THE COMPRESSION SQUARE: WEIGHT QUANTIZATION AND KV-CACHE EVICTION DO NOT COMPOSE AT THE OPERATING POINT

Anonymous authors
Paper under double-blind review

ABSTRACT

A deployed long-context language model is compressed twice: its weights are post-training quantized, and its KV cache is pruned by an eviction policy that ranks tokens with the model's own attention. Each axis is evaluated alone and the two losses are assumed to add, which misses that the scores deciding what to evict come from the quantized model. We measure the coupling with the *compression square*, a factorial over model precision and selection source whose counterfactual transplant cell splits the interaction Δ_{int} exactly into an oracle term (the quantized model kept the wrong tokens for itself) and a fragility term (it is more hurt by the same eviction). Across 499k released per-item records on Qwen2.5-7B, Llama-3.1-8B and Qwen2.5-3B at 4k–32k, the deployed cell is significantly worse than additive in 71 of 143 identifiable SnapKV cells (63 after Benjamini–Hochberg) and sub-additive in 3; on the needle tasks the effect is a tail: most items are unaffected and a few are badly hurt. Its size is set by the quantizer (two noisy RTN builds exceed the three shipped W4-g128 recipes in 32 of 33 matched conditions), but the shipped recipes are not exempt on real documents: on Llama-3.1-8B LongBench, GPTQ W4-g128 is super-additive in 9 of 9 cells over three eviction policies at 288 items (5 at 120), and on Qwen2.5-7B LongBench, where eviction alone costs 0.7% of the baseline loss and RTN W4-g128 +0.07 nats, the two together cost +0.49 [+0.14, +0.92] more than their sum. An intervention supports the channel: i.i.d. noise injected into the ranked scores alone, with the model untouched, opens it at a level within a factor of two of a quantizer's own score error. At these budgets, single-axis evaluation does not predict the deployed system. All code and the 499,289 per-item records will be made public upon acceptance at `https://github.com/[anonymized]/[anonymized]`.

1 INTRODUCTION

Serving a long-context language model means compressing it along two axes. Post-training quantization (PTQ) shrinks the weights to 3–4 bits (Frantar et al., 2023; Lin et al., 2024); KV-cache eviction discards most of the cached keys and values, keeping a few hundred tokens chosen by a policy such as SnapKV or H_2O (Li et al., 2024; Zhang et al., 2023). The two literatures are almost disjoint: a quantizer is validated with a full cache, an eviction policy on an unquantized model, and the deployed system's accuracy is inferred by assuming the two losses add.

This paper is about why that assumption fails. Quantization replaces θ by $\hat{\theta}$, so it enters the cached keys and values and the forward computation, as expected. But a query-aware eviction policy ranks tokens with attention scores the model itself computes, $g = G(\theta, x)$, and in the deployed stack those come from $\hat{\theta}$. Quantization also enters the *oracle* that decides what to throw away (the composition $\pi \circ G$ of §3), and no published evaluation reports an interaction statistic for it.

We measure it with a design we call the **compression square** (§3, Fig. 1). Its four physical cells (baseline, quantize only, evict only, both) define the interaction $\Delta_{\text{int}} = \mathcal{L}_{11} - \mathcal{L}_{10} - \mathcal{L}_{01} + \mathcal{L}_{00}$; two further cells can only be built in a laboratory: the quantized model on the tokens the *clean* model would have kept, and the converse. The first transplant splits Δ_{int} exactly, with no modelling assumption, into an *oracle* term (the quantized model kept different, and on this loss worse, tokens

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

than the clean model would have, the clean selection being a reference rather than an optimum) and a *fragility* term (it is more hurt by the same eviction). Policies that never consult the model (StreamingLLM, random) have oracle term zero by construction, so their interaction is pure fragility; only the transplant attributes Δ_{int}.

What we find. On Qwen2.5-7B (4k–32k) and Llama-3.1-8B (4k–16k), with a smaller Qwen2.5-3B check, three real quantization algorithms, and budgets calibrated to the eviction policy's failure curve, the joint cell is significantly worse than additive in 71 of 143 identifiable SnapKV cells and sub-additive in 3 (63 and 3 after Benjamini–Hochberg, Benjamini & Hochberg, 1995), but the sign is set by the quantizer, not by joint compression: where the clean policy still works, the two noisy RTN builds are super-additive in 34 of 44 cells while the three shipped W4-g128 recipes (GPTQ, AWQ, RTN) give 5 of 31 (1 significantly negative; cell-record counts), a difference of magnitude and not of presence (§5.1). Over the cells with $\Delta_{\text{int}} > 0.05$ the oracle term carries 41% of the summed interaction, though not in every cell, and is near zero on a language-modelling control which eviction hurts *more* per answer token than it hurts retrieval (§5.2). It is absent at the eviction literature's generous budget fractions and appears as the budget approaches the policy's failure point (§6).

Mechanism, and what we could not confirm. A boundary-margin bound (Prop. 2) on the *expected* attention mass displaced by score noise σ yields three predictions we treat as falsifiers: the channel opens with σ, measurable on sixteen short prompts (C1 holds, though the screen orders the quantizers at one of five identical-bit budgets and not the others, §5.5); the damage is positional (C2 holds for the noisiest quantizer only); and the ordering between policies follows their scoring window (C3 holds on Qwen2.5-7B but not on Llama-3.1-8B and is unsupported, §5.3).

Contributions. (i) A factorial design for composed compression whose transplant cell splits the interaction exactly into an oracle and a fragility term, the latter a cross-curvature of the loss (Prop. 1). (ii) The first measurement, to our knowledge, of how weight quantization interacts with named token-eviction policies: super-additive at the operating point in most identifiable cells and sub-additive in 3 of 143, retrieval-specific in its oracle component per answer token (null in summed nats), and present on real documents. (iii) A boundary-margin heuristic and a calibration screen, suggestive across the ladder and on real documents (App. G.6) but recovering the full ordering at one of five equal-bit budgets and, as a ranking, redundant with the perplexity gap (§5.5). (iv) Negative results against claims stated in advance: protecting the score-generating projections does no better than a same-cost placebo at 3 bits and worse at 4 (Table 17, App. G.11), and the predicted ordering between policies does not survive a second model family (§5.3).

2 Related work

Weight quantization. Post-training quantization to 4 bits is routine, through second-order error compensation (Frantar et al., 2023), activation-aware scaling (Lin et al., 2024) or outlier-removing rotations (Ashkboos et al., 2024; Liu et al., 2025), the difficulty traced to activation outliers (Dettmers et al., 2022; Xiao et al., 2023; Sun et al., 2024); evaluation is single-axis, with a full cache. Quantized models can match aggregate accuracy while flipping many answers (Dutta et al., 2024), the damage grows with input length (Mekala et al., 2025), and standard metrics under-report it (Jaiswal et al., 2024), but none reports an interaction statistic for a quantized model that also evicts its cache.

KV-cache compression. Token eviction keeps a budget of positions chosen by a score: accumulated attention (Zhang et al., 2023; Ge et al., 2024), attention from a recent window (Li et al., 2024), the same with layer- or head-adaptive budgets (Cai et al., 2024; Feng et al., 2025), or position alone with sinks (Xiao et al., 2024); cache quantization (Liu et al., 2024b; Hooper et al., 2024) is the other route. Retrieval heads (Wu et al., 2025) and attention sinks (Gu et al., 2025) explain why a few cached tokens matter most.

Composed compression. Harma et al. (2025) prove weight sparsity and weight quantization non-orthogonal, with a threshold readable as a sign test on a 2×2 interaction; Chhawri et al. (2025) show the order of pruning, distillation and quantization matters; SparseGPT (Frantar & Alistarh, 2023) applies two jointly in one solver; all are weight-only. KV Pareto (Gokhale et al., 2025) co-tunes AWQ weights with cache *quantization* without an interaction statistic; QET (Wang et al., 2024) quantizes weights, keys and values together but evaluates each separately; RDKV and QEvict (Zhang et al., 2026; Garg et al., 2026) combine cache quantization with eviction, without weight quantization. The

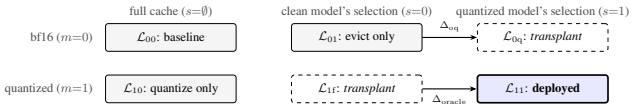

Figure 1: **The compression square.** Rows: model precision m; columns: the source of the eviction policy's scores, s ($\emptyset$: no eviction). Solid cells are physically realisable, dashed cells transplant a saved selection across models, the deployed system is bottom-right, and Δ_{int} contrasts the four solid cells. For StreamingLLM and random eviction the two right-hand columns coincide, so $\Delta_{\text{oracle}} \equiv \Delta_{\text{oq}} \equiv 0$.

closest ancestor of our mechanism is Q-Hitter (Zhang et al., 2024), which shows *cache* quantization reclassifies more than half of H_2O's heavy hitters. We study the weight pathway, perturbing the function that generates the scores rather than the values scored, and attribute downstream loss to the corrupted selection rather than stopping at set overlap, which we report only as a diagnostic (App. C).

3 The compression square

3.1 Setup and the two exact decompositions

A model with weights θ is prefilled on a prompt x of length L, producing a KV cache $C(\theta, x)$; a *cache-selection operator* π with budget B maps per-head scores $g = G(\theta, x) \in \mathbb{R}^L$ to a retain set S, $|S| = B$, and the pruned cache decodes the answer, $\hat{y} = f(\theta, C(\theta, x)|_S)$ with $S = \pi(G(\theta, x))$.

Let $\mathcal{L}$ be a per-item loss, the *tail metric*: the teacher-forced NLL of the gold answer tokens after the first (which is predicted from the prompt-final logits before any eviction, App. E), summed over them, so a "nat" is per answer, not per token (answer lengths differ by task, App. F). We index it by model precision $m \in \{0, 1\}$ (bf16, quantized) and by where the retain set comes from: the full cache ($s = \emptyset$), the bf16 model's own selection ($s = 0$) or the quantized model's ($s = 1$). Four cells are physically realisable and we write them $\mathcal{L}_{00}$, $\mathcal{L}_{10}$, $\mathcal{L}_{01}$ and the deployed $\mathcal{L}_{11}$, the first digit being m and the second whether the cache was evicted by that model's own selection. Two are counterfactual transplants, named by a letter for the *other* model's selection: $\mathcal{L}_{1f}$, the quantized model given the clean model's (*f*ull-precision) retain set, and $\mathcal{L}_{0q}$, the clean model given the *q*uantized model's (Fig. 1; the subscripts are names). Write $\Delta_{\text{quant}} = \mathcal{L}_{10} - \mathcal{L}_{00}$ and $\Delta_{\text{evict}} = \mathcal{L}_{01} - \mathcal{L}_{00}$ for the single-axis costs.

Definition 1 (Compression interaction)**.** $\Delta_{\text{int}} := \mathcal{L}_{11} - \mathcal{L}_{10} - \mathcal{L}_{01} + \mathcal{L}_{00}$*;* $\Delta_{\text{int}} > 0$ *means damage compounds,* $\Delta_{\text{int}} < 0$ *that it saturates.*

Definition 2 (Oracle/fragility split; oracle-quality gap)**.** Δ_{int} *telescopes exactly through any intermediate quantity; because* $\mathcal{L}_{1f}$ *is measured rather than assumed, this split is estimable:*

$$\Delta_{\text{int}} = \underbrace{\mathcal{L}_{11} - \mathcal{L}_{1f}}_{\Delta_{\text{oracle}}} + \underbrace{\mathcal{L}_{1f} - \mathcal{L}_{10} - \mathcal{L}_{01} + \mathcal{L}_{00}}_{\Delta_{\text{frag}}}, \qquad \Delta_{\text{oq}} := \mathcal{L}_{0q} - \mathcal{L}_{01}. \tag{1}$$

Δ_{oracle} is what the quantized model pays for keeping the tokens it chose rather than the clean model's; Δ_{frag} is the extra cost it pays for the *same* eviction; $\Delta_{\text{oq}} > 0$ says its retain set is worse even for a clean model; across the 83 identifiable SnapKV cells that carry the second transplant it is null in aggregate (median +0.007, $p = 0.155$ signed rank) yet ranks with Δ_{oracle} at Spearman +0.70 (App. G.11): where the oracle term is large the retain set is worse for the clean model too. No assumption enters the split in (1). With the retain set as mediator, Δ_{oracle} is the natural indirect effect of causal mediation analysis (Pearl, 2001; Imai et al., 2010) and Δ_{frag} the natural direct effect $\mathcal{L}_{1f} - \mathcal{L}_{01}$ net of Δ_{quant}, with the cross-world counterfactual $\mathcal{L}_{1f}$ run rather than assumed.

Proposition 1 (Cross-curvature plus selection shift)**.** *Fix any non-empty retain set* S *of prompt positions, selected once after prefill. Parameterise weight noise by* ϵ, $\theta_\epsilon = \theta + \epsilon\Delta\theta$ *with* $\Delta\theta = \hat{\theta} - \theta$*, and cache ablation by* η *through the relaxed mask* $w_j(\eta) = 1 - \eta\mathbb{1}[j \notin S]$ *multiplying the unnormalised attention weight of* prompt *position* j *in every head,* for the answer-token queries only*; the prompt's own forward pass is unmasked, so the cached keys and values are those of the full prefill, and answer tokens attend to one another unmasked (*$\eta = 1$ *is then exactly the pruned cache, positions preserved). If* $\mathcal{L}_S(\epsilon, \eta)$ *is* C^2 *on* $[0, 1]^2$*, then* $\mathcal{L}_S(1,1) - \mathcal{L}_S(1,0) - \mathcal{L}_S(0,1) + \mathcal{L}_S(0,0) =$

$\int_0^1\int_0^1 \partial^2\mathcal{L}_S/\partial\epsilon\,\partial\eta\,d\epsilon\,d\eta$. *With S the clean model's selection the left-hand side is Δ_{frag}, so Δ_{int} is this cross-curvature plus Δ_{oracle}; for a model-independent policy $\Delta_{\text{oracle}} = 0$.*

The proof (App. A) is elementary, and the proposition's use is interpretive. With the retain set fixed, "do compression errors compound?" asks about the mixed second derivative of the loss in weights and cache, not whether error norms add; the oracle term is what the selection's own weight-dependence adds.

Remark 1 (Algebraic nulls)**.** *For a policy whose selection does not depend on the model (StreamingLLM keeps sinks and a recency window; random eviction draws positions from a fixed seed), S is identical under θ and $\hat{\theta}$, so $\Delta_{\text{oracle}} \equiv \Delta_{\text{oq}} \equiv 0$ by construction and $\Delta_{\text{int}} = \Delta_{\text{frag}}$ exactly. Conversely Δ_{oracle} compares the* same *quantized model under two selections at the same budget, so "quantized models are simply more cache-sensitive" cannot produce it (a selection shift is necessary; cache-sensitivity only amplifies it); that confound lives in Δ_{frag}.*

3.2 Mechanism: boundary margins against score noise

Fix a layer, a kv-head and a scoring query i, with attention logits $a_j = \langle q_i, k_j\rangle/\sqrt{d}$. Quantizing W_Q, W_K and the hidden states feeding them perturbs the logits, $\tilde{a}_j = a_j + \varepsilon_j$, $\varepsilon_j = \langle \Delta q_i, k_j\rangle/\sqrt{d} + \langle q_i, \Delta k_j\rangle/\sqrt{d} + O(\Delta^2)$.

Assumption 1. *The ε_j are zero-mean and sub-Gaussian with variance proxy σ^2 (a per-head quantity). Independence across j is assumed only where stated: rounding in W_K, contracted with distinct x_j, gives near-independent Δk_j, but every policy here ranks a* pooled *score, which makes neighbours dependent by construction (App. G.2).*

Let $S^* = \text{top}_B(a)$, $p = \text{softmax}(a)$, $P^* = \sum_{j\in S^*} p_j$, and for $j \in S^*$ the *boundary margin* $\mu_j = a_j - a_{(B+1)} \geq 0$, the gap to the $(B{+}1)$-th largest logit.

Proposition 2 (Displacement bound)**.** *Let $\hat{S} = \text{top}_B(\tilde{a})$ and $D := \sum_{j\in S^*\setminus\hat{S}} p_j$ be the displaced mass. Under Assumption 1 with $\varepsilon_j \perp \varepsilon_k$, $\mathbb{E}[D] \leq \sum_{j\in S^*} p_j \sum_{k\notin S^*} \exp\big(-(a_j - a_k)^2/(4\sigma^2)\big)$; without independence the same argument gives $8\sigma^2$ in place of $4\sigma^2$.*

Proofs are in App. A, with a corollary bounding the head output by $2D\max_j \|v_j\|/(P^* - D)$ when $D < P^*$, which makes D, not set overlap, the mediator the theory nominates; it bounds the head, not the loss, and the $D \to \Delta_{\text{oracle}}$ link holds at 0.5B but is within noise at 7B (App. C). Since $a_j - a_k \geq \mu_j$ for $k \notin S^*$, the inner sum is dominated by excluded tokens near the boundary, and the single-competitor quantity $\hat{D} := \sum_{j\in S^*} p_j \exp(-\mu_j^2/(4\sigma^2))$ is the predictor we test, as a rank predictor rather than a bound. Summed over all of S^* (Fig. 4) it counts the pinned recency window, which C2 says cannot move; restricted to the *contestable* positions it ranks D better (Spearman +0.95 against +0.89; App. C). Three testable predictions follow, which we treat as falsifiers (§1). **(C1) Noise scaling.** The bound on $\mathbb{E}[D]$ increases with σ, which a few short prompts measure, and we test whether D and Δ_{oracle} do; the exponential dependence predicts a threshold, not a line. **(C2) Margin dependence.** D depends on the margin distribution, not on competence: near-ties at the boundary (distractor needles) are fragile at the same σ, and tokens kept for structural reasons (a recency window) are immune. **(C3) Averaging protects, but only against query-side noise.** Split $\sigma^2 = \sigma_q^2 + \sigma_k^2$ into the query-side (Δq_i) and key-side (Δk_j) parts of ε_j, taken independent; averaging n_{eff} independent queries divides the first but not the second, so the exponent becomes $-\mu^2/(4(\sigma_q^2/n_{\text{eff}} + \sigma_k^2))$ and *saturates* at a level set by σ_k. Policies differing in that window, and in pooling and forced-cache size (§5.3), test where the noise lives; the direct path can also be attacked by keeping W_K at 16 bit (§5.5, Table 17).

3.3 Metric and identifiability

Accuracy interactions are confounded by floors and ceilings: if either single-axis compression already drives accuracy to zero, the joint cell cannot be worse, and Δ_{int} computed on exact-match *error* is forced non-positive whatever the cross-curvature (given that quantizing alone does not raise accuracy). Teacher-forced NLL is unbounded above, so it has no ceiling artefact (it still saturates at the guessing entropy once retrieval fails, which the band below excludes) and is the primary metric. The same logic

dictates an *identifiability band*: a cell is interpretable only if the quantizer still solves the task with a full cache (exact match ≥ 0.15) and the clean policy still works at that budget (bf16 evict-only exact match ≥ 0.5), since otherwise the transplanted clean selection is not a gold standard. Both thresholds were fixed before interaction numbers were read; a policy's *operating budget* is the smallest budget inside its band. Both gates condition on components of Δ_{int} ($-\mathcal{L}_{10}$ and $-\mathcal{L}_{01}$), so a cell admitted because a realised single-axis loss came out small would be selected towards a positive interaction. Gating the clean policy on one half of each cell's items and estimating Δ_{int} on the other moves the median by at most 0.027 nats (within-cell difference 0.0000 in median, $p = 0.999$; App. G.11). On LongBench, where exact match is uninformative, the band is a loss ratio instead (App. B).

4 Experimental setup

Models. Qwen2.5-7B-Instruct (Qwen Team, 2024) is the primary model, with Llama-3.1-8B-Instruct (Grattafiori et al., 2024) and Qwen2.5-3B-Instruct as cross-family and cross-scale checks; all use grouped-query attention (Ainslie et al., 2023) and selections are per kv-head; a CPU-scale study on Qwen2.5-0.5B and 1.5B preceded the GPU experiments (App. D).

Quantizers. We use real post-training algorithms rather than synthetic noise: GPTQ (Frantar et al., 2023) (Hessians from 32 sequences of 2048 WikiText-2 tokens), AWQ (Lin et al., 2024), and round-to-nearest, at 3 to 6 bits with group sizes from 16 to per-channel, plus the W_K/W_V-protected variants of Table 17. Bits per parameter count a 16-bit scale and a b-bit zero per group. All are emitted as dequantised bf16, so one harness evaluates every model (perplexities in App. E).

Eviction policies. Eviction is a single selection after prefill, rotary positions preserved, followed by teacher-forced decoding of the answer through the pruned cache. That is faithful to SnapKV and PyramidKV and a post-prefill variant of TOVA, $\mathrm{H_2O}$ and StreamingLLM, which evict per decoding step; only $\mathrm{H_2O}$ is renamed, because only there does the variant change the *score* rather than when it is applied. ***SnapKV*** (Li et al., 2024) scores each position by the attention from the last 32 prompt queries, max-pooled with kernel 7, and keeps that recency window; ***TOVA*** (Oren et al., 2024) is the same with a window of one and no pooling; ***PyramidKV*** (Cai et al., 2024) uses SnapKV's score but spreads the same total budget across layers in a decreasing arithmetic sequence; $\mathrm{H_2O}$-***norm*** accumulates attention over all queries, normalised by the number that could attend (a post-prefill variant of $\mathrm{H_2O}$ (Zhang et al., 2023), whose per-step unnormalised form we did not run); ***StreamingLLM*** (Xiao et al., 2024) keeps four sinks and a recency window; *random* keeps the recency window plus seeded random positions. B counts every retained position, forced ones included (SnapKV's, PyramidKV's and random's 32-token window, TOVA's last token, $\mathrm{H_2O}$-norm's recency half, StreamingLLM's four sinks). Every scored selection is saved, so the transplant cells re-use one model's selection on the other; for StreamingLLM and random the two coincide (Remark 1).

Tasks. Two synthetic tasks follow RULER (Hsieh et al., 2024); the positional analysis follows Liu et al. (2024a). *Single needle*: one "special magic number" sentence at a depth drawn uniformly in $[0.1, 0.9]$ of WikiText-2 prose (Merity et al., 2017) (also the calibration corpus; App. G.7 controls that), queried by its key. *Distractor*: four key–number needles at independent depths, one queried, creating near-ties at the eviction boundary (C2). A *plain language-modelling* control separates "retrieval is fragile" from "everything degrades"; a variable-tracking task was run but is outside the band for every model (Table 18). Three English QA subsets of LongBench (Bai et al., 2024), namely multi-field QA (English), HotpotQA and 2WikiMQA, supply real documents (§5.4). Contexts are ≈3.8k, ≈15.0k and, on Qwen, ≈32k tokens for the synthetic tasks, 5.0–16.9k for LongBench; items are seeded and identical across every cell that shares a (task, context).

Budgets, calibrated to the policy's failure curve. A first 7B grid used the literature's fractional budgets (50% to 6.25%) and was saturated: SnapKV moved the bf16 single-needle loss only from 0.003 to 0.103 nats; that grid (12,851 records) is kept as a scope statement (§6). A calibration sweep (App. B) put SnapKV's transition on the 4k single needle between 64 and 96 tokens, so the runs use absolute budgets $\{64, 96, 160, 256\}$ at 4k and $\{96, 160, 256, 512\}$ at 16k.

Design and statistics. The 7B 4k main grid evaluates bf16 and six quantizers $\times$ 4 tasks (the two needles, variable tracking, language modelling) $\times$ 4 policies (SnapKV, $\mathrm{H_2O}$-norm, StreamingLLM, random) $\times$ 4 budgets, with every selection source of Fig. 1, on 64 items; the policy, ladder, LongBench, long-context (16k on both families, 32k on Qwen) and cross-family grids are described where they

are used. In total 499,289 per-item records were evaluated; a *cell* is one (model, context, task, quantizer, policy, budget) condition averaged over items. Every term is a mean over items of a per-item contrast, with a paired item-level bootstrap of 4,000 resamples and a percentile 95% interval (Efron & Tibshirani, 1994; Kübler et al., 2026); "significant" means the interval excludes zero, and the bootstrap p behind the multiplicity corrections is two-sided, twice the smaller tail mass at zero, floored at one over the resample count plus one (App. G.11). Exact match and the loss are computed over the answer tokens after the first, which is decided before any eviction (§3, App. E); on LongBench, whose answers are often one token, every answer token is scored through the pruned cache (§5.4).

5 RESULTS

5.1 THE LOSSES DO NOT ADD, AND THE SIZE IS THE QUANTIZER'S

Table 1: **Every base quantizer at its operating budget**, in three single-needle settings (all base quantizers, not a selection). SnapKV at B, the smallest identifiable budget for that (model, context, task); */†: 95% paired-bootstrap interval excludes zero from above/below; 8 of 13 rows positive, 0 negative. Columns: the quantizer's ΔNLL on the tail metric and on the *whole* answer ("full"), eviction's own ΔNLL and clean exact match. 5 of the 10 Qwen rows fail the full-string check of Table 24 (not evaluable on Llama), and eviction alone costs 0.4–1.2 nats in every row, so no row is free-alone (below). Table 27 lists all 101 cells.

			quantizer alone				eviction alone	
Quantizer	bits	B	ΔNLL	full	Δ_{int} [95% CI]	Δ_{ora}	ΔNLL	EM
Qwen2.5-7B, 4k, single needle								
GPTQ W4-g128	4.16	96	-0.003	+0.02	+0.57 [+0.03, +1.14]*	+0.43	+0.41	0.89
AWQ W4-g128	4.16	96	-0.004	+0.26	-0.14 [-0.58, +0.20]	-0.24	+0.41	0.89
GPTQ W3-g128	3.15	96	+0.014	-0.27	-0.19 [-0.59, +0.14]	-0.16	+0.41	0.89
RTN W4-g128	4.16	96	-0.003	-0.15	-0.16 [-0.61, +0.21]	0.00	+0.41	0.89
RTN W4-per-ch.	4.00	96	+0.002	+0.63	+0.93 [+0.27, +1.66]*	+0.54	+0.41	0.89
RTN W3-g128	3.15	96	+0.001	+0.49	+2.00 [+1.25, +2.76]*	+1.39	+0.41	0.89
Qwen2.5-7B, 16k, single needle								
GPTQ W4-g128	4.16	96	0.000	+0.03	+1.11 [-0.05, +2.37]	+0.17	+1.19	0.72
GPTQ W3-g128	3.15	96	0.000	-0.10	+1.67 [+0.01, +3.55]*	+0.80	+1.19	0.72
RTN W4-per-ch.	4.00	96	+0.014	+0.52	+2.29 [+0.86, +3.86]*	+1.11	+1.19	0.72
RTN W3-g128	3.15	96	+0.001	+0.12	+4.30 [+2.58, +6.14]*	+1.28	+1.19	0.72
Llama-3.1-8B, 4k, single needle								
GPTQ W4-g128	4.16	64	-0.009	+0.31	+0.09 [-0.22, +0.44]	+0.27	+0.63	0.79
RTN W4-per-ch.	4.00	64	+0.146	+1.03	+2.97 [+2.28, +3.72]*	+2.27	+0.63	0.79
RTN W3-g128	3.15	64	-0.010	+1.97	+1.80 [+1.28, +2.36]*	+1.54	+0.63	0.79

The effect is not universal: at 7B/4k 3 of 6 quantizers open the channel (Table 1). Inside the band of §3.3 SnapKV yields 101 cells over four (model, context) settings and twelve quantizer configurations (six base builds plus, at Qwen 4k, the W_K/W_V-protected and group-64 variants), Δ_{int} positive in 49 and negative in 1, median +0.167 nats.

