Stepwise Guided Policy Optimization: Coloring Your Incorrect Reasoning in GRPO

Peter Chen *lc3826@columbia.edu*
Department of Mathematics
Columbia University

Xiaopeng Li *xiaopengli1@cuhk.edu.cn*
School of Artificial Intelligence
The Chinese University of Hong Kong, Shenzhen

Ziniu Li *liziniu1997@gmail.com*
School of Data Science
The Chinese University of Hong Kong, Shenzhen

Xi Chen *xc13@stern.nyu.edu*
Stern School of Business
New York University

Tianyi Lin *tl3335@columbia.edu*
Department of Industrial Engineering and Operation Research
Columbia University

Reviewed on OpenReview: *https://openreview.net/forum?id=ALnVAqtshR*

Abstract

Reinforcement learning (RL) has proven effective in strengthening the reasoning capabilities of large language models (LLMs). A widely adopted method, Group Relative Policy Optimization (GRPO) (Shao et al., 2024), has shown strong empirical results in training recent reasoning models (Guo et al., 2025a), but it fails to update the policy when all responses within a group are incorrect (i.e., all-negative-sample groups). This limitation highlights a gap between artificial and human intelligence: unlike humans, who can learn from mistakes, GRPO discards these failure signals. We introduce a simple framework to mitigate the all-negative-sample issue by incorporating response diversity within groups using a *step-wise* judge model, which can be trained directly or adapted from existing LLMs. In a simplified setting, we prove that this diversification accelerates GRPO's learning dynamics. We then empirically validate Stepwise Guided Policy Optimization (SGPO) across model sizes (7B, 14B, 32B) in both offline and online training on nine reasoning benchmarks (including base and distilled variants). Overall, SGPO improves average performance and is effective in early and mid-training when all-negative groups are prevalent, while improvements are not uniform across every benchmark and depend on the structure and informativeness of negative samples. Finally, SGPO does not require the judge model to generate correct solutions, distinguishing it from knowledge distillation methods.

1 Introduction

The rise of OpenAI-o1 (Jaech et al., 2024), DeepSeek-R1 (Guo et al., 2025a), and Kimi-1.5 (Team et al., 2025) has highlighted the emergence of *large AI reasoning models*. Unlike instruction-tuned models (Brown et al., 2020; Chowdhery et al., 2023; Touvron et al., 2023; Achiam et al., 2023), which produce quick responses by

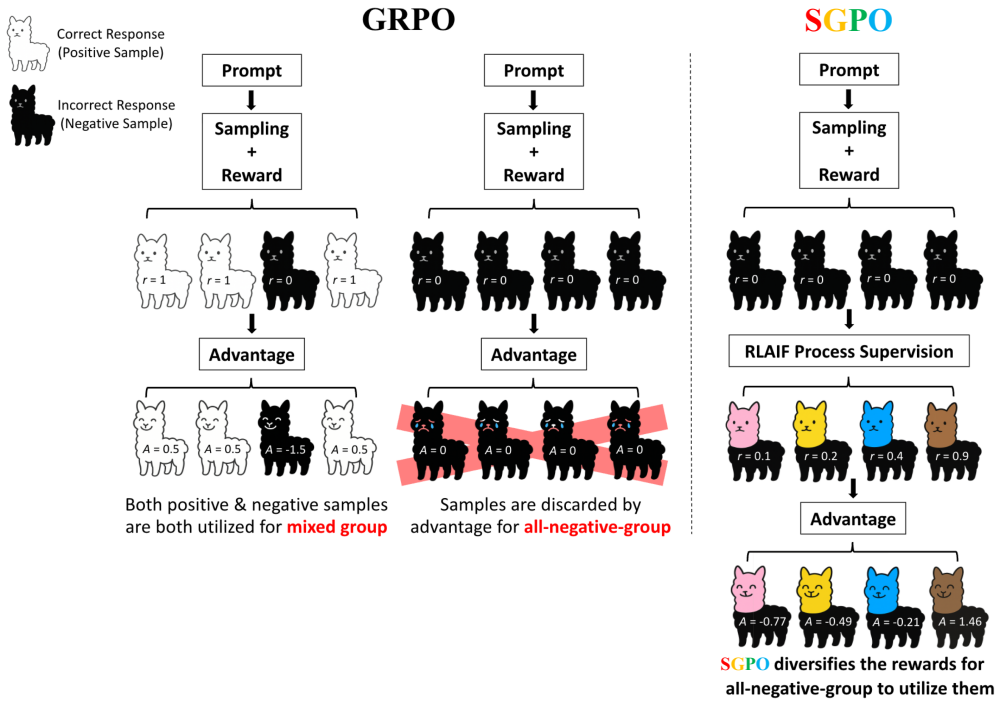

Figure 1: Main pipeline for Stepwise Guided Policy Optimization. On each llama, r indicates the reward of the sampled response and A indicates response's advantage through group relative computation.

