OFF-CONTEXT GRPO: LEARNING TO REASON ON HARD PROBLEMS USING PRIVILEGED INFORMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards (RLVR) improves reasoning in large language models. Yet, typical RLVR approaches fail on difficult problems: when a model cannot generate any correct solutions, it receives *zero* learning signal. Providing privileged guidance during training, such as solution prefixes, can help overcome this learning cliff by steering the model towards correct solutions with non-zero reward. We call these rollouts *off-context*: they are generated from a training prompt that contains privileged guidance, while the target objective is defined by the original prompt without that guidance. We introduce Off-Context GRPO (OC-GRPO), a minimally modified variant of GRPO that uses guided rollouts but applies an importance-corrected objective to steer the update back toward the original unguided objective, avoiding the mismatch that destabilizes uncorrected guided training. Empirically, our algorithm achieves a 3.8% absolute improvement (13.7% relative gain) over vanilla GRPO on average across standard mathematical reasoning benchmarks, while adding only one forward pass per training step relative to existing guided methods.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has emerged as a powerful approach for improving reasoning in large language models (Guo et al., 2025a; Yu et al., 2025; Jaech et al., 2024). The dominant algorithm for RLVR is Group Relative Policy Optimization (GRPO) (Shao et al., 2024) which itself is based on the seminal PPO algorithm (Schulman et al., 2017). In GRPO, for each problem, we first sample a group of responses from an LLM, use a verifier to score each response, and then compute in-group advantages by normalizing rewards within the group. These advantage estimates are then used to update the LLM policy via a PPO-like clipped importance-sampling surrogate. The gradient signal in this GRPO update comes entirely from within-group reward variance, i.e., GRPO learns only when some rollouts in a group succeed and others fail, reinforcing the successful ones relative to the failures.

This requirement exposes a fundamental failure mode on hard problems. When every rollout in a group fails, rewards are uniformly zero, within-group variance collapses, and the gradient is identically zero, no matter how long training continues. We refer to this as the *learning cliff* (Guo et al., 2025b; Zhang et al., 2025d; Qu et al., 2026; Yan et al., 2025). Escaping this cliff requires at least some successes within each rollout group. One workaround is to use oracle/expert solutions directly as training targets, via supervised fine-tuning or off-policy RL (Yan et al., 2025; Song et al., 2024; Agarwal et al., 2024; Huang et al., 2025; Zhang et al., 2025a;e). This breaks the cliff by construction but at a cost: it distorts the base model's native reasoning style and requires heavy stabilization machinery such as reward shaping, entropy control, and hybrid SFT–RL objectives. A second class of methods takes a more surgical approach, using oracle information not as a training target but as *context*, injecting privileged guidance into the prompt to make correct continuations reachable, while the model still samples its own rollouts. Examples include solution-prefix conditioning (Qu et al., 2026; Zhang et al., 2025e; Li et al., 2025), output-space prefix injection (Setlur et al., 2026; Amani et al., 2025), and tiered in-prompt hints (Zhang et al., 2025d). This preserves GRPO's on-policy structure while breaking the cliff. However, all of these methods share a subtle but consequential flaw we call the *off-context* problem: rollouts are sampled under a guided prompt the model never sees at deployment, yet gradients are computed as if the sampling and evaluation distributions matched. The result is either a misaligned objective or biased gradients that can destabilize training.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

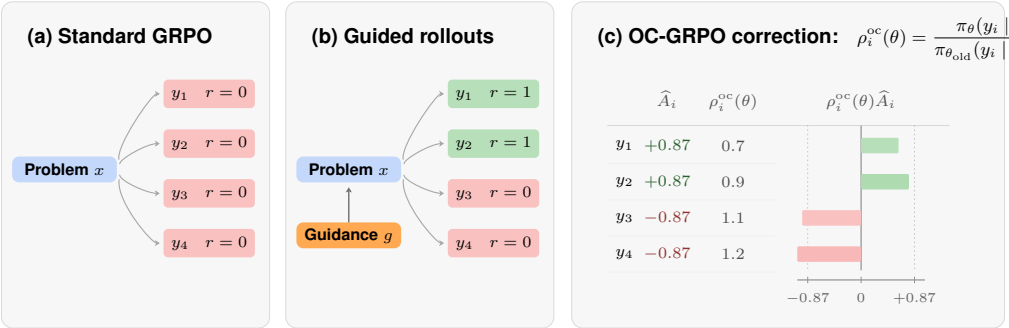

Figure 1: **Off-Context GRPO. (a)** On hard problems, unguided samples all receive reward 0, so the group-relative advantages vanish. **(b)** Guidance is supplied with the problem and produces successful rollouts, while the update still targets the unguided objective at x. **(c)** `OC-GRPO` keeps the guided rollouts and reweights their advantages with the importance weight shown in the panel. Successes made likely by guidance receive smaller positive contributions; failures receive larger negative contributions.

In this work, we propose `OC-GRPO`**: Off-context GRPO**, a minimal modification to GRPO (Figure 1). `OC-GRPO` re-weights per-token advantages by the importance ratio between the unguided and guided policies, a standard technique used for correction for off-distribution sampling (Precup et al., 2001; Degris et al., 2012). The resulting trajectory-level corrected estimator is unbiased for the original unguided objective (the implementation optimizes its standard clipped token-level surrogate, exactly as vanilla GRPO does for the on-context objective). The framework is mechanism-agnostic, applying to any form of privileged signal since what matters is the context mismatch, not its form.

Beyond fixing the objective, the importance ratio encodes a meaningful inductive bias about how much credit each trajectory generated under the guided context deserves. As illustrated in Figure 1, the correction dampens the credit assigned to correct trajectories, and the dampening is stronger for solution trajectories that rely more heavily on the guidance, limiting the influence of successes the model could not have produced on its own. We formalize this intuition in Section 5. Our contributions are as follows.

- We formalize privileged guidance in RLVR as off-context sampling and show that existing methods implicitly optimize a different objective than the one evaluated at deployment.
- We propose `OC-GRPO`, a minimal modification to GRPO that corrects for the off-context mismatch via importance sampling. The importance-corrected gradient estimator is aligned with the inference-time objective, the original reward on the unguided prompt, even though rollouts are generated with guidance.
- We show that importance sampling provides a natural credit assignment according to the model's capability (Theorem 1): when a success depends heavily on the guidance, the model receives less credit for it. The update therefore reinforces genuine problem solving rather than hint-following, without any additional reward shaping.
- We empirically validate the proposed method on multiple models and benchmarks. On Qwen2.5-7B-Instruct, `OC-GRPO` delivers a 13.7% relative gain over GRPO on math reasoning. At 3B and 1.5B scale (three seeds), `OC-GRPO-Fixed` is the only method significantly above vanilla GRPO at both scales, while none of the uncorrected guided baselines is significantly above it at either scale.

2 Preliminaries: RLVR and GRPO

RLVR trains a policy on problems with automatically checkable answers (Guo et al., 2025a). A training distribution $\mathcal{D}$ supplies problems $x \in \mathcal{X}$, and the policy samples a response $y = (y_1, \ldots, y_T)$ autoregressively from $\pi_\theta(\cdot \mid x)$, where T denotes the response length. We use the base model π_{ref} as the KL reference. A response may contain a reasoning trace, but the verifier scores only the extracted final answer: we write $a(y)$ for the answer extracted from y and $a^\star(x)$ for the correct answer to

x. The verifier returns the binary reward $r(x, y) = \mathbf{1}\{a(y) = a^\star(x)\}$. The RLVR objective is the expected verifier reward:

$$J(\theta) \;=\; \mathbb{E}_{x\sim\mathcal{D}}\,\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\big[r(x,y)\big]. \tag{1}$$

GRPO converts these binary rewards into relative advantages within a rollout group (Shao et al., 2024). For each problem x, the behavior policy $\pi_{\theta_{\text{old}}}$ generates G responses $\{y_i\}_{i=1}^G$. We write T_i for the length of y_i. Let μ_r and σ_r be the mean and standard deviation of the group rewards $\{r(x, y_i)\}_{i=1}^G$. GRPO assigns $\widehat{A}_i = (r(x, y_i) - \mu_r)/\sigma_r$. When $\sigma_r = 0$, the group contains no preference information, and the usual convention sets all advantages to zero. The policy is updated by optimizing the clipped, KL-regularized surrogate averaged over training problems:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\Bigg[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{T_i}\sum_{t=1}^{T_i}\Big\{\min\Big(\rho_{i,t}(\theta)\widehat{A}_i,\ \text{clip}\big(\rho_{i,t}(\theta), 1-\epsilon, 1+\epsilon\big)\,\widehat{A}_i\Big) - \beta\,\text{KL}_{i,t}(\theta)\Big\}\Bigg]. \tag{2}$$

The per-token PPO ratio (Schulman et al., 2017) compares the current policy with the behavior policy that produced the rollout, and the KL term penalizes drift from the reference model:

$$\rho_{i,t}(\theta) := \frac{\pi_\theta(y_{i,t} \mid x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid x, y_{i,<t})}, \qquad \text{KL}_{i,t}(\theta) := D_{\text{KL}}\big(\pi_\theta(\cdot \mid x, y_{i,<t}) \,\big\|\, \pi_{\text{ref}}(\cdot \mid x, y_{i,<t})\big). \tag{3}$$

3 Overcoming Learning Cliffs with Privileged Guidance

Challenging reasoning benchmarks often contain problems that receive almost no verifier reward under the behavior policy $\pi_{\theta_{\text{old}}}$. We call a problem x *hard* if its expected verifier reward under the behavioral policy

$$p(x) \;:=\; \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(\cdot|x)}[r(x,y)] \tag{4}$$

is close to zero. Hard problems are common in practice, and they expose a fundamental failure mode of GRPO: if every sampled response is incorrect, rewards are uniformly zero and $\sigma_r = 0$, so the group-relative advantages $\widehat{A}_i$ vanish. This means the gradient of (2) is identically zero, and nothing is learned from x. We refer to this as the *learning cliff* in RLVR.

A direct way to address the learning cliff is to use *privileged guidance*: auxiliary training-time context, unavailable at inference, that raises expected verifier reward on a hard problem x. Let g be a guidance strategy that maps each problem to a rollout prompt. It leaves non-hard problems unchanged, $g(x) = x$, and can add a solution prefix or problem-specific hint on hard problems (Appendix I). The expected reward under the guided prompt is

$$q(x) \;:=\; \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(\cdot|g(x))}[r(x,y)], \tag{5}$$

Useful guidance satisfies $q(x) > p(x)$ (see Theorem 1). Figure 2 gives an illustration of this effect for increasing solution prefixes.

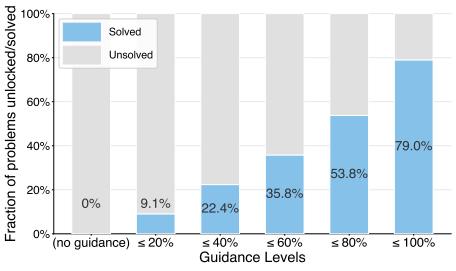

Figure 2: Solution prefixes increase empirical expected verifier reward under $\pi_{\theta_{\text{old}}}$ on hard MATH problems from 0.091 (none) to 0.790 (full prefix) (Hendrycks et al., 2021).

One way that recent methods have used privileged guidance is to sample rollouts under $g(x)$ and run GRPO as if $g(x)$ were the true task (Zhang et al., 2025d;e; Qu et al., 2026; Setlur et al., 2026; Amani et al., 2025). These guided-target methods effectively optimize

$$J^{\text{guide}}(\theta) \;:=\; \mathbb{E}_{x\sim\mathcal{D}}\,\mathbb{E}_{y\sim\pi_\theta(\cdot|g(x))}\big[r(x,y)\big], \tag{6}$$

which scores the original problem x but trains the policy under the behavior prompt $g(x)$. The corresponding per-token ratio in (3) is

$$\rho^{\text{guide}}_{i,t}(\theta) \;:=\; \frac{\pi_\theta(y_{i,t} \mid g(x), y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid g(x), y_{i,<t})}, \tag{7}$$

conditioning on $g(x)$ throughout rather than the original x for hard problems. The objective J^{guide} is therefore different from the target J because it trains the policy on responses conditioned on $g(x)$, while J evaluates responses conditioned on x. When guidance carries significant information, these two can differ substantially. We refer to this mismatch as the *off-context* problem.

Despite optimizing a misaligned objective, guided-target updates can still improve J in practice. This transfer rests on *back-generalization*, a hypothesis coined by Setlur et al. (2026) and shared by related guided-target methods (Qu et al., 2026): the rollout distributions induced by $g(x)$ and x overlap, so a policy update that improves continuations under the guided prompt also improves them under the nearby unguided prompt. Back-generalization is empirical and not guaranteed in practice: Appendix F gives a toy example where optimizing J^{guide} selects a policy that is useless under the original prompt, and our experiments in Section 6.3 suggest the transfer is capacity-dependent, with guided-target baselines degrading below vanilla GRPO at smaller model scales.

