RESPO: RESHAPED SEQUENCE POLICY OPTIMIZATION FOR GRADIENT STARVATION IN OFF-POLICY LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning from verifiable rewards (RLVR) frequently reuses rollouts across multiple policy updates, increasing the mismatch between the current policy and the data-generating policy. We identify a sign-dependent *gradient starvation* problem in clipped policy optimization: clipping suppresses under-generated positive responses at the low-importance-weight tail while permitting severely over-generated negative responses to dominate the high-weight tail. To address this, we propose **ReSPO** (Reshaped Sequence Policy Optimization), which replaces clipping with a smooth, two-branch sequence-level kernel derived from an α-divergence variational objective and an exponential variance-control tilt. The positive branch preserves a nonzero gradient weight for under-generated positive responses, while the negative branch suppresses heavily over-generated negative responses. We demonstrate that ReSPO effectively learns from long positive reasoning trajectories during early training, even when accumulated policy drift relegates them to the low-importance-weight tail. On dense and MoE Qwen3 models, ReSPO accelerates early optimization, improves final training scores, and achieves higher held-out benchmark performance under a rollout reuse, validating our approach on importance-weight tail control in off-policy learning.

1 INTRODUCTION

RL from verifiable rewards (RLVR) improves LLM reasoning using scalar rewards (Chen et al., 2025; DeepSeek-AI et al., 2025; Yang et al., 2025), and GRPO (Shao et al., 2024) computes group-relative advantages without a value function. Expensive and slow generation motivates rollout reuse, making the data *off-policy* in training: the current policy π_θ differs from the old policy π_{old} that generated the responses. The sequence importance ratio $W = \pi_\theta(\boldsymbol{o} \mid \boldsymbol{q})/\pi_{\text{old}}(\boldsymbol{o} \mid \boldsymbol{q})$ measures this mismatch on the prompt $\boldsymbol{q}$ and response $\boldsymbol{o}$. A small ratio W means that the response is now less likely to be generated by the current policy than the old policy, while a large W means that it is more likely. GRPO clips the token importance ratio $w_t = \pi_\theta(o_t|\boldsymbol{q}, \boldsymbol{o}_{<t})/\pi_{\text{old}}(o_t|\boldsymbol{q}, \boldsymbol{o}_{<t})$ and GSPO (Zheng et al., 2025) clips the length-normalized ratio $s = W^{1/|\boldsymbol{o}|}$ to stabilize training. Both methods use clipping, so their in-region gradients scale *linearly* with the importance ratio.

This linear weighting starves informative gradients in opposite ways depending on the advantage sign. We call responses with $\hat{A} > 0$ *positive responses* and those with $\hat{A} < 0$ *negative responses*, according to their group-relative reward. For positive responses, the gradient is proportional to w_t, so already familiar tokens near the upper clip threshold receive the largest updates, while under-generated tokens with $w_t \ll 1$ receive almost none. For negative responses, heavily over-generated tokens with $w_t \gg 1$ remain unclipped, so their unbounded weight can overshadow the gradient contributions from other tokens. The same failure carries over to the sequence-level ratio used by GSPO. We refer to this sign-dependent misallocation of learning signal as *gradient starvation*.

We tackle the failures by reshaping the importance weight W at the sequence level. Instead of clipping, we replace the raw W with a smooth, sign-dependent kernel $\phi^{(\pm)}(W)$, and the two failures above determine the shape we need: $\phi^{(+)}(0) > 0$ must preserve under-generated positive responses, while $\phi^{(-)}(0) = \phi^{(-)}(\infty) = 0$ must suppress both already-suppressed and severely over-generated negative responses. GRPO and GSPO clipping miss these critical tails: their effective weight van-

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.

ishes on the low-ratio tail but grows without bound on the high-ratio tail. VESPO (Shen et al., 2026) suppresses both tails, fitting negative advantages but starving positive ones at $W \to 0$. We propose **ReSPO** (Reshaped Sequence Policy Optimization), which derives a smooth two-branch kernel from an α-divergence variational objective with exponential tilting for variance control. The choice $\alpha^{(+)} > 1$ gives the required positive tail, while the $\alpha^{(-)} = 1$ limit gives both required negative tails. On DAPO-MATH (Yu et al., 2025) with Qwen3-1.7B-Base and Qwen3-30B-A3B-Base, ReSPO attains a higher late-stage training score than every clipped baseline in all six model–N settings and a higher score than VESPO in five of six.

2 Related Work

Policy optimization for LLM reasoning. PPO (Schulman et al., 2017) is the standard method for RLHF (Ouyang et al., 2022), using a clipped surrogate for stable updates. With automated verifiers (DeepSeek-AI et al., 2025), value-free alternatives have become popular. GRPO (Shao et al., 2024) computes group-relative advantages and clips per-token ratios; DAPO (Yu et al., 2025) adds decoupled clipping and dynamic sampling; and REINFORCE++ (Hu et al., 2025) uses global advantage normalization. These methods focus on token-level updates, whereas we study sequence-level gradient starvation. REAL (Zhai et al., 2026) reframes gradient starvation as a classification problem; we retain importance sampling to keep off-policy staleness explicit. This distinction matters increasingly as rollout reuse makes each batch more off-policy over successive optimizer steps.

Sequence-level and off-policy optimization. Token-level importance correction is mismatched with sequence-level rewards. GSPO (Zheng et al., 2025) uses a length-normalized sequence ratio for stable MoE training, while VESPO (Shen et al., 2026) derives $\phi_{\mathrm{KL}}(W) = W^{\beta}\exp(\lambda(1 - W))$ from a KL variational objective. Reusing rollouts across mini-batches causes off-policy drift (Noukhovitch et al., 2025), especially for MoE models (Zheng et al., 2025); Routing Replay (Ma et al., 2025) and truncated importance sampling (Liu et al., 2025) have been proposed to improve system stability. ReSPO complements these techniques by controlling how the importance weight allocates gradient across responses. We generalize VESPO's framework to the α-divergence family with a sign-dependent two-branch kernel. Sec. 3.3.4 and Fig. 1 compare these kernel designs.

Trust regions and divergence measures. TRPO (Schulman et al., 2015) limits updates with a KL bound, and PPO approximates this with hard clipping. Softer options include probability smoothing (Dwyer et al., 2025) and adaptive gating (Gao et al., 2025). Rényi-divergence bounds have also been used to analyze the variance of importance-sampling estimators in policy optimization (Metelli et al., 2018). ReSPO instead uses separable power-Bregman geometry to derive a sequence-level reshaping family. Its two branches allocate gradient according to the advantage sign and the importance-weight tail behavior that matters for off-policy optimization.

3 Method

3.1 Preliminaries: Token-Level and Sequence Policy Optimization

LLM and GRPO. A large language model (LLM) parameterized by θ defines an autoregressive policy $\pi_\theta(\boldsymbol{o} \mid \boldsymbol{q}) = \prod_{t=1}^{T} \pi_\theta(o_t \mid \boldsymbol{q}, o_{<t})$ that generates a response $\boldsymbol{o} = (o_1, \ldots, o_T)$ for a prompt $\boldsymbol{q}$. In the RLVR setting, a deterministic scalar reward $R(\boldsymbol{q}, \boldsymbol{o})$ gives a sequence-level signal. GRPO samples G responses $\{\boldsymbol{o}_i\}_{i=1}^{G}$ from the old policy π_{old} and forms the group-normalized advantage $\hat{A}_i = (R(\boldsymbol{q}, \boldsymbol{o}_i) - \mathrm{mean}_j R(\boldsymbol{q}, \boldsymbol{o}_j))/(\mathrm{std}_j R(\boldsymbol{q}, \boldsymbol{o}_j) + \epsilon_A)$; implementation details are given in App. C. GRPO then optimizes a token-level clipped surrogate:

$$\mathcal{J}_{\mathrm{GRPO}}(\theta) = \mathbb{E}_{\boldsymbol{q},\{\boldsymbol{o}_i\}}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|\boldsymbol{o}_i|}\sum_{t=1}^{|\boldsymbol{o}_i|}\min\Big(w_{i,t}\,\hat{A}_i,\ \mathrm{clip}(w_{i,t}, 1{-}\varepsilon, 1{+}\varepsilon)\,\hat{A}_i\Big)\right], \tag{1}$$

where $w_{i,t} = \pi_\theta(o_{i,t} \mid \boldsymbol{q}, \boldsymbol{o}_{i,<t})/\pi_{\mathrm{old}}(o_{i,t} \mid \boldsymbol{q}, \boldsymbol{o}_{i,<t})$ is the per-token importance ratio. Because the reward is defined at the sequence level, applying a separate single-sample ratio to each token does not recover the exact sequence-level importance correction and can introduce noisy token-wise scaling as sequence length grows (Shen et al., 2026; Zheng et al., 2025). Routing fluctuations create

an additional source of training–inference mismatch in MoE models, motivating stabilization techniques such as Routing Replay (Ma et al., 2025; Zheng et al., 2025). These instabilities compound as policy staleness grows.

Sequence-level ratio: GSPO. A natural fix is to work on the sequence-level importance weight $W(\boldsymbol{o}) = \prod_t w_t = \pi_\theta(\boldsymbol{o} \mid \boldsymbol{q})/\pi_{\text{old}}(\boldsymbol{o} \mid \boldsymbol{q})$, which aggregates token ratios at the same granularity as the reward. GSPO (Zheng et al., 2025) puts this view into practice with a length-normalized geometric-mean ratio $s_i(\theta) = (\pi_\theta(\boldsymbol{o}_i \mid \boldsymbol{q})/\pi_{\text{old}}(\boldsymbol{o}_i \mid \boldsymbol{q}))^{1/|\boldsymbol{o}_i|}$ and a clipped surrogate. GSPO trains more stably than GRPO, notably without Routing Replay on MoE models (Zheng et al., 2025), so the sequence-level weight is a natural starting point for off-policy reshaping. However, hard clipping of s_i keeps GRPO's gradient starvation inside the unclipped region. Moreover, the normalization $W^{1/|\boldsymbol{o}_i|}$ induces length-dependent bias because it tempers the sequence importance ratio by an exponent that depends on response length (see App. A.6).

3.2 Gradient Starvation

Let $\rho = w_t$ in GRPO and $\rho = s_i$ in GSPO. We call a response positive when the estimated advantage $\hat{A} > 0$ and negative when $\hat{A} < 0$. Inside the unclipped region, both objectives weight the gradient linearly by ρ. For GRPO and GSPO, the gradients are

$$\nabla_\theta \mathcal{J}_{\text{GRPO}} \propto \hat{A} w_t \nabla_\theta \log \pi_\theta(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t}), \quad \text{and} \quad \nabla_\theta \mathcal{J}_{\text{GSPO}} \propto \frac{\hat{A}_i s_i}{|\boldsymbol{o}_i|} \sum_{t=1}^{|\boldsymbol{o}_i|} \nabla_\theta \log \pi_\theta(o_{i,t} \mid \boldsymbol{q}, \boldsymbol{o}_{i,<t}).$$

Apart from the length factor in GSPO, the gradient magnitude of either objective grows linearly with $|\hat{A}|\rho$. Clipping discards large-ratio positive and small-ratio negative updates, but leaves two failures (Zhai et al., 2026):

- **Starved positive update.** When $\hat{A} > 0$ and $\rho \to 0$, a positive response is becoming less likely, but its recovery signal vanishes: $w_t = 0.01$ receives $100\times$ less weight than $w_t = 1$.
- **Dominating negative update.** When $\hat{A} < 0$ and $\rho \to \infty$, a negative response is becoming more likely. This side remains unclipped, so its weight grows without bound and can dominate repeated off-policy updates.

ReSPO addresses these GRPO failures by reshaping W. The likelihood shift encoded by W, together with the advantage sign, gives the four cases below.

How ReSPO should treat each tail. ↑ means "reinforce" and ↓ means "suppress"

Response	$W \to 0$**: less likely now**	$W \to \infty$**: more likely now**
$\hat{A} > 0$	↑**:** positive response being lost	↓**:** positive response already reinforced
$\hat{A} < 0$	↓**:** negative response already suppressed	↓**:** negative response being amplified

Therefore, ReSPO requires $\phi^{(+)}(0) > 0$ and $\phi^{(+)}(\infty) = \phi^{(-)}(0) = \phi^{(-)}(\infty) = 0$.

3.3 ReSPO: Reshaped Sequence Policy Optimization

To address gradient starvation, ReSPO applies a soft kernel to the sequence-level importance weight. The derivation has three stages: (i) interpret reshaping as a change of measure; (ii) use an α-divergence objective to obtain an unconstrained kernel ϕ_0; and (iii) project this measure under a moment constraint, producing a smooth exponential tilt ϕ. We then choose α by advantage sign so that the two branches satisfy the requirements above; App. A provides the full derivations.

