QECFM: QUANTUM ERROR CORRECTING CODE DISCOVERY VIA TRAINING FREE GUIDANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Designing quantum error-correcting codes is difficult: the search space is discrete and highly constrained, while evaluating a candidate often requires an expensive Monte Carlo estimate of its logical error rate. We introduce QECFM, a two-stage conditional flow matching model with a deterministic group-algebra readout layer, which maps each continuous sample to discrete parity-check matrices and enforces CSS commutativity. We then use D-Flow with a differentiable noise-aware critic to optimise the source variables toward codes with lower logical error rates. We evaluate the resulting method across six code families, code-capacity and circuit-level noise, and block lengths up to $n = 288$. Our method finds codes with a lower median logical error rate than both learned and classical search baselines within the Bivariate Bicycle Code family. Under code-capacity depolarising noise, our best code also outperforms the Gross code at different physical error rates.

1 INTRODUCTION

Physical qubits accumulate errors during gates, measurement, and storage, making the long computations needed for quantum simulation and integer factorisation unreliable. Quantum error correction (QEC) protects information by encoding a logical qubit across several physical qubits (Shor, 1995). The choice of code determines how efficient the code is (encoding rate) and how effective the code is, i.e. how many errors can successfully be corrected. This is called logical error rate (LER).

Two relevant approaches to code design are algebraic construction and Bayesian optimisation. The first is to hand-design quantum error correcting codes via construction schemas such as low-density parity-check (QLDPC) and bivariate-bicycle (BB) (Breuckmann and Eberhardt, 2021; Bravyi et al., 2024). The other approach uses Bayesian optimisation and LER measurements to choose candidates in parameterised spaces (Chengyu et al., 2026). Constructions do not identify the best code for a given noise model and decoder. Bayesian optimisation, while more adaptive, is a compute-intensive search process. The Calderbank–Shor–Steane (CSS) codes (Calderbank and Shor, 1996; Steane, 1996) are represented by two sparse binary (parity-check) matrices, which are the incidence matrices of a Tanner graph. The matrices cannot be chosen independently as they need to commute, meaning their product needs to be zero. This makes discovery a constrained graph-generation problem with expensive black-box feedback.

We therefore introduce QECFM, a generative model that can propose valid CSS codes directly, trained on known constructions but capable of exploring beyond them. We learn a continuous representation of these coupled graphs through their Tanner-Laplacian spectra. A two-stage flow matching prior proposes spectral states, D-Flow (Ben-Hamu et al., 2024) refines their source variables, and a deterministic readout layer over a group algebra ensures valid CSS code generation. We train the flow matching model on BB codes and test its out-of-family generalisation to other group-algebra families.

We compare learned search with random search, simulated annealing, and Bayesian optimisation across six families, code-capacity noise (perfect syndrome extraction) and circuit-level noise (faulty measurements), and block lengths up to $n = 288$. We separately measure fixed codes: at $p = 0.003$, our selected BB code has LER 3.89×10^{-6} versus 1.63×10^{-5} for the published Gross code under the same code-capacity evaluator.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Figure 1 shows how our method (QECFM) concentrates samples of valid codes which makes a search with D-Flow (Ben-Hamu et al., 2024) more effective.

Contributions In brief, our contributions are:

- **A generative spectral flow matching model for QEC code design.** We introduce a two-stage flow matching model QECFM that generates Tanner-Laplacian spectra and supports multiple QEC code families.
- **Differentiable structural code search.** We train a critic to predict logical error rates from the parity-check matrices and their spectra, enabling gradient-based optimisation through the generative flow. We dub this method QECFM+DFlow.
- **Evaluation across families, scales, and noise models.** We compare learned and classical search methods across six code families, code-capacity and circuit-level noise, and block lengths up to $n = 288$, including matched-compute evaluations. We observe a more effective search, whose advantage grows with the size of the code.

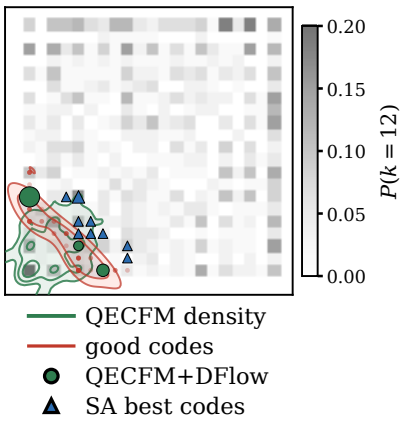

Figure 1: Sample density projection of QECFM and Simulated Annealing (SA). QECFM density significantly overlaps with the density of good QEC codes. QECFM+DFlow moves the samples towards more optimal regions

2 Background

Error correction and CSS codes Error-correcting codes protect information sent through a noisy channel by adding redundancy. A binary linear code is specified by a parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{m \times n}$, with valid codewords $\mathbf{x} \in \mathbb{F}_2^n$ satisfying

$$\mathbf{H}\mathbf{x} = \mathbf{0}. \tag{1}$$

If the channel adds an error $\mathbf{e}$, the received word is $\mathbf{y} = \mathbf{x} + \mathbf{e}$. Its syndrome is

$$\mathbf{s} = \mathbf{H}\mathbf{y} = \mathbf{H}\mathbf{e}, \tag{2}$$

which identifies the violated parity checks without depending on the transmitted codeword. A decoder then finds a likely error consistent with this syndrome.

Quantum error correction follows the same principle, but quantum noise has both bit- and phase-error components. An X error flips a qubit in the computational basis, a Z error changes its phase, and a Y error combines both. On quantum hardware, these errors arise from imperfect gates, measurements, and idle operations and accumulate with circuit depth (Georgopoulos et al., 2021; Preskill, 2018). To protect the encoded state, k logical qubits are embedded into $n > k$ physical qubits, and commuting stabiliser generators are measured to obtain error syndromes without revealing the logical information and hence destroying the system (Gottesman, 1997).

CSS codes separate the two error components using a pair of binary parity-check matrices (Calderbank and Shor, 1996; Steane, 1996),

$$\mathbf{H}_X \in \mathbb{F}_2^{m_X \times n}, \qquad \mathbf{H}_Z \in \mathbb{F}_2^{m_Z \times n}. \tag{3}$$

The X checks detect Z errors, while the Z checks detect X errors:

$$\mathbf{s}_X = \mathbf{H}_X \mathbf{e}_Z, \qquad \mathbf{s}_Z = \mathbf{H}_Z \mathbf{e}_X. \tag{4}$$

Thus, a CSS code resembles two coupled classical linear codes. The coupling is essential: their stabiliser checks must commute, which requires

$$\mathbf{H}_X \mathbf{H}_Z^{\mathsf{T}} = \mathbf{0} \qquad \text{over } \mathbb{F}_2. \tag{5}$$

The resulting code has parameters $[[n, k, d]]$, where n is the number of physical qubits, k the number of encoded logical qubits and d the code distance, meaning how many errors can be corrected. Each check matrix $\mathbf{H}$ also defines a bipartite Tanner graph $\mathcal{T}(\mathbf{H})$, with one check node per row, one qubit node per column, and an edge (i, j) whenever $H_{ij} = 1$.

Satisfying Eq. (5) is straightforward when the matrices are constructed algebraically. Two randomly sampled unstructured matrices are unlikely to fulfil Eq. (5). For a fixed $\mathbf{H}_X$ of rank r, a matrix $\mathbf{H}_Z$ with m_Z independently sampled rows satisfies Eq. (5) with probability 2^{-rm_Z}. Useful QLDPC codes must also maintain sparse checks, nonzero rate, sufficient distance, and low logical error under a chosen decoder. Product constructions address the first difficulty by enforcing commutativity by construction (Breuckmann and Eberhardt, 2021).

Decoding and the objective Two Z-errors are physically equivalent when they differ by an element of $\mathrm{rowspan}(\mathbf{H}_Z)$, since their difference is a *stabiliser* and therefore acts trivially on the code space. The decoder must consequently identify the most probable equivalence class consistent with the observed syndrome rather than the exact physical error. The logical error rate (LER) is the probability that, after physical noise and decoding, at least one encoded logical qubit is recovered incorrectly. We estimate it as the fraction of Monte Carlo trials ending in such a failure under a fixed noise model and decoder.

Distance alone is insufficient to predict finite-size code performance. For a fixed physical error rate, syndrome-extraction circuit, decoder, and circuit-level noise model, the logical error rate is of interest. However, a higher code distance corresponds to a lower logical error rate (Google Quantum AI and Collaborators, 2025). The logical error rate is typically estimated through repeated noisy-circuit simulation and Monte Carlo sampling and counting the number of decoding failures (Gidney, 2021). The resulting objective is stochastic, non-differentiable, and expensive to evaluate, particularly when enough samples are required to resolve rare logical failures. This makes search procedures requiring large numbers of sequential, high-precision objective evaluations impractical.

