RANDOMIZED HORIZONS WITH CENTERED HISTORY FOR ON-POLICY DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Teacher-logit on-policy distillation learns from a student's own continuations, making response length a joint budget for generation, teacher supervision, and student differentiation. A fixed short horizon saves work but removes late supervision from the training target. We study an alternative that retains the mean of a specified full-horizon semi-gradient while evaluating response tails intermittently. Centered-history OPD combines a continuation decision shared across the batch with a learned prediction of the omitted tail gradient. A weighted loss produces the student update in one reverse traversal, and a contrast between continuation outcomes updates the prediction without computing a separate tail gradient. We characterize the additional covariance, the role of causal centering, and the distinction between expected retained work and runtime. On MATH, the observed mean correctness is 61.662% versus 60.333% for plain OPD with a 0.6B student, and 75.410% versus 74.623% with a 1.7B student. These comparisons use all three available centered-history runs and both available plain-OPD runs at each scale. The results motivate randomized supervision allocation as a practical design direction for on-policy distillation.

1 INTRODUCTION

On-policy distillation trains a student language model at its own generated prefixes, using a stronger teacher's next-token distributions (Agarwal et al., 2023; Gu et al., 2023). For mathematical reasoning, early choices determine which intermediate states receive supervision. Each additional response position requires generation, teacher evaluation, and student differentiation.

Response length determines which reasoning states are supervised and how long generation and learning resources remain occupied. Fixed, progressive, and adaptive horizons reduce work by changing the available supervision (Zhang et al., 2026; Liang et al., 2026; Tan et al., 2026). We ask whether late supervision can be evaluated intermittently while retaining its contribution to a full-horizon mean update.

Randomized truncation samples a tail with probability p and weights its gradient by $1/p$ (Beatson & Adams, 2019). Independent response truncation may leave almost every batch with a long-running request. Sharing the stop decision shortens the batch's critical path but couples the added noise across responses. Scheduling and estimation must therefore be designed together.

Figure 1 summarizes the problem, approach, analysis, and empirical findings. Our approach uses a common prefix boundary and a single continuation decision shared across the batch. An incoming history vector predicts the aggregate tail gradient. On a short branch, the history supplies the missing contribution; on a continued branch, the exact tail is upweighted and the history contribution is corrected. The predictor's error determines the additional covariance. The remaining challenge is learning that predictor cheaply: a separate tail gradient would require another reverse traversal through the student's prefix computation.

We resolve this challenge by learning history from the gradient already used for training. A contrast between the two continuation outcomes has expectation equal to the tail-prediction error. Subtracting a predictable center changes the noise of this history update while preserving its mean. We use the previous bias-corrected Adam first moment (Kingma & Ba, 2015) as the center, keeping both the history and the center fixed until the current estimator has been formed. This yields *centered-history*

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.

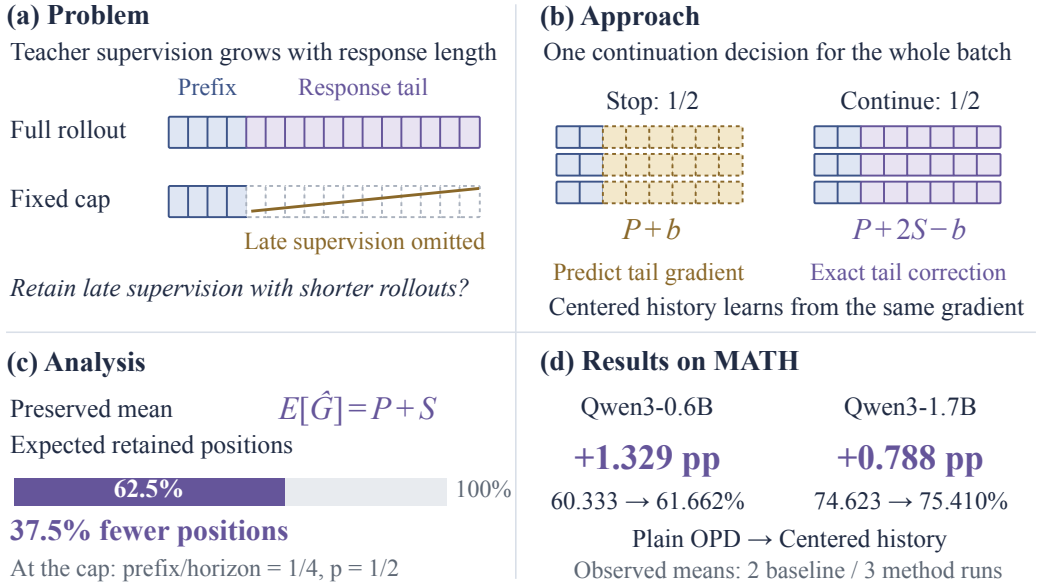

Figure 1: Overview of the work. (a) Shortening rollouts omits late teacher supervision. (b) Shared random continuation and learned tail history preserve its mean contribution, using one student VJP per response; branches show an active batch at $p = 1/2$. (c) The mean is preserved before clipping, with 62.5% expected retained positions when responses reach the cap. (d) Observed MATH mean gains. The analytical position budget and empirical accuracy results describe separate quantities.

