What Drives the Zero-to-One Jump? Diagnosing and Steering Hard-Prompt Learning in RLVR

Anonymous ACL submission

Abstract

Reinforcement Learning with Verifiable Rewards (RLVR) has substantially improved mathematical reasoning, yet progress on hard prompts often stalls: a prompt can remain unsolved for many training steps before abruptly transitioning to solved—a discrete "zero-to-one" jump whose underlying dynamics and steerability remain poorly understood. We investigate two questions: *what drives this transition, and can it be steered?* Through event-based analysis measuring the time-to-first-positive (TTFP), we find that negative rollouts preceding successful transitions consistently retain higher early-stage generation entropy, distinguishing near-solvable failures from dead ends. This signal motivates **N**egative-**A**ugmented **P**olicy **O**ptimization NAPO, a GRPO (Group Relative Policy Optimization)-integrated method that ranks negatives in unsolved groups via an early-branching driver and assigns graded advantages, injecting informative gradients precisely where standard group-relative updates vanish. Under fixed rollout budgets, NAPO accelerates early solution discovery and improves Pass@256 by 10.0 points on AIME24 and 6.6 points on AIME2025, demonstrating that zero-to-one transitions are diagnosable and steerable through driver-guided negative learning. Our code is available at `anonymous.4open.science/r/Z2O-7E6B`.

1 Introduction

Reinforcement Learning with Verifiable Rewards (RLVR) (Lambert et al., 2025) has rapidly emerged as a dominant paradigm for post-training reasoning models, particularly in mathematical problem solving where correctness can be automatically verified (Yang et al., 2025a; Guo et al., 2025). At each training step, the policy samples multiple rollouts per prompt and receives binary rewards indicating whether each final answer is correct. When some rollouts in a group succeed, the resulting contrast between positive and negative samples provides a clear gradient that drives steady improvement. However, on sufficiently hard prompts the on-policy pass rate is so low that entire rollout groups contain only incorrect samples for many consecutive steps. In group-relative methods such as GRPO (Shao et al., 2024; Guo et al., 2025), this eliminates reward contrast and yields near-zero advantages on precisely the prompts that most need exploration. As a result, hard prompts often remain stagnant for many steps and then transition abruptly from unsolved to solved — a discrete "zero-to-one" jump rather than smooth refinement. The dynamics that drive this transition, and whether they can be steered, remain largely unexplored.

Answering these questions, however, requires confronting two intertwined obstacles that current RLVR pipelines leave unresolved. The first obstacle is diagnosing zero-to-one events under binary supervision. Standard RLVR diagnostics offer little purchase on what happens inside a hard prompt before it is first solved: pass@k (Chen et al., 2021) aggregates end-of-training capability and cannot localize the moment a previously-unsolved prompt becomes solvable, while group-level rewards remain identically zero across long stretches of training and therefore carry no information about which unsolved prompts are close to a breakthrough. What is missing is (i) an event-based view of training that pinpoints when transitions occur and (ii) an internal, rollout-level signal that distinguishes near-solvable failures from true dead ends using only the information already available at sampling time.

The second obstacle is turning heterogeneous negatives into informative gradient on fully-negative groups. A majority of RLVR methods build on grouped objectives such as GRPO and its variants (Shao et al., 2024; Yu et al., 2025; Zheng et al., 2024; Wu et al., 2026b), where all-negative

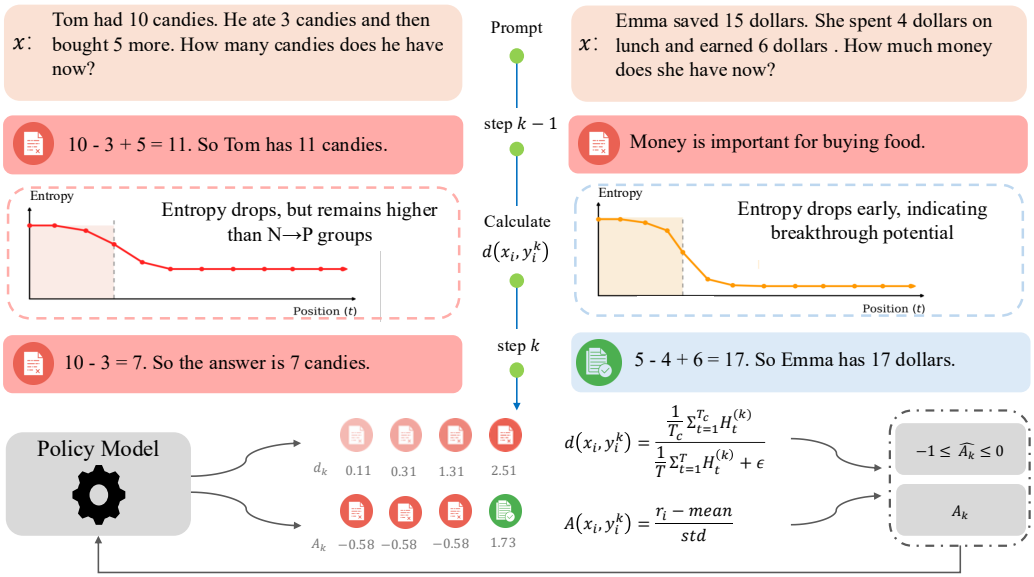


Figure 1: Z2O pipeline

groups produce vanishing advantages and provide little learning signal (Ma et al., 2026; Yu et al., 2025). Recent variants restore a gradient through weighted negative learning, virtual rewards, or entropy-aware updates (Zhu et al., 2025; Nan et al., 2025; Le et al., 2026), but they apply roughly uniform pressure across negatives. The open problem is therefore how to assign graded credit among them under only binary rewards, in a way that injects gradient into unsolved groups without destabilizing the rest of training.

We analyze token-level generation entropy during RLVR training and find that negative rollouts from prompts that later undergo a Negative-to-Positive (N→P) transition exhibit front-loaded uncertainty, with early-response entropy patterns that distinguish them from rollouts on prompts that remain negative (N→N). Based on this signal, we propose NAPO (Negative-Augmented Policy Optimization), a GRPO-integrated method that ranks negative samples by front-loaded entropy and injects informative gradients into unsolved groups.

Our contributions are threefold:

- We study zero-to-one learning dynamics in RLVR through event-based metrics, especially **N→P** transitions and **TTFP**, to characterize how hard prompts first become solvable.
- We identify a token-entropy pattern associated with N→P transitions and propose early-branching drivers that rank negative rollouts by their latent learning potential.
- We introduce NAPO, a GRPO-integrated algorithm that assigns graded advantages to negative samples according to their driver-based ranking while maintaining stable updates via in-group normalization, improving token-efficient exploration on hard prompts as reflected by better TTFP and higher pass@k.

2 Preliminaries

Related Works. Reinforcement learning (RL)(Sutton et al., 1998) has been shown to improve LLM reasoning via both exploration and exploitation (Auer et al., 2002; Cheng et al., 2026). Approaches such as RLHF (Ouyang et al., 2022), Test-Time RL (Zuo et al., 2025), and RLVR (Guo et al., 2025) train policies using verifiable rewards, often with policy-gradient methods like PPO, GRPO, and REINFORCE (Schulman et al., 2017; Shao et al., 2024; Williams, 1992). These methods can stall on very hard prompts (Ma et al., 2026). Recent work addresses this by reallocating rollouts toward learnable prompts and by extracting informative signals from negative samples, including token-level entropy to encourage exploration and reduce overfitting (Zhu et al., 2025; Le et al., 2026; Fang et al., 2026). We further introduce the **negative-to-positive transition** to predict and

leverage latent prompt learnability to over come this challenge.

GRPO for grouped RLVR. Following recent RLVR practice for reasoning models, we sample a group of G rollouts $\{y_i\}_{i=1}^{G}$ for each prompt and optimize the policy with Group Relative Policy Optimization (GRPO) (Guo et al., 2025; Shao et al., 2024). Let $\pi_{\theta_{\text{old}}}$ be the snapshot policy and define the token-level likelihood ratio $\rho_{i,t}(\theta) = \frac{\pi_\theta(a_{i,t}|s_{i,t})}{\pi_{\theta_{\text{old}}}(a_{i,t}|s_{i,t})}$. GRPO then applies the clipped surrogate objective

$$\max_\theta \mathbb{E}_x \mathbb{E}_{\,y_i\sim\pi_{\theta_{\text{old}}}} \left[\sum_{i=1}^{G} \sum_t \ell_{i,t} - \beta\, \mathrm{KL}(\pi_\theta \,\|\, \pi_{\text{ref}}) \right],$$
$$\ell_{i,t} = \min\Big(\rho_{i,t}(\theta)\hat{A}_{i,t}, \quad \mathrm{clip}(\rho_{i,t}(\theta), 1-\epsilon, 1+\epsilon)\ \hat{A}_{i,t}\Big), \tag{1}$$

where $\hat{A}_{i,t}$ is the group-relative advantage, π_{ref} is a reference policy, and β controls KL regularization. For brevity, we use π_θ and π_{ref} to denote the distributions conditioned on a given prompt x, i.e., $\pi_\theta(\cdot \mid x)$ and $\pi_{\text{ref}}(\cdot \mid x)$, respectively.

