ADWIN: Adaptive Windows for Horizon-Aware On-Policy Distillation

Kun Liang[1,2] **Chenming Tang**[1,2] **Clive Bai**[3] **Weijie Liu**[3]
Saiyong Yang[3,*] **Yunfang Wu**[1,2,*]
[1]School of Computer Science, Peking University
[2]National Key Laboratory for Multimedia Information Processing, Peking University
[3]LLM Department, Tencent
kliang25@stu.pku.edu.cn wuyf@pku.edu.cn

Abstract

On-policy distillation (OPD) transfers reasoning behavior by training a student on teacher feedback along student-generated trajectories, but standard full-rollout training ties every update to a costly completion and can over-allocate supervision to late positions with low marginal value for the current student. We revisit this assumption through the useful supervision horizon: student-induced rollouts can drift from teacher-preferred continuations, while aligned prefixes may already preserve the long-horizon OPD update direction. We propose ADWIN, an adaptive-window framework for OPD that treats rollout length as an online admissibility decision, training on short teacher-anchored prefixes while using delayed full-rollout probes to audit prefix–full alignment and adapt the next horizon with staleness control. Across math and code reasoning benchmarks in single-task, multi-task, and strong-to-weak settings, ADWIN improves the accuracy–compute trade-off over full-rollout OPD and prefix-based baselines, reducing end-to-end training cost by up to $4.1\times$ while achieving comparable or better accuracy.

1 Introduction

On-policy distillation (OPD) has emerged as an important post-training paradigm for transferring reasoning behavior from a teacher model to a student model (Agarwal et al., 2024; Gu et al., 2024; Lu and Lab, 2025). Unlike offline knowledge distillation (KD) (Hinton et al., 2015; Kim and Rush, 2016), which distills teacher behavior from trajectories generated by the teacher, OPD queries the teacher on trajectories sampled from the student, thereby retaining the dense supervision of distillation while grounding it in the contexts induced by the student itself. This on-policy nature is especially beneficial for logical reasoning, enabling supervision over the intermediate states that the student actually visits (Wang et al., 2024). Recent studies have further demonstrated this potential, showing that OPD can reproduce strong reasoning gains at substantially lower compute, especially on reasoning-heavy tasks (Yang et al., 2025; LLM-Core Xiaomi, 2025; GLM-5-Team, 2026).

Despite the effectiveness of this paradigm, making supervision on-policy does not by itself determine how dense sequential signal should be used during training. Standard OPD practices take the direct operationalization, updating teacher feedback over a complete rollout. However, this fine-grained feedback does not imply that supervision at all positions is equally useful: in autoregressive generation, the student-induced context evolves recursively, and early decisions can causally constrain or distort the subsequent reasoning path (Dziri et al., 2023), pushing the sequence into off-manifold states that resemble a teacher-side exposure bias (Ranzato et al., 2015; Bengio et al., 2015).

The same causal structure also motivates a constructive sufficiency question for prefixes: whether an early rollout region can already provide sufficient directional evidence for the long-horizon OPD update. The joint perspective reframes OPD training window as a prefix-admissibility problem, highlighting a degree of freedom left implicit in standard OPD: how far dense teacher feedback should be instantiated along a student-induced rollout. A further operational question follows: given the inherent rollout cost of on-policy training, can OPD identify supervision that would ultimately be unnecessary and truncate the rollout online, before the corresponding full trajectory and backward signal are available?

To this end, we propose ADWIN, an **AD**aptive-**WIN**dow OPD framework that treats rollout truncation as a prefix-admissibility decision. Instead of tying every update to a complete trajectory, ADWIN performs synchronous OPD updates on short teacher-anchored prefix windows whose gradients

*Corresponding Authors

Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing, pages 10371–10389
October 24-29, 2026 ©2026 Association for Computational Linguistics

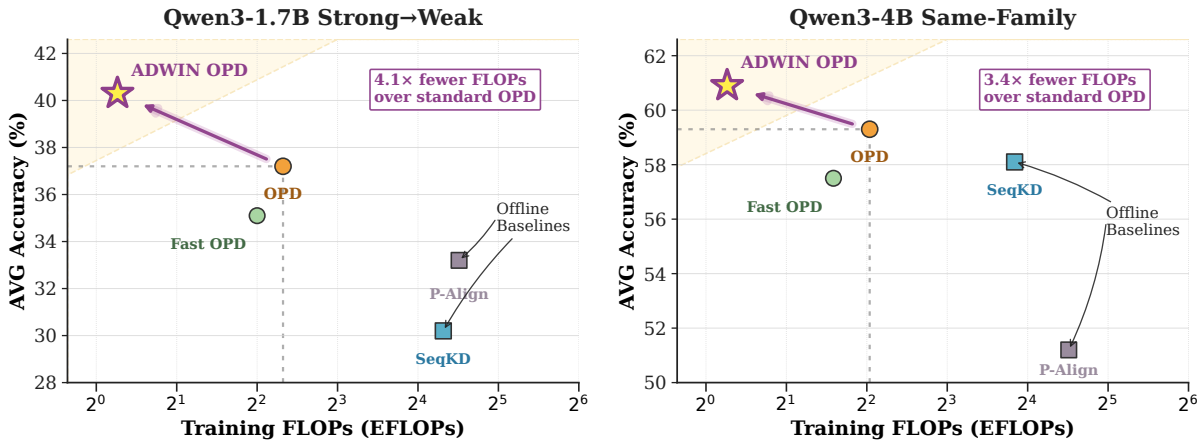

Figure 1: Accuracy–cost comparison of distillation methods. The x-axis reports log-scaled end-to-end training FLOPs and the y-axis reports average accuracy over the evaluated benchmarks. Left: Qwen3-1.7B strong-to-weak setting. Right: Qwen3-4B same-family setting, both averaged single task distillation of Math and Code tasks.

are intended to preserve the long-horizon OPD direction. In parallel, it asynchronously extends a small subset of unfinished prefixes into delayed full-rollout probes, keeping long-horizon evidence off the optimization critical path. When probes return, ADWIN uses their prefix–full consistency to audit candidate prefix window sizes and calibrate the next synchronous horizon. This allows ADWIN to select the shortest admissible window, reducing rollout cost and making updates non-blocking with respect to full-rollout completion, while still retaining long-horizon evidence for adaptive supervision.

As shown in Figure 1, ADWIN improves the OPD accuracy–cost trade-off through adaptive prefix-window updates with delayed full-rollout auditing, reducing end-to-end training FLOPs by up to 4.1× while achieving comparable or better accuracy than standard OPD and outperforming fixed or externally scheduled prefix baselines.

Our contributions are threefold:

1. We formulate prefix-window selection in OPD as a prefix-admissibility problem, linking efficient online rollout truncation to the preservation of long-horizon OPD update directions.

2. We introduce ADWIN, an adaptive-window OPD framework that combines synchronous prefix updates with delayed full-rollout probes to audit prefix–full alignment and calibrate.

3. We evaluate ADWIN on math and code benchmarks across single-task, multi-task, and strong-to-weak settings, showing that it substantially reduces rollout cost while preserving or improving OPD accuracy.

2 Preliminaries

2.1 Notation

Let D denote a prompt distribution. Given a prompt $\boldsymbol{x} \sim D$, a response sequence is $\boldsymbol{y} = (y_1, \ldots, y_{|\boldsymbol{y}|})$ with prefix $\boldsymbol{y}_{<t} = (y_1, \ldots, y_{t-1})$. We write $\pi_{\boldsymbol{\theta}}$ for the student policy, $\pi_{\boldsymbol{\phi}}$ for a frozen teacher, and $\mathcal{D}_{\mathrm{KL}}$ for the Kullback–Leibler divergence between two distributions over responses.

2.2 Offline Knowledge Distillation

KD transfers knowledge from the teacher $\pi_{\boldsymbol{\phi}}$ to the student $\pi_{\boldsymbol{\theta}}$. The general form aligns the two distributions over teacher-generated trajectories,

$$\mathcal{J}_{\mathrm{KD}}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x}\sim D,\, \boldsymbol{y}\sim\pi_{\boldsymbol{\phi}}(\cdot|\boldsymbol{x})}\Big[\mathcal{D}_{\mathrm{KL}}\big(\pi_{\boldsymbol{\phi}}(\boldsymbol{y}|\boldsymbol{x}) \,\|\, \pi_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})\big)\Big]. \quad (1)$$

In practice, obtaining the full output distribution of a teacher LLM is often computationally prohibitive or infeasible, Eq. (1) is commonly approximated via supervised fine-tuning (SFT) on teacher-generated trajectories. Either way, this *off-policy* paradigm fails to align the student's actions with its self-induced reasoning context; by imitating fixed oracle paths, the model circumvents the causal feedback of its own decisions, leading to logical drift when navigating its own prefixes at inference time (Ross et al., 2011).

2.3 On-Policy Distillation

OPD (Agarwal et al., 2024; Lu and Lab, 2025) resolves this mismatch by querying the teacher on

student-generated rollouts.

$$\mathcal{J}_{\mathrm{OPD}}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x}\sim D,\, \boldsymbol{y}\sim\pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})}\Big[\mathcal{D}_{\mathrm{KL}}\big(\pi_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}) \,\|\, \pi_{\boldsymbol{\phi}}(\boldsymbol{y}|\boldsymbol{x})\big)\Big]. \quad (2)$$

where $\boldsymbol{y}$ is sampled from the current student, making the training on-policy. Let $c_t = \log \pi_{\boldsymbol{\theta}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) - \log \pi_{\boldsymbol{\phi}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t})$ denote the step-wise cost. The standard policy gradient couples the parameter update at step t with all future costs in a return-to-go fashion (Schulman et al., 2018):

$$\nabla_{\boldsymbol{\theta}}\mathcal{J}_{\mathrm{OPD}} \approx \mathbb{E}\left[\sum_{t=1}^{|\boldsymbol{y}|} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) \sum_{t'=t}^{|\boldsymbol{y}|} \gamma^{t'-t} c_{t'}\right], \quad (3)$$

where $\gamma \in [0, 1]$ is the discount factor. Following Lu and Lab (2025), we apply a discount factor of $\gamma = 0$, resulting corresponding policy gradient:

$$\nabla_{\boldsymbol{\theta}}\mathcal{J}_{\mathrm{OPD}}^{\gamma=0} = \mathbb{E}\left[\sum_{t=1}^{|\boldsymbol{y}|} c_t \cdot \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t})\right]. \quad (4)$$

The full deduction can be found in Appendix A.1. This can be viewed as a dense on-policy RL objective, where each token carries a teacher-defined local reward signal of $-[\log \pi_{\boldsymbol{\theta}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) - \log \pi_{\boldsymbol{\phi}}(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t})]$, without waiting for rollout-level returns.

