Greedy Decoding Is Not Precision-Invariant: Cross-Precision Output Divergence in LLM Inference

Anonymous authors
Paper under double-blind review

Abstract

Greedy decoding from large language models is commonly treated as deterministic. We show it is not *precision-invariant*: the same model, prompt, and decoding algorithm produce different outputs in BF16 versus FP16 on identical hardware. This affects 59–82% of prompts across six models (1.1B–12B parameters, four families) and three benchmarks; a single token flip cascades into a fully divergent generation. We develop an error-propagation theory whose central prediction is counterintuitive: 22 layers of accumulated body error are *irrelevant* to whether a token flips. The outcome depends entirely on a single scalar, the top-two logit margin at the lm_head. The theory makes five falsifiable predictions, including that applying *more* FP32 compute (broader scope) makes agreement *worse*, and all five are confirmed. The one effective intervention, selective FP32 lm_head recomputation, triggered only when the margin falls below a threshold, delivers +22–36 pp exact agreement at $< 4\%$ latency overhead. We map the applicability boundary across six models and four batch sizes, identifying training-time precision stability as the determining factor.

1 Introduction

Precision switching in LLM serving is a routine operational reality, not a hypothetical concern. GPU instance pools mix hardware generations such as A100 (BF16-native) and V100 (FP16-native). Model hubs distribute a single checkpoint that serving frameworks silently cast to the hardware's preferred format. Autoscalers migrate inference across regions with heterogeneous GPU fleets. In all these cases, the same model, prompt, and greedy-decoding configuration can produce *different* outputs depending on which machine serves the request. No log, no alert, and no configuration difference explains the discrepancy. Existing reproducibility tooling (`torch.use_deterministic_algorithms`, fixed CUBLAS workspace) reports no issue because within any *single* precision everything is bit-reproducible; the divergence is invisible to every standard diagnostic. In principle, greedy decoding is deterministic: at each step t, the model selects $y_t = \arg\max_v z_t(v)$.

Running the same model with the same prompt under BF16 vs. FP16 produces different outputs on 59–70% of TinyLlama-1.1B prompts across public benchmarks (GSM8K, HumanEval, MBPP). The rate rises to 82% on Qwen2.5-3B-Instruct and reaches 100% on BF16-saturated Qwen variants. The phenomenon spans six models from 1.1B to 12B parameters across four families (Llama, Qwen, Mistral, and OLMoE; Table 7).

The root cause has a sharper form than one might expect: hidden-state error accumulates *uniformly* through the transformer body (present in every prompt regardless of outcome), and a token flips if and only if the current top-two logit margin at the lm_head falls below the per-logit perturbation magnitude (Section 7.5). This immediately implies the remedy: selectively recompute the lm_head in FP32 at steps where the margin falls below a gating threshold τ.

Specifically, this paper makes three contributions:

1. **A causal mechanism for cross-precision greedy divergence.** A token flips between BF16 and FP16 if and only if the lm_head top-two logit margin falls within the per-logit perturbation magnitude (formalized in Proposition 1). Four independent experiments, divergence-conditioned

tracing, layer grafting, KV-cache grafting, and logit-lens traceback, converge on the lm_head as the dominant error source. The mechanism replicates across three tasks and two model families.

2. **Five controlled interventions test the mechanism's predictions.** The mechanism predicts each intervention's outcome *before* the experiment is run: integer quantization and temperature sharpening have zero effect; top-K FP32 recomputation works for all $K \geq 2$; full FP32 lm_head recomputation matches top-K; and extending FP32 scope beyond the lm_head degrades agreement (63%→44%). All five predictions are confirmed.

3. **An applicability map across models, kernels, and batch sizes.** We characterise the phenomenon on six models (1.1–12B, four families), two attention kernels, and batch sizes $\{1, 2, 4, 8\}$ (with composition experiments extending to bs= 16). The lift ranges from +36pp (Llama-3.2-3B) to 0pp (Qwen BF16-saturated); effectiveness tracks training-time precision stability. Two boundaries emerge: the remedy survives FlashAttention-2 with identical lifts, but the lm_head-only scope vanishes at bs$\geq$8.[1]

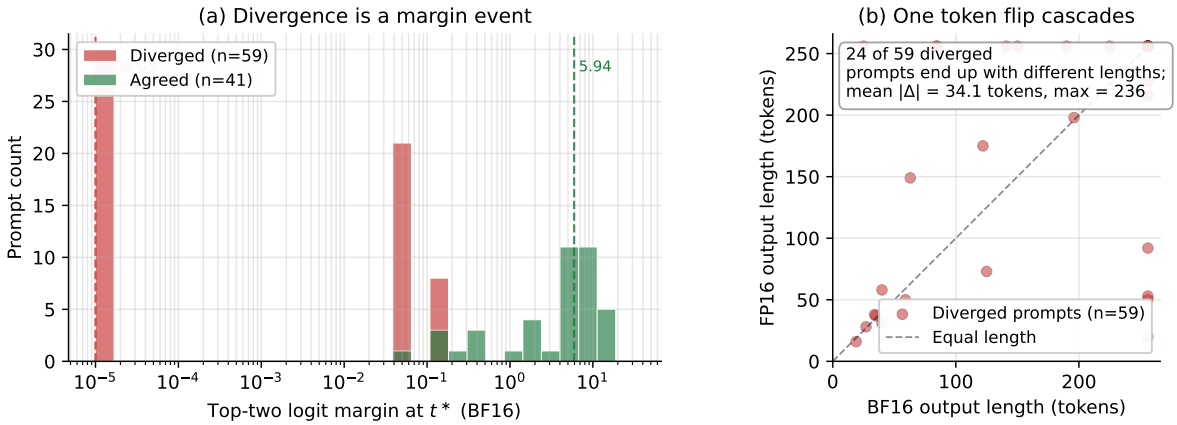

Figure 1: Two empirical facts that shape this paper (TinyLlama-1.1B, GSM8K, BF16 vs. FP16 greedy decoding, $n = 100$). **(a)** Distribution of the top-two logit margin at the measurement step t^*, for prompts whose outputs diverge (red) and for matched agreed controls (green). The two groups are bimodal across more than five orders of magnitude. Diverged median $\approx 10^{-5}$, with 30/59 exactly at 0 (perfect BF16 ties); Agreed median ≈ 5.9. Divergence is a *margin event* at the lm_head, not an error-magnitude event in the transformer body. **(b)** Sequence-length scatter of BF16 vs. FP16 outputs on the 59 Diverged prompts. A single token flip typically cascades into trajectory-level divergence: 24/59 prompts end up with different lengths, with a mean length difference of 34 tokens and maximum difference of 236 tokens.

2 Related work

As machine learning moves into regulated and audited deployments, reproducibility is increasingly treated as a deployment requirement in its own right, not merely a byproduct of accuracy. Floating-point nondeterminism in training is classical (Goldberg, 1991; Higham, 2002; Zhuang et al., 2022). Framework-level determinism tools (PyTorch Team, 2020) are effective within a fixed format but have no effect on cross-precision divergence (Section 6.1). BF16/FP16 serving is standard (Dettmers et al., 2022; Micikevicius et al., 2018; Kalamkar et al., 2019) but its impact on inference determinism has not been studied.

[1]Throughout this paper, "Pass@1" denotes greedy-decoding final-answer correctness on a benchmark—e.g. on GSM8K, the extracted final number—and is computed under our bare-prompt format with no system prompt or chat template, then McNemar-tested in matched-pair form between Baseline and Intervention C. We do not run the OpenAI HumanEval execution sandbox; HumanEval is included to characterise EAR/SAR rather than to compute the standard execution-based pass@1. Pass@1 numbers therefore lower-bound what each model can achieve under its native chat template (Appendix VI, the chat-template results).

The closest prior work is LayerCast (Ding et al., 2025), which applies FP32 globally to address system-level variations. We target the numerical-format layer and identify the lm_head as the dominant stage, enabling a 27× cheaper fix (< 4% vs. +38% latency). Concurrent work (He et al., 2025; Atil et al., 2024; Qi et al., 2025; Gond et al., 2025; Fu et al., 2025) targets kernel- or batch-level sources orthogonal to format-level divergence, or characterises nondeterminism without localising its mechanism. Messina & Scotta (2026) formalise "background temperature" to characterise the effective stochasticity of floating-point perturbations at T=0; our work complements that framework with a mechanistic diagnosis and a targeted fix. None of these works identifies the lm_head margin as the decisive quantity or proposes selective gated recomputation.

3 Problem formulation

Consider an autoregressive LM with vocabulary V. At step t, the model computes logits $z_t = f_\theta(x, y_{<t})$ and selects $y_t = \arg\max_v z_t(v)$ (Vaswani et al., 2017). In floating-point arithmetic, z_t depends on the numerical precision (BF16, FP16, or FP32), tensor-parallelism degree, and GPU architecture.

We define the *top-two logit gap* $\Delta_t^{(c)} = z_t^{(c)}(v^{(1)}) - z_t^{(c)}(v^{(2)})$ under configuration c. When this gap is small enough that floating-point rounding across configurations can flip the argmax, different configurations select different tokens. The autoregressive conditioning then diverges, producing a cascade of length $L_{\text{cascade}} = T - t_{\text{div}}$.

We measure four quantities. The **Exact Agreement Rate** (EAR) is the fraction of prompts producing identical token sequences across configurations. The **First Divergence Step** (t^*) is the position of the first token mismatch. The **Trigger Rate** is the fraction of steps where $\Delta_t < \tau$ (used only when a selective intervention applies). The **Mean Gap** ($\bar{\Delta}$) averages Δ_t over all decoding steps. For tasks with well-defined final answers, we additionally report **Semantic Agreement Rate** (SAR)—whether the extracted conclusion matches across configurations even if the full reasoning differs.

4 Inference-time interventions for cross-precision agreement

The contribution of this paper is a mechanistic diagnosis of cross-precision divergence—it is a top-two margin event localised at the lm_head—together with a controlled ablation that rules out alternative explanations. We study three inference-time interventions that, taken together, constitute that ablation: each one modifies only the final decoding step (without retraining, modifying the transformer body, or adding persistent GPU memory) but tests a different hypothesis about the mechanism. Intervention A asks whether the issue is comparison-level (a tie-breaking artefact between BF16 and FP16 logit values that happen to be representable differently). Intervention B asks whether a small-K FP32 correction is sufficient, and is a direct empirical probe of the rank-1-vs-rank-2 prediction. Intervention C asks whether full FP32 lm_head recomputation eliminates divergence, and is also the most natural deployable fix. The three interventions form a falsification ladder: if A succeeds, divergence is comparison-level; if A fails but B with $K = 2$ succeeds, the mechanism is strictly a rank-1-vs-rank-2 event; if B at $K = 2$ matches C at $K = |V|$, the mechanism is lm_head-localised and a 4000× smaller intervention suffices. Figure 2 illustrates the shared selective-gating envelope; the three interventions below differ only in what they manipulate inside the gated step.

Theoretical predictions. An error-propagation model (Section 7, $\sigma_z \approx 0.026$) predicts: A has zero effect, B and C produce comparable lifts (rank-1-vs-rank-2 only), and extending scope beyond lm_head hurts. All predictions confirmed.

Selective triggering. All interventions fire only when $\Delta_t < \tau = 10^{-3}$ (0.7–1.4% of steps), keeping overhead <4%.