The topological view of QEC codes The CSS commutation condition in Eq. (5) turns the sequence

$$\mathbb{F}_2^{m_Z} \xrightarrow{\partial_2} \mathbb{F}_2^{n} \xrightarrow{\partial_1} \mathbb{F}_2^{m_X}, \qquad \partial_2 = \mathbf{H}_Z^\top, \quad \partial_1 = \mathbf{H}_X, \tag{6}$$

into a three-term chain complex, since $\partial_1\partial_2 = 0$ (Hatcher, 2002; Bombin and Martin-Delgado, 2007). The qubits index a basis of the degree-one space C_1, while the Z- and X-checks index bases of C_2 and C_0, respectively. In the Z-error sector, undetectable errors are cycles and Z-stabilisers are boundaries.

When this abstract chain complex is the cellular chain complex of a finite CW complex, its basis elements correlate to the cell structure of those complexes: qubits as 1-cells, Z-checks as 2-cells, and X-checks as 0-cells. Under suitable locality and bounded-weight conditions, this construction yields a homological topological code like the toric code (Kitaev, 2003) as shown in Fig. 2.

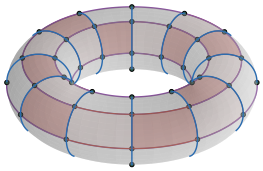

Figure 2: A $[[144, 2, \cdot]]$ bivariate bicycle code as a cell complex on the torus: • X-checks are vertices, — and — qubits are edges, and shaded Z-checks are faces. 72 vertices, 144 edges, 72 faces, $\chi = 0$.

Code family Quantum error correcting codes are categorised into various families (Albert and Faist, 2026). We visualised the relation between a few prominent families in Fig. 7. We focus on QLDPC product codes (Lin and Pryadko, 2024) because they combine sparse checks, finite rate, and low-overhead syndrome extraction. One prominent product category is the BB family to which the $[[144, 12, 12]]$ Gross code belongs which is a leading candidate for early fault-tolerant quantum memory (Bravyi et al., 2024). For a finite group G and $a, b \in \mathbb{F}_2[G]$, let

$$\mathbf{A} = \sum_{g \in \mathrm{supp}(a)} \rho_L(g), \qquad \mathbf{B} = \sum_{h \in \mathrm{supp}(b)} \rho_R(h), \qquad \mathbf{H}_X = [\mathbf{A} \mid \mathbf{B}], \quad \mathbf{H}_Z = [\mathbf{B}^\top \mid \mathbf{A}^\top], \tag{7}$$

where ρ_L and ρ_R are the left and right regular representations. For $a = \sum_{g \in G} a_g g$, we write $\mathrm{supp}(a) = g \in G \neq 0$ for the support of (a), and define $\mathrm{supp}(b)$ analogously. Because these representations commute, $\mathbf{AB} = \mathbf{BA}$ and, over $\mathbb{F}_2$, $\mathbf{H}_X\mathbf{H}_Z^\top = \mathbf{AB} + \mathbf{BA} = \mathbf{0}$. Hence, Eq. (5) holds by construction. Bivariate-bicycle (BB) codes correspond to $G = \mathbb{Z}_\ell \times \mathbb{Z}_g$. Here, ℓ and g are the orders of the two cyclic factors, so each BB code has block length $n = 2|G| = 2\ell g$. In particular, $(\ell, g) = (12, 6)$ gives $n = 144$. The Gross $[[144, 12, 12]]$ code is one member of this family (Bravyi et al., 2024).

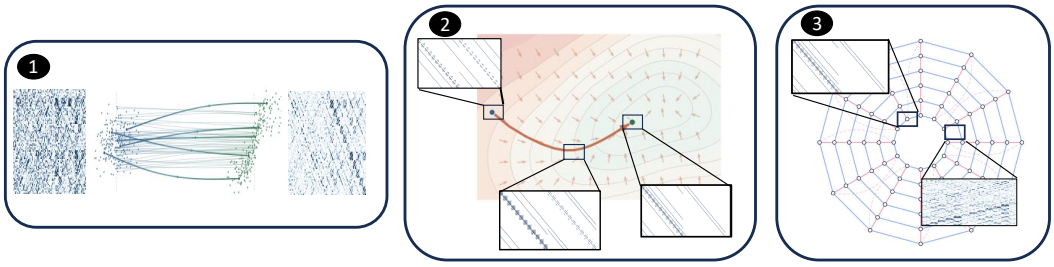

Figure 3: Method overview. 1) A conditional flow generates the Tanner spectra of $\mathbf{H}_X$ and $\mathbf{H}_Z$; 2) D-Flow optimises its source variables; and 3) the structured readout layer maps the resulting soft matrices onto a target group algebra.

3 Structured Spectral Flow Matching

Discovering a useful code poses two demands. First, we need a searchable representation that covers a useful range of structured codes without searching over every binary entry of their check matrices. Second, within that distribution, we need to find codes with a low logical error rate (LER) for a specified noise model and decoder.

We address the representation problem with QECFM. We start by learning a flow over Tanner-graph spectra from a newly generated dataset of valid BB codes. This gives us continuous representation through which candidate codes can be generated. A structured readout layer maps the resulting soft Tanner graph matrices to valid group-algebra codes. To address the selection problem, we train a separate critic model. This model is trained on a second dataset, which contains codes and their measured LERs. The model predicts the LER based on the code spectrum. We then optimise the source variables of the frozen flow toward low predicted LER. We evaluate the resulting discrete codes by Monte Carlo simulation. Figure 3 summarises this process.

3.1 QECFM: Spectral flow matching model

Searching directly over a binary check matrix $\mathbf{H}_b \in \mathbb{F}_2^{m_b \times n}$ requires choosing among $m_b n$ coupled binary entries.

For each check type $b \in \{X, Z\}$, let $\mathcal{T}_b = \mathcal{T}(\mathbf{H}_b)$ be its Tanner graph, with n qubit nodes, m_b check nodes, and $N_b = n + m_b$ nodes in total. Denoting its adjacency and degree matrices by $\mathbf{A}_b$ and $\mathbf{D}_b$, respectively, its combinatorial Laplacian has the eigendecomposition

$$\mathbf{L}_b = \mathbf{D}_b - \mathbf{A}_b = \mathbf{U}^b \operatorname{diag}(\boldsymbol{\lambda}^b)(\mathbf{U}^b)^\top, \qquad \boldsymbol{\lambda}^b \in \mathbb{R}^{N_b}. \tag{8}$$

We represent a code by $((\boldsymbol{\lambda}^X, \mathbf{U}^X), (\boldsymbol{\lambda}^Z, \mathbf{U}^Z))$. Following HOG-Diff (Huang and Birdal, 2026), we retain eigenvector bases sampled from the training corpus and model only the eigenvalue vectors. The generated state therefore contains $\mathcal{O}(N_b)$ variables per graph instead of the $\mathcal{O}(N_b^2)$ entries of its adjacency matrix.

The X and Z Tanner graphs jointly specify a code, so we model their spectra as a coupled pair. For an X-side spectrum, we define

$$\mathcal{S}^X = (\boldsymbol{\lambda}^X, \widetilde{\mathbf{A}}^X), \tag{9}$$

where $\widetilde{\mathbf{A}}^X$ is the relaxed Tanner adjacency reconstructed from $(\boldsymbol{\lambda}^X, \mathbf{U}^X)$. Let p_θ denote the model distribution over paired eigenvalue vectors, with the learnable parameters $\theta = (\theta_X, \theta_Z)$. We factorise

$$p_\theta(\boldsymbol{\lambda}^X, \boldsymbol{\lambda}^Z) = p_{\theta_X}(\boldsymbol{\lambda}^X)\, p_{\theta_Z}(\boldsymbol{\lambda}^Z \mid \mathcal{S}^X). \tag{10}$$

Thus, the p_{θ_X} captures the marginal spectral distribution, while p_{θ_Z} models its dependence on the corresponding X Tanner graph.

We implement the two factors with time-dependent graph neural velocity fields $\mathbf{v}_{\theta_X}$ and $\mathbf{v}_{\theta_Z}$. The X field is unconditional, whereas the Z field is conditioned on $\mathcal{S}^X$. Both fields are trained on paired code spectra using the standard conditional flow matching objective (Lipman et al., 2023), with a

mask excluding padded spectral entries. This training uses only the structural code corpus and does not require LER labels.

Let $\Phi_{\theta_X}^{t_s \to 0}$ and $\Phi_{\theta_Z}^{t_s \to 0}$ denote the reverse-time transports induced by the trained velocity fields. We call their sequential composition QECFM. Given independent source variables $\mathbf{z}^X$ and $\mathbf{z}^Z$, it first transports the X source, reconstructs the corresponding graph state, and then transports the Z source conditioned on that state:

$$\begin{aligned} \boldsymbol{\lambda}_0^X &= \Phi_{\theta_X}^{t_s \to 0}(\mathbf{z}^X), & \mathcal{S}_0^X &= (\boldsymbol{\lambda}_0^X, \widetilde{\mathbf{A}}_0^X), \\ \boldsymbol{\lambda}_0^Z &= \Phi_{\theta_Z}^{t_s \to 0}(\mathbf{z}^Z; \mathcal{S}_0^X), & \texttt{QECFM}_\theta^{t_s}(\mathbf{z}^X, \mathbf{z}^Z) &:= (\boldsymbol{\lambda}_0^X, \boldsymbol{\lambda}_0^Z). \end{aligned} \tag{11}$$

For ordinary generation, $t_s = 1$ and $\mathbf{z}^X, \mathbf{z}^Z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Thus, Eq. (11) defines our QEC code generative model QECFM.

