM

This appendix reviews the Laurent polynomial representation and its application to translation-invariant stabilizer codes. The formalism was first introduced in Refs. [48, 49]. In this work, we adopt the conventions and notations of Refs. [45, 97, 98]. We begin with a general $\mathbb{Z}_d$ qudit system,

where the $d \times d$ generalized Pauli matrices are defined as

$$X = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix},$$

$$Z = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & \omega & 0 & \cdots & 0 \\ 0 & 0 & \omega^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \omega^{d-1} \end{bmatrix}, \quad (\text{A1})$$

with $\omega := \exp(2\pi i/d)$. These matrices satisfy the commutation relation

$$ZX = \omega XZ. \quad (\text{A2})$$

The qubit case is recovered by setting $d = 2$.

We consider the case of two $\mathbb{Z}_d$ qudits per unit cell (e.g., one qudit on each edge of a square lattice), generalizable to k qudits per cell. Any Pauli operator—a finite tensor product of Pauli matrices—can be represented, up to an overall phase, as a column vector over the polynomial ring

$$R = \mathbb{Z}_d[x, y, x^{-1}, y^{-1}], \quad (\text{A3})$$

which includes all polynomials in $x^{\pm 1}, y^{\pm 1}$ with coefficients in $\mathbb{Z}_d$ [48]. We assign column vectors over $\mathbb{Z}_d$ to the (generalized) Pauli matrices X_{12} , Z_{12} , X_{14} , and Z_{14} , depicted in Fig. 10:

$$\mathcal{X}_{12} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathcal{Z}_{12} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathcal{X}_{14} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix},$$

$$\mathcal{Z}_{14} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \quad (\text{A4})$$

where column vector representations of operators are indicated using curly letters. The coefficients in these vectors correspond to their powers:

$$\mathcal{P} = \begin{bmatrix} i \\ j \\ k \\ l \end{bmatrix} \Rightarrow \mathcal{P}^m = \begin{bmatrix} mi \\ mj \\ mk \\ ml \end{bmatrix} \quad \forall m \in \mathbb{Z}_d. \quad (\text{A5})$$

The translation of operators is achieved using polynomials of x and y to denote translations in the x and y directions, respectively. To illustrate, translating the operator on edge

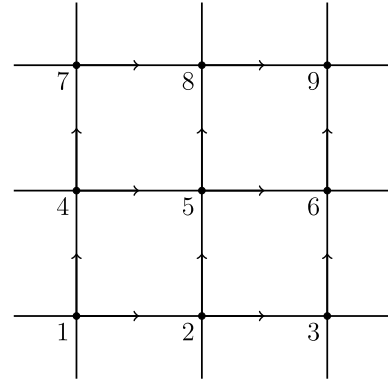


FIG. 10. A qudit is placed on each edge, with generalized Pauli operators X_e and Z_e acting on it.

e_{12} to edge e_{78} or to edge e_{58} involves multiplying the column vector of the operator by y^2 or xy , respectively:

$$\mathcal{Z}_{78} = y^2 \mathcal{Z}_{12} = \begin{bmatrix} 0 \\ 0 \\ y^2 \\ 0 \end{bmatrix}, \quad \mathcal{X}_{58} = xy \mathcal{X}_{14} = \begin{bmatrix} 0 \\ xy \\ 0 \\ 0 \end{bmatrix}. \quad (\text{A6})$$

A general Pauli operator can be expressed as

$$P = \eta X_{e_1}^{a_1} X_{e_2}^{a_2} \cdots X_{e_n}^{a_n} Z_{e'_1}^{b_1} Z_{e'_2}^{b_2} \cdots Z_{e'_m}^{b_m}, \quad (\text{A7})$$

where η represents a root of unity of order $2d$. After dropping the overall phase η , the corresponding column vector for this operator is a linear combination of individual Pauli matrices, expressed as

$$\mathcal{P} = a_1 \mathcal{X}_{e_1} + a_2 \mathcal{X}_{e_2} + \cdots + a_n \mathcal{X}_{e_n} + b_1 \mathcal{Z}_{e'_1} + b_2 \mathcal{Z}_{e'_2} + \cdots + b_m \mathcal{Z}_{e'_m}. \quad (\text{A8})$$

More examples are included in Fig. 11.

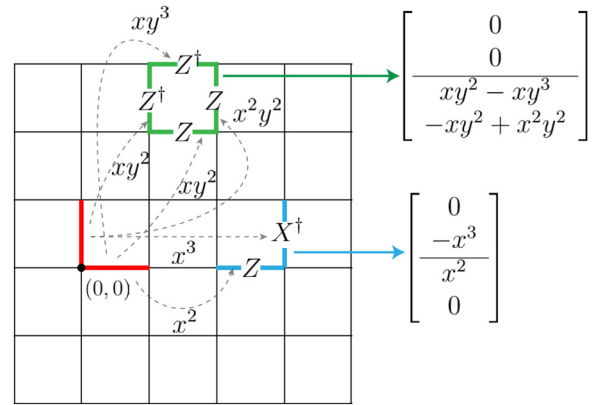


FIG. 11. Examples of polynomial expressions for Pauli strings. The flux term on a plaquette and the XZ term on edges are shown. The factors such as $x^2 y^2$ and x^2 represent the locations of the operators relative to the origin.

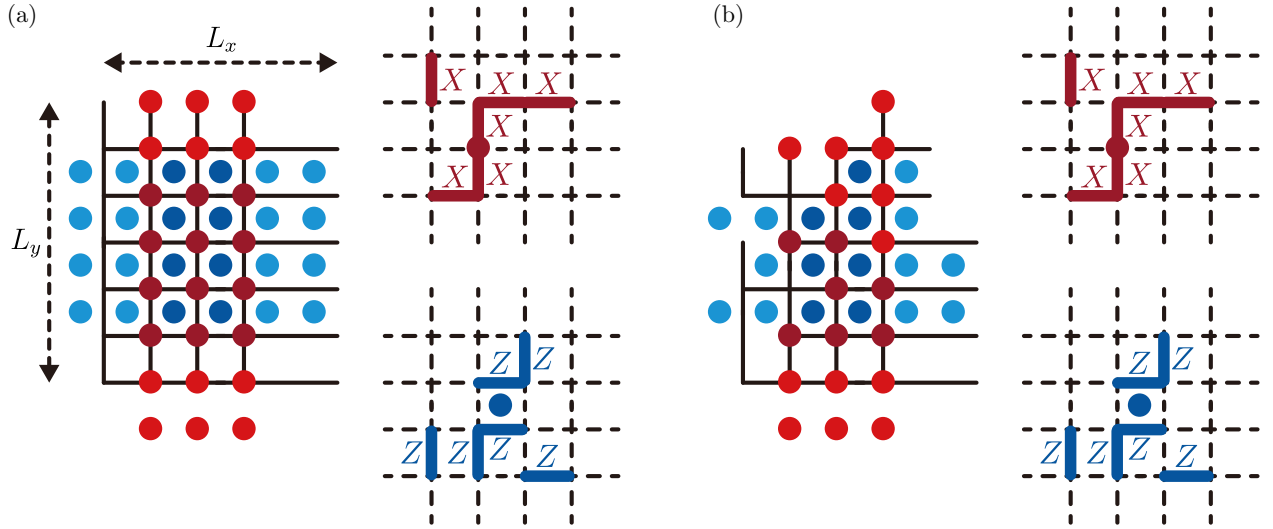


FIG. 12. (a) The $[[54, 6, 4]]$ code on an open square lattice with $L_x = 5$ and $L_y = 6$. The bulk stabilizers correspond to $f(x, y) = xy^2(1 + x + x^{-1}y^{-2})$ and $g(x, y) = x(1 + y + x^{-1}y^2)$. Each red (blue) dot represents an X - (Z -) stabilizer: deep red and blue indicate bulk stabilizers fully supported on the pruned lattice, while lighter dots near the boundary denote stabilizers with truncated edges. There are 12 bulk X (deep red), eight bulk Z (deep blue), 12 boundary X (red), and 16 boundary Z (blue) stabilizers, giving a logical dimension $k = 54 - 12 - 8 - 12 - 16 = 6$. (b) The grafted $[[44, 6, 4]]$ code, obtained by removing ten qubits. The remaining stabilizers include seven bulk X (deep red), seven bulk Z (deep blue), 13 boundary X (red), and 11 boundary Z (blue), yielding the same logical dimension $k = 44 - 7 - 7 - 13 - 11 = 6$.

Next, we introduce the *antipode map* that is a $\mathbb{Z}_d$ -linear map from R to R defined by

$$x^a y^b \rightarrow \overline{x^a y^b} := x^{-a} y^{-b}. \quad (\text{A9})$$

To determine whether two Pauli operators represented by vectors v_1 and v_2 commute or anticommute, we define the

dot product as

$$v_1 \cdot v_2 = \overline{v_1}^T \Lambda v_2, \quad (\text{A10})$$

where T is the transpose operation on a matrix and

$$\Lambda = \left[\begin{array}{cc|cc} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hline -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{array} \right] \quad (\text{A11})$$

is the matrix representation of the standard *symplectic bilinear form*. For simplicity, we denote $\overline{(\dots)}^T$ as $(\dots)^\dagger$.

Two operators commute if and only if the constant term of $v_1 \cdot v_2$ is zero. For example,

$$\begin{aligned} \mathcal{X}_{12} \cdot \mathcal{Z}_{12} &= 1 \quad (\text{anticommute by the factor of } \omega), \\ \mathcal{X}_{58} \cdot \mathcal{Z}_{14} &= x^{-1}y^{-1} \quad (\text{commute}). \end{aligned} \quad (\text{A12})$$

Furthermore, the physical interpretation of $\mathcal{X}_{58} \cdot \mathcal{Z}_{14} = x^{-1}y^{-1}$ is that shifting of $\mathcal{X}_{58}$ in $-x$ and $-y$ directions by 1 step will anticommute (by the factor of ω) with $\mathcal{Z}_{14}$.

A translation-invariant stabilizer code corresponds to an R -submodule σ such that

$$v_1 \cdot v_2 = v_1^\dagger \Lambda v_2 = 0, \quad \forall v_1, v_2 \in \sigma, \quad (\text{A13})$$

called the *stabilizer module*. The Hamiltonian could have two (or more) terms per square to have a unique ground

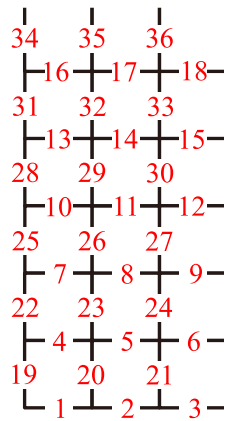


FIG. 13. Sequential qubit numbering from 1 to 36 on a 3×6 lattice. Indices are first assigned to qubits on horizontal edges, followed by those on vertical edges, proceeding from the bottom left to the top right. After lattice grafting, the indexing is obtained by removing the corresponding qubits while preserving the same ordering convention.

TABLE II. (n, d) for the $k = 6$ family from the $[[88, 6, 6]]$ code (Fig. 8) for different L_x and L_y .

L_x	L_y											
	6	7	8	9	10	11	12	13	14	15	16	17
5	(54,4)	(63,4)	(72,4)	(81,4)	(90,4)	(99,4)	(108,4)	(117,4)	(126,4)	(135,4)	(144,4)	(153,4)
6	(66,4)	(77,5)	(88,6)	(99,6)	(110,6)	(121,6)	(132,6)	(143,6)	(154,6)	(165,6)	(176,6)	(187,6)
7	(78,4)	(91,5)	(104,6)	(117,7)	(130,7)	(143,7)	(156,7)	(169,7)	(182,7)	(195,7)	(208,7)	(221,7)
8	(90,4)	(105,5)	(120,6)	(135,7)	(150,8)	(165,9)	(180,9)	(195,9)	(210,9)	(225,9)	(240,9)	(255,9)
9	(102,4)	(119,5)	(136,6)	(153,7)	(170,8)	(187,9)	(204,10)	(221,10)	(238,10)	(255,10)	(272,10)	(289,10)
10	(114,4)	(133,5)	(152,6)	(171,7)	(190,8)	(209,9)	(228,11)	(247,12)	(266,12)	(285,12)	(304,12)	(323,12)
11	(126,4)	(147,5)	(168,6)	(189,7)	(210,8)	(231,9)	(252,11)	(273,12)	(294,13)	(315,13)	(336,13)	(357,13)
12	(138,4)	(161,5)	(184,6)	(207,7)	(230,8)	(253,9)	(276,11)	(299,12)	(322,13)	(345,14)	(368,14)	(391,14)
13	(150,4)	(175,5)	(200,6)	(225,7)	(250,8)	(275,9)	(300,11)	(325,12)	(350,13)	(375,14)	(400,15)	(425,16)
14	(162,4)	(189,5)	(216,6)	(243,7)	(270,8)	(297,9)	(324,11)	(351,12)	(378,13)	(405,14)	(432,15)	(459,16)

TABLE III. (n, d) for the $k = 7$ family from the $[[131, 7, 7]]$ code (Fig. 7) for different L_x and L_y .

L_x	L_y												
	7	8	9	10	11	12	13	14	15	16	17	18	19
5	(55,4)	(64,4)	(73,4)	(82,4)	(91,4)	(100,4)	(109,4)	(118,4)	(127,4)	(136,4)	(145,4)	(154,4)	(163,4)
6	(67,4)	(78,4)	(89,5)	(100,5)	(111,5)	(122,5)	(133,5)	(144,5)	(155,5)	(166,5)	(177,5)	(188,5)	(199,5)
7	(79,4)	(92,4)	(105,5)	(118,6)	(131,7)	(144,7)	(157,7)	(170,7)	(183,7)	(196,7)	(209,7)	(222,7)	(235,7)
8	(91,4)	(106,4)	(121,5)	(136,6)	(151,7)	(166,8)	(181,8)	(196,8)	(211,8)	(226,8)	(241,8)	(256,8)	(271,8)
9	(103,4)	(120,4)	(137,5)	(154,6)	(171,7)	(188,8)	(205,10)	(222,10)	(239,10)	(256,10)	(273,10)	(290,10)	(307,10)
10	(115,4)	(134,4)	(153,5)	(172,6)	(191,7)	(210,8)	(229,10)	(248,11)	(267,12)	(286,12)	(305,12)	(324,12)	(343,12)
11	(127,4)	(148,4)	(169,5)	(190,6)	(211,7)	(232,8)	(253,10)	(274,11)	(295,12)	(316,13)	(337,14)	(358,14)	(379,14)
12	(139,4)	(162,4)	(185,5)	(208,6)	(231,7)	(254,8)	(277,10)	(300,11)	(323,12)	(346,13)	(369,14)	(392,15)	(415,15)
13	(151,4)	(176,4)	(201,5)	(226,6)	(251,7)	(276,8)	(301,10)	(326,11)	(351,12)	(376,13)	(401,14)	(426,15)	(451,16)

TABLE IV. (n, d) for the $k = 8$ family from the $[[188, 8, 9]]$ code (Fig. 6) for different L_x and L_y .