Multiplicity and grain. With the Llama 16k and Qwen 32k grids this is 143 cells, 71 positive and 3 negative; Benjamini–Hochberg (Benjamini & Hochberg, 1995) at 5% leaves 63 and 3, Bonferroni 40 and 0 (Table 25). It is a boundary phenomenon: median Δ_{int} is +0.78 nats at the smallest in-band budget of its own (model, context, task, quantizer) unit against +0.21 one budget up, 67% of the summed positive interaction at the first. Budgets within a unit share items and weights, so we also read one test per unit at that budget, fixed in advance: 24 positive and 2 negative of 46; at the next two ranks it is 22 of 45 and 20 of 39. **Tail structure.** Δ_{int} is a difference of *mean* losses, the deployment-relevant quantity, but it is a mean of an unbounded per-item contrast on which most items are unaffected: the three largest in absolute value carry a median 0.49 of the summed absolute contrast. Rebuilt from the released records on each cell's full item set (larger than the cell file's for 20 cells, hence the different counts; App. G.11), all 143 give 75/1 by the mean, 82/1 by a 10%-trimmed mean and 106/0 by a paired signed-rank test. The mean is a poor *detector* of a broad, small shift (median +0.0088 nats per item against +0.191 by the mean), most where the effect is smallest: over the shipped W4-g128 cells whose clean policy still works it is 7 of 31 by the mean (5 on the cell records) and 13 by signed rank, against 34 and 41 of 44 for the noisy builds. The negatives are read by the same rule: 1 cell is negative under the mean *and* the trimmed estimator, and the ladder's

two sub-additive W4-g128 cells are tail-driven like the positives (App. G.11); the mean remains the estimand for both signs.

Magnitude by quantizer. Matched within (model, context, task, budget), the two noisy RTN builds exceed the three shipped W4-g128 recipes in 32 of 33 conditions, medians +0.410 against +0.005 nats ($p = 5 \times 10^{-10}$, signed rank), and the split is stable as the clean-policy threshold rises (App. G.11) and is not a calibration-corpus artefact (App. G.7). That is C1's prediction, and it separates the quantizers by magnitude, not presence, since the shipped recipes are not exempt (§5.4). The ladder's 48 in-band cells give 8 positive and 7 negative and agree in sign with the 256-item power run in 11 of 12 shared cells (App. G.11, G.3). Power adds detections at 4k and on Llama and leaves 16k flat: over the 30 identifiable 4k cells the positives go 10 at $n = 64$ to 18 on the 128–176 held-out items and 16 pooled at 256, on Llama-3.1-8B 11 of 18 to 13, and at 16k 10 of 18 to 9, none with a negative cell at any n (App. G.3, G.4).

When both compressions are free. Requiring, per cell, that the quantizer be non-inferior to bf16 on full-string exact match (margin 0.05) at $|\Delta\mathrm{NLL}| \leq 0.05$, and that eviction alone cost at most 0.05 nats at clean exact match ≥ 0.95, leaves 14 cells, Δ_{int} positive in 6 and negative in 0. This is the weakest result here: the gate pins both single-axis costs, so wherever $\mathcal{L}_{11} \geq \mathcal{L}_{00}$ it bounds Δ_{int} below at -0.048 nats and a negative cell was nearly unavailable, and at four times the items 5 of 6 stay significant but only 1 passes the gate (App. G.6).

At four and eight times 4k the union adds 16 of 24 positive cells at Llama 16k (2 negative) and 6 of 18 at Qwen 32k (App. G.11).

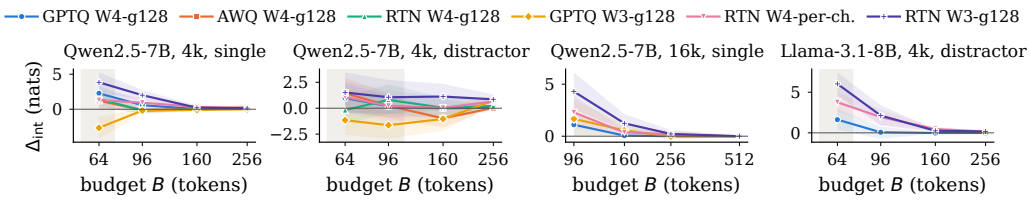

Figure 2: **Interaction against cache budget**, SnapKV, four (model, context, task) panels with their own y-scales; bands are 95% paired-bootstrap intervals; grey marks budgets where the clean policy has failed (bf16 evict-only exact match < 0.5), where Δ_{int} is not identifiable; a series missing from a panel was not run in that setting. The Llama single-needle block of Table 1 is in Table 27.

Fig. 2 shows the decay directly; below the failure point (grey) Δ_{int} takes either sign.

5.2 The oracle channel: decomposition, algebraic nulls and specificity

The split (1) attributes the interaction (Fig. 6): over the 70 identifiable SnapKV cells with $\Delta_{\mathrm{int}} > 0.05$ nats the oracle term carries 41% of the summed interaction (34% over all 101 without the cut), and both terms are often significant alone (Δ_{oracle} significantly positive in 32 of 101 and negative in 4, Δ_{frag} 39 and 1), so the quantized model keeps worse tokens *and* is more hurt by losing them. The first reading is forced (both arms of Δ_{oracle} share weights and items; only the selection moves it); the second is not: in Δ_{frag} a more fragile model and one whose full-cache damage the tail metric did not count look the same, and scoring the whole answer cannot separate them, since the first token is decided before eviction and cancels in Δ_{int} exactly (App. G.1). The null policies ($\Delta_{\mathrm{int}} \equiv \Delta_{\mathrm{frag}}$) never retain the needle, so all 216 of their cells lie outside the band.

A plain language-modelling control runs the same quantizers, policy and budgets; its answer is a median 90 tokens against the needle's 8, so only *per-token* losses are comparable. Eviction alone costs the control *more* than retrieval (median 0.205 nats per token against 0.041), while its oracle term is -0.0006 against +0.0177 on the needles ($p = 0.012$, signed rank; 0.043 on the 18 in-band pairs); the ratio form agrees ($p = 0.013$), but in summed nats, the paper's primary scale, it is null ($p = 0.86$), so specificity depends on the normalisation (Table 18).

5.3 IT IS NOT ONE POLICY, BUT THE ORDERING BETWEEN POLICIES IS MODEL-DEPENDENT

The result so far rests on SnapKV. TOVA, which C3 makes the most corruptible (though it also drops pooling), and PyramidKV ran with it in one grid on Qwen2.5-7B at 4k (six quantizers, $B \in \{64, 96, 160, 256\}$, 64 items) and on Llama-3.1-8B (three quantizers, 48 items; Table 26); all vary the scoring window, not the family of score, and the structurally different family never enters the band (§6).

Scope across policies. Each enters the band at its own budget (TOVA at $B = 64$, SnapKV at 96, PyramidKV at 160) and shows the effect inside it: on Qwen, 15 of 36 TOVA cells, 9 of 30 SnapKV and 4 of 12 PyramidKV are positive, 1 negative, PyramidKV the weak case (median -0.005 nats). On Llama, 66 of 72 are identifiable: 17/24, 13/21 and 16/21 (3 negative).

Cross-family ordering. On Qwen it holds: at matched task, quantizer and budget, TOVA's Δ_{oracle} exceeds SnapKV's in 20 of 30 cells (Wilcoxon $p = 0.001$); on Llama it does not (9 of 21, $p = 0.585$), and PyramidKV reverses, falling below SnapKV on Qwen (5 of 12) and rising above it in 21 of 21 Llama cells. The one controlled test of C3, sweeping SnapKV's window with everything else fixed on Qwen2.5-0.5B, is non-monotone inside the band (App. D). C3 is unsupported: the bound gets right that the channel opens wherever the policy consults the model, not which policy suffers more.

Table 2: **Five equal-cost configurations, three algorithms each** (Qwen2.5-7B, 4k, single needle, SnapKV at $B = 96$, 128 items). σ is σ_{cal}; within a row the bit cost is identical. */†: the oracle term's interval excludes zero from above/below. Full ladder: Table 15.

		RTN		GPTQ		AWQ	
Config	bits	σ	Δ_{oracle}	σ	Δ_{oracle}	σ	Δ_{oracle}
W3-g128	3.15	0.331	+1.02*	0.253	+0.06	0.307	+0.15
W3-g64	3.30	0.303	+0.35*	0.220	-0.02	0.274	-0.17
W4-per-ch.	4.00	0.309	+0.42*	0.227	+0.20	0.298	+0.55*
W4-g128	4.16	0.170	-0.16	0.126	+0.34*	0.159	-0.29†
W4-g64	4.31	0.151	-0.01	0.112	-0.06	0.133	-0.10

Table 3: **Protection versus placebo versus more bits** (Qwen2.5-7B, 4k, single needle, SnapKV at $B = 96$, 64 items). W_V@16 (placebo) costs the same as W_K@16 without entering its layer's scores; RTN W3-g64 (uniform) spends the bits everywhere. Only the 3-bit block has a significant unprotected baseline; both blocks, with Δ_{int}, are in Table 17.

Scheme	bits	Δ_{oracle} [95% CI]	EM
RTN W3 g128	3.15	+1.39 [+0.64, +2.17]	0.55
+ W_K@16	3.25	+0.69 [+0.03, +1.40]	0.78
+ W_V@16^p	3.25	+0.84 [+0.42, +1.29]	0.67
RTN W3 g64^u	3.30	+0.59 [+0.23, +1.01]	0.72

5.4 REAL DOCUMENTS

Because the needle is a sentence we inserted, we also run three English QA subsets of LongBench (Bai et al., 2024) verbatim on all five policies (120 items). The loss is the deployment-faithful one (§4); identifiability is a loss ratio, each compression within 1.5× the full-cache loss, fixed before any interaction was computed though after the bf16 curves were read (App. B). Three things carry over. First, which policies work: SnapKV, TOVA and PyramidKV enter the band, H_2O-norm and StreamingLLM never (28 of 80 cells: four quantizers, four budgets, five policies; Table 14). Second, super-additivity: Δ_{int} is significantly positive in 12 of 12 TOVA cells, 6 of 12 SnapKV and 2 of 4 PyramidKV, none significantly negative; at this grid's budgets eviction is never free, but at $B \in \{2048, 3072\}$ SnapKV costs 0.7–1.4% of the baseline and RTN W4-g128 is then free alone (+0.07 nats) yet +0.49 [+0.14, +0.92] costly together, a tail with 2 of 12 trimmed intervals excluding zero (App. G.9). Bands from 1.3× to 2× do not change it; at 1.2× only 3 cells remain (App. G.11). Third, coverage: the grid's two W4-g128 builds (GPTQ and RTN) are super-additive under TOVA in every cell, +0.37–+0.68 nats after a 10%-trimmed estimate (6 of 6 excluding zero) against a 9.66-nat full-cache loss, though under SnapKV they do not survive trimming (0 of 6). On the second family the effect is broad: the same design on Llama-3.1-8B (three policies, three quantizers, three budgets) is super-additive in 23 of 27 cells with none negative, a broad shift rather than a tail. At 288 items the *shipped* GPTQ W4-g128 is significant in 9 of 9 cells by the mean and 8 by a 10%-trimmed mean, 8 of them out of sample, so that grid's null under trimming at 120 items was a power limit (App. G.8).

5.5 A CALIBRATION SCREEN: SCORE NOISE AND WHICH QUANTIZERS OPEN THE CHANNEL

C1 says the oracle term should grow with the score noise σ; the σ we test is the cheap one, σ_{cal}: the across-position standard deviation of the quantized minus clean log-scores (App. C) on 16 generic 256-token prompts, ranking them as the evaluation-time σ does (+0.99). Over 24 configurations spanning 0.054–0.331 at 128 items (Table 15) it ranks each rung's mean oracle term over its two budgets at Spearman +0.56 ($p = 0.005$; +0.55 against the $B = 96$ column of Table 15; bit-width alone does as well, -0.54), with 5 of the 12 rungs above the median σ significantly positive and 1 negative, against 1 and 1 below it. Averaged over the 12 (context, task, budget) strata of the six main-grid quantizers the same correlation is +0.11: higher up there is no oracle term left to rank.

Is it σ, or bit-width? A ladder built by varying group size cannot say, because the two are collinear at -0.91 (App. G.11). The GPTQ and AWQ rungs break that by changing σ at *identical* cost (Table 2), and within a *bit* budget σ still ranks the oracle term, though not decisively (+0.51 over 15 budget-centred values, permutation $p = 0.102$, 10 of 15 concordant pairs). It dissolves where σ does not separate the algorithms, including at 4-bit group-128. As a *ranking* it is redundant: over our 9 builds σ_{cal} orders the quantizers exactly as the WikiText-2 perplexity gap does ($\rho = +1.00$; Table 5), so a practitioner who has run perplexity already has the ordering. An intervention establishes that corrupting the ranking alone suffices: i.i.d. noise injected into the ranked scores of an unquantized model raises Δ_{oq} with σ in all 6 curves (Spearman $\rho \geq +0.96$; 52 of 66 conditions significant), and the level reproducing a real quantizer's Δ_{oq} (median $\sigma = 0.024$) is within a factor of two of that quantizer's own measured error on the same array (median sd 0.040, range 0.026–0.073); we attribute the residual to the error's autocorrelation, which the probe does not manipulate (App. G.2).
Two predictions were stated in advance. Keeping W_K at 16 bit should remove the direct path of the score noise; it removes some (paired contrast +0.70 [+0.09, +1.35] at 3 bits, 64 items), but a placebo spending the same bits on W_V, which enters no score, is indistinguishable (-0.15 [-0.74, +0.43]), as is spending them uniformly, and at 4 bits, where the unprotected baseline is itself not significant, protection raises the oracle term while the placebo removes it (contrast +0.92 [+0.43, +1.48]; Table 17). The reduction comes from the extra bits rather than from the score path, and what dominates is the *indirect* route through the already-perturbed hidden states. And C2's positional prediction holds only for the noisiest quantizer: transplanting the clean selection removes over half of RTN W3-g128's early-context tilt and much less of RTN W4 per-channel's (App. C).

6 DISCUSSION, LIMITATIONS AND CONCLUSION

Scope. The effect lives at the operating point, and its absence away from it is measured: at the eviction literature's fractional budgets a strong model tolerates joint compression, and at 80 items the intervals are tight enough to say so, with $\Delta_{\mathrm{int}} = +0.001$ [-0.007, +0.008] at 12.5% of 4k and 0.000 at 25% (RTN W4-g128, SnapKV), against +0.02 [-0.17, +0.26] at 6.25%, where eviction first costs the clean model anything (+0.10 nats; App. B). The real-document cells of §5.4 share the limit (eviction alone costs +0.95–+4.14 of 9.66 nats on Qwen, +0.38–+2.54 of 5.18 on Llama), and their free-alone case rests on the large-budget run (App. G.9). For practitioners it is a tail on the needle tasks (most items are unaffected and a few are badly hurt) and a broad shift on Llama LongBench (App. G.8); either way a benchmark mean understates it. Its size scales with the quantizer's score noise, which the perplexity gap already ranks, but not to zero for the recipes that ship. The remedy scales the same way: a median $1.91\times$ the clean model's cache budget for RTN W4 per-channel and $1.23\times$ for GPTQ W4-g128 (App. G.11, with the budget-shift reading).

Limitations. The three in-band policies are settings of one scoring rule; accumulated attention enters at no budget we ran, and per-step eviction ($\mathrm{H_2O}$ and TOVA as published) and Ada-KV (Feng et al., 2025) we did not implement. The free-alone certification survives four times the items in one cell of six (App. G.6); the noise probe is i.i.d. where the quantizer's error is autocorrelated; the $D \to \Delta_{\mathrm{oracle}}$ link is within noise at 7B; and C3 is unsupported. Query-aware *sparse attention* is the natural next target. Weights are fake-quantized, so the numbers reflect arithmetic not a kernel; cache quantization is a third axis we did not compose; and the band restricts the claims to where both compressions are acceptable alone.

Conclusion. At the operating point weight quantization and KV-cache eviction do not compose, because the eviction policy's scores come from the quantized model. The interaction grows with score noise, the shipped 4-bit recipes are not exempt, and single-axis evaluation does not predict it.

REPRODUCIBILITY STATEMENT

Every reported measurement of the compression square (every cell estimate, interval, count and correlation in the main text and appendix) is generated from the released per-item records by a script that writes the paper's macros and tables, and no such value is transcribed by hand. A small number of quantities that are not measurements of the square are quoted from the implementation checks and the run logs instead, and are identified as such where they appear: the tolerances of the correctness tests and the wall-clock cost of the runs. The supplementary material contains the evaluation harness (selection operators, transplant logic, teacher-forced scoring), the layer-sequential GPTQ and AWQ implementations with their calibration-data recipe and the data-free per-matrix RTN implementation (both calibrated builds are our own re-implementations; App. E names their deviations from the reference code), the task generators with the hashed-seed scheme that makes items identical across cells, the analysis and bootstrap code, the mechanism diagnostics, and all 499,289 evaluated records with their per-item losses. App. A gives the proofs; App. E lists the implementation checks (bit-identical in-hook scores, next-token agreement of truncated logits, GPTQ-beats-RTN, selection overlap above 0.995 under a bf16 score matmul, and the transplant path itself, which a separate GPU-only script checks rather than the test suites), the tokenization and seeding conventions, the compute budget, and the three group-size configurations that the analysis scripts exclude because a fallback left them unquantized; App. B reports the budget calibration and the identifiability bands; App. F reproduces the prompt templates. Every table and macro regenerates from the committed records on a CPU; re-running the experiments themselves took some forty GPU-hours on one 24 GB GPU (App. E), needs the public checkpoints (Llama-3.1-8B-Instruct behind its licence acceptance) and requires TF32 off for the GPTQ Hessians; quantized weights are regenerated by the released scripts.

ETHICS STATEMENT

This work studies the evaluation of compressed language models on synthetic retrieval tasks built from WikiText-2 (Merity et al., 2017) and on three English QA subsets of LongBench (Bai et al., 2024), both public research corpora used under their stated licences; it involves no human subjects and no personal data, and we release no new model. Its main practical implication is cautionary: composed compression can cost far more than the sum of its parts on a tail of retrieval items without showing up in single-axis evaluation, which is relevant to anyone deploying quantized long-context models in settings where omissions matter.

AI USE STATEMENT

In this work, we used generative AI tools to assist in writing, editing, and debugging code, including code used to implement our methods and experiments. We did not use generative AI tools for any of the other tasks that require disclosure: in particular, we did not use them to generate synthetic data, to develop theoretical models or conceptual frameworks, to formulate mathematical claims or assist in proofs, to propose or refine hypotheses, to design the research methodology or experiments, to assist with translation, to clean or reformat datasets, or to support data analysis or the interpretation of results. Additionally, we used generative AI tools to edit the manuscript to improve readability and clarity, and to obtain simulated peer-review-style feedback on drafts, which informed revisions to the writing and presentation. All AI-assisted code was reviewed and tested for correctness by the authors, and all AI-suggested edits and feedback were reviewed and adopted at the authors' discretion; the scientific content, claims, and results are entirely the authors' own. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. QuaRot: Outlier-free 4-bit inference in rotated LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300, 1995.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Yucheng Li, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Junjie Hu, and Wen Xiao. PyramidKV: Dynamic KV cache compression based on pyramidal information funneling. *arXiv preprint arXiv:2406.02069*, 2024.

Shivansh Chhawri, Rahul Mahadik, and Suparna Rooj. A systematic study of compression ordering for large language models. *arXiv preprint arXiv:2511.19495*, 2025.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Abhinav Dutta, Sanjeev Krishnan, Nipun Kwatra, and Ramachandran Ramjee. Accuracy is not all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall/CRC, 1994.

Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S. Kevin Zhou. Ada-KV: Optimizing KV cache eviction by adaptive budget allocation for efficient LLM inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Elias Frantar and Dan Alistarh. SparseGPT: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning (ICML)*, 2023.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *International Conference on Learning Representations (ICLR)*, 2023.

Ayushman Garg, Akshita Gupta, Shaswata Bhattacharya, Abhishek Gupta, Sandeep Kumar, and Manoj Kumar. QEvict: Recoverable quantized KV eviction for attention-drift-robust long-context decoding. *arXiv preprint arXiv:2608.05326*, 2026.

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive KV cache compression for LLMs. In *International Conference on Learning Representations (ICLR)*, 2024.

Sai Gokhale, Devleena Das, Rajeev Patwari, Ashish Sirasao, and Elliott Delaye. KV Pareto: Systems-level optimization of KV cache and model compression for long context inference. *arXiv preprint arXiv:2512.01953*, 2025.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. When attention sink emerges in language models: An empirical view. In *International Conference on Learning Representations (ICLR)*, 2025.

Simla Burcu Harma, Ayan Chakraborty, Elizaveta Kostenok, Danila Mishin, Dongho Ha, Babak Falsafi, Martin Jaggi, Ming Liu, Yunho Oh, Suvinay Subramanian, and Amir Yazdanbakhsh. Effective interplay between sparsity and quantization: From theory to practice. In *International Conference on Learning Representations (ICLR)*, 2025.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. KVQuant: Towards 10 million context length LLM inference with KV cache quantization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *Conference on Language Modeling (COLM)*, 2024.

Kosuke Imai, Luke Keele, and Dustin Tingley. A general approach to causal mediation analysis. *Psychological Methods*, 15(4):309–334, 2010.

Ajay Jaiswal, Zhe Gan, Xianzhi Du, Bowen Zhang, Zhangyang Wang, and Yinfei Yang. Compressing LLMs: The truth is rarely pure and never simple. In *International Conference on Learning Representations (ICLR)*, 2024.

Jonas Kübler, Kailash Budhathoki, Matthäus Kleindessner, Xiong Zhou, Junming Yin, Ashish Khetan, and George Karypis. When LLMs get significantly worse: A statistical approach to detect model degradations. In *International Conference on Learning Representations (ICLR)*, 2026.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Proceedings of Machine Learning and Systems (MLSys)*, 2024.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024a.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. SpinQuant: LLM quantization with learned rotations. In *International Conference on Learning Representations (ICLR)*, 2025.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. In *International Conference on Machine Learning (ICML)*, 2024b.

Anmol Mekala, Anirudh Atmakuru, Yixiao Song, Marzena Karpinska, and Mohit Iyyer. Does quantization affect models' performance on long-context tasks? In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations (ICLR)*, 2017.

Matanel Oren, Michael Hassid, Nir Yarden, Yossi Adi, and Roy Schwartz. Transformers are multi-state RNNs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Judea Pearl. Direct and indirect effects. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 411–420, 2001.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Mingjie Sun, Xinlei Chen, J. Zico Kolter, and Zhuang Liu. Massive activations in large language models. In *Conference on Language Modeling (COLM)*, 2024.

Yanshu Wang, Wang Li, Zhaoqian Yao, and Tong Yang. QET: Enhancing quantized LLM parameters and KV cache compression through element substitution and residual clustering. *arXiv preprint arXiv:2407.03637*, 2024.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. In *International Conference on Learning Representations (ICLR)*, 2025.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning (ICML)*, 2023.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations (ICLR)*, 2024.

Junkai Zhang, Hang Guo, Luca Benini, and Yawei Li. RDKV: Rate-distortion bit allocation for joint eviction and quantization of the KV cache. *arXiv preprint arXiv:2605.08317*, 2026.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H_2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Zhenyu Zhang, Shiwei Liu, Runjin Chen, Bhavya Kailkhura, Beidi Chen, and Zhangyang Wang. Q-Hitter: A better token oracle for efficient LLM inference via sparse-quantized KV cache. In *Proceedings of Machine Learning and Systems (MLSys)*, 2024.

A PROOFS

Proposition 1. Fix S and write $\mathcal{L}_S(\epsilon, \eta)$ for the relaxed loss; the two arguments are the weight-noise and mask parameters of the proposition, not the digit pairs that name the cells in §3. By the fundamental theorem of calculus in ϵ for fixed η, $\int_0^1 \partial_\epsilon \partial_\eta \mathcal{L}_S(\epsilon, \eta)\, d\epsilon = \partial_\eta \mathcal{L}_S(1, \eta) - \partial_\eta \mathcal{L}_S(0, \eta)$, (with $\partial^2 \mathcal{L}_S / \partial\epsilon\, \partial\eta$ read as $\partial_\epsilon \partial_\eta \mathcal{L}_S$; C^2 makes the order immaterial). Integrating in η, $\int_0^1 \int_0^1 \partial_\epsilon \partial_\eta \mathcal{L}_S \, d\epsilon \, d\eta = [\mathcal{L}_S(1,1) - \mathcal{L}_S(1,0)] - [\mathcal{L}_S(0,1) - \mathcal{L}_S(0,0)]$. At $\eta = 0$ every mask entry is one and the cache is full, so $\mathcal{L}_S(\epsilon, 0)$ is the full-cache loss of θ_ϵ; at $\eta = 1$ the mask zeroes the unnormalised attention weights of every position outside S, which is the renormalised attention over S and hence the pruned cache with positions preserved. Thus $\mathcal{L}_S(0,0) = \mathcal{L}_{00}$, $\mathcal{L}_S(1,0) = \mathcal{L}_{10}$, and $\mathcal{L}_S(0,1)$, $\mathcal{L}_S(1,1)$ are the clean and quantized models evicting to S (the mask acts on prompt positions only, so answer tokens attend to one another exactly as in the pruned-cache run). With $S = S(\theta)$, the clean model's selection, $\mathcal{L}_S(0,1) = \mathcal{L}_{01}$ and $\mathcal{L}_S(1,1) = \mathcal{L}_{1\mathrm{f}}$, so the left-hand side is $\mathcal{L}_{1\mathrm{f}} - \mathcal{L}_{10} - \mathcal{L}_{01} + \mathcal{L}_{00} = \Delta_{\mathrm{frag}}$. Adding $\Delta_{\mathrm{oracle}} = \mathcal{L}_{11} - \mathcal{L}_{1\mathrm{f}}$ gives Δ_{int}. For a policy whose selection does not depend on θ (deterministic given the prompt, or sharing its seed), $S(\hat{\theta}) = S(\theta)$ and $\mathcal{L}_{11} = \mathcal{L}_{1\mathrm{f}}$ are the same run, so $\Delta_{\mathrm{oracle}} = 0$. □

The relaxed mask is a device for stating the identity; nothing in the experiments interpolates. The C^2 requirement holds for the teacher-forced NLL of the transformers evaluated here (SiLU activations, RMSNorm) as a function of their weights and of a multiplicative mask on unnormalised attention weights, both of which enter through smooth operations; the renormalised softmax is defined for any non-empty S, since $\sum_{j \in S} \exp(a_j) > 0$. Prop. 1 covers a single selection made after prefill, which is how every policy here is run (§4); a per-step eviction policy would need a query-indexed mask.

