Quantum Science and Technology

OPEN ACCESS

RECEIVED
12 January 2024

REVISED
26 April 2024

ACCEPTED FOR PUBLICATION
3 July 2024

PUBLISHED
17 July 2024

Original Content from this work may be used under the terms of the Creative Commons Attribution 4.0 licence.

Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.

PAPER

Lift-connected surface codes

Josias Old[1,2,*], Manuel Rispler[1,2] and Markus Müller[1,2]

[1] Institute for Quantum Information, RWTH Aachen University, Aachen, Germany
[2] Institute for Theoretical Nanoelectronics (PGI-2), Forschungszentrum Jülich, Jülich, Germany
[*] Author to whom any correspondence should be addressed.

E-mail: j.old@fz-juelich.de

Keywords: quantum LDPC codes, 3D local quantum codes, QEC codes for NISQ devices, surface code, quantum error correction, fault tolerance, lifted product construction

Abstract

We use the recently introduced lifted product to construct a family of quantum low density parity check codes (QLDPC codes). The codes we obtain can be viewed as stacks of surface codes that are interconnected, leading to the name lift-connected surface (LCS) codes. LCS codes offer a wide range of parameters—a particularly striking feature is that they show interesting properties that are favorable compared to the standard surface code. For example, already at moderate numbers of physical qubits in the order of tens, LCS codes of equal size have lower logical error rate or similarly, require fewer qubits for a fixed target logical error rate. We present and analyze the construction and provide numerical simulation results for the logical error rate under code capacity and phenomenological noise. These results show that LCS codes attain thresholds that are comparable to corresponding (non-connected) copies of surface codes, while the logical error rate can be orders of magnitude lower, even for representatives with the same parameters. This provides a code family showing the potential of modern product constructions at already small qubit numbers. Their amenability to 3D-local connectivity renders them particularly relevant for near-term implementations.

1. Introduction

Quantum error correcting (QEC) codes are essential for the reliable operation of quantum computers [1]. Recent advances in hardware quality and qubit count of quantum computing devices enabled first experiments realizing different aspects of quantum error correction [2–7].

QEC codes encode logical qubits in a subspace of a higher dimensional Hilbert space. For stabilizer codes, the codespace is spanned by the simultaneous $+1$-eigenspace of a set of commuting operators, the stabilizer generators. The commonly shown parameter triple $[[n,k,d]]$ denotes the number of physical qubits n employed by the error correcting code to encode k logical qubits with a minimum distance d. The latter is defined as the minimum number of single qubit Pauli operators that have non-trivial action on the codespace, i.e. the minimum weight of a logical operator. A code with distance d can correct at least for all errors up to weight $t=\lfloor\frac{d}{2}\rfloor$. A *code family* is specified by a sequence of stabilizer codes (that share most properties) with growing number of physical qubits. Promising QEC code families are *surface codes* [8]. Surface codes only require nearest neighbor connectivity in a planar 2D architecture. This makes them especially suited for experimental platforms with manifest connectivity constraints, such as superconducting qubits. Additionally, surface codes have some of the highest known thresholds for realistic circuit level noise models [9, 10].

Despite these strong upsides, a major shortcoming of surface codes is the observation that a surface code patch essentially always encodes only a single logical qubit irrespective of the size of the patch. This is captured by the so-called code rate $r=\frac{k}{n}$, i.e. the ratio of logical to physical qubit numbers, which is asymptotically zero for the surface code. In practical terms, this implies that scaling the code to improve its correction capabilities leads to a substantial qubit overhead. This in turn defines the challenge to find codes with better encoding rate while giving up as little as possible with respect to connectivity and logical

© 2024 The Author(s). Published by IOP Publishing Ltd

(threshold) performance. To highlight a result in this direction, it has been shown that codes with bounded connectivity attaining a constant encoding rate could be used for constant overhead fault-tolerant quantum computation [11]. Codes built from stabilizers with bounded degree of connectivity are called *quantum low-density parity check codes* (QLDPC). In particular, a (d_q, d_s)-QLDPC family of codes $\mathcal{Q}_i$ with $n_i \to \infty$ as $i \to \infty$ has every qubit involved in a maximum of d_q stabilizer measurements and every stabilizer measurement involving at most d_s qubits, independent of [12]. A (Q)LDPC code family with both an asymptotically constant rate and a linear distance scaling, i.e. $[[n, k \propto n, d \propto n]]$, is called *'good'*. For a recent overview of QLDPC codes see [13].

The definition of QLDPC captures the bounded connectivity found in the surface (or closely related toric) and color codes. However despite technically being QLDPC, these are far from 'good' both due to their vanishing rate and moreover their sub-optimal distance scaling $d \propto \sqrt{n}$. A big leap towards QLDPC codes with improved scalings was provided by the invention of the hypergraph product (HGP) construction [14]. Here, the product of two classical codes gives a quantum code, which remarkably preserves the (Q)LDPC property provided the two classical codes are LDPC to begin with. With this construction, the product of two good classical codes leads to a QLDPC code with constant rate while maintaining the distance scaling of the surface code ($d \propto \sqrt{n}$) [14, 15]. After a series of breakthroughs, this line of research of product code constructions recently culminated in the remarkable discovery that good quantum (LDPC) codes with constant rate and linear distance exist [16, 17]. Additional constructions of good QLDPC code families followed shortly after [18, 19]. While this showcases 'goodness' as an important guiding principle inspiring the search for better codes, a good but also practical code (family) with low enough number of qubits and embeddability so far remains elusive.

These results establish QLDPC codes as promising candidates for fault-tolerant quantum computation. By the same token, they define substantial challenges, both on the experimental and on the theory side. As a first challenge, known constructions of good codes involve prohibitively large numbers of qubits ($\sim 10^6$), leaving a substantial gap between what will be realistically available in near-term devices and what would be required for the above. As a second challenge, good QLDPC codes have been proven to require geometrically non-local connectivity. In fact, no-go theorems prevent good scaling of parameters if the connectivity of the code is restricted to some neighborhood that does not grow with the code [20, 21]. Several proposals on circuit constructions and implementation of QLDPC codes with constrained connectivity in mind have been formulated [22–24] and code constructions that explicitly leverage hardware capabilities like modularity are also considered [25]. Progress in platforms that allow for more connectivity opens the road to implement more advanced codes. Ion traps achieve all-to-all connectivity in single crystals mediated by motional modes, limited to a few tens of qubits [26]. This restriction can be overcome by the ability to shuttle ions [27]. Shuttling also enables effective all-to-all connectivity in neutral atom arrays, where coherent control of hundreds of atoms can be realized [28–33]. The potential for quantum error correction in neutral atom quantum processors has very recently been demonstrated in [7].

Further challenges from the conceptual side are logical gates and decoding. Decoders adapted from classical coding theory like belief propagation and ordered statistics decoding (BP+OSD) perform reasonably well, but symmetry of certain quantum codes and large distances pose ongoing challenges [34–37]. While efficient decoders for good QLDPC codes are in principle available, the lack of codes with reasonable size prevents benchmarking these codes [19, 38, 39]. Additionally, QLDPC codes with a property known as single-shotness allow for fault-tolerance with only a single round of (noisy) syndrome measurement, but at the cost of a large qubit overhead [40, 41].

Fault-tolerant implementations of logical gates like transversal, i.e. single qubit decomposable gates, in general rely on symmetries of the code [42–44]. Several approaches for general codes include teleportation based gates [45], generalized lattice surgery [46] or codes specifically constructed to support certain gates [47]. For HGP codes, some implementations of gates have been proposed, but they remain short of generality or practicality [48, 49].

1.1. Summary of results

In this work, we introduce a new QLDPC code family, which we call lift-connected surface (LCS) codes. For their construction, we employ the recently established lifted product (LP). This technique is a key ingredient in the recent groundbreaking discovery of good QLDPC codes. Using comparatively simple input codes, we obtain QLDPC codes that can be straightforwardly seen as sparsely interconnected copies of surface codes, leading to the name LCS. While their asymptotic scaling is not 'good' in the strict sense of the term, i.e. in a constant rate regime, the distance grows proportional to the physical qubit number up to a maximum size (see discussion around equation (28)), they demonstrate the near-term potential of QLDPC (specifically LP) codes. We benchmark LCS codes under code capacity as well as phenomenological noise (i.e. noisy syndrome

Table 1. Bit-flip noise thresholds of LCS codes compared to copies of surface codes, obtained from BP+OSD decoding of codes with increasing distance. For details of the families, refer to the text. All uncertainties here and in the following are readoff errors.

	Threshold	
Code	Code capacity	Phenomenological
LCS code family 1	$6.7 \pm 0.3\%$	$2.9 \pm 0.1\%$
LCS code family 2	$7.7 \pm 0.2\%$	$3.2 \pm 0.1\%$
d copies of distance d surface codes	$7.5 \pm 0.3\%$	$2.9 \pm 0.1\%$

measurements) using an adapted BP+OSD decoder. We find that their asymptotic thresholds are comparable to disjoint copies of surface codes, summarized in table 1. However for concrete realizations they offer substantially lower logical error rates and higher pseudo-thresholds. We show that this carries over to circuit-level noise by constructing distance preserving circuits for small LCS codes. Using only a fraction of the number of physical qubits, they achieve the same logical error rates as copies of surface codes. Given that these advantages already appear for as few as tens of qubits, these results make LCS codes promising candidates for near-term QEC experiments.

The manuscript is structured as follows. In section 2, we review the LP construction for QEC codes. In section 3, we show how LCS codes are constructed from the LP and describe the code parameters and structure. In section 4 we perform simulations over code capacity and phenomenological noise channels for several members of the LCS family, showing their error correction capabilities. Section 5 shows the construction and benchmarking of fault-tolerant syndrome readout circuits for representatives with small qubit numbers. Finally we outline a path towards logical gates in LCS codes in section 6 before concluding in section 7.

2. The Lifted Product construction

The LP construction combines classical LDPC codes based on circulant permutation matrices with the HGP construction for quantum codes [14, 50].

2.1. Hypergraph Product

A parity check matrix of a (classical code) H can be represented by the so-called Tanner graph by identifying its rows (the parity checks c) with one type of node and its columns (the (qu)-bits q) with another type of node. An edge between nodes is drawn whenever the corresponding entry H_{cq} is 1, making the graph manifestly bipartite [51]. Let $\mathcal{T}_{\mathcal{C}_1}$ and $\mathcal{T}_{\mathcal{C}_2}$ be the Tanner graphs of two classical codes $\mathcal{C}_1$ and $\mathcal{C}_2$ with binary parity check matrices $H_1 \in \mathbb{F}_2^{m_1 \times n_1}, H_2 \in \mathbb{F}_2^{m_2 \times n_2}$ respectively. We will sometimes refer to these as *base matrices*. The Tanner graph of the HGP (quantum) code $\mathcal{T}_Q$ is based on the Cartesian product of the classical Tanner graphs $\mathcal{T}_{\mathcal{C}_1}$ and $\mathcal{T}_{\mathcal{C}_2}$. A graphical construction is shown in figure 1, for details refer to [14]. Here, we review the algebraic construction rule when given two base matrices. The parity check matrices of the HGP quantum CSS code are given by

$$H = \mathrm{HGP}\left(H_1, H_2\right) = \begin{pmatrix} 0 & H_Z \\ H_X & 0 \end{pmatrix} \quad \text{with} \tag{1}$$

$$H_X = \left(\mathbf{1}_{n_1} \otimes H_2 \quad H_1^T \otimes \mathbf{1}_{m_2}\right) \tag{2}$$

$$H_Z = \left(H_1 \otimes \mathbf{1}_{n_2} \quad \mathbf{1}_{m_1} \otimes H_2^T\right). \tag{3}$$

The CSS commutativity constraint $H_Z H_X^T = 0$ is fulfilled by construction since

$$\begin{aligned} H_Z H_X^T &= \left(H_1 \otimes \mathbf{1}_{n_2} \quad \mathbf{1}_{m_1} \otimes H_2^T\right) \begin{pmatrix} \left(\mathbf{1}_{n_1} \otimes H_2\right)^T \\ \left(H_1^T \otimes \mathbf{1}_{m_2}\right)^T \end{pmatrix} \\ &= \left(H_1 \otimes \mathbf{1}_{n_2}\right)\left(\mathbf{1}_{n_1} \otimes H_2^T\right) + \left(\mathbf{1}_{m_1} \otimes H_2^T\right)\left(H_1 \otimes \mathbf{1}_{m_2}\right) \qquad (4) \\ &= H_1 \otimes H_2^T + H_1 \otimes H_2^T = 0. \qquad (5) \end{aligned}$$

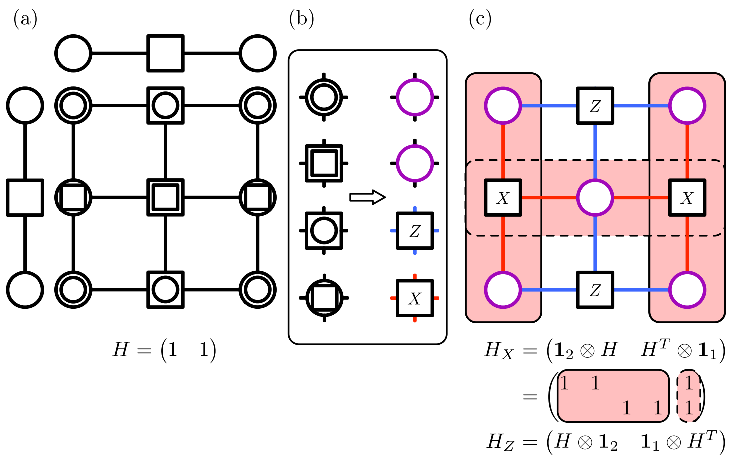
(a)
(b)
(c)
$H = (1 \quad 1)$
Z
X
$H_X = (\mathbf{1}_2 \otimes H \quad H^T \otimes \mathbf{1}_1)$
$= \begin{pmatrix} 1 & 1 & & & 1 \\ & & 1 & 1 & 1 \end{pmatrix}$
$H_Z = (H \otimes \mathbf{1}_2 \quad \mathbf{1}_1 \otimes H^T)$

Figure 1. Pictorial representation of the hypergraph product (HGP) construction. (a) Shows the Cartesian product of two repetition codes of length two with parity check matrix H. Black circles represent bits and black squares represent parity checks. With the prescription of (b), the Tanner graph (c) is obtained, which can be identified with the non-rotated $[[5,1,2]]$ surface code. Purple circles represent qubits, stabilizers are again black squares, where the edge color red (blue) denotes the Pauli-type X (Z). We furthermore highlight the X-part as a guide to the reader to identify the blocks of the algebraic construction. The vertical stripes (solid border) correspond to the left $\mathbf{1}_2 \otimes H$ part of H_X as two copies of the original repetition code. The horizontal stripe (dashed border) comes from the right $H^T \otimes \mathbf{1}_1$ block, corresponding to (one copy of) the Tanner graph of the transposed repetition code.

Any choice of binary matrices H_1, H_2 gives a valid quantum code. Notably, surface codes can be obtained from taking the HGP of the $\ell \times \ell + 1$ parity check matrices of (classical) repetition codes, i.e.

$$H_1^{(\ell)} = H_2^{(\ell)} = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 1 & 0 & \cdots & 0 \\ \vdots & & & & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 1 & 1 \end{pmatrix} \equiv H^{(\ell)}. \tag{6}$$

The resulting surface code $\mathrm{HGP}(H^{(\ell)}, H^{(\ell)})$ then has parameters $[[n,k,d]] = [[(\ell+1)^2 + \ell^2, 1, \ell+1]]$.

If the base matrices are members of a *good* classical LDPC code family with parameters $[n_{\mathrm{cl}}, k_{\mathrm{cl}} \propto n_{\mathrm{cl}}, d_{\mathrm{cl}} \propto n_{\mathrm{cl}}]$, then the resulting HGP codes have constant rate and distance $d = \Omega(\sqrt{n})$ [14].

2.2. Lifted Product

To present the LP construction, we will follow the approach by Panteleev and Kalachev [50]. For a complementary approach see [13]. We restrict ourselves to a subset of LP codes, originally referred to as quasi-cyclic generalized HGP codes [34]. We first briefly review this generalization, before carefully explaining the implications for our construction in the next section.