OPD, an implementation that combines batch-level stopping, one weighted student reverse traversal, and causal learning of the tail prediction.

We evaluate Qwen3-0.6B and Qwen3-1.7B students with a frozen Qwen3-8B teacher on MATH. Across the available runs, centered history improves observed mean correctness over plain OPD by 1.329 and 0.788 percentage points. Our contributions are the combination of shared continuation and learned tail compensation, an analysis of history-learning noise, and an evaluation at two student scales.

2 Related Work

On-Policy Distillation. Knowledge distillation transfers predictive distributions to a smaller model (Hinton et al., 2015); sequence-level distillation trains on teacher-generated outputs (Kim & Rush, 2016). Learning from the states induced by the learner's own predictions addresses distribution shift in sequential tasks (Ross et al., 2011). GKD applies this principle to distillation on self-generated sequences (Agarwal et al., 2023), and MiniLLM develops on-policy distillation with reverse-divergence objectives (Gu et al., 2023). Our objective sums conditional reverse KL over the vocabulary at sampled student prefixes. Token-level control variates reduce action-sampling noise (Oh et al., 2026); our coin controls whether a later block of prefixes is evaluated.

Rollout Horizons. TOPD and POPD investigate truncated and progressive rollouts (Zhang et al., 2026). ADWIN selects windows using gradient evidence (Liang et al., 2026), Adaptive FastOPD expands horizons using progress and utilization signals (Tan et al., 2026), and Prefix-Guided OPD allocates continuations using prefix information (Zhao et al., 2026). Centered history fixes a reference horizon and compensates for random tail inclusion to preserve its mean semi-gradient. This target is useful when late supervision merits retention; shorter targets can be preferable otherwise.

Gradient Estimation. ARTBP compensates for random truncation in recurrent computation (Tallec & Ollivier, 2017); randomized telescoping develops unbiased gradients and cost–variance allocation for loops and limits (Beatson & Adams, 2019). RELAX learns control variates through

estimator-variance objectives (Grathwohl et al., 2017). In centered-history OPD, a parameter-space predictor compensates for an omitted response tail, and a shared continuation contrast trains it from the available weighted gradient.

SVRG combines gradients of the same example at current and reference parameters with a full reference gradient (Johnson & Zhang, 2013). Its control targets sampling noise over a fixed finite sum. In OPD, the prefix distribution also changes with the student. Our history predicts an aggregate gradient contribution, rather than caching per-example losses or teacher distributions. A continuation contrast trains it without a reference pass over the training pool.

Asynchronous Distillation. AsyncOPD studies stale distillation targets under asynchronous execution (Kang et al., 2026), complementing the choice of response positions to supervise. Our experiments compare centered history with plain full-rollout OPD.

3 METHOD

3.1 REFERENCE GRADIENT

Let θ_t denote the incoming student parameters and q the frozen teacher. For a visited prefix s, the training loss is

$$f_\theta(s) = \mathrm{KL}(p_\theta(\cdot \mid s) \,\|\, q(\cdot \mid s)) \,. \tag{1}$$

Consider a global batch of B responses and a reference response horizon L. A potential full batch ω contains the responses generated with the incoming actor fixed, using a specified sampling and continuation procedure. If response i ends after N_i positions, define

$$F(\omega) = \frac{1}{BL} \sum_{i=1}^{B} \sum_{j=1}^{\min(N_i, L)} \nabla_\theta f_\theta(s_{ij})|_{\theta=\theta_t} \,. \tag{2}$$

The sampled prefixes and teacher targets are held fixed during differentiation. Thus F is the conditional-loss *semi-gradient*, excluding derivatives through the distribution that generated the prefixes. The reference includes the position predicting a genuine end-of-sequence token and excludes later positions.

Choose a prefix cut $\ell < L$ and write $F = P + S$, where P sums positions up to ℓ and S sums subsequent positions. Both use the same divisor BL. This fixed normalization is essential: normalizing a short branch by its observed length would change the relative weight of the prefix and produce a different target. A length cap terminates computation without adding an artificial end-of-sequence training target. For a batch reaching the cut at $\ell/L = 1/4$, averaging a short branch over $B\ell$ would multiply its prefix contribution by four. Tail reweighting cannot undo that change, which is why both branches retain the reference divisor.

3.2 SHARED CONTINUATION

All responses first reach the common boundary or end naturally. A single coin $C \sim \mathrm{Bernoulli}(p)$ then selects whether the unfinished responses continue. In distributed execution, every rank uses the same coin. Let Z indicate that every response has naturally ended by ℓ, and put $A = 1 - Z$. The event Z is known at the prefix boundary, and $S = 0$ on this event.

