

Zero fitted constants, measured: what the 64 calibration numbers in a ReLU-MLP mean propagator are actually absorbing

ARC White-Box Estimation Challenge (WhestBench) — Phase 2 technical write-up. Shape: width 1024, depth 16, budget 2^{41} .

Jigon Yoo (AICrowd: yujigon)

6 September 2026

This write-up refers to submission #330001 (estimator v6). Graded on the Phase 2 public split on 2026-09-06: 100/100 MLPs completed (50 public scored, 50 private sealed), 0 failures, adjusted score $1.066\text{e-}07$ on the public split.

Result

One corpus, one label, every row. 32 MLPs, seeds 200–231, 4×10^6 -sample ground truth, pre-registered before measurement (commit f2445768). **None of these networks was used to design or tune any estimator below.** Two prior contacts are disclosed rather than hidden: seeds 200–215 (16 of the 32) had been evaluated once before, at 10^6 samples, as the Phase F holdout on which v5 was chosen over v2 as the score candidate (Section 7) — the 4×10^6 -sample ground truth here is new for all 32, and seeds 216–231 had never been run; and the pre-registered branch decision (32 networks $\times$ 4M samples versus $64 \times 1\text{M}$) was taken from a GT-noise estimate on seeds 200–203 (PHASE_H_FINDINGS.md, H1), using v2 and v6 outputs on those four networks.

estimator	fitted constants	sampling	final-layer MSE	vs baseline	compute	adjusted
bundled	—	—	4.141153e-06	1.00x	2.35%	4.14e-07
03_covariance_propagation						
v2 exact ReLU covariance + fitted mean correction	64	none	1.230525e-06	3.37x	2.36%	1.23e-07
v4 + derived one-step k3 term	0	none	2.358626e-06	1.76x	3.83%	2.36e-07
v5 v4 + residual fitted correction	64	none	1.091853e-06	3.79x	3.83%	1.09e-07
v6 inherited-cumulant CP channel	0	none	1.134226e-06	3.65x	9.89%	1.13e-07

Official public-split grade of the referenced submission (#330001, v6):

split	MLPs run / scored	failed	final-layer MSE	adjusted score	vs MC	compute	participant rank (of 877)
Phase 2 public	100 / 50	0	1.066e-06	1.066e-07	11x	9.89%	64

The official public split scored v6 at $1.066\text{e-}06$, **6% better than our own 32-network holdout (1.134e-06)**. Holdout error was, if anything, pessimistic — the opposite direction from selection-bias inflation. Wall time was 3.86–4.56 s per MLP against the 120 s cap; the residual wall-time

margin discussed in Section 9 was not approached. The participant rank is the leaderboard position at the time of writing (2026-09-06, ~05:00 UTC) and moves as others submit.

The fitted comparator was graded too, after a smoke-test failure and a one-line patch (Section 8): **v5b, submission #330006 — 100/100 completed, 0 failures, final-layer MSE 1.048e-06, adjusted 1.048e-07, compute 3.83%**. On the official public split v5b is 1.7% better than v6; on our 32-network holdout v5 was 3.7% better and the difference did not resolve (Section 6.2). The two measurements agree in direction and in scale, and neither changes which submission this write-up is about — v5b carries the 64 fitted constants, v6 carries none.

The result this write-up is about is the last row of the first table.

v6 vs v2	+7.264114 pp	95% CI [+2.591824, +11.936404]	paired t p = 0.003413
		Wilcoxon p = 0.005026	wins 23/32
v6 vs v5	-4.949303 pp	95% CI [-11.092403, +1.193798]	paired t p = 0.110454
		Wilcoxon p = 0.154479	wins 11/32

An estimator with zero fitted constants and zero sampling beats the 64-constant fitted baseline, and the effect is resolved. Against the *best* fitted estimator it is not distinguished — and “not distinguished” is not “equal”: v5’s mean MSE is 3.74% lower, and this comparison had the power to detect only differences of 8.44 pp or more, so it could not have resolved the 4.95 pp it observed. That limitation is stated here rather than in a footnote.

All five sit at the $\max(0.1, C/B)$ multiplier floor, so every difference is accuracy and none of it is bought with compute. Every MSE and ratio in the first table is regenerated by `make_canonical_numbers.py` and carries its corpus in `CANONICAL_NUMBERS.md` (the compute column comes from the metered three-MLP harness runs of Section 9, and the second table from the official grader); **an earlier draft of this table had three rows from three different corpora under one label, and that is why the file exists.**

Related public Phase 2 write-ups, as of 2026-09-06. Two are on the challenge forum. `venture_agent_ai` (submission 329951, adjusted 1.32e-07 on the public leaderboard) reports the same two-part construction as this document’s v2 — an exact bivariate ReLU covariance plus a per-layer ridge calibration fitted on local MLPs, applied to the recorded output rows rather than fed back into the loop — and four negative results on sampling. `baghyeonsig` (submission 328274, 3.04e-07) blends gain-covariance propagation with whitened antithetic Monte Carlo. Neither asks what the fitted calibration is absorbing or whether it can be removed; that is the question here, and the zero-constant estimator scores 1.066e-07 against both. (Forum threads 18214 and 18200; I have read the posts, not reproduced their measurements.)

1. The premise that scoped this work, and why it was wrong

The score is $\text{final_layer_mse} \times \max(0.1, C_m / B_m)$. Everything below follows from one reading of that formula, and **the reading was wrong**. I am putting the correction first because it determined every design choice in this write-up, and a reader is entitled to know the boundary of the search before seeing its results.

What I believed. *Plain* Monte Carlo has error σ^2/k at cost proportional to k , so the product $\text{MSE} \times C/B$ is constant in k . Sampling more does not improve the score; sampling less does not either, until the 0.1 floor, after which it strictly hurts. I took that to mean **sampling cannot be tuned at all**, concluded that the problem was *maximise analytic accuracy subject to staying under 10% of budget*, and never revisited it.

Why that is false. The invariance is a property of the $O(1/k)$ variance law, not of sampling. A randomised quasi-Monte Carlo estimator — scrambled Sobol, for instance — does not obey it: for sufficiently smooth integrands scrambled-net RMSE decays faster than $k^{-1/2}$, so $\text{MSE} \times C/B$ **falls** with k and buying samples buys score. The ReLU kink degrades that rate and I have not measured what rate is actually achieved at this shape, so I make no claim about how much faster.

