MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Q-Learning Base Search Voltage-Labeled Covers for Weight-Six Bivariate-Bicycle Quantum LDPC Codes

Nourozi, Vahid; Mitchell, David; Koike-Akino, Toshiaki

TR2026-132 September 17, 2026

Abstract

Bivariate-bicycle (BB) quantum LDPC codes combine sparse stabilizers with nonzero finite-length rate, but the discrete search over polynomial supports and graph covers grows rapidly. We study a two-stage construction procedure for weight-six BB codes. Tabular Q-learning searches the supports of two threeterm bivariate polynomials. For each selected base, directional covers and monomial shifts are then evaluated. After gauge fixing, a cover of size h has h ^4 shift assignments; hence every reported case h E 2, 3, 4, 6 has at most 1296 < Nex = 6500 assignments and was exhaustively enumerated. The shifts are voltage labels, but the reported reward used no additive voltage term (Bv = 0); voltage order is therefore used only as a structural interpretation of lifted collisions. We recall the standard BB commutation/parity and voltage-lifting facts, specify the screening and exact-distance criteria, and report representative descendants including [[126, 10, 10]] and [[126, 6, 12]]. Code-capacity simulations with QBP and BP-OSD are presented as descriptive illustrations rather than matched-parameter or equal-budget superiority claims.

IEEE International Conference on Quantum Computing and Engineering (QCE) 2026

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Q-Learning Base Search Voltage-Labeled Covers for Weight-Six Bivariate-Bicycle Quantum LDPC Codes

Vahid Nourozi[†,*], David G. M. Mitchell[†], and Toshiaki Koike-Akino[*]

[†]Klipsch School of Electrical and Computer Engineering, New Mexico State University
Las Cruces, NM 88003, USA, Email: {nourozi, dgmm}@nmsu.edu.
[*]Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139, USA
Email: {nourozi, koike}@merl.com.

***Abstract*—Bivariate-bicycle (BB) quantum LDPC codes combine sparse stabilizers with nonzero finite-length rate, but the discrete search over polynomial supports and graph covers grows rapidly. We study a two-stage construction procedure for weight-six BB codes. Tabular Q-learning searches the supports of two three-term bivariate polynomials. For each selected base, directional covers and monomial shifts are then evaluated. After gauge fixing, a cover of size h has h^4 shift assignments; hence every reported case $h \in \{2,3,4,6\}$ has at most $1296 < N_{\mathrm{ex}} = 6500$ assignments and was exhaustively enumerated. The shifts are voltage labels, but the reported reward used no additive voltage term ($\beta_v = 0$); voltage order is therefore used only as a structural interpretation of lifted collisions. We recall the standard BB commutation/parity and voltage-lifting facts, specify the screening and exact-distance criteria, and report representative descendants including $[[126, 10, 10]]$ and $[[126, 6, 12]]$. Code-capacity simulations with QBP and BP-OSD are presented as descriptive illustrations rather than matched-parameter or equal-budget superiority claims.**

***Index Terms*—Quantum LDPC codes, bivariate bicycle codes, graph covers, voltage assignments, reinforcement learning.**

I. Introduction

QUANTUM low-density parity-check (QLDPC) codes are a leading route to reducing the overhead of fault-tolerant quantum computation. Asymptotic breakthroughs show that sparse quantum codes can simultaneously have nonzero rate and large distance, but near-term fault-tolerant systems also require good finite-length parameters, bounded stabilizer weight, and a practical method for discovering candidate codes. Bivariate-bicycle (BB) codes are especially useful in this finite-length setting. They are defined by two sparse polynomials in a two-variable quotient ring and give Calderbank–Shor–Steane (CSS) codes with automatic commutation. The recent fault-tolerant memory construction of Bravyi *et al.* demonstrated the practical relevance of BB codes and motivated systematic searches for strong finite-length instances [1]. Related progress includes asymptotic QLDPC constructions [2], [3], coprime-BB searches [4], generalized-toric realizations [5], and covering-graph constructions for BB cover codes [6].

Machine-learning methods have also been used to discover quantum error-correcting codes. Examples include reinforcement learning (RL) for adapting surface-code layouts [7], low-weight stabilizer-code discovery [8], and optimization of hypergraph-product codes [9]. Recent work has also used RL to improve BP decoding itself by learning sequential decoding trajectories for QLDPC codes [10]. Very recent work has also applied language-model-guided evolutionary search to BB and perturbed-BB code ansatzes [11]. Our approach differs by using a discrete tabular Q-learning policy for scalar weight-six polynomial supports and a separate cover-shift search over selected bases.