statistically inferring the next token, these new reasoning models deliberately decompose complex prompts (e.g., mathematical problems) into intermediate steps and work through chain-of-thought reasoning (Wei et al., 2022; Yao et al., 2023; Besta et al., 2024; Xiang et al., 2025). This slower yet more rigorous process yields greater accuracy and makes them more human-like, enabling success on more complex and challenging tasks (Yang et al., 2018; Shi et al., 2024; Jain et al., 2025). As generative AI applications move beyond single-turn chat and question-answering, these reasoning models are poised to become more powerful and widely adopted, positioning them as a foundational component of modern AI systems.

At the heart of this revolution lies post-training with outcome-based and verifiable rewards (Cobbe et al., 2021; Uesato et al., 2022; Zelikman et al., 2022; Singh et al., 2023; Hosseini et al., 2024; Lightman et al., 2024; Wang et al., 2024; Setlur et al., 2025; Zhang et al., 2025b), together with reinforcement learning (RL) methods (Schulman et al., 2015; 2017; Li et al., 2024b; Ahmadian et al., 2024; Shao et al., 2024; Xiong et al., 2025a), appreciated for their simplicity, intuitiveness, and practicality. A leading approach is proximal policy optimization (PPO) (Schulman et al., 2017), which relies on a critic (or value) model to estimate advantages. While essential in general RL tasks, this critic is often unnecessary in large language models (LLMs) due to their deterministic transition dynamics (Li et al., 2024b). This observation has inspired alternatives such as group relative policy optimization (GRPO) (Shao et al., 2024) and its extensions (Yu et al., 2025b; Liu et al., 2025b; Chu et al., 2025; Zhang et al., 2025a), which estimate advantages directly in a group-relative fashion (normalizing rewards across multiple samples for the same prompt).

A major limitation of these methods arises when all sampled responses in a group are incorrect (i.e., *all-negative-sample* groups), which eliminates the learning signal and halts policy updates. In GRPO, given a prompt $\mathbf{x}$, responses $\{\mathbf{y}_i\}_{i=1}^G$ are drawn from the old policy π_{old} and assigned rewards $\{r_i\}_{i=1}^G$, where $r_i = 1$ if $\mathbf{y}_i$ is correct and 0 otherwise. Advantages are obtained by normalizing r_i within the group. If $r_i = 0$ for all i, the advantage vanishes, yielding no update. Such groups are frequent in early and mid-stages of training, when reasoning ability is weak[1]. This shortcoming highlights a gap between artificial and human intelligence: humans effectively learn from mistakes, which act as essential signals during cognitive development (Chialvo

[1]To reduce computational cost, training often uses small group sizes and short rollouts, further increasing the likelihood of all-negative-sample groups.

& Bak, 1999). In mathematical reasoning, all-negative-sample groups prompt a child to revise rules and strengthen reasoning ability.

Recent studies suggest that negative samples in RL-based large reasoning model training carry more nuanced value than previously assumed (Xiong et al., 2025a). Instead of treating negative samples uniformly, they advocate for principled mechanisms to distinguish negative samples. One prominent direction is process reward models (PRMs) (Lightman et al., 2024; Wang et al., 2024; Luo et al., 2024; Setlur et al., 2025; Zhang et al., 2025b), which estimate either the probability of final success or its change after each reasoning step. However, their reliance on speculative value functions makes them prone to reward hacking (Skalse et al., 2022).

A common observation is that many reasoning tasks possess a structure where step-level correctness can be explicitly defined. This motivates the use of a step-wise judge model that evaluates trajectories by labeling each step as correct (1) or incorrect (0). Such a model can be trained directly (Xiong et al., 2025b) or adapted from existing LLMs (Zha et al., 2025; He et al., 2025)[2]. By grounding rewards in step-level correctness rather than speculative value estimates, our method mitigates reward hacking and yields clearer signals. Intuitively, this allows negative samples to be differentiated through their trajectories: while early-stage reasoning trajectories are of low-quality, these remain informative – much like partial credit in education, where intermediate steps still guide learning.