3.2 Structured readout layer

QECFM produces continuous spectra, whereas a usable code requires binary check matrices satisfying the CSS commutation condition exactly. Independently thresholding reconstructed matrix entries does not ensure this condition. We instead use the soft entries to identify group-algebra supports and rebuild the check matrices through Eq. (7).

For a generated eigenvalue vector and its basis, we reconstruct a soft parity-check matrix:

$$\widehat{\mathbf{A}} = -\mathbf{U}\,\mathrm{diag}(\boldsymbol{\lambda})\mathbf{U}^\top, \qquad \mathbf{P} = (P_{iq}), \quad P_{iq} = \sigma\left(\frac{|\widehat{A}_{n+i,q}| - \tau}{T}\right), \tag{12}$$

with threshold $\tau = 0.5$ and temperature $T = 0.08$ and σ the sigmoid function. We split the soft matrix into its two group-algebra blocks and write $\mathbf{p}_h^A$ and $\mathbf{p}_h^B$ for their score vectors in row h. For a vector $\mathbf{p}$ indexed by G, let $\mathrm{top}_w(\mathbf{p}) \subseteq G$ denote the group elements indexing its w largest entries. For each $h \in G$ and $w \in \{2, 3, 4, 5, 6\}$, we translate these entries back to the identity:

$$\mathrm{supp}(a) = \{uh^{-1} : u \in \mathrm{top}_w(\mathbf{p}_h^A)\}, \qquad \mathrm{supp}(b) = \{h^{-1}u : u \in \mathrm{top}_w(\mathbf{p}_h^B)\}. \tag{13}$$

We then rebuild $(\mathbf{H}_X, \mathbf{H}_Z)$ using Eq. (7), so every projected candidate satisfies Eq. (5). Candidates are deduplicated and ordered by agreement with the requested k and by projection defect, the Hamming distance from the rounded reconstruction.

3.3 Optimising the source variables

The learned flow describes plausible code structure, but it does not distinguish codes by performance under a particular noise model and decoder. That performance is measured by LER, whose Monte Carlo estimate is expensive and cannot be differentiated through. We therefore train a critic $\widehat{s}$ on a second dataset of BB codes with measured LERs. Given both check matrices and their spectra, it predicts log-LER. The implementation details are reported in sec. A.2.

QECFM provides a differentiable map from source variables to candidate spectra. We keep this map fixed and optimise its source variables using the critic's gradient, following the source-optimisation approach of D-Flow (Ben-Hamu et al., 2024).

Algorithm 1 Code search with QECFM+DFlow

Require: Frozen flow Φ, frozen critic $\widehat{s}$, CSS penalty $\mathcal{C}$, search steps T
1: Initialise source spectra $\mathbf{z}$
2: **for** $j = 0, \ldots, T$ **do**
3: Generate spectra $(\boldsymbol{\lambda}^X, \boldsymbol{\lambda}^Z) \leftarrow \Phi(\mathbf{z})$
4: Reconstruct soft check matrices $\mathbf{P}^X, \mathbf{P}^Z$
5: Compute the critic-plus-CSS objective from Eq. (14) as $J(\mathbf{z})$
6: Update $\mathbf{z}$ using $\nabla_{\mathbf{z}} J(\mathbf{z})$
7: **end for**
8: Project to exact CSS code
9: Evaluate code with MC sampling
10: **return** the code

Writing $\Phi = \Phi_\theta^{t_s}$ for the frozen flow, we optimise:

$$\mathbf{z}^\star = \arg\min_{\mathbf{z}} \left[\lambda_{\mathrm{CSS}}\mathcal{C}(\mathbf{P}^X, \mathbf{P}^Z) + \lambda_q \widehat{s}(\mathbf{P}^X, \mathbf{P}^Z, \boldsymbol{\lambda}^X, \boldsymbol{\lambda}^Z)\right], \qquad (\boldsymbol{\lambda}^X, \boldsymbol{\lambda}^Z) = \Phi(\mathbf{z}). \tag{14}$$

The critic $\widehat{s}$ predicts log-LER. During optimisation, we also regularise the soft check matrices $\mathbf{P}^b \in [0,1]^{m\times n}$ from Eq. (12), where $b \in \{X, Z\}$ and $m = n/2$. Their row sums $r_i^b = \sum_q P_{iq}^b$ and column sums $c_q^b = \sum_i P_{iq}^b$ give the soft check and qubit degrees. We use

$$\mathcal{C}(\mathbf{P}^X, \mathbf{P}^Z) = \sum_{b\in\{X,Z\}} \left[\frac{1}{m} \sum_i \phi_{\delta_r}(r_i^b) + \frac{1}{n} \sum_q \phi_{\delta_c}(c_q^b) \right], \qquad \phi_\delta(x) = [1-x]_+^2 + 0.1[x-\delta]_+^2, \tag{15}$$

where $[x]_+ = \max(x, 0)$, $\delta_r = \max(4, n/4)$, and $\delta_c = \max(4, m/4)$.

The term $\mathcal{C}$ discourages empty or overly connected checks and qubits while the candidate remains continuous. It guides the search toward useful soft matrices; the algebraic readout supplies exact CSS validity after optimisation.

At each search step, the flow maps the updated source variables to paired spectra, from which we reconstruct soft check matrices and evaluate the critic and degree regulariser. We then project the resulting candidates to exact codes and measure their LER independently by Monte Carlo simulation. Thus, the critic guides which candidates to test, while the final selection uses measured performance. All thresholds, loss weights, architecture choices, optimisation settings, dataset statistics, and evaluation budgets are reported in Appendix A.1.

4 Experiments

Data and code families. We evaluate codes across the bivariate-bicycle family (Bravyi et al., 2024) and five two-block group-algebra families (Lin and Pryadko, 2024). `QECFM` is trained only on BB codes. All reported candidates encode $k = 12$ logical qubits, with block length varied only in the scaling study.

Baselines. We compare our `QECFM`+DFlow setup with a diffusion model, which is trained analogously to our flow matching model and the HOG-Diff + DFlow, unguided HOG-Diff (Huang and Birdal, 2026), Bayesian optimisation (Chengyu et al., 2026), simulated annealing, and random search. Classical methods search each family's native parameter space, and all methods share the same evaluator within each comparison.

Metrics and protocols. The primary metric is LER, where lower is better. Code-capacity experiments use depolarising noise and a BP-OSD decoder (Roffe et al., 2020). Circuit-level experiments use SI1000 syndrome extraction implemented with Stim (Gidney, 2021) and Relay-BP decoding. We additionally report the target-dimension rate for structural feasibility and the number of full Monte Carlo evaluations for search cost. Full protocols are given in Appendix A.

4.1 Search efficiency under equal compute

In the first experiment we test the search efficiency of various methods when using an equal number of optimisation steps.

Figure 4 compares the first 256 optimisation steps and the LER of the samples. We notice that our method `QECFM`+DFlow improves steadily as evaluations accumulate, showing that the learned prior provides a search space, in which it is easier to optimise compared to the baselines. The separate 600-evaluation comparison is summarised in Table 1. `QECFM`+DFlow has the lowest median winner LER, 0.0310, compared with 0.0333 for diffusion D-Flow, 0.0495 for random search, and 0.0519 for simulated annealing. This shows that the learned factorised spectral generation with flow matching or diffusion provides the most suitable representation to find QEC codes with a low LER.

4.2 Scaling with block length

Next we investigate how well our method is able to scale up to codes with more physical qubits. Figure 5 shows the scaling capabilities of our method.

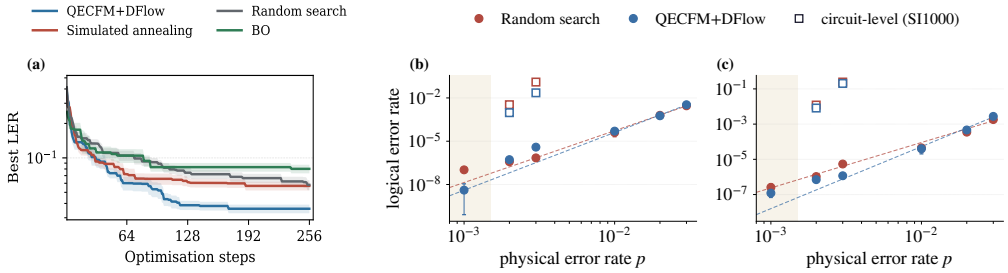

Figure 4: **(a)** Search efficiency for $[[144, 12]]$ BB codes at $p = 0.05$. Lines and bands show mean selected LER $\pm$ one standard error through 256 evaluations. All methods use ten runs. BO begins at evaluation 42 as it needs an initial pool for the optimisation procedure. (b)-(c) Low-p performance under code-capacity noise (filled markers, 95% Poisson intervals) and SI1000 circuit-level noise (open squares). **(b)** In-family BB: FM retains its advantage under both noise models. **(c)** Zero-shot D_{36}, one of the three of five audited families in which FM leads at circuit-level (Appendix Table 6).

