TPPO: TRACKING-VALUE PPO FOR LANGUAGE MODEL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Proximal Policy Optimization (PPO) is widely used in reinforcement learning with verifiable rewards for language-model reasoning. While *value evolution along trajectory* characterizes relative token-level credit, yet standard critics often fail to track it under sparse terminal rewards. To model this evolution, we analyze the relation between successive prefix values and show that they admit a Bayesian decomposition with additive updates in log-odds space. We use this additive structure to construct Tracking-Value PPO (TPPO), a cumulative value parameterization that sums position-wise scalar outputs into value logits. These outputs are learned through terminal-outcome supervision on the reconstructed values. TPPO improves aggregate mathematical reasoning scores over PPO by 7.0 and 8.4 points on Qwen3-4B-Base and Qwen3-8B-Base, respectively, while improving out-of-distribution generalization. Gains extend to Qwen3-4B, Qwen2.5-Math-1.5B, Qwen3.5-9B and to countdown, and scientific coding tasks. Further analysis shows more accurate tracking of value evolution and more reliable advantage estimates.

1 INTRODUCTION

Reinforcement Learning with Verifiable Rewards (RLVR) has substantially improved the reasoning capabilities of large language models (LLMs), enabling them to perform extended chain-of-thought (CoT) reasoning (Guo et al., 2025; Jaech et al., 2024). LLM reasoning often involves a long chain of token-level actions before arriving at a solution, making credit assignment during reinforcement learning crucial. In classical RL, the state-value function $V^\pi(\mathbf{s})$ provides a way to estimate the expected return from each state, thereby supporting credit assignment throughout the trajectory (Sutton et al., 1998). Proximal Policy Optimization (PPO) (Schulman et al., 2017) implements this idea with a critic network that estimates state values and uses them to compute advantages for policy updates.

However, assigning token-level credit with PPO remains challenging in long-CoT reasoning (Yuan et al., 2025; Chen et al., 2026; Wang et al., 2026). More important than independently estimating each reasoning prefix, the key difficulty is tracking how value evolves along the reasoning trajectory, $\Delta V_t^\pi = V^\pi(\mathbf{s}_{t+1}) - V^\pi(\mathbf{s}_t)$, which characterizes token-level credit under policy π. In standard PPO, the critic predicts each prefix value independently and does not explicitly model this evolution. Empirically, Figure 1(a) illustrates a sharp drop from the correct step "*At* $a = 0$*:* $\sin^2 x = 0$" to the incorrect claim "*The only solution is* $x = 0$", with $\Delta V_{\mathrm{MC}} = -0.992$. However, the PPO critic predicts $\Delta V_{\mathrm{PPO}} = +0.036$, missing the drop and reversing the MC-based GAE advantage ordering at the two anchors ($\gamma = \lambda = 1$). This problem is systematic: the sign accuracy remains near chance and decreases from 50.34% to 45.91% during training, as shown in Figure 1(b). This observation motivates a central question:

How can a critic better model and learn value evolution across successive states?

To answer this question, we characterize how state values evolve as reasoning unfolds (Section 4.1). Under binary terminal rewards, each prefix value is the probability that its continuation succeeds. Conditioning on the next token updates this probability, and Bayes' rule shows that the resulting updates become additive when represented in log-odds space (Equation 8). To use this additive structure in modeling value evolution, we design TPPO (Tracking-Value PPO; Section 4.2), a critic parameterization that accumulates position-wise scalar outputs into value logits and applies a sigmoid to reconstruct prefix values, as illustrated in Figure 2. The critic uses the same terminal outcomes as

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.

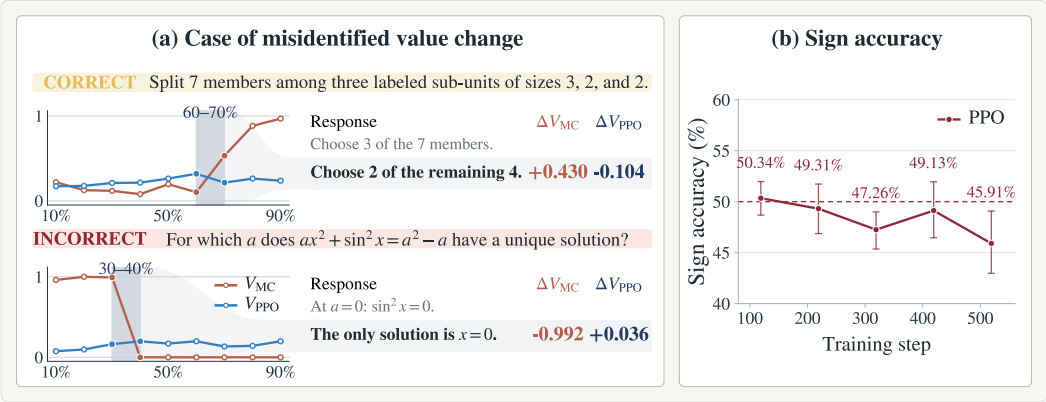

Figure 1: **Critics mispredict how value evolves along reasoning.** (a) Representative responses showing that different steps within a response can cause different MC-estimated value changes; the PPO critic predicts changes with opposite signs in these examples. (b) The sign accuracy of these predictions decreases during training. MC evaluation details are in Appendix D.3.

standard PPO and requires neither additional rollouts nor process-level annotations. The reconstructed state values can be directly incorporated into GAE and the standard PPO objective.

Extensive experiments with Qwen3-4B-Base, Qwen3-8B-Base, and Qwen3-4B demonstrate the effectiveness of TPPO across diverse reasoning tasks (Section 5). On mathematical reasoning, TPPO improves both in-distribution performance and out-of-distribution generalization. These gains extend consistently to countdown, scientific coding tasks, where TPPO outperforms the corresponding PPO baselines. Further analysis shows that TPPO more accurately tracks how values evolve along reasoning, leading to more reliable advantage estimates. Our contributions are as follows:

- **Characterizing how value evolves along reasoning.** We reveal an empirical discrepancy between Monte Carlo estimates of value evolution and the evolution captured by standard PPO critics, highlighting the challenge of learning fine-grained credit signals under sparse rewards.
- **A cumulative parameterization of prefix values.** We use the additive structure of prefix values in log-odds space to construct TPPO, a critic that directly modeling value evolution and accumulates learned scalar increments and is trained with terminal-outcome supervision.
- **Improved policy optimization.** Experiments across mathematical reasoning, Countdown, and scientific coding demonstrate that TPPO improves reasoning performance and enhances the quality of training signals across models.

2 Related Work

Reinforcement Learning and Credit Assignment for LLM Reasoning Reinforcement learning has become a widely used approach for improving LLM reasoning. Value-free methods such as GRPO (Shao et al., 2024) and DAPO (Yu et al., 2026) replace learned value with intra-group reward. They treat the entire response as a single action and apply the same state value to every token. Value-based methods such as Proximal Policy Optimization (Schulman et al., 2017) use a learned value function to estimate advantages via GAE (Schulman et al., 2015). Under sparse terminal rewards, assigning credit to intermediate states remains challenging. One line of work uses Monte Carlo sampling or tree search to obtain more accurate value estimates for intermediate states (Kazemnejad et al., 2024; Hou et al.; Li et al., 2025). These methods require additional sampling or search during training. Another line of work introduces process-level supervision to provide denser feedback over intermediate reasoning steps (Lightman et al., 2024; Wang et al., 2024; Zhang et al., 2025; Setlur et al., 2025; Qu et al., 2025; Cui et al., 2025; Chen et al., 2026). Such methods typically rely on additional process supervision or auxiliary reward models. In contrast, we focus on value-function learning as a way to improve credit assignment, without introducing additional supervision or rollout computation.

Value Function Learning Prior work also studied how value function should be learned. Distributional RL (Bellemare et al., 2017) models distributions over returns, while Stop Regressing (Fare-

brother et al., 2024) shows that categorical classification can improve value learning. HL-Gauss PPO (Zhou et al., 2026) recently applies classification-based value learning to PPO critics for LLM reinforcement learning. VAPO (Yue et al., 2025) develops a value-based PPO recipe for long-chain-of-thought training, addressing value bias, heterogeneous sequence lengths, and sparse rewards through its training design. SPPO (Wang et al., 2026) reformulates long-horizon reasoning as a sequence-level contextual bandit and uses a decoupled scalar value function to avoid unstable token-level credit assignment in PPO. TPPO addresses a different aspect of value learning: it parameterizes prefix values through cumulative scalar outputs in log-odds space.

3 PPO Critics Struggle to Track Value Evolution

In this section, we formulate autoregressive reasoning as a token-level Markov decision process and review PPO value estimation for token-level credit assignment. For distinguishing credit between successive tokens, we focus on value evolution across states. Standard PPO critics do not explicitly model this evolution, and our empirical analysis shows that they fail to track it.

3.1 Values and Token-level Credit

We model autoregressive generation as a token-level Markov decision process (Trung et al., 2024; Jaech et al., 2024). Given a prompt $\mathbf{x}$, the policy π_θ samples a response $\mathbf{y} = (y_1, \ldots, y_T)$ for a prompt $\mathbf{x}$. At time t, the state is $\mathbf{s}_t = (\mathbf{x}, \mathbf{y}_{<t})$ and the token action is $a_t = y_t \sim \pi_\theta(\cdot \mid \mathbf{s}_t)$. In the reinforcement learning with verifiable rewards setting, a scalar reward is assigned only at the terminal step: $r_t = 0$ for $t < T$ and $r_T = R(\mathbf{x}, \mathbf{y}) \in \{0, 1\}$, with the discount factor set to $\gamma = 1$ (Guo et al., 2025; Shao et al., 2024; Ouyang et al., 2022).