This paper separates the two search roles. First, tabular Q-learning modifies the supports of two three-term polynomials and supplies the base candidates in Table II. Second, each selected base is lifted through directional covers and monomial shifts. For the reported values $h \in \{2,3,4,6\}$, all h^4 gauge-fixed shift assignments fit below the exhaustive threshold and are evaluated; Table III therefore reports exhaustively selected cover descendants, not candidates whose selection can be attributed to cover-stage Q-learning. The optional Q-learning branch is retained only for larger shift spaces. Voltage labels provide a standard description of cycle lifting, but no additive voltage reward was active in the reported runs.

II. Bivariate-bicycle notation

For positive integers (ℓ, m), define

$$R_{\ell,m} = \mathbb{F}_2[x,y]/(x^\ell - 1, y^m - 1).$$

A scalar BB code is specified by two polynomials

$$P = \sum_i x^{a_i} y^{b_i}, \qquad Q = \sum_j x^{c_j} y^{e_j} \in R_{\ell,m}.$$

Let M_P and M_Q denote the binary multiplication matrices associated with the polynomials P and Q. They give the CSS pair

$$H_X = [M_P \; M_Q], \qquad H_Z = [M_Q^T \; M_P^T],$$

so that the block length is $n = 2\ell m$. The code dimension is computed from the binary ranks. We write the code parameters as $[[n, k, d]]$ with

$$d = \min(d_X, d_Z),$$

where d_X and d_Z are the two CSS logical distances. A weight-six BB code means that P and Q each contain three monomials, so every check has weight six and the maximum data degree is also bounded by six.

A cover is described by a factorization

$$h = ab,$$

where a scales the x direction and b scales the y direction. The lifted ring is

$$R_{a\ell,bm} = \mathbb{F}_2[x,y]/(x^{a\ell}-1, y^{bm}-1),$$

and the length scales as $n_h = hn$. The cover also has shift variables. A monomial $x^u y^v$ is sent to

$$x^{u+\ell s} y^{v+mt}, \qquad s \in \mathbb{Z}_a, \quad t \in \mathbb{Z}_b.$$

The vectors $\mathbf{s}_x^P, \mathbf{s}_y^P, \mathbf{s}_x^Q, \mathbf{s}_y^Q$ in Table III consistently denote the four directional shift lists for the monomials of P and Q.

III. Reinforcement-learning search method

A. Standard BB identities

The following algebraic facts are recalled to fix notation and are not claimed as new theoretical results; they are standard in the scalar BB and covering-graph literature [1], [6].

Proposition 1 (Standard BB identities)**.** *Let $P, Q \in R_{\ell,m}$ and let M_P, M_Q be their binary multiplication matrices. With*

$$H_X = [M_P \; M_Q], \qquad H_Z = [M_Q^T \; M_P^T],$$

one has $H_X H_Z^T = 0$. Moreover $\operatorname{rank} H_X = \operatorname{rank} H_Z$, so

$$k = 2\ell m - 2\operatorname{rank}(H_X)$$

is even. If P and Q have three distinct monomials each, every stabilizer has weight at most six; the same statement holds after a cover and monomial shift in the same abelian quotient-ring construction.

Proof: Here $H_X H_Z^T = M_P M_Q + M_Q M_P$. The multiplication matrices commute in the abelian quotient ring, and the sum vanishes over $\mathbb{F}_2$. Transposition is the torus involution $x \mapsto x^{-1}, y \mapsto y^{-1}$ up to permutation, while a block swap preserves rank. The CSS dimension formula gives the expression for k. Each monomial contributes one edge per row and column, and a cover or shift only relocates those edges. ∎

Corollary 1 (Parity of the scalar BB dimension)**.** *All scalar weight-six BB codes in this template have even k. A target such as $k = 13$ therefore requires a different ansatz or CSS construction.*

B. Standard voltage-lift interpretation

For two terms of P, define

$$\Delta_P(i,i') = (a_i - a_{i'}, b_i - b_{i'}) \in \mathbb{Z}_\ell \times \mathbb{Z}_m,$$

and define $\Delta_Q(j,j')$ similarly. A base collision is an equality $\Delta_P(i,i') = \Delta_Q(j,j')$. In an (a,b)-cover, let $G_h = \mathbb{Z}_a \times \mathbb{Z}_b$ and let $\lambda_{P,i}, \lambda_{Q,j} \in G_h$ denote the voltage labels.