Proposition 2. Let $j \in S^*$. If $j \notin \hat{S}$ then at least B tokens have perturbed logit at least $\tilde{a}_j$; since $|S^* \setminus \{j\}| = B - 1$, at least one of them lies outside S^*. Hence $\{j \notin \hat{S}\} \subseteq \bigcup_{k \notin S^*} \{\tilde{a}_k \geq \tilde{a}_j\} = \bigcup_{k \notin S^*} \{\varepsilon_j - \varepsilon_k \leq -(a_j - a_k)\}$. Under Assumption 1, $\varepsilon_j - \varepsilon_k$ is zero-mean sub-Gaussian with variance proxy $2\sigma^2$, so $\Pr[\varepsilon_j - \varepsilon_k \leq -t] \leq \exp(-t^2/(4\sigma^2))$ for $t \geq 0$, and $a_j - a_k \geq \mu_j \geq 0$ for $k \notin S^*$. A union bound gives $\Pr[j \notin \hat{S}] \leq \sum_{k \notin S^*} \exp(-(a_j - a_k)^2/(4\sigma^2))$. Finally $\mathbb{E}[D] = \sum_{j \in S^*} p_j \Pr[j \notin \hat{S}]$ by linearity. □

Only pairwise independence of ε_j and ε_k is used, to make their difference sub-Gaussian with proxy $2\sigma^2$ (for jointly Gaussian errors a positive correlation ρ shrinks that proxy to $2\sigma^2(1-\rho)$ and tightens the bound; in general dependence costs the factor of two in the statement). The proof needs only the weak inequality $\tilde{a}_k \geq \tilde{a}_j$, so ties are covered under any tie-breaking rule, and p_j is deterministic because p is the softmax of the clean logits. Assumption 1's zero mean can be weakened to a zero-mean *difference*, which is what App. C's across-position standard deviation estimates. The single-competitor predictor $\hat{D}$ replaces the inner sum by its largest term, $k = (B+1)$, and is what App. C computes; it is a rank predictor, not a calibrated bound.

Corollary 1 (Head-output bound; stated in §3.2)**.** *In the setting of Prop. 2 (S^* the top-B of the same logits whose softmax p weights the head), with $P^* = \sum_{j \in S^*} p_j$, Attn_S the head output with softmax renormalised over S, and $D < P^*$, $\|\mathrm{Attn}_{S^*} - \mathrm{Attn}_{\hat{S}}\| \leq 2D \max_j \|v_j\| / (P^* - D)$.*

Proof of Cor. 1. Write $P_S = \sum_{j \in S} p_j$ and $\mathrm{Attn}_S = P_S^{-1} \sum_{j \in S} p_j v_j$. Let $C = S^* \cap \hat{S}$, $U = S^* \setminus \hat{S}$ (mass D) and $V = \hat{S} \setminus S^*$ (mass D'). Because $|V| = |U|$ and every $k \in V$ lies outside the true top-B, $p_k \leq p_{(B+1)} \leq p_j$ for every $j \in U$, so $D' \leq D$; also $P_{\hat{S}} = P^* - D + D' \geq P^* - D$. With $M = \max_j \|v_j\|$,

$$\begin{aligned}\left\|\mathrm{Attn}_{S^*} - \mathrm{Attn}_{\hat{S}}\right\| &\leq \left\| \sum_{j \in C} p_j v_j \right\| \left| \tfrac{1}{P^*} - \tfrac{1}{P_{\hat{S}}} \right| + \tfrac{1}{P^*} \left\| \sum_{j \in U} p_j v_j \right\| + \tfrac{1}{P_{\hat{S}}} \left\| \sum_{k \in V} p_k v_k \right\| \\ &\leq \tfrac{P_C\,(D - D')}{P^* P_{\hat{S}}} M + \tfrac{D}{P^*} M + \tfrac{D'}{P_{\hat{S}}} M \leq \tfrac{(D-D') + D'}{P_{\hat{S}}} M + \tfrac{D}{P^*} M \leq \tfrac{2DM}{P^* - D},\end{aligned}$$

using $P_C = P^* - D \leq P^*$, $0 \leq D - D' \leq D$ and $P^* \geq P_{\hat{S}} \geq P^* - D > 0$. The hypothesis $D < P^*$ excludes the disjoint case $S^* \cap \hat{S} = \emptyset$, where the bound is undefined; the ranking hypothesis is what gives $D' \leq D$, and it is why the corollary bounds an idealised single-query head and not the pooled-score policies of §4. □

B BUDGET CALIBRATION AND THE SATURATED REGIME

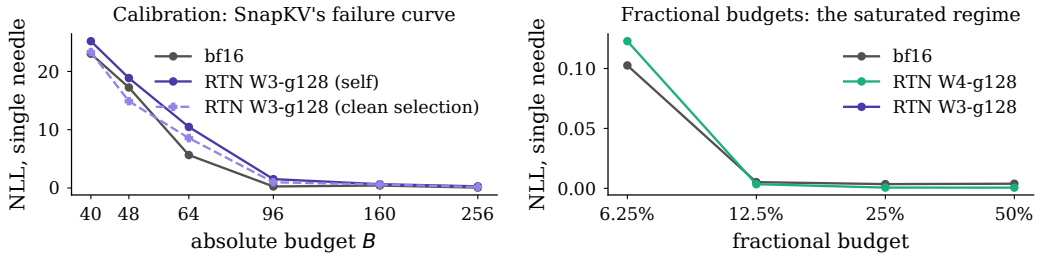

Figure 3: **Left:** the calibration sweep (Qwen2.5-7B, 4k, single needle, 12 items) that located SnapKV's failure curve and fixed the absolute budgets used in every formal run; the oracle term is already visible at 64 tokens as the gap between the deployed RTN W3 system and the same model given the clean selection—a budget outside the identifiability band of §3.3, so suggestive only. **Right:** the first main grid at fractional budgets, on which SnapKV costs the bf16 model at most 0.103 nats down to 6.25% of the context, leaving little for quantization to interact with.

The first 7B grid (12,851 records) used the fractional budgets that the 0.5B study and most of the eviction literature use. On a 7B model at 4k the single-needle loss under bf16 SnapKV was 0.003 nats with a full cache, 0.004 at 50% and 0.103 at 6.25%, with exact match 1.00, 1.00 and 0.96. At 6.25% the RTN W4-g128 cell gives $\Delta_{\text{int}} = +0.02$ [-0.17, +0.26] on 80 items with joint exact match 0.96; the 12.5% and 25% cells quoted in §6 are from the same grid at 80 items. A 12-item sweep over absolute budgets (Fig. 3, left) then located the transition: bf16 NLL 23.04, 17.25, 5.65, 0.26, 0.42 and 0.06 at 40, 48, 64, 96, 160 and 256 tokens, with exact match 0.00, 0.00, 0.08, 0.92, 0.83 and 0.92. That sweep is 12 items on the *single-needle* task alone (at that n the curve is not monotone between 96 and 160 tokens), and the two tasks do not have the same transition. Measured afterwards on the main grid, the clean policy's exact match at 4k is single $B = 64$: 0.12; single $B = 96$: 0.89; single $B = 160$: 0.95; single $B = 256$: 0.97; distractor $B = 64$: 0.00; distractor $B = 96$: 0.12; distractor $B = 160$: 0.64; distractor $B = 256$: 0.98—so the single needle first enters the band at $B = 96$, which the grid contains and which is therefore right at its knee, while the distractor task first enters at $B = 160$, which is one rung past its own knee because the grid has no budget between 96 and 160. App. G.10 runs that missing rung and finds it below the band on that task (clean exact match 0.36), so the band's exclusion of it is not hiding an identifiable cell. Only single-needle at $B \in \{96, 160, 256\}$ and distractor at $B \in \{160, 256\}$ are identifiable at 4k as a result. The consequence for the pooled counts is that "distance to the failure point" is not matched across the two tasks at the same nominal B, which is one reason §5.1 also reads one test per (model, context, task, quantizer) unit at that unit's own smallest in-band budget. A pilot protocol should therefore check that the *compressed* condition is inside the measurable band, not only that the uncompressed task is solvable.

On LongBench, where free-form answers make exact match useless as a band, identifiability is instead a loss ratio: with a bf16 model the best loss any budget buys must stay within $1.5\times$ the full-cache loss, for the quantizer and for the clean policy alike. Each policy's ratio is taken against the full-cache loss on its own item set: 1.23 for TOVA, 1.10 for SnapKV, 1.33 for PyramidKV, 1.715 for H_2O-norm and 2.24 for StreamingLLM. The threshold was set after these bf16 curves were read and before any interaction was computed, so the choice was blind to the interaction but not to which policies it admits: it keeps SnapKV, TOVA and PyramidKV and excludes H_2O-norm and StreamingLLM (random was not run on LongBench), and any threshold in $[1.326, 1.715)$ would admit the same set.

C MECHANISM DIAGNOSTICS

Fig. 4 shows the two mechanism relations at 7B and 0.5B. The depth-shape test (Fig. 5, middle) compares a configuration's per-depth mean NLL profile with the bf16 full-cache profile under the null that they differ only by a constant: the statistic is the variance across depth bins of the per-bin mean of the paired per-item differences (which removes any constant shift), and its null distribution is obtained by permuting the depth-bin labels of those per-item differences (20,000 permutations). It is an omnibus test of non-flatness; the direction is read from the bin means, which for the deployed

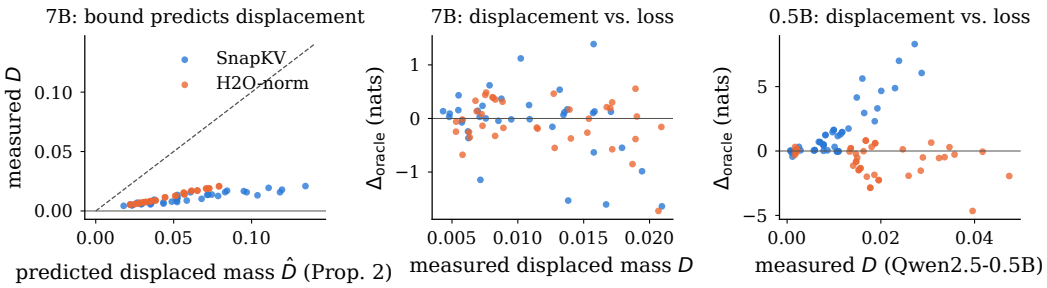

Figure 4: **Left:** Prop. 2's single-competitor predictor $\hat{D}$ against the measured displaced mass D on Qwen2.5-7B (36 cells per policy; Spearman +0.94 for SnapKV, +1.00 for H_2O-norm; the dashed line $\hat{D} = D$ is for scale only: $\hat{D}$ keeps one term of a union bound and is a rank predictor, not a bound in either direction (App. A); points sit below the line because the uncorrected $\hat{D}$ counts the pinned recency window at full weight, quantified below). **Middle:** measured D against Δ_{oracle} at 7B, where the relation is within noise (pooled Spearman -0.22 for SnapKV; within-stratum mean +0.02). **Right:** the same relation on Qwen2.5-0.5B, where it is strong for SnapKV (+0.78) and absent for H_2O-norm (-0.24).

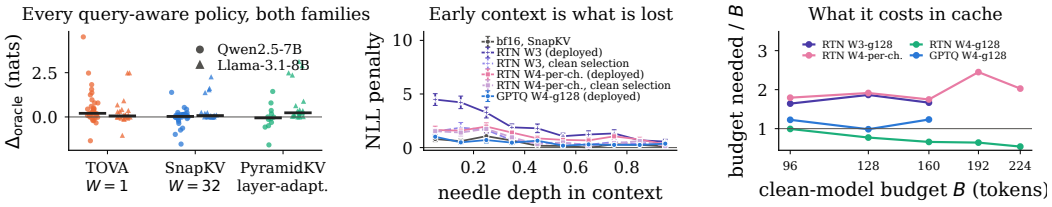

Figure 5: **Left:** oracle term per query-aware policy, one point per identifiable cell, circles Qwen2.5-7B and triangles Llama-3.1-8B at 4k, bars medians. **Middle:** NLL penalty against needle depth in ten bins from 5% to 95% (Qwen2.5-7B, 4k, SnapKV at $B = 96$, 48 items per bin, ± 1 s.e.); "clean selection" is the transplant arm $\mathcal{L}_{1f}$. **Right:** the exchange rate on the dense ten-budget 4k single-needle sweep—the budget the deployed system needs to match the clean model at reference budget B, relative to B; the reference budgets are the sweep's, not the main grid's.

RTN W3 system fall from +4.48 nats at 5% depth to +0.56 at 95%. The full per-bin profiles at 48 items per bin are: bf16 evict-only +0.80 +0.60 +1.07 +0.66 +0.17 +0.15 +0.01 +0.31 +0.26 +0.11; RTN W3-g128 deployed +4.48 +4.22 +3.27 +1.89 +1.80 +1.07 +1.22 +1.32 +0.68 +0.56 and with the clean selection transplanted +1.49 +1.86 +1.66 +0.68 +0.46 +0.44 +0.27 +0.52 +0.33 +0.40; RTN W4 per-channel deployed +1.64 +1.58 +1.96 +1.40 +0.86 +0.72 +0.67 +1.05 +0.76 +0.30 and transplanted +1.53 +1.40 +1.75 +0.96 +0.32 +0.48 +0.16 +0.41 +0.61 +0.14; GPTQ W4-g128 deployed +1.01 +0.49 +0.70 +0.46 +0.65 +0.18 +0.30 +0.25 +0.25 +0.37. The test statistic (the across-bin variance) orders the three quantizers as their calibration noise does (0.06 for GPTQ W4-g128, 0.25 for RTN W4 per-channel, 1.85 for RTN W3-g128), but not proportionally: the last two differ by 7% in σ and sevenfold in the statistic, which is what an exponential in $-\mu^2/\sigma^2$ looks like. The clean policy's own eviction sits at 0.11 with $\sigma = 0$ and is the reference, not a rung. The transplant still leaves a significant gradient ($p < 0.0001$), so the corrupted oracle explains most but not all of the shape for that quantizer. It does not generalise: applying the same arithmetic to RTN W4 per-channel, the first-three-bin mean moves only from +1.73 to +1.56, a far smaller share, so the attribution of the positional tilt to the oracle is demonstrated for the noisiest quantizer and not established for the others. On binary exact match the comparison has a minimum detectable difference of 0.14 at this sample size, so we do not read it. Qwen2.5-3B contributes only 4 identifiable cells (Table 27; App. E). Saved score tensors are stored as float16 while the self arms select from the float32 scores still in memory, so the transplant arms and the self arms differ by that rounding. Comparing the *released* (already float16) tensors against themselves cast back to float32 cannot fail, so it says nothing. The informative check needs the float32 array, which does not survive the process,

so it runs on the GPU (`code/test_transplant.py`, App. E), and the sets are *not* identical: 0.07–1.92% of selected positions move. Almost all of them are exact ties: SnapKV max-pools with kernel 7, which replicates a local maximum across up to seven consecutive positions, so at $B = 160$, 181 of the 224 (layer, head) selections on two released items are decided at the boundary by a tie that the top-k implementation breaks on index order. Three things bound what this can do. The interaction Δ_{int} never uses a transplant arm, so it is untouched. The oracle term does: $\Delta_{\text{oracle}} = \mathcal{L}_{11} - \mathcal{L}_{1\text{f}}$ takes $\mathcal{L}_{1\text{f}}$ with a slightly perturbed clean selection; the sign of that bias is not fixed (the clean selection is score-optimal, not loss-optimal for the quantized model), and where $\Delta_{\text{oracle}} > 0$ a perturbation that raises $\mathcal{L}_{1\text{f}}$ pulls it *towards zero*, where $\Delta_{\text{oracle}} < 0$ away from it, with Δ_{frag} moved by the same amount in the opposite direction. And the perturbation is small on the scale the selection uses. `code/test_transplant.py` measures it directly at a log-score standard deviation of 6×10^{-4} in median over the 12 (item, policy) pairs and 2×10^{-3} at worst, against 0.28–0.73 for a real quantizer's error on the same array and the same log scale (App. G.2), 152–1207 times smaller. On the loss it is not always negligible for a single item: over those pairs float16 storage moves an item's tail NLL by a median of 0.00 and at most 0.25 nats (5 of 12 pairs move at all), on single items at one budget rather than on cell means. All mechanism diagnostics are computed offline from the score tensors saved during prefill, without further forward passes. For each (task, item, quantizer, policy, budget) we compute, per layer and kv-head: the *overlap* $|S_{\text{fp}} \cap S_{\text{q}}|/B$ between the clean and quantized retain sets (the statistic of Zhang et al., 2024); the *displaced mass* $D = \sum_{j \in S_{\text{fp}} \setminus S_{\text{q}}} p_j$, where p is the clean model's raw attention mass over the observation window (for SnapKV) or accumulated attention (for H_2O-norm), normalised over positions; the *noise proxy* $\sigma = \text{std}_j(\log g_j^{\text{q}} - \log g_j^{\text{fp}})$ over the log of the policy's score, the quantity whose ranking the top-B operation uses (for a single scoring query this is exactly Prop. 2's logit perturbation, since the log-partition difference is constant in j and cancels under std_j; for a pooled multi-query score such as SnapKV's it is an approximation); the median *boundary margin* μ in log-score units; and the predictor $\hat{D} = \sum_{j \in S_{\text{fp}}} p_j \exp(-\mu_j^2/(4\sigma^2))$. Cell-level values are means over layers, heads and items.

A correction to that predictor. S_{fp} contains the recency window every policy pins into the cache regardless of score, and C2 says exactly those positions cannot be displaced—they are in S_{fp} and S_{q} in every cell, so their true contribution to D is zero. Summed over all of S_{fp} the predictor counts them anyway, at $\mu = 0$ and therefore weight one, and they carry the largest attention mass. On the released score samples the contestable share of $\hat{D}$ (the mean over heads of the per-head ratio) is $B = 64$: 0.67; $B = 96$: 0.80; $B = 160$: 0.90; $B = 256$: 0.96, so at the smallest budget a third of the predictor is mass that cannot move—and that contamination is most of $\hat{D}$'s budget dependence: on cell means $\hat{D}$ falls from 0.0977 at $B = 64$ to 0.0344 at $B = 256$ while the measured D moves from 0.0104 to 0.0090 and the contestable predictor from 0.0455 to 0.0307 (as a ratio of these cell means the contestable part is 0.47 and 0.89, lower than the per-head shares because the heads with the largest $\hat{D}$ are the most contaminated). `code/mechanism.py` emits both, and a dedicated run at 32 items per cell measures what the correction does: over its 48 cells the uncorrected predictor ranks the measured displaced mass at Spearman +0.89 and the contestable one at +0.95, with the gap concentrated at the small budgets where the contamination is largest (contestable share 0.68 at $B = 64$ against 0.96 at $B = 256$; within-budget +0.83 and +0.97). The figure below and the correlations quoted from it are the uncorrected form, which is what the released `mech_*` records of the main grids carry. Table 16 reports σ and top-B set overlap at $B = 256$ on the single-needle task for the six quantizers of the main grid; Table 15 reports two noise levels per quantizer: σ_{cal}, computed by the same formula on 16 generic 256-token prose prompts with no long-context evaluation at all, and σ, computed on the evaluation items themselves. They are consistently offset, because the evaluation prompts are longer and the log-score spread is therefore larger (compare the two columns of Table 15), but rank the 24 quantizers almost identically (Spearman +0.99), which is what makes the screen usable: it needs only 16 prompts of 256 tokens per quantizer. Both are across-position standard deviations of one realised perturbation, a surrogate for the per-position variance proxy of Assumption 1 that is exact only if the ε_j are identically distributed across j.

D THE CPU-SCALE STUDY

Before the GPU experiments, the whole design was run on Qwen2.5-0.5B-Instruct at ≈1k tokens on two CPU cores (40,196 records across a main grid, controls, a protection grid, the scoring-window sweep of Table 4, and a Qwen2.5-1.5B cross-check), with fractional budgets of 50% to 6.25% and 24

items per task (12 at 1.5B). Its results are the source of every claim in the main text that is attributed to 0.5B, and they differ from 7B mainly in size: a larger oracle share (0.80 of the interaction against 0.41 at 7B), a Δ_{oq} that is positive far more often, and a $D \to \Delta_{\text{oracle}}$ relation that is measurable there and within noise at 7B (App. C). In the identifiability band (defined identically), Δ_{int} was significantly positive in 27 of 44 SnapKV cells (1 negative); the oracle term carried a median 0.80 of the interaction; Δ_{oq} was significantly positive in 17 of 28 cells; the calibration σ correlated with Δ_{oracle} at $\rho = +0.78$ ($p = 5 \times 10^{-10}$, $n = 44$) for SnapKV and at -0.22 for H_2O-norm; measured displacement correlated with Δ_{oracle} at +0.78 and needle-retention loss at +0.86; and $\hat{D}$ predicted D at +0.92 (SnapKV) and +0.96 (H_2O-norm). SnapKV's Δ_{oracle} exceeded H_2O-norm's in 36 of 44 matched cells, but the 0.5B run did not record whether H_2O-norm was itself inside the band in those cells, so we do not build on that comparison. Specificity was sharper than at 7B: on RTN W4 per-channel at 12.5% budget the oracle term was +0.803 nats on the single needle against +0.0045 on the language-modelling control, a ratio of 178. On Qwen2.5-1.5B, RTN W4 with group 512 changed the full-cache loss by +0.012 nats and left exact match at 1.00, yet at 12.5% budget gave $\Delta_{\text{int}} = +1.23$ with $\Delta_{\text{oracle}} = +1.79$ [+0.48, +3.21] ($n = 12$)—the earliest instance of the "quantizer lossless alone, costly together" pattern that Table 1 reproduces at 7B and 8B (the two-sided free-alone gate of §5.1 also requires cheap eviction, which no Table 1 row has). The protection experiment at

Table 4: **The scoring-window sweep on Qwen2.5-0.5B** (single needle, 25% budget, 24 items): SnapKV's observation window W from 4 to 256 queries. The clean policy is in band (exact match ≥ 0.5) only up to $W = 32$, and *within* the band Δ_{oracle} does not fall with W. This controlled sweep does not support C3; neither does the cross-policy comparison of §5.3, which is why we report C3 as unsupported.

Scoring window W	4	8	16	32	64	128	256
bf16 evict-only EM	0.79	0.88	0.79	0.71	0.08	0.00	0.04
Δ_{evict} (bf16)	+0.47	+0.40	+0.72	+0.99	+5.52	+8.52	+12.65
Δ_{oracle}, RTN W4 per-ch.	+4.61	+3.16	+2.52	+4.15	+5.00	+5.23	+3.06
Δ_{oracle}, RTN W3-g128	+6.75	+7.29	+6.91	+7.00	+5.76	+7.27	+1.45

0.5B reached the same verdict as Table 17 (App. G.11): protecting W_K (or W_Q and W_K) did not beat the same-cost placebo, and a finer uniform group removed the oracle term. At 7B and 3 bits W_K protection (+0.69 nats) edges the placebo (+0.84) by less than their intervals, and the uniform control does at least as well (+0.59); the uniform control at 4 bits spends +0.31 extra bits per parameter against protection's +0.094. Table 17 lists every arm.

E Implementation, correctness checks and build disclosures

Harness. Selection scores are computed inside the attention function during prefill, reducing queries and keys to per-position scores in the hook so that a 16k prefill never materialises the full query tensor (the in-hook scores are bit-identical to scores recomputed after the fact from the full query and key tensors captured in a separate forward pass; maximum absolute difference 0). Prefill for cells that need no scores keeps only the last position's logits, which preserves the next-token distribution (KL 2×10^{-9} nats against the full logits on the test prompt, same argmax). The score matmul runs in bf16 with fp32 softmax and accumulation; the resulting top-B sets overlap the fp32 sets (fraction of retained positions in common, per head) by more than 0.995; the separate float16-storage drift of the transplant arms, 0.07–1.92% of positions, is measured in App. C. Pruning keeps the selected positions' keys and values and their original rotary positions; the answer is scored teacher-forced through the pruned cache with the cache position and the position ids decoupled. A CPU suite of 15 checks and a GPU suite of 12 cover these invariants together with three sanity expectations on a single test item (eviction damage falling over three budgets, GPTQ error below RTN's at equal bits, quantizer error falling with bits), and all pass; the CPU check of AWQ's clipping search (T8) accompanies the build control of App. G.7, and the other CPU checks were in place before any result was read; the expectations are checks on the build, not correctness invariants, and a violation would have been investigated rather than reported. None of them exercised the transplant path itself (the CPU suite tests only the identity prune, for which the cache position and the position ids coincide and the decoupling is a no-op), so `code/test_transplant.py` adds the check that matters:

gathering the cache down to S must equal a full-cache forward with an additive $-\infty$ mask on the complement over the answer-token queries, which is exactly the $\eta = 1$ corner of Prop. 1. It does, in 4 of 4 items, to within 0.57% of the loss—the two are different reduction orders in bf16, so they agree to arithmetic noise rather than exactly; the check is on four items because it needs a GPU forward per item and the float32 scores that do not survive a run, and it bounds the transplant construction, not the tail effects of §5.1, which App. C bounds separately by the float16 drift. The same script carries that float16 measurement (App. C).

