

23	6 Decoding across the quantum low-density parity-check code landscape (Notes on [13])	23
24	6.1 Low-density parity-check codes	23
25	6.2 Quantum coding	25
26	6.3 Quantum LDPC codes	25
27	6.4 Belief propagation decoding	26
28	6.5 Numerical Simulations	29
29	7 Notes on Fault-Tolerant Belief Propagation for Practical Quantum memory [9]	29
30	7.1 Introduction	30
31	7.2 Quantum stabilizer codes and circuit-level noise model	30
32	7.3 generalized check matrix for syndrome extraction circuit and circuit-level decoding problem and FTBP and sparse generalized check matrix for space-time Tanner graph and error merging and probabilisitic error consolidation	30
33	7.4 Generalized check matrix	32
34	7.5 BP complexity discussion for circuit-level decoding problem	35
35	7.6 Probabilistic error consolidation	35
36	7.7 adaptive sliding window	35
37	7.8 simulations	35
40	8 Notes on Dan Browne topological codes	35
41	8.1 Toric codes	35
42	8.2 Elements of Topology and Homology	38

45 1 Introduction

46 Quantum algorithms potentially speed up calculations exponentially but at the same time require thousands
 47 of gate operations. High-fidelity gates can be realized in experiments [16], but accumulation of errors can
 48 be drastic in even larger systems as required in many applications [2]. Assuming a simple model where each
 49 gate has independent stochastic errors with fidelity f , the probability that a circuit of m gates has no errors
 50 is f^m , which can be around 50% with 140 consecutive gates with fidelity $f = 99.5\%$. The probability of
 51 error-free execution drops rapidly as the circuit depth increases. For practical algorithms requiring thousands
 52 or millions of operations, such raw physical error rates are clearly insufficient. This motivates the use of
 53 quantum error correction (QEC), in which logical qubits are redundantly encoded into multiple physical
 54 qubits to actively detect and correct errors. The threshold theorem ensures that if the physical error rate is
 55 below a certain threshold value, then logical errors can be suppressed arbitrarily by increasing the code size.
 56 For surface codes, one of the most studied QEC schemes, this threshold is on the order of 1% and follows
 57 the relation [4]:

$$P_L \sim \left(\frac{P}{P_{\text{thre}}} \right)^{\frac{d+1}{2}}$$

58 to suppress the logical error rate P_L per stage, where d is the code distance, P is the physical error rate
 59 per stage, and P_{thre} is the error threshold. This relation also shows how quantum error correction mitigates
 60 logical errors P_L by mapping physical operations into logical operations.

61 Suppose one physical qubit in the Shor code suffers a small coherent error:

$$U_X(\theta) = e^{-i\frac{\theta}{2}X} = \cos\left(\frac{\theta}{2}\right)I - i\sin\left(\frac{\theta}{2}\right)X.$$

62 For small θ , this can be approximated as

$$U_X(\theta) \approx I - i\frac{\theta}{2}X.$$

63 **Action on an encoded state**

64 Let $|\psi_L\rangle$ denote the encoded logical state. After the error, the state becomes

$$U_X(\theta)|\psi_L\rangle = \cos\left(\frac{\theta}{2}\right)|\psi_L\rangle - i \sin\left(\frac{\theta}{2}\right)X_j|\psi_L\rangle,$$

65 where X_j denotes a bit-flip on physical qubit j .

66 Thus, the corrupted state is a superposition of:

- 67 • the “no error” branch with amplitude $\cos\left(\frac{\theta}{2}\right)$, and
- 68 • the “error on qubit j ” branch with amplitude $-i \sin\left(\frac{\theta}{2}\right)$.

69 In this article, we will first highlight the importance of quantum error correction. We will then examine
70 the work based mainly on [3], aiming to provide explanations, reproduce key results, and draw inspiration
71 from their findings.

72 **2 Stabilizer Formalism**

73 A well-developed framework used to efficiently characterize quantum error correction codes is the *stabilizer*
74 *formalism*. It describes a code space as the simultaneous +1 eigenspace of a set of commuting Pauli operators,
75 called *stabilizers*. Errors are detected by measuring these stabilizers: if an error anticommutes with a
76 stabilizer, the corresponding measurement outcome flips, providing an error syndrome.

77 **Stabilizer code.** A *stabilizer code* on n qubits is specified by an abelian subgroup $\mathcal{S} \subseteq \mathcal{P}_n$ (the n -qubit
78 Pauli group) that does not contain $-I$. The *codespace* \mathcal{C} is the joint +1 eigenspace of all elements of \mathcal{S} :

$$\mathcal{C} = \{ |\psi\rangle \in (\mathbb{C}^2)^{\otimes n} : S|\psi\rangle = |\psi\rangle \quad \forall S \in \mathcal{S} \}.$$

79 **Dimension.** If the code encodes k logical qubits into n physical qubits (an $[[n, k, d]]$ code), then with $n - k$
80 independent stabilizer generators we have

$$\dim \mathcal{C} = \frac{2^n}{2^{n-k}} = 2^k.$$

81 **Logical basis inside the codespace.** Because $\dim \mathcal{C} = 2^k$, we may choose an orthonormal *logical basis*
82 of \mathcal{C} ,

$$\mathcal{B}_L = \{ |x_L\rangle : x \in \{0, 1\}^k \},$$

83 such that each basis vector lies in the codespace (hence is stabilized):

$$S|x_L\rangle = |x_L\rangle \quad \forall S \in \mathcal{S}, \forall x \in \{0, 1\}^k.$$

84 **Arbitrary logical state (spanning by the logical basis).** Any generic codespace vector $|\psi\rangle \in \mathcal{C}$ can
85 be expressed in the logical basis as

$$|\psi\rangle \equiv |\psi_L\rangle = \sum_{x \in \{0, 1\}^k} \alpha_x |x_L\rangle, \quad \sum_x |\alpha_x|^2 = 1.$$

86 Thus the logical basis $\{|x_L\rangle\}$ spans the same subspace that was defined abstractly by the condition $S|\psi\rangle =$
87 $|\psi\rangle$.

88 **Expansion in the physical (computational) basis.** Each logical basis vector is itself a vector in the
 89 n -qubit Hilbert space and typically expands as a superposition of computational basis states:

$$|x_L\rangle = \sum_{i=0}^{2^n-1} c_i^{(x)} |i\rangle,$$

90 with coefficients $\{c_i^{(x)}\}$ constrained by the stabilizer conditions $S|x_L\rangle = |x_L\rangle$ for all $S \in \mathcal{S}$. These constraints
 91 select which computational-basis components may appear and with what relative phases or amplitudes.

92 **Remark (stabilizers vs. logical operators).** Stabilizers act *trivially* on every codespace vector (eigen-
 93 value +1) and thus define \mathcal{C} . By contrast, *logical operators* act *nontrivially* within \mathcal{C} ; they lie in the normalizer
 94 $N(\mathcal{S})$ of \mathcal{S} in \mathcal{P}_n but not in \mathcal{S} itself.

95 There are $n - k$ independent stabilizer generators, which generate the full stabilizer group of size $|\mathcal{S}| =$
 96 2^{n-k} . These $n - k$ constraints reduce the full 2^n -dimensional Hilbert space to the 2^k -dimensional code space.
 97 In other words:

- 98 • n physical qubits provide a Hilbert space of dimension 2^n ,
- 99 • $n - k$ stabilizer constraints remove $n - k$ degrees of freedom,
- 100 • leaving k logical qubits, i.e. a code space of dimension 2^k (same as logical state dimension).

101 Within this framework, many of the most important quantum error correction codes—including repetition
 102 codes, concatenated codes (e.g., Shor code), the color code of Hamming codes (e.g., Steane code), surface
 103 codes, and subsystem codes (e.g., Bacon-Shor code)—can be described in a unified and elegant way.

104 3 Homomorphic Logical Measurements (Notes on the Talk and 105 Paper [5, 7])

106 3.1 Surface code

107 Each stabilizer act on neighbor local qubits. The error threshold is low but the code distance cannot be well
 108 increased even with larger physical qubits number. The relation write $kd^2 = O(n)$ with n, k, d being the
 109 typical $[[n, k, d]]$ definition error code. Here we can see that if restricting on encoding rates $\frac{k}{n} \sim 1$, code
 110 distance d scales as $O(1)$. Noted that for linear code, $n \geq k + d - 1$

111 3.2 Quantum LDPC (Low-Density Parity-Check) code

112 Decoding time is large to cost computation delays, while fast decoding is an essential ingredient to fault-
 113 tolerant computation. Sparce stabilzers (low weight hamming weight) can improve the problem [14]. Quantum
 114 LDPC code provide nonlocal stabilzers, measurements. The code distance d can be increased faster not
 115 following $kd^2 = n$ (code rate: $\frac{k}{n}$). In addition, one motivation comes from when standard Shor and Steane
 116 style logical measurement cannnot be performed on large quantum LDPC code.

117 For typical surface code, code rate scales asymptotically to zero and with square root of code distance
 118 when enlarging code block. Improvement gives nonvanishing encoding rate for different surfaces (more non-
 119 trivial loops), but with code distance logarithmic in the blocklength. **Hypergraph product construction**
 120 improve this problem: First of all, we have

$$\text{Toric code} \subset \text{Hypergraph product codes} \subset \text{Homological codes} \subset \text{Stabilizer codes}.$$

121 Noted that homological codes belong to mutually orthogonal binary codes, and stabilizer codes belong to
 122 additive self-orthogonal code over GF (4) with respect to the trace Hermitian inner product

¹²³ **Theorem 1:** it guarantees that from any full-rank classical LDPC parity-check matrix H , you can systematically build a quantum LDPC code whose parameters are exactly those given.

Classical	Quantum (constructed)	Notes
Code $[n, k, d]$	$\rightarrow [[n^2 + (n - k)^2, k^2, d]]$	Quantum code parameters
LDPC (sparse) row weight i , column weight j	\rightarrow LDPC (row weight $\approx i + j$)	Sparsity preserved
Parity-check matrix H	$\rightarrow (H_X, H_Z)$ built from $H \otimes I, I \otimes H^T$	CSS-type stabilizers
Distance d	\rightarrow Distance d	Same as classical code
Rate k/n	$\rightarrow \frac{(k/n)^2}{1 + (1 - k/n)^2}$	Quantum rate expression

¹²⁵ **LDPC codes** linear codes with sparse parity check matrix and can also be described by Tanner graph
¹²⁶ denoted by bipartite $\mathcal{T}(V, C, E)$. For $H = \mathbb{F}_2^{r \times n}$, $V = 1, \dots, n$ (called variable nodes) is the columns of H and
¹²⁷ $C = \otimes_1, \dots, \otimes_r$ (check nodes) with column indices i and row indices j . There is an edge set E when $H_{ij} = 1$.

¹²⁸ **Generalizations from Toric code** An $m \times m$ toric code (V, E) can be represented as $\mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z}$, where
¹²⁹ the two-dimensional vertex set consists of coordinates (x, y) with each coordinate ranging over $\{0, 1, 2, \dots, m - 1\}$. The vertex-edge incidence matrix \mathbf{H}_1 is defined such that $(\mathbf{H}_1)_{ij} = 1$ if vertex i is incident to edge j .
¹³⁰ Each i -th row of \mathbf{H}_1 corresponds to a vertex (an X -stabilizer), and each j -th column corresponds to an
¹³¹ edge, which represents a physical qubit. Pictorially, for a four qubits repetition code (building block of toric
¹³² code) can be denoted as in Table 1. Let $H_1 \in \{0, 1\}^{r_1 \times n_1}$ and $H_2 \in \{0, 1\}^{r_2 \times n_2}$ be classical parity-check

X stabilizer (row)	edge ₀	edge ₁	edge ₂	edge ₃
X ₀	1	1	0	0
X ₁	0	1	1	0
X ₂	0	0	1	1
X ₃	1	0	0	1

Table 1: Toric code H_r matrix for 4 edges

¹³³ matrices. Define identity matrices I_a of the indicated sizes, and use the Kronecker product \otimes . Then, the
¹³⁴ CSS stabilizer matrices are given by
¹³⁵

$$H_X = [H_1 \otimes I_{n_2} \mid I_{r_1} \otimes H_2^T], \quad H_Z = [I_{n_1} \otimes H_2 \mid H_1^T \otimes I_{r_2}].$$

¹³⁶ **Toric code as a special case.** If both classical codes are chosen as the length- L repetition code with
¹³⁷ parity-check $H_r \in \{0, 1\}^{L \times L}$ (representing a cyclic ring), then the toric-code stabilizer matrices become

$$H_X = [H_r \otimes I_L \mid I_L \otimes H_r^T], \quad H_Z = [I_L \otimes H_r \mid H_r^T \otimes I_L].$$

¹³⁸ Here the rows of H_X correspond to plaquette (face) X -stabilizers and the rows of H_Z correspond to vertex
¹³⁹ Z -stabilizers, while the columns index the $2L^2$ edge qubits of the lattice.

¹⁴⁰ 3.3 Logical measurements of Shor and Steane type

¹⁴¹ Standard approach will encounter two possible limitations. First, if an error occur on the ancilla qubits, the
¹⁴² error will propagate to data qubits and cause higher weight errors. Below shows a graph of common error

143 propagations extracted from [15]

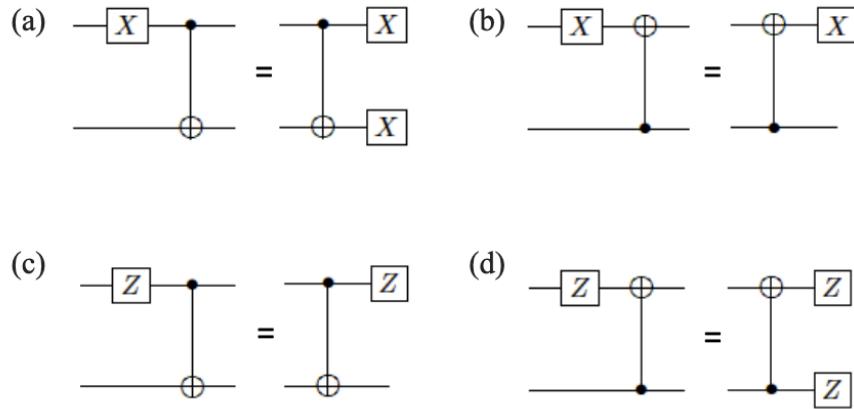


Fig. 3. Propagation of X and Z errors through the CNOT gates.

Figure 1:

144 Shor's fault-tolerant logical measurements are implemented by applying transversal gates between data
 145 qubits and ancilla GHZ (cat) states. The procedure requires multiple rounds, where each GHZ ancilla
 146 interacts transversally with the data qubits and is then measured in the X (Z) basis, corresponding to initial
 147 input state $|\bar{+}\rangle$ ($|\bar{0}\rangle$).

148 These repeated measurements allow one to perform majority voting on the syndrome outcomes, thereby
 149 suppressing the effect of measurement errors. Fault tolerance requires that errors arising at any stage do not
 150 propagate uncontrollably to the data qubits. Figure 4 illustrates this process.

151 One potential issue is that ancilla faults during syndrome extraction can propagate in such a way that
 152 errors mimic measurement errors. To avoid this mixing, each round of syndrome extraction must itself be
 153 implemented fault-tolerantly. By performing fault tolerant error correction in each state, a single fault can
 154 only corrupt the outcome and then be fixed during that round. This guarantees that majority voting across
 155 repeated rounds of cat state measurements produces valid syndrome information.

Shor's method requires repetitions of each stage to alleviate an probability

$$P = \frac{1}{2} - (1 - 2p)^d = \frac{1}{2} - \Delta$$

156 of logical error occurs, where p is the single qubit error probability and d is the circuit depth of each stage.
 157 The majority vote requires $O(e^{2d})$ repetitions.

158 In Steane method, logical measurement is performed by preparing an ancilla block encoded in the same
 159 CSS code (e.g., $|0_L\rangle = \frac{1}{\sqrt{2}}(|++\rangle + |--\rangle)^{\otimes 3}$ or $|+_L\rangle$ for the Shor code.) and coupling it to the data
 160 block with transversal CNOTs, realizing

$$\text{CNOT}^{\otimes n} = \overline{\text{CNOT}}^{\otimes k}$$

161 for an $[n, k, d]$ CSS code. This is in fact mapping the measurement outcome from data code block to ancilla
 162 code block:

163 Let the data block be $|\psi\rangle = \alpha|0_L\rangle + \beta|1_L\rangle$ and the ancilla be $|0_L\rangle$. After transversal CNOTs: $|\psi\rangle|0_L\rangle \mapsto$
 164 $\alpha|0_L\rangle|0_L\rangle + \beta|1_L\rangle|1_L\rangle$. Measuring the ancilla block in the Z basis reveals the eigenvalue of Z_L on the data
 165 block, while collapsing it into $|0_L\rangle$ or $|1_L\rangle$ accordingly.