Proposition 2 (Standard voltage-lift closure)**.** *For a base collision, define*

$$\Omega = (\lambda_{P,i} - \lambda_{P,i'}) - (\lambda_{Q,j} - \lambda_{Q,j'}) \in G_h.$$

The lifted collision closes after one traversal in the starting cover sheet if and only if $\Omega = 0$. If $\Omega \neq 0$, repeated traversal closes after $\operatorname{ord}_{G_h}(\Omega)$ traversals.

Proof: One traversal returns to the same base vertex and changes the cover sheet by Ω. The first closure occurs at the smallest positive r for which $r\Omega = 0$, which is exactly the group order of Ω. ∎

Corollary 2 (Voltage-order diagnostic)**.** *For a fixed base and cover split, $\Omega \neq 0$ certifies only that the corresponding base collision does not close after one traversal; $\operatorname{ord}_{G_h}(\Omega)$ gives the number of traversals for that collision to close. This is a local cycle-lifting diagnostic, not a distance formula or a proof of improved code quality.*

For example, in $G_h = \mathbb{Z}_3$, a voltage $\Omega = 0$ closes immediately, whereas $\Omega = 1$ has order three. In the reported experiments, however, $\beta_v = 0$ and all cover spaces were exhaustively enumerated, so voltage order is not used as evidence that the final rows were selected by voltage guidance.

C. States, actions, rewards, and archives

For a valid candidate C, the evaluator returns

$$\mathsf{Eval}(C) = (n, k, r_X, r_Z, d_{\mathrm{scr}}, d_{\mathrm{ex}}, \tau),$$

where r_X, r_Z are exact binary ranks, d_{scr} is the bounded screening value, d_{ex} is present only after exact certification, and τ records timeout or abort status. The value supplied to the learner is

$$d_{\mathrm{eff}} = \begin{cases} d_{\mathrm{ex}}, & \text{if an exact certificate is available,} \\ d_{\mathrm{scr}}, & \text{otherwise.} \end{cases}$$

The provenance of d_{eff} is stored with every candidate, and only d_{ex} is printed as an exact distance in the final tables.

Duplicate monomial residues are rejected. For distinct supports, let $m(\Delta)$ be the multiplicity of a nonzero ordered difference Δ among the ordered support differences of both P and Q, and define

$$C_{\mathrm{bad}}(P,Q) = \sum_{\Delta \neq 0} \max\{0, m(\Delta) - 1\}.$$

This count includes repeated within-polynomial differences and P–Q collision patterns. Let $B(C) = \mathbf{1}\{k(C) \geq K_0, \; d_{\mathrm{eff}}(C) \geq D_0\}$ be the binary target bonus and let $S(C) = k(C) d_{\mathrm{eff}}(C)^2 / n(C)$. The base state is the tuple of the binned values of $k - K_0$, $d_{\mathrm{eff}} - D_0$, and C_{bad} shown in Table I, together with $B(C)$. A full cover candidate is denoted

$$C_{\mathrm{cov}} = (h, a, b, \mathbf{s}_x^P, \mathbf{s}_y^P, \mathbf{s}_x^Q, \mathbf{s}_y^Q; P, Q),$$

and its state is the clipped tuple $(k - K_0, d_{\mathrm{eff}} - D_0, \min\{50, \lfloor S/2 \rfloor\})$. Thus the encoders are explicit binning maps rather than additional learned functions. Voltage quality is not a state or reward component in the reported runs.

The base action set consists of incrementing or decrementing one exponent coordinate modulo (ℓ, m), replacing one support point by a new residue, exchanging one support point between P and Q, and restarting from a random or archived candidate. The cover action set changes one free shift coordinate modulo (a, b), randomizes one label, changes labels in both polynomials, uses the zero assignment, or restarts from a random or archived assignment. For larger nonenumerated spaces, $Q_{\mathrm{pow}}(i, \lambda)$ stores an auxiliary value for assigning label $\lambda \in G_h$ to free monomial i and biases later proposals; it does not replace the state–action table and was not used to select the exhaustively enumerated rows reported here.

TABLE I
SEARCH SETTINGS AND ATTRIBUTION OF THE REPORTED RESULTS.