L_x	L_y												
	7	8	9	10	11	12	13	14	15	16	17	18	19
4	(46,3)	(53,3)	(60,3)	(67,3)	(74,3)	(81,3)	(88,3)	(95,3)	(102,3)	(109,3)	(116,3)	(123,3)	(130,3)
5	(59,4)	(68,4)	(77,4)	(86,4)	(95,4)	(104,4)	(113,4)	(122,4)	(131,4)	(140,4)	(149,4)	(158,4)	(167,4)
6	(72,4)	(83,4)	(94,5)	(105,6)	(116,6)	(127,6)	(138,6)	(149,6)	(160,6)	(171,6)	(182,6)	(193,6)	(204,6)
7	(85,4)	(98,4)	(111,5)	(124,6)	(137,7)	(150,7)	(163,7)	(176,7)	(189,7)	(202,7)	(215,7)	(228,7)	(241,7)
8	(98,4)	(113,4)	(128,5)	(143,6)	(158,7)	(173,8)	(188,9)	(203,9)	(218,9)	(233,9)	(248,9)	(263,9)	(278,9)
9	(111,4)	(128,4)	(145,5)	(162,6)	(179,7)	(196,8)	(213,9)	(230,10)	(247,10)	(264,10)	(281,10)	(298,10)	(315,10)
10	(124,4)	(143,4)	(162,5)	(181,6)	(200,7)	(219,8)	(238,9)	(257,10)	(276,11)	(295,12)	(314,12)	(333,12)	(352,12)
11	(137,4)	(158,4)	(179,5)	(200,6)	(221,7)	(242,8)	(263,9)	(284,10)	(305,11)	(326,13)	(347,14)	(368,15)	(389,15)
12	(150,4)	(173,4)	(196,5)	(219,6)	(242,7)	(265,8)	(288,9)	(311,10)	(334,11)	(357,13)	(380,14)	(403,15)	(426,16)

TABLE V. (n, d) for the $k = 8$ family from the $[[288, 8, 12]]$ code (Fig. 1) for different L_x and L_y .

	L_y										
L_x	5	6	7	8	9	10	11	12	13	14	15
5	(50,3)	(60,3)	(70,3)	(80,3)	(90,3)	(100,3)	(110,3)	(120,3)	(130,3)	(140,3)	(150,3)
6	(60,3)	(72,4)	(84,4)	(96,4)	(108,4)	(120,4)	(132,4)	(144,4)	(156,4)	(168,4)	(180,4)
7	(70,3)	(84,4)	(98,5)	(112,5)	(126,5)	(140,5)	(154,5)	(168,5)	(182,5)	(196,5)	(210,5)
8	(80,3)	(96,4)	(112,5)	(128,6)	(144,6)	(160,6)	(176,6)	(192,6)	(208,6)	(224,6)	(240,6)
9	(90,3)	(108,4)	(126,5)	(144,6)	(162,7)	(180,7)	(198,7)	(216,7)	(234,7)	(252,7)	(270,7)
10	(100,3)	(120,4)	(140,5)	(160,6)	(180,7)	(200,9)	(220,9)	(240,9)	(260,9)	(280,9)	(300,9)
11	(110,3)	(132,4)	(154,5)	(176,6)	(198,7)	(220,9)	(242,10)	(264,10)	(286,10)	(308,10)	(330,10)
12	(120,3)	(144,4)	(168,5)	(192,6)	(216,7)	(240,9)	(264,10)	(288,12)	(312,12)	(336,12)	(360,12)
13	(130,3)	(156,4)	(182,5)	(208,6)	(234,7)	(260,9)	(286,10)	(312,12)	(338,13)	(364,13)	(390,13)
14	(140,3)	(168,4)	(196,5)	(224,6)	(252,7)	(280,9)	(308,10)	(336,12)	(364,13)	(392,15)	(420,15)
15	(150,3)	(180,4)	(210,5)	(240,6)	(270,7)	(300,9)	(330,10)	(360,12)	(390,13)	(420,15)	(450,16)

TABLE VI. (n, d) for the $k = 12$ family (weight-8 stabilizers) from the $[[292, 12, 14]]$ code (Fig. 9) for different L_x and L_y .

L_x	L_y										
	14	15	16	17	18	19	20	21	22	23	24
4	(94,5)	(101,5)	(108,5)	(115,5)	(122,5)	(129,5)	(136,5)	(143,5)	(150,5)	(157,5)	(164,5)
5	(121,6)	(130,6)	(139,6)	(148,6)	(157,6)	(166,6)	(175,6)	(184,6)	(193,6)	(202,6)	(211,6)
6	(148,6)	(159,8)	(170,9)	(181,10)	(192,10)	(203,10)	(214,10)	(225,10)	(236,10)	(247,10)	(258,10)
7	(175,6)	(188,7)	(201,9)	(214,9)	(227,11)	(240,12)	(253,12)	(266,12)	(279,12)	(292,12)	(305,12)
8	(202,6)	(217,7)	(232,9)	(247,9)	(262,11)	(277,12)	(292,14)	(307,14)	(322,14)	(337,14)	(352,14)
9	(229,6)	(246,7)	(263,9)	(280,9)	(297,11)	(314,12)	(331,14)	(348,15)	(365,16)	(382,17)	(399,18)
10	(256,6)	(275,7)	(294,9)	(313,9)	(332,11)	(351,12)	(370,14)	(389,15)	(408,16)	(427,17)	(446,18)
11	(283,6)	(304,7)	(325,9)	(346,9)	(367,11)	(388,12)	(409,14)	(430,15)	(451,16)	(472,17)	(493,18)
12	(310,6)	(333,7)	(356,9)	(379,9)	(402,11)	(425,12)	(448,14)	(471,15)	(494,16)	(517,17)	(540,18)

state on a simply connected manifold, denoted as $H = -\sum_{\text{cells}} (S_1 + S_2)$ with corresponding column vectors

$$S_1 = \begin{bmatrix} f_1(x, y) \\ f_2(x, y) \\ g_1(x, y) \\ g_2(x, y) \end{bmatrix}, \quad S_2 = \begin{bmatrix} h_1(x, y) \\ h_2(x, y) \\ k_1(x, y) \\ k_2(x, y) \end{bmatrix}. \quad (\text{A14})$$

For example, the trivial phase $H_0 = -\sum_e X_e$ is

$$S_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad S_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad (\text{A15})$$

and the standard $\mathbb{Z}_d$ toric code Hamiltonian

$$H_{\text{TC}} = -\sum_v \begin{array}{c} | \\ X \\ | \\ X^\dagger \\ | \end{array} - \sum_p \begin{array}{c} Z^\dagger \\ \left[\begin{array}{c} Z^\dagger \quad p \quad Z \\ Z \end{array} \right] \end{array} \quad (\text{A16})$$

corresponds to

$$S_1 = \begin{bmatrix} 1 - \bar{x} \\ 1 - \bar{y} \\ 0 \\ 0 \end{bmatrix}, \quad S_2 = \begin{bmatrix} 0 \\ 0 \\ 1 - y \\ -1 + x \end{bmatrix}. \quad (\text{A17})$$

In the main text, we have considered a translation-invariant $\mathbb{Z}_2$ CSS code defined on the square lattice, with bulk stabilizer generators:

$$A_v = \begin{bmatrix} f(x, y) \\ g(x, y) \\ 0 \\ 0 \end{bmatrix}, \quad B_p = \begin{bmatrix} 0 \\ 0 \\ \overline{g(x, y)} \\ \overline{f(x, y)} \end{bmatrix}, \quad (\text{A18})$$

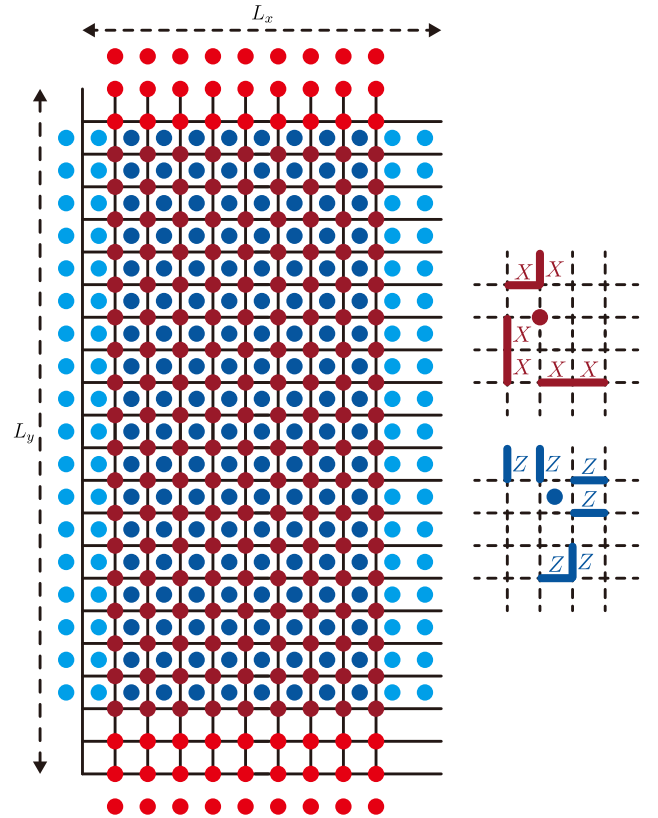


FIG. 14. The $[[441, 9, 15]]$ code on an open square lattice. Bulk stabilizers are specified by two polynomials $f(x, y) = x(1 + x + x^{-1}y^3)$ and $g(x, y) = 1 + y + xy^3$ [45]. The vertical edges of the rightmost column have been removed. Each red (blue) dot represents an X - (Z -) stabilizer. Deep red and blue dots correspond to bulk stabilizers fully supported on the lattice. In contrast, dots near the boundary represent stabilizers with truncated edges (i.e., the bulk stabilizer pattern positioned at the boundary dot, retaining only the Pauli operators on edges within the lattice). The design ensures that all truncated X and Z operators still commute, forming valid boundary stabilizers. In total, there are 162 bulk X -stabilizers (deep red), 144 bulk Z -stabilizers (deep blue), 54 boundary X -stabilizers (red), and 72 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 441 - 162 - 144 - 54 - 72 = 9$. This generates a code family of $k = 9$: $[[2L_x L_y - L_y, 9, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table VII below.

TABLE VII. (n, d) for the $k = 9$ family from the $[[441, 9, 15]]$ code for different L_x and L_y .

L_x	L_y													
	9	10	11	12	13	14	15	16	17	18	19	20	21	22
5	(81,4)	(90,4)	(99,4)	(108,4)	(117,4)	(126,4)	(135,4)	(144,4)	(153,4)	(162,4)	(171,4)	(180,4)	(189,4)	(198,4)
6	(99,4)	(110,5)	(121,5)	(132,6)	(143,6)	(154,6)	(165,6)	(176,6)	(187,6)	(198,6)	(209,6)	(220,6)	(231,6)	(242,6)
7	(117,4)	(130,5)	(143,5)	(156,6)	(169,7)	(182,7)	(195,7)	(208,7)	(221,7)	(234,7)	(247,7)	(260,7)	(273,7)	(286,7)
8	(135,4)	(150,5)	(165,5)	(180,6)	(195,7)	(210,8)	(225,9)	(240,9)	(255,9)	(270,9)	(285,9)	(300,9)	(315,9)	(330,9)
9	(153,4)	(170,5)	(187,5)	(204,6)	(221,7)	(238,8)	(255,9)	(272,10)	(289,10)	(306,10)	(323,10)	(340,10)	(357,10)	(374,10)
10	(171,4)	(190,5)	(209,5)	(228,6)	(247,7)	(266,8)	(285,9)	(304,10)	(323,10)	(342,11)	(361,12)	(380,12)	(399,12)	(418,12)
11	(189,4)	(210,5)	(231,5)	(252,6)	(273,7)	(294,8)	(315,9)	(336,10)	(357,10)	(378,11)	(399,12)	(420,13)	(441,15)	(462,15)
12	(207,4)	(230,5)	(253,5)	(276,6)	(299,7)	(322,8)	(345,9)	(368,10)	(391,10)	(414,11)	(437,12)	(460,13)	(483,15)	(506,16)

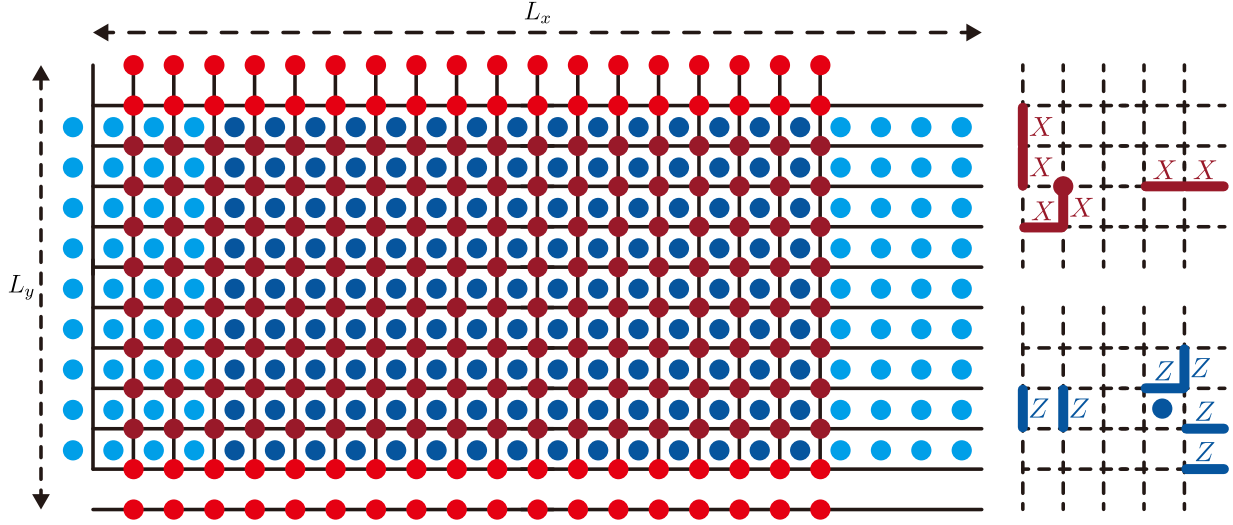


FIG. 15. The $[[432, 9, 14]]$ code on an open square lattice. Bulk stabilizers are specified by two polynomials $f(x, y) = x^3y(1 + x + x^{-3}y^{-1})$ and $g(x, y) = y(1 + y + xy^{-1})$ [45]. The vertical edges of the rightmost column have been removed. In total, there are 144 bulk X -stabilizers (deep red), 135 bulk Z -stabilizers (deep blue), 72 boundary X -stabilizers (red), and 72 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 432 - 144 - 135 - 72 \times 2 = 9$. This generates a code family of $k = 9$: $[[2L_xL_y - L_x - 3L_y + 3, 9, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table VIII below.

