
Rationale-Guided Policy Optimization: Learning to Reason with Adaptive Rationale Scaffolding

Anonymous Author(s)

Affiliation

Address

email

Abstract

1 On-policy reinforcement learning has become a central paradigm for improving
2 the reasoning abilities of large language models. However, its effectiveness is
3 often limited by reward sparsity: when a model fails to discover correct trajectories
4 for difficult problems, the optimization process receives little useful signal and
5 may stagnate. Existing approaches mitigate this issue by incorporating off-policy
6 demonstrations, expert traces, or model-generated solutions, but they typically
7 require the auxiliary data to match the format of the reinforcement-learning task,
8 often relying on rejection sampling from stronger models to obtain suitable training
9 trajectories. We introduce **Rationale-Guided Policy Optimization (RGPO)**, a
10 framework that adaptively leverages ground-truth rationale information according
11 to the model’s current capability while preserving its freedom to explore. Rather
12 than treating reference solutions as fixed imitation targets, RGPO uses them as
13 temporary scaffolds: rationales help the model generate improved responses, after
14 which only higher-reward, model-generated solutions are transferred back to the
15 original unguided setting. This design allows training to exploit available ground-
16 truth information without requiring off-policy data to follow the same format as the
17 RL task. Across both language-only and vision-language reasoning settings, RGPO
18 consistently improves performance over RLVR baselines, and ablation studies show
19 that adaptive rationale guidance is a key contributor to these gains. These results
20 suggest that RGPO offers a practical and general approach for reducing reward
21 sparsity, stabilizing reinforcement learning, and improving reasoning performance
22 in both text-only and multimodal models. Our implementation is available at
23 <https://anonymous.4open.science/r/hint-rlvr-D643/>.

24 1 Introduction

25 Recent advances in large language model reasoning have shown that reinforcement learning,
26 particularly reinforcement learning with verifiable rewards (RLVR), can elicit long-horizon reasoning
27 behaviors from pretrained models. Recent frontier reasoning models, including GPT-5 [1, 2], Gemini
28 2.5 [3], Claude Opus 4 and Claude Sonnet 4 [4], Qwen3 [5], Kimi K2 [6], and GLM-4.5 [7], have
29 demonstrated rapid progress in mathematics, coding, tool use, and agentic problem solving by using
30 reinforcement learning during post-training [8, 9]. Despite its effectiveness, on-policy reinforcement
31 learning remains fundamentally constrained by the current capability of the policy model. When
32 the model is trained on problems beyond its evolving reasoning ability, all sampled rollouts for
33 a prompt may be incorrect. In such cases, rewards become sparse or uniform, and the resulting
34 learning signal provides little information about how to reach a correct solution. This capacity-
35 difficulty mismatch is especially severe for weak or small models, where standard on-policy RL may
36 spend substantial computation sampling failed trajectories without obtaining positive examples for
37 improvement [10, 11, 12]. Consequently, the training process can stagnate precisely on the hard

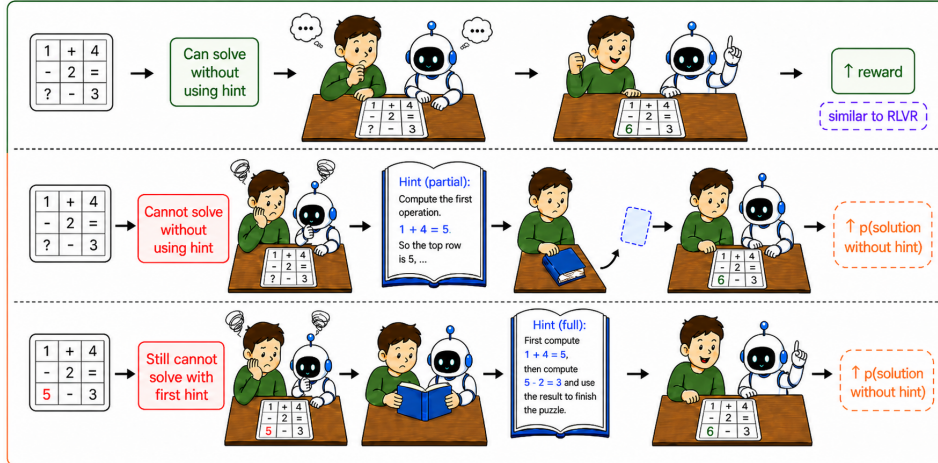


Figure 1: **Overview of the proposed method:** Inspired by how humans approach a problem, the model first tries to solve each problem without access to hints. When it succeeds, the correct response is directly reinforced. When it fails, progressively stronger rationales are introduced as temporary scaffolds to guide the model toward a correct solution.

38 examples that are most important for improving reasoning. A natural direction for mitigating reward
 39 sparsity is to incorporate external guidance, such as reference solutions or expert traces [10, 13]. Prior
 40 work has explored several forms of such guidance. However, directly using external rationales also
 41 introduces important challenges. Maximizing the likelihood of those reference solutions may force
 42 the model to imitate trajectories that are far from its own policy distribution, causing distribution
 43 mismatch, memorization, and limited generalization [14]. Off-policy expert traces can provide
 44 successful solutions, but they may not reflect the reasoning style or token-level decisions that the
 45 current model can naturally reproduce.

46 Human learning research suggests that effective reasoning instruction requires a careful balance
 47 between independent problem solving and guided assistance. Since initial struggle can prepare
 48 learners to better recognize, organize, and internalize later guidance [15, 16], they can benefit from
 49 attempting a problem before receiving explicit instruction. At the same time, intelligent tutoring
 50 research identifies an “assistance dilemma”: too little help may leave learners stuck, whereas too
 51 much help may reduce effort, encourage shallow processing, and weaken transfer [17]. Motivated by
 52 this perspective, we propose **Rationale-Guided Policy Optimization (RGPO)**, a simple yet effective
 53 framework that builds on the standard RLVR training loop for improving reasoning by using reference
 54 rationales as temporary scaffolds for exploration rather than as direct imitation targets. For each
 55 prompt, the model first generates ordinary on-policy rollouts and receives rewards from the verifier.
 56 Then, for examples where guidance may be useful, RGPO constructs a second-round refinement
 57 prompt that includes the original problem together with a rationale, solution hint, or feedback derived
 58 from the training example. The model is asked to generate a new standalone solution to the original
 59 problem. If this yields a better response than the original rollout, RGPO applies a supervised update
 60 to maximize the likelihood of the refined response conditioned only on the original prompt, rather
 61 than on the hint-augmented prompt.

62 To further balance guidance and exploration, we introduce an adaptive hinting mechanism. Rather
 63 than always revealing the full reference solution, the scheduler adjusts the amount of guidance
 64 according to the model’s current success on each problem (see Figure 1). If the model repeatedly
 65 fails to solve a problem, RGPO reveals more of the rationale to help it reach a successful trajectory.
 66 If the model can already solve the problem, the scheduler reduces or removes the hint, encouraging
 67 independent reasoning and avoiding unnecessary simplification. This adaptive mechanism allows
 68 RGPO to concentrate guidance on difficult examples while preserving on-policy exploration for
 69 problems within the model’s current capability. Overall, our contributions are summarized as follows:

- 70 • We propose **RGPO**, a rationale-guided optimization framework that augments RLVR with
 71 hint-assisted refinement. In particular, we show how reference rationales can be used as
 72 *temporary reasoning scaffolds* rather than direct imitation targets, reducing reward sparsity
 73 without requiring the rationale to match the final-answer format used in RL rollouts.

- 74 • We introduce an adaptive hint mechanism that dynamically increases or decreases the
75 amount of revealed solution guidance according to the model’s current ability.
- 76 • We conduct experiments on both language models and vision-language models, where
77 RGPO consistently shows superior training sample efficiency over standard RLVR and
78 outperforms other hybrid baselines that combine both on-policy and off-policy training.

79 2 Related Work

80 **RL with verifiable rewards.** PPO optimizes a clipped policy-gradient objective and is widely used
81 for language-model post-training [8]. GRPO replaces value estimation with group-relative reward
82 normalization, reducing training complexity for mathematical reasoning [9]. DeepSeek-R1 further
83 demonstrated that large-scale RLVR can elicit sophisticated reasoning behaviors without relying
84 entirely on supervised reasoning traces [18]. RGPO is compatible with either PPO or GRPO: it
85 leaves the first-round RL update intact and adds a second-round reward-gated distillation term.

86 **Off-policy guidance and mixed SFT–RL training.** A major limitation of pure on-policy RL is that
87 the policy can only learn from trajectories it already samples. LUFFY addresses this by incorporating
88 off-policy reasoning traces while using policy shaping to avoid rigid imitation [10]. CHORD reframes
89 SFT as a dynamically weighted auxiliary objective within RL [19]. Minimalist rejection-sampling
90 and reinforce-style methods also show that filtering positive trajectories can be a strong baseline
91 [20]. RGPO differs in that its auxiliary examples are generated online by the current policy under
92 rationale-guided refinement, then trained under the original prompt distribution. GHPO adjusts the
93 amount of guidance according to the model’s ability to solve the problem [11]. QuestA augments
94 hard questions with partial solutions [21]. BREAD inserts expert prefixes and branches rollouts from
95 them [12]. HINT supplies heuristic hints and argues that external guidance should preserve training
96 affinity [13].

97 **Failure-conditioned guidance.** The design of RGPO is primarily motivated by cognitive studies
98 on when and how guidance should be provided. Effective reasoning instruction depends on both the
99 timing and form of guidance, as learners often benefit from attempting problems before receiving
100 instruction, because initial struggle can help them notice relevant structures, activate prior knowledge,
101 and better internalize subsequent explanations [15, 16]. Besides, active recall and testing improve
102 long-term retention more effectively than passive study, motivating an initial phase of independent
103 answer generation [22]. The “assistance dilemma” [17] states that too little help may leave learners
104 stuck, while too much help may reduce effort, encourage shallow processing, and weaken transfer.
105 Studies of help seeking also show that learners may engage in hint abuse, using assistance to obtain
106 answers rather than to understand the underlying reasoning process [23]. Complementary work on
107 worked examples demonstrates that examples can support schema acquisition, but their benefits
108 depend on active engagement, such as self-explanation, rather than copying solution steps [24, 25].
109 Finally, fading worked-out steps helps learners transition from guided examples to independent
110 problem solving, suggesting that explanations and hints are most useful as temporary scaffolds rather
111 than permanent supports [26].

