# FROM REWARDING WHAT MODELS SAY TO HOW THEY THINK: ATTENTION PROCESS REWARD MODEL FOR LLM REINFORCEMENT LEARNING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Reinforcement learning (RL) has become an important component in large language models (LLMs). However, existing RL approaches focus on final outputs or textual solution trajectories, leading to coarse-grained rewards and unreliable reasoning. Such paradigms focus mainly on what the model outputs while overlooking how the model internally reasons. Therefore, can we reward how a model thinks internally rather than merely what it says? We first reveal the problem of textual reward aliasing in existing reward modeling approaches. Motivated by this, we propose the **A**ttention **P**rocess **R**eward **M**odel (APRM), which assigns internal rewards based on hidden states and attention maps to capture high-quality reasoning patterns. With only approximately 46M parameters, APRM is trained using ranking and counterfactual objectives, and its reward guides RL optimization while being periodically updated to handle distribution shifts. Integrated with representative RL methods, APRM achieved an average pass@8 gain of 2.88%. Our results demonstrate the promise of incorporating internal reasoning signals into RL for LLMs. More details in anonymous repository here.

## 1 INTRODUCTION

In recent years, large language models (LLMs) Zhao et al. (2023); Chang et al. (2024); Vaswani et al. (2017) have demonstrated remarkable capabilities in complex reasoning tasks Wang et al. (2024); Plaat et al. (2025). Reinforcement learning (RL) has consequently become an important approach for improving reasoning ability during the post-training stage Cao et al. (2024); Zhang et al. (2025).

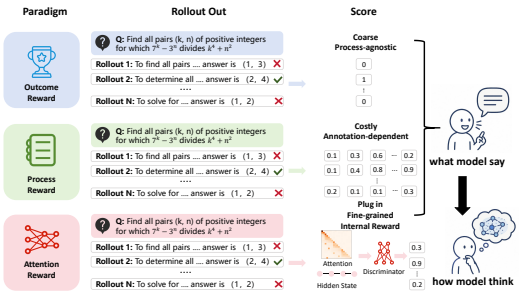


Figure 1: Comparison of outcome rewards, textual process rewards, and APRM.

Existing RL approaches for reasoning typically employ outcome reward models (ORMs) Zhong et al. (2025) and process reward models (PRMs) Xi et al. (2026); Gao et al. (2024). ORMs reward final-answer correctness, which is scalable but provides only sparse outcome-level feedback and cannot identify which reasoning steps contribute to success or failure. PRMs provide denser supervision by scoring intermediate Chain-of-Thought (CoT) steps, but they rely on natural-language rationales that are costly to annotate. Moreover, existing reward models supervise only externalized reasoning traces. However, reasoning in LLMs is fundamentally latent Chen et al. (2025); Barez et al. (2025). The internal computation that produces a textual trajectory is not directly observable from the trajectory itself.



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.

Therefore, textual reward supervision may suffer from process aliasing, where distinct internal reasoning processes induce indistinguishable textual outputs and consequently receive identical rewards. Therefore, a question arises:

*Can we reward how a model thinks internally, rather than merely what it says?*

Mechanistic interpretability studies reveal structured information routing and interpretable computations in attention components Lan et al. (2024); Merullo et al. (2024); Kissane et al. (2024); Nam et al. (2026), so we use attention-derived signals as a partial window into internal computation, extending reward modeling from *what models say* toward *how they think*. We first theoretically show that textual reward supervision is fundamentally under-observed. Distinct internal reasoning processes may collapse to indistinguishable textual trajectories and consequently receive identical rewards, a phenomenon we term **textual reward aliasing**. Motivated by this view, we propose the Attention Process Reward Model (APRM), as shown in Figure 1. The APRM is built upon an Attention-Fused Concept Pooler and an Edge-Biased Graph Attention Network (GAT) Veličković et al. (2017). APRM directly evaluates hidden states and attention maps from the reasoning process to produce internal process rewards. We use APRM in two stages. During cold-start, APRM uses positive and negative reasoning trajectories with ranking and counterfactual losses to learn preferences over internal reasoning processes. During RL, APRM reshapes the task reward so that policy optimization is guided by both outcomes and internal attention quality.

Through systematic analysis of APRM, we obtain three key findings: (1) Positive and negative trajectories exhibit systematic structural differences in the APRM concept-attention space. (2) APRM captures meaningful attention-derived signals and is sensitive to counterfactual attention perturbations. (3) APRM is not a proxy for existing metrics. It exhibits length-decoupling behavior and captures complementary process-quality information beyond existing reward supervision.

We further validate APRM by integrating it with representative RL methods. As a lightweight plug-in module with only approximately 46M parameters, APRM consistently improves avg@8 and pass@8, yielding an average pass@8 gain of 2.88%. Interestingly, APRM also encourages more efficient reasoning behaviors, with a tendency toward shorter reasoning trajectories while maintaining or improving performance. Our contributions are summarized as follows:

- We reveal the phenomenon of the textual reward aliasing problem and demonstrate the information gain of using internal signals as rewards.
- We propose APRM, a lightweight reward model with approximately 46M parameters that evaluates reasoning trajectories using attention-derived internal signals.
- We systematically analyze the properties of APRM and further validate its effectiveness on downstream reinforcement learning tasks.

## 2 Motivation and Theoretical View of APRM

**Proposition 1** ***(Textual Reward Aliasing)***. *In self-attention, representations are computed as* $\mathbf{C} = \mathbf{AV}$*, where* $\mathbf{A}$ *is the attention matrix and* $\mathbf{V}$ *is the value matrix. When* $\mathrm{rank}([\mathbf{V}, \mathbf{1}]) < T$*, where* $T$ *is the sequence length. There exist distinct attention matrices* $\mathbf{A} \neq \mathbf{A}'$ *such that* $\mathbf{AV} = \mathbf{A}'\mathbf{V}$.

Proposition 1 implies that distinct internal reasoning processes may induce indistinguishable textual trajectories and therefore receive identical rewards under outcome-level or textual supervision. In long-chain CoT scenarios, $T$ typically far exceeds $V$. We refer to this phenomenon as **textual reward aliasing**. Existing ORMs and textual PRMs only observe external behavioral signals $B = b(X, Y)$, where $X$ denotes the input prompt and $Y$ denotes the generated trajectory. In contrast, APRM additionally incorporates latent

reasoning states $Z = (\mathbf{H}, \mathbf{A})$, consisting of hidden states $\mathbf{H}$ and attention dynamics $\mathbf{A}$, thereby expanding the observable space of reward modeling. Full proofs are provided in Appendix F.

**Proposition 2** ***(Non-redundant Internal Supervision)***. *Let $Q$ denote process quality. Under log-loss, APRM provides potential non-redundant information gain: $I(Q; Z \mid X, B) > 0$.*

Proposition 2 suggests that internal attention structures contain process-quality information beyond textual outputs and existing reward signals. We empirically validate this condition in Section 4.2.2 through a residual information test. Full proofs are provided in Appendix F.

Proposition 1 shows the existence of textual reward aliasing, and Proposition 2 further shows that the lost information contains reasoning-relevant signals. Based on these findings, we propose APRM to capture this previously overlooked yet informative signal.

# 3 METHODS

## 3.1 APRM ARCHITECTURE

As shown in Figure 2, we propose APRM as an internal process reward model for Transformer-based LLMs. Given the hidden states and attention maps produced during generation, APRM estimates a scalar score that reflects the quality of internal representations.

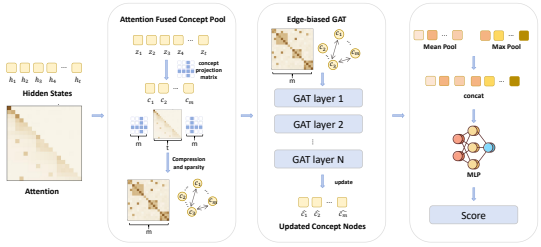


Figure 2: Architecture of APRM.

### 3.1.1 ATTENTION-FUSED CONCEPT POOLING

Reasoning structure is often distributed across tokens and attention interactions. APRM therefore compresses them into latent reasoning concepts, enabling structured process-level modeling while reducing memory overhead. For each rollout sample, APRM uses the final-layer hidden states $\mathbf{H} \in \mathbb{R}^{T \times d}$ and constructs an average attention over heads and layers from the last $K$ layers:

$$\mathbf{A} = \frac{1}{K} \sum_{\ell=L-K+1}^{L} \left( \frac{1}{N_h} \sum_{h=1}^{N_h} \mathbf{A}^{(\ell,h)} \right) \in \mathbb{R}^{T \times T},$$

where $N_h$ denotes the number of attention heads.