Our approach enables a holistic evaluation of multi-step reasoning by transforming negative samples from binary outcome rewards into graded, step-level rewards. Consider a negative sample with five reasoning steps $(a_1, a_2, a_3, a_4, a_5)$. If the first error occurs at a_3, then a_1 and a_2 are correct, yielding a correctness proportion of $\frac{2}{5}$. To improve reliability, we adopt a Grok4-Heavy -inspired strategy where multiple independent judgments are obtained from the judge model, and the error position is determined by the majority vote. We further introduce two scaling parameters β and γ to downweight noisy or unreliable signals (see Eq. (2)). Unlike PRMs, our approach avoids memory overhead and does not require costly step-level human annotations, thereby accelerating training. In this work, we focus on outcome-based post-training with group-relative updates (GRPO-style) for structured reasoning tasks; extending our approach to arbitrary reward settings is beyond the scope of this paper and left to future work.

Contribution. We propose and analyze a simple and efficient framework that introduces response diversity within all-negative-sample groups. It is both theoretically grounded in the simplified setting and empirically effective on various models, distinguishing our approach from existing heuristics. Our contributions can be summarized as follows:

1. We propose a *Stepwise Guided Policy Optimization* (SGPO) framework that leverages a step-wise judge model that identifies the first incorrect step that causes the trajectory to deviate from correctness. This makes evaluation computationally tractable and reliable. *It is important to emphasize that our contribution lies not in designing effective judge models, but in introducing a framework that leverages step-wise judges to effectively distinguish negative samples.* We also prove that SGPO outperforms GRPO in a simplified setting.

2. We conduct experiments demonstrating the effectiveness of our approach in improving LLM reasoning. Evaluations are undertaken across various model sizes (7B, 14B, 32B) in both offline and online settings with nine benchmarks, including base and distilled variants, using GRPO as the primary baseline given our focus on outcome-based group-relative RLVR. Our results reveal two key benefits: (i) SGPO delivers improvements beyond the reach of GRPO, especially in the early and mid-stages of training where all-negative-sample groups are common; (ii) SGPO does not rely on more powerful judge models generating correct answers, allowing it to be distinguished from knowledge distillation.

The additional overhead from all-negative-sample groups remains modest, since the correctness can be efficiently verified against reference solutions, enabling rapid assessment of reasoning steps. As the computational and financial costs of closed-source judge models (`o4-mini`, `Claude3.7`) rise, SGPO accelerates learning dynamics, making the trade-off worthwhile. SGPO also outperforms GRPO with less powerful and

[2]We do not have access to their judge models as it's not publicly released, so we adapt our own from existing LLMs.

more affordable open-source judge models (`DeepSeek-V3-0324`, `Qwen3-235B-A22B`, `QwQ-32B`), confirming that SGPO remains effective even without cutting-edge LLMs and underscoring its practicality in lower-resource settings.

2 Preliminaries and Technical Background

LLM finetuning. LLM finetuning typically consists of *pre-training* and *post-training*. Pre-training equips the model with broad language understanding and generation capabilities, while post-training adapts the model to specific downstream objectives (e.g., improving mathematical problem solving via reasoning). In post-training, a model usually first undergoes *imitation learning* (e.g., supervised finetuning on human or expert trajectories, or direct distillation from stronger models), and then further improves by training on self-generated responses paired with feedback indicating whether the responses are good or bad. Two common feedback-driven paradigms are *reinforcement learning from human feedback* (RLHF), where a learned reward model (trained from human preferences) assigns rewards to model outputs, and *reinforcement learning from verifiable rewards* (RLVR), where rewards are computed by an automatic verifier (often used for math problems with checkable answers). We introduce GRPO, a representative RLVR method, below.

Policy optimization. Modern LLMs are built based on the Transformer architecture (Vaswani et al., 2017) and generate responses $\mathbf{y} = (a_1, \ldots, a_H)$ to user prompts $\mathbf{x}$, where each token $a_h \in \mathcal{V}^\star$, with $\mathcal{V}$ denoting the vocabulary and $\mathcal{V}^\star$ the set of all possible token sequences. We view the LLM as a policy $\pi_\theta(\mathbf{y}|\mathbf{x})$ parameterized by θ, assigning probabilities to responses $\mathbf{y}$ given $\mathbf{x}$. The policy operates in an auto-regressive way as follows (Agarwal et al., 2020; Mei et al., 2021; Li et al., 2024b):

$$\pi_\theta(\mathbf{y}|\mathbf{x}) = \prod_{h=1}^{H} \pi_\theta(a_h \mid \mathbf{x}, a_1, \ldots, a_{h-1}).$$

For a prompt $\mathbf{x}$ with ground-truth response $\mathbf{y}^\star_{\mathbf{x}}$, performance is evaluated using a regular-expression match on the final answer: $r(\mathbf{x}, \mathbf{y}) = 1$ if $\mathbf{y}$ matches $\mathbf{y}^\star_{\mathbf{x}}$ and $r(\mathbf{x}, \mathbf{y}) = 0$ otherwise (Hendrycks et al., 2021). We consider the reasoning tasks defined over a dataset $\mathcal{D} = (\mathbf{x}, \mathbf{y}^\star_{\mathbf{x}})$, where each $\mathbf{x}$ is a problem and $\mathbf{y}^\star_{\mathbf{x}}$ its ground-truth solution.