Policy Gradient and Advantage Functions. Given the MDP formulation, policy gradient methods optimize the expected terminal reward by assigning a credit signal to each sampled token:

$$J(\theta) \doteq \mathbb{E}_{\mathbf{x}\sim\mathcal{D},\, \mathbf{y}\sim\pi_\theta(\cdot|\mathbf{x})} \left[R(\mathbf{x}, \mathbf{y}) \right], \tag{1}$$

$$\mathbf{g}_{\text{pg}} \doteq \nabla_\theta J(\theta) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D},\, \mathbf{y}\sim\pi_\theta(\cdot|\mathbf{x})} \left[\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t \mid \mathbf{s}_t) A^{\pi_\theta}(\mathbf{s}_t, a_t) \right]. \tag{2}$$

Here, $\mathcal{D}$ is the prompt distribution, and the advantage function $A^{\pi_\theta}(\mathbf{s}_t, a_t)$ measures how much better token a_t is than the policy's average choice at state $\mathbf{s}_t$. The sign and absolute magnitude of $A^{\pi_\theta}(\mathbf{s}_t, a_t)$ determine the direction and weight of the token's contribution to the policy gradient.

Advantage Estimation and Value Functions. Generalized Advantage Estimation (GAE) (Schulman et al., 2015) is a widely used advantage estimator that controls the bias–variance trade-off by combining temporal-difference (TD) residuals. With critic predictions $\widehat{V}_t = V_\phi(\mathbf{s}_t)$, $\gamma = \lambda = 1$, and terminal rewards, the GAE estimate reduces to:

$$\widehat{A}_t^{\text{GAE}} = \sum_{\ell=0}^{T-t} (\gamma\lambda)^\ell \widehat{\delta}_{t+\ell} \quad = \sum_{k=t}^{T} \widehat{\delta}_k = R(\mathbf{x}, \mathbf{y}) - \widehat{V}_t. \tag{3}$$

where $\widehat{\delta}_t \doteq r_t + \gamma \widehat{V}_{t+1} - \widehat{V}_t$ is the critic-based one-step TD residual, with $\widehat{V}_{T+1} = 0$ at the terminal state. The true value $V^{\pi_\theta}(\mathbf{s}_t)$ is the expected return from the current prefix. In this setting, the return is $G_t \doteq \sum_{k=t}^{T} r_k = R(\mathbf{x}, \mathbf{y})$, so

$$V^{\pi_\theta}(\mathbf{s}_t) \doteq \mathbb{E}_{\pi_\theta} \left[G_t \mid S_t = \mathbf{s}_t \right] = \Pr_{\pi_\theta} \left[R(\mathbf{x}, \mathbf{y}) = 1 \mid S_t = \mathbf{s}_t \right]. \tag{4}$$

Thus, $V^{\pi_\theta}(\mathbf{s}_t)$ is the probability that a continuation from the current prefix succeeds.

Estimating Advantage via Value Networks. Since the true value $V^{\pi_\theta}(\mathbf{s}_t)$ is unavailable in practice, PPO learns a value network V_ϕ to approximate $V^{\pi_\theta}(\mathbf{s}_t)$. Its predictions are used in Equation 3 to obtain advantage estimates. The network parameters ϕ are optimized alongside the policy π_θ by minimizing the mean squared error between predicted values and sampled returns G_t. For a batch $\mathcal{D}_{\pi_\theta} = \{\mathbf{y}^{(n)}\}_{n=1}^{N}$, with response n having T_n valid response tokens, this objective is

$$\widehat{\mathcal{L}}^{\text{VF}}(\phi) \doteq \frac{1}{\sum_{n=1}^{N} T_n} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \left(V_\phi(\mathbf{s}_t^{(n)}) - G_t^{(n)} \right)^2. \tag{5}$$

Under terminal rewards, $G_1^{(n)} = \cdots = G_{T_n}^{(n)} = R(\mathbf{y}^{(n)})$, every state visited by the same response receives the same observed binary target. Sharing a sampled return does not imply equal values: the expected target at each prefix remains $V^{\pi_\theta}(\mathbf{s}_t)$.

3.2 PPO Fails to Track Value Evolution

Value evolution determines relative token-level credit. Although all tokens in a response share the same terminal reward, they need not deserve the same credit. Value evolution is the change in the probability of eventual success as each token extends the prefix. Different steps in the same response can therefore have very different effects: Figure 1(a) shows a sharp MC-estimated drop across the incorrect claim "The only solution is $x = 0$" ($\Delta V_{\mathrm{MC}} = -0.992$), but an increase across the correct step "Choose 2 of the remaining 4" ($\Delta V_{\mathrm{MC}} = +0.430$). These changes indicate which steps should receive less or more relative credit. From Equation 3, the change equals the difference between adjacent tokens' advantages: for $t < T$,

$$\widehat{A}_t^{\mathrm{GAE}} - \widehat{A}_{t+1}^{\mathrm{GAE}} = \widehat{V}_{t+1} - \widehat{V}_t \doteq \Delta\widehat{V}_t. \tag{6}$$

$\Delta\widehat{V}_t > 0$ indicates increased success probability and gives a_t a larger estimated advantage than a_{t+1}; $\Delta\widehat{V}_t < 0$ indicates a decrease and reverses this ordering. Thus, predicting the opposite sign of $\Delta\widehat{V}_t$ in Figure 1 indicates a reversal of the relative credit. Since advantages weight token gradients in Equation 2, this misordering changes the relative weights of their contributions to the sampled policy update. We discuss in Appendix B.2 how value evolution affects advantage estimation for different λ.

Empirical Observation We evaluate whether PPO critics recover the direction of local value evolution. We estimate prefix values using Monte Carlo (MC) continuations (Kazemnejad et al., 2024; Chen et al., 2026) at fixed fractions of each response and measure sign agreement between MC-estimated and critic-predicted differences across adjacent anchors. Appendix B.3 details the estimation procedure and the connection between anchor-level changes and relative GAE advantages. Using this protocol, we find that the sign errors shown in Figure 1(a) persist across training: aggregate sign accuracy stays near chance, falling from 50.34% to 45.91% between the first and last evaluated checkpoints (Figure 1(b)). Continued training does not reliably recover the direction of value evolution, and poor value training introduces noisy credit assignment. This suggests that value evolution itself should be explicitly captured in the critic.

4 TPPO: Tracking-Value PPO

Our goal is to explicitly model value evolution across successive prefixes. Analyzing the relation between $V^{\pi_\theta}(\mathbf{s}_t)$ and $V^{\pi_\theta}(\mathbf{s}_{t+1})$, we find a Bayesian update structure that is additive in log-odds space. We use this structure to construct the cumulative logit parameterization of TPPO.

4.1 Bayesian Structure of Value Evolution

Recall from Equation 4 that $V^{\pi_\theta}(\mathbf{s}_t)$ is the conditional probability that a continuation from $\mathbf{s}_t$ eventually succeeds. After sampling token a_t, the next state $\mathbf{s}_{t+1} = (\mathbf{s}_t, a_t)$ conditions this probability on one additional observation. The transition from $V^{\pi_\theta}(\mathbf{s}_t)$ to $V^{\pi_\theta}(\mathbf{s}_{t+1})$ can therefore be written as a Bayesian update.

Proposition 4.1 (Bayesian update of value)**.** *Assume a fixed policy π_θ (with θ held fixed), a binary terminal reward, $\gamma = 1$, and $0 < V^{\pi_\theta}(\mathbf{s}_t) < 1$. For any sampled token a_t for which the conditional probabilities below are positive,*

$$V^{\pi_\theta}(\mathbf{s}_{t+1}) = \frac{V^{\pi_\theta}(\mathbf{s}_t)\Pr_{\pi_\theta}\left[a_t \mid \mathbf{s}_t, R=1\right]}{V^{\pi_\theta}(\mathbf{s}_t)\Pr_{\pi_\theta}\left[a_t \mid \mathbf{s}_t, R=1\right] + \left(1 - V^{\pi_\theta}(\mathbf{s}_t)\right)\Pr_{\pi_\theta}\left[a_t \mid \mathbf{s}_t, R=0\right]}. \tag{7}$$

Equivalently, taking log-odds yields

$$\operatorname{logit} V^{\pi_\theta}(\mathbf{s}_{t+1}) - \operatorname{logit} V^{\pi_\theta}(\mathbf{s}_t) = \log \frac{\Pr_{\pi_\theta}\left[a_t \mid \mathbf{s}_t, R=1\right]}{\Pr_{\pi_\theta}\left[a_t \mid \mathbf{s}_t, R=0\right]} \doteq \Delta_{t+1}^{\pi_\theta}. \tag{8}$$

The derivation, together with the boundary cases $V^{\pi_\theta}(\mathbf{s}_t) \in \{0, 1\}$, is provided in Appendix C.1. Equation 8 characterizes the true values under a fixed policy: their successive log-odds changes equal token-specific log-likelihood ratios. Applying this identity recursively along the trajectory yields the following decomposition.

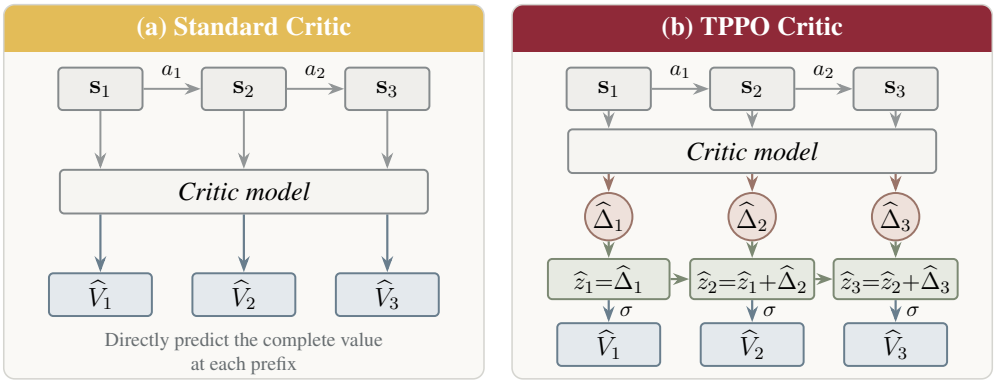

Figure 2: **Difference between standard and TPPO critic parameterizations.** (a) A standard critic directly predicts a value for each prefix. (b) TPPO accumulates token-wise critic outputs into value logits, then applies a sigmoid to obtain prefix values.

