# SHARPDRAFT: ACCELERATING LONG-CONTEXT SPECULATIVE DECODING WITH CARDINALITY-AWARE QUERY SCALING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Long-form reasoning makes inference expensive, and speculative decoding mitigates this cost by verifying multiple draft tokens in parallel. Its speedup, however, can fade as context grows and draft acceptance declines. We focus on attention-mass dilution: as softmax normalizes over more visible Keys, the mass concentrated on the highest-scoring Keys can decrease. We introduce SHARPDRAFT, a training-free method that counteracts this effect through cardinality-aware Query scaling, without the computational overhead of online adaptation. Under explicit assumptions, we derive an exact top-$k$ mass correction and deploy a closed-form fixed-slope approximation. Across AIME-26, GPQA-Diamond, and LongGenBench Diary, SHARPDRAFT achieves $2.59$–$3.19\times$ geometric-mean end-to-end speedups over target-only autoregressive decoding when applied to DFlash, PARD, and EAGLE 3.1. With DFlash, it improves decoding speed and outperforms full-parameter and LoRA-based online adaptation in end-to-end speedup, while matching the unmodified drafter's reported peak allocated GPU memory.

## 1 INTRODUCTION

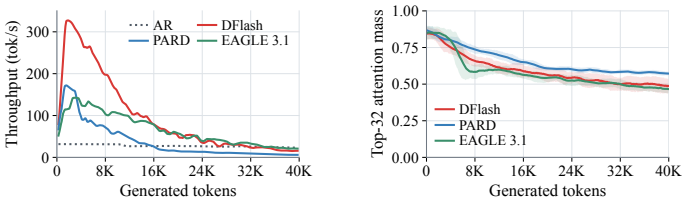
Recent advances in Large Language Models (LLMs) have expanded their use from short question answering to long-form writing, long-horizon reasoning, agentic workflows, and multi-turn dialogue (Bai et al., 2025; Ling et al., 2025; Motwani et al., 2026; Fang et al., 2026; Bai et al., 2024). These workloads either produce long outputs within a single request or repeatedly append new interactions to an already growing history. Autoregressive decoding remains slow in this setting because it generates only one token per sequential decoding step. Speculative decoding reduces the number of sequential target-model passes by using a lightweight draft model to propose several tokens for parallel verification (Leviathan et al., 2023; Chen et al., 2023). A successful verification can accept multiple draft tokens at once rather than producing a single token as in autoregressive decoding. The resulting speedup depends in part on the mean accepted length (MAL), reported here as the number of output tokens emitted per target verification.

The benefit of speculative decoding does not remain constant throughout a long generation. MAL can decrease as generation proceeds. This increases the number of target-model verification steps required per generated token. DFlash's Qwen3-8B draft model is trained with a maximum sequence length of 3,072 tokens (Chen et al., 2026). Figure 1(a) illustrates this behavior on LongGenBench Diary: DFlash, PARD, and EAGLE 3.1 achieve high throughput relative to AR early in generation, but their advantage declines as generation continues. Throughout this paper, EAGLE 3.1 refers to the post-norm EAGLE-3 configuration incorporating the normalization changes introduced in Attention Drift (Eldenk et al., 2026) and described in the EAGLE 3.1 release (EAGLE Team et al., 2026).

OSD and OnlineSpec use target feedback to adapt the draft model during inference and improve its agreement with the target (Liu et al., 2024; Qian et al., 2026). We ask whether growing context is also accompanied by a systematic change in the draft model's attention. Softmax attention can become more diffuse as the number of visible Keys grows (Veličković et al., 2025). Figure 1(b) shows a corresponding decline in the average attention mass assigned to the 32 highest-scoring Keys across these draft models. We call this change *attention-mass dilution* and investigate whether it contributes to MAL degradation and whether directly correcting it can preserve draft acceptance.



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




(a) Decode throughput (b) Top-32 attention mass

Figure 1: Long-generation behavior of Qwen3-8B on LongGenBench Diary. (a) Decode throughput of different methods in tokens per second. (b) Top-32 attention mass of different methods.

In this paper, we introduce SHARPDRAFT, a training-free method that counteracts attention-mass dilution through cardinality-aware Query scaling. At each draft step, SHARPDRAFT scales each draft Query according to its visible-Key cardinality without changing the draft-model parameters. Under explicit assumptions on the attention-score partitions, we derive an exact correction for top-$k$ attention mass. SHARPDRAFT uses a closed-form fixed-slope approximation that requires only a scalar computation and Query rescaling. The cached Keys and Values remain unchanged.

We evaluate SHARPDRAFT with DFlash, PARD, and EAGLE 3.1 on long-generation workloads. Across AIME-26, GPQA-Diamond, and LongGenBench Diary, these systems achieve geometric-mean end-to-end speedups of $2.59$–$3.19\times$ over target-only AR. In the DFlash evaluation, SHARPDRAFT improves on the unmodified drafter while matching its reported peak allocated GPU memory, and outperforms full-parameter and LoRA-based online adaptation in end-to-end speedup.

## 2 RELATED WORK

### 2.1 SPECULATIVE DECODING AND ONLINE DRAFT ADAPTATION

Speculative decoding reduces the latency of autoregressive LLM inference (Leviathan et al., 2023; Chen et al., 2023). A lightweight draft model proposes multiple tokens, which the target model verifies in parallel. The resulting speedup depends in part on how many draft tokens are accepted. Recent advances, including EAGLE-3 (Li et al., 2025) and DFlash (Chen et al., 2026), improve draft quality by using target-model features to guide token prediction. However, draft models trained on short sequences may not generalize well to longer contexts. LongSpec addresses this mismatch in part by introducing novel position indices for short-sequence draft training (Yang et al., 2026). Test-Time Speculation (TTS) reports that acceptance in EAGLE-3 and DFlash declines as generation proceeds, reducing their decoding speedup (Kumar et al., 2026).

Online draft adaptation addresses inference-time mismatch by learning from target feedback collected during serving. Online Speculative Decoding (OSD) continually distills draft models on observed queries to adapt to the serving distribution (Liu et al., 2024). OnlineSpec formulates draft–target interaction as online learning and develops algorithms for updating draft models from verification feedback (Qian et al., 2026). TTS adapts the draft model during generation using target supervision from successive verification rounds (Kumar et al., 2026). It uses strided updates and asynchronous execution to reduce the impact of training on decoding latency. Our work also addresses declining draft acceptance during long generations, but focuses on the dilution of attention mass as the number of attended Keys grows. Rather than updating the draft model weights, SHARPDRAFT mitigates this effect by scaling Queries according to the current Key count.

### 2.2 ATTENTION NORMALIZATION AND LENGTH GENERALIZATION

Attention normalization can affect how models generalize to longer contexts. LM-Infinite shows that increasing the number of attended tokens can push attention entropy beyond its training range when attention logits remain bounded (Han et al., 2024). Veličković et al. (2025) establish conditions under which softmax attention disperses as the number of attended tokens grows. They propose adjusting the inference-time temperature based on attention entropy to mitigate this effect. Mudarisov et al. (2025) analyze how softmax scaling and the geometry of token representations constrain selec-

tive attention. Scalable-Softmax mitigates attention dilution by scaling Queries logarithmically with the number of attended tokens (Nakanishi, 2025). It considers both learning the scaling parameters during pretraining and replacing softmax after pretraining.

Unlike these studies, our work addresses declining draft acceptance in speculative decoding during long generations. To that end, we analyze attention normalization to distinguish the effect of Key count from changes in the score contrast between the top-$k$ Keys and the remaining Keys. This analysis yields a Query scaling rule that preserves a reference top-$k$ attention mass under stated assumptions. SHARPDRAFT uses a lightweight approximation of this rule, which we evaluate in terms of draft acceptance and end-to-end decoding latency.

## 3 CARDINALITY-AWARE QUERY SCALING

SHARPDRAFT scales draft queries according to the number of visible keys. Our primary evaluations use a closed-form fixed-slope approximation, derived below by separating the effect of key count from attention-score contrast.

### 3.1 MOTIVATION: ATTENTION DILUTION UNDER GROWING KEY COUNTS

Full-context drafting exposes queries to more keys as generation proceeds; Figure 1 shows declining top-32 attention mass for Qwen3-8B. Adding lower-scoring keys enlarges the softmax denominator and reduces an existing top-$k$ group's mass even when all existing scores stay fixed.

For one attention head of dimension $d$, fix a query $q_t$ and keys $k_{t,i}$ after normalization and positional transformation. The unscaled scores and attention weights at query scale $\beta > 0$ are

$$z_{t,i} = \frac{q_t^\top k_{t,i}}{\sqrt{d}}, \qquad a_{t,i}(\beta) = \frac{e^{\beta z_{t,i}}}{\sum_{j=1}^{N_t} e^{\beta z_{t,j}}}, \tag{1}$$

where $\beta = 1$ is the checkpoint scale. For a fixed positive integer $k$ and $N_t > k$ keys visible under the actual mask, let $S_t$ be the top-$k$ score set, $B_t$ its complement, and $C_t = N_t - k = |B_t|$. To separate score contrast from group size, define

$$F_t(\beta) = \log\left(\frac{1}{k}\sum_{i\in S_t} e^{\beta z_{t,i}}\right) - \log\left(\frac{1}{C_t}\sum_{j\in B_t} e^{\beta z_{t,j}}\right), \tag{2}$$

The top-$k$ mass $p_t(\beta) = \sum_{i\in S_t} a_{t,i}(\beta)$ then obeys the exact identity

$$\operatorname{logit} p_t(\beta) = F_t(\beta) - \log\frac{C_t}{k}. \tag{3}$$

Here $F_t$ measures the score contrast normalized by each partition's size, while $\log(C_t/k)$ gives the explicit competitor-count term. At fixed contrast and $k$, doubling $C_t$ halves the attention odds and reduces top-$k$ mass.