Quantity	Setting
Base learning	$(K_0, D_0) = (12, 6)$, $\alpha = 0.10$, $\delta = 0.95$, $\epsilon : 0.35 \to 0.05$, 3000 episodes, 60 steps.
Optional cover learning	$(K_0, D_0) = (8, 8)$, $\alpha = 0.15$, $\delta = 0.95$, $\epsilon = 0.25$, 2000 episodes, 50 steps, $h \in \{2, 3, 4, 6\}$ and all $h = ab$.
State bins	Base: $k - K_0 : \{-10, -5, -2, -1, 0, 2, 5, 10\}$, $d_{\text{eff}} - D_0 : \{-5, -3, -2, -1, 0, 1, 2, 4\}$, $C_{\text{bad}} : \{0, 1, 2, 4, 8, 16, 32\}$. Cover: $k - K_0 \in [-8, 12]$, $d_{\text{eff}} - D_0 \in [-8, 14]$, $\lfloor S/2 \rfloor \le 50$.
Rewards	Equations (1) and (2); no additive voltage reward ($\beta_v = 0$).
Enumeration	$N_{\text{ex}} = 6500$. After gauge fixing, $N_{\text{shift}} = h^4$, giving $16, 81, 256$, and 1296 assignments for $h = 2, 3, 4, 6$. Therefore every reported cover row was selected by exhaustive enumeration, not by Q_{cov} or Q_{pow}.

Algorithm 1 Q-learning search for W6 scalar BB bases

Require: Rings (ℓ, m), exponent set $\mathcal{E}$, target (K_0, D_0), learning parameters $(\alpha, \delta, \epsilon)$.
Ensure: Archive of screened bases and the best base P^*, Q^*.
Initialize $Q_{\text{base}}(z, a) = 0$ and an empty archive.
for each ring (ℓ, m) and each learning episode **do**
  Start from a random or seeded W6 pair $|P| = |Q| = 3$.
  for $t = 1, \ldots, L$ **do**
    Evaluate the current pair with Eval and form z_t.
    Select a_t by the ϵ-greedy rule.
    Apply a_t to move, replace, swap, or restart monomial supports; cancel duplicate residues.
    Evaluate the new pair, compute R_t from (1), and update Q_{base} by (3).
    if the candidate satisfies the screened save rule **then**
      Store $(\ell, m, n, k, d_{\text{eff}}, P, Q)$ and analyzer files.
    end if
  end for
end for
return the archived bases ranked lexicographically by $(d_{\text{eff}}, k, kd_{\text{eff}}^2/n)$.

For completed evaluations, the implemented rewards are

$$R_{\text{base}}(C) = 8k + 180d_{\text{eff}} + 15d_{\text{eff}}^2 + B(C) - 2C_{\text{bad}}, \quad (1)$$

$$R_{\text{cov}}(C) = 2500S(C) + 10k + 100d_{\text{eff}} + B(C). \quad (2)$$

A timed-out or invalid evaluation receives a fixed negative reward and cannot be entered as an exact result; low-distance candidates remain negative training samples rather than being silently discarded. No additive voltage term is present: $\beta_v = 0$. The screened save rule maintains a target archive for $k \ge K_0$ and $d_{\text{eff}} \ge D_0$, and a separate per-ring or per-split trace containing the lexicographically best valid candidate under (d_{eff}, k, S); the latter is why some family rows below the nominal target are retained in the tables.

With learning rate $0 < \alpha \le 1$, discount factor $0 \le \delta < 1$, and exploration probability ϵ, the tabular update is

$$Q(z_t, a_t) \leftarrow (1 - \alpha)Q(z_t, a_t) + \alpha\Big(R_t + \delta \max_{a'} Q(z_{t+1}, a')\Big). \quad (3)$$

Actions are chosen by an ϵ-greedy policy, with probability ϵ a random exploratory action is used, and otherwise an action with largest learned Q-value is selected. Together with the state encoders, rewards in (1)–(2), update rule in (3), and the mutation, restart, label-change, and enumeration operations in Algorithm 1 and the cover-search protocol below, these settings define the complete search procedure used to generate Tables II and III.

After gauge fixing, the reported spaces contain $h^4 = 16, 81, 256$, and 1296 assignments for $h = 2, 3, 4, 6$, respectively, all below $N_{\text{ex}} = 6500$. Thus Algorithm 2 used its exhaustive branch for every row in Table III; $Q_{\text{cov}}^{(h,a,b)}$ and Q_{pow} did not determine the reported descendants.