3.3.1 Weight Reshaping as Measure Change

To find a ϕ that satisfies the properties stated in Sec. 3.2, we first ask what replacing the raw importance weight W with a generic reshaped weight $\phi(W)$ actually does to the gradient. Let $\mu = \pi_{\text{old}}$ denote the behavior policy from which rollouts are sampled and $\pi = \pi_\theta$ the current target policy, so

that the sequence-level importance weight defined in Sec. 3.1 is $W(\boldsymbol{o}) = \pi(\boldsymbol{o} \mid \boldsymbol{q})/\mu(\boldsymbol{o} \mid \boldsymbol{q})$. For the reward function $R(\cdot)$, the policy gradient of $\mathbb{E}_{\boldsymbol{o}\sim\pi} R(\boldsymbol{o})$ is

$$\nabla_\theta \mathbb{E}_{\boldsymbol{o}\sim\pi} R(\boldsymbol{o}) = \mathbb{E}_{\boldsymbol{o}\sim\pi} R(\boldsymbol{o}) \nabla_\theta \log \pi_\theta(\boldsymbol{o}) = \mathbb{E}_{\boldsymbol{o}\sim\mu} W(\boldsymbol{o}) \cdot R(\boldsymbol{o}) \cdot \nabla_\theta \log \pi_\theta(\boldsymbol{o})\,. \tag{2}$$

Applying any reshaping function $\phi : \mathbb{R}_+ \to \mathbb{R}_+$ to W implicitly defines an unnormalized measure τ. The measure-change identity is

$$\mathbb{E}_{\boldsymbol{o}\sim\mu}\big[\phi(W(\boldsymbol{o})) \cdot R(\boldsymbol{o}) \cdot \nabla_\theta \log \pi_\theta(\boldsymbol{o})\big] = \sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\, R(\boldsymbol{o}) \cdot \nabla_\theta \log \pi_\theta(\boldsymbol{o}), \tag{3}$$

where τ is given by

$$\tau(\boldsymbol{o}) = \mu(\boldsymbol{o})\, \phi(W(\boldsymbol{o})). \tag{4}$$

Eq. (4) fixes the notation used below: W is the raw sequence ratio, ϕ is its reshaping kernel, and $\tau = \mu\phi$ is an unnormalized measure. Let $Z := \sum_{\boldsymbol{o}} \tau(\boldsymbol{o}) = \mathbb{E}_\mu[\phi(W)]$ and $\bar\tau(\boldsymbol{o}) := \tau(\boldsymbol{o})/Z$. Then only $\bar\tau$ is a probability distribution, and $\sum_{\boldsymbol{o}} \tau(\boldsymbol{o}) g(\boldsymbol{o}) = Z\, \mathbb{E}_{\boldsymbol{o}\sim\bar\tau}[g]$. This reframing suggests a design strategy. Rather than hand-picking ϕ as clipping does, we first solve for an unnormalized τ, read off ϕ through Eq. (4), and finally verify its tails against the requirements in Sec. 3.2.

3.3.2 Variational Derivation of the Reshaping Kernel

Our goal is to design a measure τ that balances three goals: staying close to the behavior distribution μ for sample efficiency, moving toward the target policy π to reduce bias, and controlling variance for training stability. We begin with a brief review of the Bregman divergence.

Bregman divergence and the dual-proximity objective. Let $f : \mathbb{R}_+ \to \mathbb{R}$ be a strictly convex function, which we call the Bregman generator. The Bregman divergence induced by f for (unnormalized) measures $p, q \geq 0$ is

$$D_f(p \,\|\, q) = \sum_{\boldsymbol{o}} \Big[f\big(p(\boldsymbol{o})\big) - f\big(q(\boldsymbol{o})\big) - f'\big(q(\boldsymbol{o})\big)\,\big(p(\boldsymbol{o}) - q(\boldsymbol{o})\big) \Big]. \tag{5}$$

Borrowing the idea from annealed importance sampling (Neal, 2001), we formulate the dual-proximity objective for a general Bregman divergence:

$$\min_{\tau \geq 0} \;\; (1-\beta)\, D_f(\tau \,\|\, \mu) \;+\; \beta\, D_f(\tau \,\|\, \pi), \tag{6}$$

where $\beta \in (0, 1)$ controls the trade-off. At $\beta = 0$ we aim to recover the behavior policy ($\tau = \mu$, zero importance-weight variance but maximum bias), and at $\beta = 1$ we target the current policy ($\tau = \pi$, unbiased but with possibly high variance). Throughout this paper, τ is an unnormalized measure; only $\bar\tau$ denotes its normalized probability distribution. See App. A.1 for discussion.

Choice of Bregman generator. The choice of f determines the geometry of the interpolation and, with it, the form and tail behavior of the reshaping kernel.

KL divergence ($f(x) = x \log x - x$). This is the choice in VESPO (Shen et al., 2026). The first-order condition of Eq. (6) gives the geometric mixture $\tau^*_{\mathrm{KL}}(\boldsymbol{o}) = \mu(\boldsymbol{o})^{1-\beta} \cdot \pi(\boldsymbol{o})^{\beta}$, with reshaping kernel $\phi_{\mathrm{KL}}(W) = W^\beta$. This is also the continuous $\alpha \to 1$ limit of the power-mean family below. After exponential tilting it provides the two vanishing tails required by the negative branch, but its value at $W \to 0$ cannot satisfy the positive branch.

α-divergence ($f(x) = \frac{1}{\alpha(\alpha-1)} x^\alpha$, $\alpha \neq 0, 1$). f is strictly convex on $\mathbb{R}_+$ for all $\alpha \notin \{0, 1\}$. Solving the first-order condition gives the power-mean interpolation

$$\tau^*_0(\boldsymbol{o}) = \Big[(1-\beta)\, \mu(\boldsymbol{o})^{\alpha-1} + \beta\, \pi(\boldsymbol{o})^{\alpha-1} \Big]^{\frac{1}{\alpha-1}}, \tag{7}$$

and substituting $\pi(\boldsymbol{o}) = \mu(\boldsymbol{o})\, W(\boldsymbol{o})$ together with Eq. (4) yields the corresponding unconstrained reshaping kernel

$$\phi_0(W) = \big[(1-\beta) + \beta\, W^{\alpha-1} \big]^{1/(\alpha-1)}. \tag{8}$$

This kernel is valid for both $0 < \alpha < 1$ and $\alpha > 1$ and satisfies $\phi_0(1) = 1$ by construction. The full first-order derivation is given in App. A.1 for completeness.

3.3.3 EXPONENTIAL TILT FOR VARIANCE CONTROL

The unconstrained kernel ϕ_0 from Eq. (8) redistributes gradient weight and requires a separate variance-control step. Since $\tau = \mu\,\phi$, the natural second-moment target is $\mathbb{E}_\mu[\phi(W)^2] = \sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi(W(\boldsymbol{o}))$. The final ϕ is still unknown, so we replace it inside the constraint by the fixed proxy ϕ_0 and require $\sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi_0(W(\boldsymbol{o})) \le C_1$. This proxy is sufficient because the resulting tilt leads to bounded $\mathbb{E}_\mu[\phi(W)^2]$, as shown in App. A.2. We additionally impose $\sum_{\boldsymbol{o}} \tau(\boldsymbol{o}) \le C_2$ to prevent the measure from having unbounded mass.

For this projection step, we use the generalized KL divergence for unnormalized measures $p, q \ge 0$: $D_{\mathrm{KL}}(p\|q) = \sum_{\boldsymbol{o}} \Big(p(\boldsymbol{o}) \log \frac{p(\boldsymbol{o})}{q(\boldsymbol{o})} - p(\boldsymbol{o}) + q(\boldsymbol{o})\Big)$. We find the closest measure to τ_0^* that satisfies both constraints in this KL geometry (Csiszár, 1975):

$$\tau^* = \arg\min_{\tau}\; D_{\mathrm{KL}}(\tau \,\|\, \tau_0^*) \quad \text{s.t.} \quad \sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi_0(W(\boldsymbol{o})) \le C_1, \quad \sum_{\boldsymbol{o}} \tau(\boldsymbol{o}) \le C_2. \tag{9}$$

The KKT conditions for this KL projection give an exponential tilt of τ_0^* with sufficient statistic $\phi_0(W)$:

$$\tau^*(\boldsymbol{o}) \propto \tau_0^*(\boldsymbol{o}) \cdot \exp\big(-\lambda\,\phi_0(W(\boldsymbol{o}))\big), \tag{10}$$

where $\lambda \ge 0$ is the Lagrange multiplier. Extracting the reshaping kernel via $\tau = \mu\,\phi(W)$ and using $\phi_0(1) = 1$ to normalize $\phi(1) = 1$ gives

$$\phi(W) = \underbrace{\phi_0(W)}_{\alpha\text{ reshaping}} \cdot \underbrace{\exp\big(\lambda\,(1 - \phi_0(W))\big)}_{\text{exponential tilt}}. \tag{11}$$

The two divergences therefore play different roles: the α-divergence determines the power-mean interpolation and its tails, while the KL projection turns a linear moment constraint into a smooth multiplicative tilt rather than a hard boundary (App. A.2). As $\alpha \to 1^+$, $\phi_0(W) \to W^\beta$ (App. A.4) and the kernel becomes $W^\beta \exp(\lambda(1 - W^\beta))$. This differs from VESPO (Shen et al., 2026), whose exponential factor acts on the raw weight W rather than the reshaped weight W^β.

3.3.4 TAIL BEHAVIOR AND THE TWO-BRANCH DESIGN

The kernel of Eq. (11) depends on three hyperparameters (α, β, λ), and its tail behavior determines whether the gradient requirements of Sec. 3.2 are met. We extend the power-mean kernel continuously to $\alpha = 1$ by defining $\phi_0(W) = W^\beta$ (App. A.4). The key observation is that no single value of α satisfies both the positive- and negative-advantage requirements simultaneously.

Tail analysis. For $\lambda > 0$, the tilt sends $\phi(W)$ to zero whenever $\phi_0(W)$ diverges. For $\alpha > 1$, $\phi(0) > 0$ and $\phi(\infty) = 0$, matching the positive-branch requirements. For $0 < \alpha < 1$, $\phi(0) = 0$ but $\phi(\infty) > 0$, so extremely over-generated negative responses retain nonzero weight, and these outliers could impact the training. At the boundary $\alpha = 1$, $\phi(0) = \phi(\infty) = 0$, exactly matching the negative-branch requirements. This is where the sign-specific design enters the variational construction.

The ReSPO kernel. We use a sign-dependent parameterization: $\alpha^{(+)} > 1$ for positive advantages and the $\alpha^{(-)} = 1$ limiting form for negative advantages, both with $0 < \beta^{(\pm)} < 1$. The full ReSPO reshaping kernel is

$$\phi(W; \hat{A}) = \phi_0^{(\pm)}(W) \cdot \exp\big(\lambda^{(\pm)}\,(1 - \phi_0^{(\pm)}(W))\big), \tag{12}$$

where $(+)$ applies when $\hat{A} \ge 0$ and $(-)$ applies when $\hat{A} < 0$, and

$$\phi_0^{(+)}(W) = \Big[(1 - \beta^{(+)}) + \beta^{(+)} W^{\alpha^{(+)}-1}\Big]^{1/(\alpha^{(+)}-1)}, \qquad \phi_0^{(-)}(W) = W^{\beta^{(-)}}. \tag{13}$$

The negative branch is the $\alpha^{(-)} \to 1$ limit of the same family. Both branches satisfy $\phi(1) = 1$, and together they realize all four sign-specific tail requirements. Combining the kernel with group-normalized advantages gives the ReSPO policy gradient for a sampled batch from Eq. (3):

$$\nabla_\theta \mathcal{J}_{\mathrm{ReSPO}} = \mathbb{E}_{\boldsymbol{q},\,\{\boldsymbol{o}_i\}\sim\pi_{\mathrm{old}}} \left[\frac{1}{G}\sum_{i=1}^{G} \mathrm{sg}\left(\phi(W_i; \hat{A}_i)\right) \cdot \hat{A}_i \cdot \sum_{t=1}^{|\boldsymbol{o}_i|} \nabla_\theta \log \pi_\theta(o_{i,t} \mid \boldsymbol{q}, \boldsymbol{o}_{i,<t})\right], \tag{14}$$

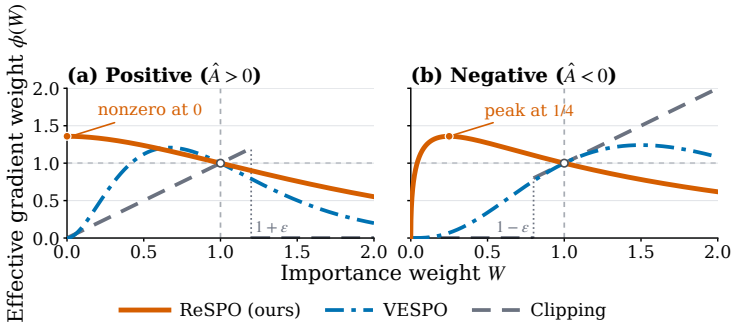
Here $\mathrm{sg}(\cdot)$ is the stop-gradient operator and $W_i = \prod_{t=1}^{|o_i|} w_{i,t}$ is the sequence-level importance weight for response i. We apply the kernel to the unnormalized ratio W, because the variational derivation (Sec. 3.3.2) operates on distributions over sequences and W is the natural Radon–Nikodym derivative; length normalization in GSPO would distort the geometry. The detached weight $\phi(W_i; \hat{A}_i)$ acts as a per-sequence gradient scaling coefficient in a REINFORCE-style estimator. For an on-policy update, $W = 1$ and both branches reduce to unit weight (App. A.5). App. B gives the numerically stable log-space form, and Alg. 1 summarizes the complete update.

3.3.5 Hyperparameter Selection

The three parameters have separate roles: α determines the tail regime, β sets the behavior–target interpolation, and λ controls the exponential tilt. The tail analysis fixes α by advantage sign. We set $\beta^{(\pm)} = 0.5$ for a symmetric interpolation, then choose $\lambda^{(\pm)}$ according to the desired local shape.

