BCH Orthogonal-Array Cubature for Deep Random ReLU MLPs

A hybrid estimator using homogeneity, exact first-layer transport, and weight-aware ReLU folding

LLM usage disclosure

This project used LLMs extensively. I directed the research, chose which experiments to run, reviewed the evidence, and made the final decisions. Codex, Claude, and Grok contributed to ideation, wrote much of the experimental and submission code, analyzed results, reviewed one another's implementations, and assisted with this write-up. I do not claim that every idea or line of code was independently produced by me. The identities and numerical claims below were checked against the submitted source, preserved experiment artifacts, and leaderboard results. Where I am offering an interpretation rather than a proved fact, I say so.

Phase 1 submission ID	Public adjusted score	Public final-layer MSE
`327651`	`1.23e-7`	`1.55e-7`

The short version

My submission is a hybrid estimator for width-256, depth-32, bias-free ReLU MLPs. It still samples the network, but the samples and the execution are chosen from the network's structure:

1. Positive homogeneity integrates the Gaussian radius analytically, leaving only a spherical integral.
2. The angular points come from a near-complete prefix of a strength-4 BCH orthogonal array rather than ordinary Monte Carlo or Sobol points.
3. A small pilot classifies neurons as almost always dead, almost always on, or uncertain. Dead paths are removed, on paths near the output are folded linearly, and full ReLU sampling is kept for uncertain neurons.
4. The first-layer BCH product and one antithetic continuation step are evaluated with exact Walsh–Hadamard identities, reducing cost without changing the estimator.

The statistical turning point was the BCH/OA table. The final leaderboard gain came from exact FWHT, antithetic, zero-skipping, and fast-matrix-multiplication identities that reduced the arithmetic actually performed. Every numerical operation remained inside the official `flopscope.numpy` interface; no work was moved outside the meter.

1. The estimator

Analytically remove the radius

The MLP has no biases, so ReLU makes the entire network positively homogeneous:

$$f(ru) = rf(u), \qquad r \geq 0.$$

For a Gaussian input $x = Ru$, where u is uniform on the sphere and R is independent,

$$\mathbb{E}[f(x)] = \mathbb{E}[R], \mathbb{E}[f(u)].$$

I therefore fix every direction at the exact mean Gaussian radius. This is not a radial approximation for this architecture: homogeneity makes the radial expectation separable. It removes one source of sampling variance and turns the remaining problem into angular cubature.

Use a BCH strength-4 design for the angles

Coordinates are indexed by $a \in \mathrm{GF}(256)$. For $r \in \mathrm{GF}(2)^{16}$, define

$$u_{r,a} = \frac{(-1)^{\langle r,(a,a^3)\rangle}}{\sqrt{256}}.$$

With the implicit leading coordinate, the columns are $[1, a, a^3]$. Any four distinct columns are independent over $\mathrm{GF}(2)$, so the complete binary sign array is an orthogonal array of strength 4. In plain language, the full table balances all interactions involving up to four coordinates exactly.

Antithetic evaluation identifies u and $-u$, leaving 65,536 line representatives in the complete design. The submitted estimator uses a fixed, target-independent permutation and the first **56,000 pairs**, or 112,000 signed rows. This prefix is no longer an exact strength-4 array, but it retained most of the empirical benefit while giving enough headroom for hard MLPs. A fixed Hadamard orientation reduces special alignment with the coordinate axes.

The reason I believe this helps is mechanistic but not a complete proof: after antithetic pairing, odd angular terms vanish; the OA then strongly suppresses the first remaining low-order interaction terms. The measured gain is unambiguous, while this harmonic explanation should be read as the most plausible account rather than a theorem about the depth-32 integrand.