The only claim I need is that the product need not be constant, and that is enough to void my premise.

What it cost. The leaderboard shows it directly. Leading Phase 2 submissions sit at a compute multiplier near 0.19, meaning they deliberately spend roughly twice the free allowance because the accuracy it buys more than pays the penalty. Every estimator in this write-up sits under a tenth of the budget, at the floor — v2 at 0.024, v4 and v5 at 0.038 with 2.6x of free compute unspent, v6 at 0.099 — and the paid region was never examined. **I did not run out of budget; I ruled it out in the second paragraph of my own design notes and never came back.**

Why this is still worth reading. The premise was wrong about *sampling*, not about the mechanistic question. What follows is a study of what the fitted constants in a Gaussian-closure propagator actually are, and that question is independent of how many samples anyone draws. Section 6 measures where the closure family stops, and Section 3 contains the same limit measured a second way. **A reader deciding whether to spend compute on samples or on closure should read those two sections and then not do what I did.**

One correction of fact carried from a review. Earlier drafts asserted “the `mlp.seed` RNG is never drawn from.” That is true of v2, v4 and v5, the three estimators that matter here, but **false as stated:** `estimator_v3.py`, which this write-up names in Section 7, does draw from it. The absolute is withdrawn; the scoped claim stands.

2. The bundled ReLU covariance is the first term of a series

The bundled estimator approximates the post-ReLU covariance by a gain factor, `cov_post[i,j]` $\approx \text{Phi}(\alpha_i) \text{Phi}(\alpha_j) \text{cov_pre}[i,j]$. For jointly Gaussian pre-activations this is exactly the $k=1$ term of the Hermite (Mehler) expansion:

$$\begin{aligned} \text{Cov}(\text{ReLU}(z_i), \text{ReLU}(z_j)) &= \sum_{k \geq 1} c_k^{(i)} c_k^{(j)} \rho^k \\ c_1 &= \sigma \text{Phi}(\alpha) \\ c_k &= \sigma \text{phi}(\alpha) \text{He}_{k-2}(-\alpha) / \sqrt{k!} \quad (k \geq 2) \end{aligned}$$

from $E[f(u) \text{He}_k(u)] = E[f'(u) \text{He}_{k-1}(u)]$ with $f'(u) = \sigma \cdot 1\{u > -\alpha\}$. Checked against 8e6-sample Monte Carlo on a (μ, ρ) grid spanning ρ from 0.30 to 0.99: the bundled $k=1$ form understates the covariance by 4–26% there, and the series has converged by $K=3$ for ρ up to about 0.8 (at $\rho = 0.9$ $K=3$ is still 2% short and $K=6$ is needed for 0.1%). $K=4$ costs $(4K+2)n^2$ per layer — about 0.014% of the budget over 16 layers.

The grid is not the operating range, and that matters. Later width diagnostics measured the off-diagonal ρ RMS this network family actually visits: **0.031 at layer 0 to 0.117 at layer 15** ($n = 1024$). At those correlations the $k=1$ deficit is **1–3%** (at $\alpha = 0$; up to 12% at $\alpha = -1$), not 4–26%, and the “converged by $K=3$ ” statement is comfortable rather than marginal. **The validation grid was chosen to stress the expansion, not to represent the problem,** and earlier drafts quoted its worst-case numbers without saying so.

Effect alone, on the original 12-network corpus (seeds 0–11, 10^6 samples, `CANONICAL_NUMBERS.md`): final-layer MSE $4.208e-06 \rightarrow 3.977e-06$. **Almost nothing — and that is the interesting result,** which is what forced the next section. (Two earlier drafts printed two other pairs for this quantity, from an unrecorded corpus and from an 11-file replica respectively; the figure above is the one the canonical script regenerates.)

3. The error is accumulated drift, and it lives in the mean

Scope of this section: every figure below is measured on a single MLP (`data/mlp0.npz`), and the “true” mean and covariance rows are themselves 10^6 -sample estimates. The statements are descriptive of that network. They are stated as general claims in places below because that is how they read, and they should not be

— one network is not a sample. Section 3.1 is where this matters most, and it carries its own scope note.

The one-step map is accurate to within a same-sample noise indicator:

```
one-step error (true layer-14 state -> layer 15)    1.50e-07
same quantity from an overlapping half-sample      1.72e-07    <- noise indicator
full propagation                                  3.82e-06
```

On that second row. Earlier drafts called it “two independent MC estimates of the same” and read it as a noise floor. `onestep.py` builds the second estimate from **the first half of the same stream** that produced the first, compared against the same reference, so it is a same-sample indicator, not two independent estimates. A disjoint-pair comparison is a different and larger number. The qualitative reading — the one-step map is far more accurate than the full propagation — survives; the phrase “exact to within measurement noise” does not.

Nearly all the error is accumulation. Swapping components of the layer-14 state for their Monte-Carlo truth localises it:

layer-14 state	final MSE
propagated mu, propagated cov	3.82e-06
true mu , propagated cov	3.18e-07
propagated mu, true cov	3.87e-06

The propagated covariance is wrong by 28% in relative off-diagonal terms and it does not matter. $E[\text{ReLU}(z_i)]$ depends on cov_pre only through $w_i^T \text{cov } w_i$, a quadratic form averaging over 10^6 entries; off-diagonal errors cancel. **On this network, the mean carries everything.**

A negative result worth keeping: injecting the true covariance at *every* layer while propagating the mean makes the answer worse ($3.82e-06 \rightarrow 6.98e-06$). The propagated mean and covariance drift together into a self-consistent shifted state, and correcting one alone breaks the consistency. Partial oracle information was harmful on this network.

3.1 The same table read as a ceiling

I built the table above as a diagnostic and read it as one for three drafts: *the error is in the mean*. It is also the strongest limit measurement in this write-up, and reading it that way changes what the rest of the work can be worth.