3 Admissible Prefix Windows for OPD

The two KD paradigms in Section 2 differ in how they handle the realizability of teacher supervision under the current student:

- **Off-policy KD** (Eq. (1)) makes teacher labels realizable, but not necessarily on student-realizable states.
- **On-policy Distillation** (Eq. (2)) makes states student-realizable, but long rollouts can introduce cascading deviations that render later teacher feedback less actionable.

The contrast reframes the problem from trajectory source to the useful horizon of on-policy supervision: where along the student rollout does teacher guidance remain useful signal rather than low-value or even noisy? We treat this as a **window-level admissibility** of locality and long-horizon fidelity: identify early student-induced region whose update remains reliable while still preserving the long-horizon OPD direction.

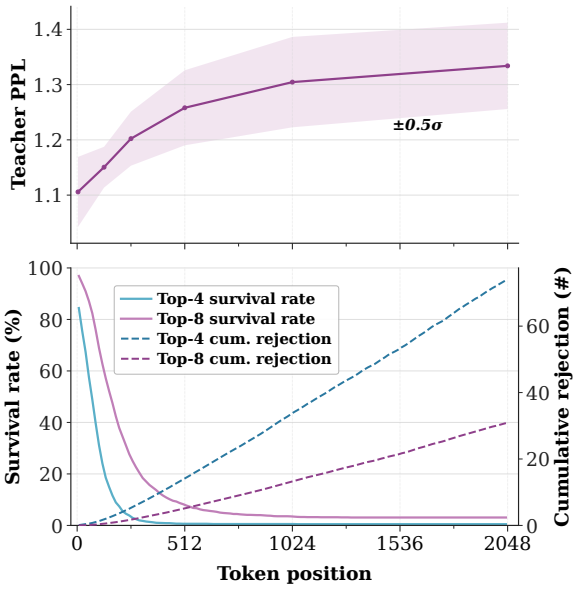

Figure 2: Per-position teacher-side drift along student-generated rollouts. Top: teacher per-position perplexity. Bottom: top-k survival rate and cumulative rejection under the teacher distribution ($N = 2048$ rollouts).

3.1 Supervision Drift in On-Policy Rollouts

For a student-generated rollout $\boldsymbol{y} = (y_1, \ldots, y_T)$, we track its prefix evolution under the teacher distribution and characterize the resulting drift with two diagnostics:

- **Teacher branching factor:**

$$\mathrm{BF}_t = \exp\Big(H[\pi_{\boldsymbol{\phi}}(\cdot \mid \boldsymbol{y}_{<t})]\Big),$$

  capturing the local multiplicity of plausible teacher paths, this quantity can be interpreted as the teacher's per-position *perplexity* (PPL).

- **Top-k rejection:** We track top-k departures over N student-generated rollouts. Let $r_{i,t}$ denote the rank of token $y_{i,t}$ under $\pi_{\boldsymbol{\phi}}(\cdot \mid \boldsymbol{y}_{i,<t})$. The survival fraction up to position T is

$$S_T^{(k)} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[\forall t \le T,\ r_{i,t} \le k],$$

  i.e., the fraction of rollouts with no top-k rejection through position T.

As shown in Figure 2, under student-induced prefixes, the teacher distribution becomes broader at later positions, as reflected by the rising PPL. Meanwhile, the rapid collapse of top-k survival and the steady growth of cumulative rejection show that realized rollouts increasingly escape the teacher's

preferred continuations, pushing later prefixes into regions the teacher would be unlikely to reach by itself. This compounding drift induces a reversed exposure bias on the teacher side. The teacher thereby does not provide uniformly reliable supervision, which can also be potentially harmful. We provide some qualitative cases in Appendix F.

3.2 Prefix Utility by Gradient Alignment

Following our observations on the locality side, we further consider a training-dynamic hinged indicator to quantify prefix region's optimization utility.

Surrogate Descent Utility. Consider a local neighborhood around $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. Let $\boldsymbol{g} := \nabla_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta}) \in \mathbb{R}^d$ denote the exact gradient of the target OPD objective, and let $\boldsymbol{M}(\boldsymbol{\theta}) \approx \mathbb{E}_{\boldsymbol{y}\sim\pi_{\boldsymbol{\theta}}}\left[(\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{y}))(\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{y}))^\top\right]$ be a local quadratic form representing curvature. The second-order surrogate improvement of a parameter update $\boldsymbol{\Delta}$ is then given by

$$\mathcal{S}_{\boldsymbol{M}}(\boldsymbol{\Delta}) := \boldsymbol{g}^\top \boldsymbol{\Delta} - \frac{1}{2}\boldsymbol{\Delta}^\top \boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{\Delta}. \tag{5}$$

For a surrogate update direction $\boldsymbol{d} \neq 0$, the maximum utility along $\boldsymbol{d}$ is

$$\mathcal{U}_{\boldsymbol{M}}(\boldsymbol{d}) := \max_{\eta\geq 0} \mathcal{S}_{\boldsymbol{M}}(\eta \boldsymbol{d}) = \frac{[\boldsymbol{g}^\top \boldsymbol{d}]_+^2}{2\,\boldsymbol{d}^\top \boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{d}}, \tag{6}$$

where $[\cdot]_+ := \max\{\cdot, 0\}$. The unconstrained optimal update in this local geometry is the natural gradient $\boldsymbol{\Delta}^\star = \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}$, which achieves the maximum local utility

$$\mathcal{U}_{\boldsymbol{M}}^\star := \frac{1}{2}\boldsymbol{g}^\top \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}. \tag{7}$$

The *relative utility* of an update $\boldsymbol{d}$ is then

$$\begin{aligned}\mathcal{E}_{\boldsymbol{M}}(\boldsymbol{d}) := \frac{\mathcal{U}_{\boldsymbol{M}}(\boldsymbol{d})}{\mathcal{U}_{\boldsymbol{M}}^\star} &= \frac{[\boldsymbol{g}^\top \boldsymbol{d}]_+^2}{(\boldsymbol{g}^\top \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g})(\boldsymbol{d}^\top \boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{d})} \\ &= \left[\cos_{\boldsymbol{M}}(\boldsymbol{d}, \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g})\right]_+^2.\end{aligned} \tag{8}$$

Evaluating Prefixes for Training Surrogates. Equation (8) condenses the local update utility into a single interpretable scalar: the metric-aware cosine between the candidate update and the ideal natural-gradient direction $\boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}$. For a prefix gradient $\boldsymbol{d}_p$ and a full-rollout gradient $\boldsymbol{d}_f$, their relative informational value is

$$\frac{\mathcal{E}_{\boldsymbol{M}}(\boldsymbol{d}_p)}{\mathcal{E}_{\boldsymbol{M}}(\boldsymbol{d}_f)} = \left(\frac{\cos_{\boldsymbol{M}}\left(\boldsymbol{d}_p, \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}\right)}{\cos_{\boldsymbol{M}}\left(\boldsymbol{d}_f, \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}\right)}\right)^2. \tag{9}$$

A prefix is therefore an effective surrogate if its gradient preserves the optimization-relevant M-cosine. In the transparent case $\boldsymbol{d}_f = \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}$, Eq. (8) reduces to $\mathcal{E}_{\boldsymbol{M}}(\boldsymbol{d}_p) = \cos^2_{\boldsymbol{M}}(\boldsymbol{d}_p, \boldsymbol{d}_f)$. This simplification provides an operational pathway for practically scheduling the training window.

4 ADWIN: Pushing Acceleration to Online Rollout Truncation

Although a prefix cannot give an unbiased estimate of the long-horizon update direction, Section 3 shows it can still yield a useful descent step on the OPD objective. We further exploit it as an operational question: given that window selection itself requires long-horizon evidence, can OPD adaptively truncate rollouts online without first incurring the full-generation cost that motivates truncation? To remedy this, we propose ADWIN, an adaptive-window training framework for OPD that makes region-size selection online and auditable, using delayed asynchronous probes to adapt the horizon of synchronous truncated updates.

4.1 Sync Updates and Async Probes

ADWIN separates OPD training into a synchronous update path and an asynchronous audit path. At optimization step t, the synchronous path uses only the current window length L_t: it samples student continuations up to L_t, evaluates the teacher-anchored OPD loss on this prefix, and immediately applies the resulting update. Thus the blocking cost of a training step is $C_{\text{sync}}(t) = C_{\text{roll}}(L_t) + C_{\text{sup}}(L_t)$, rather than the full-rollout cost $C_{\text{roll}}(L_{\max}) + C_{\text{sup}}(L_{\max})$. The full horizon is accessed only through delayed probes.

After each synchronous batch, ADWIN selects a small subset of unfinished prefixes and inserts them into a probe pool $\mathcal{P}$. Background workers continue these probes toward $L_{\max}$ over subsequent training steps, with each extension round capped by the current synchronous rollout budget. Consequently, full rollouts are not eliminated, but moved off the optimization critical path: they serve only to produce delayed audit batches for window calibration.

To control the mismatch between delayed probes and the evolving student policy, ADWIN enforces a hard staleness limit measured in optimization steps. A probe that exceeds this limit is forced to complete before being consumed by the scheduler. The returned probe batch is then passed to the admissibility audit described below.