Repair the first layer exactly

The first preactivation is exactly Gaussian, giving

$$\mathbb{E}[\mathrm{ReLU}(x^T w_j)] = \frac{\|w_j\|_2}{\sqrt{2\pi}}.$$

For each antithetic pair,

$$\mathrm{ReLU}(z) + \mathrm{ReLU}(-z) = |z|.$$

I replace the sampled layer-0 mean by the exact value and transport the mean discrepancy into the next preactivation:

$$b_1 = \left(\mu_0^{\mathrm{exact}} - \widehat{\mu}_0\right) W_1.$$

This is a cheap, known correction. Exact covariance transport helped an earlier sampling chassis but hurt the fixed-radius BCH design, so the submitted version uses mean transport only.

Spend nonlinear work on kink neurons

A diagonal Gaussian moment pass proposes three labels using $\alpha = \mu_{\text{pre}}/\sigma_{\text{pre}}$: dead for $\alpha < -3$, on for $\alpha > 3$, and kink otherwise. The first 256 antithetic pairs are also used as a pilot. A neuron is accepted as dead or on only when its observed firing rate supports the analytic label (a roughly 0.54% tail cutoff); ambiguous cases return to the kink class. These 512 pilot rows are reused from the main table, not added on top of the 112,000 submitted rows.

The labels alter the computation:

- dead columns are removed from the sampled network;
- surviving neurons are ordered by firing rate, making rarely active inputs contiguous;
- sparse rows in this cold prefix use gathered small products;
- on neurons in the final two layers are treated linearly; and
- kink neurons retain their sampled ReLU exactly.

The final-two-layer fold composes weights through the on block, while separately sampling paths that pass through a kink. This is the main white-box part of the forward engine: the weights and pilot tell us where the ReLU behaves like zero, a linear map, or a genuine kink.

This dead/on/kink execution chassis was inspired by the SOX team's public structure-aware write-up accompanying submission '319341'. I implemented and extended the idea for my estimator.

Apply the BCH structure with exact counted algorithms

The BCH sign matrix has a sparse Walsh spectrum. Its layer-0 product is therefore evaluated by an exact fast Walsh–Hadamard transform (FWHT). The submission ships only about 0.19 MB of masks, signs, and permutations rather than a large floating-point table.

Antithetic pairing gives another exact identity:

$$h_0(-u) = h_0(u) - z_0(u),$$

and therefore

$$h_0(-u)W_1 = h_0(u)W_1 - u(W_0W_1).$$

The second term is evaluated through the same BCH/FWHT map. This replaces one of the two nonlinear layer-1 products for each pair by a structured linear correction.

The remaining products use a shape-dependent two- or three-level Winograd/Strassen recursion, sparse row packing, and a neuron order propagated between layers. Winograd/Strassen reduces the number of scalar multiplications by standard exact matrix-multiplication identities; sparse packing skips products whose ReLU inputs are known zeros. These identities are exact over real arithmetic; their only numerical effect is a small change in float32 operation order. The bulk forward pass is float32.

Every runtime numerical operation including the FWHT, additions, gathers, sparse products, and dense products uses the public `flopscope.numpy` interface and is charged by the same official cost model as any other submission. No custom native kernel, unmetered numerical package, or accounting bypass is used.

What the submitted program returns

The execution is deterministic for a given MLP and the shipped BCH metadata:

1. A diagonal Gaussian pass produces means, variances, and provisional labels for all 32 layers.
2. The FWHT constructs the 56,000 positive layer-0 preactivations; antithetic identities supply the corresponding negative rows.
3. The first 256 pairs run as the dense pilot and also contribute to the final accumulated mean.
4. The remaining 55,744 pairs continue through the pruned and folded network.
5. Layer 0 is returned from its exact Gaussian formula. Layers 1–30 are returned from the diagonal analytic pass. At layer 31, on and kink neurons use the high-count continuation, while neurons conservatively classified dead use the analytic fallback.

Thus the high-accuracy sampled computation is concentrated on the scored final layer, although its trajectories still pass through the earlier layers and use their realized neuron structure.

