WHEN POLICIES DRIFT, PREFIXES MATTER: PREFIX-NORMALIZED POLICY OPTIMIZATION FOR LLM REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

In LLM reinforcement learning, the behavior policy can lag behind the current policy through asynchronous generation and updates or repeated rollout reuse; numerical differences between training and inference add further mismatch. Proximal methods such as PPO use local likelihood ratios to correct for conditional action-probability mismatch, but not explicitly for differences in prefix visitation induced by earlier decisions. In autoregressive generation, each prefix has a unique path, yielding an exact state–action change-of-measure ratio at each position: the product of local likelihood ratios through that position. However, this ratio can grow or decay rapidly with prefix length, hindering long-sequence optimization. We propose *Prefix-Normalized Policy Optimization* (PNPO), which geometrically averages local likelihood ratios over each causal prefix and constrains updates with position-adaptive clipping. Under equal optimizer-update budgets, PNPO outperforms GRPO and GSPO in mean Pass@1 across six mathematical reasoning benchmarks across different policy-lag settings, with a larger advantage under high policy lag. With four PPO epochs, PNPO achieves 52.9% mean Pass@1, compared with 50.9% for GSPO and 50.7% for GRPO. PNPO and GSPO remain stable, whereas GRPO declines late in training. PNPO also matches its one-epoch performance using one quarter as many fresh rollout batches.

1 INTRODUCTION

Policy gradient methods have become central to reinforcement learning post-training for large language models (Schulman et al., 2017; Ouyang et al., 2022; Guo et al., 2025). These methods update the current policy using rollouts generated by a behavior policy. Because autoregressive generation is time-consuming, two common ways to improve training efficiency are to execute generation and updates asynchronously and to reuse each rollout batch for more optimization steps. However, the former can cause the behavior policy that generates the data to lag behind the current policy (Espeholt et al., 2018; Fu et al., 2025), while the latter accumulates policy divergence as fixed trajectories are reused, giving rise to policy lag (Schulman et al., 2017; Ivison et al., 2024). Moreover, numerical differences between inference and training engines can introduce additional mismatch even when parameter versions are identical (DeepSeek-AI et al., 2025). Maintaining effective policy optimization under policy lag is therefore key to balancing performance and training efficiency in large language model reinforcement learning.

The difficulty of optimizing under policy lag reflects approximations in proximal surrogate objectives. The exact performance difference identity weights states by the state visitation distribution induced by the current policy. TRPO instead retains the behavior policy's state visitation distribution and restricts policy movement through a trust region to control the resulting approximation error (Schulman et al., 2015). PPO replaces the explicit trust-region constraint with a clipped local-ratio surrogate (Schulman et al., 2017), and methods such as GRPO retain this construction (Shao et al., 2024). In these surrogate objectives, the local ratio corrects the action probability given a prefix, whereas the joint probability through the current token also depends on the probability of reaching that prefix. With exact behavior-policy advantages, the unclipped surrogate matches the true objective in value and gradient at π_β. Under smoothness assumptions, this local first-order approximation may deteriorate as policy lag grows (Figure 1(a)). Further correcting the state distribution mismatch

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

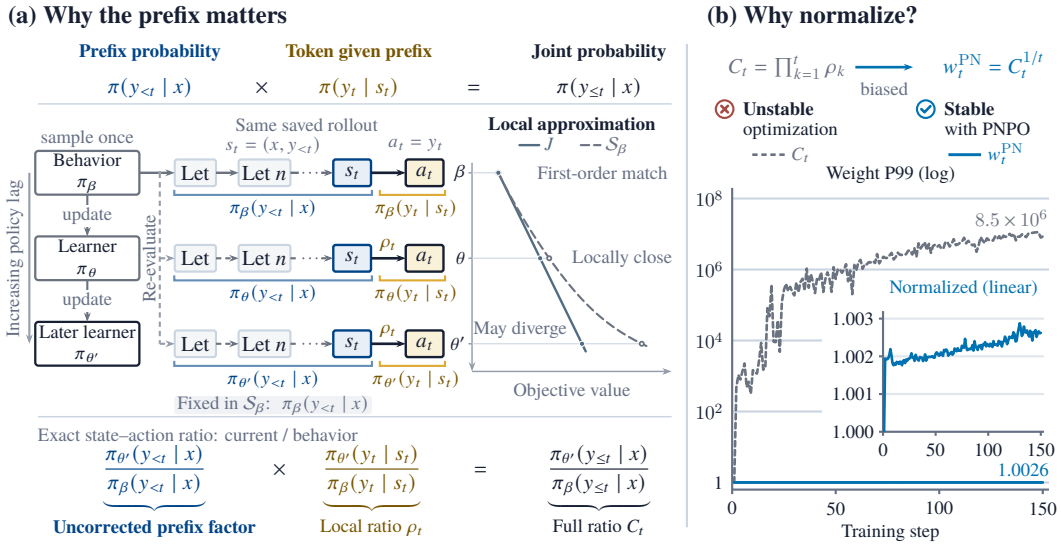

Figure 1: **Why prefixes matter and why their weights need normalization.** (a) J is the true objective; $\mathcal{S}_\beta$ is the ideal local surrogate with the behavior-policy prefix distribution held fixed. The same behavior-policy rollout is re-evaluated after policy updates. The objectives agree to first order at β, but the approximation may deteriorate as policy lag grows. The schematic shows objective values horizontally and a shared policy-update path vertically. (b) Prefix normalization compresses the dynamic range of cumulative weights. Curves show the 99th percentiles (P99) of raw and normalized weights from the same four-epoch PNPO run; the inset shows normalized-weight P99 on a linear scale.

requires computing the state visitation ratio. In a general Markov decision process, the same state can be reached through multiple paths, making this ratio difficult to compute directly. In autoregressive generation, however, the state is the full prefix and corresponds to a unique token-generation path. Let ρ_k denote the local likelihood ratio of the current policy to the behavior policy at position k. Under standard support conditions, the exact state–action change-of-measure ratio at position t is the cumulative prefix ratio $C_t = \prod_{k=1}^{t} \rho_k$ (Precup et al., 2000; Liu et al., 2020). This ratio makes the distribution correction omitted by local surrogates computable, providing a starting point for deriving unbiased gradients of the current policy. We establish and prove the equivalence of two gradient representations based on behavior-policy and current-policy advantages.

However, exactly computable cumulative prefix ratios need not be suitable for direct optimization. Even small local-ratio deviations from 1 can compound, rapidly amplifying or diminishing weights as prefixes lengthen and policy lag grows. Figure 1(b) shows raw cumulative-weight P99 reaching the order of 10^6, while prefix-normalized-weight P99 remains near 1. To control this scale, we propose *Prefix-Normalized Policy Optimization* (PNPO), which replaces the cumulative ratio at position t with the geometric mean $C_t^{1/t}$. Using only ratios through the current token, this transformation compresses cumulative log-weights at the cost of introducing bias. During training, PNPO also uses position-adaptive clipping to match acceptance intervals to weight fluctuations across positions.

With 2,400 optimizer updates, we compare one and four PPO epochs to examine PNPO's training and performance under different degrees of policy lag. Across six mathematical reasoning benchmarks, PNPO and GSPO achieve similar mean Pass@1 with one epoch (52.9% vs. 52.3%); with four epochs, PNPO reaches 52.9%, versus 50.9% for GSPO. Both remain stable, but PNPO generally achieves higher training reward, while GRPO, CISPO, and DPPO-KL exhibit marked declines. **Our main contributions are as follows:**

- We prove the equivalence of two policy-gradient representations under prefix correction, clarify their advantage–score pairings, and distinguish exact change of measure from unbiasedness of the full gradient estimator. This provides a theoretical basis for further exploration of unbiased prefix-based gradient estimators.

- We propose PNPO, using prefix normalization to control cumulative-weight scale while retaining causal prefix dependence, with accompanying position-adaptive clipping tailored to position-dependent fluctuations after normalization.
- We demonstrate PNPO's performance and training stability under greater policy lag. With four PPO epochs and the same update budget, PNPO outperforms all evaluated baselines in mean Pass@1, Pass@2, and Pass@4 across six benchmarks by 1.2–3.0, 1.2–2.4, and 0.6–2.2 percentage points, respectively. It also matches its one-epoch performance using only one quarter as many fresh rollout batches.

2 Problem Setup and Prefix Correction

2.1 Local Proximal Surrogate

Given a prompt x, let the behavior policy π_β generate a response $y_{1:L}$. At position t, the state is the full prefix $s_t = (x, y_{<t})$, the action is $a_t = y_t$, and d_t^β and d_t^θ denote the state distributions induced by the behavior and current policies, respectively. Let A_t^β denote the finite-horizon advantage under π_β, and define the local likelihood ratio as $\rho_t = \pi_\theta(a_t \mid s_t)/\pi_\beta(a_t \mid s_t)$.

The performance difference identity expresses the change in expected return from π_β to π_θ using the behavior-policy advantage A_t^β and the current policy's state–action visitation distribution. TRPO weights states by d_t^β in its surrogate objective and uses ρ_t to change only the conditional action distribution, yielding a tractable local objective. The corresponding unclipped surrogate gradient is

$$\nabla_\theta \mathcal{S}_{\beta,x}(\theta) = \sum_{t=1}^{L} \mathbb{E}_{\substack{s_t \sim d_t^\beta(\cdot|x) \\ a_t \sim \pi_\beta(\cdot|s_t)}} \left[\rho_t A_t^\beta(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t \mid s_t) \right]. \tag{1}$$

Here, both A_t^β and d_t^β are held fixed, and differentiating the local ratio yields the score-function term for the current token. Thus, at a fixed state, ρ_t exactly reweights the conditional action distribution from π_β to π_θ, but leaves the state weighting at d_t^β. This deliberate omission defines a proximal approximation, not an exact state–action correction. The gradient above coincides with the true policy gradient at $\pi_\theta = \pi_\beta$; away from the behavior policy, the accuracy of the surrogate depends on controlling the magnitude of policy movement.

TRPO constrains policy movement through a trust region (Schulman et al., 2015), PPO instead uses a clipped local-ratio surrogate (Schulman et al., 2017), and GRPO retains token-level ratios and clipping while using group-relative outcome advantages (Shao et al., 2024). These methods rely on policy proximity to limit the error introduced by state distribution shift, rather than explicitly applying the missing state ratio. Using a rollout batch for additional parameter updates may gradually weaken this proximity condition. The performance difference construction and the resulting local surrogate are derived in Appendix A.

2.2 Exact Prefix Change of Measure

In a general MDP, the state distribution ratio omitted by the local surrogate is difficult to compute. In the autoregressive process considered here, however, the full prefix is observable, and the transition deterministically appends the sampled token. The probability of reaching s_t therefore factorizes along its unique prefix path. Under standard support conditions,

$$\frac{d_t^\theta(s_t \mid x)}{d_t^\beta(s_t \mid x)} \cdot \frac{\pi_\theta(a_t \mid s_t)}{\pi_\beta(a_t \mid s_t)} = \underbrace{\prod_{k=1}^{t-1} \rho_k}_{\text{state ratio}} \underbrace{\rho_t}_{\text{action ratio}} = \prod_{k=1}^{t} \rho_k =: C_t. \tag{2}$$

The factors before position t correct for the probability of reaching the current prefix, while ρ_t corrects for the conditional probability of the current action. Thus, C_t is the exact joint state–action change-of-measure ratio at position t. Under the same support conditions, for any integrable function $f_t(s_t, a_t)$,

$$\mathbb{E}_{\substack{s_t \sim d_t^\beta(\cdot|x) \\ a_t \sim \pi_\beta(\cdot|s_t)}} \left[C_t f_t(s_t, a_t) \right] = \mathbb{E}_{\substack{s_t \sim d_t^\theta(\cdot|x) \\ a_t \sim \pi_\theta(\cdot|s_t)}} \left[f_t(s_t, a_t) \right]. \tag{3}$$

This is the autoregressive form of per-decision importance sampling (Precup et al., 2000; Zhang et al., 2026); its state-ratio factor is precisely the state distribution correction required for exact off-policy gradient estimation (Liu et al., 2020). Derivations of this factorization and the general change-of-measure identity are provided in Appendix A.