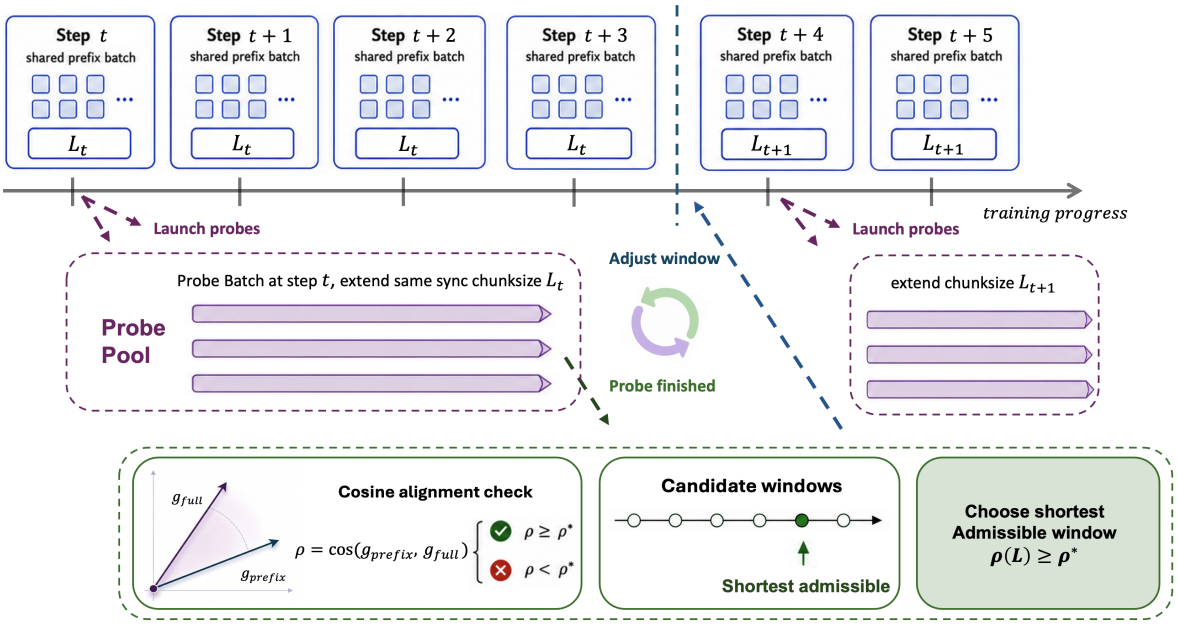

Figure 3: Workflow of ADWIN. The top row performs synchronous OPD updates on the current prefix window L_t: student rollouts are truncated at the shared window, teacher feedback is evaluated on the prefix, and the resulting loss is used for training updates. The middle probe path asynchronously continues a small probe batch of unfinished prefixes into delayed full-rollout probes. Returned probes are then used in the bottom audit panel to compute prefix–full gradient cosine for candidate window lengths, after which the scheduler updates L_{t+1} by choosing the shortest length whose alignment exceeds the admissibility threshold ρ^*.

4.2 Probe-Based Window Calibration

Upon a probe's return, we evaluate a candidate list of truncation lengths $\mathcal{L} = \{L_1, L_2, \dots\}$ to update the synchronous window L_t. Both the candidate prefix gradient $\bar{u}_t^{(L)}$ and the full-horizon probe reference u_t^{probe} are evaluated via a forward and backward pass under the current student parameters. Following the geometric criterion established in Section 3.2, we instantiate prefix admissibility using the finite-batch full-horizon probe gradient u_t^{probe} as an empirical reference and compute their metric-aware consensus $\rho_t(L) := \cos\big(u_t^{\text{probe}}, \bar{u}_t^{(L)}\big)$.

Decomposing u_t^{probe} into an aligned component and an orthogonal residual yields the prefix surrogate's effective signal-to-noise ratio: $\mathrm{SNR}_t(L) := \rho_t(L)^2/(1-\rho_t(L)^2)$. Requiring a strictly prefix signal dominated update ($\mathrm{SNR}_t(L) \geq 1$) as a uniform approximation directly derives our empirical lower bound admissibility threshold:

$$\rho_t(L) \geq \rho^* = \frac{\sqrt{2}}{2} \approx 0.707. \tag{10}$$

The scheduler updates the online training horizon to the shortest length that satisfies this alignment bound: $L_{t+1} = \min\{L \in \mathcal{L} \mid \rho_t(L) \geq \rho^*\}$. If the current prefix demonstrates strong gradient consensus ($\geq \rho^*$), the tracking window shrinks to maximize wall-clock acceleration; conversely, if the alignment drops below the conservative signal-to-noise threshold, the window gracefully expands to reclaim supervision fidelity.

5 Experiments

To comprehensively evaluate ADWIN, we conduct experiments on math and code generation tasks across three distinct settings: single-task, multi-task, and strong-to-weak.

5.1 Experimental Setup

Base Models. For the same-family single- and multi-task settings, we use Qwen3-4B-Non-Thinking (Yang et al., 2025) as the student and supervise it with the corresponding RL-tuned experts. For the strong-to-weak setting, we use Qwen3-1.7B-Non-Thinking as the student and Qwen3-30B-A3B-Instruct-2507 as the stronger teacher.

Training Details. We train the math task on Polaris-53K (An et al., 2025), and the code task on Eurus-RL-Code (Cui et al., 2025) for both RL and OPD phases. The RL experts are obtained with

Method	Mathematics			Code			Avg.	EFLOPs ↓
	AIME'24 avg@16	AIME'25 avg@16	Beyond AIME avg@8	HumanEval+ avg@4	MBPP+ avg@4	LCB v6 avg@4		
Initial Model	22.5	18.9	10.8	71.9	60.8	24.6	34.9	–
Teacher	61.3	58.9	34.0	83.4	75.6	32.0	57.5	–
(a) Single-Task								
SeqKD	61.7	56.9	34.0	82.2	75.5	38.0	58.1	14.3
P-Align	54.6	52.9	29.8	77.8	63.0	29.3	51.2	22.8
OPD	58.8	58.5	32.9	**92.6**	75.9	37.1	59.3	4.1
Fast OPD	60.4	54.9	33.3	83.3	76.0	37.3	57.5	3.0
ADWIN	**65.6**	**62.9**	**34.9**	85.9	**77.4**	**38.6**	**60.9**	**1.2**
(b) Multi-Task								
SeqKD	61.0	55.2	33.3	84.2	75.7	38.1	57.9	25.7
P-Align	53.5	50.8	31.8	77.8	65.4	31.6	51.8	46.7
OPD	60.4	56.1	**34.8**	83.8	77.0	37.1	58.2	8.2
Fast OPD	62.1	57.9	33.9	83.1	76.2	37.9	58.5	6.5
ADWIN	**64.2**	**59.8**	34.6	**84.9**	**77.4**	**38.4**	**59.9**	**2.7**

Table 1: Main results in the same-family settings. Student is Qwen3-4B-Non-Thinking; teachers are the corresponding RL-tuned checkpoints. Gray rows mark offline-distillation baselines. **Bold** marks the best student method within each setting; the highlighted row is ADWIN.

Method	Mathematics			Code			Avg.	EFLOPs ↓
	AIME'24 avg@16	AIME'25 avg@16	Beyond AIME avg@8	HumanEval+ avg@4	MBPP+ avg@4	LCB v6 avg@4		
Initial Model	13.3	7.9	4.5	57.1	40.2	13.1	22.7	–
Teacher	76.0	61.9	44.8	92.6	79.4	40.9	65.9	–
(c) Strong-to-Weak								
SeqKD	19.0	16.5	7.6	65.3	59.5	13.1	30.2	19.9
P-Align	27.1	19.6	8.8	64.7	60.9	18.1	33.2	22.8
OPD	30.4	22.5	10.8	**72.7**	64.0	22.6	37.2	5.0
Fast OPD	32.1	**27.7**	12.8	62.1	57.7	18.0	35.1	4.0
ADWIN	**34.2**	26.9	**13.4**	72.4	**67.1**	**27.7**	**40.3**	**1.2**

Table 2: Main results in the strong-to-weak setting. The student is Qwen3-1.7B-Non-Thinking, supervised by the stronger Qwen3-30B-A3B-Instruct-2507.

GRPO (Shao et al., 2024) using a binary correctness reward, with 128 prompts per batch and 8 rollouts. We implement ADWIN with candidate training windows $\mathcal{L} = \{64, 128, 256, 512, 1024, 2048\}$ under a maximum training horizon $L_{\max}$ of 8,192 tokens, selecting the shortest window whose probe-estimated prefix–full gradient cosine exceeds ρ^*; the probe batch size is set to 64 samples and maximum probe staleness is set to 5 optimization steps. See Appendix B.1 for more details.

Baselines. We evaluate ADWIN against different prefix-level designs of distillation. For OPD-family baselines, standard OPD updates on complete student trajectories; Fast OPD (Zhang et al., 2026a) uses a predefined schedule that linearly increases the prefix window; and fixed-window Prefix OPD trains with manually chosen prefixes without adaptive auditing. For offline baselines, SeqKD (Kim and Rush, 2016) trains on teacher-generated full trajectories, while P-Align (Liu et al., 2026) truncates teacher trajectories at a prefix length judged sufficient by the student, who then generates the remaining suffix for continuation. All baselines use the same student, teacher; specific implementation details can be found in Appendix B.2.

Evaluations Setup. We evaluate on three challenging mathematical reasoning tasks: AIME 2024 (AI-MO, 2024), AIME 2025 (OpenCompass, 2025), BeyondAIME (ByteDance-Seed, 2025) and three coding tasks: HumanEval+, MBPP+ (Liu et al., 2023), LiveCodeBench (v6 only) (Jain et al., 2025) . In all the evaluations, we set the tempera-

ture to 0.6, top-p to 0.95, top-k to 20 and maximum generation length to 30,720 tokens. The evaluation prompts can be found in Appendix E.

5.2 Main Results

ADWIN improves the end-to-end accuracy–cost trade-off of OPD. Tables 1 and 2 report the main results across the single-RL-teacher, multitask, and strong-to-weak settings. ADWIN achieves the best average performance among all evaluated distillation methods in all three settings while using substantially lower end-to-end compute than standard OPD. Compared with full-rollout OPD, ADWIN improves the average score from 59.3 to 60.9 in the single-RL-teacher setting, from 58.2 to 59.9 in the multitask setting, and from 37.2 to 40.3 in the strong-to-weak setting. At the same time, ADWIN reduces EFLOPs from 4.1 to 1.2, 8.2 to 2.7, and 5.0 to 1.2, respectively, corresponding to up to a $4.1\times$ reduction in end-to-end training cost.