TABLE VIII. (n, d) for the $k = 9$ family from the $[[432, 9, 14]]$ code for different L_x and L_y .

L_x	L_y									
	5	6	7	8	9	10	11	12	13	14
10	(78,4)	(95,4)	(112,4)	(129,4)	(146,4)	(163,4)	(180,4)	(197,4)	(214,4)	(231,4)
11	(87,4)	(106,4)	(125,4)	(144,4)	(163,4)	(182,4)	(201,4)	(220,4)	(239,4)	(258,4)
12	(96,4)	(117,5)	(138,5)	(159,5)	(180,5)	(201,5)	(222,5)	(243,5)	(264,5)	(285,5)
13	(105,4)	(128,5)	(151,6)	(174,6)	(197,6)	(220,6)	(243,6)	(266,6)	(289,6)	(312,6)
14	(114,4)	(139,5)	(164,7)	(189,7)	(214,7)	(239,7)	(264,7)	(289,7)	(314,7)	(339,7)
15	(123,4)	(150,5)	(177,7)	(204,8)	(231,8)	(258,8)	(285,8)	(312,8)	(339,8)	(366,8)
16	(132,4)	(161,5)	(190,7)	(219,8)	(248,9)	(277,9)	(306,9)	(335,9)	(364,9)	(393,9)
17	(141,4)	(172,5)	(203,7)	(234,8)	(265,10)	(296,10)	(327,10)	(358,10)	(389,10)	(420,10)
18	(150,4)	(183,5)	(216,7)	(249,8)	(282,10)	(315,10)	(348,10)	(381,10)	(414,10)	(447,10)
19	(159,4)	(194,5)	(229,7)	(264,8)	(299,10)	(334,11)	(369,11)	(404,11)	(439,11)	(474,11)
20	(168,4)	(205,5)	(242,7)	(279,8)	(316,10)	(353,12)	(390,12)	(427,12)	(464,12)	(501,12)
21	(177,4)	(216,5)	(255,7)	(294,8)	(333,10)	(372,12)	(411,13)	(450,13)	(489,13)	(528,13)
22	(186,4)	(227,5)	(268,7)	(309,8)	(350,10)	(391,12)	(432,14)	(473,14)	(514,14)	(555,14)
23	(195,4)	(238,5)	(281,7)	(324,8)	(367,10)	(410,12)	(453,14)	(496,15)	(539,15)	(582,15)
24	(204,4)	(249,5)	(294,7)	(339,8)	(384,10)	(429,12)	(474,14)	(519,15)	(564,15)	(609,15)
25	(213,4)	(260,5)	(307,7)	(354,8)	(401,10)	(448,12)	(495,14)	(542,16)	(589,16)	(636,16)

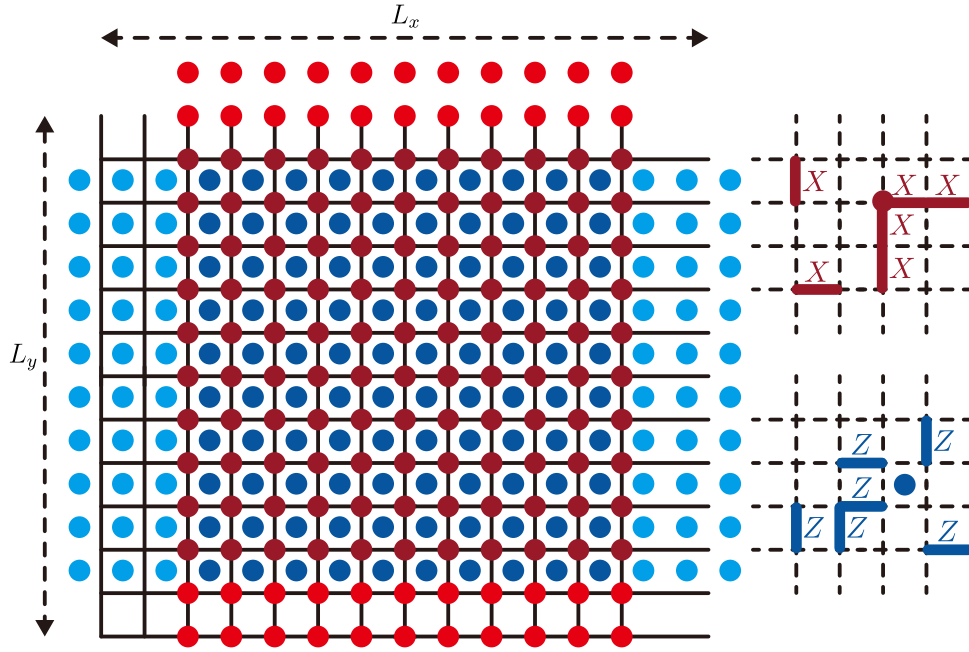


FIG. 16. The $[[324, 10, 11]]$ code on an open square lattice. Bulk stabilizers are specified by two polynomials $f(x, y) = x^2y^2(1 + x + x^{-2}y^{-2})$ and $g(x, y) = x^2(1 + y + x^{-2}y^2)$ [45]. The vertical edges of the rightmost column have been removed. In total, there are 110 bulk X -stabilizers (deep red), 100 bulk Z -stabilizers (deep blue), 44 boundary X -stabilizers (red), and 60 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 324 - 110 - 100 - 44 - 60 = 10$. This generates a code family of $k = 10$: $[[2L_xL_y - L_y, 10, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table IX below.

with

$$\begin{aligned} f(x, y) &= 1 + x + x^a y^b, \\ g(x, y) &= 1 + y + x^c y^d. \end{aligned} \quad (\text{A19})$$

We now introduce the *excitation map* for a Pauli operator $\mathcal{P}$ (represented as a column vector over R):

$$\epsilon(\mathcal{P}) := [\mathcal{S}_1 \cdot \mathcal{P}, \mathcal{S}_2 \cdot \mathcal{P}], \quad (\text{A20})$$

which encodes how $\mathcal{P}$ violates the stabilizers $\mathcal{S}_1$ and $\mathcal{S}_2$. For a single Pauli operator in the generic Hamiltonian (A14), the corresponding error syndromes are

$$\begin{aligned} \epsilon(\mathcal{X}_1) &= [\mathcal{S}_1 \cdot \mathcal{X}_{12}, \mathcal{S}_2 \cdot \mathcal{X}_{12}] = [-\bar{g}_1, -\bar{k}_1], \\ \epsilon(\mathcal{X}_2) &= [\mathcal{S}_1 \cdot \mathcal{X}_{14}, \mathcal{S}_2 \cdot \mathcal{X}_{14}] = [-\bar{g}_2, -\bar{k}_2], \\ \epsilon(\mathcal{Z}_1) &= [\mathcal{S}_1 \cdot \mathcal{Z}_{12}, \mathcal{S}_2 \cdot \mathcal{Z}_{12}] = [\bar{f}_1, \bar{h}_1], \\ \epsilon(\mathcal{Z}_2) &= [\mathcal{S}_1 \cdot \mathcal{Z}_{14}, \mathcal{S}_2 \cdot \mathcal{Z}_{14}] = [\bar{f}_2, \bar{h}_2]. \end{aligned} \quad (\text{A21})$$

TABLE IX. (n, d) for the $k = 10$ family from the $[[324, 10, 11]]$ code for different L_x and L_y .

L_x	L_y									
	6	7	8	9	10	11	12	13	14	15
7	(78,4)	(91,4)	(104,4)	(117,4)	(130,4)	(143,4)	(156,4)	(169,4)	(182,4)	(195,4)
8	(90,4)	(105,4)	(120,4)	(135,4)	(150,4)	(165,4)	(180,4)	(195,4)	(210,4)	(225,4)
9	(102,4)	(119,5)	(136,5)	(153,5)	(170,5)	(187,5)	(204,5)	(221,5)	(238,5)	(255,5)
10	(114,4)	(133,5)	(152,6)	(171,6)	(190,6)	(209,6)	(228,6)	(247,6)	(266,6)	(285,6)
11	(126,4)	(147,5)	(168,6)	(189,7)	(210,7)	(231,7)	(252,7)	(273,7)	(294,7)	(315,7)
12	(138,4)	(161,5)	(184,6)	(207,7)	(230,9)	(253,9)	(276,9)	(299,9)	(322,9)	(345,9)
13	(150,4)	(175,5)	(200,6)	(225,7)	(250,9)	(275,10)	(300,10)	(325,10)	(350,10)	(375,10)
14	(162,4)	(189,5)	(216,6)	(243,7)	(270,9)	(297,10)	(324,11)	(351,11)	(378,11)	(405,11)
15	(174,4)	(203,5)	(232,6)	(261,7)	(290,9)	(319,10)	(348,11)	(377,11)	(406,11)	(435,11)
16	(186,4)	(217,5)	(248,6)	(279,7)	(310,9)	(341,10)	(372,12)	(403,13)	(434,13)	(465,13)
17	(198,4)	(231,5)	(264,6)	(297,7)	(330,9)	(363,10)	(396,12)	(429,13)	(462,14)	(495,14)
18	(210,4)	(245,5)	(280,6)	(315,7)	(350,9)	(385,10)	(420,12)	(455,13)	(490,15)	(525,15)
19	(222,4)	(259,5)	(296,6)	(333,7)	(370,9)	(407,10)	(444,12)	(481,13)	(518,15)	(555,16)

TABLE X. (n, d) for the $k = 10$ family from the $[[381, 10, 13]]$ code for different L_x and L_y .

L_x	L_y															
	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
5	(82,4)	(91,4)	(100,4)	(109,4)	(118,4)	(127,4)	(136,4)	(145,4)	(154,4)	(163,4)	(172,4)	(181,4)	(190,4)	(199,4)	(208,4)	
6	(100,4)	(111,4)	(122,5)	(133,6)	(144,6)	(155,6)	(166,6)	(177,6)	(188,6)	(199,6)	(210,6)	(221,6)	(232,6)	(243,6)	(254,6)	
7	(118,4)	(131,4)	(144,5)	(157,6)	(170,7)	(183,7)	(196,7)	(209,7)	(222,7)	(235,7)	(248,7)	(261,7)	(274,7)	(287,7)	(300,7)	
8	(136,4)	(151,4)	(166,5)	(181,6)	(196,7)	(211,8)	(226,9)	(241,9)	(256,9)	(271,9)	(286,9)	(301,9)	(316,9)	(331,9)	(346,9)	
9	(154,4)	(171,4)	(188,5)	(205,6)	(222,7)	(239,8)	(256,9)	(273,10)	(290,10)	(307,10)	(324,10)	(341,10)	(358,10)	(375,10)	(392,10)	
10	(172,4)	(191,4)	(210,5)	(229,6)	(248,7)	(267,8)	(286,9)	(305,10)	(324,10)	(343,11)	(362,12)	(381,13)	(400,13)	(419,13)	(438,13)	
11	(190,4)	(211,4)	(232,5)	(253,6)	(274,7)	(295,8)	(316,9)	(337,10)	(358,10)	(379,11)	(400,12)	(421,13)	(442,14)	(463,14)	(484,14)	
12	(208,4)	(231,4)	(254,5)	(277,6)	(300,7)	(323,8)	(346,9)	(369,10)	(392,10)	(415,11)	(438,12)	(461,13)	(484,14)	(507,15)	(530,16)	
13	(226,4)	(251,4)	(276,5)	(301,6)	(326,7)	(351,8)	(376,9)	(401,10)	(426,10)	(451,11)	(476,12)	(501,13)	(526,14)	(551,15)	(576,16)	
14	(244,4)	(271,4)	(298,5)	(325,6)	(352,7)	(379,8)	(406,9)	(433,10)	(460,10)	(487,11)	(514,12)	(541,13)	(568,14)	(595,15)	(622,16)	
15	(262,4)	(291,4)	(320,5)	(349,6)	(378,7)	(407,8)	(436,9)	(465,10)	(494,10)	(523,11)	(552,12)	(581,13)	(610,14)	(639,15)	(668,16)	

TABLE XI. (n, d) for the $k = 11$ family from the $[[494, 11, 15]]$ code for different L_x and L_y .

L_x	L_y															
	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
5	(86,4)	(95,4)	(104,4)	(113,4)	(122,4)	(131,4)	(140,4)	(149,4)	(158,4)	(167,4)	(176,4)	(185,4)	(194,4)	(203,4)	(212,4)	(221,4)
6	(105,4)	(116,4)	(127,5)	(138,6)	(149,6)	(160,6)	(171,6)	(182,6)	(193,6)	(204,6)	(215,6)	(226,6)	(237,6)	(248,6)	(259,6)	(270,6)
7	(124,4)	(137,4)	(150,5)	(163,6)	(176,7)	(189,7)	(202,7)	(215,7)	(228,7)	(241,7)	(254,7)	(267,7)	(280,7)	(293,7)	(306,7)	(319,7)
8	(143,4)	(158,4)	(173,5)	(188,6)	(203,7)	(218,7)	(233,8)	(248,9)	(263,9)	(278,9)	(293,9)	(308,9)	(323,9)	(338,9)	(353,9)	(368,9)
9	(162,4)	(179,4)	(196,5)	(213,6)	(230,7)	(247,7)	(264,8)	(281,9)	(298,10)	(315,10)	(332,10)	(349,10)	(366,10)	(383,10)	(400,10)	(417,10)
10	(181,4)	(200,4)	(219,5)	(238,6)	(257,7)	(276,7)	(295,8)	(314,9)	(333,10)	(352,11)	(371,12)	(390,12)	(409,12)	(428,12)	(447,12)	(466,12)
11	(200,4)	(221,4)	(242,5)	(263,6)	(284,7)	(305,7)	(326,8)	(347,9)	(368,10)	(389,11)	(410,12)	(431,13)	(452,13)	(473,14)	(494,15)	(515,15)
12	(219,4)	(242,4)	(265,5)	(288,6)	(311,7)	(334,7)	(357,8)	(380,9)	(403,10)	(426,11)	(449,12)	(472,13)	(495,13)	(518,14)	(541,15)	(564,16)
13	(238,4)	(263,4)	(288,5)	(313,6)	(338,7)	(363,7)	(388,8)	(413,9)	(438,10)	(463,11)	(488,12)	(513,13)	(538,13)	(563,14)	(588,15)	(613,16)
14	(257,4)	(284,4)	(311,5)	(338,6)	(365,7)	(392,7)	(419,8)	(446,9)	(473,10)	(500,11)	(527,12)	(554,13)	(581,13)	(608,14)	(635,15)	(662,16)