Row two is an oracle. It hands the propagator the Monte-Carlo-true layer-14 mean — the single quantity Sections 4 through 7 spend their entire effort estimating — and the final MSE still stops at **3.18e-07**. Nothing downstream of that row is reachable by improving the mean. One correction to that number, carried over from an earlier review: both the injected mean and the reference are 10^6 -sample Monte-Carlo draws, and two independent 10^6 -sample targets on a fresh network differed by $1.34e-07$ in MSE (variance-predicted $1.57e-07$). Between roughly $0.7e-07$ and $1.3e-07$ of the $3.18e-07$ is therefore measurement noise, and the noise-corrected row sits near $2e-07$. The “16x” below is more honestly “10–12x”. The direction of the argument does not change; its magnitude does, and an earlier draft had mistaken this same noise for a floor.

Stated precisely, because the scope matters more than the number:

Oracle scope: Phase 2 shape ($n = 1024$, $L = 16$); the true mean injected at layer 14 only, not at every layer; covariance still propagated; measured on this work’s own corpus. A wider oracle — true marginals at *every* layer — would give a different and probably larger number. **I did not measure that, and this figure must not be quoted as “the closure ceiling” without the scope attached.**

Within that scope: leading Phase 2 submissions report final-layer MSE near **1.96e-08**. That is **16x below a bound my own family cannot pass even when handed the answer at layer**

14, which is not a statement about my execution — it is a statement about the family. A method reaching $1.96\text{e-}08$ is carrying information this propagator never had.

A second, independent measurement points the same way. In the public challenge forum, participant thylinao reports, for the *Phase 1* shape (256×32) and with **perfect marginals at every layer**, a floor of $7.07\text{e-}07$, and draws the operational conclusion in one sentence: “Give the estimator perfect marginals at every layer and leave only the correlations approximate, and it still cannot get below $7.07\text{e-}07$. That is what says ‘spend the budget on samples’.” (Source: AICrowd challenge forum, thread “*Phase 1 SOLUTION, ideas for Phase 2 and the honest public frontier (all methods combined and explained)*”, thylinao’s reply, <https://discourse.aicrowd.com/t/phase-1-solution-ideas-for-phase-2-and-the-honest-public-frontier-all-methods-combined-and-explained/18193>. Quoted verbatim from the thread as displayed on 2026-09-06; I have not reproduced the measurement.) **That number and mine are not comparable** — different shape, different oracle scope, different corpus — and I am not combining them. They agree only on the shape of the claim: *closure-family estimators have a floor that oracle marginals do not remove, and it sits well above what the leaders achieve*.

So Sections 4 to 8 should be read for what they are: **a mechanistic account of one term inside a family whose ceiling I have now measured, not a route to the top of the leaderboard**. Section 1 says why I did not find that out sooner.

4. What the 64 fitted constants actually were

v2 applied a per-layer linear correction in the loop,

```
mu_l <- b0*mu_post + b1*(sigma*alpha*phi) + b2*(sigma*phi) + b3*(sigma*alpha^2*phi)
```

with b fitted greedily, offline, on MLPs from the same specified distribution: 16 layers $\times$ 4 coefficients = 64 numbers shipped in the estimator. They are stable across seeds — the optimal final-layer scalar has mean 0.998305 and population sd $1.70\text{e-}04$ over **12** independently seeded MLPs (seeds 0–11, CANONICAL_NUMBERS.md) — but **only at this shape**; Section 8.2 discusses what happens when the width changes, and why that measurement is underpowered.

(This one figure has carried three different values across drafts — 11 / $1.8\text{e-}04$, 12 / 0.998305 / $1.70\text{e-}04$, 11 / 0.998336 / $1.63\text{e-}04$ — and every one of them was a true statement about a different copy of the corpus: a replica with eleven files prints the third; the full twelve-network corpus prints the second. **Nothing was mis-measured. The label was missing.** The value above is the one `make_canonical_numbers.py` regenerates, with its corpus attached, and that file now governs every corpus-labelled number in this document.) That worked, and it is the sort of thing that wins a leaderboard and teaches nobody anything. **The question this write-up answers is what those numbers are.**

They are an Edgeworth correction, and the basis is not a heuristic — it is forced.

Let z have mean μ , s.d. σ , and standardized cumulants k_3, k_4 beyond Gaussian. Its density has the Gram-Charlier form $p(t) = \phi(t)[1 + (k_3/6) \text{He}_3(t) + (k_4/24) \text{He}_4(t) + \dots]$ in the standardized variable. Then with $\alpha = \mu/\sigma$,

$$E[\text{ReLU}(z)] = \sigma * \int_{-\alpha}^{\infty} (t + \alpha) p(t) dt$$

and the whole calculation reduces to one identity, which is the crux:

$$\int_{-\alpha}^{\infty} (t + \alpha) \phi(t) \text{He}_k(t) dt = \phi(\alpha) * \text{He}_{k-2}(-\alpha)$$

This identity is the one thing in the derivation that everything else rests on, so it was checked numerically before it was used. `identity_check.py` in this repository runs `scipy.integrate.quad` against the closed form for $k = 3, 4, 5$ and α in $\{-1.3, -0.4, 0.0, 0.7, 2.1\}$ and reports

max_abs_error 8.882e-16 — machine precision, not approximation. (Earlier drafts quoted “9e-16” for this and shipped no script; a review found the only identity check in the repository was a different, coarser one at 1.950e-11. The script that produces the quoted figure is now included.)

Substituting $k=3$ and $k=4$ and using $\text{He}_1(-\alpha) = -\alpha$, $\text{He}_2(-\alpha) = \alpha^2 - 1$:

$$\begin{aligned}\Delta E[\text{ReLU}(z)] &= \sigma \phi(\alpha) [-(k_3/6) \alpha + (k_4/24)(\alpha^2 - 1)] \\ &= -(k_3/6) \cdot (\sigma \alpha \phi) + (k_4/24) \cdot (\sigma \alpha^2 \phi) - (k_4/24) \cdot (\sigma \phi)\end{aligned}$$

The three non-Gaussian basis functions are $\{\sigma \alpha \phi, \sigma \alpha^2 \phi, \sigma \phi\}$. **Those are exactly b_1, b_3, b_2 .** The fitted basis I had chosen for its shape is the Edgeworth basis.