During generation, content, positions, and top-$k$ membership also change, so this identity does not isolate cardinality as the cause of Figure 1's trend. Top-$k$ membership alone does not establish task relevance. For fixed scores, positive scaling preserves their ordering and, under strict top-$k$ separation, increases $F_t$ (Appendix A), motivating a correction for the count term.

### 3.2 CARDINALITY-AWARE QUERY SCALING

Choose a fixed reference count $N_0 > k$ with $C_{\mathrm{ref}} = N_0 - k$. For the derivation and numerical-inverse variant, define the reference contrast $F_{\mathrm{ref}}(\beta) = \mathbb{E}_{x\in\mathcal{C}_{\mathrm{ref}}}[F_x(\beta)]$, where $\mathcal{C}_{\mathrm{ref}}$ is a set of reference queries disjoint from evaluation. The reference mass is defined by $\operatorname{logit}\rho_{\mathrm{ref}} = F_{\mathrm{ref}}(1) - \log(C_{\mathrm{ref}}/k)$. For $C_t \geq C_{\mathrm{ref}}$ and inactive upper clipping, reference matching requires

$$F_{\mathrm{ref}}(\beta_t) - F_{\mathrm{ref}}(1) = \log\frac{C_t}{C_{\mathrm{ref}}}. \tag{4}$$

Thus, increased reference contrast offsets the log-odds loss from more competitors.

A first-order Taylor expansion around $\beta = 1$ gives $F_{\text{ref}}(\beta) \approx F_{\text{ref}}(1) + \Delta_{\text{ref}}(\beta - 1)$, with $\Delta_{\text{ref}} = F'_{\text{ref}}(1) > 0$. Our deployed fixed-slope rule replaces this derivative by a fixed $s > 0$ and caps the scale at $\beta_{\max} \geq 1$:

$$r_t = \max\left\{\frac{N_t - k}{N_0 - k}, 1\right\}, \qquad \widehat{\beta}_t = \min\left\{\beta_{\max}, 1 + \frac{\log r_t}{s}\right\}. \tag{5}$$

We apply $q_t \leftarrow \widehat{\beta}_t q_t$ using $s = 5$, $k = 32$, and $\beta_{\max} = e^{0.5}$, without constructing an empirical reference curve. The slope is a fixed setting rather than a measured derivative. Likewise, $N_0$ is a fixed experimental proxy; the values and their interpretation are given in Appendix B.1. Below the reference count, $\widehat{\beta}_t = 1$.

For $C_t \geq C_{\text{ref}}$, the reference inverse preserves $\rho_{\text{ref}}$ when current and reference contrasts match throughout the application range and upper clipping is inactive (Appendix A.3). For the fixed-slope rule, mass error also depends on the slope choice and the Taylor approximation error (Appendix A.4). This attention-mass analysis does not guarantee draft acceptance or latency.

### 3.3 Inference Integration

We integrate SharpDraft into DFlash (Chen et al., 2026) by rescaling queries after Q-normalization, sharing the scale across all five draft layers in our primary evaluation. The scaling affects query scores against both context and current-block keys. The target model and verification procedure are unchanged, as is the construction of persistent context keys and values from target-model features. Scale changes therefore require no rebuilding of cached context keys and values. Appendix B.1 specifies the BF16/FP32 arithmetic.

At each drafting step, $N_t$ includes visible persistent-context and current-draft-block keys, excluding masked and padded positions. Our full-attention implementation uses persistent-context length plus draft-block size and retains the count across turns when context is retained. With a fixed sliding window, the count and scale saturate. The parameter $k$ defines the analytical partition and does not restrict which keys receive attention. The deployed rule requires no online top-$k$ selection or reference-curve evaluation: it computes a scalar scale from the count and rescales the queries. Both operations are included in E2E latency, with no additional model forward pass, backward pass, or optimizer state.

## 4 Experiments

We evaluate how SharpDraft affects decoding latency, memory use, and draft acceptance across models and generation lengths.

**Evaluation setup.** Our primary experiments pair Qwen3-4B, Qwen3-8B, and Llama-3.1-8B with their corresponding DFlash draft models (Chen et al., 2026). We compare the unmodified draft model (Frozen), Full Adaptation, LoRA, and SharpDraft, using target-only autoregressive decoding (AR) as the latency reference. Full Adaptation updates all parameters of the draft model using target feedback during generation. LoRA (Hu et al., 2022) updates only the low-rank adapter parameters, with rank 16 unless otherwise specified. Both adaptation baselines use a forward-KL distillation objective to align the draft model's token distributions with those of the target (Liu et al., 2024; Zhou et al., 2024). They use target distributions obtained during verification and schedule periodic updates during generation, a setting similar to that of Test-Time Speculation (TTS) (Kumar et al., 2026). SharpDraft applies Equation 5 without updating model parameters. We measure end-to-end (E2E) speedup, peak allocated memory, online update cost, and mean accepted length (MAL), defined as emitted tokens per target verification. Hardware, execution backends, and measurement protocols are specified in Appendix B.

### 4.1 End-to-End Efficiency

To measure the net benefit of calibration and online adaptation, we compare total generation time, including their inference-time costs. Table 1 covers 30 AIME-26 questions from MathArena (Dekoninck et al., 2026), all 198 GPQA-Diamond questions (Rein et al., 2024), and 100 Daily Diary

prompts from LongGenBench's long split (Wu et al., 2025), abbreviated as LGB(Diary). Speedup is total AR time divided by total method time on the same prompts. Overall is the geometric mean across benchmarks.

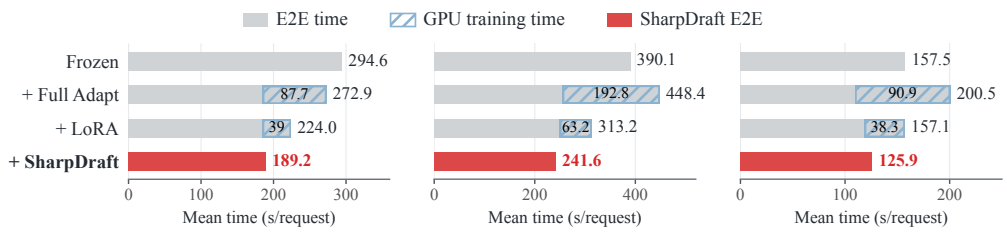
Table 1: End-to-end speedup and adaptation overhead on long-generation workloads. Speedups are relative to target-only AR; Overall is the geometric mean across the three benchmarks. Peak memory and update cost are measured on AIME-26 and GPQA-Diamond only.

| Model | Method | Trainable Params ↓ | Peak Mem. ↓ | Online Update Cost (ms/update) ↓ | Speedup ↑ | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | AIME-26 | GPQA-D | LGB(Diary) | Overall |
| Qwen3-4B | Target-only AR | 0 | 12.54 GiB | 0 | 1.000× | 1.000× | 1.000× | 1.000× |
| | DFlash (Frozen) | 0 | 16.68 GiB | 0 | 1.267× | 1.598× | 1.637× | 1.491× |
| | + Full Adaptation | 537.43M | 24.46 GiB | 103.14 | 1.368× | 1.981× | 2.478× | 1.886× |
| | + LoRA ($r = 32$) | 9.67M | 19.95 GiB | 51.26 | 1.434× | 1.952× | 2.568× | 1.930× |
| | + LoRA ($r = 16$) | 4.83M | 19.84 GiB | 49.91 | 1.589× | 2.183× | 2.865× | 2.150× |
| | + LoRA ($r = 8$) | 2.42M | 19.80 GiB | 50.13 | 1.667× | 2.267× | 3.050× | 2.259× |
| | + SHARPDRAFT | 0 | 16.68 GiB | 0 | **1.973×** | **2.666×** | **3.316×** | **2.593×** |
| Qwen3-8B | Target-only AR | 0 | 20.26 GiB | 0 | 1.000× | 1.000× | 1.000× | 1.000× |
| | DFlash (Frozen) | 0 | 26.87 GiB | 0 | 1.319× | 1.392× | 1.798× | 1.489× |
| | + Full Adaptation | 1.049B | 41.29 GiB | 199.21 | 1.148× | 1.317× | 2.112× | 1.472× |
| | + LoRA ($r = 32$) | 12.91M | 31.95 GiB | 61.65 | 1.532× | 1.815× | 2.993× | 2.027× |
| | + LoRA ($r = 16$) | 6.46M | 31.80 GiB | 61.76 | 1.613× | 1.887× | 3.125× | 2.119× |
| | + LoRA ($r = 8$) | 3.23M | 31.63 GiB | 61.76 | 1.643× | 1.884× | 3.184× | 2.144× |
| | + SHARPDRAFT | 0 | 26.87 GiB | 0 | **2.130×** | **2.316×** | **3.882×** | **2.675×** |
| Llama-3.1-8B | Target-only AR | 0 | 19.16 GiB | 0 | 1.000× | 1.000× | 1.000× | 1.000× |
| | DFlash (Frozen) | 0 | 25.62 GiB | 0 | 2.170× | 2.045× | 3.183× | 2.417× |
| | + Full Adaptation | 1.049B | 40.15 GiB | 246.19 | 1.705× | 1.596× | 2.211× | 1.819× |
| | + LoRA ($r = 32$) | 12.91M | 30.61 GiB | 102.06 | 2.044× | 1.998× | 2.675× | 2.219× |
| | + LoRA ($r = 16$) | 6.46M | 30.59 GiB | 101.62 | 2.147× | 2.067× | 2.809× | 2.319× |
| | + LoRA ($r = 8$) | 3.23M | 30.22 GiB | 101.26 | 2.175× | 2.100× | 2.854× | 2.353× |
| | + SHARPDRAFT | 0 | 25.62 GiB | 0 | **2.714×** | **2.593×** | **3.710×** | **2.967×** |

