BEYOND FINAL OUTCOMES: PROCESS-AWARE RELATIVE DISTRIBUTION MATCHING FOR LLM REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Distribution matching has emerged as a promising paradigm for LLM reasoning by shaping probability distributions over complete trajectories to preserve diverse rewarding paths and thus improve reasoning performance. However, existing approaches remain largely trajectory-level, leaving two questions unresolved when incorporating intermediate reasoning information: where along a long reasoning trajectory should process information be measured, and how should it influence the distribution-matching objective? We first examine the former through counterfactual continuations. Building on high-entropy reasoning points as candidate decision boundaries, we find that alternative continuations from these forks exhibit greater variation in downstream verified-answer progress than those from neighboring non-fork positions, supporting their use as process-sensitive boundaries. We further observe substantial heterogeneity across segments: improving and deteriorating transitions coexist within both correct and incorrect trajectories, while a small subset of segments accounts for a disproportionate fraction of progress variation. These observations motivate distinct uses of intermediate progress in distribution matching. Progress magnitude controls credit allocation, reallocating optimization emphasis across segments while preserving the distribution-matching residual, whereas progress direction shapes the target distribution by adaptively tempering the reference-policy prior. Based on this design, we propose **P**rocess-**A**ware Relative **D**istribution **M**atching (PADM). Across trajectories sampled for the same prompt, a group-relative objective aligns relative policy probabilities with the resulting process-aware target. Theoretically, we show that exact relative matching recovers the intended target distribution, while magnitude-based segment weighting preserves this matching condition. Empirically, we evaluate PADM on six challenging mathematical reasoning benchmarks across two model scales, demonstrating consistent improvements over competitive baselines, with gains of up to 4.63 points in average accuracy.

1 INTRODUCTION

Reinforcement Learning with Verifiable Rewards (RLVR) has emerged as an effective paradigm for improving the reasoning capabilities of large language models, particularly in domains such as mathematics and coding, where the final outcome can be automatically verified (Cobbe et al., 2021; Shao et al., 2024; Guo et al., 2025; Yu et al., 2026). Within this setting, distribution matching provides a promising perspective on policy optimization in LLM reasoning by specifying a desired distribution over complete reasoning trajectories and training the policy to allocate probability mass accordingly (Hu et al., 2024; Yu et al., 2025; Yoon et al., 2025; Zhu et al., 2026; Kwon et al., 2026; Liu et al., 2026; Chen et al., 2026a). By encouraging coverage of multiple rewarding trajectories, this line of work aims to preserve diverse reasoning trajectories and support generalization, addressing the risk of localizes probability mass on a narrow set of solutions. However, specifying which complete reasoning trajectories should be preferred does not by itself determine how the resulting distribution-matching correction should be allocated across their intermediate decisions. This becomes particularly pronounced in long-horizon reasoning, where a single response may contain thousands of tokens, yet learning-relevant decisions may be driven by only a small number of reasoning segments.

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.

Reasoning segments with different reasoning behaviors, i.e., routine derivations, critical decisions, incorrect detours, and subsequent self-corrections, may all receive the same trajectory-level learning signal despite playing substantially different roles in the final outcome. Nevertheless, existing distribution-matching formulations either operate primarily at the level of complete trajectories or incorporate intermediate supervision via an additional process reward model. This gap raises two distinct questions for process-aware distribution matching: where should intermediate reasoning information be localized, and how should it influence the matching objective?

The first question concerns where to localize process information. Building on high-entropy reasoning positions as candidate decision boundaries, our Observation I examines whether these positions are particularly sensitive to alternative reasoning choices. We find that alternative continuations from high-entropy positions produce greater average variation in downstream confidence in the verified answer than those from nearby lower-entropy positions. This local sensitivity supports the use of high-entropy positions as boundaries for measuring intermediate progress. Under suitable coverage assumptions, outcome supervision need not be statistically harder than process supervision, suggesting that empirical gaps may reflect algorithmic limitations (Jia et al., 2025). This finding suggests that practical gains can arise from how algorithms extract and use learning signals from trajectory-level feedback. Together, this observation motivates extracting process information from decision-sensitive reasoning transitions rather than treating all intermediate positions uniformly.

The second question concerns how localized process information should influence distribution matching. Observation II reveals two complementary forms of heterogeneity under decision-aware partitioning. First, progress varies in direction: correct trajectories may contain locally deteriorating transitions, while incorrect trajectories may contain substantial improvements. Second, progress varies strongly in magnitude, with a handful of segments accounting for most of the overall reasoning evolution. These phenomena suggest that intermediate progress should not be collapsed into a single process-level reward or uniformly propagated signal. Crucially, these two aspects of progress have different semantics in distribution matching. Progress magnitude controls optimization emphasis without changing the target distribution, while progress direction guides local target shaping. This motivates separating target-preserving optimization shaping from process-aware target shaping.

Building on this localize-then-separate view, we propose **P**rocess-**A**ware Relative **D**istribution **M**atching (PADM), a process-aware distribution-matching framework that operates at both the within-trajectory and cross-trajectory levels without requiring additional process-level labels or a separately trained process reward model. Within each trajectory, PADM localizes intermediate progress between high-entropy reasoning positions and assigns distinct roles to its magnitude and direction. Progress magnitude controls target-preserving optimization through a residual-preserving gradient surrogate, reallocating learning emphasis across segments without changing the forward-matching residual. Progress direction shapes the target by adaptively tempering reference-policy anchoring. Across trajectories sampled for the same prompt, PADM minimizes the dispersion of policy-to-target log-density mismatches, enforcing group-relative consistency so that relative policy probabilities agree with a process-aware target jointly determined by terminal outcomes and intermediate reasoning progress. In summary, our main contributions are:

- We show that high-entropy reasoning positions exhibit greater counterfactual variation in downstream verified-answer progress than neighboring lower-entropy positions, supporting their use as process-sensitive boundaries.
- We distinguish two roles of intermediate progress in trajectory distribution matching: magnitude performs target-preserving optimization shaping, while direction performs process-aware target shaping, separating how strongly to learn from what distribution to learn.
- We instantiate these principles in PADM through residual-preserving segment credit, adaptive reference tempering, and group-relative trajectory matching. We further show that exact relative matching recovers the intended process-aware target, while segment credit preserves the matching condition.
- We conduct comprehensive experiments on six mathematical reasoning benchmarks across two model scales, demonstrating consistent improvements in average accuracy compared with various related baselines, with gains of up to 3.92 points over GRPO, 3.17 points over FlowRL, and 4.63 points over GRPO-VPS.

2 PRELIMINARIES

RLVR. Let $x \sim \mathcal{X}$ denote a prompt sampled from prompt distribution $\mathcal{X}$ and $\tau = (a_1, \ldots, a_T)$ a response sampled from an autoregressive policy π_θ, with $\pi_\theta(\tau \mid x) = \prod_{t=1}^{T} \pi_\theta(a_t \mid x, a_{<t})$. A verifier provides a terminal reward $r(x, \tau)$ based on the correctness of the final answer. We further denote a fixed reference model by π_{ref}, which serves as a prior constraint on the reward distribution. For each prompt x, we sample a group of G independent trajectories $\{\tau_i\}_{i=1}^{G} \sim \pi_\theta(\cdot \mid x)$.

Trajectory-level distribution matching. Distribution-matching approaches provide a view of policy optimization by specifying a reward-induced distribution over complete trajectories. Following FlowRL (Zhu et al., 2026), they consider an unnormalized trajectory target of the form:

$$\pi^*(\tau \mid x) \propto \exp\{\zeta \hat{r}(x, \tau)\}\, \pi_{\text{ref}}(\tau \mid x) =: \rho_x(\tau),$$

where $\hat{r}_i$ denotes the normalized terminal reward among the sampled group trajectories. The corresponding desired policy is obtained by normalizing ρ_x over trajectories, i.e., $\pi^*(\tau \mid x) = \rho_x(\tau) / \sum_{\tau' \in \mathcal{T}_x} \rho_x(\tau')$. Zhu et al. (2026) optimizes this distribution-matching objective using a trajectory-level squared surrogate derived from reverse-KL minimization. For later use, we denote their trajectory-wise policy-to-target log-density difference by $q_x(\tau) := \log \pi_\theta(\tau \mid x) - \log \rho_x(\tau)$. At exact distribution matching, $q_x(\tau)$ is constant over trajectories for the same prompt x.