Positive branch: $\alpha^{(+)} = 2$, $\beta^{(+)} = 0.5$, $\lambda^{(+)} = 2$. At $\alpha^{(+)} = 2$, the power mean reduces to the linear form $\phi_0^{(+)}(W) = (1 - \beta^{(+)}) + \beta^{(+)} W$. Setting $\beta^{(+)} = 0.5$ gives the arithmetic mean $\phi_0^{(+)}(W) = \frac{1}{2}(1 + W)$, treating μ and π symmetrically. We require $\phi^{(+)\prime}(0) = 0$, which gives $\lambda^{(+)} = 1/(1 - \beta^{(+)}) = 2$ and makes $W = 0$ the global maximum. Although an importance weight cannot be negative, the linear closed form extends to the real line and has its unique maximum at zero (App. A.3). The resulting kernel is $\phi^{(+)}(W) = \frac{1+W}{2} \exp(1 - W)$, which decreases strictly for $W > 0$ and has maximum $\phi^{(+)}(0) = \frac{1}{2}e \approx 1.36$.

Negative branch: $\alpha^{(-)} = 1$, $\beta^{(-)} = 0.5$, $\lambda^{(-)} = 2$. The choice $\alpha^{(-)} = 1$ is made so that $\phi^{(-)}(0) = \phi^{(-)}(\infty) = 0$. With $\beta^{(-)} = 0.5$, its unconstrained kernel is the geometric mean $\phi_0^{(-)}(W) = \sqrt{W}$. We choose $\lambda^{(-)}$ by matching the two branches at the on-policy point: $\phi^{(+)\prime}(1) = -\frac{1}{2}$, while $\phi^{(-)\prime}(1) = \beta^{(-)}(1 - \lambda^{(-)})$. Equating the derivatives gives $\lambda^{(-)} = 2$ (App. A.3) and $\phi^{(-)}(W) = \sqrt{W} \exp\left(2(1 - \sqrt{W})\right)$. Thus the branches match in value and slope at $W = 1$. Fig. 1 shows that the negative kernel peaks at $W^* = \frac{1}{4}$ with value $e/2$ and vanishes on both tails.



Tail behavior

Method	$\phi^{(+)}$ $(0, \infty)$	$\phi^{(-)}$ $(0, \infty)$
Clipping	$(0, 0)$	$(0, \infty)$
VESPO	$(0, 0)$	$(0, 0)$
ReSPO	$(e/2, 0)$	$(0, 0)$

Figure 1: Kernel shapes and tail limits ($\varepsilon = 0.2$). The adjacent table gives the limits as $W \to (0, \infty)$. ReSPO preserves positive-branch signal near zero while suppressing both negative tails.

4 Experiments

We evaluate ReSPO on mathematical reasoning with one dense model, **Qwen3-1.7B-Base**, and one mixture-of-experts (MoE) model, **Qwen3-30B-A3B-Base**. For each model we vary the rollout-reuse ratio N to study how the methods degrade as off-policy drift grows.

4.1 Experimental Setup

Training & evaluation protocol. All experiments use `verl` (Sheng et al., 2025), asynchronous vLLM rollouts (Kwon et al., 2023), mini-batch size $M = 32$, $G = 8$ rollouts per prompt, and the GRPO advantage estimator. The dense model uses FSDP with AdamW, whereas the MoE model

uses Megatron with Adam; both use a learning rate of 10^{-6} and no KL penalty. Tab. 2 lists the shared training settings. Both models are trained on DAPO-MATH-17k (Yu et al., 2025); we use a strict-box verifier as the training reward (App. D.1). The prompt limit is $1{,}024$ tokens, and the response limits are $15{,}360$ and $8{,}192$ for Qwen3-1.7B and Qwen3-30B-A3B. The 30B setting adds a soft length penalty: it is zero through $4{,}096$ tokens and decreases linearly to -1 at the $8{,}192$-token limit, encouraging concise responses without truncating them at $4{,}096$. We set the global batch size to NM and split each rollout batch into N consecutive mini-batches. Thus larger N reuses the same π_{old} for more updates and induces more off-policy drift. We evaluate $N \in \{8, 16, 32\}$ and fix every run to 1024 policy updates, corresponding to $32{,}768$ training prompts. Our evaluation uses the full test splits of AIME 2025, AIME 2024, and AMC 2023 (OpenCompass, 2025; Maxwell-Jia, 2024; math-ai, 2023), together with OlympiadBench (He et al., 2024), MinervaMath (Lewkowycz et al., 2022), and MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024). We sample with temperature 1.0, top-$p = 0.7$, and no top-k truncation, allowing a maximum response length of $16{,}384$. For each problem, we generate 16 responses on AIME 2025, AIME 2024, and AMC 2023, and 4 on each remaining benchmark; each completion is scored using Math-Verify (Kydlíček, 2024) (App. D.2).

Baselines, metrics, and reporting. We compare with GRPO (Shao et al., 2024), GSPO (Zheng et al., 2025), and VESPO (Shen et al., 2026) under matched data and optimizer settings. Tabs. 2 and 3 list the shared and method-specific settings, respectively. In particular, we use VESPO's published best-performing sign-specific setting, $(\beta^{(+)}, \lambda^{(+)}) = (2, 3)$ and $(\beta^{(-)}, \lambda^{(-)}) = (3, 2)$. Tab. 1 reports per-benchmark results at $N \in \{8, 16, 32\}$. For training curves, we apply the affine transformation $(\texttt{score} + 1)/2$ to the recorded DAPO-MATH score and plot it against the number of policy updates, making experiments with different N directly comparable (App. D.1). We call this quantity training accuracy for 1.7B and penalized training accuracy for 30B. We report the maximum logged score within the first 256 policy updates without an error bar and the policy-step-weighted trapezoidal mean over the last 128 policy updates with its within-run temporal standard deviation. Tab. 1 uses a cluster-bootstrap standard error over evaluation problems (App. D.3). We discuss cross-seed standard deviations in Sec. 4.3.

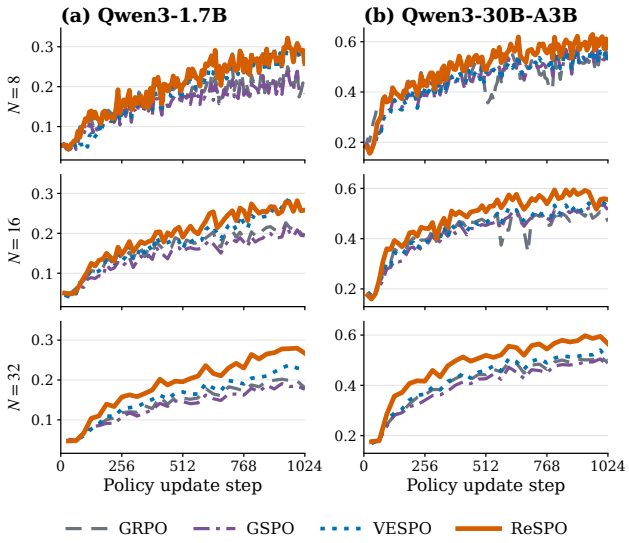
4.2 Training and Evaluation Performance

Fig. 2 and Tab. 1 compare optimization and generalization across both models and all three values of N. Figs. 5 and 6 additionally report evaluation accuracy, response length, approximate KL divergence, and entropy throughout training. Normalized training score and held-out pass@1 show the same broad pattern: the observed advantage of ReSPO is generally largest at $N = 32$.

Training performance. ReSPO improves quickly and remains higher through most of the training trajectory. It has the highest observed early peak in five of six settings; for 30B at $N = 8$, its peak is 1.0 pp below GRPO. For $N \in \{16, 32\}$, its early-peak differences from the highest baseline are 1.0 and 3.0 pp on 1.7B and 4.2 and 5.4 pp on 30B. ReSPO also has the highest observed last-128 mean in five of six settings; on 1.7B at $N = 16$, it is 0.4 pp below VESPO, and each central value lies within the other's temporal range. Its late mean exceeds the highest clipped baseline in all six settings by 4.2–8.4 pp, with non-overlapping temporal ranges. At $N = 32$, these differences reach 7.9 pp on 1.7B and 8.4 pp on 30B. App. E.1 gives the complete model- and N-specific comparison.

Evaluation performance. At $N = 8$, the macro averages are similar relative to their evaluation sampling uncertainty. For $N \in \{16, 32\}$, ReSPO has the highest observed mean in all four model–N settings, exceeding the highest baseline by 0.8 and 1.3 pp on 1.7B and by 3.2 and 1.9 pp on 30B. ReSPO has the highest value in 9 of the 12 model–benchmark comparisons at $N = 16$ and 8 at $N = 32$. The pattern is most consistent on difficult competition benchmarks, where it has the highest AIME25 and AMC23 values in every model–N setting. Averaged over AIME25, AIME24, and AMC23 on 30B, its differences from the highest baseline are 6.3, 5.8, and 3.4 pp for $N \in \{8, 16, 32\}$, respectively. App. E.1 provides a detailed summary of these comparisons.

Accuracy across response lengths. Using 10 quantile-based response-length bins shared across methods and benchmarks for each N (constructed in App. E.2), accuracy decreases toward the long-response tail for all methods (Fig. 3). ReSPO has the highest pooled accuracy in 24 of the 30 bins. At $N = 8$, it is highest in Q1–Q3 and Q5–Q9. At $N = 16$, ReSPO is highest in Q1–Q6 and Q9–Q10. The lead is largest at $N = 32$: ReSPO exceeds the highest baseline from Q1 through Q7 by 8.4, 16.0, 24.4, 21.6, 20.8, 15.1, and 8.5 pp, respectively. The shared-bin comparison therefore shows



Early peak ($t \leq 256$)

N	GRPO	GSPO	VESPO	ReSPO
Qwen3-1.7B				
8	0.169	0.158	0.161	**0.192**
16	0.156	0.127	0.151	**0.166**
32	0.118	0.106	0.127	**0.157**
Qwen3-30B-A3B				
8	**0.478**	0.416	0.414	0.468
16	0.403	0.369	0.386	**0.445**
32	0.362	0.341	0.364	**0.418**

Late mean (last 128 steps)

N	GRPO	GSPO	VESPO	ReSPO
Qwen3-1.7B				
8	$0.213_{\pm 0.017}$	$0.209_{\pm 0.015}$	$\mathbf{0.275}_{\pm 0.016}$	$\mathbf{0.286}_{\pm 0.016}$
16	$0.210_{\pm 0.007}$	$0.199_{\pm 0.009}$	$\mathbf{0.268}_{\pm 0.009}$	$\mathbf{0.264}_{\pm 0.010}$
32	$0.197_{\pm 0.005}$	$0.184_{\pm 0.003}$	$0.230_{\pm 0.004}$	$\mathbf{0.276}_{\pm 0.004}$
Qwen3-30B-A3B				
8	$0.530_{\pm 0.015}$	$0.538_{\pm 0.013}$	$0.538_{\pm 0.014}$	$\mathbf{0.595}_{\pm 0.016}$
16	$0.498_{\pm 0.010}$	$0.521_{\pm 0.008}$	$0.531_{\pm 0.012}$	$\mathbf{0.563}_{\pm 0.021}$
32	$0.504_{\pm 0.005}$	$0.493_{\pm 0.006}$	$0.523_{\pm 0.008}$	$\mathbf{0.588}_{\pm 0.007}$

Figure 2: Normalized training-score curves and summaries. Tables report the peak through step 256 and the last-128 mean. Early peaks have no error bars; late-mean subscripts are temporal standard deviations. Bold marks each row's maximum; for late means, it also marks methods whose central value and the maximum each fall within the other's range.

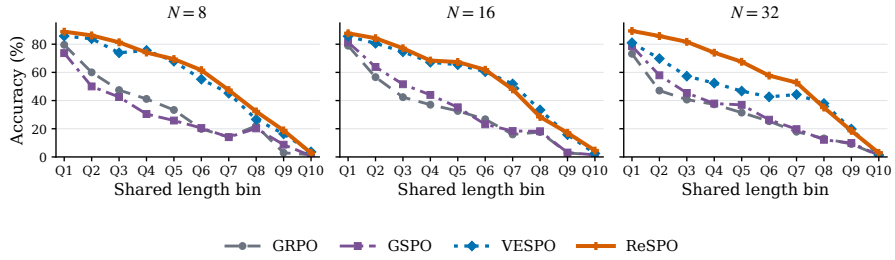


Figure 3: 1.7B accuracy across shared response-length bins. Q1–Q9 partition uncapped responses, while Q10 contains responses at the 16,384-token cap. ReSPO has the highest accuracy in 8 of 10 bins at each N.

Table 1: Final-checkpoint pass@1. Bold marks the highest mean within each model–N block.