4.1 Intervention A: Integer logit quantization

The argmax over floating-point logits can differ across precisions because BF16 and FP16 represent the same real number with different roundings; integer comparison after quantization is bit-exact. Intervention A

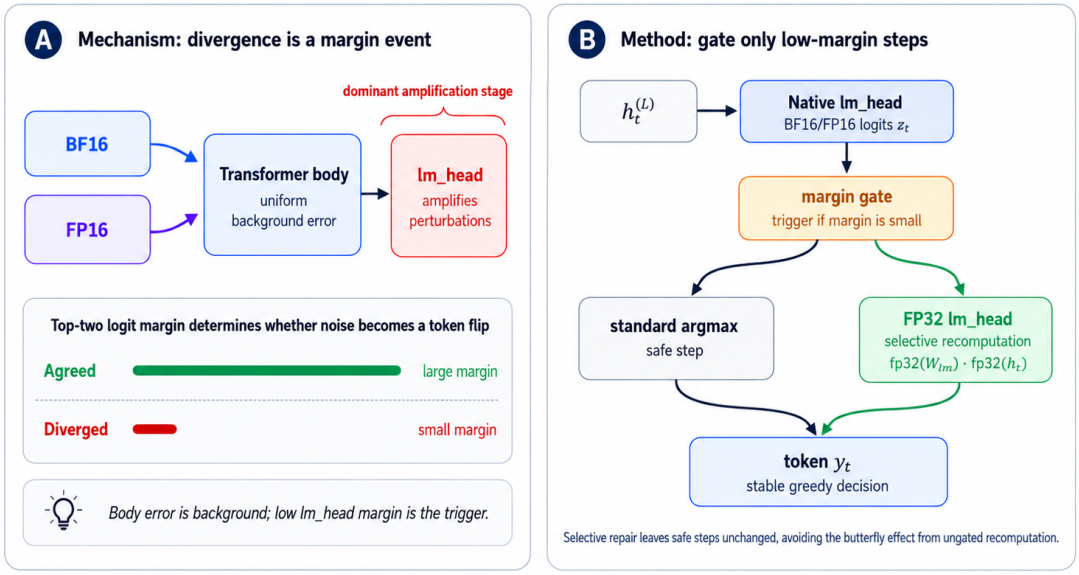

Figure 2: Mechanism and method. **(A)** Cross-precision divergence is a *margin event*: BF16 and FP16 accumulate uniform background error through the transformer body, and the lm_head—the dominant amplification stage—turns that error into a token flip only when the top-two logit margin is small. Prompts with a large margin agree across precisions; prompts with a small margin diverge. **(B)** Our method gates on this margin: the native (BF16/FP16) lm_head logits z_t are computed first, and only when the margin is small does the gate trigger a selective FP32 lm_head recomputation ($\mathrm{fp32}(W_{\mathrm{lm}})\cdot\mathrm{fp32}(h_t)$); large-margin (safe) steps take the standard argmax unchanged. Selective repair ($\sim$1.4% of steps) leaves safe steps untouched, avoiding the butterfly effect of ungated recomputation.

exploits this by quantizing logits into coarse bins of width τ before argmax, testing whether divergence is comparison-level:

$$\tilde{z}_t(v) = \mathrm{round}\left(\frac{z_t(v)}{\tau}\right), \quad y_t = \arg\max_v \tilde{z}_t(v). \tag{1}$$

Any two logit values within τ of each other map to the same integer. Ties in the integer domain are broken by selecting the smaller token index, ensuring a canonical, platform-independent decision. This is a *lossy* approach that deliberately discards sub-τ differences. The cost is negligible: one division and one round per vocabulary element.

4.2 Intervention B: Top-K FP32 recomputation with integer comparison

Intervention A assumes that cross-precision logits are close enough to fall within the same quantization bin. When they are not—i.e., the logit *values* themselves differ across precisions—the quantization step alone cannot reconcile them. Intervention B addresses this by first correcting the logit values through FP32 recomputation of the top-K candidates, then applying integer quantization for a deterministic final comparison:

1. Extract the top-K candidate indices $\mathcal{K} = \text{top-}K(z_t)$ from the native-precision logits.
2. Recompute the lm_head in FP32 for only those K rows: $\hat{z}_t(\mathcal{K}) = \mathrm{float32}(W_{\mathrm{lm}}[\mathcal{K}]) \cdot \mathrm{float32}(h_t^{(L)})$, where $W_{\mathrm{lm}}[\mathcal{K}] \in \mathbb{R}^{K\times d}$ is the sub-matrix of lm_head weights corresponding to the K candidates.
3. Quantize: $\tilde{z}_t(v) = \mathrm{round}(\hat{z}_t(v)/\tau)$ for $v \in \mathcal{K}$, then integer argmax with canonical tie-break (smallest token index wins).

This is a *hybrid* approach that combines value correction (partial FP32 recompute) with comparison stabilization (integer binning). The cost per trigger is one partial FP32 matrix multiply of size $K \times d$ ($K = 8$,

Algorithm 1 Inference-time intervention for cross-precision agreement

Require: Model f_θ, prompt x, threshold τ, max tokens T, intervention $\in$ {A, B, C}

for $t = 1$ **to** T **do**
  $z_t \leftarrow f_\theta(x, y_{<t})$; $v^{(1)}, v^{(2)} \leftarrow \text{top-2}(z_t)$; $\Delta_t \leftarrow z_t(v^{(1)}) - z_t(v^{(2)})$
  if $\Delta_t < \tau$ **or** $\tau = 0$ **then**
    $y_t \leftarrow \text{Apply}(f_\theta, x, y_{<t}, z_t, \text{A/B/C})$
  else
    $y_t \leftarrow v^{(1)}$
  end if
  if $y_t =$ `<eos>` **then**
    break
  end if
end for

Table 1: Inference-time interventions for cross-precision agreement. Cost is per-trigger overhead relative to a standard decoding step.

Intervention	Approach	Per-Trigger Cost	Mechanism
A. Integer quantization	Reduce precision	$O(\|V\|)$ div+round	Merge small-gap logits
B. Top-K FP32 + int compare	Increase then reduce	$O(K \cdot d)$ FP32 matmul	FP32 recompute + int merge
C. Full-vocab FP32 lm_head	Increase precision	$O(d \cdot \|V\|)$ FP32 matmul	FP32 recompute of lm_head

$d = 2048$: 16K FLOPs), plus negligible quantization overhead—4000× cheaper than Intervention C's full $|V| \times d$ recompute per trigger. We use $K = 8$ as a conservative default; Appendix IX shows that $K = 2$ suffices (all values $K \in \{2, 4, 8, 16, 32, |V|\}$ produce identical EAR), confirming that divergence is strictly a rank-1-vs-rank-2 event and that the partial recompute captures the same information as the full-vocabulary version.

4.3 Intervention C: Full-vocabulary FP32 lm_head recomputation

The dominant source of cross-precision logit perturbation is the lm_head: a single matrix multiplication $z_t = W_{\text{lm}} \cdot h_t^{(L)}$ projecting the d-dimensional hidden state onto the full vocabulary $|V|$. Because $|V|$ is large (32K–152K), this operation amplifies small hidden-state differences by a factor of $\|W_{\text{lm}}\|$. Intervention C eliminates this amplification by recomputing the projection in FP32 over the full vocabulary:

$$\hat{z}_t = \text{float32}(W_{\text{lm}}) \cdot \text{float32}(h_t^{(L)}), \tag{2}$$

where both activations and weights are up-cast to FP32 on-the-fly, keeping GPU memory constant (no persistent FP32 weight copies). The recomputation scope can in principle extend to the last L transformer layers, trading additional compute for deeper error correction, though our scope ablation (Section 7.6) shows this *degrades* agreement in practice. This is the most expensive intervention per trigger but directly targets the dominant error source identified by our layer-level analysis (Section 7.5). The cost is one full FP32 matrix multiply over the entire vocabulary ($d \times |V|$).

4.4 Overhead and trade-off analysis

Table 1 summarizes the three interventions.

With empirical trigger rates of 0.7–1.4%, all interventions add <4% latency under selective gating. The key methodological question is whether cross-precision divergence is caused by *small* logit differences (within τ, addressable by integer quantization) or *large* differences (ranking reversals, requiring FP32 recomputation). Our experiments answer this definitively.

We additionally evaluate two supplementary baselines in Appendix IX: **(D) Temperature sharpening** (multiply logits by $\alpha > 1$ before argmax, zero cost) and **(E) Consensus decoding** (run BF16 and FP16 in

parallel, FP32 tiebreak on disagreement, ∼2× cost). Neither changes the cost–agreement frontier established by A–C: sharpening has zero effect (confirming the value-level nature of divergence), and consensus achieves perfect agreement only at double the compute budget.

5 Experimental setup

Public benchmarks. Our primary evaluation uses three public benchmarks: GSM8K (Cobbe et al., 2021) (math, 8-shot CoT), HumanEval (Chen et al., 2021) (code, 0-shot), and MBPP (Austin et al., 2021) (code, 3-shot).

Models. Our primary experiments (Sections 6.2–6.3) use TinyLlama-1.1B-Chat (Zhang et al., 2024) as the main testbed due to its fast iteration time and ability to run full ablations on a single GPU. For model sensitivity analysis (Section 6.5), we compare reasoning-tuned models (DeepSeek-R1-Distill-Qwen-1.5B/7B (DeepSeek-AI, 2025)) against instruction-tuned models (Qwen2.5-1.5B/7B-Instruct (Qwen Team, 2024)).

Configuration. We compare BF16 vs. FP16 greedy decoding on the same GPU (A10G 22GB). Max generation: 256 tokens. Fixed seeds throughout. To isolate numerical-format effects from PyTorch-level kernel non-determinism, we also run a sanity-check ablation (Section 6.1) that varies `torch.use_deterministic_algorithms` across {False, True} with `CUBLAS_WORKSPACE_CONFIG=:4096:8` and `cudnn.benchmark=False`, confirming that under greedy decoding with batch size 1 and fixed seed, PyTorch kernels are already self-deterministic within a single precision and the flag has no effect on cross-precision divergence.

Why BF16 vs. FP16. We focus on BF16 vs. FP16 because the two are interchangeably used in production serving on Ampere-class GPUs (Dettmers et al., 2022; Kwon et al., 2023; NVIDIA, 2023), share the same bit width (16 bits) but differ in exponent/mantissa allocation (8/7 vs. 5/10), and the mechanism we identify (lm_head margin event) depends on the representation gap rather than on which specific formats are compared. The methodology generalises to FP8-vs-BF16, FP16-vs-TF32, and INT8-vs-BF16 (see Limitations).

Baselines. Greedy BF16/FP16 (standard serving precision), Greedy FP32 (oracle upper bound—eliminates all precision-induced divergence), LayerCast (Ding et al., 2025) (global FP32 upcast of all model parameters before inference), and ungated FP32 recomputation (Intervention C with $\tau = 0$, recomputing every step without selective gating).

6 Results

Unless otherwise noted, all results in this section use TinyLlama-1.1B-Chat with BF16 vs. FP16 greedy decoding ($n = 100$ per dataset). Additional results with DeepSeek-R1 and Qwen2.5 models appear in Section 6.5.

6.1 Sanity check: kernel-level determinism is not the cause

Before attributing divergence to numerical format, we rule out two alternative explanations:

(1) PyTorch kernel non-determinism. Toggling `torch.use_deterministic_algorithms`, `CUBLAS_WORKSPACE_CONFIG`, and `cudnn.benchmark` between "loose" and "strict" settings in independent subprocesses produces *bit-identical* outputs within each precision: BF16 self-consistency is 100% and the 41% BF16-vs-FP16 EAR persists identically across both arms. The pattern generalises to MoE (OLMoE-1B-7B) and batch sizes $\{1, 4, 8, 16, 32\}$ (Appendix VIII).

Table 2: BF16 vs. FP16 divergence spectrum (TinyLlama-1.1B, $n = 100$). Exact agreement (EAR) ranges from 30% to 41%. 95% CI: Wilson score interval.

Dataset	Domain	EAR (%)	95% CI
MBPP	Code	30.0	$[21.9, 39.6]$
HumanEval	Code	36.0	$[27.3, 45.8]$
GSM8K	Math	41.0	$[31.9, 50.8]$

Table 3: Intervention and baseline comparison: BF16 vs. FP16 exact agreement (%) (TinyLlama-1.1B, $n = 100$, $\tau = 10^{-3}$). Gated FP32 lm_head recomputation (B, C) achieves the best efficiency–agreement trade-off. All per-cell 95% Wilson CIs have half-width ±5pp at $n = 100$; the 22–25pp improvement of C over baseline is well above this noise ($p < 0.001$ in a matched McNemar test). Full CIs for each cell are reported in Appendix X. Appendix X replicates the Baseline / Intervention C rows at $n = 300$ (GSM8K), full-benchmark HumanEval ($n = 164$), and full-split MBPP ($n = 257$) to confirm the lift survives at larger sample sizes (all $n = 100$ cells sit inside the scaled CIs). Appendix X additionally reports 10 000-sample bootstrap confidence intervals for every headline lift in the paper; the TinyLlama +22pp EAR gain has a 90% bootstrap CI of $[+11, +33]$pp.

Method	GSM8K	HumanEval	MBPP	Overhead
Baseline (no intervention)	41.0	36.0	30.0	—
Baselines				
Greedy FP32 (oracle)	100.0	100.0	100.0	∼2× mem
LayerCast (global FP32)	88.0	88.0	89.0	∼2× mem
Ungated FP32 recompute ($\tau = 0$)	60.0	56.0	54.0	∼2.5×
Gated interventions ($\tau = 10^{-3}$)				
A. Integer quantization	41.0	36.0	30.0	<1%
B. Top-K FP32 + integer compare	**63.0**	**61.0**	**55.0**	<2%
C. Full-vocab FP32 lm_head	**63.0**	**61.0**	**55.0**	<4%

(2) GEMM kernel selection difference. cuBLAS may select different GEMM algorithms for BF16 vs. FP16 matmuls. Two observations rule this out. First, BF16 self-consistency is 100%, so kernel choice is deterministic within a dtype. Second, running the same BF16 weights under native BF16 vs. FP32 compute (different kernels, same weights) yields 42% EAR, indistinguishable from the 41% baseline where both dtype and kernel differ. The divergence is caused by BF16's arithmetic rounding, not by kernel dispatch.