One claim here does not survive, and I am withdrawing it. Earlier drafts added that the fitted coefficients *are* the Edgeworth coefficients — $-k_3/6$ and $\pm k_4/24$ averaged over the layer. The derivation forces $b_3 = -b_2$ if that were so. **In the shipped BETA, the sign of b_3 matches $+k_4/24$ in 0 of 15 layers.** The basis claim is what the derivation supports and what Section 4.1 tests; **the coefficient identification is refuted by the fitted values themselves** and was never checked before it was written. The greedy per-layer fit is absorbing more than the two Edgeworth cumulants — Section 8 lists what else is in there.

4.1 What the split-sample test actually measured — and what it did not

If the correction is really an Edgeworth term, then a residual should be predictable from the measured cumulants of the true pre-activations. That is falsifiable, and it can be made airtight by measuring the two sides on disjoint sample sets so no shared noise can manufacture agreement.

Two independent draws, seeds 1001 and 2002, 250k samples each. The residual is measured on set A, the Edgeworth prediction built from k_3, k_4 measured on set B.

layer	corr(residual, k3+k4 prediction)	k3 only	k4 only	variance explained
15	0.9749	0.9132	0.0706	0.9630
8	0.9468	—	—	0.9452
0	~0 (corr)	—	—	-0.61

The result is real. My reading of it was wrong, and an adversarial review of this draft caught it. The correction is worth more than the original claim, so it gets the space.

Which residual this is. `splitcheck.py` computes

$$\text{resid} = E[\text{ReLU}(z)]_{\text{measured}} - (\mu_{\text{true}} \cdot \Phi(\alpha_{\text{true}}) + \sigma_{\text{true}} \cdot \phi(\alpha_{\text{true}}))$$

from the **Monte-Carlo-true** pre-activation moments. That is the *one-step, oracle-input* non-Gaussianity: given the exact marginal, how much does the Gaussian closure miss? The answer is 96.3% Edgeworth at layer 15, carried almost entirely by k_3 — a clean and, I think, genuinely useful measurement.

Which residual the 64 constants absorb. Something else entirely: `gt[l] - propagated[l]`, the error of the *propagated* state, which has accumulated through fifteen layers. Measured on the same MLP:

layer 15	rms	corr with Edgeworth prediction	variance explained
oracle-input residual (what 4.1 tested)	3.27e-04	+0.9749	+0.9630
propagated residual (what the fit absorbs)	1.69e-03	-0.0593	+0.0482
propagated residual, scalar channel removed	1.13e-03	+0.1982	+0.0066

`corr( oracle-input residual , propagated residual ) = -0.0495`

They are 5.2x apart in magnitude and essentially orthogonal. So the sentence earlier drafts ended this section with — “*the 64 constants were a low-rank stand-in for the standardized third cumulant*” — **is withdrawn. The experiment does not test it, and the quantity it does test is uncorrelated with the one that would.**

4.2 The correction is a better result than the claim it replaces

Three things follow, and together they are more useful than what I was trying to say.

First, Section 3 predicted this and I did not connect it. Section 3 measured that the one-step map is accurate and that essentially all the error is accumulation. If that is true, then the propagated residual is *made of* accumulated drift, and the local non-Gaussianity — which is what Edgeworth describes — is a small orthogonal component of it. The 5.2x magnitude gap and the -0.05 correlation are exactly that statement, measured. **Two sections of this write-up were saying the same thing and I read them as unrelated for three drafts.**

Second, it explains v4. Section 5 derives the one-step k3 term mechanistically and Section 7 reports that v4 lands at 2.53e-06 against fitted v2’s 1.34e-06 (first holdout, seeds 200–215). That looked like a disappointing partial success. It is not: **v4 correctly computes the small orthogonal component, while v2’s fitted constants are absorbing the large accumulated one.** They are not competing approximations of the same quantity — they are addressing different quantities, and the fitted one is addressing the bigger one. A derived estimator that only fixes local non-Gaussianity *should* lose to a fitted one that absorbs drift, and now there is a number attached to why.

Third, it says where a derived approach would have to aim, and that turned out to be testable. Not at the ReLU non-Gaussianity — that is solved, to 96%, by the formula in Section 4. **At the accumulation.** Section 6’s inherited-cumulant channel is the one mechanism in this work aimed at accumulation rather than at the local term.

This was written as a prediction, before the measurement existed, and it held.

The reading above says a derived estimator aimed at accumulation should be the one that competes with fitted constants, and a derived estimator aimed at the local term should not. On the pre-registered 32-network holdout: the local-term estimator (v4) lands at 1.76x the baseline and loses badly to fitted v2, while the accumulation estimator (v6) reaches 3.65x and **beats** v2 by 7.26 pp with a confidence interval excluding zero.

The correction in 4.1 is the reason this write-up knew where to look. It is the only instance in this work where withdrawing a claim produced a testable one.

What survives unconditionally. The derivation in Section 4 is algebra, not measurement: the basis $\{\sigma \cdot \alpha \cdot \phi, \sigma \cdot \alpha^2 \cdot \phi, \sigma \cdot \phi\}$ is forced by the identity, and the identity is verified to 8.88e-16 by `identity_check.py` in this repository. **A fitted linear correction on that basis is fitting an Edgeworth-shaped family whether or not its coefficients are Edgeworth coefficients** — and Section 4’s closing paragraph already withdrew the claim that they are. What 4.1 adds is a measurement of the one-step term; what it does not add is a bridge from that to the fitted constants.

On the layer-0 row. Earlier drafts wrote the negative control as “~0” and called it clean. The correlation is indeed ~0, but the variance explained is **-0.61**: at layer 0 the pre-activations are exactly Gaussian, there is nothing for the prediction to explain, and subtracting it is actively

worse than predicting zero. The control behaves as it should — **but “-0.61” is the honest way to print it and “~0” was not.**

5. v4: deriving k3 instead of fitting it

If the correction is $-(k_3/6) \cdot \sigma \cdot \alpha \cdot \phi$, then removing the fitted constants means computing k_3 from the propagated state. For $z_j = \sum_i W_{ij} a_i$ with a_i the previous layer’s ReLU outputs,

$$\kappa_3(z_j) = \sum_{\{i,k,l\}} W_{ij} W_{kj} W_{lj} * \kappa_3(a_i, a_k, a_l)$$

— the *joint* third-cumulant tensor of the previous activations, contracted three times. Expanding each a in Hermite polynomials of its underlying Gaussian, that joint cumulant is a sum over connected diagrams in the correlation ρ .