Model	N	Method	AIME25	AIME24	AMC23	Olymp.	Minerva	MATH500	Avg.
Qwen3 1.7B	8	GRPO	0.042	0.075	0.353	0.284	0.286	0.665	$0.284_{\pm 0.013}$
		GSPO	0.062	0.077	0.402	0.304	0.272	0.658	$0.296_{\pm 0.013}$
		VESPO	0.073	0.127	0.478	**0.388**	**0.307**	**0.742**	$\mathbf{0.353}_{\pm 0.015}$
		ReSPO	**0.090**	**0.133**	**0.484**	0.368	0.289	0.726	$0.348_{\pm 0.016}$
	16	GRPO	0.071	0.083	0.431	0.303	0.278	0.663	$0.305_{\pm 0.014}$
		GSPO	0.046	0.075	0.419	0.319	0.283	0.672	$0.302_{\pm 0.013}$
		VESPO	0.065	0.106	0.464	**0.364**	**0.301**	0.712	$0.335_{\pm 0.014}$
		ReSPO	**0.094**	**0.123**	**0.467**	0.358	0.299	**0.717**	$\mathbf{0.343}_{\pm 0.015}$
	32	GRPO	0.052	0.040	0.383	0.289	0.278	0.653	$0.283_{\pm 0.012}$
		GSPO	0.046	0.054	0.381	0.299	**0.299**	0.663	$0.290_{\pm 0.013}$
		VESPO	0.081	**0.131**	0.456	0.341	0.282	0.698	$0.332_{\pm 0.015}$
		ReSPO	**0.088**	0.117	**0.488**	**0.367**	0.280	**0.729**	$\mathbf{0.345}_{\pm 0.015}$
Qwen3 30B-A3B	8	GRPO	0.152	0.248	0.738	0.511	0.436	0.852	$0.490_{\pm 0.018}$
		GSPO	0.181	0.267	0.713	**0.521**	0.439	**0.861**	$\mathbf{0.497}_{\pm 0.018}$
		VESPO	0.154	0.273	0.700	0.517	**0.454**	0.853	$0.492_{\pm 0.018}$
		ReSPO	**0.252**	**0.306**	**0.791**	0.427	0.411	0.783	$0.495_{\pm 0.019}$
	16	GRPO	0.138	0.246	0.733	0.490	0.424	0.831	$0.477_{\pm 0.017}$
		GSPO	0.156	0.265	0.700	0.493	0.452	0.848	$0.486_{\pm 0.018}$
		VESPO	0.152	0.300	0.700	**0.515**	0.436	0.854	$0.493_{\pm 0.018}$
		ReSPO	**0.213**	**0.321**	**0.792**	0.512	**0.457**	**0.857**	$\mathbf{0.525}_{\pm 0.018}$
	32	GRPO	0.123	**0.313**	0.717	0.503	0.444	0.850	$0.492_{\pm 0.017}$
		GSPO	0.146	0.231	0.664	0.475	0.423	0.835	$0.462_{\pm 0.018}$
		VESPO	0.177	0.277	0.709	0.500	0.444	**0.852**	$0.493_{\pm 0.018}$
		ReSPO	**0.202**	0.269	**0.795**	**0.510**	**0.451**	0.844	$\mathbf{0.512}_{\pm 0.018}$

that ReSPO's observed advantage persists within response-length ranges and is generally largest across short and medium responses at $N = 32$. In App. E.2, Tab. 4 gives the token boundaries and method-specific bin counts and Fig. 4 gives the complete per-benchmark results.

4.3 Discussion

Early learning from longer positive reasoning trajectories. The complete training trajectories in App. G complement the positive-branch diagnostics in App. E.3. Particularly at $N = 32$, ReSPO's response length and training score begin to separate from the baselines during the same early stage of optimization (Fig. 5). To examine this initial rapid improvement, App. E.3 focuses on the first 512 policy updates and the relationship between low-W positive responses and rapid response-length growth. Because $\log W$ accumulates token-level policy drift across a response, longer responses tend to enter the low-W tail. The diagnostic shows that ReSPO retains more coefficient mass in this tail while mean positive-response length grows faster. Together, these observations indicate that ReSPO can effectively learn from longer positive reasoning trajectories during the earliest training steps, when retaining their signal is most important for rapid improvement. On 30B, this early advantage remains 4.2 and 5.4 pp over the strongest baseline at $N = 16$ and $N = 32$, respectively, despite the soft length penalty. ReSPO also remains more accurate within shared response-length ranges (Fig. 3), indicating that its advantage extends beyond the change in length distribution.

Robustness across seeds. To assess the robustness of the observed differences at $N = 32$, we augment each primary ReSPO run used in Fig. 2 and Tab. 1 with two additional seeds. All means and cross-seed population standard deviations below are computed over the three runs. On Qwen3-1.7B, the three runs attain a last-128 mean of 27.07%, with a standard deviation of 0.54 pp. This mean exceeds VESPO (23.00%), GRPO (19.70%), and GSPO (18.41%) by 4.07, 7.37, and 8.66 pp. On Qwen3-30B-A3B, the three runs attain a last-128 mean of 57.92%, with a standard deviation of 0.95 pp. This mean exceeds VESPO (52.30%), GRPO (50.40%), and GSPO (49.28%) by 5.62, 7.52, and 8.64 pp. On the final-checkpoint evaluations averaged over AIME25, AIME24, and AMC23, the three-run mean is 41.94% with a standard deviation of 0.47 pp, exceeding VESPO (38.77%), GRPO (38.43%), and GSPO (34.70%) by 3.17, 3.51, and 7.24 pp. Across both model scales, the standard deviations are substantially smaller than ReSPO's performance gains over every baseline.

Combination with Routing Replay. Routing Replay (Ma et al., 2025), which reuses rollout-time expert-routing decisions during optimization, is orthogonal to ReSPO's control of sequence-level importance weights. In a matched 30B model ablation at $N = 16$, adding R3 increases the mean penalized training accuracy over the last 128 policy steps from 56.32% to 58.16%. In this comparison, Routing Replay improves late-stage training score and smoothness; App. E.4 and Tabs. 8 report the complete comparison.

5 Conclusion

We identify a sign-dependent gradient-allocation problem in off-policy learning: clipping suppresses under-generated positive responses on the low-importance-weight tail while allowing severely over-generated negative responses to dominate on the high-weight tail. ReSPO addresses both failures with separate positive- and negative-advantage reshaping branches derived from an α-divergence variational objective. The choice $\alpha^{(+)} > 1$ preserves a nonzero positive-branch limit as $W \to 0$, while the continuous $\alpha^{(-)} = 1$ limit suppresses both negative-branch extremes.

Across Qwen3-1.7B and Qwen3-30B-A3B at $N \in \{8, 16, 32\}$, ReSPO attains the highest observed early score in five of six settings and the highest observed late score in five of six settings, with the observed differences generally largest at $N = 32$. Positive-branch diagnostics show that ReSPO can effectively learn from long positive reasoning trajectories during the earliest training steps, even when accumulated policy drift might put them to the low-importance-weight tail; this early advantage remains influential on 30B despite its soft length penalty. ReSPO also obtains the highest observed final-checkpoint benchmark evaluation averages at $N = 16$ and $N = 32$ on both model scales, and has the highest accuracy in 24 of 30 response-length bins. The MoE ablation further shows that Routing Replay complements ReSPO by improving late-stage training score and smoothness. App. F discusses the limitations in compute budget and hyperparameter search.

AI USE DISCLOSURE

In this work, we used generative AI tools for the following tasks with required disclosure: *design or provide feedback on research methodology or experiments*. We have not used generative AI tools for the following other tasks with required disclosure: *help develop theoretical models or conceptual frameworks*; *formulate mathematical claims*; *provide critical ingredients for proving mathematical claims*; *propose or refine hypotheses*; *implement methods*; *assist in the writing of proofs*; and *interpret results*. The rest of the required disclosure tasks, *clean and reformat dataset*, *generate synthetic data sets*, *assist with translation*, and *support qualitative and thematic data analysis*, are not applicable to this work.

Additionally, we used generative AI tools for the following tasks with recommended disclosure: *create or modify scientific figures or images*; *create or edit software code*; *creation of artifacts*; *draft parts of a research paper*; *sourcing/searching for information*; *edit a research paper to improve readability*; *identify relevant literature*; *format references*; and *suggest a structure for a research paper*. Specifically, we used OpenAI Codex to inspect training and evaluation code, organize experimental records, write numerical comparisons and plotting scripts, revise the manuscript and appendix, adapt the source to the ICLR template, and prepare figures and tables.

We have reviewed all AI-assisted work. The authors checked AI-assisted text, code, mathematical expressions, visualizations, numerical claims, and references against the underlying source material, and compiled and inspected the final manuscript. The authors determined the method, experimental design, and final scientific claims. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Sec. 3.3 defines the ReSPO objective, its gradient, and the analytical conditions used to select the kernel. App. A provides the full derivations, while App. B gives the numerically stable log-space implementation and complete algorithm. The accompanying code artifact contains the ReSPO implementation and launch configurations for the dense and MoE experiments. Tabs. 2 and 3 record the shared training configuration and every method-specific hyperparameter, including model backends, parallelism, optimization settings, sampling parameters, response limits, batch construction, rollout-reuse ratios, and training horizon.

Sec. 4.1 specifies the training data, model checkpoints, baseline configurations, and evaluation protocol. App. D.1 documents the DAPO-MATH reward and length penalty, and App. D.2 describes the evaluation-generation pipeline and mathematical-equivalence verifier. Tab. 1 reports every per-benchmark result used in the main comparison. App. D.3 defines the evaluation resampling procedure and its random seed; Apps. E.1, E.3, A.5, E.2, E.4, and G provide the detailed comparisons, sequence-weight diagnostics, on-policy limit, response-length analysis, Routing-Replay comparison, and auxiliary training statistics. Together, these materials specify how to reproduce the optimization method, training runs, evaluation scores, uncertainty estimates, and reported figures.

REFERENCES

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, et al. ReSearch: Learning to reason with search for LLMs via reinforcement learning. *arXiv preprint arXiv:2503.19470*, 2025.

Imre Csiszár. I-divergence geometry of probability distributions and minimization problems. *The Annals of Probability*, pp. 146–158, 1975.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Madeleine Dwyer, Adam Sobey, and Adriane Chapman. It's not you, it's clipping: A soft trust-region via probability smoothing for LLM RL, 2025. URL https://arxiv.org/abs/2509.21282.

Chang Gao, Chujie Zheng, Xiong-Hui Chen, Kai Dang, Shixuan Liu, Bowen Yu, An Yang, Shuai Bai, Jingren Zhou, and Junyang Lin. Soft adaptive policy optimization, 2025. URL https://arxiv.org/abs/2511.20347.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024.

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.

Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. REINFORCE++: Stabilizing critic-free policy optimization with global advantage normalization. *arXiv preprint arXiv:2501.03262*, 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Hynek Kydlíček. Math-Verify: Math verification library. https://github.com/huggingface/Math-Verify, 2024. Software, accessed September 25, 2026.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857, 2022.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Jiacai Liu, Yingru Li, Yuqian Fu, Jiawei Wang, Qian Liu, and Zhuo Jiang. When speed kills stability: Demystifying RL collapse from the training-inference mismatch, September 2025. URL https://richardli.xyz/rl-collapse. Web article, accessed September 25, 2026.

Wenhan Ma, Hailin Zhang, Liang Zhao, Yifan Song, Yudong Wang, Zhifang Sui, and Fuli Luo. Stabilizing MoE reinforcement learning by aligning training and inference routers. *arXiv preprint arXiv:2510.11370*, 2025.

math-ai. AMC 2023 dataset release. https://huggingface.co/datasets/math-ai/amc23, 2023. Competition problems originally published by the Mathematical Association of America; accessed September 25, 2026.

Maxwell-Jia. AIME 2024 dataset release. https://huggingface.co/datasets/Maxwell-Jia/AIME_2024, 2024. Competition problems originally published by the Mathematical Association of America; accessed September 25, 2026.

Alberto Maria Metelli, Matteo Papini, Francesco Faccio, and Marcello Restelli. Policy optimization via importance sampling. *Advances in Neural Information Processing Systems*, 31, 2018.

Radford M Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001.

Michael Noukhovitch, Shengyi Huang, Sophie Xhonneux, Arian Hosseini, Rishabh Agarwal, and Aaron Courville. Asynchronous RLHF: Faster and more efficient off-policy RL for language models. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.18252.

OpenCompass. AIME 2025 dataset release. https://huggingface.co/datasets/opencompass/AIME2025, 2025. Competition problems originally published by the Mathematical Association of America; accessed September 25, 2026.

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.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897. PMLR, 2015.

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

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.

Guobin Shen, Chenxiao Zhao, Xiang Cheng, Lei Huang, and Xing Yu. VESPO: Variational sequence-level soft policy optimization for stable off-policy LLM training, 2026. URL https://arxiv.org/abs/2602.10693.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Zepeng Zhai, Meilin Chen, Jiaxuan Zhao, Junlang Qian, Lei Shen, and Yuan Lu. Rewards as labels: Revisiting RLVR from a classification perspective, 2026. URL https://arxiv.org/abs/2602.05630.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization, 2025. URL https://arxiv.org/abs/2507.18071.

A Derivations for the α Reshaping Kernel

This appendix collects the derivations referenced in Sec. 3.3.

A.1 α-Bregman divergence and first-order condition

For $f(x) = \frac{1}{\alpha(\alpha-1)}x^\alpha$ with $\alpha \neq 0, 1$, f is strictly convex on $\mathbb{R}_+$. The induced Bregman divergence is

$$D_f(p \,\|\, q) = \sum_{\boldsymbol{o}} \left[\frac{1}{\alpha(\alpha-1)} p(\boldsymbol{o})^\alpha - \frac{1}{\alpha(\alpha-1)} q(\boldsymbol{o})^\alpha - \frac{1}{\alpha-1} q(\boldsymbol{o})^{\alpha-1} \big(p(\boldsymbol{o}) - q(\boldsymbol{o})\big) \right]. \quad (15)$$

The objective in Eq. (6) has derivative

$$\frac{\partial}{\partial \tau(\boldsymbol{o})} \Big[(1-\beta)\, D_f(\tau \,\|\, \mu) + \beta\, D_f(\tau \,\|\, \pi) \Big] = (1-\beta)\big(f'(\tau) - f'(\mu)\big) + \beta\big(f'(\tau) - f'(\pi)\big),$$

so setting it to zero yields $f'(\tau^*) = (1-\beta)\, f'(\mu) + \beta\, f'(\pi)$. With $f'(x) = \frac{1}{\alpha-1}x^{\alpha-1}$ this reads

$$\frac{1}{\alpha-1}(\tau^*)^{\alpha-1} = \frac{1-\beta}{\alpha-1}\,\mu^{\alpha-1} + \frac{\beta}{\alpha-1}\,\pi^{\alpha-1}, \quad (16)$$

and solving for τ^* gives the power-mean form of Eq. (7). Substituting $\pi = \mu\, W$ and using $\tau = \mu\, \phi(W)$ (Eq. (4)) recovers the unconstrained kernel Eq. (8). The normalized distribution is obtained only afterward as $\bar{\tau} = \tau / Z$.