4 Off-Context GRPO

`OC-GRPO` addresses the off-context problem via a minimal modification to the GRPO update rule. Recall from Section 3 that existing methods effectively optimize $J^{\text{guide}}(\theta)$ by sampling rollouts under the guided prompt $g(x)$ instead of the original prompt x. Importance sampling provides a principled correction:

$$\mathbb{E}_{x\sim\mathcal{D},\, y\sim\pi_{\theta_{\text{old}}}(\cdot|g(x))}\left[\frac{\pi_\theta(y \mid x)}{\pi_{\theta_{\text{old}}}(y \mid g(x))}\, f(y)\right] \;=\; \mathbb{E}_{x\sim\mathcal{D},\, y\sim\pi_\theta(\cdot|x)}[f(y)].$$

This makes the reweighted guided expectation exactly equivalent to the target expectation. When $g(x) = x$ (i.e., no guidance is applied), the per-token importance ratio reduces to the standard GRPO ratio in (3), and the update is identical to standard GRPO.

In our setting, the sampling distribution is $\pi_{\theta_{\text{old}}}(\cdot \mid g(x))$, the guided context that produced the rollout, and the target distribution is $\pi_\theta(\cdot \mid x)$, the unguided prompt that defines $J(\theta)$ in (1). The appropriate per-token importance ratio for guided rollouts is therefore

$$\rho^{\text{oc}}_{i,t}(\theta) \;\coloneqq\; \frac{\pi_\theta(y_{i,t} \mid x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid g(x), y_{i,<t})}. \tag{8}$$

Remark 1. *One alternative is to mask the guidance tokens from the loss (Setlur et al., 2026). This restricts which tokens receive gradient, but does not change what they condition on. The per-token ratio remains $\pi_\theta(y_{i,t} \mid g(x), y_{i,<t}) \,/\, \pi_{\theta_{\text{old}}}(y_{i,t} \mid g(x), y_{i,<t})$, with $g(x)$ in both numerator and denominator, so the update still targets $J^{\text{guide}}(\theta)$ rather than $J(\theta)$. We verify this failure mode empirically in Section 6.*

Table 1 summarizes the contemporary landscape of approaches along three axes: guided sampling (for overcoming learning cliff), off-context correction, and effective objective. Table 1 highlights that `OC-GRPO` is the unique combination that achieves all three desiderata. Replacing $\rho_{i,t}(\theta)$ in (2) with $\rho^{\text{oc}}_{i,t}(\theta)$ from (8) gives the `OC-GRPO` surrogate:

$$\mathcal{L}_{\texttt{OC-GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{T_i}\sum_{t=1}^{T_i}\Big\{\min\Big(\rho^{\text{oc}}_{i,t}(\theta)\,\widehat{A}_i,\; \text{clip}\Big(\rho^{\text{oc}}_{i,t}(\theta),\, 1{-}\epsilon,\, 1{+}\epsilon\Big)\,\widehat{A}_i\Big) - \beta\,\text{KL}_{i,t}(\theta)\Big\}\right]. \tag{9}$$

Theory versus implementation. The importance-sampling identity above is exact at the trajectory level: the trajectory-level corrected estimator is unbiased for the gradient of the unguided objective $J(\theta)$ (Theorem 2 in Appendix E.1). As with vanilla GRPO, whose clipped token-level surrogate approximates the on-context policy gradient, (9) is the standard clipped token-level surrogate of this estimator. In all experiments, the per-token correction weights are additionally truncated to $[0.01, 1]$ and applied as a multiplier on the advantage (Appendix G). These surrogate approximations are shared by every method we compare.

Method	Guided Sampling	Off-Context Correction	Effective Objective is $J(\theta)$
Vanilla GRPO	✗	✗	✓
Guided, on-context Qu et al. (2026); Zhang et al. (2025e) Setlur et al. (2026); Amani et al. (2025)	✓	✗	✗
Guided, no correction	✓	✗	✗
OC-GRPO (ours)	✓	✓	✓

Table 1: Comparison of GRPO-based methods along three axes: whether guided sampling is used to address the learning cliff, whether the off-context shift is corrected in the update rule, and whether the resulting effective objective is aligned to the target objective. `OC-GRPO` is the only method that combines guided sampling with off-context correction, recovering $J(\theta)$ as the optimization target. Appendix F gives a toy example where guided-target optimization selects the opposite policy from $J(\theta)$; Appendix D demonstrates the instability of uncorrected guided training.

5 Theoretical Understanding

We now show that the importance correction in (8) has a more intuitive interpretation than simply to correct the off-context distribution mismatch. In fact, it induces a *provably* well-behaved, behavior-aware credit assignment mechanism, automatically adjusting the gradient contributions based on how much the guidance influenced each trajectory. For simplicity, the analysis in this subsection is response-level, where we treat a sampled completion $y = (y_1, \ldots, y_T)$ as the unit that receives verifier reward and group-relative advantage.

Let $\mathcal{Y}^+ = \{y : r(x, y) = 1\}$ and $\mathcal{Y}^- = \{y : r(x, y) = 0\}$ refer to the responses partitioned based on their correctness. Recall from (4) and (5) that, for a hard problem x,

$$p(x) = \mathbb{E}_{y \sim \pi_{\theta_{\text{old}}}(\cdot|x)}[r(x, y)] \approx 0, \qquad q(x) = \mathbb{E}_{y \sim \pi_{\theta_{\text{old}}}(\cdot|g(x))}[r(x, y)],$$

for the expected verifier rewards under the original and guided prompts respectively. For this response-level calculation, set $\theta = \theta_{\text{old}}$. We use the centered advantage

$$\widehat{A}(x, y) := r(x, y) - q(x) = \begin{cases} 1 - q(x), & y \in \mathcal{Y}^+, \\ -q(x), & y \in \mathcal{Y}^-, \end{cases}$$

without within-group standard-deviation normalization. Define the response-level correction multiplier and aggregate correction factors

$$\rho^{\text{oc}}_{\text{resp}}(y) := \frac{\pi_{\theta_{\text{old}}}(y \mid x)}{\pi_{\theta_{\text{old}}}(y \mid g(x))},$$
$$\lambda_+(x, g) := \mathbb{E}\left[\rho^{\text{oc}}_{\text{resp}}(y) \;\middle|\; y \in \mathcal{Y}^+\right], \qquad \lambda_-(x, g) := \mathbb{E}\left[\rho^{\text{oc}}_{\text{resp}}(y) \;\middle|\; y \in \mathcal{Y}^-\right].$$

The quantities $\lambda_+(x, g)$ and $\lambda_-(x, g)$ are conditional averages of the response-level correction multiplier over successful and failed guided rollouts, respectively. Pointwise, $\rho^{\text{oc}}_{\text{resp}}(y) > 1$ means the original prompt assigns higher probability to the same response y than the guided prompt does; $\rho^{\text{oc}}_{\text{resp}}(y) < 1$ means the guided prompt assigns higher probability to y; and $\rho^{\text{oc}}_{\text{resp}}(y) = 1$ means the two probabilities match. We also use the "ideal" *response-level* surrogate, without clipping or token-level averaging,

$$\mathcal{L}^{\text{resp}}_{\text{OC-GRPO}}(\theta) := \mathbb{E}_{x \sim \mathcal{D}}\left[\mathbb{E}_{y \sim \pi_{\theta_{\text{old}}}(\cdot|g(x))}\left[\frac{\pi_\theta(y \mid x)}{\pi_{\theta_{\text{old}}}(y \mid g(x))}\widehat{A}(x, y)\right]\right].$$

At $\theta = \theta_{\text{old}}$, the prefactor in the inner expectation is $\rho^{\text{oc}}_{\text{resp}}(y)$. We first isolate the aggregate effect of the correction on successful and failed guided rollouts. The proofs of Lemma 1 and Theorem 1 are deferred to Appendix A.

Lemma 1 (Aggregate correction factors)**.** *Consider a hard problem x with $0 < p(x) < q(x) < 1$, and assume $\pi_{\theta_{\text{old}}}(y \mid g(x)) > 0$ whenever $\pi_{\theta_{\text{old}}}(y \mid x) > 0$. Then, with all expectations taken over guided rollouts $y \sim \pi_{\theta_{\text{old}}}(\cdot \mid g(x))$,*

$$\lambda_+(x,g) = \frac{p(x)}{q(x)} < 1, \qquad \lambda_-(x,g) = \frac{1-p(x)}{1-q(x)} > 1.$$

Lemma 1 gives a direct credit-assignment interpretation, formalized in Theorem 1 below.

Theorem 1 (`OC-GRPO` Credit Assignment)**.** *Under the conditions of Lemma 1, the following hold:*

(i) ***Systematic credit redistribution.*** *The response-level correction suppresses expected credit on successful guided rollouts,*

$$\mathbb{E}\Big[\rho^{\text{oc}}_{\text{resp}}(y)\widehat{A}(x,y) \;\Big|\; y \in \mathcal{Y}^+\Big] < \mathbb{E}\Big[\widehat{A}(x,y) \;\Big|\; y \in \mathcal{Y}^+\Big],$$

and increases the expected penalty magnitude on failed guided rollouts,

$$\Big|\,\mathbb{E}\Big[\rho^{\text{oc}}_{\text{resp}}(y)\widehat{A}(x,y) \;\Big|\; y \in \mathcal{Y}^-\Big]\,\Big| > \Big|\,\mathbb{E}\Big[\widehat{A}(x,y) \;\Big|\; y \in \mathcal{Y}^-\Big]\,\Big|.$$

(ii) ***Gradient decomposition.*** *Define $s_x(y) := \nabla_\theta \log \pi_\theta(y \mid x)\big|_{\theta=\theta_{\text{old}}}$. The gradient of the response-level surrogate satisfies the following decomposition:*

$$\begin{aligned}
&\nabla_\theta \mathcal{L}^{\text{resp}}_{\text{OC-GRPO}}(\theta)\big|_{\theta=\theta_{\text{old}}} \\
&= \mathbb{E}_{x\sim\mathcal{D}}\Bigg[\underbrace{\lambda_+(x,g)(1-q(x))}_{\textit{success weight}} \cdot q(x) \cdot \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(\cdot|x)}\big[s_x(y) \mid y \in \mathcal{Y}^+\big] \\
&\qquad - \underbrace{\lambda_-(x,g)q(x)}_{\textit{failure weight}} \cdot (1-q(x)) \cdot \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(\cdot|x)}\big[s_x(y) \mid y \in \mathcal{Y}^-\big] \Bigg].
\end{aligned} \tag{10}$$

For hard prompts with $p(x) \approx 0$, standard unguided GRPO has little reward variation, while (10) *still produces a failure-side update whenever $q(x) > 0$ and the failure-conditioned score average is nonzero.*

Part (ii) shows where the correction from part (i) enters the gradient. The success branch is scaled by $\lambda_+(x,g) = p(x)/q(x)$, while the failure branch is scaled by $\lambda_-(x,g) = (1-p(x))/(1-q(x))$. The importance ratio therefore acts as a behavior-aware credit assignment mechanism, where the correction multiplier encodes the guidance's influence on each trajectory and automatically adjusts gradient contributions accordingly without any additional design choices. Since π_θ is conditioned only on x throughout (never on $g(x)$), the privileged information g plays no role at inference time, and the training and deployment objectives remain exactly aligned.

The following example illustrates this response-level mechanism for light and large hints.

Example 1 (Response-level credit assignment)**.** *In practice, the correction credits trajectories that solve a hard problem without leaning too heavily on the guidance. Consider the shared problem x, "Prove $|a\cdot b| \le \|a\|\,\|b\|$," together with three guidance strategies g_1, g_2, g_3.*

- ***Light hint, successful rollout.*** *The guidance says "consider $\|a-b\|^2 \ge 0$", and the response y expands $(a-b)\cdot(a-b)$, rearranges, and reaches the conclusion. The model essentially solves the problem on its own, so the guidance barely shifts the likelihood of y. The correction multiplier is close to one, retaining full credit.*
- ***Light hint, failed rollout.*** *The guidance gives a similarly light hint, but the model goes off-track without using it. Because this failing trajectory does not rely on the guidance, the same ratio is again close to one, so the full penalty is preserved.*
- ***Large hint, successful rollout.*** *The guidance gives away the solution, "$\|a-b\|^2 \ge 0 \Rightarrow a\cdot b \le \frac{1}{2}(\|a\|^2+\|b\|^2) \Rightarrow$ AM–GM $\Rightarrow$ result", which the response reproduces verbatim. Here the guidance is almost entirely responsible for y, so the correction multiplier is much smaller than one, heavily damping the credit even though the trajectory is correct.*

The correction preserves the learning signal when the model succeeds or fails on its own merits, and suppresses it when a success is inherited from the guidance.