Algorithm 2 Hybrid search for directional cover shifts

Require: Selected base $P, Q \in R_{\ell,m}$, cover list $\mathcal{H}$, and N_{ex}.
for $h \in \mathcal{H}$ and each factorization $h = ab$ **do**
  Set $(\ell', m') = (a\ell, bm)$ and $G_h = \mathbb{Z}_a \times \mathbb{Z}_b$; fix one label in each polynomial to $(0, 0)$.
  if $h^4 \le N_{\text{ex}}$ **then**
    Evaluate and cache every assignment in G_h^4; update the per-split archive.
  else
    Initialize $Q_{\text{cov}}^{(h,a,b)}$ and Q_{pow}; apply ϵ-greedy label changes and the screened save rule.
  end if
  Certify the shortlisted finalists and output the highest-ranked valid cover.
end for

D. Distance screening and exact certification

For a CSS code, the two logical distances are

$$d_X = \min\{\text{wt}(e) : H_Z e^T = 0,\ e \notin \text{row}(H_X)\},$$
$$d_Z = \min\{\text{wt}(e) : H_X e^T = 0,\ e \notin \text{row}(H_Z)\}.$$

During learning, a bounded search examines weights through five in the base stage and through nine in the cover stage. Its scalar output is d_{scr} and is used only for ranking or rejection. Final certification treats the two sectors separately and performs a complete binary logical-operator feasibility search in increasing weight. A value $d_X = w$ is exact only when the search proves infeasibility for every weight below w and returns an explicit weight-w vector in $\ker H_Z \setminus \text{row}(H_X)$; d_Z is certified analogously. The reported distance is $d_{\text{ex}} = \min(d_X, d_Z)$ only after both sector certificates complete.

A screening or certification call that reaches its wall-clock limit is marked by τ and is never labeled exact. Such a candidate retains only its available screening evidence or a lower/upper bound and is excluded from any table entry described as exact. Exact ranks are obtained by binary Gaussian elimination. This stopping rule distinguishes d_{scr}, the learner's d_{eff}, and the final certified d_{ex}.

IV. RESULTS

The base supports in Table II were obtained with Q-learning, but no matched-evaluation-budget random, greedy, or simulated-annealing baseline was run. The base results therefore establish constructive output, not a causal advantage of the learned Q-table. Every cover space reported in Table III was exhaustively enumerated, and several strongest descendants use the zero assignment. Thus the cover gains can arise from enlarging the torus itself; they are not attributed to cover-stage Q-learning or to an additive voltage reward. A controlled ablation remains necessary to compare base-stage Q-learning with random, greedy, and simulated-annealing search under identical evaluation counts.

The strongest compact exact row in the selected list is $[[126, 10, 10]]$. Other weight-six representatives include $[[126, 6, 12]]$, $[[192, 8, 12]]$, $[[192, 8, 10]]$, $[[180, 16, 8]]$, $[[198, 12, 8]]$, $[[240, 8, 18]]$, and $[[360, 8, 24]]$. These rows are reported as constructions found within the stated search domain; no best-known claim at matched (n, k), check weight, or certification budget is made.

TABLE II

SELECTED W6 BB BASES FROM ALGORITHM 1. THE COLUMNS P_0 AND Q_0 LIST THE THREE MONOMIAL SUPPORTS OF EACH BASE.