112 3 Background

113 3.1 Supervised Fine-Tuning

114 Supervised fine-tuning (SFT) is one of the most widely used post-training procedures for adapting a
115 pretrained language model to a target task, domain, or response format. Given a dataset of prompt–
116 response pairs $\mathcal{D}_{\text{sft}} = \{(x_i, y_i)\}_{i=1}^N$, SFT optimizes the model by maximizing the conditional
117 likelihood of the reference response:

$$\mathcal{L}_{\text{sft}}(\theta) = -\mathbb{E}_{(x,y) \sim \mathcal{D}_{\text{sft}}} \left[\sum_{t=1}^{|y|} \log \pi_{\theta}(y_t \mid x, y_{<t}) \right]. \quad (1)$$

118 In instruction-following pipelines, SFT is commonly used to initialize the policy before preference-
119 based or reward-based optimization. For example, Ouyang et al. [27] first fine-tuned a language
120 model on human-written demonstrations and then further optimize it using reinforcement learning
121 from human feedback.

122 In reasoning tasks, SFT can teach the model answer formatting, step-by-step reasoning style, and
 123 task-specific conventions. However, human-written or oracle solutions may be concise or stylistically
 124 different from model-generated reasoning (i.e. outside the model’s current reasoning distribution).
 125 As a result, simply maximizing the likelihood of such demonstrations can lead to superficial imitation
 126 or memorization [14], rather than improving the model’s ability to independently discover correct
 127 reasoning trajectories. This limitation is particularly important for weaker models, where expert
 128 traces may be too difficult to imitate directly. We can see how an 2B model cannot follow Mulberry’s
 129 reasoning template [28] and thus cannot learn how to answer user’s questions properly in the appendix.

130 3.2 Policy Optimization for Language Model Reasoning

131 Reinforcement learning (RL) provides an alternative post-training paradigm in which the model is
 132 optimized according to a reward signal rather than fixed demonstrations. For reasoning tasks, the
 133 reward can often be automatically verified, for example by checking whether the final answer matches
 134 the ground truth. Given a prompt x , the policy π_θ samples a response y , and a reward function
 135 $R(x, y)$ assigns a scalar score. The goal is to maximize expected reward:

$$J(\theta) = \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta(\cdot | x)} [R(x, y)]. \quad (2)$$

136 **Group Relative Policy Optimization (GRPO)** [9] estimates the baseline using a group of responses
 137 sampled for the same prompt. For each prompt x , the old policy samples a group of G responses:

$$\{y_1, \dots, y_G\} \sim \pi_{\theta_{\text{old}}}(\cdot | x). \quad (3)$$

138 Each response receives a reward $r_i = R(x, y_i)$. The group-relative advantage is then computed by
 139 normalizing rewards within the group:

$$\hat{A}_i = \frac{r_i - \text{mean}(\{r_j\}_{j=1}^G)}{\text{std}(\{r_j\}_{j=1}^G)}. \quad (4)$$

140 Using this relative advantage, GRPO optimizes a PPO-style clipped objective:

$$-\mathbb{E} \left[\frac{1}{G} \sum_{i=1}^G \frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \min \left(\rho_{i,t}(\theta) \hat{A}_i, \text{clip}(\rho_{i,t}(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_i \right) \right] + \beta D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}), \quad (5)$$

141 where we have defined

$$\rho_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t} | x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} | x, y_{i,<t})}. \quad (6)$$

142 **REINFORCE++** emphasizes global advantage normalization rather than prompt-local normalization
 143 [29]. If \mathcal{B} denotes a global batch of sampled responses, the normalized advantage can be written as

$$\hat{A}(x, y) = \frac{R(x, y) - \mu_{\mathcal{B}}}{\sigma_{\mathcal{B}}}, \quad (7)$$

144 where $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}$ are the mean and standard deviation of rewards or advantages over the global batch.
 145 This design aims to produce more stable advantage estimates than small prompt-level groups. The
 146 resulting objective can be viewed as a critic-free policy-gradient objective:

$$-\mathbb{E}_{(x,y) \sim \mathcal{B}} \left[\sum_{t=1}^{|y|} \hat{A}(x, y) \log \pi_\theta(y_t | x, y_{<t}) \right] + \beta D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}). \quad (8)$$

147 We follow recent reasoning-focused RL pipelines that reduce or remove KL ($\beta = 0$) both to encourage
 148 exploration [30, 31] and for simplicity.

149 4 Method: Rationale-Guided Policy Optimization

150 4.1 Problem setup

151 Let $\mathcal{D} = \{(x_i, s_i, a_i)\}_{i=1}^M$ denote the training set, where x_i is the original prompt, s_i is a reference
 152 rationale, and a_i is the final ground-truth. We assume access to a verifiable reward function
 153 $r(x, y, a) \in \mathbb{R}$, which assigns reward to a model response y for prompt x given answer a .

154 We train the model for $N + 1$ epochs with the first epoch fully unguided and is used to estimate the
 155 model’s initial capability on each training example. The following N epochs adaptively reveal or
 156 hide increments of size $1/N$ of the reference solution according to whether the current model can
 157 already solve the example. For each example, we partition the reference solution into N ordered
 158 segments,

$$s_i = [s_{i,1}; s_{i,2}; \dots; s_{i,N}],$$

159 where each segment contains approximately $1/N$ of the solution. The partition may be defined by
 160 token count, sentence boundaries, reasoning steps, or another deterministic segmentation rule. We
 161 employ raw character counts for simplicity in our experiments. Let $g_i^{(t)} \in \{0, 1, \dots, N\}$ denote the
 162 number of revealed segments for example i at epoch t . The corresponding hint is the prefix

$$h_i^{(t)} = [s_{i,1}; \dots; s_{i,g_i^{(t)}}],$$

163 with $h_i^{(t)} = \emptyset$ when $g_i^{(t)} = 0$. We initialize $g_i^{(0)} = 0$ for all examples, so the first epoch uses no hint.
 164 After each epoch, the hint level is updated adaptively: if the model produces no correct rollout for
 165 an example, then the next epoch reveals one additional segment; if the model produces at least one
 166 correct rollout, then the next epoch removes one segment. Formally, if $C_i^{(t)}$ indicates whether any
 167 rollout for example i is correct at epoch t , then

$$g_i^{(t+1)} = \begin{cases} \min(g_i^{(t)} + 1, N), & \text{if } C_i^{(t)} = 0, \\ \max(g_i^{(t)} - 1, 0), & \text{if } C_i^{(t)} = 1. \end{cases}$$

168 This update rule creates a curriculum that automatically moves each example toward the smallest
 169 amount of hinting needed to keep the problem learnable. Hard examples gradually receive more of
 170 the reference solution until the model can discover a correct trajectory, while easier examples are
 171 annealed back toward the original no-hint setting. In this way, RGPO avoids a fixed hint schedule
 172 that may over-assist easy problems or under-assist hard ones.

173 4.2 Adaptive rationale-guided rollout

174 The first epoch is a standard on-policy RL epoch without any solution hint. For each prompt x_i , we
 175 sample K rollouts from the current policy,

$$y_{i,1}^{(0)}, \dots, y_{i,K}^{(0)} \sim \pi_\theta(\cdot | x_i),$$

176 and compute their rewards $r_{i,k}^{(0)} = r(x_i, y_{i,k}^{(0)}, a_i)$. We estimate the initial solvability of each example
 177 under the current policy. We define

$$C_i^{(0)} = \mathbb{I}[\max_k r_{i,k}^{(0)} \geq \tau_{\text{correct}}]$$

178 If $C_i^{(0)} = 0$, then none of the sampled unguided responses solved the problem, and RGPO reveals the
 179 first $1/N$ of the reference solution for this example in the next epoch. If $C_i^{(0)} = 1$, then the example
 180 remains unguided in the next epoch because the model has already demonstrated that the problem is
 181 learnable without assistance.

182 For epoch $t \in \{1, \dots, N\}$, each example is assigned a hint prefix $h_i^{(t)}$ according to the previous
 183 epoch’s correctness. We then perform two rollout passes. The first pass is the ordinary unguided
 184 rollout from the original prompt,

$$y_{i,k}^{(t)} \sim \pi_\theta(\cdot | x_i),$$

185 which is used for the main on-policy RL objective. The second pass is a guided revision rollout, used
 186 only when a nonempty hint is available or when feedback from the first pass is included. The guided
 187 prompt is constructed as

$$\tilde{x}_i^{(t)} = \text{Reprompt}(x_i, h_i^{(t)}, f_i^{(t)}),$$

188 where $f_i^{(t)}$ includes verifier feedback and answer-level correctness, ... Note that $\tilde{x}_i^{(t)}$ is used only to
 189 generate candidate improved trajectories. The guided rollout is sampled as

$$\tilde{y}_{i,k}^{(t)} \sim \pi_\theta(\cdot | \tilde{x}_i^{(t)}),$$

190 and evaluated using the same reward function,

$$\tilde{r}_{i,k}^{(t)} = r(x_i, \tilde{y}_{i,k}^{(t)}, a_i).$$

191 We compare the guided candidate to the unguided rollout distribution from the same epoch. A guided
192 response is accepted into an auxiliary buffer only if it improves over the unguided attempts:

$$\tilde{y}_{i,k}^{(t)} \in \mathcal{B}_{\text{RGPO}} \iff \tilde{r}_{i,k}^{(t)} > \max_j r_{i,j}^{(t)} + \delta,$$

193 where $\delta \geq 0$ is a margin. In the simplest exact-reward setting, $\delta = 0$, so a guided response is accepted
194 whenever it achieves a higher verifier reward than all unguided rollouts for the same example. This
195 reward gate is crucial because the reference solution prefix may not always help the model reason
196 correctly. Instead of assuming that every hint-conditioned response is useful, RGPO treats the guided
197 pass as a proposal distribution and keeps only proposals that improve measured task reward.

198 4.3 Policy optimization objective

199 RGPO combines a main on-policy RL objective with an auxiliary reward-gated supervised objective.
200 The RL objective is computed on the unguided rollouts sampled from $\pi_\theta(\cdot | x_i)$, so the policy is
201 directly optimized for the deployment condition. The auxiliary supervised objective is computed on
202 accepted guided responses, but conditioned on the original prompt:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\mathbb{E}_{(x_i, \tilde{y}) \in \mathcal{B}_{\text{RGPO}}} [\log \pi_\theta(\tilde{y} | x_i)].$$

203 This objective intentionally maximizes $p_\theta(\tilde{y} | x_i)$ rather than $p_\theta(\tilde{y} | \tilde{x}_i^{(t)})$. The distinction is central to
204 RGPO, because training on $\tilde{x}_i^{(t)}$ would teach the model to rely on rationales at inference time. Training
205 on x_i instead transfers the benefit of the guided revision back into the original task distribution. The
206 full objective for each epoch is

$$\mathcal{L}_{\text{total}}(\theta) = \mathcal{L}_{\text{RL}}(\theta) + \lambda_t \mathcal{L}_{\text{SFT}}(\theta)$$

207 where \mathcal{L}_{RL} is the main RLVR objective (e.g. with GRPO or REINFORCE++), and λ_t controls the
208 weight of the auxiliary term. Our proposed method is formally presented in Algorithm 1 in the
209 appendix.