3 METHODS

In this section, we present the PADM objective. We begin with motivating observations (Section 3.1) and introduce the overall objective (Section 3.2). We then explain how progress magnitude and direction guide credit allocation and target shaping, respectively (Section 3.3), before deriving the group-relative matching formulation for trajectories sampled from the same prompt (Section 3.4).

3.1 DECISION-AWARE PROCESS LOCALIZATION AND PROGRESS HETEROGENEITY

Before designing a process-aware matching objective, we first ask where process information should be resolved and what structure this information exhibits once localized. Specifically, we investigate two complementary questions: (i) *Are some positions in a reasoning trajectory more sensitive to the choice of the next step?* (ii) *How are local improving and deteriorating steps distributed along a reasoning trajectory, and do a few steps account for most of the reasoning evolution?* These questions arise naturally because trajectory-level reward summarizes an entire reasoning process, inevitably compressing a heterogeneous reasoning process into a single final outcome. For example, two responses may reach the same correct answer through very different paths: one makes steady progress, while the other takes an unproductive detour before self-correcting. Conversely, an incorrect response may contain primarily useful intermediate steps before a later mistake leads to failure. In each case, final correctness offers an incomplete account of how the reasoning unfolds: success does not imply uninterrupted progress, and failure does not imply that every preceding decision was unproductive. To begin with, we define the trajectory partitioning and progress measures.

Given a fixed prompt $x \in \mathcal{X}$ and a group of independently sampled trajectories $\{\tau_1, \tau_2, \ldots, \tau_G\} \sim \pi_\theta(\cdot \mid x)$. Inspired by Wang et al. (2026c), we partition the reasoning path within each trajectory τ_i into internal reasoning segments using the top 20% highest-entropy tokens as decision boundaries, referred to as reasoning forks. The partition is denoted by $\mathbf{z}_i = (z_{i1}, \ldots, z_{iM_i})$. We subsequently enforce a minimum separation between selected reasoning forks, since neighboring forks may belong to the same local decision region. Then, we use changes in log-probability of the ground-truth answer to characterize these transitions (Wang et al., 2026b). Specifically, we define progress of segment z_{ik} as:

$$\Delta C_{ik} = \frac{1}{|y^*|} \big(\log \pi_\theta\left(y^* \mid x, z_{i,\le k}\right) - \log \pi_\theta\left(y^* \mid x, z_{i,\le k-1}\right) \big),$$

where y^* and $|y^*|$ are the true answer and its length. Positive and negative progress correspond to improving and deteriorating support for the true answer. We interpret progress as a local proxy for reasoning behavior evolution under policy π_θ between high-entropy decision boundaries along the reasoning trajectory. Prior work has identified high-entropy tokens as candidate reasoning forks or

related progress gains to outcome correctness. Our goal is not to re-establish entropy as an important signal, but to examine whether these positions provide informative boundaries for measuring downstream reasoning progress. We examine, from a unique perspective, the heterogeneity within individual trajectories, providing insights into the coexistence of confidence gains and declines and the concentration of substantial confidence changes in a few reasoning segments.

Observation I: Reasoning forks exhibit elevated downstream sensitivity. We examine whether high-entropy reasoning forks are particularly informative process boundaries by asking whether alternative continuations from these positions produce greater variation in downstream progress than continuations from neighboring non-fork positions. For each trajectory, we randomly select one fork and a non-fork neighbor, then sample eight continuations from each under the same generation budget. Dispersion is measured as the mean absolute pairwise difference in progress. Higher dispersion indicates that alternative continuations lead to more varied changes in support for the true answer. Figure 1 shows higher mean dispersion at forks, with a larger estimated increase for outcome-correct trajectories than for outcome-incorrect ones generated under Qwen3-4B (Yang et al., 2025). The increase is 0.29, with 0.89 for outcome-correct trajectories and 0.15 for outcome-incorrect trajectories. This comparison provides direct empirical support for an association between local policy uncertainty and counterfactual downstream sensitivity, motivating high-entropy forks as process boundaries at which intermediate progress can be measured.

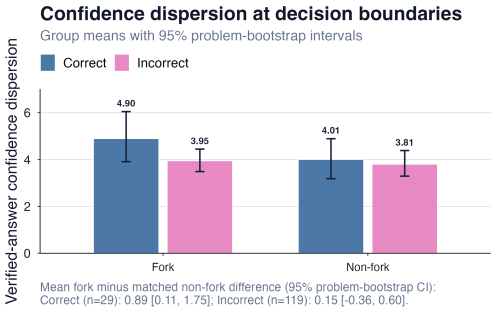

Figure 1: **Progress dispersion at forks and neighbor non-fork positions.** Across 148 valid matched pairs, bars show averages grouped by outcome correctness; Error bars indicate 95% bootstrap confidence intervals obtained by resampling the problems.

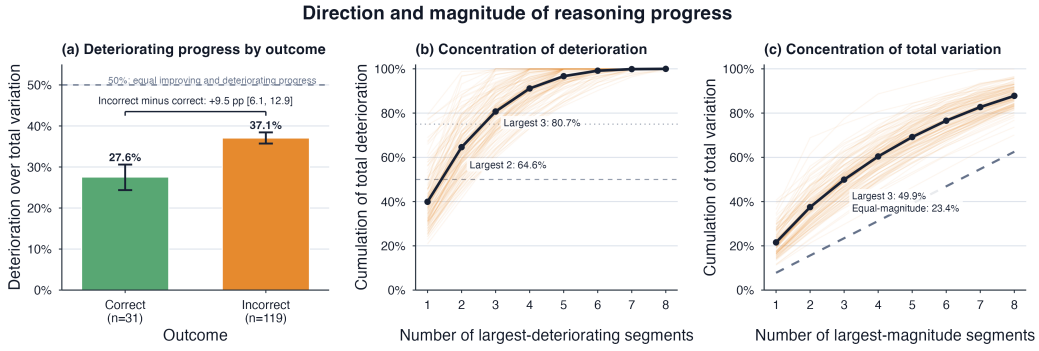

Figure 2: **Direction and magnitude of reasoning progress.** (a) Mean fraction of deteriorating transitions over total absolute progress. Error bars indicate 95% bootstrap confidence intervals. (b–c) For outcome-incorrect trajectories, cumulative fractions of total deterioration and total absolute progress captured by the largest k segment contributions, respectively. Thin curves represent individual trajectories; the dashed line in (c) shows the equal-magnitude baseline.

Observation II: Progress varies in direction and magnitude. We analyze how progress is distributed along realized trajectories generated by Qwen3-4B. Figure 2(a) shows that deteriorating transitions account for a larger proportion of total absolute progress in outcome-incorrect trajectories than in outcome-correct ones (37.1% versus 27.6%). Nevertheless, the remaining 62.9% of outcome-incorrect trajectories arise from improved transitions, whereas outcome-correct trajectories also exhibit deterioration. Thus, an incorrect answer can coexist with substantial increases in verified-answer progress along the way, and a correct answer can be reached through a trajectory with local detours. Final correctness alone does not describe the progress made at each intermediate transition. Furthermore, these changes are unevenly distributed. Among outcome-incorrect

trajectories, the three largest deteriorating segments account for 80.7% of total deterioration (Figure 2(b)). Considering both improving and deteriorating transitions, the three largest-magnitude segments account for 49.9% of total absolute progress variation, compared with 23.4% under an equal-magnitude allocation across segments (Figure 2(c)). A small subset of transitions is thereby responsible for a substantial portion of the observed changes in progress. This study reveals two forms of heterogeneity: direction, as outcome-correct trajectories may contain deteriorating transitions while outcome-incorrect ones may contain productive steps; and magnitude, as a small subset of segments accounts for much of the progress variation.

Together, Observations I and II motivate a two-stage view: localize decision-sensitive boundaries, then separate the roles of the process information measured between them. We next formulate a process-aware relative distribution-matching objective that integrates terminal outcome supervision with complementary signals of intermediate progress, using the former to shape the trajectory target and the latter to allocate optimization credit across segments.