A useful starting point before attempting to go beyond the HGP is to note a subtle requirement in fulfilling the CSS commutativity constraint of equation (5). Given the two parity check matrices H_1 and H_2 and using Dirac notation, it is indeed true that

$$\left(H_1 \otimes \mathbf{1}_{n_2}\right)\left(\mathbf{1}_{n_1} \otimes H_2^T\right) \tag{7}$$

$$= \sum_{abgh} H_{1,ab} H_{2,gh}^T |ag\rangle\langle bh| \tag{8}$$

$$= H_1 \otimes H_2^T, \tag{9}$$

however when doing the analogous calculation

$$\left(\mathbf{1}_{m_1} \otimes H_2^T\right)\left(H_1 \otimes \mathbf{1}_{m_2}\right) = \sum_{cgaf} H_{2,cg}^T H_{1,af} |ac\rangle\langle fg|, \tag{10}$$

the conclusion that this is also $H_1 \otimes H_2^T$ rested on the assumption that all $H_{2,cg}^T$ and $H_{1,af}$ commute. While this commutativity is trivially true when the entries are numbers, it turns into a non-trivial requirement as soon as we want to promote the entries to higher-dimensional objects, e.g. matrices. In turn, this suggests that as long as we fulfill commutativity on this level, it will imply the fulfillment of the CSS constraint. One choice are elements of a commutative (matrix) ring, like circulant matrices of size $L \times L$. A circulant C can be represented as sums of cyclic permutations $P^{(i)}$,

$$C = \sum_{i=0}^{L-1} c_i P^{(i)} \tag{11}$$

where c_i are binary coefficients and $P^{(i)}$ denotes the ith cyclic (right) shift. We denote $P^{(0)}$ by I. For any circulant, we can give a binary representation such that $\mathcal{B}_L(P^{(i)})$ is the ith cyclic (right) shift of the identity matrix $\mathbf{1}_L$. For example

$$\mathcal{B}_4\left(P^{(3)}\right) = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \tag{12}$$

$$\mathcal{B}_3\left(I + P^{(1)}\right) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}, \tag{13}$$

$$\mathcal{B}_2\left[\begin{pmatrix} 0 & I + P^{(1)} \\ P^{(1)} & 0 \end{pmatrix}\right] = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}. \tag{14}$$

The LP construction then takes the HGP of two matrices with circulants as entries and replaces these with the corresponding binary representations after having taken the product. This increases the number of qubits and parity checks by a factor of L, which gives the procedure its name *lifting*. Denoting matrices with circulant entries with a tilde, an LP code is obtained as

$$H = \mathrm{LP}\left(\tilde{H}_1, \tilde{H}_2\right) = \mathcal{B}_L\left(\tilde{H}\right) \quad \text{with} \tag{15}$$

$$\tilde{H} = \begin{pmatrix} 0 & \tilde{H}_Z \\ \tilde{H}_X & 0 \end{pmatrix} \quad \text{with} \tag{16}$$

$$\tilde{H}_X = \begin{pmatrix} \mathbf{1}_{n_1} \otimes \tilde{H}_2 & \left(\tilde{H}_1\right)^T \otimes \mathbf{1}_{m_2} \end{pmatrix} \tag{17}$$

$$\tilde{H}_Z = \begin{pmatrix} \tilde{H}_1 \otimes \mathbf{1}_{n_2} & \mathbf{1}_{m_1} \otimes \left(\tilde{H}_2\right)^T \end{pmatrix}. \tag{18}$$

Note that the transpose of a matrix with circulant entries $\tilde{A} = (a_{ij})_{m\times n}$ is $\tilde{A}^T = (a_{ji}^T)_{n\times m}$ and it holds that $\mathcal{B}_L(\tilde{A}^T) = \mathcal{B}_L(\tilde{A})^T$.

In graph-theoretical terminology, the parity check matrices H_X and H_Z are the *biadjacency matrices* of the Tanner graphs. We show the lifting procedure for the example equation (13) in figure 2. One edge of a Tanner graph of a HGP code with parity check matrix H between check c_i and qubit q_j is shown in figure 2(a). In the LP construction, the entries of the final parity check matrix are replaced by circulants. This can be visualized by labeling the corresponding edge in the Tanner graph, as shown in figure 2(b). The lift, in figure 2(c) with lift parameter $L = 4$, translates to copying the check and qubit nodes L times and connecting them according to the non-zero entries of the circulant. In this example, the circulant $I + P^{(1)}$ connects every copy of the check (c_i, k) with two copies of qubits, (q_j, k) and $(q_j, k+1 \mod L)$.

3. The LCS codes

3.1. Construction and parameters

We can use the LP construction to algebraically build copies of the (non-rotated) surface code. To that end, we take base matrices of size $\ell \times (\ell+1)$ of the same repetition code form as in equation (6). The binary entries are replaced the trivial circulant ($P^{(0)} = I$) of size L and zero circulant. The resulting matrix with circulant entries is denoted by $\tilde{H}^{(\ell)}$. Then the LP quantum code $\mathrm{LP}_L(\tilde{H}^{(\ell)}, \tilde{H}^{(\ell)})$ consists of L disjoint copies of distance $d = \ell + 1$ surface codes. The parameters of the code are therefore

$$\begin{aligned} [[n,k,d]] &= \left[\left[\left((\ell+1)^2 + \ell^2\right)L, L, \ell+1\right]\right], \\ \left(d_q, d_s\right) &= (4,4). \end{aligned} \tag{19}$$

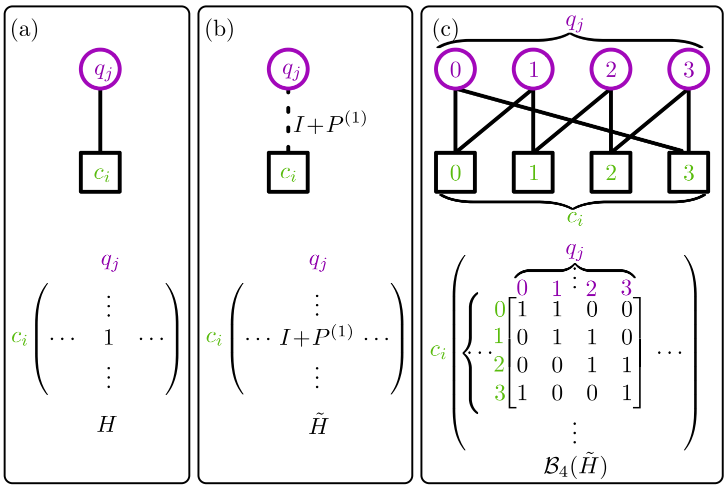

Figure 2. Lifting of edges of a graph that are part of the Tanner graph of a (quantum) code. (a) A nonzero entry of the parity check matrix H_{ij} corresponds to an edge between check c_i and qubit q_j. (b) A circulant entry like $I+P^{(1)}$ of a LP parity check matrix can be visualized by a label on the edge and a corresponding edge style. (c) The lift of the matrix corresponds to taking L (here $L=4$) copies of the check and qubit nodes. The new connectivity is according to the binary representation of the circulant. In this example, the circulant $I+P^{(1)}$ connects every copy of the check (c_i,k) with two copies of qubits, (q_j,k) and $(q_j,k+1 \mod L)$.

This insight can be used to construct interconnected surface codes. Consider matrices of size $\ell\times\ell+1$ with

$$\tilde{H}^{(\ell)} = \tilde{H}^{(\ell)}_{\text{rep.}} + \tilde{H}^{(\ell)}_{\text{int.}} \tag{20}$$

$$= \begin{pmatrix} I & I & 0 & \cdots & 0 & 0 \\ 0 & I & I & 0 & \cdots & 0 \\ \vdots & & & & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & I & I \end{pmatrix} + \begin{pmatrix} 0 & P^{(1)} & 0 & \cdots & 0 & 0 \\ 0 & 0 & P^{(1)} & 0 & \cdots & 0 \\ \vdots & & & & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 0 & P^{(1)} \end{pmatrix} \tag{21}$$

$$= \begin{pmatrix} I & I+P^{(1)} & 0 & \cdots & 0 & 0 \\ 0 & I & I+P^{(1)} & 0 & \cdots & 0 \\ \vdots & & & & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & I & I+P^{(1)} \end{pmatrix}. \tag{22}$$

The parity check matrices resulting from the first step (HGP) then naturally split into two parts,

$$\begin{aligned} \tilde{H}_X &= \left(\mathbf{1}_{\ell+1}\otimes\tilde{H}^{(\ell)}_{\text{rep.}} \quad \tilde{H}^{(\ell)T}_{\text{rep.}}\otimes\mathbf{1}_\ell\right) + \left(\mathbf{1}_{\ell+1}\otimes\tilde{H}^{(\ell)}_{\text{int.}} \quad \tilde{H}^{(\ell)T}_{\text{int.}}\otimes\mathbf{1}_\ell\right) \\ &=: \tilde{H}_{X,\text{surface}} + \tilde{H}_{X,\text{interconnection}} \end{aligned} \tag{23}$$

$$\begin{aligned} \tilde{H}_Z &= \left(\tilde{H}^{(\ell)}_{\text{rep.}}\otimes\mathbf{1}_{\ell+1} \quad \mathbf{1}_\ell\otimes\tilde{H}^{(\ell)T}_{\text{rep.}}\right) + \left(\tilde{H}^{(\ell)}_{\text{int.}}\otimes\mathbf{1}_{\ell+1} \quad \mathbf{1}_\ell\otimes\tilde{H}^{(\ell)T}_{\text{int.}}\right) \\ &=: \tilde{H}_{Z,\text{surface}} + \tilde{H}_{Z,\text{interconnection}}. \end{aligned} \tag{24}$$

The first addends $\tilde{H}_{X,\text{surface}}, \tilde{H}_{Z,\text{surface}}$ have the same structure as surface codes. The lift therefore generates L copies of surface codes. The second addends $\tilde{H}_{X,\text{interconnection}}, \tilde{H}_{Z,\text{interconnection}}$ act as additional connections in between the surface code patches. We therefore call the codes $\text{LP}_L(\tilde{H}^{(\ell)},\tilde{H}^{(\ell)})$ the (ℓ,L)-*lift-connected surface codes* or simply *LCS codes*. The interconnections amount to at most two additional connections per check, because $\tilde{H}^{(\ell)}_{\text{int.}}$ contains at most one entry per row and column. The general form of the parity-check matrices is shown in appendix A. The LCS code construction is shown pictorially in figure 3.

The LCS codes are therefore (6, 6)-QLDPC. The *dimension* of a quantum CSS code is given by

$$k = n - \text{rank}(H_X) - \text{rank}(H_Z). \tag{25}$$

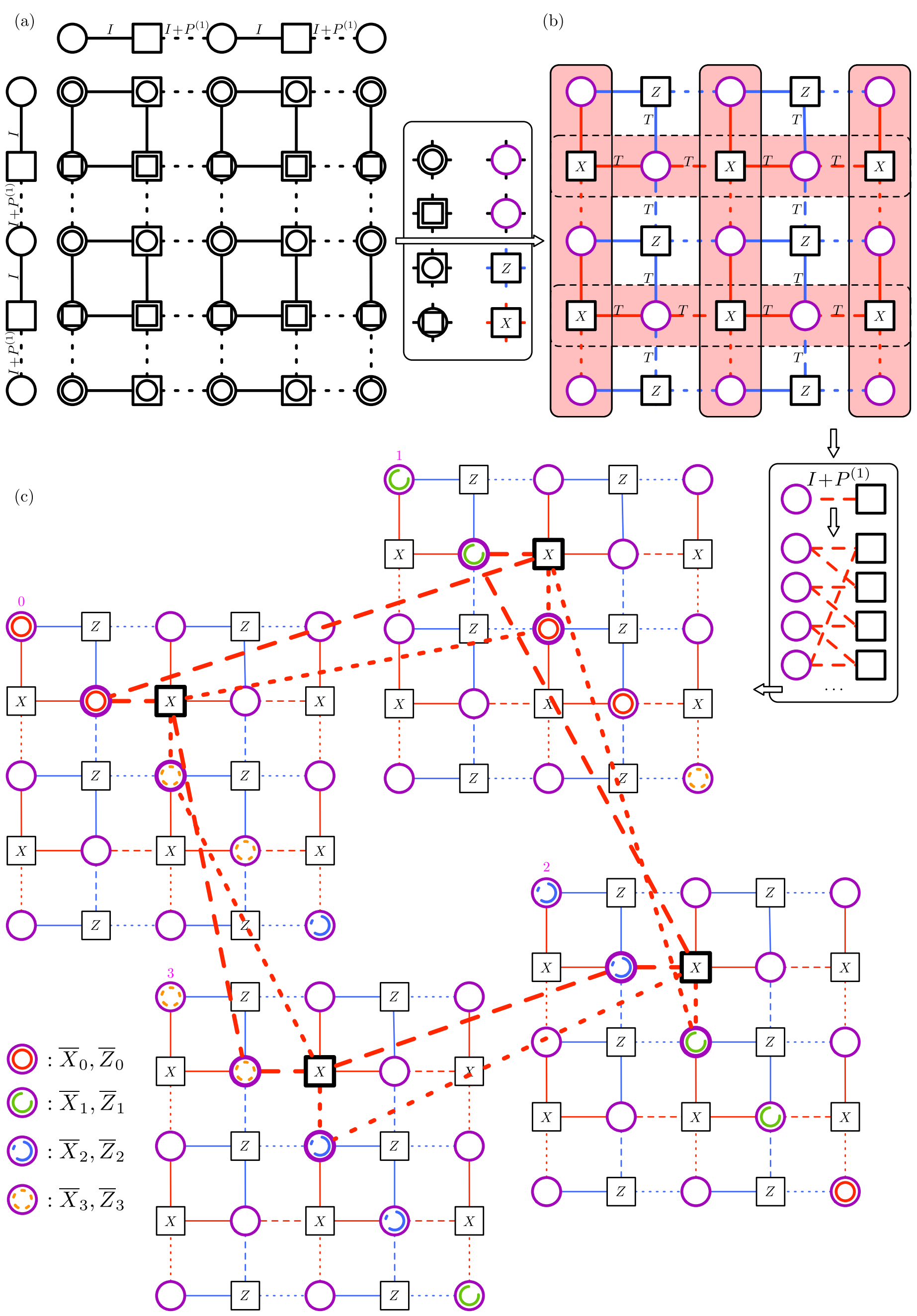
(a)
I
$I+P^{(1)}$
(b)
Z
X
T
(c)
$I+P^{(1)}$
0
1
2
3
: $\overline{X}_0, \overline{Z}_0$
: $\overline{X}_1, \overline{Z}_1$
: $\overline{X}_2, \overline{Z}_2$
: $\overline{X}_3, \overline{Z}_3$

Figure 3. Pictorial representation of the lifted product construction for LCS codes with $(\ell, L) = (2, 4)$. (a) First, we take the hypergraph product (HGP) of two repetition codes of length 3. The edges of the input Tanner graphs (top row and leftmost column) are now decorated and labeled to indicate that the scalar entries of the repetition code are being promoted (i.e. *lifted*) to matrices I (solid) and $I + P^{(1)}$ (dashed) respectively, which are both of dimension $L \times L$. This label carries through to the resulting Tanner graph. (b) Analogously to the HGP, vertices are respectively identified with qubits, X- and Z-checks. The edge label again indicates whether the corresponding entries are taken from the identity matrix I (solid) or the circulant $I + P^{(1)}$ (dashed). To obtain the stabilizers from this picture, first of all observe that since all edges contain the I matrix, we will obtain L copies of the code that we would have obtained without lifting. The non-trivial extension comes from the edges containing the additional term $P^{(1)}$, which define the additional entries in the resulting stabilizer checks between the different copies. Also note that the edges of the transposed base graphs (indicated by T) also have to be lifted by the transposed circulant. This is exemplarily spelled out in (c), where we show how the dashed edges of the given X-check in the LP are promoted to four X-checks on four copies of the underlying code and additional interconnections in the respective checks arise from the non-trivial circulants $P^{(1)}$ ($P^{(1)T}$). Note that the interconnections are highly structured, in particular they exclusively appear between neighboring code copies. We also indicate four logical representatives by small circles on the diagonal qubits of the surface code patches.