My first answer here was wrong, and the correction is the whole method. I guessed the leading connected term was the triangle $(2,2,2)$, which is $0(\rho^3)$, with coefficient 2. It is not the leading term: the **path $(2,1,1)$** is also connected and is $0(\rho^2)$ — one order lower in a quantity whose off-diagonal entries are small — so it dominates everywhere the network actually operates. Its coefficient is $3\sqrt{2}$, not 2. I had labelled the guess as a guess, and it was re-derived and falsified before it reached any code.

The path contracts to a single dense matrix multiplication per layer:

$$\kappa_3(z_j) \approx 3\sqrt{2} * \sum_i W_{ij} c_{2i} * (\sum_k \rho_{ik} W_{kj} c_{1k})^2$$

with the all-equal-index part of the truncation replaced by the **exact** marginal third cumulant of $\text{ReLU}(N(\mu_i, \sigma_i^2))$, carried forward as $(W^3)^T m_3$.

The result is an estimator with **zero fitted constants and zero sampling**: 84.2 GFLOP, 3.83% of budget (metered three-MLP harness run), still under the free floor. It recovers **40.18%** of the mean error that the $K=4$ Hermite propagation leaves behind (tuning corpus, seeds 100-107, 8 MLPs, 10^6 samples, CANONICAL_NUMBERS.md).

40% is not a win. $2.53e-06$ is worse than v2’s $1.34e-06$ on the first holdout, and $2.36e-06$ against $1.23e-06$ on the 32-network corpus. **v4 is a falsifiable ablation, not a competitive estimator**, and it is reported as one.

6. The inherited-cumulant channel — the result this work is for

v4 (Section 5) captures the third cumulant *generated* at each layer. What it does not carry is the joint third-cumulant tensor *inherited* through depth — an $n \times n \times n$ object at $n = 1024$, which is 10^9 entries and cannot be propagated at any budget. Section 4.2 argues that this inherited part, not the local one, is where the fitted constants are doing their work.

A CP decomposition of the inherited joint third cumulant carries it in $O(r \cdot n)$, and it works. (Diagnostic corpus for this table: mlp0 plus seeds 100-102, 250k-sample split A, PHASE_E_FINDINGS.md — a design-stage measurement, not the holdout of Section 6.1.)

skew correlation, before	0.4477
skew correlation, with the inherited-CP channel	0.7954
regression slope, before → after	2.62 → 2.10

Rank 3,072 = 3n cleared the pre-registered accuracy gate (corr > 0.70) at 0.7398 under the weight-only tensor-norm construction (the exact one-source-age construction reached only 0.6299 and failed it). The rank actually shipped, 4,608 = 4.5n, was set by the **10% compute ceiling**, not by accuracy: preflight measured 4,608 → 9.8894% and rejected 4,704 → 10.0093%.

estimator_v6.py is that channel, with **zero fitted constants and zero sampling**, and it is the submission this write-up is about. Its packaged bundle, manifest and archive are included.

6.1 The pre-registered test

The Phase E tuning corpus showed v6 ahead of fitted v2 by 5.13% on eight networks. **That was not evidence** — recomputing per-network gave mean +4.768%, sd 13.423pp, $t = +1.005$, $p = 0.348$, CI [-6.45, +15.99], i.e. indistinguishable from zero, and a power calculation said resolving a 4.77 pp effect needed about **62 networks**. An earlier draft of this write-up disposed of v6 in three words — “*dropped as dominated*” — giving neither the number nor the reason.

So the test was pre-registered and committed (f2445768) before any of it was measured: the decision rule, the analysis, and a variance decomposition to size the corpus.

The variance decomposition is why this worked. Replicate ground-truth draws on four networks put **67.1% of the paired variance in the ground truth itself**, not in the networks. Buying 4×10^6 samples instead of 10^6 therefore bought far more resolution than buying more networks would have, and the frozen branch rule (>30% noise → raise samples) selected **32 networks at 4M samples** rather than 64 at 1M. **Without that step this experiment would have been run at the size that could not answer it, for the third time in this project.**

6.2 What the result licenses, and what it does not

v6 vs v2 +7.264114 pp CI [+2.591824, +11.936404] $t p = 0.003413$ W p = 0.005026 23/32
v6 vs v5 -4.949303 pp CI [-11.092403, +1.193798] $t p = 0.110454$ W p = 0.154479 11/32

Licensed: a zero-fitted-constant, zero-sampling, purely mechanistic estimator beats the 64-constant fitted correction on a pre-registered holdout, by 7.26 pp, with the interval clear of zero on both a parametric and a rank test.

Not licensed, and stated because the distinction is the point:

- **“v6 is as good as v5” is not established.** The comparison did not resolve; v5’s mean is 3.74% lower; and **the achieved MDE for that comparison was 8.44 pp against an observed 4.95 pp, so it could not have detected the difference it saw.** Failing to reject is not evidence of equivalence, and an equivalence test was not pre-registered.
- **The v6-vs-v2 margin was thinner than it looks.** The achieved MDE was 6.418 pp against the 5 pp target; the corpus cleared the bar because the true effect (7.26) was larger than the Phase E estimate (4.77) that sized it, not because the design was adequate for 4.77. **Had the effect been the size I predicted, this corpus would have missed it.**
- **The channel is a cost result, not an impossibility one.** Carrying the inherited tensor at rank 4.5n costs more per contraction than the n^2 matmul it augments. Earlier drafts wrote that there was “nothing to exploit”; there was, and the price is 9.89% of budget against v5’s 3.83%. Both are under the scoring floor, so at this shape the price is zero — but it is a price, and at a tighter budget it would bind.