SHARPDRAFT achieves the highest E2E speedup on all nine model–benchmark pairs in Table 1. Overall speedups are 2.593×, 2.675×, and 2.967× for Qwen3-4B, Qwen3-8B, and Llama-3.1-8B, respectively. Full Adaptation improves over Frozen on Qwen3-4B. For Qwen3-8B and Llama-3.1-8B, however, it takes longer on both AIME-26 and GPQA-Diamond. All tested LoRA ranks trail SHARPDRAFT on every model–benchmark pair.

Query scaling also avoids the memory needed to train the draft model. SHARPDRAFT matches Frozen's reported peak memory, whereas Full Adaptation adds 7.78–14.53 GiB across the three models. LoRA reduces this overhead but still incurs additional memory and update costs. Query scaling is included in the E2E latency and peak-memory measurements. SHARPDRAFT requires no optimizer updates.

To examine the training work incurred within a request, Figure 2 overlays recorded GPU training durations on mean AIME-26 E2E times, using rank 8 for LoRA. Full Adaptation spends substantially more GPU time on training than LoRA, while SHARPDRAFT completes requests fastest on all three models without optimizer updates. Training can overlap decoding, so the hatched durations measure training work rather than a separate, additive portion of request latency (Appendix B.2).




(a) Qwen3-4B (b) Qwen3-8B (c) Llama-3.1-8B

Figure 2: Mean end-to-end latency and recorded GPU training time per AIME-26 request.

### 4.2 Generalization Across Draft Model Families

To test whether the correction is useful beyond DFlash, we use Qwen3-8B and apply Query scaling to parallel drafting with PARD (An et al., 2026) and autoregressive drafting with EAGLE 3.1 (EAGLE Team et al., 2026; Eldenk et al., 2026). The draft model weights, proposal configuration, and verification procedure remain unchanged within each family. Table 2 reports MAL and AR-relative speedup. Reference and clipping settings for each draft model are given in Appendix B.3.

For EAGLE 3.1, SharpDraft raises MAL from 3.152 to 3.302 on AIME-26, 3.659 to 3.904 on GPQA-D, and 5.668 to 5.968 on Diary. Whether the MAL gains reduce latency also depends on drafting and verification costs.

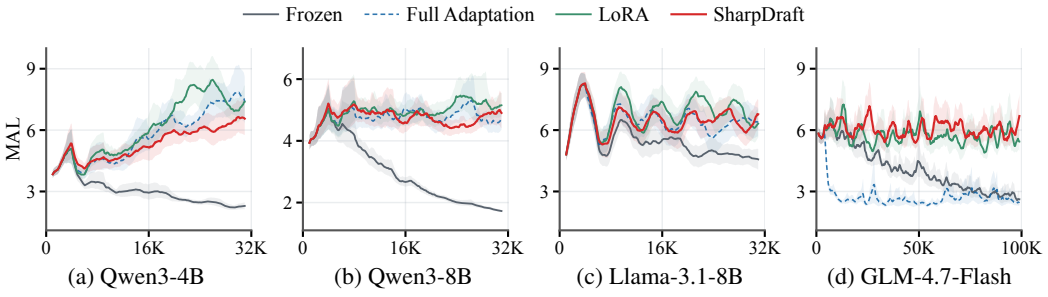
Table 2: MAL and end-to-end speedup across draft model families with Qwen3-8B. Speedups are relative to target-only AR; Overall is their geometric mean across the three benchmarks.

| Method | MAL ↑ | | | Speedup ↑ | | | |
|---|---|---|---|---|---|---|---|
| | AIME-26 | GPQA-D | LGB(Diary) | AIME-26 | GPQA-D | LGB(Diary) | Overall |
| Target-only AR | – | – | – | 1.000× | 1.000× | 1.000× | 1.000× |
| PARD | 2.368 | 2.728 | 2.252 | 2.691× | 3.389× | 2.729× | 2.919× |
| + LoRA | 1.637 | 2.224 | 1.390 | 1.015× | 1.567× | 0.923× | 1.137× |
| + SharpDraft | **2.441** | **2.906** | **2.424** | **2.773×** | **3.939×** | **2.975×** | **3.191×** |
| EAGLE 3.1 | 3.152 | 3.659 | 5.668 | 1.922× | 2.529× | 3.515× | 2.575× |
| + LoRA | 3.086 | 3.812 | 4.251 | 1.725× | 2.168× | 2.549× | 2.120× |
| + SharpDraft | **3.302** | **3.904** | **5.968** | **2.045×** | **2.624×** | **3.792×** | **2.730×** |

### 4.3 Draft Acceptance in Multi-Turn Generation

To evaluate acceptance as conversation history grows, we extend the multi-turn setting used in MT-Bench (Zheng et al., 2023) and speculative-decoding benchmarks (Li et al., 2024; Xia et al., 2024) with MATH-500 level-4/5 problems (Lightman et al., 2024). Each turn introduces a new problem after EOS or 5,000 generated tokens, retaining conversation history, KV caches, and adaptation state. Figure 3 averages thirty sessions with matched question orders and covers 32K generated tokens for the three Table 1 models. To examine longer histories, we additionally evaluate GLM-4.7-Flash with a cumulative generation budget of 100K tokens, using its larger context limit.

SharpDraft maintains higher late-position MAL than Frozen across all four models. Full Adaptation and LoRA can achieve higher MAL, particularly for Qwen3-4B, at the cost of online training. Repetitive continuations are retained, so these curves measure drafting efficiency rather than reasoning quality (Appendix B.5).




(a) Qwen3-4B (b) Qwen3-8B (c) Llama-3.1-8B (d) GLM-4.7-Flash

Figure 3: Draft acceptance during multi-turn MATH-500 generation, up to 32K tokens in (a)–(c) and 100K in (d). Lines show mean session MAL; bands indicate pointwise 95% bootstrap confidence intervals.

# 5 Analysis and Ablation Studies

## 5.1 Dynamic versus Fixed Scaling

To isolate the benefit of context-dependent scaling, we compare SharpDraft with fixed Query scales on Qwen3-8B/DFlash. The comparison includes Frozen ($\beta = 1$) and five constants, $\beta \in \{e^{0.1}, e^{0.2}, e^{0.3}, e^{0.4}, e^{0.5}\}$. All seven settings produce identical output tokens and termination on the 30 AIME-26 problems.

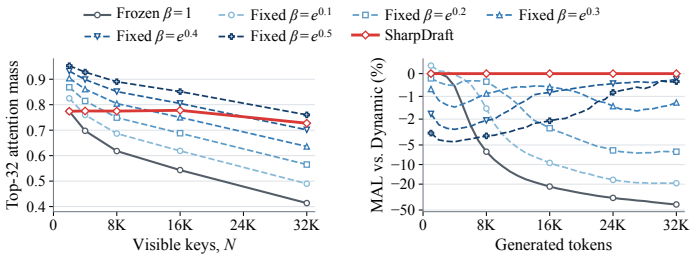
Table 3 reports whole-request MAL, weighting problems equally and grouping them by final output length. The best fixed scale is the one with the highest mean MAL among the five tested values, selected separately overall and within each length group (Appendix C.1).

Table 3: Fixed versus dynamic Query scaling on Qwen3-8B/DFlash using 30 output-matched AIME-26 problems. Values are mean whole-request MAL, grouped by final output length. Parentheses show gains over Frozen.

| Method | Final output length | | | All problems |
|---|---|---|---|---|
| | 0–10K | 10–20K | 20–32K | Overall MAL ↑ |
| Frozen ($\beta = 1$) | 4.092 | 3.113 | 2.596 | 3.182 |
| Fixed $\beta = e^{0.1}$ | 4.273 (+0.181) | 3.524 (+0.411) | 3.318 (+0.723) | 3.660 (+0.477) |
| Fixed $\beta = e^{0.2}$ | 4.305 (+0.214) | 3.726 (+0.613) | 3.785 (+1.189) | 3.925 (+0.743) |
| Fixed $\beta = e^{0.3}$ | 4.284 (+0.193) | 3.784 (+0.671) | 3.989 (+1.393) | 4.023 (+0.841) |
| Fixed $\beta = e^{0.4}$ | 4.235 (+0.143) | 3.756 (+0.643) | 4.028 (+1.433) | 4.018 (+0.835) |
| Fixed $\beta = e^{0.5}$ | 4.166 (+0.074) | 3.702 (+0.588) | 4.005 (+1.410) | 3.973 (+0.790) |
| SharpDraft (dynamic) | **4.313** (+0.221) | **3.808** (+0.694) | **4.047** (+1.451) | **4.063** (+0.880) |

Fixed scaling captures most of the gain over Frozen. The best fixed scale ($\beta = e^{0.3}$) reaches a mean MAL of 4.023, compared with 3.182 for Frozen and 4.063 for SharpDraft. The additional gain from dynamic scaling is modest (0.99%), but it improves request MAL on 26 of 30 problems and has the highest mean in each length group. The preferred fixed scale also changes with output length, from $e^{0.2}$ for short outputs to $e^{0.4}$ for long outputs. A single fixed setting must therefore compromise between these length regimes.