Then we construct attention-aware token representations by fusing each token representation with its attention-weighted context:

$$\mathbf{Z} = \mathrm{LayerNorm}\big(\mathbf{H} + \mathrm{MLP}([\mathbf{H}; \mathbf{AH}])\big).$$

To obtain latent reasoning concepts, APRM first uses a learnable projection matrix $\mathbf{W}c \in \mathbb{R}^{d \times M}$ to compute concept-assignment logits and applies softmax over the token dimension, followed by weighted pooling to derive the concept representations:

$$\mathbf{W} = \mathrm{Softmax}T\left((\mathbf{ZW}_c)^\top\right) \in \mathbb{R}^{M \times T}, \qquad \mathbf{C} = \mathbf{WZ} \in \mathbb{R}^{M \times d}.$$

Based on the same assignment matrix, APRM projects the token-level attention summary into the concept space and constructs a sparse normalized concept graph:

$$\hat{\mathbf{A}}_c = \mathrm{TopKNorm}\left(\mathbf{WAW}^\top\right) \in \mathbb{R}^{M \times M},$$

where $\mathrm{TopKNorm}(\cdot)$ denotes row-wise top-$k$ sparsification followed by row normalization.

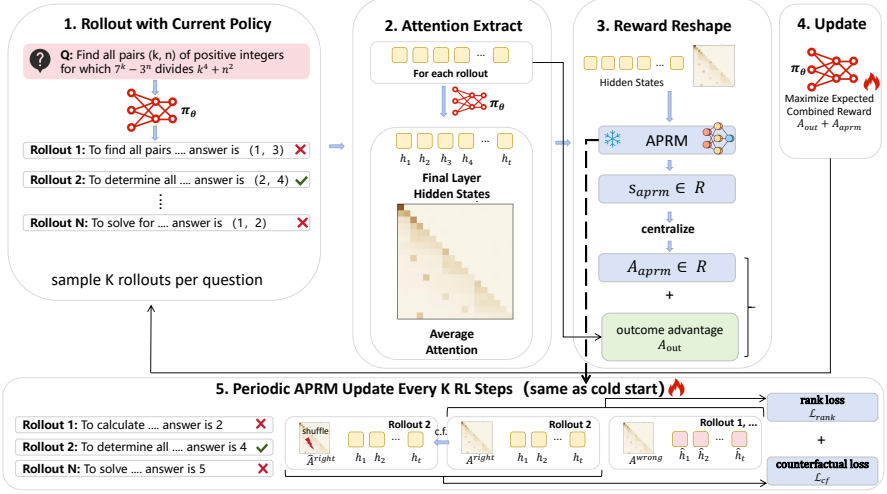


Figure 3: APRM-enhanced reinforcement learning workflow.APRM computes an internal process advantage from its reward scores and combines it with the outcome-based advantage, while being periodically refreshed to mitigate distribution shift.

### 3.1.2 Edge-Biased Graph Attention Network

To enhance APRM's ability to represent the model's internal reasoning process and better capture the interactions between attention patterns and hidden states, we propose the Edge-Biased Graph Attention Network. Given the concept representations $\mathbf{C}$ and the sparse concept graph $\hat{\mathbf{A}}_c$, APRM applies stacked Edge-Biased GAT layers to model dependencies among latent reasoning concepts. For each concept edge, $\hat{\mathbf{A}}_{c,ij}$ is used both as an additive bias in the attention logits and as a multiplicative gate on the transmitted message:

$$\alpha_{ij} = \mathrm{Softmax}_j\left(\frac{\mathbf{Q}_i\mathbf{K}_j^\top}{\sqrt{d_h}} + b(\hat{\mathbf{A}}_{c,ij})\right), \mathbf{m}_{ij} = \alpha_{ij}\, g(\hat{\mathbf{A}}_{c,ij})\, \mathbf{V}_j.$$

Here, $b(\cdot)$ and $g(\cdot)$ denote the edge bias and edge gate functions, respectively.

After graph propagation, APRM applies mean pooling and max pooling over concept nodes. The pooled representations are concatenated and fed into an MLP to produce the final internal process score:

$$s_{\mathrm{APRM}} = \mathcal{S}_\phi(\mathbf{H}, \mathbf{A}).$$

Due to space limitations, detailed layer-wise operations are provided in Appendix C.2.

## 3.2 Cold-start Training of APRM

To train APRM, we first construct a rollout dataset using the backbone LLM. For each question $x$, the model samples multiple reasoning trajectories: $\{y_i\}_{i=1}^N \sim \pi_{\theta_0}(\cdot \mid x)$, where sampling is performed with temperature and top-$p$ sampling. Each trajectory is labeled by an answer verifier: $r_i^{\mathrm{out}} \in \{0, 1\}$, where $r_i^{\mathrm{out}} = 1$ indicates that the trajectory reaches the correct final answer. For each trajectory, we extract internal features $(\mathbf{H}_i, \mathbf{A}_i)$ from the backbone LLM and use them to train APRM. Thus, the APRM training dataset consists of tuples $(x, y_i, \mathbf{H}_i, \mathbf{A}_i, r_i^{\mathrm{out}})$.

**Ranking Loss.** Given a correct trajectory $y^+$ and an incorrect trajectory $y^-$ for the same question, APRM is trained to assign a higher internal score to the correct trajectory. The pairwise ranking loss is

$$\mathcal{L}_{\text{rank}} = \max\big(0, m_{rk} - (s^+ - s^-)\big),$$

where $s^+ = \mathcal{S}_\phi(\mathbf{H}^+, \mathbf{A}^+)$, $s^- = \mathcal{S}_\phi(\mathbf{H}^-, \mathbf{A}^-)$, and $m_{rk}$ is the ranking margin.

**Counterfactual Loss.** To encourage APRM to capture informative attention structures, we construct counterfactual attention patterns from positive trajectories. Specifically, we perturb the original attention map using token-level dropout, token permutation, and Gaussian noise. Given the original attention map $\mathbf{A}$ and the perturbed counterpart $\tilde{\mathbf{A}}$, APRM is encouraged to assign a higher score to the original internal process:

$$\mathcal{L}_{\text{cf}} = \max\left(0, m_{\text{cf}} - \left[\mathcal{S}_\phi(\mathbf{H}, \mathbf{A}) - \mathcal{S}_\phi(\mathbf{H}, \tilde{\mathbf{A}})\right]\right),$$

where $m_{\text{cf}}$ is the counterfactual margin.

Finally, the overall cold-start objective is

$$\mathcal{L}_{\text{APRM}} = \lambda_{\text{rank}}\mathcal{L}_{\text{rank}} + \lambda_{\text{cf}}\mathcal{L}_{\text{cf}}.$$

### 3.3 APRM-enhanced Reinforcement Learning

As shown in Figure 3, during reinforcement learning, APRM provides an additional internal advantage signal. For each rollout group $\{y_i\}_{i=1}^{G}$, the base RL algorithm first computes its original token-level advantage $A_{i,t}^{\text{base}}$. APRM assigns an internal score to each trajectory and incorporates its group-centered value into the policy advantage:

$$A_{i,t} = A_{i,t}^{\text{base}} + \beta\, \mathbb{I}_{\text{active}}(x) \left( \mathcal{S}_\phi(\mathbf{H}_i, \mathbf{A}_i) - \frac{1}{G}\sum_{j=1}^{G} \mathcal{S}_\phi(\mathbf{H}_j, \mathbf{A}_j) \right),$$

where $\beta$ controls the contribution of the model-internal signal. In this way, APRM can be seamlessly integrated with mainstream RL algorithms such as GRPO Shao et al. (2024). To handle the drift of attention distributions during policy training, we periodically update APRM using newly generated rollouts from the latest policy. The detailed algorithmic process is provided in Appendix E.

## 4 Experiments

### 4.1 Experimental Setup

**Datasets.** We conduct experiments on four widely used reasoning benchmarks, including AIME2025, AMC, MATH500 Lightman et al. (2024), and OlympiadBench He et al. (2024). For training, we construct a unified dataset of 10K samples by randomly mixing problems from DAPO-17k Yu et al. (2026) and MATH level 3 to 5 Hendrycks et al. (2021). To reduce the impact of randomness, we report both average@8 and pass@8 as the primary evaluation metrics.

**Baselines.** We compare our method against several representative RL and PRM methods, including GRPO Shao et al. (2024), GPG Chu et al. (2025), GMPO Zhao et al. (2025), GSPO Zheng et al. (2025a), and FAPO Ding et al. (2025). We adopt Qwen3-1.7b-Base and Qwen3-4b-Base Yang et al. (2025) as the backbone models for all methods. For the APRM training, we set $\lambda_{\text{rank}} = 1.0$, $\lambda_{\text{cf}} = 0.05$, $m_{\text{rk}} = m_{\text{cf}} = 0.2$. We adopt the Verl framework to RL, and the maximum prompt length is 512 tokens. For Qwen3-1.7b-Base, the maximum response length is set to 4096 with $\beta = 0.3$. For Qwen3-4b-Base, the maximum response length is set to 3072 with $\beta = 0.1$. In addition, APRM is lightweight, introducing only about 46M additional parameters. More details about hyperparameters are shown in Appendix C.

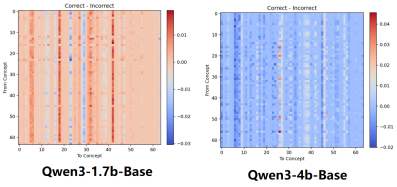


Figure 4: Concept-level representation differences between correct and incorrect reasoning trajectories.