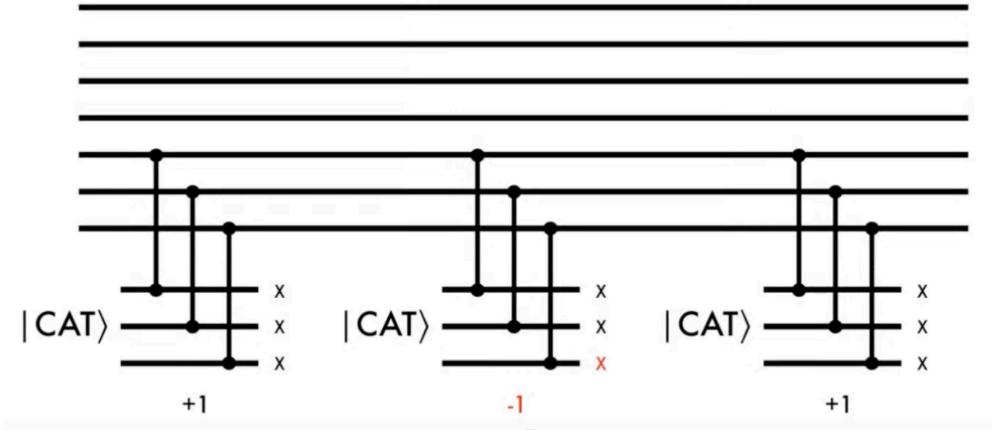


Figure 2:

Unlike Shor's cat-state method, which measures stabilizers one by one, Steane's method allows all stabilizers of one error type (either X -type or Z -type) to be extracted in a single round. One can imagine that once an measurement error occurs in Steane's parity checks, more parity checks outcome can be used to infer the syndromes compared with Shor's method that require more qubits for multiple stages measurements for one set of syndrome in each stage. The logical error rate yields

$$P = \mathcal{O} \left(p^{\frac{d-1}{2}} \right)$$

166 . Here, d is the code distance.

167 For an general $\llbracket n, k, d \rrbracket$ code, it is easily to generalize the ancilla states to $|0\rangle = \overline{|+_1 \dots 0_i \dots +_k\rangle}$ for a Z
 168 type measurements, since $X_j|+_k\rangle$ leaves no change of the state, the measurement of Z_i will only extract
 169 information from i qubits (the state $|+_k\rangle$, which treats all Z measurement outcomes on an equal footing).
 170 $|0_i\rangle$ is just the ancilla state (measurement state) for Z_i stabilizers. Dimensions of $|0_i\rangle$ state is the weight of
 171 Z_i stabilizers. Here, we have noted that for even weight of stabilizers, they are related by local Hadamard
 172 gate, called Clifford-equivalent. Also, the choice of codewords are designed by both logical operators and
 173 stabilizers. A density matrix for logical state of one-qubit encoding can be found as U

$$|0_L\rangle\langle 0_L| = \frac{1}{2^n} (I + \overline{Z}_L) \prod_j (I + S_j).$$

174 For $|1_L\rangle$, one can change the plus sign to minus sign.

175 Problem in Steane code could be ancilla states $|0_L\rangle$ preparation [12]. It can be comprised of a non-fault
 176 tolerant preapation process combined with a verification stage. The verification stage Fig. 3 requires and
 177 additional ancilla qubit to flag a successful preparation, like post-process. The whole process then become
 178 fault-tolerant but with successful rate e^{-np} , with n being number of gates and p the succesful probability
 179 of each gate. For state other than $|0_L\rangle$ can be prepared combined with Clifford operations. Noted that
 180 apart from Clifford operations, magic state injection ($T|+\rangle$) is also required to fulfill the universal quantum
 181 operations. Similarly, by state distillation or code concatenation, desired ancilla qubit states can be obtained
 182 but with large overheads [20].

183 Another trick for ancilla states creation in Steane code is by performing X -type measurements. It seems
 184 like we can create the codewords $|0_L\rangle$ by following a state projection from logical operators and stabilizers:

$$\frac{1}{2^J} (I + \overline{Z}_L) \prod_j^J (I + S_j).$$

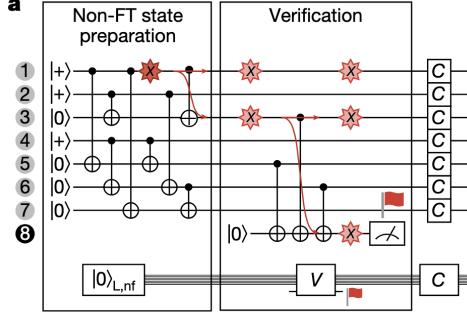


Figure 3:

185 If we perform X type measurements (mathematically described by above formula, while S_j being Z type
 186 measurements can be trivial), the mathematical description of this projecting process can be:

$$|0_L\rangle = \frac{1}{2^{J/2}} \cancel{(I + Z_L)} \prod_{j \in X\text{-type}} (I + S_j) |0^{\otimes n}\rangle + \cancel{\frac{1}{2^{J/2}} (I + Z_L)} \prod_{j \in Z\text{-type}} (I + S_j) |0^{\otimes n}\rangle. \quad (1)$$

187 Noted that Z type measurements and logical Z_L measurement act trivially on $|0^{\otimes n}\rangle$ (They are already
 188 in stabilizer group or commute with stabilizers). The formula requires projective operator which could only
 189 be done unitarily. More explicity, an arbitrary state can be written as combination of projective states with
 190 different observables, hence we could write $|0^{\otimes n}\rangle = \frac{I+S_j}{2}|0^{\otimes n}\rangle + \frac{I-S_j}{2}|0^{\otimes n}\rangle$. This also demonstrate that
 191 Eq. 1 have implicitly selectively choose the projective states $\frac{I+S_j}{2}|0^{\otimes n}\rangle$ with some probability. For Steane
 192 code, this probability is $(\frac{1}{2})^3 = \frac{1}{8}$ for three consecutive projecting process.

193 From $|0^{\otimes n}\rangle = \frac{I+S_j}{2}|0^{\otimes n}\rangle + \frac{I-S_j}{2}|0^{\otimes n}\rangle$, we could infer that the correponding error correction of Z-type
 194 could fix the problem when projecting into wrong states. Hence, the process require further fault-tolerant
 195 error correction (FTEC) following the stabilzers measurement to deterministically generate logical $|0_L\rangle$
 196 state(Above are my current understanding which may not correspond to what paper really trying to convey.).
 197 There is no need post-selection for Steane's ancilla qubit preparation as claimeed in the video for logical
 198 qubits number $k = 1$. Also, for $k > 1$ the process can be used to generate $|0^{\otimes k}\rangle$ (all Z measurement at
 199 once) but not $|+_1...0_i...+_k\rangle$ (If we want particular Z_i measurement). The reason is that $|+_1...0_i...+_k\rangle$ are
 200 not easily prepared anymore. This may make the whole preparation process as hard as directly measuring
 201 logical operators in data block.

202 A natural thoughts then will be can a new choice of ancilla code such that it can achieve a LDPC
 203 measurement on particular logical qubit. The next question is, is there any other choices of ancilla code to
 204 achieve non-postselection, no repition like Steane's method for an $[[m,1,d]]$ (Steane ancilla code: $[[n,1,d]]$ or
 205 $[[n, k ,d]]$)ancilla code. Here, the speaker aims to build a new code that could perform with $m < d$ that
 206 could be more resource freindly.

207 The speaker introduced a measurement process called *homomorphic measurement*. A toric code is an

$$[[n, k, d]] = [[2L^2, 2, L]]$$

208 defined on a torus, which can be represented as a square sheet with periodic boundary conditions. The
 209 stabilizers all commute, and the corresponding logical operators are shown in Fig. 10. The horizontal loops
 210 \bar{X}_1, \bar{Z}_2 and the vertical loops \bar{Z}_1, \bar{X}_2 correspond to logical operators that wrap around the torus in the
 211 horizontal or vertical directions.

212 3.4 Binary vector spaces

213 They construct the homomorphism between data qubits and the ancilla qubits by using CSS codes chain
 214 complexes.

215 An $r \times n$ binary matrix defines a linear map

$$H : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^r,$$

216 where $\mathbb{F}_2 = \{0, 1\}$ with addition and multiplication modulo 2. The transpose H^T is the $n \times r$ matrix with
217 rows and columns swapped. The kernel (null space) is

$$\ker(H) = \{v \in \mathbb{F}_2^n : Hv = 0\},$$

218 the image (column space) is

$$\text{im}(H) = \{Hv : v \in \mathbb{F}_2^n\},$$

219 and the row space is the span of the rows of H , denoted $\text{rs}(H)$. Note that $\dim(\text{im}(H)) = \dim(\text{rs}(H)) =$
220 $\text{rank}(H)$.

221 Given a finite set S , the vector space $\mathbb{F}_2[S]$ consists of all formal binary sums of elements in S ,

$$v = \sum_{e \in S} v_e e, \quad v_e \in \mathbb{F}_2,$$

222 which can be naturally identified with subsets of S (element e is present if $v_e = 1$). If $H : \mathbb{F}_2[A] \rightarrow \mathbb{F}_2[B]$,
223 then the transpose defines a map $H^T : \mathbb{F}_2[B] \rightarrow \mathbb{F}_2[A]$ under the corresponding bases.

224 As an example, consider

$$H = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

225 The image is spanned by the columns $(1, 0)^T$, $(0, 1)^T$, and $(1, 1)^T$, which generate all of \mathbb{F}_2^2 . The row space
226 is spanned by $(1, 0, 1)$ and $(0, 1, 1)$, giving the subspace

$$\{(0, 0, 0), (1, 0, 1), (0, 1, 1), (1, 1, 0)\} \subseteq \mathbb{F}_2^3.$$

227 In the language of quantum error correction, the row space $\text{rs}(H)$ often corresponds to the stabilizer
228 group (constraints on codewords), the image $\text{im}(H)$ corresponds to possible syndrome outcomes, and the
229 kernel $\ker(H)$ corresponds to valid codewords with no detected error.

230 A CSS code with stabilizer X type and Z type will have corresponding stabilizer group isomorphic to
231 $\text{rs}(H_X)$ and $\text{rs}(H_Z)$.

232 The quantum code can be described using two families of Pauli stabilizers. The X-type stabilizer group
233 corresponds to parity checks that involve X operators, and it is isomorphic to the row space of H_X . Similarly,
234 the Z-type stabilizer group corresponds to parity checks that involve Z operators, and it is isomorphic to
235 the row space of H_Z .

236 The X-type logical operators are elements of $\ker(H_Z)$ (like centralizer), meaning they commute with all
237 Z-type checks and therefore preserve the Z-stabilizer constraints. Likewise, the Z-type logical operators are
238 elements of $\ker(H_X)$, since they commute with all X-type checks since a logical operator will stay in the
239 codespace.

240 The number of encoded logical qubits is the number of independent logical degrees of freedom that remain
241 after imposing all stabilizer constraints:

$$k = \dim(\ker(H_X)/\text{rs}(H_Z)) = \dim(\ker(H_Z)/\text{rs}(H_X))$$

242 (quotient subgroup: The elements of the quotient space V/W are the cosets of W . Each coset is of the
243 form $v + W$ for some $v, w \in V$. Algebraically, forming the quotient space V/W , V/W means we treat all
244 vectors that differ by an element of W as equivalent. Topologically, V/W is like shrinking W space into a
245 point. It is also like finding logical qubits dimension using $\dim(2^n/2^{n-k}) = k$. This formula says that logical
246 qubits live in the space of operators that preserve one type of stabilizer (the kernel) but are not redundant
247 with the other type (the row space). The X distance d_X measures how resilient the code is against bit-flip

²⁴⁸ (X -type) errors: it is the minimum number of qubits that must be flipped to implement a nontrivial logical
²⁴⁹ X operation. Formally,

$$d_X := \min\{ |c| : c \in \ker(H_Z) \setminus \text{rs}(H_X) \}.$$

²⁵⁰ Similarly, the Z distance d_Z quantifies protection against phase-flip (Z -type) errors:

$$d_Z := \min\{ |c| : c \in \ker(H_X) \setminus \text{rs}(H_Z) \}.$$

²⁵¹ Finally, the overall *code distance* is

$$d = \min\{d_X, d_Z\},$$

²⁵² which sets the maximum number of arbitrary single-qubit errors the code can reliably detect and correct.

²⁵³ Physically, the larger the distance, the more robust the code is against noise.

²⁵⁴ Quantum error correction uses this framework because stabilizers operators naturally form abelian groups
²⁵⁵ modulo phases (self-commute).

²⁵⁶ 3.5 Algebraic Topology

²⁵⁷ Notes from the lecture [3], the **2-dimensional disk** is defined as

$$D^2 = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 \leq 1\}.$$

²⁵⁸ It consists of all points in the plane whose distance from the origin is less than or equal to 1. **Interior and boundary**

$$\text{Int}(D^2) = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 < 1\}, \quad \partial D^2 = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 = 1\} = S^1. (\partial D^n = S^{n-1})$$

²⁶⁰ **Topological Meaning** In a CW complex, D^2 serves as a **2-cell**. Attaching a 2-cell means gluing a copy of
²⁶¹ D^2 along its boundary S^1 via a continuous map:

$$f : S^1 \rightarrow X^1.$$

²⁶² For example:

- ²⁶³ • S^2 is formed by attaching one D^2 to a point ($X^1 = X^0$ here, one D^0 zero D^1 , one $D^2]$, $\chi(S^2) = 2$ (χ defined below) .
- ²⁶⁵ • A torus T^2 is formed by attaching D^2 along a loop that winds in two directions (X^0 : a point, X^1 add
²⁶⁶ two D^1 lines, $f : \partial D^1 = S^0 \rightarrow X^0$. X^2 : add a D^2 two dimensional disk $f : S^1 \rightarrow X^1$) ($X^2 : ab^{-1}a^{-1}b$,
²⁶⁷ the direction of loop are glued will result in different shape, if $X^2 : ab^{-1}ab$ is a Klein bottle). $\chi(T^2) = 0$

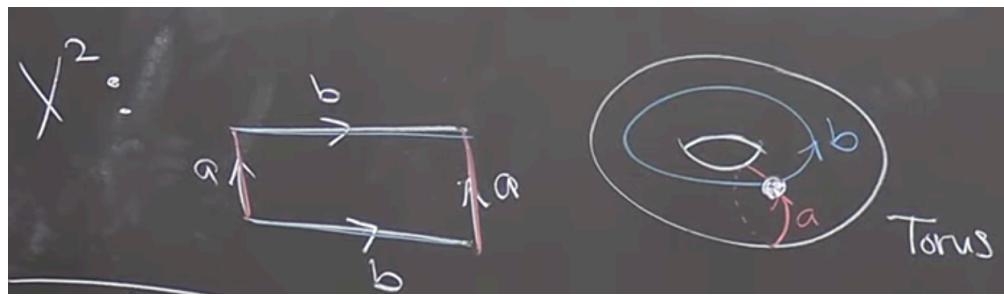


Figure 4: S^1 for torus. X^1 : add

268 **Generalization** The n -dimensional disk is

$$D^n = \{(x_1, \dots, x_n) \in \mathbb{R}^n \mid x_1^2 + \dots + x_n^2 \leq 1\},$$

$$f : S^{n-1} \rightarrow X^{n-1}.$$

270 **Euler characteristic** Vertices $D^0 = V$, Edges $D^1 = E$, Faces $D^2 = F, \dots$

$$\chi = \#\text{even dim}(D) - \#\text{odd dim}(D)$$

$$\chi(T^2) = V - E + F = 1 - E + 1 = 0 \Rightarrow E = 2$$

271 or a torus can be build from $V = 4$, $E = 8$, $F = 4$ and similar goes for S^2 but with χ fixed.

272 **Product and homology.** Noted that D^n is contractible and S^n are not.
273 **homotopy** \simeq

Spaces (X, Y)	Relationship	Intuition
D^n and a point *	$D^n \simeq *$	A disk can be shrunk to a point (contractible).
S^1 and a circle-shaped wire loop	$S^1 \simeq$ any loop	All circles have the same homotopy type
S^1 and a torus (T^2)	Not homotopy equivalent	A torus has more “holes.”
\mathbb{R}^n and a point	$\mathbb{R}^n \simeq *$	Can contract the entire space to a point.
A hollow cylinder and a circle	$S^1 \times I \simeq S^1$	The cylinder retracts onto its circular core.

275 The torus (solid torus: $D^2 \times S^1$) is defined as $T^2 = S^1 \times S^1$ and its *fundamental group* is $\pi_1(T^2) \cong \mathbb{Z} \times \mathbb{Z}$.
276 In contrast, for the circle we have $\pi_1(S^1) \cong \mathbb{Z}$. Since the integer group \mathbb{Z} is not isomorphic to the product
277 group $\mathbb{Z} \times \mathbb{Z}$, it follows that $T^2 \not\simeq S^1$. Geometrically, if one tries to shrink the torus T^2 into a circle S^1 , one
278 must collapse or “break” one of the gluing directions that form T^2 . Since this cannot be done continuously
279 without tearing the surface, T^2 and S^1 are not homotopy equivalent. $D^1 \times D^2$: a solid cylinder (sphere)
280 Some identities:

$$D^n \times D^m = D^{n+m}$$

$$\partial(X \times Y) = (\partial X \times Y) \cup (X \times \partial Y)$$

281 \cup is called union. For example, calculate $\partial(D^2 \times [1, 0]) = (\partial D^2 \times [1, 0]) \cup (D^2 \times \partial[1, 0]) =$
282 $(S^1 \times [1, 0]) + D^2 \times \{0, 1\}$ It is exactly the surface of the cylinder. Or simply, $\partial(D^2 \times [1, 0]) = \partial D^3 // = S^2$
283 So calculate $\partial(S^1 \times S^1 \times [1, 0]) = S^1 \times S^1 \times \{0, 1\}$ is two copies of torus surface.