The policy gradient methods (Williams, 1992; Sutton & Barto, 1998) aim to maximize the objective $J(\theta) = \mathbb{E}_{\mathbf{x}\sim\rho, \mathbf{y}\sim\pi_\theta(\cdot|\mathbf{x})}[r(\mathbf{x}, \mathbf{y})]$ where ρ is the prompt distribution and π_θ is an LLM policy. Parameters are updated via $\theta \leftarrow \theta + \eta \nabla_\theta J(\theta)$. In practice, trajectories are sampled from an old policy $\pi_{\theta_{\text{old}}}$, which is different from π_θ, motivating the use of importance sampling as follows:

$$J(\theta) = \mathbb{E}_{\mathbf{x}\sim\rho, \mathbf{y}\sim\pi_{\theta_{\text{old}}}(\cdot|\mathbf{x})} \left[\frac{\pi_\theta(\mathbf{y}|\mathbf{x})}{\pi_{\theta_{\text{old}}}(\mathbf{y}|\mathbf{x})} r(\mathbf{x}, \mathbf{y}) \right].$$

However, this estimator suffers from high variance when π_θ deviates from $\pi_{\theta_{\text{old}}}$. To stabilize training, clipped surrogate objectives are used as follows:

$$J(\theta) = \mathbb{E}_{\mathbf{x}\sim\rho, \mathbf{y}\sim\pi_{\theta_{\text{old}}}(\cdot|\mathbf{x})} \left[\min \left\{ \frac{\pi_\theta(\mathbf{y}|\mathbf{x})}{\pi_{\theta_{\text{old}}}(\mathbf{y}|\mathbf{x})} r(\mathbf{x}, \mathbf{y}), \texttt{clip}\left(\frac{\pi_\theta(\mathbf{y}|\mathbf{x})}{\pi_{\theta_{\text{old}}}(\mathbf{y}|\mathbf{x})}, 1-\epsilon, 1+\epsilon \right) r(\mathbf{x}, \mathbf{y}) \right\} \right],$$

where $\texttt{clip}(x, 1-\varepsilon, 1+\varepsilon) := \max\{\min\{x, 1+\varepsilon\}, 1-\varepsilon\}$. The group relative policy optimization (GRPO) and its variants (Yu et al., 2025b; Liu et al., 2025b; Chu et al., 2025; Zhang et al., 2025a) adopt this framework but estimate gradients using groups of samples. For each prompt $\mathbf{x}$, GRPO samples responses $\mathbf{y}_1, \ldots, \mathbf{y}_G$ from $\pi_{\theta_{\text{old}}}$. We aim at maximizing the objective function in the form of

$$J(\theta) = \mathbb{E}_{\mathbf{x}\sim\rho, \{\mathbf{y}_i\}_{i=1}^G \sim\pi_{\theta_{\text{old}}}(\cdot|\mathbf{x})} \left[\frac{1}{G} \sum_{i=1}^{G} \min \left\{ \frac{\pi_\theta(\mathbf{y}_i|\mathbf{x})}{\pi_{\theta_{\text{old}}}(\mathbf{y}_i|\mathbf{x})} A_i, \texttt{clip}\left(\frac{\pi_\theta(\mathbf{y}_i|\mathbf{x})}{\pi_{\theta_{\text{old}}}(\mathbf{y}_i|\mathbf{x})}, 1-\epsilon, 1+\epsilon \right) A_i \right\} \right],$$

where $\epsilon \in (0, 1)$ and the advantage A_i is computed as

$$A_i = \frac{r(\mathbf{x}, \mathbf{y}_i) - \texttt{mean}(\{r(\mathbf{x}, \mathbf{y}_1), \ldots, r(\mathbf{x}, \mathbf{y}_G)\})}{\texttt{std}(\{r(\mathbf{x}, \mathbf{y}_1), \ldots, r(\mathbf{x}, \mathbf{y}_G)\})}, \tag{1}$$

where $r(\mathbf{x}, \mathbf{y}_i) = 1$ if $\mathbf{y}_i$ matches the ground-truth answer and 0 otherwise.

Remark 2.1. *When rewards are identical across all samples within a group, $A_i = 0$ and no update occurs. This is appropriate for all-positive groups but constitutes a critical limitation for all-negative groups, where GRPO fails to exploit mistakes as learning signals.*

3 Main Results

We propose the Stepwise Guided Policy Optimization (SGPO) framework, which employs the step-wise judge model to detect the first incorrect step that leads a trajectory away from correctness. In a simplified setting, we prove that SGPO consistently accelerates GRPO's learning dynamics.