To see whether this trade-off also occurs within a generation, we track attention concentration and cumulative acceptance on the same cohort. Fixed scales raise top-32 attention mass on matched prefixes, but the mass still declines as context grows. SharpDraft stays near its short-context level through 16K before declining at 32K (Figure 4a). The live acceptance profiles show a corresponding shift toward stronger fixed scales as generation proceeds, whereas SharpDraft has the highest mean cumulative MAL from 4K onward (Figure 4b). Later positions average only the problems that reach them, as detailed in Appendices C.2 and C.3.




(a) Attention concentration (b) MAL relative to Dynamic

Figure 4: Fixed and dynamic scaling during AIME-26 generation with Qwen3-8B. (a) Top-32 attention mass on matched prefixes. (b) Mean cumulative MAL relative to SharpDraft.

### 5.2 FIXED-SLOPE APPROXIMATION

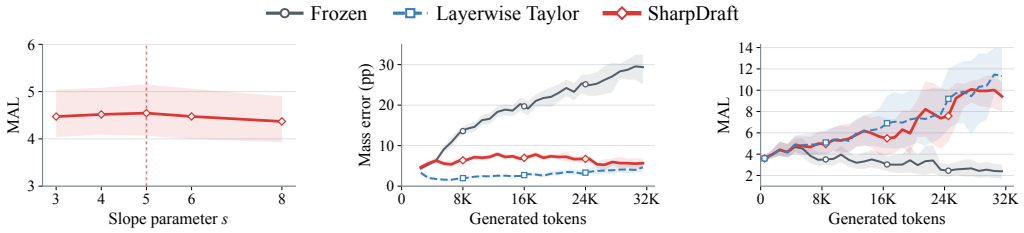
We examine whether more accurate attention-mass restoration improves acceptance enough to justify the cost of state-dependent calibration. Although SHARPDRAFT fixes the shared slope at $s = 5$, its Query scale still varies with visible-Key count through Equation 5. The approximation avoids measuring current attention statistics and computes the scale from cardinality alone.

For a more accurate correction, Layerwise Taylor evaluates the current contrast and its derivative separately in each draft layer, re-centering the Taylor expansion near the previous applied scale. This requires additional Query–Key score computation and top-32 partitioning in all five layers at each post-reference drafting step. We compare this variant with Frozen and SHARPDRAFT on AIME-26 using Qwen3-8B/DFlash with native vLLM. Appendix C.5 gives the update rule and measurement details.

Layerwise Taylor restores attention mass more accurately and yields a modest MAL gain. Relative to SHARPDRAFT, Layerwise Taylor reduces mean absolute top-32 mass error from $6.163$ to $2.388$ percentage points and raises whole-request MAL from $4.545$ to $4.667$ ($2.7\%$), averaging requests equally (Figures 5b and 5c). Most of the additional MAL gain comes from capped or repetitive outputs. These results are consistent with a benefit from mitigating dilution, but do not isolate its causal effect because layerwise correction, the scale cap (2 rather than $e^{0.5}$), and generation paths also differ.

Mean per-request E2E time is $89.70$ seconds for Frozen, $59.24$ for SHARPDRAFT, and $77.90$ for Layerwise Taylor, with Taylor taking $31.5\%$ longer than SHARPDRAFT. Because output lengths and GPU assignments differ in this full-cohort comparison, we also assess runtime on requests with identical output tokens and termination for SHARPDRAFT and Layerwise Taylor, measuring each comparison on the same GPU. On these matched requests, Layerwise Taylor increases mean E2E time from $25.87$ to $31.28$ seconds ($20.9\%$). These timings include calibration during generation but exclude separate attention-mass diagnostics. For this native-vLLM implementation, the extra calibration cost outweighs the acceptance gain, favoring the simpler fixed-slope rule.

We separately test the choice of $s = 5$ by sweeping $s \in \{3, 4, 5, 6, 8\}$ on AIME-26, holding $N_0$, $k$, and $\beta_{\max}$ fixed. The default $s = 5$ gives the highest observed whole-request MAL ($4.545$), while $s = 4$ and $s = 6$ remain within $1.6\%$ (Figure 5a). A separate pooled-reference diagnostic yields effective slopes between $4.995$ and $5.112$ (Appendix C.4). Together, these results support $s = 5$ as a practical default that retains cardinality-dependent scaling without the cost of online contrast estimation.

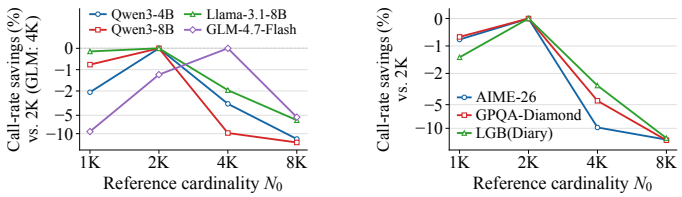



(a) Slope sensitivity (b) Top-32 mass error (c) MAL

Figure 5: Fixed-slope sensitivity and layerwise calibration on Qwen3-8B/AIME-26. (a) MAL across slope settings. (b,c) Top-32 mass error and MAL. Shading shows pointwise 95% bootstrap confidence intervals.

### 5.3 SENSITIVITY TO THE REFERENCE CARDINALITY

We vary the reference count $N_0$ to check whether the same setting remains useful across models and datasets. Scaling begins when the visible-Key count exceeds $N_0$, so a larger reference shifts its onset to a longer context. Figure 6 compares $N_0 \in \{1\text{K}, 2\text{K}, 4\text{K}, 8\text{K}\}$ for four models on AIME-26 and three datasets for Qwen3-8B, using 30 inputs per model–dataset pair. We report the reduction in total verifier calls per generated token relative to the default reference of 2K for Qwen/Llama and 4K for GLM. Positive values indicate fewer calls per token, while zero denotes the default reference.




(a) Across models (b) Across datasets

Figure 6: Sensitivity to the reference cardinality across models and datasets. Verifier-call-rate savings are relative to 2K for Qwen/Llama and 4K for GLM.

The effect of changing the reference differs across models. On AIME-26, increasing $N_0$ from 2K to 4K raises calls per token by $3.2\%$, $9.8\%$, and $2.0\%$ for Qwen3-4B, Qwen3-8B, and Llama-3.1-8B, respectively. GLM shows the opposite preference between these two settings, requiring $1.2\%$ more calls at 2K than at 4K. For Qwen3-8B, the pattern is similar across AIME-26, GPQA-Diamond, and LGB(Diary). The 1K setting stays close to 2K, while 8K requires approximately 13–$14\%$ more calls per token. These results support a shared reference across the tested Qwen3-8B workloads, but suggest checking it again when changing the model rather than treating 2K as universal. Appendix C.6 provides the generation and aggregation details.

## 6 Discussion and Limitations

Query scaling changes how sharply the draft model attends, but cannot by itself correct every source of disagreement with the target. For fixed Queries and Keys, positive scaling preserves score order and can reinforce an incorrect preference. The layerwise experiment illustrates this distinction: restoring attention mass more closely adds little to MAL. Learned updates may still be useful when the mismatch concerns content rather than concentration; combining selective online adaptation with Query scaling is a natural extension.

The theoretical guarantee is narrower than the empirical evaluation. Our reference-inverse mass-preservation result assumes $C_t \geq C_{\text{ref}}$, agreement between current and reference contrasts throughout the application range, and inactive upper clipping. The deployed rule uses a Taylor approximation and a fixed slope, whose adequacy also depends on current/reference contrast offsets (Appendix C.5). Its slope and reference count should therefore be validated for a new draft model, rather than assumed to recover a universal calibration rule.

Our end-to-end evidence is strongest for DFlash under greedy decoding; the transfer experiments show that the benefit can vary with the target–draft pair. The multi-turn study extends acceptance measurements to 100K generated tokens, not latency measurements, and retains repetitive continuations. Since repetitive text can be easy to draft, evaluating answer quality alongside latency is important before interpreting high long-context MAL as a practical gain. Whether the latency gains persist under stochastic decoding, batching, or concurrent serving remains untested.

## 7 Conclusion

We presented SharpDraft, which scales draft Queries according to visible-Key cardinality without training the draft model. Separating attention-score contrast from the number of competing Keys leads to a reference-based correction, implemented through a fixed-slope approximation derived from a Taylor expansion. On the evaluated DFlash workloads, this correction yields higher end-to-end speedup than Frozen and online adaptation while matching Frozen's peak allocated memory. The ablations show a modest but consistent benefit from varying the scale with context, while more precise mass restoration adds little further acceptance.