(ℓ_0, m_0)	Base code	P_0 terms	Q_0 terms	Selected child
$(7, 3)$	$[[42, 10, 4]]$	$\{(2, 2); (3, 1); (5, 0)\}$	$\{(2, 1); (3, 2); (5, 0)\}$	$[[126, 10, 10]]$
$(6, 5)$	$[[60, 8, 6]]$	$\{(1, 2); (2, 3); (4, 4)\}$	$\{(0, 0); (2, 3); (3, 1)\}$	$[[120, 8, 10]]$
$(7, 3)$	$[[42, 6, 6]]$	$\{(2, 2); (3, 0); (5, 0)\}$	$\{(2, 1); (3, 2); (5, 0)\}$	$[[126, 6, 12]]$
$(9, 3)$	$[[54, 8, 6]]$	$\{(0, 1); (5, 0); (7, 2)\}$	$\{(0, 1); (0, 2); (3, 0)\}$	$[[162, 8, 12]]$
$(8, 3)$	$[[48, 4, 8]]$	$\{(2, 1); (3, 0); (6, 2)\}$	$\{(1, 1); (2, 2); (4, 0)\}$	$[[192, 8, 12]]$
$(6, 5)$	$[[60, 8, 4]]$	$\{(4, 2); (5, 1); (5, 3)\}$	$\{(2, 0); (2, 1); (3, 3)\}$	$[[120, 8, 8]]$
$(31, 1)$	$[[62, 10, 4]]$	$\{(2, 0); (4, 0); (7, 0)\}$	$\{(9, 0); (22, 0); (23, 0)\}$	$[[186, 14, 10]]$
$(7, 4)$	$[[56, 6, 8]]$	$\{(0, 0); (4, 3); (6, 2)\}$	$\{(0, 0); (4, 2); (6, 3)\}$	$[[168, 18, 8]]$
$(6, 5)$	$[[60, 16, 4]]$	$\{(1, 0); (1, 1); (5, 3)\}$	$\{(3, 1); (3, 4); (5, 0)\}$	$[[180, 16, 8]]$
$(7, 3)$	$[[42, 6, 4]]$	$\{(3, 2); (5, 2); (6, 2)\}$	$\{(2, 1); (4, 0); (5, 0)\}$	$[[168, 12, 8]]$
$(7, 3)$	$[[42, 10, 2]]$	$\{(2, 2); (3, 0); (5, 1)\}$	$\{(2, 1); (3, 2); (5, 0)\}$	$[[168, 10, 8]]$
$(8, 3)$	$[[48, 8, 4]]$	$\{(2, 1); (4, 0); (6, 2)\}$	$\{(0, 1); (2, 2); (4, 0)\}$	$[[192, 8, 10]]$
$(6, 5)$	$[[60, 4, 4]]$	$\{(0, 0); (1, 3); (2, 0)\}$	$\{(1, 2); (3, 1); (5, 1)\}$	$[[180, 4, 12]]$
$(9, 3)$	$[[54, 4, 6]]$	$\{(0, 1); (5, 0); (7, 2)\}$	$\{(1, 0); (3, 0); (8, 1)\}$	$[[162, 4, 14]]$
$(3, 11)$	$[[66, 4, 8]]$	$\{(0, 6); (1, 5); (2, 6)\}$	$\{(0, 1); (1, 3); (2, 0)\}$	$[[198, 12, 8]]$

TABLE III

EXHAUSTIVELY SELECTED COVER/SHIFT DESCENDANTS FOR $h \in \{2, 3, 4, 6\}$. A DASH DENOTES THE ZERO ASSIGNMENT.