4.2 Method

To utilize the structure of value evolution derived in Proposition 4.1, we first unroll the Bayesian updates along a response.

Corollary 4.2 (Incremental decomposition of values)**.** *Applying Equation 8 recursively gives*

$$V^{\pi_\theta}(\mathbf{s}_t) = \sigma\left(\sum_{i=1}^{t} c_i^{\pi_\theta}\right), \quad c_1^{\pi_\theta} = \operatorname{logit} V^{\pi_\theta}(\mathbf{s}_1), \quad c_i^{\pi_\theta} = \Delta_i^{\pi_\theta}, \quad i > 1. \tag{9}$$

To model value evolution using this cumulative form, we propose Tracking-Value PPO (TPPO), which parameterizes the critic as an accumulation of position-wise scalar outputs: $\widehat{V}_t = \sigma\left(\sum_{i=1}^{t} f_\phi(\mathbf{s}_i)\right)$. The resulting increments describe the model's own value sequence; the Bayesian identity does not guarantee that learned outputs recover the true log-likelihood ratios in Equation 8. The comparison with a standard critic model is illustrated in Figure 2.

Under binary terminal rewards, $V^{\pi_\theta}(\mathbf{s}_t) = \Pr_{\pi_\theta}(R = 1 \mid \mathbf{s}_t)$. We train each reconstructed value against the terminal outcome $R(\mathbf{y}) \in \{0, 1\}$ using

$$\mathcal{L}_V = -\frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \left[R(\mathbf{y}) \log \widehat{V}_t + \big(1 - R(\mathbf{y})\big) \log\big(1 - \widehat{V}_t\big) \right], \tag{10}$$

where $\mathcal{T}$ denotes the valid response positions. The scalar outputs are learned jointly through this value objective, without explicit targets for log-odds changes. Each increment receives gradients from all subsequent reconstructed values that depend on it (Appendix C.2).

Cumulative parameterization improves the tracking of value evolution. To isolate the effect of parameterization, we fix the actor and train critics using either the standard parameterization with MSE or BCE, or the cumulative parameterization (TPPO) with BCE. For the BCE objective, the standard critic uses a sigmoid output to match the bounded value targets. We conduct this controlled comparison on the Qwen3-8B-Base mathematical reasoning task. As shown in Table 1, TPPO improves the directional accuracy of local value evolution over the standard BCE critic by 2.01 and 2.80 percentage points at $\tau = 0$ and $\tau = 1$, respectively. Because both critics use the same BCE objective, this improvement isolates the contribution of the cumulative parameterization and shows that its benefit extends beyond the choice of loss function.

Train with PPO The resulting $\widehat{V}_t$ is used in the standard GAE estimator, and the policy is optimized with the usual PPO clipped objective (Schulman et al., 2017),

$$\mathcal{L}_{\text{PPO}} = -\frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \min\Big(\rho_t(\theta)\widehat{A}_t,\ \operatorname{clip}\big(\rho_t(\theta), 1-\epsilon, 1+\epsilon\big)\widehat{A}_t \Big), \tag{11}$$

where $\rho_t(\theta) = \pi_\theta(a_t \mid \mathbf{s}_t)/\pi_{\text{old}}(a_t \mid \mathbf{s}_t)$. Apart from the critic parameterization, its advantage estimation and policy optimization follow standard PPO.

Table 1: **Directional accuracy of local value evolution under a fixed actor.** Entries are accuracies (%) with 95% confidence intervals in brackets, computed on intervals satisfying $\mathrm{SNR} > \tau$. The metric compares the signs of critic-predicted and MC-estimated local value evolution between evaluation anchors (Appendix D.3).

Critic + loss	Sign accuracy ($\tau = 0$)	Sign accuracy ($\tau = 1$)
Standard critic + MSE	50.51 [49.15, 51.79]	51.19 [49.57, 52.69]
Standard critic + BCE	50.70 [49.29, 52.13]	51.14 [49.54, 52.74]
TPPO critic + BCE	**52.71** [51.67, 53.77]	**53.94** [52.59, 55.33]

5 Experiments

In this section, we evaluate TPPO in terms of reasoning performance, the critic's ability to track value evolution, and the quality of advantage estimates. Specifically, we ask:

- **RQ1: Policy performance.** Does TPPO improve policy performance across model scales and tasks?
- **RQ2: Tracking value evolution.** Does the cumulative critic better track value evolution and estimate prefix values along reasoning trajectories?
- **RQ3: Credit assignment.** Do better value estimates and value-evolution tracking produce more accurate advantage estimates for policy updates?

5.1 Experimental Setup

Evaluation. For mathematical reasoning, we evaluate on ten benchmarks: in-domain tasks (AIME24/25/26, AMC (Li et al., 2024), MATH-500 (Hendrycks et al., 2021b), Minerva (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024)) and OOD tasks (ARC-C (Clark et al., 2018), AGI-Eval (Zhong et al., 2024), and BBH (Suzgun et al., 2023)). We report Avg@32 on AIME and AMC, Avg@4 on the remaining benchmarks, using temperature 1.0 and a maximum response length of 24k tokens. For scientific coding, we evaluate SciCode (Tian et al., 2024), HumanEval (Chen et al., 2021), HumanEvalFix (Muennighoff et al., 2024), LiveCodeBench v6 (Jain et al., 2025), and APPS Introductory (Hendrycks et al., 2021a), averaging three evaluation runs.

RLVR setup. For mathematical reasoning, we train on approximately 30k examples from Polaris (An et al., 2025), with $n \in \{1, 8\}$ sampled responses per prompt. Training responses are sampled at temperature 1.0 with a maximum length of 8,192 tokens. We use Math-Verify[1] to compute outcome rewards for mathematical reasoning. Experiments use the Slime framework (Zhu et al., 2025) on nodes with eight H200 GPUs each. For scientific coding, we train on data from Sciwalker (Li et al., 2026). Further implementation details are provided in Appendix D. In all main experiments, the position-wise outputs use a shared scalar head. We discuss and evaluate a separate-head alternative in Appendix E.1.

Models and baselines. Our main experiments use Qwen3-4B-Base and Qwen3-8B-Base (Yang et al., 2025). The $n = 1$ setting also includes Qwen3-4B, with thinking disabled during training. We compare TPPO with PPO (Schulman et al., 2017), GRPO (Shao et al., 2024), SPPO (Wang et al., 2026), and VAPO (Yue et al., 2025), using the same training dataset and evaluation protocol within each setting. We also report each model's performance before RL training. To test TPPO on another task and a smaller model, we train Qwen2.5-Math-1.5B on Countdown problems with three or four input numbers and evaluate on problems with three, four, or five input numbers.

Critic evaluation. At five Qwen3-8B-Base training checkpoints, we evaluate PPO and TPPO on a fixed subset of 64 prompts from the training set. The same prompts are used for both methods at every checkpoint, with each method's actor sampling 8 responses per prompt, yielding approximately 5.1k rollouts across both methods and all five checkpoints. We estimate MC values at nine anchors spanning 10–90% of each response, allocating 128 continuations per anchor (approximately 5.9M valid MC continuations in total). We measure agreement with these estimates in the direction of local value evolution and in advantages. Sampling details, metric definitions, and bootstrap estimation of the gains over PPO are provided in Appendix D.3.

[1] https://github.com/huggingface/Math-Verify

Table 2: **In-domain and out-of-domain results with $n = 8$.** All entries are accuracies (%). Orange parenthetical values show the absolute improvement over PPO in percentage points. Bold and underlined values indicate the best and second-best results within each model group, respectively.

Method	In-Domain Performance						Out-of-Domain Performance			
	AIME24/25/26	AMC	MATH500	Minerva	Olympiad	Avg.	ARC-C	AGI-Eval	BBH	OOD Avg.
Qwen3-4B-Base										
Base	4.3/3.3/3.4	27.3	43.2	22.1	22.3	18.0	34.8	24.8	12.8	24.1
GRPO	12.1/11.4/8.1	51.9	76.9	**47.6**	40.1	35.4	90.2	63.7	42.9	65.6
PPO	13.5/9.5/9.3	53.8	76.1	41.2	39.1	34.6	88.7	59.0	40.7	62.8
VAPO	8.3/8.3/6.7	48.8	72.2	44.9	34.9	32.0	88.7	58.1	41.2	62.7
SPPO	9.2/15.0/7.5	58.1	77.5	40.8	41.3	35.6	90.1	63.0	40.5	64.5
TPPO	**18.5/18.4/18.5**	**62.5**	**81.8**	46.5	**45.3**	**41.6** (+7.0)	**90.6**	**66.5**	**43.7**	**66.9**
Qwen3-8B-Base										
Base	8.2/6.8/6.4	33.8	54.3	24.9	27.2	23.1	62.4	44.2	27.8	44.8
GRPO	19.7/14.1/9.4	59.5	79.6	46.0	41.2	38.5	93.0	67.7	34.8	65.2
PPO	15.8/12.5/11.3	61.6	78.8	40.8	42.0	37.5	92.2	68.4	42.2	67.6
VAPO	14.2/13.3/8.3	59.4	79.6	47.4	40.8	37.6	91.9	67.6	40.6	66.7
SPPO	14.2/14.2/7.5	61.3	80.1	45.6	42.8	38.0	92.9	69.9	40.7	67.8
TPPO	**25.8/22.3/20.5**	**67.3**	**85.4**	**49.8**	**50.2**	**45.9** (+8.4)	**93.1**	**72.0**	**43.9**	**69.7**

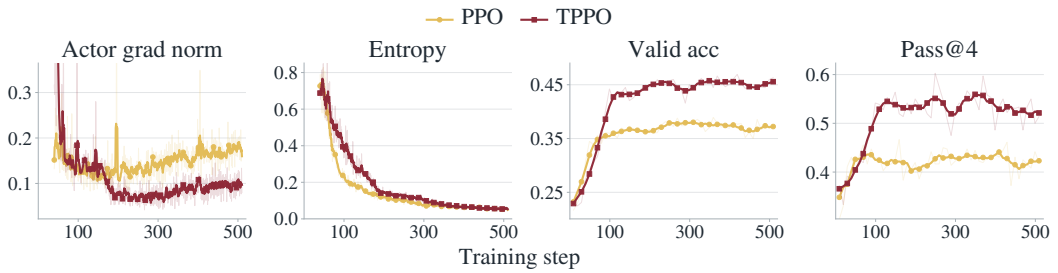

Figure 3: **Training dynamics comparing PPO and TPPO on Qwen3-8B-Base ($n = 8$).** We track actor gradient norm, policy entropy, validation accuracy, and Pass@4 over training. TPPO establishes an early lead over PPO in both performance metrics and retains it through the final checkpoint.