### AI USE STATEMENT

Generative AI tools were used as auxiliary support for implementation, debugging, software validation, and writing refinement. This included assistance with smoke testing, parity checks, and other implementation-level verification tasks, as well as improving the clarity and presentation of the manuscript. The research hypotheses and experimental protocol were formulated independently by the authors. All AI-assisted outputs were reviewed by the authors, who retain full responsibility for the content and artifacts reported in this work.

### REPRODUCIBILITY STATEMENT

Section 3 specifies SHARPDRAFT and its inference integration, while Sections 4 and 5 describe the evaluation setup and ablation studies. Appendix A provides the assumptions and theoretical derivations. Implementation details, data selection and prompt construction, hyperparameters, and measurement protocols are provided in Appendices B and C. Code and evaluation artifacts will be released upon acceptance.

## REFERENCES

Zihao An, Huajun Bai, Ziqiong Liu, Dong Li, and Emad Barsoum. PARD: Accelerating LLM inference with low-cost PARallel draft model adaptation. In *International Conference on Learning Representations*, pp. 144562–144579, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/ea17f1a1685d65e66af19da44471b9ae-Abstract-Conference.html.

Ge Bai, Jie Liu, Xingyuan Bu, Yancheng He, Jiaheng Liu, Zhanhui Zhou, Zhuoran Lin, Wenbo Su, Tiezheng Ge, Bo Zheng, and Wanli Ouyang. MT-Bench-101: A fine-grained benchmark for evaluating large language models in multi-turn dialogues. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7421–7454. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.401. URL https://aclanthology.org/2024.acl-long.401/.

Yushi Bai, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongWriter: Unleashing 10,000+ word generation from long context LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/59f278de1619bdb6b53fd04e8e0976e0-Abstract-Conference.html.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023. URL https://arxiv.org/abs/2302.01318.

Jian Chen, Yesheng Liang, and Zhijian Liu. DFlash: Block diffusion for flash speculative decoding. In *International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/64301.

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. Beyond benchmarks: MathArena as an evaluation platform for mathematics with LLMs. *arXiv preprint arXiv:2605.00674*, 2026. URL https://arxiv.org/abs/2605.00674.

EAGLE Team, vLLM Team, and TorchSpec Team. EAGLE 3.1: Advancing speculative decoding through collaboration between the EAGLE team, vLLM, and TorchSpec. vLLM Blog, 2026. URL https://vllm-project.github.io/2026/05/26/eagle-3-1.html.

Doğaç Eldenk, Payal Mohapatra, Yigitcan Comlek, Kaan Oktay, Hongyang Zhang, and Stephen Xia. Attention drift: What autoregressive speculative decoding models learn. *arXiv preprint arXiv:2605.09992*, 2026. URL https://arxiv.org/abs/2605.09992.

Shicheng Fang, Yuxin Wang, Xiaoran Liu, Jiahao Lu, Chuanyuan Tan, Xinchi Chen, Yining Zheng, Xuanjing Huang, and Xipeng Qiu. AgentLongBench: A controllable long benchmark for long-contexts agents via environment rollouts. *arXiv preprint arXiv:2601.20730*, 2026. URL https://arxiv.org/abs/2601.20730.

Chi Han, Qifan Wang, Hao Peng, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. LM-Infinite: Zero-shot extreme length generalization for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3991–4008. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-long.222. URL https://aclanthology.org/2024.naacl-long.222/.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Avinash Kumar, Sujay Sanghavi, and Poulami Das. Test-time speculation. *arXiv preprint arXiv:2605.09329*, 2026. URL https://arxiv.org/abs/2605.09329.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19274–19286. PMLR, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE: Speculative sampling requires rethinking feature uncertainty. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 28935–28948. PMLR, 2024. URL https://proceedings.mlr.press/v235/li24bt.html.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-3: Scaling up inference acceleration of large language models via training-time test. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=4exx1hUffq.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, pp. 39578–39601, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Zhan Ling, Kang Liu, Kai Yan, Yifan Yang, Weijian Lin, Ting-Han Fan, Lingfeng Shen, Zhengyin Du, and Jiecao Chen. LongReason: A synthetic long-context reasoning benchmark via context expansion. *arXiv preprint arXiv:2501.15089*, 2025. URL https://arxiv.org/abs/2501.15089.

Xiaoxuan Liu, Lanxiang Hu, Peter Bailis, Alvin Cheung, Zhijie Deng, Ion Stoica, and Hao Zhang. Online speculative decoding. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 31131–31146. PMLR, 2024. URL https://proceedings.mlr.press/v235/liu24y.html.

Sumeet Ramesh Motwani, Daniel Nichols, Charles London, Peggy Li, Fabio Pizzati, Acer Blake, Hasan Hammoud, Tavish McDonald, Akshat Naik, Alesia Ivanova, Vignesh Baskaran, Ivan Laptev, Ruben Glatt, Tal Ben-Nun, Philip Torr, Natasha Jaques, Ameya Prabhu, Brian Bartoldson, Bhavya Kailkhura, and Christian Schroeder de Witt. LongCoT: Benchmarking long-horizon chain-of-thought reasoning. *arXiv preprint arXiv:2604.14140*, 2026. URL https://arxiv.org/abs/2604.14140.

Timur Mudarisov, Mikhail Burtsev, Tatiana Petrova, and Radu State. Limitations of normalization in attention. In *Advances in Neural Information Processing Systems*, volume 38, pp. 98312–98335. Curran Associates, Inc., 2025. doi: 10.52202/085713-3285. URL https://papers.neurips.cc/paper_files/paper/2025/hash/8d6fa78de53d5f24b67faf51aec86bbc-Abstract-Conference.html.

Ken M. Nakanishi. Scalable-Softmax is superior for attention. *arXiv preprint arXiv:2501.19399*, 2025. URL https://arxiv.org/abs/2501.19399.

Yu-Yang Qian, Hao-Cong Wu, Yichao Fu, Hao Zhang, and Peng Zhao. When drafts evolve: Speculative decoding meets online learning. In *International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/63017.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof Q&A benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Petar Veličković, Christos Perivolaropoulos, Federico Barbero, and Razvan Pascanu. Softmax is not enough (for sharp size generalisation). In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 61190–61211. PMLR, 2025. URL https://proceedings.mlr.press/v267/velickovic25a.html.

Yuhao Wu, Ming Shan Hee, Zhiqing Hu, and Roy Ka-Wei Lee. LongGenBench: Benchmarking long-form generation in long context LLMs. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2409.02076.

Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 7655–7671. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.456. URL https://aclanthology.org/2024.findings-acl.456/.

Penghui Yang, Cunxiao Du, Fengzhuo Zhang, Haonan Wang, Tianyu Pang, Chao Du, and Bo An. LongSpec: Long-context lossless speculative decoding with efficient drafting and verification. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1826–1844. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.83. URL https://aclanthology.org/2026.acl-long.83/.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html. Datasets and Benchmarks Track.

Yongchao Zhou, Kaifeng Lyu, Ankit Singh Rawat, Aditya Krishna Menon, Afshin Rostamizadeh, Sanjiv Kumar, Jean-François Kagy, and Rishabh Agarwal. DistillSpec: Improving speculative decoding via knowledge distillation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=rsY6J3ZaTF.

# A Theoretical Analysis

## A.1 Exact Partition Identity

For a current query define

$$F_t(\beta) = \log\left(\frac{1}{k}\sum_{i\in S_t} e^{\beta z_{t,i}}\right) - \log\left(\frac{1}{C_t}\sum_{j\in B_t} e^{\beta z_{t,j}}\right). \tag{6}$$

For $p_t(\beta) = \sum_{i\in S_t} a_{t,i}(\beta)$, direct algebra gives

$$\operatorname{logit} p_t(\beta) = F_t(\beta) - \log\frac{C_t}{k}. \tag{7}$$

At fixed $F_t(1)$, increasing $C_t$ decreases uncalibrated top-$k$ mass, without requiring the selected keys to be task-relevant.

### A.2 Reference Contrast and Monotonicity

For numerical inversion, let $\mathcal{C}_{\text{ref}}$ be a finite query set disjoint from evaluation, with each $F_x$ normalized by its own partition sizes as in Section 3.1. The frozen reference curve is

$$F_{\text{ref}}(\beta) = \mathbb{E}_{x \in \mathcal{C}_{\text{ref}}}[F_x(\beta)]. \tag{8}$$

For a specified reference competitor count $C_{\text{ref}} = N_0 - k > 0$, define

$$\operatorname{logit} \rho_{\text{ref}} = F_{\text{ref}}(1) - \log \frac{C_{\text{ref}}}{k}. \tag{9}$$

Only the reference-inverse rule requires the measured curve; the deployed fixed-slope rule does not.

Let $w^S_{x,i}(\beta)$ and $w^B_{x,j}(\beta)$ be softmax weights within each partition. Differentiating gives

$$F'_{\text{ref}}(\beta) = \mathbb{E}_{x \in \mathcal{C}_{\text{ref}}} \left[ \sum_{i \in S_x} w^S_{x,i}(\beta) z_{x,i} - \sum_{j \in B_x} w^B_{x,j}(\beta) z_{x,j} \right]. \tag{10}$$

Under strict top-$k$ separation, every top-$k$ logit exceeds every background logit; the first weighted mean is therefore larger. For a finite calibration set this establishes strict monotonicity of the average and the existence of an inverse on the selected interval.