210 5 Experiments

211 We evaluate our method on post-training of language models and vision-language models, and find it
212 to be effective across a range of settings, often outperforming standard baselines. All experiments are
213 conducted using the verl framework [32].

214 5.1 RGPO on Language Models

215 For the language-model experiments, we follow the training protocol of HINT [13] and use Qwen2.5-
216 3B [33] as the base model. To ensure a consistent comparison across methods, we train on the
217 DAPO-Math-27K split used by HINT, which is derived from the DAPO-Math170K dataset [34]. We
218 optimize the model with GRPO [9]. Unlike HINT, which trains for 10 epochs, we train for 3 epochs,
219 as we observe that the model already outperforms the compared baselines under this setting. We use a
220 training batch size of 256, a maximum prompt length of 1024 tokens, and a maximum response length
221 of 4096 tokens. For each prompt, the rollout engine samples eight responses, four with hints and four
222 without hints to ensure a fair comparison. The base model is optimized with AdamW [35] using a
223 learning rate of 1×10^{-6} . For efficient large-model training, we use the FSDP2 [36], together with
224 bfloat16 precision [37], fused kernels, and gradient checkpointing [38]. We do not enable parameter
225 or optimizer offloading.

226 We benchmark against the base model, vanilla GRPO, PPO where applicable, and representative
227 approaches for improving rollout quality or efficiency in reinforcement learning with verifiable
228 rewards such as LUFFY [10], CHORD [19], GHPO [11], QuestA [21], BREAD [12], and HINT
229 [13]. Evaluation is conducted without providing hints at test time. We consider both in-distribution
230 and out-of-distribution settings. For mathematical reasoning, we use AIME24, MATH-500 [39],

Table 1: **Overall Performance Comparison of RGPO vs. Baselines.** RGPO achieves notable performance improvements on in-distribution datasets, surpassing the base Qwen2.5-3B model and the second-best approach by **5%**. The method further exhibits strong generalization on out-of-distribution datasets, delivering an additional **2.3%** performance gain.

Methods	In-Distribution				Avg	Out-of-Distribution			Avg
	AIME	Math	Olympiad	Minerva		ARC	GPQA	MMLU	
Vanilla	2.9	39.8	12.0	9.8	16.1	44.8	11.4	28.8	28.3
GRPO	4.3	44.0	18.2	12.2	19.7	45.0	11.8	28.0	28.3
CHORD	4.5	46.6	20.2	13.0	21.1	40.0	11.0	26.4	25.8
LUFFY	3.3	40.0	18.0	13.2	18.6	40.8	11.2	24.0	25.3
GHPO	4.0	42.2	19.6	12.8	19.7	45.5	<u>12.0</u>	28.2	28.6
QuestA	3.9	42.0	19.6	12.4	19.5	44.8	<u>12.0</u>	29.0	28.6
BREAD	4.1	44.4	<u>20.4</u>	<u>13.4</u>	20.6	45.5	11.8	29.2	28.8
HINT	<u>4.9</u>	<u>48.6</u>	20.2	<u>13.4</u>	<u>21.8</u>	<u>48.8</u>	11.8	<u>30.2</u>	<u>29.9</u>
RGPO	5.1	55.2	24.9	22.1	26.8	51.2	14.7	30.8	32.2

231 OlympiadBench [40] and Minerva Math [41]. Because AIME24 contains a relatively small number
 232 of test items, we report avg@32 on this benchmark, while pass@1 is used for the remaining datasets.
 233 To examine broader reasoning transfer, we also evaluate on ARC-Challenge [42], GPQA-Diamond
 234 [43], and MMLU-Pro [44].

235 Table 1 reports the overall performance of RGPO against the vanilla model, standard GRPO, and
 236 other hybrid RL training baselines. Overall, RGPO achieves the best result on every in-distribution
 237 benchmark, improving the average score from 19.7 with GRPO to 26.8, and outperforming the
 238 strongest prior baseline, HINT, by 5.0 points. The gains are especially pronounced on Math and
 239 Minerva, where RGPO improves over HINT by 6.6 and 8.7 points, respectively, suggesting that
 240 rationale-guided optimization is particularly effective on problems requiring longer or more structured
 241 reasoning. RGPO also generalizes well beyond the training distribution, achieving the highest scores
 242 on ARC, GPQA, and MMLU, with an out-of-distribution average of 32.2 compared with 29.9 for the
 243 best baseline.

244 5.2 RGPO on Vision-Language Models

245 For vision-language experiments, we follow the experimental protocol of MINT-CoT [45] and use
 246 Qwen2-VL-7B-Instruct [46] as the base multimodal model. We additionally conduct experiments on
 247 Qwen2-VL-2B-Instruct as a weaker base model to highlight the benefit of using rationales to guide
 248 the main RLVR training. Similar to the language model experiment, we train each model for 3 epochs
 249 with adaptive hinting. We use a training batch size of 512, a maximum prompt length of 768 tokens,
 250 and a maximum response length of 1024 tokens. Other hyperparameters are kept the same as in the
 251 language-model experiment.

252 **Evaluation.** We follow MINT-CoT and evaluate on multimodal mathematical reasoning
 253 benchmarks: MathVista-Math [47], and MMStar-Math [48]. For reference, we report results from
 254 contemporary VLMs, including R1-VL-7B [49, 50], Open-R1-Multimodal [51], Mulberry [28], MM-
 255 Eureka [52], LLaVA-OneVision-Qwen2-7B-OV [53], InternVL2-8B [54], InternVL2-8B-MPO [55],
 256 DeepSeek-VL2 [56], and Qwen2.5-VL-7B-Instruct [57]. We compare RGPO primarily against
 257 contemporary VLMs that were available under comparable experimental conditions but later-released
 258 state-of-the-art VLMs because they have often undergone stronger post-training pipelines. Our goal is
 259 to evaluate whether RGPO improves reasoning when applied to a fixed backbone and compared with
 260 contemporaneous models and controlled training baselines. Accordingly, we report the performance
 261 of on-policy and off-policy training methods on the MINT-CoT dataset of in this experiment.

262 Figure 2 compares RGPO against standard RLVR algorithms, including GRPO and REINFORCE++,
 263 on Qwen2-VL-2B-Instruct and Qwen2-VL-7B-Instruct. We report accuracy over training steps
 264 to evaluate both final reasoning performance and learning efficiency. Across all settings, RGPO
 265 improves the data efficiency over the corresponding RLVR baseline. On Qwen2-VL-2B with GRPO,
 266 RGPO reaches the baseline performance about 2.95× faster and achieves a higher final accuracy.
 267 The improvement is even larger when compared with REINFORCE++, where the baseline struggles
 268 to learn effectively, while RGPO continues to improve and obtains a substantial final gain. On the

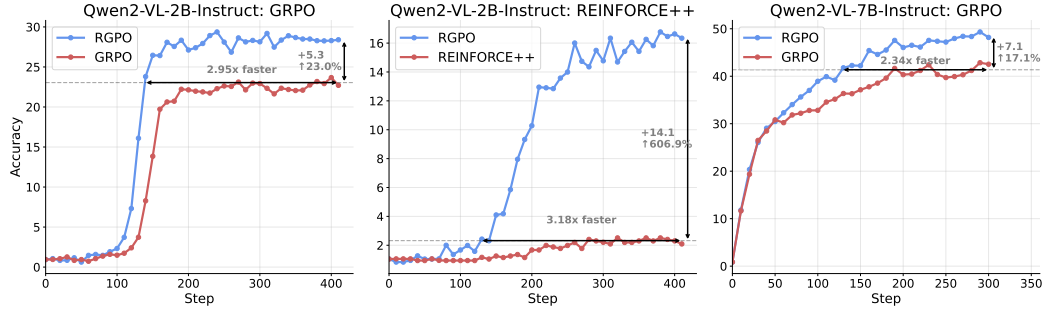


Figure 2: **RGPO accelerates and improves RLVR training.** Across Qwen2-VL-2B and Qwen2-VL-7B models, RGPO reaches the final performance of RLVR baselines substantially faster and achieves higher final accuracy. The gains are especially pronounced for weaker models and less stable RLVR algorithms (e.g. REINFORCE++).

269 stronger Qwen2-VL-7B model, both methods improve during training, but RGPO still converges
 270 faster and reaches a higher final accuracy.

271 As shown in Table 2, RGPO delivers the strongest overall performance among the compared methods.
 272 On MMStar-Math, RGPO reaches 70.1, outperforming the base Qwen2-VL-7B-Instruct model by
 273 23.7 points and surpassing other open-source multimodal reasoning models such as R1-VL-7B
 274 and Mulberry. On MathVista-Math, RGPO achieves the highest overall score of 73.65, improving
 275 substantially over the base model’s 41.11 and also outperforming strong reasoning-oriented baselines
 276 such as MM-Eureka. The gains are especially clear in the GEO, ALG, and GPS subsets, where
 277 RGPO obtains the best scores of 73.47, 73.62, and 73.89, respectively. In contrast, the gain on the
 278 Textbook Question Answering subset is relatively modest. This may be due to a small degree of
 279 domain overlap between the MINT-CoT [45] training set and this subset of MathVista, which leaves
 280 less room for additional improvement on this category.

Table 2: **Performance comparison on MMStar-Math and the mathematical subset of MathVista**
 We compare RGPO with the base Qwen2-VL-7B-Instruct model, CoT supervised fine-tuning variants,
 and representative open-source multimodal reasoning models. Dashes indicate unavailable results.