In Appendix E.2, we show that the variance of the correction grows with the length of the guidance rather than the length of the rollout, which suggests using the shortest guidance that breaks the learning cliff.

6 Empirical Results

We evaluate OC-GRPO on math reasoning benchmarks to assess whether importance-corrected updates can reliably learn from privileged guidance. We describe the experimental setup below and present results in Section 6.3.

6.1 Implementing OC-GRPO

OC-GRPO requires three components: (i) identifying which problems need guidance and at what level, (ii) constructing the augmented training dataset, and (iii) running the importance-corrected update.

Identifying hard problems. We consider the MATH dataset (Hendrycks et al., 2021), which includes full reference solutions. A problem x is hard if π_{ref} is unable to solve it in 64 attempts. We start from all the hard problems in MATH Levels 3, 4, and 5.

Using cascaded guidance. Given a hard problem $x \in \mathcal{D}$ with reference solution $z^\star(x) = (z_1^\star, \ldots, z_{T^\star}^\star)$, we define $L = 5$ guidance levels by taking solution prefixes at fixed length fractions. Let $h_\ell(x)$ denote the prefix text and $g_\ell(x)$ denote the corresponding rollout prompt:

$$h_\ell(x) \;:=\; z^\star_{1:\lceil \alpha_\ell \cdot T^\star \rceil}(x), \qquad g_\ell(x) \;:=\; (x, h_\ell(x)), \qquad \alpha_\ell \in \{0.2,\, 0.4,\, 0.6,\, 0.8,\, 1.0\}. \tag{11}$$

Levels are strictly nested: $h_\ell(x)$ contains all the information in $h_{\ell-1}(x)$ plus an additional chunk of the reference solution. Alternative guidance families (self-hints, frontier hints) are studied in Appendix I, and the same methodology and theoretical results apply to those settings.

Fixed guidance via OC-GRPO-Fixed. OC-GRPO-Fixed constructs an augmented training dataset $\mathcal{D}_{\text{aug}}$ *before* RLVR training begins, by identifying the minimum guidance level needed for each hard problem to yield at least one correct rollout under the base model π_{ref}. For each $x \in \mathcal{D}$, we start from $\ell = 1$, sample 8 rollouts from $\pi_{\text{ref}}(\cdot \mid g_\ell(x))$; if any succeeds, record $\ell^\star(x) = \ell$ and stop; otherwise adjust to $\ell + 1$. The augmented dataset is then

$$\mathcal{D}_{\text{aug}} \;=\; \mathcal{D} \;\cup\; \big\{ g_{\ell^\star(x)}(x) : x \in \mathcal{D} \big\}.$$

The guidance level $\ell^\star(x)$ is selected once using the base model and fixed throughout training. The policy never re-selects guidance as it evolves.

Adaptive guidance via OC-GRPO-Adaptive. A natural alternative is to adjust guidance adaptively during training, using the current policy π_{θ_t} to re-evaluate which problems need guidance at each step. We refer to this as OC-GRPO-Adaptive and defer the full algorithm to Appendix C. Adaptive adjustment can in principle provide more targeted guidance as the policy improves, but incurs additional inference cost at every training step. We report results for both variants, with OC-GRPO-Fixed as our recommended default: it is simpler, faster, and easier to reproduce, while remaining empirically competitive.

6.2 Experimental Setup

We construct a hard dataset $\mathcal{D}$ from the MATH training split (Levels 3–5), retaining problems on which all 64 rollouts under π_{ref} fail; this yields 595 hard problems on which vanilla GRPO receives near-zero learning signal. We train Qwen2.5-7B-Instruct (Qwen, 2024) with LoRA adapters via veRL (Sheng et al., 2024) for 4 epochs (16 rollouts/prompt, 3 seeds). We evaluate Pass@1 and

Method	Training		AIME (1983–2026)		Gaokao2023		OmniMath		Average	
	P@1	P@16	P@1	P@16	P@1	P@16	P@1	P@16	P@1	P@16
Baselines										
Base	0.7	4.7	16.8	37.5	44.2	64.7	19.5	44.8	26.8	49.0
Vanilla GRPO (Shao et al., 2024) (ref.)	0.8±0.1	7.0±0.6	16.9±0.5	38.8±1.2	43.3±2.8	64.2±2.5	21.7±1.1	**45.9±1.2**	27.3	49.6
Before-Training Guidance										
PrefixRL* (Setlur et al., 2026)	3.8±0.6 (+383.5%)	11.8±0.8 (+68.6%)	18.3±0.3 (+8.1%)	40.8±0.6 (+5.1%)	43.2±1.3 (-0.3%)	65.2±1.7 (+1.5%)	22.4±0.6 (+3.5%)	44.9±0.1 (-2.1%)	28.0 (+2.5%)	50.3 (+1.3%)
POPE* (Qu et al., 2026)	**6.3±0.3** (+704.0%)	13.2±0.5 (+88.6%)	18.8±0.1 (+10.8%)	**41.9±1.0** (+8.0%)	47.2±1.8 (+9.0%)	67.2±1.6 (+4.6%)	23.2±0.0 (+7.0%)	45.0±0.4 (-1.8%)	29.7 (+8.8%)	**51.4** (+3.5%)
OC-GRPO-Fixed	5.9±1.5 (+657.1%)	**14.8±0.3** (+111.4%)	**19.1±0.2** (+12.6%)	41.7±0.2 (+7.4%)	**50.4±2.1** (+16.3%)	**68.2±1.0** (+6.2%)	**23.7±0.3** (+9.6%)	44.3±1.0 (-3.5%)	**31.1** (+13.7%)	**51.4** (+3.6%)
During-Training Guidance										
BREAD* (Zhang et al., 2025e)	5.6±0.2 (+617.0%)	13.8±1.2 (+97.1%)	18.5±0.3 (+9.1%)	41.6±0.8 (+7.1%)	44.2±0.2 (+2.2%)	65.1±0.5 (+1.3%)	22.8±0.2 (+5.2%)	45.5±0.4 (-0.8%)	28.5 (+4.4%)	50.7 (+2.2%)
OC-GRPO-Adaptive	2.6±0.4 (+235.7%)	10.0±0.3 (+42.9%)	17.7±0.2 (+4.3%)	39.8±0.8 (+2.6%)	50.3±0.6 (+16.1%)	67.5±0.8 (+5.1%)	23.5±0.2 (+8.3%)	45.6±0.8 (-0.5%)	30.5 (+11.6%)	51.0 (+2.7%)

Table 2: Main results on Qwen2.5-7B-Instruct. Each non-base cell reports the seed-averaged (3 seeds) mean±std on the first line and, on the second line, the relative improvement over Vanilla GRPO, which serves as the reference (ref.) for all comparisons. The Training columns report Pass@1 and Pass@16 on the 595 hard training problems. The Base row's P@1 values are estimated from the first sampled trajectory of the original evaluation run; Average is computed over AIME (1983–2026) (Veeraboina, 2023; Balunović et al., 2025), Gaokao2023 (Zhang et al., 2023), and OmniMath (Gao et al., 2024). In each column, the method with the best mean is bolded; bolding considers the mean alone, with no adjustment for seed variance, and methods with equal means are both bolded.

Pass@16[1] on AIME (1983–2026), Gaokao2023, and OmniMath. Full details of the setup are in Appendix G.

Baselines. We compare `OC-GRPO-Fixed` and `OC-GRPO-Adaptive` against four baselines[2]. Each method's optimization objective is given explicitly to clarify the comparison. **Vanilla GRPO** optimizes the unguided objective $J(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[r(x,y)]$ but receives near- zero signal because every x has $\mathrm{pass@64} = 0$ under π_{ref}. **POPE*** (Qu et al., 2026) (offline) and **BREAD*** (Zhang et al., 2025e) (online) sample from $g(x)$ *and* score under $g(x)$ in both numerator and denominator, optimizing a misaligned *guided* objective $J^{\mathrm{guide}}(\theta) := \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_\theta(\cdot|g(x))}[r(x,y)]$, rather than $J(\theta)$. Appendix F gives a toy example where the two objectives prefer opposite policies. **PrefixRL*** (Setlur et al., 2026) injects solution prefixes in the *output* space rather than the prompt, with gradients masked on the prefix tokens. Although the mechanism differs, the optimized objective is similar to those in Qu et al. (2026); Zhang et al. (2025e): the policy is trained to complete from a forced prefix endpoint, and at inference time the prefix is absent. See Appendix G for full empirical implementation details.

6.3 Results

Table 2 reports the main results on Qwen2.5-7B-Instruct. `OC-GRPO-Fixed` achieves the best average Pass@1 of 31.1, a 13.7% relative gain over vanilla GRPO and 1.3 absolute points ahead of the strongest guided baseline POPE* (+8.8%). `OC-GRPO-Adaptive` is competitive (+11.6%) but trails

[1] Pass@k is the probability that at least one of k sampled responses solves the problem. We compute it with the standard unbiased estimator (Chen et al., 2021): given $n \geq k$ samples per problem of which c are correct, $\mathrm{pass@}k = \mathbb{E}\big[1 - \binom{n-c}{k} / \binom{n}{k}\big]$; we use $n = 16$.

[2] We implement each baseline in our setting with the same $\mathcal{D}_{\mathrm{aug}}$ across all methods. We mark these baselines with a * suffix.

Table 3: Average Pass@1 over AIME, Gaokao2023, and OmniMath at smaller scales (mean±std over three seeds). $^{\uparrow}/^{\downarrow}$: the 95% paired-bootstrap confidence interval of the difference to vanilla GRPO lies entirely above/below zero (Appendix H).

Method	Qwen2.5-3B	Qwen2.5-1.5B
Vanilla GRPO (ref.)	23.91±0.48	17.71±0.16
PrefixRL*	24.39±1.03	16.93±0.63
POPE*	24.46±0.78	18.41±0.58
OC-GRPO-Fixed	**25.81±0.62**$^{\uparrow}$	**19.68±0.32**$^{\uparrow}$
BREAD*	23.04±0.43	15.87±1.54$^{\downarrow}$
OC-GRPO-Adaptive	24.81±0.43	19.62±0.67$^{\uparrow}$

Table 4: Pass@1 of `OC-GRPO-Adaptive` on Qwen2.5-7B-Instruct with guidance sources other than solution prefixes (mean±std over seeds; 8 rollouts per prompt; best checkpoint per seed by validation Pass@1). Full results in Appendix I.

Guidance	AIME	Gaokao2023	OmniMath
None (vanilla GRPO)	16.8±0.3	46.1±0.7	22.6±0.2
Cascading hints	**17.9±0.2**	**49.9±0.1**	**23.2±0.2**
Self-correction	17.3±0.2	46.4±2.1	22.7±0.6
Frontier hints	17.4±0.5	48.5±1.4	23.0±0.6

its fixed counterpart with additional inference cost per training step. Across individual benchmarks, `OC-GRPO-Fixed` leads on AIME, Gaokao2023, and OmniMath.

Smaller scales. Table 3 reports results at 3B and 1.5B scale with identical hyperparameters, averaged over three seeds; per-seed results, paired confidence intervals, and per-benchmark results are in Appendix H. `OC-GRPO-Fixed` is the strongest method at both scales, improving the average Pass@1 over vanilla GRPO by 1.9 points at 3B ($+7.9\%$) and 2.0 points at 1.5B ($+11.1\%$), and it is the only method whose improvement over vanilla GRPO is significant at both scales. In contrast, none of the guided baselines that optimize a misaligned objective (Qu et al., 2026; Zhang et al., 2025e; Setlur et al., 2026) is significantly above vanilla GRPO at either scale, and BREAD* falls significantly below it at 1.5B. The gains from uncorrected guidance thus shrink or vanish at smaller scales, while the importance-corrected update retains them.

Beyond solution prefixes. The correction does not depend on the form of the guidance. Table 4 excerpts results where `OC-GRPO-Adaptive` replaces solution prefixes with cascading hints generated from the model's failed attempts (and, at higher levels, the reference solution), with self-correction (feedback on failed attempts only, without the reference solution), or with hints from an external frontier model. Appendix I defines these families and reports full results (Algorithm 2; prompts in Appendix J.2). All three match or exceed vanilla GRPO's mean Pass@1 on each benchmark, with cascading hints giving the largest gains (e.g., $+3.8$ points on Gaokao2023).

