Quantum Error Correction Meets AI for Science: Lessons from LLM-Guided Code Discovery

AI4QC Workshop — IEEE Quantum Week 2026 · Toronto

Juan Cruz-Benito
IBM Research

THE DISCOVERY — AND THE STRESS TEST

200,000 candidates. 465 codes. A new map of QEC design.

~2×10⁵
candidates screened

465
BLISS-distinct codes, n ≤ 360

97 + 368
CSS + non-CSS codes

12×
heuristic blind spot corrected

This is both a discovery result and a methods result: **a richer rate–distance and error-suppression map**, plus a verification stack strong enough to expose and correct its own blind spots.

AI for science, one step beyond

Where LLM-guided program evolution sits in the current wave

AI FOR SCIENCE IS HAVING A MOMENT

From learned predictors to verifier-guided program search

- **AlphaFold** (Jumper et al., *Nature* 596, 583–589, 2021) — structure prediction over an astronomically large conformational space.
- **GNoME** (Merchant et al., *Nature* 624, 80–85, 2023) — materials discovery over crystal-structure space via graph networks + active learning.
- **FunSearch** (Romera-Paredes et al., *Nature* 625, 468–475, 2024) — LLM proposes *programs*, not answers, scored by a verifier, kept in an evolutionary population.
- **AlphaEvolve** (Novikov et al., arXiv:2506.13131, 2025) — the same idea, scaled: an LLM-driven coding agent evolving algorithms across math and systems problems.

> The shared pattern is search over immense structured spaces with automated evaluation. **FunSearch and AlphaEvolve** add the move used here: an LLM proposes candidate programs, and a verifier drives selection.

- This talk is that pattern, applied to a domain with an unusually strict verifier: **quantum error correction**.
- Built on **OpenEvolve**, an open-source evolution engine implementing the FunSearch/AlphaEvolve pattern.

When each approach helps

RL / DL for search

Learns a policy or value function and amortizes search into weights; typically needs many task-specific interactions and a sufficiently informative reward; strongest when evaluations are cheap.

+

LLM-guided evolution

Uses an LLM's code and mathematical priors to mutate **programs**, not points, without task-specific training; useful when evaluations are sparse or costly, but still vulnerable to a gameable proxy.

- **Thesis:** LLM-guided program evolution is complementary to RL/DL, not a replacement — and qLDPC code discovery is a demanding place to stress-test it.
- **Shared risk:** either approach can exploit a weak proxy. Here, an exact solver and an analytic theorem let us prove that it happened.

Roadmap: object → method → results → transferable lessons.

The object and the problem

arXiv:2606.02418

THE OBJECT

Bivariate bicycle (BB) and perturbed BB (PBB) codes

- Read `[[n, k, d]]` as **physical qubits, logical qubits, distance**. For example, `[[144,12,12]]` uses 144 physical qubits to encode 12 logical qubits with distance 12.
- A **BB code** is CSS: two polynomials `A, B ∈ F₂[x, y]/(x^ℓ − 1, y^m − 1)`, giving `n = 2ℓm`; `k` is exact via GF(2) rank, while `d` is expensive to determine.
- A **PBB code** adds perturbation polynomials `(C, D)`, mixing X/Z stabilizers; `C = D = 0` recovers the CSS case.
- **Weight** means the maximum Pauli support of a stabilizer generator. For CSS BB codes it is usually `|A| + |B|`; overlapping X/Z support in PBB counts only once.

Figure of merit used throughout: **FOM = k·d²/n**. It is a compact rate–distance search metric, not a complete performance measure: logical error rate, stabilizer weight, and locality also matter.

HOW EXPENSIVE IS BRUTE FORCE?

Back-of-the-envelope: up to 1,700 core-years

~9T
naïve choices, all shapes at n = 360

~1.2B
left after symmetries

~30M
estimated connected, k > 1

≤1,700
core-years at 30 min/code

- At `n = 360`, nine lattice shapes yield about **9 trillion** raw symmetric weight-6 choices.
- Symmetry reduction gives a working estimate of **~1.2 billion** representatives across `100 ≤ n ≤ 360`; Monte Carlo sampling suggests roughly **30 million** have `k > 1` and connected Tanner graphs.
- At **up to 30 minutes per exact-distance attempt**, that is **~1,700 core-years**—before asymmetric, other-weight, perturbed, and product constructions.

WHY QUANTUM ERROR CORRECTION

An unusually strict testbed for AI-driven discovery

NP-hard
exact code distance

No
gradient signal

MILP
exact when optimality is certified

- Fault-tolerant quantum computing is constrained by **qubit overhead**—good qLDPC codes are scarce and hard-won.
- The search space is discrete and gradient-free; exact distance is **NP-hard**, while fast heuristics can return loose upper bounds.
- Unlike many AI-for-science benchmarks, there is a route to **certifiable ground truth**: MILP when optimality is proved, exact structural checks, and analytic theorems.

Method

Evolving programs, not codes — and why that choice matters

KEY IDEA 1

Evolve the generator, not an individual code

- The unit of evolution is a Python function `G(ℓ, m) → candidates` that *generates* polynomial pairs/tuples for a given lattice—not a single code.
- One mutation—for example, "try exponent `ℓ/3`"—can generalize across every compatible lattice instead of producing a one-off tuple.
- This is the core AlphaEvolve/FunSearch move: let the LLM edit *source code*, while evolution maintains population state outside the model's context.

> Practical upshot: a single evolved program can generalize across lattices—the search discovers a *recipe*, not an isolated answer.

- A **MAP-Elites** quality-diversity archive keeps the best-known program per structural niche, preserving structural variety.