5.2 Results

RQ1: TPPO improves policy performance across model sizes and tasks. Tables 2 and 3 show consistent in-domain gains over PPO across model sizes and rollout budgets. With $n = 8$, TPPO improves average accuracy by 7.0 and 8.4 percentage points on Qwen3-4B-Base and Qwen3-8B-Base, respectively, outperforming baselines. The gains persist with $n = 1$, including an average improvement of +10.1 points on Qwen3-4B. Figure 3 shows an early and sustained lead in validation accuracy and Pass@4, with smaller actor gradient norms and slower entropy decay suggesting more stable updates and less premature policy collapse. Results for Llama3.2-3B-Instruct on mathematical reasoning are reported in Appendix D.2, Figure 9. The gains also extend to Countdown (+4.6 on 3–4 input numbers and +3.2 on 5 input numbers) with Qwen2.5-Math-1.5B (Figure 4), and to scientific coding (+7.0 on average and +15.5 on LiveCodeBench Code Gen.v6) with Qwen3.5-9B (Table 4).

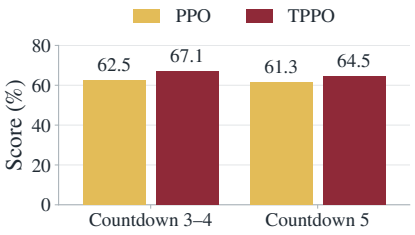

Figure 4: **Countdown task on Qwen2.5-Math-1.5B.** Trained on problems with three or four input numbers.

RQ2: TPPO better tracks value evolution during training. Figure 5 compares critic-predicted and MC-estimated local value evolution between adjacent anchors. We evaluate under two SNR thresholds ($\tau = 0, 1$) to check whether the comparison is robust to filtering out noisier estimates

Table 3: **In-domain and out-of-domain results with $n = 1$.** All entries are accuracies (%). Orange parenthetical values show the absolute improvement over PPO in percentage points. Bold and underlined values indicate the best and second-best results within each model group, respectively.

Method	In-Domain Performance						Out-of-Domain Performance			
	AIME24/25/26	AMC	MATH500	Minerva	Olympiad	Avg.	ARC-C	AGI-Eval	BBH	OOD Avg.
Qwen3-8B-Base										
Base	8.2/6.8/6.4	33.8	54.3	24.9	27.2	23.1	62.4	44.2	27.8	44.8
PPO	11.7/15.0/15.0	56.9	77.9	45.9	39.2	37.4	91.4	67.1	42.0	66.8
SPPO	15.8/10.8/15.0	56.3	78.2	43.4	41.3	37.3	91.8	66.0	40.4	66.1
TPPO	**23.3/20.0/20.8**	**68.1**	**84.9**	**46.3**	**50.9**	**44.9** (+7.5)	**93.0**	**71.8**	**46.8**	**70.5**
Qwen3-4B										
Base	19.2/24.2/17.5	57.5	79.4	47.2	43.5	41.2	91.3	74.0	46.6	70.6
PPO	44.2/35.0/34.2	79.4	85.5	47.2	52.7	54.0	91.3	78.7	50.6	73.5
SPPO	40.8/32.5/35.0	85.0	87.1	50.5	56.9	55.4	**91.6**	79.4	**52.6**	**74.5**
TPPO	**60.8/47.5/44.2**	**90.6**	**92.3**	**51.0**	**62.4**	**64.1** (+10.1)	90.3	**79.9**	51.1	73.8

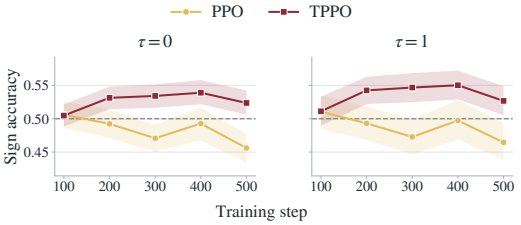

Figure 5: **Value evolution tracking.** Sign agreement between critic predictions and MC-estimated value changes for SNR $> \tau$. Results use Qwen3-8B-Base mathematical reasoning with $n = 8$.

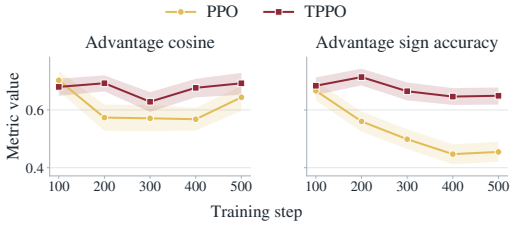

Figure 6: **Credit assignment.** Cosine similarity and sign agreement with GAE advantages computed from MC values. Results use Qwen3-8B-Base mathematical reasoning with $n = 8$.

of local value evolution (Appendix D.3). Starting from similar accuracy, TPPO maintains stronger directional agreement as training progresses, while PPO's agreement declines. Across the last four checkpoints, the gains are 3.9–6.8 percentage points for $\tau = 0$ and 4.9–7.4 for $\tau = 1$, with the corresponding 95% bootstrap intervals for the gains above zero. These results show improved tracking of the direction of local value evolution under both SNR thresholds. Appendix Figure 10 additionally compares both critics on the same actor's rollouts to control for rollout source.

RQ3: TPPO improves the quality of credit assignment. Figure 6 evaluates GAE advantages ($\lambda = 1$) against a reference computed by substituting MC estimates for critic values (details are in Appendix D.3). After the initial checkpoint, TPPO consistently achieves higher cosine similarity and sign agreement with this reference. Across the last four checkpoints, advantage sign accuracy improves by 15.3 - 19.9 percentage points, with 95% bootstrap intervals for these gains above zero. Together with RQ2, these results show that improved tracking value evolution accompanies advantage estimates that agree more closely with the MC reference.

Table 4: **Coding results on Qwen3.5-9B.** Accuracy (%) averaged over three evaluation runs. Orange parenthetical values show the improvement over PPO in percentage points. HumanEvalFix averages Python, Rust, and JavaScript.

Benchmark	Base	PPO	TPPO
SciCode	29.3	32.6	**35.0** (+2.4)
HumanEval	94.7	93.5	**96.5** (+3.0)
HumanEvalFix	58.3	70.1	**75.5** (+5.4)
LCB Code Gen.v6	61.9	57.6	**73.1** (+15.5)
APPS Introductory	69.1	72.3	**80.8** (+8.5)

6 Discussion

Ablations To test whether the gains come from the loss alone, we train a standard PPO critic with BCE (PPO + BCE), applying a sigmoid to each position's scalar output without accumulating outputs across positions. We also replace BCE with MSE in TPPO (TPPO + MSE), keeping its cumulative

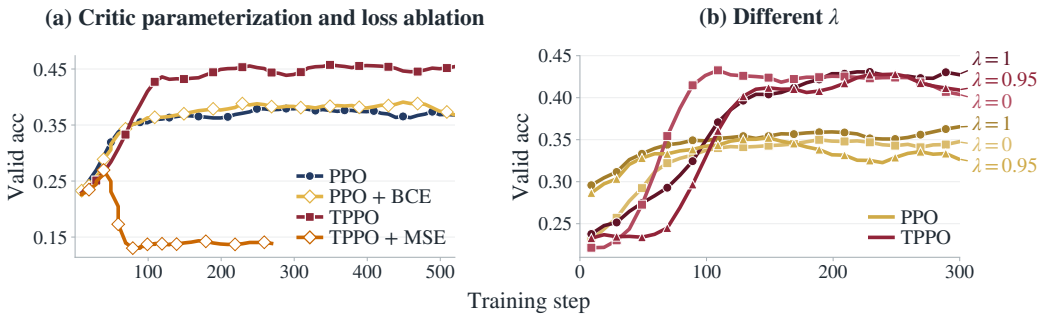

Figure 8: **Critic ablations and actor GAE coefficients.** Validation accuracy for (a) critic parameterization and loss variants, and (b) PPO and TPPO with $\lambda_{\text{actor}} \in \{1, 0.95, 0\}$. Settings for (b) are given in Appendix E.2.

parameterization. As shown in Figure 8 (a), PPO + BCE improves over PPO, but TPPO reaches a higher validation accuracy and maintains the gap. The TPPO + MSE variant deteriorates early and remains well below the other variants. The gains therefore mainly come from modeling value evolution with the cumulative parameterization, with BCE providing an additional benefit.

Modeling value evolution improves value estimates. Following the usual interpretation of a critic as an estimate of the expected terminal return from a prefix (Kazemnejad et al., 2024), we compare the reconstructed values with Monte Carlo (MC) estimates. Figure 7 shows this comparison on AIME test responses. TPPO has lower mean absolute error and higher value accuracy at every evaluated position. Its error grows towards the end of a response but remains below PPO's, while its accuracy stays higher throughout. This supports the view that learning value evolution improves the critic's value estimates, which in turn provide the baseline and local changes used by GAE.