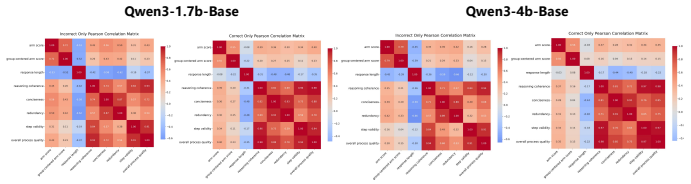


Figure 5: Stratified Pearson correlations between APRM scores and proxy quality signals within correct and incorrect samples on Qwen3-1.7b-Base and Qwen3-4b-Base backbones.

### 4.2 Evaluations and Analysis of APRM

To assess whether APRM captures attention-derived process information, we randomly shuffle attention summaries while keeping hidden states fixed. As shown in Table 1, shuffling reduces ranking accuracy from 78.55% to 62.54% for Qwen3-1.7b-Base and from 85.33% to 75.72% for Qwen3-4b-Base, indicating that APRM uses attention structure to rank reasoning trajectories rather than relying solely on hidden representations. Removing the counterfactual loss $\mathcal{L}_{\text{cf}}$ substantially reduces attention sensitivity and ranking accuracy, highlighting its importance in encouraging attention-derived reasoning signals. Further training details and ablations on latent concept counts and GAT layers are provided in Appendix C.3. Figure 4 compares mean concept graphs $\hat{\mathbf{A}}_c$ from 600 trajectories, produced by the attention-fused concept pooling module. Correct and incorrect trajectories exhibit distinct concept-transition patterns in both models: correct Qwen3-1.7b-Base samples activate a broader range of transitions more strongly, while Qwen3-4b-Base shows a more selective redistribution of transition weights. These findings suggest systematic structural differences between positive and negative trajectories in APRM's concept-attention space.

Table 1: Attention shuffle robustness analysis of APRM. Columns starting with 'S-' indicate results after attention shuffling.

| Model | Acc | S-Acc | Score | S-Score |
|---|---|---|---|---|
| Qwen3-1.7b-Base | 78.55% | 62.54% | 0.4984 | -0.9645 |
| w/o $\mathcal{L}_{\text{cf}}$ | 76.43% | 76.16% | -0.1063 | -0.4836 |
| Qwen3-4b-Base | 85.33% | 75.72% | -0.2145 | -1.0180 |

#### 4.2.1 APRM Score Correlation Analysis

To assess whether APRM captures process quality beyond final-answer correctness, we compute Pearson correlations between APRM scores and GLM-5.1 judge ratings separately within correct and incorrect samples. For each backbone, the judge evaluates 200 questions with 6 rollouts each (1,200 samples) on coherence, conciseness, redundancy-related quality, step validity, and overall process quality. Results and judge prompts appear in Figure 5 and Appendix A, respectively.

APRM retains discriminative power within both correctness subsets. Among correct samples, it correlates positively with overall process quality ($r = 0.399$ for 1.7B and $r = 0.355$ for 4B) and coherence ($r = 0.387$ and $r = 0.367$), but negligibly with response length ($r = -0.082$ and $r = -0.027$). Among incorrect samples, positive correlations persist, particularly for conciseness ($r = 0.565$ for 1.7B and $r = 0.393$ for 4B) and redundancy-related quality ($r = 0.502$ and $r = 0.419$), while negative length correlations suggest lower scores for verbose failed reasoning. Overall, APRM exhibits length-decoupling behavior and captures complementary process-quality information beyond existing reward supervision, rather than merely serving as a proxy for existing metrics.

#### 4.2.2 Information Gain and Case Study.

To test whether APRM adds information beyond existing reward signals, we compare nested regressions $Q \sim X + B$ and $Q \sim X + B + Z$, where $Q$ is LLM-judged overall process quality, $X$ denotes prompt

Table 2: Performance comparison of different RL algorithms and APRM-enhanced variants. Best results are highlighted in bold. Δ indicates the performance change of the method by applying the APRM.

| Method | AIME2025 | | AMC | | MATH500 | | Olympiad | | Average | |
|---|---|---|---|---|---|---|---|---|---|---|
| | avg@8 | pass@8 | avg@8 | pass@8 | avg@8 | pass@8 | avg@8 | pass@8 | avg@8 | pass@8 |
| **Qwen3-1.7b-Base** | | | | | | | | | | |
| Base | 1.67% | 10.00% | 27.81% | 60.00% | 47.40% | 75.80% | 20.77% | 41.99% | 24.41% | 46.95% |
| GRPO | 5.00% | 20.00% | 39.69% | 65.00% | 61.43% | 78.60% | 27.32% | 46.88% | 33.36% | 52.62% |
| GSPO | 3.75% | 13.33% | 40.94% | 72.50% | 61.32% | 78.40% | 26.80% | 47.03% | 33.20% | 52.82% |
| GPG | 5.83% | 20.00% | 39.38% | 67.50% | 63.10% | 80.00% | 29.53% | 50.15% | 34.46% | 54.41% |
| GMPO | 4.58% | 10.00% | 41.25% | **75.00%** | 62.22% | 78.40% | 29.66% | 47.77% | 34.43% | 52.79% |
| FAPO | 3.75% | 13.3% | 37.50% | 65.00% | 58.65% | 74.80% | 24.72% | 42.73% | 31.16% | 48.97% |
| **GRPO+APRM** | **6.25%** | **26.67%** | 42.50% | 67.50% | 61.82% | 80.00% | 28.62% | 49.70% | 34.80% | 55.97% |
| Δ | +1.25 | +6.67 | +2.81 | +2.50 | +0.40 | +1.40 | +1.30 | +2.82 | +1.44 | +3.35 |
| **GSPO+APRM** | 4.58% | 20.00% | **43.13%** | 72.50% | 60.90% | 78.80% | 28.15% | 48.07% | 34.19% | 54.84% |
| Δ | +0.83 | +6.67 | +2.19 | +0.00 | -0.42 | +0.40 | +1.35 | +1.04 | +0.99 | +2.03 |
| **GPG+APRM** | **6.25%** | **26.67%** | 40.62% | **75.00%** | **63.30%** | **80.40%** | **29.99%** | **50.45%** | **35.04%** | **58.13%** |
| Δ | +0.42 | +6.67 | +1.25 | +7.50 | +0.20 | +0.40 | +0.46 | +0.30 | +0.58 | +3.72 |
| **Qwen3-4b-Base** | | | | | | | | | | |
| Base | 3.75% | 13.33% | 29.38% | 77.50% | 44.98% | 81.40% | 22.55% | 51.63% | 25.16% | 55.97% |
| GRPO | 10.00% | 23.33% | 57.81% | 85.00% | 72.65% | 82.60% | 41.36% | 55.79% | 45.46% | 61.68% |
| GSPO | 10.83% | 23.33% | 57.50% | 77.50% | 73.18% | 84.40% | 40.02% | 57.72% | 45.38% | 60.74% |
| GPG | 12.50% | 26.67% | 58.13% | 85.00% | 73.33% | 84.40% | **41.69%** | 59.64% | 46.41% | 63.93% |
| GMPO | 12.50% | 26.67% | 59.69% | 80.00% | 73.70% | 83.20% | 41.26% | 58.61% | 46.79% | 62.12% |
| FAPO | 9.17% | 23.33% | 55.31% | 80.00% | 72.08% | 83.60% | 37.80% | 54.30% | 43.59% | 60.31% |
| **GRPO+APRM** | 11.67% | **30.00%** | 56.88% | **87.50%** | 73.13% | 84.00% | 41.12% | 58.75% | 45.70% | 65.06% |
| Δ | +1.67 | +6.67 | -0.94 | +2.50 | +0.47 | +1.40 | -0.24 | +2.97 | +0.24 | +3.38 |
| **GPG+APRM** | **13.33%** | **30.00%** | **60.31%** | **87.50%** | **73.82%** | **85.60%** | 40.62% | **60.24%** | **47.02%** | **65.84%** |
| Δ | +0.83 | +3.33 | +2.18 | +2.50 | +0.50 | +1.20 | -1.07 | +0.60 | +0.61 | +1.91 |

fixed effects, $B$ includes correctness and log response length, and $Z$ is the APRM score. Adding $Z$ improves $R^2$ by $0.0171$ on Qwen3-4b-Base and $0.0212$ on Qwen3-1.7b-Base, with partial correlations of $0.3332$ and $0.3167$, respectively, and permutation-test $p = 0.0033$. These results support $I(Q; Z \mid X, B) > 0$, indicating non-redundant process-quality information beyond prompt and output-level controls. Details appear in Appendix F.3.

Figure 6 illustrates concept-level attribution for rollouts of the same problem. Concepts with the largest assignment differences between correct and incorrect rollouts are projected onto tokens through the learned assignment matrix $\mathbf{W}$, grouping high-weight tokens into highlighted spans. The correct rollout scores higher, with highlights on symbolic derivation, while the incorrect rollout scores lower, with highlights on search-oriented behavior. This suggests greater sensitivity to reasoning structure than isolated numerical computation, favoring algebraically consistent, constraint-satisfying trajectories and penalizing superficially plausible but search-heavy or constraint-violating reasoning. More cases appear in Appendix B.