6.3 Why this is the claim worth making

The bundled covariance propagator is the previously-known method. Against it on the same corpus: **v5 reaches 3.79x using 64 numbers fitted offline; v6 reaches 3.65x using none.**

The 4% of accuracy that separates them is, on this evidence, not resolvable. What separates them is that **one of them can be written down.** Every coefficient in v6 comes from the Hermite expansion of the ReLU and the CP form of the inherited cumulant; there is nothing in it that was tuned to a corpus, and nothing that would have to be refitted at another width or depth — **which Section 8.2 shows is not true of the fitted constants.**

7. v5: the derived term plus what it cannot reach

Since the derived term is real but partial, v5 keeps it and re-fits a residual linear correction on top. **v5 is the better-scoring estimator and v6 is the one this write-up references**, for the reason in 6.3: on this evidence they are not distinguishable, and only one of them is derived.

A selection-bias check that matters more than the headline. **This table is an earlier, smaller corpus than the Result table** — tuning corpus seeds 100-107 (8 MLPs, 10^6 samples, PHASE_E_REPORT.md) against the first clean holdout, seeds 200-215 (16 MLPs, 10^6 samples, PHASE_F_HOLDOUT.md). Neither is the 32-network / 4×10^6 -sample corpus of the Result table, and the absolute values differ from it for that reason:

	tuning (100-107, 1M)	first holdout (200-215, 1M)	change
v2	1.365055e-06	1.338206e-06	-1.97%
v4	2.719812e-06	2.532581e-06	-6.88%
v5	1.223817e-06	1.212473e-06	-0.93%

Every estimator got better on the corpus it was never tuned on. If the fitted residual were absorbing corpus-specific noise, v5 would degrade out of sample; it does not. The Result table’s 32-network corpus and the official public split (Section Result) repeat the pattern: v6’s official grade ($1.066e-06$) is better than its own holdout estimate ($1.134e-06$).

v3 and v6 were not “dominated”, and earlier drafts saying so were wrong on both counts.

- **v3** is dropped because it is worse than v5 on accuracy *and* carries a sampling branch, which is the property this work is trying not to have. Its compute (9.92%) is **not** a reason — it is under the 10% floor, so it is free (Section 1). Saying “it costs 2.6x the compute” implied a penalty that the scoring function does not charge.
- **v6** is the strongest zero-fitted-constant result in this repository and is the referenced submission. At the time of this table (Phase E/F, 10^6 -sample corpora) it was *not* the score candidate: its improvement over v2 had not resolved and it degraded on the five networks outside its selection gate. That judgment was superseded by the pre-registered 32-network test of Section 6.1, where the v6-over-v2 effect resolved ($p = 0.003$); v5 remains the better point estimate, and Section 6.2 states what that comparison did and did not establish.

8. What did not work — including one experiment that proved nothing

Negative results constrain where the remaining error can be, so they are reported. Two of these are errors of mine that survived into a design, and the last one is the worst kind: a result that looks like a result.

- **Higher Hermite order beyond K=3.** Converged; K=6 and K=8 match K=4 to four figures (checked on the 11-file replica corpus only).

- **Covariance scale calibration.** The optimum sits at 1.000 on a seven-point grid; earlier drafts called it “sharp”, which that grid does not establish. What it does establish is that the propagated covariance scale is not the bias — only the mean is.
- **Diagonally-tracked skewness.** Earlier drafts reported 1.361e-06 with it against 1.315e-06 without, and concluded the diagonal independence approximation was too crude. **That comparison was not nested** — the skew variant replaced the α^2 feature rather than adding to it, so the two runs did not differ only in the thing being tested, and 1.315e-06 is not produced by the script that was cited. A properly nested re-test gave a noise-sized difference and **remains inconclusive**. The original conclusion may still be right; this experiment does not establish it.
- **Per-neuron feature regression at the output.** Eight propagation features beat a single scalar by **0.608%** in leave-one-out on the 12-network corpus (CANONICAL_NUMBERS.md; earlier drafts printed 0.4% and then 0.633% from an 11-file replica). The residual is still not a function of what the propagation computes.
- **The triangle diagram.** See section 5. Guessed, labelled as a guess, falsified.
- **svd_cost.** I asserted the harness exposed a metered SVD cost function. It exposes `svdvals_cost` only. The design that depended on it was discarded.
- **v5 failed the grader’s smoke test, and the failure had been found and dismissed.** Submission #330004 (v5) returned `SMOKE_FAILED · ESTIMATOR_EXCEPTION` before any MLP was scored. The cause is `estimator_v5.py:199, beta = BETA[layer_index]`, where `BETA` is a 16-row tuple: any network deeper than 16 layers raises `IndexError`. An adversarial review of the bundle **had found exactly this**, judged that it “does not fire in this round because the round is fixed at depth 16”, and recommended a one-line guard anyway. **I accepted the judgment and skipped the guard. The grader’s smoke test evidently uses a deeper shape, and it fired.** v6 and v4, which carry no fitted table, passed the same smoke test. (The same trap is reported independently in forum thread 18214: “the grader’s smoke test runs a shape other than 16 x 1024, so anything indexed by layer must be guarded” — seven failed uploads in that case.) The patched v5b (`beta = BETA[layer_index] if layer_index < len(BETA) else BETA[-1]`) is bit-identical to v5 at 1024x16 and no longer raises at depth 17 or 32. Resubmitted as #330006 it graded cleanly: 100/100, 0 failures, 1.048e-06 final-layer MSE, adjusted 1.048e-07. The transferable lesson is not about `BETA`: **when a reviewer finds a defect and the fix costs one line, do not adjudicate whether it will fire — apply it.** A “won’t fire” judgment is itself an unverified claim.

8.1 A void experiment, recorded rather than deleted

I designed a test to show that the layer-15 correction slope is a function of *depth* rather than of amplitude: scale all weights $W \rightarrow \sqrt{c} \cdot W$ for c in $\{0.7, 1.0, 1.4\}$ and check the slope is unchanged. It was unchanged, to 1.4e-06 relative:

```
c = 0.7    slope 2.617606616
c = 1.0    slope 2.617609600
c = 1.4    slope 2.617610201
```

This measures nothing. A ReLU network is positively homogeneous: $W \rightarrow \sqrt{c} \cdot W$ multiplies the layer- l activations by $c^{(l+1)/2}$ and leaves $\alpha = \mu/\sigma$, the correlation ρ , and every standardized cumulant **exactly** invariant. The knob could not have moved the quantity being measured. The agreement is a statement about my knob, not about depth, and any claim of the form “we showed the correction is a function of depth” would be false.