Table 1: Matched-search comparison at $p = 0.05$ on $[[144, 12]]$ BB codes. We optimise over 600 steps and evaluate the code with 1.2×10^7 true-MC shots per run. We report the median run-level winner with a percentile-bootstrap 95% interval. All setups use 10 runs.

Method	Median best LER ↓	Best code (95% CI)	Raw target-k yield ratio
Random search	0.0495 ± 0.0180	0.0305 [0.0282, 0.0330]	0.022
Simulated annealing	0.0519 ± 0.0092	0.0377 [0.0351, 0.0404]	0.025
HOG-Diff	0.0522 ± 0.0076	0.0456 [0.0428, 0.0486]	0.798
HOG-Diff + DFlow	0.0952 ± 0.0099	0.0873 [0.0834, 0.0913]	0.219
Bayesian optimisation	0.0679 ± 0.0192	0.0382 [0.0371, 0.0393]	0.161
QECFM	0.0361 ± 0.0047	0.0250 [0.0229, 0.0273]	0.959
Diffusion + D-Flow	0.0333 ± 0.0047	0.0293 [0.0270, 0.0317]	0.901
QECFM+DFlow (ours)	$\mathbf{0.0310 \pm 0.0068}$	**0.0247** [0.0226, 0.0269]	0.959

QECFM+DFlow trails classical search at the smallest scale, is statistically comparable at the intermediate scale, and separates from both baselines at the larger scales. This pattern is consistent with a learned prior becoming more valuable as the structured search space grows.

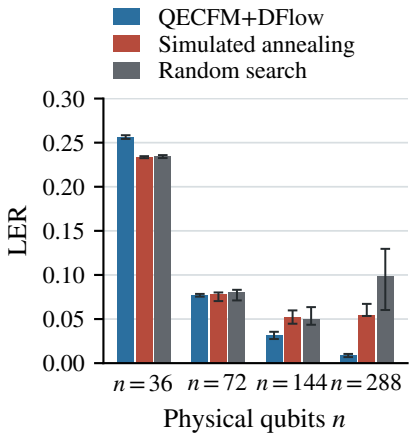

Figure 5: Scaling at k = 12 and p = 0.05. Each setup optimises for 600 steps. All reported numbers are the median LER of the codes across different seeds.

4.3 Robustness across noise regimes

The performance of a code changes with different noise levels (Fowler et al., 2012). Figure 4 shows that the QECFM+DFlow found code retains its advantage under both code-capacity and circuit-level noise. For the fixed BB pair, the circuit-level advantage is 3.6–5.6× at $p \in \{0.002, 0.003\}$ (Table 6). The zero-shot D_{36} code also remains below random search at both measured circuit points, showing that the transferred readout can survive a simultaneous change of family, noise model, and decoder. We further plotted two circuit-level noise measurements. The remaining circuit-level noise measurements are in the Appendix Table 6. Here, we similarly observe that with a change in physical error rate our found codes remain competitive compared to the ones found with random search.

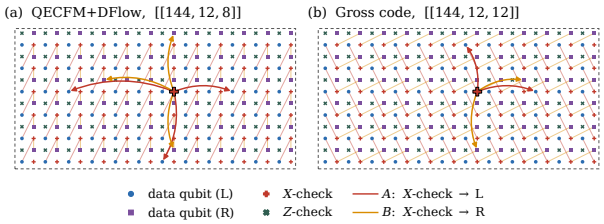

Figure 6: Bi-planar layouts of two $[[144, 12, \cdot]]$ BB codes on a 12×6 torus.

Table 2: Code-capacity comparison with the published Gross $[[144, 12, 12]]$ code in the same BP-OSD setup. We report the LER for different physical error rates.

p	QECFM+DFlow	Gross
0.05	2.59×10^{-2}	5.93×10^{-2}
0.01	5.00×10^{-5}	6.30×10^{-4}
0.003	3.89×10^{-6}	1.63×10^{-5}
0.002	5.30×10^{-7}	3.83×10^{-6}

Table 3: Transfer and noise robustness across six code families. The QECFM+DFlow model is trained on $\mathbb{Z}_{12} \times \mathbb{Z}_6$. Depolarising LER is measured at $p = 10^{-3}$ and Z- and X-heavy LERs are measured at $p = 0.01$ and reported in units of 10^{-5}.

	Target-k yield		Depolarising LER, $p = 10^{-3}$			Z-heavy LER, $p = 0.01$			X-heavy LER, $p = 0.01$		
Family	QECFM+DFlow	Random	QECFM+DFlow	SA	Random	QECFM+DFlow	SA	Random	QECFM+DFlow	SA	Random
$\mathbb{Z}_{12} \times \mathbb{Z}_6$ (BB)	**0.9594**	0.0112	$\mathbf{4.08 \times 10^{-9}}$	5.00×10^{-7}	1.04×10^{-7}	**9.20**	22.85	11.00	**10.60**	24.95	12.05
$A_4 \times \mathbb{Z}_6$	**0.9998**	0.0348	$\mathbf{1.18 \times 10^{-7}}$	2.66×10^{-7}	1.92×10^{-7}	**8.90**	15.95	15.80	**9.85**	16.95	14.70
$S_4 \times \mathbb{Z}_3$	**0.9986**	0.0276	$\mathbf{2.08 \times 10^{-7}}$	3.47×10^{-7}	3.33×10^{-7}	**12.15**	26.20	20.45	**13.40**	26.05	20.55
$D_4 \times \mathbb{Z}_9$	**0.8584**	0.0047	7.30×10^{-8}	$\mathbf{5.62 \times 10^{-9}}$	2.91×10^{-7}	6.05	**5.05**	11.15	6.10	**3.95**	8.85
D_{36}	**0.9655**	0.0096	$\mathbf{1.21 \times 10^{-7}}$	5.00×10^{-7}	2.56×10^{-7}	**9.35**	11.15	10.00	**8.20**	10.50	8.80
$\mathbb{Z}_6 \times \mathbb{Z}_6 \times \mathbb{Z}_2$	**1.0000**	0.0227	2.10×10^{-5}	$\mathbf{4.26 \times 10^{-7}}$	4.64×10^{-7}	12.75	**10.85**	12.30	**12.20**	14.10	13.85

4.4 Gross Code Comparison

We further compare in Table 2 the selected BB code with the published Gross code (Bravyi et al., 2024) in the same code-capacity setup. Our code performs better across the well-resolved operating points. The gain is largest at $p = 0.01$ ($12.6\times$) and remains $4.2\times$ and $7.2\times$ at $p = 0.003$ and $p = 0.002$.

4.5 Transfer Across Code Families

The structured projection enforces CSS commutativity by construction, so the remaining feasibility question is whether a projected code is good, meaning having a low logical error rate and a high encoding rate. When we test 10,000 samples per family, FM attains the target encoding rate of 83.8–100%, compared with 0.31–3.48% for random construction (Table 3).

However, Table 3 also shows that structural transfer does not guarantee uniform transfer in code quality. At $p = 10^{-3}$, the QECFM+DFlow point estimate is lower than random search in five of six families and lower in four code families compared to simulated annealing. Lastly, we investigate if QECFM is discovering new codes or simply generates codes from the training set. Overall, we achieve 99.9% novelty in the discovered code sets. Together, these results support transfer of structural validity and broad sampling coverage, with quality remaining family dependent.

4.6 Guidance Ablation

Lastly, we investigate the effect of using D-Flow as an optimisation method. We compare it to a no guidance setup, which is comparable to random search on the output of the flow matching model, or to use a random gradient or no gradient, when optimising the code. We observe in Table 4 that D-Flow helps to find codes with a lower LER compared to using no guidance. Here, overwriting the gradient, which is used in D-Flow to optimise the sample, is more harmful than not having any guidance at all.

Table 4: Low-p guidance ablation at $p = 0.003$. We use 20 million MC circuit simulations for each measurement.

Arm	LER ↓	Events	95% CI
QECFM+DFlow	$\mathbf{3.00 \times 10^{-6}}$	60	$[2.29 \times 10^{-6}, 3.86 \times 10^{-6}]$
$\lambda_q = 0$ (no guidance)	5.75×10^{-6}	115	$[4.75 \times 10^{-6}, 6.90 \times 10^{-6}]$
Random-guidance	6.90×10^{-6}	138	$[5.80 \times 10^{-6}, 8.15 \times 10^{-6}]$
Zero-guidance	6.90×10^{-6}	138	$[5.80 \times 10^{-6}, 8.15 \times 10^{-6}]$

5 Related Work

Quantum error-correcting (QEC) code design QEC codes have traditionally been handcrafted. For those methods, general code families have been developed such as QLDPC (Breuckmann and Eberhardt, 2021) and bivariate-bicycle codes (Bravyi et al., 2024). While being effective design frameworks the optimisation here needs to be done manually (Fowler et al., 2009). Most of learning-based QEC work primarily targets decoding, including AlphaQubit (Senior et al., 2026), hypergraph neural decoders (Bhave et al., 2026), and generative models (Cao et al., 2023). However, code discovery is a much harder problem, as it needs to ensure the validity of the code, which is a hard algebraic problem.

