β-OPSD: DERIVING WITH POLICY OPTIMIZATION, TRAINING WITH SELF-DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy self-distillation (OPSD) is a promising approach to improve reasoning language models, but it remains brittle in practice and often requires substantial engineering. Direct teacher matching can be unstable when the student–teacher gap is large, yet vanilla OPSD provides no mechanism to control the update strength. We show that vanilla OPSD is the $\beta = 1$ case of a KL-regularized objective, where β controls the trade-off between proximity to a reference policy and privileged teacher guidance. We introduce ***β*-OPSD** and derive its optimal policy as a geometric interpolation between the reference policy and the privileged teacher. Instead of optimizing this objective with costly, high-variance reinforcement learning, we implement its closed-form target through logit interpolation. By scheduling β, we control the target along the geometric path between the reference and the teacher. Because token-local updates ignore how early choices affect future mismatches, we use return-to-go credit assignment to propagate those mismatches backward. Experiments on mathematical and code reasoning benchmarks show that β-OPSD improves overall performance over vanilla OPSD across Qwen3 models from 1.7B to 8B, with gains of 1.66–5.74 percentage points in math avg@12 and 1.52–2.18 percentage points in LiveCodeBench pass@1. Thus, β-OPSD uses policy optimization to derive the target while retaining the efficiency of self-distillation.

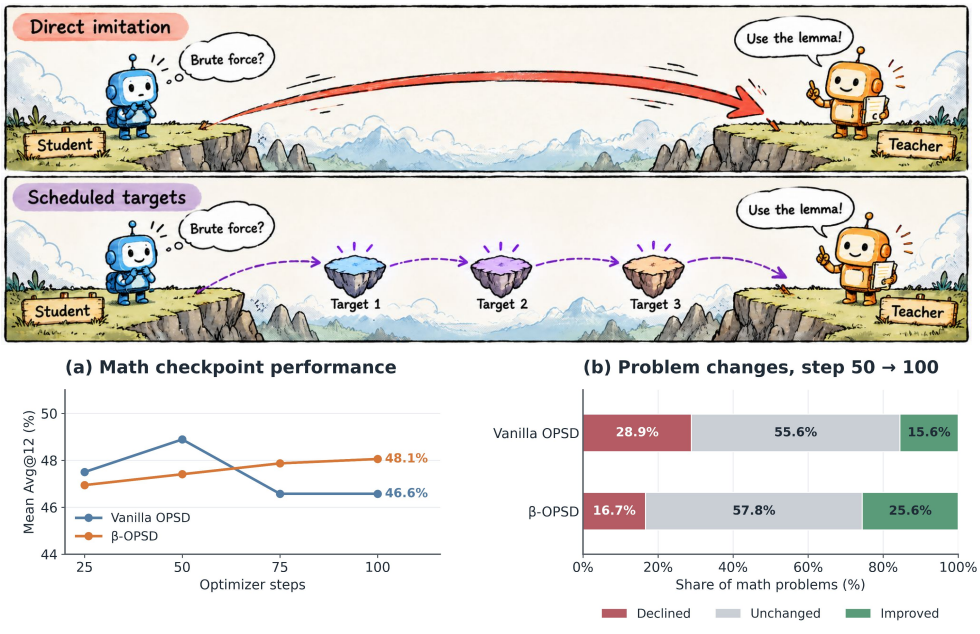


Figure 1: **Bridging the student–teacher gap and stabilizing training.** *Top:* Vanilla OPSD directly imitates a distant privileged teacher, whereas β-OPSD follows scheduled intermediate targets. *Bottom:* On Qwen3-4B math, (a) Vanilla OPSD declines after step 50 while β-OPSD continues improving; (b) more problems improve and fewer problems decline from steps 50 to 100.

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.

1 INTRODUCTION

Reasoning performance of large language models has advanced rapidly through reinforcement learning and self-improvement techniques (Shao et al., 2024; Ouyang et al., 2022; Guo et al., 2025; Huang et al., 2022). Among these approaches, on-policy self-distillation (OPSD) trains a student on its own trajectories while a teacher supplies supervision using privileged information available only during training, such as reference solutions, verified reasoning traces, or external feedback (Zhao et al., 2026; Shenfeld et al., 2026; Hübotter et al., 2026; Li et al., 2026a; Jang et al., 2026; Yang et al., 2026). Unlike offline distillation, OPSD provides supervision directly on the states and reasoning paths encountered by the student during learning. This tight coupling between student exploration and teacher supervision makes OPSD a promising framework for continually improving reasoning capabilities from on-policy experience.

Despite this promise, most existing OPSD methods begin from direct teacher imitation. Given a student-generated trajectory, the privileged teacher provides token-level predictions, and the student minimizes a reverse Kullback–Leibler (KL) divergence to those predictions (Zhao et al., 2026; Jang et al., 2026; Li et al., 2026a; Yang et al., 2026). This formulation fixes the teacher as the unique target and offers no explicit control over how aggressively the student moves away from its current or initial policy. When the teacher is far from the student, this fixed target can force an unnecessarily aggressive update and destabilize training. As a result, OPSD in practice is often brittle and requires substantial engineering to train reliably (Li et al., 2026a; Wang et al., 2026; Zhu et al., 2026).

To address this limitation, we reinterpret OPSD through the lens of KL-regularized reinforcement learning. In this formulation, the policy is encouraged to follow the privileged teacher while remaining close to a reference policy, with β controlling the strength of this regularization. Importantly, conventional direct teacher matching emerges as a special case:

$$\text{Vanilla OPSD} \;\equiv\; \text{the } \beta = 1 \text{ case of KL-regularized policy optimization.}$$

This connection reveals that teacher imitation need not be the only target of OPSD. Instead, varying β defines a family of objectives that explicitly trade off teacher guidance against preservation of the reference policy. Building on this view, we derive the optimal policy of the generalized objective in closed form and show that it is a geometric interpolation between the reference policy and the privileged teacher. At $\beta = 1$, the optimum recovers the teacher policy; as β increases, the target moves progressively toward the reference. This reference–teacher path forms the basis of ***β*-OPSD**, allowing the strength of teacher guidance to be controlled throughout training rather than committing to direct teacher matching.

The RL formulation specifies the target without requiring an RL training loop. In practice, β-OPSD constructs a tractable local approximation by interpolating reference and teacher logits, then trains with on-policy self-distillation. In this way, β-OPSD retains the simplicity and efficiency of OPSD while inheriting the explicit policy-control mechanism provided by the RL formulation. We further incorporate return-to-go credit assignment, allowing earlier tokens to receive feedback based on their effect on the remainder of the response.

We evaluate β-OPSD on mathematical and code reasoning with Qwen3 models from 1.7B to 8B parameters (Yang et al., 2025). Across AIME 2024 (Zhang & Math-AI, 2024), AIME 2025 (Zhang & Math-AI, 2025), and HMMT 2025 (Dekoninck et al., 2026), it improves avg@12 over vanilla OPSD by 1.66–5.74 percentage points, including a 9.16-point gain on AIME 2024 at 1.7B. On LiveCodeBench (Jain et al., 2024), it improves pass@1 by 1.52–2.18 points across model scales. Controlled comparisons separately test the interpolated target and return-to-go credit assignment.

Our contributions are:

- **Theory:** We show that vanilla OPSD is the $\beta = 1$ case of a KL-regularized policy-optimization family and derive its closed-form geometric optimum between the reference policy and privileged teacher.
- **Algorithm:** We turn this optimum into a practical self-distillation method using scheduled logit interpolation and return-to-go credit assignment.
- **Empirics:** We demonstrate improvements over vanilla OPSD across mathematical and code reasoning tasks, model scales, and controlled comparisons.

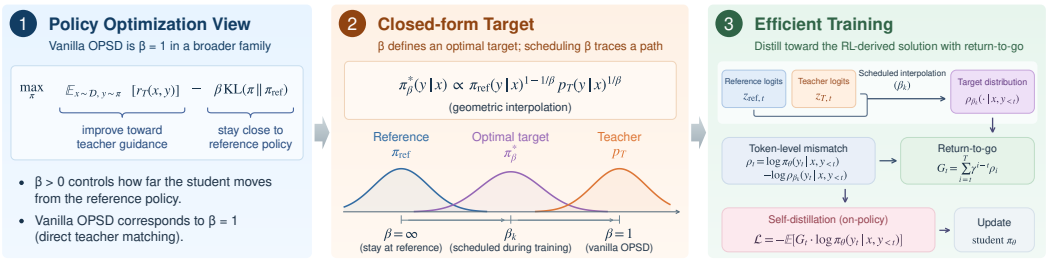


Figure 2: **Overview of β-OPSD.** (1) Vanilla OPSD is the $\beta = 1$ case of a KL-regularized policy-optimization family, where β controls movement away from the reference policy. (2) The closed-form optimum is a geometric interpolation between the reference and privileged teacher; scheduling β traces a path between them. (3) We realize this target through local logit interpolation and train with return-to-go credit assignment.

2 METHOD

We first derive the optimal target and then develop its practical distillation approximation. Figure 2 summarizes this derivation-to-training pipeline. Appendix A summarizes the notation.

2.1 VANILLA OPSD AS $\beta = 1$ POLICY OPTIMIZATION

Let $\pi_\theta(y \mid x)$ denote the student policy, $p_T(y \mid x, c)$ the privileged teacher, and $\pi_{\text{ref}}(y \mid x)$ a reference policy, such as the initial student or a stop-gradient copy of the current student. We start from the standard KL-regularized RL objective

$$\max_\theta \; \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[R(y;x,c)\right] - \beta D_{\text{KL}}\left(\pi_\theta(\cdot \mid x) \,\|\, \pi_{\text{ref}}(\cdot \mid x)\right), \tag{1}$$

where $\beta > 0$ controls how strongly the learned policy is constrained to remain close to the reference.

To embed standard on-policy self-distillation within this policy-optimization family, we choose the reward to be the teacher-to-reference log-ratio:

$$R(y;x,c) = \log \frac{p_T(y \mid x, c)}{\pi_{\text{ref}}(y \mid x)}. \tag{2}$$

This reward is high when a trajectory is more likely under the privileged teacher than under the reference policy. Substituting Eq. 2 into Eq. 1 gives the family

$$\mathcal{J}_\beta(\theta) = \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\log \frac{p_T(y \mid x, c)}{\pi_{\text{ref}}(y \mid x)}\right] - \beta D_{\text{KL}}\left(\pi_\theta(\cdot \mid x) \,\|\, \pi_{\text{ref}}(\cdot \mid x)\right). \tag{3}$$

We call this objective β-*OPSD*.

Proposition 2.1 (OPSD is the $\beta = 1$ case)**.** *When $\beta = 1$, maximizing Eq. 3 is equivalent to minimizing the standard sequence-level reverse KL,*

$$D_{\text{KL}}\left(\pi_\theta(\cdot \mid x) \,\|\, p_T(\cdot \mid x, c)\right). \tag{4}$$

At $\beta = 1$, the reference terms cancel, recovering standard reverse-KL OPSD. For $\beta > 1$, the remaining reference penalty produces more conservative updates. The proof is given in Appendix B.1.

2.2 VARIABLE β DEFINES A PATH OF OPTIMAL TARGETS

For fixed β, the β-OPSD objective has a closed-form optimum.

Proposition 2.2 (Optimal policy of β-OPSD)**.** *Assuming compatible support, the maximizer of Eq. 3 is*

$$\pi^\star_\beta(y \mid x, c) = \frac{\pi_{\text{ref}}(y \mid x)^{1-\frac{1}{\beta}} p_T(y \mid x, c)^{\frac{1}{\beta}}}{Z_\beta(x, c)}, \tag{5}$$

where $Z_\beta(x, c)$ normalizes over full trajectories.

The proof is in Appendix B.2. The solution geometrically interpolates between the reference and teacher. For $\beta \geq 1$, the teacher weight $1/\beta$ lies in $[0, 1]$: $\beta = 1$ recovers the teacher, while $\beta \to \infty$ recovers the reference.

Local logit realization. Although Eq. 5 gives the optimal trajectory-level distribution, it is not directly usable for language models because the normalizer $Z_\beta(x, c)$ ranges over all possible completions. We therefore approximate the optimal policy locally at each decoding prefix.

Let

$$h_t = (x, y_{<t}) \tag{6}$$

be the prefix at token t. Let $z_{\text{ref}}(\cdot \mid h_t)$ denote the reference or student-side logits, and let $z_T(\cdot \mid h_t, c)$ denote the privileged teacher logits. For a fixed $\beta \geq 1$, we define the local logit-interpolant target as

$$\tilde{p}_\beta(\cdot \mid h_t, c) = \text{softmax}\left(\left(1 - \frac{1}{\beta}\right) z_{\text{ref}}(\cdot \mid h_t) + \frac{1}{\beta} z_T(\cdot \mid h_t, c)\right). \tag{7}$$

At each prefix, this distribution is exactly the normalized geometric interpolation of the local reference and teacher distributions: multiplying probabilities geometrically is equivalent, up to normalization, to linearly interpolating their logits.