To recover the current policy gradient rather than a surrogate gradient with a fixed state distribution, let $J_x(\pi)$ denote the expected return for prompt x and A_t^θ the finite-horizon advantage under π_θ, and define $z_t = \nabla_\theta \log \pi_\theta(a_t \mid s_t)$. Applying Eq. (3) to the policy gradient theorem (Sutton et al., 1999) yields

$$\begin{aligned}\nabla_\theta J_x(\pi_\theta) &= \sum_{t=1}^{L} \mathbb{E}_{\substack{s_t \sim d_t^\theta(\cdot|x) \\ a_t \sim \pi_\theta(\cdot|s_t)}} \left[A_t^\theta(s_t, a_t) z_t \right] \\ &= \sum_{t=1}^{L} \mathbb{E}_{\substack{s_t \sim d_t^\beta(\cdot|x) \\ a_t \sim \pi_\beta(\cdot|s_t)}} \left[C_t A_t^\theta(s_t, a_t) z_t \right]. \end{aligned} \tag{4}$$

2.3 Exact change of measure and gradient representations

Eq. (1) uses the behavior-policy advantage A_t^β, whereas Eq. (4) uses the current-policy advantage A_t^θ. Merely substituting a prefix ratio does not eliminate this distinction: C_t corrects the state–action distribution through the current action, whereas the advantage also encodes subsequent returns under the corresponding policy. Recovering the current policy gradient therefore requires relating the gradient representations associated with these two advantages.

Differentiating the full prefix ratio in the performance difference identity, with the behavior-policy advantage held fixed, introduces the cumulative prefix score $\sum_{k=1}^{t} z_k$. The policy gradient theorem, in contrast, pairs the current-policy advantage with the current-token score z_t. Assume a fixed behavior policy, rewards and transitions with no explicit parameter dependence, and the regularity conditions in Appendix B. Both derivations then yield the same current policy gradient:

$$\mathbb{E}_{\pi_\beta}\left[\sum_{t=1}^{L} C_t A_t^\beta \sum_{k=1}^{t} z_k \right] = \mathbb{E}_{\pi_\beta}\left[\sum_{t=1}^{L} C_t A_t^\theta z_t \right] = \nabla_\theta J_x(\pi_\theta). \tag{5}$$

Here, $\mathbb{E}_{\pi_\beta}$ denotes expectation over responses generated by the behavior policy for a fixed prompt. The full proof is given in Appendix B. This equivalence pairs behavior-policy advantages with cumulative prefix scores and current-policy advantages with current-token scores; an exact prefix ratio alone therefore does not ensure an unbiased gradient estimate. We retain the current-token score and use advantages estimated from behavior-policy trajectories as proxies for current-policy advantages, preserving this advantage approximation.

3 Prefix-normalized policy optimization

3.1 Prefix-normalized policy weight

Under this advantage approximation, we address the prefix distribution shift omitted by local ratios. Both exact gradient representations in Eq. (5) use the cumulative prefix ratio C_t for state–action correction, but its dynamic range may grow with prefix length and policy lag. PNPO controls this scale while retaining position-wise prefix dependence by exponentiating the mean local log-ratio through the current token. For a response of valid length L_i, denoted $y_i = y_{i,1:L_i}$, we define the weight at position t as

$$w_{i,t}^{\mathrm{PN}} = C_{i,t}^{1/t} = \exp\left(\frac{1}{t} \sum_{k=1}^{t} \log \rho_{i,k} \right). \tag{6}$$

We call $w_{i,t}^{\mathrm{PN}}$ the *prefix-normalized policy weight*. Geometric averaging rescales the cumulative log-weight by $1/t$. GSPO shares a full-response geometric mean across positions (Zheng et al., 2025), whereas PNPO uses only current and preceding token ratios, excluding likelihood changes in the future suffix. This normalization no longer preserves the exact change of measure in Eq. (3); instead, it controls cumulative-weight scale at the cost of introducing bias. Under the support condition, for a

fixed position $t > 1$, Jensen's inequality gives $\mathbb{E}_{\pi_\beta}[w_t^{\mathrm{PN}}] \le (\mathbb{E}_{\pi_\beta}[C_t])^{1/t} = 1$, with strict inequality when C_t is nonconstant under the behavior policy. We therefore use it as a biased surrogate weight preserving causal prefix dependence, rather than an exact importance sampling ratio.

3.2 Position-adaptive clipping and optimization objective

Position-adaptive acceptance interval. Prefix-normalized weights have position-dependent fluctuation scales: short prefixes are more sensitive to likelihood changes in a few tokens, so fixed thresholds tend to concentrate clipping near the beginning of responses (Figure 2). To accommodate this scale difference, we design a gate based on the position-dependent factor L_i/t to accompany prefix normalization, with a wider acceptance interval for short prefixes that gradually narrows as the prefix grows. The default scaling factor is

$$h(t, L_i) = \sqrt{\frac{L_i}{t}}. \tag{7}$$

The default implementation uses a two-sided acceptance rule, retaining the score-function term at position t only when the following condition is satisfied:

$$M_{i,t} = \mathbf{1}\left\{1 - \epsilon_- \, h(t, L_i) \le w_{i,t}^{\mathrm{PN}} \le 1 + \epsilon_+ \, h(t, L_i)\right\}. \tag{8}$$

At $t = L_i$, the acceptance interval reduces to the base interval; the lower and upper tolerances may be asymmetric. Each position is checked independently: $M_{i,t} = 0$ removes only the current score-function term, without rejecting later positions. The gate performs hard rejection rather than clamping the weight to a boundary.

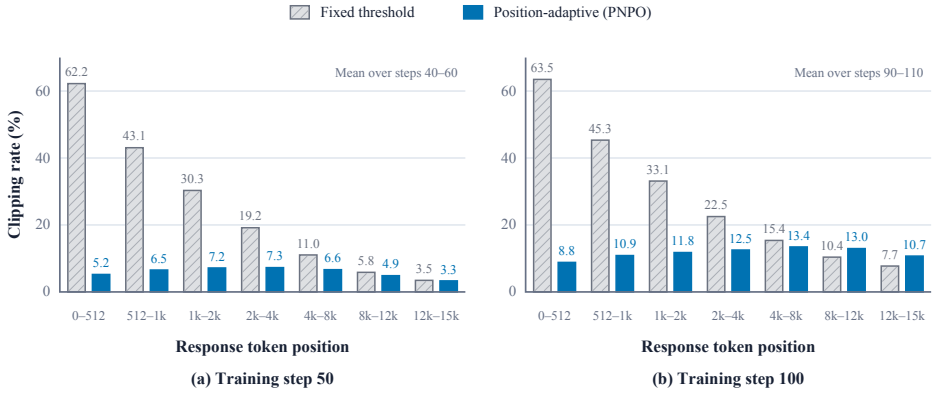

Figure 2: **Position-wise clipping rates with fixed and position-adaptive thresholds.**

Under the same base tolerances, Figure 2 shows that over steps 40–60, the mean clipping rate for the first 512 tokens decreases from 62.2% to 5.2%, with more similar clipping rates across positions.

Group-relative advantage and stop-gradient surrogate objective. For a prompt $x \sim \mathcal{D}$, independently sample G responses y_i from $\pi_\beta(\cdot \mid x)$, with rewards $R_i = R(x, y_i)$. Our experiments use GRPO's group-relative outcome advantage (Shao et al., 2024), assigning the same value to every valid token in response i:

$$\widehat{A}_{i,t} = \frac{R_i - \mathrm{mean}\big(\{R_j\}_{j=1}^G\big)}{\mathrm{std}\big(\{R_j\}_{j=1}^G\big)}. \tag{9}$$

By default, we average token contributions within each response, then equally across responses, maximizing the following stop-gradient surrogate objective:

$$\mathcal{J}_{\mathrm{PNPO}}(\theta) = \mathbb{E}_{\substack{x\sim\mathcal{D} \\ y_1,\ldots,y_G \overset{\text{i.i.d.}}{\sim} \pi_\beta(\cdot|x)}} \left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{L_i}\sum_{t=1}^{L_i} \mathrm{sg}\left[M_{i,t} w_{i,t}^{\mathrm{PN}} \widehat{A}_{i,t}\right] \log \pi_\theta(y_{i,t} \mid x, y_{i,<t})\right]. \tag{10}$$

The stop-gradient operator treats the gate, policy weight, and advantage as constants during differentiation, so gradients flow only through the current token's log-probability.

4 EXPERIMENTS

We first compare performance and training dynamics across policy-lag settings, then examine PNPO's optimization behavior through policy divergence diagnostics, weight-scale analysis, ablations, and hyperparameter sensitivity experiments.

4.1 EXPERIMENTAL SETUP

Models and training. We initialize all policies from DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025) and train them with reinforcement learning on DAPO-Math-17k (Yu et al., 2025). Rewards are $+1$ for correct and -1 for incorrect answers. Each training step uses a rollout batch of 256 prompts with 8 sampled responses each; PPO minibatches contain 64 prompt groups. With the same optimizer configuration and 2,400 optimizer updates, one and four PPO epochs per rollout batch correspond to 600 and 150 training steps, respectively (full configuration in Appendix C). A training run with four PPO epochs takes approximately 21 hours on 32 NVIDIA H20 GPUs to complete 150 training steps.

Methods compared. With one PPO epoch, we compare GRPO, GSPO, and PNPO; with four PPO epochs, we additionally include GMPO, CISPO, DPPO-KL, CTPO, and MinPRO (Shao et al., 2024; Zheng et al., 2025; Zhao et al., 2026; MiniMax, 2025; Qi et al., 2026; Zhang et al., 2026; Lei et al., 2026). Baselines use their respective weighting and clipping rules; PNPO follows Section 3.2 (hyperparameters in Appendix C). Tables 1 and 2 report single-run and three-seed results, respectively; multi-seed training curves appear in Appendix D.6.

Evaluation protocol. We evaluate Pass@k on AIME24, AIME25, AMC23, MATH-500, Minerva, and Olympiad-Bench. We sample 32 responses per problem for the first three datasets and 4 for the remaining three, and report the equally weighted six-benchmark means for Pass@1, Pass@2, and Pass@4. We evaluate the policy every 50 training steps with one PPO epoch and every 10 steps with four epochs. We select the checkpoint with the highest average score.

4.2 OVERALL PERFORMANCE AND TRAINING DYNAMICS

Table 1: **Results on six mathematical reasoning benchmarks (%).** Benchmark columns show Pass@1; Average columns show Pass@1, Pass@2, and Pass@4 averaged equally across benchmarks. Base is the shared pre-RL checkpoint. Bold marks column-wise maxima within each epoch group.