Table 5: **What each quantizer build costs on the standard benchmark**: WikiText-2 perplexity at sequence length 2048 over 64 windows of a held-out tail of WikiText-2. For the calibrated builds other than the two (C4 calib.) rows the calibration is drawn from the same corpus, so the absolute values are not directly comparable to a published test-split number, but the ordering and the gaps to bf16 are; RTN and bf16 use no calibration data. Δ is computed before rounding, so it can differ by 0.01 from the difference of the printed perplexities. The three parenthesised Qwen rows are the build-side controls of App. G.7; the Llama block has none.

model	build	bits/param	PPL	Δ vs bf16
Qwen2.5-7B	bf16	16.00	7.69	—
Qwen2.5-7B	GPTQ W4-g128	4.16	7.92	+0.22
Qwen2.5-7B	GPTQ W4-g128 (C4 calib.)	4.16	7.98	+0.29
Qwen2.5-7B	GPTQ W3-g128	3.15	8.97	+1.27
Qwen2.5-7B	GPTQ W3-g128 (C4 calib.)	3.15	9.28	+1.59
Qwen2.5-7B	AWQ W4-g128	4.16	7.99	+0.29
Qwen2.5-7B	AWQ W4-g128 (+clip search)	4.16	8.02	+0.33
Qwen2.5-7B	RTN W4-g128	4.16	8.13	+0.43
Qwen2.5-7B	RTN W4-per-ch.	4.00	10.04	+2.34
Qwen2.5-7B	RTN W3-g128	3.15	12.31	+4.61
Llama-3.1-8B	bf16	16.00	7.25	—
Llama-3.1-8B	GPTQ W4-g128	4.16	7.54	+0.29
Llama-3.1-8B	RTN W4-per-ch.	4.00	9.42	+2.17
Llama-3.1-8B	RTN W3-g128	3.15	12.21	+4.96

Quantizers. GPTQ symmetrises the Hessian and escalates damping from $1\times$ to $100\times$ when the Cholesky factorisation fails; linears in a layer are quantized in ascending input-width order and Hessians are freed as they are consumed, so the largest projection is handled when the others are gone. TF32 must be off for the Hessian accumulation: with it on, the $18{,}944$-wide down-projection Hessian became numerically indefinite. AWQ searches the per-input-channel scaling exponent on an 11-point grid against layer-output error; two deviations from the reference implementation are worth naming, since we call these "shipped" recipes: we do not search the clipping range, and the activation statistics come from the first calibration sequence rather than all of them. The first is closed in App. G.7, where a build *with* the reference clipping search behaves like ours (median Δ_{int} +0.016 against +0.011 nats for ours); the second is not. Round-to-nearest takes no calibration data and is applied per weight matrix at load time, so it is not layer-sequential. One consequence should be named because the paper's central contrast is between calibrated and data-free builds: GPTQ's Hessians and AWQ's activation statistics come from WikiText-2, and the needle haystacks are WikiText-2 prose (App. F), so the calibrated arm has seen the evaluation text distribution and the RTN arm has not. We do not think this explains the gap: the calibrated builds' median Δ_{int} is +0.027 against +0.220 for the data-free ones, a factor of 8, and the LongBench documents are not WikiText-2 yet show the same ordering. App. G.7 runs the clean control, the same algorithms at the same bit budgets calibrated on C4, which shares nothing with the haystacks. Moving the calibration off the evaluation corpus does not move the interaction when paired within cell (median -0.030 nats over 10 in-band cells, $p = 1.00$), though the unpaired median moves 25% of the way to the data-free value (App. G.7). Table 5 gives every build's WikiText-2 perplexity at sequence length 2048 over 64 windows of a held-out tail of the corpus. GPTQ is below RTN at both bit budgets and below AWQ at the one budget where all three are measured, which is the published ordering, and the gaps are the published magnitudes: +0.22 for GPTQ W4-g128 on Qwen2.5-7B and +0.29 on Llama-3.1-8B, against +2.34 and +4.61 for the two noisy RTN builds that carry most of this paper's positive cells. GPTQ is deterministic given its calibration file, which we used twice: the Qwen and Llama W4-g128 weights were rebuilt for the later runs, and on the 48 items the two grids share the rebuild reproduces the original cell

values exactly, at every budget and both tasks. Group fallback: a group larger than the input width is clamped, which made RTN W4 group-512 identical to group-256 on the 0.5B model (the duplicate was recorded, not reported twice); and a group that does not divide the input width is halved until it does, which for 96 and 192 on Qwen2.5-7B lands on 1—a per-element scale and zero, costing up to 24 bits per parameter, which is not a quantizer at all. The 3 configurations that fell into that case are dropped throughout: from the 24 rungs of §5.5 and from the 48 cells §5.1 counts. They also collapse onto one another, so keeping them would have counted one null three times.

Tokenization and seeding conventions. (i) Gold answers are tokenized without special tokens: Llama-3.1's tokenizer would otherwise prepend a beginning-of-text token and make it the first teacher-forced target, while Qwen's adds nothing. (ii) Llama's tokenizer encodes the leading space of a numeric answer as a standalone token, so full-string exact match is not comparable across families; the first answer token is in any case predicted from the prompt-final logits, before eviction acts. Exact match is therefore computed over the answer tokens after the first ("tail" exact match), and every exact-match number in this paper uses that definition except where a *full-string* exact match is named: the free-alone gate of §5.1 and Table 24. (iii) Items are keyed by a SHA-256 seed rather than Python's per-process hash, so a resumed run generates the items of the run it resumes, and every record carries the item's prompt hash, which `analyze.check_items` asserts on before any cell is built: over the 1,056 (model, task, context, item) keys of the pinned grids, 0 carry two hashes. (iv) The score capture overrides the default attention implementation in place rather than registering under a new name, so decode-time mask construction is the library's own. (v) The fractional-budget grid is kept as a scope statement (App. B).

Compute. Some forty GPU-hours on a single 24 GB GPU; the figure is read off the run logs rather than measured. It buys 499,289 records because a record is not a forward pass: one captured prefill per (model, task, context, item, quantizer) serves every policy, budget and selection source in that cell, so the 499,289 records rest on 16,512 prefills, 30.2 records each, and the transplant arms re-use the same prefill with a different retain set. Models were served as dequantised bf16 weights, so wall-clock is not representative of kernel-level quantized inference and we make no speed claims. Qwen2.5-3B's per-channel 4-bit and 3-bit RTN variants collapse outright with a full cache ($\Delta_{\text{quant}} = +40.950$ and +36.651 nats on the single needle, exact match 0.00), a known fragility of small Qwen2.5 models under naive rounding; those cells are outside the band and only its RTN W4-g128 cells enter the analysis.

F TASK TEMPLATES

Haystacks are WikiText-2 prose (Merity et al., 2017), filtered to lines longer than 200 characters, concatenated to a target length of 16,000 characters (≈3.8k tokens) or 66,000 characters (≈15.0k tokens). A needle has the form "`One of the special magic numbers for {key} is: {value}.`" with a two-word key drawn from a list of twenty and a seven-digit value; the prompt is

```
Some special magic numbers are hidden within the following
text.  Make sure to memorize them.  I will quiz you about
the numbers afterwards.\n{context}\nWhat is the special
magic number for {query} mentioned in the provided text?
```

followed by the answer prefix, and the gold continuation is the value. The single-needle task inserts one needle at a depth drawn uniformly from $[0.1, 0.9]$ (fixed to ten grid points in the depth study); the distractor task inserts four needles with distinct keys at independent depths, the queried one drawn from $[0.1, 0.9]$ as above and the three distractors from $[0.05, 0.95]$. The language-modelling control takes a prose segment of the same length and scores the next three sentences (a median 90 tokens against the needle tasks' 8, which is why §5.2 compares them per token). Variable tracking follows RULER's chain-of-assignments format with two chains of four variables. Every item is generated from a SHA-256 hash of its (seed, index, length, task) tuple, so the same item appears in every cell of the square.

G ADDITIONAL RESULTS

G.1 AN ALTERNATIVE READING OF THE INTERACTION, AND WHAT RULES IT OUT

The primary loss counts the gold answer tokens after the first (§3.3), so a quantizer whose full-cache damage lands on the first token is recorded as costing almost nothing alone while the model it describes is plainly worse—Table 1 shows exactly that for RTN W4 per-channel and RTN W3-g128, whose ΔNLL on the tail metric is near zero and on the whole answer is not, and whose full-cache full-string exact match is 0.61 and 0.67 against bf16's 0.98 on the same paired items (Table 24). That licenses an alternative reading of Δ_{int} in those rows: not a coupling at all, but the quantizer's *existing* damage becoming visible on the counted tokens once eviction removes the evidence that was masking it. We take the objection seriously because it would explain the rows with the largest interactions without any interaction.

It is ruled out for the oracle half of the decomposition, and not for the other. $\Delta_{\text{oracle}} = \mathcal{L}_{11} - \mathcal{L}_{1\text{f}}$ holds the quantized model *fixed* and changes only which tokens it kept: both arms are the same weights on the same items, so a quantizer that is merely worse cannot produce a positive Δ_{oracle}—only a quantizer that kept different, worse tokens can. Over the 70 cells with $\Delta_{\text{int}} > 0.05$ the oracle term carries 41% of the interaction; over the 101 identifiable SnapKV cells of the four main grids it is individually significant in 32. The remaining share is Δ_{frag}, and there the objection is *not* answered by the design: a more fragile model and a model whose damage was previously uncounted look the same in that term. The two-arm structure was built to separate the oracle channel from everything else, not to decompose everything else. One check a reader will ask for cannot discriminate: recomputing the counts on the whole-answer loss. The first answer token is scored from the prompt-final logits, which are computed before any eviction, so its loss is the same in $\mathcal{L}_{00}$ and $\mathcal{L}_{01}$ and the same in $\mathcal{L}_{10}$ and $\mathcal{L}_{11}$, and it cancels in Δ_{int} exactly: `code/whole_answer.py` rebuilds all 143 cells on the whole answer and reproduces every interval to 10^{-6} nats. The exclusion changes what the quantizer is charged *alone* (the two ΔNLL columns of Table 1), never the interaction, which is why the objection has to be met by the decomposition and the gate rather than by the metric.

Three further facts bear on the alternative without settling it. The injected-noise probe (App. G.2) produces Δ_{oq} with the weights held at bf16, so at least one route to the effect requires no quantizer damage of any kind. The free-alone gate of §5.1 requires full-string non-inferiority, which removes exactly the rows the objection targets, and leaves 6 significantly positive cells (of which 1 survives re-measurement—App. G.6). And on LongBench the loss is the deployment-faithful one over the whole answer with no first-token exclusion (§4), so the masking route is not available there at all, yet 23 of 27 Llama cells and 5 of 12 large-budget Qwen cells are significantly super-additive (App. G.8, G.9).

G.2 INJECTING THE NOISE: A CAUSAL TEST OF THE SCORE CHANNEL

Everything in §5.5 is observational: quantizers differ in score noise *and* in every other way, so a correlation between σ_{cal} and Δ_{oracle} cannot say noise caused it. This probe intervenes on the scores alone. Weights stay bf16, the KV cache stays bf16, the evaluated model is the bf16 model; the only change is that the policy ranks tokens with $s + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ i.i.d. over every (layer, head, position), before taking the top-B—s being the score array the policy ranks, on its own scale, not a log. The loss difference against the same policy on unperturbed scores is Δ_{oq}'s functional form (clean model, perturbed selection) with the perturbation set rather than inferred and no weight perturbation anywhere. Δ_{oq} itself uses the quantizer's selection; here the selection comes from a known noise level instead. As a check, the probe's unperturbed arms reproduce the main grid's L_{00} and L_{01} bit-identically at all eight shared cells (L_{00} on both tasks, L_{01} at the six (task, budget) pairs).

The channel is causal. The ladder runs 11 levels from $\sigma = 0.005$ to 0.60 at 64 items, three independent noise draws per item at the four smallest. Over the 66 (task, budget, σ) conditions Δ_{oq} is significantly positive in 52, and it increases with σ: Spearman ρ between σ and Δ_{oq} is +0.99 in median over the 6 (task, budget) curves and never below +0.96 (Table 7). The ordering is not strict everywhere. On the single-needle curves the three smallest levels are within noise of each other and of zero and two curves invert there, while the distractor curves at $B \geq 160$ are already significant by $\sigma = 0.02$ (Table 7); this is why we report a rank correlation rather than claim monotonicity. Each curve crosses from null into significance somewhere between $\sigma = 0.005$ and 0.050. Corrupting only the ranking signal, with the model untouched, degrades the deployed system, and more corruption

degrades it more. That is the causal content of C1 and of the oracle channel, and it is the one prediction of §3 that survives a controlled test.

How large is a real quantizer's perturbation on the same scale? The probe perturbs the array the policy ranks, on that array's own scale. It is therefore directly comparable to a quantizer's measured error on the same array—and *not* to σ_{cal}, which §5.5's screen computes as a log-scale statistic on sixteen short generic prompts; the two live on different scales and we do not equate them. Inverting each probe curve gives the i.i.d.-equivalent noise for a real quantizer cell: the level that would do that quantizer's actual damage if it were independent across positions, heads and layers. Of the 36 real quantizer cells, 19 have a positive oracle-quality gap to invert; 13 resolve inside the probed range at a median $\sigma = 0.024$ (range 0.005 to 0.036) and 6 fall below our smallest rung. The four rungs below $\sigma = 0.05$ were run so that this is a measurement inside the ladder rather than an extrapolation off its bottom.

And how large is a real quantizer's perturbation? The comparison the probe makes possible needs the quantizer's error on the *same* array, which `code/score_corr.py` measures directly from the paired score dumps: $\varepsilon = s^{\mathrm{q}} - s^{\mathrm{fp}}$ per (layer, head, position), with no model in the loop. Over the 6 quantizers of the main grid at 4k (16 items each, Table 6; two paired items per quantizer are released in `data/scores_release` so the measurement can be re-run) its standard deviation is 0.026 to 0.073, median 0.040—overlapping the i.i.d.-equivalent levels the probe returns (0.005 to 0.036, median 0.024), with a median 1.7× the probe's, and ordered exactly as σ_{cal} orders the quantizers. So the size of the perturbation on the ranked score does account for the size of the oracle term, within about a factor of two. The marginal standard deviation misses the residual factor, because the error is autocorrelated along positions (median 0.85 at lag 1, 0.21 at lag 4, 0.00 at lag 8—the width of SnapKV's pooling kernel), so it has fewer independent chances to reorder the boundary than i.i.d. noise of the same size, and the clean and quantized rankings within a head still agree at Spearman 0.960. Correlation therefore corrects the magnitude by about a factor of two rather than explaining it.

Table 6: **The quantizer's own error on the ranked score** (Qwen2.5-7B, 4k, single needle, 16 items), measured from the paired score dumps with no model in the loop. σ_{cal} is the log-scale screen of §5.5, on a different scale and shown only to fix the ordering; "sd of error" is on the ranked array's own scale and is what the probe's σ is comparable to. Spearman is between the clean and quantized rankings within a (layer, head); ACF is the autocorrelation of the error along positions.

Quantizer	σ_{cal} (log)	sd of error	Spearman	ACF(1)	ACF(4)	ACF(8)
GPTQ W4-g128	0.126	0.0259	0.985	0.85	0.24	0.00
AWQ W4-g128	0.159	0.0264	0.977	0.84	0.21	-0.04
RTN W4-g128	0.170	0.0306	0.971	0.85	0.21	0.00
GPTQ W3-g128	0.253	0.0495	0.948	0.85	0.21	-0.02
RTN W4-per-ch.	0.309	0.0686	0.920	0.86	0.17	0.01
RTN W3-g128	0.331	0.0728	0.904	0.86	0.19	0.01

What it does not rescue. The probe establishes the channel and sizes the perturbation, and it does not rescue Prop. 2 as a quantitative tool. The bound is stated in log-score units with a single boundary margin ($\mu = 0.94$ in the mechanism dump, App. C), while the quantity a top-B rule actually reads is the ranked score itself and what it loses is governed by the density of tokens *at* the boundary, whose margin is near zero at any σ. That mismatch is already visible without the probe: $\hat{D}$, a function of σ and a typical μ alone, does not predict Δ_{oracle} at 7B (App. C), and the screen of §5.5 orders quantizers only weakly ($p = 0.102$ within a bit budget). We report the screen as a screen for that reason, and we do not claim the bound is quantitative anywhere in this paper.

What the probe does not settle. The noise field is drawn independently for every (item, budget, σ, draw) from a seed that includes all four, so the item-level bootstrap resamples noise fields along with items and the monotonicity in σ is not an artefact of a shared draw. The four smallest rungs use three draws per item, the rest one, so at large σ, where the effect is not in doubt, the uncertainty we report is item-level rather than field-level. And the probe answers whether *independent* score noise opens the channel; that correlated noise of the same marginal size does not is an inference from the gap between the two, not a separate manipulation. Manipulating the correlation structure directly, at fixed marginal σ, is the experiment this suggests and we did not run.

Table 7: **Injected score noise** (Qwen2.5-7B, 4k, SnapKV, 64 items, bf16 weights throughout): Δ_{oq} in nats against the i.i.d. noise σ added to the ranked scores before top-B, on those scores' own scale. The 52 of 66 conditions whose 95% interval excludes zero are counted in the text, not marked here; the nulls are all at the smallest levels. Real quantizers never exceed 2.31 nats of Δ_{oq} in absolute value on the same cells.

task	B	0.005	0.01	0.02	0.035	0.05	0.1	0.15	0.2	0.3	0.45	0.6
single	96	-0.13	0.04	0.25	1.0	3.5	9.1	13.2	16.7	17.8	19.5	20.1
single	160	-0.01	-0.05	-0.08	0.36	1.6	6.4	11.0	14.5	16.3	18.6	19.0
single	256	0.06	0.11	0.07	0.34	1.6	5.2	8.5	11.7	13.5	17.9	17.9
distr.	96	-0.13	-0.04	0.34	0.69	1.9	8.2	11.0	11.9	13.6	15.5	15.5
distr.	160	0.15	0.02	1.1	3.1	5.3	11.0	14.7	15.5	18.1	21.0	21.3
distr.	256	0.18	0.62	2.3	4.5	6.4	11.6	13.9	18.5	20.0	22.5	22.6

G.3 A POWER CHECK AT THE OPERATING BUDGETS

The free-alone gate did not survive a fourfold power increase (App. G.6), which makes the obvious next question whether the count that carries the paper survives one. We re-measured every 4k cell of the main grid at $B \in \{64, 96, 160, 256\}$ (all six quantizers, both needle tasks, 48 cells) at 256 items against the main grid's 64 (Table 8). Held-out is defined as in App. G.6.

The band of §3.3 applies to a check as much as to a claim, and it splits this one. The clean policy's exact match is single $B = 64$: 0.13; single $B = 96$: 0.89; single $B = 160$: 0.96; single $B = 256$: 0.98; distr. $B = 64$: 0.00; distr. $B = 96$: 0.12; distr. $B = 160$: 0.68; distr. $B = 256$: 0.96—so the identifiable half is single-needle at $B \in \{96, 160, 256\}$ and distractor at $B \in \{160, 256\}$, 30 of the 48 cells, and the remaining 18 compare two failures in exactly the sense §5.1 warns about. We report the halves separately and read only the first.

In band, power adds detections rather than removing them. At the original n the 30 cells give 10 significantly positive and 1 negative (median +0.095); on the held-out items alone 18 and 1 (median +0.153); pooled 16 and 3 (median +0.149). The sign agrees between original and held-out items in 23 of 30, with 0 significant reversals. The positive count rises by roughly half again while the median effect moves from +0.095 to +0.149 nats. More items should raise the count; the moving median is not a power effect but an item-set one (the held-out items are documents no earlier grid drew), and it is the opposite of what the same increase did to the free-alone gate. The negatives rise too, from 1 to 3, and we name them below.

One caveat about the band applies inside it. The rule of §3.3 is an exact-match threshold, and it does not bound what eviction alone costs in *nats*: Δ_{evict} by block here is single $B = 96$: +0.48; single $B = 160$: +0.13; single $B = 256$: +0.02; distr. $B = 160$: +2.35; distr. $B = 256$: +0.18, so the distractor task at $B = 160$ is admitted at clean exact match 0.68 while the clean policy already pays +2.35 nats. Since Δ_{int} carries $-\mathcal{L}_{01}$, that is exactly where the selection concern of §3.3 and App. G.11 bites hardest. Dropping that block leaves 14 of 24 held-out positives (1 negative), so the count does not rest on it.

What does move is coverage—in one direction only. In the main grid the positives concentrate in the two noisy RTN configurations (51 of 66, against 10 of 42 for GPTQ and AWQ; union cells on the cell records, no clean-policy threshold). In band at four times the items, 5 of the 6 quantizers have at least one held-out significant positive (10 of 10 noisy cells, but also 7 of 15 GPTQ and AWQ cells and 1 of 5 RTN W4-g128 cells), so part of the concentration reported in §5.1 is a power limit at $n = 64$ rather than a property of the quantizer. Only part, and only for GPTQ: every one of those 7 is a GPTQ cell, AWQ W4-g128 has none out of sample or pooled (it has one at the original n; Table 8), and pooled the quantizer count is 4 of 6 because RTN W4-g128's only held-out positive (the distractor task at $B = 256$) loses significance at $n = 256$. On the 15 in-band cells of the three W4-g128 builds, the slice a deployment is most likely to ship, four times the items leaves 4 positive and 3 negative, median +0.039 nats, against 10 of 10 for the noisy builds out of sample and pooled. The one row of Table 1 this run re-measures directly (GPTQ W4-g128 on the single needle at $B = 96$, its first row) is +0.57 [+0.03, +1.14] at 64 items, +0.08 [-0.22, +0.41] on the held-out items alone and +0.23 [+0.01, +0.46] pooled: it stays significant pooled but not out of sample, and a reader should treat that row as the pooled estimate. Power buys GPTQ detections; it does not close the gap on the shipped

recipes, and §5.1's statement of that limit stands. The three in-band negatives are RTN W4-g128 on the single needle at $B \in \{96, 160\}$ and AWQ W4-g128 on the distractor task at $B = 160$; none reverses a significant positive.

On the second family the same check is entirely in band. Repeating it on Llama-3.1-8B at 4k (the three quantizers of the cross-family grid, both tasks, $B \in \{96, 160, 256\}$, 256 items against that grid's 48), the clean policy's exact match never falls below 0.83, so all 18 cells are identifiable and there is nothing to split (Table 9). The counts are 11 positive and 0 negative at the original n (median +0.098), 13 and 0 on the 208 held-out items (median +0.179), 13 and 0 pooled (median +0.172), with the sign agreeing in 15 of 18 and 0 reversals. *No cell is significantly negative at any n.* The concentration is sharper here than on Qwen: 12 of the 12 noisy-RTN cells are positive out of sample against 1 of 6 for GPTQ W4-g128. This run also re-measures the cross-family grid's own 48 items with GPTQ weights rebuilt from scratch, and reproduces them exactly (App. E).

Back on Qwen, out of band the 18 cells of Table 8's lower block give 4/2, 8/2 and 9/2, with the sign agreeing in 18 of 18; those numbers describe the failure regime and are not evidence about the interaction. Two are worth naming anyway: GPTQ W3-g128 at $B = 64$ is significantly *sub*-additive on both tasks, out of sample and pooled. Where the clean policy has already failed, the interaction is not one-signed and the most sub-additive configuration is not the quietest one.

G.4 The 16k cells at four times the items

The Qwen 16k cells are among the thinnest in the paper (32 items, against 16 at 32k), and they carry the free-alone result of §5.1. This re-measures every one of them, the 18 cells the original 16k grid and a 128-item re-run share (three quantizers, both tasks, $B \in \{160, 256, 512\}$), in the same three ways as App. G.3: at the original n, on the 96 items no released grid outside the power runs contains, and pooled (Table 10). Every cell is in band there (the clean policy's exact match never falls below 0.69), so nothing is split off.

Nothing significant reverses. Δ_{int} is significantly positive in 10 of 18 at $n = 32$, 9 out of sample and 11 pooled, with 0 significantly negative at any n; the medians are +0.475, +0.280 and +0.309 nats and the sign agrees in 15 of 18 cells. The estimate shrinks, as a winner's-curse correction should, and the count does not. Three cells lose significance out of sample and two gain it, and three flip sign, all three non-significant at both n. Like the Llama check above (four times the items, an out-of-sample column, no negative cell at any n), it is a power check the effect passes, on the setting whose thinness we flagged.

The free-alone cells are the exception, and they were before. The three 16k cells that passed the free-alone gate are re-certified in App. G.6 at 128 items rather than the 32 they were certified on. Two of the three keep a significantly positive Δ_{int} and *none* still passes the gate; over all 6 recertified cells 5 stay significantly positive, and the 1 that still passes is a 4k Qwen2.5-3B cell, not a 16k one. What does not survive is the gate, not the interaction.

G.5 The budget decay at four times the power

The claim that the interaction is a boundary phenomenon rests, in the main grid, on 64-item cells. The two 256-item runs measure the same two lossless-alone quantizers on the same items at $B = 160, 256, 384$, so the decay can be read at four times the power—and, at the largest budget, on the single-needle cells, in a regime where eviction alone is free by the gate of §5.1.

It decays as claimed. Median Δ_{int} over the four cells is +0.100 at $B = 160$ (1 positive, 1 negative), +0.097 at $B = 256$ (2, 0) and +0.012 at $B = 384$ (1, 0); the cost of eviction alone falls with it, from at most +2.350 nats to +0.058 (Table 11; the gate's ≤ 0.05 is met by the single-needle cells at $B = 256$ and $B = 384$ and by no distractor cell). The consequence for the paper's strongest claim is the one worth stating plainly: at $B = 256$, 2 cells clear the free-alone gate and 1 is significantly positive (+0.022 and +0.061 nats); at $B = 384$, 2 clear it and 0 are significant (-0.017 and +0.006). The free-alone interaction is therefore not a property of joint compression in general but of a narrow band of budgets just above the point where the clean policy starts to fail—which is the same statement §6 makes about the effect as a whole, now measured at 256 items and at a budget where neither compression costs anything on its own.