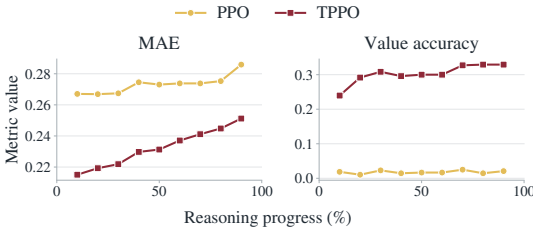

Figure 7: **Value estimates on test data.** Error and accuracy against MC estimates at different reasoning positions.

Modeling value evolution improves credit assignment for different GAE λ. We vary the actor's GAE coefficient λ, which controls how far advantage estimates propagate information from later TD residuals, to test whether learning value evolution remains useful beyond the setting used in our main experiments. In this experiment, we vary only the λ used to compute the actor's GAE advantages, keeping the critic's λ fixed at 1. Under terminal rewards and $\gamma = 1$, $\lambda = 0$ uses the local value change at nonterminal transitions, while $0 < \lambda < 1$ geometrically weights subsequent value changes and the terminal TD residual. At $\lambda = 1$, these terms telescope to the terminal reward minus the current value (Appendix B.2). Better modeling of value evolution can therefore improve credit assignment across these regimes. Our $\lambda = 0$ experiment applies this construction at the segment level. Figure 8(b) shows that TPPO retains its performance advantage over PPO in every evaluated setting; the exact training settings are given in Appendix E.2.

7 Conclusion

We presented TPPO, a cumulative parameterization for modeling value evolution in language-model reinforcement learning. Our analysis of successive prefix values reveals a Bayesian decomposition with additive updates in log-odds space. TPPO uses this structure to accumulate position-wise scalar outputs and learns these outputs through terminal-outcome supervision on the reconstructed values, which are then used by PPO. Experiments on mathematical reasoning tasks show consistent improvements over PPO across two Qwen3 model sizes, as well as gains on additional tasks and a different model family. The critic analyses show stronger agreement with Monte Carlo values and better alignment with MC-based reference advantages, while the ablations show that the cumulative parameterization and BCE training are both important. These results suggest that modeling how value evolves along a reasoning trace can improve credit assignment without additional rollout sampling or process-level labels.

AI USE STATEMENT

AI-based writing tools were used to assist with language editing, proofreading, and improving the clarity of the manuscript. The authors were responsible for the scientific ideas, methodology, experiments, analysis, and conclusions, and reviewed and approved the final text.

ETHICS STATEMENT

This work studies value-function learning for language-model reinforcement learning using existing models and datasets for mathematical reasoning, Countdown, and coding. Improving reasoning and code-generation capabilities may support beneficial applications but can also enable misuse. Our evaluation focuses on task performance and credit assignment; it does not assess harmful generation, bias, or the security of generated code. Benchmark improvements therefore do not establish suitability for deployment, which requires application-specific safety evaluation and appropriate safeguards.

REPRODUCIBILITY STATEMENT

The critic parameterization, training objective, and policy update are specified in Section 4.2. Models, datasets, baselines, and evaluation settings are described in Section 5.1, with training hyperparameters in Appendix D.1. The MC evaluation protocol, anchor selection, diagnostic metrics, and bootstrap procedure are detailed in Appendices B.3 and D.3. The theoretical assumptions and derivations are given in Section 4.1 and Appendices B and C; settings for the output-head and actor-GAE ablations are provided in Appendices E.1 and E.2. Code is included in the supplementary materials.

REFERENCES

Chenxin An, Zhihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and Lingpeng Kong. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models, 2025. URL https://hkunlp.github.io/blog/2025/Polaris.

Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International conference on machine learning*, pp. 449–458. Pmlr, 2017.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Zizhe Chen, Jiqian Dong, Yizhou Tian, Garry YANG, Yongqiang Chen, Zhitang Chen, and James Cheng. Hista and numca: Estimate state value effectively for large language model reinforcement learning. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=ff4b0ELdU0.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

Jesse Farebrother, Jordi Orbay, Quan Vuong, Adrien Ali Taïga, Yevgen Chebotar, Ted Xiao, Alex Irpan, Sergey Levine, Pablo Samuel Castro, Aleksandra Faust, et al. Stop regressing: Training value functions via classification for scalable deep rl. *arXiv preprint arXiv:2403.03950*, 2024.

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

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

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. Measuring coding challenge competence with apps. In J. Vanschoren and S. Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021a. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/c24cd76e1ce41366a4bbe8a49b02a028-Paper-round2.pdf.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021b.

Zhenyu Hou, Ziniu Hu, Yujiang Li, Rui Lu, Jie Tang, and Yuxiao Dong. Treerl: Llm reinforcement learning with on-policy tree search, 2025. *URL https://arxiv. org/abs/2506.11902*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Naman Jain, Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 58791–58831, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/94074dd5a072d28ff75a76dabed43767-Paper-Conference.pdf.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in rl training of llms. *arXiv preprint arXiv:2410.01679*, 2024.

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

Chenxi Li, Wenxuan Zeng, Yun Luo, Fangchen Yu, Peng Ye, Yu Cheng, and Jun Zhang. Sciwalker: Synthesizing scientific coding problems with operator graphs and execution feedback, 2026. URL https://arxiv.org/abs/2609.30054.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9, 2024.

Yizhi Li, Qingshui Gu, Zhoufutu Wen, Ziniu Li, Tianshun Xing, Shuyue Guo, Tianyu Zheng, Xin Zhou, Xingwei Qu, Wangchunshu Zhou, et al. Treepo: Bridging the gap of policy optimization and efficacy and inference efficiency with heuristic tree-based modeling. *arXiv preprint arXiv:2508.17445*, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Niklas Muennighoff, Qian Liu, Armel Zebaze, Qinkai Zheng, Binyuan Hui, Terry Yue Zhuo, Swayam Singh, Xiangru Tang, Leandro Von Werra, and Shayne Longpre. Octopack: Instruction tuning code large language models. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 7604–7623, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/1ec299a5229034141e58aeded0d0b9de-Paper-Conference.pdf.

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

Yuxiao Qu, Matthew YR Yang, Amrith Setlur, Lewis Tunstall, Edward Emanuel Beeching, Ruslan Salakhutdinov, and Aviral Kumar. Optimizing test-time compute via meta reinforcement fine-tuning. *arXiv preprint arXiv:2503.07572*, 2025.

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

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

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for llm reasoning. In *International Conference on Learning Representations*, volume 2025, pp. 60808–60838, 2025.

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

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

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023.

Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, Shengyan Liu, Di Luo, Yutao Ma, Hao Tong, Kha Trinh, Chenyu Tian, Zihan Wang, Bohao Wu, Yanyu Xiong, Shengzhu Yin, Minhui Zhu, Kilian Lieret, Yanxin Lu, Genglin Liu, Yufeng Du, Tianhua Tao, Ofir Press, Jamie Callan, Eliu Huerta, and Hao Peng. Scicode: A research coding benchmark curated by scientists. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 30624–30650. Curran Associates, Inc., 2024. doi: 10.52202/079017-0963. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/36850592258c8c41cecdaa3dea5ff7de-Paper-Datasets_and_Benchmarks_Track.pdf.

Luong Trung, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. Reft: Reasoning with reinforced fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7601–7614, 2024.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, 2024.

Tianyi Wang, Yixia Li, Long Li, Yibiao Chen, Shaohan Huang, Yun Chen, Peng Li, Yang Liu, and Guanhua Chen. Sppo: Sequence-level ppo for long-horizon reasoning tasks. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7831–7853, 2026.

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

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244, 2026.

Yufeng Yuan, Yu Yue, Ruofei Zhu, Tiantian Fan, and Lin Yan. What's behind ppo's collapse in long-cot? value optimization holds the secret. *arXiv preprint arXiv:2503.01491*, 2025.

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 10495–10516, 2025.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. Agieval: A human-centric benchmark for evaluating foundation models. In *Findings of the association for computational linguistics: NAACL 2024*, pp. 2299–2314, 2024.

Zhijian Zhou, Long Li, Xuan Zhang, Zongkai Liu, Yulei Qin, Ke Li, Xing Sun, Xiaoyu Tan, Chao Qu, and Yuan Qi. Start classifying: Categorical critics for llm reinforcement learning. *arXiv preprint arXiv:2608.02181*, 2026.

Zilin Zhu, Chengxing Xie, Xin Lv, and slime Contributors. slime: An llm post-training framework for rl scaling. https://github.com/THUDM/slime, 2025. GitHub repository. Corresponding author: Xin Lv.

A LIMITATIONS

Our experiments cover models up to 9B parameters, leaving the effectiveness of TPPO at substantially larger scales unverified. Although we evaluate mathematical reasoning, Countdown, and coding, broader task coverage is needed to assess its generality beyond these verifiable domains. Our critic evaluation also reflects a trade-off under computational constraints: we sample MC continuations at selected anchors on a subset of prompts, balancing trajectory coverage against the accuracy of each value estimate. These diagnostics capture value changes between anchors but can miss fluctuations within each interval, and finite MC samples introduce estimation noise. Finally, the current formulation assumes binary terminal rewards and $\gamma = 1$; extending it to dense or non-binary rewards requires further investigation.

B VALUE FUNCTIONS AND CREDIT ASSIGNMENT

B.1 WHAT DOES THE VALUE FUNCTION LEARN?