3.1 A Step-wise Judge Model

We propose a principled reward mechanism for negative samples, wherein the step-wise judge model differentiates between structurally sound but partially incorrect reasoning and entirely erroneous responses. This design is motivated by the intuition that an incorrect final answer does not invalidate the entire reasoning process. For instance, a model may follow a logically coherent sequence of steps yet make a minor error – such as an arithmetic slip – that leads to an incorrect conclusion. Treating such cases the same as fundamentally flawed or incoherent reasoning does not make sense. This refinement remains effective under constraints such as reduced output length, where a model may be unable to complete the full solution but still demonstrates a valid reasoning trajectory.

Our step-wise judge model can be either trained directly or adapted from existing LLMs. It evaluates responses sequentially, identifying the first substantive error – such as a computational slip or a logical fallacy – that causes the trajectory to deviate from correctness. To formalize this, we define the *Reasoning Trajectory Score* (RTS) for an incorrect response $\mathbf{y}$, denoted as $\texttt{RTS}(\mathbf{y}) \in [0, 1]$. The judge model checks each step in order, pinpoints the first error, and treats all preceding steps as the valid reasoning segment. $\texttt{RTS}(\mathbf{y})$ is then computed as the ratio of the valid segment length to the total trajectory length. For example, if $\mathbf{y}$ consists of five steps $(a_1, a_2, a_3, a_4, a_5)$ and the first error occurs at a_4, then $\texttt{RTS}(\mathbf{y}) = \frac{3}{5}$, indicating that three steps of reasoning are correct before erroneous.

In our experiment, we adapt the judge model from existing LLMs, either closed-source (`o4-mini`, `Claude3.7`) or open-source (`DeepSeek-V3-0324`, `Qwen3-235B-A22B`, `QwQ-32B`). To enhance reliability and further reduce variance in the reward signal, we employ the following protocol: (i) alongside the candidate response, we provide the judge with a reference solution (e.g., a gold final answer, a brief solution outline, or a full reasoning trace when available); in our experiments, we draw this solution from a supervised fine-tuning dataset with correct answers and reasoning trajectories, which anchors the intended solution path and enables error localization; and (ii) we elicit step-wise evaluation rather than holistic evaluation. The judge model justifies correctness or flags an error sentence by sentence, identifies the first clear mistake, and then traces how this error propagates to the final incorrect conclusion.

Based on the reasoning trajectory score, we introduce a new outcome reward function:

$$r_{\texttt{SGPO}}(\mathbf{y}) = \begin{cases} 1, & \text{if the final answer of } \mathbf{y} \text{ is correct,} \\ \frac{1}{1+\exp(-\beta(\texttt{RTS}(\mathbf{y})-\gamma))}, & \text{otherwise.} \end{cases} \tag{2}$$

where $\gamma > 0$ and $\beta > 0$ (taken as 10 and 0.5 in the actual implementation) are two parameters to decide scale threshold and scale intensity, respectively. This design ensures that the model receives a more informative gradient signal during training, thereby encouraging refinement of partially correct reasoning rather than indiscriminate penalization of all incorrect outputs. This specification of $r_{\texttt{SGPO}}$ can be directly incorporated into the advantage calculation in Eq. (1). As a consequence, SGPO keeps the same rollout pipeline as GRPO and the same outcome-based supervision, and only replaces the reward used in within-group advantage computation from $r(x, y)$ to $r_{\texttt{SGPO}}(y)$ in Eq. (2).

Remark 3.1. *Our approach differs from process reward models (PRMs) (e.g. Lightman et al., 2024). For a prompt $\mathbf{x}$ and a prefix of reasoning steps $(a_1, \ldots, a_t)$, a PRM typically predicts either (i) a prefix-level value $V(\mathbf{x}, a_{1:t}) = \mathbb{P}(\text{final answer correct} \mid \mathbf{x}, a_{1:t})$, or (ii) a step-level progress signal such as $\Delta_t = V(\mathbf{x}, a_{1:t}) -$*