284 Another example: $S^3 = \partial(D^4) = \partial(D^2 \times D^2) = S^1 \times D^2 \cup D^2 \times S^1$ (two tori formed by looping around
285 different directions, pictorially, draw S^1 first for first qubit and then draw D^2 connected on S^1 similar for
286 second torus but with opposite order.). Union can be think of gluing, hence gluing two tori is S^3 .

287 Examples of **Quotients** in topology: $D^1/S^0 = S^1$, $D^2/S^1 = S^2$, $S^2/S^1 = S^1 \vee S^1$, \vee (pronounce:
288 wedge), also examples in Fig. 3

289 **Homology** group is used to describe

290 3.5.1 Homology Groups

291 Vector spaces over \mathbb{F}_2 are abelian groups C_i under addition. Boundary operators ∂_i are group homomorphisms
292 (Like in toric code, logical Z_L is noncontractible loops around the torus.). Groups here are in topological
293 sense not the same as Stabilizers group in physical Pauli sense. A chain complex is just a sequence of abelian
294 groups with compatible homomorphisms, typically written as

$$C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0, \quad \text{with } \partial_1 \circ \partial_2 = 0.$$

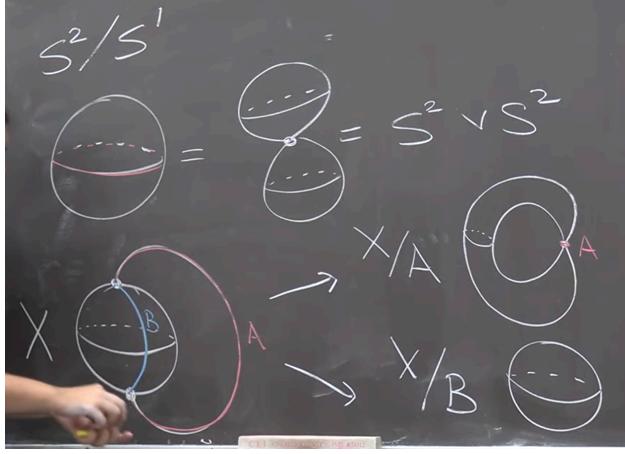


Figure 5:

295 In CSS codes, $H_X = \partial_1$ and $H_Z^T = \partial_2$ naturally satisfy $H_X H_Z^T = 0$ is the stabilizers. The above
 296 **homology group** is used to describe data qubits C_1 and logical operators ($\ker(\partial_1)/\text{im}(\partial_2)$)
 297 Logical operators are homology classes H_i . They are cycles Z_n (commute with stabilizers) but not
 298 boundaries B_n themselves (not product of stabilizers). Mathematically:

$$H_n = Z_n / B_n \quad (2)$$

$$Z_n := \ker \partial_n := \{ c \in C_n \mid \partial_n(c) = 0 \} \quad (3)$$

$$B_n := \text{im } \partial_{n+1} := \{ \partial_{n+1}(c) \mid c \in C_{n+1} \} \quad (4)$$

299 This correspond to $\dim(\ker(\partial_n)/\text{im}(\partial_{n+1})) = \dim(\ker(H_Z)/\text{rs}(H_x)) = k$. The algebra links with Fig. 10 toric
 300 code. The toric code can be expressed as the chain complex $C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0$, where qubits live on edges
 301 (C_1), X -stabilizers are associated with vertices (C_0), and Z -stabilizers with faces (C_2). The logical operators
 302 are characterized by the first homology group

$$H_1 = \ker(\partial_1) / \text{im}(\partial_2) \cong \mathbb{Z}_2 \oplus \mathbb{Z}_2 = (0,0), (1,0), (0,1), (1,1),$$

303 which corresponds to the two nontrivial loops around the torus that encode the logical qubits, while torus
 304 requires two loops to decribe its topology. In general,

$$\ker(H_Z) / \text{im}(H_X)$$

305 corresponds to the X -type logical operators, while

$$\ker(H_X) / \text{im}(H_Z)$$

306 corresponds to the Z -type logical operators. Pictorially, one can imagine that all closed loops of errors on
 307 qubits in Fig. 10 lie in $\ker(\partial_{H_Z})$, but many of them can also be formed as products of X stabilizers. The
 308 only exceptions are loops that connect opposite edges (loop around) of the torus, which give nontrivial errors
 309 that cannot be detected and by design act as logical Z operators.

310 3.6 Homomorphic logical measurements

311 As we have elaborated on Shor and Steane measurement downsides and limitations, here we dorectly go
 312 to the arthor main points, homomorphic logical measurements. They are trying to find a new code $[[m, 1
 313 ,d]]$ that could unifying or improve before mentioned downsides. The process first start from preparing 1.
 314 preparing ancilla in $|0^{\otimes k}\rangle$ 2. perform interaction Γ between ancilla and data block. 3. measured Z basis on
 315 ancilla block.

316 Data–Ancilla Interaction

317 Applying the homomorphism for CSS codes into their ancilla code construction by considering possible
 318 interaction between data-ancilla interaction (typically utilising similar mathematical but applying on different
 319 purposes.):

320 We have two CSS codes: - Data: (H_X, H_Z) of length n , - Ancilla: (H'_X, H'_Z) of length m . Before interaction,
 321 stabilizer groups are written as

$$T_Z = \text{rs} \begin{pmatrix} H_Z & 0 \\ 0 & H'_Z \end{pmatrix}, \quad T_X = \text{rs} \begin{pmatrix} H_X & 0 \\ 0 & H'_X \end{pmatrix}.$$

322 After Interaction (Γ a gate matrix $\Gamma : \mathbb{F}_2^m \rightarrow \mathbb{F}_2^n$ (CNOTs)), stabilizer groups can be written as

$$T'_Z = \text{rs} \begin{pmatrix} H_Z & 0 \\ H'_Z \Gamma^T & H'_Z \end{pmatrix}, \quad T'_X = \text{rs} \begin{pmatrix} H_X & H_X \Gamma \\ 0 & H'_X \end{pmatrix}.$$

323 To explain T'_Z further, it is like the outcome of H'_Z on ancilla qubits, is determined not only by the state
 324 initial state $|0_L\rangle$ lie in ancilla block but also $H'_Z \Gamma^T$ when performing interaction, which is like a different
 325 mapping other than Steane style, from my understanding, Steane style measurement follows $H'_Z \Gamma^T = H'_Z$
 326 (Since Γ here is like identity for transversal gates in Steane measurement) and also $H'_Z = H_Z$ since they
 327 are using same logical codewords, hence same stabilizers. Just like the author mentioned, for Shor's style
 328 measurment, $H'_Z \neq H_Z$ since H_Z should correspond to cat states stabilizers (1D).

329 The role interchange between target and controlled of X type and Z type errors can be explained by the
 330 error propagation shown in Fig. 6.

331 We also required conditions such $T'_Z = T_Z$, $T'_X = T_X$, i.e.

$$\text{rs}(H'_Z \Gamma^T) \subseteq \text{rs}(H_Z), \quad \text{rs}(H_X \Gamma) \subseteq \text{rs}(H'_X).$$

332 This ensures the interaction preserves the stabilizer groups.

333 **Definition (Homomorphic gadget).** An $[[n, k, d]]$ homomorphic gadget (H'_X, H'_Z, Γ) for an $[[n, k, d]]$
 334 CSS code (H_X, H_Z) consists of: (i) an ancilla $[[m, k', d']]$ CSS code with checks (H'_X, H'_Z) ; (ii) a gate matrix
 335 $\Gamma : \mathbb{F}_2^m \rightarrow \mathbb{F}_2^n$; such that

$$\text{rs}(H'_Z \Gamma^T) \subseteq \text{rs}(H_Z), \quad \text{rs}(H_X \Gamma) \subseteq \text{rs}(H'_X). \quad (5)$$

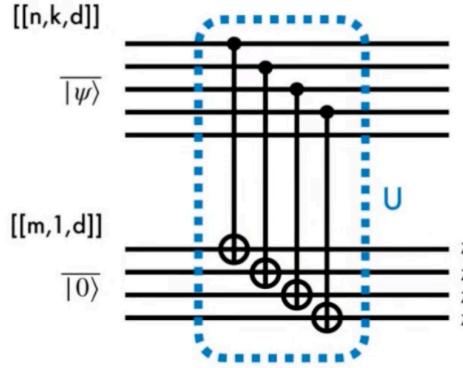


Figure 6:

336 To summarize, a stabilizer element (in fact also logical error) $v \in \ker H'_Z$ are transformed as $\Gamma v \oplus v$, acted
 337 on data block (Γv) and ancilla block (v), since ancilla state is prepared in logical $|0^{\otimes k}\rangle$, the outcome will be
 338 Γv . There are two cases: where $v \in \text{rs}(H'_Z)$ or $v \notin \text{rs}(H'_Z)$, the former under homomorphic gadget setting
 339 will preserve the structure of $v \in \text{rs}(H'_Z)$ and act as a X error detection for data block. The latter are in

fact mapping logical Z operation into ancilla block. As mentioned, the outcome will be Γv (it is measured in ancilla block but in fact bring based on data block information. One can simply assume a vector acting by matrix T'_Z to see this) which will isomorphic to $\Gamma \ker(H_X)$ (seems might encounter vector space outside $\Gamma \ker(H_X)$)

3.7 Homomorphic measurements on surface codes

Surface codes are defined as cellulations of a manifold $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, where the boundary maps $\partial_2 : \mathcal{F} \rightarrow \mathcal{E}$ and $\partial_1 : \mathcal{E} \rightarrow \mathcal{V}$ obey the CSS code condition $\partial_1 \partial_2 = 0$. (Can LDPC CSS codes, such as hypergraph product codes, have different homomorphic gadgets?) Linear maps $\gamma : \mathcal{A} \rightarrow \mathcal{D}$ connect the ancilla and data surface codes. In fact, the gate matrix is given by $\Gamma = \gamma_1$ in the paper, where $\gamma_1 : \mathcal{E}' \rightarrow \mathcal{E}$ is the linear map between qubits. The data and ancilla surface codes are defined respectively as $\mathcal{D} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, and $\mathcal{A} = (\mathcal{V}', \mathcal{E}', \mathcal{F}')$. Explicitly, the relation between data block and ancilla block:

$$\begin{array}{ccccc} \mathbb{F}_2[\mathcal{F}'] & \xrightarrow{\partial'_2} & \mathbb{F}_2[\mathcal{E}'] & \xrightarrow{\partial'_1} & \mathbb{F}_2[\mathcal{V}'] \\ \downarrow \gamma_2 & & \downarrow \gamma_1 & & \downarrow \gamma_0 \\ \mathbb{F}_2[\mathcal{F}] & \xrightarrow{\partial_2} & \mathbb{F}_2[\mathcal{E}] & \xrightarrow{\partial_1} & \mathbb{F}_2[\mathcal{V}] \end{array}$$

The above relation naturally gives homomorphic gadget conditions shown in Eq. 5, as $\gamma_1 \partial'_2 = \partial_2 \gamma_2 \subseteq \partial_2$ and $\gamma_0 \partial'_1 = \partial'_1 \gamma_1 \subseteq \partial_1$. The paper seems like weakening the global homeomorphism constraints of a usual linear map $\Gamma(\gamma_i)$ such as Steane or Shor to local homeomorphism. This generalization gives more degree of freedom to represent logical operators to a single non-contractable loop in a new manifold. This generalization do not preserve transversal gates, as we can see that γ_i local homeomorphism, or covering spaces can be many-to-one linear maps. There are also certain boundaries for manifold M , with two rough boundaries and two smooth boundaries is the planar surface codes [4].

The paper constructs homomorphic gadgets into two categories: **subspaces of data code space** \mathcal{D} and **covering space** of \mathcal{D} . For the first one, it is natural that homomorphic gadget can be constructed given Γ is injective (one-to-one, hence transversal and fault tolerant. $\mathcal{A}(\mathcal{V}', \mathcal{E}', \mathcal{F}') \in \mathcal{D}(\mathcal{V}, \mathcal{E}, \mathcal{F})$). Shor code can be thought of as $A = l \subseteq (\mathcal{V}, \mathcal{E})$ and l loops not intersecting (loops here generally mean logical operators, so not restricted on toric code loops, if loop intersects, it could involve two logical operators which is not in cat state gadget.), with $F = \emptyset$ and this indicates repetition code $0_L = \frac{1}{\sqrt{2}}(|+++...+|---...)$ will have only X stabilizers, for repetition code of Z stabilizers, one use T'_X which interchange the controlled and target qubits between data and ancilla qubits.

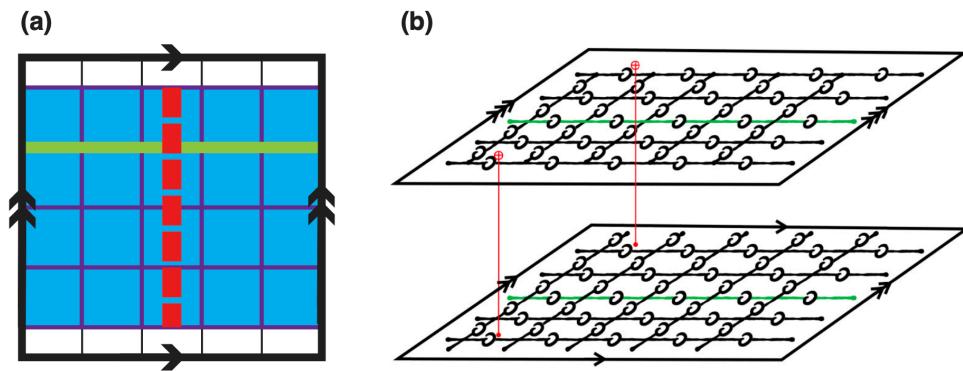


Figure 7:

Fig. 7 (a) describes a toric code \mathcal{D} and the blue region describes surface code with smooth boundaries. Red line connecting two boundaries are then logical X (or by seeing red lines crossing sets of Z stabilizers.).

368 The logical operator dimension of \mathcal{A} are then reduced from four (complete mapping correspond to Steane
 369 measurement) to two. This ensures simpler preparation of ancilla states as mentioned in 3.3, which is also a
 370 problem associated with Steane measurement if utilising a complete mapping between data (might involve
 371 two or more logicals) and ancilla block.

372 Here we move on to homomorphic gadgets from **covering spaces**. Emphasizing the motivation again,
 373 if we want to perform a single-shot nondestructive logical CSS measurements on multiple logical operators
 374 (ancilla block), then the direct mapping such as Steane code or $A \in l$ inevitably support two logicals degree
 375 of freedom but with overlapped qubits, furthermore, multiple logical qubits ancilla is hard to prepare. The
 376 idea is to unfold the manifold to make logical operators uniquely represented by a non-intersecting loop in
 377 ancilla sheets. This resolves all of the problems mentioned.

378 **Groups acting on spaces:** The infinite *simply connected* covering space U (for example, \mathbb{R}^2) is equipped
 379 with a regular tiling (cellulation: divided into cells like vertices, edges, faces, $[i, i+1] \times [j, j+1]$ for $(i, j) \in \mathbb{Z}^2$).
 380 A group G of symmetries (leave the grid-structure intact), such as translations or rotations, acts on U , and
 381 for each point $u \in U$, its orbit $Gu = \{g(u) \mid g \in G\}$ consists of all symmetry-related copies of u (all Gu
 382 collapses to one point.). The orbit space U/G is the corresponding quotient manifold (for instance, a torus),
 383 and the quotient map (continuous and open) $p_G : U \rightarrow U/G$ sends each point u to its orbit Gu . The resulting
 384 manifold $M = U/G$ is the compact surface on which the surface code is defined. Because each element of G
 385 preserves the tiling of U , the quotient map p_G induces a cellulation of M ; that is, every k -cell in U maps to
 386 a k -cell in M , preserving the lattice structure.

387 As discussed in Sec. 3.5 and illustrated in Fig. 4, one can regard the **first example: torus** as the
 388 quotient of the real plane $\mathcal{U} = \mathbb{R}^2$ by the integer translation group $G \cong \mathbb{Z} \times \mathbb{Z}$
 389 (translations $t_{r,s}(x, y) : (x, y) \rightarrow (x + dr, y + ds)$ for an $[2d^2, 2, d]$ toric code). Intuitively, this corresponds to
 390 identifying points that differ by integer shifts, i.e., taking 0 and 1 as the same point in each direction. The
 391 quotient $\mathbb{R}^2 / (\mathbb{Z} \times \mathbb{Z})$ can thus be represented by the unit square $[0, 1] \times [0, 1]$, where opposite edges—labeled
 392 a and b in Fig. 4—are glued together to form the torus topologically. Because the torus is constructed as
 393 this quotient, its *fundamental group* is isomorphic to the translation group itself,

$$\pi_1(T^2) \cong \mathbb{Z} \times \mathbb{Z},$$

394 with each generator corresponding to one of the two noncontractible loops along the a and b directions.

395 We can also consider **second example: hyperbolic surface codes**, with the universal $\mathcal{U} = \mathbb{H}^2$ which
 396 are defined on regular tilings characterized by a Schläfli symbol $\{r, s\}$ (note that this is unrelated to the
 397 integer coordinates (r, s) used earlier). Here, r indicates that each face (tile) is a regular polygon with r
 398 sides, and s means that s such faces meet at each vertex. The pair $\{r, s\}$ determines both the curvature of
 399 the surface and the stabilizer structure: if $(r - 2)(s - 2) < 4$, the surface is spherical; if $(r - 2)(s - 2) = 4$,
 400 it is Euclidean (flat, as in the toric code); and if $(r - 2)(s - 2) > 4$, it is hyperbolic. In the code, each
 401 Z -type stabilizer acts on r qubits (around a face), and each X -type stabilizer acts on s qubits (around a
 402 vertex). The **Coxeter group** $G_{r,s}$ preserve the tiling structure. Group G is chosen as the normal subgroup
 403 of $G_{r,s}$ (like relation between Pauli group and Clifford group). The parameters $\llbracket n, k, d \rrbracket$ satisfy $k = O(n)$
 404 and $d = O(\log n)$.