PPO epochs	Method	Pass@1 (%)						Average (%)		
		AIME24	AIME25	AMC23	MATH500	Minerva	Olympiad	Pass@1	Pass@2	Pass@4
—	Base	28.23	22.81	68.59	82.30	27.30	42.25	45.25	53.06	59.58
1	GRPO	33.75	28.02	**80.16**	**86.85**	30.06	50.74	51.60	58.55	64.15
	GSPO	37.81	**30.52**	79.06	86.65	30.33	49.26	52.27	59.42	64.65
	PNPO	**39.58**	29.48	79.61	86.45	**31.07**	**51.11**	**52.88**	**60.34**	**65.66**
4	GRPO	36.15	26.98	77.19	85.70	29.32	48.78	50.68	58.70	64.97
	GSPO	34.79	27.92	78.28	86.10	29.78	48.52	50.90	57.98	63.39
	GMPO	33.85	26.88	78.91	85.35	30.42	49.07	50.75	58.41	64.39
	CISPO	35.42	28.96	79.45	85.20	**30.51**	50.22	51.63	58.88	64.52
	DPPO-KL	33.44	27.08	78.12	85.40	29.87	49.96	50.65	58.18	64.35
	CTPO	33.85	26.56	76.88	84.80	29.32	47.70	49.85	57.74	63.89
	MinPRO	33.85	27.71	**80.16**	86.15	30.15	49.30	51.22	58.65	64.46
	PNPO	**38.75**	**30.94**	79.30	**86.50**	30.42	**51.22**	**52.86**	**60.11**	**65.61**

Table 1 shows that, with one PPO epoch, PNPO and GSPO achieve six-benchmark mean Pass@1 scores of 52.88% and 52.27%. With four PPO epochs, they achieve 52.86% and 50.90%, respectively, a gap of 1.96 percentage points. In this setting, PNPO leads all compared methods in mean Pass@1 and on four benchmarks. Experiments with multiple random seeds further support this advantage (Table 2). PNPO also achieves the highest six-benchmark mean Pass@2 and Pass@4 in both epoch settings. Grouped-benchmark Pass@k results are provided in Appendix D.7.

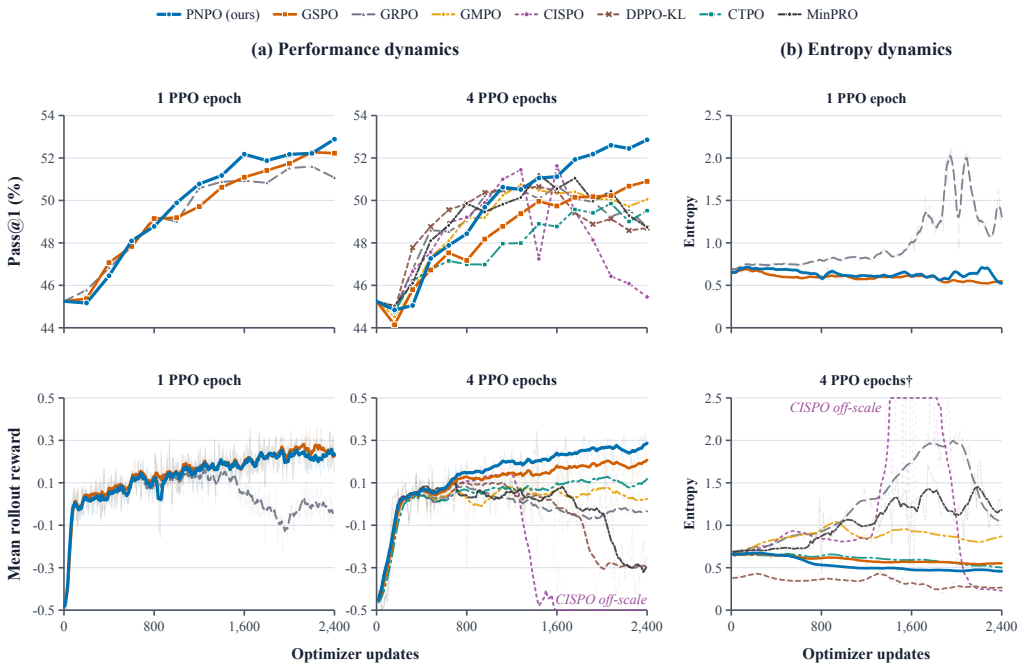

Figure 3: **Performance and training dynamics.** Curves correspond to the single runs in Table 1. Reward and entropy curves are smoothed with centered 11-step moving averages; faint lines show raw trajectories, and evaluation scores are unsmoothed. Off-scale values are shown at plot boundaries. † DPPO-KL uses different entropy aggregation; only within-run trends are compared.

Training dynamics and sustained optimization. Figure 3 compares training trajectories of PNPO and the baselines. PNPO and GSPO have broadly similar rewards with one PPO epoch. With four epochs, both train stably, but PNPO generally achieves higher reward and higher evaluation scores late in training. With 2,400 optimizer updates, PNPO achieves similar final six-benchmark mean Pass@1 with one and four PPO epochs. The four-epoch setting uses 150 fresh rollout batches—one quarter of the 600 used with one epoch. Estimated time savings at the same reward are reported in Appendix D.5. GRPO, CISPO, and DPPO-KL decline markedly late in training, while GMPO declines more mildly, potentially reflecting distribution mismatch under high policy lag. CISPO's mean Pass@1 falls from 51.6% to 45.5%, with entropy first increasing and then rapidly decreasing. MinPRO's late deterioration may reflect small preceding token ratios persistently suppressing downstream weights and weakening update signals. CTPO avoids clear collapse but makes limited gains, possibly because dynamic clipping adjusts the acceptance range without removing cumulative-weight scale differences.

Some baselines outperform PNPO and GSPO in early evaluations. To investigate the source of this advantage, we retain PNPO's prefix-normalized weighting but use GRPO-style local token clipping, likewise observing faster early improvement. This ablation shows that changing the clipping mechanism can reproduce a similar early advantage, but this advantage likewise does not persist into late training. See Section 4.4 and Appendix D.3 for the detailed comparison.

Multi-seed validation. With four PPO epochs, PNPO leads in three-seed mean Pass@1 on all six benchmarks, averaging $53.12 \pm 0.33\%$ overall (Table 2).

Table 2: Three-seed Best Pass@1 (%) with four PPO epochs.

Method	AIME24	AIME25	AMC23	MATH500	Minerva	Olympiad	Average
GRPO	35.90±1.73	27.92±1.13	77.66±1.31	85.77±0.12	30.09±0.83	49.28±0.44	51.10±0.80
GSPO	35.80±1.04	28.61±0.95	78.39±0.63	85.73±0.35	29.11±0.65	49.44±0.81	51.18±0.31
CTPO	34.03±2.24	26.39±1.21	77.27±1.56	84.87±0.16	29.20±0.74	47.42±0.46	49.86±0.59
PNPO	**38.44**±0.31	**30.45**±0.47	**80.89**±1.69	**86.47**±0.15	**31.25**±0.83	**51.22**±0.19	**53.12**±0.33

Gains over GRPO, GSPO, and CTPO are 1.94–3.26 percentage points. Scores are means ± sample SD; multi-seed training curves appear in Appendix D.6.

4.3 PPO Epoch Count and Measured Policy Divergence

To compare policy lag across update settings, we estimate token-level KL from the behavior to the current policy using K3 (Schulman, 2020), $\rho_{i,t} - \log \rho_{i,t} - 1$, on tokens sampled from the behavior policy. K1, $-\log \rho_{i,t}$, can take either sign; K3 is nonnegative per sample and has lower variance for nearby policies, making small divergences easier to resolve.

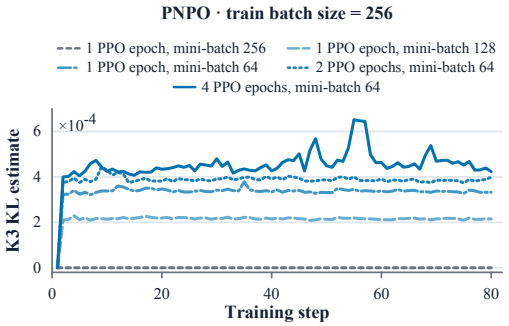

Figure 4: **K3 estimates of token-level KL across update configurations.**

Figure 4 shows higher measured divergence with more PPO epochs or smaller minibatches, consistent with greater policy lag in this setting. At a fixed minibatch size of 64 prompt groups, mean K3 is approximately 35% higher with four PPO epochs than with one. Higher divergence also coincides with larger prefix-weight upper quantiles and higher clipping rates (Appendix D.2).

4.4 Ablation Studies

Figure 5 examines the effects of prefix normalization and different clipping configurations. Without clipping, reward with raw prefix weights gradually declines after initial improvement. Adding normalization improves early reward but is followed by training collapse, indicating that weight-scale control alone is insufficient to sustain stable optimization. To compare token-level and prefix-level clipping mechanisms, we retain prefix-normalized weighting but base clipping decisions on the current token's local likelihood ratio. This control exhibits a marked decline in late-training reward (Appendix D.3), whereas both fixed-threshold and position-adaptive prefix clipping sustain improvement. Under the same base tolerances, position-adaptive thresholds reduce the concentration of clipping near the beginning of responses (Section 3.2) and yield higher late-training reward. Supplementary ablations of non-core configurations, including clipping direction and aggregation, are provided in Appendix D.4.

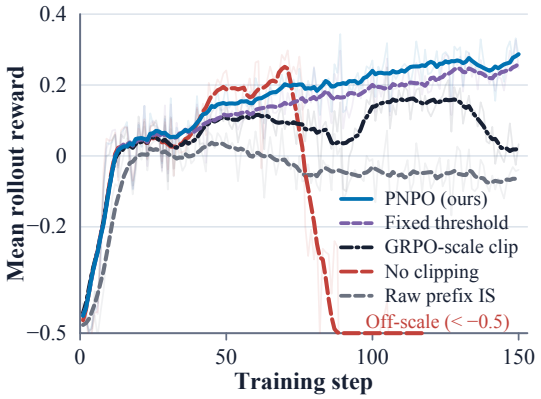

Figure 5: **Normalization and clipping ablations with four PPO epochs.**

Table 3: **Clipping sensitivity with four PPO epochs.** Other parameters retain their default values; clipping rates are averaged over steps 1–150.

Configuration	**Multiplier** m $\epsilon_\pm = m\epsilon_\pm^0$	**Exponent** α $h(t,L) = (L/t)^\alpha$	**Clipping rate** (%)	**Mean Pass@1** (%) ↑
PNPO (default)	1	0.5	11.2	52.9
Base-tolerance sweep	0.5	0.5	31.0	**53.1**
	2	0.5	4.7	53.0
Position-exponent sweep	1	0	25.1	52.4
	1	0.25	17.2	52.1
	1	0.75	9.8	**53.4**
GSPO (reference)	—	—	20.2	50.9

4.5 Hyperparameter Sensitivity

Table 3 separately examines the base-tolerance multiplier m and position exponent α. With $\alpha = 0.5$, increasing m from 0.5 to 2 reduces the mean clipping rate from 31.0% to 4.7%, while mean Pass@1 remains within 52.9%–53.1%; varying α yields scores of 52.1%–53.4%. PNPO maintains strong performance across the examined parameter range without relying on the specific default setting. Results with nonzero position exponents indicate that strong performance with the accompanying L/t-based gate does not require the exact square-root choice. Moreover, smaller tolerances $(m = 0.5)$ and fixed thresholds $(\alpha = 0)$ both yield higher clipping rates than GSPO, yet achieve higher evaluation scores and late-training rewards (Appendix D.1). Thus, the performance advantage of PNPO cannot be attributed solely to a lower clipping rate. The fixed-threshold prefix-normalized configuration also outperforms GSPO, which likewise uses geometric normalization (52.4% vs. 50.9%), indicating that causal prefix weighting brings performance gains beyond merely normalizing weights toward 1.