PSR/NSR decomposition. For verifiable-reward post-training, each rollout receives a binary outcome. We view the learning signal through the positive-sample reward (PSR) and negative-sample reward (NSR) decomposition (Zhu et al., 2025). Let $r(x, y) \in \{-1, +1\}$ be the verifiable reward for completion y on prompt x. A simplified RLVR objective is

$$\mathcal{L}_{\text{RLVR}}(\theta) \;:=\; -\mathbb{E}_{x\sim\mathcal{D}}\, \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[r(x, y)]\,. \tag{2}$$

Using $r = \mathbb{I}[r = +1] - \mathbb{I}[r = -1]$, we can write

$$\mathcal{L}_{\text{RLVR}}(\theta) = \underbrace{-\mathbb{E}_{x\sim\mathcal{D}}\, \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[\mathbb{I}[r(x, y) = +1]]}_{\mathcal{L}_{\text{PSR}}(\theta)} + \underbrace{\mathbb{E}_{x\sim\mathcal{D}}\, \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[\mathbb{I}[r(x, y) = -1]]}_{\mathcal{L}_{\text{NSR}}(\theta)}. \tag{3}$$

PSR reinforces verified rollouts, whereas NSR suppresses failed ones. Both sides can provide useful superivision on reasoning tasks (Zhu et al., 2025).

Token-level uncertainty and rollout-level drivers. To analyze how N→P happens, we analyze the patterns of negative rollouts before N→P happens. We measure token-level uncertainty with the entropy of the next-token distribution (Wang et al., 2025; Yang et al., 2025b). For each rollout, let $p_t^{(k)}$ denote the top-k truncated next-token distribution at position t after renormalization, with invalid and padding positions masked out. We define the token-level entropy sequence

$$H_t^{(k)} = -\sum_v p_t^{(k)}(v) \log p_t^{(k)}(v), \tag{4}$$

and its exponential form

$$\rho_t^{(k)} = \exp\big(H_t^{(k)}\big), \tag{5}$$

which we use as a token-level perplexity / effective branching statistic. Our goal is not to measure absolute early-stage uncertainty in isolation, but to summarize the *relative prominence* of early uncertainty within each rollout. [1]

N→P Transition. To localize progress on hard prompts, let $q_t(x)$ indicate whether evaluation at training step t contains at least one positive rollout for prompt x. We define the time-to-first-positive (TTFP) as

$$\mathrm{TTFP}(x) \;=\; \min\{t \,:\, q_t(x) = 1\},$$

with $\mathrm{TTFP}(x) = \infty$ if no positive appears within the training horizon. We define an N→P transition as

$$q_{t-1}(x) = 0 \;\wedge\; q_t(x) = 1.$$

This implies that the model solve one problem where previous attempts fails.

3 Signals of Near-Solvable Failures

This section answers the question of what drives the N→P transition.

3.1 Experiment Setup

We examines two axes of variation in this section: (i) *within-batch*, comparing rollouts inside a single all-negative group, and (ii) *between-replay*, comparing two training continuations that share a base state but differ in sampling stochasticity.

Fixed-batch protocol. RLVR on hard prompts exhibits strong temporal correlations across training steps (Li et al., 2025; Zheng et al., 2025). To expose this structure directly, we train Qwen2.5-Math-7B on a fixed set of 1024 DAPO-17k prompts for 10 consecutive gradient steps, sampling $G = 8$

[1] In off-policy settings, rollouts may come from another policy π_ϕ, but unless stated otherwise we still compute H_t under the training policy π_θ. $k = 20$ unless otherwise stated

	AMC23		AIME2025		Math-Beyond		Training Set		
Replay	N→P	P→N	N→P	P→N	N→P	P→N	N→P	P→N	TTFP
n1	**2**/3	0.33	**2.67**/20	2	**12**/134	5.33	**120**	116	**1.213**
n2	1.67/3	2.67	2/20	2.33	8.33/134	5.33	117.67	102.67	1.225

Table 1: **Transition statistics at $t^* = 1$ for the *n1–n2* replay pair.** A prompt is labeled as P if at least one rollout is correct under the fixed rollout budget, and as N otherwise. In the validation columns, a/b reports a N→P transitions among b prompts that are negative at the base checkpoint. Results are averaged over three runs, and TTFP is computed over the first three steps.

rollouts per prompt per step. Holding the prompt batch fixed across steps lets us align rollout-internal uncertainty patterns with later N→P events using the metrics of §2. Unless otherwise noted, we use temperature $T = 1.0$, maximum context length 4096, learning rate 10^{-6}, and zero entropy/KL penalties.[2]

Critical-step replay. Across the 10-step window, we identify the step t^* at which N→P events peak and treat it as the *critical checkpoint*. Starting from this shared state, we replay training with the same base policy, optimizer state, data order, and hyperparameters, varying only rollout stochasticity. This yields two representative continuations, denoted n_1 and n_2. We do not label either replay as "good" or "bad" at this point.

Outcome measures. *(i) Prompt-level transition label.* Among prompts that are negative at the base checkpoint t^*, we call a prompt **N→P** if at least one later checkpoint within the replay window produces a positive rollout for it, and **N→N** otherwise. Each negative rollout inherits its prompt's label. At training time the binary reward gives both groups identical labels; the partition into N→P and N→N is only observable *after* the replay window, which is precisely what makes the within-batch driver signal of §3.2.2 non-trivial.

(ii) Replay-level transition counts. For each replay we report the number of N→P and P→N transitions under 64 rollouts per prompt, together with the mean TTFP over prompts within the replay window. Smaller first-positive indices indicate earlier solution discovery.

(iii) Validation sets. We evaluate replay checkpoints on AMC23, AIME 2025, and Math-Beyond (Mayilvahanan et al., 2025) under the same 64-rollout budget.

[2]Full hyperparameters appear in Appendix B.1 and Table 3.

3.2 Findings and Implications

We first define a rollout-time driver that compresses the entropy–position pattern into a single scalar (§3.2.1), then validate it at each level in turn (§3.2.2 and §3.2.3).

3.2.1 An early-branching driver

Why a ratio, not an absolute level. A naive metric would be the average entropy of a rollout, but the contrast we care about is not "more entropy" in absolute terms; it is a difference in the *timing* of uncertainty—how much of a rollout's total uncertainty is allocated to its opening tokens (Fang et al., 2026).

Definition. Let $T_e = \lceil 0.2\,T \rceil$ denote the number of tokens in the first 20% of a response of length T. The *early mean-perplexity ratio* is

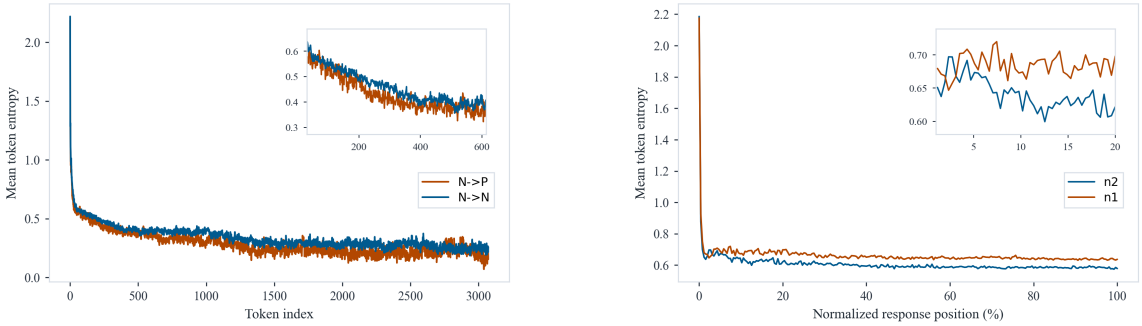
$$d_{\text{early-ppl}} = \frac{\frac{1}{T_e}\sum_{t=1}^{T_e} \rho_t^{(k)}}{\frac{1}{T}\sum_{t=1}^{T} \rho_t^{(k)}}, \tag{6}$$

where $\rho_t^{(k)} = \exp\big(H_t^{(k)}\big)$ [3]is the exponential of the top-k truncated entropy defined in Eq. (5). Values of $d_{\text{early-ppl}} > 1$ indicate that uncertainty is concentrated early in the response.

Choice of the early fraction. We compare 0.1, 0.2, and 0.5 on the n_1–n_2 replay pair (Fig. 5[4]). Among these candidates, 0.2 yields the clearest and most stable separation between N→P and N→N rollouts while remaining robust across both replays, and we use it throughout the remainder of the paper.

[3]Since entropy is measured on a logarithmic scale, $\exp(H_t^{(k)})$ can be interpreted as a top-k truncated local perplexity, i.e., the effective branching factor of plausible next-token continuations. This is related to the branching-factor view in (Yang and Holtzman, 2025).

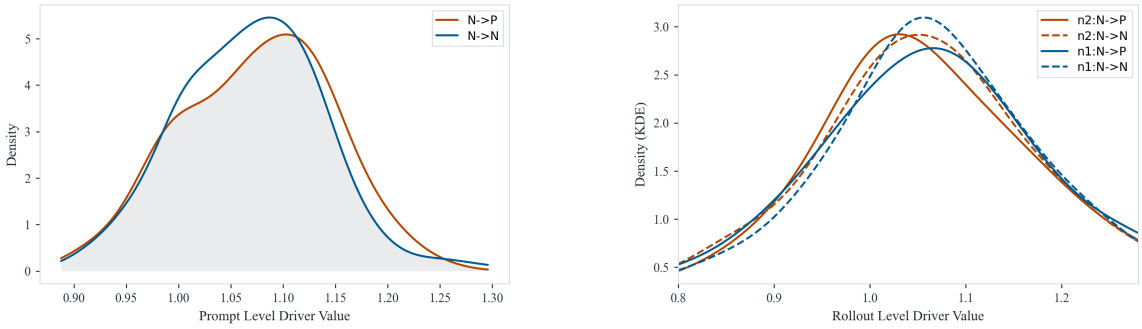
[4]Details are deferred to §5.



(a) Entropy-position curves for the bad replay *n2*.

(b) Comparison of the N→P group between *n1* and *n2*.

Figure 2: **Entropy-position plots.** The main separation is localized in the early part of generation. Prompts that later undergo N→P transitions preserve more early-stage uncertainty than prompts that remain N→N, and this pattern is sharper in the good replay *n1* than in the bad replay *n2*.