The resulting autoregressive interpolant target is

$$\tilde{p}_\beta(y \mid x, c) = \prod_{t=1}^{T} \tilde{p}_\beta(y_t \mid x, y_{<t}, c). \tag{8}$$

Because normalization is applied separately at each prefix, the resulting autoregressive distribution is a tractable local approximation to the sequence-level optimum in Eq. 5. This is the target used by the practical distillation objective below.

Remark 2.3. *The trajectory-level optimum in Eq. 5 does not in general factorize into the locally normalized interpolants in Eq. 7, because the exact conditional additionally depends on future continuation mass. We characterize this approximation and provide KL error bounds in Appendix B.4.*

Scheduling the interpolation. The construction above defines a path of local targets indexed by a fixed β. We now specify how training traverses this path. Directly matching the privileged teacher from the start can create a substantial student–teacher distribution gap, destabilizing training and degrading performance (Wang et al., 2026; Zhu et al., 2026; Li et al., 2026a). The geometric target family provides a natural curriculum for bridging this gap. By choosing a sequence of teacher weights $w_k \in [0, 1]$, we can control how strongly teacher supervision is injected throughout training, rather than forcing every update toward the teacher endpoint. A natural choice is to gradually increase the teacher weight, yielding progressively stronger teacher guidance as training proceeds. However, the formulation itself does not require monotonicity: fixed or decreasing schedules are also valid instances of the same target-control framework (Section E.3).

Concretely, let $w_k = 1/\beta_k \in [0, 1]$ denote the teacher weight at training step k, where $w_k = 0$ gives the reference endpoint and $w_k = 1$ gives the teacher endpoint. We use the bounded linear schedule

$$w_k = w_{\text{start}} + (w_{\text{end}} - w_{\text{start}}) \frac{k}{K - 1}, \qquad k = 0, \ldots, K - 1. \tag{9}$$

When $w_{\text{start}} < w_{\text{end}}$, the target transitions progressively from a more reference-like distribution to a more teacher-like one over training. At step k, we set $\beta_k = 1/w_k$ and use the local and autoregressive targets in Eqs. 7–8 with $\beta = \beta_k$.

2.3 From Policy Optimization Back to Practical Distillation

At training step k, we train the student to match the tractable interpolant $\tilde{p}_{\beta_k}$ obtained from Eq. 8 with schedule β_k. Policy optimization determines the target distribution, while distillation provides an efficient training method.

Because early tokens affect later prefixes, the sequence-level score-function gradient weights each token by its future log-ratio mismatches. For a sampled trajectory, define the token mismatch

$$\rho_t^{\beta_k} = \log \pi_\theta(y_t \mid h_t) - \log \tilde{p}_{\beta_k}(y_t \mid h_t, c). \tag{10}$$

and its discounted return-to-go

$$G_{t,\gamma}^{\beta_k} = \sum_{s=t}^{T} \gamma^{s-t} \rho_s^{\beta_k}, \qquad \gamma \in [0, 1]. \tag{11}$$

The resulting practical loss is

$$\mathcal{L}_\beta(\theta; y) = \frac{1}{T} \sum_{t=1}^{T} \mathrm{sg}\left(G_{t,\gamma}^{\beta_k}\right) \log \pi_\theta(y_t \mid h_t), \tag{12}$$

where $\mathrm{sg}(\cdot)$ stops gradients through the interpolant and return weights. The return-to-go construction is not merely heuristic: when $\gamma = 1$, its unnormalized form is an unbiased estimator of the sequence-level reverse-KL gradient, and normalization by a deterministic horizon preserves its direction. The practical loss uses the sampled-length factor $1/T$ and may use $\gamma < 1$ to control long-horizon magnitude. Appendix B.3 gives the full derivation and conditions. Relative to vanilla OPSD (Zhao et al., 2026), β-OPSD therefore changes both the target and the credit assignment: it replaces the fixed teacher with the scheduled RL-derived interpolant and local token weights with future-aware return-to-go weights.

3 Related Work

On-policy distillation and self-distillation. On-policy distillation trains the student on its own generated trajectories, using dense teacher feedback at each token. This avoids the exposure-bias mismatch of purely offline distillation, but introduces a new challenge: the teacher must provide reliable supervision on student-generated prefixes, which may be outside the teacher's typical distribution (Li et al., 2026a; Jang et al., 2026). Recent self-distillation methods use the same model as the teacher but provide it with privileged information, such as ground-truth solutions or feedback (Zhao et al., 2026; Liu et al., 2026; Yang et al., 2026; Shen et al., 2026; Zhang et al., 2026; Hübotter et al., 2026; He et al., 2026). β-OPSD belongs to this line of work, but replaces the fixed teacher target with a scheduled student–teacher interpolant and by correcting the myopic local token-KL gradient with a return-to-go estimator.

KL-regularized policy optimization and credit assignment. KL-regularized RL and RLHF optimize reward while constraining the policy toward a reference model (Ziegler et al., 2019; Ouyang et al., 2022; Schulman et al., 2017). Prior work also studies stable local token-level distillation signals (Li et al., 2026a; Armandpour et al., 2026; Shen et al., 2026; Jang et al., 2026; Kim et al., 2026), unbiased sequence-level reverse-KL gradients (Agrawal et al., 2026; Liu et al., 2026), and future-aware credit assignment in policy optimization (Williams, 1992; Schulman et al., 2015; Sutton et al., 1998). Our formulation connects these strands by deriving the target from reference-regularized policy optimization and using return-to-go to estimate its sequence-level gradient.

Interpolation, curricula, and teacher-guided learning. Recent work has explored intermediate behaviors or targets to reduce the student–teacher distribution gap. TRB (Plyusov et al., 2026) modifies the *rollout distribution* by blending student and teacher behavior during warmup, while keeping the OPD objective itself unchanged. Veto (Jang et al., 2026) and TOP-D (Xie et al., 2026) use product-of-experts and arithmetic-mixture targets, respectively. The concurrent method DemoPSD (Li et al., 2026b) derives a reverse-KL barycenter between student and teacher distributions and adapts the interpolation strength according to their disagreement; related work derives reward-tilted policies for KL-regularized learning with privileged information (Penaloza et al., 2026). Unlike heuristic behavior mixing or stabilization targets, our scheduled geometric interpolant is the closed-form optimizer induced by the generalized OPSD objective.

4 Experiments

We organize the experiments around three questions: **RQ1:** Does β-OPSD improve mathematical and code reasoning across model scales and compare favorably with concurrent OPD methods under a matched protocol? **RQ2:** Does β-OPSD retain the computational efficiency of distillation? **RQ3:** Which components produce these gains, and how sensitive are they to implementation choices?

4.1 Experimental Setup

Models and datasets. We evaluate β-OPSD on the instruct-tuned Qwen3-1.7B, Qwen3-4B, and Qwen3-8B models. For math, we train on the mathematical-reasoning subset of OpenThoughts (Guha et al., 2025); for code, we train on the 20,974-example DeepCoder dataset, including the Verified TACO (Li, 2024) and PrimeIntellect SYNTHETIC-1 (Mattern et al., 2025) subsets. In both domains, each training example contains a problem x and a ground-truth solution, which serves as privileged information c for the teacher branch. The student conditions only on x, whereas the privileged teacher conditions on both x and c during training.

For evaluation, we use AIME 2024, AIME 2025, and HMMT February 2025 for mathematical reasoning, and LiveCodeBench (Jain et al., 2024) for code reasoning.

Baselines. We compare against the base model without post-training, supervised fine-tuning (SFT) on ground-truth solutions, vanilla on-policy self-distillation (OPSD), and GRPO with outcome-level rewards. Vanilla OPSD uses a local token-KL objective to distill directly from the privileged teacher, without scheduled interpolant targets or return-to-go credit assignment. Our main tables implement this baseline with the sampled-token reverse-KL objective used in our training framework. Appendix D additionally compares this baseline with a full-vocabulary forward-KL objective. All main comparisons use the step-100 checkpoint.

Implementation details. We use the linear schedule in Eq. 9 with $w_{\text{start}} = 0.5$ and $w_{\text{end}} = 0.8$. We construct the target using a stop-gradient copy of the current student and train with the return-to-go loss using $\gamma = 0.99$. All methods use LoRA (Hu et al., 2022). Shared optimization settings, method-specific rollout limits, and further training and evaluation details are provided in Appendix C.

4.2 Results on Math and Code Reasoning

Answer to RQ 1. β-OPSD achieves the best aggregate performance across mathematical and code reasoning tasks at all three model sizes and the strongest mathematical-reasoning performance among the concurrent methods under a matched protocol.

Mathematical reasoning. Figure 3 reports avg@12 at the step-100 checkpoint. β-OPSD achieves the highest average score among all evaluated methods at each model scale, reaching 36.76, 57.87, and 60.18 for Qwen3-1.7B, Qwen3-4B, and Qwen3-8B, respectively. Relative to Vanilla OPSD, these correspond to gains of 5.74, 1.76, and 1.66 percentage points. The largest gain occurs at 1.7B, where β-OPSD improves all three benchmarks, including a 9.16-point gain on AIME 2024. Improvements are less uniform at larger scales: Qwen3-4B loses 1.11 points on AIME 2024 and Qwen3-8B loses 1.12 points on HMMT 2025 relative to Vanilla OPSD, but gains on the other benchmarks yield higher overall averages. Full results are reported in Table 4.

Code reasoning. Figure 4 reports pass@1 on LiveCodeBench. β-OPSD outperforms all evaluated baselines in overall pass@1 at every model scale, achieving 17.25%, 27.30%, and 26.26% for Qwen3-1.7B, Qwen3-4B, and Qwen3-8B, respectively. These results improve on Vanilla OPSD, the strongest overall baseline at each scale, by 1.52, 1.90, and 2.18 percentage points. The difficulty breakdown shows consistent gains on easy and hard problems across all three scales. In particular, Qwen3-8B improves on hard problems from 4.00% to 8.57%. Qwen3-4B improves across all difficulty levels, whereas medium-problem performance drops by 0.78 and 0.52 points for Qwen3-1.7B and Qwen3-8B, respectively, relative to Vanilla OPSD. Thus, the overall gains do not imply uniform improvements across difficulty levels. Full results are reported in Table 5.

Comparison with concurrent methods. Section 3 describes the distinctions among these methods. For a controlled comparison, we re-implement TRB, Veto, and TOP-D, which are general OPD methods originally evaluated with different models and tasks, in our privileged-information self-distillation setting using the same model, data, optimization budget, and evaluation protocol as β-OPSD. For each baseline, we follow the best-performing configuration reported in its original paper whenever applicable. As shown in Table 1, β-OPSD performs best on all three math benchmarks and achieves the strongest aggregate result.

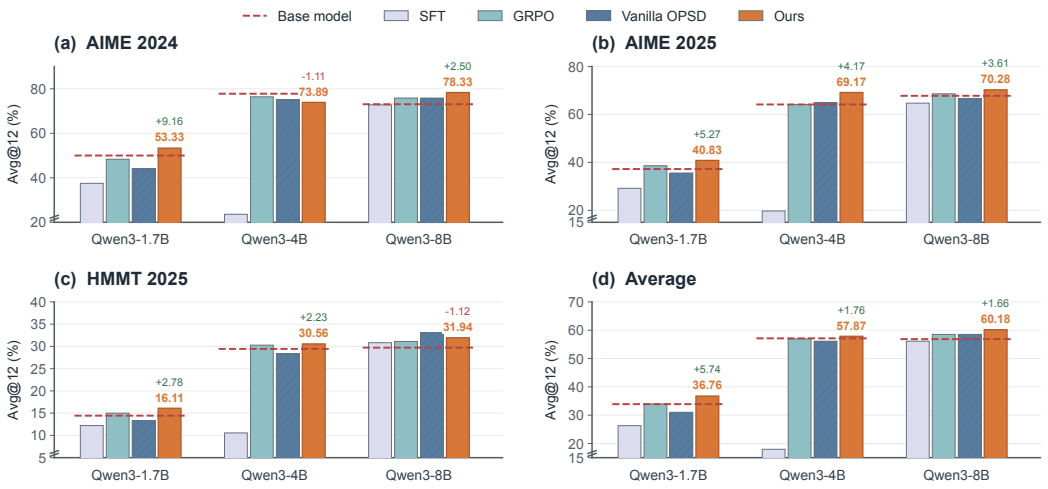


Figure 3: **Mathematical reasoning performance at step 100.** We report avg@12 (%). Gains are percentage-point differences from Vanilla OPSD at the same model scale. β-OPSD uses $w_{\text{start}} = 0.5$ and $w_{\text{end}} = 0.8$.