$V(\mathbf{x}, a_{1:t-1})$. *In practice, PRMs are trained by supervised ranking of intermediate steps and are used to re-rank trajectories or shape training at the* prefix *level, acting as approximate value (or Q-value) functions over prefixes. In contrast, SGPO introduces a different way of producing and using feedback signals: (i) Policy-guided rollouts without search. All trajectories are sampled from the current policy, without PRM-guided exploration or trajectory alteration; (ii) Post-hoc first-error identification. A step-wise judge inspects the entire trajectory, pinpoints the earliest error relative to a reference solution, and converts this into a calibrated scalar reward $r_{\texttt{SGPO}}(\mathbf{y})$ via the reasoning trajectory score; (iii) Stable credit assignment in all-negative-sample groups. By locating the first definitive mistake only after observing the full trace, SGPO eliminates the look-ahead ambiguity and feedback loops inherent to PRM-guided search (Zhang et al., 2024a), while avoiding the need for the judge to solve the problem or approximate a value function. We note that SGPO alleviates but does not fully eliminate degeneracy: if all trajectories in a group receive identical $r_{\texttt{SGPO}}$ (e.g., they fail at the same first-error position), the group-normalized advantages can still vanish, though this phenomenon is rarely observed in our experiments.*

Remark 3.2. *Our approach differs from knowledge distillation (e.g. Kang et al., 2023; Gu et al., 2024). The student model trained via knowledge distillation inherits the judge the model's failure, since it only imitates the judge model's outputs. In contrast, SGPO leverages the judge model to identify mistakes in the student's reasoning, providing learning signals that go beyond imitation and enabling improvements unattainable by knowledge distillation.*

3.2 Accelerating Learning Dynamics

We present a theoretical analysis to explain why SGPO outperforms GRPO. We study an idealized setting in which the step-wise judge provides accurate first-error localization. Even in this regime, establishing a general separation is technically subtle, so we focus on a stylized example. To this end, we consider the case when $H = 2$, where each step admits two possible actions $a_h \in 1, 2$ for $h = 1, 2$. Extending the analysis to general horizons or action spaces and imperfect (noisy or biased) judges is left to future work. This configuration follows prior works (Dayan, 1991; Li et al., 2024b), in which analogous examples were employed to validate theoretical insights. Without loss of generality, we assume a unique ground-truth response $\mathbf{y}^\star_{\mathbf{x}} = (2, 2)$ for the prompt $\mathbf{x}$. For clarity, we restrict the sample space to $(1, 1), (2, 1), (2, 2)$, excluding $(1, 2)$ since a correct reasoning step is unlikely to, and should not, follow an incorrect precursor.

To illustrate the effect of SGPO, we analyze the learning dynamics of SGPO and GRPO in this simplified setting. Under GRPO, the rewards are assigned as $r((2, 2)) = 1$ and $r((2, 1)) = r((1, 1)) = 0$, meaning that only selecting the "good" action 2 at both steps yields a positive reward. In contrast, SGPO assigns $r_{\texttt{SGPO}}((2, 2)) = 1$, $r_{\texttt{SGPO}}((2, 1)) = \frac{1}{2}$ and $r_{\texttt{SGPO}}((1, 1)) = 0$. The difference is that partial progress – choosing the "good" action 2 in the first step but failing at the second – receives no credit in GRPO yet proportional credit in SGPO. Here, $\frac{1}{2}$ is chosen for illustrative purposes to convey the qualitative behavior of the reward mechanism, while the exact values used in experiments are determined by Eq. (2).

The algorithm iteratively updates the policy parameter θ using samples drawn from the current policy π_θ. We rewrite the generic GRPO update with a step size $\eta > 0$ as follows,

$$\theta^{(k+1)} = \theta^{(k)} + \eta \cdot g(\theta), \quad \text{where } g(\theta) = \frac{1}{NGH} \left(\sum_{i=1}^{N} \sum_{k=1}^{G} \sum_{h=1}^{H} s_\theta(\mathbf{x}^i, a^{i,k}_{1:h-1}) A_{i,k} \right),$$

where N is the number of prompts, G is the number of groups, H is the number of reasoning steps in each response, $s_\theta(\mathbf{x}^i, a^{i,k}_{1:h-1}) := \nabla\theta \log \pi_\theta(a_t|\mathbf{x}, a_{1:h-1})$ is the score function, and the advantage $A_{i,k}$ is defined by Eq. (1) for each question $\mathbf{x}^i$. To distinguish, we denote $g_{\texttt{GRPO}}(\cdot)$ as the gradient estimator using classical outcome reward model r, and $g_{\texttt{SGPO}}(\cdot)$ as the gradient estimator using the reward $r_{\texttt{SGPO}}$ as proposed in Section 3.1.

