

Latent-GRPO: Group Relative Policy Optimization for Latent Reasoning

Jingcheng Deng^{1,2,*} Zihao Wei^{1,2,*} Liang Pang^{1,†} Junhong Wu^{2,3}
Shicheng Xu^{1,2} Zenghao Duan^{1,2} Huawei Shen^{1,2}

¹State Key Laboratory of AI Safety, Institute of Computing Technology, Chinese Academy of Sciences

²University of Chinese Academy of Sciences

³Institute of Automation, Chinese Academy of Sciences

{dengjingcheng23s, pangliang}@ict.ac.cn

*Equal contribution. †Corresponding author.

Abstract

Latent reasoning offers a more efficient alternative to explicit reasoning by compressing intermediate reasoning into continuous representations and substantially shortening reasoning chains. However, existing latent reasoning methods mainly focus on supervised learning, and reinforcement learning in latent space remains highly unstable. We study this problem through the lens of Group Relative Policy Optimization (GRPO), and show that directly adapting GRPO to latent reasoning is fundamentally non-trivial: latent reasoning changes both the probability density and the sampling mechanism, causing three coupled bottlenecks: absence of intrinsic latent manifolds, where unconstrained exploration pushes rollouts off the valid latent manifold; exploration-optimization misalignment, where trajectory-level rewards can induce incorrect token-level updates; and latent mixture non-closure, where jointly reinforcing multiple correct latent paths can produce an invalid averaged state. To address them, we propose **Latent-GRPO**, which combines invalid-sample advantage masking, one-sided noise sampling, and optimal correct-path first-token selection. Across eight reasoning benchmarks, Latent-GRPO improves over its latent initialization by 7.86 Pass@1 points on low-difficulty tasks and surpasses explicit GRPO by 4.27 points on high-difficulty tasks while using 3–4× shorter reasoning chains. It also achieves stronger pass@ k performance under Gumbel sampling. These results establish Latent-GRPO as an effective approach for stable and efficient latent reasoning.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable efficacy in complex reasoning tasks through the generation of discrete chains of thought (CoT) (Wei et al., 2022; Xu et al., 2026). While test-time scaling studies show that performance often improves with longer reasoning

chains (Muennighoff et al., 2025; OpenAI, 2024), this paradigm incurs substantial computational redundancy and latency (Wei et al., 2025). To reduce this cost, recent work has explored *latent reasoning*, which shifts intermediate reasoning from explicit language into continuous representations. Early methods (Hao et al., 2024; Tan et al., 2025), which directly adopt the hidden state as the latent token, often shorten reasoning chains at the expense of performance. Recently, vocabulary-superposition approaches (Zhang et al., 2025), which adopt a linear combination of the top- k vocabulary as the latent token, have emerged as an alternative. In particular, Latent-SFT (Deng et al., 2025a) demonstrates that with proper Supervised Fine-Tuning (SFT), latent reasoning can match or even surpass explicit supervised fine-tuning while substantially compressing the reasoning chain.

With the success of SFT approaches, Reinforcement Learning naturally emerges as the next key step. Previous works (see Section 2 for details) have established the fundamental tools for RL training: (1) **Sampling Mechanism:** Wu et al. (2025) proposes injecting Gumbel noise to simulate the explicit sampling process, achieving effective exploration; (2) **Probability Density Estimation:** Soft-GRPO (Zheng and Lee, 2025) adopts the Gumbel re-parameter trick to estimate the probability density of the latent token, enabling PPO-style training. However, even with these established techniques, RL for latent reasoning remains an unsolved problem. Soft-GRPO struggles to outperform standard token RL, and scaling the Gumbel temperature to encourage broader exploration invariably precipitates catastrophic model collapse. Overall, an effective RL framework to fully unleash the potential of latent reasoning remains a gap.

We study this problem with GRPO (Shao et al., 2024), a widely adopted RL approach for LLMs that underpins much of the recent progress in explicit reasoning (DeepSeek-AI, 2025). We identify

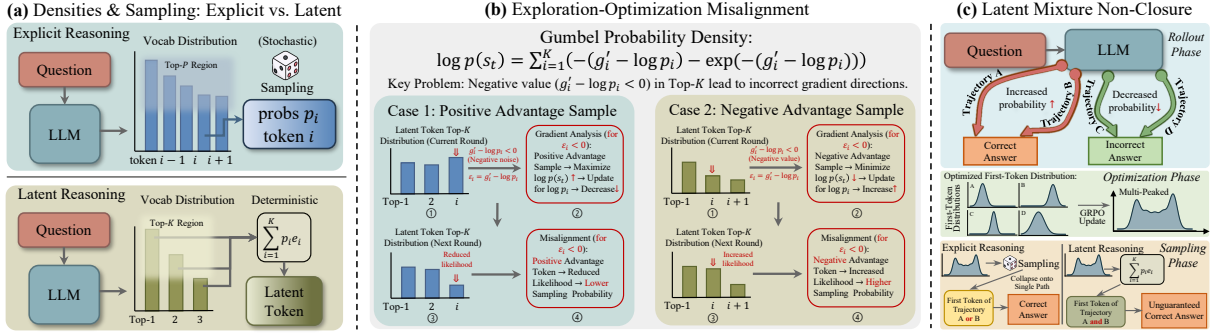


Figure 1: Key discrepancies and bottlenecks in adapting GRPO to latent-token reasoning. (a) Explicit and latent reasoning differ in both probability density and sampling mechanism. (b) Exploration-Optimization Misalignment: The standard Gumbel density objective yields incorrect directional updates, inadvertently pushing down components from positive-advantage latent trajectories while pushing up components from negative-advantage ones. (c) Latent Mixture Non-Closure: jointly reinforcing several correct latent paths can produce an averaged continuous state that does not correspond to a valid reasoning path.

three bottlenecks: (1) **Absence of Intrinsic Latent Manifolds**: direct application of Latent-RL fails to spontaneously induce latent reasoning capabilities within explicit models; such foundational structures must be injected before RL exploration, typically during SFT. Furthermore, unbounded continuous exploration can propel the model into chaotic, out-of-distribution (OOD) regions off the valid manifold, resulting in non-terminating generation. (2) **Exploration-Optimization Misalignment**: under naive Gumbel perturbations, the optimization direction can mismatch the trajectory-level advantage. Consequently, components from positive-advantage latent trajectories may receive downward probability updates, while components from negative-advantage trajectories may receive upward updates (See Figure 1(b)). (3) **Latent Mixture Non-Closure**: The set of valid latent reasoning states is not closed under jointly fitting multiple correct paths. In explicit reasoning, even when the model assigns high probabilities to several correct first-step continuations, discrete sampling ensures that reasoning proceeds along a single concrete path. In latent reasoning, however, the model must continue from a single continuous latent action under the same prefix, so jointly reinforced correct paths can induce an averaged latent state that need not support any correct continuation (See Figure 1(c)).

Building on this diagnosis, we propose three tailored designs. First, we initialize the model from Latent-SFT and apply **Invalid Sample Advantage Masking** to discard overlength trajectories that likely fall off the valid latent reasoning manifold. Second, we introduce **One-sided Noise Sam-**

pling, which constrains Gumbel perturbations to be strictly positive and thus aligns the update direction with the advantage sign. Third, we propose **Optimal Correct Path First Token Selection**, which scores each correct trajectory by its trajectory-level average surrogate log-probability under the rollout objective and updates only the first token of the selected correct path, while keeping later-token updates from other paths active. This specifically targets the shared initial step, where multiple correct trajectories under the same prefix are most vulnerable to harmful mode averaging. Together, these designs yield **Latent-GRPO**, a stable RL framework for latent-token reasoning.

Extensive experiments on LLaMA-3.2-1B-Instruct and Qwen2.5-Math-7B show that Latent-GRPO delivers stable gains across low- and high-difficulty settings. On low-difficulty tasks, it improves over Latent-SFT by 7.86 Pass@1 points on average while using a $4.44\times$ shorter reasoning chain than explicit GRPO. On high-difficulty tasks, it improves over Latent-SFT by 14.77 points on average, outperforms explicit GRPO by over 4 Pass@1 points, uses a $3.31\times$ shorter reasoning chain, and achieves the best Pass@1 on Math500, AIME24, and AIME25. Under Gumbel sampling, Latent-GRPO also retains strong pass@k capability, reaching 50+ pass@64 on the AIME benchmarks. These results validate Latent-GRPO as an effective and robust framework for efficient, high-performance latent reasoning.

Code, data, and checkpoints are available at <https://github.com/DJC-GO-SOLO/Latent-GRPO>.

2 Preliminaries

We focus this section on the technical background for Latent-GRPO; additional related work on explicit and latent reasoning appears in Appendix A.

2.1 Latent Reasoning

Standard LLMs generate explicit Chains of Thought (CoT) via an autoregressive process, where each discrete token y_t is sampled from the vocabulary distribution $P(y_t | \mathbf{x}, y_{<t})$. To alleviate the computational redundancy of discrete sequences, latent reasoning replaces intermediate explicit tokens with continuous vector representations, termed *latent tokens*.