Model	MMStar-Math	MathVista-Math				
		All	GEO	ALG	GPS	TQA
LLaVA-OneVision-Qwen2-7b-ov [53]	–	67.04	69.34	67.04	69.71	58.06
InternVL2-8B [54]	66.8	62.59	62.26	62.92	62.50	62.90
InternVL2-8B-MPO [55]	–	68.52	68.87	68.91	69.71	64.52
DeepSeek-VL2 [56]	–	65.56	63.68	65.54	63.94	70.97
Qwen2.5-VL-7B-Instruct [57]	66.8	66.66	65.56	66.29	65.87	69.35
Open-R1-Multimodal [51]	59.2	54.81	52.36	54.68	53.37	59.68
R1-VL-7B [50]	68.4	69.63	68.87	69.66	69.71	69.35
Mulberry [28]	66.8	68.52	67.92	68.54	68.75	67.74
MM-Eureka [52]	–	<u>72.59</u>	<u>71.22</u>	<u>72.66</u>	<u>72.60</u>	72.58
Qwen2-VL-7B-Instruct [58]	46.4	41.11	35.85	41.57	36.54	56.45
Original Image CoT SFT	–	40.37	38.68	40.82	39.42	43.54
Bounding Box CoT SFT	–	65.56	63.21	65.54	63.94	<u>70.97</u>
Text-only CoT SFT	<u>67.6</u>	64.07	64.15	64.04	64.42	62.90
RGPO	70.1	73.65	73.47	73.62	73.89	70.35

281 Figure 3 compares RGPO with a full-hint setting, where the model is given the complete hint during
 282 training. Although the full-hint model quickly obtains high training reward, this improvement is
 283 misleading: the model can simply use the provided hint to output the correct answer with only a few
 284 tokens, without learning to perform the reasoning process independently. This behavior indicates
 285 lazy learning, where the policy exploits the availability of the hint rather than developing robust
 286 reasoning ability. The full-hint model initially produces extremely short outputs while still receiving
 287 high reward, indicating that it can bypass reasoning and directly answer from the hint. Moreover, the
 288 policy-gradient loss and gradient-norm curves show that full-hint training is less stable, with large
 289 early spikes and sharp fluctuations. In contrast, RGPO achieves more gradual reward improvement,
 290 maintains longer reasoning responses, and exhibits smoother optimization dynamics. These results
 291 suggest that always providing hints can create a shortcut that harms stable reasoning learning, while

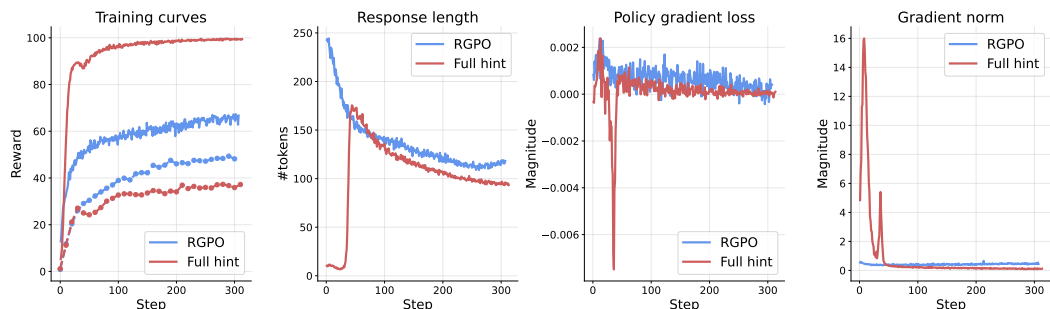


Figure 3: **Effect of failure-conditioned hinting versus always-on full hints:** Training (solid lines) and validation (dashed lines) rewards. Full hints rapidly inflate training reward but generalize poorly, while RGPO learns more gradually and achieves stronger no-hint validation performance.

292 RGPO’s failure-conditioned hinting mechanism provides guidance only when needed and avoids
 293 encouraging the model to rely on hints as a substitute for reasoning.

294 Figure 4 highlights an important
 295 interaction between prompt format,
 296 optimization objective, and exploration
 297 during reasoning training. With the
 298 concise DeepSeek-R1-style prompt, both
 299 RGPO and GRPO begin improving
 300 rapidly once the model discovers the
 301 correct answer format, but RGPO reaches
 302 a consistently higher reward plateau.
 303 At the same time, its entropy decreases
 304 smoothly and remains close to GRPO,
 305 suggesting that the performance gain
 306 does not come from simply collapsing
 307 the policy into a narrow set of outputs. In
 308 contrast, when using the more structured
 309 Mulberry-style prompt, AHPO [59],
 310 which combines off-policy expert SFT
 311 loss with RLVR, shows a delayed
 312 reward increase but undergoes a sharp
 313 entropy collapse shortly afterward. This
 314 suggests that AHPO learns to satisfy
 315 the complex template by concentrating
 316 probability mass on a limited response
 317 pattern, reducing exploration once the
 318 formatting objective is acquired. These
 319 results support the design choice of
 320 using a simpler reasoning format for RL
 321 training: it makes the reward signal easier
 322 to optimize while allowing RGPO to
 323 improve reasoning performance without
 324 prematurely sacrificing policy diversity.

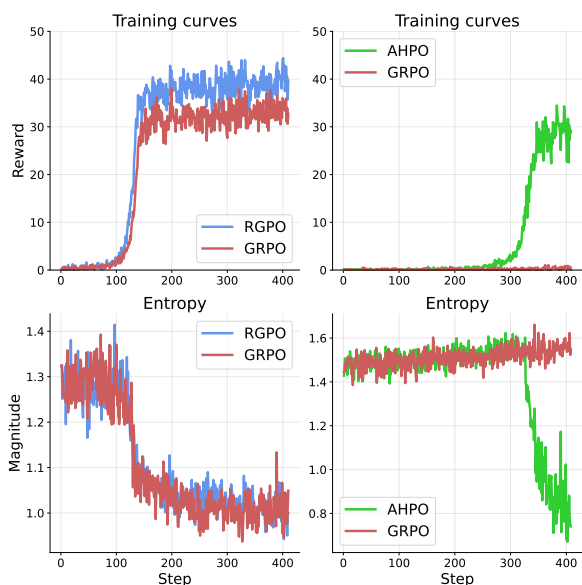


Figure 4: **Training dynamics under different reasoning prompt formats.** Left: using the concise DeepSeek-R1-style format, RGPO achieves higher reward than GRPO while maintaining a comparable entropy trajectory throughout training. Right: under the original Mulberry-style template, AHPO eventually improves reward after learning the required output structure, but this improvement is accompanied by a sharp entropy collapse, indicating reduced exploration and more deterministic generation behavior.

325 6 Conclusion

326 In this paper, we introduced Rationale-Guided Policy Optimization, an RL framework that improves
 327 reasoning by using rationales as temporary scaffolds rather than direct imitation targets. RGPO’s
 328 design reduces hint dependence by preventing the model from relying on rationales for every example,
 329 and reduces rationale memorization by avoiding direct training on the rationale text itself. Motivated
 330 by cognitive principles, RGPO treats rationales as instructional interventions that help convert failed
 331 attempts into useful learning signals. Empirically, RGPO consistently shows superior performance
 332 on both language models and vision-language models post training.

333 References

- 334 [1] Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan
335 McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card, 2025.
- 336 [2] OpenAI. Gpt-5.5 system card. <https://openai.com/index/gpt-5-5-system-card/>,
337 2026.
- 338 [3] Gemini Team. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long
339 context, and next generation agentic capabilities. *CoRR*, abs/2507.06261, 2025.
- 340 [4] Anthropic. System card: Claude opus 4 and claude sonnet 4. [https://www.anthropic.com/](https://www.anthropic.com/claude-4-system-card)
341 [claude-4-system-card](https://www.anthropic.com/claude-4-system-card), 2025. Accessed 2026-05-02.
- 342 [5] Qwen Team. Qwen3 technical report. *CoRR*, abs/2505.09388v1, 2025.
- 343 [6] Kimi Team. Kimi k2: Open agentic intelligence. *CoRR*, abs/2507.20534, 2025.
- 344 [7] GLM-4.5 Team. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *CoRR*,
345 abs/2508.06471, 2025.
- 346 [8] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal
347 policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.
- 348 [9] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li,
349 Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open
350 language models. *CoRR*, abs/2402.03300, 2024.
- 351 [10] Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang.
352 Learning to reason under off-policy guidance. *CoRR*, abs/2504.14945, 2025.
- 353 [11] Ziru Liu, Cheng Gong, Xinyu Fu, Yaofang Liu, Ran Chen, Shoubo Hu, Suiyun Zhang, Rui
354 Liu, Qingfu Zhang, and Dandan Tu. GHPO: adaptive guidance for stable and efficient LLM
355 reinforcement learning. *CoRR*, abs/2507.10628, 2025.
- 356 [12] Xuechen Zhang, Zijian Huang, Yingcong Li, Chenshun Ni, Jiasi Chen, and Samet Oymak.
357 BREAD: branched rollouts from expert anchors bridge sft rl for reasoning. In *Advances in*
358 *Neural Information Processing Systems*, 2026.
- 359 [13] Xinyi Wang, Jinyi Han, Zishang Jiang, Tingyun Li, Jiaqing Liang, Sihang Jiang, Zhaoqian Dai,
360 Shuguang Ma, Fei Yu, and Yanghua Xiao. HINT: helping ineffective rollouts navigate towards
361 effectiveness. *CoRR*, abs/2510.09388, 2025.
- 362 [14] Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans,
363 Quoc V. Le, Sergey Levine, and Yi Ma. SFT memorizes, RL generalizes: A comparative
364 study of foundation model post-training. In *Forty-second International Conference on Machine*
365 *Learning*, 2025.
- 366 [15] Manu Kapur. Productive failure. *Cognition and instruction*, 26(3):379–424, 2008.
- 367 [16] Daniel L Schwartz and John D Bransford. A time for telling. *Cognition and instruction*,
368 16(4):475–5223, 1998.
- 369 [17] Kenneth R Koedinger and Vincent Alevan. Exploring the assistance dilemma in experiments
370 with cognitive tutors. *Educational psychology review*, 19(3):239–264, 2007.
- 371 [18] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement
372 learning. *CoRR*, abs/2501.12948, 2025.
- 373 [19] Wenhao Zhang, Yuexiang Xie, Yuchang Sun, Yanxi Chen, Guoyin Wang, Yaliang Li, Bolin
374 Ding, and Jingren Zhou. On-policy RL meets off-policy experts: Harmonizing supervised
375 fine-tuning and reinforcement learning via dynamic weighting. *CoRR*, abs/2508.11408, 2025.
- 376 [20] Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang,
377 Tong Zhang, Caiming Xiong, and Hanze Dong. A minimalist approach to LLM reasoning: from
378 rejection sampling to reinforce. *CoRR*, abs/2504.11343, 2025.