(a) Driver distribution for the good replay *n1*. See Appendix C for statistics' demonstration

(b) Driver distribution for the bad replay *n2* and good replay *n1*.

Figure 3: **Within-batch separation under the practical driver $d_{\text{early-ppl}}$.** Rollouts associated with prompts that later undergo N→P transitions are systematically right-shifted relative to rollouts from prompts that remain N→N. The separation is present in both replays, but is cleaner and more pronounced in the good replay *n1*.

3.2.2 Within-batch evidence: negatives are not homogeneous

Entropy–position pattern. Figure 2a plots mean token entropy as a function of position for negative rollouts in replay n_1. The N→P group maintains noticeably higher uncertainty at the beginning of the response, whereas the N→N group collapses earlier to lower-entropy continuations. The two groups converge later in the response, confirming that the discriminative signal is localized in the early stage.

Driver separation. Figure 3a shows the distribution of $d_{\text{early-ppl}}$ for the two groups in replay n_1. Negatives associated with future N→P prompts are systematically shifted toward larger driver values than negatives associated with N→N prompts, mirroring the qualitative pattern of Fig. 2a under the single scalar of Eq. (6).

Claim A (within-batch). The driver provides a rollout-level ordering signal that separates near-solvable failures from dead ends inside an otherwise indistinguishable negative pool.

3.2.3 Between-replay evidence: the driver tracks aggregate transition tendency

Entropy–position pattern. Figure 2b restricts attention to N→P negatives and compares n_1 against n_2. The good replay n_1 preserves more early-stage uncertainty than the bad replay n_2, sharpening the front-loaded pattern of Claim A but now across replays rather than within one. This is consistent with the relevant contrast being a matter of timing rather than magnitude: the same shape of distribution is shifted earlier in n_1 than in n_2.

Driver distribution. Figure 3b overlays all four (replay × label) groups. n_1 allocates more probability mass to high-driver rollouts than n_2 does in both groups. [5]

[5] Batch-level summary statistics quantifying this right-tail dominance are reported in Appendix C.

Realized transitions. Table 1 reports the downstream outcome. Starting from an identical base state, the good replay n_1 realizes more N→P transitions on the training set and attains a smaller mean first-positive index than n_2. The ranking of realized transitions matches the ranking of right-tail driver mass, indicating that the batch-level driver shift is not merely a cosmetic difference but is related to *which* replay will subsequently break more prompts out of the all-negative regime. This is consistent with recent RLVR observations that low-success prompts exhibit structured temporal dynamics rather than noise-like fluctuations throughout training (Zheng et al., 2025).

Claim B (between-replay). The driver also functions as a batch-level diagnostic: batches whose driver distribution has a heavier right-tail mass, go on to realize more N→P events.

Takeaway

Prompts approaching the N→P boundary exhibit a distinctive form of front-loaded uncertainty in their negative rollouts. The early-branching driver $d_{\text{early-ppl}}$ captures this pattern at two levels of aggregation: it separates N→P from N→N negatives *within* a batch (Claim A), and it separates higher-yield from lower-yield batches *across* replays (Claim B).

4 Algorithm

We now turn the empirical findings above into a training design. Our goal is to translate the driver into a practical update rule for RLVR.

4.1 Theoretical Interpretation

We provide a view of Fisher information to theoretically interpret the driver on the negative rollouts.

Fisher information as a measure of policy plasticity. Larger local Fisher means that a failed continuation is easier to reshape under a small parameter update, so the policy remains more *plastic* around that rollout. Let $\mathbf{p}_t = \pi_\theta(\cdot \mid x, y_{<t})$ be the next-token distribution at step t, and let $\mathbf{F}_t$ denote the corresponding token-level Fisher contribution.

Lemma 4.1 (Token-level Fisher information and entropy)**.** *For token distribution $\mathbf{p}_t$, the token-level Fisher trace is*

$$\mathrm{Tr}(\mathbf{F}_t) = 1 - \sum_{k \in V} p_{t,k}^2.$$

Accordingly, Shannon entropy serves as a useful proxy for local Fisher information: higher-entropy token distributions typically correspond to more policy-plastic states, although the two quantities are not identical in general.

Implication for negative learning. Lemma 4.1 suggests that uniform penalties are wasteful: low-entropy failures are already rigid, so pushing them down mostly reinforces dead ends. By contrast, failures generated in higher-entropy states are more *plastic* and retain nearby alternatives, making negative updates more likely to redirect the policy toward useful exploration. Empirically, the informative plasticity is concentrated early in the rollout, which is why NAPO uses the early-branching driver rather than generic response-level entropy. In short, we want to penalize failures more strongly where the policy is still movable.

Takeaway

The driver indicates that failures arising in more *plastic* regions of the policy. The N→P phenomenon suggests that failures with smaller driver values are not random or hopeless: they come from prompts whose trajectories can cross into correctness under nearby policy shifts.

This leads to a design intuition that larger penalties can be assigned to negative trajectories with larger drivers where the policy still retains meaningful uncertainty and plausible alternatives.

4.2 Negative-Augmented Policy Optimization

Motivated by the heterogeneity of negative rollouts, we propose **NAPO**, a GRPO-integrated algorithm that augments standard policy optimization with *driver-weighted negative learning* on unsolved groups.

Driver-weighted negative advantage. NAPO uses the driver to modulate negative learning strength inside unsolved groups. Given driver scores $d(x, y_i)$ for rollouts in a fully negative group, we first re-normalize them into $[0, 1]$ to avoid exploded gradient while preserving a relevant advan-

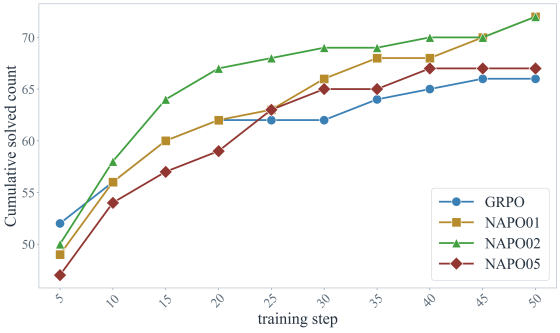


Figure 4: Cumulative solved prompts on validation sets. NAPO achieves faster early progress, solving more prompts within the first 50 steps, while maintaining comparable performance at later stages.

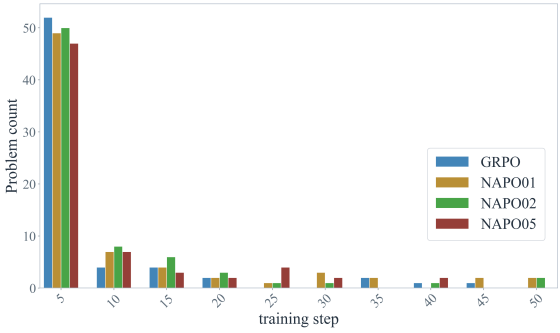


Figure 5: Number of solved problems across training steps. Setting the early-fraction gate to 0.2 leads to the fastest exploration of correct solutions, consistent with our inference-time findings.

Algorithm 1 NAPO: Negative-Augmented Policy Optimization

Sample prompts x; for each x, sample rollouts $\{y_i\}$ and rewards $r(x, y_i)$;
if $\exists i : r(x, y_i) = +1$ **then**
  Compute standard GRPO advantages $A_{\text{GRPO}}(x, y_i)$;
  Update the policy with GRPO loss.
else
  Compute driver scores for all negative rollouts by (6);
  Re-normalize scores into $[0, 1]$ by (7);
  Assign rollout-level advantages by (8);
  Update the policy with GRPO loss.
end if

tage magnitude:

$$\tilde{d}(x, y_i) = \frac{d(x, y_i) - \min_j d(x, y_j)}{\max_j d(x, y_j) - \min_j d(x, y_j) + \epsilon}. \tag{7}$$

We then assign rollout-level advantages as

$$A(x, y) = \begin{cases} A_{\text{GRPO}}(x, y), & \text{if } p(x) = 1, \\ -\tilde{d}(x, y), & \text{if } p(x) = 0, \end{cases} \tag{8}$$

where $p(x) = \mathbb{I}[\exists i : r(x, y_i) = +1]$ indicates whether the current group is solved.

Objective: Grouped advantage on solved groups and weighted NSR on unsolved groups. Let $p(x) = \mathbb{I}[\exists i : r(x, y_i) = +1]$ denote whether the current group is solved. Equivalently, the rollout-level advantage in Eq. (8) can be written as

$$A(x, y_i) = p(x)\, A_{\text{GRPO}}(x, y_i) - (1 - p(x))\, \tilde{d}(x, y_i). \tag{9}$$

When $p(x) = 1$, Eq. (9) reduces to the standard GRPO clipped update. When $p(x) = 0$, we have $A(x, y_i) = -\tilde{d}(x, y_i)$, so the same clipped surrogate becomes a weighted NSR loss in which larger-driver negatives receive larger-magnitude penalties while clipping preserves the usual PPO/GRPO trust-region effect.

5 Evaluation

We empirically evaluate the effectiveness of NAPO on hard prompts.

Setup. We evaluate NAPO under a fixed rollout budget and compare against GRPO and negative-sample-based baselines, including NSR's weighted variant (w-REINFORCE). We also evaluate the impact of early fraction ratio, where NAPO refers to taking the first 20% of the response in the driver, and NAPO05 takes 50%. We report (i) Pass@k on validation sets and (ii) TTFP. Experiments are conducted on multiple validation sets (AMC23, AIME2025, AIME24, and Math-Beyond), covering a wide range of difficulty levels. [6]

Training protocol. We use the verl framework(Sheng et al., 2025) and train on DAPO17k (Yu et al., 2025), a challenging dataset for Qwen2.5-Math-7B (Yang et al., 2024), where the initial pass rate is below 0.2. For each prompt, we sample $G = 8$ rollouts per step. Unless otherwise specified, we use temperature $T = 1.0$ during training, a maximum context length of 4096, and train for one epoch.[7] We report both large-scale validation

[6] We additionally report training-batch transition statistics, with extended ablations deferred to the appendix.