7 Conclusion

We identified the off-context problem in guided RLVR, where existing methods use privileged guidance to restore learning signal on hard problems but inadvertently optimize a different objective from the one used at deployment. `OC-GRPO` corrects for this via a minimal per-token importance reweighting, so training can sample from guided prompts while still estimating the unguided objective. The same correction also gives behavior-aware credit assignment: it discounts trajectories that leaned heavily on guidance and preserves the penalty for failures that remain under guidance. At smaller model scales, the gains from uncorrected guidance shrink or vanish, while the importance-corrected update retains a significant improvement over vanilla GRPO.

A natural extension is multi-turn or agentic RLVR, where privileged context can take the form of tool-call results, intermediate environment states, or oracle subgoals accessible during training but absent at deployment. The off-context problem appears whenever the training-time context differs from the deployment-time context, and the per-token importance correction in `OC-GRPO` generalizes directly. The broader lesson is that privileged information can guide exploration, but the update still has to account for the distribution it came from.

AI USE STATEMENT

We used generative AI tools for the following tasks that require disclosure: implementing and debugging parts of the training and evaluation code; assisting with proofs by refining them and checking them for errors; and formulating the argument in Appendix C that triggering guidance only upon failure preserves the per-problem direction of the expected update. We also used generative AI tools to draft and edit parts of the text, to improve readability, and to format the LaTeX source and references. The model-generated hints studied in Appendix I are part of the method under study and are described there. The authors reviewed all AI-assisted work, tested the AI-assisted code, and verified all claims and proofs. We take full responsibility for the content of this paper, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide anonymized source code for training and evaluation, including the configuration files for every method and model scale, as supplementary material. Section 6.1 and Appendix G describe the construction of the hard training set, the guidance levels, hyperparameters, seeds, hardware, and compute accounting. Appendix C and Algorithms 1 and 2 specify the adaptive variants, and Appendices J.1 and J.2 list all prompt templates. For the theoretical results, the assumptions are stated with each result, and complete proofs are given in Appendices A, E.1, and E.2. All models (Qwen2.5-Instruct) and datasets (MATH, AIME, Gaokao2023, and OmniMath) are publicly available.

ETHICS STATEMENT

This work studies reinforcement learning algorithms for mathematical reasoning on publicly available models and benchmarks. It involves no human subjects, personal data, or new data collection, and we do not foresee risks beyond those associated with general improvements in language-model reasoning.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*, 2024.

Mohammad Hossein Amani, Aryo Lotfi, Nicolas Mario Baldwin, Samy Bengio, Mehrdad Farajtabar, Emmanuel Abbe, and Robert West. Rl for reasoning by adaptively revealing rationales. *arXiv preprint arXiv:2506.18110*, 2025.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions, February 2025. URL `https://matharena.ai/`.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. Self-evolving curriculum for llm reasoning. *arXiv preprint arXiv:2505.14970*, 2025.

Thomas Degris, Martha White, and Richard S Sutton. Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*, 2012.

Liang Ding. Agenther: Hindsight experience replay for llm agent trajectory relabeling. *arXiv preprint arXiv:2603.21357*, 2026.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. Omni-math: A universal olympiad level mathematic benchmark for large language models. *arXiv preprint arXiv:2410.07985*, 2024.

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

Yongxin Guo, Wenbo Deng, Zhenglin Cheng, and Xiaoying Tang. Ggrpo-a: Guided group relative policy optimization with adaptive guidance. *arXiv preprint arXiv:2508.13023*, 2025b.

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. *NeurIPS*, 2021.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Michael Y Hu, Benjamin Van Durme, Jacob Andreas, and Harsh Jhamtani. Sample-efficient online learning in lm agents via hindsight trajectory rewriting. *arXiv preprint arXiv:2510.10304*, 2025.

Zeyu Huang, Tianhao Cheng, Zihan Qiu, Zili Wang, Yinghui Xu, Edoardo M Ponti, and Ivan Titov. Blending supervised and reinforcement fine-tuning with prefix sampling. *arXiv preprint arXiv:2507.01679*, 2025.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, et al. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

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

Guochao Jiang, Wenfeng Feng, Guofeng Quan, Chuzhan Hao, Yuewei Zhang, Guohua Liu, and Hao Wang. Vcrl: Variance-based curriculum reinforcement learning for large language models. *arXiv preprint arXiv:2509.19803*, 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Renda Li, Hailang Huang, Fei Wei, Feng Xiong, Yong Wang, and Xiangxiang Chu. Adacurl: Adaptive curriculum reinforcement learning with invalid sample mitigation and historical revisiting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 23123–23131, 2026.

Ziheng Li, Zexu Sun, Jinman Zhao, Erxue Min, Yongcheng Zeng, Hui Wu, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Xu Chen, et al. Staying in the sweet spot: Responsive reasoning evolution via capability-adaptive hint scaffolding. *arXiv preprint arXiv:2509.06923*, 2025.

Doina Precup, Richard S Sutton, and Sanjoy Dasgupta. Off-policy temporal-difference learning with function approximation. In *ICML*, pp. 417–424, 2001.

Yuxiao Qu, Amrith Setlur, Virginia Smith, Ruslan Salakhutdinov, and Aviral Kumar. Pope: Learning to reason on hard problems via privileged on-policy exploration. *arXiv preprint arXiv:2601.18779*, 2026.

Qwen. Qwen2.5: A party of foundation models, September 2024. URL `https://qwenlm.github.io/blog/qwen2.5/`.

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

Amrith Setlur, Zijian Wang, Andrew Cohen, Paria Rashidinejad, and Sang Michael Xie. Reuse your flops: Scaling rl on hard problems by conditioning on very off-policy prefixes. *arXiv preprint arXiv:2601.18795*, 2026.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, et al. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv:2312.06585*, 2023.

Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. Trial and error: Exploration-based trajectory optimization of llm agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7584–7600, 2024.

Alex Stein, Furong Huang, and Tom Goldstein. Gates: Self-distillation under privileged context with consensus gating. *arXiv preprint arXiv:2602.20574*, 2026.

Hemish Veeraboina. AIME problem set 1983–2024. `https://www.kaggle.com/datasets/hemishveeraboina/aime-problem-set-1983-2024`, 2023.

Zhenting Wang, Guofeng Cui, Yu-Jhe Li, Kun Wan, and Wentian Zhao. Dump: Automated distribution-level curriculum learning for rl-based llm post-training. *arXiv preprint arXiv:2504.09710*, 2025.

Yuanda Xu, Hejian Sang, Zhengze Zhou, Ran He, Zhipeng Wang, and Alborz Geramifard. Beyond grpo and on-policy distillation: An empirical sparse-to-dense reward principle for language-model post-training. *arXiv preprint arXiv:2605.12483*, 2026a.

Yuanda Xu, Zhengze Zhou, Kayhan Behdin, Jelena Markovic-Voronov, Hejian Sang, Xiaomin Li, Wenhui Zhu, Xinchen Du, Aida Rahmattalabi, Ran He, Sen Na, Zhipeng Wang, and Alborz Geramifard. Trek: Distill to explore, reinforce to refine. *arXiv preprint arXiv:2607.05339*, 2026b.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*, 2025.

Zhicheng Yang, Zhijiang Guo, Yinya Huang, Yongxin Wang, Dongchun Xie, Hanhui Li, Yiwei Wang, Xiaodan Liang, and Jing Tang. Depth-breadth synergy in rlvr: Unlocking llm reasoning gains with adaptive exploration. *arXiv preprint arXiv:2508.13755*, 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. *arXiv preprint arXiv:2503.14476*, 2025.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

Runzhe Zhan, Yafu Li, Zhi Wang, Xiaoye Qu, Dongrui Liu, Jing Shao, Derek F Wong, and Yu Cheng. Exgrpo: Learning to reason from experience. *arXiv preprint arXiv:2510.02245*, 2025.

Kaiyi Zhang, Ang Lv, Jinpeng Li, Yongbo Wang, Feng Wang, Haoyuan Hu, and Rui Yan. Stephint: Multi-level stepwise hints enhance reinforcement learning to reason. *arXiv preprint arXiv:2507.02841*, 2025a.

Shijie Zhang, Guohao Sun, Kevin Zhang, Xiang Guo, and Rujun Guo. Clpo: Curriculum learning meets policy optimization for llm reasoning. *arXiv preprint arXiv:2509.25004*, 2025b.

Wenhao Zhang, Yuexiang Xie, Yuchang Sun, Yanxi Chen, Guoyin Wang, Yaliang Li, Bolin Ding, and Jingren Zhou. On-policy rl meets off-policy experts: Harmonizing supervised fine-tuning and reinforcement learning via dynamic weighting. *arXiv preprint arXiv:2508.11408*, 2025c.

Xiaotian Zhang, Chunyang Li, Yi Zong, Zhengyu Ying, Liang He, and Xipeng Qiu. Evaluating the performance of large language models on gaokao benchmark. *arXiv preprint arXiv:2305.12474*, 2023.

Xichen Zhang, Sitong Wu, Yinghao Zhu, Haoru Tan, Shaozuo Yu, Ziyi He, and Jiaya Jia. Scaf-grpo: Scaffolded group relative policy optimization for enhancing llm reasoning. *arXiv preprint arXiv:2510.19807*, 2025d.

Xuechen Zhang, Zijian Huang, Yingcong Li, Chenshun Ni, Jiasi Chen, and Samet Oymak. Bread: Branched rollouts from expert anchors bridge sft & rl for reasoning. *arXiv preprint arXiv:2506.17211*, 2025e.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Limitations. OC-GRPO is evaluated at academic scale: we fine-tune Qwen2.5-Instruct models (1.5B–7B) with LoRA adapters on a small A100 cluster. We do not study full fine-tuning, and all comparisons with baselines are within this single model family, so behavior on other model families, on reasoning-specialized models, and at frontier scale remains to be tested. Our experiments target mathematical reasoning with verifiable answer rewards, and the gains may not transfer uniformly to other RLVR settings such as code generation, agentic tool-use, or multimodal reasoning. Finally, while the off-context framework is mechanism-agnostic and we explore hint-style variants in Appendix I, our main results rely on ground-truth solution prefixes as the privileged signal; behavior under partial, noisy, or adversarial guidance is left for future work.

A Missing Proofs

A.1 Proof of Lemma 1

Proof. Write $P(y) = \pi_{\theta_{\text{old}}}(y \mid x)$ and $Q(y) = \pi_{\theta_{\text{old}}}(y \mid g(x))$. The support assumption lets us restrict sums to responses with positive guided probability without dropping any response with positive original-prompt probability. Let $\mathcal{Y}_g^+ = \{y \in \mathcal{Y}^+ : Q(y) > 0\}$. Then

$$\begin{aligned}
\lambda_+(x,g) &= \mathbb{E}\Big[\rho^{\text{oc}}_{\text{resp}}(y) \;\Big|\; y \in \mathcal{Y}^+\Big] \\
&= \sum_{y \in \mathcal{Y}_g^+} \frac{Q(y)}{\sum_{z \in \mathcal{Y}^+} Q(z)} \frac{P(y)}{Q(y)} \\
&= \frac{\sum_{y \in \mathcal{Y}_g^+} P(y)}{\sum_{z \in \mathcal{Y}^+} Q(z)} = \frac{\sum_{y \in \mathcal{Y}^+} P(y)}{\sum_{z \in \mathcal{Y}^+} Q(z)} \\
&= \frac{\Pr_{y \sim \pi_{\theta_{\text{old}}}(\cdot|x)}(r(x,y)=1)}{\Pr_{y \sim \pi_{\theta_{\text{old}}}(\cdot|g(x))}(r(x,y)=1)} = \frac{p(x)}{q(x)}.
\end{aligned}$$

Since $p(x) < q(x)$, this ratio is less than one. The same calculation on $\mathcal{Y}^-$ gives

$$\lambda_-(x,g) = \frac{\sum_{y \in \mathcal{Y}^-} P(y)}{\sum_{y \in \mathcal{Y}^-} Q(y)} = \frac{1-p(x)}{1-q(x)} > 1,$$

where the strict inequality again follows from $p(x) < q(x)$. □

A.2 Proof of Theorem 1

Proof. We prove the two parts in order. By Lemma 1, $\lambda_+(x,g) < 1 < \lambda_-(x,g)$. On $\mathcal{Y}^+$, the centered advantage is constant and equal to $1 - q(x)$, so $\mathbb{E}[\widehat{A}(x,y) \mid y \in \mathcal{Y}^+] = 1 - q(x) > 0$. Hence

$$\begin{aligned}
\mathbb{E}\Big[\rho^{\text{oc}}_{\text{resp}}(y)\widehat{A}(x,y) \;\Big|\; y \in \mathcal{Y}^+\Big] &= (1-q(x))\mathbb{E}\Big[\rho^{\text{oc}}_{\text{resp}}(y) \;\Big|\; y \in \mathcal{Y}^+\Big] \\
&= \lambda_+(x,g)\mathbb{E}\Big[\widehat{A}(x,y) \;\Big|\; y \in \mathcal{Y}^+\Big] \\
&< \mathbb{E}\Big[\widehat{A}(x,y) \;\Big|\; y \in \mathcal{Y}^+\Big].
\end{aligned}$$