8.2 The replacement experiment, which is underpowered, and I designed that too

The properly powered version varies **width**, n in $\{512, 1024, 2048\}$ at fixed depth 16, because the off-diagonal ρ scales as $O(n^{-1/2})$ and ρ is a dimensionless state variable the estimator actually computes. Three MLPs per width, 10^6 ground-truth samples each.

The knob works, and that much is verified. Measured off-diagonal ρ RMS at layer 0 gives an endpoint ratio of 1.9973 against the predicted $\sqrt{2048/512} = 2$, and $\sqrt{n} \cdot \rho$ is flat to four figures across the three widths (0.9988, 1.0004, 1.0002). Unlike section 8.1, this perturbation demonstrably moved the quantity the estimator depends on.

The pre-registered gate fired. Against a threshold of 0.10 on the endpoint slope gap, fixed before the corpus was measured, all three diagnostic layers cross it: L8 -0.1333 , L12 $+0.1583$, L15 -0.2849 .

And the gate is smaller than the noise, which I did not check before setting it. Slopes vary across the three MLPs *within* each width by more than the gap between widths. (The gate values above are pooled-regression endpoint gaps; the table below uses the difference of per-MLP mean slopes, which is the quantity the within-width sd is defined on — hence 0.137 against -0.1333 , and so on. Both reproduce from PHASE_G_FINDINGS.md.)

layer	endpoint gap	within-width sd	gap / sd	Welch p (3 v 3)	one-way ANOVA p
L8	0.137	0.117	1.17	0.301	0.364
L12	0.164	0.361	0.46	0.673	0.853
L15	0.278	0.160	1.74	0.164	0.073

L12's between-width gap is **less than half** its within-width spread. L8 and L12 are each monotone in n — and monotone in **opposite directions**. No layer separates from zero effect at any conventional level.

To detect a 0.10 shift against a per-MLP sd of 0.16 at 80% power and alpha 0.05 requires about **40 MLPs per width**. I specified **three**.

So this experiment decides nothing. It does not show the slope is width-dependent, and it does not show it is width-invariant. It shows that a three-MLP corpus cannot tell the difference at the effect size I declared, and that I declared the effect size before I had a variance estimate to declare it against.

One procedural detail belongs here rather than in a footnote. The first implementation of the diagnostic added a requirement that all deep-layer changes share a direction, and on that stricter reading it printed WIDTH_INVARIANT_NEGATIVE. It was corrected to the pre-registered rule, which is the right call — the rule was $|\text{gap}| \geq 0.10$, and post-hoc tightening after seeing data is how false confidence gets built. But the fact that the verdict flips between two defensible readings the data cannot distinguish is the same finding as the table above, arrived at from the other side.

What this does and does not retract. Earlier drafts called the fitted coefficients a property of the model family, on the strength of their stability across 12 seeds (mean 0.998305, sd $1.70\text{e-}04$ — see Section 4). That claim is withdrawn — **but not because this experiment refuted it**. All 12 of those MLPs were at $n = 1024$. The claim was an extrapolation from a single shape the whole time, and the width experiment does not test it so much as fail to. The supportable statement is: the coefficients are stable across seeds at this shape, and nothing here licenses reusing them at another.

Why this still does not become a Phase 6. Phase 2 evaluates at one shape, $n = 1024$, $L = 16$. Even a fully resolved dependence on n carries **no score upside**: the fitted coefficients already sit at whatever value that dependence produces at $n = 1024$, so the final-layer MSE at the evaluated shape cannot move by construction. Buying the missing power costs roughly 40 MLPs at each of

three widths, and buys an expository result. The wall that *does* bear on the score is section 6's rank 4,608, and it is independent of all of this.

8.3 The pattern in 8.1 and 8.2

Two experiments, two failures, both mine, and they fail differently in a way worth naming. In 8.1 the knob could not move the target. In 8.2 the knob moved the target and the corpus could not resolve it. The first is an identification error and the second is a power error, and the common cause is that in both cases I fixed the decision rule before I had established what the apparatus could see. Neither is reported here for candour points: 8.1's slopes and 8.2's gate both produce numbers that read as findings, and a reader who saw only the numbers would draw conclusions this data does not support.

9. Reproduction and provenance

- Weights $W \sim N(0, 2/n)$, (1024, 1024), 16 layers, no biases, input $N(0, I)$.
- Ground truth 4×10^6 samples per MLP on the scoring corpus (seeds 200-231), float32 forward pass, float64 mean accumulation. The sample count was set by the variance decomposition in Section 6.1, not chosen.
- Seed protocol: `SeedSequence(input_seed).spawn(3)` — `ss[0]` weights, `ss[1]` ground-truth draws, `ss[2]` estimator, verified in `whestbench/seeds.py`. `mlp.seed` is domain-separated from the sample stream by construction. **v2, v4, v5 and v6 do not draw from it; estimator_v3.py:128 does** (Section 1).
- Two flopscope 0.12.1 defects met on the way are filed with cross-platform reproductions: AICrowd/flopscope#256 (explicit-output einsum broadcast `ValueError`) and #257 (symmetry-inferred einsum rejecting a finite symmetric result with a hard-coded inf deviation). Neither affects the referenced submission, which routes around both.
- Phase 2 rules: no numpy/scipy/BLAS/LAPACK; all arithmetic through flopscope. Every submitted estimator loops only over layers and over Hermite order, never over neurons — 112 Python iterations per `predict()`.
- **v6 on the official grader (submission #330001, 2026-09-06):** 100/100 MLPs completed, **0 failures**, wall time 3.86–4.56 s per MLP against the 120 s cap, compute utilisation reported by the grader 9.89%. The residual-time risk below did not materialise.
- **v6 on the local official harness before submission, 32 MLPs, both threading modes.** Residual wall time max 0.126644 s (default) and 0.154550 s (`OMP_NUM_THREADS=1`) against the 0.4 s cap — 0 breaches, but a **2.6x margin where v5 has 12x**, and this was the one number in the submission that is hardware-dependent. It was the reason v6 was submitted first, alone, as a test: a graded result converts a margin into a fact. Compute utilisation 9.886596% with **sd = 0.000000 pp** — FLOP counts are shape-determined, so it is exact and cannot cross the 10% floor at this shape. 0 failures, deterministic to the bit across two runs of seed 200, AST audit clean (no forbidden imports, no dynamic execution, no file I/O, no neuron loops).
- **What was independently recomputed, and by whom.** Every paired statistic in the Result table and in Section 6.2 was recomputed from the per-network table by a reader who did not run the experiment, and reproduced to the last printed digit: +7.264114 / -4.949303 pp, both confidence intervals, both t and Wilcoxon p-values, the win counts, and the achieved MDEs. The per-network table itself is in `PHASE_H_FINDINGS.md` **so that any reader can do the same**; publishing means rather than per-unit values is what made the Phase E version of this comparison unfalsifiable for three drafts.
- **What was not.** The corpus was generated once. **Earlier drafts claimed headline numbers were “re-run on a second, unshared machine”;** they were not — host metadata in all result JSONs is identical. That claim is withdrawn rather than quietly dropped. The three-MLP metered runs (v3, v4, v5) *were* repeated and matched to the digit on FLOP counts 218,241,829,871 / 84,182,335,471 / 84,182,482,927 per MLP and