405 (This passage formalizes how one can form a quotient manifold \mathcal{U}/G by identifying points under a group of
 406 **local homeomorphism**, in a way that preserves the cellulation and thus the qubit and stabilizer structure
 407 of the original topological code.)

408 The *image* of N_u , $g(N_u)$ is the set of all points in \mathcal{U}/G that are reached when applying the map p_G to every
 409 point in $N_u : p_G(N_u) = \{p_G(x) \mid x \in N_u\}$. Therefore, if N_u is a small open patch around u in the original
 410 space \mathcal{U} , then the set $N_v := p_G(N_u)$ is the corresponding small open patch and disjoint in the quotient space
 411 \mathcal{U}/G . N_u and $g(N_u)$ are homeomorphic. Also, no nontrivial $g(u) = u$ (Not fixed points mean mapping all
 412 of the points to N_v , bijective: injective and surjective)

413 **Third example: $\llbracket 2d^2, 2, d \rrbracket$ toric code.** if we choose U_u for any $u = (x, y) \in \mathcal{U} = \mathbb{R}^2$

414 The *lifting property* is the key topological feature they rely on, since it allows any logical operator—
 415 represented by a noncontractible loop on the base surface to be lifted to a non-self-intersecting path on a
 416 multi-sheeted covering manifold. The loop on \mathcal{U} starts at u and ends at some translated copy of $g(u)$.

417 **Fourth example: $[[2d^2, 2, d]]$ toric code.** Lifting a horizontal loop l on \mathcal{U}/G to \tilde{l} . Mathematically,
 418 denoted as $g(u) = t_{10}(u)$, where $u = (0, 0) \in \mathcal{U}$. One can imagine logical operator correspond to \mathcal{U}/G is like
 419 viewing $(0, y)$ and (d, y) as same point. Also, \tilde{l} is guaranteed to be a loop if and only if l is contractible on
 420 \mathcal{U}/G given U is simply connected.

421 Consider another covering map $p_G^H : \mathcal{U}/H \rightarrow \mathcal{U}/G$ defined as $p_G^H H(u) = G(u)$. When $H = \langle t_{1,0} \rangle$
 422 (horizontal translations), the intermediate covering space \mathcal{U}/H is an infinite *cylinder*, obtained by identifying
 423 points along the horizontal direction of the universal cover $U = \mathbb{R}^2$. The base space \mathcal{U}/H , where $G =$
 424 $\langle t_{1,0}, t_{0,1} \rangle$, is the *torus*, obtained by identifying both horizontal and vertical directions. On the torus \mathcal{U}/H ,
 425 the horizontal and vertical logical loops correspond to $t_{1,0}$ and $t_{0,1}$, respectively. When lifted to the cylinder
 426 \mathcal{U}/H , the horizontal loop remains closed since $t_{1,0} \in H$, while the vertical loop becomes an open segment as
 427 $t_{0,1} \notin H$. This picturizes the general relation

$$g \in H \iff \text{the lifted loop } \ell \text{ is closed on } \mathcal{U}/H.$$

428 **Fifth example: $[[2d^2, 2, d]]$ toric code** is same as the previous example for relation
 429 $g \in H \iff \text{the lifted loop } \ell \text{ is closed on } \mathcal{U}/H$.

430

431 3.8 Homomorphic gadgets for covering spaces

432 Now we can start to construct homomorphic gadgets for covering spaces. Until now, we make some remarks:
 433 $g(u)$ lives in \mathcal{U} and $p_G(g(u)) = p_G(u)$ on \mathcal{U}/G but $g(u) \neq u$ on \mathcal{U} could
 434 be possible. This could directly be seen $p_G(u) = Gu = \{g(u) \mid g \in G\}$ while p_G represented all possible
 435 $g(u) \in \mathcal{U}$ and collapse to one point in space \mathcal{U}/G by definition.

436 The task is to find $\mathcal{A} \subseteq \tilde{\mathcal{D}} = \mathcal{U}/H$ (where H is defined previously) such that $\mathcal{A} \subset l'$ and satisfies $d_{\mathcal{A}} = d_{\mathcal{D}}$.

437 As discussed before, the subgroup $H \supseteq G$, and $p = p_G^H$ is the covering map from \mathcal{U}/H to \mathcal{U}/G . If we pick
 438 $H = \langle g \rangle$ ($g \in G$), then all the loops are unfolded except the loop l corresponding to the g -translation.

439 Specifically, we map two non-contractible loops to one non-contractible loop in the ancilla block $A \subseteq$
 440 $\tilde{\mathcal{D}} = \mathcal{U}/H$. This ensures that we only have one unique logical operator in \mathcal{U}/H , where H is chosen to be
 441 $\langle t_{1,1} \rangle$ (i.e., no overlapping qubits like in the toric code with different logicals). This unique logical operator
 442 can be designed to represent $\overline{Z_1 Z_2}$, enabling single-shot measurement. Noted that ancilla block \mathcal{A} is chosen
 443 such that $d_A = d_D$ (minimum weight of a nontrivial X logical operator of \mathcal{A} , is the red line part in Fig. 9(c))

444 The induced homomorphic gadget (not necessarily transversal for covering maps) is induced by map
 445 $\gamma := p \circ \tilde{\gamma}$, where $\tilde{\gamma} : \mathcal{A} \rightarrow \tilde{\mathcal{D}}, \gamma : \mathcal{A} \rightarrow \mathcal{D}$

446 One can obtain a clearer physical picture from Fig. 9. Panel (a) shows the data-qubit manifold $\mathcal{U}/G = \mathbb{T}^2$,
 447 where the green loops represent the logical operators $\overline{Z_1 Z_2}$. In panel (b), the corresponding logical loop ℓ
 448 is lifted to the covering space \mathcal{U} , forming a path that connects the two points (x, y) and $(x + d, y + d)$
 449 (connecting two grids), pictorially, imagine two grids collapse into one grid due to translation symmetry,
 450 then the green lines in Fig. 9(b) become Fig. 9(a). Finally, panel (c) illustrates the ancilla-qubit manifold
 451 \mathcal{U}/H , with $H = \langle t_{1,1} \rangle$ which takes the form of a cylinder. The covering spaces correspond to Fig. 9(c) is
 452 shown in Fig. 8. Noted that red line in Fig. 9(c) is logical X operator, when depicting in Fig. 8, it will
 453 become a line connected two smooth boundaries.

454 3.9 Fault tolerance

455 Since there is no transversal mapping for $\gamma := p \circ \tilde{\gamma}$, while homeomorphism between data sheets and ancilla
 456 sheets in standard measurement method is leveraged to local homeomorphism between them. The mapping
 457 between edges then might encounter many-to-one coupling, $\gamma_1^T(e) \in E'$. Even under these correlations, it is
 458 shown it still have fault tolerance with X error $\min\{d_{\mathcal{A}}, d_{\mathcal{D}}\}$.

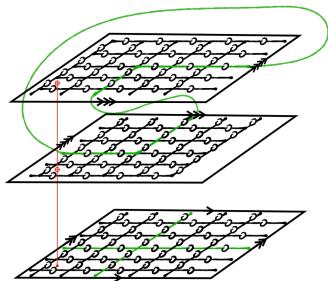


Figure 8:

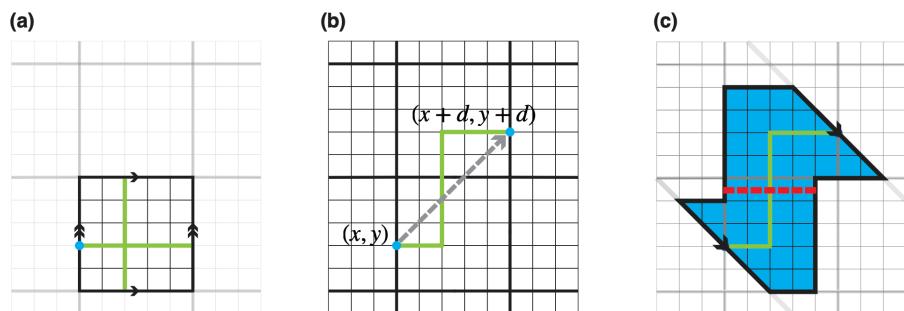


Figure 9: (a) Data-qubit manifold $\mathcal{U}/G = \mathbb{T}^2$, where the green loops represent the logical operators $\overline{Z_1 Z_2}$. (b) The covering space \mathcal{U} , showing the lifted path connecting (x, y) and $(x + d, y + d)$. (c) The ancilla-qubit manifold \mathcal{U}/H , which is topologically equivalent to a cylinder.

459 3.10 Joint measurement

460 Considering two disjoint loops l_1 and l_2 on \mathcal{U}/G , if the manifold \mathcal{M} is path connected, then logical operator
 461 can be $l_1 p l_2 p^{-1}$.

462 For two separate codes, say two ancilla blocks $\mathcal{A}_1, \mathcal{A}_2$, in order to prepare ancilla, one uses a lattice
 463 surgery approach to entangle two blocks from the initial state $|+\rangle_1|+\rangle_2$ into the logical Bell state $|+\rangle_L =$
 $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ by measuring $Z_{\mathcal{A}_1}Z_{\mathcal{A}_2}$ with some surface code A' satisfying $\partial A' = l'_1 \cup l'_2$. (Note that the
 464 results will be either $|+\rangle_L = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ or $|-\rangle_L = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$. One then applies X_1 for correction.)
 465 Just like the layer of ancilla blocks depicted in Fig. 8, $Z_{\mathcal{A}_i}$ can be a closed loop on the boundary, $l'_i \subseteq \partial \mathcal{A}_i$.
 466 After ancilla preparation, one could construct homomorphic gadget (entangle data block and ancilla block)
 467 and perform logical measurement afterwards. Ancilla states can be prepared *offline*.

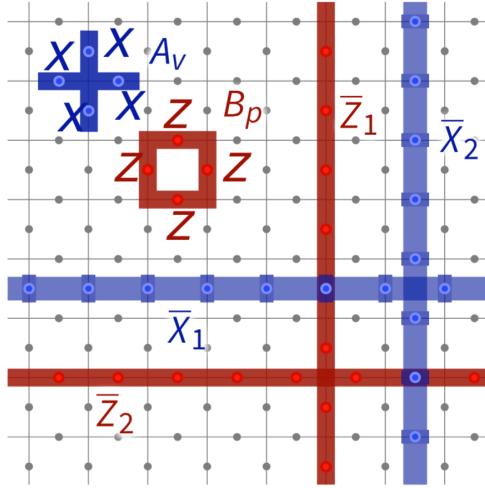


Figure 10:

469 4 Summary

470 They first establish the algebraic conditions under which the interaction matrix $\Gamma = \gamma_1$ between a data
 471 block $\llbracket n, k, d \rrbracket$ and an ancilla block $\llbracket n', k', d' \rrbracket$ preserves the stabilizer structure. Motivated by topological
 472 intuition, the authors represent intersecting logical loops as a single noncontractible loop. This construction
 473 achieves two goals: (1) it enables single-shot measurement of multiple logical operators, and (2) it simplifies
 474 the logical state preparation of the ancilla block.

475 The intuition is formalized through *covering map* between the topological structures (vertices, edges, and
 476 faces) of the data and ancilla codes. Such a map induces corresponding linear mappings between their chain
 477 complexes, ensuring that the homomorphic gadget conditions are automatically satisfied.

478 In this framework, the Steane measurement corresponds to a homeomorphic (one-to-one) chain map,
 479 while the homomorphic logical measurement generalizes it to a *covering map* (locally bijective but globally
 480 many-to-one). This broader formulation naturally supports more general and scalable constructions of logical
 481 measurements across CSS codes.

482 **5 Improved noisy syndrome decoding of quantum ldpc codes with**
 483 **sliding window (Notes on [8])**

484 The paper introduces a sliding-window decoding approach for qLDPC codes (including hypergraph product
 485 codes and lifted product codes) and demonstrates a significant improvement in the logical error rate. While
 486 sliding-window decoding has previously been studied for surface codes, its application to qLDPC codes has
 487 been lacking. The results are compared with single-shot qLDPC which requires using larger code size. The
 488 results are believed to be faster and more accurate for qLDPC codes instead of single-shot decoding.

489 **Single-shot decoding:** You can decode correctly (or nearly so) using only a single round of syndrome
 490 measurements instead of multiple rounds of syndrome measurements where is typically used. This require
 491 decoder can distinguish bewteen **data-qubit** errors and **measurement errors** and apply correction without
 492 repeated measurement rounds. Example: 3D color code with extra local containts among syndromes let you
 493 detect measurement errors. However, single-shot decoding often fail to preserve code distancen and logical
 494 error rate, this requires further larger code size, which is not resource-preserving also in a long run.

495 *Sliding-window:* Correct previous measurement syndromes and leave the latest errors for future correction.
 496 This increase logical memory lifetime and the code distance for hypergraph-product codes and lifted-product
 497 codes by using single-shot decoding. This decoding turns out to be more desirable for fast and accurate
 498 decoding for qLDPC codes.

499 **5.1 Single-shot decoding**

500 $m \times n$ parity-check matrix H (In the article, Z -type generators to correct X -tyoe error). An X error
 501 set represented by $e \in \mathbb{F}_2^n$ with ideal syndrome $\sigma_{ideal} := He \in \mathbb{F}_2^m$ and realistic syndrome denoted as
 502 $\sigma := \sigma_{ideal} + u$, where $u \subset \mathbb{F}_2^m$ represents the set of measurement errors.

503 To deal with readout errors, repetitive measurement (In [7], repetitive measurements are unnecessary
 504 because the logical operator is extracted using an ancilla block encoded in a distance- d code. The logical
 505 measurement outcome is therefore protected to the same degree as the data, allowing a single-shot proce-
 506 dure analogous to Steane's method but with substantially lower ancilla preparation overhead. Importantly,
 507 encoding a single logical operator into a distance- d ancilla is far more resource-efficient than attempting
 508 to eliminate repetition in standard syndrome extraction. In conventional error-correction protocols, avoiding
 509 repetitive stabilizer measurements would require encoding each stabilizer into its own distance- d ancilla
 510 block, which is prohibitively expensive for LDPC codes with $O(n)$ stabilizers. Thus, homomorphic logical
 511 measurement avoids repetition through an efficiently encoded ancilla, whereas repetitive measurement re-
 512 mains essential for syndrome extraction.) is common strategy but with lower error correction speed.

513 One could then consider **single-shot decoding** which correct the single shot decoder immediately after a
 514 noisy (only X errors can be detected) syndrome extraction. After a noisy decoding process, error set e be
 515 extracted as \tilde{e} , we then have $H\tilde{e} + \tilde{u} = \sigma \neq \sigma_{ideal}$ with $e \neq \tilde{e}$ (if is the ideal model, e and \tilde{e} will be different
 516 up to X -type stabilizers.). However, in single-shot correction, it is shown that if $e + \tilde{e} \in r$ where r is an X
 517 error set with $|r|$ (weight of r) is bounded.

518 Usually if fault-tolerance threshold exists, single-shot decoding is considered success (repetitive code
 519 ususally give $\rightarrow 0$ error threhold.). Therefore, this naturally gives weight $|r| \leq \alpha|u|$ with $\alpha > 0$, α is an con-
 520 stant independent of code size n . ($|r| = e + \tilde{e}$ is the weight with one-shot decoding and correction completed,
 521 therefore, if $|r|$ is smaller than measurement weight $\alpha|u|$, then somehow produce effective correcting.).

522 Threshold is not the only metric for QEC code, while it is naturally to think of space-time overheads to
 523 characterize the effectiveness of QEC code. For example, if a code performs with high distance but required
 524 a extremely large amounts of gates in encoding process and decoding process, then this might not be a
 525 good code. Another metric is called *effective distance*, d_{eff} which is upper bounded by code distance and
 526 determines the size of the code. We have logical error rate $p_L \propto p^{d_{eff}+1/2}$. This metric not only determines
 527 the circuit depths (Toric code stabilizer measurement: 4 CNOT layers + measurement depth = $O(1)$, which
 528 is quite good.) of the code, but also make it like space-time dependent (circuit depths \propto computation time
 529) which partially related to the overall computation of a round of error correction.

530 For $\alpha \leq 1$, then $d_{eff} = d$. For $\alpha \gg 1$, this shows extreme noisy decoding process with generated
 531 qubit errors $\alpha|u| \gg |u|$. The worst case would have $d_{eff} = d/\alpha$. This naturally leads to building larger
 532 code, while overall space overhead will not change using constant-rate qLDPC codes. As more logical qubits
 533 encoded in a single block code, the space-time of logical computation overhead and decoding complexity is
 534 increased. Above seems like trying to characterize what they mention before while single-shot decoding can
 535 degrade code distance d to d_{eff} , although overall space overhead (logical qubit per qubits (encoding rate)),
 536 this still requires large overheads for improving logical memory time and code distance.