2. Evidence and ablations

The public progression most relevant to this submission was:

Estimator checkpoint	Adjusted	Final MSE	Main change
Structure-aware dense sampler	`2.48e-7`	`2.87e-7`	add dead/on/kink pruning and final-two-layer linear folds
Fused L0 covariance + cost prune	`1.69e-7`	`3.36e-7`	transport exact L0 mean/covariance; remove redundant symmetric-Gram work
62k-pair BCH/OA	`1.40e-7`	`1.44e-7`	replace the input table by a near-complete BCH prefix; retain L0 mean only
Safe 56k-pair BCH + FWHT	`1.33e-7`	`1.55e-7`	reduce pair count for hard-MLP safety; factor the BCH L0 product by FWHT
Exact algebraic rewrite	`1.25e-7`	`1.55e-7`	one-write assembly, Winograd formulas, sparse packing, and propagated layout
Exact algebra + L1 antithetic fold	`1.23e-7`	`1.55e-7`	**add the exact L1 pair identity; referenced submission**

These are historical leaderboard checkpoints rather than one perfectly controlled ablation because Flopscope accounting evolved during Phase 1 (though the variation was minimal in my case). The last three are the cleanest comparison: their statistical estimator and reported raw MSE are effectively the same.

For the final algebraic pass, a representative local MLP moved from `192.566B` billed FLOPs to `178.923B`. Across four tested MLPs, the mean saving was about `6.7%`, while MSE ratios stayed within roughly `0.1%` of one. This saving came from the exact identities above and standard fast matrix multiplication, not from changing how operations were priced. The public adjusted score improved by `7.5%` from the first safe FWHT version to the final submission without a reported raw-MSE change.

The complete 65,536-pair table was more accurate locally, but hard MLPs left essentially no budget margin. A 62k FWHT version also encountered watchdog failures. I deliberately submitted 56k pairs: in this challenge, one failed MLP is much worse than a small average gain.

3. How I arrived here

The initial direction was analytic cumulant propagation. K=2 covariance propagation was useful, and higher-order diagnostics taught me why it failed at depth: important off-diagonal cumulants grow through the ReLU layers, and partial K=3 corrections can be worse than the Gaussian closure. That work produced the exact first-layer correction, but not a competitive standalone estimator.

I then tried Sobol and orthogonal QMC tables, radial rules, learned residuals, control variates, restart estimators, MUB/Kerdock designs, kernel weighting, and per-MLP table selection. Some gave good small-set or oracle results, but most failed on larger held-out MLP sets. I learned that a strong oracle is useless when its expectation or selector is as hard to estimate as the target, and that small table comparisons are noisy enough to require larger paired gates.

The decisive combination was a public structural observation dead/on/kink execution with the revisited BCH/OA table. The structure-aware engine made a previously unaffordable algebraic design shippable. The final round of work then treated the implementation as an algebra problem: preserve the estimator while using exact identities to avoid products and intermediate arrays that do not need to be computed.

4. What I think the contribution is

This is still a sampling-heavy estimator, so I do not present it as a complete solution to mechanistic white-box estimation. The contribution I would claim is narrower:

- positive homogeneity converts Gaussian integration into a purely angular problem;
- a BCH strength-4 sign design gives a large, repeatable reduction in the angular error of these deep random ReLU networks;
- exact first-layer information and neuron-state classification turn that design into a hybrid white-box estimator; and
- the BCH algebra supports exact FWHT and antithetic folds, reducing the counted arithmetic needed for each structured estimate rather than hiding work or merely increasing black-box throughput.

This estimator is specialized to width 256 and bias-free ReLU networks, but its main lesson is more general: sampling and mechanistic estimation can reinforce each other. Here, homogeneity removes the radial integral, BCH balance suppresses low-order angular error, and the realized weights reveal which ReLUs can be treated linearly. Together these ideas reduced the public adjusted score from `2.48e-7` for the first structure-aware implementation to `1.23e-7` for the selected submission. I hope this decomposition analytic symmetry, algebraic cubature, and weight-dependent local linearity is the useful result beyond the leaderboard.