At generation step t , instead of sampling a discrete token, the model computes a probability distribution over the vocabulary. To maintain computational tractability and filter long-tail noise, this distribution is typically truncated to the top- K candidates. Let $\mathcal{V}_t^{(K)}$ denote the set of the top- K tokens at step t , and let $p_{t,i}$ denote the normalized probability of the i -th token in $\mathcal{V}_t^{(K)}$. The latent token s_t is defined as the expectation of the corresponding top- K embeddings (Zhang et al., 2025; Zhuang et al., 2025):

$$s_t = \sum_{i=1}^K p_{t,i} \mathbf{e}_{t,i}, \quad (1)$$

where $\mathbf{e}_{t,i} \in \mathbb{R}^d$ is the embedding vector of the i -th token in $\mathcal{V}_t^{(K)}$. This construction maps a truncated vocabulary distribution into a continuous latent action in the embedding space.

2.2 Reinforcement Learning via Soft-GRPO

To further optimize reasoning capabilities without extensive supervised data, GRPO (Shao et al., 2024) is widely adopted. For a given query $\mathbf{x}$, GRPO samples a group of G outputs $\{o_1, o_2, \dots, o_G\}$, computes rule-based rewards $\{R_1, R_2, \dots, R_G\}$, and calculates normalized advantage $\hat{A}_j = (R_j - \mu_R) / \sigma_R$ for each output.

Adapting GRPO to the latent-token paradigm requires redefining the probability density to account for a mixed generation trajectory: a continuous latent reasoning segment $s_{1:T_{\text{lat}}}$ followed by a discrete explicit answer segment $y_{1:T_{\text{exp}}}$. **Soft-GRPO** (Zheng and Lee, 2025) models these two phases separately.

For the latent segment ($t \leq T_{\text{lat}}$), Soft-GRPO injects Gumbel noise ξ_i (Jang et al., 2017) into the

top- K components to simulate the stochasticity of explicit sampling and encourage exploration. Let $g'_{t,i} = \log p_{t,i} + \xi_i$ denote the Gumbel-perturbed score of the i -th component. During rollout, these perturbed scores are converted into noisy top- K mixture weights through a temperature-scaled softmax:

$$\alpha_{t,i} = \frac{\exp(g'_{t,i} / \tau_g)}{\sum_{u=1}^K \exp(g'_{t,u} / \tau_g)}, \quad (2)$$

$$s_t = \sum_{i=1}^K \alpha_{t,i} \mathbf{e}_{t,i},$$

where τ_g is the Gumbel temperature. Thus, the injected noise affects the forward latent token through the auxiliary perturbed-score representation. Based on the same perturbations, Soft-GRPO defines the density of each latent token, which we refer to as the *Gumbel-perturbation density*, as:

$$\log \pi_\theta(s_t) = \sum_{i=1}^K \left(-(g'_{t,i} - \log p_{t,i}) - \exp(-(g'_{t,i} - \log p_{t,i})) \right). \quad (3)$$

For the explicit segment ($t > T_{\text{lat}}$), the model reverts to standard autoregressive generation. The probability of generating the discrete token y_t is given by the standard categorical distribution over the vocabulary $\mathcal{V}$:

$$\log \pi_\theta(y_t) = \log P(y_t | \mathbf{x}, s_{1:T_{\text{lat}}}, y_{<t}). \quad (4)$$

Consequently, the policy ratio $r_{j,t}(\theta)$ utilized in the proximal policy update is piecewise defined, seamlessly bridging the continuous and discrete spaces:

$$r_{j,t}(\theta) = \begin{cases} \frac{\pi_\theta(s_{j,t} | \mathbf{x}, s_{j,<t})}{\pi_{\text{old}}(s_{j,t} | \mathbf{x}, s_{j,<t})}, & t \leq T_{\text{lat}}, \\ \frac{\pi_\theta\left(\begin{smallmatrix} y_{j,t} | \mathbf{x}, \\ s_{j,1:T_{\text{lat}}}, y_{j,<t} \end{smallmatrix}\right)}{\pi_{\text{old}}\left(\begin{smallmatrix} y_{j,t} | \mathbf{x}, \\ s_{j,1:T_{\text{lat}}}, y_{j,<t} \end{smallmatrix}\right)}, & t > T_{\text{lat}}. \end{cases} \quad (5)$$

Based on the Gumbel-perturbation density for latent tokens and the standard categorical density for explicit tokens, Soft-GRPO optimizes a clipped surrogate objective (Schulman et al., 2017) with a step-wise Kullback–Leibler (KL) penalty to constrain policy drift. Let

$$L_j = T_{\text{lat}}^{(j)} + T_{\text{exp}}^{(j)}, \quad (6)$$

denote the total length of the j -th trajectory, and let

$$\mathbb{D}_{\text{KL}}^{(j,t)} = \mathbb{D}_{\text{KL}}(\pi_{\theta}(\cdot | h_{j,t}) \| \pi_{\text{ref}}(\cdot | h_{j,t})), \quad (7)$$

denote the KL divergence at step t under the conditioning context $h_{j,t}$. The Soft-GRPO objective is then given by

$$\mathcal{J}(\theta) = \mathbb{E}_{\mathbf{x} \sim P(\mathcal{X})} \left[\frac{1}{G} \sum_{j=1}^G \frac{1}{L_j} \sum_{t=1}^{L_j} \left(\min \left(r_{j,t}(\theta) \hat{A}_j, \right. \right. \right. \\ \left. \left. \left. \text{clip}(r_{j,t}(\theta), 1 - \epsilon_{\text{clip}}, 1 + \epsilon_{\text{clip}}) \hat{A}_j \right) - \beta \mathbb{D}_{\text{KL}}^{(j,t)}(\theta) \right) \right], \quad (8)$$

where ϵ_{clip} denotes the PPO clipping parameter.

This objective has a gradient-level source of Exploration-Optimization Misalignment. For the standard two-sided Gumbel density, the local update direction of a latent component is proportional to $\hat{A}_j (1 - \exp(-\Delta_i))$, where $\Delta_i = g'_i - \log p_i(\theta)$ is the perturbation margin. Thus, even when $\hat{A}_j > 0$, a negative margin $\Delta_i < 0$ pushes the corresponding component probability downward. Conversely, for a negative-advantage trajectory, a negative margin may push the component probability upward. A detailed derivation is provided in Appendix D.

While Soft-GRPO provides a mathematical bridge from GRPO to mixed latent-discrete trajectories, its unconstrained Gumbel-based exploration introduces substantial stability challenges in latent reasoning, which motivate the design of our method.

3 Methodology: Latent-GRPO

To systematically overcome the fundamental bottlenecks of continuous latent reasoning—namely the absence of intrinsic manifolds, exploration-optimization misalignment, and latent mixture non-closure—we propose **Latent-GRPO**. Building upon the mathematical foundation of Soft-GRPO, our framework introduces three carefully designed strategies: (1) Invalid Sample Advantage Masking to establish exploration guardrails, (2) One-sided Noise Sampling to correct optimization directions, and (3) Optimal Correct Path First Token Selection to avoid harmful mode averaging. The overall training procedure is summarized in Algorithm 1.

3.1 Invalid Sample Advantage Masking for Bounded Exploration

As observed in Soft-GRPO, latent RL alone cannot reliably induce latent reasoning capabilities in an

explicit model from scratch. Latent-GRPO therefore starts from a Latent-SFT-initialized policy to provide a valid latent reasoning manifold before RL exploration. However, even with such initialization, unconstrained Gumbel exploration can still push rollout trajectories off this manifold, often causing chaotic, non-terminating generation. If these invalid trajectories are included in standard GRPO advantage normalization, they contaminate the group statistics and corrupt the normalized advantages of valid samples. To prevent this, we introduce **Invalid Sample Advantage Masking** before group-wise normalization.

For a group of G sampled trajectories, let L_j denote the total length of the j -th trajectory, and let L_{max} be the predefined maximum response length. Since rollouts are truncated at L_{max} , we treat a trajectory as valid only if it terminates before reaching this limit, i.e., it emits an EOS/stop token within the allowed budget. We define the subset of valid trajectories as

$$\mathcal{G}_{\text{valid}} = \{j \mid L_j < L_{\text{max}}\}. \quad (9)$$

The group baseline is computed only over this valid subset:

$$\mu_R = \frac{1}{|\mathcal{G}_{\text{valid}}|} \sum_{j \in \mathcal{G}_{\text{valid}}} R_j, \\ \sigma_R = \sqrt{\frac{1}{|\mathcal{G}_{\text{valid}}|} \sum_{j \in \mathcal{G}_{\text{valid}}} (R_j - \mu_R)^2}. \quad (10)$$

The valid-sample normalized advantage is then defined as

$$\hat{A}_j^{\text{valid}} = \begin{cases} \frac{R_j - \mu_R}{\sigma_R}, & \text{if } j \in \mathcal{G}_{\text{valid}}, \\ 0, & \text{if } j \notin \mathcal{G}_{\text{valid}}. \end{cases} \quad (11)$$

For numerical stability, if $|\mathcal{G}_{\text{valid}}| = 0$ or $\sigma_R = 0$, we set $\hat{A}_j^{\text{valid}} = 0$ for all j . By excluding trajectories that fail to terminate before the maximum response length from the group statistics and explicitly assigning them zero advantage, Invalid Sample Advantage Masking prevents off-manifold rollouts from interfering with the optimization of valid trajectories, thereby keeping exploration within the effective latent reasoning region.