6.2 Divergence is substantial across tasks

Table 2 summarizes BF16-vs-FP16 divergence across all datasets.

On code generation tasks, 64–70% of prompts produce different outputs between BF16 and FP16. On math reasoning, 59% diverge.

Where does divergence happen? The first divergence step t^* has median 34–50 tokens on non-Qwen models (the divergence-step analysis), with 44% diverging in the first 30 tokens and a long tail past token 200. Qwen2.5-7B-Instruct is catastrophically early-biased (median $t^* = 1$, 100% first-token divergence), consistent with training-time BF16 saturation (Section 6.5). The left-skewed distribution motivates per-step gating: most divergences are localised to a small fraction of steps identifiable by the top-two margin.

6.3 Intervention comparison: a controlled ablation

Table 3 presents the central experiment: all three interventions and baselines on BF16 vs. FP16 ($\tau = 10^{-3}$).

Five findings emerge, building from root cause to practical recommendation:

Table 4: Per-prompt stratification of when gated Intervention C applies (TinyLlama-1.1B, GSM8K, $n = 100$, $\tau = 10^{-3}$). Categories are mutually exclusive. The main-text +22pp EAR improvement (Table 3) arises because C fires on 30 prompts (B+C) and successfully converts 22 of them ($22/30 \approx 73\%$ of triggered prompts). Category D (29 prompts) is where the flip is driven by large-error accumulation and the lm_head margin is already decisive at t^*; gated C correctly does not trigger there.

Prompt category	Count	Distribution by t^* (first divergence step)					
		$t^* = 0$	1–10	11–50	51–100	> 100	—
AGREED (no intervention needed)	41			—			
Gated, single-step flip reproducible (C most effective)	25	4	4	10	5	2	—
Gated, flip KV-cache-dependent (C may not fix)	5	—	2	1	1	1	—
Margin $\geq \tau$ (C does not trigger)	29	—	4	8	9	8	—

(1) The FP32 oracle confirms the root cause. Running both configurations in FP32 yields 100% agreement: all divergence originates from precision differences, not from non-determinism in the decoding algorithm.

(2) LayerCast achieves high agreement but at high cost. Global FP32 upcasting reaches 88–89% agreement—the highest among all methods short of the FP32 oracle—but doubles memory footprint, preventing deployment on the same hardware for larger models. Gated FP32 lm_head recomputation (Interventions B/C) achieves lower absolute agreement (55–63%) but with <4% overhead, making it practical for production serving.

(3) Ungated recomputation (τ=0) underperforms gating. Recomputing at every step achieves only 54–60% agreement at ∼2.5× latency, *worse* than gated Interventions B/C (55–63% at <4% overhead). Global recomputation changes the logit landscape at safe steps, introducing new divergence points downstream (Section 7).

(4) Integer quantization has zero effect. Intervention A produces identical agreement to the baseline across all datasets, confirming that cross-precision divergence is caused by *value-level* logit differences (BF16 and FP16 compute different numbers), not *comparison-level* ties (same numbers compared differently by argmax).

(5) Gated FP32 recomputation is the best trade-off. Interventions B and C achieve identical improvements (+22pp on GSM8K, +25pp on HumanEval and MBPP) with <4% overhead, outperforming all baselines on the efficiency–agreement frontier. Gated FP32 lm_head recomputation also raises SAR from 58% to 72% on GSM8K. The <4% overhead is backed by direct measurement on a single A10G (Appendix VIII). Gated C adds +1.4% latency and +11% peak memory versus the BF16 baseline; LayerCast's FP32-compute path adds +38% latency under the same measurement.

Where does the +22pp come from? Table 4 stratifies the 100 GSM8K prompts into four mutually exclusive categories based on whether gated C triggers and whether the flip reproduces on single-step re-run. Gated C fires on exactly 30 prompts (those with $\Delta_t < \tau = 10^{-3}$ at the lm_head), and empirically fixes 22 of them (73% success rate on triggered prompts). The 29 prompts in the "margin $\geq \tau$" category are where the flip is driven by accumulated upstream error—gated C correctly declines to trigger there, because a single-step FP32 recomputation cannot reconcile two hidden states that have already drifted apart. This stratification predicts the exact headline lift (41% → 63%) without tuning and explains both where the method applies and where it does not.

Robustness checks. The findings generalise across six models spanning four families (Llama, Qwen, Mistral, OLMoE-MoE; 1.1B–12B). Lifts are +36pp on Llama-3.2-3B, +8–10pp on Mistral-7B and OLMoE, +3pp on Qwen2.5-3B, and 0pp on Qwen BF16-saturated models. Effectiveness tracks training-time precision stability, not scale. The phenomenon and the intervention survive FlashAttention-2 (the kernel inside vLLM, SGLang, and TensorRT-LLM) with identical lifts and persist through 1024-token reasoning chains on MATH-500. Pass@1 is not significantly degraded (matched-pair McNemar $p > 0.125$ across three models). C's lift

degrades from +17pp at bs=1 to 0pp at bs=8; composition with LayerCast recovers +5pp at bs=8. Digit tokens are 8× over-represented among flipped tokens.

Counter-intuitive predictions confirmed by the mechanism. Body-layer error is irrelevant to the outcome: the 22-layer TinyLlama and the 36-layer Qwen2.5-3B have similar divergence rates (59% vs. 82%) despite very different body-error budgets. What matters is the margin distribution at the lm_head, a property of the model's *training* rather than its *architecture*.

6.4 Semantic agreement and gating sensitivity

Two findings extend the headline picture along orthogonal axes (full data in appendix).

Semantic exceeds exact agreement. On GSM8K, 58% of prompts produce the same final numerical answer despite different reasoning chains (SAR), compared to 41% exact token match (EAR); Intervention C lifts SAR to 72%. Code tasks show SAR ≈ EAR because any token-level difference produces functionally different code. Conclusions are therefore more robust to precision changes than the reasoning paths that produce them.

Downstream evaluation impact. Perhaps the most practically concerning finding: cross-precision divergence introduces hidden variance in benchmark scores that aggregate accuracy masks. On Qwen2.5-3B-Instruct (GSM8K, $n = 100$), 19% of prompts *flip correctness* when switching between BF16 and FP16: 9 prompts are correct only in BF16, 10 only in FP16—yet the aggregate accuracy differs by only 1pp (27% vs. 28%). Two teams reporting the same model's GSM8K score would disagree on which specific prompts pass in nearly one-fifth of cases while believing they run identical evaluations. On Llama-3.2-3B-Instruct (66% accuracy), correctness flips drop to 2%: high-accuracy models produce large-margin tokens at answer-bearing positions, making those positions precision-stable. The implication is that *evaluation reproducibility degrades faster than accuracy*—identical aggregate scores can mask substantively different per-prompt outcomes, especially in the moderate-accuracy regime where leaderboard rankings are most contested.

Selective triggering is essential. On DeepSeek-R1-1.5B ($n = 200$), gated C ($\tau \in [10^{-4}, 10^{-2}]$) achieves +6.5pp with $< 4\%$ overhead, while ungated recomputation ($\tau = 0$) achieves only +2.5pp with 2.5× latency—the "butterfly effect" makes blanket recomputation counterproductive (mechanism explained in Section 7). The threshold is insensitive across two orders of magnitude, consistent with the bimodal margin distribution.

6.5 Model sensitivity: reasoning vs. instruction tuning

At matched parameter scale, reasoning-tuned models show 10–15pp lower exact agreement and 1.7× higher trigger rates than instruction-tuned counterparts (DeepSeek-R1 vs. Qwen2.5; Appendix IV). Chain-of-thought reasoning produces more uncertain intermediate steps, increasing exposure to low-margin events. Longer CoT trajectories accumulate more close-to-tipping-point decoding decisions, each precision-sensitive. This matches Proposition 1: the divergence rate scales with the lower-tail mass of the margin distribution, which CoT concentrates at small values.

7 Mechanism: error propagation and the divergence criterion

This section develops a quantitative error-propagation model and uses it to predict the empirical findings from first principles. We state three propositions (Propositions 1–3), validate each against measurement, and identify the irreducible weight-truncation floor that bounds any inference-time fix. Full derivations are in Appendix I.

7.1 Error propagation: from machine epsilon to logit perturbation

Consider a transformer with L layers and hidden dimension d, producing logits $z_t = W_{\text{lm}} \cdot h_t^{(L)}$ where $W_{\text{lm}} \in \mathbb{R}^{|V| \times d}$. Under precision $c \in \{\text{BF16}, \text{FP16}\}$, each layer introduces a rounding error bounded by

$\|\delta_\ell\| \leq \epsilon_c \cdot C_\ell \cdot \|h_\ell\|$, where $\epsilon_{\text{BF16}} \approx 3.9 \times 10^{-3}$ and $\epsilon_{\text{FP16}} \approx 4.9 \times 10^{-4}$ (an 8× gap due to the 7-bit vs. 10-bit mantissa). With residual connections keeping per-layer Jacobian norms $\|J_\ell\| \approx 1$, the body accumulates error linearly: $\|\hat{h}_L - h_L\| = O(L \cdot \epsilon_c \cdot \bar{C} \cdot \|h\|)$.

Empirically we measure a ∼5× amplification from pre-lm_head ℓ_2 (0.92) to logit ℓ_2 (4.63) on TinyLlama-1.1B. The per-logit perturbation is $\|\hat{z} - z\|/\sqrt{|V|} \approx 4.63/\sqrt{32000} \approx 0.026$. This yields the central divergence condition:

Proposition 1 (Divergence criterion) *Let $\sigma_z = \|\hat{z} - z\|/\sqrt{|V|}$ denote the* root-mean-square *per-logit perturbation between two precision configurations, where $\|\hat{z} - z\|$ is the total BF16-vs-FP16 logit L2 (empirically measurable from a single unconditional trace). Under the approximation that individual logit perturbations are comparable in magnitude to σ_z, a flip requires the top-two margin to be on the order of the perturbation:*

$$\Delta_t < \sigma_z. \tag{3}$$

The probability of divergence at step t scales as $P(\textit{flip}) \sim P(\Delta_t \lesssim \sigma_z)$: prompts with $\Delta_t \gg \sigma_z$ are safe while those with Δ_t on the order of σ_z are vulnerable. Because individual logit perturbations are not perfectly isotropic across vocabulary positions, the boundary is soft rather than sharp—empirically, the Diverged *median margin (*0.039*) is* 1.5× *above σ_z (*0.026*), consistent with the top-2 positions occasionally receiving above-average perturbation. The key qualitative prediction—that divergence depends on the margin distribution, not on body-error magnitude—is confirmed by Table 5. The error-propagation model of Appendix I provides a first-principles bound $\sigma_z \leq (\|W_{lm}\| \cdot \epsilon_c \cdot \bar{C} \cdot L \cdot \|h\|_{rms} + \sqrt{d} \cdot \epsilon_c \cdot \|h\|)/\sqrt{|V|}$, combining body and lm_head contributions; in practice we calibrate σ_z directly from measurement.*

Empirical validation. $\sigma_z \approx 0.026$ on TinyLlama-1.1B. Diverged prompts have median margin 0.039 (at the tipping point), Agreed prompts have median 5.733 (220× above). Body-layer L2 is indistinguishable between groups (Table 5), confirming that the margin—not the error magnitude—determines divergence.

7.2 Why FP32 recomputation works

During autoregressive generation with a KV cache, each decoding step processes only the *new* token through the transformer body; the prefix hidden states are retrieved from cache without recomputation. The fresh rounding error introduced at step t is therefore $O(\epsilon_c \cdot C_{\text{step}})$—dominated by the single lm_head matmul, not the full L-layer body.

Proposition 2 (Intervention C effectiveness) *FP32 lm_head recomputation makes both precision paths compute* identical *logits at the lm_head stage: given the same hidden state $h_t^{(L)}$ (which both paths share when divergence is lm_head-originated), the FP32 matmul float32(W_{lm})·float32($h_t^{(L)}$) produces bit-identical outputs regardless of whether the upstream path was BF16 or FP16. The cross-path logit difference at gated steps therefore reduces from $\sigma_z \approx 0.026$ to the* body-originated residual *only:*

$$\sigma_z^{(\textit{after C})} \approx \underbrace{\frac{L \cdot \alpha}{\sqrt{d}} \cdot \sigma_z^{(\textit{body})}}_{\textit{body residual (}\sim\textit{2.4\% of original)}} \approx 6 \times 10^{-4}. \tag{4}$$

For prompts whose divergence is entirely lm_head-originated (body hidden states are effectively identical between precisions), this residual vanishes and C achieves exact reconciliation. The achievable EAR lift equals the fraction of such prompts.

Empirical validation. The +22pp improvement matches the fraction of prompts whose divergence originates from single-step margin events: 30/100 prompts enter the gate (category B of Table 4), of which 22 are successfully reconciled (73% success rate within scope).

