Consistent harmonic metrics under a compute budget

Phase 2 algorithmic-contribution write-up | Submission #337582

Syeda Anika Jerin | AIcrowd: anika_jerin | 25 October 2026

ARC White-Box Estimation Challenge 2026, organized by the Alignment Research Center in partnership with AIcrowd.

Abstract

This report describes the estimator associated with successfully graded Phase 2 submission **#337582**. It extends a public factored third-cumulant engine with a consistent diagonal input metric in related fourth-order closure terms, a larger oldest-source basis, and owned-view reuse. The aim is to improve expected-activation estimates without making the additional state too expensive to carry.

On a paired panel of eight public local networks, the final combination lowers mean final-layer MSE by **3.31%** and mean adjusted error by **2.45%** relative to the preceding local release, with **0.89% more charged FLOPs**. All eight raw errors and seven adjusted errors improve; all eight resource checks pass. The recorded public grader result is **3.084e-9 adjusted error, 2.066e-8 final-layer MSE, and 50/50 public networks completed with zero failures**.

Contribution and scope

The contribution is an incremental integration and measurement study: align the metric used by several harmonic slots, spend a small amount of extra capacity on older sources, and verify whether the resulting accuracy gain pays for its cost. The underlying factored cumulant method, harmonic regeneration, skew-direction idea and recursive Winograd products are inherited or informed by public work. This report does not claim their invention or a new convergence theorem.

A longer-history experiment provides a useful counterexample: its mean raw error falls by 4.10%, but extra computation makes mean ungated adjusted error 0.51% worse and only four of eight runtime checks pass. More accurate state is beneficial only when its downstream value exceeds its transport cost.

Binding to the graded entry

This write-up is bound exclusively to #337582. The local V265 source is associated with that entry through the saved release history and upload timeline: it replaced estimator.py at 16:56 UTC on 21 October, before the 17:39 UTC upload. The submission page did not expose an uploaded-source hash, so remote byte-for-byte identity has not been independently established. Exact local hashes are supplied on page 7.

The public receipt and the local eight-network panel are different evaluations. No private score, final rank, or guaranteed generalization improvement is claimed.

1. Estimation problem and inherited engine

What is carried through the network, and why it costs computation

For standard-normal input x, the task is to estimate each post-ReLU expectation E[h_l,i(x)] from the network weights. The challenge shape is width 1,024 and depth 16. The evaluator meters analytical work rather than rewarding faster hardware. The score used in these experiments is:

```
adjusted = final_layer_MSE * max(0.1, FLOPs / 2**41)
```

The local subprocess checks used a 120 s wall limit, 400 ms residual-time limit, 5 s setup limit and an 8,192 MB worker configuration. These are the checks used for the reported local evidence; they are not inferred from execution time alone.

Inherited representation

The starting engine propagates means and covariance, carries a factored third-cumulant history, and regenerates approximate fourth-order harmonic contributions. Recent third-order sources remain in an exact factored representation, while older sources enter two shared compressed tiers. Weighted contractions of the retained factors supply the slices needed by later nonlinear updates.

The selected V265 release keeps age thresholds 4 and 8, a first older-source rank of 384, and an oldest-source rank of 224. V189 used 192 in the oldest tier. These are engine age thresholds, not a claim that the complete historical state is exact for that many additional layers.

Why higher-order state matters

A Gaussian approximation summarizes a layer by its first two moments. ReLU repeatedly changes the distribution, so omitted higher-order information can affect subsequent mean and covariance updates. Factored sources retain some of this information without materializing a dense third-order tensor. Compression and approximate fourth-order feedback introduce further bias; simply increasing every rank would also increase transport and contraction cost.

Public foundations and inspiration

The base lineage is 504aldo's factored K=3 engine [1] and LordProcciuto's derivative engine [2], both MIT-licensed. The harmonic/AUGMENT framework comes from ARC's cumulant-propagation work [3]. Public discussion of per-neuron metric variants and early harmonic strength [4] motivated the inexpensive metric ablation. The existing skew direction is related to public work on the asymmetric fourth-order slice [5]. The bias-versus-variance perspective in [6] motivated testing changes across complete networks rather than judging only a local slice.