Table 8: **The 4k cells at $B \in \{64, 96, 160, 256\}$ re-measured at 256 items** (Qwen2.5-7B, SnapKV, every main-grid quantizer). *Held-out*: items no released grid outside the power runs contains. The last column is the clean policy's own exact match: a cell is identifiable under §3.3 only at ≥ 0.5. Rows above the second rule are the 30 in-band cells; below it are the 18 that are not, kept for completeness and not read as evidence. The AWQ W4-g128 rows (five in band, three out) also compare two builds: the *original* column is the unseeded build and the other two the seeded one (App. G.11), so they are a build comparison as much as a power comparison. Held-out and original n need not sum to pooled n (Table 13). */†: interval excludes zero from above/below.

			original		held-out		pooled		
quantizer	task	B	n	Δ_{int}	n	Δ_{int}	n	Δ_{int}	clean EM
AWQ W4-g128	single	96	64	-0.14	128	-0.02	256	-0.15	0.89
AWQ W4-g128	single	160	64	+0.01	128	+0.09	256	+0.04	0.96
AWQ W4-g128	single	256	64	+0.26*	128	+0.01	256	+0.04	0.98
GPTQ W3-g128	single	96	64	-0.19	128	+0.08	256	+0.05	0.89
GPTQ W3-g128	single	160	64	-0.09	128	+0.15*	256	+0.08	0.96
GPTQ W3-g128	single	256	64	-0.02	128	+0.24*	256	+0.16*	0.98
GPTQ W4-g128	single	96	64	+0.57*	128	+0.08	256	+0.23*	0.89
GPTQ W4-g128	single	160	64	+0.03	128	0.00	256	-0.03	0.96
GPTQ W4-g128	single	256	64	+0.11	128	+0.07*	256	+0.06*	0.98
RTN W3-g128	single	96	64	+2.00*	128	+1.76*	256	+1.65*	0.89
RTN W3-g128	single	160	64	+0.21	128	+0.16*	256	+0.16*	0.96
RTN W3-g128	single	256	64	+0.15*	128	+0.15*	256	+0.16*	0.98
RTN W4-per-ch.	single	96	64	+0.93*	128	+1.02*	256	+0.91*	0.89
RTN W4-per-ch.	single	160	64	+0.33*	128	+0.21*	256	+0.21*	0.96
RTN W4-per-ch.	single	256	64	+0.17*	128	+0.12*	256	+0.14*	0.98
RTN W4-g128	single	96	64	-0.16	128	-0.11	256	-0.20†	0.89
RTN W4-g128	single	160	64	-0.03	128	-0.06†	256	-0.07†	0.96
RTN W4-g128	single	256	64	+0.12	128	0.00	256	+0.02	0.98
AWQ W4-g128	distr.	160	64	-0.98†	176	-0.39	256	-0.44†	0.68
AWQ W4-g128	distr.	256	64	+0.02	176	+0.04	256	-0.03	0.96
GPTQ W3-g128	distr.	160	64	-1.04	176	+0.58*	256	+0.16	0.68
GPTQ W3-g128	distr.	256	64	+0.72*	176	+0.32*	256	+0.36*	0.96
GPTQ W4-g128	distr.	160	64	-0.02	176	+0.85*	256	+0.65*	0.68
GPTQ W4-g128	distr.	256	64	+0.08	176	+0.20*	256	+0.13*	0.96
RTN W3-g128	distr.	160	64	+1.12	176	+2.02*	256	+1.86*	0.68
RTN W3-g128	distr.	256	64	+0.85*	176	+0.89*	256	+0.89*	0.96
RTN W4-per-ch.	distr.	160	64	+0.03	176	+0.62*	256	+0.49*	0.68
RTN W4-per-ch.	distr.	256	64	+0.66*	176	+0.25*	256	+0.31*	0.96
RTN W4-g128	distr.	160	64	+0.05	176	+0.29	256	+0.23	0.68
RTN W4-g128	distr.	256	64	+0.22	176	+0.16*	256	+0.13	0.96
AWQ W4-g128	single	64	64	+1.22*	128	+1.37*	256	+1.31*	0.13
GPTQ W3-g128	single	64	64	-2.66†	128	-1.90†	256	-1.81†	0.13
GPTQ W4-g128	single	64	64	+2.27*	128	+2.11*	256	+2.62*	0.13
RTN W3-g128	single	64	64	+3.84*	128	+4.18*	256	+4.25*	0.13
RTN W4-per-ch.	single	64	64	+1.29	128	+2.78*	256	+2.40*	0.13
RTN W4-g128	single	64	64	+1.46*	128	+2.07*	256	+1.87*	0.13
AWQ W4-g128	distr.	64	64	+1.45	176	+1.16*	256	+1.24*	0.00
AWQ W4-g128	distr.	96	64	+0.23	176	+0.91*	256	+0.81*	0.12
GPTQ W3-g128	distr.	64	64	-1.16	176	-1.27†	256	-1.20†	0.00
GPTQ W3-g128	distr.	96	64	-1.64†	176	-0.53	256	-0.65	0.12
GPTQ W4-g128	distr.	64	64	+0.97	176	+0.31	256	+0.41	0.00
GPTQ W4-g128	distr.	96	64	+0.20	176	+0.42	256	+0.36	0.12
RTN W3-g128	distr.	64	64	+1.52	176	+1.94*	256	+1.88*	0.00
RTN W3-g128	distr.	96	64	+1.06	176	+0.78	256	+0.94*	0.12
RTN W4-per-ch.	distr.	64	64	+1.07	176	+0.67	256	+0.71	0.00
RTN W4-per-ch.	distr.	96	64	+0.24	176	+0.43	256	+0.50	0.12
RTN W4-g128	distr.	64	64	-0.18	176	-0.12	256	-0.14	0.00
RTN W4-g128	distr.	96	64	+0.80	176	+0.31	256	+0.53	0.12

Table 9: **The same power check on Llama-3.1-8B** (4k, SnapKV, the three quantizers of the cross-family grid, 256 items). Every cell is inside the band (the clean policy's exact match, last column, is 0.83–1.00 throughout), so no split is needed. *Held-out*: the 208 items the cross-family grid does not contain. */†: interval excludes zero from above/below.

			original		held-out		pooled		
quantizer	task	B	n	Δ_{int}	n	Δ_{int}	n	Δ_{int}	clean EM
GPTQ W4-g128	single	96	48	+0.005	208	-0.011	256	-0.008	1.00
GPTQ W4-g128	single	160	48	+0.003	208	-0.002	256	-0.001	1.00
GPTQ W4-g128	single	256	48	-0.001	208	-0.008	256	-0.006	1.00
RTN W3-g128	single	96	48	+0.218*	208	+0.352*	256	+0.327*	1.00
RTN W3-g128	single	160	48	+0.108*	208	+0.207*	256	+0.188*	1.00
RTN W3-g128	single	256	48	+0.083*	208	+0.168*	256	+0.152*	1.00
RTN W4-per-ch.	single	96	48	+0.176*	208	+0.157*	256	+0.161*	1.00
RTN W4-per-ch.	single	160	48	+0.087*	208	+0.106*	256	+0.103*	1.00
RTN W4-per-ch.	single	256	48	+0.037	208	+0.059*	256	+0.055*	1.00
GPTQ W4-g128	distr.	96	48	+0.069	208	+0.689*	256	+0.573*	0.83
GPTQ W4-g128	distr.	160	48	-0.019	208	+0.041	256	+0.029	0.98
GPTQ W4-g128	distr.	256	48	-0.013	208	-0.007	256	-0.008	0.98
RTN W3-g128	distr.	96	48	+2.129*	208	+2.630*	256	+2.536*	0.83
RTN W3-g128	distr.	160	48	+0.254*	208	+0.375*	256	+0.352*	0.98
RTN W3-g128	distr.	256	48	+0.173*	208	+0.271*	256	+0.253*	0.98
RTN W4-per-ch.	distr.	96	48	+1.944*	208	+1.869*	256	+1.883*	0.83
RTN W4-per-ch.	distr.	160	48	+0.471*	208	+0.223*	256	+0.269*	0.98
RTN W4-per-ch.	distr.	256	48	+0.157*	208	+0.190*	256	+0.184*	0.98

Table 10: **The 16k cells re-measured at 128 items.** Qwen2.5-7B, SnapKV, the three quantizers the original 16k grid and the power run share. *Held-out*: items no released grid outside the power runs contains. The last column is the clean policy's own exact match. *: interval excludes zero from above; no cell is significantly negative at any n.

			original		held-out		pooled		
quantizer	task	B	n	Δ_{int}	n	Δ_{int}	n	Δ_{int}	clean EM
GPTQ W3-g128	single	160	32	+0.562*	96	+0.245	128	+0.324*	0.94
GPTQ W3-g128	single	256	32	-0.031	96	+0.073	128	+0.047	0.97
GPTQ W3-g128	single	512	32	0.000	96	+0.079	128	+0.060	0.98
GPTQ W4-g128	single	160	32	+0.056	96	+0.015	128	+0.025	0.94
GPTQ W4-g128	single	256	32	+0.012	96	-0.008	128	-0.003	0.97
GPTQ W4-g128	single	512	32	-0.010	96	-0.027	128	-0.023	0.98
RTN W3-g128	single	160	32	+1.235*	96	+0.764*	128	+0.882*	0.94
RTN W3-g128	single	256	32	+0.216	96	+0.315*	128	+0.291*	0.97
RTN W3-g128	single	512	32	-0.001	96	+0.038	128	+0.028	0.98
GPTQ W3-g128	distr.	160	32	+1.918*	96	+1.885*	128	+1.894*	0.69
GPTQ W3-g128	distr.	256	32	+1.337*	96	+0.928*	128	+1.030*	0.91
GPTQ W3-g128	distr.	512	32	+0.509*	96	+0.223	128	+0.295	0.96
GPTQ W4-g128	distr.	160	32	+0.357	96	+1.393*	128	+1.134*	0.69
GPTQ W4-g128	distr.	256	32	+1.041*	96	+0.401*	128	+0.561*	0.91
GPTQ W4-g128	distr.	512	32	+0.440*	96	+0.116	128	+0.197*	0.96
RTN W3-g128	distr.	160	32	+3.443*	96	+3.206*	128	+3.265*	0.69
RTN W3-g128	distr.	256	32	+2.430*	96	+2.282*	128	+2.319*	0.91
RTN W3-g128	distr.	512	32	+0.695*	96	+0.522*	128	+0.565*	0.96

G.6 A HELD-OUT REPLICATION AT HIGHER ITEM COUNT

The main grid runs 64 items at 4k, 16 at 32k and 24 on Llama at 16k, which is enough to see a large interaction and not enough to resolve a small one. We re-ran 28 of its cells (for each setting, the two quantizers and the budgets at and above the operating point where the free-alone gate can be met) with 256, 64 and 64 items. Two settings are Qwen2.5-7B and one is Llama-3.1-8B; on Llama the free-alone gate cannot be evaluated at all (its full-string exact match is uninformative, App. E), so that block tests sign and magnitude only.

The item generator and its seed are unchanged, so the new item sets *nest* the old ones: pooling is a precision gain, not an independent test, and every number below says which column it comes from.

Table 11: **Budget decay at 256 items** (Qwen2.5-7B, 4k, SnapKV, the two quantizers that are lossless alone here). Δ_{evict} is what eviction costs the bf16 model at that budget; the free-alone gate of §5.1 requires $\Delta_{\text{evict}} \leq 0.05$. $^{*}/^{\dagger}$: interval excludes zero from above/below.

quantizer	task	B	Δ_{evict}	Δ_{int}	[95% CI]	Δ_{oracle}
GPTQ W4-g128	single	160	+0.126	-0.032	[-0.118, +0.045]	+0.003
GPTQ W4-g128	single	256	+0.017	+0.061*	[+0.007, +0.131]	+0.075
GPTQ W4-g128	single	384	+0.021	+0.006	[-0.024, +0.046]	+0.001
RTN W4-g128	single	160	+0.126	-0.074†	[-0.143, -0.014]	-0.018
RTN W4-g128	single	256	+0.017	+0.022	[-0.022, +0.096]	+0.032
RTN W4-g128	single	384	+0.021	-0.017	[-0.041, +0.002]	-0.006
GPTQ W4-g128	distr.	160	+2.350	+0.651*	[+0.193, +1.121]	+0.373
GPTQ W4-g128	distr.	256	+0.181	+0.135*	[+0.006, +0.276]	+0.134
GPTQ W4-g128	distr.	384	+0.058	+0.018	[-0.040, +0.081]	+0.031
RTN W4-g128	distr.	160	+2.350	+0.232	[-0.212, +0.687]	+0.257
RTN W4-g128	distr.	256	+0.181	+0.132	[-0.005, +0.296]	+0.168
RTN W4-g128	distr.	384	+0.058	+0.033*	[+0.001, +0.073]	+0.046

Table 13 gives three per cell. *Held-out* is the strict one: an item counts only if *no* released grid outside the power run contains it at this (model, context, task). That is a stronger exclusion than dropping the main grid alone, since the 24-rung ladder re-measures two of these cells on 128 items and several supporting grids overlap the first 64, and it is why the held-out n is below $256 - 64$. For the same reason the three columns' item counts do not add: an item another grid touched is excluded from *held-out* but still enters *pooled*.

Out of sample the sign agrees with the original in 21 of 28 cells; 0 reverse sign with both estimates significant, and every disagreement involves an estimate that crosses zero in at least one column. Interval overlap is 27 of 28, but that is a weak check at a median original interval width of 0.41 nats. The sharp test is available because the two item sets are disjoint and the estimates therefore independent: a two-sample z on the difference rejects equality in 2 of 28 cells (largest $|z| = 2.95$) against 1.4 expected by chance. A nil null that fails to reject is not evidence of agreement, so we also state what the comparison could have found: the median standard error of the difference is 0.19 nats, an 80%-power minimum detectable difference of 0.53 nats, and at most 0.11 on the gated cells. The reading is therefore that no cell shifted by more than the design can resolve, and that for the small effects the design resolves little. Descriptively, the original point estimate falls outside the held-out interval in 6 cells and the held-out estimate outside the original's in 7; neither of those is a 5% test, since both endpoints are noisy, and the z test is what settles it. Power behaves as it should: the same cells give 8 significantly positive and 2 negative at the original n, 14 and 1 held out, and 15 and 1 pooled.

Two things the extra power buys are worth stating against our own claim. First, a significantly *negative* cell appears—-0.074 [-0.143, -0.014] pooled and -0.064 [-0.162, -0.003] on 128 held-out items, at $B = 160$. The interaction is not one-signed, and at $n = 64$ that cell read as null. Second, the free-alone effect is small where the quantizer is a clean 4-bit build, and pooling flatters it. 3 of the 28 cells pass the gate of §5.1. Pooled, 2 are significantly positive (+0.061 [+0.007, +0.131] and +0.056 [0.000, +0.140]), the third +0.022 [-0.022, +0.096]. Out of sample 1 is: +0.072 [+0.002, +0.168] on 128 items, with the second at +0.046 [0.000, +0.134] just missing and the third at 0.000 [-0.010, +0.008]. One gated cell therefore survives an out-of-sample test at this n, at +0.072 nats. Pooled, the median over the gate is +0.056 against +0.23 over the 25 cells outside it (both pooled columns), and both are far below the +0.44 the headline free-alone cell of §5.1 showed *as first measured*—which is itself +0.197 at 128 items (Table 12), below the larger of the two. The Llama block is the other side of that: with a noisy quantizer the effect is large and replicates cleanly out of sample in every cell (6 of 6 RTN W3-g128 cells significantly positive held out, point estimates within 0.5 nats of the original in 5 of them).

The certification is the weak point, and most of it does not survive. The 6 significant free-alone cells of §5.1 sit on the paper's smallest item sets ($n = 16$–48), and the gate is four thresholds on nuisance statistics estimated there. We re-measured all 6 at four times the items (Table 12). The interaction holds up, with 5 of 6 keeping a significantly positive Δ_{int} and point estimates that move

by less than their intervals, but the certification does not, since only 1 still passes the gate. 4 of the 5 failures are on eviction's own cost, which rises above 0.05 nats once the item pool is larger, and two of those four fail a second arm as well (Table 12); the free-alone regime at those settings was in part a property of the small pool, not of the configuration—and 1 is the exact-match arm at 32k.

The consequence for §5.1 is stated there and we repeat it here: the free-alone count is a statement at $n = 16$–48, not a claim that survives quadrupling the items. What does survive is that the interaction in those cells is real and of the size reported. The re-measured 16k cells were run with a rebuilt GPTQ W3-g128; GPTQ is deterministic given the calibration file, and the rebuild reproduces the original grid bit-identically on every cell the two share, so the two item sets are directly comparable.

Table 12: **Re-certifying the significant free-alone cells.** The 6 cells of §5.1 in which Δ_{int} is significantly positive and both compressions pass the gate, re-measured at four times the items. *: interval excludes zero from above. The last column says whether the cell still passes the gate at the larger n, and which arm fails if not.

model	ctx	quantizer	task	B	original n	original Δ_{int}	re-measured n	re-measured Δ_{int}	gate
Qwen2.5-7B	16k	GPTQ W3-g128	single	160	32	+0.562*	128	+0.324*	fails: Δ_{evict}, clean EM
Qwen2.5-7B	16k	GPTQ W3-g128	distr.	512	32	+0.509*	128	+0.295	fails: Δ_{quant}, Δ_{evict}
Qwen2.5-7B	16k	GPTQ W4-g128	distr.	512	32	+0.440*	128	+0.197*	fails: Δ_{evict}
Qwen2.5-3B	4k	RTN W4-g128	single	160	48	+0.263*	192	+0.306*	fails: Δ_{evict}
Qwen2.5-3B	4k	RTN W4-g128	single	256	48	+0.234*	192	+0.172*	passes
Qwen2.5-7B	32k	RTN W3-g128	single	1024	16	+0.313*	64	+0.264*	fails: EM

The exact-match arm of the gate is also where the two readings of "matches bf16" part company. On the main grid they agree exactly (0 cells admitted by the point rule alone, 0 by non-inferiority alone, out of 14), so there the change is a robustness statement and nothing else. Here they do not: the point rule admits 2 cells with 1 significant against 3 and 2. What separates them is how each reads a small gap at a given n: at 256 items a quantizer -0.016 behind bf16 has a lower bound of -0.035 and is non-inferior at our margin, while on the main grid a +0.125 *lead* at $n = 16$ has a lower bound of -0.125 and is not. The margin 0.05 is the exact-match counterpart of the 0.05-nat loss tolerance the gate already used and was not tuned, but it is a choice and it moves the count, so here is the whole curve on the main grid—gated cells, significant ones in brackets: $\delta = 0.00$: 10 (5); $\delta = 0.02$: 10 (5); $\delta = 0.05$: 14 (6); $\delta = 0.08$: 14 (6); $\delta = 0.10$: 14 (6). A margin of zero (require the paired lower bound to clear zero) admits 4 fewer cells and 1 fewer significant ones; every margin at or above 0.05 gives the same set. We report 0.05 and this curve rather than a single number, and a reader who prefers the strict reading should read 14 as the count at $\delta = 0.05$ and 10 at $\delta = 0$.

Does the calibration screen order the quantizers on real documents too? Table 14 is the only place to ask, since it is the only real-document grid with more than two quantizers in band. Over the 4 quantizers with an in-band SnapKV cell, σ_{cal} ranks the mean Δ_{int} over each quantizer's three SnapKV cells at Spearman +1.00 (GPTQ W4-g128 ($\sigma_{\text{cal}} = 0.13$) +0.06, RTN W4-g128 ($\sigma_{\text{cal}} = 0.17$) +0.19, RTN W4-per-ch. ($\sigma_{\text{cal}} = 0.31$) +1.07, RTN W3-g128 ($\sigma_{\text{cal}} = 0.33$) +1.52 nats, all in Table 14), and under TOVA, where every cell is in band, at +0.60: GPTQ W4-g128 +0.69, RTN W4-g128 +0.66, RTN W4-per-ch. +2.41, RTN W3-g128 +2.08, so the two noisiest builds swap. Four quantizers is an ordering check and not a correlation estimate, and we read it as consistent with §5.5 rather than as a second confirmation of it.

Table 18 gives the matched-setting values behind §5.2 for all six quantizers. The two model-independent policies, StreamingLLM and random, have $\Delta_{\text{int}} = \Delta_{\text{frag}}$ exactly (Remark 1); at these budgets they never retain the needle, so their 216 cells lie outside the band and span -7.8 to +9.4 nats, which checks the transplant pipeline rather than the split. The variable-tracking column is included for completeness; its interaction terms are large but the task is not solved by any model in our scored format, so neither term is interpretable there. On the control the interaction is often large and of either sign (the loss is summed over a long continuation, so the intervals are wide), while the oracle term stays within noise of zero for every quantizer except AWQ W4, where it is negative. The ratio form of the specificity test (the oracle term as a share of Δ_{int} rather than in nats) agrees with the per-token test of §5.2: median +0.43 on the needles against -0.004 on the control ($p = 0.013$, signed rank). We

Table 13: **Held-out replication.** Each cell re-measured at higher item count. *Original*: the main-grid estimate. *Held-out*: the same cell on items no released grid outside the power run contains—the only out-of-sample column. *Pooled*: all items, and therefore not independent of *original*. Held-out and original n need not sum to pooled n: an item another grid touched is excluded from held-out but still enters pooled (they do sum where no other grid touched the cell's items, as in the Llama 16k rows). */†: interval excludes zero from above/below.

				original		held-out		pooled	
setting	quantizer	task	B	n	Δ_{int}	n	Δ_{int}	n	Δ_{int}
Qwen 4k	GPTQ W4-g128	single	160	64	+0.031	128	-0.004	256	-0.032
Qwen 4k	GPTQ W4-g128	single	256	64	+0.113	128	+0.072*	256	+0.061*
Qwen 4k	RTN W4-g128	single	160	64	-0.029	128	-0.064†	256	-0.074†
Qwen 4k	RTN W4-g128	single	256	64	+0.117	128	0.000	256	+0.022
Qwen 4k	GPTQ W4-g128	distr.	160	64	-0.021	176	+0.846*	256	+0.651*
Qwen 4k	GPTQ W4-g128	distr.	256	64	+0.077	176	+0.200*	256	+0.135*
Qwen 4k	RTN W4-g128	distr.	160	64	+0.048	176	+0.286	256	+0.232
Qwen 4k	RTN W4-g128	distr.	256	64	+0.220	176	+0.156*	256	+0.132
Qwen 32k	GPTQ W4-g128	single	512	16	-0.047	42	-0.059	64	-0.049
Qwen 32k	GPTQ W4-g128	single	1024	16	+0.103	42	+0.046	64	+0.056*
Qwen 32k	RTN W3-g128	single	512	16	+0.291	42	+0.392	64	+0.352
Qwen 32k	RTN W3-g128	single	1024	16	+0.313*	42	+0.273*	64	+0.264*
Qwen 32k	GPTQ W4-g128	distr.	512	16	+0.095	48	+0.694*	64	+0.545*
Qwen 32k	GPTQ W4-g128	distr.	1024	16	-0.608	48	+0.025	64	-0.133
Qwen 32k	RTN W3-g128	distr.	512	16	+2.480*	48	+2.070*	64	+2.172*
Qwen 32k	RTN W3-g128	distr.	1024	16	+1.204	48	+1.273*	64	+1.255*
Llama 16k	RTN W4-g128	single	96	24	+0.025	40	+0.024	64	+0.024*
Llama 16k	RTN W4-g128	single	160	24	0.000	40	+0.017	64	+0.010
Llama 16k	RTN W4-g128	single	256	24	0.000	40	-0.015	64	-0.009
Llama 16k	RTN W3-g128	single	96	24	+0.588*	40	+0.572*	64	+0.578*
Llama 16k	RTN W3-g128	single	160	24	+0.344*	40	+0.257*	64	+0.290*
Llama 16k	RTN W3-g128	single	256	24	+0.173*	40	+0.283*	64	+0.242*
Llama 16k	RTN W4-g128	distr.	96	24	-0.933†	40	+0.069	64	-0.307
Llama 16k	RTN W4-g128	distr.	160	24	-0.017	40	-0.039	64	-0.030
Llama 16k	RTN W4-g128	distr.	256	24	-0.019†	40	-0.003	64	-0.009
Llama 16k	RTN W3-g128	distr.	96	24	+3.829*	40	+4.218*	64	+4.072*
Llama 16k	RTN W3-g128	distr.	160	24	+0.462*	40	+0.954*	64	+0.770*
Llama 16k	RTN W3-g128	distr.	256	24	+0.132*	40	+0.704*	64	+0.489*

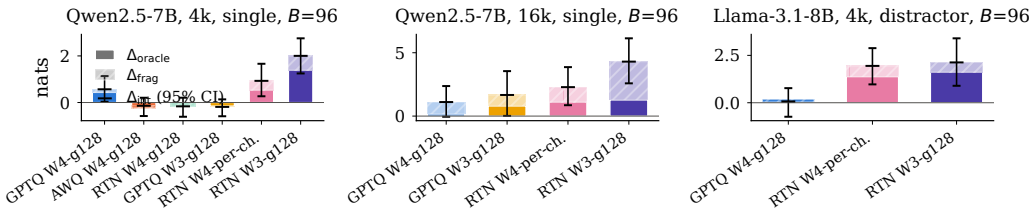

Figure 6: **Exact decomposition of the interaction** at the operating budget: solid bars Δ_{oracle} (wrong tokens kept), hatched bars Δ_{frag} (more hurt by the same tokens); the marker with interval is Δ_{int}, their sum.

report the per-token form in the body because the ratio is undefined where Δ_{int} is near zero. Neither form is assumption-free, and the third form is null: in *absolute* nats summed over the answer, the oracle term is -0.065 on the control against +0.049 on the needles, $p = 0.86$—the control's answer is 90 tokens against the needles' 8, so the summed comparison is dominated by length. The specificity claim therefore depends on putting the two tasks on a common per-token footing, and we state it that way rather than as a claim about summed loss. Read one test per (quantizer, budget) unit instead of per cell, which removes the pseudo-replication across budgets, the per-token comparison holds in 13 of 18 units ($p = 0.016$).

Table 14: **LongBench QA** (Qwen2.5-7B, 120 items, 40 each from multi-field QA, HotpotQA and 2WikiMQA; 9.66 nats with a full cache), every in-band cell of the four quantizers × four budgets × five policies grid. "trimmed" is the 10%-trimmed mean of the same per-item contrast with its own bootstrap interval; * on either column: that column's 95% interval excludes zero from above. TOVA clears the band at 1.23×, SnapKV comfortably (1.10×) and PyramidKV with less margin (1.33×); H_2O-norm never clears it (1.715×, its best over all budgets) and neither does StreamingLLM (2.24×), so neither contributes a cell. Over all 28 in-band cells, Δ_{int} is significantly positive in 12 of 12 TOVA, 6 of 12 SnapKV and 2 of 4 PyramidKV cells, with none significantly negative.