[7] All other hyperparameters follow Section 3.1, with full details provided in Appendix B.1 and Table 3.

Method	Pass@k								
k	1	2	4	8	16	32	64	128	256
AMC23									
Base	0.2847	0.4512	0.6237	0.7546	0.8400	0.9010	**0.9495**	**0.9838**	**1.0000**
GRPO	**0.7352**	**0.7910**	0.8211	0.8443	0.8681	0.8972	0.9306	0.9579	0.9750
W-REINFORCE	0.4514	0.5700	0.6570	0.7245	0.7900	0.8555	0.9086	0.9450	0.9750
NAPO02	0.7141	0.7718	0.8051	0.8328	0.8628	0.8940	0.9222	0.9467	0.9750
NAPO05	0.7191	0.7828	**0.8247**	**0.8556**	**0.8845**	**0.9147**	0.9438	0.9654	0.9750
AIME2025									
Base	0.0305	0.0561	0.0964	0.1496	0.2074	0.2657	0.3216	0.3684	0.4000
GRPO	**0.1346**	**0.1753**	**0.2165**	0.2563	0.2910	0.3271	0.3740	0.4294	0.4667
W-REINFORCE	0.0254	0.0439	0.0689	0.0970	0.1300	0.1767	0.2425	0.3278	0.4333
NAPO02	0.1220	0.1648	0.2113	**0.2631**	**0.3175**	**0.3695**	**0.4195**	**0.4745**	**0.5333**
NAPO05	0.1138	0.1576	0.2008	0.2475	0.3006	0.3555	0.4130	0.4741	**0.5333**
AIME24									
Base	0.0857	0.1508	0.2415	0.3425	0.4343	0.5112	0.5819	0.6575	0.7333
GRPO	0.2832	0.3574	0.4242	0.4860	0.5391	0.5869	0.6330	0.6719	0.7000
W-REINFORCE	0.1516	0.2322	0.3105	0.3706	0.4152	0.4500	0.4821	0.5209	0.5667
NAPO02	**0.2988**	**0.3878**	**0.4648**	**0.5314**	**0.5917**	**0.6504**	**0.7048**	**0.7564**	**0.8000**
NAPO05	0.2898	0.3762	0.4490	0.5133	0.5734	0.6301	0.6810	0.7248	0.7667
Math-Beyond									
Base	0.0426	0.0702	0.1051	0.1446	0.1879	0.2357	**0.2866**	**0.3391**	**0.3978**
GRPO	0.1337	0.1574	0.1780	0.1984	0.2205	0.2454	0.2723	0.3014	0.3370
W-REINFORCE	0.0587	0.0866	0.1147	0.1400	0.1639	0.1899	0.2197	0.2564	0.3094
NAPO02	0.1348	0.1612	0.1882	0.2130	0.2334	**0.2527**	0.2763	0.3105	0.3591
NAPO05	**0.1383**	**0.1680**	**0.1948**	**0.2164**	**0.2337**	0.2508	0.2704	0.2926	0.3204

Table 2: Pass@k comparison across datasets. The row corresponding to our method is highlighted in gray, the best value within each dataset block and column is shown in bold, and the second-best value is underlined. For W-REINFORCE method, we set the positive advantage weight to 0.5 to handle the much harder prompts.

results and controlled experiments under identical training batches (as in § 3.2).

Results. (1) Improved sample efficiency on hard prompts. As shown in Figure 4, NAPO achieves faster TTFP across datasets, solving more prompts in early training stages. This indicates improved sample efficiency, particularly in low-success regimes where standard GRPO suffers from weak learning signals.

(2) Effect of negative scaling. Comparing NAPO and NAPO05 reveals a clear trade-off rather than a uniform winner. The milder NAPO05 variant performs best on AMC23, on the highest-k region of AIME2025, and on low-k Math-Beyond, whereas full NAPO is clearly strongest on AIME24 and in the mid-k regime of AIME2025. This pattern suggests that negative updates should be tuned to benchmark difficulty: stronger shaping helps when there is substantial room for zero-to-one improvement, while milder scaling is more robust once exploration is less constrained.

6 Conclusion

We studied the zero-to-one problem in RLVR, motivated by two obstacles: diagnosing $\mathrm{N} \to \mathrm{P}$ transitions under binary supervision, and extracting informative gradients from heterogeneous negatives in fully-unsolved groups. Using event-based metrics (TTFP and $\mathrm{N} \to \mathrm{P}$ events), we showed that failures preceding successful transitions retain higher early-stage uncertainty, providing a practical signal to separate near-solvable rollouts from dead ends. Building on this, we proposed NAPO, a GRPO-integrated method that assigns graded negative advantages according to driver scores under a Fisher-information view, thereby injecting structured gradients into unsolved groups while preserving stable updates on solved ones. NAPO accelerates early solution discovery and improves Pass@256 by 10.0 points on AIME24 and 3.3 points on AIME2025, confirming that driver-guided negative learning is an effective way to broaden reasoning boundaries in low-success RLVR.

Limitations

Our analysis and method are currently focused on math-oriented RLVR settings with binary verifiers. Although these tasks provide a clean testbed for studying zero-to-one transitions, the same entropy-based driver may behave differently in broader domains, longer-horizon reasoning tasks, or environments with partial-credit, noisy, or subjective feedback. Extending NAPO to such settings may require more robust driver definitions and calibration procedures.

NAPO also relies on rollout-level statistics collected during training. Its benefits may therefore depend on the rollout budget, group size, decoding temperature, and the quality of the base policy. When the model rarely approaches a correct solution, early uncertainty patterns may be insufficient to identify useful negative samples; conversely, when success rates are already high, the additional weighting of negatives may provide limited gains. Future work should evaluate the method under more diverse model scales, verifier designs, and compute budgets.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. 2024. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12248–12267, Bangkok, Thailand. Association for Computational Linguistics.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2–3):235–256.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé 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, and 39 others. 2021. Evaluating large language models trained on code. *CoRR*, abs/2107.03374.

Zhipeng Chen, Xiaobo Qin, Youbin Wu, Yue Ling, Qinghao Ye, Wayne Xin Zhao, and Guang Shi. 2025a. Pass@k training for adaptively balancing exploration and exploitation of large reasoning models. *arXiv preprint arXiv:2508.10751*.

Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025b. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? In *Advances in Neural Information Processing Systems*, volume 38, pages 57654–57689. Curran Associates, Inc.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Xin Zhao, Zhenliang Zhang, and Furu Wei. 2026. Reasoning with exploration: An entropy perspective. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(36):30377–30385.

Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. 2026. GPG: A simple and strong reinforcement learning baseline for model reasoning. In *The Fourteenth International Conference on Learning Representations*.

Yangyi Fang, Jiaye Lin, Xiaoliang Fu, Cong Qin, Haolin Shi, Chaowen Hu, Lu Pan, Ke Zeng, and Xunliang Cai. 2026. How to allocate, how to learn? dynamic rollout allocation and advantage modulation for policy optimization. *Preprint*, arXiv:2602.19208.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

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

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Christopher Wilhelm, Luca Soldaini, and 4 others. 2025. Tulu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu Hong Hoi. 2022. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 21314–21328. Curran Associates, Inc.

Thanh-Long V. Le, Myeongho Jeon, Kim Vu, Viet Dac Lai, and Eunho Yang. 2026. No prompt left behind: Exploiting zero-variance prompts in LLM reinforcement learning via entropy-guided advantage shaping. In *The Fourteenth International Conference on Learning Representations*.

Xuefeng Li, Haoyang Zou, and Pengfei Liu. 2025. Limr: Less is more for rl scaling. *Preprint*, arXiv:2502.11886.

Ziniu Li, Congliang Chen, Tianyun Yang, Tian Ding, Ruoyu Sun, Ge Zhang, Wenhao Huang, and Zhi-Quan Luo. 2026. Knapsack RL: Unlocking exploration of LLMs via optimizing budget allocation. In *Proceedings of the 43rd International Conference on Machine Learning*.

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. 2024. ReMax: A simple, effective, and efficient reinforcement learning method for aligning large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 29128–29163. PMLR.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. Understanding r1-zero-like training: A critical perspective. In *Second Conference on Language Modeling*.

Shichao Ma, Zhiyuan Ma, Ming Yang, Xiaofan Li, Xing Wu, Jintao Du, Yu Cheng, Weiqiang Wang, Qiliang Liu, Zhengyang Zhou, and Yang Wang. 2026. Tspo: Breaking the double homogenization dilemma in multi-turn search policy optimization. *Preprint*, arXiv:2601.22776.

Prasanna Mayilvahanan, Ricardo Dominguez-Olmedo, Thaddäus Wiedemer, and Wieland Brendel. 2025. Math-beyond: A benchmark for rl to expand beyond the base model. *Preprint*, arXiv:2510.11653.

Gongrui Nan, Siye Chen, Jing Huang, Mengyu Lu, Dexun Wang, Chunmei Xie, Weiqi Xiong, Xianzhou Zeng, Qixuan Zhou, Yadong Li, and Xingzhong Xu. 2025. Ngrpo: Negative-enhanced group relative policy optimization. *Preprint*, arXiv:2509.18851.

Hieu Trung Nguyen, Bao Nguyen, Wenao Ma, Yuzhi Zhao, Ruifeng She, and Viet Anh Nguyen. 2026. Adaptive rollout allocation for online reinforcement learning with verifiable rewards. In *The Fourteenth International Conference on Learning Representations*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *EuroSys*, pages 1279–1297.