The specific work here is adapting these ideas consistently within this implementation, selecting a resource-valid combination, and documenting the measured error/cost trade-off. The final four-day optimization window built on an existing implementation and earlier project work; it was not a from-scratch invention of the base method.

2. Diagonal-metric consistency

A repeated-index identity check, followed by empirical closure evaluation

Let C denote the current covariance approximation, G a symmetric harmonic core, and M a diagonal metric. Define the symmetrized fourth-order surrogate as the average of the six placements of G and M across four indices:

```
H_ijkl = (G_ij M_kl + G_ik M_jl + G_il M_jk
          + G_jk M_il + G_jl M_ik + G_kl M_ij) / 6
```

For M = diag(m), its repeated-index slices satisfy:

```
H_iiii = G_ii * m_i
H_iijj = (G_ii * m_j + G_jj * m_i) / 6   (i != j)
H_ijjj = G_ij * m_j / 2                  (i != j)
```

The engine consumes the last matrix in a **column-cubic convention**: K(i,j,j,j). This orientation matters for asymmetric terms. A dense six-term tensor calculation on a small random instance independently checks all three identities; it does not merely compare two copies of the production expression.

Change in V265

The baseline used a scalar metric of 2. V265 computes m_i = sum_j W_ij^2 from an already-present elementwise squared-weight array, with the existing METRIC_C/2 normalization. At the selected METRIC_C=2, that normalization is one. W here is the effective row-oriented weight array used by the engine at the current step, including its active-set handling; this is not a claim to recover a full latent-space metric.

```
m       = sum(W * W, axis=1)
K4diag  = dG * m
K22_ij  = (dG_i * m_j + m_i * dG_j) / 6
K13_ij  = C_off_ij * (m_j * lambda / 2)
y_birth = w2 * m / 4
```

The off-diagonal K13 expression still uses the inherited radial approximation G_ij approximately lambda*C_off_ij; the existing skew-direction contribution is added separately. The birth expression is the affected factor in the existing source feed, not a complete new formula for every fourth-to-third-order term.

What the identity does and does not establish

The algebra shows how the selected harmonic surrogate's slots must transform under a diagonal metric. It does **not** prove that this surrogate equals the true network fourth cumulant, that the inherited off-diagonal core is optimal, or that a physically closer local cumulant always lowers final MSE. Consistency is the implementation hypothesis; full-network measurement determines whether it is useful.

The final combination also multiplies the first four harmonic-strength entries by 1.15 and uses skew gain 1.0. These are empirical settings, not derived universal constants. No new large matrix multiplication is introduced by the diagonal-metric calculation.

3. Capacity, runtime and paired results

The final combination is useful; individual effects are not isolated

The oldest basis is increased from 192 to 224 while preserving the 384-direction first older tier and the existing exact-history schedule. This targets information already entering compression rather than extending the more expensive exact young-source transport.

Runtime changes reuse owned scratch-buffer views, add fast paths for cached quadrants and slots, and cache leaf-output quadrants. Ownership references and family invalidation prevent cached views from silently referring to an old buffer after growth. The cache repair was checked for output parity before interpreting its timing results.

Paired eight-network evidence

Release	Mean final MSE	Mean adjusted	Mean FLOPs	Passes
V189	2.255682e-8	3.340907e-9	3.253024e11	8/8
V260	2.221455e-8	3.290628e-9	3.253496e11	8/8
V265	2.180916e-8	3.259215e-9	3.282108e11	8/8

V260 combines the diagonal metric and early harmonic-strength change at the preceding ranks and skew setting. V265 adds the oldest-rank/skew choices and cache refinements. This is a staged comparison, not a factorial ablation: the table cannot assign an independent gain to the metric, lambda, rank, skew or cache change.

Relative to V189, V265 improves mean MSE by 3.3146% and mean adjusted error by 2.4452%, while mean charged work rises by 0.8941%. Raw MSE improves on 8/8 networks; adjusted error improves on 7/8. The adjusted mean is computed from the per-network scores, not from the product of mean MSE and mean utilization.