Generative modelling and discrete diffusion Continuous diffusion models can generate graphs and molecules through learned stochastic dynamics (Hoogeboom et al., 2022; Jo et al., 2022). Discrete denoising models instead evolve categorical states directly (Austin et al., 2021; Campbell et al., 2022; Vignac et al., 2023; Chen et al., 2023; Yin et al., 2025). Further, topological generative models like (Huang and Birdal, 2026) can generate valid higher order structures, which fulfil the algebraic constraints.

Constraint-aware generation Classifier guidance and energy-based composition steer samples toward soft constraints (Dhariwal and Nichol, 2021; Du et al., 2023). More recent approaches use dual training, stochastic projection, or inference-time resampling for stricter requirements (Khalafi et al., 2024; Prabhakar et al., 2025; Ou et al., 2026). Our approach places the exact algebraic constraint in a deterministic readout layer.

Automated QEC discovery Most work approaches code discovery as search over codes or their construction procedures. Reinforcement learning assembles codes from reusable primitives or jointly searches for codes and encoding circuits under hardware and noise constraints (Nautrup et al., 2019; Su et al., 2025; Olle et al., 2024). Variational methods optimise shallow encoders using the Knill-Laflamme conditions (Cao et al., 2022), while genetic algorithms evolve stabiliser codes or encoding circuits (Tandeitnik and Guerreiro, 2024; Webster and Browne, 2025). Bayesian optimisation reduces the number of costly logical-error evaluations through learned code embeddings (Chengyu et al., 2026). Further, SAT and game-theoretic methods provide alternative search formulations (DalFavero et al., 2026; Guerrero, 2026). Most recently LLM systems evolve executable generators for BB and lifted-product codes (Cruz-Benito et al., 2026; Yan et al., 2026; Qian and Li, 2026). We instead learn a reusable prior and do not use a pretrained LLM for the code candidate generation.

6 Conclusion

We introduced `QECFM`+DFlow a constrained generative framework for QEC code discovery that combines a compressed Tanner-Laplacian state, a flow matching prior, source-space D-Flow optimisation, and a deterministic group-algebra readout. The readout enforces CSS commutativity exactly, separating algebraic feasibility from performance optimisation. While `QECFM` is trained on BB codes the search transfers structurally to five unseen families without retraining.

Limitations As for any learned prior, our model requires a training dataset and also the critic model requires precomputed LERs of more codes than one random search investigates. Hence, our approach is like any other neural representation compute intensive. Further, the model only works well on certain product codes, while on others, classical search methods are able to find better codes. `QECFM`+DFlow optimises LER for a fixed decoder and noise model: despite its lower distance, our selected $[\![144, 12, 8]\!]$ BB code outperforms the $[\![144, 12, 12]\!]$ Gross code under BP-OSD at the evaluated physical error rates because finite-rate performance depends on both harmful error patterns and decoder behaviour (Table 2).

Future Work In the next steps, we see a generative model over all possible QEC code families. Further, we see potential in adding circuit design to the pipeline, as the implementation of codes also relies on the physical realisation.

AI DISCLOSURE

In this work, we used generative AI tools for aiding to implement the experiments and to set up the code to generate the datasets. Generative models also provided feedback on the experimental setups as well as polishing the text. We have reviewed and tested all AI assisted code and dataset. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide anonymised supplementary code for training QECFM and the LER critic, running D-Flow, constructing the baseline methods, and reproducing the reported evaluations. The appendix specifies the implementation details further. Upon acceptance we will further make the code and dataset public. Further we provide the parameter vector and reconstruction recipe for the selected $[\![144, 12, 8]\!]$ BB code.

ETHICS STATEMENT

This work does not involve human participants, personal information, or sensitive datasets. The training and evaluation data consist of algebraically constructed quantum codes and simulated decoding outcomes. The work studies foundational methods for quantum error correction and does not introduce an immediate human-facing application. More efficient quantum error correction may advance fault-tolerant quantum computing, with its scientific benefits in optimisation, material science as well as its potential harmful applications especially in cryptography. Our results used a simulated noise model and the found code performance would need to be validated on quantum emulators and hardware. The authors have read and adhered to the ICLR Code of Ethics.

REFERENCES

Victor V. Albert and Philippe Faist, editors. *The Error Correction Zoo*. 2026. doi: 10.48550/arXiv.2606.11484. URL https://errorcorrectionzoo.org/.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pages 17981–17993, 2021.

Heli Ben-Hamu, Omri Puny, Itai Gat, Brian Karrer, Uriel Singer, and Yaron Lipman. D-flow: Differentiating through flows for controlled generation. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=SE20BFqj6J.

Ameya S. Bhave, Navnil Choudhury, and Kanad Basu. Hypernq: A hypergraph neural network decoder for quantum ldpc codes. In *ICC 2026 - IEEE International Conference on Communications*, pages 1–7, 2026. doi: 10.1109/ICC59461.2026.11587226.

Hector Bombin and Miguel A Martin-Delgado. Homological error correction: Classical and quantum codes. *Journal of mathematical physics*, 48(5), 2007.

Sergey Bravyi, Andrew W Cross, Jay M Gambetta, Dmitri Maslov, Patrick Rall, and Theodore J Yoder. High-threshold and low-overhead fault-tolerant quantum memory. *Nature*, 627(8005): 778–782, 2024.

Nikolas P. Breuckmann and Jens Niklas Eberhardt. Quantum low-density parity-check codes. *PRX Quantum*, 2:040101, Oct 2021. doi: 10.1103/PRXQuantum.2.040101. URL https://link.aps.org/doi/10.1103/PRXQuantum.2.040101.

A Robert Calderbank and Peter W Shor. Good quantum error-correcting codes exist. *Physical Review A*, 54(2):1098, 1996.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*, 35:28266–28279, 2022.

Chenfeng Cao, Chao Zhang, Zipeng Wu, Markus Grassl, and Bei Zeng. Quantum variational learning for quantum error-correcting codes. *Quantum*, 6:828, 2022.

Hanyan Cao, Feng Pan, Yijia Wang, and Pan Zhang. qecgpt: Decoding quantum error-correcting codes with generative pre-trained transformers. *arXiv preprint arXiv:2307.09025*, 2023.

Xiaohui Chen, Jiaxing He, Xu Han, and Li-Ping Liu. Efficient and degree-guided graph generation via discrete diffusion modeling. In *International Conference on Machine Learning*, pages 4585–4610. PMLR, 2023.

Yihua Chengyu, Richard Meister, Conor Carty, Sheng-Ku Lin, and Roberto Bondesan. Bayesian optimization for quantum error-correcting code discovery. *arXiv preprint arXiv:2601.18562*, 2026.

Juan Cruz-Benito, Andrew W Cross, David Kremer, and Ismael Faro. Evolutionary discovery of bivariate bicycle codes with llm-guided search. *arXiv preprint arXiv:2606.02418*, 2026.

Ben DalFavero, William M Watkins, Margarite L LaBorde, Vincent Russo, Ethan Egger, Gregory Quiroz, and Ryan LaRose. Satisfying quantum codes: Physics-informed and hardware-aware code design with sat solvers. *arXiv preprint arXiv:2608.23460*, 2026.

Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43(9):4452–4505, 2002.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Yilun Du, Conor Durkan, Robin Strudel, Joshua B Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Sussman Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc. In *International conference on machine learning*, pages 8489–8510. PMLR, 2023.

Austin G Fowler, Ashley M Stephens, and Peter Groszkowski. High-threshold universal quantum computation on the surface code. *Physical Review A—Atomic, Molecular, and Optical Physics*, 80 (5):052312, 2009.

Austin G. Fowler, Matteo Mariantoni, John M. Martinis, and Andrew N. Cleland. Surface codes: Towards practical large-scale quantum computation. *Phys. Rev. A*, 86:032324, Sep 2012. doi: 10.1103/PhysRevA.86.032324. URL https://link.aps.org/doi/10.1103/PhysRevA.86.032324.

Konstantinos Georgopoulos, Clive Emary, and Paolo Zuliani. Modeling and simulating the noisy behavior of near-term quantum computers. *Physical Review A*, 104(6):062432, 2021.

Craig Gidney. Stim: a fast stabilizer circuit simulator. *Quantum*, 5:497, 2021.

Google Quantum AI and Collaborators. Quantum error correction below the surface code threshold. *Nature*, 638(8052):920–926, 2025.

Daniel Gottesman. *Stabilizer codes and quantum error correction*. California Institute of Technology, 1997.

Rubén Darío Guerrero. Game-theoretic discovery of quantum error-correcting codes through nash equilibria. *Physical Review A*, 114(1):012465, 2026.

Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

Emiel Hoogeboom, Víctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3D. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 8867–8887. PMLR, 17–23 Jul 2022.