3.2 One-sided Noise Sampling for Aligned Optimization

The standard Gumbel-perturbation density in Soft-GRPO injects unconstrained noise ξ_i , yielding the

perturbed target $g'_i = \log p_i + \xi_i$. Since ξ_i can be negative, g'_i may fall below the original prediction $\log p_i$. This creates *Exploration-Optimization Misalignment* described in Section 1: for positive-advantage trajectories, the probability mass of the sampled latent components may be reduced when it should be increased, while for negative-advantage trajectories, it may be increased when it should be reduced.

To correct this optimization-direction mismatch, we propose **One-sided Noise Sampling**, which constructs a strictly positive perturbation margin for each latent component. Specifically, we first sample i.i.d. standard Gumbel noise and then apply a per-dimension clipped-and-shifted transformation:

$$\begin{aligned} \xi_i^+ &= \text{clip}(\xi_i, -a, b) + a + \delta, \\ \xi_i &\stackrel{\text{i.i.d.}}{\sim} \text{Gumbel}(0, 1), \end{aligned} \quad (12)$$

where $a > 0$ and $b > 0$ are the lower and upper clipping magnitudes, respectively, and $\delta > 0$ is a small positive offset. In our implementation, we use $a = 1.5$ and $b = 3.0$, corresponding to the common Gumbel clipping range $[-1.5, 3.0]$. We use this transformation as a target-margin construction rather than requiring the transformed perturbation to remain Gumbel-distributed. Since the transformation is applied independently to each dimension, it avoids coupling the top- K components while guaranteeing $\xi_i^+ \in [\delta, a + b + \delta]$ for all i . Moreover, the uniform shift $a + \delta$ does not change the normalized top- K weights because softmax is invariant to adding the same constant to all perturbed scores; it only moves the perturbation margins into the positive range. The resulting one-sided target is then written as

$$g_i^* = \log p_i(\theta_{\text{old}}) + \xi_i^+, \quad (13)$$

which satisfies $g_i^* > \log p_i(\theta_{\text{old}})$ at the beginning of optimization. The rollout latent token is then constructed by the same temperature-scaled softmax in Eq. (2), with g_i^* replacing g'_i . However, PPO-style training performs multiple update epochs on the same rollout data, so the updated prediction $\log p_i(\theta)$ may eventually exceed this fixed target. Once the target is crossed, the original one-sided assumption is violated because the effective perturbation margin becomes negative:

$$\Delta_i(\theta) = g_i^* - \log p_i(\theta). \quad (14)$$

When $\Delta_i(\theta) < 0$, optimization is no longer guided by a positive perturbation margin. To preserve the

intended one-sided behavior throughout repeated PPO updates, we apply a conditional Straight-Through Estimator (STE) (Bengio et al., 2013) to obtain a corrected one-sided margin $\tilde{\Delta}_i(\theta)$; the formal definition is provided in Appendix C. Concretely, the STE keeps the same forward value but negates the backward gradient, effectively treating the crossed target as a reflected positive-margin case. Using this one-sided margin, we define the one-sided Gumbel-margin surrogate log-likelihood as

$$\log \pi_\theta^+(s_t) = \sum_{i=1}^K \left(-\tilde{\Delta}_i(\theta) - \exp(-\tilde{\Delta}_i(\theta)) \right). \quad (15)$$

Because the conditional STE modifies the backward pass, π_θ^+ should be interpreted as a surrogate latent likelihood used for policy optimization rather than a mathematically exact probability density. Once the fixed one-sided target is exceeded, the gradient is flipped so that optimization continues to follow the intended one-sided direction throughout repeated policy updates.

3.3 Optimal Correct Path First Token Selection for Avoiding Path Harmful Averaging

Continuous latent reasoning faces a deterministic continuous-action bottleneck. Although Latent-GRPO optimizes a PPO-style objective, the latent token fed into the next step is a single continuous state rather than a sampled discrete branch. This creates a unique geometric pitfall termed *Latent Mixture Non-Closure*. In standard GRPO, multiple correct trajectories in the same group can simultaneously receive positive training signals. This is usually harmless for explicit reasoning because categorical sampling ultimately commits generation to one concrete token and hence one concrete path. In latent reasoning, however, jointly reinforcing several correct latent paths under the same prefix can instead induce mode averaging: the fitted latent token may move toward a barycentric state between several correct paths, and this averaged state need not correspond to any valid continuation. This is analogous to the multi-modal action-averaging problem in deterministic continuous control, where fitting several valid actions with one continuous action can yield an invalid action between modes.

This problem is most severe at the first reasoning step ($t = 1$). Since the prompt context $\mathbf{x}$ is identical across all trajectories, positive training signals

from multiple correct trajectories are applied to the same prefix-to-action mapping. Therefore, jointly fitting multiple correct first-step paths can push the first latent token toward a harmful averaged state under the same prefix. For later steps ($t > 1$), the generated prefixes have already diverged, which substantially alleviates this mode-averaging effect.

To address this issue, we propose **Optimal Correct Path First Token Selection**. Let $\mathcal{G}_{\text{correct}} \subseteq \mathcal{G}$ denote the subset of trajectories in the group whose final answers are verified as correct by the task-specific reward function or verifier. When $|\mathcal{G}_{\text{correct}}| > 1$, we compute a trajectory-level score for each correct path using its average per-token surrogate log-probability under the rollout objective:

$$\bar{\ell}_j = \frac{1}{L_j} \left[\sum_{t=1}^{T_{\text{lat}}^{(j)}} \log \pi_{\theta_{\text{old}}}^+(s_{j,t} | \mathbf{x}, s_{j,<t}) + \sum_{t=T_{\text{lat}}^{(j)}+1}^{L_j} \log \pi_{\theta_{\text{old}}} \left(\begin{matrix} y_{j,t} | \mathbf{x}, \\ s_{j,1:T_{\text{lat}}^{(j)}}, y_{j,<t} \end{matrix} \right) \right]. \quad (16)$$

We then select the optimal trajectory as

$$j^* = \arg \max_{j \in \mathcal{G}_{\text{correct}}} \bar{\ell}_j. \quad (17)$$

This criterion selects the correct path with the highest average surrogate log-probability under the rollout objective, providing a conservative tie-breaking rule among multiple verified correct trajectories without inducing harmful mode averaging at the shared initial step. Importantly, this path selection is used only to decide which first latent token remains active at $t = 1$; later-token updates from the other paths are preserved once their prefixes diverge. We then define a time-dependent and trajectory-dependent mask

$$M_{j,t} = \begin{cases} 0, & \text{if } j \in \mathcal{G}_{\text{correct}} \setminus \{j^*\} \\ & \text{and } t = 1, \\ 1, & \text{otherwise,} \end{cases} \quad (18)$$

and use it to construct the masked advantage

$$\tilde{A}_{j,t} = M_{j,t} \hat{A}_j^{\text{valid}}. \quad (19)$$

As a result, only the first latent token of the selected optimal correct path contributes to the update at $t = 1$, while all competing correct paths remain active at subsequent steps. Thus, the method does not collapse optimization to a single full trajectory; it only avoids harmful mode averaging at the shared initial state without suppressing later-stage latent exploration.

3.4 Final Latent-GRPO Objective

Combining the three strategies above, the final Latent-GRPO objective is defined as

$$\mathcal{J}(\theta) = \mathbb{E}_{\mathbf{x} \sim P(\mathcal{X})} \left[\frac{1}{G} \sum_{j=1}^G \frac{1}{L_j} \sum_{t=1}^{L_j} \left(\min \left(r_{j,t}^*(\theta) \tilde{A}_{j,t}, \text{clip}(r_{j,t}^*(\theta), 1 - \epsilon_{\text{clip}}, 1 + \epsilon_{\text{clip}}) \tilde{A}_{j,t} \right) - \beta \mathbb{D}_{\text{KL}}^{(j,t)} \right) \right]. \quad (20)$$

Here, ϵ_{clip} denotes the PPO clipping parameter, $\tilde{A}_{j,t}$ combines Invalid Sample Advantage Masking and Optimal Correct Path First Token Selection, and the modified PPO-style surrogate ratio is defined as

$$r_{j,t}^*(\theta) = \begin{cases} \frac{\pi_{\theta}^+(s_{j,t} | \mathbf{x}, s_{j,<t})}{\pi_{\theta_{\text{old}}}^+(s_{j,t} | \mathbf{x}, s_{j,<t})}, & t \leq T_{\text{lat}}^{(j)}, \\ \frac{\pi_{\theta} \left(\begin{matrix} y_{j,t} | \mathbf{x}, \\ s_{j,1:T_{\text{lat}}^{(j)}}, y_{j,<t} \end{matrix} \right)}{\pi_{\theta_{\text{old}}} \left(\begin{matrix} y_{j,t} | \mathbf{x}, \\ s_{j,1:T_{\text{lat}}^{(j)}}, y_{j,<t} \end{matrix} \right)}, & t > T_{\text{lat}}^{(j)}. \end{cases} \quad (21)$$

4 Experiments

4.1 Experimental Setup

We follow the experimental setting of Latent-SFT and divide our experiments into low- and high-difficulty regimes. Because the full training corpora of existing reasoning models are unavailable, we start from base instruct models and train both the explicit SFT initialization and the Latent-SFT initialization on matched data before comparing their RL performance. For low-difficulty tasks, we evaluate on GSM8K-Aug (Deng et al., 2024), GSM-Hard (Gao et al., 2023), SVAMP (Patel et al., 2021), and MultiArith (Roy and Roth, 2015), using LLaMA-3.2-1B-Instruct (Team, 2024) as the base model. For high-difficulty tasks, we evaluate on Math500 (Hendrycks et al., 2021), AIME24, AIME25 (MAA, 2024), and GPQA (Rein et al., 2023), using Qwen2.5-MATH-7B (Yang et al., 2024) as the base model. Additional implementation details are provided in Appendix E.