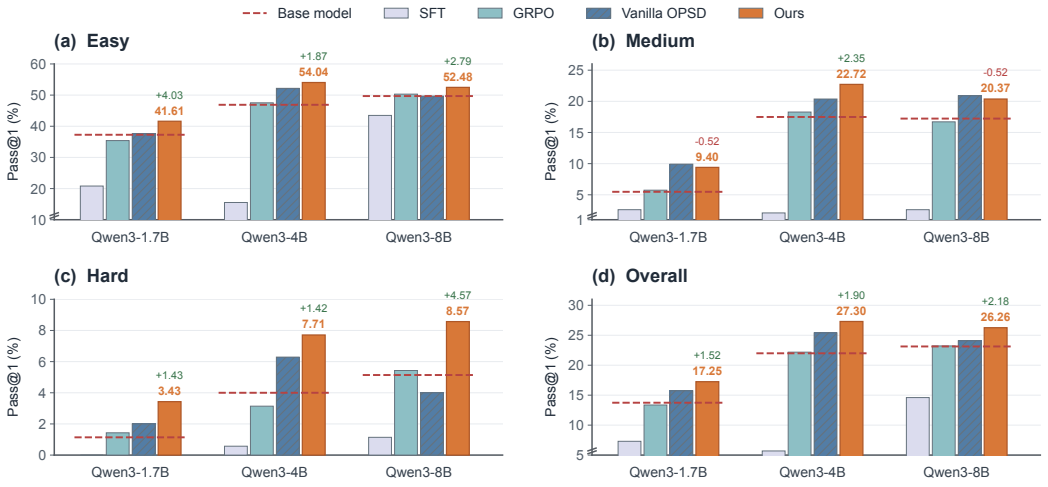


Figure 4: **LiveCodeBench performance at step 100.** We report pass@1 (%) by difficulty and overall. Gains are percentage-point differences from Vanilla OPSD at the same model scale. β-OPSD uses $w_{\text{start}} = 0.5$ and $w_{\text{end}} = 0.8$.

Table 1: **Comparison with concurrent methods on Qwen3-1.7B math avg@12 (%) under the shared training and evaluation protocol.** Bold indicates the best result in each column.

Method	AIME 2024	AIME 2025	HMMT 2025
Vanilla OPSD (Zhao et al., 2026)	44.17	35.56	13.33
TRB (Plyusov et al., 2026)	47.50	34.17	15.28
TOP-D (Xie et al., 2026)	49.44	32.78	13.06
Veto (Jang et al., 2026)	49.17	34.44	14.44
β-OPSD (ours)	**53.33**	**40.83**	**16.11**

4.3 Training Efficiency

Answer to RQ 2. β-OPSD achieves both the highest step-100 pass@1 and the largest improvement per GPU-hour, showing that its accuracy gains do not sacrifice distillation efficiency.

We compare the training efficiency of SFT, GRPO, vanilla OPSD, and β-OPSD using Qwen3-4B on coding tasks. Each method is trained for 100 optimizer steps with an effective batch size of 32 on a single H200 GPU. For comparison, both OPSD variants use a 1,024-token rollout limit; β-OPSD linearly increases the teacher target weight from 0.5 to 0.8. We evaluate checkpoints on LiveCodeBench v6 using pass@1. Equal optimizer-step budgets do not imply equal generated-token or compute budgets.

Figure 5 compares performance against training steps and GPU-hours, together with the improvement over the untrained model per GPU-hour:

$$\text{Efficiency} = \frac{\text{pass@1}_{\text{trained}} - \text{pass@1}_{\text{base}}}{\text{training GPU-hours}}.$$

At step 100, β-OPSD achieves pass@1 of 27.20 in 2.74 GPU-hours, compared with 25.12 in 3.02 GPU-hours for vanilla OPSD and 23.32 in 2.29 GPU-hours for GRPO. Relative to the base score of 21.99, β-OPSD yields 1.90 percentage points of improvement per GPU-hour, versus 1.04 for vanilla OPSD and 0.58 for GRPO. Thus, under these configurations, β-OPSD provides approximately $1.84\times$ and $3.29\times$ greater improvement per GPU-hour, respectively.

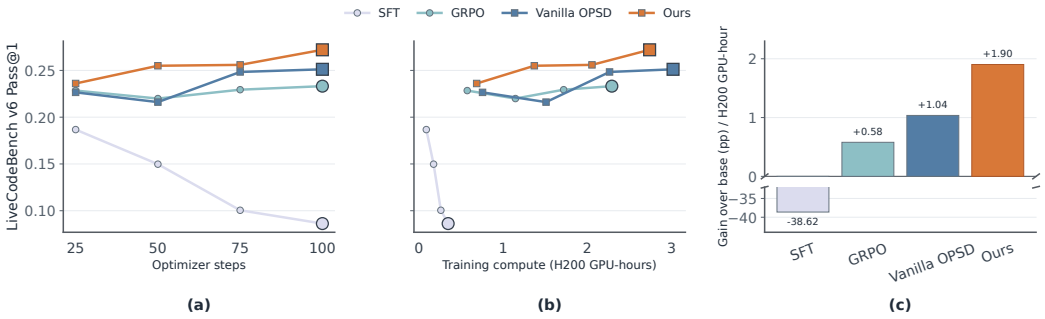


Figure 5: **Qwen3-4B LiveCodeBench efficiency.** (a) pass@1 versus optimizer steps, (b) pass@1 versus H200 GPU-hours, and (c) pass@1 improvement over the base model per H200 GPU-hour. Checkpoint compute in (b) is estimated proportionally from total training time. The enlarged marker on each curve denotes its final checkpoint. The broken axis in (c) displays SFT's negative improvement alongside the positive efficiencies.

We next isolate the contributions of the individual components.

4.4 Ablation Studies

Answer to RQ 3. Interpolating between the dynamic student and the teacher along the geometric path consistently improves performance, while RTG improves most math settings but has model-dependent effects on code.

Unless otherwise specified, all ablations use the same student, privileged teacher, data, optimization settings, training budget, and evaluation protocol as the main experiments; we vary only the component under study.

4.4.1 Effect of Logit Interpolants

We isolate the target's contribution by comparing *Teacher + Return-to-Go*, which uses the fixed privileged teacher p_T, with β-OPSD, which uses the scheduled interpolant $\tilde{p}_{\beta_k}$. Both use student on-policy trajectories and the same return-to-go loss; the teacher-only variant simply replaces $\tilde{p}_{\beta_k}$ with p_T in Eq. 10, retaining the weighting in Eq. 11.

Table 2: **Effect of the distillation target on Qwen3-1.7B math avg@12 (%).** Both variants use the same return-to-go estimator; β-OPSD uses $w_{\text{start}} = 0.5$ and $w_{\text{end}} = 0.8$. Bold indicates the best result in each column.

Method	AIME 2024	AIME 2025	HMMT 2025
Teacher + Return-to-Go	47.30	35.53	14.44
β-OPSD target	**53.33**	**40.83**	**16.11**

As shown in Table 2, the interpolant improves avg@12 by 6.03, 5.30, and 1.67 points on AIME 2024, AIME 2025, and HMMT 2025, respectively. These gains suggest that a smooth target transition reduces the student–target gap and stabilizes supervision compared with direct teacher matching.

4.4.2 Additional Ablations

Credit assignment. We compare training with and without RTG while keeping the interpolant target and all other settings fixed. Enabling RTG improves seven of nine math settings, supporting future-aware credit assignment. Its effect on code is model-dependent (Appendix E.1).

Interpolant endpoints. We compare a fixed student with a fixed teacher, a dynamic student with a dynamic teacher, and a dynamic student with a fixed teacher. The dynamic-student, fixed-teacher construction performs best on all three math benchmarks. This supports tracking the evolving student while retaining a stable teacher endpoint (Appendix E.2).

Teacher-weight schedule. We evaluate increasing, decreasing, and fixed teacher weights along the geometric reference–teacher path. Every schedule improves over Vanilla OPSD. Thus, controlling the learning target along the geometric path is more important than following any specific monotonic curriculum (Appendix E.3).

Guided rollout distribution. We additionally sample trajectories from student–teacher mixtures with per-decision importance correction. These variants obtain comparable gains but require dual-model decoding and greater sampling complexity. This trade-off supports student on-policy generation as the practical default (Appendix F).

5 Conclusion

We presented β-OPSD, which derives a reference-to-teacher target path from KL-regularized policy optimization and trains toward it using logit interpolation and return-to-go credit assignment. Experiments on competition-level mathematical reasoning and code benchmarks show higher aggregate performance than vanilla OPSD, SFT, and GRPO in the evaluated settings. In controlled ablations, the interpolant target improves all three reported math benchmarks, while return-to-go improves most math settings but has model-dependent effects on code. These results support smooth target paths as a useful principle for on-policy distillation and indicate that the benefit of trajectory-level credit assignment depends on the task and model scale. Exploring adaptive or theoretically optimized interpolation schedules, lower-variance return-to-go estimators, and applications beyond mathematical reasoning are promising directions for future work.

AI USE STATEMENT

We used generative AI tools to assist with drafting and editing manuscript text, LaTeX formatting, coding, and running experiments. We take responsibility for the final content of this work, including its text, code, claims, results, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies self-distillation for language-model mathematical reasoning using existing models, a mathematical reasoning training dataset, and competition benchmarks. The experiments assess reasoning performance rather than safety, fairness, or suitability for deployment. Improvements on these benchmarks should not be interpreted as evidence that the resulting models are free from biased, incorrect, or harmful outputs. The distillation procedure may preserve limitations of the underlying models and training data, and stronger reasoning capabilities may also support misuse. Any downstream use should therefore include application-specific safety evaluation, appropriate human oversight, and respect for the usage terms of the underlying models and datasets.

REPRODUCIBILITY STATEMENT

Section 2 specifies the policy-optimization formulation, interpolated distillation targets, and practical training objective. Appendix B provides the theoretical derivations and proofs, including their assumptions. Section 4.1 identifies the models, training data, evaluation benchmarks, and baselines. Appendix C.1 documents the shared training configuration, optimization settings, parameter-efficient tuning, generation settings, and computational budget, while Appendix C.2 describes evaluation decoding, scoring metrics, and checkpoint selection. These details are included in the same document to support reproduction and interpretation of the results.

REFERENCES

Rishabh Agrawal, Jacob Fein-Ashley, and Paria Rashidinejad. Reinforcement learning from rich feedback with distributional dagger. *arXiv preprint arXiv:2606.05152*, 2026. 5

Mohammadreza Armandpour, Fatih Ilhan, David Harrison, Ajay Jaiswal, Duc NM Hoang, Fartash Faghri, Yizhe Zhang, Minsik Cho, and Mehrdad Farajtabar. Unmasking on-policy distillation: Where it helps, where it hurts, and why. *arXiv preprint arXiv:2605.10889*, 2026. 5

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. Beyond benchmarks: Matharena as an evaluation platform for mathematics with llms. *arXiv preprint arXiv:2605.00674*, 2026. URL https://arxiv.org/abs/2605.00674. 2

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, et al. Openthoughts: Data recipes for reasoning models. *arXiv preprint arXiv:2506.04178*, 2025. 6

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. 2

Yinghui He, Simran Kaur, Adithya Bhaskar, Yongjin Yang, Jiarui Liu, Narutatsu Ri, Liam Fowl, Abhishek Panigrahi, Danqi Chen, and Sanjeev Arora. Self-distillation zero: Self-revision turns binary rewards into dense supervision. *arXiv preprint arXiv:2604.12002*, 2026. 5

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022. 6

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve, 2022. URL https://arxiv.org/abs/2210.11610. 2

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, et al. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026. 2, 5

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024. 2, 6

Ijun Jang, Jewon Yeom, Juan Yeo, Hyunggyu Lim, and Taesup Kim. Stable on-policy distillation through adaptive target reformulation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 42217–42227, 2026. 2, 5, 7

Jeonghye Kim, Xufang Luo, Minbeom Kim, Sangmook Lee, Dohyung Kim, Jiwon Jeon, Dongsheng Li, and Yuqing Yang. Why does self-distillation (sometimes) degrade the reasoning capability of llms? *arXiv preprint arXiv:2603.24472*, 2026. 5

Kaixin Li. Verified taco problems. https://huggingface.co/datasets/likaixin/TACO-verified, 2024. URL https://huggingface.co/datasets/likaixin/TACO-verified. 6, 23

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, et al. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026a. 2, 4, 5