Prefix truncation needs dynamic admissibility. ADWIN also improves over two prefix-level designs, P-Align and Fast OPD, where prefixes are selected either by offline student self-evaluation or by an externally predefined schedule. This supports the advantage of internalizing prefix selection into the OPD training dynamics: the effective horizon should be treated not as a fixed preprocessing decision, but as a training-time control variable responsive to evolving student–teacher interactions.

5.3 Analysis

For all analysis, we adopt the more discriminative strong-to-weak setting to provide a clear contrast, while the results under the peer distillation setup follow the same trajectory.

Effect of adaptive window selection. We evaluate this effect by replacing ADWIN's scheduler with fixed-window Prefix candidates used in training. As shown in Table 3, no static horizon is uniformly reliable: overly short windows can miss necessary long-horizon signal, while longer windows do not monotonically improve performance and may reintroduce low-value late-position supervision. This indicates that the useful horizon is not a tunable length fixed before training, but a training-time admissibility boundary that varies across settings and hinges on training dynamics.

Reliable gradient consensus enables window selection. We next examine an empirical observation that motivates the design of ADWIN: prefix gradients can form a reliable consensus with the full-rollout OPD gradient after batch aggregation.

Window policy	Math Avg.	Code Avg.	Overall Avg.
Prefix OPD-128	15.2	50.3	32.8
Prefix OPD-256	21.3	49.5	35.4
Prefix OPD-512	23.9	54.7	39.3
Prefix OPD-1024	24.0	**56.2**	40.1
Prefix OPD-2048	24.0	52.5	38.3
ADWIN	**24.8**	55.7	**40.3**

Table 3: Comparison of ADWIN and fixed-window policies in the strong-to-weak setting. Prefix OPD-L trains with a L-token window. Math/Code Avg. average the three benchmarks within each domain.

The cross-sample consensus. We first study the consensus effect of per-sample gradient aggregation, as illustrated in Figure 5. The batch-level micro prefix–full cosine rises quickly at short prefix lengths, even when the prefix covers only a small fraction of the full rollout. The macro cosine, obtained by averaging per-sample prefix–full alignments, remains consistently lower. This suggests a cancellation over high variance components across samples, which makes the prefix updates more reliable and useful signal even at the early positions, echoed by the non-trivial performance of aggressive truncated settings in Table 3.

The cross-batch consensus. We then examine whether the prefix-full gradient structure holds across batches, since which ensures the reliability of window selection in ADWIN's gradient probe. As shown in Figure 5, the gradient cosine remains highly stable across various probe batches. Consequently, the distribution across the threshold that triggers window decision flips remains minimal. Most importantly, we observe no cross-tier flips: sub-threshold cases do not simultaneously appear across non-adjacent window tiers, supporting the robustness of our gradient probe.

6 Related Work

Knowledge distillation for reasoning. We broadly categorize KD works by where the training data are sampled, resulting in offline (Hinton et al., 2015; Kim and Rush, 2016) and on-policy paradigms (Agarwal et al., 2024; Lu and Lab, 2025), yields different KL choices rooted in the inherent estimation principles of KL divergence, as well as subsequent works that optimize the use of these losses (e.g. adaptively combining

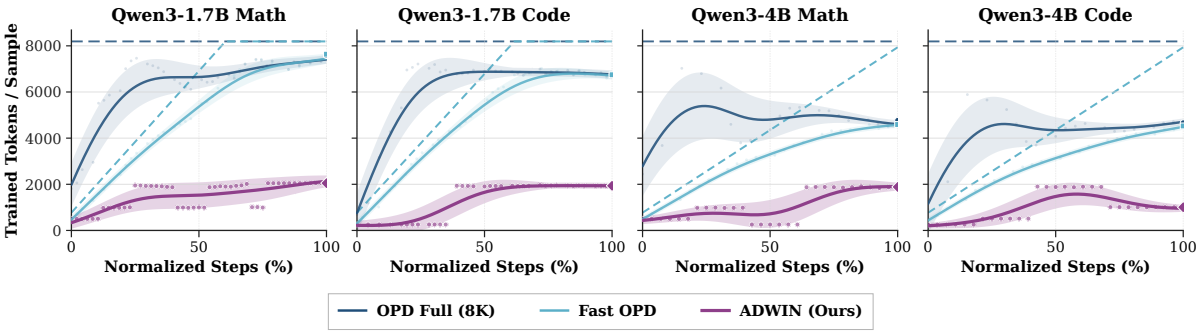

Figure 4: Evolution of the effective training horizon during OPD, with dashed lines denoting the maximum response length of the corresponding method (effectively same line for ADWIN). The curves report per step trained tokens over normalized training progress under full-rollout OPD, Fast OPD, and ADWIN. ADWIN maintains a substantially smaller horizon than full-rollout OPD and externally scheduled truncation by adapting the prefix window online.

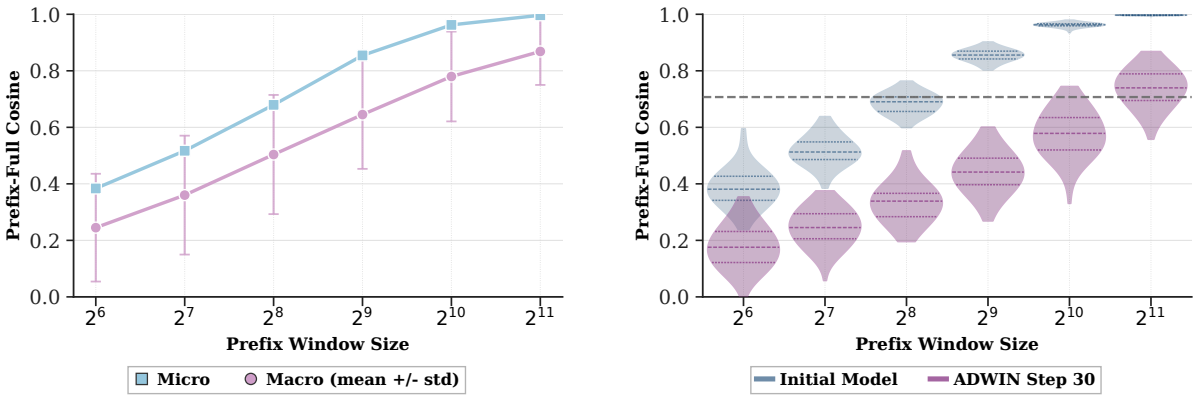

Figure 5: Prefix–full gradient cosine analysis on Qwen3-1.7B strong-to-weak setting, evaluated on Polaris. Left: macro cosine averages per-sample prefix–full alignments, while micro cosine is computed after batch-level gradient aggregation. Right: distributions of batch-level micro prefix–full cosine across 128 independent probe batches at the initial model and a middle-stage ADWIN checkpoint.

these two) (Wu et al., 2024; Wang et al., 2025). Within the focus of OPD, recent advances further push broader applications including: optimization stability (Jang et al., 2026; Fu et al., 2026), cross-family transfer (Minixhofer et al., 2025), self-distillation (Zhao et al., 2026; Hübotter et al., 2026), and distillation-to-RL extension (Yang et al., 2026b; Zhang et al., 2026b; Yang et al., 2026a). ADWIN focuses on a complementary question: selecting sufficient prefix windows for efficient OPD training.

Selective supervision for distillation. Efficient dense supervision often reduces redundant token-level updates by masking, filtering, or reweighting losses according to confidence, uncertainty, teacher–student disagreement, or estimated utility (Huang et al., 2025; Jung et al., 2025; Sang et al., 2026; Kim and Baek, 2026). Such methods improve how generated tokens are used, but typically do not reduce the rollout generation cost that dominates OPD training. Fast OPD (Zhang et al., 2026a) directly reduces this cost through externally scheduled prefix truncation, but its horizon is not audited against the current full-rollout update. ADWIN instead makes truncation an online admissibility decision: it trains on short prefix updates while using delayed probes to verify their alignment with the full-rollout OPD direction.

7 Conclusion

This paper presents ADWIN, an adaptive-window framework for efficient on-policy distillation. We show that full-rollout supervision is not uniformly necessary: student-induced drift can reduce the value of late-token feedback, while aligned prefixes can still preserve the long-horizon OPD update direction. ADWIN turns rollout truncation into a window-admissibility problem, training on short teacher-anchored prefixes while using de-

layed full-rollout probes to audit prefix–full gradient alignment and adapt the synchronous horizon. Across math and code benchmarks in single-task, multi-task, and strong-to-weak settings, ADWIN improves the accuracy–compute trade-off over full-rollout OPD and prefix-based baselines, substantially reducing end-to-end training cost.

Limitations

We elucidate the limitations of this work as follows. First, ADWIN adopts a synchronized truncation window for each training update rather than sample-wise length selection, in order to keep the online training path regular and reduce straggler-induced bubbles caused by heterogeneous rollout lengths within a batch. Consequently, the admissibility decision is made through batch-level prefix–full gradient consensus rather than per-example sufficiency. Although the threshold ρ^* provides a practical criterion for preserving the aggregated update direction and the empirical results are favorable, it does not theoretically guarantee that synchronized truncation preserves the performance of full-rollout OPD in all cases. Second, this work focuses on reasoning-intensive tasks, while the generalization of ADWIN to open-ended dialogue, instruction following, or multimodal reasoning remains untested. Third, ADWIN may be less suitable for tasks whose decisive supervision signals appear primarily near the end of a trajectory. Prefix–full gradient alignment mainly characterizes the consistency of optimization directions, and may not fully reflect suffix-specific requirements such as final-answer verification, proof closure, code executability, or long-range dependency repair.

Ethical Considerations

Use of AI Assistants

The algorithmic design and main methodology presented in this work were developed through human-led research and reasoning. During implementation, we utilized GitHub Copilot[1] for coding assistance. We affirm that the primary content and logic of the code are entirely our own work.

Potential Risks

This work introduces ADWIN (ADaptive WINdows), a framework for efficient on-policy distillation through adaptive rollout truncation. We acknowledge potential risks associated with accelerating supervision for reasoning models. Distilled students may inherit incorrect reasoning patterns, biases, or unsafe behaviors from the teacher and training data. Moreover, by reducing the cost of on-policy distillation, ADWIN may make it easier to scale the training and deployment of distilled reasoning models before their failure modes are fully understood. If used improperly, such efficiency gains could amplify teacher errors, dataset biases, or brittle reasoning behaviors across larger numbers of downstream models and applications.