KEY IDEA 2

The LLM as mutation operator

Each iteration, the LLM sees:

1. The current generator program
2. Domain knowledge about BB-code algebra
3. Feedback from the last evaluation

It proposes a **targeted code diff**—a modification to the generator, not a full rewrite or a final code.

> One evolved lineage rediscovered univariate, hypergraph-product-style structure without being told that family existed.

Novel Ansatz Discovery — Domain Knowledge for LLM

What You're Evolving

You are evolving a Python function `generate_candidates(ell, m)` that returns candidate bivariate bicycle (BB) quantum error-correcting code polynomial pairs. Each candidate is a pair of polynomials (A, B) represented as lists of `(x_exponent, y_exponent)` tuples. Polynomials may have 2-6 terms.

Your goal is to discover **new structural families (ansatze)** — algebraic templates relating A and B that produce good codes across multiple lattice sizes. Not individual codes, but repeating patterns.

Mathematical Background

A BB code is defined over the ring F_2[x,y]/(x^ell - 1, y^m - 1) by two polynomials A(x,y) and B(x,y). Each monomial x^a * y^b corresponds to a shift operator P_m^b (tensor) P_ell^a acting on the ell x m torus.

Code parameters: [[n, k, d]] where n = 2*ell*m, k = logical qubits, d = code distance.

Figure of merit: FOM = k * d^2 / n. Higher is better. Target: FOM > 12.0.

The Scientific Question

Are there structural families of BB codes beyond the 3 known ones?

All known good BB codes fall into exactly 3 families:

1. **x/y-swap trinomials**: A = x^a + y^b + y^c, B = y^d + x^e + x^f. One polynomial has one pure-x term and two pure-y terms; the other is the reverse. Best: [[360,12,≤24]] FOM=19.2.

Real excerpt from the domain-knowledge prompt supplied before mutation.

THE EVALUATION CASCADE

Cheap first, expensive last

1
Propose
LLM edits the generator

2
Generate
program emits codes across lattices

3
Evaluate
rank → BP-OSD → MILP

4
Select
MAP-Elites retains diverse winners

Evolutionary loop: archive results become context for the next targeted diff.

Independent exit path: post-campaign verification produces the catalog and never feeds claims back unchecked.

The feedback loop makes this evolution rather than one-shot generation; verification remains a separate gate.

STAGES AND COSTS

Screen ~2×10⁵ candidates to catalog 465 classes

~2 s	~30–60 s	~10 min	465
Stage 1 · GF(2) k-screen	Stage 2 · BP-OSD heuristic	Stage 3 · typical MILP attempt	catalogued BLISS classes

- Stage 1 is a fast rank computation over GF(2) — no distance estimate yet, just "is k nonzero and worth evaluating further."
- Stage 2 provides stochastic upper bounds from our original BP-OSD protocol—cheap enough to run at scale, but not sufficient for an exact claim.
- Stage 3 ran in the loop only for Campaigns 4–5; earlier campaigns used Stage 2 in the loop, then reverified every survivor with MILP. The catalog labels exact optima separately from valid upper bounds.

THE VERIFICATION STACK

Rigor is not optional when a proxy can fail

- **BP-OSD → MILP**: every exact distance is backed by an optimality certificate; unresolved results are written explicitly as upper bounds, `d ≤ …`.
- **BLISS canonical labeling**: deduplicates codes that are permutation-equivalent, via a colored Tanner graph (with a tying edge for non-CSS X/Z pairing — a real implementation subtlety).
- **Achievable-syndrome sampling** for non-CSS distance estimation—naïve per-channel random-coset sampling can select unattainable syndromes for PBB codes; this is a correctness requirement, not an optimization.
- **Decomposability and connectivity checks** catch direct sums and disconnected codes; **LC-to-CSS screening** tests equivalence within explicitly stated local-Clifford families.