The method maintains a vector b_t in student-parameter coordinates, predicting the aggregate tail gradient. With incoming $b = b_t$, the update is

$$\widehat{G} = P + \frac{C}{p} S + A \left(1 - \frac{C}{p}\right) b = F + A \left(\frac{C}{p} - 1\right) (S - b). \tag{3}$$

The first form specifies the computed contributions; the second exposes the prediction residual. On an active batch with $p = 1/2$, the two outcomes are

$$G_0 = P + b, \qquad G_1 = P + 2S - b. \tag{4}$$

Their average is $P + S$. On a known-zero batch, the history correction is bypassed and the update is P.

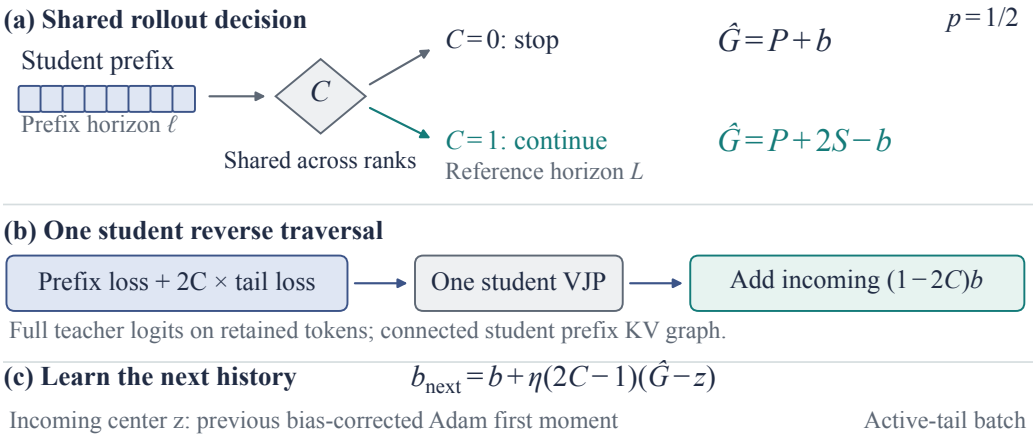

Figure 2: Centered-history OPD. A shared continuation decision controls the response tail. Prefix and retained-tail losses enter one student reverse traversal. The incoming tail history corrects the resulting gradient; a centered coin contrast updates history for the next step. The formulas show an active batch at $p = 1/2$.

Proposition 1 (Preserved mean and residual covariance)**.** *Condition on the incoming parameters, optimizer state, history, and prompt batch. Suppose the continuation coin is independent of the potential continuation and* $0 < p < 1$*. For finite second moments,*

$$\mathbb{E}_C[\widehat{G} \mid \omega] = F(\omega), \tag{5}$$

$$\mathrm{Cov}(\widehat{G}) = \mathrm{Cov}(F) + \frac{1-p}{p}\mathbb{E}\left[A(S-b)(S-b)^\top\right]. \tag{6}$$

No independence assumption between responses is required.

Proof. Given ω, the correction in the second form of Eq. (3) has mean zero because $\mathbb{E}[C/p - 1] = 0$. Its covariance is $A(1-p)(S-b)(S-b)^\top/p$, using $A^2 = A$. Applying total covariance gives Eq. (6). □

The history matters through its residual $S - b$. Predicting the tail well reduces the covariance added by random continuation. The added term is positive semidefinite, so full sampling remains the reference with no continuation noise at the same batch. The benefit being sought is a useful exchange between this noise and avoided computation. Gradient clipping and Adam subsequently apply nonlinear transformations; the preserved mean in Proposition 1 applies before those transformations.

3.3 Gradient Computation

For a continued response, prefix and tail gradients share the computation that produced the prefix states. We retain the student's prefix key–value computation graph, extend it for the tail, and assign loss weight one to prefix positions and $1/p$ to retained tail positions. The resulting local gradient is

$$U = P + \frac{C}{p}S. \tag{7}$$

A single call to automatic differentiation accumulates both adjoints through the shared graph. Detaching the student's prefix cache would remove the tail's derivative through that computation and therefore alter S. The teacher's computation is frozen and requires no reverse traversal.

Local losses use the global divisor BL. Gradients are summed across ranks, corrected once using history, and passed to clipped Adam. Each response requires one VJP, including its connected prefix computation when continued. Figure 2 details the correction and history update.

3.4 History Update

3.4.1 Error Estimation

Learning b by directly averaging S would require access to a separately computed tail gradient. Equation (7) instead returns a weighted combination of prefix and tail. We exploit the randomized branch to obtain a learning signal from this same observation.

Let z be a center fixed before the current coin is revealed. Define

$$D = A\frac{C-p}{1-p}(\widehat{G} - z) - Zb, \qquad b^+ = b + \eta D. \tag{8}$$

At general p, let $G_0 = P + b$ and $G_1 = P + S/p + (1 - 1/p)b$. The two active-batch values of D are $-p(G_0 - z)/(1-p)$ and $G_1 - z$. Their expectation is

$$\mathbb{E}_C[D \mid \omega] = p(G_1 - G_0) = S - b \qquad \text{when } A = 1. \tag{9}$$

On a known-zero batch, $D = -b = S - b$ as well. Consequently, at a fixed incoming model,

$$\mathbb{E}[b^+] = (1-\eta)b + \eta\mathbb{E}[S]. \tag{10}$$

The history tracks the tail's mean with an exponential update in expectation. It is a vector predictor learned from changing batches and parameters, rather than a stored tail that is replayed as though it were current supervision.