[1] https://github.com/features/copilot

Acknowledgments

This work is supported by the National Natural Science Foundation of China (62076008).

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*.

AI-MO. 2024. Aime 2024. https://huggingface.co/datasets/AI-MO/aimo-validation-aime.

Chenxin An, Zhihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and Lingpeng Kong. 2025. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28.

ByteDance-Seed. 2025. Beyondaime: Advancing math reasoning evaluation beyond high school olympiads. https://huggingface.co/datasets/ByteDance-Seed/BeyondAIME. Hugging Face repository.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, et al. 2025. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. 2023. Faith and fate: Limits of transformers on compositionality. *Advances in neural information processing systems*, 36:70293–70332.

Yuqian Fu, Haohuan Huang, Kaiwen Jiang, Jiacai Liu, Zhuo Jiang, Yuanheng Zhu, and Dongbin Zhao. 2026. Revisiting on-policy distillation: Empirical failure modes and simple fixes. *arXiv preprint arXiv:2603.25562*.

GLM-5-Team. 2026. Glm-5: from vibe coding to agentic engineering. *Preprint*, arXiv:2602.15763.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. MiniLLM: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Haiduo Huang, Jiangcheng Song, Yadong Zhang, and Pengju Ren. 2025. Selectkd: Selective token-weighted knowledge distillation for llms. *arXiv preprint arXiv:2510.24021*.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, et al. 2026. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2025. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*.

Ijun Jang, Jewon Yeom, Juan Yeo, Hyunggu Lim, and Taesup Kim. 2026. Stable on-policy distillation through adaptive target reformulation. *arXiv preprint arXiv:2601.07155*.

Seongryong Jung, Suwan Yoon, DongGeon Kim, and Hwanhee Lee. 2025. Todi: Token-wise distillation via fine-grained divergence control. *Preprint*, arXiv:2505.16297.

Minsang Kim and Seung Jun Baek. 2026. Explain in your own words: Improving reasoning via token-selective dual knowledge distillation. *Preprint*, arXiv:2603.13260.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2023. Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zhenghao Liu, Zhuoyang Wu, Xinze Li, Yukun Yan, Shuo Wang, Zulong Chen, Yu Gu, Ge Yu, and Maosong Sun. 2026. Long-chain reasoning distillation via adaptive prefix alignment. *arXiv preprint arXiv:2601.10064*.

LLM-Core Xiaomi. 2025. MiMo: Unlocking the Reasoning Potential of Language Model – From Pretraining to Posttraining. *Preprint*, arXiv:2505.07608.

Kevin Lu and Thinking Machines Lab. 2025. On-policy distillation. *Thinking Machines Lab: Connectionism*. Https://thinkingmachines.ai/blog/on-policy-distillation.

Benjamin Minixhofer, Ivan Vulić, and Edoardo Maria Ponti. 2025. Universal cross-tokenizer distillation via approximate likelihood matching. *Preprint*, arXiv:2503.20083.

OpenCompass. 2025. Aime 2025. https://huggingface.co/datasets/opencompass/AIME2025.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 627–635, Fort Lauderdale, FL, USA. PMLR.

Hejian Sang, Yuanda Xu, Zhengze Zhou, Ran He, and Zhipeng Wang. 2026. Not all tokens are needed (nat): token efficient reinforcement learning. *arXiv preprint arXiv:2603.06619*.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2018. High-dimensional continuous control using generalized advantage estimation. *Preprint*, arXiv:1506.02438.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Guanghui Wang, Zhiyong Yang, Zitai Wang, Shi Wang, Qianqian Xu, and Qingming Huang. 2025. ABKD: Pursuing a proper allocation of the probability mass in knowledge distillation via α-β-divergence. In *International Conference on Machine Learning*, pages 65167–65212. PMLR.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association*

for Computational Linguistics (Volume 1: Long Papers), pages 9426–9439, Bangkok, Thailand. Association for Computational Linguistics.

Taiqiang Wu, Chaofan Tao, Jiahao Wang, Runming Yang, Zhe Zhao, and Ngai Wong. 2024. Rethinking kullback-leibler divergence in knowledge distillation for large language models. *Preprint*, arXiv:2404.02657.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. 2026a. Self-distilled rlvr. *Preprint*, arXiv:2604.03128.

Wenkai Yang, Weijie Liu, Ruobing Xie, Kai Yang, Saiyong Yang, and Yankai Lin. 2026b. Learning beyond teacher: Generalized on-policy distillation with reward extrapolation. *Preprint*, arXiv:2602.12125.

Dongxu Zhang, Zhichao Yang, Sepehr Janghorbani, Jun Han, Andrew Ressler II, Qian Qian, Gregory D Lyng, Sanjit Singh Batra, and Robert E Tillman. 2026a. Fast and effective on-policy distillation from reasoning prefixes. *arXiv preprint arXiv:2602.15260*.

Zhaoyang Zhang, Shuli Jiang, Yantao Shen, Yuting Zhang, Dhananjay Ram, Shuo Yang, Zhuowen Tu, Wei Xia, and Stefano Soatto. 2026b. Reinforcement-aware knowledge distillation for llm reasoning. *Preprint*, arXiv:2602.22495.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. 2026. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*.

A Full Derivation of OPD and Descent Utility

A.1 Deriving the Token-Level OPD Gradient

We derive the token-level estimator used in Section 2. For $\boldsymbol{x} \sim \mathcal{D}$, let $\boldsymbol{y} = (y_1, \ldots, y_T) \sim \pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})$ and $\boldsymbol{s}_t = (\boldsymbol{x}, \boldsymbol{y}_{<t})$. For compactness, define the token-level gradient $\boldsymbol{g}_t$ and the teacher-relative cost c_t as:

$$\boldsymbol{g}_t := \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(y_t|\boldsymbol{s}_t), \tag{11}$$

$$c_t := \log \pi_{\boldsymbol{\theta}}(y_t|\boldsymbol{s}_t) - \log \pi_{\boldsymbol{\phi}}(y_t|\boldsymbol{s}_t). \tag{12}$$

By autoregressive factorization, the sequence-level differences decompose as:

$$\log \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})}{\pi_{\boldsymbol{\phi}}(\boldsymbol{y}|\boldsymbol{x})} = \sum_{t=1}^{T} c_t, \tag{13}$$

$$\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}) = \sum_{t=1}^{T} \boldsymbol{g}_t. \tag{14}$$

The sequence-level OPD objective can therefore be written as:

$$J_{\mathrm{OPD}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\Big[\sum_{t=1}^{T} c_t\Big], \qquad \boldsymbol{y} \sim \pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x}). \tag{15}$$

Applying the score-function identity directly yields:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} J_{\mathrm{OPD}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\Big[\Big(\sum_{t'=1}^{T} c_{t'}\Big)\Big(\sum_{t=1}^{T} \boldsymbol{g}_t\Big)\Big] \\ + \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\Big[\nabla_{\boldsymbol{\theta}} \sum_{t'=1}^{T} c_{t'}\Big].\end{aligned} \tag{16}$$

Since the teacher is fixed, $\nabla_{\boldsymbol{\theta}} c_{t'} = \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(y_{t'}|\boldsymbol{s}_{t'}) = \boldsymbol{g}_{t'}$. The second term vanishes in expectation because the expected score is zero:

$$\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\Big[\sum_{t'=1}^{T} \boldsymbol{g}_{t'}\Big] = \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})] = \boldsymbol{0}. \tag{17}$$

Thus, the gradient simplifies to the cross-terms:

$$\nabla_{\boldsymbol{\theta}} J_{\mathrm{OPD}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\Big[\sum_{t=1}^{T}\sum_{t'=1}^{T} c_{t'} \boldsymbol{g}_t\Big]. \tag{18}$$

We next remove the past-cost terms. For $t' < t$, $c_{t'}$ is determined before y_t is sampled, so:

$$\begin{aligned}\mathbb{E}[c_{t'}\boldsymbol{g}_t] &= \mathbb{E}\Big[c_{t'}\, \mathbb{E}_{y_t \sim \pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{s}_t)}[\boldsymbol{g}_t]\Big] \\ &= \mathbb{E}\Big[c_{t'} \nabla_{\boldsymbol{\theta}} \sum_{y_t} \pi_{\boldsymbol{\theta}}(y_t|\boldsymbol{s}_t)\Big] = \boldsymbol{0}.\end{aligned} \tag{19}$$

Therefore, only current and future costs remain:

$$\nabla_{\boldsymbol{\theta}} J_{\mathrm{OPD}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\Big[\sum_{t=1}^{T}\Big(\sum_{t'=t}^{T} c_{t'}\Big)\boldsymbol{g}_t\Big]. \tag{20}$$

Introducing the discounted return-to-go $G_t^{\gamma} := \sum_{t'=t}^{T} \gamma^{t'-t} c_{t'}$ for $\gamma \in [0, 1]$, yields the practical estimator:

$$\nabla_{\boldsymbol{\theta}} J_{\mathrm{OPD}}^{\gamma}(\boldsymbol{\theta}) \approx \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\Big[\sum_{t=1}^{T} G_t^{\gamma} \boldsymbol{g}_t\Big]. \tag{21}$$

With $\gamma = 0$, we have $G_t^0 = c_t$, giving the token-local estimator:

$$\nabla_{\boldsymbol{\theta}} J_{\mathrm{OPD}}^{\gamma=0}(\boldsymbol{\theta}) \approx \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\Big[\sum_{t=1}^{T} c_t \boldsymbol{g}_t\Big]. \tag{22}$$

Equivalently, each sampled token receives the immediate local reward:

$$r_t = -c_t = \log \pi_{\boldsymbol{\phi}}(y_t|\boldsymbol{s}_t) - \log \pi_{\boldsymbol{\theta}}(y_t|\boldsymbol{s}_t), \tag{23}$$

rather than a rollout-level return.

By truncating future costs, the $\gamma = 0$ estimator serves as a biased approximation of the sequence-level KL divergence, making it theoretically more rigorous to interpret as a token-level RL process rather than strict distillation. This property, collapsing return to immediate cost, enabled the prefix rollouts to compute their updates on the fly without waiting for subsequent generations, and consequently eliminates the variance caused by the dependency on future positions.