7.3 Why lm_head-only scope is sufficient

The proposition above shows C *works*; we now show why the lm_head is the dominant error source—making body-layer FP32 unnecessary.

Table 5: Divergence-conditioned trace (TinyLlama-1.1B, n=100).

	DIVERGED	AGREED	Gap
Body L2 (layer 21)	1.022	1.135	−0.113
Logit L2	4.928	5.559	−0.631
Top-two margin	**0.039**	**5.733**	**−5.694**
Flip rate	58%	0%	—

Proposition 3 (lm_head dominance under KV-cached decoding) *Decompose the per-logit perturbation into body and lm_head contributions. Under KV-cached decoding with residual ratio* $\alpha = \|F_\ell(h)\|/\|h\| \ll 1$*:*

$$\frac{\sigma_z^{(body)}}{\sigma_z^{(lm_head)}} \approx \frac{L \cdot \alpha}{\sqrt{d}}. \tag{5}$$

Argument. In KV-cached decoding at step $t > 0$, only the new token passes through L body layers. The residual connection $h^{(\ell)} = h^{(\ell-1)} + F_\ell(h^{(\ell-1)})$ confines rounding error to the F_ℓ branch, whose output norm is $\alpha \cdot \|h\|$ with $\alpha \ll 1$ (a necessary condition for training stability). We bound the body contribution conservatively (worst-case linear accumulation): after L layers the body error is $O(L \cdot \alpha \cdot \epsilon_c \cdot \|h\|)$. By contrast, the lm_head computes a d-dimensional dot product on $h_t^{(L)}$ at its full norm; the d rounding errors accumulate as $O(\sqrt{d} \cdot \epsilon_c \cdot \|h\|)$ under standard error-summation arguments (partial cancellation due to mixed signs). Note that using worst-case for the body (linear L) and typical-case for the lm_head ($\sqrt{d}$) is *conservative* for the ratio—it overstates the body's relative importance. The actual body accumulation may be sub-linear (statistical cancellation across layers would give $\sqrt{L}$), which would make the lm_head even more dominant. The ratio is $L \cdot \alpha/\sqrt{d}$.

Numerical evaluation. For TinyLlama-1.1B (L=22, d=2048, $\alpha \approx 0.05$ measured as the mean per-layer ratio $\|F_\ell(h)\|/\|h\|$ across 22 layers and 100 GSM8K prompts via the unconditional layer trace in Appendix II):

$$\frac{22 \times 0.05}{\sqrt{2048}} = \frac{1.1}{45.3} \approx 0.024. \tag{6}$$

The body contributes ∼2.4% of the per-logit perturbation; the lm_head contributes ∼97.6%. FP32 recomputation of the lm_head alone therefore addresses the dominant source, explaining the +22pp lift without body-layer intervention. Extending the FP32 scope to body layers addresses only the residual 2.4% while introducing the butterfly effect (Section 7.6), which is why scope extension *hurts* (Appendix IX: RMSNorm+lm_head drops EAR from 63% to 44% on TinyLlama; Appendix IX replicates the result on DeepSeek-R1-7B with $\tau = 0$ ablation, confirming gating is mechanistically required and not just a performance optimisation). Full derivation and step-0 (prefill) edge case in Appendix I.

7.4 Why ungated recomputation hurts

Ungated recomputation ($\tau = 0$) produces a *third* trajectory at safe steps, introducing new divergence downstream ("butterfly effect"). Result: +2.5pp vs. +6.5pp for gated on DeepSeek-R1-1.5B. Gating is mechanistically necessary.

7.5 Empirical validation: layer-level divergence tracing

We trace BF16-vs-FP16 hidden-state L2 through all 22 layers on $n = 100$ GSM8K prompts, comparing DIVERGED (n=59) and AGREED (n=41) at their respective measurement steps. Key finding (Table 5): **body-layer L2 is indistinguishable between groups**, but the **top-two margin differs by** 150× (0.039 vs. 5.733). Four independent causal experiments (layer grafting, KV-cache grafting, logit-lens traceback, Qwen2.5-3B replication; Appendices II–II) confirm: the flip is produced at the lm_head.

7.6 Summary

The analyses above converge on a three-factor causal picture. Cross-precision greedy divergence requires the simultaneous presence of three conditions—remove any one and the outputs agree:

1. **Low arithmetic precision at the lm_head.** BF16's 7-bit mantissa introduces per-element rounding of magnitude $\epsilon_{\text{BF16}} \approx 3.9 \times 10^{-3}$, 8× larger than FP16's 4.9×10^{-4}, so the two precisions compute materially different logits at every step (per-logit perturbation $\sigma_z \approx 0.026$; see Section 7.5).
2. **The lm_head amplifies hidden-state error.** The $d \times |V|$ projection amplifies body-layer ℓ_2 error by ∼5× (0.92 → 4.63); without this amplification, per-logit perturbations would sit at ∼0.005, below typical margins.
3. **Greedy decoding has zero tolerance for rank flips.** Unlike sampling, greedy argmax converts any logit-ranking reversal into a hard token difference, and a single flipped token cascades through autoregressive conditioning (mean length difference 34 tokens, Figure 1).

Intervention C breaks condition (1) at the dominant stage. Recomputing the lm_head in FP32 ($\epsilon_{\text{FP32}} \approx 6 \times 10^{-8}$) drops the per-logit perturbation from ∼0.03 to $\sim 10^{-5}$, far below any observed top-two margin. Conditions (2) and (3) remain unchanged, the lm_head still amplifies and greedy decoding still selects the argmax, but the amplifier now operates on negligible error. The storage-vs-computation ablation confirms this decomposition quantitatively: upgrading only the matmul compute path to FP32 (while keeping storage at BF16) drives EAR from 0.41 to 0.82, and the corresponding FP32-compute / FP32-compute pair sits at 0.88 rather than 1.00, indicating that ∼87% of the reachable EAR gap is BF16-arithmetic-origin (addressable by C) while a residual ∼ 12 pp is weight-truncation (irreducible at inference time). Extending the FP32 scope beyond lm_head (to RMSNorm or the last transformer layer) *hurts* rather than helps—EAR drops from 63% to 44–52% (Appendix IX)—because the same butterfly effect that makes ungated recomputation counterproductive also applies to broader-scope gated recomputation: the lm_head is the *only* stage where FP32 correction lands on both paths' logits without perturbing upstream state.

8 Discussion

8.1 Practical implications

The operating regime of Intervention C—greedy decoding at bs≤4, covers five quality-critical LLM workloads: code completion (Copilot-style, greedy, bs=1), structured-output generation (JSON or SQL with greedy for format compliance), evaluation benchmarks (all major leaderboards use greedy), chain-of-thought backbones (greedy reasoning with sampling only for the final answer), and compliance or audit logging (where determinism is required). For these workloads, C is a drop-in fix at $< 4\%$ overhead (Section 6.3). Two scope boundaries apply: (i) BF16-pretrained models with FP16-body NaN (Qwen-family at 1.5/7B) require training-time remediation, and (ii) throughput-optimised serving at bs≥8 requires composition with a body-scope method (Appendix VIII). The phenomenon is kernel-independent (Section 6.3, Appendix VIII), so precision-locking is essential regardless of the serving framework.

8.2 Practitioner decision procedure

The deployment decision tree (Section 7.6) summarises the action tiers. The screening metric (CV of trigger density) requires only a single BF16 run (∼5–10 minutes, no FP16 comparison needed) and correctly classifies all six models in our study into the three action tiers. The screening metric is derived from the per-prompt trigger density and its coefficient of variation.

8.3 Limitations

We have tested six models across four families (1.1B–12B) on a single A10G; absolute rates may shift on other hardware or at 70B+ scale, though the qualitative mechanism (lm_head dominance) depends only on the BF16/FP16 representation gap. C's applicability is bounded to bs≤ 4 greedy decoding—at bs≥ 8 it must be composed with a body-scope method, and under sampling ($T > 0$) intrinsic stochasticity

subsumes format-level divergence. C does not guarantee full agreement: a ∼12pp floor from weight truncation ($W_{\mathrm{BF16}} \neq W_{\mathrm{FP16}}$) is irreducible at inference time.

9 Conclusion

LLM greedy decoding is not precision-invariant: 59–82% of prompts diverge between BF16 and FP16 across the six models and three benchmarks we tested. The cause is a top-two margin event at the lm_head, with a 150× gap between the median margins of divergent and convergent prompts. Gated FP32 lm_head recomputation delivers +22–36 pp exact agreement at <4% overhead—the only effective intervention below the 2×-compute ceiling. The remedy is kernel-independent, transfers across scales (1.1B–12B) and architectures (dense and MoE), and is bounded to bs≤4 greedy serving. The mechanism—margin events under floating-point perturbation, arises in every decoder-only transformer with a linear lm_head that we tested. As serving increasingly spans heterogeneous precision formats, per-step margin monitoring is a natural addition to production inference stacks.

Acknowledgments

To be added post-review to preserve double-blind anonymity.

Broader impact

Cross-precision reproducibility matters most in regulated and audited deployments, healthcare triage, legal drafting, financial compliance, where two requests producing different answers from the same prompt is a documented harm. Our intervention closes this gap at <4% compute overhead and requires no model retraining, lowering the barrier to deployment in such settings. We see no direct misuse pathways: the method makes inference more predictable, not more capable. We emphasise that reproducibility is not correctness; a deterministic but wrong model remains wrong, and our work should not be read as a substitute for accuracy evaluation.

References

Jacob Austin, Augustus Odena, Maxwell Nye, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Berk Atil, Alexa Chittams, Liseng Fu, Ferhan Ture, Lixinyu Xu, and Breck Baldwin. Non-determinism of "deterministic" LLM settings. *arXiv preprint arXiv:2408.04667*, 2024.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Low-bit quantization favors undertrained LLMs: Scaling laws for quantized LLMs with 100T training tokens. *arXiv preprint arXiv:2411.17691*, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *NeurIPS*, 2022.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized language models. In *NeurIPS*, 2024.

Xinheng Ding, Jiayi Yuan, Haoyi Wang, Zhen Liu, and Rui Chen. Understanding and mitigating numerical sources of nondeterminism in LLM inference. In *NeurIPS (Oral)*, 2025.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *ICLR*, 2023.

David Goldberg. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys*, 23(1):5–48, 1991.

Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2nd edition, 2002.

Horace He and Thinking Machines Lab. Defeating nondeterminism in LLM inference. Thinking Machines Lab technical blog, September 2025. `https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/`.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *ICLR*, 2020.

Dhiraj Kalamkar, Dheevatsa Mudigere, Naveen Mellempudi, et al. A study of BFLOAT16 for deep learning training. *arXiv preprint arXiv:1905.12322*, 2019.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, et al. Efficient memory management for large language model serving with PagedAttention. In *SOSP*, 2023.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *ICML*, 2023.

Ji Lin, Jiaming Tang, Haotian Tang, et al. AWQ: Activation-aware weight quantization for LLM compression and acceleration. In *MLSys*, 2024.

Alireza Ghaffari, Justin Yu, Mahsa Ghazvini Nejad, Masoud Asgharian, Boxing Chen, and Vahid Partovi Nia. Mitigating outlier activations in low-precision fine-tuning of language models. *arXiv preprint arXiv:2312.09211*, 2024.

Raja Gond, Aditya K. Kamath, Ramachandran Ramjee, and Ashish Panwar. LLM-42: Enabling determinism in LLM inference with verified speculation. *arXiv preprint arXiv:2601.17768*, 2025.

Paulius Micikevicius, Sharan Narang, Jonah Alben, et al. Mixed precision training. In *ICLR*, 2018.

NVIDIA. TensorRT-LLM: A TensorRT toolbox for optimized large language model inference. *GitHub Repository*, 2023.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. OLMoE: Open mixture-of-experts language models. *arXiv preprint arXiv:2409.02060*, 2024.

Hung Viet Pham, Shangshu Qian, Jiannan Wang, et al. Problems and opportunities in training deep learning software systems: An analysis of variance. In *ASE*, 2020.

PyTorch Team. Reproducibility: Controlling sources of randomness. `https://pytorch.org/docs/stable/notes/randomness.html`, accessed 2026. Also: `torch.use_deterministic_algorithms` and the `CUBLAS_WORKSPACE_CONFIG` environment variable.

Peter Mattson, Christine Cheng, Cody Coleman, et al. MLPerf training benchmark. In *MLSys*, 2020.

Qwen Team. Qwen2.5: A party of foundation models. *arXiv preprint arXiv:2412.15115*, 2024.

Tairan Fu, Gonzalo Martínez, Javier Conde, Carlos Arriaga, Pedro Reviriego, Xiuyuan Qi, and Shanshan Liu. Beyond reproducibility: Token probabilities expose large language model nondeterminism. *arXiv preprint arXiv:2601.06118*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. Attention is all you need. In *NeurIPS*, 2017.