L_x	9	10	11	12	13	14	15	16	17	18	19	20	21	22
6	(108,4)	(120,4)	(132,4)	(144,4)	(156,4)	(168,4)	(180,4)	(192,4)	(204,4)	(216,4)	(228,4)	(240,4)	(252,4)	(264,4)
7	(126,4)	(140,5)	(154,5)	(168,5)	(182,5)	(196,5)	(210,5)	(224,5)	(238,5)	(252,5)	(266,5)	(280,5)	(294,5)	(308,5)
8	(144,4)	(160,5)	(176,5)	(192,6)	(208,6)	(224,6)	(240,6)	(256,6)	(272,6)	(288,6)	(304,6)	(320,6)	(336,6)	(352,6)
9	(162,4)	(180,5)	(198,5)	(216,6)	(234,7)	(252,7)	(270,7)	(288,7)	(306,7)	(324,7)	(342,7)	(360,7)	(378,7)	(396,7)
10	(180,4)	(200,5)	(220,5)	(240,6)	(260,7)	(280,8)	(300,9)	(320,9)	(340,9)	(360,9)	(380,9)	(400,9)	(420,9)	(440,9)
11	(198,4)	(220,5)	(242,5)	(264,6)	(286,7)	(308,8)	(330,10)	(352,10)	(374,10)	(396,10)	(418,10)	(440,10)	(462,10)	(484,10)
12	(216,4)	(240,5)	(264,5)	(288,6)	(312,7)	(336,8)	(360,10)	(384,11)	(408,11)	(432,12)	(456,12)	(480,12)	(504,12)	(528,12)
13	(234,4)	(260,5)	(286,5)	(312,6)	(338,7)	(364,8)	(390,10)	(416,11)	(442,11)	(468,12)	(494,13)	(520,13)	(546,13)	(572,13)
14	(252,4)	(280,5)	(308,5)	(336,6)	(364,7)	(392,8)	(420,10)	(448,11)	(476,11)	(504,12)	(532,13)	(560,14)	(588,15)	(616,15)
15	(270,4)	(300,5)	(330,5)	(360,6)	(390,7)	(420,8)	(450,10)	(480,11)	(510,11)	(540,12)	(570,13)	(600,14)	(630,15)	(660,16)

L_x	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
5	(105,4)	(114,4)	(123,4)	(132,4)	(141,4)	(150,4)	(159,4)	(168,4)	(177,4)	(186,4)	(195,4)	(204,4)	(213,4)	(222,4)	(231,4)	(240,4)	(249,4)	(258,4)	(267,4)
6	(128,4)	(139,4)	(150,5)	(161,5)	(172,6)	(183,6)	(194,6)	(205,6)	(216,6)	(227,6)	(238,6)	(249,6)	(260,6)	(271,6)	(282,6)	(293,6)	(304,6)	(315,6)	(326,6)
7	(151,4)	(164,4)	(177,5)	(190,5)	(203,6)	(216,7)	(229,7)	(242,7)	(255,7)	(268,7)	(281,7)	(294,7)	(307,7)	(320,7)	(333,7)	(346,7)	(359,7)	(372,7)	(385,7)
8	(174,4)	(189,4)	(204,5)	(219,5)	(234,6)	(249,7)	(264,8)	(279,9)	(294,9)	(309,9)	(324,9)	(339,9)	(354,9)	(369,9)	(384,9)	(399,9)	(414,9)	(429,9)	(444,9)
9	(197,4)	(214,4)	(231,5)	(248,5)	(265,6)	(282,7)	(299,8)	(316,9)	(333,10)	(350,10)	(367,10)	(384,10)	(401,10)	(418,10)	(435,10)	(452,10)	(469,10)	(486,10)	(503,10)
10	(220,4)	(239,4)	(258,5)	(277,5)	(296,6)	(315,7)	(334,8)	(353,9)	(372,10)	(391,10)	(410,10)	(429,10)	(448,11)	(467,12)	(486,13)	(505,13)	(524,13)	(543,13)	(562,13)
11	(243,4)	(264,4)	(285,5)	(306,5)	(327,6)	(348,7)	(369,8)	(390,9)	(411,10)	(432,10)	(453,10)	(474,10)	(495,11)	(516,12)	(537,13)	(558,13)	(579,14)	(600,14)	(621,14)
12	(266,4)	(289,4)	(312,5)	(335,5)	(358,6)	(381,7)	(404,8)	(427,9)	(450,10)	(473,10)	(496,10)	(519,10)	(542,11)	(565,12)	(588,13)	(611,13)	(634,14)	(657,15)	(680,16)

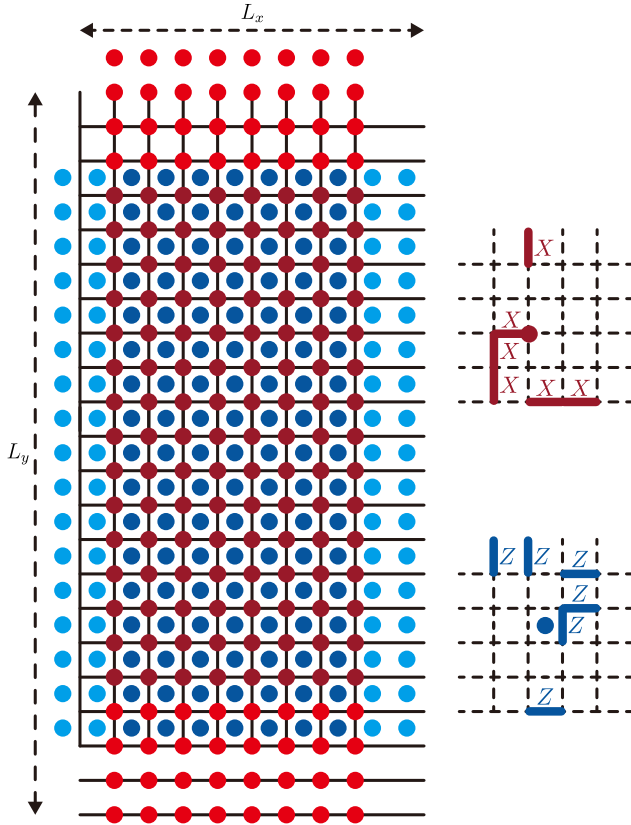


FIG. 17. The $[[381, 10, 13]]$ code on an open square lattice. Bulk stabilizers are specified by two polynomials $f(x, y) = x(1 + x + x^{-1}y^2)$ and $g(x, y) = 1 + y + xy^4$ [45]. The vertical edges of the rightmost column have been removed. In total, there are 120 bulk X -stabilizers (deep red), 119 bulk Z -stabilizers (deep blue), 64 boundary X -stabilizers (red), and 68 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 381 - 120 - 119 - 64 - 68 = 10$. This generates a code family of $k = 10$: $[[2L_x L_y - 2L_x - L_y + 2, 10, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table X below.

Each syndrome is thus a two-component row vector over $\mathbb{Z}_d[x, y, x^{-1}, y^{-1}]$. In the more general case with t stabilizer generators $S_1, \dots, S_t$, the error syndrome becomes a t -component row vector over the same ring.

With the excitation map ϵ in hand, the *topological-order (TO) condition* can be expressed as

$$\ker \epsilon = \sigma, \quad (\text{A22})$$

where σ denotes the stabilizer module, and $\ker \epsilon$ represents all local Pauli operators that commute with the stabilizers. This requirement means that any such commuting operator must itself be a product of stabilizers (i.e., an element of σ). If the TO condition holds, no local observable can distinguish different ground states. In particular, the ground-state degeneracy depends on the topology of the manifold: on a torus, it equals the number of anyon types,

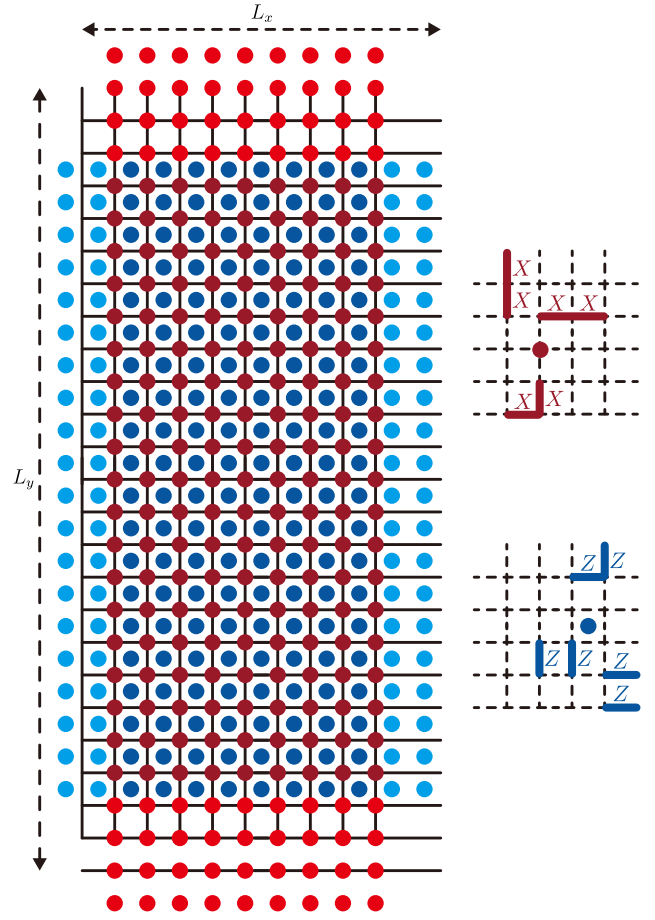


FIG. 18. The $[[494, 11, 15]]$ code on an open square lattice. Bulk stabilizers are specified by two polynomials $f(x, y) = xy^3(1 + x + x^{-1}y^{-3})$ and $g(x, y) = y^3(1 + y + xy^{-3})$ [45]. The vertical edges of the two rightmost columns and the bottom-most row have been removed. There are 171 bulk X -stabilizers (deep red), 160 bulk Z -stabilizers (deep blue), 72 boundary X -stabilizers (red), and 80 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 494 - 171 - 160 - 72 - 80 = 11$. This generates a code family of $k = 11$: $[[2L_x L_y - L_x - L_y + 1, 11, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table XI below.

with each ground state associated with an anyon string operator.

APPENDIX B: $[[54, 6, 4]]$ CODE AND ITS GRAFTED $[[44, 6, 4]]$ CODE

In this appendix, we present a specific construction of the $[[54, 6, 4]]$ code, which achieves the smallest code parameters listed in Table I. For this construction, we use slightly different stabilizers $f(x, y)$ and $g(x, y)$ from those shown in Table I. The code and its grafted version, the $[[44, 6, 4]]$ code, are illustrated in Figs. 12(a) and 12(b), respectively.

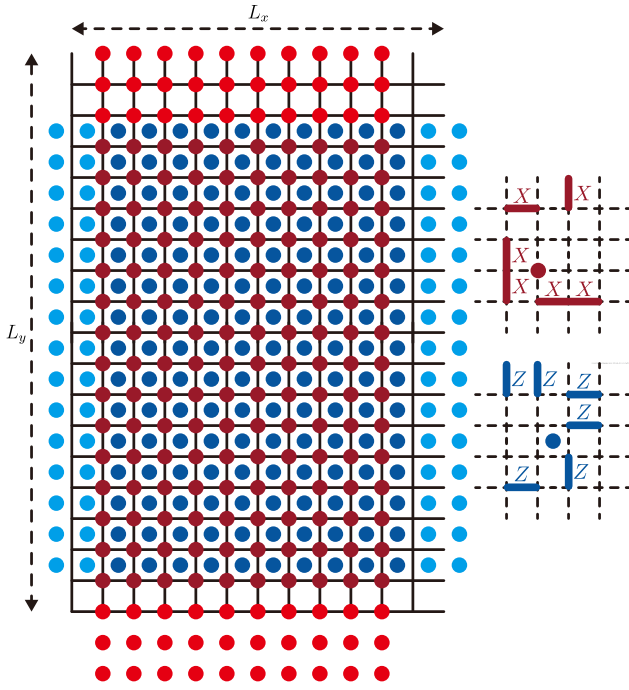


FIG. 19. The $[[432, 12, 12]]$ code on an open square lattice. Bulk stabilizers are specified by two polynomials $f(x, y) = x(1 + x + x^{-1}y^3)$ and $g(x, y) = 1 + y + x^2y^3$ [45]. There are 150 bulk X -stabilizers (deep red), 150 bulk Z -stabilizers (deep blue), 60 boundary X -stabilizers (red), and 60 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 432 - 150 \times 2 - 60 \times 2 = 12$. This generates a code family of $k = 12$: $[[2L_xL_y, 12, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table XII below.

APPENDIX C: QUBIT NUMBERING

In this appendix, we describe our convention for labeling qubits on the open square lattice, as illustrated in Fig. 13. This labeling allows all of our planar code constructions to be expressed as parity-check matrices. The Supplemental Material [125] provides the complete parity-check matrices for all codes presented in this paper, where H_X and H_Z denote the parity-check matrices corresponding to X -type and Z -type stabilizers, respectively, since we focus on CSS codes. Interested readers may use these matrices to verify the code properties with standard tools for qLDPC codes.

APPENDIX D: FAMILIES OF PLANAR qLDPC CODES

In this appendix, we first present tables of the code distances $d(L_x, L_y)$ as functions of the lattice sizes L_x and L_y for the families of planar qLDPC codes derived from the examples in Sec. IV:

- (a) The $k = 6$ code family, derived from the $[[88, 6, 6]]$ code (shown in Fig. 8) in Sec. IV C, has its code distances listed in Table II.

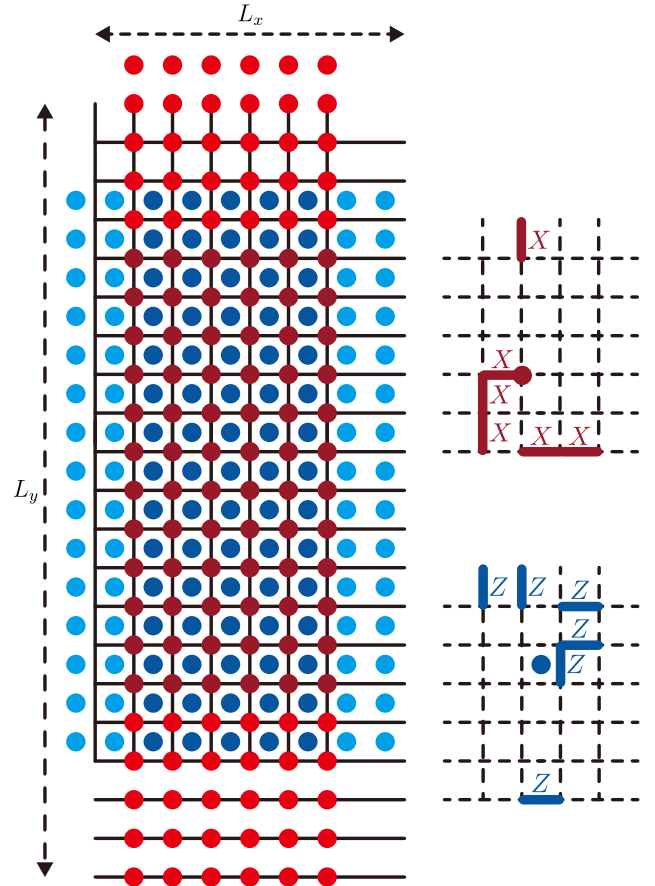


FIG. 20. The $[[279, 12, 9]]$ code on an open square lattice. Bulk stabilizers are specified by two polynomials $f(x, y) = x(1 + x + x^{-1}y^2)$ and $g(x, y) = 1 + y + xy^5$ [45]. There are 72 bulk X -stabilizers (deep red), 75 bulk Z -stabilizers (deep blue), 60 boundary X -stabilizers (red), and 60 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 279 - 72 - 75 - 60 \times 2 = 12$. This generates a code family of $k = 12$: $[[2L_xL_y - 3L_x - L_y + 3, 12, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table XIII below.