A.2 Local Surrogate Improvement and Descent Utility

Here, we derive the optimization utility of the prefix in Sec 3.2. In a local neighborhood around the parameter $\boldsymbol{\theta} \in \Theta$. Let $\boldsymbol{g} := \nabla_{\boldsymbol{\theta}} J_{\mathrm{OPD}}(\boldsymbol{\theta})$ denote the exact full-horizon gradient. We define $\boldsymbol{M}(\boldsymbol{\theta})$ as the local quadratic form representing the geometric curvature (e.g., the empirical Fisher Information Matrix):

$$\begin{aligned}\boldsymbol{M}(\boldsymbol{\theta}) \approx \mathbb{E}_{\boldsymbol{y}\sim\pi_{\boldsymbol{\theta}}}\Big[\big(\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{y})\big) \\ \times \big(\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{y})\big)^{\top}\Big].\end{aligned} \tag{24}$$

The second-order surrogate improvement of a parameter update step $\boldsymbol{\Delta}$ is given by its Taylor expansion up to the second order:

$$\mathcal{S}_{\boldsymbol{M}}(\boldsymbol{\Delta}) := \boldsymbol{g}^{\top}\boldsymbol{\Delta} - \frac{1}{2}\boldsymbol{\Delta}^{\top}\boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{\Delta}. \tag{25}$$

Suppose we are restricted to updating the parameters along a specific surrogate direction $\boldsymbol{d} \neq \boldsymbol{0}$ with a step size $\eta \geq 0$. The update is $\boldsymbol{\Delta} = \eta \boldsymbol{d}$, and the surrogate improvement becomes a scalar quadratic function of η:

$$\mathcal{S}_{\boldsymbol{M}}(\eta \boldsymbol{d}) = \eta(\boldsymbol{g}^\top \boldsymbol{d}) - \frac{\eta^2}{2}(\boldsymbol{d}^\top \boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{d}). \quad (26)$$

Taking the derivative with respect to η and setting it to zero yields the unconstrained optimal step size. Since we require $\eta \geq 0$ to move strictly in the descent direction, the projected optimal step size is:

$$\eta^* = \frac{[\boldsymbol{g}^\top \boldsymbol{d}]_+}{\boldsymbol{d}^\top \boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{d}}. \quad (27)$$

Substituting η^* back into the surrogate objective gives the maximum utility achievable along direction $\boldsymbol{d}$:

$$\begin{aligned} \mathcal{U}_{\boldsymbol{M}}(\boldsymbol{d}) &:= \max_{\eta \geq 0} \mathcal{S}_{\boldsymbol{M}}(\eta \boldsymbol{d}) \\ &= \frac{[\boldsymbol{g}^\top \boldsymbol{d}]_+ (\boldsymbol{g}^\top \boldsymbol{d})}{\boldsymbol{d}^\top \boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{d}} - \frac{[\boldsymbol{g}^\top \boldsymbol{d}]_+^2}{2\,\boldsymbol{d}^\top \boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{d}} \\ &= \frac{[\boldsymbol{g}^\top \boldsymbol{d}]_+^2}{2\,\boldsymbol{d}^\top \boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{d}}. \end{aligned} \quad (28)$$

If we instead do not restrict the update direction, we can find the globally optimal update $\boldsymbol{\Delta}^*$ by setting the gradient of $\mathcal{S}_{\boldsymbol{M}}(\boldsymbol{\Delta})$ to zero:

$$\begin{aligned} \nabla_{\boldsymbol{\Delta}} \mathcal{S}_{\boldsymbol{M}}(\boldsymbol{\Delta}) = \boldsymbol{g} - \boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{\Delta} = \boldsymbol{0} \\ \implies \boldsymbol{\Delta}^* = \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}, \end{aligned} \quad (29)$$

which recovers exactly the natural gradient. Plugging $\boldsymbol{\Delta}^*$ into the improvement function yields the absolute maximum local utility:

$$\begin{aligned} \mathcal{U}_{\boldsymbol{M}}^* &= \boldsymbol{g}^\top (\boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}) \\ &\quad - \frac{1}{2}(\boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g})^\top \boldsymbol{M}(\boldsymbol{\theta})(\boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}) \\ &= \frac{1}{2}\boldsymbol{g}^\top \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}. \end{aligned} \quad (30)$$

To evaluate the quality of the surrogate direction $\boldsymbol{d}$, we calculate its relative utility ratio compared to the optimal natural gradient:

$$\mathcal{E}_{\boldsymbol{M}}(\boldsymbol{d}) := \frac{\mathcal{U}_{\boldsymbol{M}}(\boldsymbol{d})}{\mathcal{U}_{\boldsymbol{M}}^*} = \frac{[\boldsymbol{g}^\top \boldsymbol{d}]_+^2}{\left(\boldsymbol{g}^\top \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}\right)\left(\boldsymbol{d}^\top \boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{d}\right)}. \quad (31)$$

By defining the $\boldsymbol{M}$-weighted inner product $\langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\boldsymbol{M}} = \boldsymbol{u}^\top \boldsymbol{M}\boldsymbol{v}$ and its induced norm $\|\boldsymbol{u}\|_{\boldsymbol{M}} = \sqrt{\boldsymbol{u}^\top \boldsymbol{M}\boldsymbol{u}}$, the cosine similarity between the candidate direction $\boldsymbol{d}$ and the optimal natural update $\boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}$ is:

$$\begin{aligned} &\cos_{\boldsymbol{M}}\left(\boldsymbol{d}, \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}\right) \\ =& \frac{\boldsymbol{d}^\top \boldsymbol{M}(\boldsymbol{\theta})\left(\boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}\right)}{\|\boldsymbol{d}\|_{\boldsymbol{M}} \|\boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}\|_{\boldsymbol{M}}} \\ =& \frac{\boldsymbol{d}^\top \boldsymbol{g}}{\sqrt{\boldsymbol{d}^\top \boldsymbol{M}(\boldsymbol{\theta})\boldsymbol{d}}\sqrt{\boldsymbol{g}^\top \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}}}. \end{aligned} \quad (32)$$

Squaring this expression exactly recovers the relative utility ratio. Thus, for any descent direction ($\boldsymbol{g}^\top \boldsymbol{d} \geq 0$), we mathematically establish that:

$$\mathcal{E}_{\boldsymbol{M}}(\boldsymbol{d}) = \cos^2_{\boldsymbol{M}}\left(\boldsymbol{d}, \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}\right). \quad (33)$$

The derivation above demonstrates that the relative descent utility of any proxy gradient is strictly governed by its cosine alignment with the natural gradient. Given a prefix gradient $\boldsymbol{d}_p$ and a full-rollout gradient $\boldsymbol{d}_f$, we can compare their informational value:

$$\frac{\mathcal{E}_{\boldsymbol{M}}(\boldsymbol{d}_p)}{\mathcal{E}_{\boldsymbol{M}}(\boldsymbol{d}_f)} = \left(\frac{\cos_{\boldsymbol{M}}\left(\boldsymbol{d}_p, \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}\right)}{\cos_{\boldsymbol{M}}\left(\boldsymbol{d}_f, \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}\right)}\right)^2. \quad (34)$$

In the practical regime where we take empirical full-rollout gradient as proxy, for the exact target update ($\boldsymbol{d}_f \approx \boldsymbol{M}(\boldsymbol{\theta})^{-1}\boldsymbol{g}$), the relative utility of truncating a rollout reduces to:

$$\mathcal{E}_{\boldsymbol{M}}(\boldsymbol{d}_p) \approx \cos^2_{\boldsymbol{M}}(\boldsymbol{d}_p, \boldsymbol{d}_f). \quad (35)$$

B Implementation Details

B.1 Training Configurations

We detailed our training hyperparameters in Table 4, Table 5 and Table 6. For all experiments within the OPD family, we train the single-task setting for 30 optimization steps, whereas the multi-task and strong-to-weak settings are trained for 50 steps. For offline KD, these numbers are adjusted to 240 and 400 steps, respectively.

B.2 Implementation details of Baselines

Following (Zhang et al., 2026a), we implement Fast OPD using the same linear window scheduling, which starts at 256 tokens and extends by an additional 256 tokens per step. For P-Align (Liu et al., 2026), we strictly follow the original self-evaluation procedure and the binary-search-based sufficiency criterion. Since the original framework focused exclusively on mathematical reasoning, we

Hyper-parameter	Value
Train Batch Size	128
Micro Batch Size	128
Rollouts n	8
Maximum Prompt Length	2,048
Maximum Response Length	8,192
Sampling Temperature	1.0
Top-p	1.0
Learning Rate	1e-6
Optimization Steps	500
Clip Coefficient ϵ_{high}	0.28

Table 4: Implementation details for GRPO training. The same configuration is used for both math and code RL.

Hyper-parameter	Value
Train Batch Size	1024
Rollouts n	1
Maximum Prompt Length	2,048
Maximum Response Length	8,192
Sampling Temperature	1.0
Top-p	1.0
Learning Rate	3e-6

Table 5: Implementation details for OPD-family training

adapt and extend the code block separation signs to operate at the logical line level. Furthermore, we observed that instruction-following capability becomes a bottleneck when prompting the student to continue generation using the teacher's prefix as a user-side input draft, which consistently degrades performance. To mitigate this and ensure data quality, we instead prefill the teacher-generated segment as a direct prefix during student generation. We directly adopted standard setups for all other baselines.

C Additional Analysis Results

C.1 Loss Distribution on Prefix

We consider the token level loss additionally as for reference, illustrated in Figure 6, where we observed a rather uniform distribution as in cumulative way.

Hyper-parameter	Value
Train Batch Size	128
Maximum Sequence Length	10240
Learning Rate	1e-5
Warm-up Ratio	0.03
LR scheduler	Cosine

Table 6: Implementation details for Offline KD family.

C.2 The Cascading Effect on Suffix Loss

We further conducted an analysis experiment, where we generate the full rollout, but applied a loss mask on its suffix and monitor the training process. Our empirical findings in Figure 7 reveal a compelling cascading effect: optimizing the prefix implicitly yet effectively drives down the suffix loss. While this confirms the prefix's outsized influence on the trajectory, computing the full sequence merely to mask it defeats the computational purpose of truncation. This underscores the necessity of our proposed approach: an online, training-dynamic-aware mechanism that can adaptively identify the effective prefix boundary in practice, without relying on expensive full-rollout references.