Yunhe Li, Hao Shi, Wenhao Liu, Mengzhe Ruan, Hanxu Hou, Zhongxiang Dai, Shuang Qiu, and Linqi Song. Demopsd: Disagreement-modulated policy self-distillation, 2026b. URL https://arxiv.org/abs/2607.02502. 5

Yifeng Liu, Shiyuan Zhang, Yifan Zhang, and Quanquan Gu. Self-distilled policy gradient. *arXiv preprint arXiv:2606.04036*, 2026. 5

Justus Mattern, Sami Jaghouar, Manveer Basra, Jannik Straube, Matthew Di Ferrante, Felix Gabriel, Jack Min Ong, Vincent Weisser, and Johannes Hagemann. Synthetic-1: Two million collaboratively generated reasoning traces from deepseek-r1, 2025. URL https://www.primeintellect.ai/blog/synthetic-1-release. 6, 23

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022. 2, 5

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models, 2026. URL https://arxiv.org/abs/2602.04942. 5

Daniil Plyusov, Alexey Gorbatovski, Alexey Malakhov, Nikita Balagansky, Boris Shaposhnikov, Daria Korotyshova, and Daniil Gavrilov. Trust-region behavior blending for on-policy distillation, 2026. URL https://arxiv.org/abs/2605.31159. 5, 7

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015. 5

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 5

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. 2

Guobin Shen, Xiang Cheng, Chenxiao Zhao, Lei Huang, Jindong Li, Dongcheng Zhao, and Xing Yu. Anti-self-distillation for reasoning rl via pointwise mutual information. *arXiv preprint arXiv:2605.11609*, 2026. 5

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026. 2

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998. 5

Rui Wang, Hongru Wang, Yi Chen, Boyang Xue, Tianqing Fang, Wenhao Yu, and Kam-Fai Wong. Demystifying on-policy distillation: Roles, pathologies, and regulations, 2026. URL https://arxiv.org/abs/2607.13399. 2, 4

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992. 5

Zhengpeng Xie, Li Lyna Zhang, Zeke Xie, and Mao Yang. Trust region policy distillation, 2026. URL https://arxiv.org/abs/2607.04751. 5, 7

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. 2

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled rlvr. *arXiv preprint arXiv:2604.03128*, 2026. 2, 5

Xinsen Zhang, Zhenkai Ding, Tianjun Pan, Run Yang, Chun Kang, Xue Xiong, and Jingnan Gu. Opsdl: On-policy self-distillation for long-context language models. *arXiv preprint arXiv:2604.17535*, 2026. 5

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2024, 2024. 2

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2025, 2025. 2

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026. 2, 5, 7

Siqi Zhu, Xuyan Ye, Hongyu Lu, Weiye Shi, and Ge Liu. The many faces of on-policy distillation: Pitfalls, mechanisms, and fixes, 2026. URL https://arxiv.org/abs/2605.11182. 2, 4

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019. 5

APPENDIX: SUPPLEMENTARY MATERIAL

A Notation Reference **15**

B Theoretical Proofs **16**

B.1 Proof of Proposition 2.1 16

B.2 Proof of Proposition 2.2 16

B.3 Proof of the Return-to-Go Gradient Estimator 17

B.4 Global Optimum Versus Local Realization 18

B.5 Dynamic References as Iterative Geometric Proximal Updates 20

C Experimental Details **22**

C.1 Training Configuration 22

C.2 Evaluation Configuration 24

D Additional Full-Vocabulary Forward-KL Comparison **24**

E Additional Ablation Results **25**

E.1 Effect of Return-to-Go Credit Assignment 25

E.2 Effect of Interpolant References 25

E.3 Effect of the Interpolation Schedule 26

F Student–Teacher Mixture Sampling **26**

A NOTATION REFERENCE

Table 3: **Summary of notation.** For readability, we omit conditioning variables such as (x, c) when they are clear from context. Throughout the paper, π_θ denotes the student policy, π_{ref} denotes the reference policy, and p_T denotes the privileged teacher.

Notation	Description
x	Input prompt.
c	Privileged context available only during training.
$y = (y_1, \ldots, y_T)$	Generated reasoning trajectory.
y_t	Token at decoding step t.
$y_{<t}$	Prefix before token t.
$h_t = (x, y_{<t})$	Decoding history or prefix state at step t.
$\pi_\theta(y \mid x)$	Student policy parameterized by θ.
$p_T(y \mid x, c)$	Privileged teacher policy conditioned on prompt x and context c.
$\pi_{\text{ref}}(y \mid x)$	Reference policy, e.g., the initial student or a stop-gradient copy.
$\bar{\theta}$	Stop-gradient copy of the current student parameters.
$R(y; x, c)$	Trajectory-level reward in the KL-regularized RL formulation.
$\mathcal{J}_\beta(\theta)$	β-OPSD objective, obtained by using the teacher-to-reference log-ratio reward.
β	KL regularization coefficient in β-OPSD.
β_k	Regularization coefficient used at training step k.
$\pi^\star_\beta(y \mid x, c)$	Closed-form optimal policy of the β-OPSD objective.
$z_{\text{ref}}(\cdot \mid h_t)$	Reference/student logits at prefix h_t.
$z_T(\cdot \mid h_t, c)$	Privileged teacher logits at prefix h_t.
$\tilde{p}_{\beta_k}(\cdot \mid h_t, c)$	Local logit-interpolant target at step k.
$\tilde{p}_{\beta_k}(y \mid x, c)$	Autoregressive sequence-level logit-interpolant target.
$s_k = \frac{k}{K-1}$	Normalized training progress.
$w_k = 1/\beta_k$	Teacher weight in the logit-interpolant target at training step k.
η_k	Teacher weight in the optional mixture-sampling proposal (Appendix F); distinct from w_k.
$\rho_t^{\beta_k}(y)$	Token-level log-ratio between the student and interpolant target.
$G_{t,\gamma}^{\beta_k}(y)$	Discounted return-to-go from token t using future interpolant log-ratios.
$\omega_t^{\eta_k}(y)$, $\Omega_t^{\eta_k}(y)$	Token-level and cumulative importance ratios for mixture sampling.
γ	Discount factor for the return-to-go.
$\text{sg}(\cdot)$	Stop-gradient operator.
$D_{\text{KL}}(\cdot \Vert \cdot)$	Kullback–Leibler divergence.
$\mathbb{E}_{y \sim \pi_\theta}$	Expectation over trajectories sampled from the student policy.

B Theoretical Proofs

B.1 Proof of Proposition 2.1

Proof. Fix a prompt x and privileged context c. For readability, we suppress the conditioning on (x, c) when it is unambiguous. Expanding Eq. 3, we have

$$\mathcal{J}_\beta(\pi_\theta) = \mathbb{E}_{y\sim\pi_\theta}\left[\log \frac{p_T(y)}{\pi_{\text{ref}}(y)}\right] - \beta D_{\text{KL}}\left(\pi_\theta \,\|\, \pi_{\text{ref}}\right) \tag{13}$$

$$= \mathbb{E}_{y\sim\pi_\theta}\left[\log p_T(y) - \log \pi_{\text{ref}}(y)\right] - \beta \mathbb{E}_{y\sim\pi_\theta}\left[\log \pi_\theta(y) - \log \pi_{\text{ref}}(y)\right] \tag{14}$$

$$= \mathbb{E}_{y\sim\pi_\theta}\left[\log p_T(y) - \beta \log \pi_\theta(y) + (\beta - 1)\log \pi_{\text{ref}}(y)\right]. \tag{15}$$

On the other hand,

$$- D_{\text{KL}}\left(\pi_\theta \,\|\, p_T\right) - (\beta - 1) D_{\text{KL}}\left(\pi_\theta \,\|\, \pi_{\text{ref}}\right) \tag{16}$$

$$= -\mathbb{E}_{y\sim\pi_\theta}\left[\log \pi_\theta(y) - \log p_T(y)\right] - (\beta - 1)\mathbb{E}_{y\sim\pi_\theta}\left[\log \pi_\theta(y) - \log \pi_{\text{ref}}(y)\right] \tag{17}$$

$$= \mathbb{E}_{y\sim\pi_\theta}\left[\log p_T(y) - \beta \log \pi_\theta(y) + (\beta - 1)\log \pi_{\text{ref}}(y)\right]. \tag{18}$$

Comparing this expression with Eq. 15, we obtain the equivalent decomposition

$$\mathcal{J}_\beta(\pi_\theta) = -D_{\text{KL}}\left(\pi_\theta(\cdot \mid x) \,\|\, p_T(\cdot \mid x, c)\right) - (\beta - 1) D_{\text{KL}}\left(\pi_\theta(\cdot \mid x) \,\|\, \pi_{\text{ref}}(\cdot \mid x)\right). \tag{19}$$

Setting $\beta = 1$ cancels the reference-regularization term:

$$\mathcal{J}_1(\pi_\theta) = -D_{\text{KL}}\left(\pi_\theta(\cdot \mid x) \,\|\, p_T(\cdot \mid x, c)\right). \tag{20}$$

Therefore,

$$\max_\theta \mathcal{J}_1(\pi_\theta) \quad \Longleftrightarrow \quad \min_\theta D_{\text{KL}}\left(\pi_\theta(\cdot \mid x) \,\|\, p_T(\cdot \mid x, c)\right), \tag{21}$$

which is exactly the standard sequence-level OPSD objective in Eq. 4. This proves the proposition. □

B.2 Proof of Proposition 2.2

Proof. The β-OPSD objective:

$$\mathcal{J}_\beta(\pi) = \mathbb{E}_{y\sim\pi(\cdot|x)}\left[\log \frac{p_T(y \mid x, c)}{\pi_{\text{ref}}(y \mid x)}\right] - \beta D_{\text{KL}}\left(\pi(\cdot \mid x) \,\|\, \pi_{\text{ref}}(\cdot \mid x)\right). \tag{22}$$

Expanding the KL term gives

$$\mathcal{J}_\beta(\pi) = \mathbb{E}_{y\sim\pi}\left[\log p_T(y) - \log \pi_{\text{ref}}(y)\right] - \beta \mathbb{E}_{y\sim\pi}\left[\log \pi(y) - \log \pi_{\text{ref}}(y)\right] \tag{23}$$

$$= \mathbb{E}_{y\sim\pi}\left[\log p_T(y) - \beta \log \pi(y) + (\beta - 1)\log \pi_{\text{ref}}(y)\right]. \tag{24}$$

Now define

$$\pi^\star_\beta(y \mid x, c) = \frac{\pi_{\text{ref}}(y \mid x)^{1-\frac{1}{\beta}} p_T(y \mid x, c)^{\frac{1}{\beta}}}{Z_\beta(x, c)}, \tag{25}$$

where

$$Z_\beta(x, c) = \sum_{y'} \pi_{\text{ref}}(y' \mid x)^{1-\frac{1}{\beta}} p_T(y' \mid x, c)^{\frac{1}{\beta}}. \tag{26}$$

Taking logs, we have

$$\log \pi^\star_\beta(y \mid x, c) = \left(1 - \frac{1}{\beta}\right)\log \pi_{\text{ref}}(y \mid x) + \frac{1}{\beta}\log p_T(y \mid x, c) - \log Z_\beta(x, c). \tag{27}$$

Multiplying both sides by β gives

$$\beta \log \pi^\star_\beta(y \mid x, c) = (\beta - 1)\log \pi_{\text{ref}}(y \mid x) + \log p_T(y \mid x, c) - \beta \log Z_\beta(x, c). \tag{28}$$

Substituting Eq. 28 into Eq. 24, we obtain

$$\mathcal{J}_\beta(\pi) = \mathbb{E}_{y\sim\pi}\left[\beta \log \pi^\star_\beta(y \mid x, c) + \beta \log Z_\beta(x, c) - \beta \log \pi(y \mid x)\right] \tag{29}$$

$$= -\beta \mathbb{E}_{y\sim\pi}\left[\log \frac{\pi(y \mid x)}{\pi^\star_\beta(y \mid x, c)}\right] + \beta \log Z_\beta(x, c) \tag{30}$$

$$= -\beta D_{\text{KL}}\left(\pi(\cdot \mid x) \,\|\, \pi^\star_\beta(\cdot \mid x, c)\right) + \beta \log Z_\beta(x, c). \tag{31}$$

The second term is independent of π, while the KL term is nonnegative and is minimized to zero if and only if

$$\pi(\cdot \mid x) = \pi^{\star}_{\beta}(\cdot \mid x, c). \tag{32}$$

Therefore, $\pi^{\star}_{\beta}$ is the maximizer of Eq. 3. This proves the proposition. □

B.3 Proof of the Return-to-Go Gradient Estimator

The exact result concerns the unnormalized return-to-go estimator, or normalization by a deterministic horizon, rather than the sampled length in Eq. 12.