Yiming Huang and Tolga Birdal. HOG-diff: Higher-order guided diffusion for graph generation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=FZJow6BWiM.

Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 10362–10383. PMLR, 17–23 Jul 2022.

Shervin Khalafi, Dongsheng Ding, and Alejandro Ribeiro. Constrained diffusion models via dual training. *Advances in Neural Information Processing Systems*, 37:26543–26576, 2024.

A Yu Kitaev. Fault-tolerant quantum computation by anyons. *Annals of physics*, 303(1):2–30, 2003.

Alexey A. Kovalev and Leonid P. Pryadko. Quantum kronecker sum-product low-density parity-check codes with finite rate. *Phys. Rev. A*, 88:012311, Jul 2013. doi: 10.1103/PhysRevA.88.012311. URL https://link.aps.org/doi/10.1103/PhysRevA.88.012311.

Hsiang-Ku Lin and Leonid P. Pryadko. Quantum two-block group algebra codes. *Phys. Rev. A*, 109: 022407, Feb 2024. doi: 10.1103/PhysRevA.109.022407. URL https://link.aps.org/doi/10.1103/PhysRevA.109.022407.

Ting-Chun Lin, Adam Wills, and Min-Hsiu Hsieh. Geometrically local quantum and classical codes from subdivision. *arXiv preprint arXiv:2309.16104*, 2023.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

D.J.C. MacKay, G. Mitchison, and P.L. McFadden. Sparse-graph codes for quantum error correction. *IEEE Transactions on Information Theory*, 50(10):2315–2330, 2004. doi: 10.1109/TIT.2004.834737.

Hendrik Poulsen Nautrup, Nicolas Delfosse, Vedran Dunjko, Hans J. Briegel, and Nicolai Friis. Optimizing Quantum Error Correction Codes with Reinforcement Learning. *Quantum*, 3:215, December 2019. ISSN 2521-327X. doi: 10.22331/q-2019-12-16-215. URL https://doi.org/10.22331/q-2019-12-16-215.

Jan Olle, Remmy Zen, Matteo Puviani, and Florian Marquardt. Simultaneous discovery of quantum error correction codes and encoders with a noise-aware reinforcement learning agent. *npj Quantum Information*, 10(1):126, 2024.

Zijing Ou, Chinmay Pani, and Yingzhen Li. Inference-time scaling of discrete diffusion models via importance weighting and optimal proposal design. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=7wbrFQvfdH.

Pavel Panteleev and Gleb Kalachev. Quantum ldpc codes with almost linear minimum distance. *IEEE Transactions on Information Theory*, 68(1):213–229, 2022a. doi: 10.1109/TIT.2021.3119384.

Pavel Panteleev and Gleb Kalachev. Asymptotically good quantum and locally testable classical ldpc codes. In *Proceedings of the 54th annual ACM SIGACT symposium on theory of computing*, pages 375–388, 2022b.

Chinmay Prabhakar, Suprosanna Shit, Tamaz Amiranashvili, Hongwei Bran Li, and Bjoern Menze. Semantically Consistent Discrete Diffusion for 3D Biological Graph Modeling . In *proceedings of Medical Image Computing and Computer Assisted Intervention – MICCAI 2025*, volume LNCS 15971. Springer Nature Switzerland, September 2025.

John Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, 2018.

Dongheng Qian and Tianyi Li. Multi-agent discovery of practical quantum ldpc codes. *arXiv preprint arXiv:2608.08996*, 2026.

Eric M. Rains, R. H. Hardin, Peter W. Shor, and N. J. A. Sloane. A nonadditive quantum code. *Phys. Rev. Lett.*, 79:953–954, Aug 1997. doi: 10.1103/PhysRevLett.79.953. URL https://link.aps.org/doi/10.1103/PhysRevLett.79.953.

Joschka Roffe, David R. White, Simon Burton, and Earl Campbell. Decoding across the quantum low-density parity-check code landscape. *Phys. Rev. Res.*, 2:043423, Dec 2020. doi: 10.1103/PhysRevResearch.2.043423. URL https://link.aps.org/doi/10.1103/PhysRevResearch.2.043423.

Andrew W. Senior, Thomas Edlich, Francisco J. H. Heras, Lei M. Zhang, Oscar Higgott, James S. Spencer, Taylor Applebaum, Sam Blackwell, Justin Ledford, Akvilė Žemgulytė, Augustin Žídek, Noah Shutty, Andrew Cowie, Yin Li, George Holland, Peter Brooks, Charlie Beattie, Michael Newman, Alex Davies, Cody Jones, Sergio Boixo, Hartmut Neven, Pushmeet Kohli, and Johannes Bausch. A scalable and real-time neural decoder for topological quantum codes. *arXiv preprint arXiv:2512.07737*, 2026.

Peter W Shor. Scheme for reducing decoherence in quantum computer memory. *Physical review A*, 52(4):R2493, 1995.

Andrew Steane. Multiple-particle interference and quantum error correction. *Proceedings: Mathematical, Physical and Engineering Sciences*, pages 2551–2577, 1996.

Vincent Paul Su, ChunJun Cao, Hong-Ye Hu, Yariv Yanay, Charles Tahan, and Brian Swingle. Discovery of optimal quantum codes via reinforcement learning. *Phys. Rev. Appl.*, 23:034048, Mar 2025. doi: 10.1103/PhysRevApplied.23.034048. URL https://link.aps.org/doi/10.1103/PhysRevApplied.23.034048.

Daniel Tandeitnik and Thiago Guerreiro. Evolving quantum circuits. *Quantum Information Processing*, 23(3):109, 2024.

Jean-Pierre Tillich and Gilles Zémor. Quantum ldpc codes with positive rate and minimum distance proportional to the square root of the blocklength. *IEEE Transactions on Information Theory*, 60 (2):1193–1202, 2014. doi: 10.1109/TIT.2013.2292061.

Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=UaAD-Nu86WX.

Mark A Webster and Dan E Browne. Engineering quantum error correction codes using evolutionary algorithms. *IEEE Transactions on Quantum Engineering*, 6:1–14, 2025.

Ge Yan, Shanchuan Li, Pengyue Ma, Qixin Zhang, Pingchuan Ma, Jianping Wang, Min-Hsiu Hsieh, and Yuxuan Du. Omniqec: discovering practical quantum error-correcting codes by an ai scientist. *arXiv preprint arXiv:2607.25865*, 2026.

Shu Yin, Dongpeng Hou, Lianwei Wu, Xianghua Li, and Chao Gao. A prior-based discrete diffusion model for social graph generation. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, 2025.

APPENDIX

A Additional Experimental Details **14**

A.1 Experimental setup 14

A.2 Logical-error-rate critic implementation details 14

A.3 QECFM training and integration details 15

B Additional results **15**

B.1 Sampling Diversity 16

B.2 Encoding of the selected BB code. 17

C Relation between quantum-code families **17**

A ADDITIONAL EXPERIMENTAL DETAILS

A.1 EXPERIMENTAL SETUP

Data and training. We train `QECFM` on 30,000 unique BB codes with $(\ell, g) = (12, 6)$ with $k > 0$. The H_X and H_Z spectral flow models each use five message-passing layers with hidden dimension 256 and are trained for 20,000 iterations using Adam with learning rate 10^{-4} and batch size 8. The H_Z flow matching model is conditioned on the reconstructed H_X state as described in sec. 3.1. The LER critic is trained on 50,000 labelled codes, including 20,000 codes from the low-LER tail.

Code search. `QECFM`+DFlow optimises each source spectrum for 80 steps using Adam with learning rate 10^{-3}. We integrate the learned velocity field using four midpoint steps from $t_s = 0.7$ and weight the structural and predicted-quality objectives equally.

Evaluation. We evaluate code-capacity performance under depolarising and biased Pauli noise using BP-OSD with minimum-sum belief propagation and order-2 combination-sweep OSD. Circuit-level experiments use the SI1000 noise model and Relay-BP, with 12 syndrome-extraction rounds. Logical error rates are estimated by Monte Carlo simulation.

Computing resources. Experiments were implemented in PyTorch and run on NVIDIA H200 NVL GPUs. Training the H_X and H_Z flow models took approximately 2 h 14 min and 27 min, respectively. A single FM–DFlow search took approximately 37 minutes for $n = 36$ and 1 h 16 min for $n = 72$.

A.2 LOGICAL-ERROR-RATE CRITIC IMPLEMENTATION DETAILS

D-Flow requires an objective that can be differentiated with respect to the source spectra, whereas evaluating the logical error rate (LER) requires discrete decoding and Monte Carlo sampling. We therefore train a critic $\widehat{s}_\psi$ to estimate the log-LER of a candidate under the fixed noise model and decoder used to label its training corpus. Its inputs are the two check matrices and their Tanner-graph spectra,

$$\widehat{s}_\psi\big(\mathbf{P}^X, \mathbf{P}^Z, \boldsymbol{\lambda}^X, \boldsymbol{\lambda}^Z\big) \approx \log \mathrm{LER}(\mathbf{H}_X, \mathbf{H}_Z), \tag{16}$$

Architecture. The critic processes the X and Z checks in separate branches. In each branch, the rows and columns of $\mathbf{P}^b$ are embedded as tokens, with their degrees included as additional features. The spectrum $\boldsymbol{\lambda}^b$ is mapped to a further token. These tokens are passed through a three-layer Transformer of width 256, with weights shared between the two branches. The resulting X and Z representations interact through cross-attention. We then apply mean and maximum pooling and concatenate the pooled representations with summary features describing the degree distributions, check overlaps, soft ranks, and spectra. A two-layer MLP predicts the normalised log-LER.