- (b) The $k = 7$ code family, derived from the $[[131, 7, 7]]$ code (shown in Fig. 7) in Sec. IV B, has its code distances listed in Table III.
- (c) The $k = 8$ code family, derived from the $[[188, 8, 9]]$ code (shown in Fig. 6) in Sec. IV A, has its code distances listed in Table IV.
- (d) The $k = 8$ code family, derived from the $[[288, 8, 12]]$ code (shown in Fig. 1) in Sec. II, has its code distances listed in Table V.
- (e) The $k = 12$ code family (weight-8 stabilizers), derived from the $[[292, 12, 14]]$ code (shown in Fig. 9) in Sec. IV D, has its code distances listed in Table VI.

Furthermore, we introduce additional families of planar qLDPC codes with logical dimensions ranging from $k = 9$ to $k = 13$:

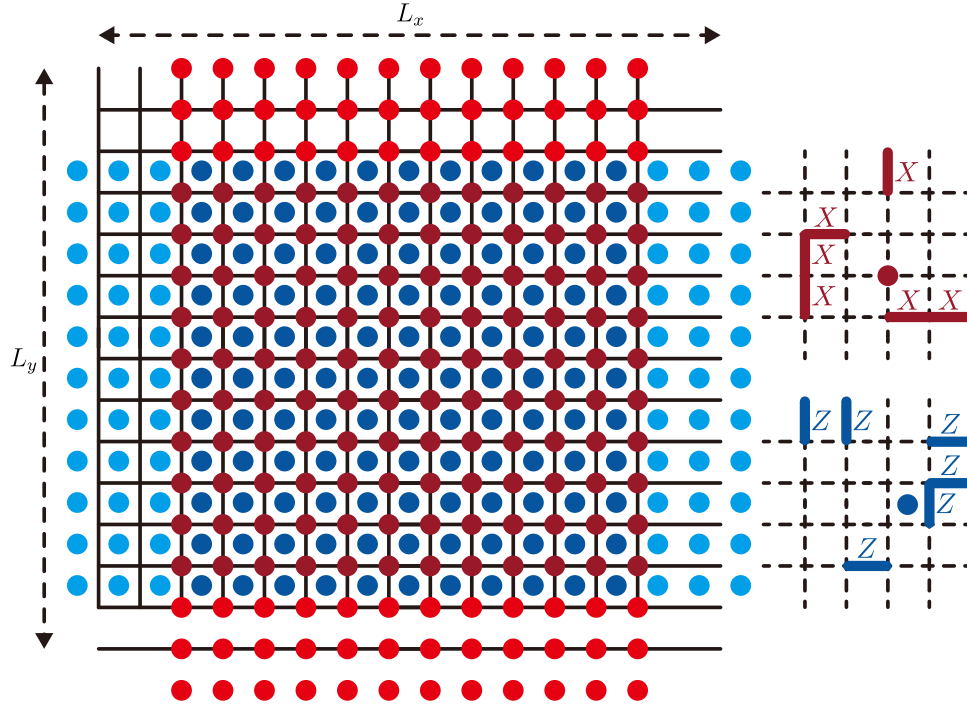


FIG. 21. The $[[392, 13, 11]]$ code on an open square lattice. Bulk stabilizers are specified by two polynomials $f(x, y) = x^2(1 + x + x^{-2}y^2)$ and $g(x, y) = 1 + y + x^2y^3$ [45]. The vertical edges of the rightmost column and the bottommost row have been removed. There are 120 bulk X -stabilizers (deep red), 121 bulk Z -stabilizers (deep blue), 72 boundary X -stabilizers (red), and 66 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 392 - 120 - 121 - 72 - 66 = 13$. This generates a code family of $k = 13$: $[[2L_xL_y - L_x - L_y + 1, 13, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table XIV below.

- (a) The $k = 9$ code family, derived from the $[[441, 9, 15]]$ code (shown in Fig. 14), has its code distances listed in Table VII.
 - (b) The $k = 10$ code family, derived from the $[[381, 10, 13]]$ code (shown in Fig. 17), has its code distances listed in Table X.
 - (c) The $k = 11$ code family, derived from the $[[494, 11, 15]]$ code (shown in Fig. 18), has its code distances listed in Table XI.
 - (d) The $k = 12$ code family, derived from the $[[432, 12, 12]]$ code (shown in Fig. 19), has its code distances listed in Table XII.
 - (e) The $k = 13$ code family, derived from the $[[392, 13, 11]]$ code (shown in Fig. 21), has its code distances listed in Table XIV.
- We have enumerated these codes for various values of L_x and L_y so that the code distances fall within the range $4 \leq$

TABLE XIV. (n, d) for the $k = 13$ family from the $[[392, 13, 11]]$ code for different L_x and L_y .

L_x	L_y											
	7	8	9	10	11	12	13	14	15	16	17	18
8	(98,4)	(113,4)	(128,4)	(143,4)	(158,4)	(173,4)	(188,4)	(203,4)	(218,4)	(233,4)	(248,4)	(263,4)
9	(111,4)	(128,4)	(145,5)	(162,5)	(179,5)	(196,5)	(213,5)	(230,5)	(247,5)	(264,5)	(281,5)	(298,5)
10	(124,4)	(143,4)	(162,5)	(181,5)	(200,5)	(219,5)	(238,5)	(257,5)	(276,5)	(295,5)	(314,5)	(333,5)
11	(137,4)	(158,4)	(179,5)	(200,6)	(221,6)	(242,6)	(263,6)	(284,6)	(305,6)	(326,6)	(347,6)	(368,6)
12	(150,4)	(173,4)	(196,5)	(219,6)	(242,7)	(265,7)	(288,7)	(311,7)	(334,7)	(357,7)	(380,7)	(403,7)
13	(163,4)	(188,4)	(213,5)	(238,6)	(263,7)	(288,9)	(313,9)	(338,9)	(363,9)	(388,9)	(413,9)	(438,9)
14	(176,4)	(203,4)	(230,5)	(257,6)	(284,7)	(311,9)	(338,9)	(365,9)	(392,9)	(419,9)	(446,9)	(473,9)
15	(189,4)	(218,4)	(247,5)	(276,6)	(305,7)	(334,9)	(363,10)	(392,11)	(421,11)	(450,11)	(479,11)	(508,11)
16	(202,4)	(233,4)	(264,5)	(295,6)	(326,7)	(357,9)	(388,10)	(419,11)	(450,11)	(481,12)	(512,12)	(543,12)
17	(215,4)	(248,4)	(281,5)	(314,6)	(347,7)	(380,9)	(413,10)	(446,11)	(479,11)	(512,13)	(545,13)	(578,13)
18	(228,4)	(263,4)	(298,5)	(333,6)	(368,7)	(403,9)	(438,10)	(473,11)	(508,11)	(543,13)	(578,14)	(613,14)
19	(241,4)	(278,4)	(315,5)	(352,6)	(389,7)	(426,9)	(463,10)	(500,11)	(537,11)	(574,13)	(611,14)	(648,16)

TABLE XV. (n, d) for the $k = 13$ family from the $[[495, 13, 13]]$ code for different L_x and L_y .

L_x	L_y								
	5	6	7	8	9	10	11	12	13
13	(109,4)	(133,4)	(157,4)	(181,4)	(205,4)	(229,4)	(253,4)	(277,4)	(301,4)
14	(118,4)	(144,4)	(170,4)	(196,4)	(222,4)	(248,4)	(274,4)	(300,4)	(326,4)
15	(127,4)	(155,5)	(183,5)	(211,5)	(239,5)	(267,5)	(295,5)	(323,5)	(351,5)
16	(136,4)	(166,5)	(196,5)	(226,5)	(256,5)	(286,5)	(316,5)	(346,5)	(376,5)
17	(145,4)	(177,6)	(209,6)	(241,6)	(273,6)	(305,6)	(337,6)	(369,6)	(401,6)
18	(154,4)	(188,6)	(222,7)	(256,7)	(290,7)	(324,7)	(358,7)	(392,7)	(426,7)
19	(163,4)	(199,6)	(235,7)	(271,8)	(307,8)	(343,8)	(379,8)	(415,8)	(451,8)
20	(172,4)	(210,6)	(248,7)	(286,9)	(324,9)	(362,9)	(400,9)	(438,9)	(476,9)
21	(181,4)	(221,6)	(261,7)	(301,9)	(341,10)	(381,10)	(421,10)	(461,10)	(501,10)
22	(190,4)	(232,6)	(274,7)	(316,9)	(358,10)	(400,10)	(442,10)	(484,10)	(526,10)
23	(199,4)	(243,6)	(287,7)	(331,9)	(375,10)	(419,10)	(463,10)	(507,10)	(551,10)
24	(208,4)	(254,6)	(300,7)	(346,9)	(392,10)	(438,10)	(484,10)	(530,10)	(576,10)
25	(217,4)	(265,6)	(313,7)	(361,9)	(409,10)	(457,11)	(505,11)	(553,11)	(601,11)
26	(226,4)	(276,6)	(326,7)	(376,9)	(426,10)	(476,12)	(526,12)	(576,12)	(626,12)
27	(235,4)	(287,6)	(339,7)	(391,9)	(443,10)	(495,13)	(547,13)	(599,13)	(651,13)
28	(244,4)	(298,6)	(352,7)	(406,9)	(460,10)	(514,13)	(568,13)	(622,13)	(676,13)
29	(253,4)	(309,6)	(365,7)	(421,9)	(477,10)	(533,13)	(589,14)	(645,14)	(701,14)
30	(262,4)	(320,6)	(378,7)	(436,9)	(494,10)	(552,13)	(610,15)	(668,15)	(726,15)
31	(271,4)	(331,6)	(391,7)	(451,9)	(511,10)	(571,13)	(631,15)	(691,16)	(751,16)

$d \leq 16$. This regime allows us to compute the code distances precisely, using the integer-programming approach [104]. In contrast, for codes with larger distances, the exact method is computationally prohibitive, so the distances are estimated using a probabilistic algorithm [126], which provides only an upper bound and may not be tight.

APPENDIX E: GRAFTED PLANAR qLDPC CODES

In this appendix, we apply the “lattice-grafting” optimization to the examples from the previous code families. The grafted codes are presented below. It is important

to note that lattice grafting involves randomized choices (since enumerating exponentially many iterative cases is impractical); therefore, we run the algorithm several hundred times for each parent code in Appendix D and select the instance with the smallest number of physical qubits n :

- (a) The grafted $[[78, 6, 6]]$ code, with $kd^2/n = 2.77$, is illustrated in Fig. 23 and is obtained from the $[[88, 6, 6]]$ code (shown in Fig. 8) by removing ten physical qubits.

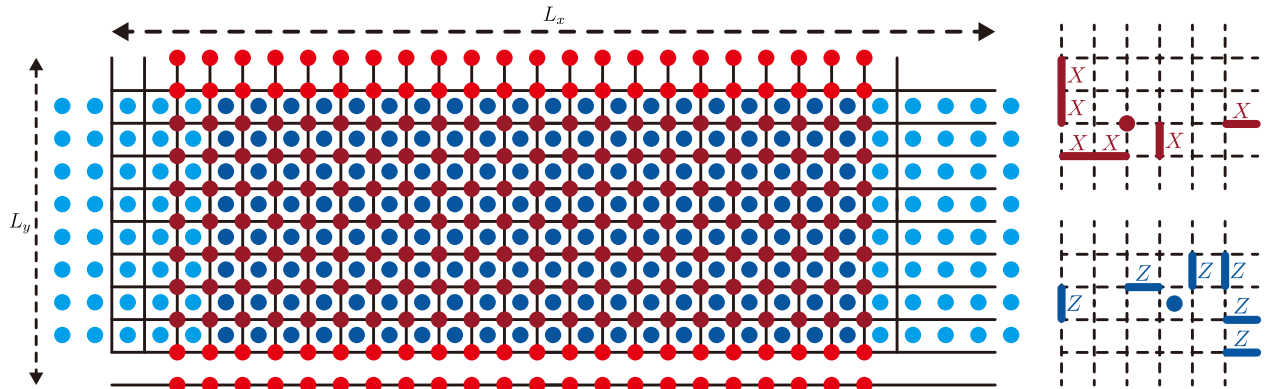


FIG. 22. The $[[495, 13, 13]]$ code on an open square lattice. Bulk stabilizers are specified by two polynomials $f(x, y) = 1 + x + x^5y$ and $g(x, y) = y(1 + y + x^3y^{-1})$ [45]. The vertical edges of the rightmost column and the bottommost row have been removed. There are 154 bulk X -stabilizers (deep red), 160 bulk Z -stabilizers (deep blue), 88 boundary X -stabilizers (red), and 80 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 495 - 154 - 160 - 88 - 80 = 13$. This generates a code family of $k = 13$: $[[2L_xL_y - L_x - 2L_y + 2, 13, d(L_x, L_y)]]$, with code distances $d(L_x, L_y)$ listed in Table XV below.