5 Related Work

Off-policy reinforcement learning for language models. Off-policy updates use behavior-policy trajectories to update the current policy, potentially introducing distribution shift. PPO-style minibatch updates can make even fresh rollouts off-policy, while training–inference mismatch introduces a related discrepancy (Schulman et al., 2017; DeepSeek-AI et al., 2025). Replay buffers resample older trajectories across multiple policy updates (Arnal et al., 2026), whereas OAPL introduces an explicitly off-policy objective for data from lagging behavior policies (Ritter et al., 2026). Concurrent generation and optimization in asynchronous systems can further exacerbate policy lag (Espeholt et al., 2018; Fu et al., 2025; Hu et al., 2026); VACO and SAO address policy lag through algorithmic modifications (Honari et al., 2026; Hou et al., 2026). Despite different mismatch sources, these settings share the challenge of importance weighting. We study a controlled source of policy lag by varying the number of updates per fixed rollout batch.

Likelihood-ratio granularity in reinforcement learning for reasoning. Recent critic-free reasoning objectives differ in advantage estimation, clipping, and likelihood-ratio granularity. GRPO and DAPO use token-level ratios and group-relative outcome advantages (Shao et al., 2024; Yu et al., 2025), whereas GSPO uses length-normalized response ratios in a sequence-level objective (Zheng et al., 2025). PNPO uses an intermediate prefix-level granularity.

Prefix-aware importance weighting. Classical per-decision importance sampling provides the foundation for distribution correction along trajectory prefixes (Precup et al., 2000). CTPO retains cumulative prefix ratios and addresses their scale growth through position-dependent log-space clipping (Zhang et al., 2026); MinPRO instead uses the product of the current-token ratio and the minimum ratio over preceding tokens (Lei et al., 2026). In contrast, PNPO averages log-ratios across the entire prefix to compress the cumulative scale while incorporating ratios from every prefix token, and adapts clipping intervals to position-dependent fluctuations in the normalized weights.

6 Conclusion and Future Work

We prove the equivalence of two policy-gradient representations, clarifying the relationship between exact change of measure over prefix distributions and unbiased policy-gradient estimation, and providing a theoretical foundation for prefix correction. Building on this foundation, we propose PNPO, combining prefix normalization with its accompanying position-adaptive gate to maintain stable training and improve performance under greater policy lag. Under the same optimizer-update budget, PNPO with four PPO epochs achieves mean Pass@1 comparable to that with one epoch using one quarter as many fresh rollout batches. Building on this equivalence, future work could develop a unified off-policy optimization framework for distribution correction, advantage estimation, and gradient structure, and characterize the trade-offs among unbiasedness, estimation variance, and optimization stability. We also aim to evaluate larger models and a broader range of tasks, and study how stale-trajectory reuse and online exploration can complement each other in asynchronous and replay-based training.

AI USE STATEMENT

In this work, we used generative AI tools to assist with manuscript translation and checking. Apart from these uses, we did not use generative AI tools for other tasks requiring disclosure.

The authors have manually checked the entire manuscript word by word and take responsibility for its final content.

REFERENCES

Charles Arnal, Vivien Cabannes, Taco Cohen, Julia Kempe, and Rémi Munos. Efficient RL training for LLMs with experience replay. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Mark Chen et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

DeepSeek-AI et al. DeepSeek-V3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1407–1416, 2018.

Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, Tongkai Yang, Binhang Yuan, and Yi Wu. AReaL: A large-scale asynchronous reinforcement learning system for language reasoning. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Daya Guo et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 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. doi: 10.18653/v1/2024.acl-long.211.

Homayoun Honari, Roger Creus Castanyer, Michael Przystupa, Michael Noukhovitch, Pablo Samuel Castro, and Glen Berseth. Align and filter: Improving performance in asynchronous on-policy RL. *arXiv preprint arXiv:2603.01365*, 2026.

Zhenyu Hou, Yujiang Li, Jie Tang, and Yuxiao Dong. Single-rollout asynchronous optimization for agentic reinforcement learning. *arXiv preprint arXiv:2607.07508*, 2026.

Tianhao Hu, Xiangcheng Liu, Yuchun Miao, Youshao Xiao, Hongyu Zang, Yang Zheng, Xuan Huang, Jinrui Ding, Yufei Zhang, et al. DORA: A scalable asynchronous reinforcement learning system for language model training. *arXiv preprint arXiv:2604.26256*, 2026.

Hamish Ivison, Yizhong Wang, Jiacheng Liu, Zeqiu Wu, Valentina Pyatkin, Nathan Lambert, Noah A. Smith, Yejin Choi, and Hannaneh Hajishirzi. Unpacking DPO and PPO: Disentangling best practices for learning from preference feedback. In *Advances in Neural Information Processing Systems*, volume 37, pp. 36602–36633, 2024. doi: 10.52202/079017-1154.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Shiye Lei, Zhihao Cheng, and Dacheng Tao. A step back: Prefix importance ratio stabilizes policy optimization. *arXiv preprint arXiv:2601.22718*, 2026.

Aitor Lewkowycz et al. Solving quantitative reasoning problems with language models. *arXiv preprint arXiv:2206.14858*, 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. *arXiv preprint arXiv:2305.20050*, 2023.

Yao Liu, Adith Swaminathan, Alekh Agarwal, and Emma Brunskill. Off-policy policy gradient with stationary distribution correction. In *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence*, volume 115 of *Proceedings of Machine Learning Research*, pp. 1180–1190, 2020.

Mathematical Association of America. American mathematics competitions, n.d. URL https://maa.org/student-programs/amc/.

MiniMax. MiniMax-M1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*, 2025.

Long Ouyang et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022.

Doina Precup, Richard S. Sutton, and Satinder P. Singh. Eligibility traces for off-policy policy evaluation. In *Proceedings of the 17th International Conference on Machine Learning*, pp. 759–766, 2000.

Penghui Qi, Xiangxin Zhou, Zichen Liu, Tianyu Pang, Chao Du, Min Lin, and Wee Sun Lee. Rethinking the trust region in LLM reinforcement learning. *arXiv preprint arXiv:2602.04879*, 2026.

Daniel Ritter, Owen Oertell, Bradley Guo, Jonathan Chang, Kianté Brantley, and Wen Sun. LLMs can learn to reason via off-policy RL. *arXiv preprint arXiv:2602.19362*, 2026.

John Schulman. Approximating KL divergence, 2020. URL http://joschu.net/blog/kl-approx.html.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, 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, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

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*, 2025. doi: 10.1145/3689031.3696075.

Richard S. Sutton, David A. McAllester, Satinder P. Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 12, 1999.

Qiying Yu et al. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Yuheng Zhang, Chenlu Ye, Shuowei Jin, Changlong Yu, Wei Xiong, Saurabh Sahu, and Nan Jiang. Rethinking importance sampling in LLM policy optimization: A cumulative token perspective. *arXiv preprint arXiv:2605.07331*, 2026.

Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen, Xun Wu, Yaru Hao, Tengchao Lv, Shaohan Huang, Lei Cui, Qixiang Ye, Fang Wan, and Furu Wei. Geometric-mean policy optimization. In *International Conference on Learning Representations*, 2026.

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. *arXiv preprint arXiv:2507.18071*, 2025.

A FINITE-HORIZON LOCAL SURROGATE AND EXACT PREFIX CHANGE OF MEASURE

Fix a prompt x and consider an undiscounted autoregressive process with a maximum horizon of L. At position t, let $s_t = (x, y_{<t})$ and $a_t = y_t$. For a policy π, define

$$d_t^\pi(s \mid x) = \Pr(s_t = s \mid x, \pi), \tag{11}$$

and let $J_x(\pi)$ and A_t^π denote its expected return and finite-horizon advantage, respectively. For responses that terminate before L, the same derivation applies up to the actual termination position. Throughout, we assume that the state–action distribution of the current policy is absolutely continuous with respect to that of the behavior policy.

A.1 LOCAL-RATIO SURROGATE

Expanding the finite-horizon state visitation distribution by position, the performance difference identity is

$$J_x(\pi_\theta) - J_x(\pi_\beta) = \sum_{t=1}^{L} \mathbb{E}_{\substack{s_t \sim d_t^\theta(\cdot|x) \\ a_t \sim \pi_\theta(\cdot|s_t)}} \left[A_t^\beta(s_t, a_t) \right]. \tag{12}$$

Define the local action ratio

$$\rho_t = \frac{\pi_\theta(a_t \mid s_t)}{\pi_\beta(a_t \mid s_t)}. \tag{13}$$

For a fixed state s_t, the action distribution admits the following exact change of measure:

$$\begin{aligned} &\mathbb{E}_{a_t \sim \pi_\theta(\cdot|s_t)} \left[A_t^\beta(s_t, a_t) \right] \\ &= \sum_a \pi_\theta(a \mid s_t) A_t^\beta(s_t, a) \\ &= \sum_a \pi_\beta(a \mid s_t) \frac{\pi_\theta(a \mid s_t)}{\pi_\beta(a \mid s_t)} A_t^\beta(s_t, a) \\ &= \mathbb{E}_{a_t \sim \pi_\beta(\cdot|s_t)} \left[\rho_t A_t^\beta(s_t, a_t) \right]. \end{aligned} \tag{14}$$

Proximal policy methods use a local surrogate that fixes the state distribution at each position in Eq. (12) to the corresponding behavior-policy state distribution:

$$\mathcal{S}_{\beta,x}(\theta) = J_x(\pi_\beta) + \sum_{t=1}^{L} \mathbb{E}_{\substack{s_t \sim d_t^\beta(\cdot|x) \\ a_t \sim \pi_\beta(\cdot|s_t)}} \left[\rho_t A_t^\beta(s_t, a_t) \right]. \tag{15}$$

Combining Eqs. (12) and (14) gives the exact gap

$$\begin{aligned} J_x(\pi_\theta) - \mathcal{S}_{\beta,x}(\theta) = \sum_{t=1}^{L} \sum_s &\left[d_t^\theta(s \mid x) - d_t^\beta(s \mid x) \right] \\ &\cdot \sum_a \pi_\theta(a \mid s) A_t^\beta(s, a). \end{aligned} \tag{16}$$

Thus, ρ_t exactly changes the conditional action distribution at a given state, while the surrogate still weights states according to d_t^β rather than d_t^θ.