On $\mathcal{Y}^-$, the same calculation uses $\widehat{A}(x,y) = -q(x)$ and $\lambda_-(x,g) > 1$, which gives

$$\begin{aligned}
\Big|\,\mathbb{E}\Big[\rho^{\text{oc}}_{\text{resp}}(y)\widehat{A}(x,y) \;\Big|\; y \in \mathcal{Y}^-\Big]\,\Big| &= \lambda_-(x,g)\Big|\,\mathbb{E}\Big[\widehat{A}(x,y) \;\Big|\; y \in \mathcal{Y}^-\Big]\,\Big| \\
&> \Big|\,\mathbb{E}\Big[\widehat{A}(x,y) \;\Big|\; y \in \mathcal{Y}^-\Big]\,\Big|.
\end{aligned}$$

For the second part, fix x and write $P(y) = \pi_{\theta_{\text{old}}}(y \mid x)$ and $Q(y) = \pi_{\theta_{\text{old}}}(y \mid g(x))$. Differentiating $\mathcal{L}^{\text{resp}}_{\text{OC-GRPO}}$ and evaluating at $\theta = \theta_{\text{old}}$ gives

$$\nabla_\theta \mathcal{L}^{\text{resp}}_{\text{OC-GRPO}}(\theta)\big|_{\theta=\theta_{\text{old}}} = \mathbb{E}_{x \sim \mathcal{D}}\left[\mathbb{E}_{y \sim Q}\Big[\rho^{\text{oc}}_{\text{resp}}(y)\widehat{A}(x,y)s_x(y)\Big]\right].$$

Because $\widehat{A}(x,y) = 1 - q(x)$ on $\mathcal{Y}^+$ and $\widehat{A}(x,y) = -q(x)$ on $\mathcal{Y}^-$, the law of total expectation gives

$$\begin{aligned}\mathbb{E}_{y\sim Q}\left[\rho^{\mathrm{oc}}_{\mathrm{resp}}(y)\widehat{A}(x,y)s_x(y)\right] &= q(x)(1-q(x))\mathbb{E}_{y\sim Q}\left[\rho^{\mathrm{oc}}_{\mathrm{resp}}(y)s_x(y) \;\middle|\; y\in\mathcal{Y}^+\right] \\ &\quad - q(x)(1-q(x))\mathbb{E}_{y\sim Q}\left[\rho^{\mathrm{oc}}_{\mathrm{resp}}(y)s_x(y) \;\middle|\; y\in\mathcal{Y}^-\right].\end{aligned}$$

For any $S \in \{\mathcal{Y}^+, \mathcal{Y}^-\}$, the correction changes the conditional law from Q to P:

$$\begin{aligned}\mathbb{E}_{y\sim Q}\left[\rho^{\mathrm{oc}}_{\mathrm{resp}}(y)s_x(y) \;\middle|\; y\in S\right] &= \sum_{y\in S}\frac{Q(y)}{\sum_{z\in S}Q(z)}\frac{P(y)}{Q(y)}s_x(y) \\ &= \frac{\sum_{z\in S}P(z)}{\sum_{z\in S}Q(z)}\mathbb{E}_{y\sim P}\left[s_x(y) \;\middle|\; y\in S\right].\end{aligned}$$

Taking $S = \mathcal{Y}^+$ gives the factor $p(x)/q(x) = \lambda_+(x,g)$, and taking $S = \mathcal{Y}^-$ gives the factor $(1-p(x))/(1-q(x)) = \lambda_-(x,g)$. Substitution gives (10). □

B Extended Related Works

B.1 Self-Distillation in RLVR

A complementary line of work uses the model itself, conditioned on richer information, as a self-teacher. **OPSD** (Zhao et al., 2026) instantiates a single LLM as both privileged teacher (conditioned on ground-truth answers) and unprivileged student (conditioned only on the problem), training via per-token KL/JSD divergence on student rollouts. **SDPO** (Hübotter et al., 2026) extends this to rich-feedback settings such as code execution traces, treating the feedback-conditioned policy as a self-teacher. **GATES** (Stein et al., 2026) applies a similar idea with consensus gating to filter unreliable supervision in document-grounded question answering. Conceptual ancestors include **STaR** (Zelikman et al., 2022), which bootstraps rationales conditioned on the correct answer, and on-policy distillation (Agarwal et al., 2024). In the same distillation line, Xu et al. (2026a) study how scarce labeled data should be allocated across a pipeline that runs sparse-reward RL on a large teacher and then compresses the result into a smaller student via forward-KL warmup and on-policy distillation; the question addressed there is data allocation across model scales and reward densities for compression, rather than learning on problems where the policy receives no reward at all, and no correction of the training-deployment distribution mismatch is involved.

OPSD and SDPO share with OC-GRPO the use of training-time-only privileged context, but the methods differ in three essential ways. *First, they are distillation methods, not RL methods*: training minimizes a divergence to the teacher's logit distribution rather than a verifier reward, and correctness is never directly consulted during the update. *Second, the two paradigms inject privileged context at different points in the pipeline*: OPSD and SDPO sample rollouts from the unguided student and use privileged information only to shape the teacher's per-token distribution, whereas OC-GRPO uses the privileged context to drive the rollout distribution itself and reserves the unguided context for the importance correction. SDPO is flexible in the kinds of feedback the self-teacher can consume; neither method, however, specifically targets the learning-cliff regime, and without further algorithmic modifications both rely in this regime on teacher signal computed over student-sampled traces that are typically incorrect, whereas OC-GRPO's guided rollouts more frequently produce reward-bearing trajectories. *Third, OC-GRPO's trajectory-level corrected estimator is provably unbiased for the gradient of the original unguided RLVR objective* $J(\theta)$, while OPSD and SDPO optimize a distillation divergence that is structurally distinct from $J(\theta)$. The two paradigms are complementary rather than competing, distillation provides dense token-level supervision when the teacher distribution is reliable, RL provides reward-anchored supervision when it is not.

B.2 Hybrid SFT and RL

CHORD (Zhang et al., 2025c) reframes SFT as a dynamically weighted auxiliary objective inside on-policy RL, with a global coefficient and a token-wise weighting function balancing imitation against exploration. **Prefix-RFT** (Huang et al., 2025) blends supervised demonstrations with RFT by sampling prefixes from offline traces as starting points for on-policy continuation, scheduled

via a cosine decay over prefix length. **ReST-EM** (Singh et al., 2023) iterates between rejection-sampling rollouts and SFT on filtered successes, viewing the procedure as expectation-maximization. **TREK** (Xu et al., 2026b) is a recent stage-wise instantiation of this family targeting hard prompts: for problems with low unaided pass rates, it collects verified trajectories from a proposal source (an external teacher, or the model itself given additional inference-time context), filters them by student likelihood, distills them into the policy with a brief forward-KL phase, and then returns to standard on-policy GRPO. These approaches harmonize the two losses; OC-GRPO instead harmonizes the two distributions: guided rollouts provide the exploration that SFT-RL hybrids would obtain through expert demonstrations, while importance correction plays the role that the SFT loss plays in those methods, namely keeping training aligned with the original deployment objective. The trade-off between the two designs is explicit in TREK's staging: a forward-KL phase can pull in solution modes that lie entirely outside the current policy's support, where an importance correction has little leverage, but during that phase the optimized objective is a divergence to the proposals rather than the deployment objective, and the correctness of the final policy rests on the subsequent uncorrected RL stage; OC-GRPO instead optimizes a single, deployment-aligned objective throughout training.

B.3 Self-Curriculum and Adaptive Difficulty

A separate body of work treats problem difficulty as a curriculum variable controlled across training. Methods in this family, including **SEC** (Chen et al., 2025), **VCRL** (Jiang et al., 2025), **CLPO** (Zhang et al., 2025b), **AdaCuRL** (Li et al., 2026), **DUMP** (Wang et al., 2025), and **DARS** (Yang et al., 2025), adjust which problems the policy trains on or how much rollout budget each receives, often using rollout reward variance, advantage magnitude, or learning progress as control signals. We list these works for completeness; OC-GRPO is not a curriculum method and operates in a different regime. Curriculum approaches assume that the policy can learn from at least some problems on its own and select among them; we target the regime where every remaining hard problem has $\text{PASS@}k \approx 0$ without guidance, so privileged guidance is the lever that makes such problems trainable in the first place, before any curriculum question arises.

B.4 Sparse-Reward RL: Experience Reuse and Hindsight Relabeling

Hindsight experience replay (HER) (Andrychowicz et al., 2017) is the canonical sparse-reward technique: failed trajectories are relabeled with the goals they actually achieved, converting zero-reward experience into useful learning signal. Recent work lifts this idea to language settings: **AgentHER** (Ding, 2026) relabels failed multi-step agent trajectories with hindsight goals via a four-stage pipeline, and Hu et al. (2025) extend experience replay to LM agents through hindsight trajectory rewriting. **ExGRPO** (Zhan et al., 2025) maintains a replay buffer of partially correct rollouts organized by correctness, prioritizes valuable experiences via correctness and entropy signals, and blends replayed and on-policy data through a mixed-policy objective. HER and OC-GRPO are mirror images: HER changes the goal post-hoc to match the trajectory, while OC-GRPO changes the prompt pre-hoc to make the trajectory reachable, then importance-corrects the gradient back to the original goal. ExGRPO and replay methods are orthogonal to OC-GRPO, they reuse trajectories the policy did sample, rather than altering the sampling distribution to obtain new ones.

C `OC-GRPO-Adaptive`: Adaptive Guidance Adjustment

`OC-GRPO-Adaptive` (adaptive augmentation). At each training step t, given a batch $\mathcal{B} \subset \mathcal{D}$:

1. **Try unguided first.** For each $x \in \mathcal{B}$, sample N rollouts from $\pi_{\theta_t}(\cdot \mid x)$. If at least one is correct, keep the unguided group for this problem.
2. **Adjust guidance if all rollouts fail.** If all N unguided rollouts fail, form the guided prompt $g_1(x)$ and sample N new rollouts from $\pi_{\theta_t}(\cdot \mid g_1(x))$. Continue adjusting through levels $\ell = 2, 3, 4, 5$ until at least one rollout is correct, and use that group for the update.
3. **Proceed with the policy update.** The batch now contains a mix of unguided groups (for easier problems at the current policy) and guided groups (for problems that are currently too hard).

OC-GRPO-Adaptive differs from OC-GRPO-Fixed in how the guidance level is selected: the fixed variant fixes $\ell^\star(x)$ once using the base model π_{ref}, while the adaptive variant re-selects guidance at every training step with respect to the current policy π_{θ_t}. Adaptive adjustment selects a behavior prompt $b \in \{x, g_\ell(x)\}$, where ℓ is the guidance level that first produces a correct rollout, and the importance ratio uses the realized behavior prompt b of the retained group in its denominator. OC-GRPO-Adaptive pays an inference cost at every training step: when the unguided rollouts fail, the model must re-sample N rollouts for each guidance level, potentially up to L times per problem. In contrast, OC-GRPO-Fixed pays this cost once with π_{ref} before training, and $\mathcal{D}_{\text{aug}}$ can be reused across runs. Empirically, OC-GRPO-Fixed is competitive with or stronger than OC-GRPO-Adaptive (Table 2), so we recommend the fixed variant as the default.

Triggering guidance only upon failure. In OC-GRPO-Adaptive, whether a problem receives guidance depends on the outcome of its unguided rollouts, and a guided group is retained only if it contains a success. This does not change the direction of the expected trajectory-level update for any problem, for three reasons. First, every guided group is sampled afresh after the decision to escalate, so, conditional on reaching level ℓ, its rollouts are i.i.d. draws from $\pi_{\theta_t}(\cdot \mid g_\ell(x))$ (for hints built from a failed attempt, the same holds conditional on that attempt). Second, a group in which every rollout fails has zero group-relative advantage and contributes no gradient, so retaining only groups that contain a success does not change the expected update. Third, with binary rewards and full support, the importance ratio maps the expected score over guided successes (failures) to the expected score over unguided successes (failures), so each level contributes a non-negative multiple of $\nabla_\theta J(\theta; x)$. Consequently, for each problem x, the expected trajectory-level update equals $\kappa(x)\,\nabla_\theta J(\theta; x)$ for some $\kappa(x) \geq 0$ determined by how often the policy fails with and without guidance. Adaptive triggering therefore reweights problems, placing more weight on problems the policy currently fails, but never moves the update toward $J^{\text{guide}}(\theta)$; the standard-deviation normalization of vanilla GRPO reweights problems in the same way. The argument concerns the trajectory-level estimator; like every method we compare, the implementation uses the clipped token-level surrogate.