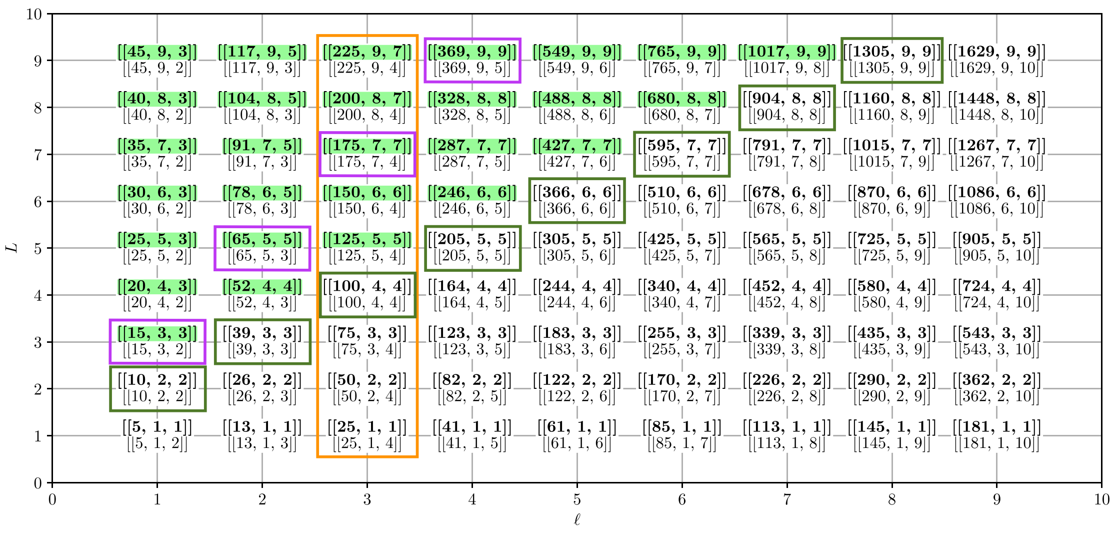

Figure 4. Code parameters $[[n,k,d]]$ of LCS (top, bold font) and copies of surface codes (bottom) for different (ℓ, L). The codes marked in dark green correspond to $L = \ell + 1$ where both surface and LCS codes have the same parameters (family 1). Above that line for fixed (ℓ, L), the LCS codes have a larger distance compared to the copies of surface codes, indicated by the green shade behind the corresponding parameters. The purple boxes mark LCS code family 2 where for a fixed rate, codes with maximum distance and minimum number of physical qubits are chosen. The orange box ($\ell = \text{const.}$) shows how for the LCS codes, k, d scale linearly in n up until $L = 2\ell + 1$.

Since both H_X and H_Z are already in a row echelon form and contain no zero rows, they are full rank and

$$k = \left((\ell+1)^2 + \ell^2\right) L - 2\ell(\ell+1) L = L. \tag{26}$$

There is no general recipe to get the distance of LP codes from the ingredients of the construction [50]. We can, however, calculate it via brute force by searching through all stabiliser equivalent representatives of logical operators or bound the minimum distance by probabilistic methods using the `GAP` package `QDistRnd` [52]. We find for parameters $(\ell = 1, L < 100), (\ell = 2, L < 10), (\ell = 3, L < 5)$ that

$$d = \min(L, 2\ell + 1). \tag{27}$$

All data shown in this manuscript support this conjecture. In appendix A, we show a constructive approach to establishing the minimum distance based on the block structure of the parity check matrices. There, we also show sets of logical operators that can be understood from the point of view of interconnected surface codes. In (regular) surface codes, there exist representatives of X- and Z-logical operators with the same support on the diagonal of the surface code patch. These have weight $2\ell + 1$ for surface code distance $\ell + 1$. While the canonical logical operators of surface codes do not 'survive' the lift trivially, the operators on the diagonal can be lifted using consecutive shifts of the identity. In figure 3, we indicate these logical operators, that have support on 2, 2 and 1 qubits of successive copies of surface codes. In summary, the parameters of the LCS codes are

$$\begin{aligned}[[n,k,d]] &= \left[\left[\left((\ell+1)^2 + \ell^2\right) L, L, \min(L, 2\ell+1)\right]\right], \\ (d_q, d_s) &= (6, 6).\end{aligned} \tag{28}$$

Available codes are shown in figure 4 and discussed in the following section. When unclear, we denote the construction parameters as a superscript $[[n,k,d]]^{(\ell,L)}$. Note that for fixed ℓ and varying lift-parameter $L \leqslant 2\ell + 1$, both the number of encoded logical qubits and the distance scale linearly in the number of physical qubits with constant rate $r = (2\ell^2 + 2\ell + 1)^{-1}$.

Asymptotically for $L \propto \ell$, the LCS codes achieve a scaling $[[n, \mathcal{O}(n^{\frac{1}{3}}), \mathcal{O}(n^{\frac{1}{3}})]]$. Compared to other 3D-local codes, this is better than 3D surface and toric codes with $[[n, \mathcal{O}(1), \mathcal{O}(n^{\frac{1}{3}})]]$ (or $[[n, \mathcal{O}(n^{\frac{1}{4}}), \mathcal{O}(n^{\frac{1}{4}})]]$ when considering $n^{\frac{1}{3}}$ copies of 3D surface/toric codes). Recent advances on 3D local codes from subdivisions or layer codes achieve the optimal scaling $[[n, \mathcal{O}(n^{\frac{1}{3}}), \mathcal{O}(n^{\frac{2}{3}})]]$ [20, 21, 53–55]. It is, however, not clear if and how small examples can be constructed and how they perform.

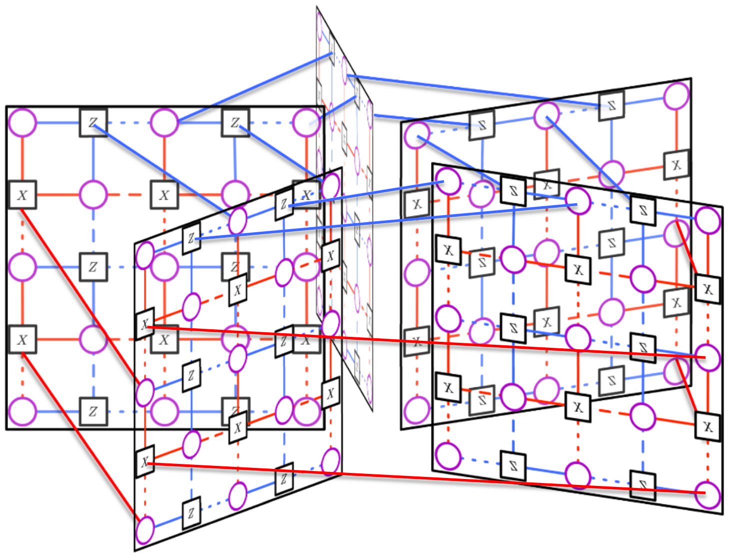

Figure 5. Possible arrangement of qubits of the $[[65,5,5]]^{(2,5)}$ LCS code, corresponding to 5 interconnected $[[13,1,3]]$ surface codes. While only a few interconnections of the patches are shown, note that all connectivity is restricted to neighbors of neighboring patches and therefore 3D-local.

3.2. Connectivity

Firstly, the LCS codes are QLDPC with $(d_q,d_s)_{\text{lcs}}=(6,6)$ being slightly larger than the surface codes' $(d_q,d_s)_{\text{s}}=(4,4)$. The additional connectivity however is limited and indeed 3D-local. We label qubits with the tuple (q_i,s) where qubit $q_i, i\in\{0,1,\dots,(\ell+1)^2+\ell^2-1\}$ of patch $s\in\{0,1,\dots L-1\}$ (and checks correspondingly with (c_i,s)). An edge then is the tuple $e=[(c_i,s),(q_j,s')]$. The circulant $P^{(1)}$ in the interconnection part of the parity check matrices interconnects only neighboring patches s and $s+1 \mod L$. The transposed circulant $P^{(1)T}=P^{(L-1)}$ turns the cyclic right shift by one into a cyclic left shift by one, keeping the restricted connectivity by only interconnecting neighboring patches s and $s-1 \mod L$. Hence, there only exist edges $e=[(c_i,s),(q_j,s')]$ with $s'=s+1 \mod L$. Also, for every edge $e=[(c_i,s),(q_j,s')]$ with $s\neq s'$ there exists an edge $e=[(c_i,s),(q_j,s)]$, i.e. interconnection only involve checks and qubits that would have been connected within one patch. An example for $(\ell,L)=(2,5)$ is shown in figure 5, where the underlying surface codes are arranged as slices of a torus. Only a few interconnections are shown, but these have a bounded length. These observations imply that these codes might be suitable, e.g. for implementation in static 3D Rydberg atom array structures [56], without the need for shuttling. This only holds if the qubits at the outer edge are close enough to the respective ancillary qubit of the neighboring slice to perform entangling gates for stabilizer measurements. Concrete implementations and optimizations of qubit positions in 3D space are left as future work.

Note that the choice of taking $P^{(1)}$ on every entry of $\tilde{H}^{(\ell)}_{\text{int.}}$ is motivated by the restricted connectivity. Taking $P^{(j)}$ with $j\neq 1$ and $j\neq\frac{L}{2}$ constructs equivalent codes up to permuting the surface code patches accordingly. For $j=\frac{L}{2}$, $P^{(j)T}=P^{(j)}$ and both additional connections of a check go to the same copy of the underlying surface code. In that case, the code decouples into $\frac{L}{2}$ disjoint copies and has distance $d=2$. Putting different $P^{(j)}$ matrices as entries of $\tilde{H}^{(\ell)}_{\text{int.}}$ renders the codes non-local, which makes it less straightforward to analyze. However, heuristically we find that such codes typically have the same parameters compared to their 3D local partners with the same (ℓ,L).

4. Code performance over noisy quantum channels

4.1. Sampling codes in quantum channels

Here we discuss several standard methods for (Monte Carlo) sampling of the combination of a given decoder and a noise model. We consider two types of noise models:

- *Code capacity* model: only the data qubits are affected by a single qubit i.i.d. noisy quantum channel (e.g. a bit-flip channel). The syndrome measurement is assumed to be perfect, see also algorithm 3.
- *Phenomenological noise* model: this noise model extends the code capacity model by additionally including noisy syndrome measurements. This can e.g. be modeled by a perfect measurement followed by a (classical) bit-flip channel on the result, see algorithm 4.

Typically, to ensure a fault-tolerance level of $t = \lfloor \frac{d}{2} \rfloor$, the number of noisy syndrome measurement cycles is chosen to be $n_{\mathrm{sc}} = d$. This is based on the fact that, even if the t errors occur on measurements of the same syndrome in the measurements cycles, the $d = 2t + 1$ repetitions allow to correctly identify these errors. We can also think of this as defining repetition codes on the syndrome measurement outcomes, with extra variables introduced to model syndrome flips. In order to preserve the distance d of the code, the length of these repetition codes is chosen to be d. This procedure is also shown in figure 6 for a three-bit repetition code.

4.2. Decoding

A decoder takes syndrome data and potentially parameters of the noise model as input and returns an inference of the underlying error configuration (which in turn determines the appropriate recovery operation), that is consistent with the observed syndrome. Based on this error guess, a correction is applied that puts the state back to the codespace. The ideal maximum likelihood decoder picks a guess from the most likely error (MLE) class, taking into account the code degeneracy, i.e. that distinct error configurations can be logically equivalent. Because this is in general computationally hard, practical decoders return an approximation, trading computational efficiency for non-optimality of the suggested recovery operation. E.g. the *MLE* decoder tries to determine the MLE configuration, ignoring potential degeneracies. To be efficient, practical decoders generally try to exploit structure in the code and noise model, such that not every existing decoder is suitable for application to a general code. Notably, a matching-based decoder like *minimum weight perfect matching* works well whenever elementary errors violate two parity checks [57]. This is not the case for LCS codes, since here a single error on a qubit violates up to 3 parity checks (considering only X- and Z- errors). We therefore resort to two general purpose decoders, namely a *MLE* decoder and a decoder based on *Belief Propagation* and *Ordered Statistics Decoding (BP+OSD)* [34, 58].

Given our codes are symmetric with respect to X and Z, we consider one Pauli type only, e.g. (single qubit, i.i.d) bit-flips with probability p, for a performance benchmark of the codes. The probability of a fixed configuration $E = \{E_q\}_{q=0}^{n-1}$ of errors given an observed syndrome $\mathbf{s}$ is

$$\begin{aligned} p(E|\mathbf{s}) &\propto \prod_{q=0}^{n-1} p(E_q) \prod_{c=0}^{n_c-1} \delta(\langle E, S_c\rangle = s_c) && (29)\\ &= p^w (1-p)^{n-w} && (30)\\ &\propto \left(\frac{p}{1-p}\right)^w, && (31) \end{aligned}$$

where w is the weight of the configuration $w = |E|$. $S_c \in \mathcal{S}$ denotes the stabilizer c and s_c the bit c of the observed syndrome $\mathbf{s}$. We write $\langle P, P'\rangle$ with $P, P' \in \mathcal{P}_n$ for the function that indicates commutation,

$$\langle P, P'\rangle = \begin{cases} 0 & \text{if} \quad [P, P'] = 0 \\ 1 & \text{if} \quad \{P, P'\} = 0. \end{cases} \qquad (32)$$

With that notation, the syndrome of error E can be written as

$$\sigma(E) = \left(\langle E, S_c\rangle\right)_{c=0}^{n_c-1}. \qquad (33)$$

4.2.1. MLE Decoding with i.i.d. noise

For $p < 0.5$ ($p > 0.5$), the MLE is the one with the lowest (highest) weight w. Landahl *et al* give an intuitive implementation of an MLE decoder for quantum codes which we follow here [59].

4.2.1.1. Code capacity noise

Considering pure bit-flip noise, we have one (error free) syndrome $\mathbf{s}_Z = \{s_{Z,c}\}_{c=0}^{n_c-1}$. Let $\mathbf{x} = \{x_q\}_{q=0}^{n-1}$ be the binary representation of the Pauli error. Then MLE decoding can be stated as the optimization problem

$$\begin{aligned} &\min \sum_q x_q && (34)\\ &\text{subject to} \bigoplus_{q\in\Gamma(c)} x_q = s_{Z,c} \qquad \forall c && (35)\\ &\text{with} \quad x_q \in \mathrm{GF}(2) = \{0,1\}. && (36) \end{aligned}$$

Here, $\Gamma(c)$ is the set of qubits involved in stabilizer measurement c and $\oplus$ indicates addition modulo 2. In words, this procedure states: minimize the weight of an error that is compatible with the syndrome. In vectorised form with Z-parity check matrix H_Z, this can be written as

$$\min \quad \mathbf{1}^T\mathbf{x} \tag{37}$$
$$\text{subject to } H_Z\mathbf{x} = \mathbf{s} \mod 2 \tag{38}$$
$$\text{with} \quad \mathbf{x} \in \mathrm{GF}(2)^n = \{0,1\}^n. \tag{39}$$

4.2.1.2. Phenomenological noise

With multiple rounds of noisy syndrome measurements, data errors after time step t are indicated by differences of syndrome bits from t to $t+1$. We write

$$\Delta\mathbf{s}_t = \mathbf{s}_t - \mathbf{s}_{t-1} = \mathbf{s}_t + \mathbf{s}_{t-1} \mod 2 \quad \forall t \tag{40}$$

with $\mathbf{s}_{t=0} := \mathbf{0}$ for the observed syndrome measurement outcomes. We introduce d data-error vectors $\mathbf{x}_0, \ldots \mathbf{x}_{d-1} \in \mathrm{GF}(2)^n$ and d syndrome measurement-error vectors $\tilde{\mathbf{s}}_0, \ldots \tilde{\mathbf{s}}_{d-1} \in \mathrm{GF}(2)^{n_c}$ for d noisy rounds of error correction. This allows one to formulate the MLE decoding optimization problem as

$$\min \sum_t \mathbf{1}^T\mathbf{x}_t \tag{41}$$
$$\text{subject to } H\mathbf{x}_t + \tilde{\mathbf{s}}_t + \tilde{\mathbf{s}}_{t-1} = \Delta\mathbf{s}_t \mod 2 \quad \forall t, \tag{42}$$

requiring that the noise-less syndrome together with syndrome errors are compatible with the observed syndrome differences. A last round of perfect syndrome measurements allows us to verify the correction returned by the optimization (see appendix 4). Vectorized, this reads

$$\min \quad \mathbf{1}^T\mathbf{y} \tag{43}$$
$$\text{subject to } A\mathbf{y} = \Delta\mathbf{s} \mod 2 \tag{44}$$
$$\text{with} \quad \mathbf{y} = \left(\mathbf{x}_0^T, \ldots, \mathbf{x}_{d-1}^T, \tilde{\mathbf{s}}_0^T, \ldots \tilde{\mathbf{s}}_{d-1}^T\right) \in \mathrm{GF}(2)^{2dn}. \tag{45}$$

Here

$$A = \left(\begin{array}{cccc|cccc} H & & & & I & & & \\ & H & & & I & I & & \\ & & \ddots & & & & \ddots & \\ & & & H & & & I & I \end{array}\right). \tag{46}$$

We implement the MLE decoder in python using the `optlang` interface [60] to the GNU Linear Programming Kit `GLPK` [61] to solve the constrained minimization. The run-time of decoding is exponential in the number of qubits and (noisy) error correction rounds. In practice, this results in a limitation on the feasibility of simulating the decoding of codes. We restrict the simulations to codes with $nd \leqslant 250$.