Consider the population value-regression objective under the trajectory distribution induced by policy π:

$$\mathcal{L}(V) = \mathbb{E}_{\tau\sim\pi}\left[\big(V(\mathbf{s}_t) - G_t\big)^2\right]. \tag{12}$$

Conditioned on a particular state $\mathbf{s}_t = \mathbf{s}$, its contribution is

$$\mathcal{L}_{\mathbf{s}}(V) = \mathbb{E}_{\pi}\left[\big(V(\mathbf{s}) - G_t\big)^2 \mid \mathbf{s}_t = \mathbf{s}\right]. \tag{13}$$

Differentiating with respect to $V(\mathbf{s})$ gives

$$\frac{\partial \mathcal{L}_{\mathbf{s}}}{\partial V(\mathbf{s})} = 2\left(V(\mathbf{s}) - \mathbb{E}_{\pi}[G_t \mid \mathbf{s}_t = \mathbf{s}]\right). \tag{14}$$

Therefore, the population minimizer is

$$V^*(\mathbf{s}_t) = \mathbb{E}_{\pi}[G_t \mid \mathbf{s}_t] = V^{\pi}(\mathbf{s}_t). \tag{15}$$

For a terminal-only reward, $r_k = 0$ for $k < T$ and $r_T = R(\mathbf{y})$, so

$$G_t = \gamma^{T-t} R(\mathbf{y}), \tag{16}$$

and hence

$$V^{\pi}(\mathbf{s}_t) = \mathbb{E}_{\mathbf{y}_{\geq t}\sim\pi(\cdot|\mathbf{s}_t)}\left[\gamma^{T-t} R(\mathbf{y})\right]. \tag{17}$$

When $\gamma = 1$ and $R(\mathbf{y}) \in \{0, 1\}$, this reduces to

$$V^{\pi}(\mathbf{s}_t) = \mathbb{E}_{\pi}[R(\mathbf{y}) \mid \mathbf{s}_t] \tag{18}$$

$$= \Pr_{\pi}(R(\mathbf{y}) = 1 \mid \mathbf{s}_t). \tag{19}$$

Thus, the critic estimates the success probability of continuations produced by π. In PPO, the relevant policy is typically the rollout policy π_{old}, so the value target changes as the policy is updated.

B.2 VALUE CHANGES AND PPO CREDIT ASSIGNMENT

We connect changes in prefix values to token-level advantages and to the GAE signals used by PPO. For autoregressive generation, sampling token a_t deterministically produces the next state $\mathbf{s}_{t+1} = (\mathbf{s}_t, a_t)$. For nonterminal steps ($t < T$), zero intermediate rewards and $\gamma = 1$ give

$$Q^{\pi}(\mathbf{s}_t, a_t) = \mathbb{E}_{\pi}\left[G_t \mid \mathbf{s}_t, a_t\right] = \mathbb{E}_{\pi}\left[G_{t+1} \mid \mathbf{s}_{t+1}\right] = V^{\pi}(\mathbf{s}_{t+1}), \tag{20}$$

so

$$A^{\pi}(\mathbf{s}_t, a_t) = V^{\pi}(\mathbf{s}_{t+1}) - V^{\pi}(\mathbf{s}_t) \doteq \Delta V_t^{\pi}. \tag{21}$$

Thus, the true token-level advantage is the change in the probability of eventual success induced by the token.

Let λ_{actor} denote the GAE coefficient used for the policy update, and let $\widehat{V}$ be a fixed critic, with $\widehat{V}_t = \widehat{V}(\mathbf{s}_t)$. With the value after termination set to zero,

$$\widehat{A}_t^{(\lambda_{\text{actor}})} = \sum_{\ell=0}^{T-t} \lambda_{\text{actor}}^{\ell} \widehat{\delta}_{t+\ell}, \qquad \widehat{\delta}_k = r_k + \widehat{V}_{k+1} - \widehat{V}_k. \tag{22}$$

For terminal reward R, this can be written directly in terms of local value changes:

$$\begin{aligned} \widehat{A}_t^{(\lambda_{\text{actor}})} = & \sum_{\ell=0}^{T-t-1} \lambda_{\text{actor}}^{\ell} \big(\widehat{V}_{t+\ell+1} - \widehat{V}_{t+\ell}\big) \\ & + \lambda_{\text{actor}}^{T-t} \big(R - \widehat{V}_T\big). \end{aligned} \tag{23}$$

How $\lambda < 1$ affects credit assignment. At $\lambda_{\text{actor}} = 0$, the nonterminal signal is the current local change,

$$\widehat{A}_t^{\text{GAE}}(0) = \widehat{V}(\mathbf{s}_{t+1}) - \widehat{V}(\mathbf{s}_t) = \Delta\widehat{V}_t, \tag{24}$$

whereas at $\lambda_{\text{actor}} = 1$ the terms telescope to $R - \widehat{V}(\mathbf{s}_t)$. More generally, intermediate values of λ_{actor} geometrically weight local changes at increasing distances. At the terminal step, all settings give $\widehat{A}_T = R - \widehat{V}_T$, since the value after termination is zero. With exact values, the nonterminal one-step estimator equals the true advantage in Equation 21.

For $0 < \lambda_{\text{actor}} < 1$, a value change ℓ steps ahead contributes to the current token's advantage with weight $\lambda_{\text{actor}}^{\ell}$. Smaller λ_{actor} therefore places more emphasis on nearby changes and attenuates the distant terminal signal. Since advantages weight token gradients in Equation 2, errors in these nearby changes can distort the direction or strength of the policy update. Accurately tracking value evolution can improve these credit signals, especially when the terminal outcome is many steps away.

This also changes the relative credit between adjacent tokens. For $t < T$, the GAE recursion gives

$$\widehat{A}_t^{(\lambda_{\text{actor}})} - \widehat{A}_{t+1}^{(\lambda_{\text{actor}})} = \Delta\widehat{V}_t - (1 - \lambda_{\text{actor}})\widehat{A}_{t+1}^{(\lambda_{\text{actor}})}. \tag{25}$$

Thus, for $\lambda_{\text{actor}} < 1$, adjacent-token credit depends on both the local value change and the downstream advantage; the local change alone determines their difference only at $\lambda_{\text{actor}} = 1$.

The $\lambda = 1$ case. For completeness, when $\gamma = \lambda = 1$ as in the main experiments,

$$\widehat{A}_t = R(\mathbf{y}) - \widehat{V}_t, \tag{26}$$

and, if $\widehat{V} = V^{\pi}$, then for $t < T$

$$\mathbb{E}_{\pi}\left[\widehat{A}_t \mid \mathbf{s}_t, a_t\right] = V^{\pi}(\mathbf{s}_{t+1}) - V^{\pi}(\mathbf{s}_t) = A^{\pi}(\mathbf{s}_t, a_t). \tag{27}$$

The sampled $\lambda = 1$ estimator therefore recovers the token-level value change in expectation. For adjacent positions in the same response, the shared terminal return also gives the sample-wise ordering identity

$$\widehat{A}_t - \widehat{A}_{t+1} = \widehat{V}_{t+1} - \widehat{V}_t = \Delta\widehat{V}_t. \tag{28}$$

B.3 Monte Carlo Evaluation

To visualize how values evolve over the course of reasoning, we evaluate states at a fixed set of relative positions along each response. For a response containing T tokens, let

$$0 < \rho_1 < \rho_2 < \cdots < \rho_M \leq 1$$

denote predefined fractions of the response length. The corresponding token positions are

$$t_j = \lfloor \rho_j T \rfloor, \qquad j = 1, \ldots, M. \tag{29}$$

Thus, responses of different lengths are evaluated at comparable stages of generation.

For two consecutive evaluation points, we define the coarse-grained value change as

$$\Delta_j^{(\rho)} V \doteq V(\mathbf{s}_{t_{j+1}}) - V(\mathbf{s}_{t_j}). \tag{30}$$

Although Equation 30 measures the change over a segment of the response rather than between two adjacent tokens, it is directly related to the token-level value changes discussed in Section 3. By telescoping,

$$\Delta_j^{(\rho)} V = \sum_{t=t_j}^{t_{j+1}-1} \left[V(\mathbf{s}_{t+1}) - V(\mathbf{s}_t) \right]. \tag{31}$$

Hence, the change between two evaluation points summarizes the cumulative token-level value evolution over the corresponding segment of reasoning.

Monte Carlo value estimates. For each selected state $\mathbf{s}_{t_j}$, we estimate its policy value by sampling K_j valid continuations from the rollout policy at anchor j:

$$\widehat{V}_{\mathrm{MC}}(\mathbf{s}_{t_j}) \doteq \frac{1}{K_j} \sum_{k=1}^{K_j} R\Big(\mathbf{y}^{(k)}\Big), \qquad \mathbf{y}^{(k)} \sim \pi(\cdot \mid \mathbf{s}_{t_j}). \tag{32}$$

Because

$$V^{\pi}(\mathbf{s}_{t_j}) = \mathbb{E}_{\pi}\left[R(\mathbf{y}) \mid S_{t_j} = \mathbf{s}_{t_j} \right], \tag{33}$$

$\widehat{V}_{\mathrm{MC}}(\mathbf{s}_{t_j})$ is a Monte Carlo estimate of the same state value targeted by the PPO critic.

We then compute the Monte Carlo value change between consecutive evaluation points as

$$\Delta_j^{(\rho)} V_{\mathrm{MC}} \doteq \widehat{V}_{\mathrm{MC}}(\mathbf{s}_{t_{j+1}}) - \widehat{V}_{\mathrm{MC}}(\mathbf{s}_{t_j}), \tag{34}$$

and the corresponding PPO-predicted change as

$$\Delta_j^{(\rho)} V_{\mathrm{PPO}} \doteq V_\phi(\mathbf{s}_{t_{j+1}}) - V_\phi(\mathbf{s}_{t_j}). \tag{35}$$

Figure 1(a) illustrates these estimates for a correct and an incorrect response.

Connection to GAE advantage ordering. Under terminal-only rewards and $\gamma = \lambda = 1$, let $\widehat{A}_t^{\mathrm{GAE}}[V] = R(\mathbf{y}) - V(\mathbf{s}_t)$ denote the advantage computed using a value function V, with zero terminal-state value. Applying Equation 28 across the interval between two anchors gives

$$\widehat{A}_{t_j}^{\mathrm{GAE}}[V] - \widehat{A}_{t_{j+1}}^{\mathrm{GAE}}[V] = V(\mathbf{s}_{t_{j+1}}) - V(\mathbf{s}_{t_j}) = \Delta_j^{(\rho)} V. \tag{36}$$

The endpoint identity also holds when MC estimates replace the critic values at the two anchors. Thus, when the critic and MC changes are both nonzero and have opposite signs at the same anchor pair for the same rollout policy, the critic-based advantages reverse the ordering given by the MC-based reference. This reference uses sampled estimates rather than exact policy values. The comparison concerns relative advantages at the anchors; it does not establish the sign of every individual token-level value change within the interval.