Per-network adjusted error (x 1e-9)

Local index	V189	V265	Resource pass
92	3.63127	3.40974	yes
93	3.61462	3.56984	yes
94	2.60735	2.54622	yes
95	4.06791	3.81792	yes
96	3.74754	3.76029	yes
97	3.00616	2.99720	yes
98	2.82202	2.79865	yes
99	3.23037	3.17386	yes

Maximum V265 residual time is 396.84 ms, leaving only 3.16 ms against the local cap. Nominal preallocated scratch capacity is 5.0198 GiB. The worker was configured for 8,192 MB; the harness RSS field measures the parent process and is not evidence of worker peak memory.

Recorded public receipt for #337582: adjusted 3.084e-9, final-layer MSE 2.066e-8, all-layer mean MSE 8.02e-9, and 50/50 public cases completed with zero failures. These public figures are not the eight-network local averages and are not a private-evaluation result.

4. What the error-cost trade-off revealed

Why a lower MSE was not enough to justify a replacement

For a single network operating above the 0.1 utilization floor, an MSE reduction r and cost increase c change the score by the factor (1-r)(1+c). The modification pays only if:

```
c < r / (1 - r)
```

This inequality is a per-network diagnostic. With different changes across networks, the aggregate decision must use the actual mean of their adjusted errors. It also says nothing about hard resource failures: a candidate that exceeds a limit is not acceptable merely because its ungated error looks attractive.

Longer exact history: an informative rejected experiment

V239 delayed first-stage compression by one layer from evaluation index 9 onward, keeping ranks 384/192. Its complete eight-network panel gave mean raw MSE 2.163242e-8, a 4.098% reduction against V189, but mean charged work increased by 4.826%. Mean ungated adjusted error rose to 3.357985e-9, 0.511% worse. Only four of eight residual-time checks passed.

The practical interpretation is limited but clear: on this measured panel, the additional exact history recovered information, yet its transport cost and runtime prevented a usable adjusted-score improvement. It does not prove that every adaptive-history policy fails.

Cheap consistency versus costly extra state

V265 instead spends a small amount on an already-available metric and an intermediate oldest rank. The combination passes the complete local panel with lower mean MSE and adjusted error. This supports that particular allocation of accuracy budget in this engine; it does not establish the metric variant as a universally better closure.

Other checks that narrowed the search

Increasing the skew gain to 3.0 worsened both development pilots in the tested configuration. A wider first-four-layer harmonic-strength sweep did not produce a resource-valid replacement. Narrowing the longer-history first tier to 288 lost too much accuracy. Batched block-copy/addition experiments preserved pilot outputs, but counted reshapes and runtime overhead prevented a usable gain.

An attempted warm start of a nonfixed basis update reproduced the control because the active fixed-basis mode bypassed that code path. Inspection identified the inactive configuration, and the experiment was stopped. This was a research-configuration correction, not a defect repaired in the selected submitted estimator.

Uncertainty and selection effects

The local eight-network panel was consulted repeatedly during development. It is not a pristine final holdout and no confidence interval or significance claim is attached to its final improvement. The public aggregate is a separate observation, but iterative public feedback can also influence selection. Fresh private seeds are the relevant remaining test.

5. Verification, reproducibility and disclosure

Evidence sufficient for the stated claims; limits kept explicit

Numerical and implementation verification

The diagonal metric identities were checked against an independently assembled symmetric fourth-order tensor. Five growing/shrinking buffer cases checked cached multiplication and summed contractions against the preceding implementation, including bitwise parity, dense references and ownership inventory. Three deep recursive rectangular contractions were checked against dense products. Six generic small/random estimator cases tested the estimator contract beyond the competition shape.

WhestBench estimator validation, packaging and package validation passed for the frozen original V265 source. The clean archive contained only estimator.py and manifest.json. The released comment-cleaned copies preserve the original Python AST and also pass contract validation. This is not a claim that every future package/runtime combination has been benchmarked.