537 5.2 Sliding window decoding

538 Instead of using large codeblock for an improved effective distance, the author considers sliding-window
 539 decoding, which use sets of syndromes accumulated multiple rounds of stabilizer measurements. A sliding
 540 window decoding (W (width), F (offset)) means for each W syndrome measurements and decoding we apply
 541 correction on F rounds and leaving $W - F$ rounds for the future correctin cycles.

542 The measured syndrome σ_t at time t obeys

$$\sigma_t = H \left(\sum_{j=1}^t e_j \right) + u_t.$$

543 Then the decoder \mathcal{D}_{win} takes $(\sigma_1, \dots, \sigma_W)$ as input to estimate $\tilde{e}_1, \dots, \tilde{e}_W$ and $\tilde{u}_1, \dots, \tilde{u}_W$ such that

$$H \left(\sum_{j=1}^t \tilde{e}_j \right) + \tilde{u}_t = \sigma_t.$$

544 The decoding part in this paper uses BP-OSD method which will be introduced in next Section [19] and
 545 with phenomenological error model with measurement error u_t and qubit error e_j and probability p and
 546 independent to each other.

547 To emphasize, the decoder cannot recover the exact errors e_t and u_t (while exact errors seem like could
 548 be traced in classical computers as random generating lists), but only get approximations \tilde{e}_t and \tilde{u}_t . After
 549 getting σ_t from decoder \mathcal{D}_{win} , we then apply

$$\xi := \sum_{j=1}^F \tilde{e}_j$$

550 on the qubits. Then update each syndrome σ_t for $t \in [F+1, W]$ as

$$\sigma'_t := \sigma_t + H\xi.$$

551 which can simply be changed manually with actual operaion in code. For next round, we change

$$\sigma_t = H \left(\sum_{j=W+1}^t \tilde{e}_j \right) + u_t$$

552 with t ranges from $W+1, \dots, W+F$ which we the generates sets of syndrome $(\sigma_{W+1}, \dots, \sigma_{W+F})$. Combined
 553 with previous round generated $(\sigma_{F+1}, \dots, \sigma_W)$. We can then take W inputs $(\sigma_{F+1}, \dots, \sigma_{W+F})$ to compute the
 554 next round decoder $\mathcal{D}_{win} := \sigma_t = H \left(\sum_{j=F+1}^{W+F} \tilde{e}_j \right) + \tilde{u}_t$.

555 Different choices of W, F might benefit the qLDPC codes compared with single-shot $(W, F) = (1, 1)$.
 556 While non-overlapping decoding will necessarily suffer from a residual error where they are not able to
 557 correct measurement/qubit errors occur on $W - 1$ timestep.

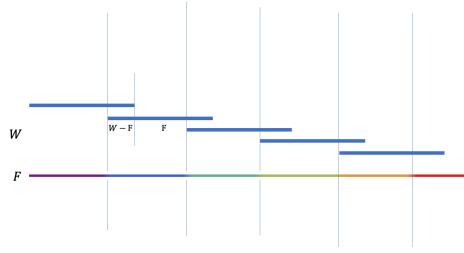


Figure 11:

558 5.3 Numerical simulations

559 Consider a phenomenological error model in which a qubit error e_j occurs with probability p , independently
 560 of a measurement error u_j , which also occurs with probability p . The criteria for a good performance of
 561 (W, F) -decoding is perform multiple rounds of syndrome extractions, decoding, and correction when an
 562 actual logical error occurs. The memory *lifetime* $T := (N - 1)F$ if the logical fail occurs when after N EC.
 563 Also, T is a function of error probability p .

564 Along the cycles, we keep track on a quantity *residual qubit error* $r := \sum_{i=1}^{NF} e^i + \sum_{j=1}^N \xi_j$, once the
 565 quantity r is not a **correctable** error set even in the absence of measurement errors.

566 To determine whether r is correctable or not, an *ideal decoder* $\mathcal{D}_{\text{ideal}} : \mathbb{F}^m \rightarrow \mathbb{F}^n$ is applied, namely
 567 $\tilde{r} := \mathcal{D}_{\text{ideal}}(Hr)$ to get \tilde{r} . If $r + \tilde{r}$ is non-trivial logical operator, then the logical error occurs.

568 The strategy is to check whether the following relation holds:

$$r + \tilde{r} \in \ker(H_X) \setminus \text{rs}(H_Z).$$

569 Note that now $r + \tilde{r}$ is guaranteed to lie in normalizer group.

570 BP-OSD is used for characterizing D_{ideal} and D_{win} (BP: *belief propagation* OSD: *ordered-statistics decod-*
 571 *ing*, a post-processing algorithm), with open source [13] and combination sweep method [11]. Combination-
 572 sweep parameter () is set to 40 in this paper.

573 Instead of using $(\sigma_1, \sigma_2, \dots, \sigma_W)$, we use $(\sigma_1, \sigma_2 - \sigma_1, \dots, \sigma_W - \sigma_W)$ which is by mapping relation
 574 $(\sigma_t = H_{\text{win}} \left(\sum_{j=1}^t e_j \right) + u_t)$ with $H_{\text{win}} = [I_W \otimes H | B \otimes I_m]$, where B is a $W \times W$ matrix and H is a $m \times n$
 575 matrix.

576 *Hypergraph Product Codes*: a linear code with $m_A \times n_A$ parity-check matrix A and $HGP(A)$ has X and
 577 Z parity-check matrices:

$$\begin{aligned} H_X &= [A \otimes I_{n_A} \quad | \quad I_{m_A} \otimes A^T], \\ H_Z &= [I_{n_A} \otimes A \quad | \quad A^T \otimes I_{m_A}]. \end{aligned}$$

578 An $(r, s) = (3, 4)$ LDPC matrix means that each column has exactly 3 ones and each row has exactly 4
 579 ones, with all other entries equal to zero. The left and middle panels of Fig. 12 show comparisons between
 580 non-overlapping decoding and overlapping decoding. The panel on the right shows that, for larger/more
 581 complex quantum codes, copies of overlapping decoding with same logical qubits as larger quantum codes
 582 still outperforms single-shot decoding in terms of logical lifetime.

583 **Decoding volume** $V = W(n - k)/2$ is a measure of decoding complexity. While 4 copies of [[625, 25, 8]]
 584 have same decoding volume as [2500, 100], it still yields longer logical memory lifetime.

585 *Lifted product codes*: Single-shot, overlapping, and non-overlapping decoding results for the [714, 100, \leq
 586 16] and [1428, 184, ≤ 24] lifted product codes are shown in Fig. 13, reproduced from [17].

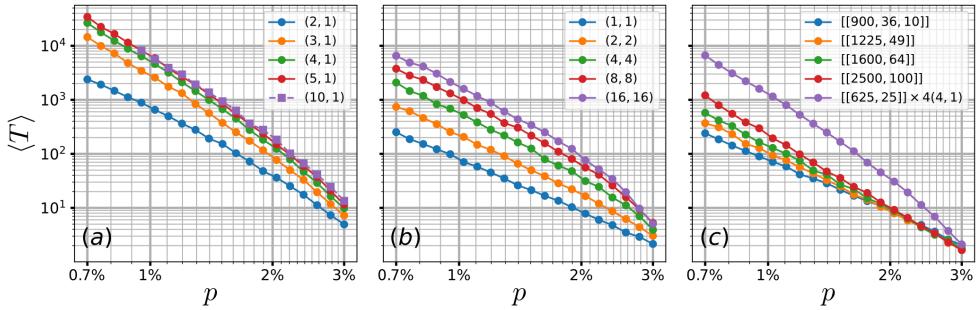


Figure 12:

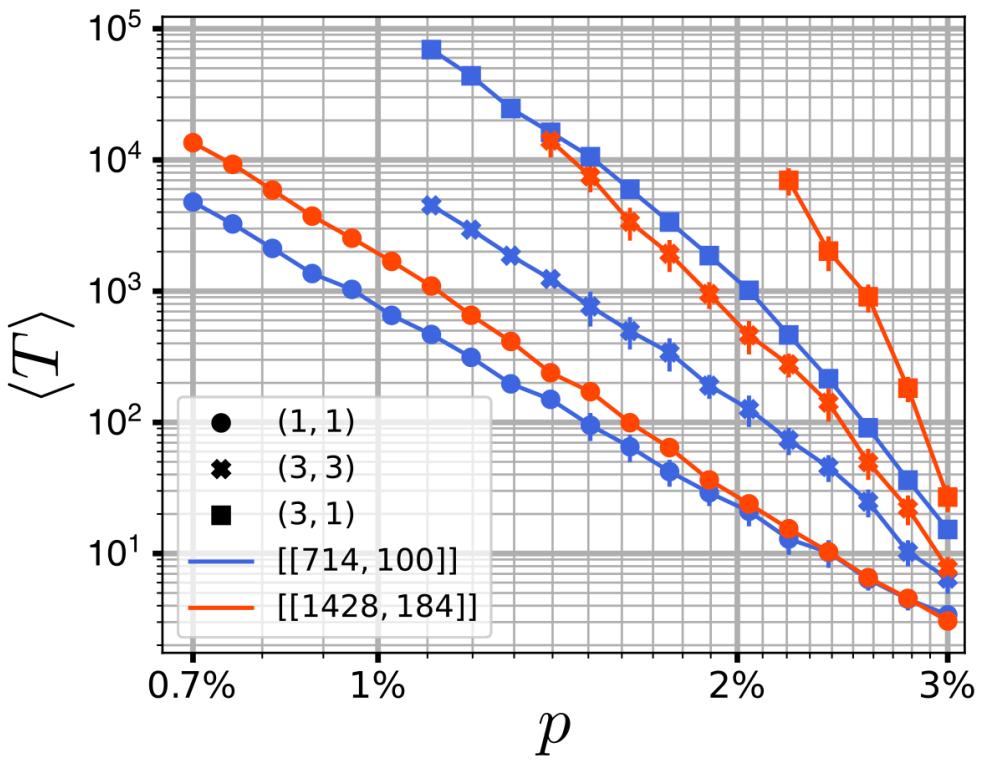


Figure 13:

588 **6 Decoding across the quantum low-density parity-check code land-**
 589 **scape (Notes on [13])**

590 Numerical simulations for previous paper based on studying paper [13].

591 1. Three families of hyperproduct code: topological codes, fixed-rate random codes, semitopological
 592 codes (shared properties of topological and hypergraph code allows for a trade-off between code threshold
 593 and stabilizer locality). 2. Exponential suppression in lower error regime for all three families

594 3. Compared with previous belief propagation decoders, the paper provides threshold comparable with
 595 minimum-weight perfect matching algorithm (golden standard on threshold, but) for toric codes which is
 596 also expected and subsequently demonstrated in the paper.

597 4. BP-OSD can be generalized to all QLDPC codes that can be constructed by hypergraph product while
 598 MWPM, union-find (UF) cannot.

599 **topological QLDPC codes** have local stabilizers embedded in some D-dimensional space with high
 600 error threshold but lower logical encoding rate.

601 **Random QLDPC codes** are hyperproduct of high-performance classical LDPC codes with higher en-
 602 coding rates (Unlike tend to zero with increasing block length on topological codes) and lower threhold than
 603 topological counterparts but with distant stabilzers that is harder to implement in various quantum platform
 604 without full connectivity.

605 The paper exapnds on the results that *BP*(belief propagation) + *OSD*(ordered statistics decoding) de-
 606 coder is a general decoder for all QLDPC codes that can be constructed from hypergraph product.

607 **semitopologcal codes:** Interplay between local topological codes and nonlocal random QLDPC codes.
 608 Firststep: *adge augmentation*: replacing each parity check edge with a length-g section of repetition code. Then
 609 semitopologcal code is obtained by hypergraph product. It can be thought of as surface codep patches
 610 connected to one another at their boundaries enabling long-range interaction.

611 For random QLDPC codes, quantum degeneracy (multiple equivalent solutions to the decoding problem)
 612 can be resolved by combining BP and OSC post-processing. When BP fails, OSD with matrix inversion is
 613 introduced to resolve ambiguities in quantum degeneracy.

614 Not only random QLDPC codes, but also topological QLDPC codes and semitopological codes are im-
 615 proved by using BP+OSD in this paper. Toric code threshold is found to be near the minimum weight-
 616 perfect-mathcing algorithm. For large code block size of semitopological codes (more local stabilziers.) the
 617 threshold approaches toric code results.

618 **6.1 Low-density parity-check codes**

619 *Classical error correction.*

620 Classical error correction. A classical error-correction code \mathcal{C}_H describes a redundant encoding $b \rightarrow c$
 621 from a k -bit data string b to an n -bit codeword c (with $n > k$). The codewords $c \in \mathcal{C}_H$ are defined as the
 622 null-space (kernel) vectors of an $m \times n$ binary parity-check matrix H such that

$$Hc \bmod 2 = 0,$$

623 that is,

$$\mathcal{C}_H = \ker(H) = \{ c \in \mathbb{F}_2^n \mid Hc = 0 \},$$

624 where the arithmetic is taken over the finite field \mathbb{F}_2 .

625 By the rank–nullity theorem, a parity-check matrix permits $k = n - \text{rank}(H)$ linearly independent
 626 codewords. If a codeword is subject to an error e , the parity-check matrix yields an m -bit syndrome

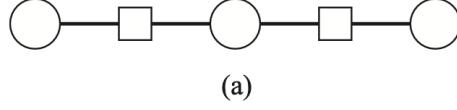
$$s = H(c + e) = He.$$

627 The syndrome is nonzero for all errors of Hamming weight less than the code distance, i.e. $|e| < d$. In general,
 628 classical codes are labeled using the notation $[n, k, d]$, where n is the codeword length, k is the number of

629 encoded bits, and d is the code distance. The code rate is given by

$$R = \frac{k}{n}.$$

630 *Factor graphs:* Data nodes $V = \{v_j | j = 1, \dots, n\}$ drawn as circles parity nodes $U = \{u_i | i = 1, \dots, m\}$ drawn
 631 as squares graph edge $\Lambda_{ij} \in \Lambda$ is drawn between when $H_{ij} = 1$ From bipartite graph $G = (V, U, \Lambda)$, one can
 632 easily visualize the corresponding parity-check matrix (adjacency matrix) H . *LDPC codes.*



(a)

Figure 14: [3,1,3] repetition code with parity check matrix (adjacency matrix) $H = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

633 *LDPC codes* A family of (l, q) -LDPC codes with column and row weights upper bounded by l and q .
 634 Either randomly producing column and row with weights set (l, q) and systematically modifying factor graph
 635 from a base code is possible to construct an (l, q) LDPC codes.

636 *Edge-augmented LDPC codes* Edge augmentation is a tool to create an LDPC code family from any
 637 "parent" factor graph $G = (V, U, \Lambda)$. **semitopological codes** are created by taking the hypergraph product
 638 of such augmented codes.

639 Connecting nodes with a single edge λ_{ij} in the parent code, the augmentation process involves adding a
 640 graph chain segment $G^g = \{V^g, U^g, \Lambda^g\}$. The adjacency matrix of the graph chain segment for $g = 4$ is

$$H^{g=4} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

641 The form is constructed by adding a diagonal $g - 1$ entries to the identity matrix with dimension g . To
 642 paraphrase again and make it intuitively consider (iv_j, λ_{ij}) which naturally gives a matrix with $i \times j$ elements
 643 with u_i and v_j indicating the row and column indices and λ_{ij} determine the either being 0 or 1. Now, if we
 644 consider

$$G' = (V \cup V_g, U \cup U_g, \{\lambda_{ij}\} \cup \Lambda^g \cup \Lambda^w),$$

645 it is like we concatenate every data nodes v_j (column) and parity nodes u_i (row) (Graphically, every circles
 646 and squares) with additional two edges $\Lambda^w = \{\lambda_{1j}^g, \lambda_{ig}^g\}$ connected to $\{v_j, u_1^g\}$ and $\{v_g^g, u_i\}$ and nodes
 647 (defined as Λ^g which is graph chain segment.) , for example, $(1 \cup V^g, 2 \cup U^g, \Lambda \cup \Lambda^g \cup \Lambda^w)$

648 Therefore, the g -augmented factor graph $G^{*g} = (V^{*g}, U^{*g}, \Lambda^{*g})$ is obtained by edge-augmenting every
 649 edge of the parent graph $G = (V, U, \Lambda)$ with a length- g **graph chain segment**.

650 If the parent graph G corresponds to an $[n, k, d]$ code, then the augmented graph G^{*g} corresponds to an
 651 $[n + g|\Lambda|, k, d']$ code, where the new distance satisfies

$$d' \geq (1 + g\mu) d,$$

652 where $|\Lambda|$ is the number of all edges in the parent graph and μ is the minimum degree over all data nodes
 653 in G .

654 The resulting code rate is

$$R^{*g} = \frac{k}{n + g|\Lambda|} = \frac{R}{1 + g|\Lambda|/n},$$

655 which reflects the trade-off between increasing the code distance (from d to d') and decreasing the rate.