What if we optimize over a normalized distribution? To distinguish this alternative from our unnormalized measure τ, let q denote the optimization variable. For $\alpha > 1$, imposing $\sum_{\boldsymbol{o}} q(\boldsymbol{o}) = 1$ with multiplier η gives the KKT conditions for $q \geq 0$:

$$f'(q^*) = (1-\beta)\, f'(\mu) + \beta\, f'(\pi) - \eta,$$

which yield

$$q^\star(\boldsymbol{o}) = [\max(0, (1-\beta)\mu(\boldsymbol{o})^{\alpha-1} + \beta\pi(\boldsymbol{o})^{\alpha-1} - \eta(\alpha-1))]^{1/(\alpha-1)},$$

including a positivity projection. Substituting $q = \mu\phi_0(W)$ and $\pi = \mu W$ gives

$$\phi_0\big(W; \mu(\boldsymbol{o})\big) = \big[(1-\beta) + \beta W^{\alpha-1} - \eta(\alpha-1)\cdot\mu(\boldsymbol{o})^{1-\alpha}\big]_+^{1/(\alpha-1)}.$$

This loses the ratio-only form needed for a reshaping kernel. The normalization multiplier enters as $\eta(\alpha-1)\mu(\boldsymbol{o})^{1-\alpha}$ rather than as a constant, so the optimum depends on the pair $(W, \mu(\boldsymbol{o}))$ instead of on W alone. Depending on the multiplier and local behavior mass, the positivity projection can also set the weight to zero, reintroducing a hard boundary. Optimizing over the unnormalized measure avoids both complications and yields the smooth kernel in Eq. (8).

A.2 Positive branch: exponential tilt derivation

Given the unconstrained solution τ_0^* from the dual-proximity problem (Sec. 3.3.2), we impose the second-moment constraint $\sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi_0(W(\boldsymbol{o})) \leq C_1$ by solving the KL projection in Eq. (9):

$$\tau^* = \arg\min_{\tau}\; D_{\mathrm{KL}}(\tau \,\|\, \tau_0^*) \qquad \text{s.t.} \qquad \sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\, \phi_0(W(\boldsymbol{o})) \leq C_1, \quad \sum_{\boldsymbol{o}} \tau(\boldsymbol{o}) \leq C_2. \quad (17)$$

Taking the functional derivative of the Lagrangian $\mathcal{L}(\tau, \lambda, \gamma) = D_{\mathrm{KL}}(\tau \| \tau_0^*) + \lambda(\sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi_0(W(\boldsymbol{o})) - C_1) + \gamma(\sum_{\boldsymbol{o}} \tau(\boldsymbol{o}) - C_2)$ gives

$$\log \tau^*(\boldsymbol{o}) = \log \tau_0^*(\boldsymbol{o}) - \lambda\, \phi_0(W(\boldsymbol{o})) - \gamma, \quad (18)$$

and hence $\tau^*(\boldsymbol{o}) = \tau_0^*(\boldsymbol{o}) \cdot \exp(-\lambda\, \phi_0(W(\boldsymbol{o})) - \gamma)$, as stated in Eq. (10). This is the standard KL information projection under a linear constraint (Csiszár, 1975): among all measures satisfying $\sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi_0(W(\boldsymbol{o})) \leq C_1$, the one closest to τ_0^* in KL divergence is obtained by tilting the density with $\exp(-\lambda\phi_0(W))$. Substituting $\tau_0^* = \mu\, \phi_0(W)$ and extracting ϕ via $\tau^* = \mu\, \phi(W)$ gives

$$\phi(W) = \phi_0(W) \exp(-\lambda\, \phi_0(W) - \gamma). \quad (19)$$

The response-independent factor $\exp(-\gamma)$ changes only the total mass and can be absorbed into the constraint constants C_1 and C_2. Rescaling the kernel so that $\phi(1) = 1$, using $\phi_0(1) = 1$, gives Eq. (11).

Why use the proxy ϕ_0? The natural variance-control target is $\mathbb{E}_\mu[\phi(W)^2] \le C$, which controls the second-moment term in the denominator of the standard importance-weight effective sample size. Since $\tau = \mu\,\phi$, this is equivalent to $\sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi(W(\boldsymbol{o})) \le C$. However, the final kernel ϕ is itself the unknown we are solving for: writing $\phi = \phi_0 \cdot h$ where h is the tilt function, the constraint becomes $\sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi_0(W(\boldsymbol{o}))h(W(\boldsymbol{o})) \le C'$, which is *not* linear in τ (since h depends on τ through the measure-change identity). The Gibbs variational principle requires a constraint of the form $\sum_{\boldsymbol{o}} \tau(\boldsymbol{o})g(W(\boldsymbol{o})) \le C$ for a *fixed* function g.

We therefore replace ϕ with its known proxy ϕ_0 (the unconstrained kernel from Step 1) and constrain $\sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi_0(W(\boldsymbol{o})) \le C$. This is linear in τ, so the KL projection yields the clean exponential tilt $\exp(-\lambda\phi_0(W))$. Under this proxy, we prove that the derived τ satisfies bounded moments. Since the tilt satisfies $h(W) = \exp(\lambda(1-\phi_0(W))) \le \exp(\lambda)$, we have

$$\mathbb{E}_\mu[\phi(W)^2] = \sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi(W(\boldsymbol{o})) \le \exp(\lambda) \sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi_0(W(\boldsymbol{o})). \tag{20}$$

Thus, bounding $\sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi_0(W(\boldsymbol{o}))$ ensures $\mathbb{E}_\mu[\phi^2]$ is also bounded.

Why KL and not α-divergence for the projection? One might ask why we use the KL divergence in the projection step (Step 2) rather than the same α-divergence used in Step 1. Replacing the KL with $D_f(\tau\|\tau_0^*)$ for $f(x) = \frac{1}{\alpha(\alpha-1)}x^\alpha$ and using the linear constraint $\sum_{\boldsymbol{o}} \tau(\boldsymbol{o})\phi_0(W(\boldsymbol{o})) \le C$, the first-order condition becomes

$$\frac{1}{\alpha-1}\tau^{*\alpha-1} = \frac{1}{\alpha-1}\tau_0^{*\alpha-1} - \lambda\,\phi_0(W). \tag{21}$$

Extracting ϕ yields the constraint term inside the power-mean bracket:

$$\phi(W) \propto \left[\phi_0(W)^{\alpha-1} - \lambda(\alpha-1)\,\mu^{1-\alpha}\,\phi_0(W)\right]_+^{1/(\alpha-1)}. \tag{22}$$

The factor $\mu^{1-\alpha}$ makes this expression depend on both W and the behavior mass. For sufficiently small μ, the bracket can become negative and activate the $[\cdot]_+$ projection; when $1 < \alpha < 2$, the subtracted term also grows faster than the first term as $\phi_0(W)$ increases. The resulting kernel can therefore acquire a hard cutoff rather than the smooth, ratio-only decay we seek.

The key distinction is geometric. The KL first-order condition is additive in *log-space* ($\log\tau^* = \log\tau_0^* - \lambda\phi_0 - \gamma$), so the constraint enters as a *multiplicative* exponential factor on τ_0^*. In the power-Bregman alternative above, the constraint instead enters inside the power-mean bracket, introducing behavior-mass dependence and a possible hard boundary. The two-step procedure therefore uses each divergence where it is most natural: α-divergence for reshaping (power-mean interpolation between μ and π), and KL for variance control (multiplicative exponential tilt).

A.3 Hyperparameter derivations

Positive branch: deriving $\lambda^{(+)}$. For $\phi(W) = \phi_0(W)\cdot\exp(\lambda(1-\phi_0(W)))$, differentiating $\log\phi$ gives

$$\frac{d\log\phi}{dW} = \phi_0'(W)\left(\frac{1}{\phi_0(W)} - \lambda\right). \tag{23}$$

At $\alpha^{(+)} = 2$, the unconstrained kernel is $\phi_0^{(+)}(W) = (1-\beta^{(+)}) + \beta^{(+)}W$, so $\phi_0'(W) = \beta^{(+)} > 0$ is constant. At $W = 0$ we have $\phi_0(0) = 1-\beta^{(+)}$, so

$$\left.\frac{d\log\phi^{(+)}}{dW}\right|_{W=0} = \beta^{(+)}\left(\frac{1}{1-\beta^{(+)}} - \lambda^{(+)}\right).$$

We additionally require the full kernel to be stationary at the origin, $\phi^{(+)\prime}(0) = 0$. Because $\phi^{(+)}(0) > 0$, this is equivalent to setting the log derivative above to zero, which gives

$$\lambda^{(+)} = \frac{1}{1-\beta^{(+)}}. \tag{24}$$

For any $W > 0$, $\phi_0(W) > 1-\beta^{(+)} = 1/\lambda^{(+)}$, so $1/\phi_0(W) < \lambda^{(+)}$ and $\frac{d\log\phi^{(+)}}{dW} < 0$: the kernel is strictly decreasing on its feasible domain $(0,\infty)$. Moreover, because $\alpha^{(+)} = 2$ makes ϕ_0 affine,

the closed form has a differentiable extension to every $W \in \mathbb{R}$, even though importance weights themselves satisfy $W \geq 0$. On this extension,

$$\phi^{(+)\prime}(W) = -\frac{(\beta^{(+)})^2}{1-\beta^{(+)}}\, W \exp\Big(\lambda^{(+)}(1-\phi_0^{(+)}(W))\Big),$$

which is positive for $W < 0$, zero at $W = 0$, and negative for $W > 0$. Thus $W = 0$ is the unique global maximizer both on the real-valued extension and on the feasible domain, with

$$\phi^{(+)}(0) = (1-\beta^{(+)})\, e^{\,\beta^{(+)}/(1-\beta^{(+)})}. \tag{25}$$

With $\beta^{(+)} = 0.5$, this gives $\lambda^{(+)} = 2$ and $\phi^{(+)}(0) = \frac{1}{2}e^1 \approx 1.36$.

Why $\alpha = 2$ is the natural choice. We seek a non-degenerate local condition at the origin: $\phi_0'(0)$ should be finite and strictly positive, so requiring $\phi^{(+)\prime}(0) = 0$ uniquely determines $\lambda^{(+)}$. For $1 < \alpha < 2$, $\phi_0'(W) \propto W^{\alpha-2}$ is singular as $W \to 0^+$; for $\alpha > 2$, $\phi_0'(0) = 0$, so stationarity of the full kernel holds for any λ and does not determine the tilt. At $\alpha = 2$, $\phi_0'(0) = \beta^{(+)}$ is finite and strictly positive, and ϕ_0 is linear in W. Thus $\alpha^{(+)} = 2$ uniquely combines an affine unconstrained kernel, a non-degenerate zero-derivative condition that fixes $\lambda^{(+)}$, and the real-line maximum at $W = 0$ shown above.

Negative branch: matching the derivative at $W = 1$. With $\alpha^{(-)} = 1$ and $\beta^{(-)} = 0.5$, the continuous-limit kernel is

$$\phi_0^{(-)}(W) = W^{\beta^{(-)}} = \sqrt{W}.$$

We share $\beta = 0.5$ across the two branches so that both unconstrained kernels interpolate symmetrically between the behavior and target policies. The remaining parameter $\lambda^{(-)}$ is fixed by matching the derivatives of the full kernels at the on-policy point. For the positive branch,

$$\phi^{(+)}(W) = \frac{1+W}{2}e^{1-W}, \qquad \frac{d\phi^{(+)}}{dW} = -\frac{W}{2}e^{1-W},$$

and hence $\phi^{(+)\prime}(1) = -\frac{1}{2}$. For the negative branch,

$$\phi^{(-)}(W) = W^{\beta^{(-)}} \exp\Big(\lambda^{(-)}(1-W^{\beta^{(-)}})\Big).$$

Since $\phi^{(-)}(1) = 1$, its derivative at $W = 1$ is

$$\phi^{(-)\prime}(1) = \beta^{(-)}\big(1-\lambda^{(-)}\big). \tag{26}$$

Requiring $\phi^{(-)\prime}(1) = \phi^{(+)\prime}(1)$ and substituting $\beta^{(-)} = 0.5$ gives

$$\frac{1}{2}\big(1-\lambda^{(-)}\big) = -\frac{1}{2} \qquad \Longrightarrow \qquad \lambda^{(-)} = 2.$$

Thus the positive and negative kernels have identical value and first derivative at $W = 1$, making their local response to off-policy drift symmetric to first order.

A.4 Continuous limit as $\alpha \to 1$

Taylor-expanding $W^{\alpha-1}$ around $\alpha = 1$ gives $W^{\alpha-1} = e^{(\alpha-1)\log W} \approx 1 + (\alpha-1)\log W$ to first order in $\alpha - 1$. Substituting this expansion into the α factor of Eq. (11) gives

$$\phi_0(W) = \big[1+\beta(W^{\alpha-1}-1)\big]^{1/(\alpha-1)} \approx \big[1+\beta(\alpha-1)\log W\big]^{1/(\alpha-1)},$$

which tends to W^β as $\alpha \to 1$. The full kernel therefore reduces to $W^\beta \exp(\lambda(1-W^\beta))$, which is the form used by ReSPO's negative branch. For comparison, VESPO's gamma-IS kernel (Shen et al., 2026) is $W^\beta \exp(\lambda(1-W))$: the exponential tilt penalizes the raw importance weight W rather than the reshaped weight $\phi_0(W) = W^\beta$.