Joe Suk and Yaqi Duan. 2025. On the optimization dynamics of RLVR: Gradient gap and step size scaling. In *First Workshop on Foundations of Reasoning in Language Models*.

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

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, and 77 others. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *Preprint*, arXiv:2501.12599.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. In *Advances in Neural Information Processing Systems*, volume 38, pages 115452–115486. Curran Associates, Inc.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256.

Fang Wu, Weihao Xuan, Ximing Lu, Mingjie Liu, Yi Dong, Zaid Harchaoui, and Yejin Choi. 2026a. The invisible leash: Why rlvr may or may not escape its origin. *Preprint*, arXiv:2507.14843.

Junkang Wu, Kexin Huang, Jiancan Wu, An Zhang, Xiang Wang, and Xiangnan He. 2026b. Quantile advantage estimation: Stabilizing RLVR for LLM reasoning. In *The Fourteenth International Conference on Learning Representations*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *CoRR*, abs/2409.12122.

Chenghao Yang, Lin Gui, Chenxiao Yang, Victor Veitch, Lizhu Zhang, and Zhuokai Zhao. 2025b. Let it calm: Exploratory annealed decoding for verifiable reinforcement learning. *Preprint*, arXiv:2510.05251.

Chenghao Yang and Ari Holtzman. 2025. Llm probability concentration: How alignment shrinks the generative horizon. *Preprint*, arXiv:2506.17871.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, juncai liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, and 17 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, pages 113222–113244. Curran Associates, Inc.

Jixiao Zhang and Chunsheng Zuo. 2025. GRPO-LEAD: A difficulty-aware reinforcement learning approach for concise mathematical reasoning in language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5642–5654, Suzhou, China. Association for Computational Linguistics.

Zhi Zhang, Zhen Han, Costas Mavromatis, Qi Zhu, Yunyi Zhang, Sheng Guan, Dingmin Wang, Xiong Zhou, Shuai Wang, Soji Adeshina, Vassilis Ioannidis, and Huzefa Rangwala. 2026. Train less, learn more: Adaptive efficient rollout optimization for group-based reinforcement learning. *Preprint*, arXiv:2602.14338.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. 2024. Group sequence policy optimization. *Qwen Team, Alibaba Inc.*

Haizhong Zheng, Yang Zhou, Brian Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. 2025. Act only when it pays: Efficient reinforcement learning for llm reasoning via selective rollouts. In *Advances in Neural Information Processing Systems*, volume 38, pages 124321–124346. Curran Associates, Inc.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. 2025. The surprising effectiveness of negative reinforcement in llm reasoning. In *Advances in Neural Information Processing Systems*, volume 38, pages 126546–126573. Curran Associates, Inc.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, yuchen zhang, Xinwei Long, Ermo Hua, Biqing Qi, Youbang Sun, Zhiyuan Ma, Lifan Yuan, Ning Ding, and Bowen Zhou. 2025. Ttrl: Test-time reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 38, pages 131459–131483. Curran Associates, Inc.

Checklist

A. For every submission

A1. Did you describe the limitations of your work? Yes. The paper includes a dedicated *Limitations* section discussing the current focus on math-oriented RLVR with binary verifiers, the dependence on rollout-level statistics, and the need to evaluate the method across broader domains, model scales, verifier designs, and compute budgets.

A2. Did you discuss any potential risks of your work? No. This work is primarily methodological and studies reinforcement learning for mathematical reasoning under verifiable-reward settings (§2 and §4). It does not introduce new user-facing applications, collect sensitive data, or target high-risk domains. A relevant limitation is that improvements on verifiable mathematical benchmarks may not directly transfer to open-ended or safety-critical reasoning tasks. We therefore present NAPO as a research method and evaluate it only in controlled benchmark settings.

B. Did you use or create scientific artifacts?

B1. Did you cite the creators of artifacts you used? Yes. The paper cites the Qwen2.5-Math-7B model and DAPO17k training data in the training protocol (§5), cites the Math-Beyond benchmark in the controlled setup (§3.1), and cites prior RLVR/GRPO-related methods in §2.

B2. Did you discuss the license or terms for use and/or distribution of any artifacts? Yes. The current draft cites the main artifacts in §3.1, §5, and §2.Our experiments use publicly available datasets and model checkpoints, including Qwen2.5-Math-7B and Llama-3.1-8B-Instruct, under their respective original licenses and terms of use. We use these artifacts strictly for non-commercial research purposes. The training and evaluation code is released anonymously for research use only.

B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use? Yes. The paper uses public mathematical reasoning datasets and pretrained language models for research evaluation and method development, as described in §3.1 and §5.

B4. Did you discuss the steps taken to check whether the data that was collected/used contains information that names or uniquely identifies individual people or offensive content,

and the steps taken to protect/anonymize it? N/A. The work does not introduce newly collected human-subject or personally identifying text data; it uses mathematical reasoning prompts and benchmark evaluations described in §3.1 and §5. If any released processed data are derived from existing datasets, the final artifact documentation should still state that no personal data were intentionally collected.

B5. Did you provide documentation of the artifacts? Yes. The paper documents the training and evaluation artifacts at the level needed for the experiments, including the model, training set, validation benchmarks, rollout budget, decoding temperature, and prompt format in §3.1, §5, Appendix B.1, and Table 3. We provide our code at .

B6. Did you report relevant statistics such as the number of examples and train/test/dev splits? Yes. The paper reports the use of DAPO17k in §5, a fixed 1024-prompt controlled training batch in §3.1, and transition/evaluation denominators in Table 1. The draft should add explicit train/validation/test sizes for every benchmark split used in the experiments where applicable.

C. Did you run computational experiments?

C1. Did you report the number of parameters in the models used, the total computational budget, and computing infrastructure used? Yes. The paper reports the model family/checkpoint, Qwen2.5-Math-7B, in §3.1, §5, and Table 3, and the appendix hyperparameters include GPU count in Table 3. The computing infrastructure is discussed in §B.1.

C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values? Yes. The controlled setup is described in §3.1; the Evaluation section (§5) describes the training protocol, rollout count, decoding temperature, context length, and evaluated datasets; Appendix B.1 and Table 3 provide the prompt template and hyperparameters. The early-fraction choice is discussed in §3.2.1 and evaluated in Figure 5.

C3. Did you report descriptive statistics about your results, and is it transparent whether you are reporting the max, mean, or a single run? Yes. The paper reports Pass@k results in Table 2, TTFP results in Figure 4 and Appendix D, and controlled replay averages in Table 1. The current draft does not consistently provide error bars, confidence intervals, or multi-seed summaries for all main results, so this limitation should be made explicit if additional runs are not available.

C4. If you used existing packages for preprocessing, normalization, or evaluation, did you report implementation, model, and parameter settings used? Yes. The paper reports important model and training settings, including the vLLM rollout setup and prompt template in Appendix B.1 and Table 3, and the NAPO update is specified in Algorithm 1 and Eqs. (7)–(9). The final version should include versioned implementation details for major packages and evaluation scripts if code is released.

D. Did you use human annotators or research with human participants?

D1. Did you report the full text of instructions given to participants? N/A. The work does not use human annotators or human participants.

D2. Did you report information about how you recruited and paid participants? N/A. The work does not recruit or compensate human participants.

D3. Did you discuss whether and how consent was obtained from people whose data you are using/curating? N/A. The work does not collect or curate new data from human participants.

D4. Was the data collection protocol approved or determined exempt by an ethics review board? N/A. The work does not involve human-subject data collection.

D5. Did you report the basic demographic and geographic characteristics of the annotator population? N/A. There is no annotator population in this work.

E. Did you use AI assistants in your research, coding, or writing?

E1. If you used any AI assistants, did you include information about your use? N/A We use LLMs only for writing, editing, organizing codes and formatting assistance. Their use does not affect the core methodology, scientific rigor, experimental results, or originality of the research.

A Related Works

Recent work has shown that reinforcement learning (RL)(Sutton et al., 1998) can effectively improve LLM reasoning through both exploitation and exploration(Auer et al., 2002; Cheng et al., 2026; Chen et al., 2025a). Methods like Reinforcement Learning from Human Feedback (RLHF)(Ouyang et al., 2022), Test-Time Reinforcement Learning (Zuo et al., 2025)and Reinforcement with Verifiable Reward(RLVR) (Guo et al., 2025)have made great process in training Large Reasoning Models(Team et al., 2025). In RLVR, an LLM is trained as a policy that generates a chain-of-thought (CoT) process and receives a deterministic reward based on whether the final answer can be algorithmically verified. On reasoning tasks such as math and coding(Le et al., 2022), this paradigm has led to strong gains with policy-gradient methods such as PPO(Schulman et al., 2017), GRPO(Shao et al., 2024) and REINFORCE(Williams, 1992) based methods(Hu et al., 2025; Li et al., 2024; Ahmadian et al., 2024). In particular, GRPO has become one of the most widely adopted formulations, and many subsequent methods build on it ((Yu et al., 2025; Zheng et al., 2024; Chu et al., 2026; Zhang and Zuo, 2025; Liu et al., 2025; Wu et al., 2026b)). These methods compute advantages from grouped rollout rewards, but on overly hard prompts, training can stall when all sampled outputs are negative, since binary terminal rewards provide little effective learning contrast within fully negative groups ((Ma et al., 2026; Yu et al., 2025)). To better utilize collected rollouts and improve learning efficiency, recent studies mainly address this problem from two directions: reallocating rollout budget toward more learnable prompts, and extracting more informative learning signals from negative samples.