At $p = 1/2$, the active update is particularly simple:

$$b^+ = b + \eta(2C-1)(\widehat{G} - z). \tag{11}$$

A continued branch adds the centered observation, and a short branch subtracts it. The opposite signs cancel the prefix contribution in expectation and leave the error in the tail prediction. No additional teacher query or student reverse traversal is needed to form this signal.

3.4.2 Centering

The same incoming b yields the same current $\widehat{G}$ regardless of z. Centering changes the noise of the history update, which affects future predictions. For a fixed active batch, define

$$z_* = (1-p)G_1 + pG_0. \tag{12}$$

The difference between the two outcomes of D is $(z_* - z)/(1-p)$, giving

$$\mathrm{Cov}_C(D \mid \omega) = \frac{p}{1-p}(z_* - z)(z_* - z)^\top. \tag{13}$$

At $p = 1/2$, $z_* = P + S = F$. Thus a predictor of the full gradient can stabilize learning of the tail history even though it never enters the current gradient correction directly.

We use the previous bias-corrected Adam first moment,

$$z_t = \begin{cases} m_t/(1-\beta_1^t), & t > 0, \\ 0, & t = 0, \end{cases} \tag{14}$$

where m_t and t refer to the incoming optimizer state. This predictor is already available to the optimizer and costs one vector rescaling. It summarizes past clipped gradients at earlier parameters, so its utility depends on how much their direction predicts the next full gradient. Crucially, the newly updated moment is not used to center the same step: that would make z depend on the current coin and invalidate Eq. (9).

For an active fixed batch at $p = 1/2$, let $e = S - b$ and $h = F - z$ denote the tail and full-gradient prediction errors. The two updates then have the pathwise forms

$$\widehat{G} = F + (2C-1)e, \qquad D = e + (2C-1)h. \tag{15}$$

The first residual controls the noise injected into the student gradient; the second controls how noisily history learns that residual. The shared coin couples these deviations, so the history update is not an independent observation of the tail. Keeping both predictors fixed until the coin is drawn preserves the mean identities despite this dependence. The trace variance of D changes from $\|F\|^2$ under zero centering to $\|F - z\|^2$. Centering therefore helps exactly when $2F^\top z > \|z\|^2$. Adam's own moment update remains unchanged.

3.4.3 TRACKING DYNAMICS

When every response ends before the cut, the current gradient needs no tail correction and history decays as $(1-\eta)b$. This rule distinguishes the implemented prediction target from the variance-optimal constant under the active gate. Let $\alpha = \Pr(A=1) > 0$ and $\mu = \mathbb{E}[S]$. Then

$$\mathbb{E}[A(S-b)(S-b)^\top] = \alpha \operatorname{Cov}(S \mid A=1) + \alpha(\mu/\alpha - b)(\mu/\alpha - b)^\top. \tag{16}$$

The constant minimizing this expression is μ/α, whereas Eq. (10) tracks μ. The implementation therefore learns the unconditional mean tail. When all batches remain active, the two targets coincide. When early termination is common, this distinction matters in interpreting the quality of the predictor.

The learning rate controls history tracking. On active short and continued branches, the coefficients of the old history are respectively $1 - \eta p/(1-p)$ and $1 - \eta(1-p)/p$. Choosing

$$0 \le \eta \le \min\left\{\frac{p}{1-p}, \frac{1-p}{p}\right\} \tag{17}$$

keeps both coefficients in $[0, 1]$ with the remaining inputs held fixed.

3.5 IMPLEMENTATION

3.5.1 ROLLOUT SCHEDULING

A shared decision requires a common execution boundary. Requests pause at the prefix cut, and termination status is aggregated before unfinished responses either resume or stop. Tokens produced before cancellation still count as work. The numerical rollout and reference-implementation details are given in Appendix A.3.

The scheduling reason for sharing the decision can be seen in an idealized batch where all responses would reach L and decode time scales with the maximum response length. If each of the B responses continues independently with probability p, the expected maximum is

$$\mathbb{E}[L_{\max}^{\text{ind}}] = \ell + (L-\ell)\left[1 - (1-p)^B\right]. \tag{18}$$

A shared decision instead gives $\mathbb{E}[L_{\max}^{\text{shared}}] = \ell + p(L-\ell)$. At $p = 1/2$, at least one independently sampled tail remains with probability $1 - 2^{-B}$, approaching one as the batch grows. Shared continuation gives the batch a short branch with probability one half. This critical-path argument concerns a simplified decoding model; real execution also includes prompt prefill, scheduling, teacher evaluation, and learning.