### A.3 Reference Inverse and Conditional Mass Preservation

For $\beta_{\max} \geq 1$, the complete reference-inverse rule is

$$r_t = \max\left\{ \frac{C_t}{C_{\text{ref}}}, 1 \right\}, \tag{11}$$

$$\beta_t = F^{-1}_{\text{ref}}(\min\{F_{\text{ref}}(1) + \log r_t,\ F_{\text{ref}}(\beta_{\max})\}). \tag{12}$$

The inverse is evaluated numerically on $F_{\text{ref}}([1, \beta_{\max}])$ using a frozen curve or lookup table. Apply $q_t \leftarrow \beta_t q_t$; the clamps give unit scale below $C_{\text{ref}}$ and cap it at $\beta_{\max}$. The conditional guarantee below applies when neither clamp changes the matching solution and the stated contrast assumptions hold.

**Proposition 1** (Conditional reference-mass preservation)**.** *Assume $C_t \geq C_{\text{ref}}$, inactive upper clipping, and $F_t(\beta) = F_{\text{ref}}(\beta)$ throughout the application range. The reference inverse in equation 12 then gives $p_t(\beta_t) = \rho_{\text{ref}}$. If instead $|F_t(\beta_t) - F_{\text{ref}}(\beta_t)| \leq \varepsilon$, the same count and clipping conditions give*

$$|p_t(\beta_t) - \rho_{\text{ref}}| \leq \varepsilon/4. \tag{13}$$

*Proof.* Since $C_t \geq C_{\text{ref}}$ and upper clipping is inactive, equation 12 gives $F_{\text{ref}}(\beta_t) = F_{\text{ref}}(1) + \log(C_t/C_{\text{ref}})$. Using contrast equality and equation 7,

$$\begin{aligned} \operatorname{logit} p_t &= F_t(\beta_t) - \log(C_t/k) \\ &= F_{\text{ref}}(1) + \log(C_t/C_{\text{ref}}) - \log(C_t/k) \\ &= F_{\text{ref}}(1) - \log(C_{\text{ref}}/k) = \operatorname{logit} \rho_{\text{ref}}. \end{aligned}$$

Thus $p_t = \rho_{\text{ref}}$. With contrast error at most $\varepsilon$, log-odds error is at most $\varepsilon$. The logistic function is $1/4$-Lipschitz, yielding equation 13. □

### A.4 Training-Distribution Reference and Approximation Scope

When the training-query distribution is available, define $F_{\text{train}}(\beta) = \mathbb{E}_{x \sim \mathcal{D}_{\text{train}}}[F_x(\beta)]$ and the effective competitor count

$$C_{\text{train}} = \exp(\mathbb{E}_{x \sim \mathcal{D}_{\text{train}}}[\log(N_x - k)]). \tag{14}$$

If the same training distribution defines $F_{\text{train}}$ and $C_{\text{train}}$, then

$$\mathbb{E}_{x\sim\mathcal{D}_{\text{train}}}[\operatorname{logit} p_x(1)] = F_{\text{train}}(1) - \log\frac{C_{\text{train}}}{k}. \tag{15}$$

The effective total-Key reference $N_{\text{train}} = C_{\text{train}} + k$ generally differs from $\exp(\mathbb{E}[\log N_x])$. The exact ratio $(N_t - k)/C_{\text{train}}$ is approximated by total-length ratios only when $N$ is large relative to $k$.

This reference represents mean log-odds rather than mean attention mass. Without the training distribution, $N_0$ is a fixed experimental proxy (Appendix B.1), not a measurement of training-time attention.

The first-order Taylor expansion around $\beta = 1$ underlying the fixed-slope approximation is

$$F_{\text{ref}}(\beta) \approx F_{\text{ref}}(1) + \Delta_{\text{ref}}(\beta - 1), \qquad \Delta_{\text{ref}} = F'_{\text{ref}}(1) > 0, \tag{16}$$

which would give

$$\beta_t \approx \min\left\{\beta_{\max}, 1 + \frac{\log r_t}{\Delta_{\text{ref}}}\right\}. \tag{17}$$

Equation 5 replaces $\Delta_{\text{ref}}$ with the fixed slope $s$. Its error depends on slope choice, the Taylor remainder, and current/reference contrast mismatch. The fixed-slope rule inherits the guarantee of Proposition 1 when its assumptions hold and the reference contrast is linear with slope $s$.

# B Implementation and Evaluation Protocols

## B.1 Query-Scaling Implementation

The DFlash experiments use $k = 32$, $s = 5$, and $\beta_{\max} = e^{0.5}$, with $N_0 = 2048$ for primary requests and Qwen/Llama multi-turn sessions and $N_0 = 4096$ for GLM. These are fixed experimental proxies, not estimates of the training-time visible-Key distribution. GLM's setting is informed by a separate held-out MATH-500 replay comparison; it is not selected on the reference-sensitivity evaluation inputs.

**Finite-precision application.** For the BF16 Q-normalization output $y$, the primary implementation applies

$$y' = y + \operatorname{cast}_{\text{BF16}}\big(\operatorname{float}(y)(\widehat{\beta}_t - 1)\big). \tag{18}$$

The gain delta and product are computed in FP32, and the product is cast to BF16 before addition; normalization weights remain unchanged.

**Cost accounting.** In full attention, evaluating the scalar requires only context length and block size; rescaling is proportional to the current query tensor size. Its arithmetic and temporary allocations are included in E2E time and peak memory, without a training copy, gradients, or optimizer state.

## B.2 End-to-End Evaluation

Table 1 uses matched DFlash draft models with PyTorch SDPA on one NVIDIA RTX 6000 Ada GPU per run. The cohort comprises 30 AIME-26 questions, all 198 GPQA-Diamond questions, and 100 LongGenBench Daily Diary prompts per model. GPQA answer choices are permuted with seed zero in source order. Speculative arms undergo a first-question token and finish-reason parity check against Frozen before independent generation of subsequent requests. Methods share prompts, target–draft precision, and verification policy. AR uses the same target checkpoint and precision without a draft model.

**Online adaptation.** Full Adaptation uses Adam with learning rate $5\times10^{-5}$ and momentum coefficients $(0.9, 0.999)$. LoRA uses AdamW with learning rate $10^{-3}$, momentum coefficients $(0.9, 0.95)$, and zero weight decay. Both use an update stride of five verification rounds.

**Metrics.** MAL is emitted tokens, including correction or bonus tokens, divided by target verification rounds. E2E includes prefill, drafting, verification, calibration or learning, and synchronization. Speedup is total reference E2E divided by total method E2E, with a geometric mean across benchmarks. Tables 1 and 2 both define E2E speedup relative to target-only AR on the same model, prompts, and hardware. Peak memory is the maximum GPU allocation across the 228 AIME-26 and GPQA-Diamond requests. Update cost is training-stream CUDA time divided by completed optimizer updates on these requests; zero denotes no optimizer updates.

**Daily Diary prompts and stopping.** We use prompt ordinals 100–199 from the official long split, each requesting 365 entries. Every request starts from fresh state with the raw prompt and BOS when defined, without a chat template or added reasoning instruction. Greedy decoding stops at EOS, the first decoded `*** finished`, 32,000 new tokens, or the 40,960-token context limit. The Llama SHARPDRAFT run instead used a 32,768-token output cap, with seven outputs exceeding 32,000. All outputs are retained, without repetition guards. Runs use BF16, PyTorch 2.11.0+cu128, Transformers 5.5.4, and SDPA with flash, memory-efficient, and math backends enabled and cuDNN disabled.

**Request-time visualization.** Figure 2 averages AIME-26 E2E times and per-request training-stream CUDA durations. Training may overlap decoding: the right-aligned highlights neither locate execution intervals nor form an additive critical-path decomposition. Drafting, verification, and calibration are not timed separately, and the unhighlighted portion is not a measured decoding component.

### B.3 TRANSFER ACROSS DRAFT MODEL FAMILIES

Table 2 evaluates Qwen3-8B on H200 using 30 AIME-26 questions, 73 GPQA-Diamond questions, and 100 Daily Diary prompts. For EAGLE 3.1, we use the Attention Drift authors' released checkpoint `Dogacel/paper-attention-drift-qwen3-post` at revision `6e6ee69bf2e72e0e0654425bdca06d7b097e52a9` (Eldenk et al., 2026). Its configuration enables FC normalization and post-norm hidden-state feedback (`fc_norm=true`, `norm_output=true`). PARD uses twelve proposals and $N_0 = 8192$; EAGLE 3.1 uses eight proposals and $N_0 = 2048$. Both use $k = 32$, $s = 5$, and a log-gain cap of $0.5$. The LoRA controls schedule updates every five verification rounds; the EAGLE 3.1 run starts at round five and permits up to 8,192 updates.

### B.4 LONG-GENERATION MOTIVATION

Figure 1 uses Qwen3-8B on the first LongGenBench Diary prompt in our evaluation (source ordinal 100), separately from the aggregate latency evaluation. The throughput measurements use an RTX 6000 Ada GPU, BF16 greedy decoding, and a 40,000-token output cap. DFlash, PARD, and EAGLE 3.1 use frozen draft models; EAGLE 3.1 uses top-1 drafting.

**Throughput.** Panel (a) averages three fresh-process runs per method, including a separately measured AR baseline. Decode rates use 512-token elapsed-time windows at a stride of 64 tokens, with timestamps interpolated within each run; prefill is excluded. For display, each mean-rate curve, including AR, is smoothed over a centered 1,024-token span. The linear vertical axis shows absolute decode throughput in tokens per second. Methods may generate different output sequences, so positions represent generated-token counts rather than matched text prefixes.