4.2 Main Results

Overall Comparison. We organize the main results around two complementary comparisons. First, we compare *explicit reasoning* and *latent reasoning* under matched base models and data, in order to evaluate whether latent reasoning can preserve task performance while reducing reasoning

Method	GSM8K-Aug		GSM-Hard		SVAMP		MultiArith		Average			
	Pass@1 ↑	#L ↓	Pass@1 ↑	#L ↓	Pass@1 ↑	#L ↓	Pass@1 ↑	#L ↓	Pass@1 ↑	#L ↓	Δ Pass@1 ↑	GRPO/#L ↑
<i>Explicit Reasoning</i>												
SFT	49.51	77.46	11.67	85.78	57.38	44.10	92.23	45.49	52.70	63.20	-	-
GRPO	62.26	111.1	14.70	130.8	60.78	69.40	94.16	65.50	57.98	94.20	+5.28	1.00×
<i>Latent Reasoning (No Sampling)</i>												
Latent-SFT	47.55	22.00	10.25	25.50	53.22	15.30	90.80	16.30	50.46	19.78	-	-
Soft-GRPO	48.34	21.60	10.30	25.60	53.19	15.30	91.07	16.40	50.73	19.73	+0.27	4.77×
Latent-GRPO	66.29	23.80	15.83	28.00	56.50	15.80	94.65	17.20	58.32	21.20	+7.86	4.44×

Table 1: Main results on the low-difficulty benchmark suite using LLaMA-3.2-1B-Instruct. We report Pass@1 and reasoning length (#L) for each dataset, together with averaged performance across benchmarks. Explicit reasoning uses the standard discrete-token CoT pattern, while latent reasoning uses the No Sampling mode. Δ denotes the average Pass@1 gain over the corresponding SFT baseline, i.e., GRPO over SFT for explicit reasoning and Soft-GRPO or Latent-GRPO over Latent-SFT for latent reasoning. GRPO/#L denotes the average-length compression ratio relative to explicit GRPO, computed as the average reasoning length of explicit GRPO divided by that of the current RL method; larger values indicate shorter reasoning. “-” indicates that the corresponding quantity is not applicable.

Method	Math500		AIME24		AIME25		GPQA		Average			
	Pass@1 ↑	#L ↓	Pass@1 ↑	#L ↓	Pass@1 ↑	#L ↓	Pass@1 ↑	#L ↓	Pass@1 ↑	#L ↓	Δ Pass@1 ↑	GRPO/#L ↑
<i>Explicit Reasoning</i>												
SFT	67.40	2662	13.39	6423	6.77	6550	32.07	3555	29.91	4798	-	-
GRPO	75.15	2982	16.61	7500	19.79	6815	38.25	4568	37.45	5466	+7.54	1.00×
<i>Latent Reasoning (No Sampling)</i>												
Latent-SFT	64.85	1070	6.67	1651	6.61	1299	29.67	1214	26.95	1309	-	-
Soft-GRPO	60.60	1133	6.20	1894	3.59	1511	27.12	1426	24.38	1491	-2.57	3.67×
Latent-GRPO	80.40	929	26.56	2548	23.23	1735	36.68	1384	41.72	1649	+14.77	3.31×

Table 2: Main results on the high-difficulty benchmark suite using Qwen2.5-Math-7B.

length. Second, within the latent-reasoning setting, we further compare **Latent-GRPO** against **Soft-GRPO** to demonstrate the effectiveness of Latent-GRPO for latent reasoning; directly applying standard GRPO to train the Latent-SFT model consistently leads to training collapse, so we do not report its performance in the main tables. Across both settings, we report Pass@1 together with reasoning length (#L).

Low-difficulty Performance. Table 1 reports the low-difficulty results for LLaMA-3.2-1B-Instruct. On the in-domain GSM8K-Aug benchmark, Latent-GRPO delivers a substantial improvement over the latent initialization, raising Pass@1 from 47.55 to 66.29, while the response length changes only slightly from 22.00 to 23.80. It also clearly outperforms Soft-GRPO (48.34) and even surpasses GRPO (62.26), while remaining much shorter than GRPO in response length (23.80 vs. 111.1). On the OOD benchmarks, the same trend largely persists: Latent-GRPO consistently improves over the latent baselines and achieves the best performance on GSM-Hard and MultiArith, suggesting that its gains are not tied to a single training distribution.

Moreover, it maintains short responses on these OOD datasets, e.g., 28.00 vs. 130.8 on GSM-Hard and 17.20 vs. 65.50 on MultiArith when compared with GRPO. Although Latent-GRPO is slightly below GRPO on SVAMP (56.50 vs. 60.78), we attribute this gap to the mismatch between the explicit and latent starting points on that dataset, since the SFT and Latent-SFT models are not equally strong there. Consistent with this interpretation, the average improvement of Latent-GRPO over Latent-SFT is 7.86 points, exceeding the 5.28-point gain of GRPO over SFT. At the same time, it remains 4.44× shorter than GRPO on average. Figure 3 further shows that Latent-GRPO maintains stable training dynamics for 10K steps.

High-difficulty Performance. Table 2 reports the results on the high-difficulty benchmark suite, where we evaluate the same comparison protocol on Qwen2.5-Math-7B across more challenging mathematical and scientific reasoning tasks. Compared with the low-difficulty setting, where Latent-GRPO improves over Latent-SFT by 7.86 points on average, the gain here rises to 14.77 points, nearly doubling the relative improvement. This advantage

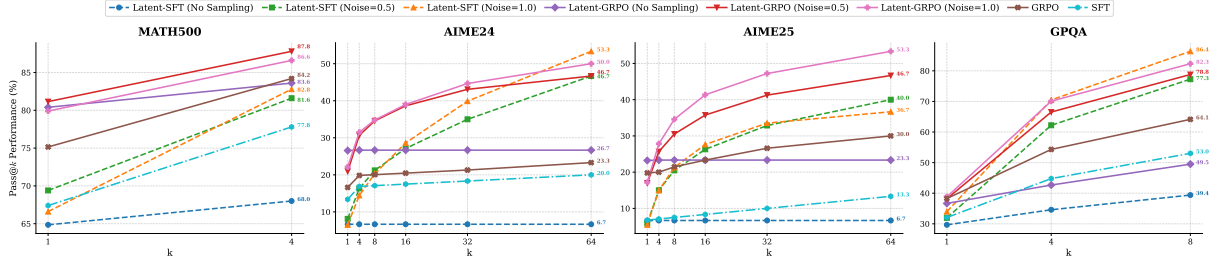


Figure 2: Pass@ k performance of different models on the high-difficulty benchmarks.

is also substantially larger than the 7.54-point gain achieved by explicit GRPO over SFT. Moreover, Latent-GRPO surpasses explicit GRPO by 4.27 points in average Pass@1, while still remaining $3.31\times$ shorter in reasoning length on average. Consistent with this overall trend, it achieves the best Pass@1 on all three mathematical reasoning benchmarks: Math500, AIME24, and AIME25. Figure 4 further shows that Latent-GRPO maintains strong performance over most of the high-difficulty training window. The final checkpoint exhibits a late-stage drop, which we attribute to a manifold-boundary effect in latent RL: after many updates, stochastic latent rollouts may reach under-trained regions outside the Latent-SFT-induced reasoning manifold, leading to malformed or non-terminating generations that hit the maximum response length (see Appendix B).

4.3 Pass@ k Performance of Latent-GRPO

Latent reasoning in the No Sampling mode is deterministic and therefore unsuitable for pass@ k evaluation. To assess sampling capability, we follow prior work (Wu et al., 2025) and inject Gumbel noise into latent-token construction during inference, enabling diverse latent reasoning trajectories from the same prompt.