Policy	Quantizer	B	ΔNLL alone	Δ_{evict}	Δ_{int} [95% CI]	trimmed	Δ_{oracle}
TOVA	GPTQ W4-g128	256	-0.27	+3.95	+0.69 [+0.08, +1.29]*	+0.59*	+0.12
TOVA	GPTQ W4-g128	512	-0.27	+2.87	+0.78 [+0.21, +1.35]*	+0.49*	+0.15
TOVA	GPTQ W4-g128	1024	-0.27	+2.24	+0.61 [+0.17, +1.07]*	+0.37*	+0.12
TOVA	RTN W4-g128	256	+0.07	+3.95	+0.80 [+0.17, +1.43]*	+0.68*	+0.34
TOVA	RTN W4-g128	512	+0.07	+2.87	+0.62 [+0.14, +1.13]*	+0.39*	+0.21
TOVA	RTN W4-g128	1024	+0.07	+2.24	+0.54 [+0.05, +1.08]*	+0.48*	+0.17
TOVA	RTN W4-per-ch.	256	+0.91	+3.95	+2.57 [+1.62, +3.59]*	+2.08*	+1.32
TOVA	RTN W4-per-ch.	512	+0.91	+2.87	+2.50 [+1.52, +3.54]*	+1.83*	+0.79
TOVA	RTN W4-per-ch.	1024	+0.91	+2.24	+2.16 [+1.27, +3.09]*	+1.69*	+0.91
TOVA	RTN W3-g128	256	+2.17	+3.95	+2.10 [+1.12, +3.11]*	+1.81*	+0.19
TOVA	RTN W3-g128	512	+2.17	+2.87	+2.30 [+1.36, +3.28]*	+1.68*	+0.22
TOVA	RTN W3-g128	1024	+2.17	+2.24	+1.84 [+0.90, +2.80]*	+1.30*	+0.16
SnapKV	GPTQ W4-g128	256	-0.27	+4.14	-0.06 [-0.68, +0.57]	0.00	-0.25
SnapKV	GPTQ W4-g128	512	-0.27	+2.17	+0.05 [-0.35, +0.44]	0.00	-0.13
SnapKV	GPTQ W4-g128	1024	-0.27	+0.95	+0.21 [-0.13, +0.55]	+0.11	+0.07
SnapKV	RTN W4-g128	256	+0.07	+4.14	+0.09 [-0.62, +0.82]	+0.16	+0.30
SnapKV	RTN W4-g128	512	+0.07	+2.17	+0.19 [-0.31, +0.71]	+0.08	+0.26
SnapKV	RTN W4-g128	1024	+0.07	+0.95	+0.31 [-0.04, +0.70]	+0.06	+0.03
SnapKV	RTN W4-per-ch.	256	+0.91	+4.14	+1.30 [+0.20, +2.45]*	+1.06*	+0.91
SnapKV	RTN W4-per-ch.	512	+0.91	+2.17	+1.02 [+0.25, +1.87]*	+0.49	+0.55
SnapKV	RTN W4-per-ch.	1024	+0.91	+0.95	+0.89 [+0.20, +1.61]*	+0.32	+0.45
SnapKV	RTN W3-g128	256	+2.17	+4.14	+1.64 [+0.46, +2.92]*	+1.06*	+1.68
SnapKV	RTN W3-g128	512	+2.17	+2.17	+1.55 [+0.57, +2.63]*	+0.97*	+1.17
SnapKV	RTN W3-g128	1024	+2.17	+0.95	+1.38 [+0.47, +2.41]*	+0.77*	+0.59
PyramidKV	GPTQ W4-g128	1024	-0.27	+3.15	+0.62 [-0.06, +1.36]	+0.24	+0.46
PyramidKV	RTN W4-g128	1024	+0.07	+3.15	+0.63 [+0.05, +1.27]*	+0.26	+0.28
PyramidKV	RTN W4-per-ch.	1024	+0.91	+3.15	+0.87 [-0.10, +1.86]	+0.69*	+0.71
PyramidKV	RTN W3-g128	1024	+2.17	+3.15	+1.70 [+0.46, +3.02]*	+1.04*	+1.63

G.7 Two build-side controls: the calibration corpus, and AWQ's clipping search

App. E names two properties of our builds that bear on the quantizer contrast. First, GPTQ's Hessians and AWQ's activation statistics are estimated on WikiText-2 and the needle haystacks *are* WikiText-2 prose, so the calibrated arm has seen the evaluation text distribution and the data-free RTN arm has not, and the paper's second claim contrasts exactly those two arms. Second, reference AWQ searches a per-group clipping range as well as the scales, and ours did not, while the paper calls AWQ a shipped recipe. Both are tested here (Table 19): `gptqc4_*` is the same algorithm at the same bit budget with its calibration drawn from C4, which shares nothing with the haystacks, and `awqclip_*` adds the clipping search. Everything else (items, budgets, policy, seeds) is held fixed, and the comparison is against the same 64 items the base builds were measured on ($B \in \{64, 96, 160, 256\}$, both tasks, SnapKV; 24 matched cells, 15 in band for both members of a pair).

The calibration corpus does not carry the effect. If the WikiText-2 overlap were doing the work, moving the calibration to C4 should push the calibrated builds towards the data-free ones, whose median Δ_{int} is +0.220 nats. Paired within cell it does not: over the 10 in-band GPTQ pairs the median difference is -0.030 nats ($p = 1.00$, signed rank), and the two GPTQ builds move in opposite directions, W4-g128 by -0.035 and W3-g128 by +0.124 nats, the latter towards the data-free value. The significant-positive count goes 2 to 3 of 10, and unpaired the medians move +0.007 to +0.062, 25%

Table 15: **The full noise ladder** (Qwen2.5-7B, 4k, single needle, SnapKV at $B = 96$, 128 items), ordered by the calibration noise σ_{cal}. σ and $\hat{D}$ are the evaluation-time noise and the bound's predicted displaced mass, measured on this grid's own items at $B = 96$; they therefore differ from Table 16, which reads σ off the main grid at $B = 256$, by at most 0.002 except for AWQ W4-g128 (0.005), which is additionally the seeded build here against the unseeded build there. $^*/^\dagger$: the oracle term's interval excludes zero from above/below.

Quantizer	bits/param	σ_{cal}	σ	$\hat{D}$	Δ_{oracle} [95% CI]
RTN W6-g128	6.17	0.054	0.103	0.0148	-0.06 [-0.26, +0.13]
RTN W5-g32	5.66	0.069	0.140	0.0197	-0.08 [-0.30, +0.13]
RTN W5-g64	5.33	0.080	0.155	0.0217	+0.01 [-0.25, +0.27]
RTN W6-per-ch.	6.00	0.089	0.164	0.0230	-0.16 [-0.43, +0.09]
RTN W5-g128	5.16	0.091	0.180	0.0254	-0.11 [-0.36, +0.12]
GPTQ W4-g64	4.31	0.112	0.217	0.0316	-0.06 [-0.23, +0.11]
RTN W4-g16	5.25	0.113	0.224	0.0329	-0.21 [-0.44, -0.02]†
GPTQ W4-g128	4.16	0.126	0.237	0.0349	+0.34 [+0.01, +0.69]*
RTN W4-g32	4.62	0.129	0.259	0.0388	-0.06 [-0.29, +0.16]
AWQ W4-g64	4.31	0.133	0.253	0.0378	-0.10 [-0.33, +0.11]
RTN W4-g64	4.31	0.151	0.282	0.0429	-0.01 [-0.25, +0.22]
RTN W5-per-ch.	5.00	0.157	0.290	0.0439	-0.09 [-0.34, +0.15]
AWQ W4-g128	4.16	0.159	0.279	0.0421	-0.29 [-0.58, -0.05]†
RTN W4-g128	4.16	0.170	0.328	0.0518	-0.16 [-0.38, +0.04]
GPTQ W3-g64	3.30	0.220	0.399	0.0680	-0.02 [-0.28, +0.25]
GPTQ W4-per-ch.	4.00	0.227	0.404	0.0682	+0.20 [-0.28, +0.73]
GPTQ W3-g128	3.15	0.253	0.447	0.0783	+0.06 [-0.23, +0.36]
RTN W3-g32	3.59	0.265	0.490	0.0876	+0.41 [+0.11, +0.75]*
AWQ W3-g64	3.30	0.274	0.493	0.0890	-0.17 [-0.40, +0.01]
AWQ W4-per-ch.	4.00	0.298	0.523	0.0956	+0.55 [+0.15, +0.98]*
RTN W3-g64	3.30	0.303	0.562	0.1043	+0.35 [+0.12, +0.59]*
AWQ W3-g128	3.15	0.307	0.537	0.0991	+0.15 [-0.17, +0.48]
RTN W4-per-ch.	4.00	0.309	0.560	0.1036	+0.42 [+0.02, +0.88]*
RTN W3-g128	3.15	0.331	0.622	0.1187	+1.02 [+0.52, +1.53]*

Table 16: **The six main-grid quantizers**: evaluation-time score noise σ (not the calibration screen σ_{cal} of Table 2), the fraction of SnapKV's top-256 retain set that survives quantization, and full-cache *tail* exact match on each needle task, i.e. after the first answer token—not the full-string check of Table 24 (Qwen2.5-7B, 4k; σ and overlap are measured at $B = 256$ on the single needle; the AWQ row is the unseeded build).

Quantizer	bits	σ	overlap	EM single	EM distr.
bf16 (reference)	16.00	–	1.00	1.00	1.00
GPTQ W4-g128	4.16	0.238	0.92	1.00	1.00
AWQ W4-g128	4.16	0.284	0.90	1.00	1.00
RTN W4-g128	4.16	0.330	0.89	1.00	1.00
GPTQ W3-g128	3.15	0.448	0.85	0.98	0.98
RTN W4-per-ch.	4.00	0.561	0.81	1.00	0.98
RTN W3-g128	3.15	0.622	0.78	1.00	1.00

of the way to the data-free value; the unpaired shift is not significant at this n, but it is not negligible either. The C4-calibrated builds are slightly *noisier* on the calibration screen (σ_{cal} 0.126→0.142 at W4-g128 and 0.253→0.278 at W3-g128) and slightly worse on WikiText-2 perplexity, which is what one expects when the calibration corpus no longer matches the perplexity corpus; that is a fraction of the distance to the data-free builds (Table 15). The confound is present, and at this power we cannot rule out that it carries part of the unpaired gap.

The clipping search does not change AWQ. With the reference per-group clipping range searched, AWQ W4-g128's median Δ_{int} goes +0.011 to +0.016 nats (paired median +0.005), its σ_{cal} 0.159 to 0.160, and its WikiText-2 perplexity gap +0.29 to +0.33. The build we call a shipped AWQ

Table 17: **Protection versus placebo versus more bits, both bit-widths** (Qwen2.5-7B, 4k, single needle, SnapKV at 96 tokens, 64 items); the 3-bit block alone is Table 3.

Scheme	bits	Δ_{oracle} [95% CI]	Δ_{int}	EM
RTN W4 per-ch.	4.00	+0.54 [-0.05, +1.22]	+0.93	0.72
+ W_K@16	4.10	+0.90 [+0.24, +1.63]	+1.23	0.64
+ W_V@16 (placebo)	4.10	-0.01 [-0.57, +0.55]	+0.38	0.81
RTN W4 g64 (uniform)	4.31	+0.01 [-0.38, +0.42]	-0.01	0.92
RTN W3 g128	3.15	+1.39 [+0.64, +2.17]	+2.00	0.55
+ W_K@16	3.25	+0.69 [+0.03, +1.40]	+0.96	0.78
+ W_V@16 (placebo)	3.25	+0.84 [+0.42, +1.29]	+0.90	0.67
RTN W3 g64 (uniform)	3.30	+0.59 [+0.23, +1.01]	+0.66	0.72

Table 18: **Specificity at the matched 7B setting** (4k, SnapKV, 96 tokens): interaction and oracle terms on the single needle, on variable tracking (outside the band: exact match 0.00 even for bf16 with a full cache) and on the plain language-modelling control.

	single needle		variable tracking		plain LM	
Quantizer	Δ_{int}	Δ_{oracle}	Δ_{int}	Δ_{oracle}	Δ_{int}	Δ_{oracle}
GPTQ W4-g128	+0.57	+0.43	+1.52	+0.55	+0.17	-0.12
AWQ W4-g128	-0.14	-0.24	+1.76	-0.28	-1.31	-1.25
RTN W4-g128	-0.16	0.00	+1.49	+0.77	+0.10	-0.19
GPTQ W3-g128	-0.19	-0.16	+2.04	-2.52	+4.49	+0.36
RTN W4-per-ch.	+0.93	+0.54	+5.99	-1.85	+1.19	-0.04
RTN W3-g128	+2.00	+1.39	+13.00	-3.50	-2.96	-0.34

therefore behaves like the one with the search. The second deviation, activation statistics from the first calibration sequence rather than all of them, remains open.

Neither control is powered to detect a small shift: at 64 items per cell the intervals are wide—they overlap in all 15 pairs, and in 13 of 15 each member's estimate also lies inside the other's interval. Pooled over the 15 pairs the paired shift is +0.024 nats [-0.055, +0.144] (Hodges–Lehmann, signed-rank interval), which excludes the +0.21 nats either mechanism would have to supply to close the calibrated-versus-data-free gap on its own. Each control taken alone is five pairs per build and excludes nothing: at $n = 5$ the signed-rank test cannot reach $p < 0.05$ at any shift. The pooled interval is the one claim here, and it mixes the two mechanisms.

Table 19: **The two build-side controls.** Qwen2.5-7B at 4k, SnapKV, 64 items, $B \in \{64, 96, 160, 256\}$, both needle tasks; medians over the 5 in-band cells of each pair. "ΔNLL alone" is the quantizer's full-cache cost on the single needle. "as run" is the build the paper reports everywhere else. σ_{cal} is the score-noise calibration screen of §5.5; "ΔNLL alone" is the quantizer's full-cache cost; "sig. pos." counts cells whose Δ_{int} interval excludes zero from above.

Build	variant	σ_{cal}	ΔNLL alone	median Δ_{int}	sig. pos.
GPTQ W4-g128	as run	0.126	-0.003	+0.077	1/5
	+ C4 calibration	0.142	-0.004	+0.045	1/5
GPTQ W3-g128	as run	0.253	+0.014	-0.091	1/5
	+ C4 calibration	0.278	+0.021	+0.108	2/5
AWQ W4-g128	as run	0.159	-0.004	+0.011	1/5
	+ clipping search	0.160	-0.002	+0.016	0/5

G.8 LongBench on the second model family

The real-document result of §5.4 is one model. This repeats the design on Llama-3.1-8B with a reduced grid: the same three English QA subsets, the same length window, 120 items, 3 of the four quantizers (GPTQ W4-g128, RTN W4-per-channel and RTN W3-g128; AWQ and RTN W4-g128

were not run on Llama), 3 of the five policies (SnapKV, TOVA and PyramidKV; the two that never enter the band on Qwen were not run) at $B \in \{256, 512, 1024\}$, without the $B = 128$ rung, with the deployment-faithful loss and the same loss-ratio band. Every one of the 27 cells is in band here (the full-cache loss is 5.18 nats and no single-axis condition exceeds the ratio, the largest being 1.49×), so nothing is excluded (Table 20).

Δ_{int} is significantly positive in 23 of the 27 cells and negative in 0, median +0.87 nats. It holds in every policy (7 of 9 under SnapKV, 7 of 9 under TOVA, 9 of 9 under PyramidKV) and in every quantizer, including the one shipped calibrated recipe in the grid: GPTQ W4-g128 costs +0.24 nats alone on these documents and is significantly super-additive in 5 of its 9 cells, at up to +0.78 [+0.20, +1.37]. The noisy builds are larger, as everywhere: RTN W4-per-channel 9 of 9 (median +0.87) and RTN W3-g128 9 of 9 (median +1.63).

The effect here is not only a tail: over the 27 cells, 17 10%-trimmed intervals exclude zero and 17 reach signed-rank $p < 0.05$. The Qwen grid of §5.4 is comparable (18 of 28 trimmed); what is not is the large-budget grid of App. G.9, where only 2 of 12 survive trimming. The robust survivors are the noisy builds: restricted to the 9 GPTQ W4-g128 cells, 0 trimmed intervals exclude zero (trimmed -0.07–+0.25 nats). So on the second family the *phenomenon* is broad and not tail-driven, while the claim about a shipped recipe remains a claim about means, exactly as it is on Qwen.

Is the shipped recipe's result a tail, or was it underpowered? At 120 items GPTQ W4-g128 (the one calibrated 4-bit recipe in the grid, and the arm that carries the paper's coverage claim) is significant on the mean in 5 of its 9 cells and on no tail-insensitive estimator at all. Those two readings separate at a larger n, and a larger n is available: the length window admits far more than 120 documents and the loader's round-robin is nested, so item i names the same document at 288 items as at 120. We re-ran the bf16 and GPTQ arms there (9 cells: 3 policies at $B \in \{256, 512, 1024\}$; Table 21). The count was power-limited. Pooled at 288 items the interaction is significantly positive in 9 of 9 cells by the mean, 8 by a 10%-trimmed mean and 8 by a paired signed-rank test; on the 168 items no earlier grid contains, the only out-of-sample column, it is 8, 8 and 8 of 9. The median trimmed estimate goes +0.065 at 120 items to +0.144 nats at 288, while the median mean barely moves (+0.309 to +0.357). Two things are in that: the count rose because the trimmed estimator had too few documents at 120 to detect a real, broad shift, and the trimmed *magnitude* rose because the 168 held-out documents differ from the first 120 (held-out trimmed median +0.155), which is an item-set difference and not a power effect. The single cell whose trimmed interval still covers zero is tova at $B = 1024$, the largest budget of the least-averaged policy, where the mean itself is smallest.

This bears on whether the effect is confined to quantizers nobody ships. On real documents, on the second model family, under three eviction policies and at every budget we ran, a shipped GPTQ W4-g128 build costs more deployed than the sum of what it costs alone and what eviction costs alone—and in 8 of the 9 it survives an estimator that discards the tail.

Two limits. Eviction alone is not cheap at these budgets (+0.38–+2.54 nats on a 5.18-nat baseline), so this is evidence about super-additivity and not about "free alone" (App. G.9 is the budget for that). And the band admits every cell here because this grid omits the two policies and the smallest budget that fail it on Qwen, not because the band is looser: as a ratio it is tighter on Llama in absolute nats (0.5×5.18 against 0.5×9.66), and the worst cell, at 1.49×, only just clears it; the identifiability argument is unchanged but it does no filtering in this grid.

G.9 LongBench at a budget where eviction alone is cheap

The LongBench grid of §5.4 runs $B \in \{128, \ldots, 1024\}$ on documents of 5.0–16.9k tokens, and at those budgets eviction alone is not cheap: the clean policy already pays 10–43% of the full-cache loss (+0.95–+4.14 nats of 9.66). That is a fair objection to reading the result as evidence about a deployed system, because a deployment would not ship a cache budget that costs it a third of its loss. The budget it would ship is higher, so we ran the same three subsets, the same 120 items and the same length window at $B \in \{2048, 3072\}$ (Table 22).

Eviction alone is then cheap under SnapKV: its clean eviction costs +0.06–+0.14 nats, which is 0.7–1.4% of the 9.66-nat full-cache loss. TOVA's costs +0.89–+1.55 nats (9.2–16.0%), which is still not cheap: it does fall as B rises, but TOVA ranks on a single query and pays more than SnapKV at every budget we ran, so the "free alone" reading below is a statement about the SnapKV rows

Table 20: **LongBench on Llama-3.1-8B.** 120 items over the same three English QA subsets and length window as §5.4, full-cache loss 5.18 nats. "trimmed" is the 10%-trimmed mean of the same per-item contrast with its own bootstrap interval. * on either column: that column's 95% interval excludes zero from above; no cell is significantly negative.

policy	build	B	ΔNLL quant. alone	eviction alone	Δ_{int}	trimmed
snapkv	GPTQ W4-g128	256	+0.24	+2.07	+0.57 [+0.14, +1.03]*	+0.18
snapkv	GPTQ W4-g128	512	+0.24	+0.99	+0.31 [0.00, +0.65]	+0.02
snapkv	GPTQ W4-g128	1024	+0.24	+0.38	+0.11 [-0.04, +0.27]	+0.03
snapkv	RTN W3-g128	256	+1.66	+2.07	+1.86 [+1.15, +2.60]*	+1.16*
snapkv	RTN W3-g128	512	+1.66	+0.99	+1.63 [+0.90, +2.42]*	+0.79*
snapkv	RTN W3-g128	1024	+1.66	+0.38	+0.92 [+0.42, +1.51]*	+0.37*
snapkv	RTN W4-per-ch.	256	+0.81	+2.07	+1.07 [+0.45, +1.78]*	+0.48*
snapkv	RTN W4-per-ch.	512	+0.81	+0.99	+0.66 [+0.25, +1.14]*	+0.25*
snapkv	RTN W4-per-ch.	1024	+0.81	+0.38	+0.41 [+0.11, +0.73]*	+0.12
tova	GPTQ W4-g128	256	+0.24	+1.46	+0.46 [+0.12, +0.84]*	+0.20
tova	GPTQ W4-g128	512	+0.24	+1.06	+0.17 [-0.09, +0.46]	+0.04
tova	GPTQ W4-g128	1024	+0.24	+0.75	+0.04 [-0.18, +0.28]	-0.07
tova	RTN W3-g128	256	+1.66	+1.46	+2.64 [+1.79, +3.60]*	+1.58*
tova	RTN W3-g128	512	+1.66	+1.06	+1.88 [+1.11, +2.75]*	+1.06*
tova	RTN W3-g128	1024	+1.66	+0.75	+1.30 [+0.60, +2.10]*	+0.68*
tova	RTN W4-per-ch.	256	+0.81	+1.46	+1.36 [+0.75, +2.05]*	+0.72*
tova	RTN W4-per-ch.	512	+0.81	+1.06	+1.12 [+0.55, +1.78]*	+0.46*
tova	RTN W4-per-ch.	1024	+0.81	+0.75	+0.71 [+0.23, +1.30]*	+0.23*
pyramidkv	GPTQ W4-g128	256	+0.24	+2.54	+0.78 [+0.20, +1.37]*	+0.25
pyramidkv	GPTQ W4-g128	512	+0.24	+1.47	+0.50 [+0.09, +0.94]*	+0.14
pyramidkv	GPTQ W4-g128	1024	+0.24	+0.63	+0.28 [+0.05, +0.52]*	+0.06
pyramidkv	RTN W3-g128	256	+1.66	+2.54	+2.29 [+1.40, +3.23]*	+1.49*
pyramidkv	RTN W3-g128	512	+1.66	+1.47	+1.55 [+0.85, +2.31]*	+0.80*
pyramidkv	RTN W3-g128	1024	+1.66	+0.63	+1.21 [+0.55, +2.01]*	+0.52*
pyramidkv	RTN W4-per-ch.	256	+0.81	+2.54	+1.41 [+0.73, +2.16]*	+0.69*
pyramidkv	RTN W4-per-ch.	512	+0.81	+1.47	+0.87 [+0.35, +1.50]*	+0.33*
pyramidkv	RTN W4-per-ch.	1024	+0.81	+0.63	+0.65 [+0.25, +1.09]*	+0.31*

Table 21: **The shipped recipe at more than twice the items.** GPTQ W4-g128 on Llama-3.1-8B LongBench, the same three subsets and length window, 288 items. *Held-out*: the 168 documents no earlier grid contains. "trimmed" is the 10%-trimmed mean of the same per-item contrast. *: 95% interval excludes zero from above.

		original ($n = 120$)		held-out ($n = 168$)		pooled ($n = 288$)	
policy	B	mean	trimmed	mean	trimmed	mean	trimmed
pyramidkv	256	+0.78*	+0.25	+0.37	+0.31*	+0.54*	+0.27*
pyramidkv	512	+0.50*	+0.14	+0.39*	+0.22*	+0.44*	+0.19*
pyramidkv	1024	+0.28*	+0.06	+0.41*	+0.14*	+0.36*	+0.11*
snapkv	256	+0.57*	+0.18	+0.55*	+0.35*	+0.56*	+0.28*
snapkv	512	+0.31	+0.02	+0.33*	+0.16*	+0.32*	+0.10*
snapkv	1024	+0.11	+0.03	+0.35*	+0.14*	+0.25*	+0.09*
tova	256	+0.46*	+0.20	+0.38*	+0.15*	+0.41*	+0.17*
tova	512	+0.17	+0.04	+0.46*	+0.22*	+0.34*	+0.14*
tova	1024	+0.04	-0.07	+0.23*	+0.07	+0.15*	+0.01

only. RTN W4-g128 costs +0.07 nats alone on the same documents. Both compressions are therefore individually almost free, and they still do not compose: at $B = 2048$ under SnapKV the deployed cell is +0.49 [+0.14, +0.92] nats worse than the sum of the two single-axis costs—an interaction seven to eight times the cost of either axis alone. Over the 12 cells, 5 are significantly positive and 0 negative. It is the paper's only instance of the pattern on real documents at a budget where *both* compressions are individually near-free. By the paper's own robustness criterion it is also the weakest of the LongBench results, which the next paragraph states.

Two limits. Four of the five positives are RTN W3-g128, the noisiest build in the grid, and the fifth is RTN W4-g128; GPTQ W4-g128, the quietest, is not significant in any of its four cells—the same magnitude ordering as everywhere else (§5.1). And at these budgets the effect is almost entirely a tail. Over all 12 in-band cells, 2 10%-trimmed intervals exclude zero (both RTN W3-g128 under TOVA) and none reaches signed-rank $p < 0.05$; over the 8 shipped-build cells, 0 do (trimmed 0.00–+0.22 nats against means up to +0.49). A handful of documents are badly hurt and the rest are untouched. Since Δ_{int} is a difference of mean losses and the mean is what a deployment pays, we report it as a real effect on the quantity that matters and say plainly that it is not a broad shift; it is the weakest of the LongBench results by the paper's own robustness criterion, and the abstract's quotation of its mean and interval should be read with that.

Table 22: **LongBench where eviction alone is cheap.** Qwen2.5-7B, 120 items over the same three English QA subsets and length window as §5.4, deployment-faithful loss, full-cache loss 9.66 nats. "ΔNLL quant. alone" is the quantizer's cost with a full cache and "eviction alone" the clean model's cost at that budget; "trimmed" is the 10%-trimmed mean of the per-item contrast with its own bootstrap interval. */†: that column's 95% paired-bootstrap interval excludes zero from above/below.

policy	build	B	ΔNLL quant. alone	eviction alone	Δ_{int}	trimmed
snapkv	GPTQ W4-g128	2048	-0.27	+0.06	+0.13 [-0.13, +0.41]	+0.04
snapkv	GPTQ W4-g128	3072	-0.27	+0.14	+0.01 [-0.23, +0.23]	+0.01
snapkv	RTN W3-g128	2048	+2.17	+0.06	+0.96 [+0.36, +1.66]*	+0.26
snapkv	RTN W3-g128	3072	+2.17	+0.14	+0.57 [+0.14, +1.06]*	+0.06
snapkv	RTN W4-g128	2048	+0.07	+0.06	+0.49 [+0.14, +0.92]*	+0.06
snapkv	RTN W4-g128	3072	+0.07	+0.14	+0.12 [-0.17, +0.41]	0.00
tova	GPTQ W4-g128	2048	-0.27	+1.55	+0.31 [-0.05, +0.68]	+0.22
tova	GPTQ W4-g128	3072	-0.27	+0.89	+0.12 [-0.22, +0.51]	0.00
tova	RTN W3-g128	2048	+2.17	+1.55	+1.13 [+0.25, +2.04]*	+0.70*
tova	RTN W3-g128	3072	+2.17	+0.89	+1.15 [+0.25, +2.12]*	+0.50*
tova	RTN W4-g128	2048	+0.07	+1.55	+0.37 [-0.09, +0.88]	+0.19
tova	RTN W4-g128	3072	+0.07	+0.89	+0.17 [-0.28, +0.69]	+0.01

G.10 Two selection-side controls

The rung the grid did not have. App. B discloses that the distractor task first enters the identifiability band at $B = 160$, one rung above its own knee, because the main grid has no budget between 96 and 160. We ran it: the same 6 quantizers, both tasks, $B \in \{96, 128, 160, 256\}$, 64 items, SnapKV (48 cells; Table 23). At $B = 128$ the clean policy's exact match on the distractor task is 0.36 and eviction alone already costs +6.02 nats, so the rung sits *below* the band, not inside it: 0 of its cells are identifiable, and the band's exclusion of it is the band working, not a gap in the grid. The single-needle column of the same table is the sanity check (there every budget from 96 up is in band, with clean exact match 0.89 to 0.97), and it also shows the interaction is not monotone in the budget: the median Δ_{int} is +0.202 at $B = 96$, +0.029 and +0.020 in the middle, and +0.115 at $B = 256$. We do not have an account of that shape and do not build on it.