Results

CSS and non-CSS

THE CATALOG

465 BLISS-distinct codes at n ≤ 360

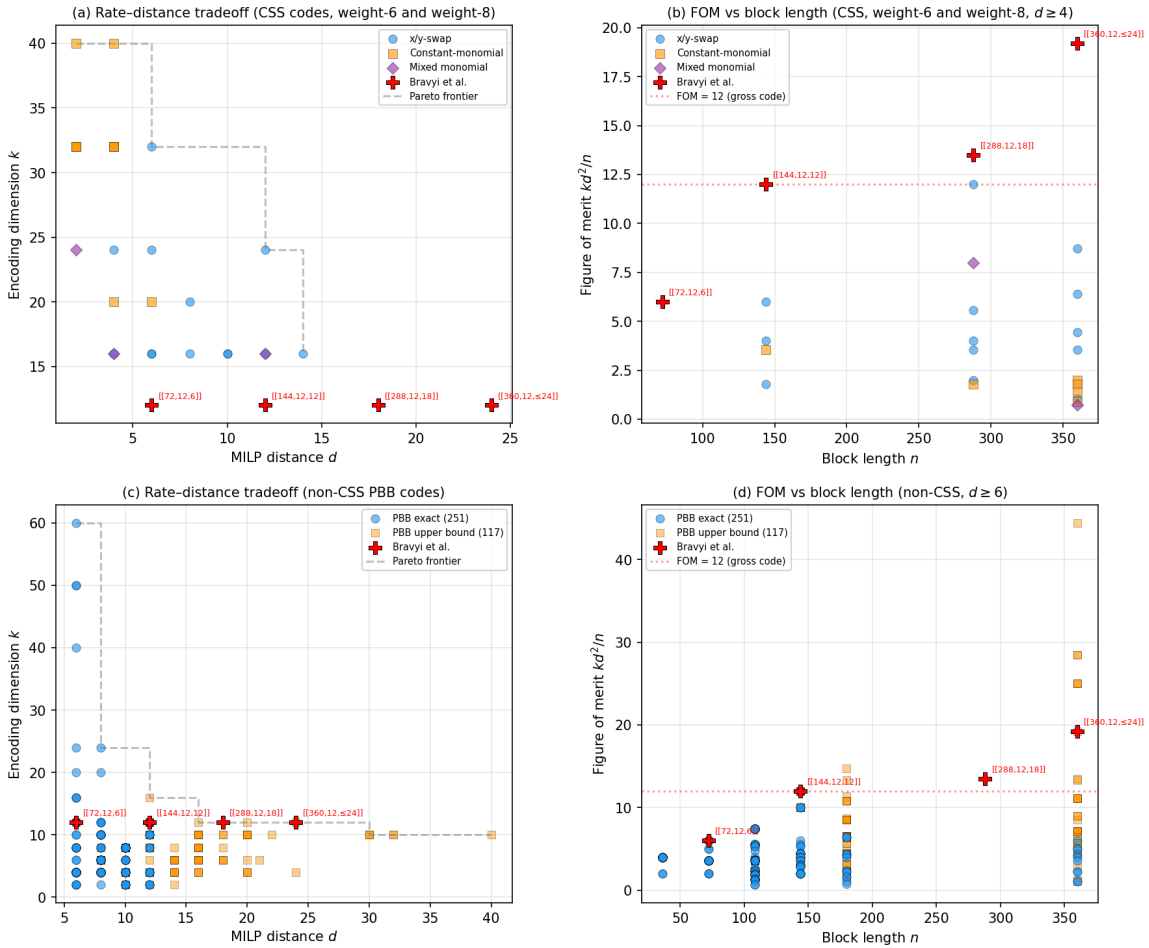
465
distinct under colored-Tanner permutation equivalence

97
CSS BB codes

368
non-CSS PBB

The catalog expands the rate–distance map across CSS and non-CSS constructions. The talk evaluates that map through distance, stabilizer weight, and selected logical-error simulations—not one leaderboard number. Broader equivalence notions may merge additional entries.

Rate–distance and FOM across the catalog



Rate–distance tradeoff and FOM versus block length: CSS above, non-CSS PBB below. “+” markers show the four Bravyi et al. baselines; upper-bound rows are plotted at incumbent ceilings, not achieved distances or FOMs.

CAMPAIGNS 1–3 · WEIGHT-6 TRINOMIALS

Within the original trinomial search, distance ≥ 6 lives in the x/y-swap family CSS

- Among the weight-6 trinomials searched in Campaigns 1–3, only the **x/y-swap family** (`A = x^a + y^b + y^c` , `B = y^d + x^e + x^f`) reaches `d ≥ 6` .
- Standout: `[[288,16,12]]` , FOM = 8.0, exact, with every lattice shift ≤ 3—short-range under the assumed periodic-grid layout.
- Empirically, indecomposable weight-6 `d = 12` codes in our catalog have `k ≤ 16` ; entries with `k > 24` have `d ≤ 4` .

CAMPAIGN 4 · MIXED-MONOMIAL AND HIGHER-WEIGHT CODES

Enlarging the structural vocabulary CSS

- Campaign 4 admitted mixed monomials `x^a y^b` and stabilizer weights 6–12, reaching higher `k` than the Campaigns 1–3 families at the matched block lengths shown here.
- Standout: **`[[288,50,8]]`**, FOM = 11.1, exact, "cross-factored" structure—the highest exact FOM among the new indecomposable codes at `n = 288`.
- At `n = 144`, the highest-`k` weight-8 code found is `[[144,54,≤4]]`: **4.5× the rate** of Bravyi's `[[144,12,12]]`, but with only the upper bound `d ≤ 4`. The maximum-`k` and maximum-FOM codes differ.

At matched block length, relative to Bravyi et al.:

n = 144: max-*k* `[[144,54,≤4]]` (weight 8); highest new exact FOM 8.0 comes from `[[144,8,12]]` (weight 6)

n = 288: `[[288,50,8]]` (weight 8) gives 4.2× more logical qubits and exact FOM 11.1

n = 360: max-*k* `[[360,40,2]]` is weight 6; the weight-8 `[[360,20,≤14]]` has FOM ≤10.9

NON-CSS PBB — AND TWO HONEST TIES

368 non-CSS codes, compared with prior art

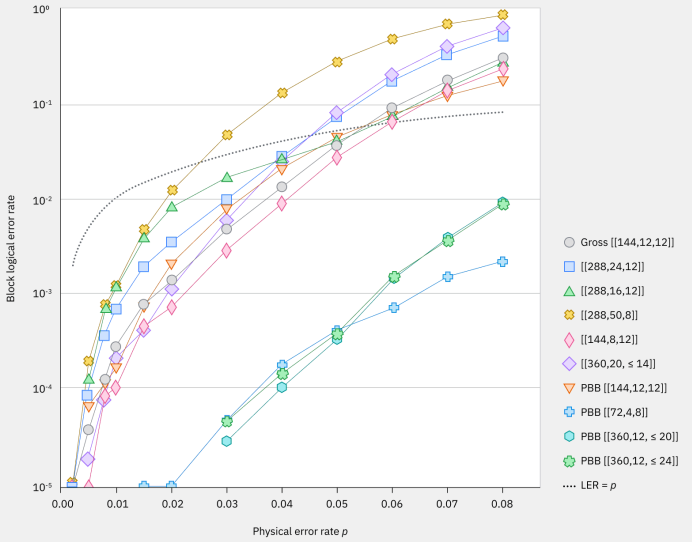
- Non-CSS "gross code" analogue: `[[144,12,12]]` PBB, FOM = 12.0, exact.
- Largest published PBB FOM upper bound in arXiv v1: `[[360,12,≤24]]`, FOM ≤ 19.2, weight 10—and it **ties**, rather than beats, Bravyi et al.'s identically parametrized weight-6 **CSS** baseline. Structurally different codes, same `(n, k, d_upper)` and same FOM bound: a tie of upper bounds, not a record.
- A second, quieter tie sits at `n = 72`, where our Pareto frontier touches Bravyi's `[[72,12,6]]` marker exactly.
- Eleven of 368 are CSS-equivalent under the tested local-Clifford patterns; no equivalence was found for the remaining 357, subject to the coverage gap described in backup.