**Attention concentration.** Panel (b) replays each draft model along its saved token sequence and samples top-32 attention mass every 64 generated tokens, averaging over draft layers, heads, and proposal queries. Lines show centered 2,048-token local means; shaded bands show $\pm 2$ population standard deviations of the sampled values within the same window. Windows use only observed positions at the boundaries, without padding or extrapolation. The bands describe local variation across token positions, not confidence intervals or uncertainty across repeated runs. The draft architectures have different query scopes and visible-Key sets, so cross-method mass levels are descriptive.

### B.5 Multi-Turn Acceptance

Figure 3 uses a 223-question evaluation subset of MATH-500 (Lightman et al., 2024), with thirty order seeds (2026092600–2026092629) shared across methods within each model. The source is the `test.jsonl` file from `HuggingFaceH4/MATH-500` at revision `6e4ed1a2a79af7d8630a6b768ec859cb5af4d3be`. Of its 500 questions, 262 have difficulty level 4 or 5 (128 and 134, respectively). For each original row, `source_id` is the literal prefix `math500:` followed by its `unique_id`; if that field is absent, we append the decimal zero-based row index in the original file, before filtering or shuffling. For example, `test/precalculus/881.json` becomes `math500:test/precalculus/881.json`. We hash the UTF-8 encoding of `source_id` with SHA-256, interpret the full hexadecimal digest as a base-16 integer $h$, and retain the question if $h \bmod 5 \neq 0$. This selects 223 evaluation questions and excludes the other 39. For each seed, the full 262-question pool is shuffled before applying this split filter, and questions are not reused within a session. EOS or 5,000 new tokens advances to the next question while retaining caches and adaptation state; state resets between sessions. Repetitive sessions are retained, so MAL measures acceptance on the observed continuations rather than answer quality.

**GLM checkpoints.** The GLM target is `zai-org/GLM-4.7-Flash` at revision `7dd20894a642a0aa287e9827cb1a1f7f91386b67`. Its DFlash draft is `Dariusfar/GLM-4.7-Flash-DFlash` at revision `d08993a8ec6fc0b50eb539bafa9f77a0de9a60ac`. All thirty seeds and four methods initialize from this target–draft pair and use the same pinned MATH-500 source and evaluation split.

**Generation and aggregation.** Runs use H200, vLLM, BF16, and greedy decoding. Qwen enables thinking with 16-token draft blocks; Llama uses its default template with 10-token blocks; GLM disables thinking with 16-token blocks. Qwen/Llama use a 40,960-token context limit and output budgets of 40,000 tokens for ten seeds and 32,000 for twenty; panels (a)–(c) use the common first 32,000 tokens. GLM uses a 100,000-token output budget and 131,072-token context limit. Budgets exclude input tokens; context exhaustion may stop a session early.

Complete 2,048-token windows advance every 128 tokens and are plotted at their midpoints. Rounds are assigned by their starting generated-token positions; each session contributes emitted tokens divided by verification rounds, and session ratios receive equal weight. Incomplete windows are omitted without padding. Bands use 20,000 paired seed-bootstrap resamples for pointwise 95% percentile intervals. Overlapping windows are correlated; late estimates use only contributing sessions.

## C Analysis Protocols

### C.1 Fixed vs. Dynamic Scaling

Table 3 uses Qwen3-8B/DFlash-b16 on all 30 AIME-26 questions with H200, BF16, PyTorch 2.11.0+cu130, and Transformers 5.5.4. SDPA enables flash, memory-efficient, and math backends; cuDNN and TF32 are disabled. Thinking is enabled, with greedy decoding to EOS or 32,768 new tokens and no repetition filtering. Frozen, five fixed scales $\beta = e^{0.1j}$ ($j = 1, \ldots, 5$), and SharpDraft share the primary implementation and Equation 18. Each problem runs all seven conditions on the same GPU, with Frozen first and the remaining order rotated across problems. Inclusion requires exact agreement in complete output tokens and termination across all conditions. All 30 problems pass, with 26 EOS-terminated and four capped outputs per condition.

**Aggregation and length groups.** For $P$ paired problems, let $T_i$ be emitted tokens after the initial prefill token and $V_i$ the verification-round count. The table averages whole-request ratios,

$$\mathrm{MAL}_{\mathrm{overall}} = \frac{1}{P} \sum_{i=1}^{P} \frac{T_i}{V_i}, \tag{19}$$

rather than the pooled-round ratio $\sum_i T_i / \sum_i V_i$. Final output length $L_i = T_i + 1$ assigns each problem to exactly one group: $[0, 10240)$, $[10240, 20480)$, or $[20480, 32768]$, including capped

outputs in the last group. The groups contain 9, 8, and 13 problems, respectively. Each group averages the same whole-request ratios; Overall is their problem-count-weighted mean. Global and group-specific best fixed scales are selected on this cohort, not on held-out data. These H200 acceptance measurements are separate from the RTX 6000 Ada latency results.

### C.2 Attention on Matched Prefixes

Figure 4a probes the Frozen histories from Appendix C.1, rebuilding target features in 512-token chunks. At nominal $N \in \{2048, 4096, 8192, 16384, 32768\}$, it evaluates visible-Key counts $N - 192, N - 128, N - 64, N$. All seven settings share the prefix, target features, anchor token, and 15 masked proposal slots, without a persistent draft cache. The scale uses the actual visible-Key count, including the 16-token draft block.

Attention logits and log-sum-exp reductions are reconstructed in FP32 from BF16 Queries and Keys. Top-32 mass is averaged equally over five layers, 32 heads, and 15 proposal positions, then over four probes per problem and finally over eligible problems. A problem contributes only when all four probes exist in every condition; terminated histories are not extended. The five windows contain 30/29/26/15/4 problems. Pointwise 95% percentile intervals use 20,000 problem-bootstrap resamples paired across settings; the figure displays means only. Instrumentation preserves draft hidden states bitwise at each problem's first and last probes.

### C.3 Cumulative Acceptance

Figure 4b uses the live verification rounds from the same paired cohort at checkpoints $x \in \{1024, 2048, \ldots, 32768\}$. Let $S(x) = \{i : L_i \geq x\}$ be the eligible problems, $\ell_{imr}$ the tokens emitted by method $m$ in round $r$, and $R_{im}(x)$ the number of complete rounds ending by $x$. The equal-problem mean is

$$\overline{\mathrm{MAL}}_m(x) = \frac{1}{|S(x)|} \sum_{i \in S(x)} \frac{\sum_{r=1}^{R_{im}(x)} \ell_{imr}}{R_{im}(x)}.$$

Emitted tokens include correction or bonus tokens but exclude the initial prefill token. Rounds are not split or interpolated; the final included round may end up to 15 tokens before the checkpoint. Long and capped outputs contribute at every earlier checkpoint, without extending terminated outputs.

The plot reports $100[\overline{\mathrm{MAL}}_m(x)/\overline{\mathrm{MAL}}_{\mathrm{dynamic}}(x) - 1]$: a ratio of means over identical problems, not a mean of per-problem percentages. The y-axis is linear within $\pm 2$ percentage points and logarithmic outside. The 1K/8K/16K/24K/32K checkpoints contain 30/26/15/9/4 problems. Pointwise intervals for absolute MAL use 20,000 paired problem-bootstrap resamples and are retained in the data; the figure shows relative means. Evaluating each problem at its own endpoint recovers Table 3; the 32K checkpoint instead uses only the 4 problems reaching it.

### C.4 Fixed-Slope Sensitivity and Reference Validation

**Slope sweep.** Figure 5a compares $s \in \{3, 4, 5, 6, 8\}$ on all 30 AIME-26 problems with Qwen3-8B/DFlash-b16, using native vLLM 0.27.1 in BF16 on H200. Only $s$ changes, with $N_0 = 2048$, $k = 32$, and $\beta_{\max} = e^{0.5}$ held fixed. Each setting uses greedy decoding with thinking enabled and a 32,768-token output cap, stopping at EOS but not on repetition. All 150 problem–setting pairs are complete, with one run per pair and no exclusions based on termination or repetition. The line averages whole-request MAL equally across problems, and the band shows pointwise 95% percentile bootstrap confidence intervals of this mean. We use 20,000 resamples of the 30 problems (seed 20260926), sharing the sampled problem indices across slopes to preserve pairing. These intervals quantify uncertainty across problems, not repeated-run variation or simultaneous coverage across slopes. MAL excludes the initial prefill token and divides the remaining emitted tokens by verification calls. Nine problems have identical outputs across all five settings, so this is a full-generation comparison rather than an output-matched comparison like Table 3. The mean fraction of post-reference verification rounds at the upper scale cap is 23.9% for $s = 3$, 2.8% for $s = 4$, and zero for $s \in \{5, 6, 8\}$. Thus, clipping affects the smaller slopes and should not be mistaken for slope insensitivity.