4.2.2. BP+OSD

To enable benchmarking beyond low qubit numbers, we employ BP+OSD. BP is a well known decoder for classical codes, where it works particularly well for good LDPC expander codes [62, 63]. BP calculates single-qubit error probabilities using statistical inference on the *Tanner graph* of the code. While BP can be shown to be exact on trees, Tanner graphs usually contain cycles, which can hinder the performance of BP (see e.g. [58, 64]). One way to overcome these problems is to use OSD as a post-processor. While BP may be inconclusive in its output, it usually points to a subset of likely erroneous qubits, OSD then brute-forces the solution of the decoding problem on that subset [34].

We denote the qubits participating in syndrome measurement c by $\Gamma(c) \subset Q$ with Q the set of all qubits. The syndrome measurements in which qubit q participates are $\Gamma(q)$. To approximate single qubit marginal probabilities, two types of quantities, termed *messages*, are updated iteratively until convergence or a maximum number of iterations is reached. The messages $m_{q\to c}(E_q)$ from qubit q to check c correspond to the current estimate of error probabilities for E_q on qubit q given the estimates of all neighboring checks except c,

$$m^{(i+1)}_{q\to c} \propto p_0\left(E_q\right) \prod_{c' \in \Gamma(q)\backslash c} m^{(i)}_{c'\to q}\left(E_q\right). \tag{47}$$

The messages $m_{c\to q}(E_q)$ from check c to qubit q collect incoming probability estimates of qubits except q and sum over all compatible configurations fixing the error E_q,

$$m_{c\to q}^{(i)} \propto \sum_{E_{\Gamma(c)\setminus q}} \delta\left[\sigma(E)_c = s_c\right] \prod_{q'\in\Gamma(c)\setminus q} m_{q'\to c}^{(i)}\left(E_{q'}\right). \tag{48}$$

The product of all these messages incoming at a qubit node gives an estimate for the marginal probabilities of errors on that qubit, called *belief*,

$$b_q\left(E_q\right) \propto p_0\left(E_q\right) \prod_{c\in\Gamma(q)} m_{c\to q}^{(i_{\max})}\left(E_q\right). \tag{49}$$

The log-likelihood ratios

$$l_q = \log\frac{1-b_q(E)}{b_q(E)} \tag{50}$$

of qubit q are calculated and used to sort the qubits by their likelihood of being erroneous. In first order OSD, the parity check matrix H is truncated to a full rank square matrix $H_{[R]}$, with $R\subseteq Q$ a set of qubits which are most likely to have an error based on the log-likelihood ratios. Here $H_{[R]}$ refers to a restriction of H to the columns specified in the set R. The OSD decoder then assumes no error on the set of remaining qubits $Q\setminus R$ and inverts the truncated matrix to get as a total error guess (up to reordering of the qubits)

$$\mathbf{x} = \left(H_{[R]}^{-1}\mathbf{s}, \mathbf{0}_{[Q\setminus R]}\right). \tag{51}$$

Higher-order OSD improves upon that by considering non-zero configurations $\mathbf{x}_{[Q\setminus R]}$ for the remaining qubits and adapting the reliable part to ensure validity as

$$\mathbf{x} = \left(H_{[R]}^{-1}\mathbf{s} + H_{[R]}^{-1}H_{[Q\setminus R]}\mathbf{x}_{[Q\setminus R]}, \mathbf{x}_{[Q\setminus R]}\right). \tag{52}$$

BP+OSD provides a general purpose decoder that is efficient enough for reasonable benchmarking and has a well tested implementation [35, 65, 66]. In appendix B, we provide details on chosen parameters. For small qubit numbers, it is typically observed that the performance of BP+OSD comes close to most-likely error decoding, see appendix C for a comparison.

As BP+OSD essentially provides a solution $\mathbf{x}$ to $H\mathbf{x} = \mathbf{s} \mod 2$, we can also use the same algorithm to find a solution to $A\mathbf{y} = \Delta\mathbf{s} \mod 2$ (equation (44)) representing the phenomenological noise model. In the graphical picture, this corresponds to taking d copies of the code's Tanner graph, adding new variable nodes for every check and interconnecting them according to the right part of the matrix A. An example of the adapted Tanner graph for a three-bit repetition code using three rounds of noisy measurements is shown in figure 6. Note that also here, we simulate a final destructive single qubit readout by a round of noiseless syndrome measurement.

4.3. Simulation results

In the following section, we present results we obtain from sampling LCS codes for code capacity and phenomenological noise channels, which we compare to the performance of the standard surface code. Because LCS and surface codes are both CSS and X- and Z-stabilizers are symmetric up to permutations, it suffices to focus on pure bit-flip noise. We start with a discussion of the results for the smallest distance instances of LCS codes. We then move to larger codes, where we first discuss how to compare different finite rate codes and then present the core results regarding logical performance and thresholds of LCS codes.

4.3.1. Logical error rate and pseudo-threshold

As we will be considering codes with more than one logical qubit, we will declare logical failure whenever any of the constituent logical qubits has an error. The logical error rate is given by

$$p_{\mathrm{ler}} = \sum_{w=t+1}^{n} N_{\mathrm{ler}}(w)\, p^w (1-p)^{n-w} \tag{53}$$

$$= (1-p)^n \sum_{w=t+1}^{n} N_{\mathrm{ler}}(w)\left(\frac{p}{1-p}\right)^w, \tag{54}$$

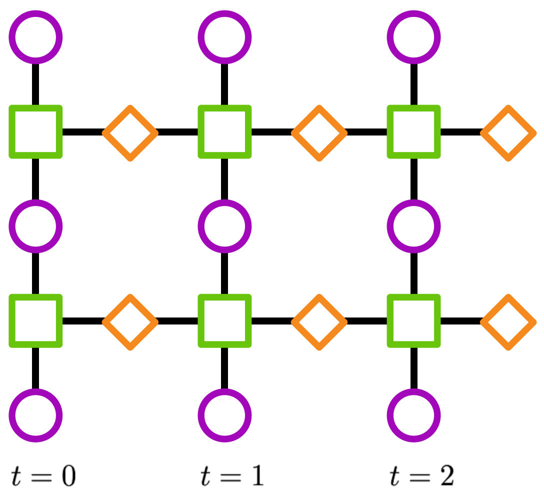

Figure 6. Phenomenological noise Tanner graph for a three-bit repetition code using three rounds of noisy measurements. Newly introduced syndrome-error variables $\tilde{\mathbf{s}}$ are represented by orange rhombi. Note that this Tanner graph is (up to the rightmost rhombi) equivalent to the Tanner graph of a distance 3 surface code.

where $N_{\mathrm{ler}}(w)$ is the number of failure configurations of weight w and $t=\lfloor\frac{d}{2}\rfloor$ is the minimum number of correctable errors. Let us start by looking at small LCS codes with $\ell=1$, $L\in\{3,4,5\}$ and parameters $[[15,3,3]]$, $[[20,4,3]]$ and $[[25,5,3]]$ respectively with a rate of $r=1/5$. The first two codes are also shown in figure 14. The logical error rate against the physical error rate is shown in figure 8(a).

We make the following observations about the given distance 3 codes: first of all, the different codes show a similar logical error rate across the range of physical error rates, where the logical error rate is larger for larger physical qubit number, as expected when fixing the distance. Furthermore the logical error rate scaling at low p is consistent with $p_{\mathrm{ler}}\propto p^2$, indicating that arbitrary single qubit errors are corrected. Remarkably, 3 copies of distance 3 surface codes with parameters $[[39,3,3]]$ have a logical error rate comparable to the $[[20,4,3]]$ LCS code encoding one logical qubit more with fewer physical qubits.

To further assess the performance of the error correcting codes, we can look at the physically motivated *pseudo-threshold* $p^{\star}_{\mathrm{th}}$ [67]. It is the value of physical error rate p below which the logical error rate is lower than the physical error rate. In general, we call the presence of any faulty (logical) qubit in our computational (logical) Hilbert space a (logical) failure. This implies that for k bare physical qubits each failing with p, their total failure probability is given by

$$p_{\mathrm{fail}}(k)=\sum_{l=1}^{k}\binom{k}{l}(1-p)^{k-l}p^{l} \tag{55}$$

$$=1-(1-p)^{k}. \tag{56}$$

This reduces to the well known case for $k=1$, $p_{\mathrm{fail}}=p$ that is often considered for (planar) color or surface codes hosting one logical qubit.

As shown in figure 8(a), The pseudo-thresholds are in the range of 8%–9%.

4.3.2. Logical error rate per logical qubit

In a practical setting, we cannot distinguish which of the logical qubits has a logical error and therefore, any error is bad. Additionally, a rescaling of the logical error rate to a *logical error rate per logical qubit* in order to e.g. compare codes with a different number of logical qubits k is only meaningful if they fail independently of each other. Figure 7 shows the probability of logical errors in the $[[15,3,3]]$ code for a bit-flip error rate $p_{\mathrm{x}}=0.01$. Most notably, $p(\overline{X}_0\overline{X}_1\overline{X}_2)\neq p(\overline{X}_0)p(\overline{X}_1)p(\overline{X}_2)$ indicates that the logical qubits do not fail independently. This is in contrast to using multiple codes each encoding only a single logical qubit. We will therefore always consider the logical failure of the block as soon as any logical qubit comprising it fails. Whenever we compare to surface codes with 1 logical qubit, we obtain the logical error rate for k copies of surface codes by rescaling the single-logical qubit error rate as

$$p_{\mathrm{L}}^{(k)}=1-\left(1-p_{\mathrm{L}}^{(1)}\right)^{k}. \tag{57}$$

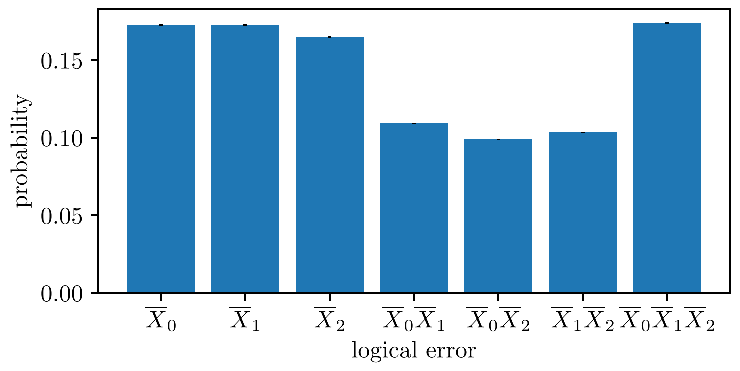

Figure 7. Probability of logical errors at $p_x = 0.01$ for the [[15, 3, 3]] LCS code. The logical qubits do not fail independently as $p(\overline{X}_i)p(\overline{X}_j) \neq p(\overline{X}_i\overline{X}_j)$. Because of the symmetry of the code and the noise, we expect the same logical error rates for e.g. $\overline{X}_0$ and $\overline{X}_2$. We attribute the difference seen above to the specific implementation of the decoder that returns only one solution whenever there are multiple with the same likelihood.

Table 2. Number of low weight failure configurations of three copies of distance 3 surface codes compared to the [[39, 3, 3]]-LCS code. Up to and including weight $w = 5$, the interconnected LCS codes have fewer failure configurations, explaining the lower logical error rates.

Weight w	$3 \cdot$ [[13, 1, 3]] Surface	[[39, 3, 3]] LCS
2	69	12
3	2253	852
4	34 152	23 093
5	321 690	307 976
6	2 163 638	2 378 071

Table 3. Pseudo-thresholds of small LCS codes, compared to three copies of surface codes. Note that the LCS codes have 26% − 40% higher pseudo-thresholds than surface codes for code capacity and 40%–56% higher pseudo-thresholds compared to surface codes with phenomenological noise.

	Pseudo-threshold	
Code	code capacity	phenomenological
[[15, 3, 3]] LCS	$8.1 \pm 0.1\%$	$4.3 \pm 0.1\%$
[[20, 4, 3]] LCS	$8.6 \pm 0.1\%$	$4.2 \pm 0.1\%$
[[25, 5, 3]] LCS	$8.4 \pm 0.1\%$	$4.2 \pm 0.1\%$
[[39, 3, 3]] LCS	$8.7 \pm 0.1\%$	$4.6 \pm 0.1\%$
[[39, 3, 3]] surface	$6.4 \pm 0.1\%$	$3.0 \pm 0.1\%$
[[75, 3, 3]] LCS	$9.0 \pm 0.1\%$	$4.7 \pm 0.1\%$

4.3.3. Copies of single-logical qubit codes or single-block codes?

In the following, we will discuss and compare logical error rates of various LCS and surface codes encoding the same number of logical qubits $k = 3$ and distance $d = 3$. This corresponds to $\ell \in \{1, 2, 3\}$, $L = 3$. Figure 8(b) shows the logical error rate of these codes sampled over a code capacity channel and decoded with the MLE decoder. For the interconnected [[39, 3, 3]] LCS codes, the logical error rate is almost an order of magnitude lower for small physical bit-flip rates p_x compared to three copies of [[13, 1, 3]] surface codes. To explain this remember that at low p the logical error rate is dominated by low-weight failure configurations. Using a look-up table decoder, we count the number of failure configurations, shown in table 2 for a weight $w \leqslant 5$. The number of failure configurations is significantly lower for the LCS code encoding the 3 logical qubits in one block which results in the significantly lower logical error rate. Also the pseudo-threshold is significantly higher with $p^{\star}_{\text{th}}([[39,3,3]] - \text{LCS}) \approx 8.6\%$ against $p^{\star}_{\text{th}}(3 \cdot [[13,1,3]] - \text{S}) \approx 6.4\%$. Note that the codes with smaller encoding rate $r = \frac{k}{n}$ show a lower logical error rate, which is plausible given their correspondingly higher number of stabilizers. However, the [[75, 3, 3]]-LCS code only has a marginally lower logical error rate compared to the [[39, 3, 3]]-LCS code, hinting at a tradeoff between physical qubit number and degeneracy. Table 3 summarizes the pseudo-thresholds for these small codes, which are all in the range of $8 - 9\%$.

For phenomenological noise, we set the probability of a noisy syndrome measurement to $q = p_x =: p$. Note that slightly different notions of *error correction* and *syndrome measurement cycles* are in use in

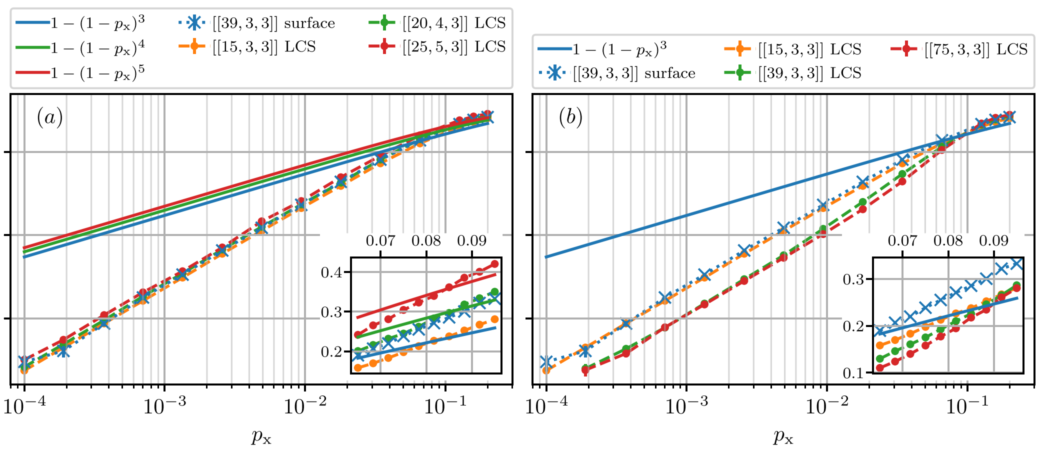

Figure 8. Logical error rates of small LCS and surface codes, sampled over a bit-flip channel and decoded using the MLE decoder (section 4.2.1). (a) Codes with $\ell = 1$ and $L = 2, 3, 4$, compared to error probabilities of 3, 4 and 5 physical qubits (solid lines), respectively. The smallest distance 3 code has the lowest logical error rate. The curves cross the physical qubit curves at the pseudo-threshold $p^{\star}_{\text{th}} \approx 8.6\%$. (b) Codes with $\ell = 1, 2, 3$ and $L = 3$, compared to error probabilities of 3 physical qubits and 3 copies of [[13, 1, 3]] surface codes. Interconnected codes can achieve the same logical error rate as surface codes using less physical qubits (here 15 vs. 39). Using more physical qubits increasing the redundancy gives an improvement of almost one order of magnitude for low physical error probabilities. The pseudo-threshold is also increased from $p^{\star}_{\text{th}}(3 \cdot [[13, 1, 3]] - \text{S}) \approx 6\%$ to $p^{\star}_{\text{th}}([[39, 3, 3]] - \text{LCS}) \approx 8.6\%$. In this plot and all following, statistical error bars corresponding to one standard deviation in the Monte Carlo sampling are drawn (occasionally smaller than markers).