Why non-CSS? Dropping the CSS restriction enlarges the design space, but commutativity is no longer automatic, distance becomes symplectic, and standard sampling can produce undecodable syndromes. Achievable-syndrome sampling, symplectic MILP, and tying-edge graph-isomorphism checks make that larger space verifiable.

ERROR SUPPRESSION ACROSS THE CATALOG

Error suppression for ten selected codes

- These are **code-capacity simulations under independent X-only bit-flip noise**, not circuit-level hardware predictions.
- Several indecomposable $d \geq 12$ codes matched gross-code pseudo-thresholds while encoding up to **1.7× more logical qubits**.
- Equal FOM does not imply equal logical error rate; block LER also depends on k, decoder, and code structure.
- Higher stabilizer weight requires more two-qubit interactions and typically greater syndrome-extraction depth.



Block logical error rate versus physical error rate for ten selected codes under the decoder reported in the paper.

Lessons

What the search revealed—and what transfers

LESSON 1

Trust but verify—our original BP-OSD configuration overestimates by up to 12×

147/154
bounds tightened (95.5%)

~8.4
mean distance reduction (points)

up to 12×
loosest original BP-OSD bound

33/149
PBB entries corrected (22%)

- Across 154 Campaigns 1–3 polynomial representations, our 150k-trial, multi-decoder BP-OSD protocol tightened 147 earlier bounds, by ~8.4 points on average.
- In a separate PBB cohort, deep MILP revised 33 of 149 catalog entries downward; the largest correction was `d: 24 → 16` at `n = 360`.
- Webster, Jacob & Higgott independently corroborate BP-OSD's sensitivity to configuration and sampling (arXiv:2603.22532).

LESSON 2

Reward hacking, quantum edition

- **Theorem** ("distance trap"): for the searched constructions with polynomial weight at least 2, every CSS BB code with `A = B` and `k > 0` has distance exactly 2; this is analytically proved.
- For PBB codes, `A = B` yields weight-2 normalizer candidates; every such catalog entry was MILP-confirmed at `d = 2`.
- The failure mode: `A = B` maximizes several cheap structural proxies while forcing distance 2—a textbook reward-hacking pattern expressed in ring theory rather than an RL reward function.
- In our original 150k-trial BP-OSD run, **all 30 batches missed the weight-2 operator**. At 1.5M trials, 29 of 30 still missed it and one found it; MILP confirms it in under a second.

> A powerful search paired with a heuristic verifier *can expose* the heuristic's blind spot. The safeguard here was an analytic theorem plus an exact solver that does not rely on sampling.

LESSON 3

The shortcut the pipeline detected

- `[[288,24,12]]` looked as though it broke the weight-6 rate ceiling: `k = 24` at `d = 12`, above the catalog's indecomposable `k ≤ 16` pattern.
- A connectivity check reveals two components; BLISS identifies each with the `[[144,12,12]]` gross code. The result is **`[[144,12,12]] ⊕ [[144,12,12]]`**, with MILP-proven `d = 12`.
- The same direct sum reappears independently at `(ℓ, m) = (24, 6)`, again confirmed by connectivity and BLISS.

> Structural guardrails belong *inside* the loop, not only after it. Without decomposability and connectivity checks, the search can keep rewarding split or disconnected codes.

LESSON 4 — THE ONE WHERE THE BASELINE WINS

Does the LLM earn its keep?

- On the deterministic proxy `Σk`, GA-G reached **`1,037 ± 22`** versus the LLM-guided run's **704**, using ~14,000 versus ~213,000 evaluations. Its selected codes all had `d ≤ 2`.
- The LLM-guided run produced indecomposable codes up to FOM **8.0**; FOM **12.0** includes the decomposable direct sum from Lesson 3.
- Random search reached **`Σk = 299 ± 26`** at 80,000 evaluations.

> The GA won on the chosen proxy; the LLM-guided run produced useful-distance codes. This exposes a mismatch between the proxy and the scientific objective—it does not establish a universal winner.

Caveat: budgets were unequal; GA-G received no distance feedback; the LLM-guided fitness used imperfect BP-OSD estimates.

LESSON 5 — THE HONEST BOUNDARY

The catalog expanded; the certified fixed-n FOM frontier held

- The catalog adds many rate–distance points, but **no discovered indecomposable code establishes a new certified fixed-`n` FOM record**.
- The highlighted comparisons are ties: an **exact tie** at `[[72,12,6]]` and a **tie of published upper bounds** at `[[360,12,≤24]]`; neither is a record.
- The largest partial valid FOM upper bound in the catalog is **≤44.4** (`[[360,10,≤40]]`), not an achieved FOM.
- One limitation: Campaign 5 never ran on the two lattices where the highest-FOM CSS codes live—`(ℓ, m) = (12, 12)` and `(15, 12)`—so whether PBB can escape the envelope there remains open.