**Reference-curve diagnostic.** To check whether a slope of five approximates the reference correction, we numerically invert a measured $F_{\text{ref}}$ rather than using the deployed scaling rule. The curve pools 19,200 Query rows from two Qwen3-8B/DFlash reference trajectories at $N_0 = 2048$, evaluated at 513 scales between 1 and $e^{0.5}$. For each $N > N_0$, we solve Equation 4 by interpolation to obtain $\beta^*(N)$, then compute the effective reference slope

$$s_{\text{eff}}^{\text{ref}}(N) = \frac{\log[(N-k)/(N_0-k)]}{\beta^*(N)-1} = \frac{F_{\text{ref}}(\beta^*(N)) - F_{\text{ref}}(1)}{\beta^*(N)-1}. \tag{20}$$

This is a secant slope, distinct from the local derivative $\Delta_{\text{ref}} = F'_{\text{ref}}(1)$. The diagnostic uses 256-Key increments through 40K, excluding $N = N_0$ where the ratio is undefined. All numerical solutions lie inside the measured scale range, without clipping or extrapolation.

Figure 7 shows that the inferred slope stays between $4.995$ and $5.112$ over this grid. Thus, $s = 5$ closely approximates the correction on this pooled reference curve. This is not a distribution of live-state slopes or evidence of MAL optimality, since current-state contrast can differ from the reference. The state-dependent offset is examined in Appendix C.5.

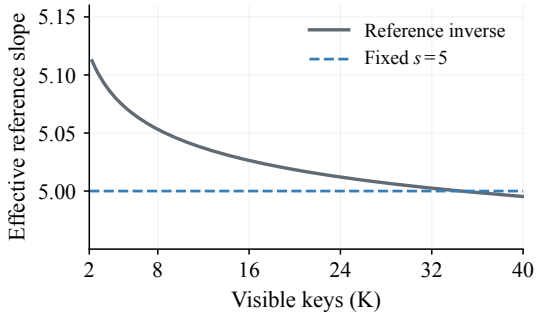



Figure 7: Effective slope inferred from the measured reference contrast for Qwen3-8B/DFlash. The dashed line marks the deployed setting $s = 5$.

### C.5 Layerwise Taylor Diagnostics

Figures 5b and 5c compare Frozen, the deployed fixed-slope rule, and Layerwise Taylor using Qwen3-8B/DFlash-b16 on H200 with native vLLM 0.27.1 and BF16. Each of the 30 AIME-26 requests is run once per method, with greedy decoding, thinking enabled, and a 32,768-token output cap. The analysis retains all requests, including capped and repetitive outputs. Frozen, SHARP-DRAFT, and Layerwise Taylor have five, four, and five capped outputs, respectively, with detected tail cycles in three, three, and four requests. Six requests have identical complete outputs across all three methods; the plotted profiles follow each method's full generation.

**Layerwise expansion and diagnostic measurements.** Layerwise Taylor uses the previous applied scale rounded to the nearest $0.1$ as its expansion center,

$$\widehat{\beta}_{\ell,t} = \text{clip}_{[1,2]}\left[c_{\ell,t} + \frac{F_{\text{ref},\ell}(1) + \log r_t - F_{\ell,t}(c_{\ell,t})}{F'_{\ell,t}(c_{\ell,t})}\right], \qquad c_{\ell,t} = \text{round}_{0.1}(\beta_{\ell,t-1}). \tag{21}$$

Here $F_{\ell,t}$ averages the contrast within layer $\ell$, and $r_t$ is the cardinality ratio from Equation 5. The layerwise contrast averages 32 heads and 15 proposal Queries in each of five draft layers. The reference contrasts and arithmetic-mean top-32 masses reuse two fixed SDPA calibration trajectories without refitting to vLLM. Both scaling methods use $N_0 = 2048$ and $k = 32$; the deployed scale is capped at $e^{0.5}$, while Layerwise Taylor is capped at 2 and remains one below $N_0$. Across the 4,450 post-reference layer–position diagnostic samples, the largest Layerwise Taylor scale is $1.916$ and 183 samples exceed $e^{0.5}$. Each layer uses the hidden state produced by preceding scaled layers. A separate diagnostic generation samples attention every 512 context tokens and reproduces the clean run's output tokens, verification rounds, and scales. Taylor errors at center one and at the adaptive center are evaluated on the same Taylor states and applied scales, including finite-precision Query

scaling and positional transformations. Each expansion uses its measured local derivative, so this comparison isolates re-centering rather than the use of the fixed slope $s = 5$. Holding the state and applied scale fixed on the Taylor trajectories, mean absolute contrast error falls from $0.0433$ at center one to $0.0023$ with re-centering. These diagnostics measure the centers visited during generation, not an independent sweep over fixed centers.

**Output-matched timing comparison.** The timing comparison in Section 5.2 uses seven requests with identical output tokens and termination between SHARPDRAFT and Layerwise Taylor, measured on the same physical GPU: 2026-I-01, 2026-I-04, 2026-I-05, 2026-I-07, 2026-I-09, 2026-II-01, and 2026-II-05. Matching is required between these two methods, not against Frozen. Six requests terminate naturally, while 2026-I-09 reaches the 32,768-token cap under both methods. The additional output-matched request, 2026-II-12, uses different GPUs and is excluded only from this timing comparison. Summed clean-run E2E times are $181.08$ seconds for SHARPDRAFT and $218.95$ seconds for Layerwise Taylor, giving per-request means of $25.87$ and $31.28$ seconds ($20.9\%$ longer for Taylor). The timing window includes prefill, decoding, calibration, and streaming delivery, but excludes model loading, reset, final trace serialization, and the separate diagnostic generations. These measurements describe the current native-vLLM implementation rather than an isolated kernel-overhead benchmark or a repeated-run latency estimate.

**Aggregation.** Mass error is the absolute percentage-point deviation from each layer's reference top-32 mass, using samples with $N_t \geq 2048$. MAL includes all verification rounds. Whole-request summaries weight all 30 requests equally. For the figure, observations are first averaged within each request's 1,024-token generated-position bins, then across contributing requests. Bands show pointwise 95% percentile bootstrap confidence intervals of the mean, using 20,000 whole-request resamples (seed 20260926) shared across methods and positions. Each resample averages only the requests observed in that bin; bins with one request have no band. Terminated generations are not extended, so both the number and composition of contributing requests can change with position. Points are placed at bin centers: the final point at 31.5K includes observations from the 31–32K bin, with five requests for Frozen, four for the deployed rule, and five for Layerwise Taylor. Matching a layer's mean contrast does not require zero arithmetic-mean mass error, because the contrast and probability averages are different objectives.

**Local slope and reference offset.** At the single-head level, reference-mass matching requires $F_t(\beta_t) = F_{\mathrm{ref}}(1) + L_t$, where $L_t = \log r_t$. Writing $A_t = F_{\mathrm{ref}}(1) - F_t(1)$ and $\Delta_t = F_t'(1)$, a first-order Taylor expansion gives $\beta_t - 1 \approx (A_t + L_t)/\Delta_t$. Representing this correction as $L_t/\Delta_{\mathrm{eff},t}$ would require

$$\Delta_{\mathrm{eff},t} = \frac{\Delta_t L_t}{A_t + L_t}. \tag{22}$$

The expression is undefined when $A_t + L_t = 0$ and can be nonpositive. In the 30-request diagnostic, median layer-mean derivatives $F'_{\ell,t}(c_{\ell,t})$ on the Taylor trajectories are $4.84$, $4.39$, $5.26$, $4.72$, and $4.64$. Effective denominators computed on Frozen states instead have medians $3.68$, $6.25$, $11.30$, $9.83$, and $7.69$. These are measurements on different trajectories; they illustrate that a local derivative near five does not account for the current/reference offset or justify a universal effective slope.

### C.6 REFERENCE-CARDINALITY SENSITIVITY

Figure 6(a) uses all 30 AIME-26 questions for Qwen3-4B, Qwen3-8B, Llama-3.1-8B, and GLM-4.7-Flash. Panel (b) reuses Qwen3-8B's AIME runs and adds the first 30 requests in the ordered GPQA73 selection and Diary source rows 100–129. Each of the six model–dataset groups uses Frozen and $N_0 \in \{1024, 2048, 4096, 8192\}$ on identical inputs, retaining capped and repetitive outputs.

Runs use H200, vLLM 0.27.1, BF16, and greedy decoding; each setting generates independently from fresh single-turn state, stopping at EOS or 32,768 new tokens. Diary also stops at `*** finished`. All Diary requests reach the cap in every arm. The engine uses tensor parallelism one, one active request, eager execution, and no prefix caching or asynchronous scheduling. Qwen reasoning prompts enable thinking, GLM disables it, and Llama uses its default template; Diary uses the raw primary prompt. Draft blocks contain 16 tokens for Qwen/GLM and 10 for Llama.

The implementation counts persistent context plus 16 for every model, retaining $k = 32$, $s = 5$, and $\beta_{\max} = e^{0.5}$.

For setting $j$, let $V_{i,j}$ and $T_{i,j}$ be verification calls and generated tokens after prefill. The pooled call rate and savings relative to reference $b$ are

$$R_j = \frac{\sum_i V_{i,j}}{\sum_i T_{i,j}}, \qquad S(j; b) = 100 \left(1 - \frac{R_j}{R_b}\right). \tag{23}$$

Reference $b$ is fixed at 2K for Qwen/Llama and 4K for GLM, rather than selected from the sweep. Pointwise 95% percentile intervals use 20,000 paired input-case bootstrap resamples, recomputing both totals in each draw; the figure displays point estimates. The y-axis is linear within $\pm 2$ percentage points and logarithmic outside, without omitting any range.