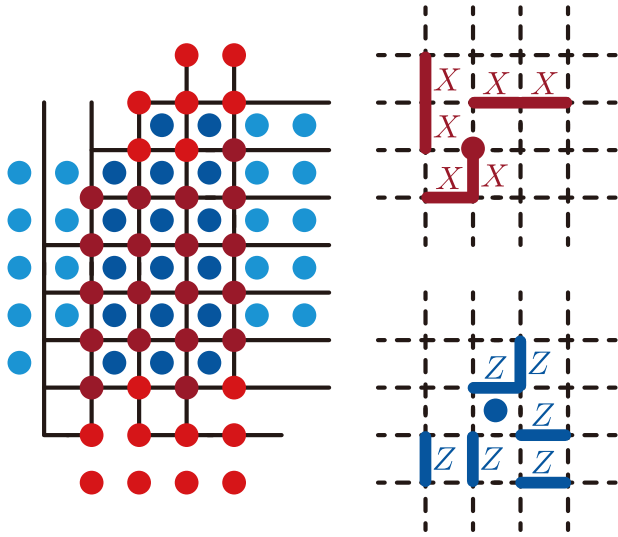


FIG. 23. The grafted $[[78, 6, 6]]$ code. Each red (blue) dot represents an X - (Z -) stabilizer. Deep red and blue dots correspond to bulk stabilizers that are fully supported on the pruned lattice, while dots near the boundary represent stabilizers with truncated edges (i.e., the bulk stabilizer pattern positioned at the boundary dot, retaining only the Pauli operators on edges within the pruned lattice). The design ensures that all truncated X and Z operators still commute, forming valid boundary stabilizers. In total, there are 19 bulk X -stabilizers (deep red), 17 bulk Z -stabilizers (deep blue), 17 boundary X -stabilizers (red), and 19 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 78 - 19 \times 2 - 17 \times 2 = 6$.

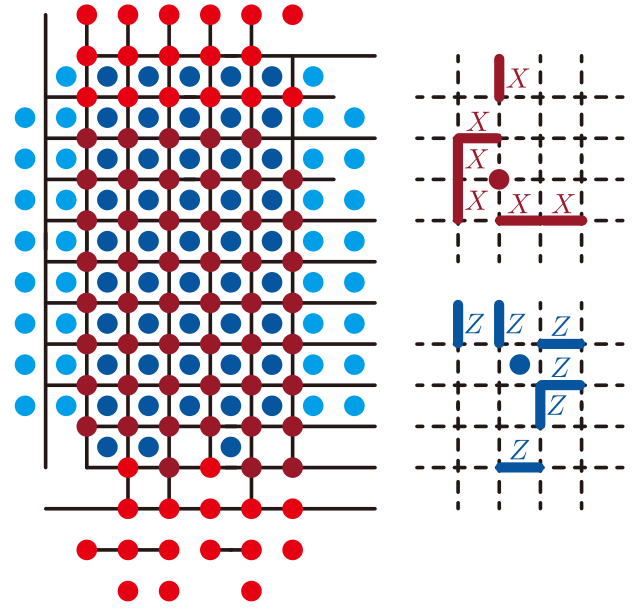


FIG. 25. The grafted $[[173, 8, 9]]$ code. Each red (blue) dot represents an X - (Z -) stabilizer. Deep red and blue dots correspond to bulk stabilizers that are fully supported on the pruned lattice, while dots near the boundary represent stabilizers with truncated edges. There are 50 bulk X -stabilizers (deep red), 48 bulk Z -stabilizers (deep blue), 33 boundary X -stabilizers (red), and 34 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 173 - 50 - 48 - 33 - 34 = 8$.

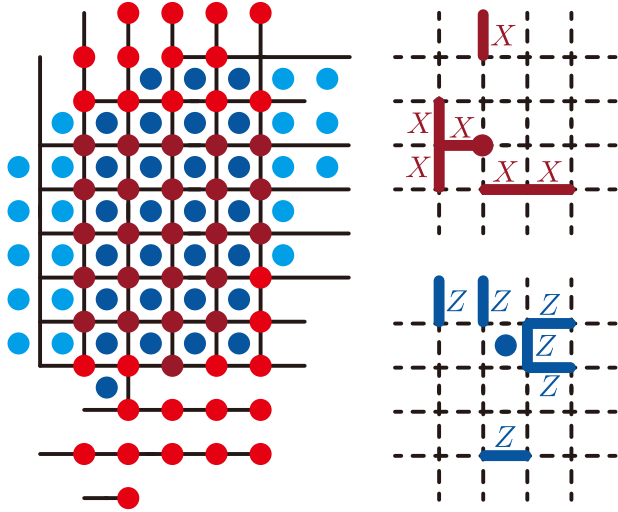


FIG. 24. The grafted $[[107, 7, 7]]$ code. Each red (blue) dot represents an X - (Z -) stabilizer. Deep red and blue dots correspond to bulk stabilizers that are fully supported on the pruned lattice, while dots near the boundary represent stabilizers with truncated edges. There are 24 bulk X -stabilizers (deep red), 28 bulk Z -stabilizers (deep blue), 29 boundary X -stabilizers (red), and 19 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 107 - 24 - 28 - 29 - 19 = 7$.

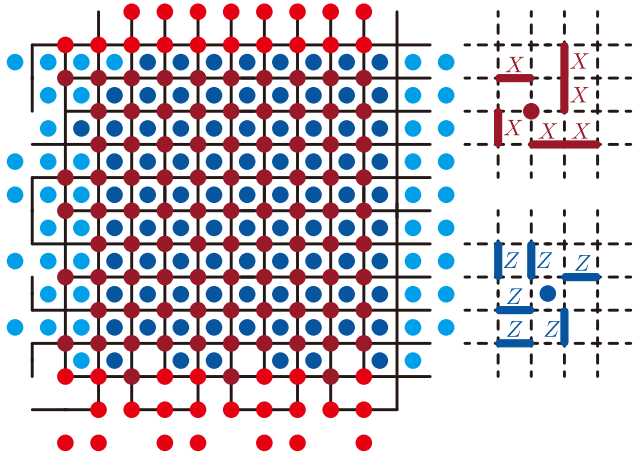


FIG. 26. The grafted $[[268, 8, 12]]$ code. Each red (blue) dot represents an X - (Z -) stabilizer. Deep red and blue dots correspond to bulk stabilizers that are fully supported on the pruned lattice, while dots near the boundary represent stabilizers with truncated edges. There are 89 bulk X -stabilizers (deep red), 87 bulk Z -stabilizers (deep blue), 41 boundary X -stabilizers (red), and 43 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 268 - 89 - 87 - 41 - 43 = 8$.

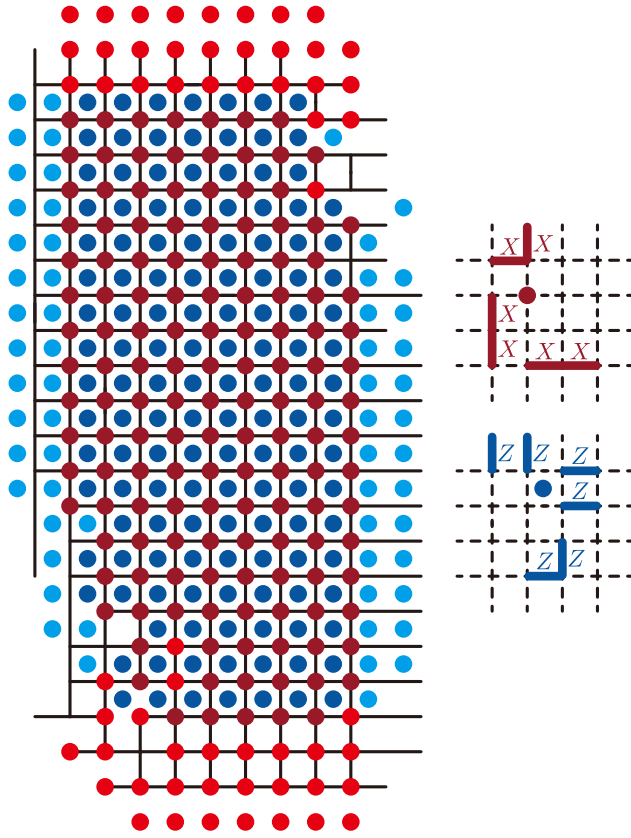


FIG. 27. The grafted $[[405, 9, 15]]$ code. Each red (blue) dot represents an X - (Z -) stabilizer. Deep red and blue dots correspond to bulk stabilizers that are fully supported on the pruned lattice, while dots near the boundary represent stabilizers with truncated edges. There are 143 bulk X -stabilizers (deep red), 136 bulk Z -stabilizers (deep blue), 58 boundary X -stabilizers (red), and 59 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 405 - 143 - 136 - 58 - 59 = 9$.

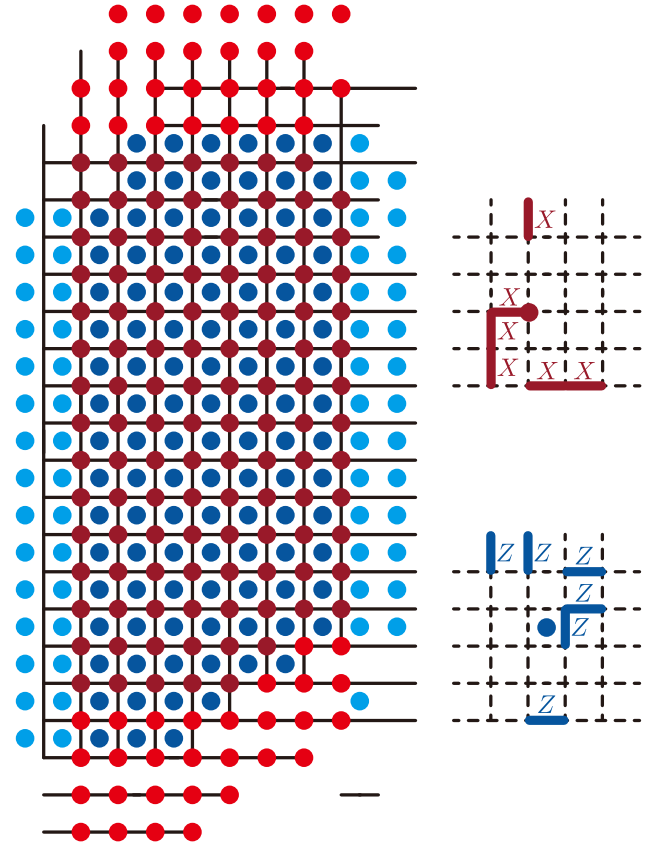


FIG. 28. The grafted $[[348, 10, 13]]$ code. Each red (blue) dot represents an X - (Z -) stabilizer. Deep red and blue dots correspond to bulk stabilizers that are fully supported on the pruned lattice, while dots near the boundary represent stabilizers with truncated edges. There are 114 bulk X -stabilizers (deep red), 109 bulk Z -stabilizers (deep blue), 57 boundary X -stabilizers (red), and 58 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 348 - 114 - 109 - 57 - 58 = 10$.

- (b) The grafted $[[107, 7, 7]]$ code, with $kd^2/n = 3.21$, is illustrated in Fig. 24 and is obtained from the $[[131, 7, 7]]$ code (shown in Fig. 7) by removing 24 physical qubits.
- (c) The grafted $[[173, 8, 9]]$ code, with $kd^2/n = 3.75$, is illustrated in Fig. 25 and is obtained from the $[[188, 8, 9]]$ code (shown in Fig. 6) by removing 15 physical qubits.
- (d) The grafted $[[268, 8, 12]]$ code, with $kd^2/n = 4.30$, is illustrated in Fig. 26 and is obtained from the $[[288, 12, 12]]$ code (shown in Fig. 1) by removing 20 physical qubits.
- (e) The grafted $[[405, 9, 15]]$ code, with $kd^2/n = 5.00$, is illustrated in Fig. 27 and is obtained from the $[[441, 9, 15]]$ code (shown in Fig. 14) by removing 36 physical qubits.
- (f) The grafted $[[348, 10, 13]]$ code, with $kd^2/n = 4.86$, is illustrated in Fig. 28 and is obtained from the $[[381, 10, 13]]$ code (shown in Fig. 17) by removing 33 physical qubits.
- (g) The grafted $[[450, 11, 15]]$ code, with $kd^2/n = 5.50$, is illustrated in Fig. 29 and is obtained from the $[[494, 11, 15]]$ code (shown in Fig. 18) by removing 44 physical qubits.
- (h) The grafted $[[386, 12, 12]]$ code, with $kd^2/n = 4.48$, is illustrated in Fig. 30 and is obtained from the $[[432, 12, 12]]$ code (shown in Fig. 19) by removing 46 physical qubits.
- (i) The grafted $[[362, 13, 11]]$ code, with $kd^2/n = 4.35$, is illustrated in Fig. 31 and is obtained from the $[[392, 13, 11]]$ code (shown in Fig. 21) by removing 30 physical qubits.

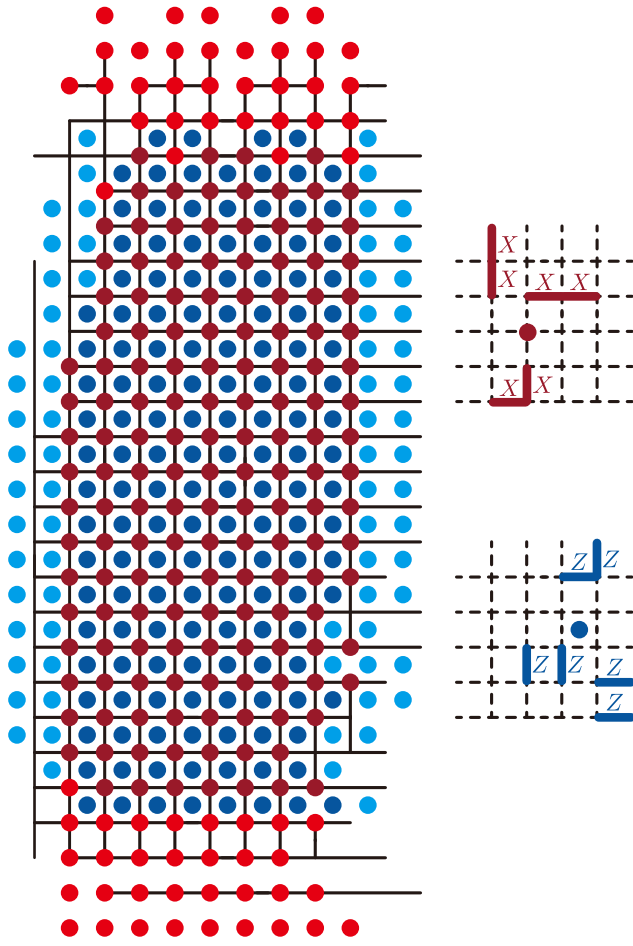


FIG. 29. The grafted $[[450, 11, 15]]$ code. Each red (blue) dot represents an X - (Z -) stabilizer. Deep red and blue dots correspond to bulk stabilizers that are fully supported on the pruned lattice, while dots near the boundary represent stabilizers with truncated edges. There are 154 bulk X -stabilizers (deep red), 148 bulk Z -stabilizers (deep blue), 66 boundary X -stabilizers (red), and 71 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 450 - 154 - 148 - 66 - 71 = 11$.

- (j) The grafted $[[282, 12, 14]]$ code (weight-8 stabilizers), with $kd^2/n = 8.34$, is illustrated in Fig. 32 and is obtained from the $[[292, 12, 14]]$ code (shown in Fig. 9) by removing ten physical qubits.