Proposition B.1 (Unbiased return-to-go gradients with deterministic normalization)**.** *Fix x, c, k and hold $\tilde{p}_{\beta_k}(\cdot \mid x, c)$ fixed during differentiation. Let the vocabulary be finite and $H \geq 1$ be a deterministic generation cap independent of θ and the sampled trajectory. Both student and target terminate at the first EOS or at the cap, with $T(y) \in \{1, \ldots, H\}$ counting the sampled tokens, including EOS if emitted. After EOS, extend both processes by identical deterministic absorbing transitions. On the resulting finite trajectory space, assume the student has θ-independent support in a neighborhood of the current parameters, differentiable positive probabilities on that support, and a target positive there. Define*

$$\mathcal{L}_{\beta_k}(\theta) = D_{\mathrm{KL}}(\pi_\theta(\cdot \mid x) \,\|\, \tilde{p}_{\beta_k}(\cdot \mid x, c)) \,. \tag{33}$$

The distributions in this KL include the stated stopping and truncation convention. For $t \leq T(y)$, let $\rho_t^{\beta_k}$ be given by Eq. 10 and let $g_t(y) = \nabla_\theta \log \pi_\theta(y_t \mid x, y_{<t})$; set both terms to zero for $t > T(y)$. For $\gamma = 1$, define

$$G_t^{\beta_k}(y) = \sum_{s=t}^{H} \rho_s^{\beta_k}(y), \qquad \widehat{g}_{\mathrm{RTG}}(y) = \sum_{t=1}^{H} G_t^{\beta_k}(y) g_t(y), \qquad \widehat{g}_H(y) = \frac{1}{H} \widehat{g}_{\mathrm{RTG}}(y). \tag{34}$$

Then

$$\mathbb{E}_{y \sim \pi_\theta}[\widehat{g}_{\mathrm{RTG}}(y)] = \nabla_\theta \mathcal{L}_{\beta_k}(\theta), \qquad \mathbb{E}_{y \sim \pi_\theta}[\widehat{g}_H(y)] = \frac{1}{H} \nabla_\theta \mathcal{L}_{\beta_k}(\theta). \tag{35}$$

Proof. Suppress x, c, k for readability. Absorbing padding is a one-to-one extension of stopped trajectories and does not change their probabilities. Thus, $\mathcal{L}(\theta) = \mathbb{E}_{\pi_\theta}[R(y)]$, where

$$R(y) := \sum_{t=1}^{H} \rho_t(y) = \log \frac{\pi_\theta(y)}{\tilde{p}(y)}, \qquad S(y) := \sum_{t=1}^{H} g_t(y) = \nabla_\theta \log \pi_\theta(y). \tag{36}$$

The fixed target gives $\nabla_\theta R(y) = S(y)$, and normalization of the student gives $\mathbb{E}_{\pi_\theta}[S(y)] = 0$. Differentiating the finite sum defining $\mathcal{L}$ therefore yields

$$\nabla_\theta \mathcal{L}(\theta) = \mathbb{E}_{\pi_\theta}[R(y)S(y) + S(y)] = \mathbb{E}_{\pi_\theta}\left[\sum_{t=1}^{H} g_t(y) \sum_{s=1}^{H} \rho_s(y)\right]. \tag{37}$$

Let $\mathcal{F}_{t-1}$ contain the padded prefix before step t. At an active prefix, the conditional score has mean zero because $\sum_v \pi_\theta(v \mid h_t) \nabla_\theta \log \pi_\theta(v \mid h_t) = \nabla_\theta \sum_v \pi_\theta(v \mid h_t) = 0$; at an absorbed prefix, $g_t = 0$. Thus $\mathbb{E}[g_t \mid \mathcal{F}_{t-1}] = 0$ in both cases. For $s < t$, ρ_s is $\mathcal{F}_{t-1}$-measurable, so

$$\mathbb{E}[g_t \rho_s] = \mathbb{E}[\rho_s \, \mathbb{E}[g_t \mid \mathcal{F}_{t-1}]] = 0. \tag{38}$$

Removing these past terms in Eq. 37 gives

$$\nabla_\theta \mathcal{L}(\theta) = \mathbb{E}\left[\sum_{t=1}^{H} g_t \sum_{s=t}^{H} \rho_s\right] = \mathbb{E}[\widehat{g}_{\mathrm{RTG}}]. \tag{39}$$

All post-termination terms vanish, so the sums may equivalently end at $T(y)$. Since H is deterministic, division by H proves the second identity. □

Surrogate gradients and scope. Differentiating Eq. 33 with sampled tokens and return weights held fixed gives exactly $\widehat{g}_H$. The expectation of this samplewise surrogate gradient, not differentiation through the sampling operation, is the quantity in Eq. 35. The result is local to the current update: refreshing student-derived target logits between updates does not make it a total-derivative statement through the changing target. The cap H defines the trajectory laws being compared; padding alone does not equate a capped KL with an untruncated KL. The unnormalized result also extends to almost-surely finite, unbounded stopping times when differentiation may be interchanged with expectation and $\mathbb{E}[\sum_t \|g_t\| + \sum_{t,s} \|g_t\| \, |\rho_s|] < \infty$. No such extension is needed for the bounded-horizon statement above.

Why normalization by sampled length is different. Even at $\gamma = 1$, the gradient of Eq. 12 is $\widehat{g}_{\mathrm{RTG}}(y)/T(y)$. Although $T(y)$ is held fixed for a sampled trajectory's backward pass, it remains random under $y \sim \pi_\theta$. In particular,

$$\mathbb{E}\left[\frac{\widehat{g}_{\mathrm{RTG}}}{T}\right] = \mathbb{E}[T^{-1}]\, \nabla_\theta \mathcal{L}_{\beta_k}(\theta) + \mathrm{Cov}\big(T^{-1}, \widehat{g}_{\mathrm{RTG}}\big)\,, \tag{40}$$

where the covariance is taken componentwise. It need not vanish or be parallel to the KL gradient. Nor can the past-term cancellation be repeated after inserting $1/T(y)$, since the final length generally depends on the current and future tokens and is not measurable with respect to $\mathcal{F}_{t-1}$.

For a concrete counterexample, consider the three trajectories EOS, a EOS, and b EOS, of lengths $1, 2, 2$. Let their student probabilities be $\mathrm{softmax}(\theta_1, \theta_2, 0)$ and their fixed target probabilities be $(1/4, 1/4, 1/2)$. After a or b, EOS is deterministic under both models, so only the first token contributes a mismatch or score. At $\theta = (\log 2, 0)$, the student probabilities are $(1/2, 1/4, 1/4)$, and direct summation gives

$$\nabla_\theta \mathcal{L} = \log 2 \begin{pmatrix} 3/8 \\ -1/16 \end{pmatrix}, \qquad \mathbb{E}\left[\frac{\widehat{g}_{\mathrm{RTG}}}{T}\right] = \log 2 \begin{pmatrix} 5/16 \\ -3/32 \end{pmatrix}. \tag{41}$$

These vectors are not scalar multiples, demonstrating that sampled-length normalization can change the expected update direction, not merely its scale.

It also does not generally give the exact gradient of the explicitly length-weighted log-ratio objective $\mathcal{F}_{\mathrm{len}}(\theta) := \mathbb{E}_{\pi_\theta}[R(y)/T(y)]$. Under the same assumptions,

$$\begin{aligned} \nabla_\theta \mathcal{F}_{\mathrm{len}}(\theta) &= \mathbb{E}\left[\frac{R(y)+1}{T(y)} S(y)\right] \\ &= \mathbb{E}\left[\frac{\widehat{g}_{\mathrm{RTG}}(y)}{T(y)}\right] + \mathbb{E}\left[\frac{1}{T(y)} \sum_{t=1}^{H} g_t(y) \left(1 + \sum_{s<t} \rho_s(y)\right)\right]. \end{aligned} \tag{42}$$

The extra terms arise from the parameter dependence of R and the failure of length-weighted past-term cancellation. Hence, we retain $1/T(y)$ only as an explicit practical heuristic, not as an unbiased gradient estimator of either objective. When $T(y) \equiv H$, the normalization is deterministic and the exact result applies; for random $T(y)$, replacing $1/T(y)$ by $1/H$ and setting $\gamma = 1$ recovers it. Discounting with $\gamma < 1$ is a separate approximation and does not in general preserve unbiasedness for the undiscounted sequence-level KL, even with fixed normalization.

B.4 Global Optimum Versus Local Realization

Unlike local logit interpolation, the global optimum accounts for the geometric reference–teacher mass of future continuations, denoted by $Z_t(h_t)$. The two conditionals coincide exactly when this continuation mass is independent of the current action. The following result quantifies their difference.

Proposition B.2 (Global optimum and local-realization error)**.** *Fix $\beta \geq 1$, a finite vocabulary, and a deterministic horizon, with reference and teacher policies positive on common allowed token supports. At every reachable prefix h_t,*

$$\pi^\star_\beta(a \mid h_t, c) = \tilde{p}_\beta(a \mid h_t, c) \frac{Z_{t+1}(h_t a)}{\mathbb{E}_{a' \sim \tilde{p}_\beta(\cdot \mid h_t, c)}[Z_{t+1}(h_t a')]}. \tag{43}$$

Let $\delta_t(h_t)$ be the range of $\log Z_{t+1}(h_t a)$ over supported actions. Then

$$D_{\mathrm{KL}}\big(\tilde{p}_\beta(\cdot \mid h_t, c) \,\big\|\, \pi^\star_\beta(\cdot \mid h_t, c)\big) \le \frac{\delta_t(h_t)^2}{8}, \tag{44}$$

$$D_{\mathrm{KL}}\big(\pi^\star_\beta(\cdot \mid h_t, c) \,\big\|\, \tilde{p}_\beta(\cdot \mid h_t, c)\big) \le \frac{\delta_t(h_t)^2}{8}. \tag{45}$$

Proof. Fix x, c and $\beta \ge 1$, and write $w = 1/\beta$. Assume a finite vocabulary and a deterministic horizon H, with reference and teacher policies positive on common allowed token supports. Variable-length trajectories capped at H are padded using identical absorbing transitions after EOS; these contribute a factor of one. All trajectory distributions below use the same stopping and truncation convention.

Continuation normalizers and exact conditionals. Define the geometric token factor

$$\psi_t(a \mid h_t, c) := \pi_{\mathrm{ref}}(a \mid h_t)^{1-w} p_T(a \mid h_t, c)^w, \qquad w = \frac{1}{\beta}, \tag{46}$$

and the backward continuation partition function

$$Z_t(h_t) := \sum_{y_{t:H}} \prod_{s=t}^{H} \psi_s(y_s \mid h_s, c), \qquad Z_{H+1}(\cdot) = 1. \tag{47}$$

The sum is over allowed continuations, and c is suppressed in Z_t. Writing $h_t a$ for the prefix obtained by appending a gives

$$Z_t(h_t) = \sum_a \psi_t(a \mid h_t, c) Z_{t+1}(h_t a). \tag{48}$$

These normalizers are finite and positive on reachable prefixes, and $Z_1(x) = Z_\beta(x, c)$ for the trajectory law in Eq. 5.

The global optimum assigns a trajectory unnormalized mass $\prod_{s=1}^{H} \psi_s(y_s \mid h_s, c)$. Summing over continuations and cancelling the common prefix mass and global normalizer yields

$$\pi^\star_\beta(a \mid h_t, c) = \frac{\psi_t(a \mid h_t, c) Z_{t+1}(h_t a)}{Z_t(h_t)}. \tag{49}$$

In contrast, the logit-interpolant target in Eq. 7 normalizes only the current token factors:

$$\tilde{p}_\beta(a \mid h_t, c) = \frac{\psi_t(a \mid h_t, c)}{\sum_{a'} \psi_t(a' \mid h_t, c)}. \tag{50}$$

Therefore,

$$Z_t(h_t) = \left(\sum_a \psi_t(a \mid h_t, c) \right) \mathbb{E}_{a' \sim \tilde{p}_\beta(\cdot \mid h_t, c)}[Z_{t+1}(h_t a')], \tag{51}$$

which proves Eq. 43.