Figure 2 shows that Latent-GRPO retains strong pass@ k capability under Gumbel sampling and consistently outperforms explicit GRPO on the mathematical benchmarks, with especially large gains on AIME24 and AIME25 at larger k . The results also reveal a trade-off: noisy latent decoding can reduce pass@1 on the hardest benchmarks, but it substantially improves pass@ k by increasing the diversity of sampled latent reasoning paths.

4.4 Ablation Study

Figures 3 and 4 provide the ablation results for the two key designs introduced in Section 3.2 and Section 3.3, namely **One-sided Noise Sampling** and **First Token Selection**. The most important

Dataset	Pass@1 $\uparrow$		#L $\downarrow$	
	Full	w/o Mask	Full	w/o Mask
Math500	80.40	78.10	929	1030
AIME24	26.56	23.28	2548	2822
AIME25	23.23	19.84	1735	2454
GPQA	36.68	36.17	1384	1408
Average	41.72	39.35	1649	1928

Table 3: Ablation of Invalid Sample Advantage Masking on the high-difficulty suite with Qwen2.5-Math-7B. “w/o Mask” retains overlength trajectories in both the group statistics and the policy update. Removing the mask costs 2.37 average Pass@1 points and lengthens reasoning by 16.9%.

factor is One-sided Noise Sampling. On both the low-difficulty GSM8K-Aug setting and the high-difficulty Math500 setting, removing this design causes the validation performance to drop sharply and eventually collapse, while the response length becomes much less controlled. This aligns with Section 3.2: without one-sided constraints, update directions can misalign with trajectory advantages.

By contrast, removing First Token Selection leads to a milder but still clear degradation. On the low-difficulty setting, the model can still achieve a noticeable performance increase in the early stage, showing that latent RL remains learnable without this design, but the improvement is consistently smaller than that of full Latent-GRPO and the training remains less stable. On the high-difficulty setting, the gap becomes much more pronounced, with severe instability appearing much earlier. This is consistent with Section 3.3, where jointly fitting multiple correct paths induces mode averaging at the shared first latent token.

We further ablate the remaining design, **Invalid Sample Advantage Masking**, on the high-difficulty suite by letting overlength trajectories enter both the group statistics and the policy update. Table 3 shows that this design improves accuracy and length control at the same time: re-

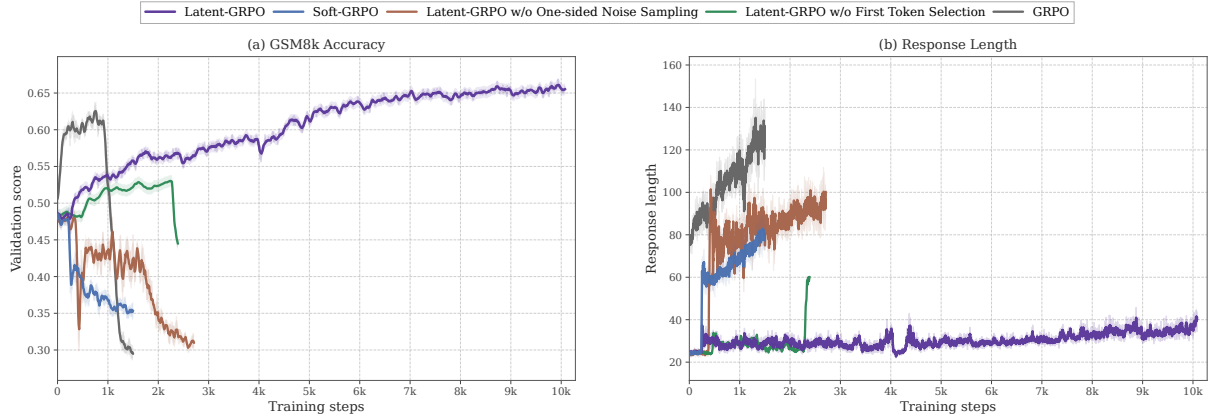


Figure 3: RL training dynamics on the low-difficulty setting for LLaMA-3.2-1B-Instruct. The figure shows validation-set performance together with the evolution of response length during training on GSM8K-Aug. Ablated variants are plotted alongside full Latent-GRPO.

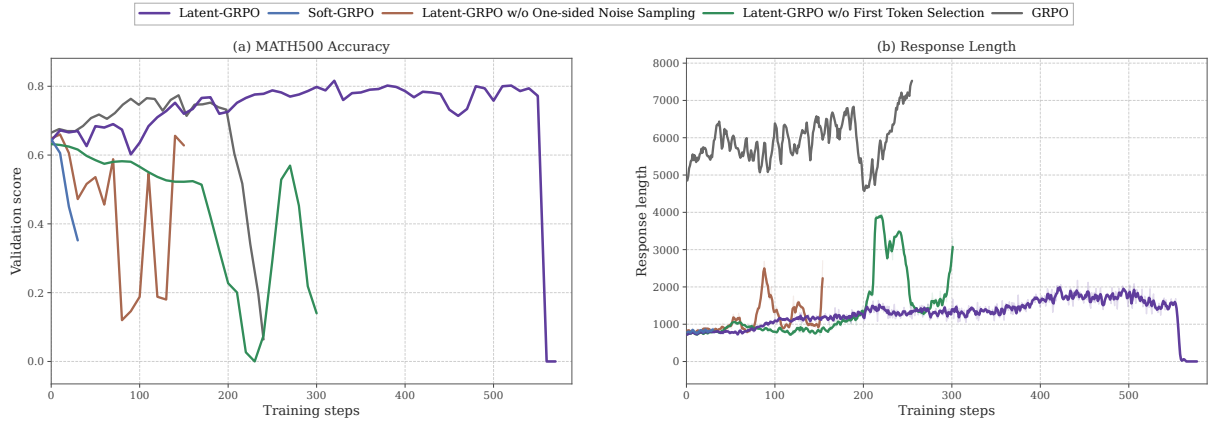


Figure 4: RL training dynamics on the high-difficulty setting for Qwen2.5-Math-7B. The figure shows validation-set performance on Math500 together with the evolution of response length during training on DAPO-Math-En. Ablated variants are plotted alongside full Latent-GRPO.

moving it lowers average Pass@1 from 41.72 to 39.35 (−2.37) while inflating the average reasoning length from 1649 to 1928 tokens (+16.9%). The degradation concentrates on the hardest benchmarks, with AIME25 losing 3.39 points and growing 41.4% longer. This matches the motivation in Section 3.1: off-manifold rollouts fail to emit a stop token and are truncated at $L_{\max}$, so response length directly identifies them, and masking them keeps μ_R and σ_R from depending on the stochastic number of such failures.

Average latent responses are far below the limits (21.2 vs. 128 tokens at low difficulty; 1649 vs. 4096 at high difficulty), so masking cap-reaching samples preserves terminating trajectories without favoring short solutions. Length filtering identifies failures, whereas advantage masking excludes them from optimization.

Overall, One-sided Noise Sampling is the primary driver of training stability, Invalid Sample

Advantage Masking is what keeps exploration bounded and reasoning length under control, and First Token Selection provides an additional improvement that is necessary for stronger and more reliable gains.

5 Conclusion

We study RL post-training for latent reasoning and show that direct GRPO adaptation is unstable due to off-manifold exploration, optimization mismatch under noisy sampling, and harmful joint fitting over multiple correct paths. We propose Latent-GRPO with invalid-sample advantage masking, one-sided noise sampling, and optimal correct-path first-token selection, achieving stable performance gains and substantially shorter reasoning chains than explicit GRPO; pass@ k and ablations confirm its sampling capability and core designs, showing that stable RL post-training is a practical and effective route to efficient latent reasoning.

Limitations

This work focuses on mathematical reasoning benchmarks, where latent reasoning can be evaluated with relatively clear correctness signals. The experiments are also limited to two model families and a fixed latent reasoning framework, so the generality of Latent-GRPO across broader tasks, larger models, and alternative latent representations remains to be further studied.

Ethical Considerations

Latent-GRPO is intended to improve the efficiency of reasoning models, but it may inherit factual errors, biases, and unsafe behaviors from the underlying base models and training data. Because latent reasoning compresses intermediate reasoning into continuous states, it can also reduce interpretability relative to explicit chain-of-thought outputs, making failures harder to diagnose. These risks are especially important in high-stakes domains, where model outputs should be used with human oversight, task-specific evaluation, and appropriate deployment safeguards.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 62676390 and 62276248, by the Beijing Nova Program under Grant No. 20250484765, by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant No. XDB0680302, by the Key Research and Development Program of Xinjiang Uyghur Autonomous Region under Grant No. 2024B03026, by the Director’s Fund Project of State Key Laboratory of AI Safety, and by the Youth Innovation Promotion Association of the Chinese Academy of Sciences under Grant No. 2023111.