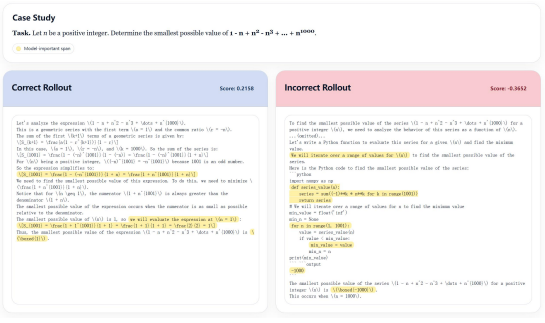


Figure 6: Representative concept-level attribution for correct and incorrect rollouts from the same problem.

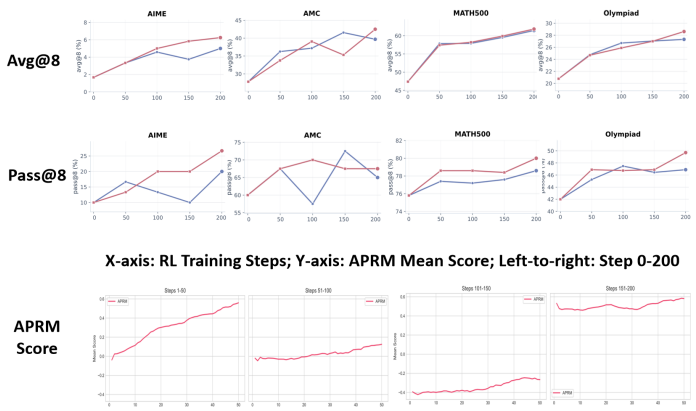


Figure 7: Training dynamics of GRPO and GRPO+APRM.

Table 4: Median response token length across benchmarks. Each entry reports baseline / APRM / relative change.

| Method | Model | Category | Average |
|---|---|---|---|
| GPG | Qwen3-1.7b-Base | Correct | 701.62 / 661.38 / -5.74% |
| | | Incorrect | 900.88 / 884.62 / -1.80% |
| | | Overall | 809.38 / 784.88 / -3.03% |
| | Qwen3-4b-Base | Correct | 847.12 / 723.50 / -14.59% |
| | | Incorrect | 1102.00 / 902.00 / -18.15% |
| | | Overall | 928.25 / 769.75 / -17.08% |
| GRPO | Qwen3-1.7b-Base | Correct | 564.88 / 693.75 / +22.81% |
| | | Incorrect | 706.00 / 880.12 / +24.66% |
| | | Overall | 636.38 / 778.00 / +22.25% |
| | Qwen3-4b-Base | Correct | 743.00 / 670.12 / -9.81% |
| | | Incorrect | 912.25 / 823.25 / -9.76% |
| | | Overall | 783.50 / 721.25 / -7.95% |
| GSPO | Qwen3-1.7b-Base | Correct | 634.00 / 602.62 / -4.95% |
| | | Incorrect | 777.75 / 842.00 / +8.26% |
| | | Overall | 712.62 / 724.25 / +1.63% |

### 4.3 Main Results on RL Training

Table 2 shows that adding APRM generally improves the corresponding RL algorithms across both model scales, especially on pass@8. On Qwen3-1.7b-Base, GRPO+APRM raises average pass@8 from 52.62% to 55.97%, while GPG+APRM achieves the best result with 58.13%. On Qwen3-4b-Base, GRPO+APRM improves pass@8 from 61.68% to 65.06%, and GPG+APRM obtains the best average performance with 47.02% avg@8 and 65.84% pass@8. The larger improvement on pass@8 suggests that APRM mainly enhances the quality and diversity of the candidate solution set rather than only improving the top-ranked single output. By providing a process-level auxiliary signal, APRM encourages the policy to assign higher probability to trajectories with more coherent intermediate reasoning structures.

#### 4.3.1 RL Training Dynamics

Figure 7 compares the training curves of GRPO and GRPO+APRM. Both methods improve rapidly in the early stage, while APRM gives more consistent gains in later training, especially on pass@8. On AIME, MATH500, and Olympiad, GRPO+APRM ends with a higher pass@8 than the GRPO baseline. The avg@8 curves show smaller but generally positive gains. Moreover, Figure 7 shows the mean APRM score within each 50-step training phase. In the early phase, the score increases steadily, indicating that the policy quickly learns to produce trajectories preferred by APRM. In later phases, the upward trend becomes slower but remains positive, suggesting continued improvement in APRM-valued reasoning quality. The phase-wise shifts are caused by periodic APRM refreshing, which changes the score scale after each update.

Table 3: Ablation study on different $\beta$.

| $\beta$ | AIME2025 | | AMC | | MATH500 | |
|---|---|---|---|---|---|---|
| | avg@8 | pass@8 | avg@8 | pass@8 | avg@8 | pass@8 |
| 0.1 | 5.83% | 26.67% | 41.88% | 67.50% | 61.55% | 79.00% |
| 0.2 | 4.58% | 16.67% | 41.25% | 72.50% | 60.88% | 77.40% |
| 0.3 | 6.25% | 26.67% | 42.50% | 67.50% | 61.82% | 80.00% |
| 0.4 | 5.83% | 23.33% | 40.31% | 70.00% | 60.62% | 76.80% |
| 0.5 | 5.00% | 20.00% | 39.69% | 62.50% | 61.78% | 78.40% |

#### 4.3.2 Ablation Study and Analysis on Response Length

To further examine the behavioral effect of APRM, we compare the response length of APRM-enhanced models with their corresponding baselines across benchmarks. We report the median generated token length. The results are summarized in Table 4. Detailed results and analysis are shown in Appendix D. APRM optimizes the model's reasoning patterns, which may affect the output length differently across various methods. Overall, however, it exhibits a tendency to produce more concise responses. The largest overall decrease is observed for GPG on Qwen3-4b-Base (-17.08%), suggesting that APRM helps the model adapt

its token budget to problem difficulty during internal deliberation. Table 3 studies the effect of the $\beta$ on the final advantage. A moderate value performs best overall, with $\beta = 0.3$ achieving the strongest avg@8 on AIME2025 and AMC, as well as the highest pass@8 on MATH500. Smaller values underuse the process signal, while larger values can overemphasize APRM and weaken the outcome reward.

# 5 Related Works

## 5.1 RL for LLM

Early RL-based approaches commonly employ PPO-style Schulman et al. (2017) optimization. By eliminating the value model, GRPO Shao et al. (2024) has largely emerged as a widely adopted alternative to PPO. Following this direction, GPG Chu et al. (2025) revisits vanilla policy gradient and directly optimizes the original RL objective without relying on surrogate losses; GSPO Zheng et al. (2025a) moves policy optimization from token-level importance ratios to sequence-level likelihood ratios; and GMPO Zhao et al. (2025) introduces a geometric-mean objective to reduce sensitivity to outlier rewards and extreme importance ratios. Another line of work focuses on designing richer process reward signals to address the sparsity of outcome supervision Zhang et al. (2024); Setlur et al. (2025); Gao et al. (2024). These methods have shown that process-level supervision can substantially improve mathematical reasoning. RLAnything Wang et al. (2026) proposes a dynamic closed-loop RL framework that jointly evolves the environment, policy, and reward model. However, these approaches still primarily define rewards over external outcomes, textual trajectories, or task-level interactions, leaving model-internal reasoning signals largely unexplored.

## 5.2 Attention Mechanism in RL

Recent circuit analysis studies have shown that attention-based sub-circuits contribute to and implement interpretable model behaviors Lan et al. (2024); Merullo et al. (2024). Furthermore, causal intervention methods have demonstrated that certain attention heads have identifiable functional contributions to model outputs Kissane et al. (2024); Nam et al. (2026). These findings suggest that attention patterns are not merely superficial statistical correlations with model outputs, but can provide partial observations of the internal computation underlying model behavior. TCRM Nikulkov (2026) exploits causal attention with temporal regularization for smooth token-level values. Extending beyond linear reasoning, GraphPRM Peng et al. (2025) models trajectories as graphs with GAT-based dependency learning for error localization. To mitigate PRM-induced verbosity and length bias, CoLD Zheng et al. (2025b) uses feature perturbation to force a uniform self-attention distribution in the bias estimator, precisely filtering out the PRM's length bias via counterfactual inference. Multilingual PRM Wang et al. (2025) further aligns attention representations across languages for robust multilingual policies. Unlike reward-centric methods, RAL Li et al. (2026) directly optimizes self-attention distributions as the RL policy itself. Unlike prior work, APRM is the first to directly supervise a model's internal signals, scoring attention and hidden states directly in the latent space.