656 An example of (2, 3)-LDPC parent code with check matrix

$$H^{g=4} = \begin{pmatrix} 1 & 1 \\ & 1 \\ & & 1 \\ & & & 1 \\ & & & & 1 \end{pmatrix}.$$

657 augmented with $g = 1, g = 2$ graph chain segment shows g -augmented graph G^{*1}, G^{*2} in Fig. [?] with code
658 parameters $[9, 2, 6], [15, 2, 10]$ It is easy to see that the check matrix shown her

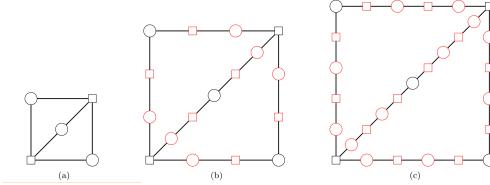


Figure 15: The left graph is the parent code. the red part correspond to $g = 1$ and $g = 2$ with check matrices being (1) and $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$, where Λ^w connected nodes $\{v_j, u_1^g\}$ and $\{v_g^g, u_i\}$ (connected to circle and square)

659 6.2 Quantum coding

660 The binary representation of a three-qubit Pauli operator $K = X_2Z_1$ (or error (\mathbf{x}, \mathbf{z})) is $k = (010, 100)$.

661 The CSS code checking matrix $H_{\text{CSS}} = \begin{pmatrix} H_Z & 0 \\ 0 & H_X \end{pmatrix}$. The corresponding quantum syndrome \mathbf{s}_Q will be

662 $\mathbf{s}_Q = (\mathbf{s}_X, \mathbf{s}_Z) = (H_Z \cdot \mathbf{x}, H_X \cdot \mathbf{z})$

663 CSS codes then can be described as two separately classical codes $\mathcal{C}(H_Z)$ and $\mathcal{C}(H_X)$ to detect X and Z
664 type errors.

665 *Hypergraph product codes* \mathcal{HGP} . For a classical code $n, k, d, \text{HGP}(\mathcal{C}_H)$ is a CSS code with

$$H_X = (H \otimes I_n \mid I_m \otimes H^T), \quad H_Z = (I_n \otimes H \mid H^T \otimes I_m).$$

666 The quantum code $\mathcal{HGP}(\mathcal{C}_H)$ parameters are

$$[[n^2 + m^2, k^2 + (k^T)^2, \min(d, d^T)]]$$

667 where H^T is the check matrix for code $[m, k^T, d^T]$.

668 6.3 Quantum LDPC codes

669 If the check matrix H_{CSS} has row and column weights bounded by l_Q and q_Q , then it defines a family of
670 (l_Q, q_Q) -QLDPC CSS codes(Hypergraph product preserve the sparsity).

671 There are two important classes of hypergraph codes: (1) topological QLDPC codes or surface code
672 family. (2) Random QLDPC codes which is constructed by randomly generated classical LDPC codes. The
673 paper propose a new class of semitopological codes contructed by hypergraph product of augmented LDPC
674 codes aims to combine both of its advantage. One can think of the surface-code family as having local
675 stabilizers described by a sparse parity-check matrix, and these stabilizers form a CSS code defined by two
676 matrices H_X and H_Z . Moreover, CSS codes naturally satisfy a chain-complex structure, which provides an
677 intuitive way to view the surface code: the stabilizers arise from taking the boundary operators between faces,
678 edges, and vertices of the underlying lattice. This correspondence between algebra and topology can also
679 be extended to the notion of covering spaces, which likewise obey chain-complex relations. Such structure

strengthens certain surface-code families [6] in their ability to support homomorphic logical measurements. To sum up, the paper [7] found an "analytic" way to systematically find the useful check matrix H_X and H_Z that are restricted in local stabilizers parity check matrix driven by topological intuition.

With random generated parity check matrix, is there an insight to construct "useful" parity matrices? Seems like [18] provides the answer.

Simply from Eq. 6.2, A (l_Q, q_Q) -QLDPC code family will have $l_Q = \max(2l, 2q)$ and $q_Q = l + q$.

Topological (4,4)-QLDPC codes. Each stabilizers are composed of four qubits and each qubit is involved in four stabilizers (With boundary, it might involve three or two qubits.). **hypergraph product** of $[n, 1, n]$ full-rank (The number of independent rows equals the number of rows. While $k = n - \text{rank}(H)$, hence parity check H is full rank.) repetition code gives an surface code with paramters $[[n^2 + (n - 1)^2, 1, n]]$. Toric code $[2n^2, 2, n]$ is given by taking hypergraph product of ring code $[n, 1, n]$. The shortcoming of *Topological (4,4)-QLDPC codes* is the encoding rate scales poorly as $R = k/n \rightarrow 0$ as $d \rightarrow 0$ Noted that for CSS codes, we have relations $k = n - \text{rank}(H) = n - \text{rank}(H_X) - \text{rank}(H_Z)$.

Random QLDPC codes. The merit of random QLDPC codes is the finite encoding rate per block compared with surface code due to the usage distant stabilizers. **4-cycle** in tanner graph negatively impacts iterative decoding in error correction, therefore, have to be avoided using Mackay-Neal method in the paper [10] (might be involved in their provided Python package). The corresponding $(8, 7)$ -QLDPC codes from $(3, 4)$ -LDPC codes produced encoding rate $R = k/n = 0.4$. In realistic hardware and in codespace, typically require more qubits to realize distant interaction in stabilizer checks. A mean weight for random codes is 7 in $(8, 7)$ -QLDPC codes, while $(4, 4)$ -topological QLDPC codes are 4.

semitopological codes. As mentioned, semitopological codes are formed by the hypergraph product of augmented LDPC codes. In the paper, it is constructed by taking the hypergraph product from $(2, 3)$ -LDPC codes with repetition code, which is the building blocks of surface code (local behavior). The codes combine the features of two types of codes. The comparisons can be seen in Fig. 16

\mathcal{C}_H	\mathcal{C}_H^T	$\mathcal{HGP}(\mathcal{C}_H)$	$R = k/n$	\bar{w}	g	\mathcal{C}_H^{*g}	$(\mathcal{C}_H^{*g})^T$	$\mathcal{HGP}(\mathcal{C}_H^{*g})$	R	\bar{w}
$[16, 4, 6]$	$[12, 0, \infty]$	$[[400, 16, 6]]$	0.04	7.0	0	$[3, 2, 2]$	$[2, 1, 1]$	$[[13, 5, 2]]$	0.385	5.00
$[20, 5, 8]$	$[15, 0, \infty]$	$[[625, 25, 8]]$	0.04	7.0	1	$[9, 2, 6]$	$[8, 1, 8]$	$[[145, 5, 6]]$	0.0345	4.25
$[24, 6, 10]$	$[18, 0, \infty]$	$[[900, 36, 10]]$	0.04	7.0	2	$[15, 2, 10]$	$[14, 1, 14]$	$[[421, 5, 10]]$	0.0119	4.14
					3	$[21, 2, 14]$	$[20, 1, 20]$	$[[841, 5, 14]]$	0.00595	4.10
					9	$[57, 2, 38]$	$[56, 1, 56]$	$[[6385, 5, 38]]$	0.000783	4.04

Figure 16: Tables of random LDPC codes and Semitopological codes performances, where \mathcal{HGP} is the hyperproduct code and with the classical codes C , encoding rate R , and mean weights \bar{w}

6.4 Belief propagation decoding

Receiving a syndrome $\mathbf{f} = H \cdot \mathbf{e}$, we look for the maximum-likelihood error

$$\mathbf{e}_{\text{MW}} = \arg \max_{\mathbf{e}} P(\mathbf{e} \mid \mathbf{s}), \quad \mathbf{e} = (e_1, e_2, e_3, \dots),$$

which ranges over 2^n possible error strings. The marginal probability (soft decisions) $P(e_i)$ is calculated to be

$$P(e_i = 1 \mid s) = \sum_{\substack{e_1, \dots, e_n \\ e_i = 1 \\ He = s}} P(e_1, \dots, e_n \mid s).$$

and subsequently with final decoding estimate (hard decoders) by:

$$P^1(e_i) = \sum_{\sim e_i} P(e_1, e_2, e_i = 1, e_3, \dots, e_n \mid s).$$

709 BP marginals perform less accurate with short loops in tanner graph and become better with long loops.
 710 BP decoder take check matrix H and \mathbf{s} as input and also suffer from quantum degeneracy mainly due to the
 711 quantum codes separate errors into cosets from stabilizer formalism but classical codes will have a nearly
 712 direct bijective relation between error and syndrome.

713 *BP decoding of quantum codes:*

(soft decoding (passing belief) \rightarrow hard decision \rightarrow check syndrome if $H\mathbf{e} = \mathbf{s}$)^{iterate until last step is satisfied}

714 In detail, BP algorithms is trying to compute fast for $P_1(\mathbf{e})$ first. We extracted the process from [19],
 715 while we follow notations from this paper might be different from the *BP – OSD* paper.

716 Consider a $[7, 1, 3]$ Steane code shown in Fig. 17, μ_{v_i} represents each bit flip error for each qubit node v_i ,
 717 and each check node c_i is connected to syndrome bit $s_{x,i}$. First, an initial LLR (log-likelihood ratio) of X_i
 718 (correspond to error on node)

$$\mu_{v_i} = \log \frac{\Pr(X_i = 1)}{\Pr(X_i = 0)} = \log \frac{p_x}{1 - p_x}.$$

719 is sent from each variable/parity node v_i to check node/syndrome node c_j as $m_{v_i \rightarrow c_j}^{(0)} = \mu_{v_i}$.

720 Then each iteration contains two steps of propagating back and forth between v_i and c_i . The governing
 721 equations are as follows:

$$m_{c_j \rightarrow v_i}^{(t)} = (-1)^{s_{x,j}} \tanh^{-1} \left(\tanh \left(\prod_{v_k \in \partial c_j \setminus v_i} m_{v_k \rightarrow c_j}^{(t)} \right) \right).$$

$$m_{v_i \rightarrow c_j}^{(t+1)} = \mu_{v_i} + \sum_{c_k \in \partial v_i} m_{c_k \rightarrow v_i}^{(t)}.$$

722 where $s_{x,j} \in \{0, 1\}$ is the syndrome corresponding to Pauli Z measurement. ∂c_i and ∂v_i denotes nodes which
 723 is connected v_i and c_i .

724 After numerous iterations, we compute marginal belief/bias $m_{v_i}^{(t)}$ as

$$m_{v_i}^{(t)} = \mu_{v_i} + \sum_{c_k \in \partial v_i} m_{c_k \rightarrow v_i}^{(t)} \approx \log \frac{\Pr(X_i = 1 \mid S_x = s_x)}{\Pr(X_i = 0 \mid S_x = s_x)}.$$

725 .
 726 and check for each syndrome measurement $\hat{x}^{(t)} H_1^T = s_x$. by using

$$x_i^{(t)} = \begin{cases} 0, & m_{v_i}^{(t)} > 0, \\ 1, & m_{v_i}^{(t)} \leq 0. \end{cases}$$

727 to characterize errors $\hat{x}^{(t)} = (\hat{x}_1^{(t)}, \hat{x}_2^{(t)}, \dots, \hat{x}_n^{(t)})$. Noted that some notations and differnet from the
 728 essential main paper I refer to due to definitions of probability.

729 *Ordered statistics decoding:*

730 Going back to the main paper, it aims to show that BP–OSD also applies to toric codes and semitopo-
 731 logical codes, in addition to random QLDPC codes, with the corresponding open-source implementation
 732 available on GitHub.

733 https://github.com/quantumgizmos/bp_osd

734 In this section, the authors describe OSD as a classical decoding procedure that can be applied equally well
 735 to the H_X and H_Z parity-check matrices of a CSS code.

736 **Method:**

737 For a $m \times n$ matrix H , we will always have $\text{rank}(A) \leq \min(m, n)$.

738 There is no full column rank for LDPC codes check matrix H as $n > m$ which can make $H^{-1} \cdot \mathbf{s} = \mathbf{e}$
 739 impossible. We then find subset of columns $[S]$ which is linearly independent and with associated submatrix
 740 $H_{[S]}$ with full column rank. Then the submatrix $H_{[S]}$ can be invertible following $H_{[S]}^{-1} \cdot \mathbf{s} = \mathbf{e}_{[S]}$.

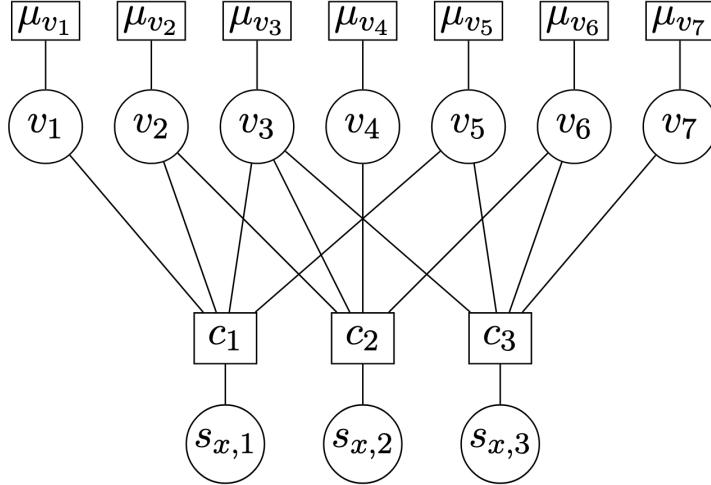


Figure 1: Tanner graph of H_1 for the $[[7, 1, 3]]$ Steane code

Figure 17: Tanner graph for $[7, 1, 3]$ Steane code H

Noted that different choice of basis $[S]$ will lead to different unique solution $\mathbf{e}_{[S]}$ which could eliminate possible quantum degeneracy.

OSD post-processing method is by modifying soft decisions in BP to obtain the basis set $[S]$ (qubits that involved in errors) with the highest probability of error occurs.

OSD-0 algorithm: Below show the following steps for BP-OSD when BP decoder fails.

(1) use BP soft decision form vector $P_1(\mathbf{e})$ to order the qubits which are being flipped or not in terms of probability, $[O_{BP}]$.

(2) Then rearrange the check matrix $H \rightarrow H_{[O_{BP}]}$ which is the matrix following the order of O_{BP}

(3) Use the first $\text{RANK}(H)$ linearly independent columns of $H_{[O_{BP}]}$ as $H_{[S]}$

(4) Calculate $\mathbf{e}_S = H_{[S]}^{-1} \cdot \mathbf{S}$, which is the OSD-O solution.

(5) The OSD-O solution can then be expressed as $\mathbf{e}_{S,T} = (\mathbf{e}_{[S]}, \mathbf{e}_{[T]}) = (\mathbf{e}_{[S]}, 0)$ while $[T] \notin [S]$. Then above guarantees $H_{[S,T]} \cdot \mathbf{e}_{[S,T]} = \mathbf{s}$ (while $H_{[S,T]}$ is the original H in (S, T) order)

(6) Mapping back to the original ordering, $\mathbf{e}_{[S,T]} \rightarrow \mathbf{e}_{OSD-0}$

It is like modifying check matrix from the original to improve the part where original BP will fail to predict.

Higher-order OSD: Considering where $e_T \neq 0$. While performing the 6 steps as in BP-OSD-0. They now give solution

$$(\mathbf{e}_{[S,T]} = H_{[S]}^{-1} \mathbf{e}_{[S]} + H_{[S]}^{-1} H_{[T]} \mathbf{e}_{[T]}, \mathbf{e}_{[T]})$$

while simply $H_{[S,T]} \cdot \mathbf{e}_{[S,T]}$ will give $\mathbf{e}_s + 2H_{[T]} \cdot \mathbf{e}_{[T]} = \mathbf{s}$ for all possible $\mathbf{e}_{[T]}$.

In principle, there are $2^{n-\text{RANK}(H)}$ configurations for \mathbf{e}_T , which gives difficulty to find the lowest weight of $\mathbf{e}_{S,T}$. However, combine with BP-OSD-0 ordering for matrix H . The task can be facilitated by applying weighted greedy search routine which prioritizes the more probable configurations of \mathbf{e}_T according to the soft decisions $P^1(\mathbf{e})$.

Greedy searching strategies for higher-order OSD:

For the numerical simulations, a strategy steps are listed as follows:

(1) As mentioned, \mathbf{e}_T is put in the order decided by BP soft decisions.

(2) Search over all weight-one configurations of $\mathbf{e}_{[T]}$

767 (3) And search for weight-two configurations in the first λ bits. This combined with weight-one searching
768 lead to $n - \text{RANK}(H) + \binom{\lambda}{2}$

769 The decoders using this combination sweep greedy search algorithm is called *BP + OSD – CS*. The
770 simulations in this paper use parameter $\lambda = 60$.

771 6.5 Numerical Simulations

772 *Simulation methodology for BP+OSD decoding.* If consider symmetric hypergraph product code, the decoding
773 problems for X and Z type errors are the same.

774 The residual error is given by $\mathbf{x}_R = \mathbf{x} + \mathbf{x}_{\text{OSD(BP)}}$. By checking the commutation relations with logical
775 Z operator sets $L_Z : L_Z \cdot \mathbf{x}_R = \mathbf{0}$. it shows zero when it \mathbf{x}_R and L_Z commutes, and with possible elemens
776 being one if anticommutate. Noted that errors anticommutates with stabilizers and stabilizers commutes with
777 logicals. Correctable errors commute with logicals, while one can think of correctable errors do not change
778 the subspace outside the encoding qubit. While errors anticommutate with logicals is itself logicals up to
779 possible stabilizers.

780 *Topological QLDPC codes*

781 Below error threshold, you can decrease logical error rate by increasing code distance or code concatenation.
782

783 The papers aims to conctruct a new type of codes which could have finite encoding rate and have low
784 logical error rate and high error threshold.

785 7 Notes on Fault-Tolerant Belief Propagation for Practical Quan- 786 tum memory [9]

787 A fault-tolerant belief propagation decoder that utilise a **sapce-time tanner graph** across **multiple**
788 **rounds of syndromes measurement** with mixed alphabet error variables.