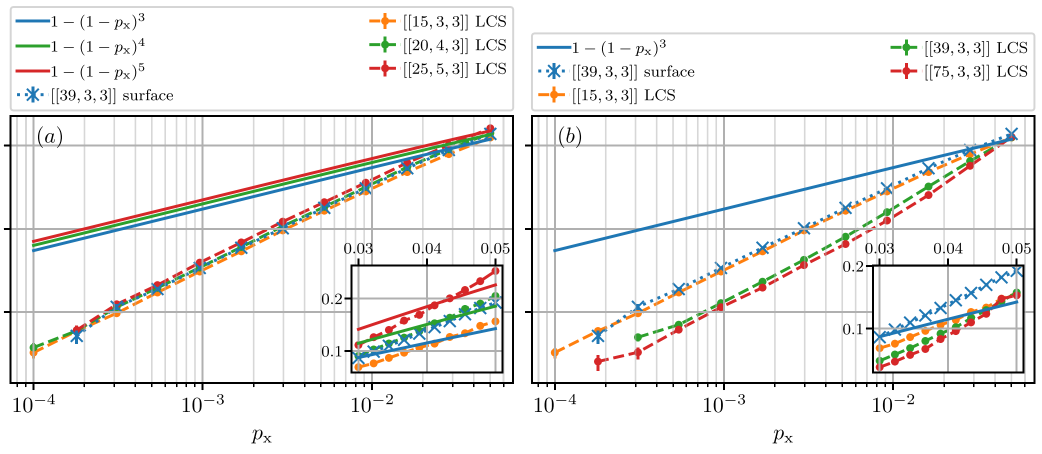

Figure 9. Logical error rates of small LCS and surface codes, sampled over a bit-flip channel with d noisy syndrome measurements and decoded using the MLE decoder (section 4.2.1). (a) Codes with $\ell = 1$ and $L = 2, 3, 4$, compared to error probabilities of 3, 4 and 5 physical qubits, respectively. The smallest distance 3 code has the lowest logical error rate. The curves cross the physical qubit curves at the pseudo-threshold $p^{\star}_{\text{th}} \approx 4.2\%$. (b) Codes with $\ell = 1, 2, 3$ and $L = 3$, compared to error probabilities of 3 physical qubits and 3 copies of [[13, 1, 3]] surface codes. Interconnected codes can achieve the same logical error rate as surface codes using less physical qubits (here 15 vs. 39). Using more physical qubits increasing the redundancy gives an improvement of almost one order of magnitude for low physical error probabilities. The pseudo-threshold is also increased from $p^{\star}_{\text{th}}(3 \cdot [[13, 1, 3]] - \text{S}) \approx 3\%$ to $p^{\star}_{\text{th}}([[39, 3, 3]] - \text{LCS}) \approx 4.6\%$.

literature. We follow [6, 24] and report a lower bound on the *logical error rate per syndrome measurement cycle* by rescaling the total logical error rate after n_{sc} cycles as

$$p_{\text{L}}(p) \geqslant 1 - (1 - p_{\text{L}}(n_{\text{sc}}))^{\frac{1}{n_{\text{sc}}}}. \tag{58}$$

This can then be compared to the failure probability of k unencoded qubits to get the pseudo-threshold as a solution to $p_{\text{L}}(p) = 1 - (1 - p)^k$, shown for the MLE decoder and distance 3 bounded LCS codes in figure 9. The pseudo-thresholds are summarized in table 3 and range between 4 and 5%, higher than the pseudo-threshold of 3 copies of distance $d = 3$ surface codes of ≈3%. Also note that the quadratic scaling in the low error rate regime is consistent with the ability to correct for any order p error, data qubit or measurement error.

4.3.4. Code families and their thresholds

Beyond pseudo-thresholds of specific codes, the asymptotic *threshold* of families of code is of interest, in particular in order to compare the performance of different code families.

It is important to note that the existence of a threshold has been proven for concatenated codes [1, 68], codes with an asymptotically constant encoding rate [11, 69] and topological codes with rate vanishing $r \propto n^{-1}$ [8]. For general QLDPC codes, the existence of and lower bounds on a threshold were proven subsequently [70]. The threshold of a code family under a specific noise model and decoder is typically determined by Monte Carlo sampling and plotting the logical error rate versus the physical error rate for representatives of the code family with growing number of physical qubits and code distance. Heuristically, a crossing of these curves at a point $p = p^\star$ indicates the existence of a threshold at $p^\star$. This is supported by arguments from statistical mechanics mappings of error correcting codes and is particularly successful for topological codes with well defined scaling properties [71–73]. Hence for independent copies of surface codes we can expect the same asymptotic threshold as the copies fail independently. Results for $k = 1$ logical qubits that are rescaled according to equation (57) however have finite size crossing points of the curves that deviate from the asymptotic threshold. For general QLDPC codes, the statistical mechanical picture is more intricate and far less well understood and part of ongoing research [74, 75]. Nevertheless, crossing points of such curves give an estimate on the value of the threshold. In the following, crossing points are obtained from reading off the crossings of quadratic least square fits to the data points. Indicated uncertainties correspond to estimates of how well we can resolve the respective crossing point.

From all available code parameters when sweeping ℓ, L (figure 4) we can define code families which are suitable for comparative simulations of error correcting properties. In the following, we will define three relevant scenarios and report the numerical findings. In the following, we report data for surface codes that is obtained with the same BP+OSD decoder as used for LCS codes. While BP+OSD is a general purpose decoder, surface codes are amenable to tailored decoders, most notably weight matching [76]. Comparing the performance of different decoders can be subtle, however we observed a better logical error rate when decoding with BP+OSD and therefore use the latter for a direct comparison. We discuss this in more detail in appendix C.

1. **Same parameter LCS code**: enforcing the parameters $[[n,k,d]]$ of surface (subindex s) and LCS codes to be the same corresponds to taking L copies of distance L surface codes and interconnecting them. Hence, this scenario explores the influence of the higher stabilizer weights with 2D non-local connectivity. This is achieved by setting $\ell_s = \ell = L - 1 = L_s - 1$, such that the parameters are

$$[[n,k,d]]_{\mathrm{lcs}} = [[n,k,d]]_{\mathrm{s}} = \left[\left[\left(2L^2 - 2L + 1\right)L, L, L\right]\right]. \tag{59}$$

   These families are marked dark green in figure 4.

   Under code capacity noise, figure 10(a) shows the logical error rate of any logical qubit compared to the physical bit-flip probability. With growing n, d, the logical error rate is increasingly suppressed. Compared to copies of surface codes, the interconnected LCS codes show a logical error rate that is up to 2 orders of magnitude lower at physical error rates on the order of 10^{-2}. We observe a crossing of curves with at $p^\star \approx 6.5 - 7\%$ indicating a threshold of $\approx 6.7 \pm 0.3\%$, slightly lower than the crossing point for surface codes at $p^\star \approx 7.5 \pm 0.3\%$.

   Results using the phenomenological noise model decoded with the adapted BP+OSD decoder are shown in figure 11(a). The results qualitatively show similar trends as for the previous code capacity case. To pick out a striking example, at a physical error rate of 10^{-2}, the logical error rate of the $[[595, 7, 7]]$ LCS code is 2 orders of magnitude lower compared to the surface code. The estimated threshold value we report for this code family under phenomenological noise is $\approx 2.9 \pm 0.1\%$ which within resolution coincides with the surface code family threshold $\approx 2.9 \pm 0.1\%$.
2. **Highest rate LCS code** for fixed dimension and distance. We can see in figure 4 that for the same k, d, we can tune the parameter ℓ for the LCS codes to a higher and lower rate. The highest rate is achieved by setting $\ell = \frac{L-1}{2}$. These LCS codes are depicted in figure 4 by purple rectangles. This family includes representatives of codes with the smallest number of physical qubits while having distances $d \geqslant 3$. We compare this family to surface codes with the same number of logical qubits k and distance d, which gives

$$[[n,k,d]]_{\mathrm{lcs}} = \left[\left[\frac{1}{2}\left(L^2 + 1\right)L, L, L\right]\right], \tag{60}$$

$$[[n,k,d]]_{\mathrm{s}} = \left[\left[\left(2L^2 - 2L + 1\right)L, L, L\right]\right]. \tag{61}$$

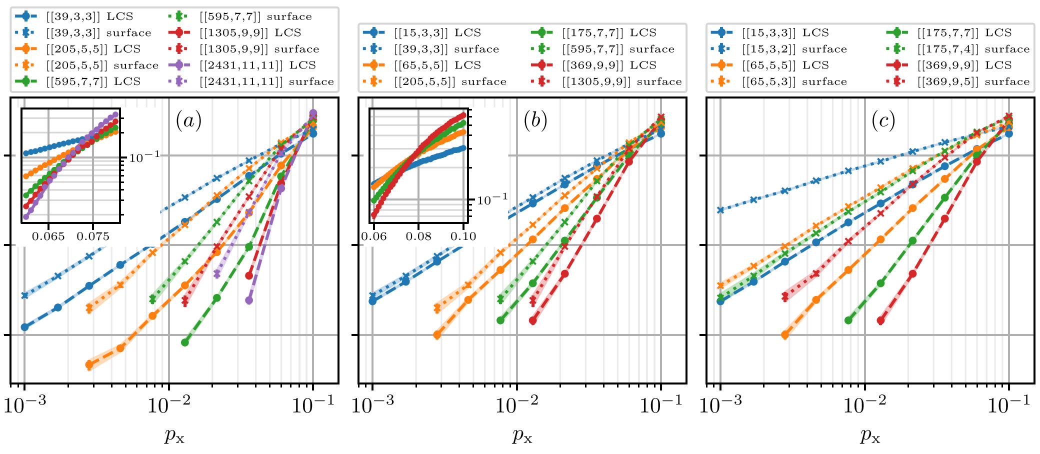

Figure 10. Logical error rate of LCS codes under *code-capacity bit-flip noise* using the BP+OSD decoder compared to surface codes. We compare three different parameter scaling choices, for all of which we observe a reduction of logical error rate with growing n, d. (a) Codes are chosen such that LCS and surface codes have the same parameters $[[n,k,d]]$. The logical error rate for the interconnected codes is up to 2 orders of magnitude lower than for the surface codes at a physical error rate in the order of 10^{-2}. We also observe a crossing of the curves indicating the existence of a threshold at $\approx 6.7 \pm 0.3\%$. (b) LCS codes are chosen such that they have the lowest qubit number n for same k, d. As explained in the text, there is still a reduction in logical error rate compared to disjoint surface codes, however this is less pronounced than in the setting discussed previously. The crossing of the curves indicates a slightly higher threshold at $\approx 7.7 \pm 0.2\%$. (c) Codes chosen such that for the same n, k, LCS codes have highest available distance. Here we observe the largest reduction in logical error rate. Note that this is consistent with the expectation that the distance is closely related to the number of correctable errors.

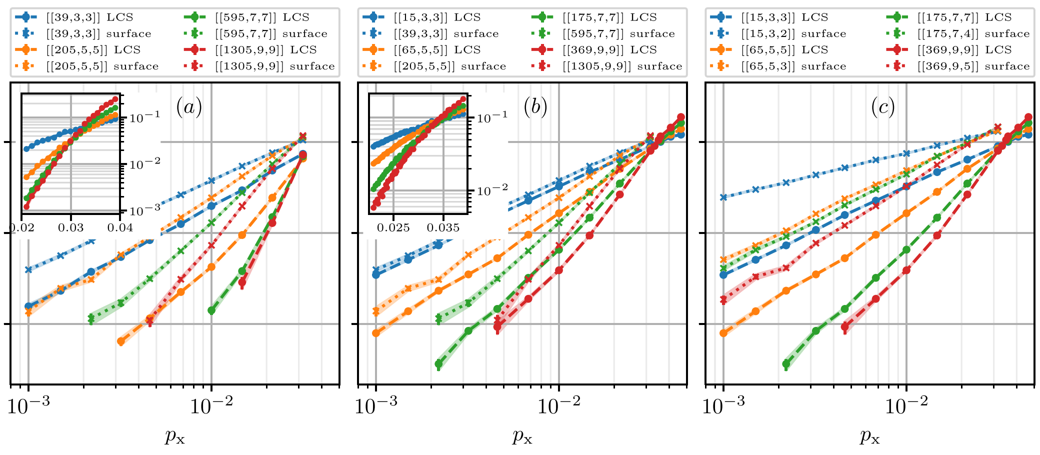

Figure 11. Logical error rate of LCS codes under *phenomenological bit-flip noise* using the BP+OSD decoder compared to surface codes. We simulate d cycles of syndrome measurements and report the logical error rate per cycle. We compare three different parameter scaling choices, for all of which we observe a reduction of logical error rate with growing n, d. (a) Codes are chosen such that LCS and surface codes have the same parameters $[[n,k,d]]$. The logical error rate for the interconnected codes is up to 2 orders of magnitude lower than for the surface codes at a physical error rate in the order of 10^{-2}. We also observe a crossing of the curves in the vicinity of $\approx 2.9 \pm 0.1\%$ indicating the existence of a threshold. (b) LCS codes are chosen such that they have the lowest qubit number for same k, d. As explained in the text, there is still a reduction in logical error rate compared to disjoint surface codes, however not as large. The crossing of the curves indicates a slightly higher threshold at $\approx 3.2 \pm 0.1\%$. (c) Codes chosen such that for the same n, k, LCS codes have highest available distance. Again, note that this is consistent with the expectation that the distance is closely related to the number of correctable errors.

This implies a saving of a factor 4 in qubit number for the same rate and distance for sufficiently large L.

The simulation results under code capacity noise are shown in figure 10(b). Also in this scenario, we find a lower logical error rate for the LCS family, which can be attributed to the smaller number of physical qubits. However the gain is smaller compared to lower rate codes which can be explained with the higher degeneracy of lower rate codes. We also observe a crossing of curves at a slightly higher $p^\star \approx 7.7 \pm 0.2\%$ indicating a threshold in that vicinity.

Under phenomenological noise (figure 11(b)) for the given scenario the crossing indicates a threshold estimate of $\approx 3.2 \pm 0.1\%$ compared to $\approx 2.9 \pm 0.1\%$ for the corresponding surface codes. The slight deviation in favor of the LCS family should be interpreted cautiously.

3. **Largest distance LCS code** for fixed physical and logical qubit number. This compares codes with parameters

$$[[n,k,d]]_{\mathrm{lcs}} = \left[\left[\frac{1}{2}\left(L^2+1\right)L, L, L\right]\right], \tag{62}$$

$$[[n,k,d]]_{\mathrm{s}} = \left[\left[\frac{1}{2}\left(L^2+1\right)L, L, \frac{L+1}{2}\right]\right]. \tag{63}$$

This implies a factor of 2 improvement in distance for same qubit number and rate for sufficiently large L. The logical error rates for this scenario under code capacity noise are shown in figure 10(c) and under phenomenological noise in figure 11(c). Here, we observe the largest gain among the three scenarios, i.e. when fixing the number of physical qubits and the desired code rate, choosing LCS codes over surface codes leads to the largest improvement in logical error rate. This can likely be attributed to the fact that the corresponding surface codes have a smaller distance $d_{\mathrm{s}} = \frac{d_{\mathrm{lcs}}+1}{2}$. Compared to the example in the 'same parameter scenario', here we observe an improvement of 2 orders of magnitude at physical error rate $p = 10^{-2}$ already for the even smaller $[[369,9,9]]$ LCS code compared to the corresponding nine copies of distance five surface codes with parameters $[[369,9,5]]$. Note that the threshold values for this scenario are the same as in the previous scenario.

Overall, these results indicate that LCS codes systematically offer improved performance in terms of logical error rate compared to standard surface codes under both code capacity and phenomenological noise.

5. Fault tolerant syndrome measurement circuits

The simulations with a phenomenological noise model show the high intrinsic error correction capabilities of LCS codes compared to surface codes, even under erroneous syndrome measurements. It is, however, not clear how this translates to an implementation with full circuit level noise because of the weight-6 instead of weight-4 stabilizers. Coupling in ancillary qubits increases the number of failure configurations and can even reduce the effective distance of the code. A noteworthy example are hook errors in topological color codes. Circuits have to be designed carefully, such that faults occurring during the syndrome measurement do not spread in an uncontrollable way [59].

For HGP codes, it was shown that any order of two qubit gates does not spread errors catastrophically, because the relevant stabilizers overlap with any logical operator on one location only [77], which is a sufficient condition. For (non-rotated) surface codes constructed as HGP of two classical repetition codes, the order does therefore also not matter. The argument of [77] does not generalize to LP codes. This can be seen in figure 15, where the first Z-stabilizer overlaps with the first logical Z-operator on two locations.