This lattice-grafting process removes a subset of the physical qubits while preserving the code distances, geometric locality, and maximum stabilizer weights. Although the process typically produces highly irregular lattice boundaries, the qubits can be rearranged to regularize the boundaries for experimental implementations.

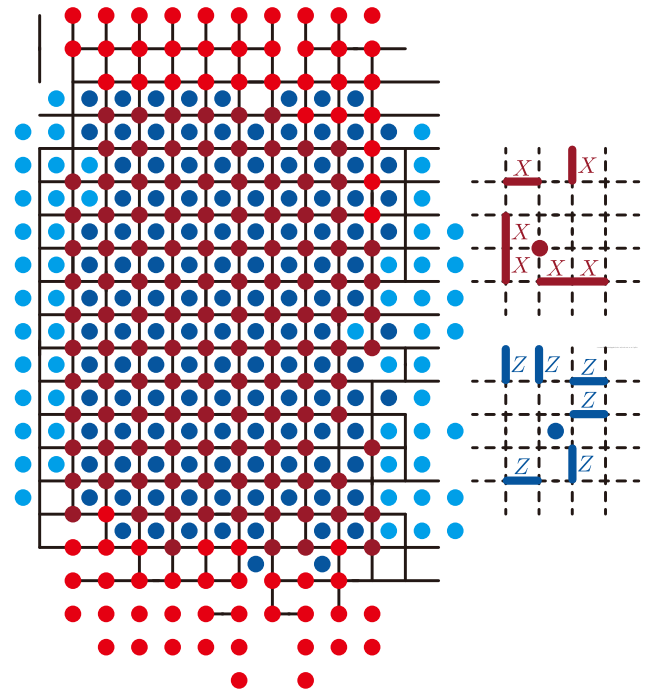


FIG. 30. The grafted $[[386, 12, 12]]$ code. Each red (blue) dot represents an X - (Z -) stabilizer. Deep red and blue dots correspond to bulk stabilizers that are fully supported on the pruned lattice, while dots near the boundary represent stabilizers with truncated edges. There are 121 bulk X -stabilizers (deep red), 128 bulk Z -stabilizers (deep blue), 71 boundary X -stabilizers (red), and 54 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 386 - 121 - 128 - 71 - 54 = 12$.

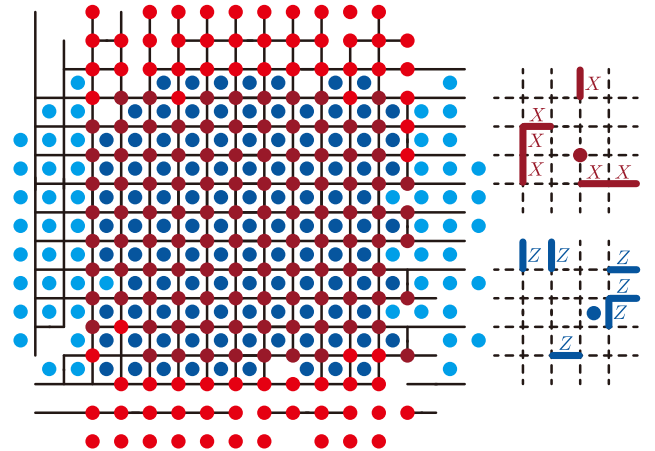


FIG. 31. The grafted $[[362, 13, 11]]$ code. Each red (blue) dot represents an X - (Z -) stabilizer. Deep red and blue dots correspond to bulk stabilizers that are fully supported on the pruned lattice, while dots near the boundary represent stabilizers with truncated edges. There are 107 bulk X -stabilizers (deep red), 110 bulk Z -stabilizers (deep blue), 76 boundary X -stabilizers (red), and 56 boundary Z -stabilizers (blue), yielding a logical dimension of $k = 362 - 107 - 110 - 76 - 56 = 13$.

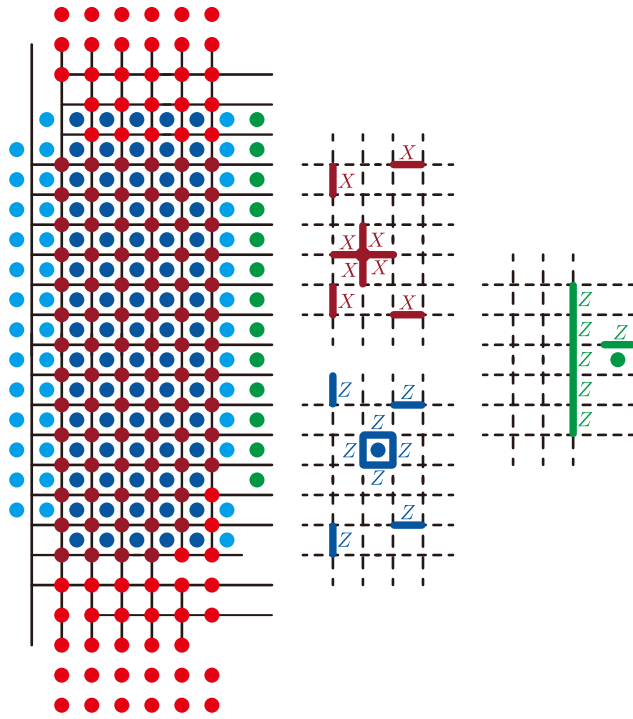


FIG. 32. The grafted $[[282, 12, 14]]$ code. Each red (blue) dot represents an X - (Z -) stabilizer. Deep red and blue dots correspond to bulk stabilizers that are fully supported on the pruned lattice, while dots near the boundary represent stabilizers with truncated edges. Furthermore, the green terms indicate the secondary boundary gauge operators that do not originate from truncating the bulk stabilizers. There are 80 bulk X -stabilizers (deep red), 75 bulk Z -stabilizers (deep blue), 61 boundary X -stabilizers (red), 41 boundary Z -stabilizers (blue) and 13 boundary Z -stabilizers (green), yielding a logical dimension of $k = 282 - 80 - 75 - 61 - 41 - 13 = 12$.

[1] P. W. Shor, Scheme for reducing decoherence in quantum computer memory, *Phys. Rev. A* **52**, R2493 (1995).
[2] A. M. Steane, Error correcting codes in quantum theory, *Phys. Rev. Lett.* **77**, 793 (1996).
[3] E. Knill and R. Laflamme, Theory of quantum error-correcting codes, *Phys. Rev. A* **55**, 900 (1997).
[4] D. Gottesman, Stabilizer codes and quantum error correction, *ArXiv:quant-ph/9705052*.
[5] A. Kitaev, Fault-tolerant quantum computation by anyons, *Ann. Phys. (NY)* **303**, 2 (2003).
[6] S. B. Bravyi and A. Y. Kitaev, Quantum codes on a lattice with boundary, *ArXiv:quant-ph/9811052*.
[7] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, *J. Math. Phys.* **43**, 4452 (2002).
[8] B. M. Terhal, Quantum error correction for quantum memories, *Rev. Mod. Phys.* **87**, 307 (2015).
[9] G. Semeghini, H. Levine, A. Keesling, S. Ebadi, T. T. Wang, D. Bluvstein, R. Verresen, H. Pichler, M.

Kalinowski, R. Samajdar, A. Omran, S. Sachdev, A. Vishwanath, M. Greiner, V. Vuletić, and M. D. Lukin, Probing topological spin liquids on a programmable quantum simulator, *Science* **374**, 1242 (2021).
[10] R. Verresen, M. D. Lukin, and A. Vishwanath, Prediction of toric code topological order from Rydberg blockade, *Phys. Rev. X* **11**, 031005 (2021).
[11] D. Bluvstein, H. Levine, G. Semeghini, T. T. Wang, S. Ebadi, M. Kalinowski, A. Keesling, N. Maskara, H. Pichler, M. Greiner, V. Vuletić, and M. D. Lukin, A quantum processor based on coherent transport of entangled atom arrays, *Nature* **604**, 451 (2022).
[12] Google Quantum AI, Suppressing quantum errors by scaling a surface code logical qubit, *Nature* **614**, 676 (2023).
[13] Google Quantum AI and Collaborators, Non-Abelian braiding of graph vertices in a superconducting processor, *Nature* **618**, 264 (2023).
[14] Google Quantum AI and Collaborators, Quantum error correction below the surface code threshold, *Nature* **638**, 920 (2024).
[15] M. Iqbal, N. Tantivasadakarn, T. M. Gatterman, J. A. Gerber, K. Gilmore, D. Gresh, A. Hankin, N. Hewitt, C. V. Horst, M. Matheny, T. Mengle, B. Neyenhuis, A. Vishwanath, M. Foss-Feig, R. Verresen, and H. Dreyer, Topological order from measurements and feed-forward on a trapped ion quantum computer, *Commun. Phys.* **7**, 205 (2024).
[16] M. Iqbal, N. Tantivasadakarn, R. Verresen, S. L. Campbell, J. M. Dreiling, C. Figgatt, J. P. Gaebler, J. Johansen, M. Mills, S. A. Moses, J. M. Pino, A. Ransford, M. Rowe, P. Siegfried, R. P. Stutz, M. Foss-Feig, A. Vishwanath, and H. Dreyer, Non-Abelian topological order and anyons on a trapped-ion processor, *Nature* **626**, 505 (2024).
[17] I. Cong, N. Maskara, M. C. Tran, H. Pichler, G. Semeghini, S. F. Yelin, S. Choi, and M. D. Lukin, Enhancing detection of topological order by local error correction, *Nat. Commun.* **15**, 1527 (2024).
[18] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Surface codes: Towards practical large-scale quantum computation, *Phys. Rev. A* **86**, 032324 (2012).
[19] D. Litinski, A game of surface codes: Large-scale quantum computing with lattice surgery, *Quantum* **3**, 128 (2019).
[20] N. Delfosse, P. Iyer, and D. Poulin, Generalized surface codes and packing of logical qubits, *ArXiv:1606.07116*.
[21] N. P. Breuckmann and J. N. Eberhardt, Quantum low-density parity-check codes, *PRX Quantum* **2**, 040101 (2021).
[22] P. Panteleev and G. Kalachev, Degenerate quantum LDPC codes with good finite length performance, *Quantum* **5**, 585 (2021).
[23] N. P. Breuckmann and J. N. Eberhardt, Balanced product quantum codes, *IEEE Trans. Inf. Theory* **67**, 6653 (2021).
[24] P. Panteleev and G. Kalachev, in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022 (Association for Computing Machinery, New York, 2022), p. 375.
[25] T.-C. Lin and M.-H. Hsieh, in *2022 IEEE International Symposium on Information Theory (ISIT)* (2022), p. 1175.

- [26] I. Dinur, M.-H. Hsieh, T.-C. Lin, and T. Vidick, in *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023 (Association for Computing Machinery, New York, 2023), p. 905.
- [27] S. Bravyi and B. Terhal, A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes, *New J. Phys.* **11**, 043029 (2009).
- [28] S. Bravyi, D. Poulin, and B. Terhal, Tradeoffs for reliable quantum information storage in 2D systems, *Phys. Rev. Lett.* **104**, 050503 (2010).
- [29] A. A. Kovalev and L. P. Pryadko, Quantum Kronecker sum-product low-density parity-check codes with finite rate, *Phys. Rev. A* **88**, 012311 (2013).
- [30] R. Wang and L. P. Pryadko, Distance bounds for generalized bicycle codes, *Symmetry* **14**, 1348 (2022).
- [31] S. Bravyi, A. W. Cross, J. M. Gambetta, D. Maslov, P. Rall, and T. J. Yoder, High-threshold and low-overhead fault-tolerant quantum memory, *Nature* **627**, 778 (2024).
- [32] M. Wang and F. Mueller, Coprime bivariate bicycle codes and their properties, [ArXiv:2408.10001](https://arxiv.org/abs/2408.10001).
- [33] M. Wang and F. Mueller, in *2024 IEEE International Conference on Quantum Computing and Engineering (QCE) (2024)*, Vol. 2, p. 412.
- [34] R. Tiew and N. P. Breuckmann, Low-overhead entangling gates from generalised Dehn twists, *IEEE Trans. Inf. Theory* **71**, 5452 (2025).
- [35] S. Wolanski and B. Barber, Ambiguity clustering: An accurate and efficient decoder for qLDPC codes, [ArXiv:2406.14527](https://arxiv.org/abs/2406.14527).
- [36] A. Gong, S. Cammerer, and J. M. Renes, Toward low-latency iterative decoding of qLDPC codes under circuit-level noise, [ArXiv:2403.18901](https://arxiv.org/abs/2403.18901).
- [37] A. S. Maan and A. Paler, Machine learning message-passing for the scalable decoding of QLDPC codes, *npj Quantum Inf.* **11**, 78 (2025).
- [38] A. Cowtan, SSIP: Automated surgery with quantum LDPC codes, [ArXiv:2407.09423](https://arxiv.org/abs/2407.09423).
- [39] M. H. Shaw and B. M. Terhal, Lowering connectivity requirements for bivariate bicycle codes using morphing circuits, *Phys. Rev. Lett.* **134**, 090602 (2025).
- [40] A. Cross, Z. He, P. Rall, and T. Yoder, Linear-size ancilla systems for logical measurements in QLDPC codes, [ArXiv:2407.18393](https://arxiv.org/abs/2407.18393).
- [41] L. Voss, S. J. Xian, T. Haug, and K. Bharti, Multivariate bicycle codes, *Phys. Rev. A* **111**, L060401 (2025).
- [42] N. Berthusen, D. Devulapalli, E. Schoute, A. M. Childs, M. J. Gullans, A. V. Gorshkov, and D. Gottesman, Toward a 2D local implementation of quantum low-density parity-check codes, *PRX Quantum* **6**, 010306 (2025).
- [43] J. N. Eberhardt and V. Steffan, Logical operators and fold-transversal gates of bivariate bicycle codes, *IEEE Trans. Inf. Theory* **71**, 1140 (2025).
- [44] H.-K. Lin, X. Liu, P. K. Lim, and L. P. Pryadko, Single-shot and two-shot decoding with generalized bicycle codes, [ArXiv:2502.19406](https://arxiv.org/abs/2502.19406).
- [45] Z. Liang, K. Liu, H. Song, and Y.-A. Chen, Generalized toric codes on twisted tori for quantum error correction, *PRX Quantum* **6**, 020357 (2025).
- [46] Z. Liang and Y.-A. Chen, Self-dual bivariate bicycle codes with transversal Clifford gates, [ArXiv:2510.05211](https://arxiv.org/abs/2510.05211).
- [47] H. Bombin, Structure of 2D topological stabilizer codes, *Commun. Math. Phys.* **327**, 387 (2014).
- [48] J. Haah, Commuting Pauli Hamiltonians as maps between free modules, *Commun. Math. Phys.* **324**, 351 (2013).
- [49] J. Haah, Algebraic methods for quantum codes on lattices, *Rev. Colomb. Mat.* **50**, 299 (2016).
- [50] J. Haah, Classification of translation invariant topological Pauli stabilizer codes for prime dimensional qudits on two-dimensional lattices, *J. Math. Phys.* **62**, 012201 (2021).
- [51] B. Ruba and B. Yang, Homological invariants of Pauli stabilizer codes, *Commun. Math. Phys.* **405**, 126 (2024).
- [52] R. Dijkgraaf and E. Witten, Topological gauge theories and group cohomology, *Commun. Math. Phys.* **129**, 393 (1990).
- [53] X.-G. Wen, Topological order and edge structure of $\nu = 1/2$ quantum Hall state, *Phys. Rev. Lett.* **70**, 355 (1993).
- [54] A. Kitaev, Anyons in an exactly solved model and beyond, *Ann. Phys.* **321**, 2 (2006), January Special Issue.
- [55] H. Bombin and M. A. Martin-Delgado, Topological quantum distillation, *Phys. Rev. Lett.* **97**, 180501 (2006).
- [56] M. Levin and X.-G. Wen, Detecting topological order in a ground state wave function, *Phys. Rev. Lett.* **96**, 110405 (2006).
- [57] X. Chen, Z.-C. Gu, and X.-G. Wen, Complete classification of one-dimensional gapped quantum phases in interacting spin systems, *Phys. Rev. B* **84**, 235128 (2011).
- [58] M. Levin and Z.-C. Gu, Braiding statistics approach to symmetry-protected topological phases, *Phys. Rev. B* **86**, 115109 (2012).
- [59] X. Chen, Z.-C. Gu, Z.-X. Liu, and X.-G. Wen, Symmetry-protected topological orders in interacting bosonic systems, *Science* **338**, 1604 (2012).
- [60] L. Cincio and G. Vidal, Characterizing topological order by studying the ground states on an infinite cylinder, *Phys. Rev. Lett.* **110**, 067208 (2013).
- [61] Z.-C. Gu, Z. Wang, and X.-G. Wen, Lattice model for fermionic toric code, *Phys. Rev. B* **90**, 085140 (2014).
- [62] C.-M. Jian and X.-L. Qi, Layer construction of 3D topological states and string braiding statistics, *Phys. Rev. X* **4**, 041043 (2014).
- [63] H. Bombin, Gauge color codes: Optimal transversal gates and gauge fixing in topological stabilizer codes, [ArXiv:1311.0879](https://arxiv.org/abs/1311.0879).
- [64] C. Wang and M. Levin, Topological invariants for gauge theories and symmetry-protected topological phases, *Phys. Rev. B* **91**, 165119 (2015).
- [65] P. Ye and Z.-C. Gu, Vortex-line condensation in three dimensions: A physical mechanism for bosonic topological insulators, *Phys. Rev. X* **5**, 021029 (2015).
- [66] B. Yoshida, Topological phases with generalized global symmetries, *Phys. Rev. B* **93**, 155131 (2016).
- [67] P. Ye and Z.-C. Gu, Topological quantum field theory of three-dimensional bosonic Abelian-symmetry-protected topological phases, *Phys. Rev. B* **93**, 205157 (2016).
- [68] A. Kapustin and R. Thorngren, in *Algebra, Geometry, and Physics in the 21st Century: Kontsevich Festschrift*, edited by D. Auroux, L. Katzarkov, T. Pantev, Y. Soibelman, and Y. Tschinkel (Springer International Publishing, Cham, 2017), p. 177.