Dynamic rollout allocation. One line of work reduces wasted rollout budget on unpromising prompts by estimating their expected pass rates and allocating computation accordingly. (Nguyen et al., 2026) predicts prompt difficulty from prior and posterior success rates and reallocates rollout budget according to batch-level uncertainty. (Li et al., 2026) estimates the value of additional exploration using policy-gradient-based approximations and formulates rollout allocation as an optimization problem. (Zhang et al., 2026) mitigates vanished groups by using a smaller initial inference budget to test prompts' difficulties and iteratively rescuing prompts that would otherwise receive no effective learning signal.

Learning from negative samples. Another line of work focuses on extracting useful training signals from negative samples rather than discarding them. Zhu et al. (2025) shows that learning from negative samples can increase response diversity and help break the bottlenecks of the base model. Building on this insight, they further applies weighted REINFORCE to bypass the limitations of standard group-relative advantage estimation on fully negative groups. Nan et al. (2025) introduces virtual rewards so that fully negative groups can still provide non-trivial optimization signals. Le et al. (2026) further incorporates token-level entropy into the update magnitude, encouraging exploration while reducing overfitting.

Recently, Fang et al. (2026) further connects rollout allocation with sample-level learning, suggesting that the two directions are complementary. Overall, these works share a common goal: identifying and exploiting the hidden learning value of difficult prompts or negative rollouts. However, existing methods still lack a unified bridge between trajectory-internal signals and prompt-level learnability.

Concretely, we are the first to define the **negative-to-positive** transition as a key behavior that captures a prompt's latent learnability. Building on this perspective, we propose a framework that links observable subsignals to future N→P behavior through a driver, so that negative responses which are likely to become positive can receive predicted advantages. In this way, our framework leverages naturally collected rollouts and learns from structured negatives on hard prompts, while maintaining a balanced advantage structure, as demonstrated in Suk and Duan (2025) and further analyzed in Section 4. Thus, our method takes a step toward addressing the limitation raised in Chen et al. (2025b) and Wu et al. (2026a).

B Experiment Settings

B.1 Chat template

Chat template for QWEN

System: `You are a helpful assistant.`

User: `{input} Please reason step by step, and put your final answer within \boxed{}.`

Assistant: `Model generates a step-by-step solution; the final answer is enclosed in \boxed{}.`

B.2 Hyperparameters setting

Category	Setting
Model	
Model	`Qwen2.5-Math-7B`
Data	
Train batch size	1024
Max prompt length	1024
Max response length	3072
RL Algorithm	
KL coefficient	0.0
Entropy coefficient	0.0
Critic warmup	0
Optimization	
Learning rate	1×10^{-6}
PPO minibatch size	256
Sampling	
top_k	-1
top_p	1
temperature	1.0

Table 3: Training hyperparameters for GRPO-based RLVR with vLLM rollouts.

All experiments are conducted on a single node equipped with eight NVIDIA H100 GPUs. We report final evaluation results using the checkpoint at step 200 for Qwen2.5-Math-7B and the checkpoint at step 100 for Llama-3.1-8B-Instruct.

C Definitions of Batch-Level Driver Statistics

In this appendix, we define the batch-level summary statistics used in Table 4 and Figure 3 to characterize the distribution of rollout-time drivers within the N→P group. For Figure 3 We use kernel density estimation (KDE) only for visualization, to provide a smoothed view of the empirical distribution of the driver scores. KDE is a standard nonparametric density estimation method that places a smooth kernel around each observed sample and aggregates these kernels to estimate a continuous density. It does not affect training, evaluation, or the computation of the proposed driver; all quantitative analyses are performed on the original empirical scores.

Let $\{d_i\}_{i=1}^n$ denote the driver values computed for the rollouts in a given subset (e.g., the N→P rollout group of a batch), where n is the number of rollouts in that subset.

Replay	Training N→P	`msk`	`rt`
n1	120	0.924	0.039
n2	117.67	0.812	0.026

Table 4: Representative batch-level driver statistics for a strong/weak replay comparison. Larger `msk` and `rt` indicate more mass in the high-driver region.

Mean-shift statistic (`msk`). The statistic `msk` summarizes the average magnitude of the driver values within the selected rollout group:

$$\texttt{msk} = \frac{1}{n}\sum_{i=1}^{n} d_i. \tag{10}$$

In our analysis, larger `msk` indicates that the corresponding batch tends to allocate relatively more uncertainty to the early stage of generation.

Right-tail mass statistic (`rt`). The statistic `rt` summarizes the heaviness of the right tail of the driver distribution. Let τ be a fixed upper-tail threshold (we set $\tau = 1.95$ in practice). We define

$$\texttt{rt} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}[d_i > \tau], \tag{11}$$

where $\mathbf{1}[\cdot]$ is the indicator function. Larger `rt` indicates that a larger fraction of rollouts fall into the high-driver region, i.e., the distribution has a heavier right tail.

Interpretation. These statistics are not intended to serve as calibrated predictors of future success or as exact estimates of future reward. In the main text, we use them only to support a weaker but more practically relevant claim: larger driver values, especially a heavier right tail, are enriched in rollout groups associated with stronger N→P tendency. This is the only property required by our method design, where the driver is used as a surrogate ordering signal rather than as a direct estimator of future pass rate.

D Additional results

E Theoretical Derivation: Fisher Information and Negative-Augmented Policy Optimization

In this section, we provide a theoretical justification for the proposed driver-based weighting scheme. We demonstrate that penalizing negative samples with uniform advantage (standard GRPO) wastes