789 (1) **Probabilistic error consolidation** to mitigate degeneracy effects and short cycles

790 (2) Adaptive sliding window that can capture long error events.

791 (3) high error thresholds of 0.4% – 0.87% and strong error-floor performance for various types of topo-
792 logical codes.

793 (4) Reduction of generalized check matrix for computation complexity.

794 (5)The fundamental purpose of generalized check matrix is by adding temporal and genral faults (circuit-
795 level) in to the matrix. If standard raw syndrome do not involve temporal and fault-tolerance. Hence the
796 design may not directly tackle temporal and fault-tolerance.

797 (6) Above tempreal addition make them possible to connect variable node connects to at most two rounds
798 of check nodes. This extend each stabilizer lives in each stabilizer. They can termporally addind effects to
799 refine the code. This might make errors propagate through rounds, but overall the effects are better.

800 (7) They merge errors temporally if with two intuition: sparse matrix computation or when errors are
801 degenerate (errors consolidation). Noted that this does not change the real circuits but computational
802 complexity. However, computational complexity do benefit real hardware fidelity if the decoding time can
803 be lower. The computation can then carry on if decoding process is fast. While adaptive sliding window and
804 their circuit-level choice do benefit real physics in a more fundamental way. Since it is about adding more
805 degrees of freedom to change the system.

806 (8) Qubit *rightarrow location* in this paper. For example, there many locations for an error $E =$
807 (E_1, E_2, \dots) . However, E_1, E_2, E_3, E_4 correspond to different errors in the first qubit.

808 (9) Representing errors in terms of binary bits, which requires 4-bit strings in circuit level model. The
809 real computations involve enlarging locations into column vectors.

810 (10) These all process, first starting from quantum computation, suddenly, we are trying to decode the
811 circuits. First, they apply decoders on not one round but three round in a role. This process also includes
812 circuit-level model and allow temporal connection between nodes and checks, called space-time tanner graph.

813 So this "choice" of decoding instead of each round give them degree of freedom to optimize the decoding
 814 process, which could be more efficient, for example, adding temporal checks between different rounds. Now,
 815 we are entering the decoding process, not only merging errors, they also constructed careful intuition on
 816 how to decompose errors and observe their degeneracy, by merging degeneracy, the decoder will become
 817 more efficient. At last, move into a larger picture when we are performing quantum error correction, they
 818 further improve the standard sliding window to adaptive sliding window, which further improve the decoding
 819 performance, this could lead to a high threshold. Here I have think of something, if we can follow the same
 820 temporal analogy like syndrome extraction, we memorize the adaptive syndrome, and we do not correct
 821 it in each adaptive sliding window, we are further optimizing the overall error threshold in a even higher
 822 percentage. Since correction also includes time. is it like passive error correction. The sliding window
 823 process involve multiple syndrome measurements, all of them but not just the last one are used in BP to
 824 find good correction.

825 Toric code is LDPC codes but not QLDPC codes, while QLDPC codes should have finite encoding rate.

826 **7.1 Introduction**

827 Typically, a code distance d is protecting the quantum memory but requires $O(d)$ rounds of syndrome
 828 measurements resulting in $O(d^3)$ potential error locations for a code length $O(d^2)$ ($O(d^2)$ qubits).

829 A decoder with complexity nearly in linear with the number of error variables. Apart from perfect
 830 error syndromes, quantum data-syndrome codes, and the phenomenological noise model. Here introduce a
 831 fault-tolerant belief propagation (FTBP) for general QLDPC, CSS, and no-CSS codes under noise model.

832 By incorporating both spatial and temporal correlations, this sparse graph representation allow more
 833 effective error correction. This also mitigates effect when gates are suppressed by probability p realistic
 834 spontaneous emission error which is related to time.

835 Noted that errors related by stabilizers must be in same syndrome and errors note related by stabilizers
 836 can also be in same syndromes.

837 Two-qubit gates introduces in the method cause more short cycles. In this paper, **probabilistic error**
 838 **consolidation** is introduced. The process put degeneracy errors and errors with same residue errors into a
 839 new single representative set. This process decouples higher- order error variables into lower-order ones for
 840 probabilistic consolidation. This can reduce short cycles in the Tanner graph.

841 In addition, they introduce dynamic sliding window to make widow offset being effective and efficient.
 842 Noted that non-overlapping sliding window in temporal majority voting is useless while (W, W) is same as
 843 $(1, 1)$ but with overlapping (W, F) could help improve code distance $\boxed{8}$.

844 FTBP applies to general QLDPC codes and with nearly linearly scaling results in $O(d^3 \log(d))$ computa-
 845 tionally intensive for a a code of distnace d .

846 **7.2 Quantum stabilizer codes and circuit-level noise model**

847 **7.3 generalized check matrix for syndrome extraction circuit and circuit-level** 848 **decoding problem and FTBP and sparse generalized check matrix for space-** 849 **time Tanner graph and error merging and probabilisitc error consolidation**

850 Stabilizer S is abelian group. Degenerate errors can be thought of as the elements in the cosets of same
 851 syndrome measurement results.

852 With multiple rounds of raw syndrome extraction for codes $[[d^2, 2, d]]$ rotated toric codes (with even d)
 853 $[\text{?}, \text{?}]$, the $[[\frac{9}{8}d^2, 4, d]]$ rotated 6.6.6 toric color codes (with d a multiple of 4) $[\text{?}, \text{?}]$, and the $[\frac{d^2+1}{2}, 1, d]$
 854 twisted XZZX toric codes (with odd d) $[\text{?}]$.

855 Lattice representation of $[[d^2, 2, d]]$ for $d = 2$. The entangling gate in order $(i, j, k, l) = (1, 4, 3, 2)$ for
 856 ancilla 1 within in a Z type stabilizers $((2, 3, 4, 1)$ for ancilla 2) which is shown in . $\boxed{18}$.But each stabilizer
 857 S_1, S_2, \dots can be performed in parallel.

858 **Circuit-level noise model**

859 -errors $\{I, X, Y, Z\}$

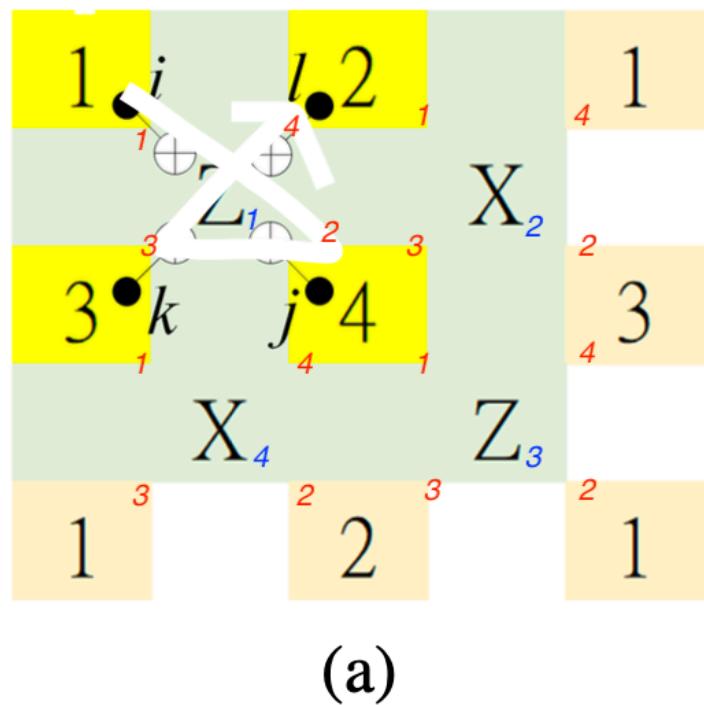


Figure 18: Blue numbers are the ancilla order, and red numbers are the entangling gate order. This involve four depths.

860 Assuming a circuit-level noise model in the quantum memory, each potential error source in the syndrome
 861 extraction circuit is referred to as a *location*. After perfect ancilla preparation, it experiences subsequent
 862 bit-flip or phase flip errors. Additionally, idle qubits will also suffer from Pauli errors which have to be
 863 taken into account.

864 7.4 Generalized check matrix

865 a M syndrome bits $s = \mathfrak{M} = \mathcal{H} \star E$, where \mathcal{H} is with dimension M (Belief propagation decoding of quantum LDPC codes with
 866 $N(N$ locations) as generalized check matrix.

$$E \star F = \sum_{k: E_k = D_i \text{ or } C_{ij}} E_k * F_k + \sum_{k: E_k = b_i \text{ or } m_i} E_k \cdot F_k \pmod{2}.$$

Here, H , E , and F take values in a mixed alphabet

$$\{I, X, Y, Z\}, \quad \{I, X, Y, Z\}^2, \quad \text{and} \quad \{0, 1\},$$

corresponding to the four types of circuit-level errors we consider. Specifically, the ancillary qubit preparation error is given by

$$b_i \in \{0, 1\},$$

which corresponds to either $\{I, X\}$ or $\{I, Z\}$. The CZ or CNOT gate error acting on ancilla i and data qubit j is represented by

$$C_{ij} \in \{I, X, Y, Z\}^2.$$

The idle-qubit error on data qubit i is denoted by

$$D_i \in \{I, X, Y, Z\},$$

and the measurement error at ancilla i is specified by

$$m_i \in \{0, 1\}.$$

867 The result is a binary syndrome $s \in \{0, 1\}$, with

$$E * F = \begin{cases} 0, & \text{if } E \text{ and } F \text{ commute,} \\ 1, & \text{if } E \text{ and } F \text{ anticommute.} \end{cases}$$

$$E \star F = \sum_{k: E_k = D_i \text{ or } C_{ij}} E_k * F_k + \sum_{k: E_k = b_i \text{ or } m_i} E_k \cdot F_k \pmod{2}.$$

868 Here H, E, F take values in a mixed alphabet $\{I, X, Y, Z\}$, $\{I, X, Y, Z\}^2$, and $\{0, 1\}$. The result is a
 869 binary syndrome $s \in \{0, 1\}^M$.

$$E * F = \begin{cases} 0, & \text{if } E \text{ and } F \text{ commute,} \\ 1, & \text{if } E \text{ and } F \text{ anticommute.} \end{cases}$$

870 **Proposition 1.** The generalized check matrix H is constructed as follows:

871 1. If location k corresponds to an ancilla preparation error b_i or a measurement error m_i , then the k -th
 872 column of H lies in $\{0, 1\}^M$ and is

$$H_k = \mathfrak{M}(\{E_k = 1\}) = (0, 0, 0, 0, \dots, E_k = 1, \dots, 0).$$

873 This is k th role of the check matrix.

874 2. If location k corresponds to an idle qubit error D_i , then the k -th column of H lies in $\{I, X, Y, Z\}^M$.
 875 Since both X and Z syndromes must be captured, set

$$H_k = Z^u X^v, \quad u = \mathfrak{M}(\{E_k = X\}), \quad v = \mathfrak{M}(\{E_k = Z\}),$$

876 where

$$u = \{u_1, u_2, \dots, u_M\}, \quad v = \{v_1, v_2, \dots, v_M\}.$$

877 3. If location k corresponds to a CZ or CNOT gate error C_{ij} , then the k -th column of H lies in
 878 $\{I, X, Y, Z\}^{M \times 2}$. The relevant error bases are

$$u = \mathfrak{M}(\{E_k = X \otimes I\}), \quad v = \mathfrak{M}(\{E_k = Z \otimes I\}),$$

$$u' = \mathfrak{M}(\{E_k = I \otimes X\}), \quad v' = \mathfrak{M}(\{E_k = I \otimes Z\}).$$

880

Thus,

$$H_k = \begin{pmatrix} Z^{u_1} X^{v_1} & Z^{u'_1} X^{v'_1} \\ \vdots & \vdots \\ Z^{u_M} X^{v_M} & Z^{u'_M} X^{v'_M} \end{pmatrix},$$

which is interpreted as a length- M vector over $\{I, X, Y, Z\}^2$.

882 The above relation are utilised to construct "generalized parity check matrix" which is shown in Fig. 19 with
 883 raw check matrix being $\begin{bmatrix} Z & Z \\ X & X \end{bmatrix}$

	$D_1^{(1)}$	$D_2^{(1)}$	$C_{1,1}^{(1)}$	$C_{2,2}^{(1)}$	$C_{1,1}^{(2)}$	$C_{2,1}^{(1)}$	$m_1^{(1)}$	$m_2^{(1)}$	$b_1^{(1)}$	$b_2^{(1)}$	$D_1^{(2)}$	$D_2^{(2)}$	$C_{1,1}^{(2)}$	$C_{2,2}^{(2)}$	$C_{1,2}^{(2)}$	$C_{2,1}^{(2)}$	$m_1^{(2)}$	$m_2^{(2)}$	$b_1^{(2)}$	$b_2^{(2)}$	$D_1^{(3)}$	$D_2^{(3)}$	$C_{1,1}^{(3)}$	$C_{2,2}^{(3)}$	$C_{1,2}^{(3)}$	$C_{2,1}^{(3)}$	$m_1^{(3)}$	$m_2^{(3)}$	$b_1^{(3)}$	$b_2^{(3)}$
(a)	Z	Z	ZI	IZ	ZI	1	1				Z	Z	ZI	IZ	ZI		1			Z	Z	ZI	IZ	ZI		1				
	X	X	IX	XI	XI			1	1		X	X	IX	XI	XI		1	1	1	X	X	IX	XI	XI		1	1	1		
	Z	Z	IZ	ZZ	IZ	IZ					Z	Z	IZ	ZZ	IZ	IZ				Z	Z	ZI	IZ	ZI		1				
	X	X	XX	IX	IX	IX					X	X	XX	IX	IX	IX				X	X	XX	IX	IX		1				
	Z	Z	IZ	ZZ	IZ	IZ					Z	Z	IZ	ZZ	IZ	IZ				Z	Z	ZI	IZ	ZI		1				
	X	X	XX	IX	IX	IX					X	X	XX	IX	IX	IX				X	X	XX	IX	IX		1				
	Z	Z	IZ	ZZ	IZ	IZ					Z	Z	IZ	ZZ	IZ	IZ				Z	Z	IZ	ZZ	IZ	IZ					
	X	X	XX	IX	IX	IX					X	X	XX	IX	IX	IX				X	X	XX	IX	IX	IX					

Figure 19:

The circuits for three rounds of raw syndrome extraction of check matrix $\begin{bmatrix} Z & Z \\ X & X \end{bmatrix}$ and corresponding error variables is shown in Fig. 20.

⁸⁸⁶ Let $R(\mathcal{M}, E) \in \{I, X, Y, Z\}^n$ denote the residual Pauli error on the data qubits of the n -qubit code after
⁸⁸⁷ the execution of circuit \mathcal{M} .

888 Circuit-level decoding problem

Given a syndrome extraction circuit \mathfrak{M} , syndrome bits $s = \mathfrak{M}(E) \in \{0, 1\}^M$, find an estimate \hat{E} such that

$$\mathfrak{M}(\hat{E}) = s$$

891 and

$$R(\mathfrak{M}, \hat{E}) R(\mathfrak{M}, E)$$

⁸⁹² is correctable by perfect QEC.

893 Definition 3

After building stabilizers, they constructed error vector E from the following settings:

- ⁸⁹⁵ 1) Each single-qubit Pauli variable D_j is independently generated with depolarizing rate $\epsilon \in [0, 3/4)$,
⁸⁹⁶ following the distribution $(p_j^I, p_j^X, p_j^Y, p_j^Z) = (1 - \epsilon, \epsilon/3, \epsilon/3, \epsilon/3)$.

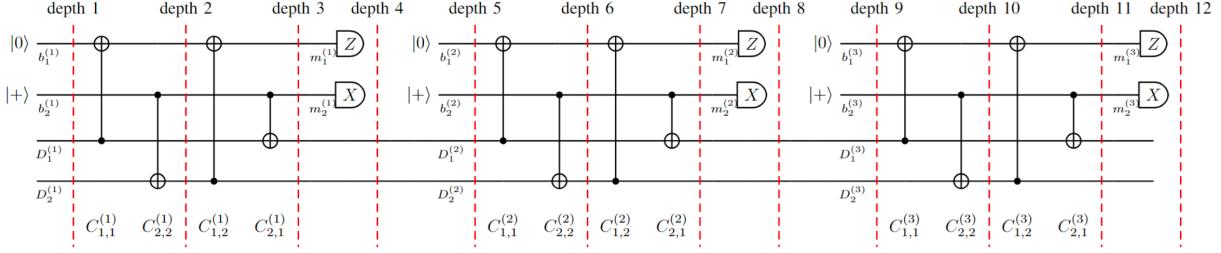


Figure 20:

897 2) Each two-qubit Pauli variable C_{ij} is independently generated with depolarizing rate $\epsilon \in [0, 3/4)$, such
898 that $C_{ij} = I$ with probability $1 - \epsilon$ and C_{ij} is a non-identity two-qubit Pauli with probability $\epsilon/15$.

899 3) Each syndrome bit error m_j or ancillary preparation error b_j is an independent bit-flip or phase-flip
900 error with rate $\epsilon_b \in [0, 1/2)$, following the distribution $(q_j^{(0)}, q_j^{(1)}) = (1 - \epsilon_b, \epsilon_b)$.