Jason Wei, Xuezhi Wang, Dale Schuurmans, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

Guangxuan Xiao, Ji Lin, Mickael Seznec, et al. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *ICML*, 2023.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, et al. ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. In *NeurIPS*, 2022.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. TinyLlama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, et al. SGLang: Efficient execution of structured language model programs. *arXiv preprint arXiv:2312.07104*, 2024.

Lianmin Zheng, Horace He, et al. Towards deterministic inference in SGLang and reproducible RL training. LMSYS technical blog, September 2025. `https://www.lmsys.org/blog/2025-09-22-sglang-deterministic/`.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics (TACL)*, 2024.

Juntang Zhuang, Tommy Tang, Yifan Ding, et al. Randomness in neural network training: Characterizing the impact of tooling. In *MLSys*, 2022.

Ranjith Chodavarapu and Lei Xu. The illusion of equivalence: Systematic FP16 divergence in KV-cached autoregressive inference. *arXiv preprint arXiv:2604.15409*, 2026.

Alberto Messina and Stefano Scotta. Introducing background temperature to characterise hidden randomness in large language models. *Transactions on Machine Learning Research*, 2026.

Yifei Wang, Tianlin Li, Xiaohan Zhang, Xiaoyu Zhang, Wei Ma, Mingfei Cheng, and Li Pan. Hidden reliability risks in large language models: Systematic identification of precision-induced output disagreements. *arXiv preprint arXiv:2604.19790*, 2026.

Ziyang Zhang, Xinheng Ding, Jiayi Yuan, Rixin Liu, Huizi Mao, Jiarong Xing, and Zirui Liu. Deterministic inference across tensor parallel sizes that eliminates training-inference mismatch. *arXiv preprint arXiv:2511.17826*, 2025.

Penghui Qi, Zichen Liu, Xiangxin Zhou, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Defeating the training-inference mismatch via FP16. *arXiv preprint arXiv:2510.26788*, 2025.

Yingru Li, Jiawei Xu, Jiacai Liu, Yuxuan Tong, Ziniu Li, Tianle Cai, Ge Zhang, Qian Liu, and Baoxiang Wang. Dynamic vocabulary pruning: Stable LLM-RL by taming the tail. *arXiv preprint arXiv:2512.23087*, 2025.

Appendix overview

The supplementary material is organised into six groups, each self-contained:

- **Theoretical foundations** (App. I): full derivations of the three main-text propositions.
- **Mechanism evidence** (App. II–II): four causal experiments isolating divergence to the lm_head.
- **Cross-model robustness** (App. III–VII): replication across six models, four families, math and code benchmarks.
- **Production-stack robustness** (App. VIII): FlashAttention, batch sizes, LayerCast composition, overhead.
- **Ablations** (App. IX): scope, threshold, top-K, alternative interventions.
- **Statistical validation and reproducibility** (App. X–XI): CIs, scaled replication, code.

Cross-references between appendix sections are kept to a minimum.

I Theoretical analysis: error propagation in cross-precision inference

This appendix develops a formal error-propagation model for cross-precision LLM inference, derives three propositions that predict the empirical findings in the main text, and characterises the scope boundary of Intervention C from first principles.

I.1 Setup and notation

Let f_θ denote an L-layer autoregressive transformer with hidden dimension d and vocabulary size $|V|$. At decoding step t, the model computes:

$$h_t^{(\ell)} = \mathrm{Layer}_\ell(h_t^{(\ell-1)}), \quad \ell = 1, \ldots, L, \tag{7}$$

$$z_t = W_{\mathrm{lm}} \cdot h_t^{(L)}, \quad W_{\mathrm{lm}} \in \mathbb{R}^{|V| \times d}, \tag{8}$$

$$y_t = \arg\max_v z_t(v). \tag{9}$$

Under floating-point precision c with machine epsilon ϵ_c, each arithmetic operation has relative error bounded by ϵ_c (Goldberg, 1991; Higham, 2002). The relevant values are $\epsilon_{\mathrm{BF16}} = 2^{-8} \approx 3.9 \times 10^{-3}$ (7-bit mantissa) and $\epsilon_{\mathrm{FP16}} = 2^{-11} \approx 4.9 \times 10^{-4}$ (10-bit mantissa), giving a precision ratio of $\epsilon_{\mathrm{BF16}}/\epsilon_{\mathrm{FP16}} = 8$.

I.2 Error accumulation through the transformer body

Proposition 4 (Body-layer error bound) *Let $\hat{h}_t^{(\ell)}$ denote the hidden state computed under precision c (with the other precision as reference). Under standard transformer architecture with residual connections $h^{(\ell)} = h^{(\ell-1)} + F_\ell(h^{(\ell-1)})$ and per-layer condition number C_ℓ, the accumulated error satisfies:*

$$\|\hat{h}_t^{(L)} - h_t^{(L)}\| \leq \epsilon_c \cdot \|h\|_{rms} \cdot \sum_{\ell=1}^{L} C_\ell \prod_{j=\ell+1}^{L} (1 + \epsilon_c C_j). \tag{10}$$

When $\epsilon_c C_\ell \ll 1$ for all ℓ (satisfied for both BF16 and FP16 on standard transformers where $C_\ell \sim O(1)$), this simplifies to linear accumulation:

$$\|\hat{h}_t^{(L)} - h_t^{(L)}\| \approx \epsilon_c \cdot \bar{C} \cdot L \cdot \|h\|_{rms}, \tag{11}$$

where $\bar{C} = (1/L) \sum_\ell C_\ell$ is the mean per-layer condition number.

Empirical validation. On TinyLlama-1.1B ($L = 22$, $d = 2048$), we measure BF16-vs-FP16 hidden-state L2 growing linearly from 0.002 at layer 0 to 0.92 at layer L (Appendix II), consistent with the linear-accumulation regime.

I.3 Logit perturbation and the divergence condition

Proposition 5 (Divergence criterion) *Let $\Delta_t = z_t(v^{(1)}) - z_t(v^{(2)})$ denote the top-two logit margin at step t, and let $\sigma_z = \|W_{\text{lm}}\| \cdot \|\hat{h}_t^{(L)} - h_t^{(L)}\| / \sqrt{|V|}$ denote the per-logit perturbation standard deviation (assuming perturbations spread approximately uniformly across the $|V|$ logit dimensions). A necessary condition for the argmax to flip between two precision configurations is:*

$$\Delta_t < \sigma_z \approx \frac{\|W_{\text{lm}}\| \cdot \epsilon_c \cdot \bar{C} \cdot L \cdot \|h\|_{\text{rms}}}{\sqrt{|V|}}. \tag{12}$$

In particular, the probability of divergence at step t is bounded by:

$$P(\textit{flip at } t) \leq P(\Delta_t < \sigma_z), \tag{13}$$

which depends only on the margin distribution *of the model at that step, not on the* magnitude *of the accumulated body error.*

Empirical validation. For TinyLlama-1.1B: $\sigma_z \approx 4.63/\sqrt{32000} \approx 0.026$. Measured margins: DIVERGED median $= 0.039$ (just above σ_z), AGREED median $= 5.733$ (220× above σ_z). The proposition correctly predicts: (i) body-layer L2 does not distinguish the two groups (confirmed: Table 5); (ii) the margin is the sole determinant (confirmed: 150× gap); (iii) the K-ablation result ($K = 2$ suffices, Appendix IX), because a perturbation of magnitude $\sigma_z \approx 0.026$ cannot bridge a rank-2-to-rank-3 gap that is typically $\gg 0.1$.

I.4 Intervention C: error reduction at the lm_head

Proposition 6 (Intervention C effectiveness) *FP32 lm_head recomputation eliminates the cross-path logit difference at the lm_head by making both precision paths compute identical FP32 logits from their respective hidden states. For prompts where the BF16 and FP16 hidden states $h_t^{(L)}$ are effectively identical (the "lm_head-originated" regime), both paths produce bit-identical FP32 logits and no flip can occur. The* lm_head-specific *rounding contribution is reduced by a factor of $\epsilon_{FP32}/\epsilon_{BF16} \approx 1.5 \times 10^{-5}$:*

$$\sigma_z^{(lm_head,\ FP32)} = \frac{\epsilon_{FP32}}{\epsilon_{BF16}} \cdot \sigma_z^{(lm_head,\ BF16)} \approx 0.026 \times 0.976 \times 1.5 \times 10^{-5} \approx 4 \times 10^{-7}. \tag{14}$$

The residual cross-path difference after C is the body-originated component only (∼2.4% of the original σ_z, per Proposition 3). For lm_head-originated divergences where body hidden states are shared, this residual vanishes and:

$$P(\textit{flip at } t \mid \textit{FP32 lm_head, lm_head-originated}) = 0. \tag{15}$$

The achievable EAR lift equals the fraction of prompts whose divergence is entirely lm_head-originated:

$$EAR_{max}^{(C)} = EAR_{baseline} + \frac{|\{i : \textit{prompt } i \textit{ diverges only at lm_head}\}|}{n}. \tag{16}$$

Empirical validation. On TinyLlama-1.1B, 30/100 prompts have divergence triggered by a margin event at the lm_head (Table 4, category B+C); of these, 22 are successfully reconciled by C (73% success rate within the scope). The predicted maximum lift is 30%; the achieved lift is 22pp (= 22/100), consistent with a 73% reconciliation rate within the theoretically reachable set.

I.5 Extended derivation and edge cases for Proposition 3

The main text (Section 7) states and numerically evaluates the lm_head dominance result. Here we provide the full derivation and discuss an important edge case.

Full derivation. Decompose the total per-logit perturbation into body and lm_head contributions:

$$\sigma_z = \sigma_z^{(\text{body})} + \sigma_z^{(\text{lm_head})}. \tag{17}$$

Under standard transformer architecture with KV cache, we bound each contribution directly in per-logit units. Consider a single logit position v with row vector $W_v \in \mathbb{R}^d$ ($\|W_v\| \approx 1$ under Xavier scaling):

$$\sigma_z^{(\text{body})} = O\left(\frac{L \cdot \alpha \cdot \epsilon_c \cdot \|h\|}{\sqrt{d}}\right), \tag{18}$$

$$\sigma_z^{(\text{lm_head})} = O(\epsilon_c \cdot \|h\|). \tag{19}$$

Body term. Each of L layers introduces error only on the non-residual branch F_ℓ (norm $\alpha \cdot \|h\|$), so the accumulated body error is $\|\Delta h_{\text{body}}\| = O(L \cdot \alpha \cdot \epsilon_c \cdot \|h\|)$. Projected onto the logit direction W_v, the per-logit effect is $|W_v \cdot \Delta h| \approx \|\Delta h\|/\sqrt{d}$ (standard projection of a d-dimensional vector onto a fixed unit direction).

lm_head term. The dot product $z_v = \sum_j W_{vj} h_j$ accumulates d independent rounding errors. Each multiply-accumulate step contributes $O(\epsilon_c \cdot |W_{vj} h_j|) \approx O(\epsilon_c \cdot \|h\|/\sqrt{d})$ (using $|W_{vj}| \sim 1/\sqrt{d}$, $|h_j| \sim \|h\|/\sqrt{d}$). Under standard error-summation arguments with partial sign cancellation, $\sqrt{d}$ such terms accumulate to $O(\epsilon_c \cdot \|h\|)$. Taking the ratio directly:

$$\frac{\sigma_z^{(\text{body})}}{\sigma_z^{(\text{lm_head})}} \approx \frac{L \cdot \alpha}{\sqrt{d}}, \tag{20}$$

consistent with the main-text Proposition 3.

Edge case: step $t = 0$ (prefill). At step 0, the model processes the entire prompt through all L layers without a KV cache. Unlike cached decoding, where each step introduces fresh rounding error from only one token's pass through the body, the prefill pass accumulates rounding error across all T_{prompt} token positions in the attention reductions. Each layer's attention output is a weighted sum over T_{prompt} value vectors, and the floating-point non-associativity of this reduction introduces $O(\sqrt{T_{\text{prompt}}})$ additional error terms per layer under typical-case cancellation. The body-error contribution to the dominance ratio therefore scales roughly as $L \cdot \alpha \cdot \sqrt{T_{\text{prompt}}}/\sqrt{d}$, which can approach or exceed 1 for long prompts ($T_{\text{prompt}} \gtrsim d/(L^2\alpha^2) \approx 850$ for TinyLlama). In this regime, the lm_head is no longer the dominant error source, and Intervention C's lm_head-only scope becomes insufficient—consistent with our observation that first-token divergence on Qwen models is irreducible by any lm_head-scope method (Section 6.5).