> This boundary makes the broader map credible: strong codes across underexplored regions, two honest ties, and a clear target for the next search.

OUTLOOK

What transfers beyond quantum codes

- The reusable recipe: **evolve programs → cheap-to-expensive verification cascade → explicit reward-hacking guards → independent cross-checks.** None of that is quantum-specific.
- **RL/LLM hybrids** are a natural next step — this talk deliberately framed LLM-guided evolution as a lever alongside RL/DL, not a replacement; combining a trained value function with LLM-proposed structural mutations is unexplored territory here.
- **Immediate PBB experiment:** run the two omitted high-FOM CSS lattices, `(ℓ, m) = (12, 12)` and `(15, 12)`, with exact verification in the loop.
- **Richer objectives:** optimize logical-error behavior, stabilizer weight, and locality alongside FOM, rather than treating one scalar as the whole target.

> If you're searching a large, discrete, gradient-free space with an expensive-but-checkable verifier — this pattern is worth trying, whether or not the domain is quantum at all.

TAKEAWAYS

Three takeaways

1. **Evolve recipes, not isolated answers.** Program-level mutations can reveal reusable structure across an entire design space.
2. **Put exact and structural checks in the loop.** A strong search will find weak proxies, decomposable shortcuts, and hidden equivalences.
3. **Use a multidimensional scorecard.** Report the expanded map, logical-error behavior, hardware costs, and the certified frontier that did not move.

Thank you

Paper:
Cruz-Benito, J., Cross, A. W., Kremer, D., & Faro, I. (2026). Evolutionary Discovery of Bivariate Bicycle Codes with LLM-Guided Search. *arXiv preprint arXiv:2606.02418.*

Repository:
`https://github.com/qiskit-community/qcode-discovery`

> **Let models propose; let exact tools decide.**

https://arxiv.org/abs/2606.02418

Backup

Primer, glossary, campaign details, proofs, and more

BACKUP · ELEMENTARY QEC PRIMER

Why you can't just checksum a qubit

- A classical bit is 0 or 1. Worried about noise? Copy it and vote — triple modular redundancy.
- A qubit **can't be copied** (no-cloning theorem), and measuring it directly to "check its value" **destroys the superposition** you're protecting.
- The fix: encode one logical qubit into many physical qubits, and measure only *joint parity checks* (stabilizers) that reveal error syndromes without revealing — or collapsing — the encoded state.

> A quantum LDPC code's job: pack the most logical qubits (`k`), at the largest protective distance (`d`), into the fewest physical qubits (`n`) — with *sparse*, low-weight stabilizers, since every stabilizer measurement is itself a noisy physical operation.

BACKUP · GLOSSARY

Extended glossary

- **CSS code** — stabilizers split into X-type and Z-type; for BB codes, the construction enforces commutativity automatically.
- **Non-CSS / PBB** — mixed X/Z stabilizers via perturbation polynomials (C, D); commutativity must be checked explicitly.
- **FOM** — $k \cdot d^2/n$, the objective the search maximizes.
- **BP-OSD** — belief propagation with ordered-statistics decoding; a fast, configuration-sensitive distance heuristic.
- **MILP** — mixed-integer linear programming; it proves an exact minimum when optimality is certified, while a timeout may leave only a valid upper bound.
- **BLISS** — a graph-isomorphism solver, used here for canonical labeling / dedup via colored Tanner graphs.
- **LC-equivalence** — local-Clifford equivalence; a non-CSS code that's LC-equivalent to some CSS code is not "genuinely" non-CSS.
- **MAP-Elites** — a quality-diversity evolutionary algorithm keeping the best solution per structural niche.
- **Weight (of a stabilizer)** — number of qubits on which a single check acts nontrivially.

BACKUP · AI FOR MATHEMATICAL SCIENCE

AI is also proving, refuting, and formalizing

- **Constructing a counterexample.** Navier–Stokes (OpenAI, Sep 2026): finite-time blow-up for 3D incompressible flow **when smooth external forcing is allowed**. Under review
- **Proving.** Riemann zeta (Anthropic, Aug 2026): a lower bound of **67.2%**, up from 41.6%, for zeros on the critical line; formalized in Lean.
- **Formalizing.** Fermat's Last Theorem (Anthropic, Sep 2026): an end-to-end computer-checked proof in Lean.

Verifier-guided search *scores* candidates with an approximate, gameable proxy. Formal proof checks claims against a specification; the remaining risk lies in the specification itself.

All three are **lab-published results with Lean artifacts, not peer-reviewed papers**. The Navier–Stokes result concerns smooth external forcing; Clay still lists the problem as active.

BACKUP · RL/DL COMPARISON

Why LLM-guided evolution fit this search

- **What RL/DL would need here:** many evaluations of an informative reward and a policy that transfers across lattice sizes. Here, evaluations are expensive and the signal is sparse.
- **What LLM priors provide:** code and mathematical structure from pretraining, auditable diffs, and no task-specific training loop.
- The search rediscovered hypergraph-product-style structure without being told that family existed.

Caveat: a GA later beat the LLM-guided run on proxy `Σk`, but selected only `d ≤ 2` codes. Lesson 4 gives the budgets and limitations.