# 6 Conclusion

We reveal textual reward aliasing, where identical or similar external trajectories may conceal distinct internal reasoning processes and thus receive indistinguishable rewards. To address this, we propose APRM, a lightweight internal process reward model with 46M parameters that scores hidden-state and attention-based reasoning signals. Integrated with existing RL algorithms, APRM provides complementary supervision and consistently improves reasoning performance. Interestingly, APRM also encourages more efficient reasoning behaviors, with a tendency toward shorter reasoning trajectories while maintaining or improving performance. These results suggest a broader direction for RL that rewards not only what models say but also how they think.

## REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide detailed descriptions of our experimental configurations, model sources, and hyperparameter settings in Section 4.1. Additional implementation details, experimental settings, prompts, and analysis are provided in Appendix C. Furthermore, we provide an anonymous code repository here containing the core implementation and results for our experiments.

## AI USE STATEMENT

In the process of writing this paper, we used AI assistants to check for syntactic and grammatical errors in the manuscript. We carefully reviewed and verified all revised content.

## REFERENCES

Fazl Barez, Tung-Yu Wu, Iván Arcuschin, Michael Lan, Vincent Wang, Noah Siegel, Nicolas Collignon, Clement Neo, Isabelle Lee, Alasdair Paren, et al. Chain-of-thought is not explainability. *Preprint, alphaXiv*, pp. v1, 2025.

Yuji Cao, Huan Zhao, Yuheng Cheng, Ting Shu, Yue Chen, Guolong Liu, Gaoqi Liang, Junhua Zhao, Jinyue Yan, and Yun Li. Survey on large language model-enhanced reinforcement learning: Concept, taxonomy, and methods. *IEEE Transactions on Neural Networks and Learning Systems*, 36(6):9737–9757, 2024.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM transactions on intelligent systems and technology*, 15(3):1–45, 2024.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, et al. Reasoning models don't always say what they think. *arXiv preprint arXiv:2505.05410*, 2025.

Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. Gpg: A simple and strong reinforcement learning baseline for model reasoning. *arXiv preprint arXiv:2504.02546*, 2025.

Yuyang Ding, Chi Zhang, Juntao Li, Haibin Lin, and Min Zhang. Fapo: flawed-aware policy optimization for efficient and reliable reasoning. *arXiv preprint arXiv:2510.22543*, 2025.

Jiaxuan Gao, Shusheng Xu, Wenjie Ye, Weilin Liu, Chuyi He, Wei Fu, Zhiyu Mei, Guangju Wang, and Yi Wu. On designing effective rl reward at training time for llm reasoning. *arXiv preprint arXiv:2410.15115*, 2024.

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.

Connor Kissane, Robert Krzyzanowski, Joseph Isaac Bloom, Arthur Conmy, and Neel Nanda. Interpreting attention layer outputs with sparse autoencoders. *arXiv preprint arXiv:2406.17759*, 2024.

Michael Lan, Philip Torr, and Fazl Barez. Towards interpretable sequence continuation: Analyzing shared circuits in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 12576–12601, 2024.

Bangzheng Li, Jianmo Ni, Chen Qu, Ian Miao, Liu Yang, Xingyu Fu, Muhao Chen, and Derek Zhiyuan Cheng. Reinforced attention learning. *arXiv preprint arXiv:2602.04884*, 2026.

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.

Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. Talking heads: Understanding inter-layer communication in transformer language models. *Advances in Neural Information Processing Systems*, 37:61372–61418, 2024.

Andrew Nam, Henry Conklin, Yukang Yang, Tom Griffiths, Jonathan D Cohen, and Sarah-Jane Leslie. Causal head gating: A framework for interpreting roles of attention heads in transformers. *Advances in Neural Information Processing Systems*, 38:115609–115631, 2026.

Alex Nikulkov. Reward models are secretly value functions: Temporally coherent reward modeling. *arXiv preprint arXiv:2604.22981*, 2026.

Miao Peng, Nuo Chen, Zongrui Suo, and Jia Li. Rewarding graph reasoning process makes llms more generalized reasoners. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 2257–2268, 2025.

Aske Plaat, Max van Duijn, Niki Van Stein, Mike Preuss, Peter van der Putten, and Kees Joost Batenburg. Agentic large language models, a survey. *Journal of Artificial Intelligence Research*, 84, 2025.

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

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

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, 2024.

Weixuan Wang, Minghao Wu, Barry Haddow, and Alexandra Birch. Demystifying multilingual chain-of-thought in process reward modeling. *arXiv preprint arXiv:2502.12663*, 2025.

Yinjie Wang, Tianbao Xie, Ke Shen, Mengdi Wang, and Ling Yang. Rlanything: Forge environment, policy, and reward model in completely dynamic rl system. *arXiv preprint arXiv:2602.02488*, 2026.

Zhiheng Xi, Chenyang Liao, Guanyu Li, Zhihao Zhang, Wenxiang Chen, Binghai Wang, Senjie Jin, Yuhao Zhou, Jian Guan, Wei Wu, et al. Agentprm: Process reward models for llm agents via step-wise promise and progress. In *Proceedings of the ACM Web Conference 2026*, pp. 4184–4195, 2026.

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

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

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. Rest-mcts*: Llm self-training via process reward guided tree search. *Advances in Neural Information Processing Systems*, 37: 64735–64772, 2024.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, et al. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*, 2025.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2):1–124, 2023.

Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen, Xun Wu, Yaru Hao, Tengchao Lv, Shaohan Huang, Lei Cui, Qixiang Ye, et al. Geometric-mean policy optimization. *arXiv preprint arXiv:2507.20673*, 2025.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025a.

Congmin Zheng, Jiachen Zhu, Jianghao Lin, Xinyi Dai, Yong Yu, Weinan Zhang, and Mengyue Yang. Cold: Counterfactually-guided length debiasing for process reward models. *arXiv preprint arXiv:2507.15698*, 2025b.

Jialun Zhong, Wei Shen, Yanzeng Li, Songyang Gao, Hua Lu, Yicheng Chen, Yang Zhang, Wei Zhou, Jinjie Gu, and Lei Zou. A comprehensive survey of reward models: Taxonomy, applications, challenges, and future. *arXiv preprint arXiv:2504.12328*, 2025.

## A Prompt Details

For the LLM-as-judge evaluation, we use GLM-5.1 to assess the process quality of APRM-scored reasoning samples. Each judged instance contains the original math question, the gold final answer, the model-generated solution, and a binary correctness flag. The judge is instructed to evaluate reasoning quality rather than final-answer correctness, and returns integer scores from 0 to 10 on five dimensions: reasoning coherence, conciseness, redundancy, step validity, and overall process quality. We use deterministic decoding with temperature 0 and top-$p$ 1.0, and require the judge to output a JSON object with the five scores and a brief rationale. The returned scores are parsed, clipped to the range $[0, 10]$, and stored together with the corresponding APRM score for correlation analysis. The prompt we use is as follows:

**Teacher LLM's System Prompt**

```
You are an expert evaluator for mathematical reasoning quality.

You will be given:
1. A math question
2. The gold final answer
3. A model-generated solution
4. Whether the final answer is correct

Your job is to evaluate the reasoning process quality of the model-generated solution, not just the final answer
      correctness.

Please score the solution on the following five dimensions, each on an integer scale from 0 to 10:

1. reasoning_coherence
- Does the reasoning flow logically from step to step?
- Are the transitions natural and internally faithful to the line of thought?
- High score: coherent, well-connected reasoning chain.
- Low score: abrupt jumps, contradictions, disconnected reasoning.

2. conciseness
- Does the solution use an efficient amount of text to communicate the reasoning?
- High score: compact, information-dense, little unnecessary verbosity.
- Low score: overly long, bloated, too much unnecessary explanation.

3. redundancy
- How much repetitive or unnecessary repetition is present?
- High score: little repetition, low redundancy.
- Low score: repetitive statements, circular restatements, duplicated work.

4. step_validity
- Are the intermediate reasoning steps mathematically valid and locally sound?
- High score: most steps are valid and justified.
- Low score: frequent invalid transformations, unsupported claims, local mathematical errors.

5. overall_process_quality
- Overall quality of the reasoning process, considering the above dimensions together.
- High score: strong reasoning process even if the final answer is wrong due to a small slip.
- Low score: poor reasoning process even if the final answer happens to be correct by luck.

Important instructions:
- Do not simply map final correctness to a high score. A correct answer with poor reasoning can still score low.
- Do not punish a small final arithmetic slip too harshly if the process is otherwise strong.
- Focus on the reasoning process quality, not surface style alone.
- Return only a single JSON object, with no markdown fences and no extra text.
- All scores must be integers between 0 and 10.

Return this JSON schema exactly:
{
  "reasoning_coherence": 0,
  "conciseness": 0,
  "redundancy": 0,
  "step_validity": 0,
  "overall_process_quality": 0,
  "brief_rationale": ""
}

The "brief_rationale" should be a short English explanation in 1 to 3 sentences summarizing the judgment.
```




Figure 8: Additional APRM case study 1.

## B Additional APRM Case Studies

Due to space limitations in the main text, we provide additional APRM case studies in the appendix. These examples further illustrate that APRM does not simply act as a proxy for response length, but assigns scores according to the structure and reliability of the reasoning process.