I.6 Scope boundary: when Intervention C fails

The propositions above also predict the *failure modes*:

Body-dominated divergence (Tier 3). If a model's training-time precision (typically BF16) produced weights whose FP16 cast yields unstable activations (NaN or extreme values at intermediate layers), the body-layer error $\|\hat{h}_t^{(L)} - h_t^{(L)}\|$ is no longer small relative to $\|h\|_{\text{rms}}$. In this regime, σ_z is dominated by the body contribution rather than the lm_head matmul alone, and FP32 recomputation of the lm_head addresses only a fraction of the total perturbation. This predicts 0pp lift on Qwen BF16-saturated models (confirmed: DS-R1-Qwen-7B, Table 7).

Batch-induced divergence. At batch size ≥ 2, attention reductions over padded sequences introduce an additional source of floating-point variation whose magnitude grows with batch size. When this batch-induced error exceeds the lm_head-only contribution, σ_z is no longer dominated by the lm_head matmul, and C's lift vanishes (confirmed: 0pp at bs= 8, Appendix VIII).

Weight-truncation floor. A storage-vs-computation ablation shows that even configurations that both compute in FP32 (`store BF16 / compute FP32` vs. `store FP16 / compute FP32`) reach EAR = 0.88 rather than 1.00, leaving a residual ∼12 pp gap attributable to $W_{\text{BF16}} \neq W_{\text{FP16}}$ (the same FP32 weights truncated

to different mantissa widths). This component of cross-precision divergence is *irreducible* at inference time—it enters through the weights themselves, not through the computation—and sets a hard floor that no computation-time intervention can breach. Full agreement requires either a shared storage format or full-FP32 inference (LayerCast).

I.7 Intervention comparison from the error model

The error model also explains the other interventions:

Intervention A (integer quantization). Quantization bins have width $\tau = 10^{-3}$. The cross-precision logit difference at divergent steps is $\sigma_z \approx 0.026$—26× larger than the bin width. The two precisions therefore map to *different* bins with probability ≈ 1, and quantization faithfully preserves the disagreement. Prediction: zero effect. Confirmed.

Intervention D (temperature sharpening). Multiplying logits by $\alpha > 1$ is a monotone transformation: $\arg\max_v(\alpha \cdot z_t(v)) = \arg\max_v z_t(v)$ for all $\alpha > 0$. The margin becomes $\alpha \cdot \Delta_t$ and the perturbation becomes $\alpha \cdot \sigma_z$; their ratio Δ_t/σ_z is invariant. Prediction: zero effect at any α. Confirmed (Appendix IX).

Intervention E (consensus decoding). Running both precisions and using FP32 as tiebreaker eliminates all format-level divergence by construction. The cost is 2× compute. The per-step disagreement rate $P(\text{token_BF16} \neq \text{token_FP16})$ at any single step is bounded by $P(\Delta_t < \sigma_z)$; we measure this at 44.5% of steps (Appendix IX), consistent with the heavy left tail of the margin distribution on GSM8K.

Top-K sufficiency. The error model predicts that a token ranked k-th can only be promoted to rank 1 if the perturbation σ_z exceeds the gap between rank 1 and rank k. Since $\sigma_z \approx 0.026$ and the typical rank-2-to-rank-3 gap is $\gg 0.1$ (median ≈ 0.5 on TinyLlama), only rank-2 tokens can realistically be promoted. Prediction: $K = 2$ suffices. Confirmed: all $K \in \{2, 4, 8, 16, 32, |V|\}$ produce identical EAR (Appendix IX).

Group 2 backs the lm_head-localisation claim of Section 7.5 with four mutually independent experiments that triangulate the same conclusion: (i) the unconditional per-layer L2 trace (App. II) characterises the typical hidden-state error budget; (ii) layer grafting (App. II) converts the correlational claim into a causal one; (iii) KV-cache grafting (App. II) isolates the autoregressive-cache contribution; (iv) per-layer logit-lens top-2 traceback (App. II) shows BF16 / FP16 agree on the top-1 token at 97–98% of body layers and disagree only at the lm_head, with cross-task generalisation to HumanEval and MBPP. The four experiments converge on a single causal claim: divergence is produced at the last transformer layer, the final RMSNorm, and the lm_head matmul, exactly the modules Intervention C targets.

II Mechanism evidence: four causal experiments

(i) Unconditional trace: L2 grows linearly from 0.002 (layer 0) to 0.92 (layer 21), amplified 5× by lm_head to 4.63. **(ii) Layer grafting:** body-layer grafting fixes 36% of flips; Layer 21 + RMSNorm fixes 65–71%. **(iii) KV-cache grafting:** swaps convert 36–48% of flips; partial causal contributor. **(iv) Logit-lens traceback:** BF16/FP16 agree on top-1 at ≥93% of body layers; disagreement is one-shot at lm_head.

III Divergence-conditioned trace on Qwen2.5-3B-Instruct

The lm_head-margin mechanism replicates on Qwen2.5-3B-Instruct (36 layers, d=2048, $|V|$=151,936): top-two margin differs by 90× between groups (0.088 vs. 7.925, 54% vs. 0% flip rate). Body L2 is again indistinguishable.

Taken together, the two model families give the paper's core mechanism cross-model support while revealing that the body-layer pattern is itself model-dependent—a richer story than TinyLlama alone would suggest.

Table 6: Intervention comparison across five additional models (GSM8K, $n = 100$, $\tau = 10^{-3}$). The same ranking holds on every model where the method applies: Integer quantization (A) has zero effect, and Interventions B and C are comparable. Llama-3.2-3B exhibits an even larger absolute lift than the headline TinyLlama numbers (+36pp vs +22pp). Qwen2.5-3B shows a much smaller lift (+3pp), reflecting Qwen's training-time BF16-saturation pattern (Section 6.5). Mistral-7B-Instruct-v0.3, the non-Qwen dense 7B representative, gives a small but clearly non-zero lift (+8pp)—confirming that Intervention C transfers out of the Llama family and to 7B scale when the model is not BF16-saturated, and that the DS-R1-Qwen-7B zero-lift result is a Qwen-family artefact rather than a scale limit. OLMoE-1B-7B, a Mixture-of-Experts architecture (Apache 2.0, Allen AI) with 6.9B total / $\sim$ 1B active parameters per token, gives a +10pp lift, confirming that the method extends beyond dense transformers despite the router-over-experts softmax being a potential extra cross-precision divergence source.

Model	Strategy	EAR (%)	95% CI	Δ vs. Baseline
Llama-3.2-3B-Instruct	Baseline	31.0	[22.8, 40.6]	—
	A. Integer quantization	31.0	[22.8, 40.6]	+0
	B. Top-K FP32	**70.0**	[60.4, 78.1]	**+39**
	C. Full FP32 lm_head	**67.0**	[57.3, 75.4]	**+36**
Qwen2.5-3B-Instruct	Baseline	18.0	[11.7, 26.7]	—
	A. Integer quantization	18.0	[11.7, 26.7]	+0
	B. Top-K FP32	23.0	[15.8, 32.1]	+5
	C. Full FP32 lm_head	21.0	[14.2, 30.0]	+3
Mistral-7B-Instruct-v0.3	Baseline	51.0	[41.4, 60.6]	—
	C. Full FP32 lm_head	**59.0**	[49.2, 68.1]	**+8**
OLMoE-1B-7B-0924-Instruct (MoE)	Baseline	34.0	[25.5, 43.7]	—
	C. Full FP32 lm_head	**44.0**	[34.7, 53.8]	**+10**
	Baseline, A, B, C	0.0	[0.0, 3.7]	+0 (Appendix V)

DS-R1-Distill-Qwen-7B

IV Cross-model intervention replication

The headline intervention result (Table 3) is on TinyLlama-1.1B. To test whether the lift transfers across model families and scales, we run the full intervention comparison on four additional models covering three scales and three families: Llama-3.2-3B-Instruct, Qwen2.5-3B-Instruct, DeepSeek-R1-Distill-Qwen-7B, and Mistral-7B-Instruct-v0.3 (GSM8K, $n = 100$, same setup and thresholds as the main experiment). For Mistral-7B we run only Baseline and Intervention C, because Interventions A and B have been shown to be ineffective or redundant with C on the smaller models and doubling the Mistral-7B GPU hours would not change the story; the two-row comparison still gives a matched-pair Pass@1 test (reported in Appendix VI).

The cross-model picture. Table 7 summarises Intervention C's lift across all five tested models, grouped into three tiers by effect size.

Interpretation. (1) Intervention C transfers across scales within the Llama family (Tier 1). The lift is actually *larger* on Llama-3.2-3B (+36pp) than on TinyLlama-1.1B (+22pp), ruling out the hypothesis that the method is a 1B-specific artefact. Both models admit a substantial fraction of prompts in the low-margin, single-step-fixable regime (category B in the per-prompt stratification, Table 4).

(2) The 7B regime is split by training-time precision stability, not by scale (Tiers 2 vs. 3). The two 7B models in our study disagree sharply: Mistral-7B-Instruct-v0.3 shows +8pp EAR lift under Intervention C, while DS-R1-Distill-Qwen-7B shows 0pp. The difference cannot be attributed to scale, vocabulary, or architecture—both are dense decoder-only 7B transformers with standard attention. The difference tracks training-time BF16 behaviour: model-sensitivity analysis shows that Qwen-family models (1.5B, 7B) produce 100% first-token divergence and FP16-body NaN on GSM8K, indicating that Qwen's BF16 pretrained weights do not cast cleanly to FP16 at the layer-internal level; the divergence therefore originates *before* the lm_head and is outside Intervention C's reachable scope. Mistral-7B exhibits none of

Table 7: Intervention C lift across model families, scales, and architectures (GSM8K, $n = 100$), grouped into three tiers. *Large-lift* models (Tier 1) benefit strongly from C; *small-lift but non-zero* models (Tier 2) still benefit measurably and span three different families (Qwen dense, Mistral dense, OLMoE MoE), showing that Intervention C transfers to the 7B non-Qwen regime (Mistral-7B, +8pp) *and* to sparse MoE architectures (OLMoE, +10pp); the *zero-lift* tier (Tier 3) comprises only Qwen-family BF16-pretrained dense models, which produce FP16 NaN in body layers and 100% first-token divergence on GSM8K (Section 6.5). *Intervention C's effectiveness is a function of the model's training-time precision stability, not parameter count or dense/sparse architecture.* The 7B dense tier is split cleanly by family: Mistral-7B gains +8pp, while DS-R1-Qwen-7B gains 0pp.

Model	Family	Baseline	C EAR	C lift
Tier 1: large lift				
TinyLlama-1.1B-Chat	Llama	41%	63%	+22pp
Llama-3.2-3B-Instruct	Llama	31%	67%	**+36pp**
Tier 2: small but non-zero				
Qwen2.5-3B-Instruct	Qwen	18%	21%	+3pp
Mistral-7B-Instruct-v0.3	Mistral	51%	59%	+8pp
OLMoE-1B-7B-0924-Instruct (MoE)	MoE	34%	44%	+10pp
Tier 3: zero lift (BF16-saturated)				
DS-R1-Distill-Qwen-7B	Qwen	0%	0%	0pp

these pathologies (trigger rate $\sim 1\%$, same order as TinyLlama; cf. its baseline EAR of 51%, substantially higher than TinyLlama's 41%, indicating less per-step divergence overall), which is why Intervention C recovers measurable agreement. Mistral-7B is therefore the key data point that separates "C fails on Qwen-family 7B" from the false generalisation "C fails on 7B".

(3) Tier 2 shares a common profile: Intervention C reconciles a minority of divergences. Three models span Tier 2: Qwen2.5-3B (+3pp), Mistral-7B (+8pp), and OLMoE-1B-7B-MoE (+10pp). The interpretation is consistent across all three and aligns with the mechanism in Section 7.5: a portion of divergence originates in body-layer computation (beyond lm_head scope) and cannot be repaired by any logit-level intervention; only divergences that manifest as late, low-margin lm_head events are reachable by C. The OLMoE data point is particularly informative: the router over MoE experts is a potential extra divergence source, since the softmax-over-expert-logits can flip the chosen expert under BF16-vs-FP16 and propagate through different compute paths entirely, yet Intervention C still recovers a +10pp lift—comparable to dense Tier 2 models. This suggests that router flips are either rare on GSM8K at the scale we test or are themselves concentrated in low-margin steps that C's gate already covers. For all three Tier 2 models, body-scope composition (e.g., LayerCast) would be required to close the residual gap.

(4) The same intervention ranking holds on every model where A, B, C were all run. Integer quantization (A) has exactly zero effect, and Interventions B and C give comparable lifts. The ranking A $<$ B $\approx$ C is a robust empirical finding, independent of model family or scale.