(ℓ, m)	(h, a, b)	Code	Cover	Nonzero shifts	P terms	Q terms
$(7, 3)$	base	$[[42, 10, 4]]$	base	–	$\{(2, 2); (3, 1); (5, 0)\}$	$\{(2, 1); (3, 2); (5, 0)\}$
$(7, 6)$	$(2, 1, 2)$	$[[84, 10, 6]]$	2	–	$\{(2, 2); (3, 1); (5, 0)\}$	$\{(2, 1); (3, 2); (5, 0)\}$
$(7, 9)$	$(3, 1, 3)$	$[[126, 10, 10]]$	3	–	$\{(2, 2); (3, 1); (5, 0)\}$	$\{(2, 1); (3, 2); (5, 0)\}$
$(14, 6)$	$(4, 2, 2)$	$[[168, 20, 6]]$	4	$s_y^P = (0, 0, 1)$	$\{(2, 2); (3, 1); (5, 3)\}$	$\{(2, 1); (3, 2); (5, 0)\}$
$(6, 5)$	base	$[[60, 8, 6]]$	base	–	$\{(1, 2); (2, 3); (4, 4)\}$	$\{(0, 0); (2, 3); (3, 1)\}$
$(12, 5)$	$(2, 2, 1)$	$[[120, 8, 10]]$	2	–	$\{(1, 2); (2, 3); (4, 4)\}$	$\{(0, 0); (2, 3); (3, 1)\}$
$(12, 10)$	$(4, 2, 2)$	$[[240, 8, 18]]$	4	–	$\{(1, 2); (2, 3); (4, 4)\}$	$\{(0, 0); (2, 3); (3, 1)\}$
$(12, 15)$	$(6, 2, 3)$	$[[360, 8, 24]]$	6	–	$\{(1, 2); (2, 3); (4, 4)\}$	$\{(0, 0); (2, 3); (3, 1)\}$
$(7, 3)$	base	$[[42, 6, 6]]$	base	–	$\{(2, 2); (3, 0); (5, 0)\}$	$\{(2, 1); (3, 2); (5, 0)\}$
$(7, 6)$	$(2, 1, 2)$	$[[84, 12, 6]]$	2	$s_y^Q = (0, 1, 1)$	$\{(2, 2); (3, 0); (5, 0)\}$	$\{(2, 1); (3, 5); (5, 3)\}$
$(7, 9)$	$(3, 1, 3)$	$[[126, 6, 12]]$	3	–	$\{(2, 2); (3, 0); (5, 0)\}$	$\{(2, 1); (3, 2); (5, 0)\}$
$(7, 12)$	$(4, 1, 4)$	$[[168, 24, 6]]$	4	$s_y^P = (0, 2, 2)$; $s_y^Q = (0, 1, 3)$	$\{(2, 2); (3, 6); (5, 6)\}$	$\{(2, 1); (3, 5); (5, 9)\}$
$(6, 5)$	base	$[[60, 8, 4]]$	base	–	$\{(4, 2); (5, 1); (5, 3)\}$	$\{(2, 0); (2, 1); (3, 3)\}$
$(6, 10)$	$(2, 1, 2)$	$[[120, 8, 8]]$	2	–	$\{(4, 2); (5, 1); (5, 3)\}$	$\{(2, 0); (2, 1); (3, 3)\}$
$(6, 30)$	$(6, 1, 6)$	$[[360, 8, 12]]$	6	–	$\{(4, 2); (5, 1); (5, 3)\}$	$\{(2, 0); (2, 1); (3, 3)\}$
$(7, 4)$	base	$[[56, 6, 8]]$	base	–	$\{(0, 0); (4, 3); (6, 2)\}$	$\{(0, 0); (4, 2); (6, 3)\}$
$(7, 8)$	$(2, 1, 2)$	$[[112, 6, 10]]$	2	–	$\{(0, 0); (4, 3); (6, 2)\}$	$\{(0, 0); (4, 2); (6, 3)\}$
$(7, 12)$	$(3, 1, 3)$	$[[168, 18, 8]]$	3	$s_y^P = (0, 0, 1)$; $s_y^Q = (0, 1, 0)$	$\{(0, 0); (4, 3); (6, 6)\}$	$\{(0, 0); (4, 6); (6, 3)\}$
$(31, 1)$	base	$[[62, 10, 4]]$	base	–	$\{(2, 0); (4, 0); (7, 0)\}$	$\{(9, 0); (22, 0); (23, 0)\}$
$(31, 2)$	$(2, 1, 2)$	$[[124, 10, 8]]$	2	$s_y^P = (0, 0, 1)$	$\{(2, 0); (4, 0); (7, 1)\}$	$\{(9, 0); (22, 0); (23, 0)\}$
$(31, 3)$	$(3, 1, 3)$	$[[186, 14, 10]]$	3	$s_y^P = (0, 1, 2)$; $s_y^Q = (0, 1, 2)$	$\{(2, 0); (4, 1); (7, 2)\}$	$\{(9, 0); (22, 1); (23, 2)\}$
$(31, 6)$	$(6, 1, 6)$	$[[372, 14, 16]]$	6	$s_y^P = (0, 1, 2)$; $s_y^Q = (0, 1, 2)$	$\{(2, 0); (4, 1); (7, 2)\}$	$\{(9, 0); (22, 1); (23, 2)\}$

V. CONCLUSION

We presented a two-stage construction procedure in which Q-learning searches weight-six BB base supports and the reported directional cover spaces are exhaustively enumerated. Because all listed h satisfy $h^4 < N_{\text{ex}}$, the cover rows cannot be attributed to cover-stage Q-learning, and the exact-distance acceptance rule now separates screening values, timeouts, bounds, and completed certificates. Because $\beta_v = 0$, voltage order is used only to interpret lifted collisions; matched-budget random, greedy, and simulated-annealing baselines remain necessary to quantify any Q-learning advantage. The tables are constructive examples, while the simulations are descriptive code-capacity results rather than evidence of best-known parameters or equal-budget superiority; trial counts and confidence intervals remain necessary.

ACKNOWLEDGEMENT

D.G.M. Mitchell is partly supported by the National Science Foundation under Grant No. CCF-2145917.