D Collapse of Off-Context GRPO without IS Correction

As discussed in the main text, naively using guided rollouts $y_i \sim \pi_{\theta_{\text{old}}}(\cdot \mid g(x))$ while updating as if they came from the original prompt x gives a biased gradient estimator. We call this the "Masked no-IC" update. Useful guidance changes the rollout distribution, and no-IC credits guided trajectories as though they were unguided samples from the target objective. During training, this repeated mismatch can accumulate, inflating the gradient norm early in training and eventually causing reward collapse, as shown in Figure 3.

The source of the bias is the off-context mismatch itself. If guidance makes a correct response much more likely under $g(x)$ than under x, then no-IC assigns too much credit to the unguided policy for a trajectory produced by the guided rollout distribution. Masking the guidance tokens does not fix this, since the generated response still came from the guided rollout distribution. OC-GRPO removes this mismatch by weighting each trajectory by the ratio between the original-prompt probability and the guided-prompt probability, so the corrected estimator targets $J(\theta)$ by Theorem 2.

E Additional Theoretical Results

E.1 Response-Level Unbiasedness

The exact unbiasedness statement is a trajectory-level change of measure. It is separate from the credit-assignment identities in Theorem 1.

Theorem 2 (Trajectory-level unbiasedness of the importance correction)**.** *Fix a problem x and assume the support condition $\pi_{\theta_{\text{old}}}(y \mid g(x)) > 0$ whenever $\pi_\theta(y \mid x) r(x, y) > 0$. For a guided rollout $y \sim \pi_{\theta_{\text{old}}}(\cdot \mid g(x))$, define the trajectory-level estimator*

$$\widehat{G}_{\text{oc}}(\theta; x, y) := \frac{\pi_\theta(y \mid x)}{\pi_{\theta_{\text{old}}}(y \mid g(x))} r(x, y) \nabla_\theta \log \pi_\theta(y \mid x).$$

Then

$$\mathbb{E}_{y \sim \pi_{\theta_{\text{old}}}(\cdot \mid g(x))}\big[\widehat{G}_{\text{oc}}(\theta; x, y)\big] = \mathbb{E}_{y \sim \pi_\theta(\cdot \mid x)}\big[r(x, y) \nabla_\theta \log \pi_\theta(y \mid x)\big] = \nabla_\theta J(\theta; x).$$

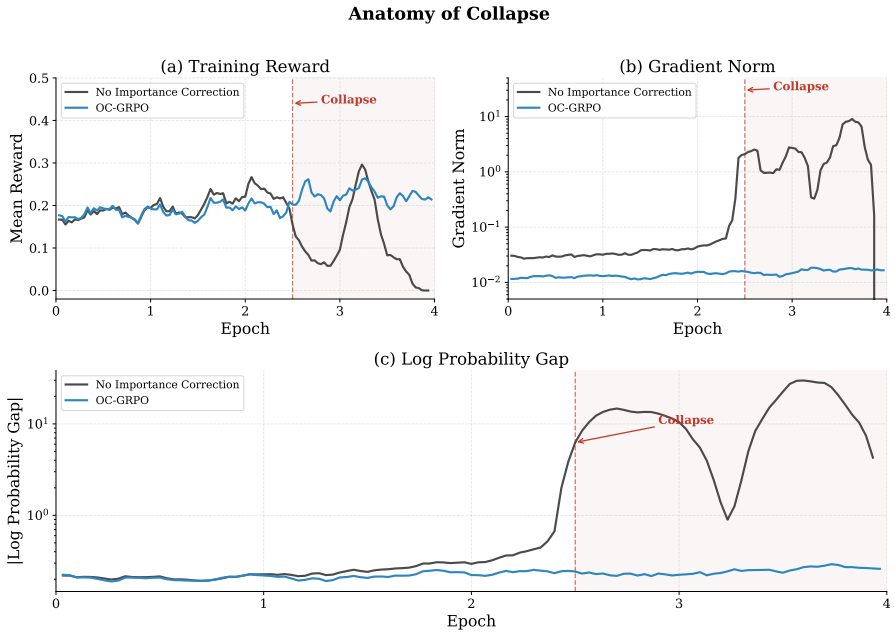

Figure 3: **Importance correction prevents collapse.** Without the importance ratio correction, guided rollouts are scored as if they came from the unguided context, producing growing objective mismatch. This leads to reward collapse, gradient norm spikes, and a widening log-probability gap. `OC-GRPO` keeps the update aligned with the unguided objective.

Proof. By direct substitution,

$$\begin{aligned}\mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(\cdot|g(x))}\left[\widehat{G}_{\text{oc}}(\theta;x,y)\right] &= \sum_y \pi_{\theta_{\text{old}}}(y \mid g(x))\frac{\pi_\theta(y \mid x)}{\pi_{\theta_{\text{old}}}(y \mid g(x))} r(x,y)\nabla_\theta \log \pi_\theta(y \mid x) \\ &= \sum_y \pi_\theta(y \mid x) r(x,y)\nabla_\theta \log \pi_\theta(y \mid x) \\ &= \nabla_\theta \sum_y \pi_\theta(y \mid x) r(x,y) = \nabla_\theta J(\theta;x).\end{aligned}$$

The support condition makes the ratio well-defined on every reward-bearing response that can contribute to the target expectation. □

E.2 Bounded Off-Context Correction Term

A practical concern with importance sampling is variance inflation, since large IS weights can destabilize training. The `OC-GRPO`-specific component of the importance ratio is bounded with controlled variance. Decompose the per-token importance ratio as

$$\rho^{\text{oc}}_{i,t}(\theta) = \underbrace{\frac{\pi_\theta(y_{i,t} \mid x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid x, y_{i,<t})}}_{\rho_{i,t}(\theta)\ \text{(on-policy drift)}} \cdot \underbrace{\frac{\pi_{\theta_{\text{old}}}(y_{i,t} \mid x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid g(x), y_{i,<t})}}_{\gamma_{i,t}\ \text{(off-context correction)}}. \tag{12}$$

The factor $\rho_{i,t}(\theta)$ is the standard GRPO drift ratio, controlled by clipping in $[1-\epsilon, 1+\epsilon]$. The `OC-GRPO`-specific factor is $\gamma_{i,t}$, the context correction.

We use the following regularity assumption.

Assumption 1 (Per-guidance-token regularity)**.** *There exists $\eta > 0$ such that for every guidance token index $k \in \{1, \ldots, n\}$ ($g = u_{1:n}$), the inequality*

$$\left|\log \frac{\pi_{\theta_{\text{old}}}(y_{i,t} \mid x, u_{1:k-1}, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid x, u_{1:k}, y_{i,<t})}\right| \le \eta$$

holds almost surely over the realized token $y_{i,t} \sim Q_{i,t}$ and the rollout history $y_{i,<t}$.

Assumption 1 states that adding any single guidance token shifts the old policy's next-token log-probability by at most η in either direction. For a well-calibrated language model on natural-language input, this is small: each additional context token refines but does not radically reshape the next-token distribution.

Theorem 3 (Bounded off-context correction term)**.** *Under Assumption 1, for every (i, t), every history $y_{i,<t}$, and every realized token $y_{i,t}$:*

1. (Deterministic bound) $e^{-n\eta} \le \gamma_{i,t} \le e^{n\eta}$.
2. (Rényi-2 variance bound) $\mathbb{E}_{Q_{i,t}}[\gamma_{i,t}^2 \mid y_{i,<t}] = e^{D_2(P_{i,t} \,\|\, Q_{i,t})} \le e^{2n\eta}$, *hence* $\mathrm{Var}_{Q_{i,t}}[\gamma_{i,t} \mid y_{i,<t}] \le e^{2n\eta} - 1$,

where n is the number of tokens in the guidance text added by $g(x)$, $Q_{i,t} = \pi_{\theta_{\text{old}}}(\cdot \mid g(x), y_{i,<t})$, and $P_{i,t} = \pi_{\theta_{\text{old}}}(\cdot \mid x, y_{i,<t})$.

Remark 2 (Utility of Theorem 3)**.** *The bound is a worst-case certificate and is loose in absolute terms, since $e^{2n\eta} - 1$ grows exponentially in n. Its useful content is the scaling: the variance of the correction is controlled by the length n of the guidance, not by the length T of the rollout, because the guidance enters only through the conditioning context. This suggests using the shortest guidance that restores learning signal, which is how guidance levels are chosen in our cascade (Section 6.1): each problem receives the smallest prefix under which at least one rollout succeeds. Empirically, the realized correction is mild: the mean per-token correction weight is about 0.94 throughout training (Appendix G).*

Proof. **(i)** By a telescoping decomposition,

$$\log \gamma_{i,t} \;=\; \sum_{k=1}^{n} \log \frac{\pi_{\theta_{\text{old}}}(y_{i,t} \mid x, u_{1:k-1}, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid x, u_{1:k}, y_{i,<t})}.$$

By Assumption 1 each summand is in $[-\eta, \eta]$, so $|\log \gamma_{i,t}| \le n\eta$. Exponentiating gives $e^{-n\eta} \le \gamma_{i,t} \le e^{n\eta}$.

(ii) Recall that for $P \ll Q$, the exponentiated Rényi-2 divergence is $e^{D_2(P\|Q)} = \sum_v P(v)^2/Q(v) = 1 + \chi^2(P\|Q)$. With $P_{i,t} = \pi_{\theta_{\text{old}}}(\cdot \mid x, y_{i,<t})$ and $Q_{i,t} = \pi_{\theta_{\text{old}}}(\cdot \mid g(x), y_{i,<t})$, the second moment under $Q_{i,t}$ is

$$\mathbb{E}_{Q_{i,t}}[\gamma_{i,t}^2 \mid y_{i,<t}] = \sum_v Q_{i,t}(v)\big(P_{i,t}(v)/Q_{i,t}(v)\big)^2 = \sum_v P_{i,t}(v)^2/Q_{i,t}(v) = e^{D_2(P_{i,t}\|Q_{i,t})}.$$

By the deterministic bound from (i), $\gamma_{i,t} \le e^{n\eta}$ almost surely under $Q_{i,t}$, hence $\mathbb{E}_{Q_{i,t}}[\gamma_{i,t}^2] \le e^{2n\eta}$. Finally, since $\gamma_{i,t} = P_{i,t}/Q_{i,t}$ is a likelihood ratio under $Q_{i,t}$, $\mathbb{E}_{Q_{i,t}}[\gamma_{i,t}] = 1$, so $\mathrm{Var}_{Q_{i,t}}[\gamma_{i,t}] = \mathbb{E}_{Q_{i,t}}[\gamma_{i,t}^2] - 1 \le e^{2n\eta} - 1$. □

F Toy Example: Guided-Target Misalignment

Consider a single problem x with guided version $g(x)$, and suppose the policy chooses between two reasoning modes, SHORTCUT and ROBUST, with SHORTCUT chosen with probability $\theta \in [0, 1]$. The verifier always evaluates the original problem x. The conditioning on x or $g(x)$ below only specifies which prompt generated the response. Assume

$$\Pr[r = 1 \mid x, \textsc{shortcut}] = 0, \quad \Pr[r = 1 \mid g(x), \textsc{shortcut}] = 1,$$
$$\Pr[r = 1 \mid x, \textsc{robust}] = \beta, \quad \Pr[r = 1 \mid g(x), \textsc{robust}] = \beta,$$

for some $\beta \in (0, 1)$. The target objective $J(\theta)$ is the expected verifier reward when the response is generated from the original prompt x, while the guided objective $J^{\text{guide}}(\theta)$ is the expected verifier reward when the response is generated from the guided prompt $g(x)$. Since the policy selects SHORTCUT with probability θ and ROBUST with probability $1 - \theta$, we have:

$$J(\theta) = \theta \cdot 0 + (1 - \theta)\beta = (1 - \theta)\beta, \qquad J^{\text{guide}}(\theta) = \theta \cdot 1 + (1 - \theta)\beta = \theta + (1 - \theta)\beta.$$

Therefore $\arg\max_\theta J(\theta) = 0$ and $\arg\max_\theta J^{\text{guide}}(\theta) = 1$. The two objectives prefer extreme opposite policies. Furthermore, $J^{\text{guide}}(\theta) - J(\theta) = \theta \ge 0$, so the guided objective is uniformly larger but rewards a deployment- useless behavior.