D Additional Results

D.1 Additional ablations

On the admissibility threshold. We use $\rho^* = \sqrt{2}/2$ as the default admissibility threshold in ADWIN, and provide additional sensitivity results in Table 7 to show how varying this threshold affects the accuracy–efficiency trade-off.

On probe batch size. We ablate the probe batch size under the setting of Figure 5, doubling it from 64 to 128; since 64 is already small relative to the training batch size of 1024, further reducing it is of limited practical interest. As shown in Table 9, the cosine means shift by at most 0.005 and no window crosses ρ^*, while the standard deviations shrink by a factor close to $1/\sqrt{2} \approx 0.707$, as expected from averaging twice as many samples under the i.i.d. assumption, indicating the probe estimator is already well concentrated at the choice of 64.

On probe staleness. We evaluate staleness sensitivity in the Qwen3-4B setting by reconstructing 32 probe batches generated $s \in \{0, 3, 5, 10\}$ steps earlier at evaluation checkpoints $t \in \{5, 10, 15, 30\}$,

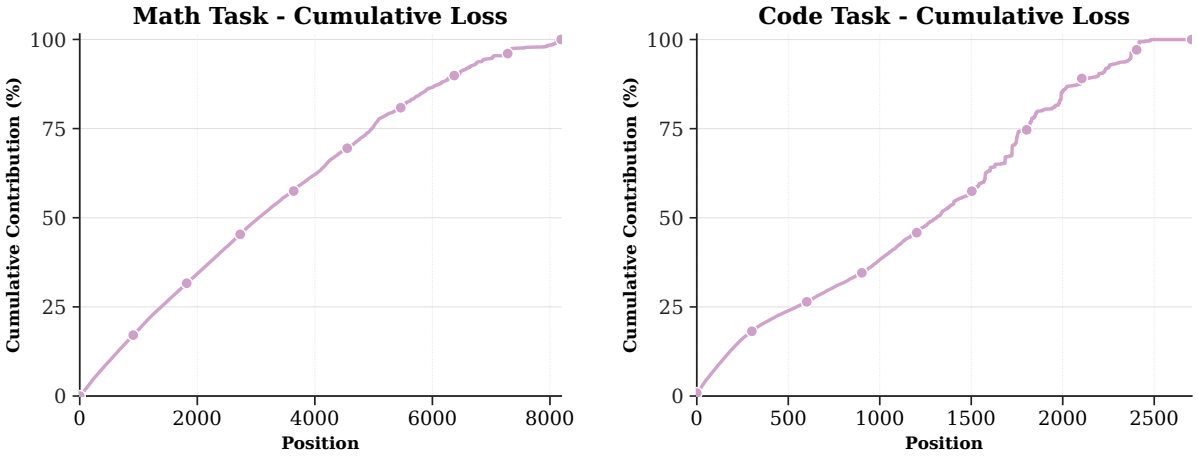

Figure 6: Cumulative distribution of token-level OPD loss over response positions on Qwen3-1.7B setting. The x-axis denotes the token position, and the y-axis reports the cumulative percentage of total loss accumulated up to that position. Results are shown separately for math and code tasks, illustrating loss distributed along student-generated rollouts are rather uniform than concentrated.

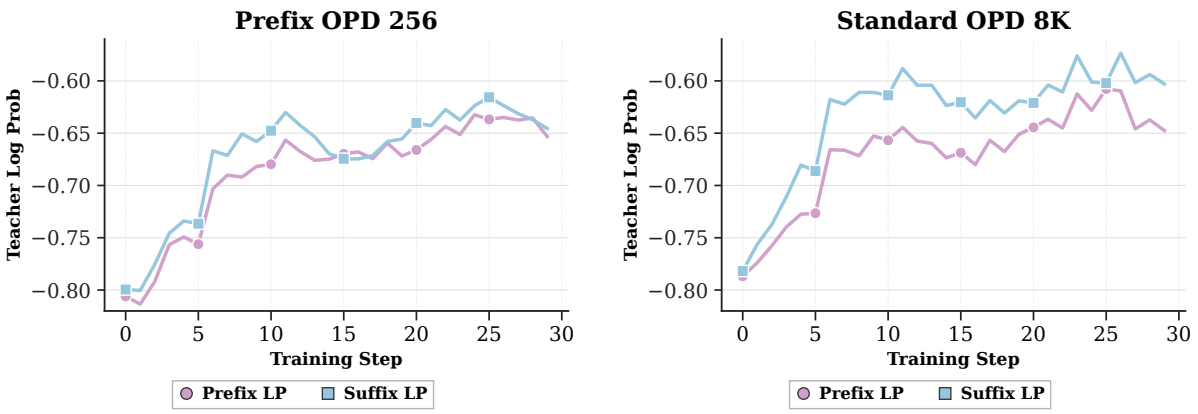

Figure 7: Prefix and suffix teacher log probability dynamics on Qwen3-1.7B setting. We compare Prefix OPD with a 256-token training window and standard full-rollout OPD with an 8K-token horizon. For each training step, teacher log probability is measured separately on the prefix and suffix regions of the generated response, showing that prefix optimization can also improve the downstream suffix region with a considerable extent.

Method	Mathematics			Code			Avg.	EFLOPs ↓
	AIME'24 avg@16	AIME'25 avg@16	Beyond AIME avg@8	HumanEval+ avg@4	MBPP+ avg@4	LCB v6 avg@4		
Initial Model	13.3	7.9	4.5	57.1	40.2	13.1	22.7	–
Teacher	76.0	61.9	44.8	92.6	79.4	40.9	65.9	–
ADWIN $\rho = 0.5$	32.3	24.8	**13.9**	68.7	65.9	**29.1**	39.1	**0.6**
ADWIN $\rho = 0.6$	32.5	25.4	**13.9**	69.3	66.1	27.4	39.1	0.8
ADWIN $\rho = \rho^*$	**34.2**	26.9	13.4	**72.4**	67.1	27.7	**40.3**	1.2
ADWIN $\rho = 0.8$	32.1	**29.8**	11.4	70.6	**68.8**	**29.1**	**40.3**	1.5

Table 7: Sensitivity analysis of the admissibility threshold ρ in ADWIN across the strong-to-weak setting. Lower thresholds reduce compute but may accept insufficiently aligned prefixes, while stricter thresholds increase the training horizon without yielding consistent accuracy gains. The default threshold ρ^* lies near this transition region, attaining the highest average accuracy among the evaluated choices (less compute than 0.8).

Method	Mathematics			Code			Avg.	EFLOPs ↓
	AIME'24 avg@16	AIME'25 avg@16	Beyond AIME avg@8	HumanEval+ avg@4	MBPP+ avg@4	LCB v6 avg@4		
Prefix OPD-128	22.5	15.2	8.0	68.3	61.0	21.5	32.8	**0.4**
Prefix OPD-256	31.0	22.1	10.8	67.6	59.6	21.3	35.4	0.5
Prefix OPD-512	31.7	26.0	**13.9**	**75.2**	65.6	23.3	39.3	0.7
Prefix OPD-1024	34.1	24.0	13.8	74.8	66.8	26.9	40.1	1.0
Prefix OPD-2048	**34.4**	24.8	12.7	72.2	64.0	21.4	38.3	1.5
ADWIN (ρ^*)	34.2	**26.9**	13.4	72.4	**67.1**	**27.7**	**40.3**	1.2

Table 8: Detailed fixed window comparison results in ADWIN under strong-to-weak setting.

Window	Batch 64		Batch 128		Std. Ratio
	Mean	Std.	Mean	Std.	
64	0.384	0.064	0.380	0.047	0.73
128	0.517	0.050	0.512	0.034	0.68
256	0.685	0.035	0.681	0.023	0.67
512	0.855	0.020	0.853	0.014	0.70
1024	0.962	0.007	0.962	0.005	0.69
2048	0.996	0.001	0.996	0.001	0.82

Table 9: Effect of the probe batch size on prefix–full gradient cosine statistics, measured under the setting of Figure 5. Ratio is the standard deviation of batch 128 over that of batch 64.

and report the fraction whose admissibility decision differs from the zero-staleness evaluation, where $s = 0$ serves as a consistency check. In Table 10, the flip rates remain around 10% for $s = 3$ and $s = 5$, while $s = 10$ raises the average to 18.8%, but rarely reached in practice since probes typically conclude before this forced synchronization. These results support 5 as a balanced choice for asynchronous efficiency against the risk from staleness.

On window candidates. The geometric window set follows the exponential-backoff design widely adopted in computer systems, keeping the number of comparisons against the full-gradient reference at the $\mathcal{O}(\log L_{\max})$ level. We also enabled the rest of the windows ranging from 1–8192 outside of 64–2048, but none were triggered in our several reruns empirically.

D.2 Compute-Aligned Comparison

Main results evaluate methods at convergence under their original training recipes, reflecting achievable distillation quality. To decouple accuracy from compute, we also match all baselines to ADWIN's 1.2-EFLOP budget by shortening their training schedules while preserving recipe ratios.

Table 11 reports the resulting performance. ADWIN remains best overall, exceeding the strongest compute-matched baseline by 4.8 points in the Single-Task setting and 6.2 points in the Strong-to-Weak setting. The convergence-based and compute-aligned comparisons provide complementary views of achievable quality and performance under a common compute envelope.

E Instruction Templates

The instructions we used for both training and evaluation are detailed below.

Prompt Template for Math Reasoning:

User Message

```
<|im_start|>user
{question}
Please reason step by step, and put your
final answer within \boxed{}.<|im_end|>
<|im_start|>assistant
```

Assistant Message

```
{response}
```

Prompt Template for Code Generation:

User Message

```
<|im_start|>user
{question}
Write Python code to solve the problem.
Present the code in
```python
Your code
```
at the end.
You need to think first then write the
Python code.<|im_end|>
<|im_start|>assistant
```

Assistant Message

```
{response}
```

Staleness	Evaluation Checkpoint				Avg.
	Step 5	Step 10	Step 15	Step 30	
0	0/32 (0.0%)	0/32 (0.0%)	0/32 (0.0%)	0/32 (0.0%)	0.0%
3	2/32 (6.2%)	2/32 (6.2%)	4/32 (12.5%)	4/32 (12.5%)	9.4%
5	2/32 (6.2%)	6/32 (18.8%)	6/32 (18.8%)	2/32 (6.2%)	12.5%
10	–	8/32 (25.0%)	5/32 (15.6%)	5/32 (15.6%)	18.8%

Table 10: Sensitivity of probe staleness in the Qwen3-4B setting. Each cell reports the number and fraction of the 32 reconstructed probe batches whose selected window differs from the zero-staleness evaluation at the same checkpoint. Staleness 10 is unavailable at step 5, as no probe can be launched that many steps earlier.