REFERENCES

[1] S. Bravyi *et al.*, "High-threshold and low-overhead fault-tolerant quantum memory," *Nature*, vol. 627, pp. 778–782, 2024.

[2] P. Panteleev and G. Kalachev, "Quantum LDPC codes with almost linear minimum distance," *IEEE Trans. Inf. Theory*, vol. 68, no. 1, pp. 213–229, 2022.

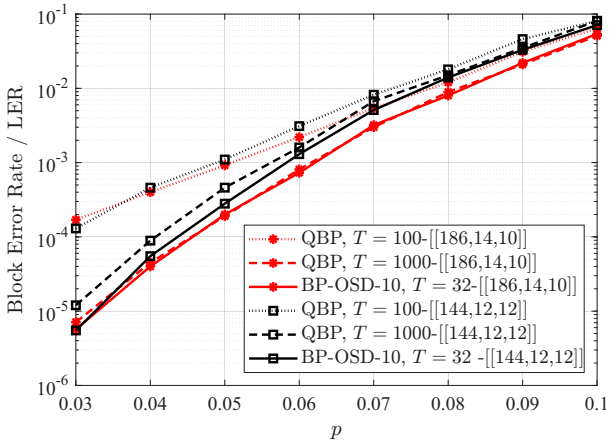

Fig. 1. Logical failure rate under depolarizing noise for the constructed $[[186, 14, 10]]$ code and the reference $[[144, 12, 12]]$ BB code. QBP uses $T \in \{100, 1000\}$; BP-OSD uses $T = 32$ and OSD order 10.

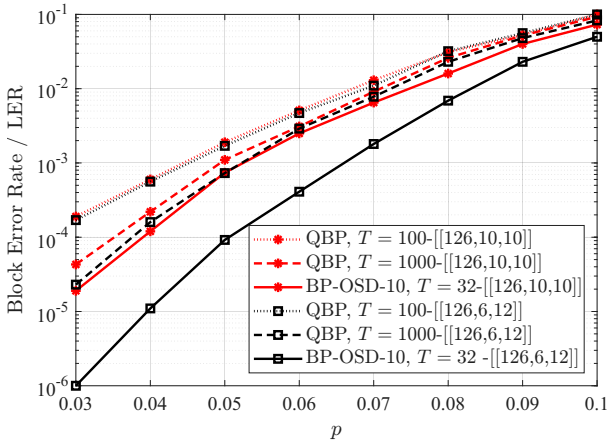

Fig. 2. Logical failure rate under depolarizing noise for the constructed $[[126, 10, 10]]$ and $[[126, 6, 12]]$ codes. QBP uses $T \in \{100, 1000\}$; BP-OSD uses $T = 32$ and OSD order 10.

[3] A. Leverrier and G. Zémor, "Quantum Tanner codes," in *Proc. IEEE FOCS*, 2022, pp. 872–883.

[4] M. Wang and F. Mueller, "Coprime bivariate bicycle codes and their layouts on cold atoms," *Quantum*, vol. 10, Art. no. 2009, 2026.

[5] Z. Liang, K. Liu, H. Song, and Y.-A. Chen, "Generalized toric codes on twisted tori for quantum error correction," *PRX Quantum*, vol. 6, Art. no. 020357, 2025.

[6] B. C. B. Symons, A. Rajput, and D. E. Browne, "Sequences of bivariate bicycle codes from covering graphs," arXiv:2511.13560, 2025.

[7] H. P. Nautrup, N. Delfosse, V. Dunjko, H. J. Briegel, and N. Friis, "Optimizing quantum error correction codes with reinforcement learning," *Quantum*, vol. 3, Art. no. 215, 2019.

[8] A. Y. He and Z.-W. Liu, "Discovering highly efficient low-weight quantum error-correcting codes with reinforcement learning," arXiv:2502.14372, 2025.

[9] B. C. A. Freire, N. Delfosse, and A. Leverrier, "Optimizing hypergraph product codes with random walks, simulated annealing and reinforcement learning," arXiv:2501.09622, 2025.

[10] M. Moradi, V. Nourozi, S. Habib, and D. G. M. Mitchell, "Learning to decode quantum LDPC codes via belief propagation," arXiv:2603.10192, 2026.

[11] J. Cruz-Benito, A. W. Cross, D. Kremer, and I. Faro, "Evolutionary discovery of bivariate bicycle codes with LLM-guided search," arXiv:2606.02418, 2026.