G Experimental Details

Dataset construction. We construct our training set from the MATH dataset (Hendrycks et al., 2021), restricting to problems at difficulty Levels 3–5 (intermediate through advanced), which yields 5,586 problems from the training split. We identify problems on which the base model π_{ref} receives no learning signal under standard GRPO. For each problem x, we generate $M = 64$ independent rollouts from the model under training using nucleus sampling , and retain only those problems for which *every* rollout fails. This filter yields **595** hard problems for the 7B model. Similarly, for the 3B and 1.5B models, we repeat this filtering process to identify model-specific hard problems. For fixed guidance methods (including our `OC-GRPO-Fixed`), we construct an augmented training set $\mathcal{D}_{\text{aug}}$ as described in Section 6. For adaptive guidance methods (including `OC-GRPO-Adaptive`), no dataset augmentation is performed.

RL training configuration. All models are trained with Group Relative Policy Optimization (GRPO) (Shao et al., 2024) and algorithms derived from GRPO implemented in veRL (Sheng et al., 2024), using vLLM (Kwon et al., 2023) for rollout generation. We fine-tuned with LoRA (Hu et al., 2022) adapters (rank 64, $\alpha = 128$, dropout 0.05) applied to all linear layers. The base model weights are frozen; only LoRA parameters are updated. We use AdamW with learning rate 1×10^{-5}, weight decay 0.01, and gradient clipping at norm 1.0. Training runs for 4 epochs on the training set. The batch size is 32 prompts with $N = 16$ rollouts per prompt, PPO mini-batch size 32, and 1 PPO epoch per GRPO iteration. We apply the clipped surrogate objective with clip ratio $\epsilon = 0.2$, remove the KL penalty ($\beta = 0$) following recent RLVR practice (Zhang et al., 2025d; Qu et al., 2026; Yu et al., 2025), and use no entropy bonus. Losses are aggregated at the token level. Importance correction is applied at the token level, with per-token correction weights truncated to $[0.01, 1]$ in all experiments; the realized correction is mild: the mean per-token weight is ≈ 0.94, and the per-step mean stays within $[0.91, 0.96]$ throughout training. Training rollouts use temperature 0.7, top-$p = 0.95$. We use a binary verifiable reward: 1 if the model's extracted answer (from `\boxed{}`) matches the ground-truth answer under symbolic equivalence, and 0 otherwise.

Fair comparison across methods. For any given model, all methods in our experiments are *epoch-matched*: each is trained for 4 epochs over the same training datasets: $\mathcal{D}_{\text{aug}}$ (for the fixed-guidance methods) and $\mathcal{D}$ (for the adaptive-guidance methods).

Infrastructure. Adaptive-guidance runs use $4\times$ NVIDIA A100 40GB GPUs with FSDP (CPU parameter and optimizer offloading) and tensor parallelism 2 for vLLM rollouts. Fixed-guidance and vanilla runs use $2\times$ A100 GPUs with tensor parallelism 1. vLLM GPU memory utilization is set to 0.5–0.6.

Seeds. All 7B model configurations are trained with 3 random seeds (0, 123, 456), and results are reported as mean $\pm$ standard deviation across seeds.

Baseline methods. We compare our methods, `OC-GRPO-Fixed` and `OC-GRPO-Adaptive`, against four baselines: vanilla GRPO, POPE (Qu et al., 2026), PrefixRL (Setlur et al., 2026), and BREAD (Zhang et al., 2025e). Note that all the guided baselines have some version of filtering for hard problems as well. For a fair and uniform comparison we implement all of them under the same training dataset setting as ours. We re-implement the core algorithmic idea of each method within our veRL training loop, using the same base model, LoRA configuration, optimizer settings, and evaluation protocol as our own methods, so that differences in final performance reflect algorithmic choices rather than infrastructure differences.

Metrics. For each checkpoint and benchmark, we generate 16 trajectories per problem and report pass@1 as the primary metric, with pass@16 reported alongside to assess the breadth of the solution distribution. Pass@k is computed using the unbiased estimator formula for the 7B results in Table 2. For the smaller-scale results (Appendix H) and the guidance-source ablation (Appendix I), Pass@1 is estimated from the first sampled trajectory per problem.

Model selection. Checkpoints are evaluated at each epoch boundary. For each seed, we select the fourth epoch checkpoint and report that checkpoint's results across all benchmarks. We selected 4

checkpoints for training as we observed the training gains on held out validation taper beyond the fourth checkpoint.

G.1 Compute Accounting

All methods use the identical per-step budget: 32 prompts $\times$ 16 rollouts. The comparison then splits into two internally compute-matched blocks. The before-training guidance family (POPE*, PrefixRL*, and `OC-GRPO-Fixed`) trains on the identical augmented dataset (at 7B: $536 \rightarrow 960$ rows, since only the 424 unlocked problems receive a guided copy) for the identical 120 optimizer steps, so these comparisons are compute-matched by construction. Vanilla GRPO, BREAD*, and `OC-GRPO-Adaptive` train on the un-augmented 536-row set for the identical 64 steps (an epoch is 30 steps on the augmented set versus 16 on the un-augmented one). `OC-GRPO-Adaptive`'s online cascade additionally issues extra generation calls when it escalates guidance, so its budget is matched on data and parameter updates rather than wall-clock.

Overhead of the correction. Relative to guided baselines such as POPE* and PrefixRL*, `OC-GRPO` adds one extra log-probability computation per step: a single no-grad forward pass evaluating the unguided log-probabilities of the guided rollouts. Measured from the training logs, this costs 5.5–5.7% of wall-clock step time at 7B, 3.6% at 3B, and 3.1% at 1.5B, and nothing at inference. At 7B (4$\times$A100), the median step takes 183–188 s for vanilla GRPO and 192–199 s for `OC-GRPO-Fixed`.

Comparison to vanilla GRPO at matched budget. Because the augmented set is $1.79\times$ larger, `OC-GRPO-Fixed`'s full four epochs cost $1.88\times$ vanilla's rollout budget ($1.92\times$ its GPU-hours). The comparison at matched budget is therefore carried by the intermediate checkpoints (evaluated for seed 1; GPU-hours measured from the training logs):

Checkpoint	steps	rollout seqs.	GPU-h	$\times$ vanilla	Average P@1
Vanilla GRPO, 4 epochs (its full budget)	64	32,768	13.7	1.00$\times$	28.9
`OC-GRPO-Fixed`, epoch 1	30	15,360	7.0	0.51$\times$	30.0
`OC-GRPO-Fixed`, epoch 2	60	30,720	13.6	0.99$\times$	30.8
`OC-GRPO-Fixed`, epoch 4	120	61,440	26.3	1.92$\times$	31.7

At half of vanilla's budget, `OC-GRPO-Fixed` already exceeds vanilla's final Average Pass@1 (30.0 vs 28.9), and its epoch-2 checkpoint lands almost exactly on vanilla's full four-epoch budget (60 vs 64 optimizer steps, $0.94\times$ the generated rollouts, $0.99\times$ the wall-clock) while ahead by $+1.9$ points. The additional epochs purchase the capability the method targets: breaking the learning cliff on the hard training set. The hard-problem filter and guidance search that produce the augmented dataset are preprocessing costs of the learning-cliff setting itself, shared by every guided baseline, not costs of `OC-GRPO`.

H Results at Smaller Scales

We train Qwen2.5-3B-Instruct and Qwen2.5-1.5B-Instruct with the same hyperparameters as the 7B experiments and three seeds per method (the data order is held fixed across seeds). Table 5 reports, for each seed, the average Pass@1 over AIME, Gaokao2023, and OmniMath, together with the difference to vanilla GRPO and its 95% confidence interval. The interval is a paired bootstrap over problems: for each problem, we average the difference to vanilla GRPO over the three seeds, resample problems within each benchmark (10,000 replicates), and weight the three benchmarks equally. Pass@1 is estimated from the first sampled trajectory per problem. Tables 6 and 7 report per-benchmark results for the first seed.

Table 5: Average Pass@1 over AIME, Gaokao2023, and OmniMath for each seed at smaller scales, and the difference to vanilla GRPO with a 95% paired-bootstrap confidence interval. Intervals that exclude zero are bolded.

Method	**Seed 1**	**Seed 2**	**Seed 3**	**Mean±std**	**Δ vs. vanilla [95% CI]**
Qwen2.5-3B-Instruct					
Vanilla GRPO	23.44	23.87	24.41	23.91±0.48	ref.
PrefixRL*	23.39	24.35	25.44	24.39±1.03	+0.49 [−0.36, +1.38]
POPE*	24.02	24.00	25.36	24.46±0.78	+0.55 [−0.28, +1.39]
BREAD*	22.65	23.50	22.97	23.04±0.43	−0.87 [−1.84, +0.10]
OC-GRPO-Fixed	25.16	25.87	26.40	25.81±0.62	**+1.90 [+0.88, +2.93]**
OC-GRPO-Adaptive	24.42	24.75	25.28	24.81±0.43	+0.91 [−0.07, +1.88]
Qwen2.5-1.5B-Instruct					
Vanilla GRPO	17.61	17.62	17.90	17.71±0.16	ref.
PrefixRL*	16.37	17.61	16.83	16.93±0.63	−0.78 [−1.80, +0.22]
POPE*	17.90	19.04	18.29	18.41±0.58	+0.70 [−0.30, +1.70]
BREAD*	17.49	14.44	15.68	15.87±1.54	**−1.84 [−2.83, −0.87]**
OC-GRPO-Fixed	19.40	20.03	19.63	19.68±0.32	**+1.97 [+0.89, +3.07]**
OC-GRPO-Adaptive	18.93	19.67	20.26	19.62±0.67	**+1.91 [+0.87, +2.93]**

Method	**AIME (1983–2026)**		**Gaokao2023**		**OmniMath**		**Average**	
	P@1	**P@16**	**P@1**	**P@16**	**P@1**	**P@16**	**P@1**	**P@16**
Baselines								
Base	9.2	29.2	38.2	64.4	18.5	**44.2**	22.0	45.9
Vanilla GRPO (ref.)	10.2	30.8	41.6	63.6	18.6	43.5	23.5	46.0
Before-Training Guidance								
PrefixRL*	10.1 (-1.0%)	**32.1** (+4.2%)	41.3 (-0.6%)	64.4 (+1.2%)	18.8 (+1.1%)	41.7 (-4.1%)	23.4 (-0.3%)	**46.1** (+0.2%)
POPE*	9.8 (-4.0%)	30.2 (-2.0%)	43.4 (+4.4%)	62.3 (-2.0%)	18.9 (+1.6%)	41.3 (-5.1%)	24.0 (+2.4%)	44.6 (-3.0%)
OC-GRPO-Fixed	**11.0** (+7.9%)	30.6 (-0.7%)	**45.7** (+10.0%)	64.7 (+1.6%)	18.8 (+1.1%)	42.0 (-3.4%)	**25.2** (+7.2%)	45.8 (-0.4%)
During-Training Guidance								
BREAD*	9.1 (-10.9%)	31.3 (+1.6%)	40.0 (-3.8%)	62.1 (-2.4%)	18.9 (+1.6%)	42.6 (-2.1%)	22.7 (-3.4%)	45.3 (-1.4%)
OC-GRPO-Adaptive	10.7 (+5.0%)	29.3 (-4.9%)	43.4 (+4.4%)	**65.2** (+2.4%)	**19.2** (+3.2%)	42.3 (-2.8%)	24.4 (+4.1%)	45.6 (-0.8%)

Table 6: Results with Qwen2.5-3B-Instruct (single seed), reported on the three held-out benchmarks. The empirical setting is identical to Table 2. Per-benchmark results are for the first seed; three-seed averages are in Table 5. Pass@1 is estimated from the first sampled trajectory, and the Average column averages the rounded per-benchmark values.

Method	AIME (1983–2026)		Gaokao2023		OmniMath		Average	
	P@1	P@16	P@1	P@16	P@1	P@16	P@1	P@16
Baselines								
Base	3.8	18.7	34.8	56.9	13.0	37.6	17.2	37.7
Vanilla GRPO (ref.)	4.6	21.2	33.5	59.0	14.7	36.9	17.6	39.0
Before-Training Guidance								
PrefixRL*	3.9 (-15.2%)	20.7 (-2.4%)	31.2 (-7.0%)	57.7 (-2.2%)	14.0 (-4.8%)	37.7 (+2.2%)	16.4 (-7.2%)	38.7 (-0.9%)
POPE*	4.1 (-10.9%)	21.1 (-0.5%)	33.8 (+0.8%)	59.7 (+1.3%)	15.8 (+7.5%)	38.1 (+3.3%)	17.9 (+1.7%)	39.6 (+1.5%)
OC-GRPO-Fixed	4.2 (-8.7%)	19.5 (-8.1%)	**37.7** (+12.4%)	59.7 (+1.3%)	**16.3** (+10.9%)	36.4 (-1.4%)	**19.4** (+10.2%)	38.5 (-1.4%)
During-Training Guidance								
BREAD*	4.6 (+0.0%)	19.3 (-9.0%)	33.2 (-0.8%)	60.0 (+1.8%)	14.6 (-0.7%)	**38.2** (+3.5%)	17.5 (-0.7%)	39.2 (+0.4%)
OC-GRPO-Adaptive	**4.8** (+4.3%)	**21.3** (+0.5%)	35.8 (+7.0%)	**62.9** (+6.6%)	16.1 (+9.5%)	**38.2** (+3.5%)	18.9 (+7.2%)	**40.8** (+4.6%)

Table 7: Results with Qwen2.5-1.5B-Instruct (single seed), reported on the three held-out benchmarks. The empirical setting is identical to Table 2. Per-benchmark results are for the first seed; three-seed averages are in Table 5. Pass@1 is estimated from the first sampled trajectory, and the Average column averages the rounded per-benchmark values.