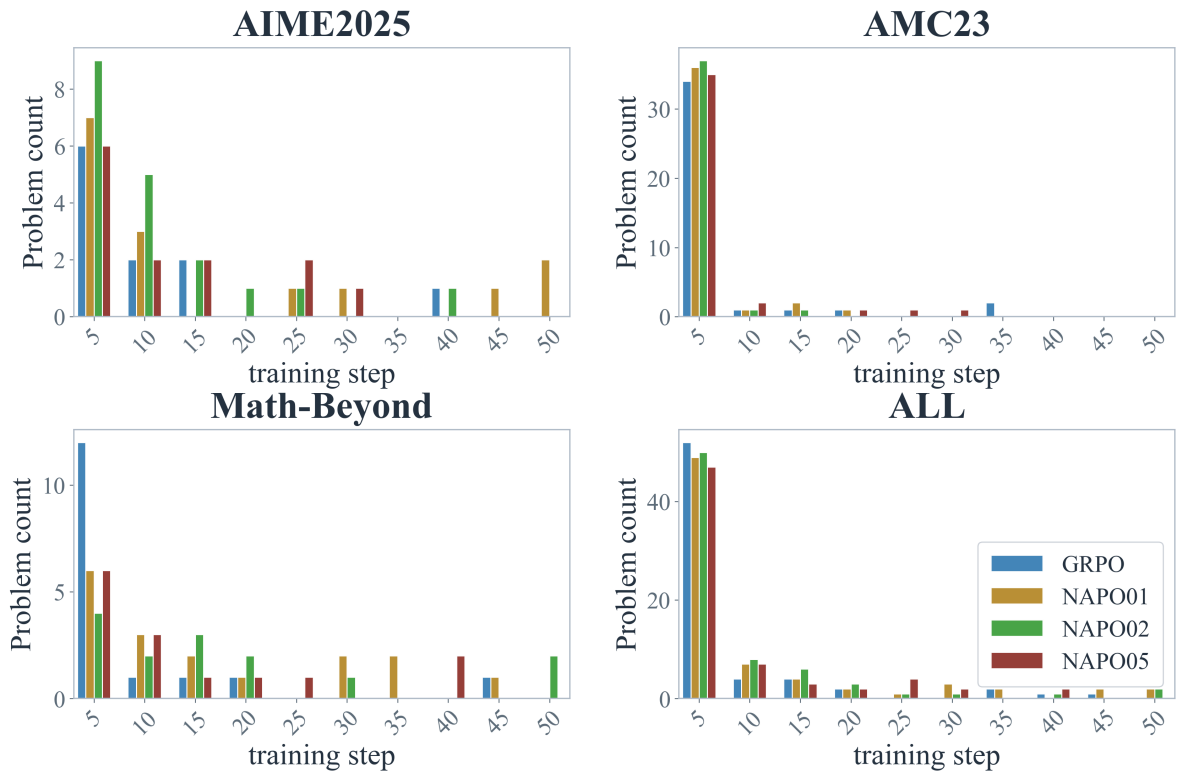


Figure 6: TTFP distribution - all datasets combined

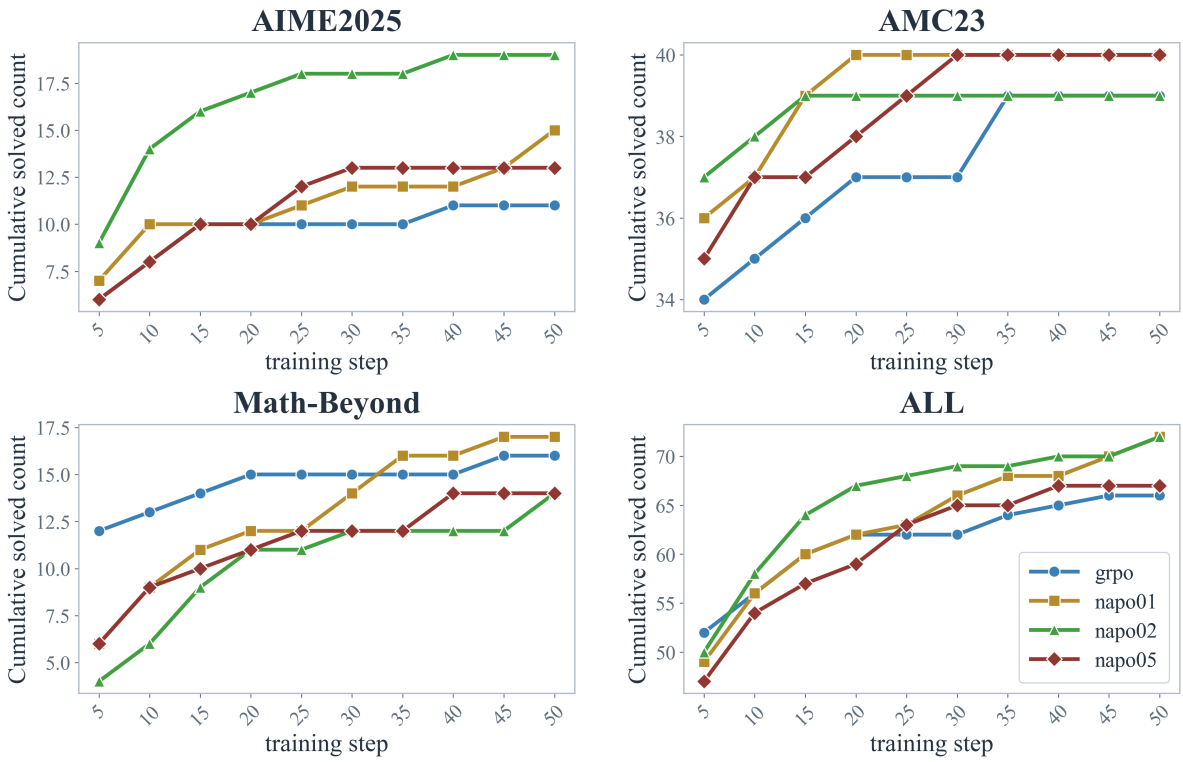


Figure 7: Cumulative solved number: details

Method (LLaMA)	Pass@k								
k	1	2	4	8	16	32	64	128	256
AMC23									
Base	0.1817	0.2718	0.3798	0.5040	0.6303	**0.7489**	**0.8465**	**0.9122**	**0.9500**
GRPO	0.3805	**0.4776**	**0.5518**	**0.6007**	**0.6415**	0.6758	0.7020	0.7312	0.7750
NAPO02	**0.4036**	0.4760	0.5305	0.5731	0.6128	0.6544	0.7016	0.7528	0.8000
AIME2025									
Base	0.00195	0.00390	0.00776	0.01537	0.03017	0.05817	0.10836	**0.18973**	**0.3000**
GRPO	**0.00247**	**0.00492**	**0.00973**	**0.01901**	**0.03630**	**0.06625**	**0.11080**	0.15956	0.2000
NAPO02	0.00052	0.00104	0.00208	0.00417	0.00833	0.01667	0.03333	0.06667	0.1333
AIME24									
Base	0.03177	0.05817	0.09923	0.15312	**0.21399**	**0.28016**	**0.34512**	**0.40348**	**0.46667**
GRPO	**0.07187**	**0.10633**	**0.13924**	**0.16717**	0.19260	0.21620	0.23945	0.27503	0.33333
NAPO02	0.04609	0.06417	0.08776	0.11927	0.15662	0.20341	0.26284	0.31524	0.33333
Math-Beyond									
Base	0.02132	0.03654	0.05716	0.08174	**0.11030**	**0.14436**	**0.18454**	**0.23267**	**0.29282**
GRPO	0.04821	0.06297	0.07633	0.08858	0.10032	0.11343	0.12962	0.14834	0.16575
NAPO02	**0.05792**	**0.07203**	**0.08411**	**0.09577**	0.10926	0.12481	0.14162	0.15875	0.17680

Table 5: Pass@k comparison across datasets for **Llama-3.1-8B-Instruct** (Chen et al., 2021) models at temperature $T = 1.0$. The row corresponding to NAPO02 is highlighted in gray, and the best value between GRPO and NAPO02 within each dataset block and column is shown in bold. NAPO02 mitigates the mode-collapse issue observed in GRPO, although its performance degrades at larger values of k, likely due to model limitations. Due to limited computational resources, we evaluate the checkpoint at step 100.

optimization budget on samples that provide negligible Fisher Information. We then derive Negative-Augmented Policy Optimization (NAPO), where the advantage is proportional to the Fisher Information of the trajectory, effectively maximizing the information gain of the policy update.

E.1 Fisher Information of Negative Samples

Consider a prompt x with a rollout group $\mathcal{G}^- = \{y_i\}_{i=1}^G$ where all rollouts are negative, i.e., $r(x, y_i) = -1$. In policy optimization, the Fisher Information Matrix (FIM) $\mathbf{F}(\theta)$ measures the curvature of the KL divergence and the informativeness of a sample gradient. For a specific trajectory y, the contribution to the FIM is given by the outer product of the score function:

$$\mathbf{F}_y(\theta) = \nabla_\theta \log \pi_\theta(y|x) \, \nabla_\theta \log \pi_\theta(y|x)^\top. \quad (32)$$

A sample y is considered informative for optimization if its gradient vector has a large magnitude in the parameter space, indicating a significant potential to reshape the policy distribution.

We analyze the Fisher Information at the token level. Let $\mathbf{p}_t = \pi_\theta(\cdot|x, y_{<t})$ be the probability vector over the vocabulary V at step t. Let $\mathbf{F}_t$ denote the Fisher Information contribution from position t.

Lemma E.1 (Fisher Information and Entropy)**.** *Let $H_t = -\sum_k p_k \log p_k$ be the entropy of the distribution at step t. The trace of the token-level Fisher Information matrix satisfies:*

$$\mathrm{Tr}(\mathbf{F}_t) = 1 - \sum_{k \in V} p_k^2. \quad (33)$$

Furthermore, for a fixed vocabulary size V, the quantity $1 - \sum p_k^2$ and the entropy H_t are positively associated: both are maximized by the uniform distribution (at $\frac{V-1}{V}$ and $\log V$, respectively) and both equal zero only for deterministic distributions. In particular, $H_t \to 0$ implies $\mathrm{Tr}(\mathbf{F}_t) \to 0$.

Proof. The score function for a sampled token y_t is $\nabla_\theta \log \pi_\theta(y_t|x, y_{<t})$. Under the common softmax parameterization $p_k = \frac{e^{z_k}}{\sum e^{z_k}}$, the gradient of the log-probability with respect to the logits $\mathbf{z}$ is:

$$\nabla_{\mathbf{z}} \log p(y_t) = \mathbf{e}_{y_t} - \mathbf{p}_t, \quad (34)$$

where $\mathbf{e}_{y_t}$ is the one-hot vector for the sampled token. The Fisher Information matrix is the covariance of this gradient:

$$\mathbf{F}_t = \mathbb{E}_{y_t}\left[(\mathbf{e}_{y_t} - \mathbf{p}_t)(\mathbf{e}_{y_t} - \mathbf{p}_t)^\top\right] \quad (35)$$

$$= \mathbb{E}[\mathbf{e}_{y_t}\mathbf{e}_{y_t}^\top] - \mathbf{p}_t\mathbf{p}_t^\top. \quad (36)$$

Since $\mathbb{E}[\mathbf{e}_{y_t}\mathbf{e}_{y_t}^\top] = \mathrm{diag}(\mathbf{p}_t)$, the trace of the FIM is:

$$\mathrm{Tr}(\mathbf{F}_t) = \mathrm{Tr}\big(\mathrm{diag}(\mathbf{p}_t)\big) - \mathrm{Tr}(\mathbf{p}_t\mathbf{p}_t^\top) \quad (37)$$

$$= \sum_k p_k - \sum_k p_k^2 = 1 - \sum_k p_k^2. \quad (38)$$

The quantity $G := 1 - \sum p_k^2$ is the Gini impurity (equivalently, the Tsallis entropy of order 2). Both G and the Shannon entropy $H = -\sum p_k \log p_k$ are *Schur-concave* functions of the probability vector: they are both uniquely maximized by the uniform distribution and both equal zero if and only if the distribution is deterministic. While G and H are not in general strict monotonic functions of each other for $V \geq 3$, they are tightly coupled via the Rényi entropy ordering $H_1 \geq H_2$, which gives:

$$1 - \sum_k p_k^2 \;\leq\; 1 - e^{-H}.$$

In particular, $H \to 0$ forces $\mathrm{Tr}(\mathbf{F}_t) \to 0$, and conversely the uniform distribution uniquely maximizes both quantities. Hence, higher-entropy distributions correspond, in a strong qualitative sense, to larger local Fisher information. □

E.2 Analysis of Standard NSR on Unsolved Groups

Standard NSR assigns uniform negative advantages $A(y) \approx -1$ to all samples in an unsolved group. From the perspective of Fisher Information, this implies that the optimization update treats samples with zero information (confident errors) and high information (uncertain errors) equally.

Proposition E.2 (Gradient Saturation and Fisher Information)**.** *Consider a negative sample y generated from a low-entropy (confident) policy state.*

1. ***Zero Information:*** *If $H_t \to 0$, then $\mathrm{Tr}(\mathbf{F}_t) \to 0$. The gradient $\nabla_\theta \log \pi_\theta(y_t)$ vanishes.*

2. ***Saddle Points:*** *Negative samples generated with high confidence correspond to "saddle points" or "dead ends" in the likelihood landscape. Uniform penalization ($A = -1$) attempts to push probability mass away from these points, but the vanishing Fisher Information implies that $\Delta\theta \approx 0$ regardless of the advantage magnitude.*

Thus, uniform advantage assignment wastes gradient budget on samples that lie in low-information regions of the parameter space.