3.5.2 BATCH CORRELATION

Fix a potential batch whose responses all remain active at the cut. Write $S = \sum_{i=1}^{B} s_i$, where each response's tail gradient s_i includes the divisor BL, and define $r_i = s_i - b/B$. This decomposition introduces no per-response history state. The shared coin gives

$$\Sigma_{\text{shared}} = \frac{1-p}{p}\left(\sum_i r_i\right)\left(\sum_i r_i\right)^\top. \tag{19}$$

Independent inclusion coins applied to the same residuals instead give $\Sigma_{\text{ind}} = (1-p)\sum_i r_i r_i^\top/p$, so

$$\operatorname{tr}(\Sigma_{\text{shared}} - \Sigma_{\text{ind}}) = 2\frac{1-p}{p}\sum_{i<j} r_i^\top r_j. \tag{20}$$

Aligned residuals amplify shared-coin noise; opposing residuals can cancel. The batch size alone therefore does not determine the variance penalty. Residual geometry matters alongside the probability of eliminating all tails from the schedule. Both covariances condition on the same potential batch, while their decoding schedules differ.

An aggregate history matches this structure: the shared estimator depends on $S - b$, even when individual tails remain variable. Predicting their sum can reduce the randomization error without predicting every response separately. Shared stopping creates the short batch; history determines how accurately its omitted contribution is represented.

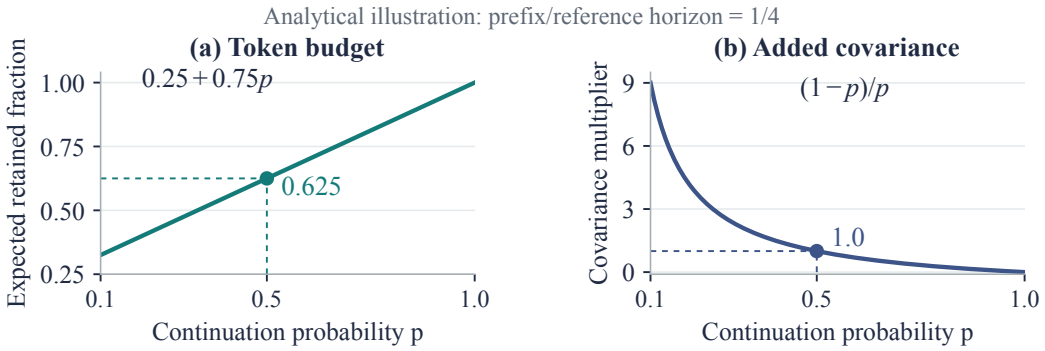

Figure 3: Analytical work–noise tradeoff at $\ell/L = 1/4$. Expected retained response positions follow Eq. (21); the coefficient $(1-p)/p$ multiplies the residual covariance in Eq. (6). The marked setting $p = 1/2$ retains 62.5% of positions when all responses reach the cap. These curves are calculations, not measured runtime or accuracy.

3.5.3 Computational Cost

For a fixed potential batch, let n_P and n_S count the prefix and tail loss positions. Expected retained supervision is $n_P + pn_S$. If every response reaches the cap, the fraction relative to full supervision is

$$r_{\text{positions}} = \frac{\ell + p(L-\ell)}{L}. \tag{21}$$

For illustration, $\ell/L = 1/4$ and $p = 1/2$ give $r_{\text{positions}} = 0.625$: a 37.5% reduction in expected retained response positions in this cap-reaching case. Natural EOS changes n_P and n_S; prompt processing and request overshoot add costs outside this expression.

Equation (6) shows why reducing p indefinitely is undesirable: the residual-covariance coefficient grows as $(1-p)/p$. More accurate history reduces the quantity this coefficient multiplies. Under a fixed incoming model and history, a simplified expected cost $c_P + pc_S$ and a penalty λ on added trace covariance give

$$J(p) = c_P + pc_S + \lambda \frac{1-p}{p} R_b, \qquad R_b = \mathbb{E}[A\|S-b\|^2]. \tag{22}$$

For $c_S > 0$ and $R_b > 0$, its minimizer on $(0, 1]$ is $p_* = \min\{1, \sqrt{\lambda R_b / c_S}\}$. This calculation explains the dependence on tail cost and prediction error; it is not the controller used in our experiments, which hold p fixed. Figure 3 displays the two terms before introducing such an application-specific tradeoff weight.

3.5.4 Optimizer State

Table 1 specifies the order required to preserve causality. The student, optimizer moments, and history are committed together after each step. History learning uses the raw corrected gradient, while Adam receives its clipped version. The incoming history is never replaced by a newly observed tail before the current correction is applied.

With d trainable coordinates, history adds one persistent vector and $O(d)$ storage and arithmetic per step. The center reuses an existing Adam moment. Connected prefix activations are retained on the continued branch, so teacher positions, gradient arithmetic, activation memory, and wall-clock time must be distinguished.