The tie-break. SnapKV max-pools its score with kernel 7 before ranking, which replicates a local maximum across up to seven consecutive positions and so puts an exact tie at the top-B boundary in most heads—at $B = 160$, 181 of 224 (layer, head) selections (App. C). Where the boundary is a tie, which position is retained is settled by the top-k implementation's index order rather than by the score, and a reader is entitled to ask whether the interaction is an artefact of that tie-break rather than of the quantizer's effect on the score. The control ranks the *same* score without the max-pool (`snapkv_nopool`, "no-pool" below; "pooled" in this paragraph means max-pooled, not the item-pooled columns of Tables 8–13), which removes the ties. It is not a free change: pooling is there because it helps, and without it the clean policy is much worse—bf16 evict-only exact match peaks at 0.92 against 0.98 pooled, and on the distractor task it never clears the band at all (0.36 at best). Only 12 of the 48 matched cells are in band for the no-pool rule against 36 for the max-pooled one, so the comparison below is on the single-needle task at $B \in \{160, 256\}$ and nowhere else. Over the 12 cells in band for both policies (of 48 matched; $B \in \{96, 128, 160, 256\}$, 64 items) the max-pooled policy gives 3 significantly positive and 0 negative at a median Δ_{int} of +0.072 nats, and the no-pool policy 7 and 0 at a median of +0.536; the paired difference has median +0.508 nats and

Table 23: **The missing rung.** Qwen2.5-7B at 4k, SnapKV, 64 items, the 6 quantizers of the main grid. "clean EM" is the bf16 evict-only exact match that the band thresholds at 0.5 and Δ_{E} is what eviction alone costs the bf16 model. $B = 128$ is the rung the main grid lacks; on the distractor task it is below the band.

Task	B	clean EM	Δ_{E}	in band	median Δ_{int}
single needle	96	0.89	+0.41	6/6 (3 pos.)	+0.202
single needle	128	0.94	+0.22	6/6 (1 pos.)	+0.029
single needle	160	0.95	+0.17	6/6 (1 pos.)	+0.020
single needle	256	0.97	+0.03	6/6 (2 pos.)	+0.115
distractor	96	0.12	+10.04	0/6	–
distractor	128	0.36	+6.02	0/6	–
distractor	160	0.64	+3.20	6/6 (0 pos.)	+0.003
distractor	256	0.98	+0.02	6/6 (3 pos.)	+0.440

the sign of Δ_{int} agrees in 8 of 12 cells. Removing the ties therefore makes the interaction *larger*, not smaller: the no-pool estimate is the larger of the two in 11 of 12 cells ($p = 0.0010$, signed rank). The pooling is suppressing the interaction, which is what one would expect of an operation that smooths the score before the comparison. Two limits on how far this can be pushed: the no-pool variant is a diagnostic, not a recipe anyone ships, so its larger interaction is not a claim about deployed systems; and it is a different selection rule, so this is a comparison between two policies and not a within-policy ablation of the tie-break alone. What it rules out is the specific alternative explanation, that index-order tie-breaking manufactures the effect, since the effect survives, and grows, when the ties are gone.

G.11 Further disclosures

The exchange rate. One reading of the whole effect is a budget shift: quantization costs the policy effective cache, so any decreasing loss-versus-budget curve makes $\Delta_{\mathrm{int}} > 0$ where it is steep and near zero where it has saturated, which is the shape of Fig. 2; the square adds what the shift alone cannot say, namely which part of it is the selection's. Priced in the currency a practitioner controls, the interaction costs a median $1.91\times$ the clean model's budget for RTN W4 per-channel and $1.23\times$ for GPTQ W4-g128, against $0.66\times$ for RTN W4-g128, whose interaction here is not significant. The remedy is small: 120 extra cache tokens at $B = 160$ (ratio $1.75\times$ at that reference budget; the medians above are over all reference budgets), less than 4-bit weights save below 1,664 concurrent sequences. Single-axis evaluation does not reveal the need for it. Fig. 5 (right) inverts the loss-versus-budget curve on the 4k single needle: for each reference budget B at which the clean policy is in band, we find the smallest budget at which the deployed system reaches the clean model's loss at B, interpolating in $\log_2 B$ over the ten-budget sweep of the dense grid. Reference budgets at which the clean curve has already saturated (within 0.05 nats of its own floor) are excluded, because inverting a flat curve amplifies noise, and references the deployed system never reaches within the swept range are dropped, so the reported ratios are lower bounds. The sweep is SnapKV only. Break-even batch sizes assume a bf16 KV cache of $28 \times 4 \times 128 \times 2 \times 2$ bytes per token and a weight saving of $(16 - b)/8$ bytes per parameter over 7.62B parameters.

$\mathrm{H_2O}$-norm at large budgets. Because $\mathrm{H_2O}$-norm never entered the band at the needle-task budgets calibrated to SnapKV, we ran a separate grid at $B \in \{512, 768, 1024, 1536\}$—up to 40% of the 3841-token context. It still does not: at $B = 1536$ the bf16 evict-only exact match is 0.09 on the single needle and 0.25 on the distractor task, both below the 0.5 threshold. On LongBench it does not clear the loss ratio either ($1.715\times$ at its best budget against a $1.5\times$ threshold), so the accumulated-attention family is outside the identifiability argument at every budget we ran, on both kinds of task. Since this exclusion is load-bearing, we state its scope exactly. It is a statement about post-prefill accumulated attention, not about published $\mathrm{H_2O}$, which evicts greedily at every decoding step and which we did not implement. It is *not* an artefact of our normalisation: the saturated first grid also ran the un-normalised accumulated score, and on the 4k single needle with a bf16 model at half the cache the two are indistinguishable (exact match 0.12 at 17.7 nats un-normalised against 0.14 at 17.8 normalised, on 80 items), while random eviction at the same budget reaches 0.25 at 7.0 and SnapKV 1.00 at 0.0. A post-prefill accumulated-attention selection is beaten by chance at

half the cache in this harness, which is why the family never enters the band; whether a per-step implementation behaves differently is exactly the question we cannot answer here.

Full-string exact match. The primary metrics exclude the first answer token, which is predicted from the prompt-final logits before any eviction, so that a cell contrasts the eviction and not the prefill. Full-string exact match with a *full* cache is the strict check on "lossless alone", and Table 24 reports it for every (model, context, task, quantizer), not for one setting: which quantizers pass is strongly setting-dependent (RTN W3-g128 is at 0.67 on Qwen2.5-7B at 4k but at bf16's own value at 32k, and GPTQ W3-g128 matches bf16 at 16k), which is why §5.1 applies the test cell by cell rather than importing it, and App. G.1 takes up what a failing row does and does not license one to conclude about Δ_{int} there. On Qwen2.5-7B at 4k specifically, RTN W4-g128 (1.00) exceeds bf16 (0.99) and GPTQ W4-g128 (0.98) is within one item of it, while AWQ W4-g128 (0.83), RTN W4 per-channel (0.61) and RTN W3-g128 (0.67) are clearly below. Every value in the table comes from the single canonical grid for that setting, not from a merge across files: merging would average the unseeded and seeded AWQ builds and report an accuracy no build has (the two differ by 14 points on this row).

Table 24: **Full-string exact match with a full cache**, every setting. Each entry is the quantizer's value followed by bf16's on the *same* paired items, bold when the quantizer is strictly informative and at least bf16 there—the gate §5.1 applies; the Llama rows are $0.00/0.00$ throughout for the tokenizer reason below and are never bold, since a tie at zero is not a pass. The "bf16 (all)" column is bf16 over all of its own items and is a reference only, not the comparison. "–": not run in that setting. Llama-3.1-8B is uninformative throughout for the tokenizer reason below, so no Llama cell supports the "lossless alone" half of the claim.

Model	Ctx	Task	bf16 (all)	GPTQ W4-128	GPTQ W3-128	AWQ W4-128	RTN W4-128	RTN W4-pc	RTN W3-128
Qwen2.5-7B	16k	single	1.00	**1.00**/1.00	**1.00**/1.00	–	–	0.81/1.00	0.94/1.00
Qwen2.5-7B	16k	distractor	0.88	**0.94**/0.88	**0.97**/0.88	–	–	0.38/0.88	0.75/0.88
Qwen2.5-7B	32k	single	1.00	**1.00**/1.00	–	–	–	0.94/1.00	**1.00**/1.00
Qwen2.5-7B	32k	distractor	0.69	**0.81**/0.69	–	–	–	0.38/0.69	0.62/0.69
Qwen2.5-7B	4k	single	0.99	**0.98**/0.98	**0.98**/0.98	0.83/0.98	**1.00**/0.99	0.61/0.98	0.67/0.98
Qwen2.5-7B	4k	distractor	0.89	**1.00**/0.86	**0.98**/0.86	0.83/0.86	**0.94**/0.89	0.20/0.86	0.47/0.86
Llama-3.1-8B	16k	single	0.00	–	–	–	0.00/0.00	0.00/0.00	0.00/0.00
Llama-3.1-8B	16k	distractor	0.00	–	–	–	0.00/0.00	0.00/0.00	0.00/0.00
Llama-3.1-8B	4k	single	0.00	0.00/0.00	–	–	–	0.00/0.00	0.00/0.00
Llama-3.1-8B	4k	distractor	0.00	0.00/0.00	–	–	–	0.00/0.00	0.00/0.00
Qwen2.5-3B	4k	single	1.00	–	–	–	**1.00**/1.00	0.00/1.00	0.00/1.00
Qwen2.5-3B	4k	distractor	0.94	–	–	–	0.69/0.94	0.00/0.94	0.00/0.94

On Llama-3.1-8B the tokenizer makes the first generated token differ from the gold string for every spec including bf16, so full-string exact match is 0.00 throughout. Token accuracy there is 0.750 for bf16 and for all three quantizers with a full cache, but that value is exactly $3/4$ on a four-token answer whose first token is always wrong, which restates the tail metric rather than checking it, so we have *no* independent full-cache verification on that family, and no Llama cell is offered as evidence for the "lossless alone" claim. The claim rests on the Qwen rows, where the check is available.

Reproducibility of the quantizer builds, and which AWQ build is canonical. GPTQ and RTN builds are deterministic given the calibration file; the AWQ scale search subsamples activations, and in the first AWQ build that subsample was drawn from an unseeded generator, so two AWQ W4-g128 builds differ by up to 0.43 nats on a cell mean and by 14 points of full-cache full-string exact match (0.83 against 0.97). Later builds seed the generator. Precedence keeps the first file a cell appears in, which for AWQ is the unseeded build in `grid_m7_4kb`; every later grid that contains AWQ (`grid_s1_pol`, `grid_s4_lad`, `grid_s30_op256`) holds the seeded build and the three agree with each other bit-for-bit on shared items. As a result, AWQ contributes 5 of the 143 union cells (1 positive, 1 negative) and one row of Table 1, and those values, like Table 16's AWQ row and Table 24's Qwen 4k entry (the 0.83 of the 0.83-against-0.97 pair), are the unseeded build, the canonical one by precedence, while Table 2's and Table 15's AWQ rungs are the seeded one. In Table 8 the AWQ rows compare the two (the *original* column is the unseeded build and the other two the seeded one), so those rows (five in band, three out) are a build comparison as much as a power comparison and should be read as such.

Of the 101 identifiable SnapKV cells, Δ_{oracle} is significantly positive in 32 and negative in 4, and Δ_{frag} in 39 and 1. Table 27 lists every identifiable SnapKV retrieval cell (101 rows) with its interaction, decomposition, oracle-quality gap and exact-match values. **Length or budget fraction.**

Table 25: **Every (model, context) setting, all six needle grids.** In-band SnapKV cells and the sign of Δ_{int} at nominal 95%. The first four rows are the grids of Table 27; the last two have their own cell files. The abstract's counts are the "All" row; after a Benjamini–Hochberg correction at 5% over all 143 cells they are 63 and 3, and after Bonferroni 40 and 0.

Model	Ctx	paired items	in-band cells	sig. > 0	sig. < 0	median Δ_{int}
Qwen2.5-7B	4k	64	48	16	1	+0.102
Qwen2.5-7B	16k	32	28	16	0	+0.475
Llama-3.1-8B	4k	48	21	13	0	+0.108
Qwen2.5-3B	4k	48	4	4	0	+0.864
Llama-3.1-8B	16k	24	24	16	2	+0.169
Qwen2.5-7B	32k	16	18	6	0	+0.281
All			143	71	3	+0.173

§5.1's pooled 16k-vs-4k comparison mixes lengths with budget fractions: the 4k budgets compared here ($B = 160$ and 256) are 4.2–6.7% of a 3841-token context while the 16k one ($B = 512$) is 3.4%. At matched fraction on the single needle, 3.4% of 16k gives median $\Delta_{\text{int}} = -0.006$ against +0.121 at 4.2% of 4k, and on the distractor task +0.475 against +0.689 at 6.7%—so length is not what drives the magnitude; distance to the policy's failure point is.

What this means for the recipes practitioners ship. Restricted to the three W4-g128 builds (GPTQ, AWQ and RTN, the configurations a deployment is most likely to use), the union has 47 cells, 8 significantly positive and 3 significantly negative on the released cell records (11 and 1 on the rebuilt item sets of `code/robust.py`, below), median +0.024 nats, against a median +0.337 for RTN W3-g128 and RTN W4-per-channel. §5.1 reads the same slice with eviction's own damage held down (clean exact match ≥ 0.9), which is the fairer comparison because the noisy builds are otherwise credited with budgets the quiet ones are never evaluated at: 5 of 31 at a median +0.011 against 34 of 44 at +0.217 (cell records; 7 of 31 rebuilt). For most of those 47 cells we cannot resolve the interaction at this n; where we can, it is a few hundredths of a nat at the free-alone points (App. G.6) and it can be negative at the operating budget (above). The strong form of the effect (tenths of a nat, in a majority of cells) belongs to the noisy quantizers. That is what C1 predicts and it is the limit of the paper's coverage: what generalises to shipped 4-bit recipes is the *design*, and the finding that single-axis evaluation cannot see either sign, not the magnitude we measure on noisy builds.

Is the split really exact? The split (1) is an identity per item, so it is an identity in the mean only when all three terms are taken over the same items. The released Δ_{int} is computed on the items the four physical cells share, which is a superset of the items the transplant arm also has; taking the interaction on the transplant-complete set instead (`d_int_c`) makes $\Delta_{\text{oracle}} + \Delta_{\text{frag}} = \Delta_{\text{int}}$ hold to machine precision in 532 of 532 cells (worst residual 6×10^{-14}). Against the wider set the worst discrepancy anywhere in the released file is 0.072 nats, in a variable-tracking cell at a fractional budget that is outside the band and is not reported.

Is the interaction a mean artefact? Δ_{int} is a mean of an unbounded per-item contrast, and in the cells that matter most items are unaffected: the three largest in absolute value carry a median 0.49 of the summed absolute contrast. `code/robust.py` reads the contrasts three ways over all 143 counted cells, and it rebuilds each contrast vector from the released records, which for 20 cells pools two long-context grids that the cell file keeps separate and so estimates on more items than the released cell does. Two of the union's three significantly negative cells are in that set, and they lose significance to the larger item set rather than to the estimator; we say so rather than counting them as artefacts of the mean. By the mean the union is 75 significantly positive and 1 negative of 143; by a 10%-trimmed mean with a bootstrap interval, 82 and 1; by a paired signed-rank test on the per-item contrasts, 106 and 0. The three are ordered by tail-insensitivity, so they are not three independent tests; the robust readings being more positive says that the bulk shift is broad and small while the tail is what makes a cell significant. The gap is largest where the effect is smallest. Restricting both families the same way, to cells whose clean policy still works: the shipped W4-g128 cells go 7 of 31 by the mean to 13 by signed rank, the noisy builds 34 of 44 to 41. Two things this does *not* license. The mean remains the estimand and the deployment-relevant quantity (the expected loss), so a robust estimator is a statement about detection, not a replacement; and it does not make every negative cell

an artefact. 1 cell is negative under both the mean and the trimmed estimator, AWQ W4-g128 on the distractor task at $B = 160$ (mean -0.98, trimmed -0.56 [-1.23, -0.11]), whose five most extreme items are -17.6 to -6.7 against $+8.0$ to $+0.8$: a real left tail, and exactly what Prop. 1's reading predicts once the retrievable information is gone. That cell is the unseeded AWQ build (*Reproducibility of the quantizer builds*, above); the seeded build gives -0.39 [-0.85, +0.08] on 176 held-out items and -0.44 [-0.84, -0.05] pooled (Table 8), so the sign is reproduced and the magnitude is not. The same estimators give the σ ladder's two sub-additive W4-g128 cells the profile of the positive tail cells rather than of a broad shift: AWQ W4-g128 has mean -0.270 with item median +0.00064 and trimmed -0.018 [-0.123, +0.016] over 128 items, and RTN W4-g128 mean -0.297 with signed-rank $p = 0.842$, while GPTQ W4-g128's +0.383 survives at +0.167 [+0.049, +0.358]. One rule for both signs: a cell significant by the mean whose trimmed interval covers zero is a tail-driven effect on the expected loss, whether it is these two, the positive tail cells of the union, or the LongBench cell of App. G.9; none is an artefact of the mean, and none is a broad shift. All three estimators are computed on the same items: taking the mean's interval from the released cell record instead, whose item set is a subset for a few cells, would make the comparison partly one between item sets.

Is the family contrast an artefact of unmatched conditions? The union's noisy and shipped cells are not evaluated at the same budgets, so a count comparison could reflect where each was measured. Matched within (model, context, task, budget) and comparing the two families' medians, the noisy builds are larger in 32 of 33 conditions (medians +0.410 against +0.005 nats, signed-rank $p = 5 \times 10^{-10}$). Raising the clean-policy threshold does not change it either; positives, noisy against shipped, at exact match ≥ 0.50: 52/66 against 11/47; ≥ 0.70: 51/62 against 10/42; ≥ 0.90: 34/44 against 7/31; ≥ 1.00: 25/26 against 5/15.

Does σ beat the perplexity gap? Over the 9 builds for which both are measured, σ_{cal} and the WikiText-2 perplexity gap of Table 5 order the quantizers identically (Spearman $+1.00$), and both get 4-bit group-128 wrong in the same way. As a *ranking* the screen is therefore redundant with a number practitioners already have. What σ_{cal} is for is that it is an across-position surrogate for the quantity Prop. 2 names, and the same kind of quantity the injected-noise probe of App. G.2 perturbs, on a different scale; the perplexity gap is a correlate of it, not a substitute in the mechanism.

Does the band's gate manufacture the effect? Δ_{int} carries $-\mathcal{L}_{01}$ and the band conditions on the arm that realises $\mathcal{L}_{01}$, so admission could select towards a positive interaction. `code/band_split.py` splits each cell's items by a fixed hash, applies the gate on one half and estimates Δ_{int} on the other, so selection and estimate share no item. Over the 150 cells the standard band admits (the 143 union cells plus the seven RTN W4-g128 SnapKV cells of the Llama cross-policy grid, which the union leaves to its own run, Table 26), the standard estimate has median +0.169 nats and the two split-sample estimates +0.142 and +0.178. Within a cell, the half the gate used exceeds the half it did not by 0.0000 nats in median over 276 admissions, each direction of the split counted (signed rank $p = 0.999$). The bias we were worried about is not measurable at this n; the band's real cost is the one App. G.3 names, that it is an exact-match rule and does not bound eviction's own loss.

Protecting the scoring projections, in full. Section 3.2 predicted, before the run, that keeping W_K at 16 bit would remove the direct path of the score noise; we test it against a placebo spending the same bits on W_V, which enters no score, and a uniform control. Protection does remove some (at 3 bits, where the unprotected oracle term is +1.39 [+0.64, +2.17], the paired contrast over the 64 shared items is +0.70 [+0.09, +1.35]), but the placebo is indistinguishable from it (-0.15 [-0.74, +0.43]), as is spending the bits uniformly, and all three arms are better models than the baseline (Table 3), which is what the placebo controls for. At 4 bits the prediction fails outright rather than tying: the unprotected oracle term is +0.54 [-0.05, +1.22], protecting W_K leaves it at +0.90 (paired contrast, unprotected minus protected, -0.37 [-0.78, +0.01] over 64 items), while the placebo lowers it to -0.01 (+0.55 [+0.08, +1.03]), so the protected arm's oracle term exceeds the placebo's by +0.92 [+0.43, +1.48]. The main text reports the 3-bit block because only there is the unprotected term itself significant, but the 4-bit block is the stronger evidence against the direct path: spending bits on the projection that enters no score removes more of the oracle term than spending them on the one that does. The bits buy the reduction, not the score path: what dominates is the *indirect* route, the already-perturbed hidden states entering every layer's attention, which no local protection of W_Q, W_K removes (Table 17).

Where in the context the damage lands, in full. C2 says structurally kept tokens are immune, so an early needle survives only if the vote finds it. At ten fixed depths (Fig. 5, middle) the clean policy already tilts early (+0.82 against +0.23 nats over the first and last three bins, $p = 0.003$) and

quantization amplifies it as score noise predicts (RTN W3-g128 +3.99 against +0.86, $p < 0.0001$); the clean selection transplanted removes over half of it (+3.99→+1.67) but much less of RTN W4 per-channel's, so the tilt is the corrupted oracle for the noisiest quantizer only (App. C).

Are the percentile intervals calibrated? The per-item interaction contrast is skewed, and a percentile interval is not second-order accurate on a skewed statistic. We therefore recomputed all 143 cells of the union with a bias-corrected and accelerated (BCa) interval on the same resamples, each cell on the grid its record was built from. The counts move *outwards* in both directions: percentile gives 71 significantly positive and 3 negative, BCa gives 79 and 6 (on the 101 cells of the four main grids alone, 49/1 against 54/3; on the 42 long-context cells, 22/2 against 25/3). So the percentile intervals used throughout are the conservative choice on the positive side, and the paper's headline count is not an artifact of the interval method; what BCa adds is negative cells (the abstract's "3 sub-additive" would read 6 under BCa), which is the same message as the ladder grid's excess of them. We report percentile intervals because they were fixed in advance and because they are the more conservative of the two for the claim we make.

The cross-policy grid is a separate run. The grid behind §5.3 was run at the same conditions as the main one but separately, so its SnapKV counts (Table 26) are not those of Table 25: on the 30 base-build cells the two share, the significant counts are 10/1 (main grid) against 9/0, and the sign of Δ_{int} agrees in 29 of 30. We report each within its own run rather than pooling them, by the precedence rule stated under *Reproducibility of the quantizer builds* above.

The free-alone cells, item by item. The mean is exposed to one heavy item, so we also read the item contrasts directly: a per-cell signed-rank test is significant in all 6 of the 6 free-alone cells whose mean is already significant, 4 at the Bonferroni threshold over all 143. Two more per-item statistics: across the 6 significant free-alone cells the largest single item carries a median 0.41 of the summed interaction and up to 0.96, while the median item is +0.009 nats. The effect is therefore a tail phenomenon in these cells, where most items are unaffected, but the tail is one-sided: a median 0.26 of items exceed additivity by more than 0.05 nats and a median 0.00 (at most 0.06) fall below -0.05.

The denominator. 18 of the 143 union cells are derivatives of two base RTN configurations at one setting (the W_K/W_V-protection arms of Table 17 and their same-cost group-size controls) and contribute 6 of the 71 positives. They are legitimate cells but they are not 18 independent quantizers. Without them the count is 65 of 125 positive and 3 negative, median +0.191 nats. The union's cells also share only 37 distinct clean-model arms, which is why §5.1 reads one test per unit, at a budget fixed in advance, alongside the pooled count.

LongBench item identity. The synthetic items are generated from a hash of (seed, index, length, task), so index i names the same prompt in every grid. LongBench items are not: they are positions in a round-robin over the requested subsets, so a grid that loaded one subset and a grid that loaded three disagree about what item i is. No LongBench cell mixes grids that loaded different subsets (each comes from one grid over the three subsets), and the only pooling is Table 21's, which pools two runs over the same three subsets, where the round-robin is nested so item i names the same document at 120 and at 288; the loader raises an error rather than skipping a subset it cannot find, and the prompt-hash check above is applied per source file for this task.