Local environment and evidence files

The paired local runs used Python 3.12, NumPy 2.4.6, FlopScope 0.12.3 and WhestBench 0.16.2, sequentially through the subprocess runner. The public configuration page recorded Python 3.10.21, NumPy 2.2.6, FlopScope client/server 0.12.3 and WhestBench 0.16.1. Local and grader environments are therefore not represented as identical.

```
research/v189_eight_confirmation.json
research/v260_eight_confirmation.json
research/v265_eight_confirmation.json
research/v265_comparison.json
research/v239_eight_confirmation.json
research/test_v256_metric.py
research/test_v180_cache.py
research/test_v188_deep_hub.py
research/benchmark_public_seeds.py
```

The public local truth is from the mini_truth.npz corpus in the referenced fourth-cumulant project [7]. Local indices 92 through 99 identify the paired cases. The release includes a compact validation-summary.json and source-manifest.json; original logs and scripts are retained locally. The corpus is not bundled in the PDF, and the short contract-validation command is not a full benchmark reproduction.

LLM assistance and participant role

OpenAI Codex generated and modified substantial portions of the project-specific code and experiment scripts, assisted with derivations, executed local checks through tools, and drafted this report under participant direction. The participant specified the optimization objective, supplied public grading receipts, and selected the two entries for private evaluation. The reference implementation also has its own disclosed LLM-assisted provenance. No claim of unaided authorship is made.

Explanations of performance are empirical interpretations unless explicitly identified as algebraic identities. Numerical tests validate selected identities and implementation behavior; they are not formal proofs of the full estimator or of the physical correctness of its closure.

Conclusion

The measured result is a modest but resource-valid improvement from consistent metric handling and a small capacity change within a known factored engine. The report contributes a traceable implementation and error-cost comparison, not a claim to have solved full fourth-order propagation within the challenge budget.

Appendix. Source identity and references

Exactly one graded entry is the subject of this report: #337582

Original local source

```
research/v265_submission/estimator.py
SHA256:
73c4da9d820d323e5bd90f6acc2876eaab
2b69e68bf4d62026f46d78ae1fa5d3
```

The two displayed hash lines concatenate to one 64-character SHA256. The active original aicrowd_submission/estimator.py has the same hash. The original validated archive is submission-v265-8net-validated.tar.gz.

Comment-cleaned publication copy

```
submissions/v265/estimator.py
SHA256:
a5de7a2ee846022503e100219213ab19c0
c279b1a5bdc4e712e6a283434f6103
```

The differing hash is expected after removing non-license comments. Parsed code is unchanged. The graded upload itself has not been independently downloaded and hashed; the association limitation on page 1 remains applicable. Public-source licenses and authorship notices are retained.

References

[1] 504aldo. Factored K=3 cumulant-propagation engine and findings log. MIT. Local reference commit 1a4083f89184d6b1cc55d2e55a5a561d2ae50a8d. discourse.aicrowd.com/t/18218

[2] LordProcciuto. Public derivative engine, MIT. github.com/siming767-creator/whestbench-lordprocciuto

[3] Wu, Lecomte, Winer, Robinson, Hilton and Christiano. Estimating the expected output of wide random MLPs more efficiently than sampling. arxiv.org/abs/2605.05179. Reference code: alignment-research-center/mlp_cumulant_propagation.

[4] il.miscusi. Carried fourth-order core and scale-mode extraction; related metric/early-strength discussion. discourse.aicrowd.com/t/18248. Scale-mode extraction is not a component claimed for V265.

[5] kai_leal_gonzalez. Two-term fourth-order (3,1) slice study. discourse.aicrowd.com/t/18246.

[6] Bias-variance error budget for factored K=3 cumulant propagation. discourse.aicrowd.com/t/18261.

[7] logxio. Fourth-cumulant project, public data and research code, MIT. github.com/logxio/fourth-cumulant.

[8] AIcrowd. Submission/write-up mechanics and LLM transparency. discourse.aicrowd.com/t/18041. Deadline update: discourse.aicrowd.com/t/18251. Challenge: AIcrowd ARC White-Box Estimation Challenge 2026.