C Bayesian Value Evolution and TPPO Details

C.1 Proof of the Bayesian Prefix-Value Update

Because the next state is the deterministic concatenation $\mathbf{s}_{t+1} = (\mathbf{s}_t, a_t)$,

$$V^{\pi_\theta}(\mathbf{s}_{t+1}) = \Pr_{\pi_\theta}\left[R = 1 \mid \mathbf{s}_t, a_t \right] \tag{37}$$

$$= \frac{\Pr_{\pi_\theta}\left[R = 1, a_t \mid \mathbf{s}_t \right]}{\Pr_{\pi_\theta}\left[a_t \mid \mathbf{s}_t \right]}, \tag{38}$$

where the denominator is positive by assumption. If $0 < V^{\pi_\theta}(\mathbf{s}_t) < 1$, the numerator and denominator factor as

$$\Pr_{\pi_\theta}\left[R = 1, a_t \mid \mathbf{s}_t \right] = V^{\pi_\theta}(\mathbf{s}_t) \Pr_{\pi_\theta}\left[a_t \mid \mathbf{s}_t, R = 1 \right], \tag{39}$$

$$\Pr_{\pi_\theta}\left[a_t \mid \mathbf{s}_t \right] = V^{\pi_\theta}(\mathbf{s}_t) \Pr_{\pi_\theta}\left[a_t \mid \mathbf{s}_t, R = 1 \right] + \big(1 - V^{\pi_\theta}(\mathbf{s}_t)\big) \Pr_{\pi_\theta}\left[a_t \mid \mathbf{s}_t, R = 0 \right], \tag{40}$$

which proves Equation 7. If the current value is zero, every positive-probability action has zero joint success probability; the value-one case follows analogously from zero joint failure probability.

Under the positivity assumptions in Proposition 4.1, let $\Lambda_t = \Pr_{\pi_\theta}[a_t \mid \mathbf{s}_t, R=1] / \Pr_{\pi_\theta}[a_t \mid \mathbf{s}_t, R=0]$. Dividing the posterior success probability by its complement gives

$$\frac{V^{\pi_\theta}(\mathbf{s}_{t+1})}{1 - V^{\pi_\theta}(\mathbf{s}_{t+1})} = \frac{V^{\pi_\theta}(\mathbf{s}_t)}{1 - V^{\pi_\theta}(\mathbf{s}_t)} \Lambda_t. \tag{41}$$

Taking logarithms proves Equation 8; iterating this identity gives the cumulative decomposition in Corollary 4.2.

C.2 Learning Signals in the Cumulative Parameterization

TPPO learns position-wise scalar outputs through supervision on the reconstructed values. Here we derive the suffix-aggregated learning signal for each logit increment. This derivation explains how terminal outcomes train the cumulative parameterization; it does not establish recovery of the true Bayesian log-likelihood ratios.

Learning signal for each increment. For a response of length T, let

$$\widehat{z}_t = \widehat{z}_1 + \sum_{j=2}^{t} \widehat{\Delta}_j, \qquad \widehat{V}_t = \sigma(\widehat{z}_t). \tag{42}$$

Under a terminal binary outcome $R \in \{0, 1\}$, the BCE objective is

$$\mathcal{L}_V = \frac{1}{T} \sum_{t=1}^{T} \ell(\widehat{z}_t, R), \tag{43}$$

where

$$\ell(\widehat{z}_t, R) = -R \log \sigma(\widehat{z}_t) - (1 - R) \log(1 - \sigma(\widehat{z}_t)). \tag{44}$$

For $i \geq 2$, the increment $\widehat{\Delta}_i$ affects every $\widehat{z}_t$ with $t \geq i$, and

$$\frac{\partial \widehat{z}_t}{\partial \widehat{\Delta}_i} = \mathbb{1}\{t \geq i\}. \tag{45}$$

Therefore, by the chain rule,

$$\begin{aligned} \frac{\partial \mathcal{L}_V}{\partial \widehat{\Delta}_i} &= \frac{1}{T} \sum_{t=1}^{T} \frac{\partial \ell(\widehat{z}_t, R)}{\partial \widehat{z}_t} \frac{\partial \widehat{z}_t}{\partial \widehat{\Delta}_i} \\ &= \frac{1}{T} \sum_{t=i}^{T} \frac{\partial \ell(\widehat{z}_t, R)}{\partial \widehat{z}_t}. \end{aligned} \tag{46}$$

For BCE with logits,

$$\frac{\partial \ell(\widehat{z}_t, R)}{\partial \widehat{z}_t} = \sigma(\widehat{z}_t) - R = \widehat{V}_t - R, \tag{47}$$

which gives

$$\frac{\partial \mathcal{L}_V}{\partial \widehat{\Delta}_i} = \frac{1}{T} \sum_{t=i}^{T} \left(\widehat{V}_t - R \right). \tag{48}$$

Hence, each increment receives the aggregate value-learning signal from the suffix of reconstructed values that depends on it. No direct target for $\widehat{\Delta}_i$ is required. Although all positions in a response share the same terminal outcome R, different increments receive different gradients because they affect different suffixes of the value sequence.

Indeed, adjacent increments satisfy

$$\frac{\partial \mathcal{L}_V}{\partial \widehat{\Delta}_i} - \frac{\partial \mathcal{L}_V}{\partial \widehat{\Delta}_{i+1}} = \frac{1}{T} \left(\widehat{V}_i - R \right). \tag{49}$$

Thus, the cumulative parameterization converts terminal value supervision into position-dependent, suffix-aggregated learning signals for the token-wise increments.

D Experimental Details and Additional Results

D.1 Training Setup

Training data and rewards. Mathematical reasoning uses Polaris with binary answer-verification rewards. Scientific coding uses Qwen3.5-9B on a multi-turn dataset of 8,178 main problems and 37,820 subproblems, with binary code-execution rewards and a 120-second training timeout. Thinking is enabled and the training seed is 42.

Table 5: **Training, rollout, and optimization settings.** Shared settings apply to both domains unless a domain-specific value is shown.

Parameter	Math	Scientific coding
Maximum response length	8,192	20,000
Temperature	1.0	1.0
Top-p	0.95	1.0
Batch size	64 prompts (Base) / 128 prompts (4B)	32 main problems $\times$ 8 trajectories
Minibatch size	32 prompts (Base) / 64 prompts (4B)	256 responses
Actor learning rate	10^{-6}	10^{-6}
Critic learning rate	4×10^{-6}	10^{-5}
Learning-rate schedule	Constant	Constant
Weight decay	0.1	0.1
Gradient clipping	1.0	1.0
KL / entropy coefficient	0 / 0	0 / 0
Importance-weight upper threshold	2.0	2.0
Training framework	Slime / Megatron	VERL[2] / Megatron
GPUs per node	8	8

Baselines. PPO uses token-level values, GAE with $\gamma = \lambda = 1$, clipped MSE, and clips the actor's policy probability ratio $\rho_t(\theta) = \pi_\theta(a_t \mid \mathbf{s}_t)/\pi_{\text{old}}(a_t \mid \mathbf{s}_t)$ in its surrogate objective to $[1-0.2, 1+0.2]$. VAPO uses the Clip-Higher variant, which clips this same policy ratio asymmetrically with $\epsilon_{\text{low}} = 0.2$ and $\epsilon_{\text{high}} = 0.28$, together with adaptive actor $\lambda(L)$ and an NLL term on correct responses. GRPO standardizes rewards within response groups, while SPPO broadcasts a prompt-level success baseline to all response tokens.

D.2 Additional Results

In Tables 2 and 3, Avg. averages the seven in-domain benchmarks equally, counting each AIME year separately; OOD Avg. averages the three OOD benchmarks equally.

Figure 9 reports the additional Llama3.2-3B-Instruct comparison on mathematical reasoning, complementing the model comparisons in Section 5.2. The overall scores are 11.60 for the base model, 16.30 for PPO, 19.03 for GRPO, and 19.88 for TPPO; thus, TPPO improves over PPO by 3.58 points on this model.

Controlling for rollout source. Figure 10 compares PPO and TPPO critics on the same actor rollouts. This additional result controls for the fact that the main comparison evaluates each critic on responses generated by its own actor.

[2]VAPO uses the VERL framework; see https://arxiv.org/pdf/2504.05118.

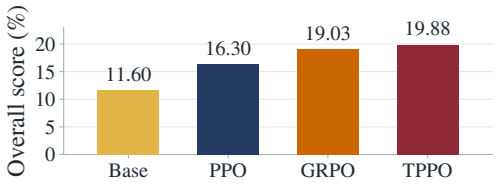

Figure 9: **Additional results on Llama3.2-3B-Instruct.** Overall scores for the base model, PPO, GRPO, and TPPO (%).

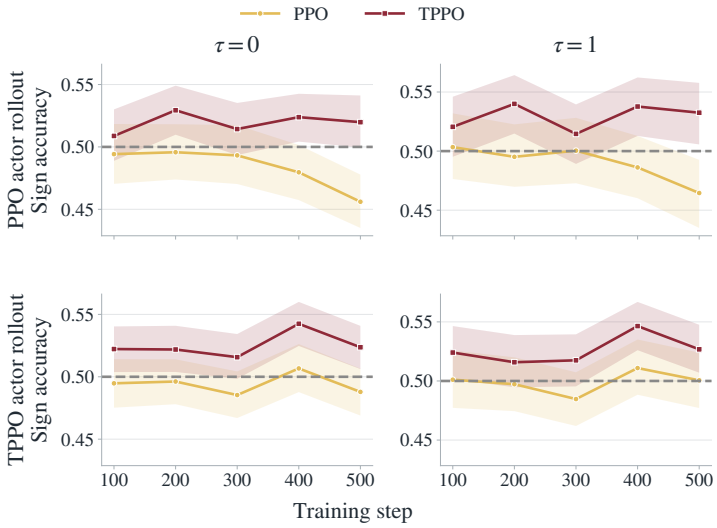

Figure 10: **Value-change sign accuracy by rollout source.** Rows distinguish PPO-actor and TPPO-actor rollouts; columns show the SNR thresholds $\tau = 0$ and $\tau = 1$.