- 379 [21] Jiazheng Li, Hong Lu, Kaiyue Wen, Zaiwen Yang, Jiakuan Gao, Hongzhou Lin, Yi Wu, and
380 Jingzhao Zhang. Questa: Expanding reasoning capacity in llms via question augmentation.
381 *CoRR*, abs/2507.13266, 2025.
- 382 [22] Henry L Roediger III and Jeffrey D Karpicke. Test-enhanced learning: Taking memory tests
383 improves long-term retention. *Psychological science*, 17(3):249–255, 2006.
- 384 [23] Vincent Aleven, Bruce McLaren, Ido Roll, and Kenneth Koedinger. Toward tutoring help
385 seeking: Applying cognitive modeling to meta-cognitive skills. In *International conference on*
386 *intelligent tutoring systems*, pages 227–239. Springer, 2004.
- 387 [24] John Sweller and Graham A Cooper. The use of worked examples as a substitute for problem
388 solving in learning algebra. *Cognition and instruction*, 2(1):59–89, 1985.
- 389 [25] Michelene TH Chi, Miriam Bassok, Matthew W Lewis, Peter Reimann, and Robert Glaser. Self-
390 explanations: How students study and use examples in learning to solve problems. *Cognitive*
391 *science*, 13(2):145–182, 1989.
- 392 [26] Robert K Atkinson, Alexander Renkl, and Mary Margaret Merrill. Transitioning from studying
393 examples to solving problems: Effects of self-explanation prompts and fading worked-out steps.
394 *Journal of educational psychology*, 95(4):774, 2003.
- 395 [27] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin,
396 Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton,
397 Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano,
398 Jan Leike, and Ryan Lowe. Training language models to follow instructions with human
399 feedback. *CoRR*, abs/2203.02155, 2022.
- 400 [28] Huanjin Yao, Jiaying Huang, Wenhao Wu, Jingyi Zhang, Yibo Wang, Shunyu Liu, Yingjie
401 Wang, Yuxin Song, Haocheng Feng, Li Shen, and Dacheng Tao. Mulberry: Empowering
402 MLLM with o1-like reasoning and reflection via collective monte carlo tree search. *CoRR*,
403 abs/2412.18319, 2024.
- 404 [29] Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. REINFORCE++: stabilizing critic-free
405 policy optimization with global advantage normalization. *CoRR*, abs/2501.03262, 2025.
- 406 [30] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum.
407 Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base
408 model. *arXiv preprint arXiv:2503.24290*, 2025.
- 409 [31] Zhezheng Hao, Hong Wang, Haoyang Liu, Jian Luo, Jiarui Yu, Hande Dong, Qiang Lin,
410 Can Wang, and Jiawei Chen. Rethinking entropy interventions in rlvr: An entropy change
411 perspective. *arXiv preprint arXiv:2510.10150*, 2025.
- 412 [32] Guangming Sheng, Chi Zhang, Zilinfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua
413 Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient RLHF framework. In
414 *Proceedings of the Twentieth European Conference on Computer Systems*, 2025.
- 415 [33] Qwen Team. Qwen2.5 technical report. *CoRR*, abs/2412.15115, 2024.
- 416 [34] Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan,
417 Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan
418 Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiase Chen, Jiangjie Chen,
419 Chengyi Wang, Hongli Yu, Weinan Dai, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu,
420 Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO:
421 an open-source LLM reinforcement learning system at scale. *CoRR*, abs/2503.14476, 2025.
- 422 [35] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International*
423 *Conference on Learning Representations*, 2019.
- 424 [36] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright,
425 Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Pritam Damania,
426 Bernard Nguyen, Geeta Chauhan, Yuchen Hao, Ajit Mathews, and Shen Li. Pytorch FSDP:
427 experiences on scaling fully sharded data parallel. *Proc. VLDB Endow.*, 16(12):3848–3860,
428 2023.

- 429 [37] Dhiraj D. Kalamkar, Dheevatsa Mudigere, Naveen Mellempudi, Dipankar Das, Kunal Banerjee,
430 Sasikanth Avancha, Dharma Teja Vooturi, Nataraj Jammalamadaka, Jianyu Huang, Hector Yuen,
431 Jiyan Yang, Jongsoo Park, Alexander Heinecke, Evangelos Georganas, Sudarshan Srinivasan,
432 Abhisek Kundu, Misha Smelyanskiy, Bharat Kaul, and Pradeep Dubey. A study of BFLOAT16
433 for deep learning training. *CoRR*, abs/1905.12322, 2019.
- 434 [38] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear
435 memory cost. *CoRR*, abs/1604.06174, 2016.
- 436 [39] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn
437 Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH
438 dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets
439 and Benchmarks*, 2021.
- 440 [40] Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi
441 Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun.
442 Olympiadbench: A challenging benchmark for promoting AGI with olympiad-level bilingual
443 multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association
444 for Computational Linguistics*, 2024.
- 445 [41] Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski,
446 Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu,
447 Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems
448 with language models. In *Advances in Neural Information Processing Systems*, 2022.
- 449 [42] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick,
450 and Oyvind Tafjord. Think you have solved question answering? try arc, the AI2 reasoning
451 challenge. *CoRR*, abs/1803.05457, 2018.
- 452 [43] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien
453 Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a
454 benchmark. *CoRR*, abs/2311.12022, 2023.
- 455 [44] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo,
456 Weiming Ren, Aaran Arulraj, Xuan He, Ziyang Jiang, Tianle Li, Max Ku, Kai Wang, Alex
457 Zhuang, Rongqi Fan, Xiang Yue, and Wenhui Chen. Mmlu-pro: A more robust and challenging
458 multi-task language understanding benchmark. In *Advances in Neural Information Processing
459 Systems*, 2024.
- 460 [45] Xinyan Chen, Renrui Zhang, Dongzhi Jiang, Aojun Zhou, Shilin Yan, Weifeng Lin, and
461 Hongsheng Li. Mint-cot: Enabling interleaved visual tokens in mathematical chain-of-thought
462 reasoning. *CoRR*, abs/2506.05331, 2025.
- 463 [46] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing
464 Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men,
465 Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-
466 language model’s perception of the world at any resolution. *CoRR*, abs/2409.12191, 2024.
- 467 [47] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao
468 Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical
469 reasoning of foundation models in visual contexts. In *International Conference on Learning
470 Representations (ICLR)*, 2024.
- 471 [48] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan,
472 Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large
473 vision-language models? In *Advances in Neural Information Processing Systems*, 2024.
- 474 [49] Liang Chen, Lei Li, Haozhe Zhao, Yifan Song, and Vinci. R1-v: Reinforcing super
475 generalization ability in vision-language models with less than \$3. [https://github.com/
476 Deep-Agent/R1-V](https://github.com/Deep-Agent/R1-V), 2025. Accessed: 2026-05-02.
- 477 [50] Jingyi Zhang, Jiaying Huang, Huanjin Yao, Shunyu Liu, Xikun Zhang, Shijian Lu, and Dacheng
478 Tao. R1-VL: learning to reason with multimodal large language models via step-wise group
479 relative policy optimization. *CoRR*, abs/2503.12937, 2025.

- 480 [51] EvolvingLMMs-Lab. open-r1-multimodal: A fork to add multimodal model training to open-
481 r1. <https://github.com/EvolvingLMMs-Lab/open-r1-multimodal>, 2025. Accessed:
482 2026-05-2.
- 483 [52] Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu,
484 Botian Shi, Wenhai Wang, Junjun He, Kaipeng Zhang, Ping Luo, Yu Qiao, Qiaosheng Zhang,
485 and Wenqi Shao. Mm-eureka: Exploring visual aha moment with rule-based large-scale
486 reinforcement learning. *CoRR*, abs/2503.07365, 2025.
- 487 [53] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan
488 Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer.
489 *Trans. Mach. Learn. Res.*, 2025, 2025.
- 490 [54] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong
491 Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. Intern
492 VL: scaling up vision foundation models and aligning for generic visual-linguistic tasks. In
493 *Conference on Computer Vision and Pattern Recognition*, 2024.
- 494 [55] Weiyun Wang, Zhe Chen, Wenhai Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Jinguo Zhu,
495 Xizhou Zhu, Lewei Lu, Yu Qiao, and Jifeng Dai. Enhancing the reasoning ability of multimodal
496 large language models via mixed preference optimization. *CoRR*, abs/2411.10442, 2024.
- 497 [56] Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao,
498 Yiyang Ma, Chengyue Wu, Bingxuan Wang, Zhenda Xie, Yu Wu, Kai Hu, Jiawei Wang, Yaofeng
499 Sun, Yukun Li, Yishi Piao, Kang Guan, Aixin Liu, Xin Xie, Yuxiang You, Kai Dong, Xingkai Yu,
500 Haowei Zhang, Liang Zhao, Yisong Wang, and Chong Ruan. Deepseek-vl2: Mixture-of-experts
501 vision-language models for advanced multimodal understanding. *CoRR*, abs/2412.10302, 2024.
- 502 [57] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng
503 Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Ming-Hsuan Yang, Zhaohai Li,
504 Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao
505 Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl
506 technical report. *CoRR*, abs/2502.13923, 2025.
- 507 [58] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing
508 Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men,
509 Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-
510 language model’s perception of the world at any resolution. *CoRR*, abs/2409.12191, 2024.
- 511 [59] Xiangyu Zhao, Junming Lin, Tianhao Liang, Yifan Zhou, Wenhao Chai, Yuzhe Gu, Weiyun
512 Wang, Kai Chen, Gen Luo, Wenwei Zhang, et al. Mm-helix: Boosting multimodal long-chain
513 reflective reasoning with holistic platform and adaptive hybrid policy optimization. *arXiv*
514 *preprint arXiv:2510.08540*, 2025.
- 515 [60] Kai Sun, Yushi Bai, Ji Qi, Lei Hou, and Juanzi Li. Mm-math: Advancing multimodal math
516 evaluation with process evaluation and fine-grained classification. In *Findings of the Association*
517 *for Computational Linguistics: EMNLP 2024*, pages 1358–1375, 2024.

518 Due to space constraints, some implementation details, additional analyses, and supplementary results
519 are omitted from the main paper. In this appendix, we provide a more complete description of the
520 RGPO training procedure, including the prompt templates, algorithmic details, and hyperparameter
521 settings. We also include additional experimental results and ablation studies to better understand the
522 effects of rationale guidance, prompt format, and failure-conditioned hinting on model learning and
523 final performance.

524 A Experimental Details

525 A.1 Algorithm

526 Algorithm 1 summarizes Rationale-Guided Policy Optimization. In our experiments, we set the
527 correctness threshold to $\tau_{\text{correct}} = 1.0$, requiring an unguided rollout to be perfectly correct in order
528 to be considered solved. We set the acceptance margin to $\delta = 0$, so a rationale-guided revision
529 is added to the auxiliary buffer whenever it achieves a strictly higher verifier reward than the best
530 unguided rollout. The rationale-guided loss coefficient λ_t is tuned over $10^{-2}, 10^{-3}, 10^{-4}$.