I CASCADING HINTS

In this section we consider 5 levels of hints, in contrast with solution prefixes. Algorithm 1 describes the implementation of cascading hints style guidance. Appendix J.2 contains the prompt templates for generating hints. Unlike solution prefixes, guidance with the hints is a two step process: (a) generation of the hint; (b) constructing a guided prompt using hints.

Guidance families. The adaptive hint variants all use the cascade of Algorithm 2 with the importance correction, and differ only in how the hints are produced. *Hints*: the policy itself generates the hint at each level L1–L5 from the templates in Appendix J.2; L1 uses only a failed attempt, while L2–L5 also use the reference solution. *Self-Correction*: only level L1, i.e., feedback on a failed attempt, so no reference solution is used. *Frontier Hints*: the same five levels, generated by an external frontier model (Claude Sonnet 4) instead of the policy. `OC-GRPO-Fixed` (Hints) instead trains on hint-augmented data constructed before training. All hint experiments use $N = 8$ rollouts per prompt.

Method	Training		AIME (1983–2026)		Gaokao2023		OlympiadBench		Minerva		OmniMath	
	P@1	P@16	P@1	P@16	P@1	P@16	P@1	P@16	P@1	P@16	P@1	P@16
Baselines												
Base	0.4	4.7	16.1	37.5	42.0	64.7	28.1	48.2	26.0	45.2	20.8	44.8
Vanilla GRPO	0.7±0.2 ref.	7.2±2.0 ref.	16.8±0.3 ref.	38.5±1.0 ref.	46.1±0.7 ref.	65.9±0.5 ref.	29.6±0.1 ref.	48.7±0.1 ref.	26.5±0.1 ref.	45.0±1.1 ref.	22.6±0.2 ref.	45.8±0.6 ref.
Fixed Hints												
`OC-GRPO-Fixed` (Hints)	**2.7±1.2** (+276.8%)	**11.7±2.6** (+62.0%)	16.6±2.6 (-1.3%)	**40.5±1.2** (+5.1%)	46.5±6.0 (+0.9%)	67.0±2.4 (+1.7%)	28.4±2.4 (-4.3%)	48.3±0.7 (-0.7%)	26.7±2.2 (+0.6%)	45.5±0.7 (+1.1%)	22.0±2.2 (-2.8%)	44.6±0.3 (-2.5%)
Adaptive Hints												
`OC-GRPO-Adaptive` (Hints)	*1.4±0.2* *(+91.3%)*	*7.8±0.5* *(+8.5%)*	***17.9±0.2*** *(+6.4%)*	39.8±0.2 (+3.4%)	***49.9±0.1*** *(+8.3%)*	***67.6±0.7*** *(+2.6%)*	***30.0±0.3*** *(+1.1%)*	48.7±0.3 (+0.1%)	***27.7±0.5*** *(+4.4%)*	***46.2±1.1*** *(+2.7%)*	***23.2±0.2*** *(+2.7%)*	***46.2±0.9*** *(+0.9%)*
`OC-GRPO` (Self-Correction)	0.7±0.1 (-2.4%)	6.9±0.4 (-4.7%)	17.3±0.2 (+3.1%)	*40.3±0.9* *(+4.5%)*	46.4±2.1 (+0.6%)	66.5±1.0 (+0.9%)	29.8±0.4 (+0.7%)	49.2±1.0 (+1.0%)	27.0±0.9 (+1.9%)	45.2±0.8 (+0.5%)	22.7±0.6 (+0.4%)	45.9±0.2 (+0.4%)
`OC-GRPO-Adaptive` (Frontier Hints)	0.8±0.3 (+6.8%)	6.2±1.6 (-14.7%)	17.4±0.5 (+3.4%)	40.1±1.5 (+4.0%)	48.5±1.4 (+5.2%)	67.2±0.2 (+2.0%)	29.9±0.4 (+0.8%)	***49.4±0.4*** *(+1.4%)*	27.5±0.5 (+3.6%)	45.5±0.6 (+1.1%)	23.0±0.6 (+1.6%)	45.7±0.4 (-0.1%)

Table 8: Hint ablation on Qwen2.5-7B-Instruct with 8 rollouts per prompt, using the best checkpoint per seed selected by validation Pass@1. Each non-base cell reports seed-averaged mean±std on the first line and relative improvement over Vanilla GRPO on the second line. Within the Adaptive Hints section, the best method per metric is shown in italics. Overall best run(s) for each metric are bolded.

Algorithm 1: Cascading self-guidance rollout construction (per training step)

Input: Batch $\mathcal{B} = \{x^{(d)}\}_{d=1}^{D} \subset \mathcal{D}_{\text{hard}}$; snapshot policy $\pi_{\theta_{\text{old}}}$; verifier v; group size N; max hint level L.
Output: Final rollout groups $\{G_{\text{final}}^{(d)}\}_{d=1}^{D}$ and behavior contexts $\{b^{(d)}\}_{d=1}^{D}$.

```
foreach x ∈ B do
    G ← ∅;
    for i ← 1 to N do
        sample y_i ~ π_{θ_old}(· | x);
        G ← G ∪ {y_i};
    compute rewards {r_i}_{i=1}^N ← v(x, G);
    if Σ_{i=1}^N r_i > 0 then                    ▷ At least one success under the original prompt
        G_final ← G; b ← x.
    else                                          ▷ Learning cliff: all N rollouts fail
        (G_final, b) ← RegenerateWithSelfGuidance(x, G, π_{θ_old}, v, N, L);
    store (G_final^(d), b^(d)) ← (G_final, b);
return {(G_final^(d), b^(d))}_{d=1}^D;
```

Algorithm 2: REGENERATEWITHSELFGUIDANCE: adjusting hint levels until success

Input: Problem x; failed unguided group $G = \{y_i\}_{i=1}^{N}$; snapshot policy $\pi_{\theta_{\text{old}}}$; verifier v; group size N; max hint level L.
Output: A final group G_{final} and its behavior context $b \in \{g_\ell(x)\}_{\ell=1}^{L} \cup \{x\}$.

```
for ℓ ← 1 to L do
    generate hint h_ℓ ← HINTGEN_ℓ(x, G) ▷ L1–L4 use failures; L2–L5 may use y*(x)
    form guided prompt g_ℓ(x) ← (x, h_ℓ);
    G' ← ∅;
    for i ← 1 to N do
        sample y'_i ~ π_θold(· | g_ℓ(x));
        G' ← G' ∪ {y'_i};
    compute rewards {r'_i}_{i=1}^N ← v(x, G');
    if Σ_{i=1}^N r'_i > 0 then                ▷ Stop as soon as any guided rollout succeeds
        return (G', g_ℓ(x));
return (G, x)                                 ▷ Fallback: no success at any hint level;
```

J ALL PROMPT TEMPLATES

J.1 SOLUTION PREFIX AS GUIDANCE

In place of LLM-generated hints, the prefix cascade exposes increasing portions of the reference solution itself as the privileged scaffold. Let $y^\star(x)$ denote the reference solution after stripping non-text artifacts (e.g., Asymptote diagram blocks). We define a prefix operator $\pi_\alpha(y^\star)$ that returns the first α fraction of $y^\star$ by character count, snapped to the nearest preceding word boundary, and we instantiate the cascade at fractions $\alpha \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$. At $\alpha < 1.0$ the prefix is a strict initial segment of the reference solution that the model is asked to *complete*; at $\alpha = 1.0$ the prefix is the full reference solution and the model is asked to *verify and reproduce* it. Unlike the hint cascade, no LLM call is required to produce a prefix: the operator π_α is a pure string operation on the ground-truth solution, so the cascade is deterministic and free.

The cascade is applied identically in the adaptive and fixed settings. In both cases, prefix-augmented prompts are used only to generate trajectories; the policy gradient is taken with respect to the original unprefixed prompt under an IS correction (Section 4).

Prefixes are then injected into a *solve prompt* for trajectory generation. The key design is that the model is always asked to produce a full solution and a final boxed answer, while the prefix length varies by level.

Prefix-conditioned solve prompt (used for guided rollout generation)

```
(partial prefix levels) Problem: {question}
Partial reference solution: {prefix}
Complete the rest of the solution step by step and output the final
    answer within \boxed{}. You can use the partial reference solution
    that follows the question to help you solve the problem.

system prompt for partial prefix levels: You are a math problem solver
    . When given a partial reference solution, use it to guide your
    reasoning, but write your solution independently. Do not repeat,
    copy, or paraphrase the reference text. Show your own step-by-step
    work and arrive at the answer through your own reasoning.

(full solution level) Problem: {question}
Reference solution: {solution}
Rephrase the solution above in your own words. Show your step-by-step
    work and output the final answer within \boxed{}.
```

The two framings are kept distinct on purpose. For partial prefixes ($\alpha < 1$) the anti-repetition system prompt discourages the model from merely echoing the given segment and pushes it to produce its own continuation, so that the trajectory contains genuine model-generated reasoning over which the policy gradient can be computed. For the full solution ($\alpha = 1$) we drop the anti-repetition prompt and ask explicitly for a paraphrase, the goal at this level is to provide a guaranteed-correct trajectory whose answer-extraction reward is 1, anchoring the IS-corrected gradient even on problems the base policy never solves on its own.

J.2 Cascading Hint Templates and Hinted Solve Prompts

We employ a *hierarchy* of hint generators with increasing privileged information. Let $y^\star(x)$ denote the reference solution and let y denote a failed model trajectory. We generate hints at levels: (i) feedback on the failed attempt, (ii) conceptual comparison to the reference, (iii) high-level strategy, (iv) detailed guidance, and (v) full-solution paraphrase.

Cascading hint generation templates (L1–L5)

```
L1 (blind; uses failure only):
You are an expert math tutor. A student attempted this problem but got
     it wrong.

Problem:
{problem}

Student's failed attempt:
{trajectory}

Analyze the student's work and think about the concepts in the problem
    . Provide a guidance or hint that redirects the student towards the
     correct approach.

Hint:


L2 (conceptual; uses reference + failure):
You are an expert math tutor. A student attempted this problem but got
     it wrong.

Problem:
{problem}

Reference solution:
{solution}

Student's failed attempt:
{trajectory}

Compare the student's approach with the correct solution and think
    about the key concepts that the student missed and provide guidance
     or hint that redirects the student to the correct solution. Do not
     reveal full solution.

Hint:


L3 (strategic; uses reference):
You are an expert math tutor. A student is struggling with this
    problem.

Problem:
{problem}
```

```
Reference solution:
{solution}

Provide a STEP-BY-STEP STRATEGY for solving this problem as a numbered
     list. Highlight the key approach needed. Do not reveal final
    answer.

Hint:


L4 (detailed; uses reference):
You are an expert math tutor. A student is struggling with this
    problem.

Problem:
{problem}

Reference solution:
{solution}

Provide DETAILED GUIDANCE to help the student solve this problem.
    Include:
- The correct approach with key intermediate calculations
- Specific values and expressions the student needs
- The critical steps that lead to the answer
You may reveal most of the solution path, but do NOT state the final
    boxed answer.

Hint:


L5 (paraphrase; uses reference):
Rephrase the reference solution in your own words. Show your step-by-
    step work and output the final answer within \boxed{}.
```

Hints are then injected into a *solve prompt* for a re-attempt. The key design is that the model is always asked to produce a full solution and a final boxed answer, while the hint varies by level.

Hint-conditioned solve prompt (used for re-attempt generation)

```
(L1--L4) Problem: {question}

Hint: {hint}

Let's think step by step and output the final answer within \boxed{}.
    You can use the hint that follows the question to help you solve
    the problem.

system prompt for L1--L4: You are a math problem solver. When given a
    hint, use it to guide your reasoning, but write your solution
    independently. Do not repeat, copy, or paraphrase the hint text.
    Show your own step-by-step work and arrive at the answer through
    your own reasoning.


(L5) Problem: {question}

Full solution: {solution}

Rephrase the solution above in your own words. Show your step-by-step
    work and output the final answer within \boxed{}.
```