In our analysis, we examine the population-level learning dynamics with $G = 2$, omitting clipping and importance sampling. In practice, importance sampling and clipping are important for training stability; we omit them here to simplify the analysis and leave their theoretical treatment to future work. For simplicity, we perform our analysis in the likelihood space rather than in the parameter space directly. Indeed, we

define the key quantities as follows

$$p \doteq \pi_{\theta_1}(a_1 = 2, |, \mathbf{x}) = \frac{e^{\theta_1^{\mathbf{x},2}}}{e^{\theta_1^{\mathbf{x},1}} + e^{\theta_1^{\mathbf{x},2}}}, \qquad q \doteq \pi_{\theta_2}(a_2 = 2, |, \mathbf{x}, a_1 = 2) = \frac{e^{\theta_2^{\mathbf{x},2,2}}}{e^{\theta_2^{\mathbf{x},2,1}} + e^{\theta_2^{\mathbf{x},2,2}}}.$$

Note that the original 4-dimensional parameter space defined by $\theta_1^{\mathbf{x},1}$, $\theta_1^{\mathbf{x},2}$, $\theta_2^{\mathbf{x},2,1}$ and $\theta_2^{\mathbf{x},2,2}$ in $\mathbb{R}$ is reduced to a 2-dimensional likelihood space defined by $p, q \in [0, 1]$.

For our simple stylized model, we compute the score functions in terms of likelihood parameters p, q as follows,

$$s(a_1 = 1 \,|\, \mathbf{x}) = \begin{bmatrix} p \\ -p \\ 0 \\ 0 \end{bmatrix}, \quad s(a_1 = 2 \,|\, \mathbf{x}) = \begin{bmatrix} p-1 \\ 1-p \\ 0 \\ 0 \end{bmatrix},$$

and

$$s(a_2 = 1 \,|\, \mathbf{x}, a_1 = 2) = \begin{bmatrix} 0 \\ 0 \\ q \\ -q \end{bmatrix}, \quad s(a_2 = 2 \,|\, \mathbf{x}, a_1 = 2) = \begin{bmatrix} 0 \\ 0 \\ q-1 \\ 1-q \end{bmatrix}.$$

The responses can be drawn i.i.d. from the distribution as follows,

$$(a_1, a_2) = \begin{cases} (1,1), & \text{w.p. } 1-p, \\ (2,1), & \text{w.p. } p(1-q), \\ (2,2), & \text{w.p. } pq. \end{cases}.$$

The SGPO and GRPO training dynamics with population-level policy gradient can be computed exactly for the stylized model as follows,

$$\bar{g}_{\texttt{SGPO}}(\theta) = \mathbb{E}[g_{\texttt{SGPO}}(\theta)] = \tfrac{1}{2} \begin{bmatrix} p(p-1) \\ p(1-p) \\ p^2 q(q-1) \\ p^2 q(1-q) \end{bmatrix}, \quad \bar{g}_{\texttt{GRPO}}(\theta) = \mathbb{E}[g_{\texttt{GRPO}}(\theta)] = \tfrac{1}{2} \begin{bmatrix} p(p-1)q \\ p(1-p)q \\ pq(q-1) \\ pq(1-q) \end{bmatrix}.$$

Since $g_{\texttt{GRPO}}(\theta)$ and $g_{\texttt{SGPO}}(\theta)$ concentrate around $\bar{g}_{\texttt{GRPO}}(\theta)$ and $\bar{g}_{\texttt{SGPO}}(\theta)$ when the number of samples in each group is sufficiently large, it is reasonable to analyze the population-level dynamics for illustration. Note that the high-probability guarantees for the sample-level dynamics can be derived using concentration inequalities under certain conditions.

Denote $p_{\texttt{GRPO}}^{(k)}$ and $q_{\texttt{GRPO}}^{(k)}$ as the value of quantity p and q at iteration k under GRPO. Analogously, $p_{\texttt{SGPO}}^{(k)}$ and $q_{\texttt{SGPO}}^{(k)}$ are the corresponding probabilities p and q at iteration k under SGPO. We can explicitly write down the SGPO and GRPO update rules with $\eta = 1$ as follows,

$$\begin{cases} p_{\texttt{SGPO}}^{(k+1)} = \exp(f_{11}(p_{\texttt{SGPO}}^{(k)})), \\ q_{\texttt{SGPO}}^{(k+1)} = \exp(f_{12}(p_{\texttt{SGPO}}^{(k)}, q_{\texttt{SGPO}}^{(k)})), \end{cases} \quad \text{and} \quad \begin{cases} p_{\texttt{GRPO}}^{(k+1)} = \exp(f_{21}(p_{\texttt{GRPO}}^{(k)}, q_{\texttt{GRPO}}^{(k)})), \\ q_{\texttt{GRPO}}^{(k+1)} = \exp(f_{22}(p_{\texttt{GRPO}}^{(k)}, q_{\texttt{GRPO}}^{(k)})), \end{cases}, \tag{3}$$

where the functions f_{ij} are defined by

$$\begin{aligned} f_{11}(p) &= \log(p) + p(1-p) - \log(1 - p + pe^{p(1-p)}), \\ f_{21}(p,q) &= \log(p) + p(1-p)q - \log(1 - p + pe^{p(1-p)q}), \\ f_{12}(p,q) &= \log(q) + p^2 q(1-q) - \log(1 - q + qe^{p^2 q(1-q)}), \\ f_{22}(p,q) &= \log(q) + pq(1-q) - \log(1 - q + qe^{pq(1-q)}). \end{aligned} \tag{4}$$

Our theoretical findings are summarized in the following theorem.