- [69] Y.-A. Chen, A. Kapustin, and D. Radicevic, Exact bosonization in two spatial dimensions and a new class of lattice gauge theories, *Ann. Phys. (NY)* **393**, 234 (2018).
- [70] T. Lan, L. Kong, and X.-G. Wen, Classification of $(3 + 1)$ D bosonic topological orders: The case when pointlike excitations are all bosons, *Phys. Rev. X* **8**, 021074 (2018).
- [71] M. Cheng, N. Tantivasadakarn, and C. Wang, Loop braiding statistics and interacting fermionic symmetry-protected topological phases in three dimensions, *Phys. Rev. X* **8**, 011054 (2018).
- [72] AtMa P. O. Chan, P. Ye, and S. Ryu, Braiding with Borromean rings in $(3 + 1)$ -dimensional spacetime, *Phys. Rev. Lett.* **121**, 061601 (2018).
- [73] Y.-A. Chen, A. Kapustin, A. Turzillo, and M. You, Free and interacting short-range entangled phases of fermions: Beyond the tenfold way, *Phys. Rev. B* **100**, 195128 (2019).
- [74] B. Han, H. Wang, and P. Ye, Generalized Wen-Zee terms, *Phys. Rev. B* **99**, 205120 (2019).
- [75] Q.-R. Wang, Y. Qi, and Z.-C. Gu, Anomalous symmetry protected topological states in interacting fermion systems, *Phys. Rev. Lett.* **123**, 207003 (2019).
- [76] Y.-A. Chen and A. Kapustin, Bosonization in three spatial dimensions and a 2-form gauge theory, *Phys. Rev. B* **100**, 245127 (2019).
- [77] T. Lan and X.-G. Wen, Classification of $3 + 1$ D bosonic topological orders (ii): The case when some pointlike excitations are fermions, *Phys. Rev. X* **9**, 021005 (2019).
- [78] Y.-A. Chen, Exact bosonization in arbitrary dimensions, *Phys. Rev. Res.* **2**, 033527 (2020).
- [79] Y.-A. Chen, T. D. Ellison, and N. Tantivasadakarn, Disentangling supercohomology symmetry-protected topological phases in three spatial dimensions, *Phys. Rev. Res.* **3**, 013056 (2021).
- [80] M. Barkeshli, Y.-A. Chen, P.-S. Hsin, and N. Manjunath, Classification of $(2 + 1)$ D invertible fermionic topological phases with symmetry, *Phys. Rev. B* **105**, 235143 (2022).
- [81] T. Johnson-Freyd, On the classification of topological orders, *Commun. Math. Phys.* **393**, 989 (2022).
- [82] T. D. Ellison, Y.-A. Chen, A. Dua, W. Shirley, N. Tantivasadakarn, and D. J. Williamson, Pauli stabilizer models of twisted quantum doubles, *PRX Quantum* **3**, 010353 (2022).
- [83] Y.-A. Chen and S. Tata, Higher cup products on hypercubic lattices: Application to lattice models of topological phases, *J. Math. Phys.* **64**, 091902 (2023).
- [84] Y.-A. Chen and P.-S. Hsin, Exactly solvable lattice Hamiltonians and gravitational anomalies, *SciPost Phys.* **14**, 089 (2023).
- [85] M. Barkeshli, Y.-A. Chen, S.-J. Huang, R. Kobayashi, N. Tantivasadakarn, and G. Zhu, Codimension-2 defects and higher symmetries in $(3+1)$ D topological phases, *SciPost Phys.* **14**, 065 (2023).
- [86] R. Kobayashi and G. Zhu, Cross-cap defects and fault-tolerant logical gates in the surface code and the honeycomb Floquet code, *PRX Quantum* **5**, 020360 (2024).
- [87] M. Barkeshli, Y.-A. Chen, P.-S. Hsin, and R. Kobayashi, Higher-group symmetry in finite gauge theory and stabilizer codes, *SciPost Phys.* **16**, 089 (2024).
- [88] M. Barkeshli, P.-S. Hsin, and R. Kobayashi, Higher-group symmetry of $(3+1)$ D fermionic $\mathbb{Z}_2$ gauge theory: Logical CCZ, CS, and T gates from higher symmetry, *SciPost Phys.* **16**, 122 (2024).
- [89] R. Kobayashi, Y. Li, H. Xue, P.-S. Hsin, and Y.-A. Chen, Generalized statistics on lattices, *ArXiv:2412.01886*.
- [90] P.-S. Hsin, R. Kobayashi, and G. Zhu, Classifying logical gates in quantum codes via cohomology operations and symmetry, *ArXiv:2411.15848*.
- [91] M. Sun, B. Yang, Z. Wang, N. Tantivasadakarn, and Y.-A. Chen, Clifford quantum cellular automata from topological quantum field theories and invertible subalgebras, *ArXiv:2509.07099*.
- [92] J. N. Eberhardt, F. R. F. Pereira, and V. Steffan, Pruning qLDPC codes: Towards bivariate bicycle codes with open boundary conditions, *ArXiv:2412.04181*.
- [93] V. Steffan, S. H. Choe, N. P. Breuckmann, F. R. F. Pereira, and J. N. Eberhardt, Tile codes: High-efficiency quantum codes on a lattice with boundary, *ArXiv:2504.09171*.
- [94] A. Dua, D. J. Williamson, J. Haah, and M. Cheng, Compactifying fracton stabilizer models, *Phys. Rev. B* **99**, 245135 (2019).
- [95] A. Dua, P. Sarkar, D. J. Williamson, and M. Cheng, Bifurcating entanglement-renormalization group flows of fracton stabilizer models, *Phys. Rev. Res.* **2**, 033021 (2020).
- [96] J. Sullivan, A. Dua, and M. Cheng, Fractonic topological phases from coupled wires, *Phys. Rev. Res.* **3**, 023123 (2021).
- [97] Z. Liang, Y. Xu, J. T. Iosue, and Y.-A. Chen, Extracting topological orders of generalized Pauli stabilizer codes in two dimensions, *PRX Quantum* **5**, 030328 (2024).
- [98] Z. Liang, B. Yang, J. T. Iosue, and Y.-A. Chen, Operator algebra and algorithmic construction of boundaries and defects in $(2 + 1)$ D topological Pauli stabilizer codes, *ArXiv:2410.11942*.
- [99] L. Z. Cohen, I. H. Kim, S. D. Bartlett, and B. J. Brown, Low-overhead fault-tolerant quantum computing using long-range connectivity, *Sci. Adv.* **8**, eabn1717 (2022).
- [100] A. Cowtan, Z. He, D. J. Williamson, and T. J. Yoder, Parallel logical measurements via quantum code surgery, *ArXiv:2503.05003*.
- [101] S. Bravyi and A. Kitaev, Universal quantum computation with ideal Clifford gates and noisy ancillas, *Phys. Rev. A* **71**, 022316 (2005).
- [102] Y. Wang and Y. Gu, Magic teleportation with generalized lattice surgery, *ArXiv:2503.19758*.
- [103] C. T. Aitchison, D. Bulmash, A. Dua, A. C. Doherty, and D. J. Williamson, Boundaries and defects in the cubic code, *Phys. Rev. B* **109**, 205125 (2024).
- [104] A. Landahl, *Fault-tolerant quantum computing with color codes* (University of Southern California (USC), 2011).
- [105] E. Rowell, R. Stong, and Z. Wang, On classification of modular tensor categories, *Commun. Math. Phys.* **292**, 343 (2009).
- [106] Z. Wang, *Topological Quantum Computation* (American Mathematical Society, Providence, RI, 2010), Vol. 112.
- [107] L. Wang and Z. Wang, In and around Abelian anyon models, *J. Phys. A: Math. Theor.* **53**, 505203 (2020).

- [108] J. Plavnik, A. Schopieray, Z. Yu, and Q. Zhang, Modular tensor categories, subcategories, and Galois orbits, *Transform. Groups* **29**, 1623 (2023).
- [109] The Hausdorff dimension of the Sierpinski triangle is $d = \frac{\log 3}{\log 2}$, obtained by solving $2^d = 3$.
- [110] For $L > 16$, the minimal-weight operator begins to transition from fractal-like to stringlike (though the resulting string remains relatively wide and irregular). As L increases, the Pauli weight of fractal operators grows superlinearly (scaling approximately as $L^{1.59}$), making such operators quickly unfavorable.
- [111] This condition arises from the TQO-1 condition as defined in Refs. [127,128] for a broader range of commuting projector Hamiltonians.
- [112] This terminology is inspired by subsystem codes, where the gauge operators do not commute, and their commutants form the stabilizer group.
- [113] H. Bombin, Gauge color codes: Optimal transversal gates and gauge fixing in topological stabilizer codes, *New J. Phys.* **17**, 083002 (2015).
- [114] D. Poulin, Stabilizer formalism for operator quantum error correction, *Phys. Rev. Lett.* **95**, 230504 (2005).
- [115] D. Bacon, Operator quantum error-correcting subsystems for self-correcting quantum memories, *Phys. Rev. A* **73**, 012340 (2006).
- [116] T. D. Ellison, Y.-A. Chen, A. Dua, W. Shirley, N. Tantivasadakarn, and D. J. Williamson, Pauli topological subsystem codes from Abelian anyon theories, *Quantum* **7**, 1137 (2023).
- [117] P. Etingof, D. Nikshych, and V. Ostrik, Fusion categories and homotopy theory, *Quantum Topol.* **1**, 209 (2010).
- [118] A. Kapustin and N. Saulina, Topological boundary conditions in Abelian Chern-Simons theory, *Nucl. Phys. B* **845**, 393 (2011).
- [119] L. Kong, Anyon condensation and tensor categories, *Nucl. Phys. B* **886**, 436 (2014).
- [120] J. Kaidi, Z. Komargodski, K. Ohmori, S. Seifnashri, and S.-H. Shao, Higher central charges and topological boundaries in 2+1-dimensional TQFTs, *SciPost Phys.* **13**, 067 (2022).
- [121] Y.-A. Chen and Y. Xu, Equivalence between fermion-to-qubit mappings in two spatial dimensions, *PRX Quantum* **4**, 010326 (2023).
- [122] If the lattice dimensions are not multiples of 217, the maximal logical dimension is not achieved. For example, on a 7×7 torus the $(-2, 2)$ -BB code yields a $[[98, 6, 12]]$ code [45].
- [123] K. Chen, Y. Liu, Y. Zhang, Z. Liang, Y.-A. Chen, K. Liu, and H. Song, Anyon theory and topological frustration of high-efficiency quantum low-density parity-check codes, *Phys. Rev. Lett.* **135**, 076603 (2025).
- [124] At the corners, translation symmetry is not well defined, and X and Z operators may overlap and anticommute. To resolve this, we adopt the convention that if an X term and a Z term anticommute, neither is included in the stabilizer Hamiltonian.
- [125] See Supplemental Material at <http://link.aps.org/supplemental/10.1103/qv65-vmzr> for the parity-check matrices of planar qLDPC codes constructed in this manuscript.
- [126] L. P. Pryadko, V. A. Shabashov, and V. K. Kozin, QDisTnd: A GAP package for computing the distance of quantum error-correcting codes, *J. Open Source Softw.* **7**, 4120 (2022).
- [127] S. Bravyi, M. B. Hastings, and S. Michalakis, Topological quantum order: Stability under local perturbations, *J. Math. Phys.* **51**, 093512 (2010).
- [128] S. Bravyi and M. B. Hastings, A short proof of stability of topological order under local perturbations, *Commun. Math. Phys.* **307**, 609 (2011).