A.2 TRPO TRUST REGION AND PPO CLIPPING

The difference in state visitation distributions in Eq. (16) motivates limiting each policy update. TRPO optimizes the surrogate with a frozen state visitation distribution under a policy-distance constraint; its practical update uses an average KL constraint

$$\begin{aligned} \max_\theta \quad & \mathcal{S}_{\beta,x}(\theta) \\ \text{s.t.} \quad & \overline{D}_{\mathrm{KL}}^\beta(\theta) \le \delta, \end{aligned} \tag{17}$$

where

$$\overline{D}_{\mathrm{KL}}^{\beta}(\theta) = \frac{1}{L}\sum_{t=1}^{L}\mathbb{E}_{s_t\sim d_t^{\beta}(\cdot|x)}\left[D_{\mathrm{KL}}\big(\pi_\beta(\cdot \mid s_t)\|\pi_\theta(\cdot \mid s_t)\big)\right], \tag{18}$$

and δ is the trust-region radius. PPO replaces explicit constrained optimization with a clipped local-ratio surrogate (Schulman et al., 2017):

$$\mathcal{J}_{\mathrm{clip}}(\theta) = \sum_{t=1}^{L}\mathbb{E}_{\substack{s_t\sim d_t^{\beta}(\cdot|x)\\ a_t\sim\pi_\beta(\cdot|s_t)}}\Big[\min\big(\rho_t A_t^{\beta}(s_t,a_t), \\ \mathrm{clip}(\rho_t, 1-\epsilon, 1+\epsilon)A_t^{\beta}(s_t,a_t)\big)\Big]. \tag{19}$$

Here, ϵ is the clipping tolerance. Once a sampled ratio moves outside the prescribed interval, clipping suppresses advantage-improving updates; it does not impose a hard constraint on the KL divergence of the full policy. GRPO retains PPO-style token-level action ratios and clipping, while replacing critic-based advantages with group-relative outcome signals.

A.3 Exact state–action change of measure for autoregressive policies

The local surrogate in Eq. (15) does not explicitly correct the state-distribution mismatch. In the autoregressive process considered here, s_t contains the full generated prefix, and transitions deterministically append the sampled token. The probability of reaching s_t therefore factorizes as

$$d_t^{\pi}(s_t \mid x) = \Pr_{\pi}(y_{<t} \mid x) = \prod_{k=1}^{t-1}\pi(a_k \mid s_k). \tag{20}$$

Along a sampled response,

$$\frac{d_t^{\theta}(s_t \mid x)}{d_t^{\beta}(s_t \mid x)} = \frac{\prod_{k=1}^{t-1}\pi_\theta(a_k \mid s_k)}{\prod_{k=1}^{t-1}\pi_\beta(a_k \mid s_k)} \\ = \prod_{k=1}^{t-1}\rho_k. \tag{21}$$

Multiplying by the action ratio at position t gives the full state–action density ratio:

$$\frac{d_t^{\theta}(s_t \mid x)\pi_\theta(a_t \mid s_t)}{d_t^{\beta}(s_t \mid x)\pi_\beta(a_t \mid s_t)} \\ = \underbrace{\prod_{k=1}^{t-1}\rho_k}_{\text{state ratio}}\ \underbrace{\rho_t}_{\text{action ratio}} \\ = \prod_{k=1}^{t}\rho_k =: C_t. \tag{22}$$

Thus, for any integrable function $f_t(s_t, a_t)$,

$$\begin{aligned}
&\mathbb{E}_{\substack{s_t\sim d_t^{\theta}(\cdot|x)\\ a_t\sim\pi_\theta(\cdot|s_t)}}\left[f_t(s_t,a_t)\right] \\
&= \sum_{s_t} d_t^{\theta}(s_t \mid x)\sum_{a_t}\pi_\theta(a_t \mid s_t)f_t(s_t,a_t) \\
&= \sum_{s_t} d_t^{\beta}(s_t \mid x)\sum_{a_t}\pi_\beta(a_t \mid s_t)\frac{d_t^{\theta}(s_t \mid x)\pi_\theta(a_t \mid s_t)}{d_t^{\beta}(s_t \mid x)\pi_\beta(a_t \mid s_t)}f_t(s_t,a_t) \\
&= \mathbb{E}_{\substack{s_t\sim d_t^{\beta}(\cdot|x)\\ a_t\sim\pi_\beta(\cdot|s_t)}}\left[C_t f_t(s_t,a_t)\right].
\end{aligned} \tag{23}$$

In summary, starting from the finite-horizon performance difference identity, this section identifies the gap between the local-ratio surrogate and the true objective arising from state-visitation distribution

mismatch. The probability factorization of autoregressive prefixes then yields the exact state–action change-of-measure ratio $C_t = \prod_{k=1}^{t} \rho_k$. This result corrects the joint distribution of prefix visitation and the current action, but obtaining an unbiased policy gradient also depends on pairing the advantage with the appropriate score function. Appendix B establishes the corresponding exact gradient representations.

B Exact gradient representations and the practical PNPO surrogate

We retain the setup of Appendix A. The behavior policy π_β is fixed, rewards and autoregressive transitions do not explicitly depend on θ, and the support condition above holds. Define the score at position t as

$$z_t = \nabla_\theta \log \pi_\theta(a_t \mid s_t). \tag{24}$$

For compactness, let $\mathbb{E}_{\pi_\beta}[\cdot] := \mathbb{E}_{\tau\sim\pi_\beta(\cdot|x)}[\cdot]$, with analogous notation for π_θ. We write $C_t(\theta)$ when emphasizing the parameter dependence.

Appendix A established that C_t is the exact state–action change-of-measure ratio at position t. Exactness of the ratio alone, however, does not determine an unbiased gradient estimator: the advantage and score-function form must also be compatible. Below, we derive two exact representations of the current-policy gradient under behavior-policy rollouts, then relate them to the proximal approximation used by the practical PNPO objective.

B.1 Gradient of the performance difference identity

Applying Eq. (23) to each state–action expectation in Eq. (12) gives

$$J_x(\pi_\theta) - J_x(\pi_\beta) = \mathbb{E}_{\tau\sim\pi_\beta(\cdot|x)}\left[\sum_{t=1}^{L} C_t(\theta) A_t^\beta(s_t, a_t)\right]. \tag{25}$$

Since

$$\begin{aligned}\nabla_\theta C_t(\theta) &= C_t(\theta)\nabla_\theta \log C_t(\theta)\\ &= C_t(\theta)\sum_{k=1}^{t}\nabla_\theta \log \pi_\theta(a_k \mid s_k)\\ &= C_t(\theta)\sum_{k=1}^{t} z_k,\end{aligned} \tag{26}$$

and both π_β and A_t^β are independent of θ, differentiating Eq. (25) yields

$$\nabla_\theta J_x(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_\beta(\cdot|x)}\left[\sum_{t=1}^{L} C_t A_t^\beta(s_t, a_t)\sum_{k=1}^{t} z_k\right]. \tag{27}$$

B.2 Direct change of measure for the policy gradient

The policy gradient theorem gives

$$\begin{aligned}\nabla_\theta J_x(\pi_\theta) &= \sum_{t=1}^{L}\mathbb{E}_{\substack{s_t\sim d_t^\theta(\cdot|x)\\ a_t\sim\pi_\theta(\cdot|s_t)}}\left[A_t^\theta(s_t, a_t) z_t\right]\\ &= \sum_{t=1}^{L}\mathbb{E}_{\substack{s_t\sim d_t^\beta(\cdot|x)\\ a_t\sim\pi_\beta(\cdot|s_t)}}\left[C_t A_t^\theta(s_t, a_t) z_t\right],\end{aligned} \tag{28}$$

The second equality applies Eq. (23): the cumulative ratio changes the sampled state–action distribution from π_β to π_θ, while $A_t^\theta z_t$ remains the score-function integrand of the current policy. Next, we verify that this current-token score form and the cumulative-score form in Eq. (27) are equal in expectation.

Exchanging the order of summation in Eq. (27) gives

$$\mathbb{E}_{\pi_\beta}\left[\sum_{t=1}^{L} C_t A_t^\beta \sum_{k=1}^{t} z_k\right] = \sum_{k=1}^{L} \mathbb{E}_{\pi_\beta}\left[z_k \sum_{t=k}^{L} C_t A_t^\beta\right]. \tag{29}$$

Let $\mathcal{F}_k = \sigma(x, a_{1:k})$ denote the prefix information up to and including a_k. Since $s_k = (x, a_{<k})$, conditioning on $\mathcal{F}_k$ is equivalent to conditioning on (s_k, a_k). For $t > k$,

$$C_t = C_k \prod_{j=k+1}^{t} \rho_j. \tag{30}$$

Therefore,

$$\mathbb{E}_{\pi_\beta}\left[C_t A_t^\beta(s_t, a_t)\middle|\mathcal{F}_k\right] = C_k \mathbb{E}_{\pi_\beta}\left[\left(\prod_{j=k+1}^{t} \rho_j\right) A_t^\beta(s_t, a_t)\middle|\mathcal{F}_k\right], \tag{31}$$

where C_k is $\mathcal{F}_k$-measurable. The remaining conditional expectation is

$$\begin{aligned}
&\mathbb{E}_{\pi_\beta}\left[\left(\prod_{j=k+1}^{t} \rho_j\right) A_t^\beta(s_t, a_t)\middle|\mathcal{F}_k\right] \\
&= \sum_{a_{k+1:t}} \left[\prod_{j=k+1}^{t} \pi_\beta(a_j \mid s_j) \frac{\pi_\theta(a_j \mid s_j)}{\pi_\beta(a_j \mid s_j)}\right] A_t^\beta(s_t, a_t) \\
&= \sum_{a_{k+1:t}} \left[\prod_{j=k+1}^{t} \pi_\theta(a_j \mid s_j)\right] A_t^\beta(s_t, a_t) \\
&= \mathbb{E}_{\pi_\theta}\left[A_t^\beta(s_t, a_t)\middle|\mathcal{F}_k\right].
\end{aligned} \tag{32}$$

Substituting Eq. (32) into Eq. (31), for every $t > k$, we have

$$\mathbb{E}_{\pi_\beta}\left[C_t A_t^\beta(s_t, a_t)\middle|\mathcal{F}_k\right] = C_k \mathbb{E}_{\pi_\theta}\left[A_t^\beta(s_t, a_t)\middle|\mathcal{F}_k\right]. \tag{33}$$

The current term $C_k A_k^\beta(s_k, a_k)$ is already determined by $\mathcal{F}_k$. Separating it from the future terms and applying Eq. (33) gives

$$\begin{aligned}
&\mathbb{E}_{\pi_\beta}\left[\sum_{t=k}^{L} C_t A_t^\beta(s_t, a_t)\middle|\mathcal{F}_k\right] \\
&= C_k A_k^\beta(s_k, a_k) + \sum_{t=k+1}^{L} \mathbb{E}_{\pi_\beta}\left[C_t A_t^\beta(s_t, a_t)\middle|\mathcal{F}_k\right] \\
&= C_k A_k^\beta(s_k, a_k) + C_k \sum_{t=k+1}^{L} \mathbb{E}_{\pi_\theta}\left[A_t^\beta(s_t, a_t)\middle|\mathcal{F}_k\right] \\
&= C_k \mathbb{E}_{\pi_\theta}\left[A_k^\beta(s_k, a_k) + \sum_{t=k+1}^{L} A_t^\beta(s_t, a_t)\middle|\mathcal{F}_k\right] \\
&= C_k \mathbb{E}_{\pi_\theta}\left[\sum_{t=k}^{L} A_t^\beta(s_t, a_t)\middle| s_k, a_k\right].
\end{aligned} \tag{34}$$

Let $r_t(s_t, a_t)$ denote the conditional mean immediate reward, and set $V^\beta_{L+1} = 0$. Under deterministic token-appending transitions,

$$A^\beta_t(s_t, a_t) = r_t(s_t, a_t) + V^\beta_{t+1}(s_{t+1}) - V^\beta_t(s_t). \tag{35}$$