Continuation variation and ratio bound. Let $\mathcal{A}_t(h_t)$ be the support of $\tilde{p}_\beta(\cdot \mid h_t, c)$. Define the continuation log-value and its action-wise oscillation by

$$V_{t+1}(h_t, a) := \log Z_{t+1}(h_t a), \tag{52}$$

$$\delta_t(h_t) := \max_{a \in \mathcal{A}_t(h_t)} V_{t+1}(h_t, a) - \min_{a \in \mathcal{A}_t(h_t)} V_{t+1}(h_t, a). \tag{53}$$

Fix a reachable prefix and abbreviate the local and global conditionals as $q(a) = \tilde{p}_\beta(a \mid h_t, c)$ and $p(a) = \pi^\star_\beta(a \mid h_t, c)$. Write $V(a) = V_{t+1}(h_t, a)$, $m = \min_a V(a)$, $M = \max_a V(a)$, and $\delta = M - m$, taking extrema over the common support. The correction identity gives $p(a) = q(a) e^{V(a)} / \mathbb{E}_q[e^V]$. Since $e^m \le \mathbb{E}_q[e^V] \le e^M$, it follows that

$$e^{-\delta_t(h_t)} \le \frac{\pi^\star_\beta(a \mid h_t, c)}{\tilde{p}_\beta(a \mid h_t, c)} \le e^{\delta_t(h_t)}, \qquad a \in \mathcal{A}_t(h_t). \tag{54}$$

KL bounds and exactness. Let $f(s) = \log \mathbb{E}_q[e^{sV}]$ and $q_s(a) = q(a)e^{sV(a)-f(s)}$ for $s \in [0,1]$. Then $q_0 = q$, $q_1 = p$, $f(0) = 0$, and

$$f'(s) = \mathbb{E}_{q_s}[V], \qquad f''(s) = \mathrm{Var}_{q_s}(V) \le \mathbb{E}_{q_s}\left[\left(V - \frac{m+M}{2}\right)^2\right] \le \frac{\delta^2}{4}. \tag{55}$$

Hence,

$$D_{\mathrm{KL}}(q\|p) = f(1) - f'(0) = \int_0^1 (1-s)f''(s)\,ds \le \frac{\delta^2}{8}, \tag{56}$$

$$D_{\mathrm{KL}}(p\|q) = f'(1) - f(1) = \int_0^1 sf''(s)\,ds \le \frac{\delta^2}{8}, \tag{57}$$

proving Eqs. 44 and 45. The identity $p = q$ holds if and only if $Z_{t+1}(h_t a)$ is constant across actions in the support of q. Thus, local interpolation omits an action-dependent continuation correction and is accurate when the log continuation mass varies little across actions. Large variation permits, but does not guarantee, a large discrepancy between the two conditionals. □

B.5 Dynamic References as Iterative Geometric Proximal Updates

Our practical implementation uses the current stop-gradient student as the reference endpoint while retaining a fixed privileged teacher. This admits a simple distribution-space interpretation.

Fix a prefix h and a finite action set on which the initial student and fixed teacher are strictly positive. Write

$$\pi_k(a) := \pi_{\theta_k}(a \mid h), \qquad p_T(a) := p_T(a \mid h, c).$$

For teacher weight $w_k = 1/\beta_k$, define

$$q_{k+1}(a) = \frac{\pi_k(a)^{1-w_k} p_T(a)^{w_k}}{\sum_{a'} \pi_k(a')^{1-w_k} p_T(a')^{w_k}}. \tag{58}$$

Proposition B.3 (Reverse-KL barycentric interpretation)**.** *For $w_k \in [0,1]$ and compatible support, q_{k+1} is the unique solution of*

$$q_{k+1} = \arg\min_q \left\{(1-w_k)D_{\mathrm{KL}}(q\|\pi_k) + w_k D_{\mathrm{KL}}(q\|p_T)\right\}. \tag{59}$$

Proof. Let $C_{k+1} = \sum_a \pi_k(a)^{1-w_k} p_T(a)^{w_k}$ be the positive normalizer in Eq. 58. For any distribution q on the action set,

$$(1-w_k)D_{\mathrm{KL}}(q\|\pi_k) + w_k D_{\mathrm{KL}}(q\|p_T) = D_{\mathrm{KL}}(q\|q_{k+1}) - \log C_{k+1}. \tag{60}$$

The last term is independent of q, and the KL is nonnegative, with equality only at $q = q_{k+1}$. This proves uniqueness, including the endpoints $w_k = 0$ and $w_k = 1$. For $w_k > 0$, dividing the objective by w_k gives the equivalent proximal form

$$q_{k+1} = \arg\min_q \left\{D_{\mathrm{KL}}(q\|p_T) + \frac{1-w_k}{w_k} D_{\mathrm{KL}}(q\|\pi_k)\right\}. \tag{61}$$

At $w_k = 0$, the update leaves the student unchanged. □

Thus, rebuilding the interpolant around the current student can be viewed as repeatedly constructing a reverse-KL barycenter between the current policy and the fixed privileged teacher.

The resulting geometry becomes particularly transparent in an idealized exact-target-tracking limit.

Proposition B.4 (Composition of dynamic-reference targets)**.** *Assume that, at a fixed prefix, the student exactly reaches the target after every update,*

$$\pi_{k+1} = q_{k+1}. \tag{62}$$

Define

$$\alpha_K := \prod_{j=0}^{K-1} (1-w_j). \tag{63}$$

Then after K updates,

$$\pi_K(a) = \frac{\pi_0(a)^{\alpha_K} p_T(a)^{1-\alpha_K}}{\sum_{a'} \pi_0(a')^{\alpha_K} p_T(a')^{1-\alpha_K}}. \tag{64}$$

Hence the cumulative effective teacher weight is

$$W_K = 1 - \alpha_K = 1 - \prod_{j=0}^{K-1} (1 - w_j). \tag{65}$$

Proof. Set $\alpha_0 = 1$, so Eq. 64 holds at $K = 0$. If it holds at step k, exact target tracking and Eq. 58 give

$$\pi_{k+1}(a) \propto \left[\pi_0(a)^{\alpha_k} p_T(a)^{1-\alpha_k}\right]^{1-w_k} p_T(a)^{w_k} = \pi_0(a)^{\alpha_k(1-w_k)} p_T(a)^{1-\alpha_k(1-w_k)}. \tag{66}$$

The normalizer inherited from π_k is independent of a and cancels. Thus $\alpha_{k+1} = \alpha_k(1 - w_k)$, proving the composition formula and $W_K = 1 - \alpha_K$ by induction. □

Partial target tracking. The exact-target recursion is an idealized limit. To model incomplete optimization, suppose instead that the next distribution takes a geometric step of size $\eta_k \in [0, 1]$ toward q_{k+1}:

$$\pi_{k+1} \propto \pi_k^{1-\eta_k} q_{k+1}^{\eta_k}. \tag{67}$$

Using Eq. (70),

$$\pi_{k+1} \propto \pi_k^{1-\eta_k w_k} p_T^{\eta_k w_k}, \tag{68}$$

and therefore

$$\Delta_{k+1}(a, b) = (1 - \eta_k w_k)\Delta_k(a, b). \tag{69}$$

Hence the effective residual reference weight after K steps is

$$\alpha_K = \prod_{j=0}^{K-1} (1 - \eta_j w_j). \tag{70}$$

This idealized model separates the geometry of the target, controlled by w_k, from how closely optimization tracks that target, represented by η_k. It should not be interpreted as an exact model of parameter-space SGD.

An equivalent characterization follows from teacher-relative log odds. For any two actions a, b, define

$$\Delta_k(a, b) := \log \frac{\pi_k(a)}{\pi_k(b)} - \log \frac{p_T(a)}{p_T(b)}. \tag{71}$$

Then

$$\Delta_{k+1}(a, b) = (1 - w_k)\Delta_k(a, b), \qquad \Delta_K(a, b) = \alpha_K \Delta_0(a, b). \tag{72}$$

Therefore, whenever

$$\prod_{k=0}^{\infty} (1 - w_k) = 0, \tag{73}$$

the idealized iterates converge to the privileged teacher. For $w_k \in [0, 1)$, a sufficient condition is $\sum_k w_k = \infty$.

Indeed, taking action log-ratios in Eq. 58 cancels its normalizer and proves Eq. 72. If $\alpha_K \to 0$, positivity and finiteness of the action set imply that the numerator in Eq. 64 converges to $p_T(a)$ and its normalizer to one; hence $\pi_K \to p_T$ in total variation. For $w_k < 1$, $\prod_{j=0}^{K-1}(1 - w_j) \le \exp(-\sum_{j=0}^{K-1} w_j)$, establishing the stated sufficient condition. If any $w_k = 1$, the next idealized iterate equals p_T and remains there.

This analysis describes the geometry of the moving target rather than the literal neural-network optimization dynamics: in practice, each update is a finite parameter-space step and is estimated only on sampled prefixes. The exact-target recursion should therefore be interpreted as an idealized limiting model of dynamic-reference distillation.

Table 4: **Mathematical reasoning performance at step 100.** We report avg@12 (%) as mean ± standard error over 10 evaluation runs. Gains are percentage-point differences from Vanilla OPSD at the same model scale. β-OPSD uses $w_{\text{start}} = 0.5$ and $w_{\text{end}} = 0.8$. Bold indicates the best result at each model scale.

Method	AIME 2024	AIME 2025	HMMT 2025	Average
Qwen3-8B				
Base model	$73.06_{\pm 0.74}$	$67.78_{\pm 0.78}$	$29.72_{\pm 0.76}$	$56.85_{\pm 0.44}$
Vanilla OPSD	$75.83_{\pm 0.71}$	$66.67_{\pm 0.79}$	$\mathbf{33.06}_{\pm 0.78}$	$58.52_{\pm 0.44}$
SFT	$72.78_{\pm 0.74}$	$64.72_{\pm 0.80}$	$30.83_{\pm 0.77}$	$56.11_{\pm 0.44}$
GRPO	$75.83_{\pm 0.71}$	$68.61_{\pm 0.77}$	$31.11_{\pm 0.77}$	$58.52_{\pm 0.43}$
β-OPSD (ours)	$\mathbf{78.33}_{\pm 0.69}$ (+2.50)	$\mathbf{70.28}_{\pm 0.76}$ (+3.61)	$31.94_{\pm 0.78}$ (-1.12)	$\mathbf{60.18}_{\pm 0.43}$ (+1.66)
Qwen3-4B				
Base model	$\mathbf{77.78}_{\pm 0.69}$	$64.17_{\pm 0.80}$	$29.44_{\pm 0.76}$	$57.13_{\pm 0.43}$
Vanilla OPSD	$75.00_{\pm 0.72}$	$65.00_{\pm 0.79}$	$28.33_{\pm 0.75}$	$56.11_{\pm 0.44}$
SFT	$23.61_{\pm 0.71}$	$19.72_{\pm 0.66}$	$10.56_{\pm 0.51}$	$17.96_{\pm 0.37}$
GRPO	$76.39_{\pm 0.71}$	$64.17_{\pm 0.80}$	$30.28_{\pm 0.77}$	$56.95_{\pm 0.44}$
β-OPSD (ours)	$73.89_{\pm 0.73}$ (-1.11)	$\mathbf{69.17}_{\pm 0.77}$ (+4.17)	$\mathbf{30.56}_{\pm 0.77}$ (+2.23)	$\mathbf{57.87}_{\pm 0.44}$ (+1.76)
Qwen3-1.7B				
Base model	$50.00_{\pm 0.83}$	$37.22_{\pm 0.81}$	$14.44_{\pm 0.59}$	$33.89_{\pm 0.43}$
Vanilla OPSD	$44.17_{\pm 0.83}$	$35.56_{\pm 0.80}$	$13.33_{\pm 0.57}$	$31.02_{\pm 0.43}$
SFT	$37.50_{\pm 0.81}$	$29.17_{\pm 0.76}$	$12.22_{\pm 0.55}$	$26.30_{\pm 0.41}$
GRPO	$48.33_{\pm 0.83}$	$38.61_{\pm 0.81}$	$15.00_{\pm 0.60}$	$33.98_{\pm 0.44}$
β-OPSD (ours)	$\mathbf{53.33}_{\pm 0.83}$ (+9.16)	$\mathbf{40.83}_{\pm 0.82}$ (+5.27)	$\mathbf{16.11}_{\pm 0.61}$ (+2.78)	$\mathbf{36.76}_{\pm 0.44}$ (+5.74)

Table 5: **LiveCodeBench performance at step 100.** We report pass@1 (%) by difficulty and overall as mean ± standard error over 10 evaluation runs. Gains are percentage-point differences from Vanilla OPSD at the same model scale. β-OPSD uses $w_{\text{start}} = 0.5$ and $w_{\text{end}} = 0.8$. Bold indicates the best result at each model scale.