BACKUP · ANALYTIC CONTEXT

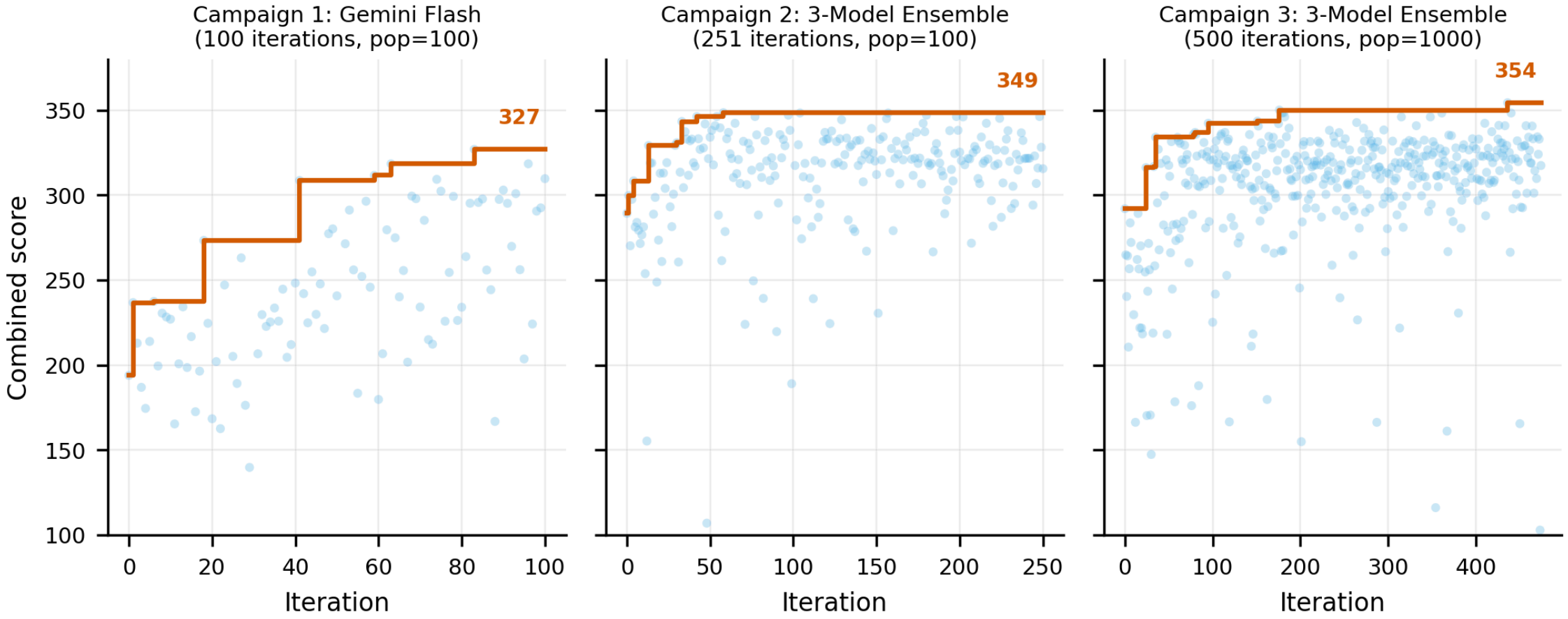
What the analytic bounds do—and do not—say

- The raw exponent-tuple space per lattice (ℓ, m) scales like $O(\ell^6 m^6)$: discrete, combinatorial, and gradient-free.
- Exact distance is NP-hard, and a fast heuristic can be systematically loose.
- Before this work, published indecomposable weight-6 BB codes with $d = 12$ had $k \le 16$; this is the ceiling the search tried to move.

- **BPT bound:** `k·d² = O(n)` applies to geometrically 2D-local stabilizer codes. It motivates FOM here, but does **not** generally bound arbitrary qLDPC or BB layouts. - **BB asymptotics:** Postema & Kokkelmans show `d = O(√n)` for the BB family.

BACKUP · SEARCH TRAJECTORY

The search optimized an in-loop proxy



Combined in-loop fitness by iteration across three campaigns—not post-verification FOM. Campaign 3 reached the highest proxy score before plateauing, motivating the mixed-monomial and non-CSS campaigns.

BACKUP · STATE OF THE ART

Selected exact results and published upper bounds

Source	Code	Type	FOM	Weight	Distance status	Note
Bravyi et al.	[[360,12,≤24]]	CSS	≤19.2	6	Incumbent upper bound	Largest published FOM upper bound
Symons et al.	[[144,14,14]]	CSS	19.1	8	Exact	33% more CNOTs than gross code
Bravyi et al.	[[288,12,18]]	CSS	13.5	6	Exact	
Bravyi et al.	[[144,12,12]]	CSS	12.0	6	Exact	Gross code
This work	[[360,12,≤24]]	**non-CSS**	≤19.2	10	Incumbent upper bound	Published upper-bound tie
This work	[[288,50,8]]	CSS	11.1	8	Exact	Cross-factored
This work	[[144,12,12]]	**non-CSS**	12.0	8	Exact	Non-CSS gross-code analogue
This work	[[288,16,12]]	CSS	8.0	6	Exact	All shifts ≤ 3

A concurrent Bayesian-optimization effort (Chengyu et al., arXiv:2601.18562) reports a `[[144,36,?]]` candidate whose distance has not yet been independently verified.

BACKUP · WEIGHT 5 — A SEPARATE STUDY

Exploring weight 5 to be submitted

- A dedicated follow-up campaign searched **weight-5 codes**; it is separate from arXiv:2606.02418.
- The campaign retained **1,142 proposals** in its weight-5 catalog: 912 CSS + 230 PBB.
- Strongest code found by the campaign: **`[[180,4,14]]`**, FOM = 4.356, at `(ℓ, m) = (10, 9)`.
- Other certified codes: `[[140,6,10]]`, `[[96,4,10]]`, and `[[216,4,10]]`, the strongest exact connected non-CSS code in that study.