4 Experiments

4.1 Experimental Setup

We evaluate Qwen3-0.6B and Qwen3-1.7B students supervised by a frozen Qwen3-8B teacher (Yang et al., 2025) on MATH (Hendrycks et al., 2021). Plain OPD and centered history use the

Table 1: One centered-history update. t is the incoming optimizer counter. The known-zero rule takes precedence; active batches at $p = 1$ skip history learning.

1	Snapshot θ_t, m_t, v_t, and b_t; form z_t from Eq. (14).
2	Generate prefixes to ℓ or EOS; determine Z across ranks.
3	Draw shared C; continue unfinished responses to L if $C = 1$, otherwise stop.
4	Compute full-teacher KL losses with weights 1 and C/p; take one student VJP per response.
5	Sum local gradients using divisor BL; add $(1 - Z)(1 - C/p)b_t$ to obtain $\widehat{G}$.
6	Compute b_{t+1} from Eq. (8) and the incoming center.
7	Clip $\widehat{G}$, apply Adam, and commit all next-step state.

Table 2: Full-MATH mean per-sample correctness (%). All available runs are shown. A dash marks an unavailable result; means use three centered-history runs or two plain-OPD runs. The gain is the difference of available-run means in percentage points, computed before rounding.

Student	Method	Run 1	Run 2	Run 3	Mean	Gain
0.6B	Plain OPD	60.290	60.375	—	60.333	—
	Centered history	61.835	61.975	61.175	**61.662**	+1.329
1.7B	Plain OPD	74.745	74.500	—	74.623	—
	Centered history	75.745	75.085	75.400	**75.410**	+0.788

same conditional reverse-KL objective, a common reference horizon, and matched training configurations within each student size. The comparison changes how tail supervision is allocated: plain OPD retains full rollouts, while centered history samples tails and compensates for their omission.

Training uses teacher distributions at student-generated prefixes. Gold answers are reserved for evaluation, and development questions are separated from the training and test partitions. Appendix A gives the data preparation, model adaptation, optimization, and decoding settings.

4.2 Evaluation Protocol

The primary metric is mean per-sample correctness on the full MATH test partition. For M questions with K sampled answers each and correctness indicators a_{jk}, the score is

$$\text{Accuracy} = \frac{100}{MK} \sum_{j=1}^{M} \sum_{k=1}^{K} a_{jk}. \tag{23}$$

Both methods use the same answer extraction and mathematical grading rules. This metric measures the correctness of an individual sampled response, rather than whether any answer for a question succeeds. Sampling and grading details are reported in Appendix A.2.

Each student size has three centered-history runs and two plain-OPD runs. Run indices enumerate entries within a method without asserting paired initializations or trajectories. Means use available runs; the missing third baseline is excluded. Question-level confidence intervals require the corresponding response records.

4.3 Results

Table 2 reports all ten available scores. Centered history improves mean correctness from 60.333% to 61.662% for Qwen3-0.6B and from 74.623% to 75.410% for Qwen3-1.7B. Differences computed before rounding are 1.329 and 0.788 percentage points.

Figure 4 shows consistent ordering across the available runs. The lowest centered-history score exceeds the highest plain-OPD score by 0.800 points at 0.6B and 0.340 points at 1.7B. Centered-history ranges are 0.800 and 0.660 points; plain-OPD ranges are 0.085 and 0.245 points. These are descriptive ranges with unequal repetition counts.

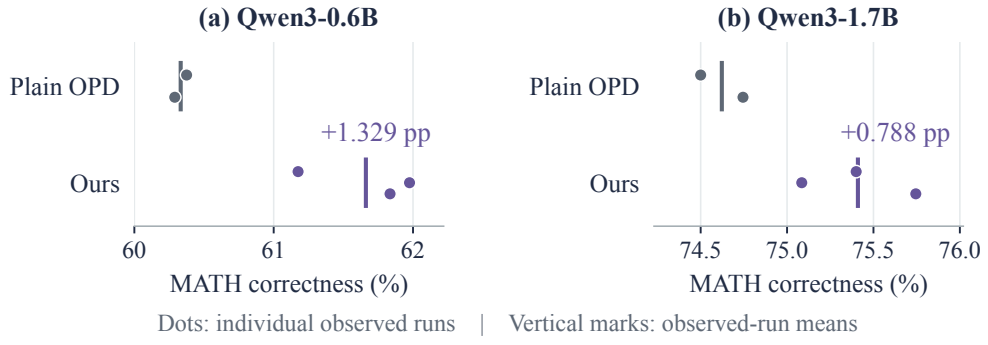

Figure 4: Every available run at each student scale. Individual accuracies expose the observed spread without imposing a pairing between methods. Mean markers summarize the available runs; they are descriptive averages, with no confidence intervals inferred from the aggregate table.

The smaller student shows the larger mean gain. Both scales produce competitive learned models while intermittently omitting tail computation; attributing the gain to individual components requires separate measurements.