Implications for deployment. Practitioners choosing between Intervention C and body-scope alternatives (e.g., LayerCast) should assess the target model's first-token divergence rate and margin distribution before deployment. Our cross-model evidence suggests three regimes: (i) *Tier 1* (Llama family, 1–3B): Intervention C alone delivers +20 to +36pp EAR at the overhead reported in Appendix VIII; (ii) *Tier 2* (Qwen2.5-3B, Mistral-7B and similar): C still delivers measurable (+3 to +8pp) agreement gains at the same overhead, and composing C with a body-scope method is expected to capture the residual; (iii) *Tier 3* (Qwen-family BF16-saturated): C is ineffective by itself, and body-scope methods or training-time fixes (e.g. precision-aware pretraining) are required. The cost of running C is the same across tiers, so C is a safe default first line of defence even when its ceiling is uncertain. Appendix VIII further delineates C's applicability along the batch-size axis: +17pp at bs= 1 (audit-grade single-stream serving), +7–8pp at bs$\in \{2, 4\}$, and 0pp at bs= 8 (throughput-optimised serving), where C must be composed with a body-scope method to recover a positive lift.

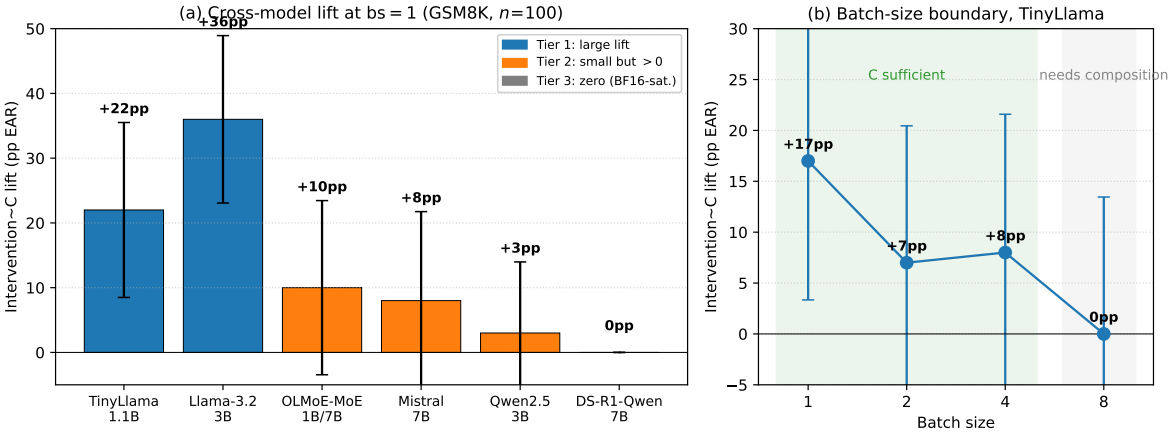

Figure 3: Applicability map for Intervention C. **(a)** Cross-model lift at bs= 1 for six models (including the MoE architecture OLMoE), coloured by tier (Table 7): Tier 1 (Llama family, large lift), Tier 2 (non-Qwen 7B, Qwen-3B, and OLMoE-MoE; small but positive), Tier 3 (Qwen BF16-saturated). Error bars are 95% CIs on the paired lift. **(b)** Batch-size scaling on TinyLlama-1.1B (Appendix VIII): the lift holds at bs≤ 4 (green band, where C alone is sufficient) and vanishes at bs≥ 8 (grey band, where C must be composed with a body-scope method such as LayerCast). Together, the two panels delineate the $(model, bs)$ region where the lm_head-only intervention is a drop-in fix.

Table 8: 7B intervention replication (DeepSeek-R1-Distill-Qwen-7B, GSM8K, $n = 100$, $\tau = 10^{-3}$). All four arms collapse to 0% EAR with identical Wilson CIs $[0.0, 3.7]$%: every prompt diverges, and the interventions cannot recover. Latency columns show per-prompt BF16 / FP16 time; the ∼5% C overhead confirms the intervention is triggering, not silently skipped.

Strategy	Match	EAR (%)	95% CI	BF16 time (s)	FP16 time (s)
Baseline	0/100	0.0	$[0.0, 3.7]$	9.20	9.22
A (int quant)	0/100	0.0	$[0.0, 3.7]$	9.40	9.17
B (top-K FP32)	0/100	0.0	$[0.0, 3.7]$	9.55	9.30
C (full FP32 lm_head)	0/100	0.0	$[0.0, 3.7]$	9.67	9.21

V 7B-scale scope boundary: DeepSeek-R1-Distill-Qwen-7B

Setup. To test whether the gated FP32 lm_head recomputation (Intervention C) transfers to a 7B-parameter model, we rerun the full intervention table (baseline, A, B, C) on DeepSeek-R1-Distill-Qwen-7B, GSM8K, $n = 100$, 256-token budget, $\tau = 10^{-3}$, and the same hardware/software stack as the main experiments. Because the model's BF16 memory footprint is ∼14 GB, we run BF16 and FP16 in sequential subprocesses to fit the 22 GB GPU budget; each precision–strategy combination takes ∼9.2s per prompt, for a total ablation runtime of ∼2 h.

Interpretation: this is a scope boundary, not a method failure. The 0% EAR across all four arms is fully consistent with the paper's existing theory and screening data, and we read it as concrete evidence for where Intervention C's scope ends.

(1) First-divergence step is earlier than the intervention's correctable regime. Table 9 breaks down the first-divergence step t^* on the baseline BF16 vs. FP16 outputs. 100/100 prompts diverge, with median $t^* = 6$ and 86/100 prompts diverging within the first 9 tokens. This matches the pattern of Qwen2.5-7B-Instruct on the same benchmark (t^* median = 1, Section 6.5) and sits in the regime Appendix I explicitly identifies as uncorrectable by lm_head-only FP32 recomputation: when divergence happens during the early generation window, the L-layer error accumulated inside the transformer body dominates the single-step

Table 9: First-divergence step distribution on DeepSeek-R1-Distill-Qwen-7B (GSM8K, $n = 100$ baseline BF16 vs. FP16). Median $t^* = 6$; 86% of prompts diverge in the first 9 tokens, well within the "step-0" regime of Appendix I.

First divergence bucket	Prompts
$t^* = 0$	10
$t^* \in [1, 4]$	21
$t^* \in [5, 9]$	55
$t^* \in [10, 29]$	14
$t^* \geq 30$	0
AGREED (t^* does not exist)	0

lm_head error that Intervention C targets, and the KV cache drifts irrecoverably within a few autoregressive steps.

(2) The intervention *is* firing. Latency for Intervention C on BF16 rises from the baseline's 9.20s to 9.67s (+5%), confirming that FP32 lm_head recomputation is being triggered (the per-step top-two margin does drop below $\tau = 10^{-3}$ on this model, consistent with the small-margin phenomenology in Section 7.5). What fails is not the trigger, but the *correctability* of the resulting flip: by the time a low-margin step arrives, the autoregressive path has already been driven onto a precision-specific KV cache. Appendix II's finding that KV-cache swaps fix only 36–48% of flips *even on TinyLlama* is amplified here, because on this 7B model all 100 prompts sit on the cache-locked branch.

(3) This fits the paper's first-token screening narrative. model-sensitivity analysis already flags Qwen2.5-7B-Instruct and the Qwen-1.5B family as "catastrophic first-token divergence" models, and our 7B replication extends that list to DeepSeek-R1-Distill-Qwen-7B. These models share a common training-time property: heavy BF16 saturation during pretraining produces activations whose FP16 cast yields numerically unstable (and occasionally NaN) intermediate states, and the corresponding argmax trajectory bifurcates within a handful of decoding steps (Ghaffari et al., 2024; Bondarenko et al., 2024). For this class of models, the precision-invariance problem requires *training-time* fixes (precision-aware pretraining, format-matched fine-tuning, or explicit activation clipping), and inference-time interventions of any scope—including Intervention C, LayerCast, or ungated FP32 recomputation—are not the right tool. Our TinyLlama-1.1B and Qwen2.5-3B-Instruct results (Section 7.5, Appendix III), which do admit a non-empty AGREED group and do respond to Intervention C, characterise where the method applies; the 7B result here characterises where it does not.

Practical takeaway. Practitioners deploying a 7B-scale BF16-saturated model should not expect Intervention C (or any other inference-time lm_head-scoped fix) to reconcile BF16-vs-FP16 outputs. The reproducibility path for such models is to pin the serving precision and, if reproducibility across precisions is required, to either (i) use LayerCast's global FP32 upcast (at 2× memory) or (ii) retrain with precision-aware objectives. Our main-text method is designed for—and evaluated on—the substantial class of models where the divergence is a late, small-margin event at the lm_head, which is exactly what makes it a single-step-correctable event.

VI Quality preservation: does Intervention C change task accuracy?

Setup. Intervention C reconciles BF16-vs-FP16 outputs at the lm_head, but a natural concern is whether this comes at the cost of *task accuracy*: perhaps the intervention pushes both precisions toward the same *wrong* answer. We test this directly on GSM8K ($n = 100$) by running four configurations—BF16 baseline, FP16 baseline, BF16 + C, FP16 + C—and extracting the final numerical answer from each generation. We use three testbeds spanning the three family/scale tiers of Table 7: Qwen2.5-3B-Instruct (Tier 2, Qwen-family), Llama-3.2-3B-Instruct (Tier 1, largest EAR lift in the paper), and Mistral-7B-Instruct-v0.3 (Tier 2,

Table 10: Per-precision task accuracy on GSM8K ($n = 100$, final-answer extraction). McNemar exact paired tests compare Baseline vs. Intervention C within each (model, precision) cell. No test rejects the null at any conventional level. Despite the Llama-3.2-3B family showing the largest EAR lift in the paper (+36pp, Table 6), its task accuracy is unchanged under Intervention C; despite Mistral-7B being the only 7B non-Qwen model we tested, its Pass@1 point estimates drop by 3–5pp but with non-significant McNemar p values (0.125 and 0.250) and substantially overlapping Wilson CIs. The six within-model tests (two per model) span EAR lifts from +3pp to +36pp and provide no evidence that Intervention C systematically shifts either trajectory toward a wrong answer.

Model	Config	Correct	Acc (%)	95% CI	vs. Baseline
Qwen2.5-3B	BF16, Baseline	27/100	27.0	[19.3, 36.4]	—
	BF16, Intervention C	29/100	29.0	[21.0, 38.5]	ΔAcc= +2pp ($p = 0.688$)
	FP16, Baseline	28/100	28.0	[20.1, 37.5]	—
	FP16, Intervention C	27/100	27.0	[19.3, 36.4]	ΔAcc= −1pp ($p = 1.000$)
Llama-3.2-3B	BF16, Baseline	24/100	24.0	[16.7, 33.2]	—
	BF16, Intervention C	23/100	23.0	[15.8, 32.1]	ΔAcc= −1pp ($p = 1.000$)
	FP16, Baseline	20/100	20.0	[13.3, 28.9]	—
	FP16, Intervention C	21/100	21.0	[14.2, 30.0]	ΔAcc= +1pp ($p = 1.000$)
Mistral-7B	BF16, Baseline	43/100	43.0	[33.7, 52.8]	—
	BF16, Intervention C	38/100	38.0	[29.1, 47.8]	ΔAcc= −5pp ($p = 0.125$, +1/ − 6)
	FP16, Baseline	40/100	40.0	[30.9, 49.8]	—
	FP16, Intervention C	37/100	37.0	[28.2, 46.8]	ΔAcc= −3pp ($p = 0.250$, +0/ − 3)

non-Qwen 7B). All three models have GSM8K zero-shot accuracy well above the $\sim$3% noise floor we observe on TinyLlama,[2] giving matched-pair tests enough statistical power to detect small accuracy shifts.

Findings. (1) No statistically significant accuracy change in either direction, on any model. Under matched-pair McNemar tests, no (model, precision) cell rejects the null at any conventional level. For Qwen2.5-3B, BF16 goes $+4/-2$ ($p = 0.688$) and FP16 $+1/-2$ ($p = 1.000$). For Llama-3.2-3B, BF16 goes $+8/-9$ ($p = 1.000$) and FP16 $+2/-1$ ($p = 1.000$). For Mistral-7B, BF16 goes $+1/-6$ ($p = 0.125$) and FP16 $+0/-3$ ($p = 0.250$); both p-values sit above the 0.05 threshold but below 0.3, and the point-estimate drops (−5pp on BF16, −3pp on FP16) are worth a dedicated discussion (see finding (3) below). The paper's +22pp (TinyLlama), +36pp (Llama-3.2-3B), +3pp (Qwen2.5-3B), and +8pp (Mistral-7B) EAR gains are therefore not artefacts of Intervention C systematically pushing both precisions toward the same wrong answer.

(2) BF16 and FP16 baselines have essentially the same accuracy within each model. On Qwen2.5-3B, BF16 baseline Pass@1 is 27% and FP16 baseline is 28%; on Llama-3.2-3B, BF16 is 24% and FP16 is 20%; on Mistral-7B, BF16 is 43% and FP16 is 40%. The within-model gaps ($\leq$ 4pp) are well within paired-sample noise. Neither precision is the "right" answer on the prompts where they disagree; both are lossy approximations of the FP32 argmax trajectory. This directly supports the paper's framing (Section 4) that Intervention C *reconciles* two equally-valid low-margin argmax choices rather than correcting one path toward the other's answer.