Substituting this into the final expectation in Eq. (34) yields

$$\begin{aligned}
&\mathbb{E}_{\pi_\theta}\left[\sum_{t=k}^{L} A^\beta_t(s_t, a_t)\,\middle|\, s_k, a_k\right] \\
&= \mathbb{E}_{\pi_\theta}\left[\sum_{t=k}^{L}\big(r_t + V^\beta_{t+1} - V^\beta_t\big)\,\middle|\, s_k, a_k\right] \\
&= \mathbb{E}_{\pi_\theta}\left[\sum_{t=k}^{L} r_t\,\middle|\, s_k, a_k\right] - V^\beta_k(s_k) \\
&= Q^\theta_k(s_k, a_k) - V^\beta_k(s_k),
\end{aligned} \tag{36}$$

where

$$Q^\theta_k(s_k, a_k) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=k}^{L} r_t\,\middle|\, s_k, a_k\right].$$

Combining Eqs. (34) and (36) gives

$$\mathbb{E}_{\pi_\beta}\left[\sum_{t=k}^{L} C_t A^\beta_t\,\middle|\, \mathcal{F}_k\right] = C_k\left[Q^\theta_k(s_k, a_k) - V^\beta_k(s_k)\right]. \tag{37}$$

Since z_k is $\mathcal{F}_k$-measurable, applying the tower property to Eqs. (29) and (37) yields

$$\begin{aligned}
&\sum_{k=1}^{L}\mathbb{E}_{\pi_\beta}\left[z_k \sum_{t=k}^{L} C_t A^\beta_t\right] \\
&= \sum_{k=1}^{L}\mathbb{E}_{\pi_\beta}\left[z_k \mathbb{E}_{\pi_\beta}\left[\sum_{t=k}^{L} C_t A^\beta_t\,\middle|\, \mathcal{F}_k\right]\right] \\
&= \sum_{k=1}^{L}\mathbb{E}_{\pi_\beta}\left[C_k\big(Q^\theta_k(s_k, a_k) - V^\beta_k(s_k)\big) z_k\right].
\end{aligned} \tag{38}$$

Now decompose

$$Q^\theta_k(s_k, a_k) - V^\beta_k(s_k) = A^\theta_k(s_k, a_k) + V^\theta_k(s_k) - V^\beta_k(s_k). \tag{39}$$

The second term depends only on s_k. Using the state–action factorization in Eq. (22), its score contribution vanishes:

$$\begin{aligned}
&\mathbb{E}_{\pi_\beta}\left[C_k\big(V^\theta_k(s_k) - V^\beta_k(s_k)\big) z_k\right] \\
&= \mathbb{E}_{s_k \sim d^\beta_k(\cdot|x)}\left[\frac{d^\theta_k(s_k \mid x)}{d^\beta_k(s_k \mid x)}\big(V^\theta_k(s_k) - V^\beta_k(s_k)\big)\right. \\
&\qquad\qquad \left.\cdot \sum_a \pi_\beta(a \mid s_k)\frac{\pi_\theta(a \mid s_k)}{\pi_\beta(a \mid s_k)}\nabla_\theta \log \pi_\theta(a \mid s_k)\right] \\
&= \mathbb{E}_{s_k \sim d^\beta_k(\cdot|x)}\left[\frac{d^\theta_k(s_k \mid x)}{d^\beta_k(s_k \mid x)}\big(V^\theta_k(s_k) - V^\beta_k(s_k)\big)\nabla_\theta \sum_a \pi_\theta(a \mid s_k)\right] \\
&= 0.
\end{aligned} \tag{40}$$

Substituting Eqs. (39) and (40) into Eq. (38) gives

$$\begin{aligned}
&\mathbb{E}_{\pi_\beta}\left[\sum_{t=1}^{L} C_t A_t^\beta \sum_{k=1}^{t} z_k\right] \\
&= \sum_{k=1}^{L} \mathbb{E}_{\pi_\beta}\left[C_k A_k^\theta(s_k, a_k) z_k\right] \\
&= \nabla_\theta J_x(\pi_\theta).
\end{aligned} \tag{41}$$

Thus, Eqs. (27) and (28) are two exact representations of the same current-policy gradient. The former pairs A_t^β with the cumulative prefix score, whereas the latter pairs A_t^θ with the current-token score. Their expectations are equal by Eq. (41), but these components are not interchangeable: in general, $C_t A_t^\beta z_t$ is not an exact current-policy gradient term. When the assumptions above hold and the advantages are exact, a Monte Carlo estimate of either representation is unbiased for $\nabla_\theta J_x(\pi_\theta)$.

The corresponding stop-gradient surrogate forms are

$$\begin{aligned}
&\mathbb{E}_{\pi_\beta}\left[\sum_{t=1}^{L} \mathrm{sg}\left[C_t A_t^\beta\right] \sum_{k=1}^{t} \log \pi_\theta(a_k \mid s_k)\right], \\
&\mathbb{E}_{\pi_\beta}\left[\sum_{t=1}^{L} \mathrm{sg}\left[C_t A_t^\theta\right] \log \pi_\theta(a_t \mid s_t)\right].
\end{aligned} \tag{42}$$

At the current parameter values, their gradients recover Eqs. (27) and (28), respectively.

B.3 Practical PNPO surrogate

In the TRPO/PPO local surrogate, the behavior-policy advantage A_t^β comes from the performance difference identity; its use does not itself approximate the current-policy advantage. With the behavior-policy state visitation distribution held fixed, differentiating the local ratio gives the current-token score. Restoring full prefix distribution correction still allows exact use of A_t^β, but requires the cumulative prefix score; retaining the current-token score instead gives the corresponding exact representation with A_t^θ. Practical PNPO retains the current-token score and uses behavior-policy-side advantage estimates as proxies for the current-policy advantage, adopting

$$A_t^\theta(s_t, a_t) \approx A_t^\beta(s_t, a_t). \tag{43}$$

In the objective reported here, the group-relative signal $\widehat{A}_{i,t}$ obtained from rollouts serves as this behavior-policy-side proxy. As shown above, pairing this proxy with the current-token score is a proximal approximation, not the exact cumulative-score representation using A_t^β. PNPO further replaces the exact cumulative ratio with the prefix-normalized policy weight:

$$w_{i,t}^{\mathrm{PN}} = C_{i,t}^{1/t}. \tag{44}$$

When $t > 1$, using $C_{i,t}^{1/t}$ in place of $C_{i,t}$ does not preserve either exact gradient identity; the normalization deliberately trades exact change of measure for control over the scale of cumulative log-weights. Using the response-level notation and gate $M_{i,t}$ defined in Section 3.2, the surrogate reported here is

$$\mathcal{J}_{\mathrm{PNPO}}(\theta) = \mathbb{E}_{\substack{x\sim\mathcal{D} \\ y_1,\ldots,y_G \overset{\text{i.i.d.}}{\sim} \pi_\beta(\cdot|x)}}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{L_i}\sum_{t=1}^{L_i} \mathrm{sg}\left[M_{i,t} w_{i,t}^{\mathrm{PN}} \widehat{A}_{i,t}\right] \log \pi_\theta(y_{i,t} \mid x, y_{i,<t})\right]. \tag{45}$$

Thus, the practical PNPO update combines the current-token score with a behavior-policy-side advantage proxy, prefix-normalized weights, position-dependent gating, and response-level aggregation. Together, these choices make the full objective a deliberately biased proximal surrogate for the current-policy gradient, with C_t remaining its exact change-of-measure reference.

C FULL EXPERIMENTAL CONFIGURATION

Our evaluation benchmarks comprise AIME24, AIME25, and AMC23 from the American Mathematics Competitions (Mathematical Association of America, n.d.), MATH-500 (Lightman et al., 2023), Minerva (Lewkowycz et al., 2022), and Olympiad-Bench (He et al., 2024).

Table 4: Training and evaluation configuration for the main experiments. The policy-lag diagnostic additionally varies the number of PPO epochs and minibatch size; see Figure 4 for the specific settings.

Item	Configuration
Base model	DeepSeek-R1-Distill-Qwen-1.5B
Training data	DAPO-Math-17k
Rollout batch	256 prompts $\times$ 8 responses/prompt
PPO minibatch	64 prompt groups (512 response sequences)
Number of PPO epochs	1 or 4
Maximum length	1,024 prompt tokens; 15,360 response tokens
Advantage estimator	Group-relative outcome advantages (same as GRPO)
Policy model optimization	Learning rate 10^{-6}; 10-step warmup; weight decay 0.1
Rollout sampling	Temperature 1.0
Additional regularization	No reference KL in the reward or policy loss; entropy coefficient 0
Implementation	Based on veRL (Sheng et al., 2025), with vLLM (Kwon et al., 2023) for rollout generation
Reporting budget	2,400 optimizer updates; the first 600/150 training steps for one/four PPO epochs, respectively
Evaluation benchmarks	AIME24, AIME25, AMC23, MATH-500, Minerva, Olympiad-Bench
Evaluation sampling	32 responses per question for the first three benchmarks, 4 for the last three; temperature 0.7, top-$p = 0.9$
Main-experiment evaluation points	Every 50 steps (one PPO epoch); every 10 steps (four PPO epochs), plus a step-0 evaluation of the base model
Reported metrics	Pass@1 on each benchmark and equally weighted six-benchmark means for Pass@1, Pass@2, and Pass@4
Runs	The main comparison uses one run; Table 2 reports three-seed Best means and sample standard deviations for four-epoch GRPO, GSPO, CTPO, and PNPO; training curves appear in Appendix D.6
GRPO	Token-mean aggregation; local-ratio clipping tolerances $(0.2, 0.28)$; dual-clip coefficient 10.0
GSPO	Average token-level score-function terms within each response, then average over responses; sequence-ratio clipping tolerances $(3 \times 10^{-4}, 4 \times 10^{-4})$
PNPO	Average token-level score-function terms within each response, then average over responses; position-dependent gating with base tolerances $(7 \times 10^{-4}, 9.5 \times 10^{-4})$
GMPO	Geometric-mean objective, equal-response aggregation; clipping configuration fields `clip_ratio_low/high` set to 0.2/0.28
CISPO	Equal-response aggregation; clipping configuration fields `clip_ratio_low/high` set to 1/4
DPPO-KL	Divergence threshold 0.05; `clip_ratio_c` set to 10,000; `seq-mean-token-sum-norm` aggregation
CTPO	Cumulative token ratios, equal-response aggregation; position-adaptive clipping fields `la_clip_low/high` set to 0.025/0.05, `la_clip_c` to 0.05, and `clip_ratio_la_power` to 0.5
MinPRO	`token-mean` aggregation; `clip_ratio_low/high` set to 1/4

The baseline parameters in the table are the configurations used in these experiments. Both the main table and training dynamics are restricted to the common optimizer-update budget; some original training runs contain records beyond this budget, which are excluded from the comparisons here. K3 KL and prefix-weight quantiles are measured at each training step; interval means are the arithmetic averages of the per-step measurements within the corresponding intervals.

Reward and entropy curves use a centered 11-step moving average per run. The appendix multi-seed figure then computes cross-seed means and min–max ranges; evaluation scores are unsmoothed.

D SUPPLEMENTARY EXPERIMENTAL ANALYSIS

D.1 TRAINING PERFORMANCE AT HIGHER CLIPPING RATES

In Table 3, the PNPO configurations with halved base tolerances and fixed thresholds both have higher clipping rates than GSPO, yet still achieve higher mean Pass@1. To further examine their training trajectories, we compare the reward and clipping-rate curves of these two configurations with those of GSPO. The two configurations use $m = 0.5, \alpha = 0.5$ and $m = 1, \alpha = 0$, respectively; parameter definitions and base tolerances are given in Table 3. All three experiments use four PPO epochs.