Proof. The gradient vector for the sampled token y_t is $\mathbf{g}_t = \mathbf{e}_{y_t} - \mathbf{p}_t$, where $\mathbf{e}_{y_t}$ is the one-hot encoding of the sampled token and $\mathbf{p}_t$ is the probability vector $\pi_\theta(\cdot|x, y_{<t})$. The squared norm of this gradient is:

$$\|\mathbf{g}_t\|_2^2 = \|\mathbf{e}_{y_t} - \mathbf{p}_t\|_2^2 \quad (39)$$

$$= (1 - \pi(y_t))^2 + \sum_{v \neq y_t} \pi(v)^2. \quad (40)$$

As $\pi(y_t) \to 1$ (confidence approaches certainty, entropy $H_t \to 0$), the first term approaches 0. Simultaneously, the mass of the probability vector concentrates on y_t, forcing $\pi(v) \to 0$ for all other v. Thus, $\|\mathbf{g}_t\|_2 \to 0$.

This implies that confidently generated tokens (which characterize the "dead-end" negative rollouts in Section 3) contribute negligible gradient signal. Uniformly weighting such samples wastes optimization budget on saturated gradients that cannot effectively redirect the policy. □

E.3 Derivation of NAPO

To maximize the efficiency of the update, we should allocate the "negative learning budget" to the samples whose penalization best reduces the total likelihood of failed trajectories. We formalize this as a constrained optimization over per-rollout weights and show that the solution naturally connects to Fisher Information and, through it, to trajectory entropy.

Exact formulation via the Gram matrix. Let $\mathcal{G}^- = \{y_i\}_{i=1}^G$ be a fully negative rollout group for prompt x. Write $g_y = \nabla_\theta \log \pi_\theta(y \mid x)$ for the trajectory-level score vector, and define the *Gram matrix* $\mathbf{G} \in \mathbb{R}^{G\times G}$ with entries

$$G_{ij} = g_{y_i}^\top g_{y_j}. \quad (43)$$

Given a weight vector $w = (w_1, \ldots, w_G)^\top$, the parameter update is $\delta = -\eta \sum_i w_i g_{y_i}$. The total first-order reduction in the log-likelihood of every negative rollout in the group is

$$R(w) \;=\; -\sum_{j=1}^{G} g_{y_j}^\top \delta \;=\; \eta\, \mathbf{1}^\top \mathbf{G}\, w. \quad (44)$$

We seek the weight vector that maximizes $R(w)$ subject to a variance budget $\|w\|^2 \leq B$ (which controls the variance of the stochastic gradient and ensures stable updates):

$$\max_w \; \mathbf{1}^\top \mathbf{G}\, w \qquad \text{s.t.} \qquad \|w\|^2 \leq B. \quad (45)$$

By the Cauchy–Schwarz inequality, $\mathbf{1}^\top \mathbf{G}\, w \le \|\mathbf{G}^\top \mathbf{1}\|\, \|w\|$, with equality when $w \propto \mathbf{G}^\top \mathbf{1}$. Since $\mathbf{G}$ is symmetric, the exact solution is

$$w^* \propto \mathbf{G}\,\mathbf{1}, \qquad w^*(y_i) \propto \big[\mathbf{G}\,\mathbf{1}\big]_i = \sum_{j=1}^{G} g_{y_i}^\top g_{y_j}. \tag{46}$$

Computing $\mathbf{G}\,\mathbf{1}$ exactly is impractical (it requires full parameter-space inner products between all pairs of rollout gradients in a multibillion-parameter model). We therefore seek a *tractable diagonal proxy* and state a structural condition under which the ranking induced by $\mathbf{G}\,\mathbf{1}$ coincides with the ranking induced by the diagonal alone.

Uniform-correlation condition. We decompose each entry of $\mathbf{G}\,\mathbf{1}$:

$$\big[\mathbf{G}\,\mathbf{1}\big]_i = \underbrace{\|g_{y_i}\|^2}_{=\,\mathrm{Tr}(\mathbf{F}_{y_i})} + \sum_{j\neq i} g_{y_i}^\top g_{y_j}.$$

The diagonal term is exactly the trajectory-level Fisher Information; the off-diagonal sum captures cross-sample interference. Directly computing the latter is as expensive as the full Gram matrix. We therefore introduce the following structural assumption.

Assumption E.3 (Uniform pairwise correlation)**.** There exists a constant $c \ge 0$ such that, for all distinct rollouts $y_i, y_j \in \mathcal{G}^-$,

$$g_{y_i}^\top g_{y_j} \;\approx\; c\,\|g_{y_i}\|\,\|g_{y_j}\|.$$

It states that *all same-prompt rollout gradients are roughly equally correlated with each other*—a plausible condition because they share the same prompt, model architecture, and sampling distribution, so no single pair of rollouts is systematically more aligned than any other. The discriminating factor across rollouts is then their *gradient magnitude*, not their mutual orientation.

Theorem E.4 (Optimality of Fisher-Weighted Advantage)**.** *Under Assumption E.3, the optimal weight $w^*(y)$ from Eq. (46) is a monotonically increasing function of $\mathrm{Tr}(\mathbf{F}_y) = \|g_y\|^2$. By Lemma E.1 and the token-independence approximation, this quantity is positively associated with the trajectory entropy $\sum_t H_t$. Therefore, we define the* Entropy-Weighted Advantage*:*

$$A_{\text{NAPO}}(y) = -\frac{d(y)}{\frac{1}{|\mathcal{G}|}\sum_{y'\in\mathcal{G}} d(y')}, \tag{47}$$

where $d(y)$ is the driver score (normalized entropy) defined in Eq. (6) of the main text.

Proof. **Step 1: Exact solution (Gram matrix).** The constrained optimum of Eq. (45) is $w^* \propto \mathbf{G}\,\mathbf{1}$ as derived in Eq. (46).

Step 2: Reduction under uniform correlation. Let $S = \sum_{j=1}^{G} \|g_{y_j}\|$ denote the total gradient norm across the group. Under Assumption E.3:

$$\begin{aligned}\big[\mathbf{G}\,\mathbf{1}\big]_i &= \|g_{y_i}\|^2 + c\,\|g_{y_i}\| \sum_{j\neq i} \|g_{y_j}\| \\ &= \|g_{y_i}\|^2 + c\,\|g_{y_i}\|\big(S - \|g_{y_i}\|\big) \\ &= (1-c)\,\|g_{y_i}\|^2 + c\,S\,\|g_{y_i}\|. \end{aligned} \tag{48}$$

Define $f(u) = (1-c)\,u^2 + c\,S\,u$ for $u \ge 0$. Its derivative is $f'(u) = 2(1-c)\,u + c\,S$. Since $0 \le c \le 1$ (cosine similarity is bounded by one) and $S \ge 0$, we have $f'(u) \ge 0$ for all $u \ge 0$. Hence f is monotonically increasing in $u = \|g_{y_i}\|$, and therefore also in $\|g_{y_i}\|^2 = \mathrm{Tr}(\mathbf{F}_{y_i})$.

The ranking of $w^*(y)$ across the group thus coincides with the ranking of $\mathrm{Tr}(\mathbf{F}_y)$.

Step 3: Connection to trajectory entropy. Since $g_y = \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(y_t|x, y_{<t})$, under the approximation that token-level score vectors are approximately uncorrelated across positions we have:

$$\|g_y\|^2 \;\approx\; \sum_{t=1}^{T} \big\|\nabla_\theta \log \pi_\theta(y_t|x, y_{<t})\big\|^2.$$

Taking the expectation over the sampled token at each position and applying Lemma E.1:

$$\mathbb{E}\big[\|g_y\|^2\big] \;\approx\; \sum_{t=1}^{T} \mathrm{Tr}(\mathbf{F}_t) = \sum_{t=1}^{T}\Big(1 - \sum_k p_{t,k}^2\Big),$$

which is positively associated with $\sum_t H_t$ by Lemma E.1.

The driver $d(y)$ in Eq. (6) of the main text is a normalized measure of early-stage entropy and thus serves as a tractable proxy for $\mathrm{Tr}(\mathbf{F}_y)$. The advantage assignment A_{NAPO} therefore implements an approximation to the optimal Fisher-weighted update, prioritizing samples where the policy is "plastic" (high entropy) over samples where it is rigid (low entropy). □

Remark E.5 (Interpretation as a diagonal preconditioner). The uniform-correlation reduction plays the role of a *diagonal preconditioner* for the Gram matrix. Just as diagonal preconditioning of a full

Newton step retains the correct coordinate-wise scaling while ignoring off-diagonal curvature, ranking by $\mathrm{Tr}(\mathbf{F}_y)$ retains the correct per-rollout scaling while treating cross-rollout interactions as uniform. When the uniform-correlation assumption is violated (e.g., if two rollouts share unusually long common prefixes), the proxy may over- or under-weight individual rollouts, but the overall ranking remains a principled first-order approximation.

Remark E.6 (Relation to orthogonality). Setting $c = 0$ recovers the special case of orthogonal score vectors, for which $w^*(y) \propto \|g_y\|^2$ exactly. However, same-prompt rollouts typically share early tokens and pass through shared model parameters, making $c > 0$ the realistic regime. The uniform-correlation condition accommodates this shared structure while still yielding a tractable ranking criterion.

E.4 The NAPO Algorithm

The resulting algorithm replaces the discrete selection in Algorithm 1 with the continuous weighting in Eq. (47). The objective function for an unsolved group is:

$$\mathcal{L}_{\text{NAPO}}(\theta) = \sum_{y \in \mathcal{G}} \sum_{t=1}^{T} A_{\text{NAPO}}(y) \, \log \pi_\theta(y_t | x, y_{<t}). \tag{49}$$

This formulation ensures that negative rollouts exhibiting higher Fisher Information (latent learning potential) receive proportionally stronger penalization signals, thereby efficiently driving exploration towards the $\mathrm{N} \to \mathrm{P}$ boundary without wasting gradient budget on saturated "dead-end" samples.