In Figure 8, APRM assigns a higher score to the rollout that correctly uses the algebraic property of the nonreal cubic root and derives the final value, while the lower-scored rollout reaches an incorrect result despite presenting a compact derivation. In Figure 9, the higher-scored rollout applies the appropriate trigonometric relation and obtains the correct side length, whereas the lower-scored rollout uses an incorrect secant-based approximation and produces a wrong answer. Figure 10 shows a base-conversion problem where APRM favors the solution with a more faithful conversion and digit-sum computation over a shorter but incorrect trajectory.

Overall, these cases show that APRM scores are sensitive to reasoning validity and structural correctness rather than verbosity. Short responses can still receive low scores when their reasoning is flawed.

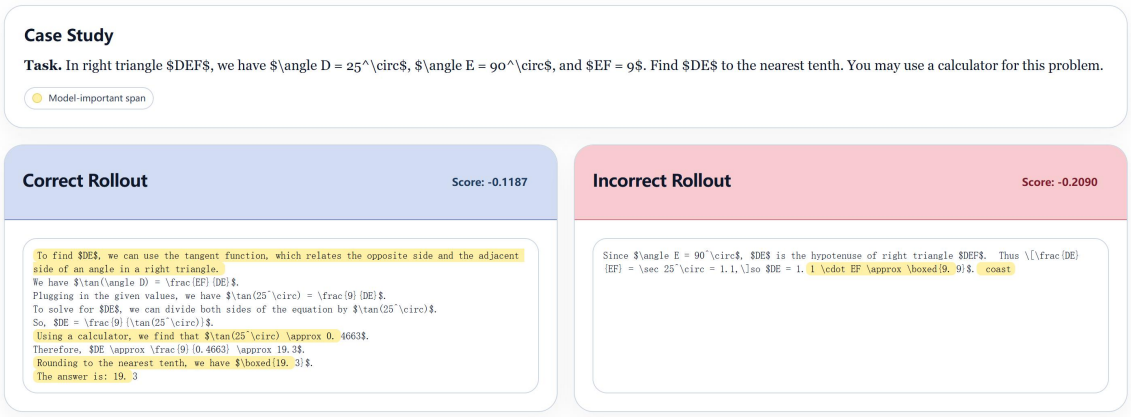


Figure 9: Additional APRM case study 2.

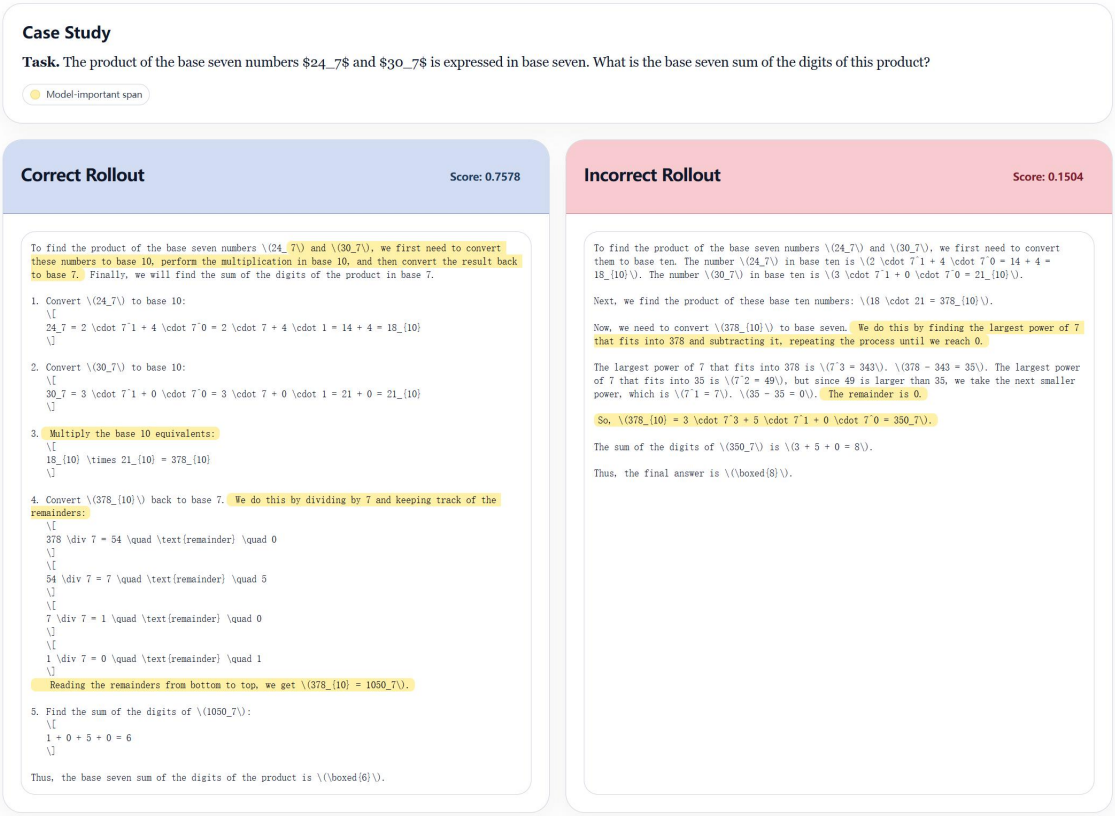


Figure 10: Additional APRM case study 3.

# C EXPERIMENTAL DETAILS

## C.1 TRAINING DATASETS AND TEST DATASETS

For APRM training, we construct a mixed mathematical reasoning corpus from DAPO-17k and MATH. DAPO-17k provides diverse competition-style math prompts for RL post-training, while MATH contains problems with graded difficulty levels. We use MATH problems from Level 3 to Level 5 to emphasize medium-to-hard reasoning cases. The final training pool contains 10K questions, mixed with a 7:3 ratio. 7K questions are sampled from DAPO-17k, and 3K questions are sampled from MATH Level 3 to 5.

During evaluation, all methods use the same prompt template:

```
{problem}. Let's think step by step and output the final answer within \boxed{}.
```

We report both avg@$k$ and pass@$k$ with $k = 8$. For each problem $i$, we sample $k$ responses and denote whether the $j$-th response is correct by $c_{i,j} \in \{0, 1\}$. Then

$$\text{avg@}k = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{k} \sum_{j=1}^{k} c_{i,j},$$

which measures the average correctness over all sampled responses. We also compute

$$\text{pass@}k = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I} \left[ \sum_{j=1}^{k} c_{i,j} > 0 \right],$$

which measures whether at least one correct response appears among the $k$ samples.

## C.2 APRM ARCHITECTURE DETAILS

We provide the implementation details of the Edge-Biased GAT used in APRM. Given concept node representations

$$\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\}, \quad \mathbf{x}_i \in \mathbb{R}^{d_g},$$

and the sparsified concept graph

$$\mathbf{E} = \hat{\mathbf{A}}_c \in \mathbb{R}^{M \times M}, \quad e_{ij} = \hat{\mathbf{A}}_{c,ij},$$

each Edge-Biased GAT layer first normalizes the node features:

$$\bar{\mathbf{x}}_i = \text{LayerNorm}(\mathbf{x}_i).$$

For each attention head $h$, query, key, and value vectors are computed as

$$\mathbf{q}_i^{(h)} = \mathbf{W}_Q^{(h)} \bar{\mathbf{x}}_i, \quad \mathbf{k}_j^{(h)} = \mathbf{W}_K^{(h)} \bar{\mathbf{x}}_j, \quad \mathbf{v}_j^{(h)} = \mathbf{W}_V^{(h)} \bar{\mathbf{x}}_j.$$

The edge value $e_{ij}$ is used in two ways. First, it produces an additive edge bias,

$$b_{ij}^{(h)} = \frac{1}{2} \tanh \left( \mathbf{w}_b^{(h)} e_{ij} + b_b^{(h)} \right),$$

which modifies the attention logit:

$$\ell_{ij}^{(h)} = \frac{\mathbf{q}_i^{(h)} (\mathbf{k}_j^{(h)})^\top}{\sqrt{d_h}} + b_{ij}^{(h)}.$$

The attention weight is then computed by

$$\alpha_{ij}^{(h)} = \mathrm{Softmax}_j\left(\ell_{ij}^{(h)}\right).$$

Second, the same edge value produces an edge gate,

$$g_{ij}^{(h)} = \sigma\left(\mathbf{w}_g^{(h)} e_{ij} + b_g^{(h)}\right),$$

which controls the strength of the transmitted message. The message aggregated by node $i$ in head $h$ is

$$\mathbf{m}_i^{(h)} = \sum_{j=1}^{M} \alpha_{ij}^{(h)} g_{ij}^{(h)} \mathbf{v}_j^{(h)}.$$

The outputs of all heads are concatenated and projected:

$$\mathbf{m}_i = \mathrm{Concat}\left(\mathbf{m}_i^{(1)}, \ldots, \mathbf{m}_i^{(H)}\right), \quad \tilde{\mathbf{x}}_i = \mathbf{W}_O \mathbf{m}_i.$$

Finally, the layer applies a residual connection and normalization:

$$\mathbf{x}_i' = \mathrm{LayerNorm}\left(\mathbf{x}_i + \mathrm{Dropout}(\tilde{\mathbf{x}}_i)\right).$$

### C.3 APRM Training Details

In our implementation, APRM uses 64 latent concepts. The attention-fused concept pooler uses a hidden dimension of 1024. The concept graph is processed by an 8-layer Edge-Biased GAT with hidden dimension 1024, 8 attention heads, dropout 0.1, and top-16 edge sparsification. Overall, APRM contains about 46M parameters.