A.5 On-policy limit

In the strict on-policy limit, $\pi_\theta = \pi_{\text{old}}$ and every sample has $W = 1$. Setting $N = 1$ removes staleness from rollout reuse and approximates this condition. At $W = 1$, the two kernels used in our experiments satisfy

$$\phi^{(+)}(1) = \frac{1+1}{2}\exp(1-1) = 1, \qquad \phi^{(-)}(1) = \sqrt{1}\exp\Big(2(1-\sqrt{1})\Big) = 1. \tag{27}$$

Consequently, ReSPO assigns unit reshaping weight in the exact on-policy limit: its per-sequence multiplier is one. This recovers standard on-policy training, in which no importance-ratio weighting is applied.

A.6 Length-dependent bias from sequence normalization

Shen et al. (2026) identify a length-dependent bias in the geometric-mean normalization used by GSPO. Let $T = |\boldsymbol{o}|$, let $\ell_t = \log w_t$ be the per-token log-ratio, and write $\bar{\ell} = T^{-1}\sum_{t=1}^{T}\ell_t$. The exact sequence importance weight and the GSPO weight are

$$W = \exp(T\bar{\ell}), \qquad s = W^{1/T} = \exp(\bar{\ell}). \tag{28}$$

The exponent $1/T$ controls variance, but it also changes the implied measure correction with response length. For a fixed length T, weighting samples from μ by s induces the proposal

$$Q_T(\boldsymbol{o}) \propto \mu(\boldsymbol{o})s(\boldsymbol{o}) = \mu(\boldsymbol{o})^{1-1/T}\pi(\boldsymbol{o})^{1/T}. \tag{29}$$

Thus the target-policy exponent decreases as T grows, and Q_T is increasingly tempered toward the behavior policy μ. In this sense, the strength of the off-policy correction depends on length rather than only on the likelihood shift between π and μ.

The normalization also conflates sequences with different total likelihood shifts. If two responses have lengths $T_1 \neq T_2$ but the same average log-ratio $\bar{\ell}$, GSPO assigns both the same weight, $s_1 = s_2 = e^{\bar{\ell}}$. Their true sequence weights are instead $W_1 = e^{T_1\bar{\ell}}$ and $W_2 = e^{T_2\bar{\ell}}$, with

$$\frac{W_2}{W_1} = \exp\big((T_2 - T_1)\bar{\ell}\big). \tag{30}$$

Consequently, a short response and a much longer response can receive the same GSPO weight even when their sequence-level distribution shifts differ exponentially. Clipping can further truncate these weights, but the length dependence is already introduced by the $1/T$ normalization.

ReSPO instead applies a fixed kernel $\phi(W)$ to the unnormalized sequence weight. Like any non-identity reshaping, this intentionally trades bias for variance. The same map ϕ is used at every response length, so length enters through the actual sequence likelihood ratio W without an additional length-dependent tempering exponent.

B Log-Space Implementation of the Two-Branch Kernel

The reshaping weight $\phi(W_i; \hat{A}_i)$ is evaluated per-sequence from the sequence-level log-ratio

$$\log W_i = \sum_{t=1}^{|\boldsymbol{o}_i|}\log w_{i,t}, \qquad w_{i,t} = \pi_\theta(o_{i,t} \mid \boldsymbol{q}, \boldsymbol{o}_{i,<t}) \,/\, \pi_{\text{old}}(o_{i,t} \mid \boldsymbol{q}, \boldsymbol{o}_{i,<t}).$$

We clamp $\log W_i$ to $[-20, 20]$ before any non-linear operation and detach it from the gradient graph, since ϕ plays the role of a gradient scaling coefficient. The branch computations use numerically stable forms. The tail limits analyzed above refer to the unclamped analytical kernels; the implementation evaluates a finite-range approximation to avoid numerical overflow. We describe the two branches separately; which one applies to a given sequence is selected by the sign of $\hat{A}_i$.

Positive branch ($\hat{A}_i \geq 0$). With $\alpha^{(+)} = 2$, $\beta^{(+)} = 0.5$, $\lambda^{(+)} = 2$, the kernel simplifies to

$$\phi^{(+)}(W_i) = \frac{1+W_i}{2}\, e^{1-W_i}. \tag{31}$$

No log-space tricks are needed: $(1 + W_i)/2 \geq 1/2 > 0$ for all $W_i \geq 0$.

Algorithm 1 ReSPO: Reshaped Sequence Policy Optimization

Require: Policy π_θ, prompts $\{\boldsymbol{q}\}$, group size G
Require: $\alpha^{(+)}$=2, $\alpha^{(-)}$=1, $\beta^{(+)}$=$\beta^{(-)}$=0.5, $\lambda^{(+)}$=2, $\lambda^{(-)}$=2
for each training iteration **do**
  Sample a batch of prompts; for each $\boldsymbol{q}$, sample G responses $\{\boldsymbol{o}_i\}_{i=1}^{G} \sim \pi_{\text{old}}$
  Compute rewards $R(\boldsymbol{q}, \boldsymbol{o}_i)$ and GRPO group-normalized advantages $\hat{A}_i$
  for each response $\boldsymbol{o}_i$ **do**
    $\log W_i \leftarrow \sum_t \log[\pi_\theta(o_{i,t} \mid \boldsymbol{q}, \boldsymbol{o}_{i,<t}) / \pi_{\text{old}}(o_{i,t} \mid \boldsymbol{q}, \boldsymbol{o}_{i,<t})]$; clamp to $[-20, 20]$
    if $\hat{A}_i \geq 0$ **then**
      {positive branch}
      $\phi_i \leftarrow (1 + e^{\log W_i})/2 \cdot \exp(1 - e^{\log W_i})$
    else
      {negative branch}
      $\ell_i \leftarrow \beta^{(-)} \log W_i$
      $\phi_i \leftarrow \exp(\ell_i + \lambda^{(-)}(1 - e^{\ell_i}))$
    end if
    $\phi_i \leftarrow \phi_i$.detach()
  end for
  $\nabla_\theta \mathcal{J} \leftarrow \frac{1}{G} \sum_i \phi_i \cdot \hat{A}_i \cdot \sum_t \nabla_\theta \log \pi_\theta(o_{i,t} \mid \boldsymbol{q}, \boldsymbol{o}_{i,<t})$
  Update θ by gradient ascent using $\nabla_\theta \mathcal{J}$
end for

Negative branch ($\hat{A}_i < 0$). With $\alpha^{(-)} = 1$, the continuous-limit kernel has $\phi_0^{(-)}(W_i) = W_i^{\beta^{(-)}}$. We therefore compute it directly in log-space:

$$\log \phi_0^{(-)}(W_i) = \beta^{(-)} \log W_i. \tag{32}$$

The full kernel adds the exponential tilt:

$$\log \phi^{(-)}(W_i) = \log \phi_0^{(-)}(W_i) + \lambda^{(-)} \Big(1 - \exp\Big(\log \phi_0^{(-)}(W_i)\Big)\Big). \tag{33}$$

Combining the branches. Given the two log-space scalars, we compute $\phi_i = \exp(\log \phi_i)$ on the branch selected by $\text{sign}(\hat{A}_i)$, detach the result, and multiply it into the token-level loss as summarized in Alg. 1. In practice we compute both branches for every sequence and select the applicable one with a sign mask, which is simpler on a GPU than using a data-dependent branch and has negligible overhead since both kernels are $O(1)$ per sequence.

C Detailed Hyperparameters

Tab. 2 lists the training hyperparameters for both experimental settings. Tab. 3 lists the method-specific hyperparameters for each baseline and ReSPO.

Group-normalized advantages. The implementation computes GRPO advantages using the sample standard deviation with $\epsilon_A = 10^{-6}$ added to the denominator. When all rewards in a group are identical, the centered numerators are zero, so all advantages are zero rather than undefined.

Why this normalization? For a fixed prompt, Eq. (3) is an expectation over responses sampled from $\mu = \pi_{\text{old}}$. Given G such responses, its standard Monte Carlo estimator is therefore the sequence mean $G^{-1} \sum_{i=1}^{G}$ used in Eq. (14) and Alg. 1; the outer expectation over prompts supplies the corresponding prompt average. The factor $1/G$ averages sampled sequences and does not normalize the unnormalized measure τ or its weights, so replacing it by $1/\sum_i \phi_i$ would instead produce a different, self-normalized estimator. In the reported experiments, we use the batch-wide token-mean normalization $1/\sum_i |\boldsymbol{o}_i|$ shown in Tab. 2 for improved training stability. Relative to the sequence mean, this applies a common batch-level scale to all response contributions and therefore preserves their relative ReSPO weights within each batch.

Table 2: Training hyperparameters across experimental settings.

Hyperparameter	**Qwen3-1.7B**	**Qwen3-30B-A3B**
Model & infrastructure		
Training backend	FSDP	Megatron
Tensor parallelism (train)	–	2
Expert parallelism (train)	–	8
Tensor parallelism (rollout)	2	4
Data		
Training set	DAPO-MATH-17k	DAPO-MATH-17k
Max prompt length	1,024	1,024
Max response length	15,360	8,192
Optimization		
Optimizer	AdamW	Adam
Learning rate	1×10^{-6}	
Warmup ratio	5%	
Weight decay	0.1	
Gradient clip norm	1.0	
KL penalty	None	
Rollout		
Rollout engine	vLLM (async)	
Responses per prompt (G)	8	
Temperature (train)	1.0	
Top-p (train)	1.0	
Batching & budget		
Mini-batch size (M)	32	
Rollout-reuse ratio (N)	$\{8, 16, 32\}$	
Global batch size	$N \times 32$	
Total policy updates	1024	
Advantage estimator	GRPO (group-normalized)	
Aggregation	token-mean	

Table 3: Method-specific hyperparameters. All methods share the training configuration in Tab. 2.

Method	Hyperparameter	Value
GRPO	Clip ratio ε	0.2
GSPO	Clip ratio low ε_{low}	3×10^{-4}
	Clip ratio high $\varepsilon_{\text{high}}$	4×10^{-4}
VESPO	$\beta^{(+)}$	2
	$\lambda^{(+)}$	3
	$\beta^{(-)}$	3
	$\lambda^{(-)}$	2
ReSPO	$\alpha^{(+)}$	2
	$\beta^{(+)}$	0.5
	$\lambda^{(+)}$	2
	$\alpha^{(-)}$	1
	$\beta^{(-)}$	0.5
	$\lambda^{(-)}$	2

The VESPO values in Tab. 3 are the best-performing sign-specific configuration reported in the hyperparameter study of the original VESPO paper (Shen et al., 2026), and we use them directly. ReSPO uses the analytically motivated parameters derived in App. A.

D EVALUATION IMPLEMENTATION

In this section, we describe the training reward implementation and the evaluation scoring procedure.

D.1 DAPO-MATH TRAINING REWARD IMPLEMENTATION

In the reported runs, training examples carry the data-source key `math_dapo`, which selects the strict-box verifier rather than the Math-Verify evaluator used for Tab. 1. The relevant logic is summarized below; names are simplified for clarity.

```
# Training-reward pseudocode
if use_strict_box_reward:
    strict_score = strict_box_verify(response, ground_truth)
    correct = (strict_score == +1)
    reward = +1.0 if correct else -1.0

if overlong_penalty_enabled:
    reward -= max(0.0, (response_length - 4096) / 4096)
```

The strict-box verifier returns $+1$ or -1 after comparing the ground truth with the final boxed answer in the response suffix. Its wrapper explicitly checks for $+1$ before mapping the result to the floating-point reward $+1.0$ or -1.0. For the reported 30B experiments, the DAPO reward manager additionally applies the configured overlong-buffer term. With `max_resp_len`$=$ 8,192, buffer length 4,096, and penalty factor 1, the additive penalty is $-\max\{0, (L-4{,}096)/4{,}096\}$ for response length L. It is zero through 4,096 tokens and reaches -1 at the response limit. The dense-model experiments disable this term and use only the signed strict-box score.

Range of the penalized training accuracy. For the 1.7B experiments, the signed strict-box score is in $\{-1, 1\}$, so the affine transformation $(\texttt{score}+1)/2$ is in $\{0, 1\}$. For the 30B experiments, the total score also includes the additive penalty in $[-1, 0]$, so the transformed quantity can range from -0.5 to 1. We retain the name *penalized training accuracy* because the quantity reduces to ordinary training accuracy when the penalty is zero and uses the same affine scale; a negative value denotes an incorrect overlong response with an additional length penalty, not a negative probability.

D.2 MATH-VERIFY SCORING IMPLEMENTATION

The evaluator tags all six benchmarks as `MATH-lighteval`, which the `verl` score dispatcher routes to Math-Verify rather than the DAPO strict-box verifier. The evaluation logic is summarized as follows.

```
# Evaluation-scoring pseudocode
gold = parse_latex(box(ground_truth))
predictions = parse(completion, modes=[expression, latex])
correct = any(
    mathematically_equivalent(g, p)
    for g in gold for p in predictions
)
score = 1.0 if correct else 0.0
```

Parsing the completion in both modes lets the verifier accept free-form final answers in addition to strict `\boxed{}` answers. If either side cannot be parsed, no pair is mathematically equivalent, or the scoring worker exceeds its 30-second timeout, the completion receives zero credit. The pass@1 reported in Tab. 1 is the mean of this binary correctness indicator over samples and problems. The auxiliary majority-vote metric separately uses the DAPO boxed-answer utilities.