Remarkably, we find that syndrome measurement circuits based on the *coloration circuit* of [23] are distance preserving. We generate the required edge coloring for parallelizing entangling gates according to algorithm 1. Since the maximum degree of the edges in the Tanner graphs of LCS codes is $\Delta = 6$, this algorithm colors the edges of the Tanner graphs of LCS codes in 6 colors [78]. From this coloring, we construct circuits as shown in algorithm 2 and implement them using `stim` [79]. One syndrome measurement cycle of the $[[15,3,3]]$ code is shown in figure 12. We implement a circuit level noise model where each circuit element fails with probability p, i.e. after single qubit operations, a single qubit depolarizing error $E \in \{X,Y,Z\}$ occurs with probability p. After two qubit entangling operations, one of fifteen $E \in \{I,X,Y,Z\}^{\otimes 2} \setminus \{I,I\}$ is placed with probability p. Initialization and measurements (of ancilla and data qubits) are also noisy, modeled by single qubit bit (phase) flips with probability p after the Z (X)-initialization or before the Z (X)-measurement. Idling positions suffer from uniform depolarizing noise with $p_{\mathrm{idle}} = p/10$ [80].

The resulting circuits preserve the distance of the static code in the sense that it requires d circuit level faults (including two qubit errors after entangling gates) to cause a logical error without violating a stabilizer [81].

Algorithm 1. Edge coloring of Tanner graph of LCS codes.

```
  Input: $X$- or $Z$- Tanner graph $T$ of LCS code $\mathcal{Q}$
  Output: Edge coloring $C$
1 $C = []$ (empty list)
2 while $T$ is not empty do
3     $m =$ maximal matching of edges in $G$
4     $C \leftarrow [m]$
5     remove edges $m$ from $T$
6 end
7 return $C$
```

Algorithm 2. Coloration syndrome readout circuits for LCS codes.

```
   Input: $X$- and $Z$- edge colorings $C_X, C_Z$ of Tanner graphs of $(\ell, L)$-LCS code $\mathcal{Q}$
   Output: Syndrome readout circuit
1  for Cycle in range($L$) do
2      Initialize $\frac{n-k}{2}$ $Z$-ancilla qubits in $|0\rangle$
3      for color $c$ in $C_Z$ do
4          for edge $e = (d, a)$ in color $c$ do
5              $\mathrm{CX}_{d\to a}$
6          end
7      end
8      Measure $\frac{n-k}{2}$ $Z$-ancilla qubits in $Z$-basis
9      Initialize $\frac{n-k}{2}$ $X$-ancilla qubits in $|+\rangle$
10     for color $c$ in $C_X$ do
11         for edge $e = (d, a)$ in color $c$ do
12             $\mathrm{CX}_{a\to d}$
13         end
14     end
15     Measure $\frac{n-k}{2}$ $X$-ancilla qubits in $X$-basis
16 end
17 return $C$
```

We can then benchmark the memory capabilities of the LCS codes by sandwiching the coloration syndrome readout circuits with initialization in $|0\rangle^{\otimes n}$ $(|+\rangle^{\otimes n})$ and single-qubit measurement $M_Z^{\otimes n}(M_X^{\otimes n})$ for the $Z(X)$-logical operators. Both these operations are also assumed noisy (with probability p), modeled by single qubit bit (phase) flips after the $Z(X)$- initialization or before the $Z(X)$-measurement .

For decoding, we generate a circuit level parity check matrix. The rows represent *detectors*, i.e. sets of deterministic measurements such as two consecutive stabilizer measurement outcomes. The columns represent *error mechanisms* [82]. We finally decode using BP+OSD similar to the code capacity and phenomenological noise model.

Figure 13(a) shows the logical error rate per syndrome measurement cycle after d cycles. We show $[[15,3,3]]$, $[[39,3,3]]$ and $[[65,5,5]]$ LCS codes compared to d copies of distance 3, 5 and 7 surface codes. This data verifies the fault-tolerance of the circuits by showing a scaling of $p_L \propto p^{t+1}$ in the regime of low physical error rates. Sub-threshold and within error bars, the logical error rates of the $[[15,3,3]]$ and $[[65,5,5]]$ LCS codes are the same as their surface code counterparts with parameters $[[39,3,3]]$ and $[[205,5,5]]$. LCS codes can therefore reach the same logical error rate using 2.78 and 3.24 times fewer physical qubits. Again, encoding 3 logical qubits using more redundancy, i.e. 39 physical qubits decreases the logical error rate by almost an order of magnitude.

Figure 13(b) shows that pseudo-thresholds of LCS codes are significantly higher, $p^\star_{\text{th}}([[15,3,3]]) \approx 0.45\%$ and $p^\star_{\text{th}}(([[39,3,3]]) \approx 0.5\%$, compared to surface codes with $p^\star_{\text{th}}([[39,3,3]]) \approx 0.39\%$. This is a strong indication that already at small qubit numbers, even under circuit level noise, LCS codes can outperform surface codes.

Even if phenomenological noise thresholds are similar, the (asymptotic) value of the circuit level threshold is typically observed to be reduced for higher stabilizer weights [83]. We observe this in figure 13(c), showing the crossing of logical error rates for members of the highest rate LCS codes. Their crossing point is at $p_{\text{th}} \approx 0.5\%$ compared to $p_{\text{th}} \approx 0.9\%$ for surface codes, indicating a threshold in that vicinity.

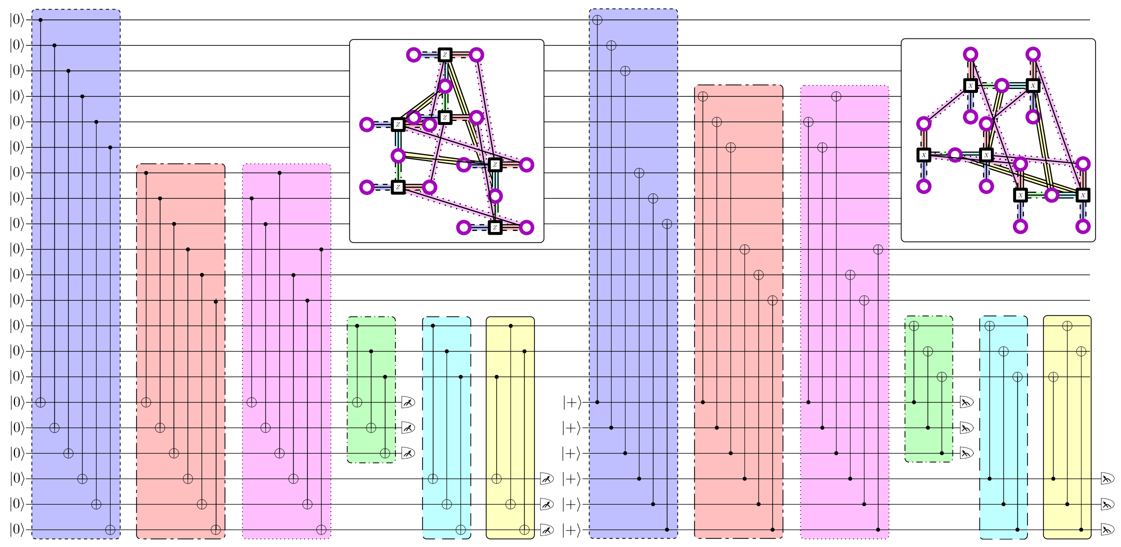

Figure 12. A syndrome measurement cycle for the $[[15,3,3]]$-LCS code constructed using the coloration circuit as described in the main text. Z- and X-stabilizer measurements are performed one after another in six layers respectively, showing potential for further parallelization. Inset are the Z- and X-Tanner graphs with edges backed with the same colored and shapes as the corresponding parallel blocks of CXs in the circuit.

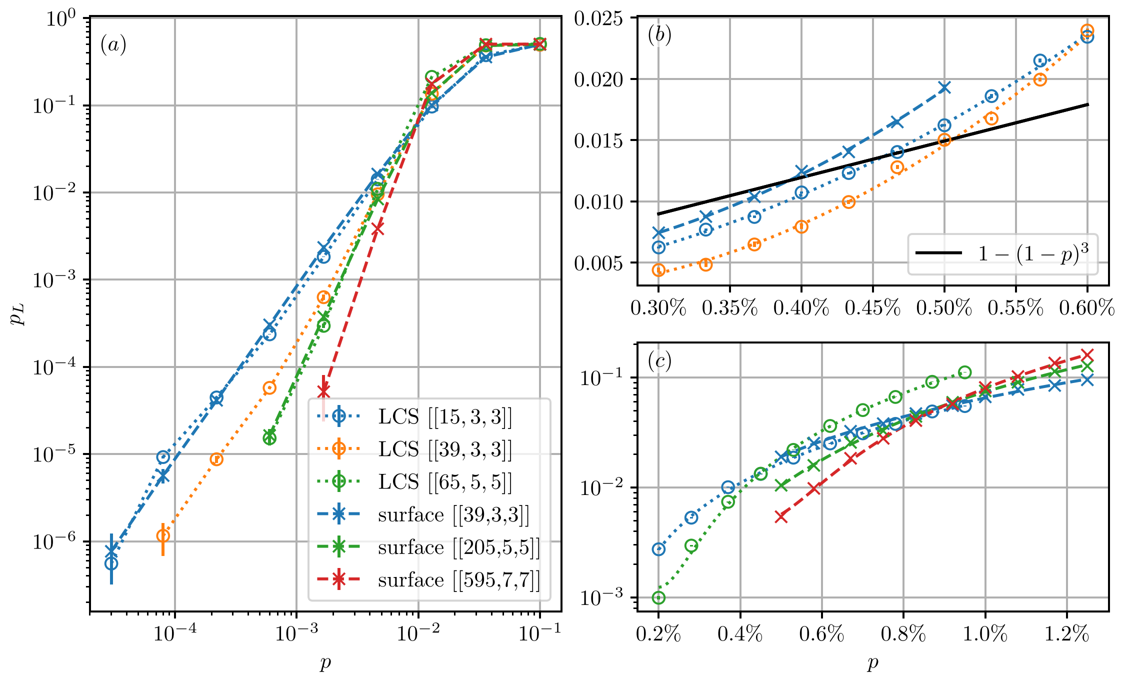

Figure 13. Logical error rates of LCS codes under *circuit level noise*, decoded using BP+OSD, compared surface codes decoded using a minimum weight perfect matching decoder. We simulate d syndrome measurement cycles and report the logical error rate per cycle. (a) Distance 3 and 5 LCS codes show the expected scaling $p_L \propto p^2$ and $p_L \propto p^3$ for small p, confirming the fault-tolerance property of the syndrome readout circuit implementations. Sub-threshold and within error bars, the logical error rates of the $[[15,3,3]]$ and $[[65,5,5]]$ LCS codes are the same as their surface code counterparts with parameters $[[39,3,3]]$ and $[[205,5,5]]$. (b) Pseudo-thresholds of LCS codes are significantly higher, $p^\star_{\text{th}}([[15,3,3]]) \approx 0.45\%$ and $p^\star_{\text{th}}(([[39,3,3]]) \approx 0.5\%$, compared to surface codes with $p^\star_{\text{th}}([[39,3,3]]) \approx 0.39\%$. (c) The crossing point of highest rate LCS codes is at $p_{\text{th}} \approx 0.5\%$ compared to $p_{\text{th}} \approx 0.9\%$ for surface codes, indicating threshold in that vicinity.

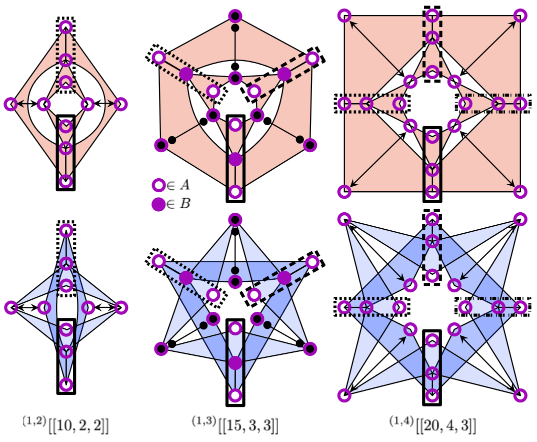

Figure 14. First three members of $\ell = 1$ LCS codes. X- (top) and Z-stabilizers (bottom). Logical operators shown in squares with different line styles for X and Z logical operators respectively. Note that they can be chosen to have same support. Swapping pairs of qubits on a diagonal as shown exemplarily for the $k = 2$ and $k = 4$ codes swaps X and Z stabilizers while leaving logical operators invariant. For $k = 3$ (center), the partitions A and B, as well the CZ operations of the fold-transversal Phase gate are indicated (cf equation (70)).

6. Towards logical gates in LCS codes

Implementing logical gates in QLDPC codes is a hard challenge, in particular if the implementation is to be fault-tolerant. In the following, we show a particular set of fault-tolerant gates for LCS codes when fixing $\ell = 1$. These have an almost planar geometrical representation and useful symmetries. Three representatives ($L = 2,3,4$) are shown in figure 14. They correspond to codes with parameters $[[n,k,d]] = [[10,1,2]]$, $[[15,3,3]]$ and $[[20,4,3]]$ respectively. They have a symmetry also called ZX-duality that exchanges X- and Z-stabilizers by simple swaps but keeps logical operators invariant. This symmetry is easily seen in the parity check matrices for $\ell = 1$,

$$\tilde{H}_X = \begin{pmatrix} 1 & 1+P^{(1)} & 0 & 0 & 1 \\ 0 & 0 & 1 & 1+P^{(1)} & 1+P^{(1)T} \end{pmatrix} \tag{64}$$

$$\tilde{H}_Z = \begin{pmatrix} 1 & 0 & 1+P^{(1)} & 0 & 1 \\ 0 & 1 & 0 & 1+P^{(1)} & 1+P^{(1)T} \end{pmatrix}. \tag{65}$$

Swapping the second and third block, i.e. applying the column permutation

$$\tau = \begin{pmatrix} L & 2L \end{pmatrix} \begin{pmatrix} L+1 & 2L+1 \end{pmatrix} \dots \begin{pmatrix} 2L-1 & 3L-1 \end{pmatrix} \tag{66}$$

brings $\tilde{H}_X \to \tilde{H}_Z$ and vice versa. Logical operators of these codes are given by

$$\tilde{L}_X = \begin{pmatrix} 1 & 0 & 0 & P^{(1)} & 1 \end{pmatrix} \tag{67}$$

$$\tilde{L}_Z = \begin{pmatrix} 1 & 0 & 0 & P^{(1)} & 1 \end{pmatrix} \tag{68}$$

and therefore left invariant under τ.

This duality enables several *fold-transversal* gates, in particular a fold-transversal Hadamard that implements a global logical Hadamard [44],

$$\overline{H} = \bigotimes_{i<\tau(i)} \mathrm{SWAP}\left(i,\tau(i)\right) \bigotimes_{i=1}^{n} H_i = \bigotimes_{i=1}^{k} \overline{H}_i \tag{69}$$

and a fold-transversal Phase gate

$$\overline{S} = \bigotimes_{i\in A} S_i \bigotimes_{i\in B} S_i^{\dagger} \bigotimes_{\substack{i=1,\dots,n \\ i<\tau(i)}} \mathrm{CZ}_{i,\tau(i)}. \tag{70}$$

In the latter, A and B are partitions of the qubits not involved in τ such that every X-stabilizer has half support on A and B respectively. These are also indicated by empty and filled circles in figure 14 for the [[15, 3, 3]]-LCS code. The CZs ensure that the stabilizer group is invariant by mapping $S_X \to S_X S_Z$ since for single-qubit Paulis $\mathrm{CZ}_{ij} X_i \mathrm{CZ}_{ij} = X_i Z_j$. The phase gate acts as $S_i X_i S_i^\dagger = i X_i Z_i$ and $S_i^\dagger X_i S_i = -i X_i Z_i$, such that the partitioning guarantees unwanted phase factors to cancel out. We can verify that the fold-transversal Phase gate implements a global logical Phase gate $\overline{S} = \otimes_{i=1}^{k} \overline{S}_i$.

These gates can be supplemented by the transversal $\overline{\mathrm{CNOT}}$ [84] to generate the full Clifford group on the set of logical qubits of copies of these $d = 3$ LCS codes. A universal set of fault-tolerant gates operating on single logical qubits is an open challenge. Approaches towards that goal may be based on (generalized) lattice surgery and gate teleportation [45, 46, 85], pieceable fault-tolerance [86] or code switching [87, 88]. Generalizing existing approaches for HGP codes [40, 48] to LP codes are another promising avenue.