The campaign rediscovered **`[[132,4,12]]`** (FOM = 4.364), first reported in the Lin–Pryadko archive. The `[[180,4,14]]` point is 0.008 lower.

These 1,142 proposals are **not included** in the 465-code main catalog.

BACKUP · ERROR-RATE KEY

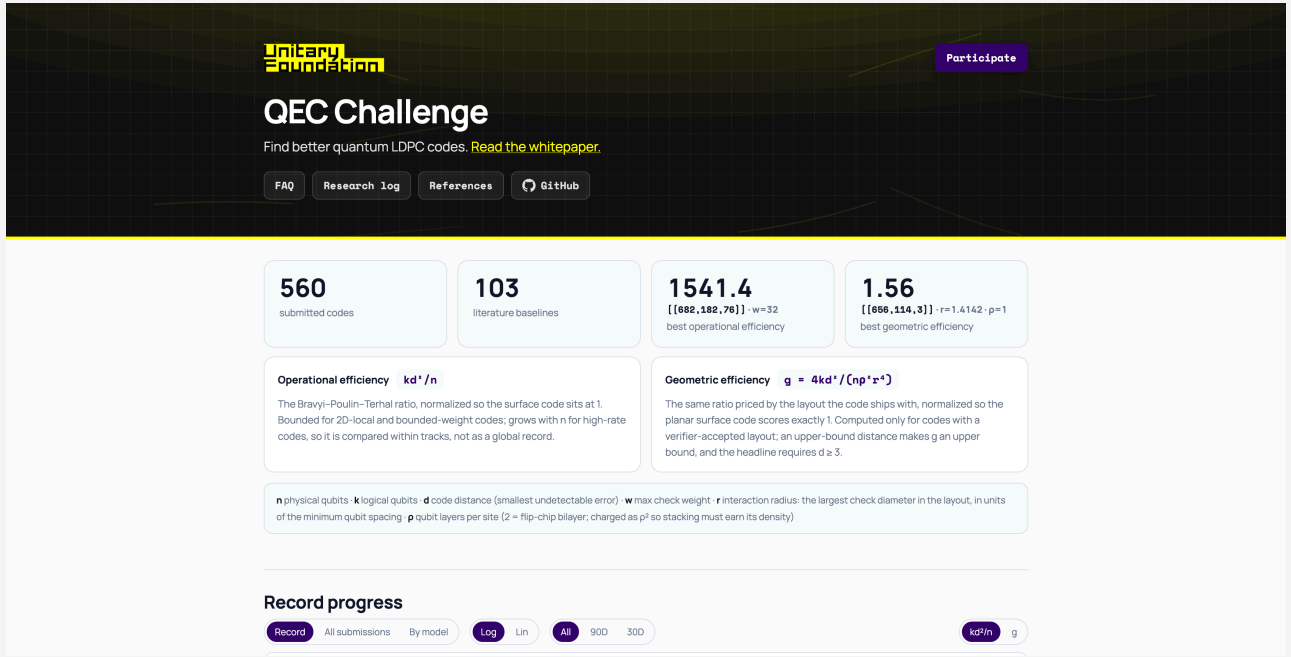
The ten codes behind the simulation plot

Code	Type	Weight	FOM	Note
[[144,12,12]]	CSS	6	12.0	Gross code (Bravyi et al.)
[[288,24,12]]	CSS	6	12.0	Decomposable, 2× gross
[[288,16,12]]	CSS	6	8.0	Shifts ≤ 3; indecomposable
[[288,50,8]]	CSS	8	11.1	Cross-factored
[[144,8,12]]	CSS	6	8.0	Mixed-monomial
[[360,20,≤14]]	CSS	8	≤10.9	Campaign 4
PBB [[144,12,12]]	non-CSS	8	12.0	Non-CSS gross-code analogue
PBB [[72,4,8]]	non-CSS	11	3.6	$d/\sqrt{n} \approx 0.94$
PBB [[360,12,≤20]]	non-CSS	8	≤13.3	Upper bound
PBB [[360,12,≤24]]	non-CSS	10	≤19.2	Published upper-bound tie

- Within our catalog, the largest reported FOM bound occurs at weight 10; FOM remains a comparison metric, not a validated proxy for logical error rate.
- Higher weight means more two-qubit interactions and typically greater syndrome-extraction depth.

BACKUP · COMMUNITY VERIFICATION

A public leaderboard provides an independent check



- The Unitary Foundation qLDPC Challenge runs an automated, **bounded refutation search** against every submission. Only CSS codes are eligible.
- Passing means no counterexample was found within the CI budget; it is **not** an exact-distance certificate.
- As of 17 September 2026: **560 submitted codes** and **103 literature baselines**.
- Third-party automated refutation complements our internal MILP/BLISS checks.