We train APRM for 5 epochs using AdamW with learning rate $1 \times 10^{-4}$, weight decay $1 \times 10^{-2}$, cosine learning-rate scheduling, and a warmup ratio of 0.05. Training uses bfloat16 precision, group-level batch size 2, gradient accumulation over 4 steps, and DeepSpeed acceleration. APRM training loss is shown in Figure 11.

| **Number of Concepts** | | | | |
|---|---|---|---|---|
| Concepts | 16 | 32 | 64 | 128 |
| Rank Acc. (%) | 72.68 | 73.38 | **78.55** | 72.82 |
| **Layer Index** | | | | |
| Layer | 4 | 6 | 8 | 12 |
| Rank Acc. (%) | 72.31 | 72.59 | **78.55** | 73.17 |

Table 5: Ablation results of different numbers of concepts and layers.

We additionally provide ablation results on the number of latent concepts and the depth of the Edge-Biased GAT in Table 5. APRM obtains the best rank accuracy of 78.55% with 64 latent concepts. Using fewer concepts limits the representational capacity of the concept space, while using too many concepts may introduce redundant or noisy semantic factors, both of which negatively affect ranking performance. For the GAT depth, the 8-layer model performs best. Shallower GATs are less capable of modeling complex attention-guided concept interactions, whereas overly deep GATs may suffer from over-smoothing, causing concept representations to lose discriminative information.

### C.4 RL Training Details

We use the verl framework as the training backbone for all reinforcement learning experiments. For each RL algorithm, including GRPO, GPG, GSPO, GMPO, and FAPO, we keep the default hyperparameters provided by its verl implementation unless otherwise specified. To ensure a fair comparison, each baseline and its APRM-enhanced variant use the same GPU configuration, batch size, and training schedule.

Due to GPU computational constraints, we set the per-GPU batch size to 6. For experiments run on 4 GPUs, the global batch size is 48 and the model is trained for 200 steps. For experiments run on 2 GPUs, the global batch size is 24 and the model is trained for 400 steps. This keeps the total number of training samples approximately aligned across different settings. When comparing an RL method with and without APRM, all default RL hyperparameters and hardware settings are kept unchanged, except for the additional APRM-related hyperparameters introduced by our method. For APRM-enhanced RL, the APRM advantage coefficient is denoted by $\beta$. We use $\beta = 0.3$ for Qwen3-1.7b-Base and $\beta = 0.1$ for Qwen3-4b-Base. As an exception, for GSPO we use $\beta = 0.2$.

## D Detailed Analysis of Response Length

Table 6 reports the median response length across benchmarks, separated by method, model scale, and correctness. The comparison suggests that APRM does not impose a fixed length preference. Instead, its effect depends on the underlying RL algorithm and model scale.

For GPG, APRM consistently shortens responses. The reduction is mild on Qwen3-1.7b-Base, with the overall average length decreasing by 3.03%, and becomes more pronounced on Qwen3-4b-Base, where the overall average length decreases by 17.08%. A similar pattern appears for GRPO on Qwen3-4b-Base, where APRM reduces the overall response length by 7.95%. These results indicate that APRM often suppresses unnecessarily long reasoning traces, especially for larger backbones. For GRPO on Qwen3-1.7b-Base and GSPO on Qwen3-1.7b-Base, APRM increases the overall response length by 22.25% and 1.63%, respectively. This suggests that APRM changes the model's reasoning behavior rather than simply penalizing long outputs. In some cases, it encourages more complete intermediate reasoning when the baseline responses are relatively short.

Overall, APRM tends to reduce verbosity in most settings while remaining sensitive to the optimization dynamics of different RL algorithms. This supports the view that APRM is not a length proxy, but a process-level signal that reshapes reasoning trajectories according to their internal structure and quality.

## E Algorithm Process

This section summarizes the training procedure of APRM-enhanced reinforcement learning. APRM is used as a plug-in internal reward module. The overall process of our algorithm is shown in Algorithm 1.

## F Proof of Propositions

### F.1 Proof of Proposition 1

Consider a causal self-attention matrix $\mathbf{A} \in \mathbb{R}^{T\times T}$ and a value matrix $\mathbf{V} \in \mathbb{R}^{T\times d_v}$, where $T$ is the sequence length and $d_v$ is the value dimension. The contextual representation is

$$\mathbf{C} = \mathbf{AV}.$$

**Algorithm 1** APRM-enhanced reinforcement learning

**Require:** Initial policy $\pi_\theta$, APRM $\mathcal{S}_\phi$, outcome verifier $V$, APRM coefficient $\beta$, update interval $U$

**for** RL step $= 1, 2, \ldots$ **do**
  Sample a batch of questions $\{x\}$
  **for** each question $x$ **do**
    Generate a rollout group $\{y_i\}_{i=1}^{G} \sim \pi_\theta(\cdot \mid x)$
    Compute outcome rewards $r_i^{\text{out}} = V(x, y_i)$
    Compute the base token-level advantages $A_{i,t}^{\text{base}}$ using the underlying RL algorithm
    Extract hidden states and attention summaries $(\mathbf{H}_i, \mathbf{A}_i)$ from $\pi_\theta$
    Compute APRM scores $s_i^{\text{APRM}} = \mathcal{S}_\phi(\mathbf{H}_i, \mathbf{A}_i)$
    Mean-center APRM scores within the rollout group:

$$\Delta_i^{\text{APRM}} = s_i^{\text{APRM}} - \frac{1}{G}\sum_{j=1}^{G} s_j^{\text{APRM}}$$

    Set $\mathbb{I}_{\text{active}}(x) = 1$ if the outcome rewards in the group are not all identical, otherwise $0$
    Form the APRM-enhanced advantage:

$$A_{i,t} = A_{i,t}^{\text{base}} + \beta\, \mathbb{I}_{\text{active}}(x) \Delta_i^{\text{APRM}}$$

  **end for**
  Update $\pi_\theta$ with the original RL objective using $A_{i,t}$ in place of $A_{i,t}^{\text{base}}$
  **if** step is divisible by $U$ **then**
    Generate fresh rollouts with the latest policy
    Update APRM $\mathcal{S}_\phi$ using ranking and counterfactual objectives
  **end if**
**end for**

For standard causal softmax attention,

$$A_{ij} = 0 \quad (j > i), \qquad A_{ij} > 0 \quad (j \le i), \qquad \mathbf{A}\mathbf{1} = \mathbf{1}.$$

Assume that $\operatorname{rank}([\mathbf{V}, \mathbf{1}]) < T$. By the rank–nullity theorem, there exists a nonzero vector $\mathbf{u} \in \mathbb{R}^T$ satisfying

$$\mathbf{u}^\top \mathbf{V} = \mathbf{0}, \qquad \mathbf{u}^\top \mathbf{1} = 0.$$

Let $\mathbf{e}_T$ denote the $T$-th standard basis vector and define

$$\mathbf{\Delta} = \mathbf{e}_T \mathbf{u}^\top, \qquad \mathbf{A}' = \mathbf{A} + \epsilon \mathbf{\Delta},$$

where

$$0 < |\epsilon| < \min_{j : u_j \neq 0} \frac{A_{Tj}}{|u_j|}.$$

This bound is strictly positive because $A_{Tj} > 0$ for all $j \le T$.

The perturbation modifies only the last row, whose causal support includes all positions. Thus, $\mathbf{A}'$ preserves causal support. Moreover, for $u_j \neq 0$,

$$A'_{Tj} = A_{Tj} + \epsilon u_j \ge A_{Tj} - |\epsilon||u_j| > 0,$$

while all other entries remain unchanged. Row normalization is also preserved:

$$\mathbf{A}'\mathbf{1} = \mathbf{A}\mathbf{1} + \epsilon \mathbf{e}_T \mathbf{u}^\top \mathbf{1} = \mathbf{1}.$$

Hence, $\mathbf{A}'$ is a valid causal attention matrix.

Finally,

$$\mathbf{A}'\mathbf{V} = \mathbf{A}\mathbf{V} + \epsilon \mathbf{e}_T \mathbf{u}^\top \mathbf{V} = \mathbf{A}\mathbf{V}.$$

Since $\epsilon \neq 0$ and $\mathbf{u} \neq \mathbf{0}$, we have $\mathbf{A}' \neq \mathbf{A}$. Therefore, under the stated rank condition, distinct valid causal attention matrices produce identical contextual representations, establishing that the mapping $\mathbf{A} \mapsto \mathbf{A}\mathbf{V}$ is non-injective over the space of valid causal attention matrices.