3.2 Process-Aware Relative Distribution Matching

Building on the preceding decision-aware localization and role separation, we now present the PADM objective. We first introduce the overall formulation and introduce its components in the following subsections: (i) the progress decomposition mechanism in Section 3.3, and (ii) the process-aware target distribution and derived group-relative matching formulation in Section 3.4. For a fixed prompt $x \in \mathcal{X}$ and a group of independently sampled trajectories from the frozen rollout policy $\{\tau_1, \tau_2, \ldots, \tau_G\} \sim \pi_{\theta_{\text{old}}}(\cdot \mid x)$, it optimizes the policy via the following:

$$\mathcal{L}_{\text{PADM}}(x;\theta) = \frac{1}{G}\sum_{i=1}^{G} \text{sg}\left[\text{clip}\left(\frac{\pi_\theta(\tau_i \mid x)}{\pi_{\theta_{\text{old}}}(\tau_i \mid x)}, 1-\epsilon, 1+\epsilon\right)\right]\left(\tilde{q}_x(\tau_i;\theta) - \bar{q}_x\right)^2, \tag{3.1}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operator. The quantity $\tilde{q}_x(\tau_i;\theta)$ is a residual-preserving surrogate for the policy-to-target log-density mismatch associated with trajectory τ_i, which we refer to as the log-density mismatch for simplicity. Further, $\bar{q}_x$ denotes the group mean of the log-density mismatch within prompt x. Consequently, the centered residual $\tilde{q}_x(\tau_i;\theta) - \bar{q}_x$ measures how much this mismatch deviates from those of alternative reasoning trajectories generated for the same prompt. Minimizing Equation 3.1 therefore encourages all within-prompt trajectories to share a common mismatch, so that their relative probabilities agree with the relative masses prescribed by our intended process-aware target. This relative consistency is sufficient to recover the corresponding normalized target distribution, as stated in Theorem 3.3. Specifically, same as the observation study in Section 3.1, we partition each trajectory into internal reasoning segments $\mathbf{z}_i = (z_{i1}, \ldots, z_{iM_i})$ and its final answer y_i, i.e., $\tau_i = (\mathbf{z}_i, y_i)$. For simplicity, we use the last segment z_{i,M_i+1} to denote y_i. The log-density mismatch for τ_i is defined as:

$$\tilde{q}_x(\tau_i;\theta) = -\zeta\hat{r}_i + \sum_{k=1}^{M_i+1}\left[\tilde{\ell}_{ik}(\theta) + (1-\gamma_{ik})\log \pi_{\text{ref}}(z_{ik} \mid z_{i,<k}, x)\right], \quad \forall i \in 1, \ldots, G. \tag{3.2}$$

At a high level, the summation across reasoning segments incorporates process-aware distribution matching via the surrogate segment ratio $\tilde{\ell}_{ik}$ and the process-aware power term γ_{ik}. We next delve further into the within-trajectory process-aware level design and explain each of the components.

3.3 Within-Trajectory Progress Decomposition

Observation II in Section 3.1 does not merely suggest decomposing a scalar progress signal; it indicates that intermediate progress should play distinct roles in distribution matching. We therefore use progress magnitude for credit allocation, controlling the relative optimization emphasis across reasoning segments while preserving the matching residual, and progress direction for target shaping, adaptively modifying the reference preferences that define the target distribution. This decomposition incorporates intermediate progress as local process evidence for distribution matching, without treating it as an additional reward or segment-level advantage.

Progress Magnitude for Credit Allocation Progress magnitude reflects how consequential a transition is for optimization, rather than how desirable that transition should be under the target distribution. Accordingly, magnitude should control the strength of the update without redefining the

distribution that PADM aims to match. We define the weight w_{ik} for segment z_{ik} as:

$$w_{ik} := \alpha|\Delta C_{ik}|\Big(1 + \frac{1}{M_i - 1}\sum_{j=k+1}^{M_i} \mathbb{I}\left[\text{sign}(\Delta C_{ik})\,(C_{ij} - C_{i,k-1}) > 0\right]\Big) \tag{3.3}$$

with confidence $C_{ij} = \pi_\theta\left(y^* \mid x, z_{i,\le j}\right)$. The first term, absolute progress, controls nonnegative progress segment weights, conceptually similar to prioritizing transitions using absolute temporal-difference errors (Schaul et al., 2016). The second term, downstream consistency, supplements local magnitude by measuring a proxy for the effect of the transition that persists beyond the local boundary instead of being immediately reversed, especially in early steps. This connects to trajectory consistency in self-rewarding LLMs (Zhang et al., 2026) and early-exit training (Dai et al., 2026). These approaches construct intrinsic rewards or favor earlier correct answers; PADM instead uses persistence to allocate nonnegative segment-level credit within distribution matching. As a result, w_{ik} controls the overall strength of process-guided credit allocation and emphasizes reasoning transitions that induce a significant local change and whose effects are reflected in subsequent reasoning. Next, we show that our magnitude-based credit is governed by the total variation of the confidence along the trajectory and remains bounded within a constant factor of this overall signal.

Proposition 3.1 (Total-Variation Credit)**.** *Consider a confidence sequence $C_0, C_1, \ldots, C_M$ with raw progress $\Delta C_k = C_k - C_{k-1}$, and let the raw process weight be w_k as in Equation 3.3. Define the unnormalized raw total variation $\text{TV}(C) := \sum_{k=1}^{M} |\Delta C_k|$, then the total process credit satisfies $\alpha\ \text{TV}(C) \le \sum_{k=1}^{M} w_k \le 2\alpha\ \text{TV}(C)$ with $\text{TV}(C) = |C_M - C_0| + 2\min\{P, N\}$, where $P = \sum_{k=1}^{M}[\Delta C_k]_+, N = \sum_{k=1}^{M}[-\Delta C_k]_+$.*

Basically, total variation contains more information than the net change at the endpoints: the additional term $2\min\{P, N\}$ captures progress that is canceled by transitions in the opposite direction. Thus, a trajectory containing a significant detour followed by self-correction can receive meaningful process-level credit even when its initial and final confidence are similar. Our weighting preserves consequential intermediate dynamics that would be invisible under endpoint-only aggregation, while the persistence factor selectively strengthens transitions whose effects remain downstream. However, a direct multiplication would alter the underlying distribution matching, which is undesirable since w_{ik} is intended to control the optimization emphasis. Therefore, to preserve the numerical value of the trajectory-level mismatch, we apply the weight w_{ik} only during the backward pass, using the stop-gradient operator $\text{sg}[\cdot]$, which carries no gradient. We formulate the residual-preserving stop-gradient surrogate $\tilde{\ell}_{ik}$ of ratio $\ell_{ik} = \log \pi_\theta(z_{ik} \mid z_{i,<k}, x) - \log \pi_{\text{ref}}(z_{ik} \mid z_{i,<k}, x)$ as:

$$\tilde{\ell}_{ik}(\theta) = \text{sg}(\ell_{ik}(\theta)) + \text{sg}(w_{ik})\left[\ell_{ik}(\theta) - \text{sg}(\ell_{ik}(\theta))\right].$$

With the $\text{sg}[\cdot]$ operator, we allow w_{ik} to change the optimization path while keeping the numerical distribution-matching residual unchanged. Mathematically, $\nabla_\theta \mathcal{L}(\tau_i; \theta) \propto \sum_k w_{ik} \nabla_\theta \ell_{ik}$. Thus, magnitude changes how the policy approaches the target, rather than which target it approaches.

Then, given that the reference policy provides a local anchor for controlling policy deviation (Chen et al., 2026b), from which the policy continuously deviates during training, we set it as the default behavior. When the progress signal suggests a productive transition, we allow the target to be tempered for greater deviation from the reference policy to unlock creative potential for capturing this useful reasoning pattern. When the progress signal indicates an unproductive transition, stronger reference anchoring discourages the learned policy from amplifying this local deviation, causing it to resort to the reference policy's relative mode rankings. Next, we present more details.

Progress Direction for Distribution Shaping We design the power γ_{ik} for segment z_{ik} as a local inverse temperature applied to reference policy π_{ref}:

$$d_{ik} = \text{sign}(\Delta C_{ik}), \qquad \gamma_{ik} = 1 - \beta d_{ik}, \quad 0 < \beta < 1. \tag{3.4}$$

Answer y_i inherits the weight and power of the last reasoning segment. In general, $\gamma_{ik} > 1$ amplifies the reference policy's preference for likely continuations, while $0 < \gamma_{ik} < 1$ allows more deviation from the reference policy. We present a rigorous explanation in the following proposition.

Proposition 3.2 (Adaptive Distribution Shaping)**.** *Let d_k, γ_k be Equations 3.4, 3.3. For a reference distribution p with full support, we define $p_\gamma(z) = p(z)^\gamma / \sum_{z'} p(z')^\gamma$. For every $\gamma > 0$,*

$$\frac{\partial}{\partial \gamma} H(p_\gamma) = -\gamma\ \text{Var}_{z \sim p_\gamma}[\log p(z)] \le 0.$$

Hence, if $\Delta C_k > 0$, $H(p_{\gamma_k}) \geq H(p)$. *If* $\Delta C_k < 0$, $H(p_{\gamma_k}) \leq H(p)$.