5 Discussion

Estimator Properties. The invariant is the conditional mean of the chosen semi-gradient before clipping. Different stochastic steps change parameters, later responses, and clipping events, allowing either higher or lower final accuracy. Proposition 1 characterizes the local target; the MATH gains describe the observed training runs.

Component Analysis. The predictor b enters the current gradient and determines the randomization residual $S - b$. The center z changes how noisily that predictor is learned. Setting $b = 0$ isolates uncompensated randomization, while retaining history with $z = 0$ isolates centering. The evaluation compares the complete method with plain OPD; component-wise measurements remain unavailable. The covariance identities specify the quantities these comparisons should inspect.

History must track both the student's changing prefix distribution and the changing gradient at each prefix. Faster tracking responds sooner to this drift but incorporates more noise from continuation decisions. Centering reduces that learning noise when the incoming moment is predictive, linking the quality of future tail compensation to the recent optimization trajectory.

Computational Efficiency. Retained positions measure analytical work. Runtime also depends on early EOS, teacher batching, prefix-cache memory, synchronization, and optimizer overhead. The available scores have no associated timing measurements, so the 37.5% cap-reaching reduction remains separate from the quality results.

Limitations. The reported comparison covers one benchmark, two student scales, one teacher, and uneven repetition counts. Per-question outputs linked to the reported entries are unavailable in the released summary, so the analysis is descriptive rather than a significance test.

6 Conclusion

Centered-history OPD treats response tails as randomly sampled supervision whose mean contribution can be retained through compensation. Sharing the continuation decision creates a batch-level opportunity to stop; learning the tail history from a centered coin contrast makes that estimator compatible with one student reverse traversal. The resulting method achieves higher observed mean MATH correctness than plain OPD in the reported 0.6B and 1.7B runs.

AI USE STATEMENT

Generative AI tools were used only for writing and language polishing, and for retrieval and discovery of relevant prior work. They were not used to generate research ideas, design methods or experiments, conduct analyses, or draw conclusions. All scientific content was developed and verified by the authors.

REPRODUCIBILITY STATEMENT

The anonymous code package contains the centered-history update, the connected-prefix gradient calculation, an HF reference training and evaluation workflow, configuration files, and numerical tests. Public model and dataset identifiers and acquisition instructions are provided; datasets and model weights are not embedded. Appendix A records the concrete experimental settings. The manuscript reports every available aggregate run. Per-response records associated with these entries would be needed to reproduce their grading and calculate question-level statistical intervals.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-Policy Distillation of Language Models: Learning from Self-Generated Mistakes. *arXiv preprint arXiv:2306.13649*, 2023. URL https://arxiv.org/abs/2306.13649v3.

Alex Beatson and Ryan P. Adams. Efficient optimization of loops and limits with randomized telescoping sums. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 534–543, 2019. URL https://proceedings.mlr.press/v97/beatson19a.html.

Will Grathwohl, Dami Choi, Yuhuai Wu, Geoffrey Roeder, and David Duvenaud. Backpropagation through the Void: Optimizing control variates for black-box gradient estimation. *arXiv preprint arXiv:1711.00123*, 2017. URL https://arxiv.org/abs/1711.00123v3.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: On-Policy Distillation of Large Language Models. *arXiv preprint arXiv:2306.08543*, 2023. URL https://arxiv.org/abs/2306.08543v6.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset. *arXiv preprint arXiv:2103.03874*, 2021. URL https://arxiv.org/abs/2103.03874v2.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. URL https://arxiv.org/abs/1503.02531.

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. *arXiv preprint arXiv:2106.09685*, 2021. URL https://arxiv.org/abs/2106.09685.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, volume 26, 2013. URL https://proceedings.neurips.cc/paper_files/paper/2013/hash/ac1dd209cbcc5e5d1c6e28598e8cbbe8-Abstract.html.

Wonjun Kang, Kevin Galim, Seunghyuk Oh, Minjun Kang, Sanghyun Park, Donghoon Kim, Minjae Lee, Minseo Kim, Rishabh Tiwari, Yuchen Zeng, Hyung Il Koo, and Kangwook Lee. AsyncOPD: How Stale Can On-Policy Distillation Be? *arXiv preprint arXiv:2606.24143*, 2026. URL https://arxiv.org/abs/2606.24143v1.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1317–1327. Association for Computational Linguistics, 2016. doi: 10.18653/v1/D16-1139. URL https://aclanthology.org/D16-1139/.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

Kun Liang, Chenming Tang, Clive Bai, Weijie Liu, Saiyong Yang, and Yunfang Wu. ADWIN: Adaptive Windows for Horizon-Aware On-Policy Distillation. *arXiv preprint arXiv:2605.28396*, 2026. URL https://arxiv.org/abs/2605.28396v1.

Minjae Oh, Sangjun Song, Gyubin Choi, Yunho Choi, and Yohan Jo. KL for a KL: On-Policy Distillation with Control Variate Baseline. *arXiv preprint arXiv:2605.07865*, 2026. URL https://arxiv.org/abs/2605.07865v1.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. 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*, pp. 627–635, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

Corentin Tallec and Yann Ollivier. Unbiasing Truncated Backpropagation Through Time. *arXiv preprint arXiv:1705.08209*, 2017. URL https://arxiv.org/abs/1705.08209v1.