Algorithm 1 Rationale-Guided Policy Optimization

Require: Training set $\mathcal{D} = \{(x_i, s_i, a_i)\}_{i=1}^M$, policy π_θ , verifier reward r , number of hint segments
 N , rollouts per prompt K , acceptance margin δ

- 1: Partition each solution s_i into N segments $[s_{i,1}; \dots; s_{i,N}]$
- 2: Initialize hint level $g_i^{(0)} \leftarrow 0$ for all examples
- 3: **for** $t = 0$ to N **do**
- 4: Clear auxiliary buffer $\mathcal{B}_{\text{RGPO}}$
- 5: **for** each minibatch of examples **do**
- 6: Sample unguided rollouts $y_{i,1}^{(t)}, \dots, y_{i,K}^{(t)} \sim \pi_\theta(\cdot | x_i)$
- 7: Compute rewards $r_{i,k}^{(t)} = r(x_i, y_{i,k}^{(t)}, a_i)$ and advantages for the main RL objective
- 8: **if** $t > 0$ and $g_i^{(t)} > 0$ **then**
- 9: Construct hint prefix $h_i^{(t)} = [s_{i,1}; \dots; s_{i,g_i^{(t)}}]$
- 10: Construct guided prompt $\tilde{x}_i^{(t)} = \text{Reprompt}(x_i, h_i^{(t)}, f_i^{(t)})$
- 11: Sample guided revisions $\tilde{y}_{i,1}^{(t)}, \dots, \tilde{y}_{i,K}^{(t)} \sim \pi_\theta(\cdot | \tilde{x}_i^{(t)})$
- 12: Compute guided rewards $\tilde{r}_{i,k}^{(t)} = r(x_i, \tilde{y}_{i,k}^{(t)}, a_i)$
- 13: Add $(x_i, \tilde{y}_{i,k}^{(t)})$ to $\mathcal{B}_{\text{RGPO}}$ if $\tilde{r}_{i,k}^{(t)} > \max_j r_{i,j}^{(t)} + \delta$
- 14: **end if**
- 15: Update π_θ using $\mathcal{L}_{\text{RL}} + \lambda_t \mathcal{L}_{\text{SFT}}$
- 16: **end for**
- 17: **for** each example i **do**
- 18: Set $C_i^{(t)} = \mathbb{I}[\max_k r_{i,k}^{(t)} \geq \tau_{\text{correct}}]$ using unguided rollouts
- 19: **if** $C_i^{(t)} = 0$ **then**
- 20: $g_i^{(t+1)} \leftarrow \min(g_i^{(t)} + 1, N)$
- 21: **else**
- 22: $g_i^{(t+1)} \leftarrow \max(g_i^{(t)} - 1, 0)$
- 23: **end if**
- 24: **end for**
- 25: **end for**

531 A.2 Prompt templates

532 RGPO uses rationales as temporary scaffolds to elicit an improved response. To implement this
533 idea, we design a refinement prompt that conditions the model on three pieces of information:
534 the original problem, a partial solution prefix, and feedback on its previous attempt. The partial
535 solution provides a rationale guide, while the feedback signal indicates whether the previous answer
536 failed due to formatting, accuracy, or both. Importantly, the model is instructed to generate a new

537 standalone solution to the original question, rather than to comment on or revise the previous response.
538 This prevents the model from learning a conversational repair pattern and instead encourages it to
539 internalize the reasoning needed to solve the original task.

Refinement prompt

```
{prompt}
This is the (partial) solution without formatting, you can leverage
it to check your answer:
{solution}
Previous response:
{response}
Feedback for your previous response:
Format score: {format_score}/1.0
Accuracy score: {accuracy_score}/1.0
Overall score: {score}/1.0
You should maximize the overall score by improving both the format
and accuracy of your answer.
Write a new standalone solution to the original question without
mentioning the previous responses or feedback. You first think
through the reasoning process as an internal monologue, enclosed
within <think> </think> tags. Then, provide your final answer
enclosed within \boxed{}. Please strictly follow the format.
```

540

541 The refinement prompt is shown above. Here, {prompt} denotes the original question, {solution}
542 denotes the currently revealed rationale segment, {response} denotes the model’s previous unguided
543 response, and the feedback fields are obtained from the verifier reward. The instruction explicitly
544 asks the model to maximize both format and accuracy, while enforcing the same answer format used
545 during training and evaluation. This is because we found in practice that the model can sometimes
546 respond with the correct option but in the wrong format due to the long refinement-instruction prompt.
547 For the language-only experiments, we use the following prompt, closely following [13].

DAPO’s system prompt

```
You are a helpful AI assistant. A conversation takes place between the User and the Assistant.
The User asks a question, and the Assistant solves it. Please help me solve this question.
Wrap only the final answer in \boxed{ }.
```

548

549 For training vision-language models, we adopt a concise reasoning format inspired by DeepSeek-R1.
550 The model is instructed to place its reasoning inside <think> and </think> tags and to wrap the
551 final answer in \boxed{}. Compared with the Mulberry prompt used in MINT-CoT, this format is
552 simpler and imposes fewer structural constraints on the model output. In the appendix, we further
553 analyze how this prompt design affects learning dynamics and final performance.

DeepSeek-R1’s system prompt

```
You first think through the reasoning process as an internal monologue, enclosed within
<think> </think> tags. Then, provide your final answer enclosed within \boxed{ }.
```

554

555 This prompt is more concise and straightforward than the Mulberry-style prompt used in the MINT-
556 CoT dataset, which decomposes the response into multiple explicit sections, including image
557 description, rationale, step-by-step reasoning, and final answer. While such a structured format
558 may encourage detailed multimodal reasoning, it also imposes a more complex output schema.

Mulberry’s system prompt

Generate an image description based on the question.
Then, provide a rationale to analyze the question.
Next, generate a step-by-step reasoning process to solve the problem. Ensure the steps are logical and concise.
Finally, provide a concise summary of the final answer in the following format: The final answer is: xxx.
Format your response with the following sections, separated by ###:
Image Description:
Rationales:
Let’s think step by step.
Step 1:
Step 2:
...
The final answer is:

559

560 A.3 Additional experimental results

561 Figure 5 presents a category-level comparison of the base model, GRPO, and RGPO on MathVista.
562 RGPO achieves the highest overall accuracy, improving from 36.7 for the base model and 39.6 for
563 GRPO to 40.7. The gains are consistent across several reasoning categories, including geometry
564 reasoning, algebraic reasoning, geometry problem solving, and textbook question answering. The
565 improvement is especially pronounced on textbook question answering, where RGPO reaches
566 53.2, outperforming GRPO by 6.4 points and the base model by 8.0 points. This suggests that
567 rationale-guided refinement is particularly helpful for problems requiring structured interpretation
568 and multi-step reasoning. On statistical reasoning, all methods achieve perfect accuracy, while on
569 arithmetic reasoning both GRPO and RGPO improve over the base model. Overall, these results
show that RGPO provides a stronger optimization signal than standard RLVR-style training, leading to better no-hint reasoning performance across most MathVista categories.

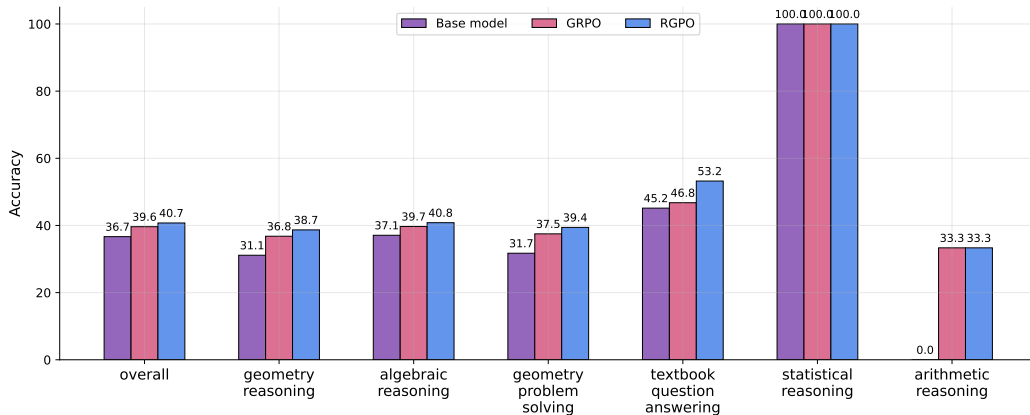


Figure 5: **Category-wise performance comparison on MathVista.** We compare the base model, GRPO, and RGPO across different mathematical reasoning categories. RGPO achieves the best overall accuracy and consistently improves over both the base model and GRPO on most categories, with the largest gain observed on textbook question answering.

571

572 Figure 6 reports the category-wise performance of the base model, GRPO, and RGPO on MMStar.
573 RGPO achieves the best overall accuracy, improving from 36.8 for the base model and 41.2 for
574 GRPO to 45.2. The largest gain appears in geometry, where RGPO reaches 34.5, outperforming
575 GRPO by 7.8 points and the base model by 10.4 points. RGPO also improves numeric commonsense
576 and calculation accuracy to 58.3, compared with 56.2 for GRPO and 50.0 for the base model. On
577 statistical reasoning, RGPO matches GRPO at 52.3, with both methods outperforming the base

578 model. These results show that RGPO provides consistent benefits on MMStar, especially for visually
 579 grounded and calculation-heavy reasoning tasks where rationale-guided refinement can help convert
 580 failed attempts into more informative training signals.

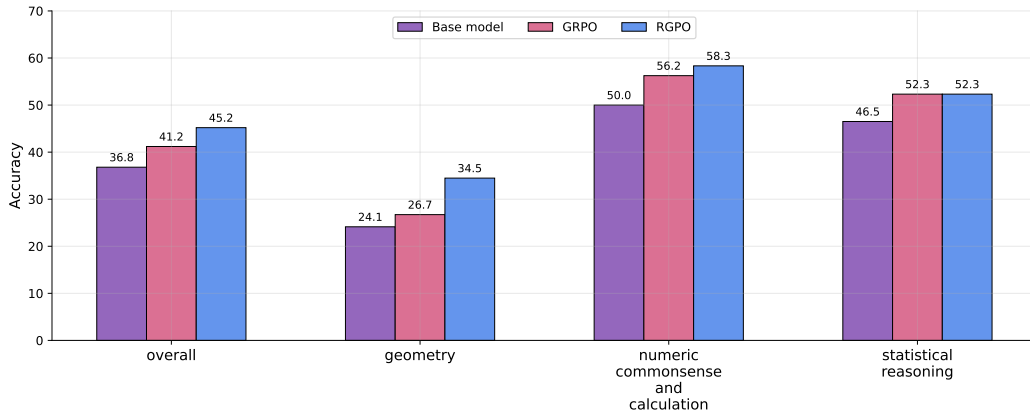


Figure 6: **Category-wise performance comparison on MMStar:** We compare the base model, GRPO, and RGPO across the mathematical reasoning categories in MMStar. RGPO achieves the highest overall accuracy and improves over both baselines on geometry and numeric commonsense/calculation tasks, while matching GRPO on statistical reasoning.

581 Figure 7 illustrates how RGPO adaptively controls the use of rationale guidance during training.
 582 In the first epoch, no hints are provided, allowing the algorithm to estimate the model’s unguided
 583 problem-solving ability. Once rationale-guided refinement is activated in the second epoch, the
 584 non-empty hint fraction rises to around 20%, indicating that only a minority of examples require
 585 scaffolded assistance. By the third epoch, this fraction drops to roughly 15%, suggesting that many
 586 examples that previously required guidance can now be solved without hints. This behavior supports
 587 the central design of RGPO: rationales are not used as always-on supervision, but as temporary
 588 scaffolds for difficult examples. As training progresses, the model becomes less dependent on external
 rationale guidance, making full-solution exposure for every sample unnecessary.