References

- Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. 2013. [Estimating or propagating gradients through stochastic neurons for conditional computation](#). *CoRR*, abs/1308.3432.
- Natasha Butt, Ariel Kwiatkowski, Ismail Labiad, Julia Kempe, and Yann Ollivier. 2025. [Soft tokens, hard truths](#). *CoRR*, abs/2509.19170.
- Yukun Cao, Shuo Han, Zengyi Gao, Zezhong Ding, Xike Xie, and S. Kevin Zhou. 2025. [Graphinsight: Unlocking insights in large language models for graph structure understanding](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2025, Vienna, Austria, July 27 - August 1, 2025, pages 12096–12134. Association for Computational Linguistics.
- Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024. [Alphamath almost zero: Process supervision without process](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.
- Xinghao Chen, Anhao Zhao, Heming Xia, Xuan Lu, Hanlin Wang, Yanjun Chen, Wei Zhang, Jian Wang, Wenjie Li, and Xiaoyu Shen. 2025. [Reasoning beyond language: A comprehensive survey on latent chain-of-thought reasoning](#). *CoRR*, abs/2505.16782.
- DeepSeek-AI. 2025. [Deepseek-r1 incentivizes reasoning in llms through reinforcement learning](#). *Nat.*, 645(8081):633–638.
- Jingcheng Deng, Liang Pang, Zihao Wei, Shicheng Xu, Zenghao Duan, Kun Xu, Yang Song, Huawei Shen, and Xueqi Cheng. 2025a. [Latent reasoning in llms as a vocabulary-space superposition](#). *CoRR*, abs/2510.15522.
- Jingcheng Deng, Zihao Wei, Liang Pang, Hanxing Ding, Huawei Shen, and Xueqi Cheng. 2025b. [Everything is editable: Extend knowledge editing to unstructured data in large language models](#). In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.
- Yuntian Deng, Yejin Choi, and Stuart M. Shieber. 2024. [From explicit cot to implicit cot: Learning to internalize cot step by step](#). *CoRR*, abs/2405.14838.
- Zenghao Duan, Liang Pang, Zihao Wei, Wenbin Duan, Yuxin Tian, Shicheng Xu, Jingcheng Deng, Zhiyi Yin, and Xueqi Cheng. 2026a. [Circular reasoning: Understanding self-reinforcing loops in large reasoning models](#). *CoRR*, abs/2601.05693.
- Zenghao Duan, Zhiyi Yin, Zhichao Shi, Liang Pang, Shaoling Jing, Zihe Huang, Jiayi Wu, Yu Yan, Jingcheng Deng, Huawei Shen, and Xueqi Cheng. 2026b. [Projecting out the malice: A global subspace approach to LLM detoxification](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2026, San Diego, California, United States, July 2-7, 2026, pages 35697–35719. Association for Computational Linguistics.
- Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. [PAL: program-aided language models](#). In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, Proceedings of Machine Learning Research, pages 10764–10799. PMLR.

- Shuo Han, Yukun Cao, Zezhong Ding, Zengyi Gao, S. Kevin Zhou, and Xike Xie. 2026. [See or say graphs: Agent-driven scalable graph understanding with vision-language models](#). In *Findings of the Association for Computational Linguistics, ACL 2026, San Diego, California, United States, July 2-7, 2026*, pages 41565–41589. Association for Computational Linguistics.
- Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. [Training large language models to reason in a continuous latent space](#). *CoRR*, abs/2412.06769.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. [Measuring mathematical problem solving with the MATH dataset](#). In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [Lora: Low-rank adaptation of large language models](#). In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.
- Hugging Face. 2025. [Open r1: A fully open reproduction of deepseek-r1](#).
- Eric Jang, Shixiang Gu, and Ben Poole. 2017. [Categorical reparameterization with gumbel-softmax](#). In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.
- MAA. 2024. American invitational mathematics examination (AIME) 2024. [Official MAA webpage](#).
- Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel J. Candès, and Tatsunori Hashimoto. 2025. [s1: Simple test-time scaling](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 20275–20321. Association for Computational Linguistics.
- OpenAI. 2024. [Openai o1 system card](#). *CoRR*, abs/2412.16720.
- Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. [Are NLP models really able to solve simple math word problems?](#) In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2080–2094. Association for Computational Linguistics.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Driani, Julian Michael, and Samuel R. Bowman. 2023. [GPQA: A graduate-level google-proof q&a benchmark](#). *CoRR*, abs/2311.12022.
- Subhro Roy and Dan Roth. 2015. [Solving general arithmetic word problems](#). In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1743–1752. The Association for Computational Linguistics.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. [Proximal policy optimization algorithms](#). *CoRR*, abs/1707.06347.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. [Deepseekmath: Pushing the limits of mathematical reasoning in open language models](#). *CoRR*, abs/2402.03300.
- Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. 2025. [CODI: compressing chain-of-thought into continuous space via self-distillation](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 677–693. Association for Computational Linguistics.
- Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. [Hybridflow: A flexible and efficient RLHF framework](#). In *Proceedings of the Twentieth European Conference on Computer Systems, EuroSys 2025, Rotterdam, The Netherlands, 30 March 2025 - 3 April 2025*, pages 1279–1297. ACM.
- Jingwei Shi, Xinxiang Yin, Jing Huang, Shengyu Tao, and Jinman Zhao. 2026. [Codehacker: Automated test case generation for detecting vulnerabilities in competitive programming solutions](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pages 2352–2382. Association for Computational Linguistics.
- Wenhui Tan, Jiaze Li, Jianzhong Ju, Zhenbo Luo, Jian Luan, and Ruihua Song. 2025. [Think silently, think fast: Dynamic latent compression of LLM reasoning chains](#). *CoRR*, abs/2505.16552.
- Yao Tang, Li Dong, Yaru Hao, Qingxiu Dong, Furu Wei, and Jiatao Gu. 2026. [Multiplex thinking: Reasoning via token-wise branch-and-merge](#). *CoRR*, abs/2601.08808.
- Llama Team. 2024. [The llama 3 herd of models](#). *CoRR*, abs/2407.21783.
- Yifan Wang, Shiyu Li, Peiming Li, Xiaochen Yang, Yang Tang, and Zheng Wei. 2026. [Render-of-thought: Rendering textual chain-of-thought as images for visual latent reasoning](#). *CoRR*, abs/2601.14750.

- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. [Chain-of-thought prompting elicits reasoning in large language models](#). In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.
- Zihao Wei, Liang Pang, Jiahao Liu, Jingcheng Deng, Shicheng Xu, Zenghao Duan, Jingang Wang, Fei Sun, Xunliang Cai, Huawei Shen, and Xueqi Cheng. 2025. [Stop spinning wheels: Mitigating LLM overthinking via mining patterns for early reasoning exit](#). *CoRR*, abs/2508.17627.
- Junhong Wu, Jinliang Lu, Zixuan Ren, Gangqiang Hu, Zhi Wu, Dai Dai, and Hua Wu. 2025. [LLMs are single-threaded reasoners: Demystifying the working mechanism of soft thinking](#). *CoRR*, abs/2508.03440.
- Shicheng Xu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-Seng Chua. 2024. [Search-in-the-chain: Interactively enhancing large language models with search for knowledge-intensive tasks](#). In *Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, May 13-17, 2024*, pages 1362–1373. ACM.
- Shicheng Xu, Liang Pang, Yunchang Zhu, Jia Gu, Zihao Wei, Jingcheng Deng, Feiyang Pan, Huawei Shen, and Xueqi Cheng. 2026. [RLKD: distilling llms’ reasoning via reinforcement learning](#). In *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pages 34151–34159. AAAI Press.
- An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. [Qwen2.5-math technical report: Toward mathematical expert model via self-improvement](#). *CoRR*, abs/2409.12122.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. [Tree of thoughts: Deliberate problem solving with large language models](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.
- Runyang You, Yongqi Li, Meng Liu, Wenjie Wang, Liqiang Nie, and Wenjie Li. 2025. [Parallel test-time scaling for latent reasoning models](#). *CoRR*, abs/2510.07745.
- Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, 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, and 16 others. 2025. [DAPO: an open-source LLM reinforcement learning system at scale](#). *CoRR*, abs/2503.14476.
- Xinlei Yu, Zhangquan Chen, Yongbo He, Tianyu Fu, Cheng Yang, Chengming Xu, Yue Ma, Xiaobin Hu, Zhe Cao, Jie Xu, Guibin Zhang, Jiale Tao, Jiayi Zhang, Siyuan Ma, Kaituo Feng, Haojie Huang, Youxing Li, Ronghao Chen, Huacan Wang, and 18 others. 2026. [The latent space: Foundation, evolution, mechanism, ability, and outlook](#). *CoRR*, abs/2604.02029.
- Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, Shuohang Wang, Yelong Shen, and Xin Eric Wang. 2025. [Soft thinking: Unlocking the reasoning potential of llms in continuous concept space](#). *CoRR*, abs/2505.15778.
- Jinman Zhao, Erxue Min, Hui Wu, Ziheng Li, Zexu Sun, Hengyi Cai, Shuaiqiang Wang, Xu Chen, and Gerald Penn. 2026. [Beyond step pruning: Information theory based step-level optimization for self-refining large language models](#). In *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pages 34941–34949. AAAI Press.
- Zhi Zheng and Wee Sun Lee. 2025. [Soft-grpo: Surpassing discrete-token LLM reinforcement learning via gumbel-reparameterized soft-thinking policy optimization](#). *CoRR*, abs/2511.06411.
- Yufan Zhuang, Liyuan Liu, Chandan Singh, Jingbo Shang, and Jianfeng Gao. 2025. [Text generation beyond discrete token sampling](#). *CoRR*, abs/2505.14827.