This proposition characterizes the isolated effect of γ_k on the local reference factor. Positive progress relaxes the local reference preference by increasing target entropy, thereby allowing more diversity and creativity by shifting probability mass away from reference-preferred continuations (Chen et al., 2026a). Negative progress has the opposite effect, sharpening the local target and thereby imposing a more conservative reference anchor. Together with Proposition 3.1, this establishes two distinct roles for the progress signal. Using both the magnitude and direction of progress, we can characterize our desired process-aware target distribution. In the next subsection, we present our main group-relative distribution-matching objective, which aims to attain this desired distribution.

3.4 Cross-Trajectory Relative Consistency

Having constructed the within-trajectory decomposition design, we now introduce the process-aware target distribution that our policy aims to match and then derive the group-relative objective introduced earlier in Section 3.2, which induces within-prompt cross-trajectory relative consistency. For a fixed prompt x, we define the desired unnormalized process-aware target density:

$$\pi^*(\tau \mid x) \propto \exp\{\zeta \hat{r}(x,\tau)\} \prod_{k=1}^{M(\tau)+1} \pi_{\mathrm{ref}}(z_k \mid z_{<k}, x)^{\gamma_k} =: \rho_x(\tau), \tag{3.5}$$

and the normalized target distribution $\pi^*(\tau \mid x) = \rho_x(\tau)/Z_\rho(x)$ with $Z_\rho(x) := \sum_{\tau' \in \mathcal{T}_x} \rho_x(\tau')$, and $\mathcal{T}_x$ denotes the trajectory space conditioned on x. Here, the progress-dependent power γ_k is computed using the rollout policy $\pi_{\theta_{\mathrm{old}}}$ and treated as fixed during the corresponding policy update. The target $\rho_x(\tau)$ combines two sources of supervision: (1) the terminal reward $\hat{r}(x,\tau)$ determines the overall desirability of the trajectory; (2) the progress-dependent powers $\{\gamma_k\}_{k=1}^M$ reshape the reference-policy preference according to Equation 3.4. Then, we define distributional mismatch through the trajectory-level log-density error without relying on the normalization constant $Z_\rho(x)$:

$$q_x(\tau;\theta) := \log \pi_\theta(\tau \mid x) - \log \rho_x(\tau) = -\log Z_\rho(x), \qquad \forall \tau \in \mathcal{T}_x, \tag{3.6}$$

where the last equality holds when the policy matches the target distribution in Equation 3.5. Furthermore, we observe that $q_x(\tau;\theta)$ is independent of the specific trajectory and depends only on the conditioning prompt x in such a case. Hence, matching $\pi_\theta(\cdot \mid x)$ to the process-aware target is equivalent to requiring that $q_x(\tau;\theta)$ be constant across trajectories associated with the same prompt x. In particular, for any two trajectories $\tau_i, \tau_j \in \mathcal{T}_x$, relative consistency requires:

$$\log \frac{\pi_\theta(\tau_i \mid x)}{\pi_\theta(\tau_j \mid x)} = \log \frac{\rho_x(\tau_i)}{\rho_x(\tau_j)}, \qquad i \neq j.$$

Therefore, it is sufficient for the learned policy to reproduce the relative probability mass prescribed by the process-aware target. In other words, if one reasoning trajectory receives a larger target mass than another, the policy should preserve the same relative preference between them. This pairwise relative-consistency view is closely related to CTB (Kwon et al., 2026), which avoids unstable partition-function estimation by matching trajectory probability ratios to reward ratios. CTB stabilizes the generation of adversarial prompts in LLM red-teaming with a fixed meta-prompt, whereas PADM addresses RLVR by using our process-aware trajectory distribution for each reasoning problem. Our contribution here is therefore not relative consistency in isolation, but its use within a process-aware framework that explicitly separates segment-level credit allocation from target shaping. In our setting, relative consistency characterizes probability allocation among alternative responses to the same prompt, forming a natural rollout group, with target masses determined jointly by final outcomes and intermediate reasoning progress. For a group $\{\tau_i\}_{i=1}^G$, we abbreviate $q_i := q_x(\tau_i;\theta)$. The pairwise condition derived above requires $\{q_i - q_j = 0\}_{i,j=1}^G$. Thus, all trajectories generated for the same prompt should share a common policy-to-target log mismatch. This suggests directly minimizing the dispersion around their within-prompt mean. In the unweighted case, define $\bar{q} = \frac{1}{G}\sum_{i=1}^G q_i$ gives the exact identity: $\frac{1}{G}\sum_{i=1}^G (q_i - \bar{q})^2 = \frac{1}{2G^2}\sum_{i=1}^G \sum_{j=1}^G (q_i - q_j)^2$. Minimizing the within-group variance is equivalent to minimizing the aggregate pairwise relative-consistency error, encouraging a common policy-to-target mismatch across trajectories for the same prompt. Under i.i.d. sampling with a fixed target, the expected unweighted loss is proportional to the within-prompt variance of this mismatch. Consequently, the objective can be understood from

two equivalent perspectives: (1) pairwise comparison matches the relative policy probability of two trajectories to their relative target mass; (2) group centering minimizes the empirical variance of within-prompt log-density mismatches, driving them toward a common value. Finally, we incorporate the detached clipped importance weights u_i. Together with the residual-persevering surrogate with w_{ik} for optimization dynamics in Equation 3.3 , we obtain our ultimate objective and its gradient:

$$\mathcal{L}_{\mathrm{PADM}}(x;\theta) = \frac{1}{G}\sum_{i=1}^{G} \mathrm{sg}[u_i](\tilde{q}_i - \bar{q})^2,$$

$$\nabla_\theta \mathcal{L}_{\mathrm{PADM}} = \frac{2}{G}\sum_{i=1}^{G} \mathrm{sg}[u_i](\tilde{q}_i - \bar{q}) \sum_{k=1}^{M_i+1} \mathrm{sg}[w_{ik}]\nabla_\theta \ell_{ik},$$

where the weighted group mean $\bar{q} = \sum_{i=1}^{G} \mathrm{sg}[u_i]\tilde{q}_i / \sum_{i=1}^{G} \mathrm{sg}[u_i]$. For the gradient of PADM, $\tilde{q}_i - \bar{q}$ is the cross-trajectory relative correction, w_{ik} is the within-trajectory segment credit, and $\nabla_\theta \ell_{ik}$ is the segment policy update, highlighting two levels of credit assignment in our formulation. Further, we provide theoretical justifications for the relative formulation by showing that the proposed within-prompt relative objective exactly recovers the aimed process-aware target distribution.

Theorem 3.3 (Target Distribution Recovery)**.** *Fix a prompt x and the unnormalized target $\rho_x(\tau)$ as Equation 3.5 with finite normalizer $Z(x) = \sum_{\tau\in\mathcal{T}_x} \rho_x(\tau)$. For $G \geq 2$ i.i.d. trajectories from a full-support policy $\pi_\theta(\cdot \mid x)$ and $u_i > 0$ almost surely, the expected PADM loss $\mathcal{J}_{\mathrm{PADM}}(\theta \mid x) = \mathbb{E}[\mathcal{L}_{\mathrm{PADM}}(\tau_{1:G}; x, \theta)]$ satisfies $\mathcal{J}_{\mathrm{PADM}}(\theta \mid x) = 0$ if and only if $\pi_\theta(\tau \mid x) = \rho_x(\tau)/Z(x), \forall \tau \in \mathcal{T}_x$.*

Theorem 3.3 shows that the relative formulation preserves the desired solution: driving the within-prompt relative residual to zero is equivalent to matching the normalized process-aware target distribution. Importantly, the magnitude-based surrogate does not alter this recovery condition. For a fixed process-aware target, the surrogate preserves the numerical value of the log-density mismatch and leaves the zero-loss relative matching condition unchanged.