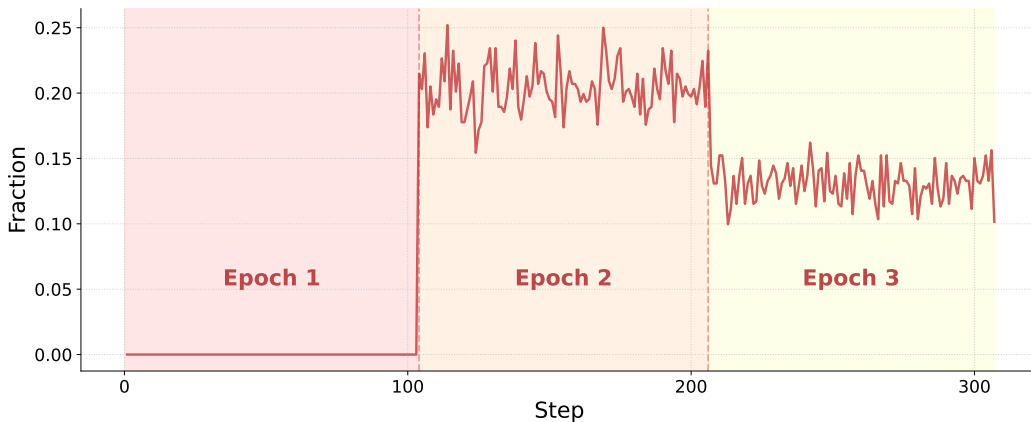


Figure 7: **Fraction of training examples receiving rationale guidance:** The hint rate remains zero during the probing epoch, rises when RGPO begins guided refinement, and decreases as the model learns to solve more examples independently.

589
 590 In Figure 8, we show that prompt format has a significant effect on both optimization and efficiency.
 591 The Mulberry-style prompt, which requires image descriptions, rationales, step-by-step sections,
 592 additional delimiters, and line breaks, produces much longer responses throughout training, typically
 593 around 380–400 tokens. Despite this higher generation cost, its reward remains nearly flat, suggesting

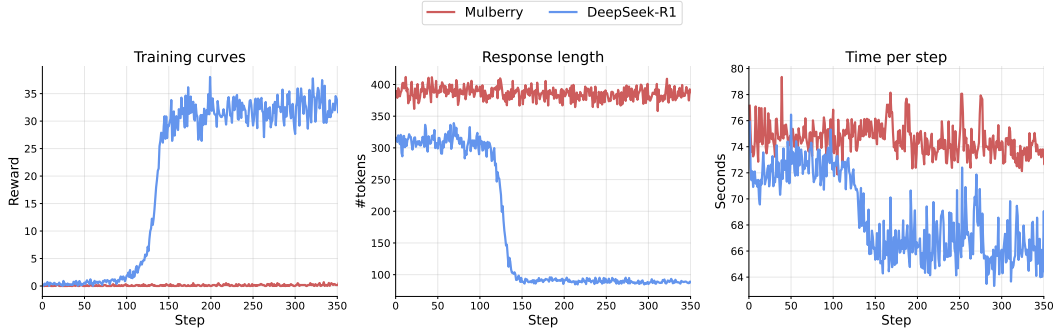


Figure 8: **Effect of prompt format on training dynamics and efficiency.** We compare the Mulberry-style structured prompt with the simpler DeepSeek-R1-style prompt in terms of reward, response length, and runtime per training step. The DeepSeek-R1-style prompt leads to substantially better learning while producing shorter responses and reducing training time.

594 that the complex output schema makes the format difficult to learn and does not translate into better
 595 task performance without additional supervision (e.g. from SFT loss in the AHPO baseline in the
 596 main paper). In contrast, the DeepSeek-R1-style prompt shows a clear reward increase after training
 597 begins to improve, while its response length drops sharply to below 100 tokens, indicating that the
 598 model learns a more concise and effective reasoning format. This reduction in response length also
 599 lowers the runtime per step, making training more efficient. These results motivate our use of the
 600 simpler DeepSeek-R1-style prompt: it reduces formatting burden, avoids unnecessary verbosity, and
 601 enables more stable learning without requiring extra supervised fine-tuning to enforce the response
 602 structure.

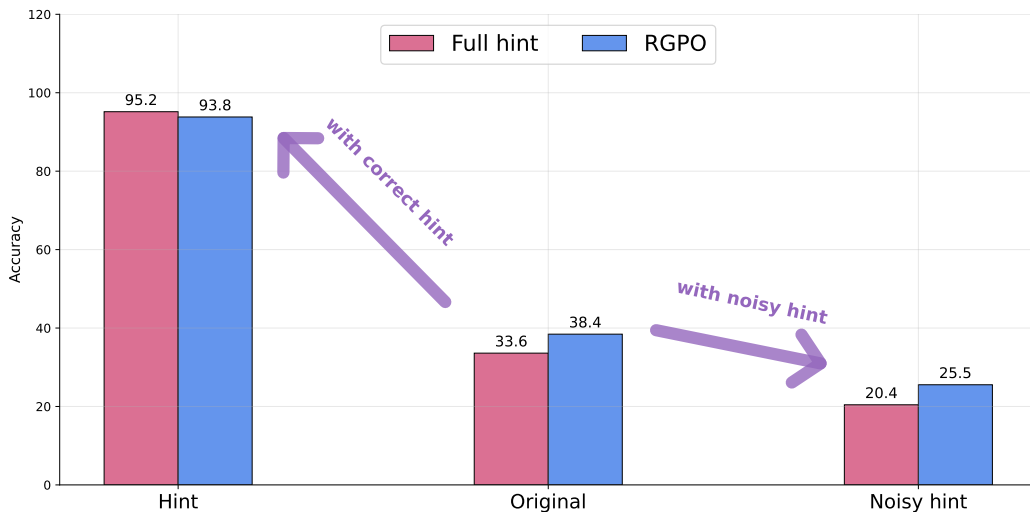


Figure 9: **Robustness to helpful and noisy rationale hints on MMMath:** We compare RGPO with a full-hint baseline under three evaluation settings: correct rationale hints, the original no-hint benchmark, and shuffled noisy hints. While the full-hint baseline performs slightly better when the correct CoT is provided, RGPO achieves higher accuracy in the original and noisy-hint settings, indicating stronger robustness and less dependence on external rationales.

603 Figure 9 evaluates whether training with rationale guidance leads to genuine reasoning improvement
 604 or merely teaches the model to rely on hints. On the left, both methods are evaluated with the correct
 605 CoT appended to the question as a rationale hint. The full-hint baseline achieves the highest accuracy,
 606 95.2, slightly outperforming RGPO at 93.8, which is expected because the baseline is optimized
 607 to maximally exploit the provided hint. However, this advantage disappears in the original no-hint
 608 setting, where RGPO improves over the full-hint baseline from 33.6 to 38.4. The difference becomes

609 more pronounced under noisy hints, where the CoT is shuffled and appended with an instruction that
 610 the model may use it if helpful. In this setting, the full-hint baseline drops to 20.4, suggesting that it
 611 tends to trust and follow the misleading rationale rather than solve the problem independently. RGPO
 612 achieves a higher score of 25.5, indicating better resistance to corrupted guidance. These results
 613 support the motivation of RGPO: rationales should serve as temporary scaffolds during training,
 614 not as permanent dependencies. By selectively using hints only after failure and optimizing for
 615 de-hinted solutions, RGPO improves no-hint reasoning while reducing over-reliance on potentially
 616 noisy rationales from MMMath [60].

617 Below, we provide qualitative examples comparing the responses generated by the base model, GRPO,
 618 and RGPO. Compared with the base model, GRPO and RGPO often yield more focused reasoning
 619 processes.

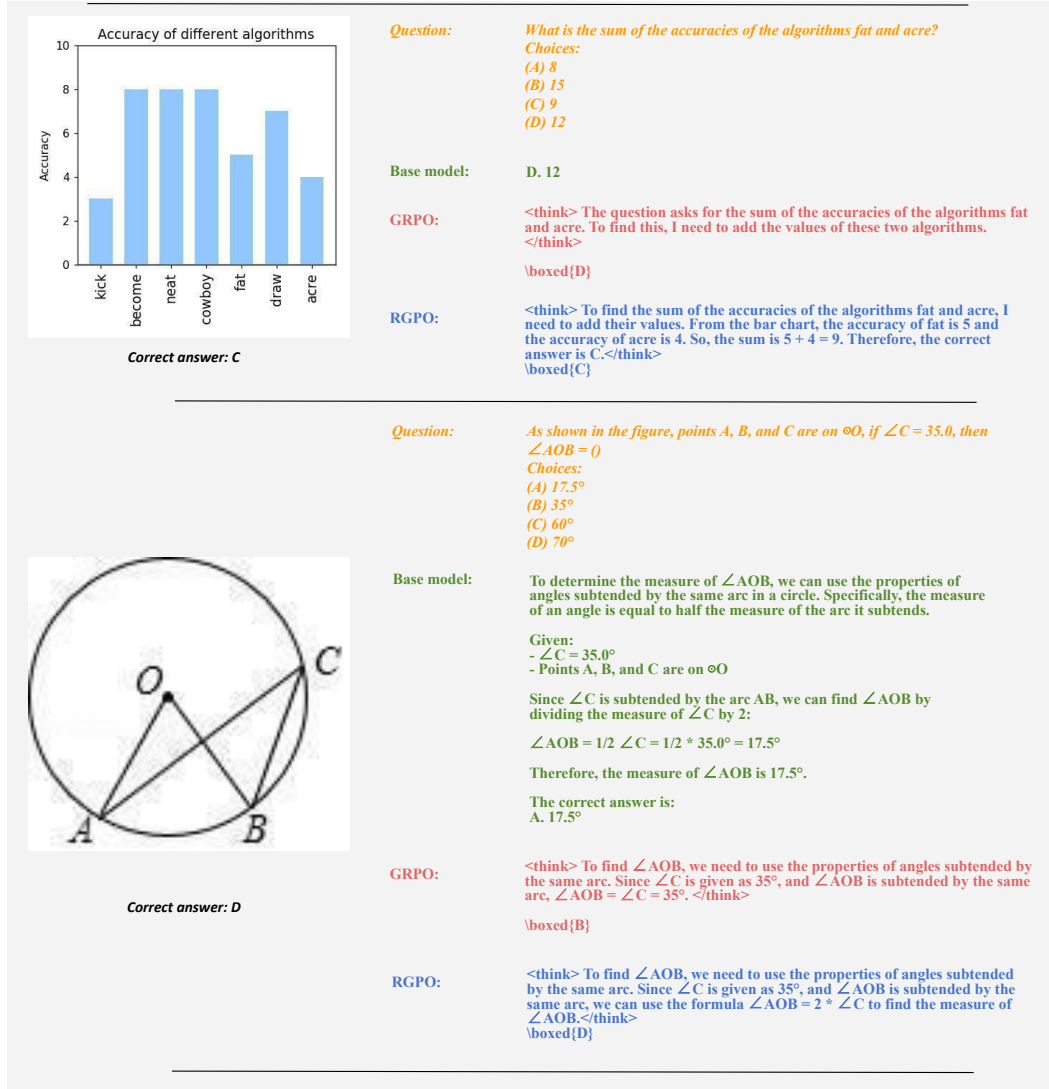


Figure 10: **Qualitative comparison of model responses:** Example outputs from the base model, GRPO, and RGPO.