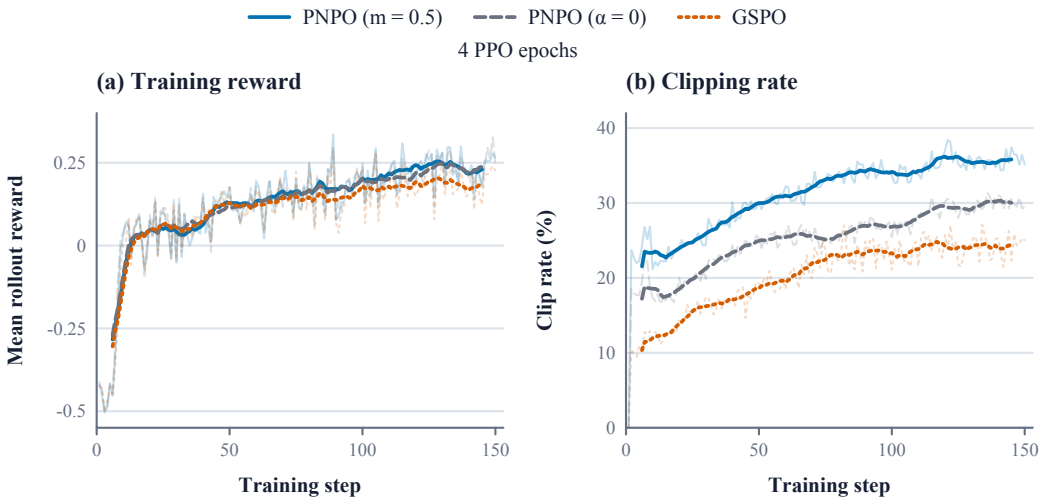

Figure 6: **Training trajectories of higher-clipping-rate PNPO configurations and GSPO.** (a) Rollout reward; (b) clipping rates, computed using the gate-rejection criterion of PNPO and the clipping-surrogate criterion of GSPO, respectively. The horizontal axis covers training steps 1–150. Light lines show raw curves, and dark lines show centered 11-step moving averages, plotted only over steps 6–145 where complete windows are available. Each curve corresponds to one run.

Figure 6 shows that both PNPO configurations retain higher late-training reward despite their higher clipping rates. The mean clipping rates with halved tolerances and fixed thresholds are 31.0% and 25.1%, respectively, both exceeding the 20.2% of GSPO. Over steps 100–150, their mean rewards reach 0.229 and 0.221, respectively, also exceeding the 0.184 of GSPO. Together with the evaluation results in the table, these findings show that the performance advantage of PNPO cannot be attributed solely to a lower clipping rate.

D.2 ACCOMPANYING DIAGNOSTICS OF CLIPPING RATES AND PREFIX WEIGHTS

Using the five PNPO runs in Figure 4, we further compare clipping rates and the upper tail of log prefix-normalized weights. All configurations use a training batch size of 256, with the same base tolerances, position-adaptive gating, and other training hyperparameters. The three configurations with a minibatch size of 64 use one, two, and four PPO epochs, respectively; the other two use one PPO epoch with minibatch sizes of 256 and 128. Both metrics are computed over the shared interval of steps 1–80 and aligned to the same training steps as the K3 KL curves in the main text.

Figure 7 shows that, with a fixed minibatch size of 64, increasing the PPO epoch count raises the upper tail of prefix weights and makes clipping more frequent. Increasing from one to four PPO epochs raises the mean clipping rate from 2.03% to 8.84% and the mean P99 of log weights from 2.045×10^{-3} to 3.185×10^{-3}, an increase of approximately 56%. In particular, the clipping rate with four PPO epochs rises further later in training, widening its gap from the other two configurations.

With one PPO epoch, reducing the minibatch size also raises both metrics. Both are close to zero with a minibatch size of 256; reducing it to 128 and 64 increases the mean clipping rate to 1.75%

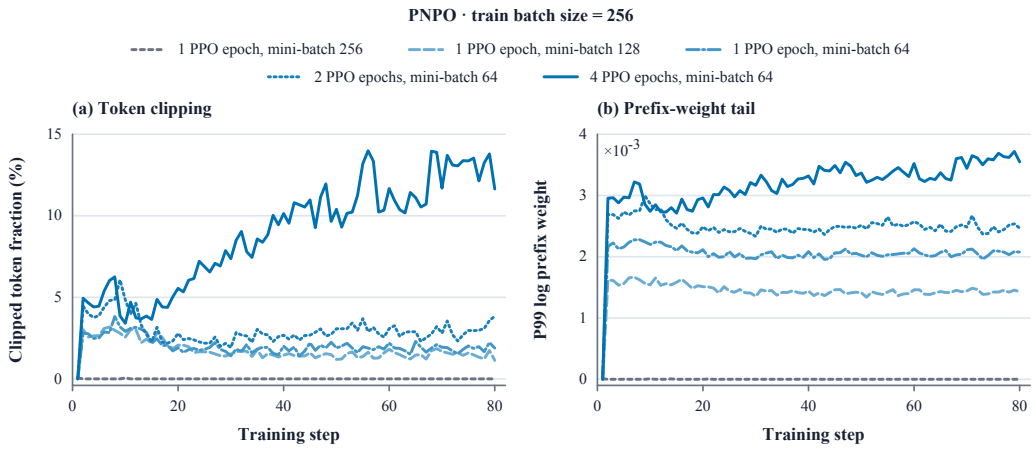

Figure 7: **Clipping rates and the upper tail of prefix weights under different update configurations.** (a) Fraction of token score terms rejected by the PNPO gate; (b) P99 of log prefix-normalized weights. The five configurations are the same as in Figure 4; each shows the raw, unsmoothed curve of one run over training steps 1–80. The horizontal axis is aligned by rollout-batch index, with different numbers of parameter updates per step across configurations. The mean P99 reported in the text is the arithmetic mean of per-step P99 values over this interval.

and 2.03%, respectively, and progressively raises the P99 curves of log weights. These changes accompany the K3 KL results in the main text, indicating that greater measured policy divergence coincides with changes in the upper tail of prefix weights and gating behavior. Even with only one PPO epoch, the current policy gradually diverges from the behavior policy through successive minibatch updates on the same rollout batch.

D.3 Clipping Configuration and Training Performance

The early validation advantages of some baselines in Figure 3 are accompanied by lower clipping rates. GRPO uses local token ratios, whereas GSPO and PNPO use length-normalized weights. These two types of weights differ in numerical scale, and their corresponding clipping tolerances also differ by orders of magnitude. Table 5 shows that, over the early interval of steps 1–50 with four PPO epochs, the mean clipping rates of GRPO, GMPO, CISPO, and DPPO-KL are all below 0.011%, compared with 14.54% for GSPO and 5.85% for full PNPO.

To examine how the clipping configuration affects early improvement and subsequent training, we retain the prefix-normalized weights and response-level aggregation of PNPO while adopting GRPO-scale clipping, and denote this control as **PNPO + GRPO-scale clipping**. Specifically, this configuration makes a one-sided clipping decision based on the local token ratio at the current position, with lower and upper tolerances of 0.2 and 0.28, respectively, a Dual-Clip parameter of 3, and no position-adaptive thresholds; all other training settings are the same as for full PNPO. Figure 8 shows the reward and evaluation curves of this control, full PNPO, and the baselines under the same parameter-update budget.

With GRPO-scale clipping, the early mean clipping rate drops from the 5.85% of full PNPO to 0.0072%, the same order of magnitude as the 0.0077% of GRPO. At the same time, the early evaluation scores of this control improve more rapidly. As shown in Figure 8(b), its six-benchmark mean Pass@1, averaged over the first five evaluation points after training begins (steps 10–50), reaches 47.6%, exceeding the 46.7% of full PNPO.

However, this early advantage does not persist into late training. Figure 8(a) shows a clear late-training reward decline for this control: its mean reward over steps 101–150 is only 0.119, less than half that of full PNPO (0.254); its six-benchmark mean Pass@1 at step 150 is 51.8%, also below the 52.9% of full PNPO. Nevertheless, this control still outperforms GRPO, GMPO, CISPO, and DPPO-KL, which likewise have low early clipping rates: all four baselines have late-training mean reward of approximately 0.044 or lower and Pass@1 at step 150 no greater than 50.1%.

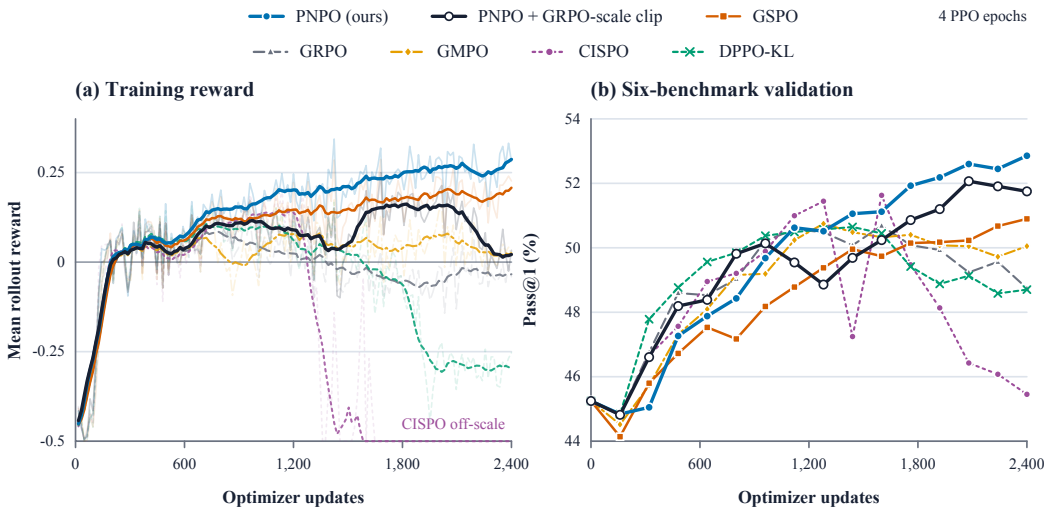

Figure 8: **Clipping-configuration comparison with four PPO epochs.** (a) Rollout reward: light lines show raw records, and dark lines show centered 11-step moving averages, using available points at the boundaries; (b) six-benchmark mean Pass@1, connecting measured checkpoints and including the shared step-0 evaluation. The horizontal axis corresponds to a budget of 2,400 parameter updates. Black denotes PNPO + GRPO-scale clipping, and blue denotes full PNPO; CISPO rewards below -0.5 are shown along the lower boundary. Each configuration has one run.

Table 5: **Mean clipping rates (%) with four PPO epochs.** Per-step clipping rates are arithmetically averaged over the early interval of steps 1–50 and the full reported interval of steps 1–150, respectively. Each method is measured according to its own clipping rule.

Method	Early (steps 1–50)	Full (steps 1–150)
GRPO	0.0077	0.3716
GMPO	0.0104	0.0939
CISPO	1.47×10^{-5}	1.2752
DPPO-KL	2.13×10^{-4}	0.0874
GSPO	14.54	20.18
PNPO + GRPO-scale clipping	0.0072	0.1813
Full PNPO	5.85	11.19

Thus, the clipping configuration can explain some differences in the speed of early improvement, but faster initial improvement does not lead to better late-training performance. The advantage of this control over other low-clipping-rate baselines provides additional evidence that prefix normalization can retain optimization benefits with less clipping; the higher late-training reward and final Pass@1 of full PNPO demonstrate the complementary roles of prefix weights and their accompanying clipping in sustained optimization.