7. Conclusion

We have introduced LCS codes as a new family of QLDPC codes based on the lifted product construction. We have shown how they can be viewed as interconnected copies of surface codes and how this additional connectivity leads to favorable parameters compared to disjoint copies of surface codes. . We adapted a BP+OSD decoder to the phenomenological noise setting and performed benchmarking of LCS codes for code-capacity and phenomenological noise. We observed that, while asymptotic thresholds are comparable to those of standard surface codes, the pseudo-thresholds can be significantly higher and logical error rates much lower. These advantages in particular also hold for small codes with qubit number below 100.

Implementing the stabilizer measurements of the codes in general involves coupling the data qubits to ancilla qubits by gates. These introduce another source of noise, that is captured by a *circuit level* noise model. Investigating the performance of LCS codes under circuit level noise will hinge on the construction of fault-tolerant circuits. For HGP codes, constructions of distance-preserving circuits have been recently introduced [23, 33, 77], a generalization to LP codes would be highly desirable. We have constructed fault-tolerant circuits for three members of the LCS code family that already demonstrate properties outperforming copies of surface codes. They also show potential for further improvement, for example by interleaving X- and Z- stabilizer measurements. Other modern fault-tolerant circuit constructions such as using flag-qubits [89] should facilitate resource efficient and fault-tolerant stabilizer readout protocols. The circuit construction also depends on assumptions on available gates and connectivity. Given that LCS codes are embeddable in three dimensions with local connectivity, this makes them highly attractive for near-term platforms and particularly suited for the emerging platforms of static 3D optical lattices or reconfigurable 2D arrays of Rydberg atoms [7, 56].

Data availability statement

The code and data that support the findings of this study are publicly available at the following URL: https://doi.org/10.5281/zenodo.10495421.

Acknowledgments

This research is part of the Munich Quantum Valley (K-8), which is supported by the Bavarian state government with funds from the Hightech Agenda Bayern Plus. We additionally acknowledge support by the BMBF project MUNIQC-ATOMS (Grant No. 13N16070). The authors gratefully acknowledge funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy 'Cluster of Excellence Matter and Light for Quantum Computing (ML4Q) EXC 2004/1' 390534769. Furthermore, we receive funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement No. 101114305 ('MILLENION-SGA1' EU Project) and ERC Starting Grant QNets through Grant No. 804247. M M also acknowledges support for the research that was sponsored by IARPA and the Army Research Office, under the Entangled Logical Qubits program through Cooperative Agreement Number W911NF-23-2-0216. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of IARPA, the Army Research Office, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. The authors gratefully acknowledge the computing time provided to them at the NHR Center NHR4CES at RWTH Aachen University (Project No. p0020074). This is funded by the Federal Ministry of Education and Research and the state governments participating on the basis of the resolutions of the GWK for national high performance computing at universities.

Appendix A. Block structure of parity check matrices and distance of LCS codes

The block structure of the parity check matrices of the LCS codes is shown in figure 15, exemplarily for $(\ell, L) = (2,4)$ ($[[52,4,4]]$). The parity check matrices inherit the block structure from the HGP, which can for example be seen in the block-diagonal structure of H_X. In total, the binary parity check matrices consist of $\ell(\ell+1)\times(2\ell^2+2\ell+1)$ blocks of size $L\times L$. Note that both H_X and H_Z are in row echelon form.

This particular form of the parity check matrices allows us to construct set of logical operators. If we have a set of k independent generators of logical operators and find their minimal weight, we can find the minimum distance of the code. By carefully inspecting the parity check matrices, we explicitly construct $L = k$ logical operators of weight $2\ell+1$. To that end, it is useful to first realize that, in regular surface codes, we can choose representatives of logical X- and Z- operators to have the same support by putting them on the diagonal. Indexing each column of the left with tuples $(i,j)\in\{0,1,\dots,\ell\}^2$ and for the right block with tuples $(i,j)\in\{0,1,\dots,\ell-1\}^2$, these diagonal qubits can be identified with columns $(0,0),(1,1),(2,2),\dots,(\ell,\ell)$ in the left part of the parity check matrices and columns $(0,0),(1,1),(2,2),\dots,(\ell-1,\ell-1)$ of the right part. These positions are also indicated by red boxes at the top of the parity check matrices in figure 15. The lift requires to select circulants at these positions, such that the resulting operators are also logical operators. It turns out that choosing logicals (before lifting L times) of the form

$$\Big(\underbrace{\underbrace{1\quad 0\quad \cdots\quad 0}_{\ell+1}|0\quad P^{(1)}\quad 0\quad\cdots\quad 0|0\quad 0\quad P^{(2)}\quad 0\quad\cdots\quad 0|\cdots|0\quad\cdots\quad 0\quad P^{(\ell)}}_{\ell+1}|$$
$$|\underbrace{\underbrace{1\quad 0\quad\cdots\quad 0}_{\ell}|0\quad P^{(1)}\quad 0\quad\cdots\quad 0|\cdots|0\quad\cdots\quad 0\quad P^{(\ell-1)}}_{\ell}\Big) \tag{A1}$$

ensures that every stabilizer has even overlap with the logicals. Note that in column (i,i) (of both left and right part), the circulant $P^{(i)}$ is placed. These are also shown in figure 15 for the $[[52,4,4]]$ code. As binary representations of logical X- and Z-operators, these are L pairs of disjoint operators since they only consist of circulants with one term, i.e. cyclic permutation matrices. With the respective partner (L_X^i, L_Z^i), the anti-commutation is guaranteed by the odd weight $2\ell+1$. Finally, these operators cannot have their weight reduced by adding stabilizers, which can also be verified using the block-structure of the parity check matrices. Every attempt to reduce the weight necessarily introduces new qubit connectivity.

However, taking the product of all these L operators results in an operator with ones in all the columns specified above. Multiplying stabilizers of rows $(0,0),(1,1),\dots,(\ell-1,\ell-1)$ will give an operator of (potentially lower) weight L.

We therefore found a set of L independent operators, each of minimum weight $2\ell+1$. Their sum has minimum weight L and all other combinations of stabilizer and logical operator have weight $\geqslant 2\ell+1$. We have therefore further evidence that the minimum distance of the code is

$$d = \min(2\ell+1, L)\,. \tag{A2}$$

$$H_X = \mathcal{B}_4\left[\left(\mathbf{1}_3 \otimes \begin{pmatrix} I & I+P^{(1)} & 0 \\ 0 & I & I+P^{(1)} \end{pmatrix} \quad \begin{pmatrix} I & 0 \\ I+P^{(1)T} & I \\ 0 & I+P^{(1)T} \end{pmatrix} \otimes \mathbf{1}_2\right)\right]$$

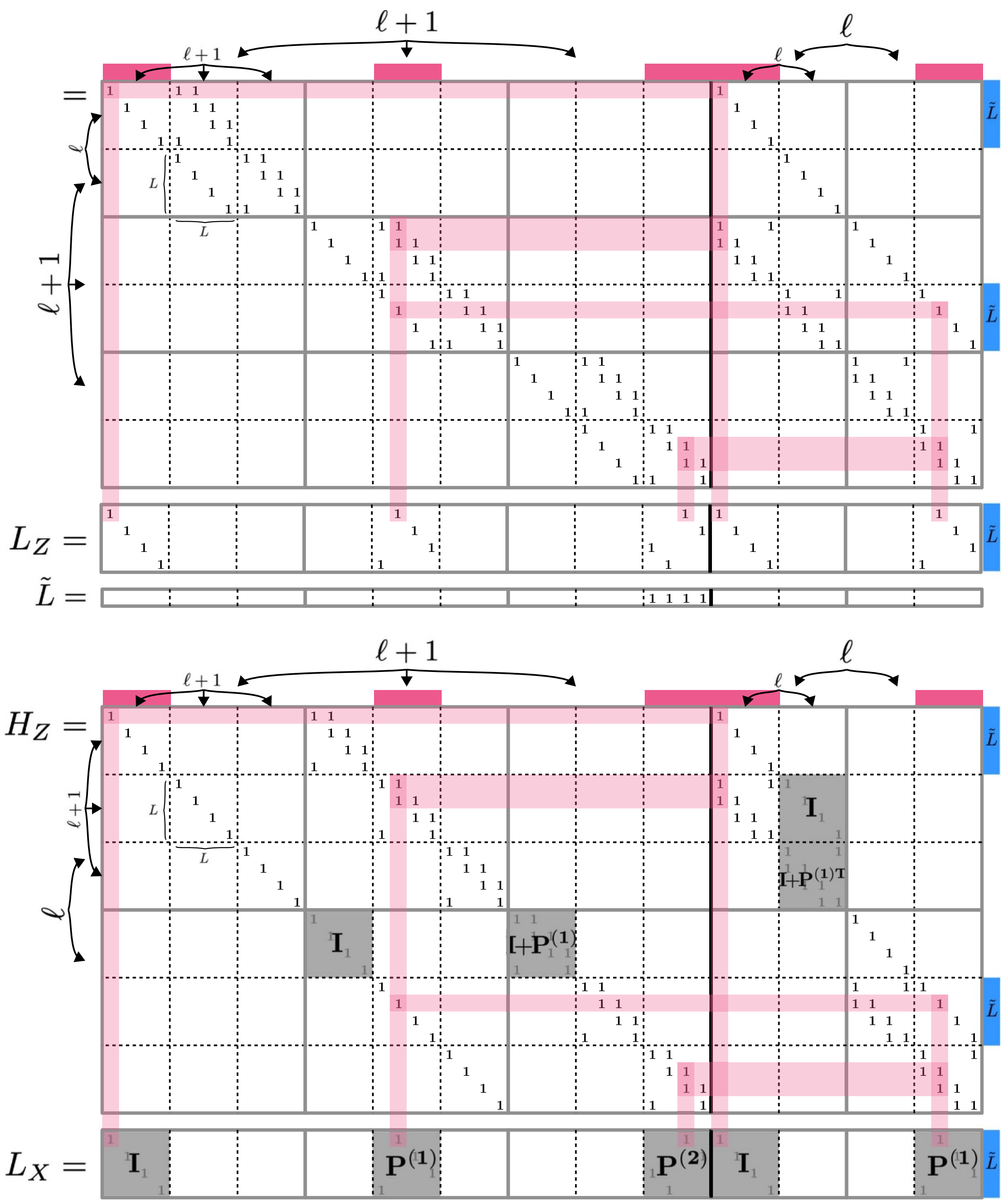

Figure 15. Block structure of the parity check matrices of X- (top) and Z-stabilizers (bottom), exemplarily for $(\ell, L) = (2, 4)$. Note that the block structure is inherited from the HGP product. The red rectangles on top of the matrices indicate the positions of the diagonal logicals in the regular HGP surface code. Also shown are matrices L_Z and L_X that are the binary representation of logical Z- and X-operators. They can be found as described in the text and have weight $2\ell + 1$. For the first logical operators, the overlap with the stabilizers of the respective other Pauli type is drawn in a red shade, visualizing the commutation. Additionally, we show the operator $\tilde{L}$ of weight L which can be constructed from products of all logical operators and some stabilizers as described in the text and shown with blue rectangles on the right side of the matrices.

Appendix B. Parameters for BP+OSD decoding

The BP+OSD decoder used in section 4 has a range of parameters that influence the decoding performance. For a comprehensive overview, refer to the source code [65]. Important parameters used in these simulations are shown in table 4. We observed that the (standard, but more complex) *product sum* method of BP (also described in the text) performs better than the lower complexity *minimum sum* method also provided by the

Table 4. Parameters used for BP+OSD decoding. Here, $[[n,k,d]]$ refer to the code parameters.

Parameter	Value
BP method	product sum
maximum BP iterations	$\lfloor \frac{d}{2} \rfloor$
osd order	$\min(d^2, 60)$

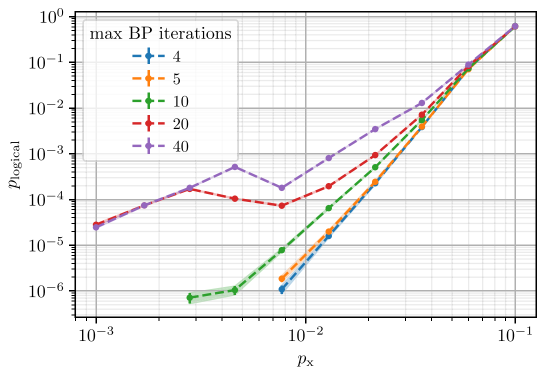

Figure 16. Logical error rate for the $[[369,9,9]]$-code using BP+OSD with different maximum BP iterations. Using only four iterations gives the lowest logical error rate.

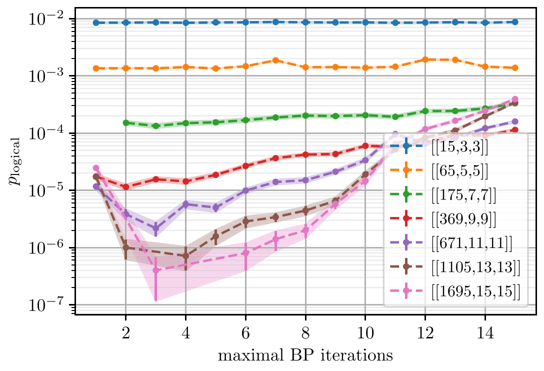

Figure 17. Logical error rates for the LCS-codes of family 2 (see main text section 4.3.4) using BP+OSD with different maximum BP iterations at a fixed physical error rate $p = 0.0129$. In particular for larger codes, the logical error rate is highly dependent on the maximum number of BP iterations before OSD post-processing.

package. The maximum number of BP iterations and the OSD order are also set heuristically based on observations in the decoding. For the $[[369,9,9]]$-LCS code, logical error rates for code capacity noise and different numbers of BP iterations are shown in figure 16. For different codes, we show the logical error rate for different numbers of BP iterations at a fixed physical error rate $p = 0.0129$ in figure 17. Heuristically, we find good performance if we set the maximum number of BP iterations to $\lfloor \frac{d}{2} \rfloor$. We attribute this to the observation that if BP is not successful, a large number of iterations will lead to an 'overfitting' and move us away from configurations, where the OSD post-processing step guesses the logical error correctly. A similar observation was also reported in [41]. Increasing the OSD order gives improved results at the cost of run time, which is why we limit the order to 60.

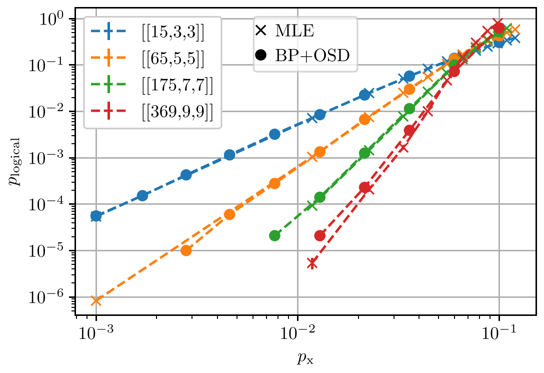

Figure 18. Logical error rate for LCS codes (family 2, see main text section 4.3.4) using MLE and BP+OSD. In particular for low qubit numbers, BP+OSD comes close to MLE decoding.

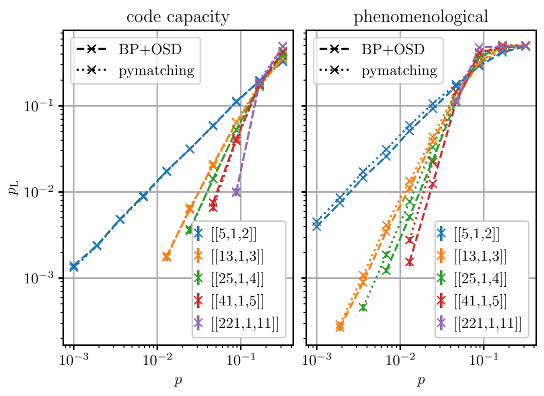

Figure 19. Logical error rates for surface codes using pymatching and BP+OSD with a code capacity (left) and phenomenological (right) noise model. BP+OSD achieves a comparable or lower logical error rate.

Appendix C. Validity of BP+OSD decoding

To verify the validity of BP+OSD decoding, we compare the most-likely error decoder to the BP+OSD decoder. As can be seen in figure 18, they perform very similarly for small qubit numbers, but for many qubits and low error rate, the decoding performance of BP+OSD decreases.

We compare the performance of BP+OSD with pymatching in figure 19. While it is typically observed that decoding surface codes with BP+OSD results in a lower threshold (see [35]), we observe that BP+OSD achieves lower logical error rates, in particular for the phenomenological noise model. For this reason, we compare the results for LCS codes with results for surface codes decoded with BP+OSD in the main text.

Appendix D. Sampling methods

The implementation details of the sampling for code capacity noise and phenomenological noise are shown in algorithms 3 and 4 respectively.