Training objective. For a labelled code i, we use the standardised target

$$y_i = \frac{\log\big(\max\{\mathrm{LER}_i, 10^{-12}\}\big) - \mu}{\sigma}, \qquad s_i = f_\psi\big(\mathbf{P}_i^X, \mathbf{P}_i^Z, \boldsymbol{\lambda}_i^X, \boldsymbol{\lambda}_i^Z\big), \tag{17}$$

where μ and σ are computed on the training split. Since the critic is used to identify promising candidates, its primary objective is to preserve the ordering of low-LER codes. Let π order the targets in a minibatch such that $y_{\pi_1} \le \cdots \le y_{\pi_K}$, where $K = \min\{64, B\}$ and B is the batch size. We minimise the Plackett–Luce loss

$$\mathcal{L}_{\mathrm{PL}} = -\frac{1}{K-1} \sum_{r=1}^{K-1} \log \frac{\exp(-s_{\pi_r})}{\sum_{j=r}^{K} \exp(-s_{\pi_j})}. \tag{18}$$

The negative sign converts predicted log-LER into a utility for which larger values are preferred. We supplement the ranking objective with a Smooth-L1 regression loss on y_i and a binary cross-entropy loss $\mathcal{L}_k$ for predicting whether $k > 0$:

$$\mathcal{L}_{\mathrm{critic}} = \mathcal{L}_{\mathrm{PL}} + 0.25\,\mathcal{L}_{\mathrm{reg}} + 0.10\,\mathcal{L}_k. \tag{19}$$

The regression term calibrates the scale of the predictions, while the listwise term concentrates learning on the relative ordering needed during search.

A.3 QECFM TRAINING AND INTEGRATION DETAILS

In the main text we define `QECFM` in Eq. (11). Next we specify its training and sampling procedure.

Training. For each part of the flow matching model $b \in \{X, Z\}$, let $\boldsymbol{\lambda}_0^b$ be a training spectrum, $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $t \sim \mathcal{U}(0, 1)$. We use the linear path

$$\boldsymbol{\lambda}_t^b = (1-t)\boldsymbol{\lambda}_0^b + t\boldsymbol{\varepsilon}, \qquad \dot{\boldsymbol{\lambda}}_t^b = \boldsymbol{\varepsilon} - \boldsymbol{\lambda}_0^b, \tag{20}$$

and minimise

$$\mathcal{L}_{\mathrm{FM}}^b = \mathbb{E}\left[\frac{\left\|\mathbf{m} \odot \left[\mathbf{v}_{\theta_b}(\boldsymbol{\lambda}_t^b, t, \mathbf{c}_b) - (\boldsymbol{\varepsilon} - \boldsymbol{\lambda}_0^b)\right]\right\|_2^2}{\|\mathbf{m}\|_1}\right], \tag{21}$$

where $\mathbf{m}$ excludes padded spectral entries. The conditioning states are $\mathbf{c}_X = \emptyset$ and $\mathbf{c}_Z = \mathcal{S}^X$.

Sampling. Starting from $\boldsymbol{\lambda}_1^b \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, we integrate

$$\frac{\mathrm{d}\boldsymbol{\lambda}_t^b}{\mathrm{d}t} = \mathbf{v}_{\theta_b}(\boldsymbol{\lambda}_t^b, t, \mathbf{c}_b) \tag{22}$$

from $t = 1$ to $t = 0$. We first solve the unconditional X flow, construct $\mathcal{S}_0^X$, and then solve the Z flow conditioned on this state. We use a four-step midpoint solver for both flows.

B ADDITIONAL RESULTS

We further report additional results, which have been omitted from the experimental section.

Table 5 expands the main transfer summary of `QECFM`+DFlow, simulated annealing (annealing) and random search (random). We report how often we observed a logical error (events) and the number of sampled shots (trials) as well as the LER and the confidence interval of it.

Table 5: Family-level code-capacity LER at $p = 10^{-3}$. FM finds the code with the lowest LER versus random search in five of six families and versus annealing in four of six.

Family	method	events/trials	LER [95% CI]
$\mathbb{Z}_{12}\times\mathbb{Z}_6$	**QECFM+DFlow (ours)**	4/981M	$\mathbf{4.08 \times 10^{-9}}$ $[1.11 \times 10^{-9}, 1.04 \times 10^{-8}]$
	annealing	30/60M	5.00×10^{-7} $[3.37 \times 10^{-7}, 7.14 \times 10^{-7}]$
	random	30/289M	1.04×10^{-7} $[7.00 \times 10^{-8}, 1.48 \times 10^{-7}]$
$A_4\times\mathbb{Z}_6$	**QECFM+DFlow (ours)**	20/170M	$\mathbf{1.18 \times 10^{-7}}$ $[7.19 \times 10^{-8}, 1.82 \times 10^{-7}]$
	annealing	25/94M	2.66×10^{-7} $[1.72 \times 10^{-7}, 3.93 \times 10^{-7}]$
	random	20/104M	1.92×10^{-7} $[1.17 \times 10^{-7}, 2.97 \times 10^{-7}]$
$S_4\times\mathbb{Z}_3$	**QECFM+DFlow (ours)**	20/96M	$\mathbf{2.08 \times 10^{-7}}$ $[1.27 \times 10^{-7}, 3.22 \times 10^{-7}]$
	annealing	25/72M	3.47×10^{-7} $[2.25 \times 10^{-7}, 5.13 \times 10^{-7}]$
	random	20/60M	3.33×10^{-7} $[2.04 \times 10^{-7}, 5.15 \times 10^{-7}]$
D_{36}	**QECFM+DFlow (ours)**	20/165M	$\mathbf{1.21 \times 10^{-7}}$ $[7.40 \times 10^{-8}, 1.87 \times 10^{-7}]$
	annealing	26/52M	5.00×10^{-7} $[3.27 \times 10^{-7}, 7.33 \times 10^{-7}]$
	random	20/78M	2.56×10^{-7} $[1.57 \times 10^{-7}, 3.96 \times 10^{-7}]$
$D_4\times\mathbb{Z}_9$	QECFM+DFlow (ours)	20/274M	7.30×10^{-8} $[4.46 \times 10^{-8}, 1.13 \times 10^{-7}]$
	annealing	7/1246M	$\mathbf{5.62 \times 10^{-9}}$ $[2.26 \times 10^{-9}, 1.16 \times 10^{-8}]$
	random	25/86M	2.91×10^{-7} $[1.88 \times 10^{-7}, 4.29 \times 10^{-7}]$
$\mathbb{Z}_6\times\mathbb{Z}_6\times\mathbb{Z}_2$	QECFM+DFlow (ours)	21/1M	2.10×10^{-5} $[1.30 \times 10^{-5}, 3.21 \times 10^{-5}]$
	annealing	20/47M	$\mathbf{4.26 \times 10^{-7}}$ $[2.60 \times 10^{-7}, 6.57 \times 10^{-7}]$
	random	26/56M	4.64×10^{-7} $[3.03 \times 10^{-7}, 6.80 \times 10^{-7}]$

Further we measure the performance the codes found in Table 5 with circuit-level noise with the SI1000 noise model. In Table 6 we report the family-level circuit comparison together with channel-derived intervals and event/trial counts. Here we observe that in 3 out of the 5 code families we retain an advantage compared to the codes found by random search.

Table 6: Comparing the best found codes under the SI1000 noise and decoded with Relay-BP. The ratio is random/FM, so values above one favour ours.