Qian Tan, Huaifei Liang, Xuanyu Zhu, Lei Jiang, and Yuqiang Li. Adaptive FastOPD: Progress-Aware Rollout Horizon Expansion for Efficient On-Policy Distillation. *arXiv preprint arXiv:2607.29494*, 2026. URL https://arxiv.org/abs/2607.29494v1.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 Technical Report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388v1.

Yaocheng Zhang, Jiajun Chai, Yuqian Fu, Songjun Tu, Xiaohan Wang, Wei Lin, Guojun Yin, Qichao Zhang, Yuanheng Zhu, and Dongbin Zhao. Are Full Rollouts Necessary for On-Policy Distillation? *arXiv preprint arXiv:2605.31490*, 2026. URL https://arxiv.org/abs/2605.31490v2.

Qingfei Zhao, Huan Song, Shuyu Tian, Jiawei Shao, and Xuelong Li. Prefix-Guided On-Policy Distillation: Mining Golden Trajectories from Rollouts. *arXiv preprint arXiv:2606.21994*, 2026. URL https://arxiv.org/abs/2606.21994v1.

A Experimental Details

A.1 Models, Training, and Data

We study Qwen3-0.6B and Qwen3-1.7B students supervised by Qwen3-8B (Yang et al., 2025). The implementation freezes the base student and teacher weights and trains rank-16 LoRA adapters (Hu et al., 2021) with scaling $\alpha = 32$ on attention and MLP projections. Student differentiation and serialized optimizer state use FP32; the teacher runs in BF16. The reference optimizer uses learning rate 10^{-4}, Adam coefficients $(0.9, 0.999)$, epsilon 10^{-8}, global norm clipping at 1 with stabilizer 10^{-6}, and zero weight decay. The objective is Eq. (1). Plain OPD retains the full response to the common horizon and uses the same conditional objective.

Training prompts come from the MATH training partition after deduplication and overlap filtering (Hendrycks et al., 2021). A separate 512-question development partition is reserved for development. The prepared training pool has 6,686 questions before tokenizer-dependent prompt-length filtering; the native configuration retains 6,680. Gold answers are used for evaluation, while distillation trains on teacher distributions at student-generated prefixes. The documented training configuration uses 64 warm-up updates followed by 128 method updates, with 32 responses per update. Table 3 lists the common method settings. Within each student size, both methods use the same training configuration. Native and HF rollout implementations retain their respective numerical and sampling configurations.

Table 3: Method and evaluation configuration. Native training uses BF16 sampling with temperature 1 and full support; the HF reference uses FP32 sampling with temperature 0.7, top-p 0.8, and top-k 20. These are distinct execution recipes.

Setting	Value
Students / frozen teacher	Qwen3-0.6B, Qwen3-1.7B / Qwen3-8B
Trainable adapters	LoRA rank 16, $\alpha = 32$; attention and MLP projections
Global responses per update	32; one response per prompt
Prompt cap / reference response horizon	1,024 / 2,048 tokens
Centered-history cut / continuation / learning rate	$\ell = 512$ / $p = 0.5$ / $\eta = 0.05$
Primary evaluation	5,000 MATH questions; four responses per question
Evaluation sampling	Temperature 0.7; top-p 0.8; top-k 20
Evaluation response / context cap	8,192 / 16,384 tokens; thinking disabled

A.2 Evaluation Settings

The primary metric is mean per-sample correctness on the full 5,000-question MATH test partition, with four sampled answers per question. For an actor with correctness indicators a_{jk}, its score is

$$\text{Accuracy} = \frac{100}{5000 \cdot 4} \sum_{j=1}^{5000} \sum_{k=1}^{4} a_{jk}. \tag{24}$$

Evaluation uses native BF16 decoding with Table 3's sampling settings and fixed mathematical grading rules. The metric averages correctness over 20,000 responses, rather than measuring whether any answer per question succeeds.

A.3 Execution and Reference Implementation

Native execution pauses requests at the prefix boundary, communicates termination status, and broadcasts the coin. Unfinished requests resume on a continued branch and are cancelled otherwise. Tokens generated before cancellation still count as work. The HF reference draws the cap before sampling from a separate coin stream, preserving the inclusion rule with a distinct numerical rollout procedure.

The training configuration uses $p = 0.5$ and history learning rate $\eta = 0.05$. Gradients are summed in fixed rank order. History is stored as one FP32 vector over the trainable coordinates, requiring $4d$

bytes for d coordinates. The center reuses the incoming bias-corrected Adam first moment. These storage and numerical choices instantiate the update described in the main text.

The aggregate scores in the paper use native BF16 evaluation. The anonymous code package provides an HF numerical reference with its own rollout recipe. The native and HF recipes are listed separately in Table 3; the reference package does not establish numerical identity with the native pause-and-continue runtime.