unitaryfoundation.github.io/qldpc-challenge/

BACKUP · RELATED WORK

Several groups are pushing this frontier

Group	Contribution	Venue
Bravyi et al.	The gross code	Nature 2024
Liang, Liu, Song & Chen	Generalized toric codes on twisted tori	PRX Quantum 2025
Wang & Mueller	Coprime BB codes; cold-atom layouts	Quantum 2026
Symons, Rajput & Browne	BB-code sequences from covering graphs	arXiv:2511.13560, 2025
Khesin & Lu	Mirror codes beyond the CSS regime	arXiv:2603.05496, 2026
Voss, Sim, Haug & Bharti	Multivariate bicycle codes	PRA 111, L060401, 2025
Lin & Pryadko	2-block group-algebra codes	PRA 109, 022407, 2024
Postema & Kokkelmans	Existence and asymptotic characterization of BB codes	arXiv:2502.17052, 2025
Webster, Jacob & Higgott	Distance-finding algorithms; BP-OSD sensitivity	arXiv:2603.22532, 2026
Chengyu et al.	Bayesian optimization for QEC-code discovery	arXiv:2601.18562, 2026
Liu & Marquardt	LLM-guided structured concept evolution	arXiv:2606.24808, 2026
Yan et al.	OmniQEC: AI-scientist-driven discovery	arXiv:2607.25865, 2026
Qian & Li	Multi-agent search with MILP witnesses	arXiv:2608.08996, 2026
Hong	Design-rate-1/5 qLDPC codes	arXiv:2607.27644, 2026
Bhardwaj et al.	High-rate qLDPC processor architectures	arXiv:2607.28795, 2026

RL-for-QEC lineage: Nautrup et al. (Quantum 2019); Su et al. (Phys. Rev. Applied 2025); Olle et al. (npj Quantum Information 2024); He & Liu (arXiv:2502.14372); Freire, Delfosse & Leverrier (arXiv:2501.09622).

BACKUP · CAMPAIGN DETAIL

Nine campaigns, two studies

weight-5 study to be submitted

Campaign	Family	Models	Codes	Notes
1	CSS trinomial	single model	—	Seed campaign
2	CSS trinomial	3-model ensemble	—	251 iterations, best = 349
3	CSS trinomial	3-model ensemble	—	500 iterations, best = 354
4	CSS mixed-monomial	ensemble, MILP-in-loop	97 CSS total (Campaigns 1–4)	Server config, pop = 750
5	Non-CSS PBB	ensemble, MILP-in-loop	368 PBB	~11 h, ~$100
Weight-5 companion study — to be submitted				
W5-1	CSS-small	Opus 5 + GPT-5.6 Sol (49/51%)	1,142 proposals total (912 CSS + 230 PBB)	
W5-2	CSS-large	same ensemble	seed 42, temperature 1.0	
W5-3	PBB-small	same ensemble		
W5-4	PBB-large	same ensemble		

1,142 weight-5 proposals is a genuinely separate count from the 465-code main catalog — never sum them.

BACKUP · COST AND REPRODUCIBILITY

Cost, compute, and reproducibility

~$400	~140 h	~1,650	~2×10⁵
main-study LLM inference	campaign wall time	main-study iterations	candidates screened

- LLM-cost breakdown: ~$237 across the five campaigns, ~$70 for the ablation, and ~$90 for exploratory and unsuccessful runs; these are dashboard estimates, not exact billing.
- The ~140 h figure is campaign wall time and excludes the post-campaign deep-MILP audit. Campaigns 4–5 ran on a 64-core, 251 GB server.
- The five-campaign main study and verification stack are open source at `github.com/qiskit-community/qcode-discovery`; the separate weight-5 study is not yet public.
- This was not a hyperscale-computing result. The bottleneck was verification rigor, not GPU-hours.

BACKUP · EXTERNAL TOOLCHAIN CROSS-CHECK

Cross-checking our catalog with external tooling

- Beyond the paper, we internally ran Webster, Jacob & Higgott's independent `codedistance` implementation on a selected set of **198 codes**: 43 CSS and 155 exact PBB entries. The toolchain uses OR-Tools/SCIP MIP plus Brouwer–Zimmermann enumeration where feasible.
- Result: **196 matching verified witness weights; 2 looser, compatible upper bounds; 0 contradictions**.
- Only **16 codes** received an independent exact Brouwer–Zimmermann proof. Most matches corroborate our upper-bound witnesses; they do not independently certify exact distance.

Important distinction: this cross-check is **our own internal verification material**. The *external* citation on the related-work slide, Webster, Jacob & Higgott (arXiv:2603.22532, 2026), is the concurrent published-methods paper their tooling comes from — a separate, citable thing from our internal use of it.

BACKUP · THE DISTANCE TRAP, IN FULL

Theorem: A = B forces d = 2

Theorem ("distance trap"). For the searched constructions with polynomial weight at least 2, every CSS bivariate bicycle code with `A = B` and `k > 0` has distance exactly `d = 2`.

- **Proof sketch:** CSS BB stabilizers lie in the diagonal subspace. When `k > 0`, some vector `(e_i, e_i)` lies in the normalizer but outside the stabilizer row space, producing a weight-2 logical operator.
- For construction polynomials of weight at least 2, the BB column structure rules out a weight-1 logical operator; hence `d = 2`.
- For PBB codes with `A = B`, analogous weight-2 operators lie in the normalizer; every such catalog entry was MILP-confirmed at `d = 2`.

BACKUP · VERIFICATION MACHINERY DETAIL

MILP formulation and the LC-equivalence caveat

- **CSS MILP**: minimum-Hamming-weight logical operator, solved via HiGHS/SciPy; exact only when optimality is certified.
- **Non-CSS (symplectic) MILP**: binary OR linearization of per-qubit X/Z support, since non-CSS weight is symplectic, not Hamming.
- **BLISS non-CSS extension**: the colored Tanner graph needs a *tying edge* between each stabilizer's X-support and Z-support vertices—without it, BLISS permits independent X/Z permutations, a broader relation than the intended stabilizer permutation equivalence.

LC-equivalence coverage: we tested all 36 uniform per-block assignments plus nonuniform `{I,S}`, `{H,HS}`, and `{I,H}` families. Gaps remain for nonuniform `{SH,HSH}` on block 1 and mixed-family nonuniform assignments; the 357-code claim is scoped accordingly.