utilisation 0.099245 / 0.038282 / 0.038282 (v2: 51,843,867,631, 0.023576).

Disclosure of LLM use

The challenge rules ask for full transparency about the extent of LLM involvement, and warn that unhedged claims will make the rest of a submission suspect. This section is written to that standard rather than to a comfortable one.

Who did what. The author is a human working with two LLM systems in a fixed division:

	Role
Human author	Direction, scope, go/no-go decisions, the decision to run adversarial review, and every judgement about what to claim and what to withdraw
Claude (Anthropic)	Research direction, the Edgeworth derivation, experiment designs, task packets for the implementing agent, and the prose of this document
Codex (OpenAI) agent	Diagnostic programs, coefficient derivations, <code>estimator_v4.py</code> and the later variants, all comparison runs, and the phase reports this write-up draws on

What “I” means in this document. It refers to that combination. Where the text says “*I guessed the leading connected term was the triangle*” (Section 5) or “*I designed a test*” (Section 8.1), the guess and the design were Claude’s, the implementation and falsification were the Codex agent’s, and the decision to publish the failure was the human author’s. **The first person is a convenience, not a claim of sole human authorship.** LLM_DISCLOSURE.md in the repository carries the per-phase breakdown through Phase G, including which conclusions were LLM inference that was later corrected; where that file and this section differ (it predates Phase H, and records write-up prose as a human action), this section is the current statement.

No claim in this document was verified by a human working by hand. Verification was done by running measurements, and by giving this write-up to **independent LLM reviewers that did not write it**, with instructions to falsify rather than confirm. That is a weaker guarantee than a domain expert checking the algebra, and it is stated plainly rather than glossed.

The review process changed the result, not just the prose. The largest finding below (item 1) withdrew this document’s central claim, and the corrected reading *predicted* where a derived estimator would have to aim. That prediction was then pre-registered and tested on 32 networks with new 4×10^6 -sample ground truth (16 of them previously seen only as the Phase F holdout — see the Result section), and it held (Sections 4.2 and 6). **The submission this write-up references exists because a reviewer refuted the draft’s main claim.**

What that review process actually found, in this draft. It is reported because it is the best available evidence about how much of this document to trust:

1. **Section 4.1’s headline result tested the wrong residual.** The 0.9630 variance explained is real, but it describes the one-step oracle-input residual, which correlates -0.05 with the residual the fitted constants absorb. The write-up’s central bridge — “*the 64 constants were a low-rank stand-in for the third cumulant*” — was withdrawn. **The reviewer found it by reading twelve lines of `splitcheck.py` and re-running the correlation.**
2. **Section 6’s `estimator_v6.py`** — then the strongest zero-constant result reported here — **was not in the submission bundle**, and neither were the scripts behind the headline holdout table. Both are now included, and v6 is the referenced submission.
3. **Section 1’s premise** — that sampling cannot be tuned — was false, and had scoped the entire search.
4. A seed-count figure that a *correction pass* had asserted by reading a constant ($N_{MLP} = 12$) instead of counting files. The count later proved right for the full corpus — 12, with an 11-file replica explaining the other value (Section 4) — but the pass that asserted it had not checked, and that is the defect.

What is verified, and how. The identity in Section 4 by quadrature (`identity_check.py`, 8.88e-16); the Hermite truncation in Section 2 against 8e6-sample Monte Carlo; the split-sample design in Section 4.1 on disjoint seeds so shared noise cannot manufacture agreement; the statistics in Sections 6.2 and 8.2 recomputed independently from the per-network tables; and the submission bundle checked against the real grader (`flopscope` 0.12.1, `whestbench` 0.16.1) — 0 failures, residual wall time at most 0.155 s against the 0.4 s cap for v6 (35 ms for v4/v5), no forbidden imports, no neuron loops.

What is reported but not verified here: everything drawn from `PHASE_E_REPORT.md`, `PHASE_F_HOLDOUT.md` and `PHASE_G_FINDINGS.md`, whose corpora are not in this bundle; the forum quotation in Section 3.1, which is quoted as displayed and not reproduced; the closure-ceiling figure of Section 3.1 outside its stated oracle scope; and the propagated-residual comparison in Section 4.1 (second table and the -0.0495 correlation), which was produced during review from the same split-sample data and is reported from that run rather than from a script shipped in this bundle.

The pattern worth naming. Every serious error in this document was of one kind: **a true measurement attached to the wrong claim.** The 0.963 was real and described something else. The rank-4,608 number was real and was called an impossibility instead of a cost. The $W \rightarrow \sqrt{c} \cdot W$ slopes agreed to 1.4e-06 and measured nothing. **Sections 8.1 to 8.3 are about that pattern, and Section 4.1 turned out to be its largest instance.** If there is one transferable result in this write-up, it is that a correct number is not a correct claim, and that the gap between them is not visible to whoever produced the number.