D.3 BOOTSTRAP EVALUATION UNCERTAINTY

The error bars in Tab. 1 use a cluster bootstrap over evaluation problems, using $B = 10{,}000$ replicates with random seed 0. For problem i in benchmark j, let $y_{jik} \in \{0, 1\}$ denote the correctness of

sampled completion k. We first average the n_{ji} completions for that problem,

$$z_{ji} = \frac{1}{n_{ji}} \sum_{k=1}^{n_{ji}} y_{jik}, \qquad \widehat{a}_j = \frac{1}{m_j} \sum_{i=1}^{m_j} z_{ji}, \qquad \widehat{A} = \frac{1}{6} \sum_{j=1}^{6} \widehat{a}_j, \tag{34}$$

where m_j is the number of problems in benchmark j: 30 each for AIME25 and AIME24, 40 for AMC23, 674 for OlympiadBench, 272 for MinervaMath, and 500 for MATH-500. The reported macro average assigns equal weight to each benchmark.

For bootstrap replicate b, we independently draw m_j problem indices with replacement within each benchmark, keeping all completions for a sampled problem together. If these indices are $I_{j1}^{(b)}, \ldots, I_{jm_j}^{(b)}$, we recompute

$$\widehat{a}_j^{(b)} = \frac{1}{m_j} \sum_{\ell=1}^{m_j} z_{jI_{j\ell}^{(b)}}, \qquad \widehat{A}^{(b)} = \frac{1}{6} \sum_{j=1}^{6} \widehat{a}_j^{(b)}. \tag{35}$$

The subscripted error bar is the bootstrap standard error,

$$\mathrm{SE}_{\mathrm{boot}} = \sqrt{\frac{1}{B-1} \sum_{b=1}^{B} \left(\widehat{A}^{(b)} - \overline{\widehat{A}} \right)^2}, \tag{36}$$

so an entry $\widehat{A}_{\pm \mathrm{SE}_{\mathrm{boot}}}$ reports one bootstrap standard error. For method contrasts, the same resampled problem indices are applied to both methods before taking the difference, giving a paired cluster bootstrap; percentile quantiles of that difference distribution form a confidence interval. These calculations quantify finite-evaluation sampling uncertainty conditional on the fixed final checkpoint and its stored generations.

E Additional Robustness Checks

E.1 Detailed training and evaluation comparisons

Within the first 256 policy updates, ReSPO has the highest peak normalized training score in five of six model–N settings. The exception is 30B at $N = 8$, where GRPO is higher by 1.0 pp. ReSPO's differences from the next-highest method are 2.3, 1.0, and 3.0 pp on 1.7B for $N \in \{8, 16, 32\}$, respectively, and 4.2 and 5.4 pp on 30B at $N = 16, 32$. Its curves remain above the baselines through most of the subsequent trajectory.

Over the last 128 policy steps, ReSPO has the highest mean in five of six settings; on 1.7B at $N = 16$, it is 0.4 pp below VESPO, and each central value lies within the other's temporal standard-deviation range. It exceeds the highest clipped baseline in all six settings by 4.2–8.4 pp, with non-overlapping temporal ranges. From $N = 8$ to $N = 32$, its mean decreases by 1.0 pp on 1.7B and 0.7 pp on 30B, compared with 2.5 and 4.5 pp for GSPO and 4.5 and 1.5 pp for VESPO. At $N = 32$, ReSPO exceeds the highest clipped baseline by 7.9 and 8.4 pp, and VESPO by 4.6 and 6.5 pp. Across the early and late summaries, ReSPO is highest in 10 of 12 settings. It is 1.0 pp below the early maximum on 30B at $N = 8$ and 0.4 pp below the late maximum on 1.7B at $N = 16$, where the temporal ranges overlap.

At $N = 8$, the evaluation macro averages are similar relative to their bootstrap uncertainty: ReSPO is 0.5 pp below VESPO on 1.7B and 0.2 pp below GSPO on 30B. For $N \in \{16, 32\}$, ReSPO has the highest observed mean in all four model–N settings, exceeding the highest baseline by 0.8 and 1.3 pp on 1.7B and by 3.2 and 1.9 pp on 30B. It has the highest value in 6, 9, and 8 of the 12 model–benchmark pairs for $N \in \{8, 16, 32\}$, respectively.

The largest observed differences occur on difficult competition benchmarks. On AIME25, ReSPO exceeds the highest baseline in all six model–N settings by 1.7, 2.3, and 0.7 pp on 1.7B and by 7.1, 5.7, and 2.5 pp on 30B as N increases. It also has the highest AMC23 value in all six settings and the highest AIME24 value in four of six. Averaged over these three benchmarks on 30B, its differences from the highest baseline are 6.3, 5.8, and 3.4 pp for $N \in \{8, 16, 32\}$, respectively. At $N = 8$, the other three benchmarks offset this difference in the macro average; at larger N, ReSPO also attains the highest observed overall mean.

E.2 Accuracy across response-length ranges

We analyze the stored Qwen3-1.7B evaluation generations underlying Tab. 1 for all four methods, all six benchmarks, and $N \in \{8, 16, 32\}$. Each method contributes 7,384 responses per value of N, and response length is the number of Qwen3-1.7B tokenizer tokens in the generated text. For each value of N, we pool the observed lengths to construct one set of boundaries shared by every method and benchmark. The generation cap produces a point mass at 16,384 tokens, so Q10 contains exactly the capped responses; Q1–Q9 divide the uncapped pooled responses into nine equal-frequency intervals. This tie-preserving construction keeps identical capped lengths in the same bin. Accuracy within a bin is the fraction of its responses accepted by the evaluation verifier.

Tab. 4 reports the resulting token upper bounds and the pooled number of responses from each method in every bin. These counts expose the method-dependent response-length distributions used to compute the pooled accuracies below.

Table 4: Shared response-length-bin upper bounds (tokens) and pooled response counts for Qwen3-1.7B. Counts aggregate the six evaluation benchmarks; each method contributes 7,384 responses for each N. Q10 contains responses at the 16,384-token cap.

N	Entry	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
8	Upper bound	355	480	611	761	954	1,263	2,009	4,404	16,383	16,384
	GRPO count	1,043	908	820	855	816	809	573	163	102	1,295
	GSPO count	1,253	1,087	1,131	1,074	917	775	420	79	46	602
	VESPO count	386	466	456	475	539	671	1,037	1,504	1,376	474
	ReSPO count	207	429	482	482	605	630	856	1,140	1,360	1,193
16	Upper bound	366	481	604	736	907	1,168	1,785	4,252	16,383	16,384
	GRPO count	997	989	1,009	1,009	916	797	601	136	71	859
	GSPO count	873	869	857	952	883	873	734	247	64	1,032
	VESPO count	438	470	481	445	527	618	822	1,139	841	1,603
	ReSPO count	506	483	450	405	472	506	640	1,281	1,827	814
32	Upper bound	371	483	596	711	856	1,057	1,421	2,732	16,383	16,384
	GRPO count	1,172	1,013	1,000	906	849	698	542	289	109	806
	GSPO count	949	908	874	905	831	832	730	386	81	888
	VESPO count	516	556	555	605	693	762	795	973	975	954
	ReSPO count	255	394	453	469	499	589	805	1,230	1,714	976

Fig. 4(a) shows the pooled curves. Accuracy generally decreases toward the long tail, and the ReSPO curve decreases monotonically for all three values of N. ReSPO has the highest accuracy in eight of ten bins at each N. Its separation is widest at $N = 32$: the differences from the highest baseline are 8.4, 16.0, 24.4, 21.6, 20.8, 15.1, and 8.5 pp from Q1 through Q7. VESPO is higher in Q8 and Q9 by 2.9 and 1.2 pp, while ReSPO is higher in the cap bin by 1.5 pp. At $N = 8$ and $N = 16$, the two reshaped methods are closer, but ReSPO still has the highest accuracy in most short, medium, and uncapped long-response bins.

Fig. 4(b,c) separates the six benchmarks. The middle-length advantage at $N = 32$ appears broadly on AMC 2023, OlympiadBench, and MATH-500. The AIME panels have fewer observations and lower base accuracy, producing more irregular bin-level curves. Across benchmarks, capped responses are generally among the least accurate groups.

E.3 Positive-response weight and length dynamics

Sec. 4.3 suggests that ReSPO learns earlier from longer positive responses. To isolate this initial rapid-improvement phase, the diagnostic runs cover the first 16 rollout batches, or 512 of the 1024 policy updates used in the complete experiments. Within this window, we ask two questions: when a positive response moves into the low-W tail, how much learning coefficient does it retain, and how does this allocation evolve with response length? Since $W = \pi_\theta(\boldsymbol{o})/\pi_{\text{old}}(\boldsymbol{o})$, $W < 1$ means that the response has become less likely under the updated policy. For $\hat{A} > 0$, this is precisely a response whose probability the update should increase.

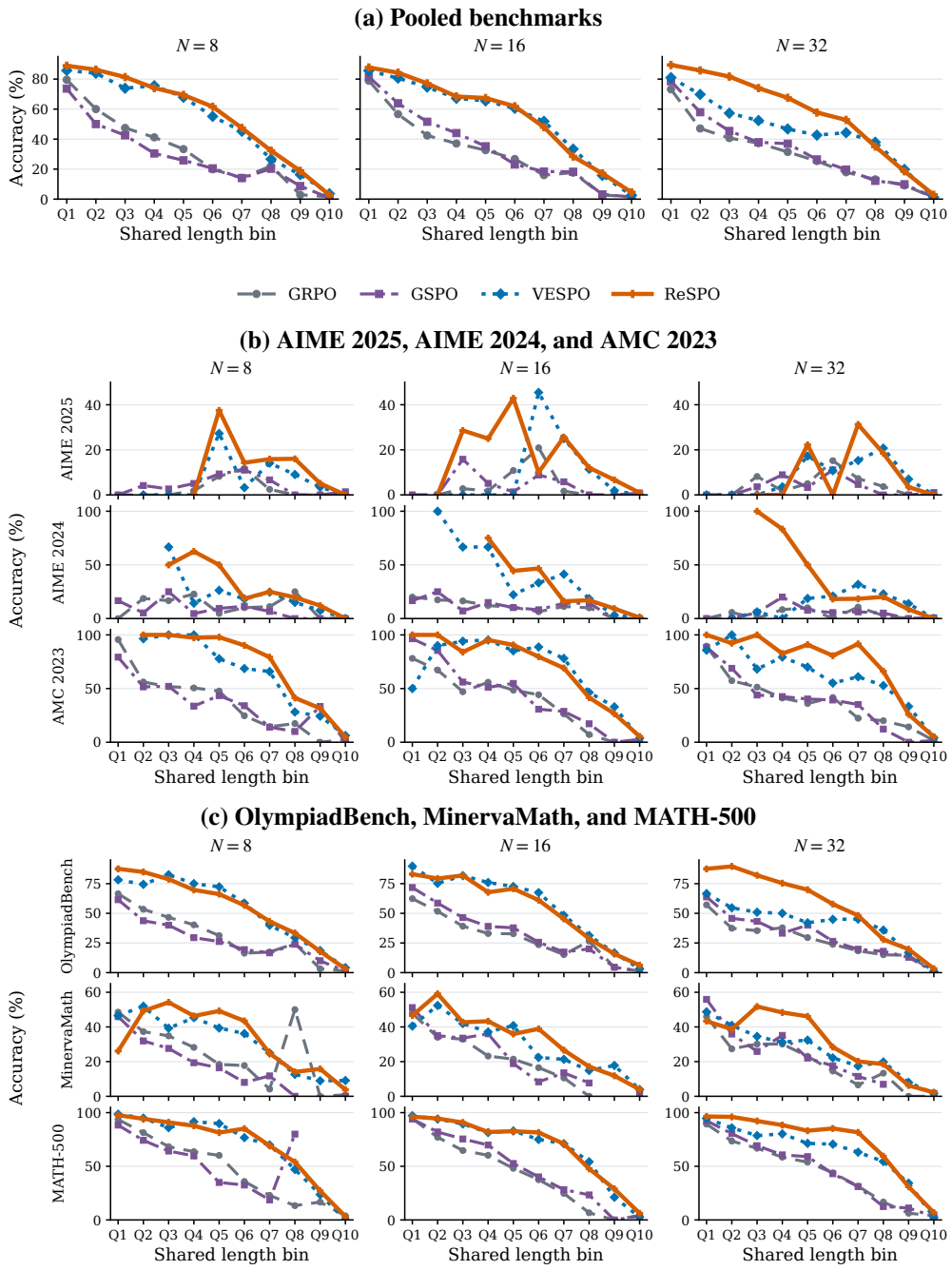


Figure 4: Qwen3-1.7B accuracy by shared response-length bin. Q1–Q9 partition uncapped responses using boundaries shared across methods, while Q10 contains responses at the 16,384-token cap. Columns correspond to $N \in \{8, 16, 32\}$; gaps in the benchmark panels indicate empty method–benchmark bins.