D.4 Comparisons of Clipping and Aggregation

While retaining the same prefix-normalized weights, we separately compare fixed and position-adaptive thresholds, one-sided and two-sided clipping, and equal-response aggregation (SMTM) and token-mean aggregation (TM). Default PNPO uses position-adaptive thresholds, two-sided clipping, and SMTM; each comparison replaces only one component while keeping all other settings unchanged. TM averages valid token score terms over the entire batch, whereas SMTM first averages within each response and then aggregates responses with equal weights. Table 6 summarizes the six-benchmark evaluation results for each configuration and additionally reports individual scores for Raw prefix IS and the fixed-threshold configuration from the core ablations in Section 4.4.

Table 6: **Comparison of design variants with four PPO epochs, Pass@1 (%).** Avg is the equally weighted mean across the six datasets; bold indicates the best value in each column. † Raw prefix IS removes both prefix normalization and prefix-clipping protection.

Configuration	AIME24	AIME25	AMC23	MATH-500	Minerva	Olympiad-Bench	Avg
Raw prefix IS†	29.1	24.0	73.9	83.9	28.6	46.2	47.6
Fixed thresholds	35.7	30.3	81.6	87.1	29.4	50.5	52.4
One-sided clipping	**39.4**	30.2	82.5	86.7	28.2	50.2	52.9
Token-mean aggregation	37.3	30.1	**82.9**	**87.3**	29.2	51.0	**53.0**
PNPO (default)	38.8	**30.9**	79.3	86.5	**30.4**	**51.2**	52.9

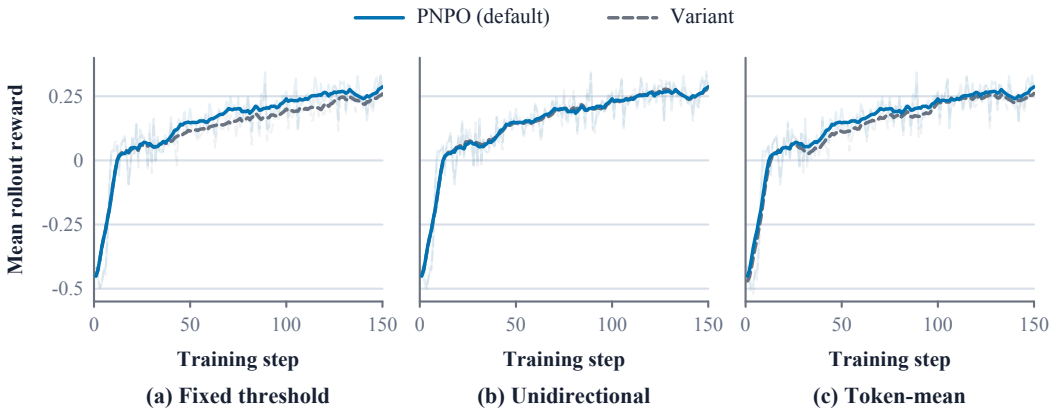

Figure 9: **Comparisons of thresholds, clipping direction, and aggregation with four PPO epochs.** Each panel uses default PNPO as the reference and replaces one component with (a) fixed thresholds; (b) one-sided clipping; or (c) token-mean aggregation over the entire batch. Solid blue lines denote default PNPO, and dashed gray lines denote the corresponding variant; light lines show raw records, and dark lines show centered 11-step moving averages, using available points at the boundaries. Curves cover training steps 1–150.

Position-adaptive thresholds. Figure 5 in the main text already shows the advantage of position-adaptive thresholds in late-training reward; Figure 9(a) presents this comparison separately. Their six-benchmark mean Pass@1 scores are relatively close: 52.9% with position-adaptive thresholds and 52.4% with fixed thresholds. The difference is primarily reflected in training reward; position-wise clipping behavior is shown in Figure 2.

Clipping direction. The reward curves for one-sided and two-sided clipping are broadly similar, and both achieve a six-benchmark mean Pass@1 of 52.9%. With position-adaptive thresholds retained, the two acceptance rules show no clear separation in training or evaluation performance.

Aggregation. SMTM yields slightly higher late-training reward than TM, but this difference does not translate into a higher evaluation score: their six-benchmark mean Pass@1 scores are 52.9% and 53.0%, respectively, indicating similar performance.

D.5 Training-Time Savings

Within the given update budget, the rollout rewards of GSPO and PNPO with one PPO epoch approach convergence around 0.25, our target in Figure 10. We estimate time as the threshold-crossing step times a shared mean step duration for each epoch setting. We average `timing_s/step` within each run's retained head/tail log excerpts, then equally across GRPO, GSPO, and PNPO, obtaining approximately 283 and 510 seconds for one and four PPO epochs. This compares progress on a common time scale, not directly measured cumulative wall-clock times.

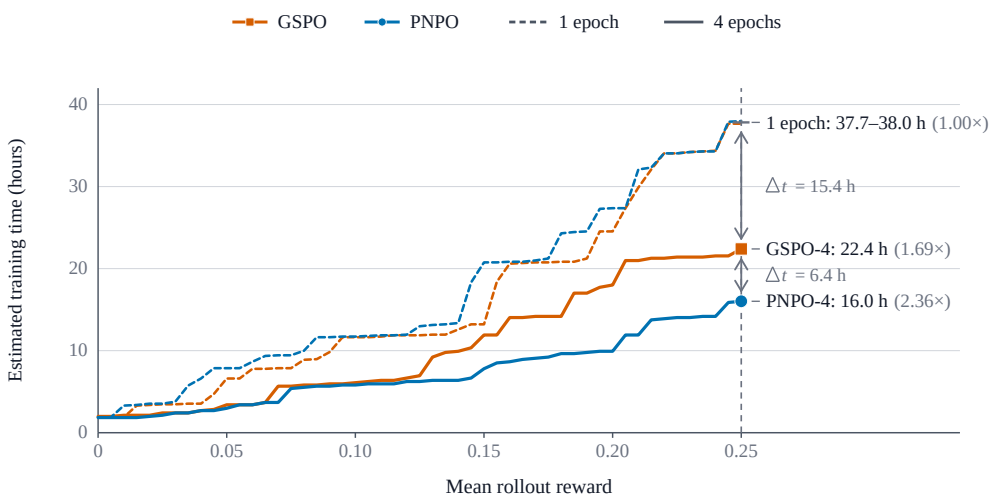

Figure 10: **Estimated training time to reach the same rollout reward.** Time equals the first threshold-crossing step of the 11-step centered reward average times the epoch-specific mean step duration. Centered averages use five subsequent steps; four-epoch records extend beyond the main 150-step budget.

With one PPO epoch, both methods reach the target at an estimated 38 hours. With four epochs, PNPO and GSPO reach it at steps 113 and 158, or approximately 16.0 and 22.4 hours: an estimated $2.36\times$ speedup for PNPO over the single-epoch reference and a 6.4-hour saving over four-epoch GSPO. The latter uses recorded continuation, not extrapolation; GSPO does not reach the target within the main 150-step budget.

D.6 Multi-seed Validation

With four PPO epochs, we evaluate three seeds each for GRPO, GSPO, CTPO, and PNPO. The Best average of PNPO is $53.12 \pm 0.33\%$, compared with $51.18 \pm 0.31\%$ for GSPO, $51.10 \pm 0.80\%$ for GRPO, and $49.86 \pm 0.59\%$ for CTPO.

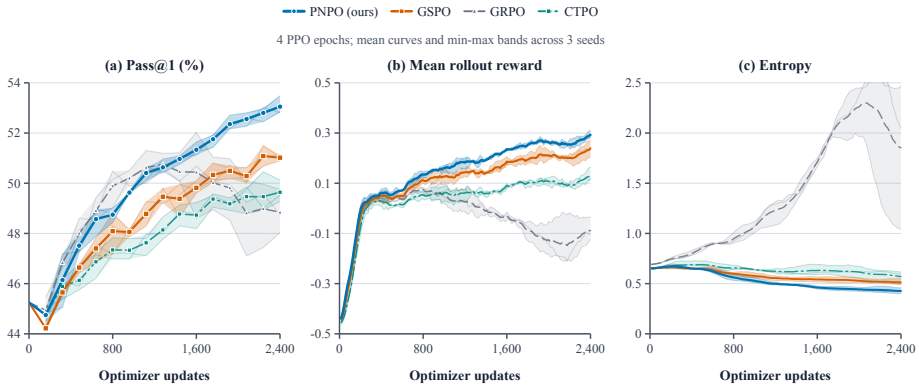

Figure 11: Multi-seed training dynamics with four PPO epochs. Main lines show three-seed means and bands show min–max ranges. Reward and entropy are smoothed per run before aggregation; evaluation scores are unsmoothed.

D.7 Multi-sample Evaluation

We use the same single runs and Best checkpoints as the main table, keeping the checkpoint fixed across all k. For c correct answers among n samples per problem, Pass@k is estimated as $1 - \binom{n-c}{k}/\binom{n}{k}$ (Chen et al., 2021), then averaged over problems and benchmarks. The first three benchmarks have 32 samples per problem and the remaining three have four.

Table 7: Pass@k at Best checkpoints (%). Each panel averages its three benchmarks equally; averaging panels (a) and (b) at $k = 1, 2, 4$ recovers the six-benchmark means in the main table. Bold denotes the highest score within each epoch group.

(a) MATH-500, Minerva, and Olympiad-Bench average

PPO epochs	Method	Pass@1	Pass@2	Pass@4
—	Base	50.62	57.21	62.39
1	GRPO	55.88	61.63	**66.05**
	GSPO	55.41	60.99	64.47
	PNPO	**56.21**	**61.83**	65.22
4	GRPO	54.60	61.17	65.87
	GSPO	54.80	59.93	63.64
	GMPO	54.95	61.29	**66.08**
	CISPO	55.31	61.10	65.11
	DPPO-KL	55.08	61.16	65.74
	CTPO	53.94	60.38	64.88
	MinPRO	55.20	61.26	65.52
	PNPO	**56.05**	**61.85**	65.97

(b) AIME24, AIME25, and AMC23 average

PPO epochs	Method	Pass@1	Pass@2	Pass@4	Pass@8	Pass@16	Pass@32
—	Base	39.88	48.91	56.77	62.98	67.68	71.67
1	GRPO	47.31	55.46	62.26	67.90	72.68	76.94
	GSPO	49.13	57.86	64.84	70.25	73.82	76.11
	PNPO	**49.56**	**58.85**	**66.11**	**71.74**	**76.11**	**79.44**
4	GRPO	46.77	56.24	64.07	70.04	74.79	79.44
	GSPO	47.00	56.04	63.15	68.99	74.08	78.33
	GMPO	46.55	55.53	62.71	68.04	72.26	76.11
	CISPO	47.94	56.66	63.93	69.82	74.56	77.22
	DPPO-KL	46.22	55.20	62.95	69.68	**75.71**	**80.56**
	CTPO	45.76	55.11	62.90	69.17	74.58	79.44
	MinPRO	47.24	56.04	63.41	69.59	74.02	76.94
	PNPO	**49.66**	**58.38**	**65.26**	**70.66**	74.85	78.33