4 Experimental Results

Training and Evaluation Setup. For the policy model π_θ, we conduct training on two model scales, using both Qwen3-1.7B-Base and Qwen3-4B-Base (Yang et al., 2025). The reference model π_{ref} is set to the corresponding fixed pretrained model. We use the training set collected from DAPO (Yu et al., 2026). We evaluate all methods on six challenging mathematical reasoning benchmarks: AIME24, AIME25 (MAA, 2025), AMC23 (MAA, 2023), MATH500 (Hendrycks et al., 2021), Minerva Math (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024). We compare our method against three baselines: (i) popular baseline GRPO (Shao et al., 2024); (ii) distribution-matching baseline FlowRL (Zhu et al., 2026); (iii) GRPO-VPS (Wang et al., 2026b), a process-supervised extension of GRPO that uses progress signals to refine credit assignment across reasoning segments. We follow the official veRL recipe for GRPO and the method-specific methods reported in the respective papers for FlowRL and GRPO-VPS. For a controlled comparison, all methods use the same training setting. For more details on training configurations, please refer to the Appendix C.3.

Main Results. The results in Table 1 demonstrate that PADM consistently improves average accuracy over GRPO, FlowRL, and GRPO-VPS across both model scales. PADM achieves the highest average accuracy at both model scales, reaching 30.87% with Qwen3-1.7B-Base and 41.45% with Qwen3-4B-Base. At the 1.7B scale, it surpasses GRPO, FlowRL, and GRPO-VPS by 3.92, 1.65, and 2.04 points, respectively. The corresponding gains at the 4B scale are 1.14, 3.17, and 4.63 points. These improvements extend across both broad mathematical reasoning and competition-style benchmarks. At 1.7B scale, PADM leads five of six benchmarks, including all three competition-style benchmarks, while remaining competitive on OlympiadBench. At 4B scale, it outperforms FlowRL on all six benchmarks, with particularly notable gains on AIME25 and OlympiadBench. It also surpasses GRPO and GRPO-VPS on four benchmarks each, including all three broad mathematical reasoning benchmarks, while maintaining competitive performance on AMC23. The consistently higher average accuracy across both scales, including over GRPO-VPS, which already incorporates confidence-based process supervision. These results support the practical value of incorporating intermediate reasoning progress into distribution matching via decision-aware segmentation and the

distinct use of progress for credit allocation and target shaping, rather than treating process information as a uniform learning signal.

Table 1: Evaluation on mathematical reasoning benchmarks. For *broad math reasoning* benchmarks (MATH500, Minerva Math, and OlympiadBench), we generate a single response per problem, denoted as Avg@1. For harder *competition-style* benchmarks (AIME24, AIME25, and AMC23), we generate 32 responses per problem and average the resulting accuracies, denoted as Avg@32. All methods use $G = 4$, meaning each prompt generates 4 response rollouts as the group size.

Method	**MATH500** Avg@1	**Minerva Math** Avg@1	**OlympiadBench** Avg@1	**AIME24** Avg@32	**AIME25** Avg@32	**AMC23** Avg@32	**Avg**
Qwen3-1.7B-Base	59.00	25.00	22.67	2.19	0.83	16.56	21.04
GRPO	64.20	26.84	27.74	6.04	4.17	36.33	27.55
FlowRL	<u>65.60</u>	<u>29.04</u>	**29.08**	8.33	5.31	37.97	<u>29.22</u>
GRPO-VPS	65.00	25.00	<u>28.78</u>	<u>8.85</u>	<u>5.63</u>	<u>39.69</u>	28.83
PADM	**67.60**	**30.15**	28.64	**10.31**	**6.98**	**41.56**	**30.87**
Qwen3-4B-Base	60.80	30.15	34.42	9.48	8.44	39.53	30.47
GRPO	77.80	35.66	40.50	<u>16.98</u>	13.44	**57.50**	<u>40.31</u>
FlowRL	<u>78.20</u>	<u>36.40</u>	<u>40.65</u>	12.92	6.88	54.61	38.28
GRPO-VPS	68.00	33.09	34.27	**17.29**	**15.62**	52.66	36.82
PADM	**78.60**	**38.42**	**44.21**	16.04	<u>14.06</u>	<u>57.34</u>	**41.45**

Effectiveness of magnitude. On Qwen3-1.7B-Base (Table 2), removing magnitude weighting ($w_{ik} = 1$) lowers average accuracy to 29.05%, with declines on five of six benchmarks. With direction-aware target shaping retained, this comparison supports the complementary role of magnitude weighting in reallocating optimization emphasis across reasoning segments.

Effectiveness of direction. Removing direction-aware reference tempering ($\gamma_{ik} = 1$) lowers average accuracy to 30.02% (Table 2), with declines on MATH500 and all three competition-style benchmarks. Since magnitude weighting remains active, these results support the additional value of using progress direction to shape the target distribution.

Effectiveness of decision-aware segmentation. On Qwen3-1.7B-Base (Table 2), PADM with uniform partitioning and the same minimum gap reduces average accuracy to 25.92%, with declines across all six benchmarks. PADM with random partitioning and the same number of boundaries achieves 29.29% and underperforms on four benchmarks. This consistent degradation supports the benefit of high-entropy segmentation for extracting and using intermediate progress in PADM.

Table 2: Ablation study with Qwen3-1.7B-Base as the base model.

Method	**MATH500** Avg@1	**Minerva Math** Avg@1	**OlympiadBench** Avg@1	**AIME24** Avg@32	**AIME25** Avg@32	**AMC23** Avg@32	**Avg**
PADM	67.60	30.15	28.64	10.31	6.98	41.56	**30.87**
– Magnitude	66.00	22.06	28.34	10.62	6.35	40.94	29.05
– Direction	65.60	30.51	31.16	8.85	4.17	39.84	30.02
w/ Uniform Part.	58.40	22.79	24.04	6.35	3.02	40.94	25.92
w/ Random Part.	65.40	25.37	31.45	6.56	8.02	38.91	29.29

5 Conclusion

We introduced PADM, which measures progress at high-entropy boundaries and uses its magnitude for residual-preserving credit allocation and its direction for target shaping, guided by our two observational studies. Group-relative matching aligns alternative trajectories with the resulting process-aware target. Experiments on six mathematical reasoning benchmarks at two model scales demonstrate consistently higher average accuracy than the evaluated baselines.

AI USE STATEMENT

In this work, we used generative AI tools for [assisting in polishing writing, assisting in the writing of proofs, interpreting results, implementing methods, and brainstorming]. We have not used generative AI tools for [generating synthetic data sets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing methods, assisting with translation, cleaning and reformatting datasets, supporting qualitative and thematic data analysis, supporting qualitative and thematic data analysis, formulate questions for surveys or interviews, creating or modifying scientific figures or images, suggesting experimental parameters, creating artifacts, summarizing or analyzing existing literature, formatting references] that are not applicable to this work. We have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness by 2 authors. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Ruishuo Chen, Yu Chen, Zhuoran Li, and Longbo Huang. Powerflow: Unlocking the dual nature of llms via principled distribution matching. In *International Conference on Machine Learning*, 2026a.

Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *Advances in Neural Information Processing Systems*, 38:57654–57689, 2026b.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Muzhi Dai, Chenxu Yang, and Qingyi Si. S-grpo: Early exit via reinforcement learning in reasoning models. *Advances in Neural Information Processing Systems*, 38:48178–48204, 2026.

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.

Yiran Guo, Lijie Xu, Jie Liu, Dan Ye, and Shuang Qiu. Segment policy optimization: Effective segment-level credit assignment in rl for large language models. *Advances in Neural Information Processing Systems*, 38:114399–114431, 2026.

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

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

Edward Hu, Moksh Jain, Eric Elmoznino, Younesse Kaddar, Guillaume Lajoie, Yoshua Bengio, and Nikolay Malkin. Amortizing intractable inference in large language models. In *International Conference on Learning Representations*, volume 2024, pp. 43053–43083, 2024.

Zeyu Jia, Alexander Rakhlin, and Tengyang Xie. Do we need to verify step by step? rethinking process supervision from a theoretical perspective. In *International Conference on Machine Learning*. PMLR, 2025.

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.

Minchan Kwon, Sunghyun Baek, Minseo Kim, Jaemyung Yu, Dongyoon Han, and Junmo Kim. Stable-gflownet: Toward diverse and robust llm red-teaming via contrastive trajectory balance. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

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

Hunter Lightman, Vineet Kosaraju, 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.

Xiaodong Liu, Michael Xu, Jack W Stokes, Paul Smolensky, Doug Burger, and Jianfeng Gao. Gflowrl: Scaling distribution-matching rl to large language models. *arXiv preprint arXiv:2607.13394*, 2026.

MAA. American mathematics competitions (amc). `https://maa.org/`, 2023.

MAA. American invitational mathematics examination (aime). `https://maa.org/`, 2025.