Which quantizers carry the count, and the ladder grid. Of the 71 significantly positive cells, 51 come from RTN W3-g128 and RTN W4-per-channel, 10 of 42 from GPTQ and AWQ (4 of 30 from the two most-shipped 4-bit recipes, GPTQ and AWQ W4-g128), and the remaining 10 from the other RTN rungs and the protection arms (6 of the 18 derivative cells). This is what C1 predicts (the noisiest quantizers open the channel widest), and it is also the limit on coverage. The ladder's two design variables are collinear at -0.91, and the rank partial correlations lean towards σ (+0.22 given bits, against -0.07 the other way), which is why §5.5 leans on the iso-bit rungs instead. The same point is visible in the ladder grid of §5.5, which is not pooled into the 143 (App. G.11: precedence keeps one canonical grid per setting) and is not the same conditions either—it is 24 quantizer configurations against the main grid's six, on the single needle only, at two budgets and 128 items. Over its 48 in-band cells Δ_{int} is significantly positive in 8 and negative in 7, median -0.022. That balance is a property of the quantizer set, not a contradiction: 18 of its rungs are quieter builds the main grid does not contain, where a null is the right answer. Restricted to the 12 cells it shares with the independent 256-item power run of App. G.3 (the six main-grid quantizers at the two shared budgets), the two agree in sign in 11 of 12, and in 11 of 12 against that run's strictly disjoint held-out items, with 4

Table 26: **The two cross-policy grids of §5.3**, one row per (policy, task, budget) block: cells run, cells inside the band of §3.3, the clean policy's mean exact match over those cells, the number significantly positive and negative, and the median Δ_{int} and Δ_{oracle} over the in-band cells (Qwen2.5-7B at 4k: six quantizers, 64 items; Llama-3.1-8B at 4k: three quantizers, 48 items). Blocks with no in-band cell are the budgets at which the clean policy fails on that task.

policy	task	B	cells	in band	clean EM	sig. > 0	sig. < 0	med. Δ_{int}	med. Δ_{oracle}
Qwen2.5-7B, 4k									
SnapKV	single	64	6	0	–	–	–	–	–
SnapKV	single	96	6	6	0.89	3	0	+0.202	+0.217
SnapKV	single	160	6	6	0.95	1	0	+0.020	+0.030
SnapKV	single	256	6	6	0.97	2	0	+0.115	+0.085
SnapKV	distr.	64	6	0	–	–	–	–	–
SnapKV	distr.	96	6	0	–	–	–	–	–
SnapKV	distr.	160	6	6	0.64	0	0	+0.003	-0.687
SnapKV	distr.	256	6	6	0.98	3	0	+0.440	+0.147
TOVA	single	64	6	6	0.61	2	0	+0.511	+0.658
TOVA	single	96	6	6	0.83	3	0	+0.415	+0.401
TOVA	single	160	6	6	0.95	2	0	+0.099	+0.023
TOVA	single	256	6	6	0.95	0	0	+0.033	+0.075
TOVA	distr.	64	6	0	–	–	–	–	–
TOVA	distr.	96	6	0	–	–	–	–	–
TOVA	distr.	160	6	6	0.66	3	0	+0.763	+0.308
TOVA	distr.	256	6	6	0.86	5	0	+0.721	+0.414
PyramidKV	single	64	6	0	–	–	–	–	–
PyramidKV	single	96	6	0	–	–	–	–	–
PyramidKV	single	160	6	6	0.58	2	1	-0.548	-0.401
PyramidKV	single	256	6	6	0.97	2	0	+0.060	-0.004
PyramidKV	distr.	64	6	0	–	–	–	–	–
PyramidKV	distr.	96	6	0	–	–	–	–	–
PyramidKV	distr.	160	6	0	–	–	–	–	–
PyramidKV	distr.	256	6	0	–	–	–	–	–
Llama-3.1-8B, 4k									
SnapKV	single	64	3	3	0.79	2	0	+1.800	+1.539
SnapKV	single	96	3	3	1.00	2	0	+0.176	+0.088
SnapKV	single	160	3	3	1.00	2	0	+0.087	+0.054
SnapKV	single	256	3	3	1.00	1	0	+0.037	+0.036
SnapKV	distr.	64	3	0	–	–	–	–	–
SnapKV	distr.	96	3	3	0.81	2	0	+1.944	+1.378
SnapKV	distr.	160	3	3	0.98	2	0	+0.254	+0.112
SnapKV	distr.	256	3	3	0.98	2	0	+0.157	-0.001
TOVA	single	64	3	3	0.62	2	0	+1.023	+0.325
TOVA	single	96	3	3	0.83	2	1	+0.745	+0.368
TOVA	single	160	3	3	0.92	2	1	+0.334	+0.008
TOVA	single	256	3	3	0.94	2	1	+0.229	+0.015
TOVA	distr.	64	3	3	0.90	2	0	+0.947	-0.117
TOVA	distr.	96	3	3	1.00	3	0	+1.021	+0.102
TOVA	distr.	160	3	3	1.00	2	0	+0.859	+0.005
TOVA	distr.	256	3	3	1.00	2	0	+0.650	+0.028
PyramidKV	single	64	3	3	0.71	2	0	+3.432	+2.369
PyramidKV	single	96	3	3	0.98	3	0	+0.480	+0.334
PyramidKV	single	160	3	3	1.00	3	0	+0.066	+0.080
PyramidKV	single	256	3	3	1.00	2	0	+0.058	+0.046
PyramidKV	distr.	64	3	0	–	–	–	–	–
PyramidKV	distr.	96	3	3	0.85	2	0	+3.799	+2.493
PyramidKV	distr.	160	3	3	1.00	2	0	+0.686	+0.358
PyramidKV	distr.	256	3	3	1.00	2	0	+0.202	+0.109

positive and 2 negative in the ladder itself. Its positives are the noisy rungs (RTN W3 at three group sizes, RTN and AWQ W4 per-channel) plus GPTQ W4-g128, the quietest build in the ladder, which is the inversion the next sentence describes; its negatives, in full and ordered by magnitude, are AWQ

W3-g64 at $B = 96$ -0.343, RTN W4-g128 at $B = 96$ -0.297, AWQ W4-g128 at $B = 96$ -0.270, GPTQ W4-g64 at $B = 160$ -0.149, AWQ W3-g64 at $B = 160$ -0.125, RTN W6-g128 at $B = 160$ -0.086, RTN W5-g32 at $B = 160$ -0.074 nats, 3 of them at the operating budget $B = 96$, and two of those three (RTN and AWQ W4-g128) are 4-bit group-128 recipes. The sign structure follows σ in the aggregate but not within a bit budget: at $B = 96$ the three W4-g128 builds give +0.383 [+0.050, +0.723] under GPTQ, -0.270 [-0.537, -0.040] under AWQ and -0.297 [-0.575, -0.048] under RTN, two of the three significantly *sub*-additive at identical cost. And 7 significant negatives against the ≈ 1.2 a nominal 95% interval would give over 48 cells is more than chance. That is not an interval artefact: the BCa recount (*Are the percentile intervals calibrated?*, above) returns *more* negatives, not fewer, so we read the excess as real sub-additivity at those rungs. Pooling the ladder with the 143 would give 79 of 191 positive and 10 negative. We report the unpooled count because the precedence rule was fixed in advance, and this one because a reader should not have to reconstruct it.

Does the LongBench conclusion depend on the band? The band there is a loss ratio fixed in advance at $1.5\times$ the full-cache loss, which is generous: an in-band cell can cost the clean model half its loss again. Tightening and loosening it does not change the reading. At $1.3\times$, 16 cells remain in band and Δ_{int} is significantly positive in 12 with 0 negative—TOVA is 8 of 8. At $1.2\times$ only 3 cells survive at all, too few to read. Loosening to $2.0\times$ admits 52 cells, 24 positive and 0 negative, and it is the first band at which the accumulated-attention family enters: 8 H_2O-norm cells, of which 0 are significantly positive. A policy family that does not rank by the query has no oracle term to open, and at the only band where it is measurable at all, it shows none. That is what the identifiability argument predicts.

LongBench, per subset. The pooled counts of §5.4 are heterogeneous across the three QA subsets. Under TOVA, Δ_{int} is significantly positive in 12 of 12 multi-field QA cells, 6 of 12 on 2WikiMQA and 3 of 12 on HotpotQA; under SnapKV, 5, 2 and 3 of 12 each. That is also the order of how close the clean policy is to failing on each: the best bf16 SnapKV loss ratio is 1.15 on multi-field QA, 1.08 on 2WikiMQA and 1.01 on HotpotQA, against full-cache losses of 13.8, 6.9 and 8.3 nats. The effect appears in every subset, and under the policy with the most in-band cells it is ordered as the budget story predicts: largest where eviction alone already costs the clean model most. The SnapKV ordering is not monotone, which at four quantizers and three budgets per subset we do not read as more than noise. The 6 significant free-alone cells of §5.1 likewise span 3 of the 4 (model, context) settings in which the full-string gate can be evaluated at all, and none is on Qwen2.5-7B at 4k, where the item set is largest.

Bootstrap p-values use 4000 paired resamples, except for cells whose p reached that resolution's floor, which `code/hires_p.py` recomputes at 400,000 so the Bonferroni threshold $0.05/143$ is resolvable at all—with 4000 resamples the achievable p below 10^{-3} are only $\{2.5 \times 10^{-4}, 5 \times 10^{-4}\}$ and the threshold falls between them. The smallest p in the set is 2×10^{-6}. A p-value is a Monte-Carlo estimate, so re-running that script gives values within resampling error of the released ones; the counts it feeds are unchanged. A significant interaction is not the same thing as a material accuracy loss, and it is not usually harmless either: of the 71 significantly positive cells in the paper, 23 cost the deployed system at most 5 exact-match points against the clean policy at the same budget (median 1.6 over those), while the remaining 48 cost more, and the median over all 71 is 9.4 exact-match points. The 6 significant free-alone cells lose a median 8.9. Two properties of the free-alone regime of §5.1 also belong here. The band itself conditions on a component of the estimand: Δ_{int} carries $-\mathcal{L}_{01}$, and requiring the bf16 evict-only arm to work selects mildly towards a smaller realised $\mathcal{L}_{01}$ and so towards positive Δ_{int}. The floor below bounds that bias, and the effects it applies to are five to ten times larger, but the selection runs in the direction of the finding and we say so rather than arguing the band only in the other direction. Because that gate pins both single-axis costs to within 0.05 nats of the baseline, $\Delta_{\text{int}} = \mathcal{L}_{11} - \mathcal{L}_{10} - \mathcal{L}_{01} + \mathcal{L}_{00}$ is bounded below by -0.048 nats across those cells: a significantly negative cell was arithmetically almost unavailable, so the informative quantity there is how large Δ_{int} is, not that it is positive. And the decomposition is not uniform across the regime—summed over the 6 significant free cells the oracle term carries 22% of Δ_{int} against 41% over the 70 cells with $\Delta_{\text{int}} > 0.05$, and it is individually significant in 3 of them against 5 for the fragility term. Where both compressions are free the interaction is real but more often the deployed model's extra cache-sensitivity than a corrupted retain set; where the quantizer is noisy the oracle term takes over (Table 27). Of these, 83 have a measurable oracle-quality gap Δ_{oq} (the protection arms do not); it is significantly positive in 15 and negative in 9—null in aggregate, and we count it among the paper's negative results. Over those 83 the median is +0.007 nats and a signed-rank

test gives $p = 0.155$, yet across the same cells Δ_{oq} ranks with Δ_{oracle} at Spearman +0.70. It is not uniformly null. Over the wider union of 125 identifiable SnapKV cells that carry a Δ_{oq} (the 143 minus the derivative grids), it is positive in 20 and negative in 13, tracks Δ_{oracle} at Spearman +0.52, and is positive in 21 of the 30 cells with $\Delta_{\mathrm{oracle}} > 0.3$, median +0.34 nats. The reading is therefore not that the quantized retain set is merely mismatched: where the channel is open it is worse for the clean model as well, as Prop. 2 requires. The aggregate null comes from the many cells with a near-zero oracle term, where Δ_{oq} is symmetric noise around zero, and from the smaller item sets that Δ_{oq}'s two arms share; at 0.5B, where the effect sizes are larger, it is positive far more often.

Table 27: **Every identifiable SnapKV cell in the four main grids** (the two further long-context grids of §5.1, Llama-3.1-8B at 16k and Qwen2.5-7B at 32k, are separate files and add 42 more, 22 significantly positive and 2 negative; the abstract counts the union of all six). *: Δ_{int} interval excludes zero from above; †: from below. $\mathrm{EM}_{\mathrm{bf16,evict}}$ is the clean policy's exact match at this budget (the band requires ≥ 0.5); $\mathrm{EM}_{\mathrm{joint}}$ is the deployed system's.

Model	Ctx	Task	Quantizer	B	Δ_{int} [95% CI]	Δ_{oracle}	Δ_{frag}	Δ_{oq}	$\mathrm{EM}_{\mathrm{bf16,evict}}$	$\mathrm{EM}_{\mathrm{joint}}$
Qwen2.5-7B	16k	single	GPTQ W4-g128	96	+1.11 [-0.05, +2.37]	+0.17	+0.94	-0.26	0.72	0.62
Qwen2.5-7B	16k	single	GPTQ W3-g128	96	+1.67 [+0.01, +3.55]*	+0.80	+0.86	+0.12	0.72	0.53
Qwen2.5-7B	16k	single	RTN W4-per-ch.	96	+2.29 [+0.86, +3.86]*	+1.11	+1.18	+1.37	0.72	0.50
Qwen2.5-7B	16k	single	RTN W3-g128	96	+4.30 [+2.58, +6.14]*	+1.28	+3.02	+1.53	0.72	0.34
Qwen2.5-7B	16k	single	GPTQ W4-g128	160	+0.06 [-0.01, +0.15]	-0.11	+0.17	+0.03	1.00	0.97
Qwen2.5-7B	16k	single	GPTQ W3-g128	160	+0.56 [+0.16, +1.11]*	-0.09	+0.66	+0.09	1.00	0.81
Qwen2.5-7B	16k	single	RTN W4-per-ch.	160	+0.36 [+0.10, +0.73]*	+0.02	+0.34	-0.01	1.00	0.91
Qwen2.5-7B	16k	single	RTN W3-g128	160	+1.24 [+0.51, +2.11]*	+0.85	+0.38	+0.27	1.00	0.81
Qwen2.5-7B	16k	single	GPTQ W4-g128	256	+0.01 [-0.22, +0.23]	+0.03	-0.02	-0.10	0.97	0.94
Qwen2.5-7B	16k	single	GPTQ W3-g128	256	-0.03 [-0.25, +0.12]	-0.19	+0.16	-0.11	0.97	0.97
Qwen2.5-7B	16k	single	RTN W4-per-ch.	256	+0.19 [-0.15, +0.57]	-0.21	+0.40	-0.11	0.97	0.94
Qwen2.5-7B	16k	single	RTN W3-g128	256	+0.22 [-0.12, +0.59]	+0.13	+0.08	-0.11	0.97	0.94
Qwen2.5-7B	16k	single	GPTQ W4-g128	512	-0.01 [-0.16, +0.12]	+0.04	-0.05	-0.05	0.97	0.97
Qwen2.5-7B	16k	single	GPTQ W3-g128	512	0.00 [-0.14, +0.12]	-0.01	+0.01	-0.05	0.97	0.97
Qwen2.5-7B	16k	single	RTN W4-per-ch.	512	-0.04 [-0.15, +0.02]	-0.29	+0.25	-0.05	0.97	1.00
Qwen2.5-7B	16k	single	RTN W3-g128	512	0.00 [-0.13, +0.09]	+0.01	-0.01	0.00	0.97	1.00
Qwen2.5-7B	16k	distractor	GPTQ W4-g128	160	+0.36 [-0.74, +1.56]	+0.28	+0.08	+0.19	0.69	0.53
Qwen2.5-7B	16k	distractor	GPTQ W3-g128	160	+1.92 [+0.28, +3.50]*	+1.06	+0.86	+0.37	0.69	0.28
Qwen2.5-7B	16k	distractor	RTN W4-per-ch.	160	+1.37 [-0.07, +2.87]	+0.96	+0.40	-0.26	0.69	0.38
Qwen2.5-7B	16k	distractor	RTN W3-g128	160	+3.44 [+1.00, +5.79]*	+0.67	+2.78	-1.38	0.69	0.28
Qwen2.5-7B	16k	distractor	GPTQ W4-g128	256	+1.04 [+0.20, +2.07]*	+0.43	+0.61	+0.49	0.88	0.84
Qwen2.5-7B	16k	distractor	GPTQ W3-g128	256	+1.34 [+0.36, +2.47]*	+0.67	+0.66	+0.56	0.88	0.72
Qwen2.5-7B	16k	distractor	RTN W4-per-ch.	256	+0.39 [+0.01, +0.78]*	-0.41	+0.80	-0.33	0.88	0.78
Qwen2.5-7B	16k	distractor	RTN W3-g128	256	+2.43 [+1.18, +3.84]*	-0.78	+3.21	-0.40	0.88	0.53
Qwen2.5-7B	16k	distractor	GPTQ W4-g128	512	+0.44 [+0.06, +0.94]*	+0.38	+0.06	+0.23	1.00	0.88
Qwen2.5-7B	16k	distractor	GPTQ W3-g128	512	+0.51 [0.00, +1.32]*	+0.05	+0.46	+0.01	1.00	0.91
Qwen2.5-7B	16k	distractor	RTN W4-per-ch.	512	+0.26 [+0.05, +0.58]*	+0.05	+0.21	-0.01	1.00	0.91
Qwen2.5-7B	16k	distractor	RTN W3-g128	512	+0.70 [+0.04, +1.44]*	-1.16	+1.85	+0.49	1.00	0.84
Qwen2.5-7B	4k	single	GPTQ W4-g128	96	+0.57 [+0.03, +1.14]*	+0.43	+0.14	+0.31	0.89	0.81
Qwen2.5-7B	4k	single	AWQ W4-g128	96	-0.14 [-0.58, +0.20]	-0.24	+0.10	-0.09	0.89	0.86
Qwen2.5-7B	4k	single	RTN W4-g128	96	-0.16 [-0.61, +0.21]	0.00	-0.17	-0.01	0.89	0.91
Qwen2.5-7B	4k	single	GPTQ W3-g128	96	-0.19 [-0.59, +0.14]	-0.16	-0.03	-0.22	0.89	0.91
Qwen2.5-7B	4k	single	RTN W4-per-ch.	96	+0.93 [+0.27, +1.66]*	+0.54	+0.39	+0.51	0.89	0.72
Qwen2.5-7B	4k	single	RTN W3-g128	96	+2.00 [+1.25, +2.76]*	+1.39	+0.61	+1.15	0.89	0.55
Qwen2.5-7B	4k	single	RTN W4-per-ch. + W_K@16	96	+1.23 [+0.52, +2.02]*	+0.90	+0.32	–	0.89	0.64
Qwen2.5-7B	4k	single	RTN W4-per-ch. + W_V@16	96	+0.38 [-0.18, +0.96]	-0.01	+0.40	–	0.89	0.81
Qwen2.5-7B	4k	single	RTN W4-g64	96	-0.01 [-0.50, +0.47]	+0.01	-0.03	–	0.89	0.92
Qwen2.5-7B	4k	single	RTN W3-g128 + W_K@16	96	+0.96 [+0.35, +1.60]*	+0.69	+0.27	–	0.89	0.78
Qwen2.5-7B	4k	single	RTN W3-g128 + W_V@16	96	+0.90 [+0.39, +1.38]*	+0.84	+0.06	–	0.89	0.67
Qwen2.5-7B	4k	single	RTN W3-g64	96	+0.66 [+0.19, +1.17]*	+0.59	+0.06	–	0.89	0.72
Qwen2.5-7B	4k	single	GPTQ W4-g128	160	+0.03 [-0.18, +0.22]	+0.03	0.00	+0.05	0.95	0.97
Qwen2.5-7B	4k	single	AWQ W4-g128	160	+0.01 [-0.23, +0.25]	-0.07	+0.09	-0.13	0.95	0.94
Qwen2.5-7B	4k	single	RTN W4-g128	160	-0.03 [-0.15, +0.08]	+0.03	-0.06	+0.05	0.95	0.97
Qwen2.5-7B	4k	single	GPTQ W3-g128	160	-0.09 [-0.32, +0.10]	-0.01	-0.08	-0.10	0.95	0.97
Qwen2.5-7B	4k	single	RTN W4-per-ch.	160	+0.33 [+0.03, +0.69]*	+0.12	+0.21	-0.08	0.95	0.84
Qwen2.5-7B	4k	single	RTN W3-g128	160	+0.21 [-0.05, +0.48]	+0.13	+0.08	+0.07	0.95	0.89
Qwen2.5-7B	4k	single	RTN W4-per-ch. + W_K@16	160	+0.09 [-0.08, +0.27]	-0.08	+0.17	–	0.95	0.92
Qwen2.5-7B	4k	single	RTN W4-per-ch. + W_V@16	160	+0.17 [-0.05, +0.40]	+0.04	+0.13	–	0.95	0.91
Qwen2.5-7B	4k	single	RTN W4-g64	160	-0.12 [-0.31, +0.01]	-0.11	-0.01	–	0.95	0.98
Qwen2.5-7B	4k	single	RTN W3-g128 + W_K@16	160	+0.09 [-0.06, +0.25]	+0.06	+0.03	–	0.95	0.91
Qwen2.5-7B	4k	single	RTN W3-g128 + W_V@16	160	+0.07 [-0.12, +0.28]	+0.06	+0.01	–	0.95	0.92
Qwen2.5-7B	4k	single	RTN W3-g64	160	+0.01 [-0.14, +0.17]	+0.02	0.00	–	0.95	0.95
Qwen2.5-7B	4k	single	GPTQ W4-g128	256	+0.11 [-0.03, +0.33]	+0.14	-0.02	+0.09	0.97	0.97
Qwen2.5-7B	4k	single	AWQ W4-g128	256	+0.26 [+0.01, +0.66]*	+0.16	+0.11	+0.05	0.97	0.94
Qwen2.5-7B	4k	single	RTN W4-g128	256	+0.12 [-0.04, +0.39]	+0.14	-0.03	+0.14	0.97	0.97
Qwen2.5-7B	4k	single	GPTQ W3-g128	256	-0.02 [-0.06, +0.02]	-0.02	0.00	-0.02	0.97	1.00
Qwen2.5-7B	4k	single	RTN W4-per-ch.	256	+0.17 [+0.04, +0.32]*	+0.07	+0.10	+0.01	0.97	0.89
Qwen2.5-7B	4k	single	RTN W3-g128	256	+0.15 [+0.02, +0.32]*	+0.10	+0.05	+0.08	0.97	0.95

Model	Ctx	Task	Quantizer	B	Δ_{int} [95% CI]	Δ_{oracle}	Δ_{frag}	Δ_{oq}	$\text{EM}_{\text{bf16,evict}}$	EM_{joint}
Qwen2.5-7B	4k	single	RTN W4-per-ch. + W_K@16	256	+0.16 [+0.04, +0.33]*	+0.06	+0.11	–	0.97	0.94
Qwen2.5-7B	4k	single	RTN W4-per-ch. + W_V@16	256	+0.11 [+0.01, +0.23]*	0.00	+0.11	–	0.97	0.92
Qwen2.5-7B	4k	single	RTN W4-g64	256	-0.03 [-0.07, 0.00]	-0.01	-0.02	–	0.97	1.00
Qwen2.5-7B	4k	single	RTN W3-g128 + W_K@16	256	+0.11 [-0.01, +0.29]	+0.08	+0.04	–	0.97	0.97
Qwen2.5-7B	4k	single	RTN W3-g128 + W_V@16	256	+0.08 [-0.01, +0.22]	+0.08	0.00	–	0.97	0.94
Qwen2.5-7B	4k	single	RTN W3-g64	256	+0.05 [-0.04, +0.17]	+0.01	+0.03	–	0.97	0.97
Qwen2.5-7B	4k	distractor	GPTQ W4-g128	160	-0.02 [-0.90, +0.87]	-0.37	+0.35	-0.15	0.64	0.61
Qwen2.5-7B	4k	distractor	AWQ W4-g128	160	-0.98 [-1.85, -0.18]†	-1.15	+0.17	-1.32	0.64	0.69
Qwen2.5-7B	4k	distractor	RTN W4-g128	160	+0.05 [-0.96, +1.06]	-0.04	+0.09	-0.39	0.64	0.62
Qwen2.5-7B	4k	distractor	GPTQ W3-g128	160	-1.04 [-2.17, +0.05]	-1.53	+0.50	-2.01	0.64	0.62
Qwen2.5-7B	4k	distractor	RTN W4-per-ch.	160	+0.03 [-1.09, +1.12]	-0.63	+0.66	-1.21	0.64	0.47
Qwen2.5-7B	4k	distractor	RTN W3-g128	160	+1.12 [-0.16, +2.35]	-0.99	+2.11	-1.38	0.64	0.31
Qwen2.5-7B	4k	distractor	GPTQ W4-g128	256	+0.08 [-0.02, +0.21]	+0.09	-0.01	+0.07	0.98	0.97
Qwen2.5-7B	4k	distractor	AWQ W4-g128	256	+0.02 [-0.03, +0.09]	-0.03	+0.06	-0.01	0.98	0.97
Qwen2.5-7B	4k	distractor	RTN W4-g128	256	+0.22 [-0.03, +0.70]	+0.24	-0.02	+0.19	0.98	0.98
Qwen2.5-7B	4k	distractor	GPTQ W3-g128	256	+0.72 [+0.31, +1.22]*	+0.25	+0.47	+0.02	0.98	0.86
Qwen2.5-7B	4k	distractor	RTN W4-per-ch.	256	+0.66 [+0.35, +1.03]*	+0.17	+0.49	+0.02	0.98	0.80
Qwen2.5-7B	4k	distractor	RTN W3-g128	256	+0.85 [+0.45, +1.34]*	+0.12	+0.73	+0.09	0.98	0.81
Llama-3.1-8B	4k	single	GPTQ W4-g128	64	+0.09 [-0.22, +0.44]	+0.27	-0.18	+0.23	0.79	0.83
Llama-3.1-8B	4k	single	RTN W4-per-ch.	64	+2.97 [+2.28, +3.72]*	+2.27	+0.70	+2.21	0.79	0.44
Llama-3.1-8B	4k	single	RTN W3-g128	64	+1.80 [+1.28, +2.36]*	+1.54	+0.26	+2.30	0.79	0.65
Llama-3.1-8B	4k	single	GPTQ W4-g128	96	0.00 [-0.03, +0.03]	+0.01	-0.01	+0.01	1.00	1.00
Llama-3.1-8B	4k	single	RTN W4-per-ch.	96	+0.18 [+0.10, +0.28]*	+0.09	+0.09	+0.03	1.00	0.98
Llama-3.1-8B	4k	single	RTN W3-g128	96	+0.22 [+0.15, +0.29]*	+0.11	+0.11	+0.03	1.00	1.00
Llama-3.1-8B	4k	single	GPTQ W4-g128	160	0.00 [-0.02, +0.02]	+0.01	-0.01	0.00	1.00	1.00
Llama-3.1-8B	4k	single	RTN W4-per-ch.	160	+0.09 [+0.04, +0.14]*	+0.08	+0.01	0.00	1.00	1.00
Llama-3.1-8B	4k	single	RTN W3-g128	160	+0.11 [+0.07, +0.15]*	+0.05	+0.05	0.00	1.00	1.00
Llama-3.1-8B	4k	single	GPTQ W4-g128	256	0.00 [-0.02, +0.02]	+0.01	-0.01	0.00	1.00	1.00
Llama-3.1-8B	4k	single	RTN W4-per-ch.	256	+0.04 [-0.01, +0.08]	+0.04	0.00	0.00	1.00	1.00
Llama-3.1-8B	4k	single	RTN W3-g128	256	+0.08 [+0.05, +0.11]*	+0.04	+0.04	-0.01	1.00	1.00
Llama-3.1-8B	4k	distractor	GPTQ W4-g128	96	+0.07 [-0.74, +0.77]	+0.19	-0.12	-0.14	0.81	0.83
Llama-3.1-8B	4k	distractor	RTN W4-per-ch.	96	+1.94 [+0.97, +2.88]*	+1.38	+0.57	+0.96	0.81	0.56
Llama-3.1-8B	4k	distractor	RTN W3-g128	96	+2.13 [+0.89, +3.40]*	+1.62	+0.51	+0.83	0.81	0.56
Llama-3.1-8B	4k	distractor	GPTQ W4-g128	160	-0.02 [-0.09, +0.02]	+0.01	-0.02	-0.02	0.98	1.00
Llama-3.1-8B	4k	distractor	RTN W4-per-ch.	160	+0.47 [+0.24, +0.76]*	+0.25	+0.23	0.00	0.98	0.94
Llama-3.1-8B	4k	distractor	RTN W3-g128	160	+0.25 [+0.12, +0.39]*	+0.11	+0.14	+0.01	0.98	0.92
Llama-3.1-8B	4k	distractor	GPTQ W4-g128	256	-0.01 [-0.05, +0.02]	+0.01	-0.02	+0.05	0.98	1.00
Llama-3.1-8B	4k	distractor	RTN W4-per-ch.	256	+0.16 [+0.07, +0.25]*	0.00	+0.16	0.00	0.98	0.98
Llama-3.1-8B	4k	distractor	RTN W3-g128	256	+0.17 [+0.10, +0.25]*	+0.05	+0.13	+0.08	0.98	0.98
Qwen2.5-3B	4k	single	RTN W4-g128	96	+2.11 [+0.90, +3.34]*	+0.70	+1.41	+0.82	0.58	0.27
Qwen2.5-3B	4k	single	RTN W4-g128	160	+0.26 [+0.10, +0.49]*	+0.04	+0.22	+0.02	1.00	0.96
Qwen2.5-3B	4k	single	RTN W4-g128	256	+0.23 [+0.09, +0.42]*	+0.11	+0.13	+0.01	1.00	0.92
Qwen2.5-3B	4k	distractor	RTN W4-g128	256	+1.47 [+0.88, +2.11]*	+0.98	+0.49	+0.74	0.98	0.62