A Additional Related Work

LLMs and Explicit Reasoning. Large language models have rapidly become a general foundation for a wide range of tasks (Deng et al., 2025b; Duan et al., 2026b), including mathematical problem solving (Chen et al., 2024), agent scenarios (Yao et al., 2023; Xu et al., 2024; Han et al., 2026), and other complex reasoning settings (Cao et al., 2025; Duan et al., 2026a; Shi et al., 2026; Zhao et al., 2026). A central driver of this progress is *explicit reasoning*, where the model produces discrete chains of thought to decompose a difficult problem into intermediate steps (Wei et al., 2022). Subsequent test-time scaling studies further showed that performance can often be improved by allocating more reasoning tokens at inference time (Muenighoff et al., 2025; OpenAI, 2024). In parallel, RL post-training has become an increasingly important way to strengthen explicit reasoning, especially in mathematical domains. In particular, GRPO shows

that policy optimization over explicit reasoning trajectories can yield strong gains with a simple and effective group-relative objective (Shao et al., 2024), and DeepSeek-R1 further demonstrates that large-scale RL can incentivize strong reasoning behavior in LLMs (DeepSeek-AI, 2025). However, this paradigm remains inherently verbose: better performance is frequently obtained by generating longer and longer reasoning traces, which introduces substantial redundancy, latency, and computational cost. This limitation motivates latent reasoning (Yu et al., 2026; Chen et al., 2025), which shifts the intermediate reasoning process from explicit language into continuous representations in order to retain the benefits of reasoning while using far fewer reasoning tokens.

Training-free Latent Reasoning. Training-free methods modify the inference process without updating model parameters. Soft Thinking constructs each latent token as a probability-weighted mixture of discrete token embeddings, showing that continuous concept tokens can improve reasoning to some extent while remaining interpretable (Zhang et al., 2025). Follow-up analysis in (Wu et al., 2025) argues that vanilla soft thinking is still dominated by the highest-probability token, which leads to a greedy feedback loop and motivates stochastic variants such as Gumbel-Softmax (Jang et al., 2017) for stronger exploration. Parallel test-time scaling further extends inference-time exploration by combining stochastic latent sampling with a latent reward model for trajectory selection in continuous space (You et al., 2025). Overall, these methods show that inference-time stochasticity can make latent reasoning more exploratory, but their performance gains are generally modest and they do not establish the strong reasoning-length advantage targeted in our setting.

Training-based Latent Reasoning. Training-based methods instead learn latent reasoning representations directly. Coconut first showed that feeding continuous hidden states back into the model can support latent reasoning and encode multiple alternative next steps during reasoning (Hao et al., 2024). CODI compresses explicit CoT into latent states via self-distillation and matches explicit CoT on GSM8K with substantial compression (Shen et al., 2025). CoLaR dynamically compresses reasoning chains and further improves them with RL (Tan et al., 2025). Render-of-Thought moves latent reasoning into a visual space by rendering textual

CoT as images, improving traceability while retaining compression (Wang et al., 2026). Latent-SFT develops a unified latent reasoning framework with latent vocabulary, latent chain construction, and stochastic optimization, showing that latent reasoning can outperform explicit SFT with shorter traces (Deng et al., 2025a). More recent work explores RL and stochastic soft decoding more directly. Continuous CoT RL in (Butt et al., 2025) introduces scalable reinforcement learning for continuous CoT, Multiplex Thinking performs token-wise branch-and-merge to optimize stochastic soft rollouts (Tang et al., 2026), and Soft-GRPO adapts GRPO to Gumbel-reparameterized soft thinking (Zheng and Lee, 2025). Overall, training-based methods establish latent reasoning as a strong alternative to explicit CoT. By contrast, our work starts directly from a Latent-SFT model and studies how to achieve stable performance gains with only a small number of reasoning tokens, thereby obtaining dual benefits in both task performance and reasoning efficiency.

B Late-stage Drop in High-difficulty Training

The final drop of Latent-GRPO in Figure 4 differs from the early collapse observed in ablated variants. The full Latent-GRPO model remains effective over most of training, but can eventually reach the boundary of the Latent-SFT-induced latent reasoning manifold. Once Gumbel-perturbed rollouts enter under-trained semantic regions, the decoder may fail to produce a well-formed terminating trajectory. Such samples are truncated at $L_{\max}$ and masked as invalid samples during training. The response-length curve reports only terminating validation outputs and therefore excludes these over-long rollouts; as invalid samples become more frequent near the final checkpoint, the reported length decreases even though individual failed rollouts would hit $L_{\max}$. This suggests that the late-stage drop is a boundary over-exploration effect, rather than a failure of the core Latent-GRPO objective.

C Conditional STE for One-sided Noise Sampling

When repeated PPO updates cross the fixed one-sided target and produce $\Delta_i(\theta) < 0$, we apply a conditional STE directly to the perturbation margin.

Specifically, we define the one-sided margin as

$$\tilde{\Delta}_i(\theta) = \begin{cases} \text{FlipGrad}(\Delta_i(\theta)), & \text{if } \Delta_i(\theta) < 0, \\ \Delta_i(\theta), & \text{otherwise,} \end{cases} \quad (22)$$

where $\text{FlipGrad}(x)$ acts as the identity in the forward pass but negates the backward gradient, i.e.,

$$\nabla_x \text{FlipGrad}(x) = -1. \quad (23)$$

In this way, the PPO ratio still uses the original forward value, while the backward pass treats a crossed target as if its perturbation margin were reflected back to the positive side.

D Gradient Analysis of Exploration-Optimization Misalignment

We provide a gradient-level explanation for the Exploration-Optimization Misalignment discussed in Section 3.2. Consider one latent step and let $\mathcal{V}_t^{(K)}$ denote the top- K candidate set used to construct the latent token, consistent with Section 2.1. During rollout, the perturbed score

$$q_i = g'_i = \log p_i(\theta_{\text{old}}) + \xi_i \quad (24)$$

is fixed for each $i \in \mathcal{V}_t^{(K)}$. During optimization, the current policy changes $\log p_i(\theta)$, so the standard Gumbel perturbation margin is

$$\Delta_i(\theta) = q_i - \log p_i(\theta). \quad (25)$$

The latent-segment Gumbel log-density used by Soft-GRPO can be written as

$$\ell_G(\theta) = \sum_{i \in \mathcal{V}_t^{(K)}} [-\Delta_i(\theta) - \exp(-\Delta_i(\theta))]. \quad (26)$$

Differentiating with respect to the current log-probability of the i -th component gives

$$\frac{\partial \ell_G(\theta)}{\partial \log p_i(\theta)} = 1 - \exp(-\Delta_i(\theta)). \quad (27)$$

In the local region where PPO clipping is inactive and the importance ratio is close to one, the corresponding contribution to the GRPO update has gradient

$$\frac{\partial \mathcal{J}}{\partial \log p_i(\theta)} \approx \hat{A}_j [1 - \exp(-\Delta_i(\theta))]. \quad (28)$$

Therefore, for a positive-advantage trajectory, a component is reinforced only when $\Delta_i(\theta) > 0$.

If $\Delta_i(\theta) < 0$, the same positive-advantage trajectory instead decreases $\log p_i(\theta)$. Conversely, for a negative-advantage trajectory, a negative margin can increase the component probability. Since the original Gumbel perturbation is two-sided, $\Delta_i(\theta_{\text{old}}) = \xi_i$ can be either positive or negative at the beginning of optimization. This is the gradient-level origin of Exploration-Optimization Misalignment.

One-sided Noise Sampling removes this sign ambiguity. In Latent-GRPO, the target is

$$g_i^* = \log p_i(\theta_{\text{old}}) + \xi_i^+, \quad \xi_i^+ > 0, \quad (29)$$

so at the beginning of optimization $\Delta_i(\theta_{\text{old}}) = \xi_i^+ > 0$. If repeated PPO updates later cross the fixed target and produce $\Delta_i(\theta) < 0$, the conditional STE in Appendix C flips the backward gradient of the margin. For the one-sided surrogate log-likelihood

$$\ell_G^+(\theta) = \sum_{i \in \mathcal{V}_t^{(K)}} [-\tilde{\Delta}_i(\theta) - \exp(-\tilde{\Delta}_i(\theta))], \quad (30)$$

the effective derivative becomes

$$\frac{\partial \ell_G^+(\theta)}{\partial \log p_i(\theta)} = \begin{cases} 1 - \exp(-\Delta_i(\theta)), & \Delta_i(\theta) \geq 0, \\ \exp(-\Delta_i(\theta)) - 1, & \Delta_i(\theta) < 0. \end{cases} \quad (31)$$

This derivative is non-negative for all $\Delta_i(\theta)$ and is strictly positive whenever $\Delta_i(\theta) \neq 0$. Because the conditional STE modifies the backward pass, ℓ_G^+ and π_θ^+ should be understood as a surrogate latent likelihood with a custom backward rule rather than a mathematically exact probability density. Consequently, in the unclipped local regime,

$$\frac{\partial \mathcal{J}_{\text{Latent-GRPO}}}{\partial \log p_i(\theta)} \approx \hat{A}_j \frac{\partial \ell_G^+(\theta)}{\partial \log p_i(\theta)}, \quad (32)$$

so the direct component-wise score with respect to $\log p_i(\theta)$ is aligned with the trajectory advantage $\hat{A}_j$ rather than with the accidental sign of a two-sided Gumbel perturbation.