Chaoli Mou, Zhan Zhuang, Xinning Chen, and Yu Zhang. Beyond uniform credit assignment: Selective eligibility traces for rlvr. *arXiv preprint arXiv:2605.05965*, 2026.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized Experience Replay. In *International Conference on Learning Representations*, 2016. URL `https://mlanthology.org/iclr/2016/schaul2016iclr-prioritized/`.

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.

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

Wei Sun, Wen Yang, Pu Jian, Qianlong Du, Fuwei Cui, Shuo Ren, and Jiajun Zhang. Ktae: A model-free algorithm to key-tokens advantage estimation in mathematical reasoning. *Advances in Neural Information Processing Systems*, 38:161880–161908, 2026.

Haozhe Wang, Qixin Xu, Che Liu, Junhong Wu, Fangzhen Lin, and Wenhu Chen. Emergent hierarchical reasoning in llms through reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 74519–74543, 2026a.

Jingyi Wang, Lei Zhu, Tengjin Weng, Song-Li Wu, Haochen Tan, Jierun Chen, Chaofan Tao, Haoli Bai, Lu Hou, Lifeng Shang, et al. Grpo-vps: Enhancing group relative policy optimization with verifiable process supervision for effective reasoning. In *International Conference on Learning Representations*, 2026b.

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.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *Advances in Neural Information Processing Systems*, 38:115452–115486, 2026c.

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.

Eunseop Yoon, Hee Suk Yoon, Jaehyun Jang, SooHwan Eom, Qi Dai, Chong Luo, Mark A Hasegawa-Johnson, and Chang D Yoo. Pacr: Progressively ascending confidence reward for llm reasoning. *arXiv preprint arXiv:2510.22255*, 2025.

Adam Younsi, Ahmed Attia, Abdalgader Abubaker, Mohamed El Amine Seddik, Hakim Hacid, and Salem Lahlou. Accurate and diverse llm mathematical reasoning via automated prm-guided gflownets. *arXiv preprint arXiv:2504.19981*, 2025.

Fangxu Yu, Lai Jiang, Haoqiang Kang, Shibo Hao, and Lianhui Qin. Flow of reasoning: Training llms for divergent reasoning with minimal examples. In *International Conference on Machine Learning*. PMLR, 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.

Yurun Yuan, Fan Chen, Zeyu Jia, Alexander Rakhlin, and Tengyang Xie. Trajectory bellman residual minimization: A simple value-based method for llm reasoning. *Advances in Neural Information Processing Systems*, 38:116994–117032, 2026.

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.

Kongcheng Zhang, Qi Yao, Shunyu Liu, Yingjie Wang, Baisheng Lai, Jieping Ye, Mingli Song, and Dacheng Tao. Consistent paths lead to truth: Self-rewarding reinforcement learning for llm reasoning. *Advances in Neural Information Processing Systems*, 38:59849–59887, 2026.

Xuekai Zhu, Daixuan Cheng, Dinghuai Zhang, Hengli Li, Kaiyan Zhang, Che Jiang, Youbang Sun, Ermo Hua, Yuxin Zuo, Xingtai Lv, Qizheng Zhang, Lin Chen, Fanghao Shao, Bo Xue, Yunchong Song, Zhenjie Yang, Ganqu Cui, Ning Ding, Jianfeng Gao, Xiaodong Liu, Bowen Zhou, Hongyuan Mei, and Zhouhan Lin. FlowRL: Matching reward distributions for LLM reasoning. In *International Conference on Learning Representations*, 2026.

A Related Works

Credit Assignment in RLVR. Reinforcement learning with verifiable rewards (RLVR) has become a widely used paradigm for improving LLM reasoning from outcome-level feedback (Cobbe et al., 2021; Shao et al., 2024; Guo et al., 2025; Yu et al., 2026). Critic-free algorithms such as GRPO estimate advantages from groups of sampled responses but propagate the resulting trajectory-level signal uniformly across the response, leaving the heterogeneous contributions of intermediate reasoning decisions unresolved (Shao et al., 2024). Recent work, therefore, develops finer-grained credit through several complementary mechanisms. Value- and rollout-based approaches estimate intermediate state quality or future success (Yue et al., 2025; Kazemnejad et al., 2024); process-supervision methods introduce step- or segment-level feedback through process reward models, early-exit rollouts, or confidence-based progress signals (Lightman et al., 2024; Wang et al., 2024; Dai et al., 2026; Guo et al., 2026; Wang et al., 2026b; Yoon et al., 2025); and selective-credit methods exploit entropy, information salience, or temporal structure to concentrate updates on consequential reasoning decisions (Wang et al., 2026c;a; Sun et al., 2026; Mou et al., 2026). Another line of work uses a principal alternative value-based RL method (Yuan et al., 2026).

These works collectively demonstrate that intermediate reasoning decisions are highly heterogeneous and that non-uniform credit can improve learning efficiency. However, they primarily incorporate process information through rewards, advantages, value estimates, or gradient weighting. In contrast, we study credit assignment within a distribution-matching objective and separate the two roles of intermediate progress. This distinction is consistent with the complementary theoretical perspective showing that outcome supervision need not be inherently more statistically difficult than process supervision under suitable coverage assumptions, suggesting that the practical benefit of intermediate signals may arise partly from how trajectory-level feedback is exploited algorithmically (Jia et al., 2025).

Distribution Matching for LLM Policy Optimization. Distribution-matching objectives provide an alternative to conventional reward-maximizing policy optimization by fitting a policy to a reward-induced distribution over complete trajectories. Recent work adapts this perspective to LLM reasoning through trajectory-level reward matching, diverse reasoning-path generation, and power-transformed target distributions in RLIF (Hu et al., 2024; Yu et al., 2025; Zhu et al., 2026; Chen et al., 2026a). Other formulations exploit pairwise or group-relative consistency among trajectories (Kwon et al., 2026; Liu et al., 2026). These methods largely formulate the matching signal at the level of complete trajectories or derive relative consistency from terminally specified targets. Most closely related to process-aware distribution matching, Younsi et al. (2025) incorporate step-level guidance into mathematical reasoning through an explicit process reward model. In contrast, we do not introduce a separate process reward objective: we condition distribution matching directly on progress measured at decision-aware reasoning transitions. Thus, our focus is not on a new relative-matching mechanism itself, but on how intermediate reasoning progress should modify the optimization and target geometry of distribution matching.

B PROOFS

B.1 PROOF FOR PROPOSITION 3.1

Proof. Since $0 \le p_k \le 1$, the definition $w_k = \alpha|\Delta C_k|(1+p_k)$ immediately gives

$$\alpha|\Delta C_k| \le w_k \le 2\alpha|\Delta C_k|,$$
$$\alpha \sum_{k=1}^{M} |\Delta C_k| \le \sum_{k=1}^{M} w_k \le 2\alpha \sum_{k=1}^{M} |\Delta C_k|,$$

which proves Equation 3.1. For the second claim, by definition of P and N and the telescope of progress increments,

$$C_M - C_0 = \sum_{k=1}^{M} \Delta C_k = P - N,$$
$$\sum_{k=1}^{M} |\Delta C_k| = P + N, \quad \sum_{k=1}^{M} \Delta C_k = P - N.$$

Therefore,

$$\begin{aligned} \mathrm{TV}(C) &= P + N \\ &= |P - N| + 2\min\{P, N\} \\ &= |C_M - C_0| + 2\min\{P, N\}, \end{aligned}$$

which proves the decomposition of TV distance. □

B.2 PROOF FOR PROPOSITION 3.2

Proof. Define $Z(\gamma) = \sum_z p(z)^\gamma$, such that $p_\gamma(z) = \exp(\gamma \log p(z) - \log Z(\gamma))$. The entropy can be written as

$$H(p_\gamma) = -\gamma \mathbb{E}_{p_\gamma}[\log p(z)] + \log Z(\gamma).$$

Moreover,

$$\begin{aligned} &\frac{\partial}{\partial \gamma} \log Z(\gamma) = \mathbb{E}_{p_\gamma}[\log p(z)], \\ &\frac{\partial}{\partial \gamma}[-\gamma \mathbb{E}_{p_\gamma}[\log p(z)]] = -\mathbb{E}_{p_\gamma}[\log p(z)] - \gamma \frac{\partial}{\partial \gamma} \mathbb{E}_{p_\gamma}[\log p(z)], \\ &\frac{\partial}{\partial \gamma} \mathbb{E}_{p_\gamma}[\log p(z)] = \mathrm{Var}_{p_\gamma}[\log p(z)], \\ &\frac{\partial}{\partial \gamma} H(p_\gamma) = -\mathbb{E}_{p_\gamma}[\log p(z)] - \gamma \,\mathrm{Var}_{p_\gamma}[\log p(z)] + \frac{\partial}{\partial \gamma} \log Z(\gamma) \\ &\qquad = -\gamma \,\mathrm{Var}_{p_\gamma}[\log p(z)] \le 0. \end{aligned}$$

Thus, decreasing γ flattens the powered reference distribution, whereas increasing γ sharpens it. Since $\gamma_k = 1 - \beta d_k$, positive and negative progress induce the stated opposite changes in target entropy. □

B.3 Proof for Theorem 3.3

Proof. We first define the log-density error of a trajectory $\tau \in \mathcal{T}_x$ as $q(\tau) := q_x(\tau)$ in Section 3.4.