Method	Easy	Medium	Hard	Overall pass@1
Qwen3-8B				
Base model	$49.69_{\pm 0.88}$	$17.23_{\pm 0.61}$	$5.14_{\pm 0.37}$	$23.13_{\pm 0.41}$
Vanilla OPSD	$49.69_{\pm 0.88}$	$\mathbf{20.89}_{\pm 0.66}$	$4.00_{\pm 0.33}$	$24.08_{\pm 0.42}$
SFT	$43.48_{\pm 0.87}$	$2.61_{\pm 0.26}$	$1.14_{\pm 0.18}$	$14.60_{\pm 0.34}$
GRPO	$50.31_{\pm 0.88}$	$16.71_{\pm 0.60}$	$5.43_{\pm 0.38}$	$23.22_{\pm 0.41}$
β-OPSD (ours)	$\mathbf{52.48}_{\pm 0.88}$ (+2.79)	$20.37_{\pm 0.65}$ (-0.52)	$\mathbf{8.57}_{\pm 0.47}$ (+4.57)	$\mathbf{26.26}_{\pm 0.43}$ (+2.18)
Qwen3-4B				
Base model	$46.89_{\pm 0.88}$	$17.49_{\pm 0.61}$	$4.00_{\pm 0.33}$	$21.99_{\pm 0.40}$
Vanilla OPSD	$52.17_{\pm 0.88}$	$20.37_{\pm 0.65}$	$6.29_{\pm 0.41}$	$25.40_{\pm 0.42}$
SFT	$15.53_{\pm 0.64}$	$2.09_{\pm 0.23}$	$0.57_{\pm 0.13}$	$5.69_{\pm 0.23}$
GRPO	$47.52_{\pm 0.88}$	$18.28_{\pm 0.62}$	$3.14_{\pm 0.29}$	$22.18_{\pm 0.40}$
β-OPSD (ours)	$\mathbf{54.04}_{\pm 0.88}$ (+1.87)	$\mathbf{22.72}_{\pm 0.68}$ (+2.35)	$\mathbf{7.71}_{\pm 0.45}$ (+1.42)	$\mathbf{27.30}_{\pm 0.43}$ (+1.90)
Qwen3-1.7B				
Base model	$37.27_{\pm 0.85}$	$5.48_{\pm 0.37}$	$1.14_{\pm 0.18}$	$13.74_{\pm 0.34}$
Vanilla OPSD	$37.58_{\pm 0.85}$	$\mathbf{9.92}_{\pm 0.48}$	$2.00_{\pm 0.24}$	$15.73_{\pm 0.35}$
SFT	$20.81_{\pm 0.72}$	$2.61_{\pm 0.26}$	$0.00_{\pm 0.00}$	$7.30_{\pm 0.25}$
GRPO	$35.40_{\pm 0.84}$	$5.74_{\pm 0.38}$	$1.43_{\pm 0.20}$	$13.36_{\pm 0.33}$
β-OPSD (ours)	$\mathbf{41.61}_{\pm 0.87}$ (+4.03)	$9.40_{\pm 0.47}$ (-0.52)	$\mathbf{3.43}_{\pm 0.31}$ (+1.43)	$\mathbf{17.25}_{\pm 0.37}$ (+1.52)

C Experimental Details

C.1 Training Configuration

Table 6 summarizes the settings shared by the controlled comparisons.

Algorithm 1 Training β-OPSD with Scheduled Logit Interpolants and Return-to-Go

Require: Student π_θ, privileged teacher p_T, dataset $\mathcal{D}$, training budget K, schedule endpoints $w_{\text{start}}, w_{\text{end}}$, discount factor γ

for $k = 0, \ldots, K-1$ **do**
  Sample a minibatch $(x, c) \sim \mathcal{D}$
  Sample on-policy responses $y = (y_1, \ldots, y_T) \sim \pi_\theta(\cdot \mid x)$
  Set normalized progress $s_k = k/(K-1)$
  Set teacher weight w_k by Eq. 9 and $\beta_k = 1/w_k$
  Set $\bar{\theta} \leftarrow \text{sg}(\theta)$
  Construct interpolant targets $\tilde{p}_{\beta_k}(\cdot \mid h_t, c)$ along the sampled prefixes using Eq. 7
  Compute token mismatches $\rho_t^{\beta_k}(y)$ using Eq. 10
  Compute discounted return-to-go weights $G_{t,\gamma}^{\beta_k}(y)$ using Eq. 11
  Minimize $\mathcal{L}_{\beta\text{-OPSD}}(\theta; y)$ in Eq. 12 and update θ
end for

Table 6: **Shared settings for the main controlled training comparisons.**

Setting	Value
Models	Qwen3-1.7B, Qwen3-4B, Qwen3-8B
Training budget	200 optimizer steps
Per-device batch size	1
Gradient accumulation	32 steps
Effective batch size	32
Learning rate	5×10^{-6}
Gradient clipping	0.1
LoRA rank / scaling	64 / 128
LoRA modules	Q/K/V/O and gate/up/down projections
Precision / attention	bfloat16 / FlashAttention-2
Training seed	42
Hardware	1 NVIDIA H200 per run
Saved checkpoints	25, 50, 75, 100, 200
Evaluation checkpoint	100

Method-specific length limits. For code, all methods use the 20,974-example DeepCoder dataset, including the Verified TACO (Li, 2024) and PrimeIntellect SYNTHETIC-1 (Mattern et al., 2025) subsets. SFT uses a 20,000-token sequence limit without rollouts. GRPO uses a 16,000-token completion limit and samples eight rollouts per problem. Vanilla OPSD and β-OPSD use an 8,192-token completion limit within a 20,000-token total sequence limit.

For math, all methods use OpenThoughts Math 30K. SFT uses a 20,000-token sequence limit without rollouts. GRPO uses a 16,000-token completion limit and samples eight rollouts per problem. Vanilla OPSD and β-OPSD use a 1,024-token completion limit.

On-policy generation uses a colocated vLLM instance with tensor-parallel size 1 and GPU-memory utilization 0.5. Practical OPSD rollouts use temperature 1.1, top-$p = 0.95$, and top-$k = 20$.

Sampling convention. In the training objective and gradient analysis, π_θ denotes the temperature-scaled student policy, $\pi_\theta(a \mid h) = \text{softmax}(z_\theta(h)/\tau)_a$, where h is the current prefix and $\tau = 1.1$ for OPSD training. Student, reference, and teacher log-probabilities in the loss are computed using the same temperature. Our exact on-policy estimation statements assume trajectories sampled directly from this policy, $y \sim \pi_\theta(\cdot \mid x)$, without top-$k$ or top-p filtering. Practical rollouts additionally use top-$k = 20$ and top-$p = 0.95$; these filters alter the sampling distribution and therefore introduce sampling bias relative to the theoretical estimator. Along each sampled trajectory, we evaluate the token-level mismatch between the student and the interpolant target:

Table 7: **Evaluation configurations for mathematical and code reasoning.**

Setting	Mathematical reasoning	Code reasoning
Benchmark	AIME 2024, AIME 2025, HMMT February 2025	LiveCodeBench `release_v6` (1,055 problems)
Metric	pass@12 and avg@12	pass@1 overall and by difficulty
Samples per problem	12	1
Thinking mode	Enabled	Disabled
Temperature	0.7	0.2
Top-p	0.95	0.95
Top-k	50	No additional top-k setting
Maximum generated tokens	No separate completion cap	16,384
Maximum model length	40,960	32,768
Evaluation batch size	32	Implementation default
Execution	—	2 workers; 6-second timeout per test
Code extraction	—	Last complete Python fenced block
Evaluated checkpoint	Step 100	Step 100

C.2 Evaluation Configuration

Table 7 gives the complete decoding, scoring, and execution settings for both domains.

We load each LoRA adapter on its corresponding original Qwen3 checkpoint and use tensor-parallel size 1 with GPU-memory utilization 0.9. The reported checkpoints are trained with seed 42. We conduct 10 independent decoding runs and report means with standard errors. For math, pass@12 is the fraction of problems solved by at least one of 12 samples, while avg@12 averages correctness across all 12 samples. For code, pass@1 is the fraction of problems whose extracted program passes every test case. All metrics are multiplied by 100 and reported as percentages.

D Additional Full-Vocabulary Forward-KL Comparison

The main-paper Vanilla OPSD baseline uses the sampled-token reverse-KL signal $\log \pi_\theta(y_t \mid h_t) - \log p_T(y_t \mid h_t, c)$ on student-generated tokens. As an additional objective comparison, we also implement the full-vocabulary forward KL $D_{\mathrm{KL}}(p_T(\cdot \mid h_t, c) \| \pi_\theta(\cdot \mid h_t))$ at each visited prefix. We train and evaluate both objectives while holding the model scale, data, training budget, and evaluation protocol fixed. This comparison isolates sensitivity to the KL direction and token aggregation; it is not presented as the original formulation of Vanilla OPSD.

Table 8: **Additional comparison of sampled-token reverse KL and full-vocabulary forward KL for Vanilla OPSD on mathematical reasoning.** We report avg@12 (%). Each Δ is the percentage-point difference between forward and reverse KL at the same model scale (forward minus reverse).

		AIME 2024		AIME 2025		HMMT 2025		Average	
Model	KL implementation	Score	Δ	Score	Δ	Score	Δ	Score	Δ
Qwen3-1.7B	Tinker sampled reverse KL	44.17	—	35.56	—	13.33	—	31.02	—
	Full-vocab. forward KL	49.44	+5.27	36.67	+1.11	15.00	+1.67	33.70	+2.68
Qwen3-4B	Tinker sampled reverse KL	75.00	—	65.00	—	28.33	—	56.11	—
	Full-vocab. forward KL	73.06	-1.94	61.94	-3.06	30.56	+2.22	55.19	-0.92
Qwen3-8B	Tinker sampled reverse KL	75.83	—	66.67	—	33.06	—	58.52	—
	Full-vocab. forward KL	75.56	-0.28	66.39	-0.28	31.67	-1.39	57.87	-0.65

Table 8 shows that full-vocabulary forward KL raises the three-benchmark average avg@12 by 2.68 points at 1.7B, but lowers it by 0.92 and 0.65 points at 4B and 8B, respectively. Table 9 shows a more scale-dependent coding result: forward KL is lower overall by 0.28 and 0.95 points at 1.7B and 4B, but higher by 4.36 points at 8B. Thus, the KL direction and token aggregation

Table 9: **Additional comparison of sampled-token reverse KL and full-vocabulary forward KL for Vanilla OPSD on LiveCodeBench.** We report pass@1 (%). Each Δ is the percentage-point difference between full-vocabulary forward KL and sampled-token reverse KL at the same model scale (forward minus reverse).

		Easy		Medium		Hard		Overall pass@1	
Model	KL implementation	Score	Δ	Score	Δ	Score	Δ	Score	Δ
Qwen3-1.7B	Sampled reverse KL	37.58	—	9.92	—	2.00	—	15.73	—
	Full-vocab. forward KL	39.13	+1.55	8.36	-1.56	1.43	-0.57	15.45	-0.28
Qwen3-4B	Sampled reverse KL	52.17	—	20.37	—	6.29	—	25.40	—
	Full-vocab. forward KL	52.17	0.00	19.06	-1.31	4.86	-1.43	24.45	-0.95
Qwen3-8B	Sampled reverse KL	49.69	—	20.89	—	4.00	—	24.08	—
	Full-vocab. forward KL	56.52	+6.83	24.28	+3.39	7.14	+3.14	28.44	+4.36

materially affect the Vanilla OPSD baseline, and neither implementation uniformly dominates across domains and scales. There is a performance–efficiency tradeoff: the substantially more expensive full-vocabulary objective can outperform β-OPSD in one setting, whereas β-OPSD improves the lightweight sampled-token formulation under a practical training budget.

E Additional Ablation Results

E.1 Effect of Return-to-Go Credit Assignment

We isolate the contribution of return-to-go credit assignment. Both variants use the same logit-interpolant target $\tilde{p}_{\beta_k}$ and the same student-generated trajectories, but differ in how the token-level signal is assigned.

The *Local Token Gradient* variant uses only the instantaneous log-ratio:

$$\widehat{g}_{\mathrm{local}}(y) = \sum_{t=1}^{T} \rho_t^{\beta_k}(y) \nabla_\theta \log \pi_\theta(y_t \mid x, y_{<t}). \tag{74}$$

In contrast, the *Return-to-Go* variant in β-OPSDuses the cumulative future mismatch:

$$\widehat{g}_{\mathrm{RTG}}(y) = \sum_{t=1}^{T} G_{t,\gamma}^{\beta_k}(y) \nabla_\theta \log \pi_\theta(y_t \mid x, y_{<t}). \tag{75}$$

Figure 6 compares both gradient variants against Vanilla OPSD across domains and model sizes. On math tasks, return-to-go yields larger gains than local token gradients in seven of nine benchmark–model settings, with one tie and one exception (Qwen3-4B on AIME 2024). Local token gradients also improve several settings, but their gains are smaller and less consistent. On code tasks, the comparison is more model-dependent: return-to-go gives higher overall pass@1 at 4B, whereas local token gradients perform better overall at 1.7B and 8B. Return-to-go nevertheless improves easy and hard problem performance over Vanilla OPSD at every scale, despite declines on medium problems at 1.7B and 8B. Together with the aggregate results in Section 4.2, these findings show consistent overall gains from β-OPSD over Vanilla OPSD across both domains and all three model sizes, suggesting benefits that extend beyond mathematical reasoning. They do not, however, imply that return-to-go uniformly outperforms local credit assignment; its relative advantage depends on the domain, model size, and difficulty.