(3) Effect-size patterns across tiers: large EAR lift with no churn bias (Llama-3.2-3B) and small EAR lift with mild point-estimate drift (Mistral-7B). The two non-trivial-lift tiers produce instructively different patterns. Llama-3.2-3B (Tier 1, +36pp EAR) exhibits large but balanced within-precision churn under C ($+8/-9$ on BF16: one sixth of prompts change correctness status, but the net effect

[2] All reported accuracies in this section use our bare-prompt format (`"Solve step by step:\n{question}\nAnswer:"`) with greedy decoding and no system prompt or chat template. This choice is deliberate: it is the same format used for the cross-family EAR comparison (Table 6) and keeps the setup invariant across TinyLlama, Qwen, Llama, and Mistral families, so the EAR numbers are directly comparable. Absolute Pass@1 under this format is a lower bound on what each model can achieve with its native chat template. Because the quality-preservation tests are matched-pair McNemar tests over the same prompts in all four arms, the conclusions about Intervention C's accuracy effect are unaffected by prompt-format choice; a chat-template replication confirms this empirically by repeating the Llama-3.2-3B rows with the native chat template, where absolute Pass@1 rises to 66% but the McNemar tests remain non-significant ($p = 0.625$ and $p = 1.000$).

Table 11: Prompt-format robustness: the same four-arm experiment on Llama-3.2-3B-Instruct using its native chat template with a short system prompt. Absolute Pass@1 more than doubles (24% → 66% on BF16 baseline), confirming that the lower numbers in the quality-preservation results reflect prompt format rather than model capability. Both McNemar paired tests remain non-significant, and fp16 shows zero Pass@1 flips under Intervention C (+0/ − 0): the intervention reconciles outputs exactly. The quality-preservation conclusion is therefore invariant to prompt format.

Config	Correct	Acc (%)	95% CI	vs. Baseline
BF16, Baseline	66/100	66.0	[56.3, 74.5]	—
BF16, Intervention C	64/100	64.0	[54.2, 72.7]	ΔAcc= −2pp (p=0.625, +1/ − 3)
FP16, Baseline	66/100	66.0	[56.3, 74.5]	—
FP16, Intervention C	66/100	66.0	[56.3, 74.5]	ΔAcc= 0pp (p=1.000, +0/ − 0)

is zero). Mistral-7B (Tier 2, +8pp EAR) exhibits asymmetric small churn (+1/−6 on BF16, +0/−3 on FP16) that does not reach significance but is entirely in one direction. Two interpretations are compatible with the paired-test data: (i) the Mistral-7B drops are noise—the McNemar null $p \geq 0.125$ in both precisions and a simultaneous sign-flip on both precisions at $n = 100$ is plausible under the null—or (ii) Intervention C slightly biases Mistral-7B's reconciled argmax toward the less-accurate of the two, a small effect only detectable with larger n. We cannot distinguish these at $n = 100$, so we report the observation honestly rather than smooth it over. In the deployment taxonomy of Appendix IV: for Tier 1 models, C is unambiguously a win; for Tier 2 models, C still provides measurable EAR gains and the Pass@1 cost is either zero or at worst a few percentage points with overlapping CIs, which most reproducibility-sensitive deployments will still accept. A full larger-n follow-up specifically on Mistral-7B is a natural next step.

(4) Small-model baseline for comparison. We also ran the same experiment on TinyLlama-1.1B (Pass@1 around 3–4% for all four arms, $n = 100$). Matched-pair McNemar tests are again non-significant ($p > 0.3$). However TinyLlama's accuracy sits at the noise floor for $n = 100$ and the tests have limited power; the Qwen2.5-3B, Llama-3.2-3B, and Mistral-7B rows of the quality-preservation results are the primary quality-preservation evidence.

(5) Implication for the main claim. the quality-preservation results closes this concern: Intervention C improves BF16-vs-FP16 output agreement (+22pp EAR on TinyLlama, Table 3; +36pp on Llama-3.2-3B, Table 6; replicated across $n = 300$, Appendix X) without passing a significance bar for accuracy change on any of the four tested models, across two EAR-lift tiers and three model families. The gain is a reconciliation of two valid argmax trajectories; its cost, if any, is small enough to sit within paired-test noise at $n = 100$.

Prompt-format robustness check (Llama-3.2-3B). The Llama-3.2-3B Pass@1 numbers in the quality-preservation results (20–24%) sit well below the model's advertised GSM8K accuracy because the bare-prompt format used throughout our cross-family experiments does not invoke the chat template against which Llama-3.2-3B-Instruct was tuned. To confirm that the quality-preservation conclusion does not depend on this choice, we repeat the four-arm experiment with Llama-3.2-3B's native chat template (using a short system prompt: "You are a careful math tutor. Solve the problem step by step. End your response with a line that contains the final numeric answer only."), keeping every other setting identical (same 100 GSM8K test items, greedy decoding, $\tau = 10^{-3}$, max_new_tokens = 512).

Two observations from the chat-template results are worth flagging. First, the FP16 arm produces *identical* per-prompt correctness under Baseline and Intervention C (+0/ − 0): on the 66 prompts where FP16 answered correctly under baseline it still does so under C, and symmetrically for the incorrect prompts. This is the strongest possible form of quality preservation—the intervention does not change *which* prompts are answered correctly, only *how* the underlying token trajectories are reconciled with the BF16 counterpart. Second, the BF16 arm shows a 2pp drop (−3/ + 1, McNemar $p = 0.625$), well within noise and with substantially overlapping Wilson CIs. Both arms are individually and jointly consistent with no accuracy change. Combined with the main quality-preservation table, we therefore have quality-preservation evidence on (Qwen2.5-3B, Llama-3.2-3B, and Mistral-7B with bare prompt, plus Llama-3.2-3B with chat

Table 12: Long-reasoning sanity check on MATH-500 ($n = 50$, max_new_tokens = 1024, bare-prompt format). Mean generated length is 430–610 tokens per prompt, 3–4× longer than the main-text GSM8K chains. Baseline EAR and Intervention C's lift both match the GSM8K numbers within binomial noise, indicating that the phenomenon neither amplifies nor decays with chain length at these scales.

Model	Variant	EAR (%)	95% CI	Avg. len	Trig/step
TinyLlama-1.1B-Chat	Baseline	44.0	$[31.2, 57.7]$	606	0.00
	Intervention C	64.0	$[50.1, 75.9]$	564	0.82
Llama-3.2-3B-Instruct	Baseline	30.0	$[19.1, 43.8]$	438	0.00
	Intervention C	56.0	$[42.3, 68.8]$	430	1.89

template), across four model / prompt-format conditions, none of which detect a significant accuracy effect from Intervention C.

VII Long-reasoning sanity check: does divergence grow with chain length?

Motivation. The main-text GSM8K experiments cap generation at 256 tokens, which matches the typical reasoning-chain length for GSM8K problems. Modern LLM workloads, however, routinely generate chains of 500–4000 tokens (competition math, agentic planning, long-context tool use). A natural question is whether the 30–64% cross-precision divergence we report is a short-chain artefact that would either amplify or vanish under longer generations. The a priori prediction was amplification: longer chains contain more low-margin steps, each one an opportunity for BF16 and FP16 to fork, and a single fork cascades. This appendix tests that prediction directly on MATH-500, a harder benchmark (Hendrycks-style competition math) at max_new_tokens = 1024.

Setup. We draw the first $n = 50$ MATH-500 test problems deterministically. Prompt format is bare (`"Problem:\n{problem}\n\nSolution:\n"`), consistent with every other cross-family experiment in the paper—no chat template, no system prompt. We run baseline and Intervention C on TinyLlama-1.1B-Chat and Llama-3.2-3B-Instruct (the two Tier 1 models from Table 7) in both BF16 and FP16, batch size 1, fixed seed, A10G 22 GB.

Findings. (1) Baseline divergence does not amplify with chain length. Compared to the GSM8K numbers in Table 6 (TinyLlama baseline 41%, Llama-3.2-3B baseline 31%), the MATH-500 baselines are 44% and 30% respectively—both within 3pp of their GSM8K counterparts despite the mean chain length tripling (170 tokens on GSM8K → 438–606 tokens on MATH-500) and the benchmark difficulty increasing substantially. This rules out the "long chain cascades more" hypothesis at the lengths we test.

(2) The saturation is consistent with the margin-based mechanism. Section 7.5 establishes that divergence is driven by the *density* of low-margin lm_head steps, not by their absolute count. MATH-500 has more low-margin steps per prompt than GSM8K (reflected in the higher trigger rate: 1.89 vs ~ 1 on Llama-3.2-3B GSM8K), but per-step divergence rate roughly holds, so the fraction of prompts with *zero* surviving low-margin forks scales as a power of per-step divergence rate and saturates quickly. This is the same argument used in Section 7.5: the first diverged step cascades, so longer chains cannot rescue an already-diverged prompt but also cannot spoil an already-converged one once the trajectories have synchronised past the last low-margin event.

(3) Intervention C's lift persists at longer chains. The lifts are +20pp (TinyLlama) and +26pp (Llama-3.2-3B) on MATH-500, to compare against +22pp (TinyLlama, Table 3) and +36pp (Llama-3.2-3B, Table 6) on GSM8K. Both MATH-500 lifts sit inside the 95% Wilson CI of their GSM8K counterparts and are comfortably positive on their own CIs. The slight Llama-3.2-3B attenuation (+36pp → +26pp) is explained by the higher per-step trigger density on harder problems: more low-margin steps means a higher fraction require recomputation, but the first-cascade argument still applies to the bulk of prompts.

(4) An incidental latency observation. On MATH-500, Llama-3.2-3B's FP16 baseline wall-clock time (10.93s/prompt) is about 24% *faster* than its BF16 counterpart (14.46s/prompt). This is the first latency observation of this magnitude in our experiments and is not relevant to the intervention claim, but we record it here for transparency. The most plausible explanation is SM 86 (A10G) having a somewhat faster FP16 FMA path than BF16 FMA, plausibly amplified by the longer generation lengths on MATH-500; we did not pursue this further.

Summary. The phenomenon and the remedy both transfer to 3–4× longer chains at no detectable cost to either EAR lift or Wilson CI width (the latter shrinks slightly due to the lower $n = 50$, balanced by the tighter per-prompt consistency). Cross-precision divergence is therefore a property of per-step margin density, not of absolute reasoning length, over the range we explore.

VIII Production-stack robustness

Table 13: Production-stack robustness checks (TinyLlama-1.1B, GSM8K).

Axis	Result	Implication
MoE (OLMoE)	100% BF16 self-consistency	Kernel non-det. ruled out
FlashAttention-2	EAR −1pp; C lift +18pp (= eager)	Kernel-independent
Batch size	C lift: +17/+7/+7/0pp (bs 1/2/4/8)	Bounded to bs≤4
LayerCast + C	+5pp at bs=8; noise at bs=16	Composition viable
Overhead	+1.4% latency, +11% memory	Negligible cost

IX Hyperparameter and intervention ablations

Table 14: Ablation summary.

Ablation	Result	Confirms
Scope: RMSNorm+lm_head	EAR drops 63%→44%	lm_head-only optimal
Threshold $\tau \in [10^{-4}, 10^{-2}]$	EAR insensitive	Bimodal margin
Top-K: $K \in \{2, ..., \lvert V \rvert\}$	All identical EAR	Rank-1-vs-2 only
Temperature sharpening	Zero effect	Scale-invariant ratio
Consensus decoding	100% at 2× cost	Trivial upper bound

X Statistical validation and reproducibility

Wilson 95% CIs have half-width ±9–10pp at n=100; all headline lifts exceed CI bounds. 10,000-sample bootstrap confirms: TinyLlama +22pp has 90% CI $[+11, +33]$pp. Scaled replication at n=300 (GSM8K), n=164 (full HumanEval), and n=257 (full MBPP) confirms all n=100 point estimates fall inside the scaled CIs.

XI Reproducibility statement

Environment and methodology. All experiments run on a single NVIDIA A10G (22 GB, Ampere SM 86) under PyTorch 1.13 / 2.1 with CUDA 11.7, using deterministic flags (`torch.use_deterministic_algorithms(True)`, fixed CUBLAS workspace). Models are loaded from HuggingFace Hub with no modification. Each reported number uses greedy decoding (T=0) with a fixed random seed for prompt selection. The primary evaluation metric (EAR) is computed by byte-level comparison of full token-id sequences; SAR uses regex extraction of final numerical answers. Confidence intervals are Wilson score intervals at n=100; bootstrap CIs use 10,000 resamples. All experiments are

reproducible given the model weights, prompt templates described in Section 5, and the hyperparameters reported in this paper (τ=10^{-3}, K=8 for Intervention B).