Forward direction ($\Leftarrow$)**.** Suppose that the policy matches the target distribution, i.e., $\pi_\theta(\tau \mid x) = \rho_x(\tau)/Z(x)$ for every $\tau \in \mathcal{T}_x$. Substituting this relation into the definition of $q(\tau)$ gives

$$q(\tau) = \log \frac{\rho_x(\tau)}{Z(x)} - \log \rho_x(\tau) = -\log Z(x), \qquad \forall \tau \in \mathcal{T}_x.$$

Hence, all trajectories associated with the same prompt share exactly the same log-density error. In particular, for any sampled group $\tau_{1:G}$, we have $q_i = -\log Z(x)$ for every i, and therefore the importance-weighted group mean also satisfies $\bar{q} = -\log Z(x)$. Consequently, every group-relative residual $q_i - \bar{q}$ vanishes, so the PADM loss is zero for every sampled group. Taking the expectation over $\tau_{1:G} \sim \pi_\theta(\cdot \mid x)$ yields

$$\mathcal{J}_{\mathrm{PADM}}(\theta \mid x) = 0.$$

Backward direction ($\Rightarrow$)**.** Suppose that $\mathcal{J}_{\mathrm{PADM}}(\theta \mid x) = 0$. Since the PADM loss is a non-negative weighted sum of squared deviations and $\mathrm{sg}[u_i] > 0$, zero expected loss implies that the loss itself is zero almost surely. Thus, every trajectory in a sampled group must have the same log-density error, i.e., $q(\tau_i) = q(\tau_j), \forall i, j \in \{1, \ldots, G\}$ almost surely. Equivalently, letting $a_i = \mathrm{sg}[u_i]$ and $A = \sum_{i=1}^{G} a_i$, the weighted within-group deviation can be written as

$$\sum_{i=1}^{G} a_i (q_i - \bar{q})^2 = \frac{1}{2A} \sum_{i=1}^{G} \sum_{j=1}^{G} a_i a_j (q_i - q_j)^2.$$

Because all a_i are strictly positive, this quantity is zero if and only if $q_i = q_j$ for every pair in the group. Since $\pi_\theta(\cdot \mid x)$ has full support and the trajectories are sampled independently, every pair $\tau, \tau' \in \mathcal{T}_x$ has positive probability of appearing together in a sampled group. Therefore, zero PADM loss requires $q(\tau)$ to be constant over the entire trajectory space $\mathcal{T}_x$. Namely, there exists a prompt-dependent constant $C(x)$ such that

$$\log \pi_\theta(\tau \mid x) - \log \rho_x(\tau) = C(x), \qquad \forall \tau \in \mathcal{T}_x.$$

Exponentiating gives $\pi_\theta(\tau \mid x) = e^{C(x)} \rho_x(\tau)$. Using the normalization of $\pi_\theta(\cdot \mid x)$ together with the definition of $Z(x)$, we obtain

$$1 = \sum_{\tau \in \mathcal{T}_x} \pi_\theta(\tau \mid x) = e^{C(x)} \sum_{\tau \in \mathcal{T}_x} \rho_x(\tau) = e^{C(x)} Z(x),$$

and hence $e^{C(x)} = 1/Z(x)$. It follows that

$$\pi_\theta(\tau \mid x) = \frac{\rho_x(\tau)}{Z(x)}, \qquad \forall \tau \in \mathcal{T}_x.$$

This proves the equivalence. This proof draws on Theorem 4.1 in CTB (Kwon et al., 2026). □

C Implementation Details

C.1 Observation Study

Observation I. We sample 150 problems from DAPO-MATH-17k with seed 42, using temperature 1.0 and `top_p` = 1.0. Fork selection uses the 80th entropy percentile and a minimum spacing of 256 tokens. Matched controls must lie within 0.10 in normalized trajectory depth and at least 128 tokens from the selected fork. Both positions must leave at least 64 tokens in the source reasoning span: $L_i - t \geq 64$, where L_i is the reasoning length and t is the prefix length. From each position, we sample $K = 8$ independent continuations of exactly 64 tokens. Given their progress values $\Delta C^{(1)}, \ldots, \Delta C^{(K)}$, dispersion is the mean absolute pairwise difference, $D = \frac{2}{K(K-1)} \sum_{a<b} |\Delta C^{(a)} - \Delta C^{(b)}|$. We construct 95% confidence intervals using 10,000 problem-level bootstrap resamples, preserving fork–control pairing.

Observation II. For each original trajectory, we measure progress between consecutive reasoning boundaries and normalize each segment's magnitude as $s_{ik} = |\Delta C_{ik}| / \sum_j |\Delta C_{ij}|$. These shares quantify how total absolute progress is distributed across segments; summing the largest shares measures its concentration in a few transitions. For sign-based summaries, segments with $|\Delta C_{ik}| \leq 10^{-8}$ are classified as near-zero. Trajectories with total absolute progress $\sum_j |\Delta C_{ij}| \leq 10^{-12}$ are excluded from ratio-based summaries. We construct 95% confidence intervals using $20{,}000$ problem-level bootstrap resamples.

C.2 Main Algorithm

Training configurations. We implement PADM and all baselines in the veRL FlowRL framework (Sheng et al., 2025; Zhu et al., 2026). All methods are trained on DAPO-MATH-17k. Within each model scale, all methods use the same initial checkpoint and share the following training settings. Each rollout batch contains 128 prompts, with $G = 4$ responses per prompt, yielding 512 trajectories. The maximum prompt and response lengths are $1{,}024$ and $4{,}096$ tokens, respectively, with left truncation applied to overlong prompts. We use a learning rate of 10^{-6}, 10 warmup steps, a weight decay of 0.1, and an actor minibatch size of 32. All 1.7B and 4B models are trained for 360 and 180 steps, respectively, and evaluated using their final checkpoints. For method-specific hyperparameters, including KL-loss coefficients and process-supervision settings, we follow the official veRL recipe for GRPO (Shao et al., 2024) and the recommended configurations for FlowRL (Zhu et al., 2026) and GRPO-VPS (Wang et al., 2026b), adapting these recipes to the shared training setup above. Experiments use four NVIDIA RTX PRO 6000 Blackwell Server Edition GPUs on a shared cluster managed by Slurm. Rollout tensor parallelism is set to 1, and the rollout engine's GPU memory utilization fraction is set to 0.6.

Hyperparameters. We set the terminal-reward coefficient to $\zeta = 15.0$ following Zhu et al. (2026), the progress-magnitude coefficient to $\alpha = 1.5$, the weight clipping factor to 5.0, and the reference-tempering coefficient to $\beta = 0.25$. The minimum distance between consecutive reasoning boundaries is $g = 128$ tokens for the 1.7B model and $g = 256$ tokens for the 4B model. These parameters govern PADM's process-aware target construction and segment-level credit allocation; the remaining training settings are shared with the baselines at the corresponding model scale.

Evaluation configuration. We evaluate PADM and all baselines using the same benchmark inputs, decoding settings, and answer-verification procedure. We use the final checkpoints after 360 training steps for the 1.7B models and 180 steps for the 4B models. Across all benchmarks and model scales, we sample responses with temperature 0.6 and $\texttt{top_p} = 0.95$, with maximum prompt and response lengths of 1024 and 4096 tokens, the same as the training configuration. For MATH500, Minerva Math, and OlympiadBench, we sample one response per problem and report $\mathrm{Avg@1}$. For AIME24, AIME25, and AMC23, we sample 32 responses per problem and report $\mathrm{Avg@32}$, computed by averaging response correctness across samples and problems. The overall score is the unweighted mean of the six benchmark accuracies.