Family	p	QECFM+DFlow [95% CI]; events/trials	random search [95% CI]; events/trials	random/FM
$\mathbb{Z}_{12}\times\mathbb{Z}_6$ (BB)	0.002	$\mathbf{9.50 \times 10^{-4}}$ $[4.51 \times 10^{-4}, 1.77 \times 10^{-3}]$ 19/40k	3.40×10^{-3} $[1.98 \times 10^{-3}, 5.43 \times 10^{-3}]$ 34/20k	3.58×
	0.003	$\mathbf{2.26 \times 10^{-2}}$ $[1.69 \times 10^{-2}, 2.96 \times 10^{-2}]$ 104/9,216	1.26×10^{-1} $[9.90 \times 10^{-2}, 1.57 \times 10^{-1}]$ 133/2,048	5.56×
$A_4\times\mathbb{Z}_6$	0.002	2.60×10^{-3} $[1.39 \times 10^{-3}, 4.44 \times 10^{-3}]$ 26/20k	$\mathbf{8.50 \times 10^{-4}}$ $[3.85 \times 10^{-4}, 1.64 \times 10^{-3}]$ 17/40k	0.33×
	0.003	1.11×10^{-1} $[8.55 \times 10^{-2}, 1.41 \times 10^{-1}]$ 117/2,048	$\mathbf{2.86 \times 10^{-2}}$ $[2.19 \times 10^{-2}, 3.67 \times 10^{-2}]$ 118/8,192	0.26×
$S_4\times\mathbb{Z}_3$	0.002	6.49×10^{-3} $[4.48 \times 10^{-3}, 9.11 \times 10^{-3}]$ 65/20k	$\mathbf{7.33 \times 10^{-4}}$ $[3.67 \times 10^{-4}, 1.31 \times 10^{-3}]$ 22/60k	0.11×
	0.003	1.79×10^{-1} $[1.48 \times 10^{-1}, 2.14 \times 10^{-1}]$ 192/2,048	$\mathbf{1.65 \times 10^{-2}}$ $[1.25 \times 10^{-2}, 2.15 \times 10^{-2}]$ 110/13,312	0.09×
D_{36}	0.002	$\mathbf{8.18 \times 10^{-3}}$ $[5.88 \times 10^{-3}, 1.11 \times 10^{-2}]$ 82/20k	1.21×10^{-2} $[9.00 \times 10^{-3}, 1.59 \times 10^{-2}]$ 99/17,168	1.48×
	0.003	$\mathbf{2.04 \times 10^{-1}}$ $[1.71 \times 10^{-1}, 2.41 \times 10^{-1}]$ 221/2,048	2.45×10^{-1} $[2.10 \times 10^{-1}, 2.83 \times 10^{-1}]$ 267/2,048	1.20×
$D_4\times\mathbb{Z}_9$	0.002	$\mathbf{7.88 \times 10^{-3}}$ $[5.63 \times 10^{-3}, 1.07 \times 10^{-2}]$ 79/20k	1.91×10^{-2} $[1.44 \times 10^{-2}, 2.49 \times 10^{-2}]$ 107/13,312	2.43×
	0.003	$\mathbf{2.13 \times 10^{-1}}$ $[1.79 \times 10^{-1}, 2.50 \times 10^{-1}]$ 231/2,048	2.63×10^{-1} $[2.27 \times 10^{-1}, 3.02 \times 10^{-1}]$ 289/2,048	1.24×

B.1 Sampling Diversity

Among 20,000 generated codes, QECFM produces the target encoding rate of 12 in 95.90% of samples, compared with 79.75% for HOG-Diff and 2.24% for random construction. Of the valid QECFM samples, 93.91% are unique up to symmetry and 99.91% are absent from the training corpus. This shows that QECFM does not simply remember the training corpus but is capable of sampling unseen codes.

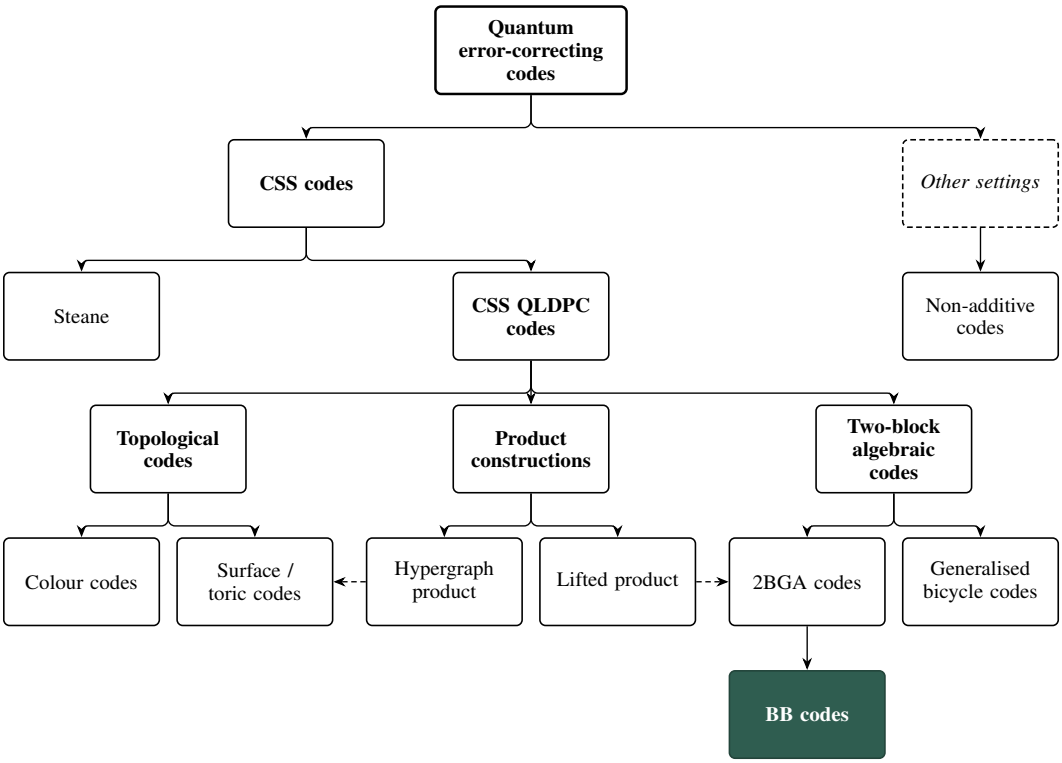

Figure 7: Taxonomy of quantum error-correcting codes. Solid arrows denote containment; dashed arrows mark a family that also arises from that construction.

B.2 Encoding of the selected BB code.

We use the BB construction of Bravyi et al. (2024), viewed as the Abelian case of the two-block group-algebra construction (Lin and Pryadko, 2024). A BB code with parameters (ℓ, g) is represented by a binary vector of length $2(\ell + g - 1)$. Each half contains the coefficients of one polynomial in the ordered basis

$$(1, x, \ldots, x^{\ell-1}, y, \ldots, y^{g-1}).$$

For our selected $[\![144, 12, 8]\!]$ code, $(\ell, g) = (12, 6)$ and the parameter vector is

$$\texttt{000000000001100001001011000001010 0}.$$

Splitting the vector after 17 entries gives

$$a = x^{11} + y, \qquad b = 1 + x^3 + x^5 + x^6 + y + y^3,$$

or equivalently

$$\mathrm{supp}(a) = \{(11, 0), (0, 1)\}, \qquad \mathrm{supp}(b) = \{(0, 0), (3, 0), (5, 0), (6, 0), (0, 1), (0, 3)\}.$$

To fix the matrix representation, we define

$$(S_r)_{i,(i+1) \bmod r} = 1, \qquad x = S_{12} \otimes I_6, \qquad y = I_{12} \otimes S_6,$$

with indices $i = 0, \ldots, r - 1$. Evaluating the two polynomials at these shifts gives $A = a(x, y)$ and $B = b(x, y)$, from which

$$\mathbf{H}_X = [A \mid B], \qquad \mathbf{H}_Z = [B^\top \mid A^\top].$$

C Relation between quantum-code families

Scope of the taxonomy. Figure 7 summarises the part of the quantum-code landscape relevant for our work. CSS codes describe X- and Z-type errors using separate binary parity-check matrices (Calderbank and Shor, 1996; Steane, 1996). The $[\![7, 1, 3]\!]$ Steane code is a canonical example of

this construction. Quantum LDPC codes (Breuckmann and Eberhardt, 2021) are CSS codes whose stabiliser weights and the number of stabiliser checks acting on each qubit remain bounded as the block length grows. Non-additive codes, such as the $\big((5, 6, 2)\big)$ Rains–Hardin–Shor–Sloane code (Rains et al., 1997), fall outside the CSS framework and are shown only to indicate that the CSS codes form one branch of the broader QEC landscape.

Topological and product constructions. Topological codes obtain their checks from geometric or cellular structure. Surface and toric codes associate qubits and stabilisers with cells of a two-dimensional complex, while colour codes use suitably colourable lattices (Dennis et al., 2002; Kitaev, 2003; Bombin and Martin-Delgado, 2007). Product constructions build codes from parity-check matrices. Hypergraph-product codes achieve finite rate and square-root distance scaling (Tillich and Zémor, 2014), and some surface and toric codes can themselves be expressed as products of repetition codes. Lifted-product codes replace the scalar entries of these products by matrices or group-algebra elements, substantially enlarging the construction space (Panteleev and Kalachev, 2022a;b). These overlaps are shown by dashed arrows in Fig. 7

Two-block group-algebra and BB codes. Bicycle codes originated as sparse quantum codes built from circulant matrices (MacKay et al., 2004), with generalised bicycle and hyperbicycle constructions extending this idea to pairs of commuting circulants (Kovalev and Pryadko, 2013). Two-block group-algebra (2BGA) codes replace the cyclic algebra by the group algebra of an arbitrary finite group G and can also be viewed as the smallest lifted-product codes (Lin et al., 2023). BB codes are the Abelian specialisation $G = \mathbb{Z}_\ell \times \mathbb{Z}_g$, for which the two cyclic group coordinates give the commuting bivariate shifts used in their check matrices (Bravyi et al., 2024). The $[\![144, 12, 12]\!]$ Gross code is a prominent member of this family.