E.2 Effect of Interpolant References

We ablate how the two endpoints of the logit interpolant are constructed. Let θ_0 denote the initial student parameters, let θ_k denote the student parameters at training step k, and let $\bar{\theta}_k = \mathrm{sg}(\theta_k)$ denote a stop-gradient copy of the current student. In our self-distillation setting, the teacher branch uses the same model architecture but additionally conditions on privileged information c.



Figure 6: **Effect of Return-to-Go credit assignment.** All variants use the same scheduled logit-interpolant target with $w_{\text{start}} = 0.5$ and $w_{\text{end}} = 0.8$. We report math avg@12 (%) and LiveCodeBench pass@1 (%) across model scales and difficulty levels.

We compare three reference choices. First, *fixed student + fixed teacher* constructs both endpoints from the initial model:

$$\tilde{p}_w^{\text{FF}}(\cdot \mid h_t, c) = \text{softmax}\left((1-w)z_{\theta_0}(\cdot \mid h_t) + wz_{\theta_0}(\cdot \mid h_t, c)\right). \tag{76}$$

Second, *dynamic student + dynamic teacher* refreshes both endpoints from the current model:

$$\tilde{p}_w^{\text{DD}}(\cdot \mid h_t, c) = \text{softmax}\left((1-w)z_{\bar{\theta}_k}(\cdot \mid h_t) + wz_{\bar{\theta}_k}(\cdot \mid h_t, c)\right). \tag{77}$$

Third, *dynamic student + fixed teacher*, our default construction, uses the current student as the student-side endpoint and a fixed privileged teacher as the teacher-side endpoint:

$$\tilde{p}_w^{\text{DF}}(\cdot \mid h_t, c) = \text{softmax}\left((1-w)z_{\bar{\theta}_k}(\cdot \mid h_t) + wz_{\theta_0}(\cdot \mid h_t, c)\right). \tag{78}$$

To isolate the effect of the interpolant references, we evaluate all three constructions using the same return-to-go estimator and identical training settings. We keep the teacher weight fixed at $w = 0.5$, so that the two endpoints contribute equally to the target distribution. As shown in Figure 7, the dynamic-student and fixed-teacher construction performs best across all three benchmarks. These results suggest that the interpolant benefits from tracking the evolving student distribution while retaining a stable privileged teacher as its supervisory anchor.

E.3 Effect of the Interpolation Schedule

Finally, we study how the interpolation schedule affects performance. We focus on linear schedules for the teacher weight $w_k = 1/\beta_k$, varying the start and end values in Eq. 9. We also include a fixed $w_k = 0.5$ baseline.

Table 10 shows that all tested linear schedules, whether increasing or decreasing the teacher weight, improve all three benchmarks over Vanilla OPSD (Table 4). For a consistent comparison, we use $0.5 \rightarrow 0.8$ for the main results at every model scale rather than selecting a separate schedule for each model. These results suggest that the benefit is not tied to a particular monotonic curriculum. Increasing, decreasing, and fixed intermediate targets can all outperform direct teacher matching, indicating that the key mechanism is controlling the target distribution along the reference–teacher path rather than simply annealing toward the teacher.

F Student–Teacher Mixture Sampling

In addition to student on-policy sampling, we evaluate a guided sampling variant that constructs trajectories from a mixture of the student and privileged teacher. The distillation target retains the

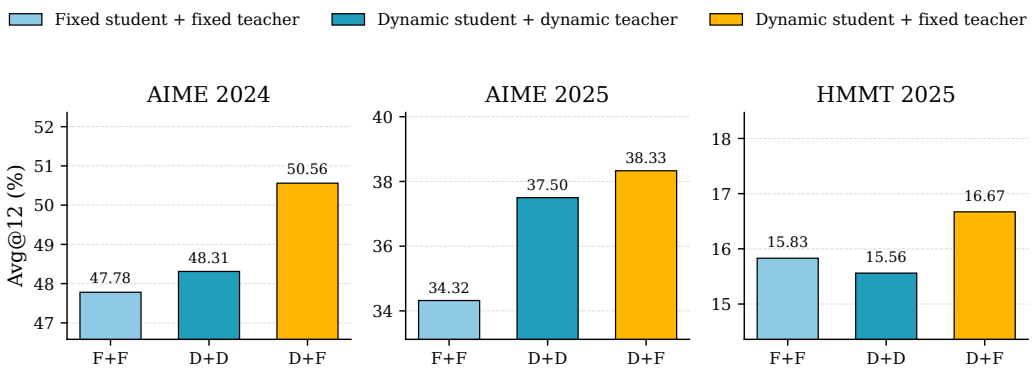


Figure 7: **Effect of interpolant endpoint choices on Qwen3-1.7B math avg@12 (%).** All variants use the same return-to-go estimator and fixed teacher weight $w = 0.5$. Benchmark-specific vertical ranges highlight relative differences; the dynamic-student, fixed-teacher construction performs best on all three benchmarks.

Table 10: **Effect of the teacher-weight schedule on Qwen3-1.7B math avg@12 (%).** The notation $a \to b$ denotes a linear schedule from $w_{\text{start}} = a$ to $w_{\text{end}} = b$; fixed 0.5 holds the weight constant. Bold indicates the best result in each column.

Schedule	AIME 2024	AIME 2025	HMMT 2025
Fixed 0.5	50.56	38.33	**16.67**
Linear $0.2 \to 0.8$	51.39	37.50	16.39
Linear $0.8 \to 0.2$	51.94	38.06	15.83
Linear $0.5 \to 0.8$	53.33	**40.83**	16.11
Linear $0.8 \to 0.5$	**54.72**	38.61	15.83

main-paper teacher weight $w_k = 1/\beta_k$ and token mismatch $\rho_t^{\beta_k}$; we use a separate coefficient $\eta_k \in [0, 1]$ for the teacher weight in the proposal distribution. At training step k, the token-level proposal is

$$m_{\bar{\theta},\eta_k}(\cdot \mid h_t, c) = (1 - \eta_k)\, \pi_{\bar{\theta}}(\cdot \mid h_t) + \eta_k\, p_T(\cdot \mid h_t, c), \tag{79}$$

where $\pi_{\bar{\theta}}$ is a stop-gradient copy of the current student. Thus, $\eta_k = 0$ recovers student on-policy sampling, whereas $\eta_k = 1$ samples entirely from the privileged teacher. The resulting autoregressive proposal is

$$m_{\bar{\theta},\eta_k}(y \mid x, c) = \prod_{t=1}^{T} m_{\bar{\theta},\eta_k}(y_t \mid h_t, c). \tag{80}$$

Because the interpolant KL in Eq. 33 is defined under the student trajectory distribution π_θ, sampling from $m_{\bar{\theta},\eta_k}$ introduces a distribution mismatch. We correct this mismatch using per-decision importance sampling. For a trajectory $y \sim m_{\bar{\theta},\eta_k}(\cdot \mid x, c)$, define the token-level and cumulative importance ratios

$$\omega_t^{\eta_k}(y) = \frac{\pi_\theta(y_t \mid h_t)}{m_{\bar{\theta},\eta_k}(y_t \mid h_t, c)}, \qquad \Omega_t^{\eta_k}(y) = \prod_{i=1}^{t} \omega_i^{\eta_k}(y). \tag{81}$$

Using the main-paper token mismatch $\rho_s^{\beta_k}$, the importance-weighted return-to-go is

$$G_{t,\gamma}^{\beta_k,\eta_k,\text{mix}}(y) = \sum_{s=t}^{T} \gamma^{s-t} \Omega_s^{\eta_k}(y) \rho_s^{\beta_k}(y), \tag{82}$$

where $\rho_s^{\beta_k}$ is defined in Eq. 10. We optimize the corresponding mixed-sampling extension of the main loss,

$$\mathcal{L}_\beta^{\text{mix}}(\theta; y) = \frac{1}{T} \sum_{t=1}^{T} \text{sg}\left(G_{t,\gamma}^{\beta_k,\eta_k,\text{mix}}(y)\right) \log \pi_\theta(y_t \mid h_t). \tag{83}$$

We note that this variant is inherently off-policy and requires combining the student and teacher distributions at every decoding step. At the time of writing, such dual-model sampling is **not** directly supported by standard vLLM decoding pipelines, resulting in additional (nearly double) implementation complexity and generation overhead. Although mixed student–teacher sampling yields comparable improvements over vanilla OPSD, its reduced sampling efficiency makes it less attractive in practice.

Results. We evaluate fixed proposal teacher weights $\eta_k \in \{0.2, 0.5, 0.8\}$ and linear schedules that either increase or decrease η_k during training. Here, η_k and $1 - \eta_k$ are respectively the teacher and student weights in Eq. 79; they are distinct from the target teacher weight $w_k = 1/\beta_k$. All mixed-sampling variants use the same logit-interpolant target notation as the main method, discount factor $\gamma = 0.99$, and per-decision importance sampling (IS).

Table 11: **Student–teacher mixture-sampling results on Qwen3-1.7B math.** All non-baseline variants use β-OPSD with discount factor $\gamma = 0.99$ and per-decision importance sampling. Here, η_k is the teacher weight in the proposal distribution, distinct from the target teacher weight $w_k = 1/\beta_k$. We report pass@12 and avg@12 (%). Bold indicates the best result in each metric column.

Proposal schedule	Step	AIME 2024		AIME 2025		HMMT 2025	
		pass@12	avg@12	pass@12	avg@12	pass@12	avg@12
Vanilla OPSD	200	76.67	46.11	63.33	35.56	43.33	15.83
Fixed proposal: $\eta_k = 0.2$							
Mixed Sampling	50	76.67	50.83	60.00	37.50	40.00	15.83
Mixed Sampling	75	80.00	50.00	66.67	38.06	40.00	14.44
Mixed Sampling	100	76.67	52.78	53.33	35.83	33.33	13.61
Mixed Sampling	200	73.33	51.94	63.33	37.22	43.33	16.11
Fixed proposal: $\eta_k = 0.5$							
Mixed Sampling	50	80.00	50.83	66.67	36.39	43.33	16.11
Mixed Sampling	75	80.00	52.22	66.67	37.78	46.67	16.11
Mixed Sampling	100	76.67	**54.17**	63.33	39.44	46.67	17.22
Mixed Sampling	200	76.67	53.33	63.33	38.06	36.67	15.83
Fixed proposal: $\eta_k = 0.8$							
Mixed Sampling	50	80.00	50.83	60.00	36.94	43.33	15.83
Mixed Sampling	75	80.00	51.39	70.00	**41.11**	46.67	16.11
Mixed Sampling	100	73.33	51.94	60.00	36.11	40.00	15.83
Mixed Sampling	200	73.33	51.39	66.67	38.61	36.67	13.89
Linear proposal schedule: $\eta_k : 0.2 \to 0.8$							
Mixed Sampling	50	80.00	50.83	70.00	38.06	36.67	13.61
Mixed Sampling	75	76.67	50.56	70.00	36.39	**53.33**	18.06
Mixed Sampling	100	73.33	51.94	66.67	36.67	36.67	16.39
Mixed Sampling	200	76.67	50.56	60.00	**41.11**	43.33	16.94
Linear proposal schedule: $\eta_k : 0.8 \to 0.2$							
Mixed Sampling	50	76.67	50.56	**76.67**	36.39	30.00	13.61
Mixed Sampling	75	73.33	47.78	73.33	36.39	50.00	**18.89**
Mixed Sampling	100	80.00	50.56	60.00	38.61	36.67	15.83
Mixed Sampling	200	76.67	52.78	60.00	39.72	36.67	15.56

Table 11 shows that mixed student–teacher sampling can produce improvements comparable to those of the main on-policy β-OPSD method. At 200 steps, every evaluated schedule improves avg@12 over Vanilla OPSD on both AIME benchmarks. Across all checkpoints, the best observed gains over Vanilla OPSD are 8.06 points on AIME 2024, 5.55 points on AIME 2025, and 3.06 points on HMMT 2025. However, no single proposal schedule dominates across all benchmarks, indicating that the off-policy variant is sensitive to both the mixture coefficient and the training checkpoint. Given its additional dual-model decoding cost and the variance introduced by importance sampling, we use student on-policy sampling as the default implementation of β-OPSD.