D.3 Diagnostic Metrics

Value-estimation metrics on test responses. In Figure 7, value accuracy is the fraction of prefixes satisfying $|\widehat{V} - \widehat{V}_{\mathrm{MC}}| \leq 0.05$ at each evaluated reasoning position.

Sampling protocol. The diagnostics use Qwen3-8B-Base actor updates at steps 100, 200, 300, 400, and 500. At each step, the 64 prompts are a fixed subset of the training set, and exactly the same subset is used for PPO and TPPO at all five steps. Each method's own actor generates eight responses per prompt, giving 512 responses per method and checkpoint. Each response has nine MC anchors at approximately 10%, 20%, ..., 90% of its length. This yields 4,608 anchors and 4,096 adjacent-anchor intervals per method and checkpoint. Figure 1(a) shows individual examples; the aggregate diagnostics use this sampling protocol.

Each anchor is allocated a budget of 128 MC continuations under the corresponding actor. Across both methods and all five checkpoints, 46,080 anchors have 128 valid samples. The total number of valid MC samples is 5.90M. MC means and standard errors use the actual valid sample count at each anchor.

These training-set diagnostics measure value-change sign agreement and agreement with MC-based reference advantages, as defined below. Figure 10 additionally controls for rollout source when evaluating value-change sign agreement.

SNR-conditioned value-change sign accuracy. Let $\widehat{V}_{\mathrm{MC},j}$ be the MC value estimate at anchor j, and let s_j and n_j denote the standard deviation of its sampled continuation returns and the number of

MC samples, respectively. For adjacent anchors, define

$$\Delta\mathrm{MC}_j = \widehat{V}_{\mathrm{MC},j+1} - \widehat{V}_{\mathrm{MC},j},$$
$$\mathrm{SNR}_j = \frac{|\Delta\mathrm{MC}_j|}{\sqrt{s_j^2/n_j + s_{j+1}^2/n_{j+1}}}. \tag{50}$$

The denominator estimates the standard error of the MC difference; the calculation omits the covariance between the two anchor estimates. For a critic-predicted change ΔV_j, we compute the empirical conditional sign agreement

$$\mathrm{SignAcc}(\tau) = \Pr[\mathrm{sign}(\Delta V_j) = \mathrm{sign}(\Delta\mathrm{MC}_j) \,|\, \mathrm{SNR}_j > \tau]\,. \tag{51}$$

The threshold $\tau = 0$ retains nonzero MC changes, while $\tau = 1$ retains changes whose magnitude exceeds one estimated standard error. The selection depends solely on the MC samples and does not use critic predictions. Figure 5 reports each method's absolute sign accuracy under these two thresholds. Filtering is performed separately for each method, and each accuracy is normalized by its own retained interval count, rather than by all 4,096 candidates. Table 6 lists these denominators.

Table 6: **Intervals retained by SNR filtering.** Counts per method and checkpoint, with percentages of the 4,096 candidate intervals in parentheses.

	$\tau = 0$		$\tau = 1$	
Actor update	PPO	TPPO	PPO	TPPO
100	2,765 (67.5%)	2,843 (69.4%)	1,655 (40.4%)	1,972 (48.1%)
200	2,403 (58.7%)	2,968 (72.5%)	1,586 (38.7%)	1,930 (47.1%)
300	2,186 (53.4%)	2,793 (68.2%)	1,468 (35.8%)	1,860 (45.4%)
400	1,944 (47.5%)	2,755 (67.3%)	1,323 (32.3%)	1,852 (45.2%)
500	1,991 (48.6%)	2,767 (67.6%)	1,339 (32.7%)	1,830 (44.7%)

MC-based reference advantages. For these diagnostics, $\gamma = \lambda = 1$, intermediate rewards are zero, there is no KL reward shaping, and the value after termination is zero. GAE therefore reduces to the terminal outcome minus the value at the anchor:

$$\widehat{A}_{t_j} = R - \widehat{V}_{t_j}, \qquad \widehat{A}^{\mathrm{MC}}_{t_j} = R - \widehat{V}_{\mathrm{MC}}(\mathbf{s}_{t_j}). \tag{52}$$

Here R is the binary terminal reward of the original response. The reference replaces the critic value with the MC estimate at the same anchor. Both metrics use all 4,608 anchors per method and checkpoint. For cosine similarity, we concatenate the critic advantages and reference advantages into two vectors and compute their overall cosine similarity. Sign accuracy is the fraction of anchors at which their signs agree; zero values are included with $\mathrm{sign}(0) = 0$. Figure 6 reports each method's absolute metrics, while Table 7 reports their differences.

Bootstrap estimation of metric gains. To quantify uncertainty in the gains over PPO, we draw 2,000 bootstrap resamples of the 512 response indices with replacement, using random seed 1234. Each sampled response contributes all its anchors and intervals, preserving their within-response dependence. The same resampled `sample_index` sequence is used for both methods; each method retains its own generated responses, MC estimates, and SNR-filtered intervals. We recompute the aggregate metrics in each resample and take the 2.5th and 97.5th percentiles of the TPPO-minus-PPO differences. These intervals describe response-resampling uncertainty conditional on the fixed prompts and trained checkpoints; the resampling unit is a response, with eight responses per prompt.

Across actor update steps 200–500, the 95% intervals for advantage sign-accuracy gains and value-change sign-accuracy gains under both SNR thresholds are above zero. These are intervals for metric differences, distinct from the individual-method curves in Figures 5 and 6.

E Additional Ablations and Discussion

E.1 Shared and Separate Output Heads

Our default critic uses the shared scalar head described in the experimental setup (Section 5.1): the first output supplies the initial logit and subsequent outputs supply logit increments.

Table 7: **Diagnostic gains of TPPO over PPO.** Sign-accuracy gains are in percentage points; cosine gains are absolute differences. Rows correspond to actor update steps.

	Advantage		Value-change sign accuracy	
Actor update	Cosine	Sign accuracy	$\tau = 0$	$\tau = 1$
100	-0.0228	$+1.76$	-0.12	$+0.06$
200	$+0.1189$	$+15.34$	$+3.90$	$+4.94$
300	$+0.0574$	$+16.60$	$+6.35$	$+7.40$
400	$+0.1089$	$+19.92$	$+4.62$	$+5.29$
500	$+0.0485$	$+19.44$	$+6.76$	$+6.23$

Separate-head parameterization. We also tested a variant with two independent scalar heads,

$$\widehat{z}_1 = f_{\phi_0}(\mathbf{h}_1), \qquad \widehat{\Delta}_t = f_{\phi_\Delta}(\mathbf{h}_t), \quad t \geq 2. \tag{53}$$

This variant uses separate output parameters for the initial logit and subsequent increments. The cumulative reconstruction and terminal-outcome supervision remain the same.

Across runs, the separate-head variant was less stable and learned different allocations between the initial logit and accumulated increments. Table 8 summarizes the output statistics: the initial logits have substantially larger magnitude and variance, while the increment scales remain comparable. Figure 11 compares validation accuracy, response length, and actor gradient norms for the head variants and PPO.

These observations motivate using a shared head in all main experiments. Sharing ties the initial-logit and increment outputs through the same parameters, without additional losses or regularization terms. The Bayesian identity itself does not prescribe whether the heads should be shared.

Table 8: **Output statistics for shared and separate heads.** The separate head has substantially larger initial-logit magnitude and variance, while increment statistics remain comparable.

	Initial logit		Increment	
Head	Mean	SD	Mean	SD
Shared	-0.0273	0.0052	-0.00054	0.00210
Separate	-0.8657	0.3016	-0.00132	0.00257

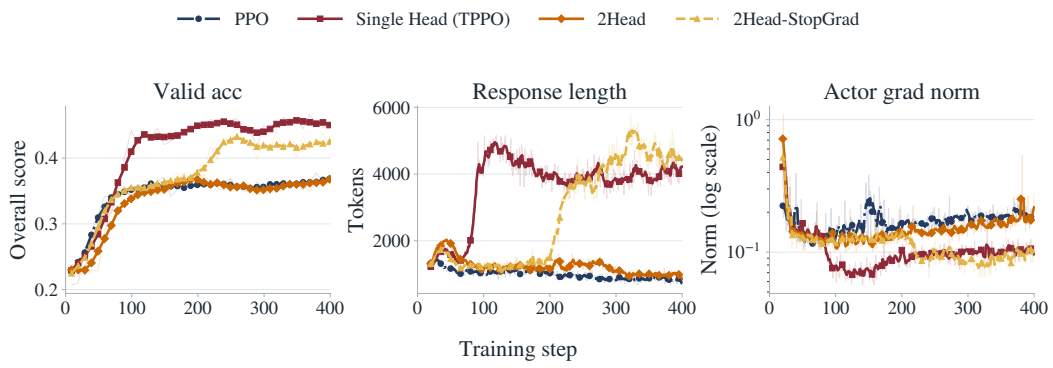

Figure 11: **Shared and separate output heads.** Validation accuracy, response length, and actor gradient norm for PPO, the shared-head TPPO critic, and the two separate-head variants shown in the legend.

E.2 Actor GAE Coefficients

For the comparison in Section 6 and Figure 8(b), we evaluate $\lambda_{\text{actor}} \in \{1, 0.95, 0\}$ while keeping the critic's λ fixed at 1. The critic is warmed up for 200 steps with the actor held fixed before joint actor–critic training begins. The $\lambda_{\text{actor}} = 0$ condition computes advantages at the segment level:

with zero intermediate rewards and $\gamma = 1$, each nonterminal segment TD residual is the predicted value difference between its boundaries. Following the segment-level credit assignment used in VinePPO (Kazemnejad et al., 2024), we assign the same segment advantage to every token within that segment. The other settings use the corresponding GAE construction in Appendix B.2.