901 The log-likelihood ratio then can be defined consequently based on above.

902 In addition, the Tanner graph has N variable nodes corresponding to E_1, \dots, E_N and M check nodes
903 corresponding to the rows of H . The neighbors of a check node i are

$$N(i) = \{j : H_{ij} \neq I \text{ or } H_{ij} \neq 0\},$$

904 Here j mean collect variable nodes (in index j) directly connected to check node i . For example, $N(i)$ is the
905 set such that $H_{i1} = X, H_{i2} = Z, \dots$

906 The neighbors of a variable node j are

$$M(j) = \{i : H_{ij} \neq I \text{ or } H_{ij} \neq 0\}.$$

907 This shows trivial operator in \mathcal{H} for identity I and 0 actually do not contribute to "stabilize", therefore no
908 need to perform entangling gates corresponding to this entry.

909 Belief propagation generates LLR vectors $\Gamma_1, \dots, \Gamma_N$ for estimating the error variables. The FTBP
910 algorithm performs binary, quaternary, and 16-ary message computations.

911 **FTBP Algorithm**

912 Variable-to-check (V-to-C): $\Gamma_{j \rightarrow i}$ Check-to-variable (C-to-V): $\Delta_{j \rightarrow i}$

913 If the syndrome does not match, then the process continue until the maximum of T_{\max} iterations.

914 We first quickly summarize the main difference between FTBP and standard BP:

915 (1) Space-time Tanner graph while standard BP is spatial only.

916 (2) FTBP variable update (Pauli case): instead of Z or X type error, here with full circuit level error which
917 would include 2/4/16-ary instead of all 2 ary, since we have to characterize errors like two-qubit errors. A
918 matrix as shown in Fig. 19 shows this for label C denoting rows and columns.

919 (3) Horizontal step do not change from BP

920 (4) Vertical Step (Marginal Distribution Part):

$$\Gamma_j^W = \Lambda_j^W + \frac{1}{\alpha} \sum_{\substack{i \in \mathcal{M}(j) \\ W \star H_{ij} = 1}} \Delta_{i \rightarrow j}, \quad W \neq I.$$

921 This is the quantum twist.

922 For each Pauli $W \in \{X, Y, Z\}$:

923 - Add only the check messages for checks where $W \in \{X, Y, ZXI, IX, XY, XZ, \dots\}$ anticommutes with
924 the check-matrix entry H_{ij} . In fact, this can be thought of as if error is X and the parity check matrix

925 element are also X , then this do not cause an error. But in classical BP, while there are just $0 \cdot 0, 0 \cdot 1, 1 \cdot 0,$
 926 $1 \cdot 1$, which naturally can be expressed as
 927 Because anticommutate = 1 in the syndrome.

928 7.5 BP complexity discussion for circuit-level decoding problem

929 Window W size is smaller to reduce generalized check matrix. A n qubit stabilizer code with m stabilizers
 930 with weight w requires mw CNOT or CZ gates, and m ancilla preparations. The decoding procedure is
 931 applied to r rounds of syndrome extraction. The generalized matrix will then be $rm \times r(n + mw + 2m)$
 932 (where n qubits plus possible errors on m ancilla preparations and m measurements and mw gates)

933 Lemma

934 Reduce the complexity of FTBP generalized check matrix. The complexity of FTBP for an n -qubit
 935 quantum code with $O(1)$ stabilizer weights and a window of $O(\sqrt{n})$ rounds of syndrome extraction is $O(n1.5$
 936 $\log n)$, achieved using a sparse generalized check matrix. Additionally, each variable node connects to at
 937 most two rounds of check nodes in the corresponding Tanner graph.

938 7.6 Probabilistic error consolidation

939 The trick is to changing the input of errors that could combine degenerate errors into one with two times
 940 probability and the other degenerate one being zero. Noted that errors vector and check matrix are invertible.
 941 So labeling degenerate errors on parity check elements is like telling you which type of error should be merged.

942 7.7 adaptive sliding window

943 7.8 simulations

944 8 Notes on Dan Browne topological codes

945 8.1 Toric codes

946 all boundaries are cycles as $\partial_{n+1}\partial_n = 0$

947 **Boundaries** $B_n = \text{im } \partial_{n+1}$ **Cycles (closed loops)** $Z_n = \ker \partial_n$

$$B_n = \text{im}(\partial_{n+1}), \quad Z_n = \ker(\partial_n), \quad B_n \subseteq Z_n.$$

948 In the toric code, $Z_1 = \ker(\partial_1)$ is the group of 1-chains with zero boundary, (Mathematically, centralizer
 949 of stabilizer group) pictorially it can be thought of as all closed loops that pass through X stabilizers (vertices),
 950 this form sets of logical Z and other closed loops formed by Z stabilizers (when pass through $Z_1 = \ker(\partial_1)$
 951 $= Z_1 = \ker(H_x)$, it means formed by Z stabilizers, and vice versa, one can pictorially imagine that.) In the
 952 chain complex, this is consistent with the relation $\partial_2 \circ \partial_1 = 0$, which ensures that every boundary is a cycle.
 953 However, there are loops that are not contractible, represented by $\ker(\partial_1)/\text{im}(\partial_2)$, pictorially $\text{im}(\partial_2)$ is all
 954 loops that can be formed by stabilizers multiplication, and $\ker(\partial_1)/\text{im}(\partial_2)$ satisfies $\ker(\partial_1) - \text{im}(\partial_2) \subset \text{im}(\partial_2)$.
 955 Or simply thinking, make group generated by LS, S, L with $S = I$, then the remaining will be only logicals
 956 L .

957 Noted that $\ker(\partial_1)/\text{im}(\partial_2)$ characterize Z logical and $\ker(\partial_2)/\text{im}(\partial_1)$ characterize X logical, pictorially
 958 and algebraically imagine no trivial commuted with their dual types of operators.

959 This can further deduce, that Z logicals lies on the "edges" and X logicals lies on the "faces" as shown in
 960 Fig. 21 toric codes pictorial interpretations.

961 Two error models: **depolarizing**: each qubit independently suffer from three types of error and **uncorrelated model**: usually problems can be separated into X and Z type decoding problems.

963 Minimum weight checking: graph characterize the weights between syndrome measurement that show
 964 errors.

965 Fig. 22 shows a process when decoding a syndrome measurement based on MWPM.

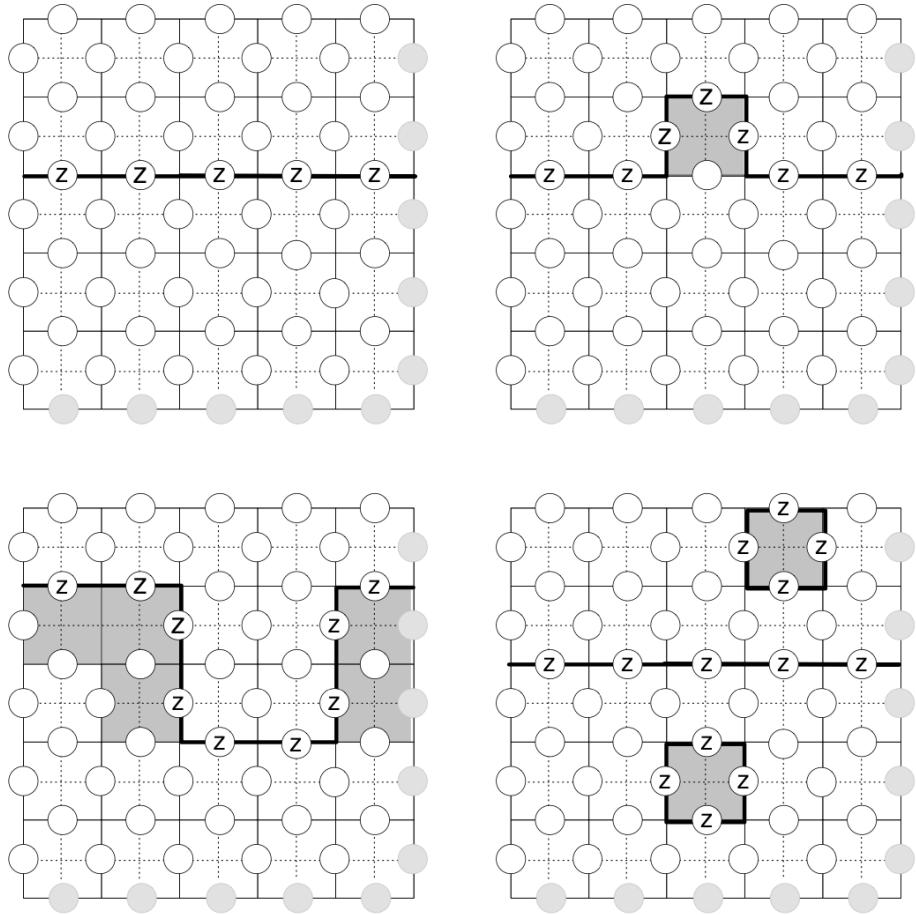


Figure 21:

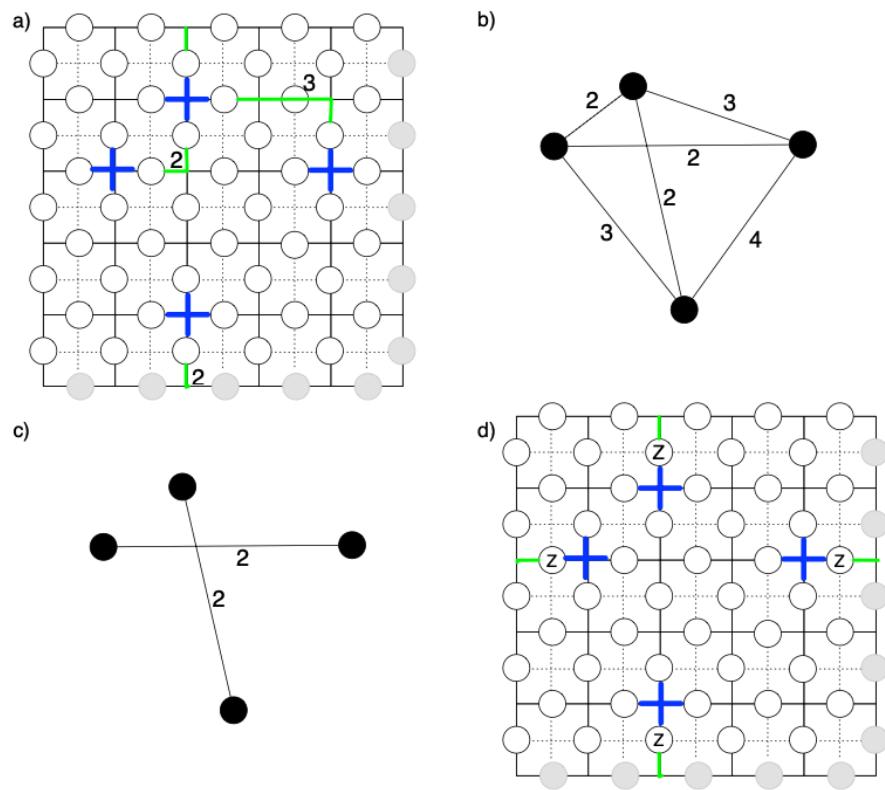


Figure 22:

966 The runtime scales with V^3 V is the vertices of the graph shown in Fig. 22(b).

967 8.2 Elements of Topology and Homology

968 *Homeomorphism*: an object can include stretching, bending, but not (in general) tearing or glueing. For-
 969 mally: Continuous mapping between two objects or topology: continuous, bijective, continuous inverse function.
 970 Function mapping is shown in Fig. 23.

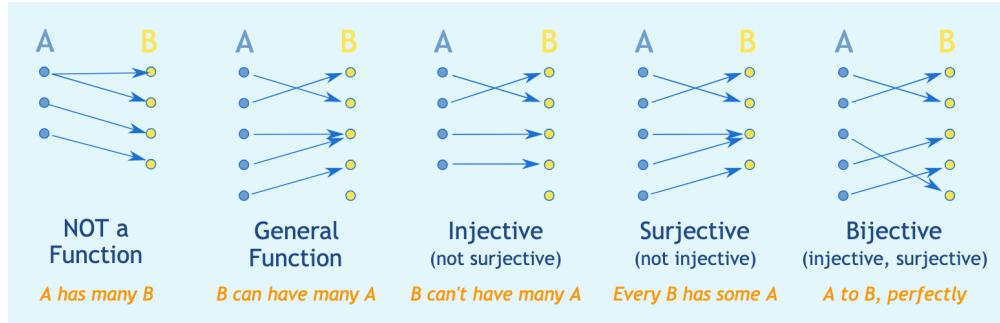


Figure 23:

971 *Cellulation (CW complex)*: Euler characteristic of any cellulation of the sphere is 2, and that more
 972 generally, for any surface, χ is independent of the cellulation chosen and is a topological invariant.
 973 The surface of any convex polyhedron is homeomorphic to a sphere.

974 References

- 975 [1] <https://sites.google.com/site/danbrownnecl/teaching/lectures-on-topological-codes-and-quantum-computation>.
- 977 [2] Quantum error correction below the surface code threshold. *Nature*, 638(8052):920–926, 2025.
- 978 [3] Anthony Bosman. Algebraic topology, 2023.
- 979 [4] Austin G Fowler, Matteo Mariantoni, John M Martinis, and Andrew N Cleland. Surface codes: To-
 980 wards practical large-scale quantum computation. *Physical Review A—Atomic, Molecular, and Optical Physics*, 86(3):032324, 2012.
- 982 [5] Shilin Huang. Homomorphic logical measurements | qiskit seminar series with shilin huang. 2023.
- 983 [6] Shilin Huang, Kenneth R. Brown, and Marko Cetina. Comparing shor and steane error correction using
 984 the bacon-shor code. *Science Advances*, 10(45):eadp2008, 2024.
- 985 [7] Shilin Huang, Tomas Jochym-O'Connor, and Theodore J Yoder. Homomorphic logical measurements. *PRX Quantum*, 4(3):030301, 2023.
- 987 [8] Shilin Huang and Shruti Puri. Improved noisy syndrome decoding of quantum ldpc codes with sliding
 988 window. *arXiv preprint arXiv:2311.03307*, 2023.
- 989 [9] Kao-Yueh Kuo and Ching-Yi Lai. Fault-tolerant belief propagation for practical quantum memory.
 990 *arXiv preprint arXiv:2409.18689*, 2024.
- 991 [10] David JC MacKay and Radford M Neal. Near shannon limit performance of low density parity check
 992 codes. *Electronics letters*, 32(18):1645–1646, 1996.

- 993 [11] Pavel Panteleev and Gleb Kalachev. Degenerate quantum ldpc codes with good finite length performance.
 994 *Quantum*, 5:585, 2021.
- 995 [12] Lukas Postler, Sascha Heußen, Ivan Pogorelov, Manuel Rispler, Thomas Feldker, Michael Meth, Chris-
 996 tian D Marcinia̧k, Roman Stricker, Martin Ringbauer, Rainer Blatt, et al. Demonstration of fault-
 997 tolerant universal quantum gate operations. *Nature*, 605(7911):675–680, 2022.
- 998 [13] Joschka Roffe, David R White, Simon Burton, and Earl Campbell. Decoding across the quantum
 999 low-density parity-check code landscape. *Physical Review Research*, 2(4):043423, 2020.
- 1000 [14] Jean-Pierre Tillich and Gilles Zémor. Quantum ldpc codes with positive rate and minimum distance pro-
 1001 portional to the square root of the blocklength. *IEEE Transactions on Information Theory*, 60(2):1193–
 1002 1202, 2013.
- 1003 [15] Andre Van Rynbach, Ahsan Muhammad, Abhijit C Mehta, Jeffrey Hussmann, and Jungsang
 1004 Kim. A quantum performance simulator based on fidelity and fault-path counting. *arXiv preprint*
 1005 *arXiv:1212.0845*, 2012.
- 1006 [16] Ye Wang, Stephen Crain, Chao Fang, Bichen Zhang, Shilin Huang, Qiyao Liang, Pak Hong Leung, Ken-
 1007 neth R Brown, and Jungsang Kim. High-fidelity two-qubit gates using a microelectromechanical-system-
 1008 based beam steering system for individual qubit addressing. *Physical Review Letters*, 125(15):150505,
 1009 2020.
- 1010 [17] Qian Xu, J Pablo Bonilla Ataides, Christopher A Pattison, Nithin Raveendran, Dolev Bluvstein,
 1011 Jonathan Wurtz, Bane Vasić, Mikhail D Lukin, Liang Jiang, and Hengyun Zhou. Constant-overhead
 1012 fault-tolerant quantum computation with reconfigurable atom arrays. *Nature Physics*, 20(7):1084–1090,
 1013 2024.
- 1014 [18] Qian Xu, Hengyun Zhou, Guo Zheng, Dolev Bluvstein, J Pablo Bonilla Ataides, Mikhail D Lukin, and
 1015 Liang Jiang. Fast and parallelizable logical computation with homological product codes. *Physical*
 1016 *Review X*, 15(2):021065, 2025.
- 1017 [19] Hanwen Yao, Waleed Abu Laban, Christian Häger, Alexandre Graell i Amat, and Henry D Pfister.
 1018 Belief propagation decoding of quantum ldpc codes with guided decimation. In *2024 IEEE International*
 1019 *Symposium on Information Theory (ISIT)*, pages 2478–2483. IEEE, 2024.
- 1020 [20] Yi-Cong Zheng, Ching-Yi Lai, and Todd A Brun. Efficient preparation of large-block-code ancilla states
 1021 for fault-tolerant quantum computation. *Physical Review A*, 97(3):032331, 2018.