We measure W directly in matched Qwen3-1.7B-Base runs at $N = 32$ with a 16,384-token response limit. ReSPO and VESPO use the same DAPO-MATH data, optimizer settings, GRPO advantages, and token-mean aggregation. Each run uses this first-half diagnostic window. For every sequence, we record its advantage sign and $\log W = \sum_t (\log \pi_\theta - \log \pi_{\text{old}})$ in fixed bins. Because this sum accumulates policy drift token by token, longer responses provide more opportunities for negative drift to build up and therefore tend to occur in the low-W tail. For a positive-branch bin B, we report

$$p_B^{(+)} = \frac{\sum_{i:\,\hat{A}_i>0} \mathbf{1}\{\log W_i \in B\}}{\sum_{i:\,\hat{A}_i>0} 1}, \qquad m_B^{(+)} = \frac{\sum_{i:\,\hat{A}_i>0} |\hat{A}_i| \phi^{(+)}(W_i) \mathbf{1}\{\log W_i \in B\}}{\sum_{i:\,\hat{A}_i>0} |\hat{A}_i| \phi^{(+)}(W_i)}.$$

Here, $p_B^{(+)}$ is the fraction of positive responses in the bin, whereas $m_B^{(+)}$ is the fraction of the total positive-branch loss coefficient $|\hat{A}|\phi^{(+)}(W)$ assigned to that bin. Their ratio $m_B^{(+)}/p_B^{(+)}$ is the amplification: a value above 1 means that each response in the bin receives more coefficient mass than the positive-branch average, while a value below 1 indicates suppression. The diagnostic logs each numerator and denominator under the same distributed reduction, so these ratios remain directly comparable.

Table 5: Positive-branch deep-tail allocation on Qwen3-1.7B-Base at $N = 32$. Values are means over rollout batches 9–16; amplification is the coefficient-mass share divided by the sequence share.

Statistic	ReSPO	VESPO
Positive responses with $\log W < -1$	22.8%	16.7%
Coefficient mass on $\log W < -1$	38.5%	15.5%
Amplification on $\log W < -1$	1.69×	0.93×
Positive responses in $-5 < \log W \le -2$	7.4%	3.7%
Coefficient mass in $-5 < \log W \le -2$	14.8%	1.1%
Amplification in $-5 < \log W \le -2$	2.00×	0.30×

The distinction between response share and coefficient-mass share isolates where the methods differ. In $\log W < -1$, ReSPO observes 22.8% of its positive responses but assigns them 38.5% of its positive-branch coefficient mass, giving $1.69\times$ amplification. VESPO observes a similar 16.7% response share but assigns the region only 15.5% of its mass. The contrast sharpens in $-5 < \log W \le -2$: ReSPO amplifies the bin to $2.00\times$ its response share, whereas VESPO suppresses it to $0.30\times$. At the final rollout batch, ReSPO places 58.5% of its positive-branch mass in $-5 < \log W \le -0.5$; VESPO instead places its largest share, 35.6%, in the region immediately below $W = 1$, $-0.5 < \log W \le 0$.

Table 6: Mean $\log W$ and response length on the positive branch during the $N = 32$ diagnostic runs.

	Mean $\log W$		Mean response length	
Batch	ReSPO	VESPO	ReSPO	VESPO
1	−0.021	+0.134	752	829
8	−0.188	−0.064	829	843
16	−0.385	−0.064	1076	915

The weight and length statistics evolve together. By batch 16, mean $\log W$ reaches -0.385 for ReSPO and -0.064 for VESPO, corresponding to geometric-mean weights of approximately 0.68 and 0.94. Over the same interval, mean positive-response length grows from 752 to 1076 under ReSPO (43%) and from 829 to 915 under VESPO (10%). Thus, the faster growth of ReSPO's positive responses coincides with their movement farther into the low-W region where ReSPO retains more coefficient mass. In the complete 1024-update trajectories, ReSPO's aggregate response length and training score also separate from the baselines during the early stage of optimization and continue to diverge at $N = 32$ (Fig. 5). The two views together are consistent with ReSPO learning earlier from longer positive responses: this diagnostic identifies the retained learning coefficient, while App. G shows the corresponding full-training trajectory.

To test the length–weight relationship directly, a follow-up diagnostic records positive-response length within each $\log W$ bin. Tab. 7 reports count-weighted mean lengths, with each value also

normalized by the corresponding positive-branch mean. This directly shows that low-likelihood positive responses are longer on average in both methods; the kernel determines whether their learning coefficient is retained.

Table 7: Early positive-response length by sequence-weight bin at $N = 32$. Parentheses give length relative to the method's overall positive-branch mean; "–" denotes an empty bin.

$\log W$ bin	W range	ReSPO length	VESPO length
$(-10, -5]$	4.5×10^{-5}–0.007	1241 (1.52×)	–
$(-5, -2]$	0.007–0.14	891 (1.09×)	1163 (1.48×)
$(-2, -1]$	0.14–0.37	848 (1.04×)	811 (1.03×)
$(-1, -0.5]$	0.37–0.61	904 (1.11×)	796 (1.01×)
$(-0.5, 0]$	0.61–1	656 (0.80×)	641 (0.81×)
$(0, 0.5]$	1–1.65	773 (0.95×)	717 (0.91×)
$(0.5, 1]$	1.65–2.7	827 (1.01×)	758 (0.96×)
$(1, 2]$	2.7–7.4	920 (1.13×)	870 (1.10×)
$(2, 5]$	7.4–148	1063 (1.30×)	1238 (1.57×)
Positive-branch mean	–	817	788

The allocation follows directly from the positive-branch kernels. ReSPO uses

$$\phi^{(+)}(W) = \frac{1+W}{2}\exp(1-W),$$

which approaches $e/2 \approx 1.359$ as $W \to 0$ and remains 1.348 at $\log W = -2$. VESPO uses $\phi^{(+)}(W) = W^2\exp(3(1-W))$, which vanishes as $W \to 0$ and falls to 0.245 at $\log W = -2$ and 0.006 at $\log W = -4$. At the same low value of W, ReSPO therefore retains a near-maximal coefficient on a positive response, whereas VESPO progressively removes its contribution. This kernel difference explains the coefficient-mass allocation in Tab. 5.

E.4 MoE training and Routing Replay

Both Qwen3-30B-A3B ReSPO runs at $N = 16$ reach the full budget of 1024 policy updates. The Routing-Replay ablation adds R3 while matching the model, data, optimization, response cap, batch sizes, and update budget. Both use the current ReSPO parameters $(\alpha^{(+)}, \alpha^{(-)}) = (2, 1)$, $(\beta^{(+)}, \beta^{(-)}) = (0.5, 0.5)$, and $(\lambda^{(+)}, \lambda^{(-)}) = (2, 2)$.

Tab. 8 uses the same reporting protocol as Fig. 2. "Early peak" is the maximum through policy step 256; late metrics are policy-step-weighted trapezoidal means over $[896, 1024]$; and "final" is the value at step 1024. The score is $(\texttt{critic/score/mean} + 1)/2$. Because the 30B reward includes the soft overlong penalty described in App. D.1, this is a penalized training accuracy. The subscript on the late score denotes its within-run temporal standard deviation.

Table 8: Matched Qwen3-30B-A3B ReSPO training diagnostics at $N = 16$, with and without R3 Routing Replay. All columns except the early peak and final score are time-weighted means over policy steps $[896, 1024]$.

Setting	Early peak	Late score	Final score
ReSPO	0.445	$0.563_{\pm 0.021}$	0.555
ReSPO + R3	0.446	$0.582_{\pm 0.008}$	0.584

Setting	Resp. len.	Cap-hit rate	Approx. KL	Entropy	Grad. norm
ReSPO	1992	3.505%	3.37×10^{-4}	0.128	1.13
ReSPO + R3	1213	0.011%	4.77×10^{-4}	0.114	1.21

R3 leaves the early peak essentially unchanged but raises the late score by 1.84 pp and reduces its temporal standard deviation from 2.11 to 0.75 pp. It also substantially shortens responses and nearly eliminates response-cap hits in this window. Approximate KL rises slightly. In this matched $N = 16$ run, R3 combines with ReSPO, raising the late-stage score and reducing its temporal variation.

F LIMITATIONS

Compute resource limitation. Our compute budget does not support a longer Qwen3-30B-A3B study with a longer training response limit and additional policy updates. The reported 30B experiments instead use an 8,192-token limit with the soft length penalty described in Sec. 4.1. Under our current training setup, extending both the context length and training horizon could require several weeks on 8×H200 GPUs. Owing to the same compute constraints, we run multiple ReSPO seeds only for $N = 32$, where the separation from the baselines is largest and robustness is most important to verify.

Hyperparameter optimality. The ReSPO hyperparameters are selected primarily based on the analytical tail requirements and derivative matching at $W = 1$. This study does not exhaustively search over α, β, and λ; other choices may further improve particular model scales, reuse ratios, or response-length regimes. However, we want to emphasize that without extensive hyperparameter tuniing, matching the tail requirements only is enough to find a good algorithm.

G DETAILED TRAINING STATISTICS AND REPRODUCIBILITY

Fig. 2 uses the recorded `critic/score/mean` history. Each rollout batch supplies N policy updates, so the horizontal coordinate is `training/global_step`$\times N$. The plotting script truncates every curve at 1024 policy updates, applies $(\texttt{score} + 1)/2$ to the signed DAPO-MATH score, and validates the response cap and ReSPO negative-branch parameters. The early statistic is the maximum logged value through step 256 and is reported without an error bar. For the late statistic, we linearly interpolate the curve over $[896, 1024]$ and integrate its first and second moments to obtain the time-weighted mean and standard deviation. Figs. 5 and 6 use the same experiments and additionally report AIME25 evaluation accuracy, response length, approximate KL divergence, and entropy. For 1.7B at $N = 32$, App. E.3 resolves the aggregate response-length separation into positive-branch importance-weight and coefficient-mass statistics. The KL panel uses `actor/ppo_kl` for GRPO/GSPO and `actor/approx_kl` for VESPO/ReSPO.

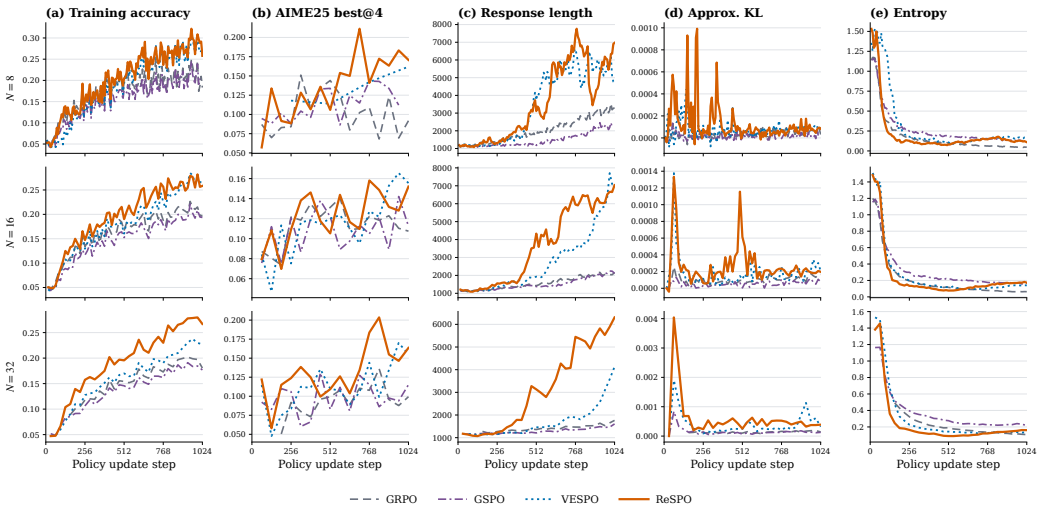


Figure 5: Detailed training statistics for Qwen3-1.7B-Base on DAPO-MATH. Rows correspond to $N \in \{8, 16, 32\}$. Training accuracy is $(\texttt{critic/score/mean} + 1)/2$, and the evaluation panel reports AIME25 best@4 accuracy.

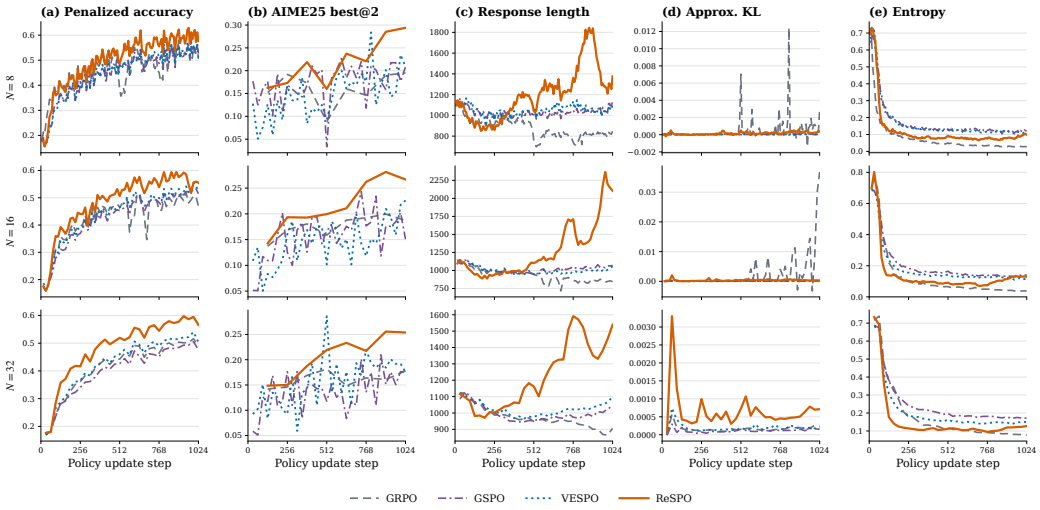


Figure 6: Detailed training statistics for Qwen3-30B-A3B-Base on DAPO-MATH. The training panel reports penalized training accuracy, and the evaluation panel reports AIME25 best@2 accuracy, the largest common best-of-k metric available for all four methods; the other panels follow Fig. 5.