Method	Mathematics			Code			Avg.
	AIME'24 avg@16	AIME'25 avg@16	Beyond AIME avg@8	HumanEval+ avg@4	MBPP+ avg@4	LCB v6 avg@4	
(a) Single-Task							
OPD	57.3	55.6	31.6	84.4	65.2	35.0	54.9
Fast OPD	61.0	52.3	31.9	79.6	76.4	35.4	56.1
SeqKD	59.2	52.7	29.6	77.5	73.2	34.3	54.4
P-Align	50.4	47.9	28.5	73.9	61.3	27.1	48.2
ADWIN	**65.6**	**62.9**	**34.9**	**85.9**	**77.4**	**38.6**	**60.9**
(b) Strong-to-Weak							
OPD	26.7	22.7	10.4	65.9	61.2	17.9	34.1
Fast OPD	30.4	24.4	11.1	58.1	57.4	14.5	32.7
SeqKD	16.6	15.8	5.5	55.7	54.8	11.4	26.6
P-Align	21.9	18.5	7.1	57.1	53.2	13.6	28.6
ADWIN	**34.2**	**26.9**	**13.4**	**72.4**	**67.1**	**27.7**	**40.3**

Table 11: Compute-aligned results under a common training budget of 1.2 EFLOPs. We align each baseline with ADWIN's end-to-end compute budget by proportionally scaling down its training schedule while maintaining its original recipe ratios.

F Qualitative Cases of Student Drift

We provide qualitative cases here where student context drift is severe, and the teacher's supervision provides noise rather than reliable guidance. Green denotes a positive teacher feedback on that token while red denotes penalize, with the color intensity reflecting the magnitude of the feedback value.

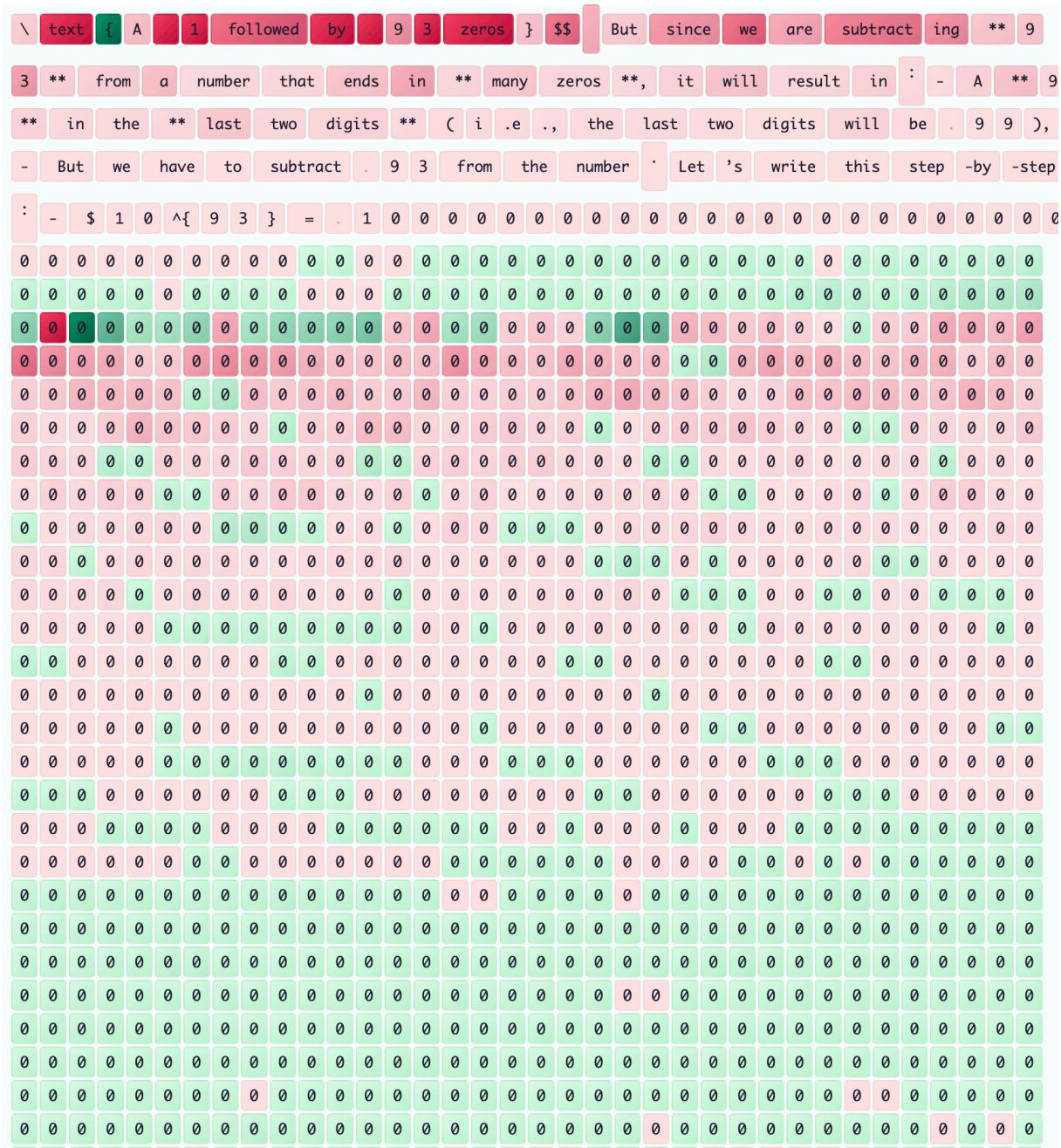

Figure 8: Qualitative case of student context drift into a repetitive loop on mathematical reasoning. The student model enters a meaningless repetition sequence, where teacher distribution allocates a significant proportion of positive reward signals (denoted in green) over these degenerate tokens.

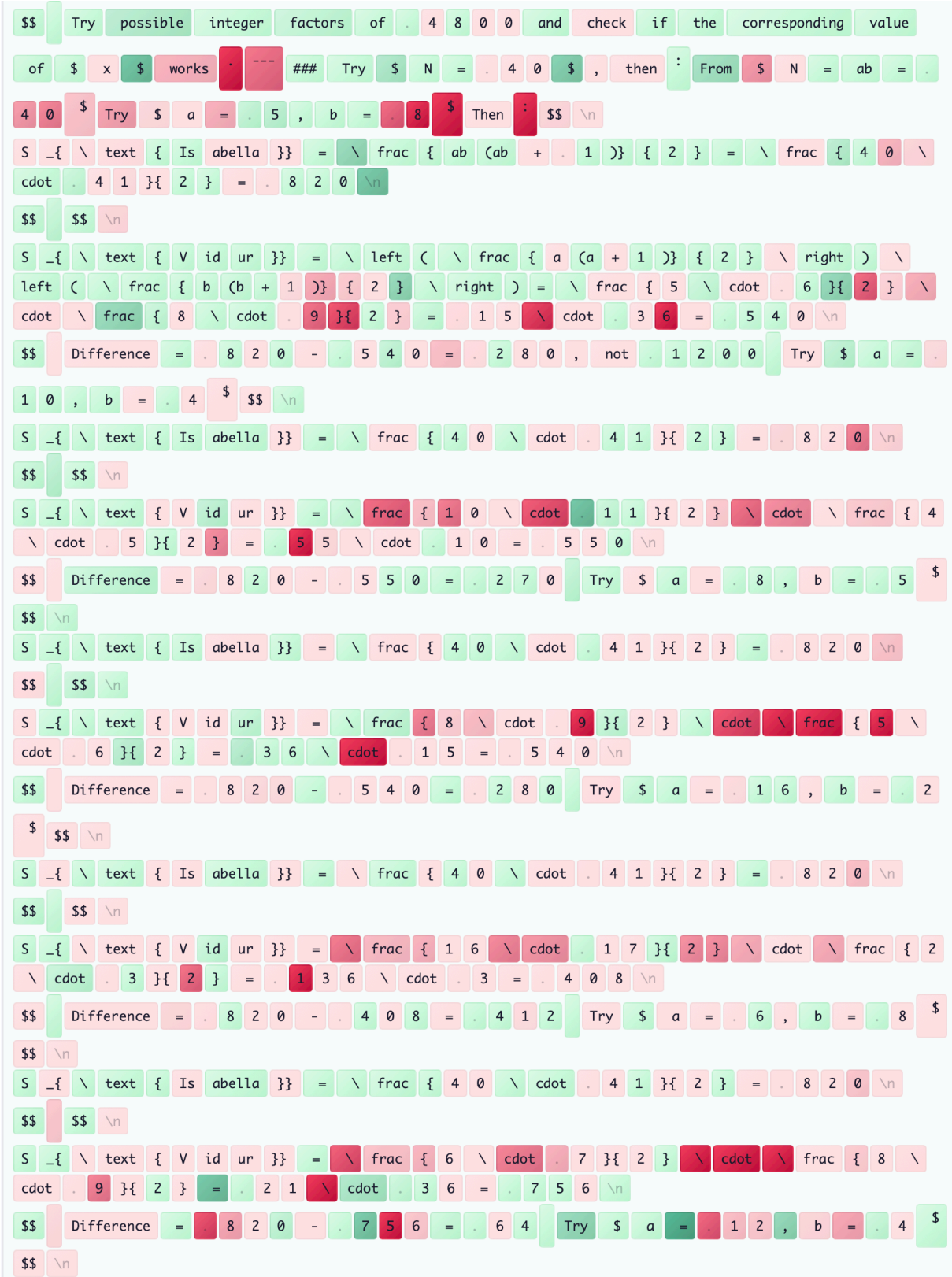

Figure 9: The student model enters a trial-and-error loop of evaluating different candidate integer factors (a and b), where the teacher provides a non-trivial proportion positive feedback (denoted in green).