620 A.4 Hardware configuration

621 All experiments were conducted on high-performance machines equipped with Intel Xeon CPUs and
 622 NVIDIA GPUs, selected to accommodate varying computational needs and optimize job priority
 623 scheduling across different tasks. Specifically, we utilized two machine configurations: (1) INTEL(R)

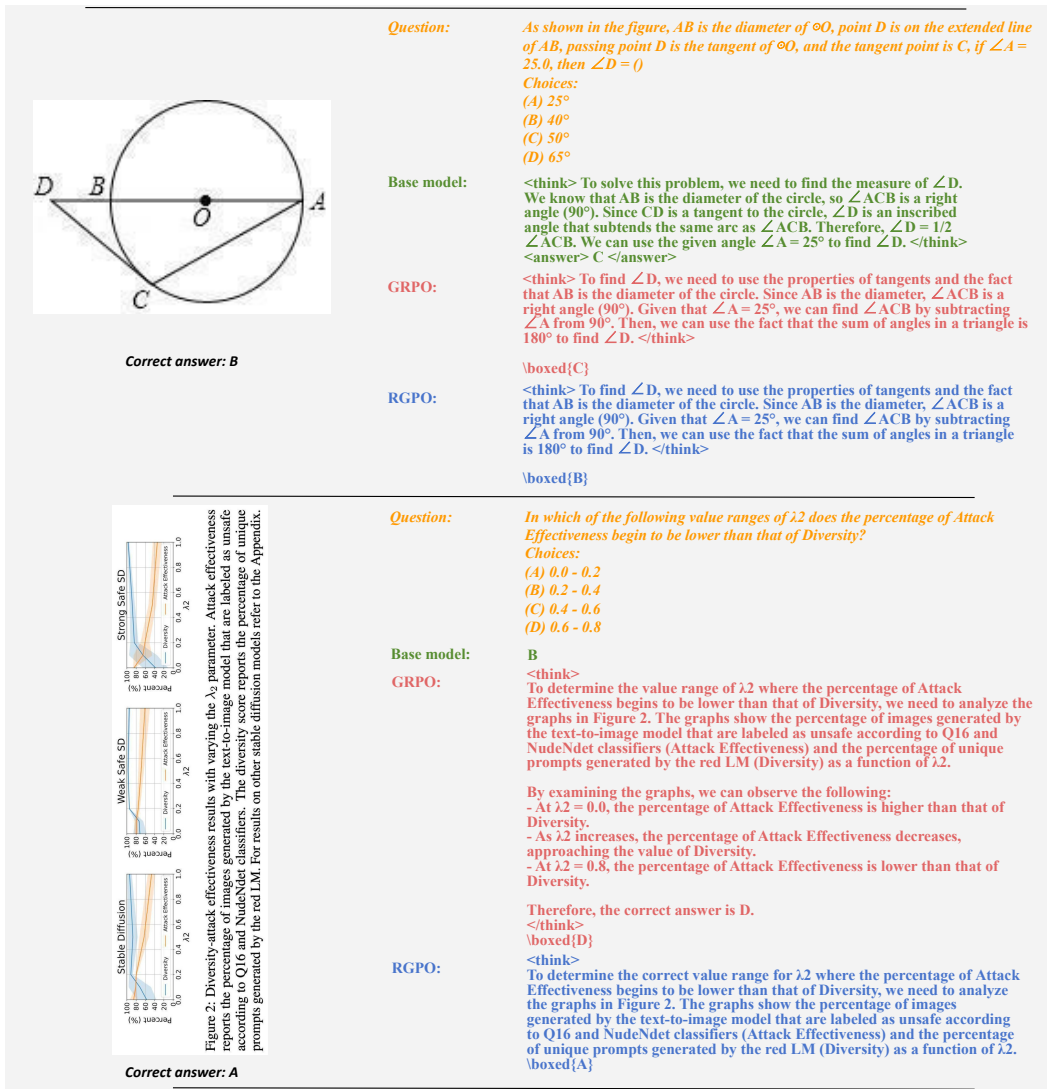


Figure 11: **Qualitative comparison of model responses:** Example outputs from the base model, GRPO, and RGPO.

624 XEON(R) PLATINUM 8592+ with 2.0Ti RAM and H200 SXM 141GB GPUs and (2) Intel(R)
 625 Xeon(R) Platinum 8358 CPU @ 2.60GHz with 503Gi RAM and NVIDIA A100 SXM4 80GB
 626 GPUs. Although different GPU types were used to balance workload priorities, we ensured that
 627 all comparisons across methods were performed on the same hardware configuration for a given
 628 model and dataset to eliminate hardware-induced variability and maintain consistency and fairness in
 629 evaluation.

630 B Limitations

631 RGPO relies on the availability of reference solutions; its effectiveness therefore depends on both the
 632 quality of those solutions and the granularity of the segmentation. If the revealed prefix contains too
 633 much answer-specific information, the method may encourage shortcut learning or memorization
 634 rather than transferable reasoning; if it contains too little or is stylistically far from the model's own
 635 reasoning distribution, it may fail to rescue hard examples from sparse-reward collapse. Moreover, the
 636 reward-gated acceptance rule only guarantees improvement under the available verifier, not genuine
 637 reasoning quality, so RGPO can still inherit verifier bias, reward hacking, or spurious formatting gains.

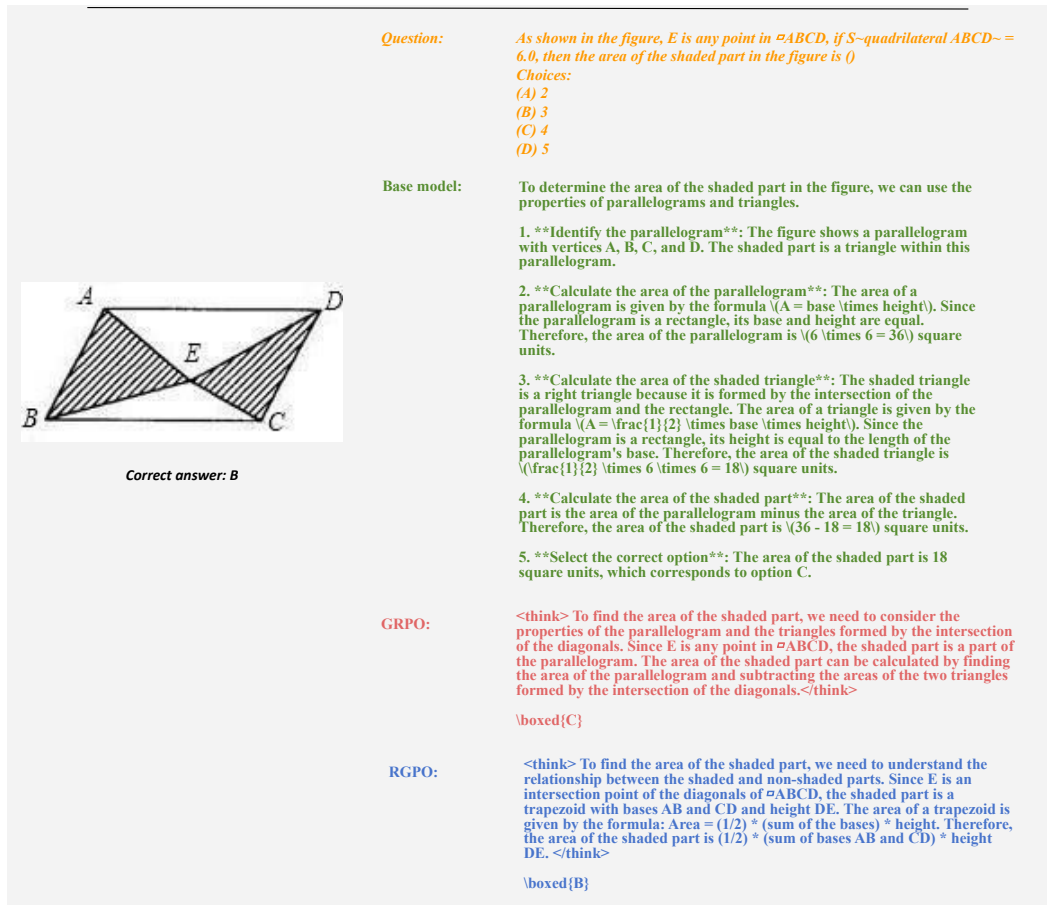


Figure 12: **Qualitative comparison of model responses:** Example outputs from the base model, GRPO, and RGPO.

638 The adaptive schedule also introduces additional compute from guided rollouts and auxiliary SFT
 639 updates, making the method more expensive than a single-pass PPO or GRPO baseline. However, we
 640 keep the number of rollouts per prompt the same as in the GRPO baseline to ensure a fair comparison.

641 C Broader Impacts

642 RGPO aims to make reasoning-oriented post-training more sample-efficient by using reference
 643 rationales as adaptive training-time scaffolds, which may help smaller language and vision-language
 644 models learn from sparse verifiable rewards without relying as heavily on large teacher models,
 645 rejection sampling, or fully formatted supervised traces; this could benefit education, scientific
 646 analysis, software assistance, mathematical problem solving, and multimodal question answering.
 647 However, stronger reasoning ability is inherently dual-use: the same improvements that help benign
 648 problem solving may also assist harmful planning, automated persuasion, misinformation generation,
 649 cyber misuse, or other strategic tasks. RGPO also improves verifier reward rather than guaranteeing
 650 faithful reasoning, so generated rationales may remain incomplete, misleading, or optimized for the
 651 reward function rather than for truth, especially when verifiers or reference solutions contain biases,
 652 shortcuts, or annotation artifacts. These risks suggest that RGPO should be paired with careful verifier
 653 design, dataset auditing, robustness and out-of-distribution evaluation, and appropriate deployment
 654 safeguards for high-capability models. The method also adds compute overhead through guided
 655 rollouts and auxiliary distillation, although this cost may be offset in some settings if adaptive hinting
 656 improves sample efficiency or enables smaller models to reach competitive reasoning performance.