Algorithm 3. Code capacity sampling of quantum codes.

```
Input: Quantum Channel $\mathcal{E}$, physical error probabilities $\mathbf{p}_{\text{phys}} = (p_x, p_y, p_z)$, number of samples $n$, Quantum Code $\mathcal{Q}$
Output: Number of successful runs $n_{\text{success}}$
$n_{\text{success}} = 0$
for $i = 0, i < n, i = i + 1$ do
    $E = \mathcal{E}(0, \mathbf{p})$
    $\mathbf{s} = \sigma(E)$
    $E^\star = \mathrm{Dec}(\mathbf{s}, \mathbf{p}, \dots)$
    if $\sigma(EE^\star) = 0$ & $\langle EE^\star, L\rangle = 0 \quad \forall L \in \mathcal{L}(\mathcal{Q})$ then
        $n_{\text{success}} = n_{\text{success}} + 1$
    end
end
return $n_{\text{success}}$
```

Algorithm 4. Phenomenological noise sampling of quantum codes.

```
Input: Quantum Channel $\mathcal{E}$, physical qubit error probabilities $\mathbf{p}_{\text{phys}} = (p_x, p_y, p_z)$, Classical Channel $\mathcal{B}$, physical
    syndrome error probability $q$, number of samples $n$, Quantum Code $\mathcal{Q}$
Output: Number of successful runs $n_{\text{success}}$
$n_{\text{success}} = 0$
for $i = 0, i < n, i = i + 1$ do
    $E = 0$
    $\mathbf{s} = (0)^{((t+1)\times n_c)}$
    for $t = 0, t, d(\mathcal{Q}), t = t + 1$ do
        $E = \mathcal{E}(E, \mathbf{p})$
        $\mathbf{s}_{t+1} = \mathcal{B}(\sigma(E))$
    end
    $E^\star = \mathrm{Dec}_1(\mathbf{s}, \mathbf{p}, q, \dots)$
    $\mathbf{s}^\star = \sigma(EE^\star)$
    $E^{\star\star} = \mathrm{Dec}_2(\mathbf{s}^\star, \mathbf{p}, \dots)$
    if $\sigma(EE^\star E^{\star\star}) = 0$ & $\langle EE^\star E^{\star\star}, L\rangle = 0 \quad \forall L \in \mathcal{L}(\mathcal{Q})$
    then
        $n_{\text{success}} = n_{\text{success}} + 1$
    end
end
return $n_{\text{success}}$
```

ORCID iDs

Josias Old https://orcid.org/0009-0002-8811-1401
Markus Müller https://orcid.org/0000-0002-2813-3097

References

[1] Knill E, Laflamme R and Zurek W H 1998 Resilient quantum computation: error models and thresholds *Proc. R. Soc.* A **454** 365–84
[2] Ryan-Anderson C, Bohnet J G, Lee K, Gresh D, Hankin A, Gaebler J P, Francois D, Chernoguzov A, Lucchetti D and Brown N C *et al* 2021 Realization of real-time fault-tolerant quantum error correction *Phys. Rev.* X **11** 041058
[3] Postler L *et al* 2022 Demonstration of fault-tolerant universal quantum gate operations *Nature* **605** 675–80
[4] Hilder J, Pijn D, Onishchenko O, Stahl A, Orth M, Lekitsch B, Rodriguez-Blanco A, Müller M, Schmidt-Kaler F and Poschinger U G 2022 Fault-tolerant parity readout on a shuttling-based trapped-ion quantum computer *Phys. Rev.* X **12** 011032
[5] Krinner S *et al* 2022 Realizing repeated quantum error correction in a distance-three surface code *Nature* **605** 669–74
[6] Google Quantum AI 2023 Suppressing quantum errors by scaling a surface code logical qubit *Nature* **614** 676–81
[7] Bluvstein D *et al* 2024 Logical quantum processor based on reconfigurable atom arrays *Nature* **626** 58–65
[8] Kitaev A Y 1997 Quantum computations: algorithms and error correction *Russ. Math. Surv.* **52** 1191
[9] Stephens A M 2014 Fault-tolerant thresholds for quantum error correction with the surface code *Phys. Rev.* A **89** 022321
[10] Fowler A G, Mariantoni M, Martinis J M and Cleland A N 2012 Surface codes: towards practical large-scale quantum computation *Phys. Rev.* A **86** 032324
[11] Gottesman D2013 Fault-tolerant quantum computation with constant overhead (arXiv:1310.2984)

[12] MacKay D J C, Mitchison G and McFadden P L 2004 Sparse-graph codes for quantum error correction *IEEE Trans. Inf. Theory* **50** 2315–30
[13] Breuckmann N P and Eberhardt J N 2021 Quantum low-density parity-check codes *PRX Quantum* **2** 040101
[14] Tillich J-P and Zemor G 2013 Quantum ldpc codes with positive rate and minimum distance proportional to the square root of the blocklength *IEEE Trans. Inf. Theory* **60** 1193–202
[15] Sipser M and Spielman D A 1996 Expander codes *IEEE Trans. Inf. Theory* **42** 1710–22
[16] Breuckmann N P and Eberhardt J N 2021 Balanced product quantum codes *IEEE Trans. Inf. Theory* **67** 6653–74
[17] Panteleev P and Kalachev G 2022 Asymptotically good quantum and locally testable classical ldpc codes *Proc. 54th Annual ACM SIGACT Symp. on Theory of Computing* pp 375–88
[18] Leverrier A and Zémor G2022 Quantum tanner codes (arXiv:2202.13641)
[19] Dinur I, Hsieh M-H, Lin T-C and Vidick T 2023 *Good Quantum Ldpc Codes With Linear Time Decoders* (Association for Computing Machinery) pp 905–18
[20] Bravyi S and Terhal B 2009 A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes *New J. Phys.* **11** 043029
[21] Bravyi S, Poulin D and Terhal B 2010 Tradeoffs for reliable quantum information storage in 2d systems *Phys. Rev. Lett.* **104** 050503
[22] Delfosse N, Beverland M E and Tremblay M A 2021 Bounds on stabilizer measurement circuits and obstructions to local implementations of quantum ldpc codes (arXiv:2109.14599)
[23] Tremblay M A, Delfosse N and Beverland M E 2022 Constant-overhead quantum error correction with thin planar connectivity *Phys. Rev. Lett.* **129** 050504
[24] Bravyi S, Cross A W, Gambetta J M, Maslov D, Rall P and Yoder T J 2024 High-threshold and low-overhead fault-tolerant quantum memory *Nature* **627** 778–82
[25] Strikis A and Berent L 2023 Quantum low-density parity-check codes for modular architectures *PRX Quantum* **4** 020321
[26] Bruzewicz C D, Chiaverini J, McConnell R and Sage J M 2019 Trapped-ion quantum computing: progress and challenges *Appl. Phys. Rev.* **6** 021314
[27] Kaushal V, Lekitsch B, Stahl A, Hilder J, Pijn D, Schmiegelow C, Bermudez A, Müller M, Schmidt-Kaler F and Poschinger U 2020 Shuttling-based trapped-ion quantum information processing *AVS Quantum Sci.* **2** 014101
[28] Saffman M 2016 Quantum computing with atomic qubits and Rydberg interactions: progress and challenges *J. Phys. B: At. Mol. Opt. Phys.* **49** 202001
[29] Moses S A *et al* 2023 A race track trapped-ion quantum processor *Phys. Rev.* X **13** 041052
[30] Bluvstein D *et al* 2022 A quantum processor based on coherent transport of entangled atom arrays *Nature* **604** 451–6
[31] Manigrasso J, Chillón I, Genna V, Vidossich P, Somarowthu S, Pyle A M, De Vivo M and Marcia M 2022 Erasure conversion for fault-tolerant quantum computing in alkaline earth Rydberg atom arrays *Nat. Commun.* **13** 1–7
[32] Cong I, Levine H, Keesling A, Bluvstein D, Wang S-T and Lukin M D 2022 Hardware-efficient, fault-tolerant quantum computation with rydberg atoms *Phys. Rev.* X **12** 021049
[33] Xu Q, Ataides J P B, Pattison C A, Raveendran N, Bluvstein D, Wurtz J, Vasic B, Lukin M D, Jiang L and Zhou H 2023 Constant-overhead fault-tolerant quantum computation with reconfigurable atom arrays (arXiv:2308.08648)
[34] Panteleev P and Kalachev G 2021 Degenerate quantum LDPC codes with good finite length performance *Quantum* **5** 585
[35] Roffe J, White D R, Burton S and Campbell E 2020 Decoding across the quantum low-density parity-check code landscape *Phys. Rev. Res.* **2** 043423
[36] Delfosse N, Londe V and Beverland M E 2022 Toward a union-find decoder for quantum LDPC codes *IEEE Trans. Inf. Theory* **68** 3187–99
[37] Berent L, Burgholzer L and Wille R 2023 Software tools for decoding quantum low-density parity-check codes *ASPDAC '23: Proc. 28th Asia and South Pacific Design Automation Conf.* (Association for Computing Machinery) pp 709–14
[38] Leverrier A and Zémor G 2023 Efficient decoding up to a constant fraction of the code length for asymptotically good quantum codes *Proc. 2023 Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)* (Society for Industrial and Applied Mathematics) pp 1216–44
[39] Gu S, Pattison C A and Tang E 2023 An efficient decoder for a linear distance quantum ldpc code *Proc. 55th Annual ACM Symp. on Theory of Computing (STOC 2023)* (Association for Computing Machinery) pp 919–32
[40] Quintavalle A O, Vasmer M, Roffe J and Campbell E T 2021 Single-shot error correction of three-dimensional homological product codes *PRX Quantum* **2** 020340
[41] Higgott O and Breuckmann N P 2023 Improved single-shot decoding of higher-dimensional hypergraph-product codes *PRX Quantum* **4** 020332
[42] Eastin B and Knill E 2009 Restrictions on transversal encoded quantum gate sets *Phys. Rev. Lett.* **102** 110502
[43] Jochym-O'Connor T, Kubica A and Yoder T J 2018 Disjointness of stabilizer codes and limitations on fault-tolerant logical gates *Phys. Rev.* X **8** 021047
[44] Breuckmann N P and Burton S 2024 Fold-transversal clifford gates for quantum codes *Quantum* **8** 1372
[45] Brun T A, Zheng Y-C, Hsu K-C, Job J and Lai C-Y 2015 Teleportation-based fault-tolerant quantum computation in multi-qubit large block codes (arXiv:1504.03913)
[46] Cohen L Z, Kim I H, Bartlett S D and Brown B J 2022 Low-overhead fault-tolerant quantum computing using long-range connectivity *Sci. Adv.* **8** eabn1717
[47] Jochym-O'Connor T 2019 Fault-tolerant gates via homological product codes *Quantum* **3** 120
[48] Krishna A and Poulin D 2021 Fault-tolerant gates on hypergraph product codes *Phys. Rev.* X **11** 011023
[49] Quintavalle A O, Webster P and Vasmer M 2023 Partitioning qubits in hypergraph product codes to implement logical gates *Quantum* **7** 1153
[50] Panteleev P and Kalachev G 2022 Quantum LDPC codes with almost linear minimum distance *IEEE Trans. Inf. Theory* **68** 213–29
[51] Tanner R 1981 A recursive approach to low complexity codes *IEEE Trans. Inf. Theory* **27** 533–47
[52] Pryadko L P, Shabashov V A and Kozin V K 2022 Qdistrnd: a gap package for computing the distance of quantum error-correcting codes *J. Open Source Softw.* **7** 4120
[53] Haah J 2021 A degeneracy bound for homogeneous topological order *SciPost Phys.* **10** 011
[54] Lin T-C, Wills A and Hsieh M-H 2023 Geometrically local quantum and classical codes from subdivision (arXiv:2309.16104)
[55] Williamson D J and Baspin N 2023 Layer codes (arXiv:2309.16503)
[56] Barredo D, Lienhard V, de Léséleuc S, Lahaye T and Browaeys A 2018 Synthetic three-dimensional atomic structures assembled atom by atom *Nature* **561** 79–82

[57] Fowler A G, Whiteside A C and Hollenberg L C L 2012 Towards practical classical processing for the surface code *Phys. Rev. Lett.* **108** 180501
[58] Poulin D and Chung Y 2008 On the iterative decoding of sparse quantum codes (arXiv:0801.1241)
[59] Landahl A J, Anderson J T and Rice P R 2011 Fault-tolerant quantum computing with color codes (arXiv:1108.5738)
[60] Jensen K, Cardoso J G and Sonnenschein N 2017 Optlang: an algebraic modeling language for mathematical optimization *J. Open Source Softw.* **2** 139
[61] Free Software Foundation 2011 GNU Linear ProgrammingKit (available at: https://www.gnu.org/software/glpk/glpk.html)
[62] Gallager R 1962 Low-density parity-check codes *IRE Trans. Inf. Theory* **8** 21–28
[63] MacKay D J C and Neal R M 1997 Near Shannon limit performance of low density parity check codes *Electron. Lett.* **33** 457–8
[64] Roffe J 2019 Quantum error correction: an introductory guide *Contemp. Phys.* **60** 226–45
[65] Roffe J 2022 BP+OSD: a decoder for quantum LDPC codes (available at: https://github.com/quantumgizmos/bp_osd)
[66] Roffe J 2022 LDPC: Python tools for low density parity check codes (available at: https://pypi.org/project/ldpc/)
[67] Preskill J 1997 Fault-tolerant quantum computation (arXiv:quant-ph/9712048)
[68] Aharonov D and Ben-Or M 2008 Fault-tolerant quantum computation with constant error rate *SIAM J. Comput.* **38** 1207–82
[69] Kovalev A A and Pryadko L P 2013 Fault tolerance of quantum low-density parity check codes with sublinear distance scaling *Phys. Rev.* A **87** 020304
[70] Dumer I, Kovalev A A and Pryadko L P 2015 Thresholds for correcting errors, erasures and faulty syndrome measurements in degenerate quantum codes *Phys. Rev. Lett.* **115** 050502
[71] Dennis E, Kitaev A, Landahl A and Preskill J 2002 Topological quantum memory *J. Math. Phys.* **43** 4452–505
[72] Wang C, Harrington J and Preskill J 2003 Confinement-Higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory *Ann. Phys., NY* **303** 31–58
[73] Ohno T, Arakawa G, Ichinose I and Matsui T 2004 Phase structure of the random-plaquette Z2 gauge model: accuracy threshold for a toric quantum memory *Nucl. Phys.* B **697** 462–80
[74] Kovalev A A, Prabhakar S, Dumer I and Pryadko L P 2018 Numerical and analytical bounds on threshold error rates for hypergraph-product codes *Phys. Rev.* A **97** 062320
[75] Ippoliti M, Li Y, Rakovszky T and Khemani V 2023 The physics of (good) LDPC codes I. Gauging and dualities (arXiv:2310.16032)
[76] Higgott O 2021 PyMatching: A Python package for decoding quantum codes with minimum-weight perfect matching (arXiv:2105.13082)
[77] Manes A G and Claes J 2023 Distance-preserving stabilizer measurements in hypergraph product codes (arXiv:2308.15520)
[78] We implement the edge coloring using the `maximal_matching` algorithm of the `NetworkX` `python` library [90]
[79] Gidney C 2021 Stim: a fast stabilizer circuit simulator (arXiv:2103.02202)
[80] We use $p/10$ instead of p to account for the optimization potential in parallelizing the circuits
[81] This can be done in `stim` using `circuit.search_for_undetectable_logical_errors(...)`
[82] To generate the decoding matrix, we use `stim`'s automated detector error model generation
[83] Ruiz D, Guillaud J'emie, Leverrier A, Mirrahimi M and Vuillot C 2024 LDPC-cat codes for low-overhead quantum computing in 2D (arXiv:2401.09541)
[84] Shor P W 1996 Fault-tolerant quantum computation (arXiv:quant-ph/9605011)
[85] Horsman C, Fowler A G, Devitt S and Van Meter R 2012 Surface code quantum computing by lattice surgery *New J. Phys.* **14** 123011
[86] Yoder T J, Takagi R and Chuang I L 2016 Universal fault-tolerant gates on concatenated stabilizer codes *Phys. Rev.* X **6** 031039
[87] Beverland M E, Kubica A and Svore K M 2021 Cost of universality: a comparative study of the overhead of state distillation and code switching with color codes *PRX Quantum* **2** 020341
[88] Butt F, Heußen S, Rispler M and Müller M 2024 Fault-tolerant code switching protocols for near-term quantum processors *PRX Quantum* **5** 020345
[89] Chamberland C and Beverland M E 2018 Flag fault-tolerant error correction with arbitrary distance codes *Quantum* **2** 53
[90] Hagberg A, Swart P and Chult D S 2008 Exploring network structure, dynamics, and function using Network *Technical Report* (Los Alamos National Lab.(LANL), Los Alamos, NM)