### F.2 Proof of Proposition 2

Under log-loss, the Bayes-optimal predictor outputs the true posterior probability, and its expected risk is bounded below by the conditional entropy. Let $Q$ denote process quality, $B$ denote existing reward signals, and $Z = (\mathbf{H}, \mathbf{A})$ denote APRM internal states.

Under log-loss, the Bayes-optimal predictor using only $(X, B)$ achieves risk

$$\mathcal{L}^*_{\text{base}} = H(Q \mid X, B).$$

If the predictor additionally conditions on $Z$, the optimal risk becomes

$$\mathcal{L}^*_{\text{APRM}} = H(Q \mid X, B, Z).$$

Therefore, the improvement obtained by conditioning on APRM internal states is

$$\begin{aligned}\Delta_{\text{IG}} &= \mathcal{L}^*_{\text{base}} - \mathcal{L}^*_{\text{APRM}} \\ &= H(Q \mid X, B) - H(Q \mid X, B, Z).\end{aligned}$$

By the definition of conditional mutual information,

$$I(Q; Z \mid X, B) = H(Q \mid X, B) - H(Q \mid X, B, Z).$$

Thus,

$$\Delta_{\text{IG}} = I(Q; Z \mid X, B) \geq 0.$$

Moreover,

$$I(Q; Z \mid X, B) > 0$$

if and only if

$$Q \not\perp\!\!\!\perp Z \mid X, B.$$

The internal attention structures remain predictive of process quality after conditioning on existing reward signals. Specifically, in the next subsection F.3, we experimentally demonstrate why this condition holds.

### F.3 Additional Evidence for Information Gain

We provide a supplementary residual information test for Proposition 2. We evaluate two independently sampled rollout sets, generated by Qwen3-4b-Base and Qwen3-1.7b-Base, respectively.

For each dataset, we regress the LLM-judged process quality $Q$ on existing reward signals $B$, including correctness and log response length, together with prompt fixed effects $X$. We then test whether adding the APRM score $Z$ improves the prediction:

$$Q \sim X + B, \qquad Q \sim X + B + Z.$$

As shown in Figure 12, the consistent positive residual trends in both rollout sets show that APRM retains predictive signal for process quality after removing correctness, response length, and prompt-specific effects, thereby empirically supporting Proposition 2 that $I(Q; Z \mid X, B) > 0$.




Figure 11: Loss curve during APRM training.

| Method | Model | Category | Type | AIME | AMC23 | MATH | Olympiad | Average |
|---|---|---|---|---|---|---|---|---|
| GPG | Qwen3-1.7b-Base | Correct | Baseline | 1009.00 | 715.00 | 453.00 | 629.50 | 701.62 |
| | | | Ours | 835.00 | 701.50 | 450.00 | 659.00 | 661.38 |
| | | | Δ | -17.24% | -1.89% | -0.66% | +4.69% | -5.74% |
| | | Incorrect | Baseline | 1031.00 | 908.00 | 782.00 | 882.50 | 900.88 |
| | | | Ours | 983.00 | 946.00 | 757.50 | 852.00 | 884.62 |
| | | | Δ | -4.66% | +4.19% | -3.13% | -3.46% | -1.80% |
| | | Overall | Baseline | 1031.00 | 852.00 | 548.50 | 806.00 | 809.38 |
| | | | Ours | 962.00 | 847.50 | 544.00 | 786.00 | 784.88 |
| | | | Δ | -6.69% | -0.53% | -0.82% | -2.48% | -3.03% |
| | Qwen3-4b-Base | Correct | Baseline | 1204.50 | 901.00 | 524.00 | 759.00 | 847.12 |
| | | | Ours | 1119.00 | 696.00 | 435.00 | 644.00 | 723.50 |
| | | | Δ | -7.10% | -22.75% | -16.98% | -15.15% | -14.59% |
| | | Incorrect | Baseline | 1177.00 | 1247.00 | 913.00 | 1071.00 | 1102.00 |
| | | | Ours | 972.00 | 978.00 | 780.00 | 878.00 | 902.00 |
| | | | Δ | -17.42% | -21.57% | -14.57% | -18.02% | -18.15% |
| | | Overall | Baseline | 1179.50 | 1023.50 | 583.00 | 927.00 | 928.25 |
| | | | Ours | 992.00 | 819.00 | 494.00 | 774.00 | 769.75 |
| | | | Δ | -15.90% | -19.98% | -15.27% | -16.50% | -17.08% |
| GRPO | Qwen3-1.7b-Base | Correct | Baseline | 728.50 | 622.00 | 368.00 | 541.00 | 564.88 |
| | | | Ours | 1071.00 | 681.00 | 417.00 | 606.00 | 693.75 |
| | | | Δ | +47.01% | +9.49% | +13.32% | +12.01% | +22.81% |
| | | Incorrect | Baseline | 767.00 | 761.00 | 610.00 | 686.00 | 706.00 |
| | | | Ours | 969.00 | 966.50 | 737.00 | 848.00 | 880.12 |
| | | | Δ | +26.34% | +27.00% | +20.82% | +23.62% | +24.66% |
| | | Overall | Baseline | 767.00 | 698.50 | 442.00 | 638.00 | 636.38 |
| | | | Ours | 975.00 | 868.00 | 502.00 | 767.00 | 778.00 |
| | | | Δ | +27.12% | +24.27% | +13.57% | +20.22% | +22.25% |
| | Qwen3-4b-Base | Correct | Baseline | 1099.50 | 758.00 | 456.00 | 658.50 | 743.00 |
| | | | Ours | 955.00 | 705.50 | 410.00 | 610.00 | 670.12 |
| | | | Δ | -13.14% | -6.93% | -10.09% | -7.37% | -9.81% |
| | | Incorrect | Baseline | 945.00 | 1034.00 | 799.50 | 870.50 | 912.25 |
| | | | Ours | 874.00 | 887.00 | 734.00 | 798.00 | 823.25 |
| | | | Δ | -7.51% | -14.22% | -8.19% | -8.33% | -9.76% |
| | | Overall | Baseline | 969.00 | 872.50 | 516.50 | 776.00 | 783.50 |
| | | | Ours | 898.50 | 795.50 | 473.00 | 718.00 | 721.25 |
| | | | Δ | -7.28% | -8.83% | -8.42% | -7.47% | -7.95% |
| GSPO | Qwen3-1.7b-Base | Correct | Baseline | 796.00 | 680.00 | 445.00 | 615.00 | 634.00 |
| | | | Ours | 863.00 | 601.50 | 389.00 | 557.00 | 602.62 |
| | | | Δ | +8.42% | -11.54% | -12.58% | -9.43% | -4.95% |
| | | Incorrect | Baseline | 822.00 | 785.00 | 699.00 | 805.00 | 777.75 |
| | | | Ours | 921.00 | 922.00 | 702.00 | 823.00 | 842.00 |
| | | | Δ | +12.04% | +17.45% | +0.43% | +2.24% | +8.26% |
| | | Overall | Baseline | 819.50 | 741.00 | 530.00 | 760.00 | 712.62 |
| | | | Ours | 911.50 | 776.50 | 473.00 | 736.00 | 724.25 |
| | | | Δ | +11.23% | +4.79% | -10.75% | -3.16% | +1.63% |

Table 6: Median response length across benchmarks. Δ denotes the relative difference of ours compared with the baseline.

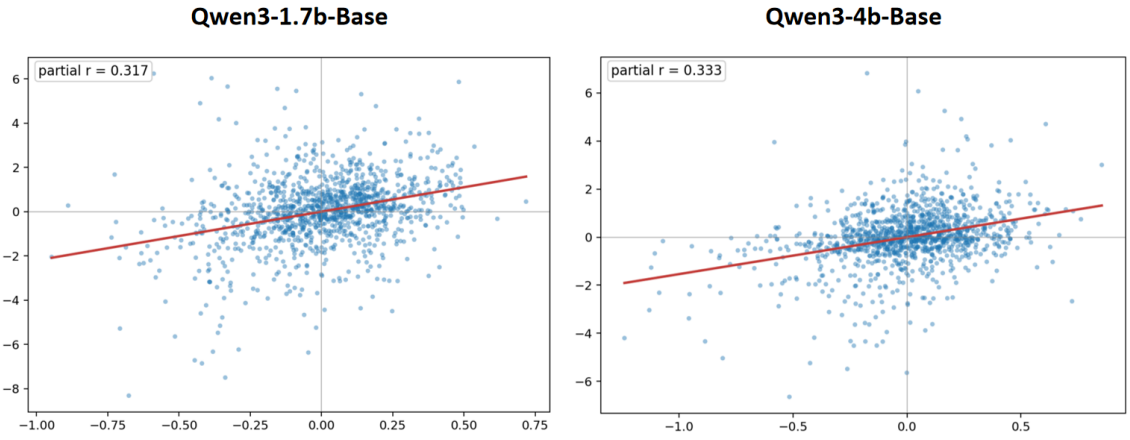


Figure 12: Residual information test for APRM. The two panels show the residual correlation between APRM score and LLM-judged process quality after controlling for correctness, response length, and prompt fixed effects.