This conclusion is local to the coordinates $\log p_i(\theta)$ and should not be read as a guarantee that every selected raw logit increases individually. Under the full-softmax parameterization, let z_l denote the raw logit of token l and define

$$h_i = \frac{\partial \ell_G^+(\theta)}{\partial \log p_i(\theta)}, \quad H = \sum_{i \in \mathcal{V}_t^{(K)}} h_i. \quad (33)$$

Since

$$\frac{\partial \log p_i(\theta)}{\partial z_l} = \mathbf{1}[i = l] - p_l(\theta), \quad (34)$$

we have

$$\frac{\partial \ell_G^+(\theta)}{\partial z_l} = \sum_{i \in \mathcal{V}_t^{(K)}} h_i (\mathbf{1}[i = l] - p_l(\theta)). \quad (35)$$

Thus, for $l \in \mathcal{V}_t^{(K)}$,

$$\frac{\partial \ell_G^+(\theta)}{\partial z_l} = h_l - p_l(\theta)H, \quad (36)$$

whose sign is not guaranteed to be positive for every selected component due to softmax competition. For $l \notin \mathcal{V}_t^{(K)}$,

$$\frac{\partial \ell_G^+(\theta)}{\partial z_l} = -p_l(\theta)H \leq 0, \quad (37)$$

so non-selected tokens receive an aggregate downward signal when $H \geq 0$. Therefore, One-sided Noise Sampling aligns the direct component-wise score $\partial \ell_G^+ / \partial \log p_i(\theta)$ with the trajectory advantage and induces an aligned aggregate learning signal under softmax normalization, but it does not strictly imply that every selected token logit increases individually.

E Implementation Details

This appendix provides the detailed implementation settings used in our experiments.

Data and Evaluation Setup. In the low-difficulty setting, both the explicit SFT model and the Latent-SFT model are trained on the GSM8K-Aug training split. Because in-domain RL data are unavailable, the RL stage uses an LLM-generated dataset derived from the GSM8K-Aug training set. In the high-difficulty setting, both models are trained on the same subset of Open-R1 (Hugging Face, 2025) with 4K-length reasoning chains, and the RL stage uses DAPO-Math-En (Yu et al., 2025).

Latent-SFT. We first describe the implementation details of the Latent-SFT initialization used in our experiments. To facilitate downstream Latent-GRPO training, we set $K = 10$ in Section 2.1, so that each latent token is constructed from the top-10 explicit tokens. In practice, using a larger K increases the risk of out-of-memory issues during Latent-GRPO training.

Hyperparameter	LLaMA-3.2-1B	Qwen2.5-Math-7B
Train batch size	64	32
PPO mini-batch size	16	32
Rollouts	8	8
Max response length	128	4096

Table 4: Latent-GRPO hyperparameters for the two experimental settings.

For the training setup, in Stage 1 we train 10 epochs per step with a learning rate of 5×10^{-5} and a compression ratio of 2. In Stage 2, the learning rate is set to 3×10^{-4} and training runs for 70 epochs. We further add the Gumbel training objective with Gumbel noise set to 1.0 and Gumbel temperature set to 1.0.

Latent-GRPO. We then describe the implementation details of Latent-GRPO. For each setting, we select the best-performing Latent-SFT checkpoint as the initialization model for Latent-GRPO. The learning rate is fixed at 1×10^{-6} throughout training.

SFT. For completeness, we also summarize the implementation details of the explicit SFT baseline. We train the SFT models with LoRA (Hu et al., 2022), using a uniform rank of 64 and 10 training epochs for all settings. The learning rate is set to 5×10^{-5} for LLaMA-3.2-1B-Instruct and 2×10^{-5} for Qwen2.5-Math-7B.

GRPO. Finally, we summarize the implementation details of the explicit GRPO baseline. We use version 0.7.0 of ver1 (Sheng et al., 2025) and run the experiments with the official scripts. The released Latent-GRPO implementation instead builds on a modified ver1 0.4.x codebase; version 0.7.0 is used only for the explicit GRPO baseline. The learning rate is uniformly set to 1×10^{-6} . For LLaMA-3.2-1B-Instruct, we set `max_response_length=512`. For Qwen2.5-Math-7B, we set `max_response_length=8192`.

All experiments are conducted on 8 A100 GPUs.

F Use of AI Assistance

AI tools were used only for language polishing, including grammar, clarity, and fluency improvements, and for reviewing parts of the code. All scientific content, claims, experiments, analyses, code, and final writing were reviewed and verified by the authors.

Algorithm 1 Latent-GRPO Training Procedure

Require: Prompt distribution $\mathcal{P}(\mathcal{X})$; Latent-SFT initialized policy π_θ ; reference policy π_{ref} ; group size G ; top- K size; maximum length L_{max} ; Gumbel temperature τ_g ; one-sided noise parameters a, b, δ ; PPO clipping coefficient ϵ_{clip} ; KL coefficient β .

- 1: Initialize π_θ from the Latent-SFT checkpoint.
- 2: **for** each RL iteration **do**
- 3: Set the rollout policy $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$.
- 4: **for** each prompt $x \sim \mathcal{P}(\mathcal{X})$ **do**
- 5: Sample a group of G latent-discrete trajectories $\{o_j\}_{j=1}^G$ with $\pi_{\theta_{\text{old}}}$.
- 6: **for** each trajectory o_j **do**
- 7: **for** each latent step $t \leq T_{\text{lat}}^{(j)}$ **do**
- 8: Obtain top- K probabilities $p_{j,t,i}(\theta_{\text{old}})$ and embeddings $e_{j,t,i}$.
- 9: Sample $\xi_{j,t,i} \sim \text{Gumbel}(0, 1)$ and construct $\xi_{j,t,i}^+ = \text{clip}(\xi_{j,t,i}, -a, b) + a + \delta$.
- 10: Set the fixed one-sided target $g_{j,t,i}^* = \log p_{j,t,i}(\theta_{\text{old}}) + \xi_{j,t,i}^+$.
- 11: Construct the latent token $s_{j,t} = \sum_{i=1}^K \text{softmax}(g_{j,t,i}^*/\tau_g)_i e_{j,t,i}$, and store $\{g_{j,t,i}^*\}_{i=1}^K$.
- 12: **end for**
- 13: Generate the explicit suffix, compute reward R_j , and record total length L_j .
- 14: **end for**
- 15: **Invalid-sample advantage masking.**
- 16: Define $G_{\text{valid}} = \{j \mid L_j < L_{\text{max}}\}$.
- 17: Compute μ_R and σ_R only over G_{valid} , and obtain $\hat{A}_j^{\text{valid}}$ by valid-sample normalization.
- 18: If $|G_{\text{valid}}| = 0$ or $\sigma_R = 0$, set $\hat{A}_j^{\text{valid}} = 0$ for all j .
- 19: **Correct-path first-token selection.**
- 20: Define $G_{\text{correct}} = \{j \mid o_j \text{ is verified as correct}\}$.
- 21: If $|G_{\text{correct}}| > 1$, compute $\bar{\ell}_j$ using the average rollout surrogate log-probability and select $j^* = \arg \max_{j \in G_{\text{correct}}} \bar{\ell}_j$.
- 22: If $|G_{\text{correct}}| = 1$, let j^* be the sole correct trajectory; otherwise set $j^* = \emptyset$.
- 23: Define the first-token selection mask $M_{j,t}$ and the final advantage $\tilde{A}_{j,t} = M_{j,t} \hat{A}_j^{\text{valid}}$.
- 24: **end for**
- 25: **PPO-style Latent-GRPO update.**
- 26: **for** each PPO epoch and minibatch **do**
- 27: **for** each stored latent step $t \leq T_{\text{lat}}^{(j)}$ **do**
- 28: Compute $\Delta_{j,t,i}(\theta) = g_{j,t,i}^* - \log p_{j,t,i}(\theta)$.
- 29: Apply the conditional STE from Appendix C to obtain $\tilde{\Delta}_{j,t,i}(\theta)$.
- 30: Compute the surrogate latent likelihood $\pi_\theta^+(s_{j,t} \mid x, s_{j,<t})$.
- 31: **end for**
- 32: For explicit steps, use the standard categorical likelihood.
- 33: Compute the modified ratio $r_{j,t}^*(\theta)$.
- 34: Update θ by maximizing the Latent-GRPO objective.
- 35: **end for**
- 36: **end for**