Ablation Study. To isolate the effect of fork-based boundary placement, we consider two partition ablations that preserve the original number of reasoning segments for each trajectory. We first apply PADM's high-entropy partitioner to obtain the trajectory-specific segment count M. Uniform partitioning divides the reasoning span into M contiguous segments with approximately equal token counts, while random partitioning samples $M - 1$ internal cutpoints from valid token positions using seed 42, without restricting them to high-entropy forks. Both variants preserve the original minimum-gap constraints and final-answer boundary while maintaining exactly M reasoning segments. Progress signals are recomputed on the resulting segments, and all remaining PADM computations and hyperparameters are unchanged.

C.3 Examples

We show excerpts from every reasoning segment of two Qwen3-4B responses to DAPO problems in Tables 3 and 4 to further illustrate our Observation II in Section 3.1. Each ΔC_k measures the full segment's change in answer-length-normalized log progress in the verified answer. Blue/orange indicate gains/declines, not step correctness. Boxes add emphasis; ellipses mark omissions. Wording

is preserved with mathematical typesetting. Indeed, the examples indicate the coexistence of positive and negative gains along both outcome-correct and outcome-incorrect trajectories.

Table 3: **Outcome-correct trajectory with local progress declines.** The response ultimately answers correctly, despite substantial intermediate declines in verified-answer progress followed by recovery.

Qwen3-4B	**Problem and selected reasoning segments**
Problem	When written in standard form, how many zeroes are in the product $(9.2 \times 10^2)(8 \times 10^6)$?
Answers	Truth: 7. Model: $\boxed{7}$ **(correct)**.
Segment 1 $\Delta C_1 = +3.969$	"Hmm, standard form usually means scientific notation, right? Or maybe it's just the regular decimal form?"
Segment 2 $\Delta C_2 = +7.869$	"Then the powers of 10: $10^2 \times 10^6 = 10^{(2+6)} = 10^8$. So the product in scientific notation is $\boxed{7.36 \times 10^8}$. Wait, but 73.6×10^7 would be equivalent? Wait, no, because 7.36×10^8 is the correct scientific notation."
Segment 3 $\Delta C_3 = +20.226$	"If we have 7.36×10^8, converting that to decimal would be 736,000,000. Let's count the zeros. 736,000,000. Let me write that out: $\boxed{736 \text{ followed by 7 zeros?}}$ Wait, no."
Segment 4 $\Delta C_4 = -15.122$	"Wait, that would be 10 digits. But 10^8 is 100,000,000 which is 9 digits. $\boxed{\text{Wait, maybe I messed up.}}$ … So between 736 and the end, there are 7 zeros. But the number is 736,000,000. Let me count the digits."
Segment 5 $\Delta C_5 = -2.978$	"So the answer would be 7 zeros? Wait, but maybe I made a mistake in the calculation. … Wait, hold on! $\boxed{\text{Wait, that contradicts my previous calculation.}}$"
Segment 6 $\Delta C_6 = +4.109$	"I think I made an error here. … So we need to adjust that. 73.6×10^8 is equal to $7.36 \times 10^1 \times 10^8 = 7.36 \times 10^9$. Therefore, the product in standard form (scientific notation) is $\boxed{7.36 \times 10^9}$."
Segment 7 $\Delta C_7 = -3.560$	"Therefore, the number of zeros is 7. But wait, hold on. Wait when I first calculated it as 7.36×10^8, $\boxed{\text{which was wrong.}}$"
Segment 8 $\Delta C_8 = -6.018$	"I see the correct value is 7.36×10^9, which is 7,360,000,000. Which has 7 zeros. So that would mean the answer is 7. … But earlier when I thought it was 7.36×10^8, $\boxed{\text{that was a mistake.}}$"
Segment 9 $\Delta C_9 = +12.137$	"But then 73.6×10^8 is 7.36×10^9, which is correct. Therefore, the standard form decimal is $\boxed{7{,}360{,}000{,}000}$. … So the number of zeros is 7. Therefore, the answer is 7."
Segment 10 $\Delta C_{10} = -0.650$	"But let me check if the problem is asking for the number of zeros in standard form. … Therefore, I think the correct interpretation is $\boxed{\text{decimal form}}$, which is 7,360,000,000, which has 7 zeros. Therefore, the answer is 7."
Segment 11 $\Delta C_{11} = +12.518$	"So the number of zeros is 7. $\boxed{\text{Therefore, the answer is 7.}}$ Hence, Answer: 7."
Confidence trace	Normalized log confidence decreases from -11.312 at boundary 3 to -34.880 at boundary 8, then recovers to -10.875 at boundary 11.
Concentration	Segments 3, 4, and 11 account for **53.7%** of total absolute progress across 11 segments; $\sum_{k=1}^{11} \lvert\Delta C_k\rvert = 89.155$.

Table 4: **Outcome-incorrect trajectory with substantial local confidence gains.** The response contains both positive and negative local progress despite ultimately producing an incorrect answer.

Qwen3-4B	**Problem and selected reasoning segments**
Problem *(paraphrased)*	Find the smallest possible value of $(p+q+r)^2$, where p, q, r are nonzero integers satisfying $p^2+q^2=r^2$.
Answers	Truth: 4. Model: $\boxed{144}$ **(incorrect)**.
Segment 1 $\Delta C_1 = -0.688$	"Okay, so I need to find the smallest possible value of $(p+q+r)^2$ where p, q, r are nonzero integers satisfying $p^2+q^2=r^2$. Hmm, let me think."
Segment 2 $\Delta C_2 = +10.068$	"If $p=3$, $q=4$, $r=5$, then $p^2+q^2=9+16=25=5^2$. Then $\boxed{(3+4+5)^2=12^2=144}$. Is that the smallest possible? Maybe there's a smaller triple?"
Segment 3 $\Delta C_3 = -7.012$	"Wait, maybe there is a non-primitive triple with $\boxed{\text{smaller sum?}}$ Let me check. Wait, the $3, 4, 5$ triple is the smallest. Let me check if there are any smaller numbers."
Segment 4 $\Delta C_4 = +2.667$	"For example, $4, 3, 5$ is same as $3, 4, 5$. What about $5, 12, 13$? No. Wait, maybe $7, 24, 25$? No. So $3, 4, 5$ is the smallest. But maybe there is a different triple with $\boxed{\text{smaller sum?}}$"
Segment 5 $\Delta C_5 = -1.536$	"Case 1: $r-q=1$, $r+q=4$. Adding equations: $2r=5 \Rightarrow r=2.5$, which is not integer. $\boxed{\text{Disregard.}}$"
Segment 6 $\Delta C_6 = +4.855$	"Then p, q, r are $\boxed{\text{positive integers}}$ such that $p^2+q^2=r^2$ and $p+q+r=10$. Let's see. Maybe r is 5? Then $p+q=5$. So p and q are positive integers adding up to 5. Let's check possible pairs:"
Segment 7 $\Delta C_7 = +6.206$	"For example, $r=5$: $p+q=6$. Check $(3,3)$: $9+9=18$. Not square. $(2,4)$: $4+16=20$. Not square. $(1,5)$: $1+25=26$. No."
Segment 8 $\Delta C_8 = -6.140$	"So no solution for sum 11. How about sum 12? Which is $3+4+5$. That works. $\boxed{\text{So that's the minimal.}}$"
Segment 9 $\Delta C_9 = -1.393$	"For example, maybe $6, 8, 10$ is 24. No, that's bigger. What about $5, 12, 13$? Sum is 30. So no. $\boxed{\text{So } 3, 4, 5 \text{ is the minimal.}}$"
Segment 10 $\Delta C_{10} = +10.769$	"If I take $m=3$, $n=2$. Then $p=9-4=5$, $q=2\times3\times2=12$, $r=9+4=13$. So that gives $5, 12, 13$. Sum $5+12+13=30$. Which is bigger than 12. So worse. Alternatively, take $m=2$, $n=1$. Which gives $3, 4, 5$. So that's the minimal. So yes, $\boxed{144 \text{ is the answer.}}$"
Answer error	The displayed reasoning focuses on positive triples, overlooking the admissible choice $(p, q, r) = (3, 4, -5)$, which achieves the minimum value 4.
Concentration	Segments 2 and 10 account for **40.6%** of total absolute progress across 10 segments; $\sum_{k=1}^{10} \lvert\Delta C_k\rvert = 51.333$. Both gains are measured in confidence in the reference answer 4.