Theorem 3.3. *Suppose that $p^{(0)}_{GRPO} = q^{(0)}_{GRPO} = p^{(0)}_{SGPO} = q^{(0)}_{SGPO} = \frac{1}{2}$ and $\eta = 1$[3] for GRPO and SGPO. Then, we have that (i) GRPO and SGPO achieve successful learning: $p^{(k)}_{GRPO}, q^{(k)}_{GRPO}, p^{(k)}_{SGPO}, q^{(k)}_{SGPO} \to 1$ as $k \to +\infty$; (ii) SGPO outperforms GRPO in learning the "good" action in the first step: $p^{(k)}_{SGPO} > p^{(k)}_{GRPO}$ for all $k \geq 1$; (iii) SGPO outperforms GRPO in learning the optimal policy: $p^{(k)}_{SGPO} q^{(k)}_{SGPO} > p^{(k)}_{GRPO} q^{(k)}_{GRPO}$ for all $k \geq 1$.*

Proof. To show (i), recall that the sequence $(p^{(k)}_{\texttt{SGPO}})_{k\in\mathbb{N}}$ is strictly increasing and bounded in $(0,1)$ from Lemmas B.4(i) and B.4(ii), so it converges to some value $c \in (0,1]$. Taking limit as $k \to \infty$:

$$1 = \lim_{k\to\infty} \frac{p^{(k+1)}_{\texttt{SGPO}}}{p^{(k)}_{\texttt{SGPO}}} = \lim_{k\to\infty} \frac{1}{(1-p^{(k)}_{\texttt{SGPO}})e^{-p^{(k)}_{\texttt{SGPO}}(1-p^{(k)}_{\texttt{SGPO}})}+p^{(k)}_{\texttt{SGPO}}} = \frac{1}{(1-c)e^{-c(1-c)}+c}.$$

Using the simple Taylor lower bound $e^{-x} \geq 1 - x$, we have

$$1 = \frac{1}{(1-c)e^{-c(1-c)}+c} \geq \frac{1}{(1-c)(1-c(1-c))+c} \implies (c-1)^2 \leq 0 \implies c = 1.$$

This shows $p^{(k)}_{\texttt{SGPO}} \to 1$ as $k \to \infty$. Similarly, we can show $q^{(k)}_{\texttt{GRPO}}, p^{(k)}_{\texttt{SGPO}}, q^{(k)}_{\texttt{SGPO}} \to 1$ as $k \to \infty$.

To show (ii), consider the base case:

$$\begin{aligned}
p^{(1)}_{\texttt{SGPO}} &= \exp(f_{11}(p^{(0)}_{\texttt{SGPO}})) = \exp(\log p^{(0)}_{\texttt{SGPO}} + h_{p^{(0)}_{\texttt{SGPO}}}(p^{(0)}_{\texttt{SGPO}}(1-p^{(0)}_{\texttt{SGPO}}))) \\
&= \exp(\log p^{(0)}_{\texttt{GRPO}} + h_{p^{(0)}_{\texttt{GRPO}}}(p^{(0)}_{\texttt{GRPO}}(1-p^{(0)}_{\texttt{GRPO}}))) \\
&> \exp(\log p^{(0)}_{\texttt{GRPO}} + h_{p^{(0)}_{\texttt{GRPO}}}(p^{(0)}_{\texttt{GRPO}}(1-p^{(0)}_{\texttt{GRPO}})q^{(0)}_{\texttt{GRPO}})) = \exp(f_{21}(p^{(0)}_{\texttt{GRPO}}, q^{(0)}_{\texttt{GRPO}})) = p^{(1)}_{\texttt{GRPO}},
\end{aligned}$$

where the inequality follows from Lemma B.1(ii). Thus, we use induction and assume $p^{(k)}_{\texttt{SGPO}} > p^{(k)}_{\texttt{GRPO}}$ for some $k \geq 1$. Then, we have

$$\begin{aligned}
p^{(k+1)}_{\texttt{SGPO}} &= \exp(f_{11}(p^{(k)}_{\texttt{SGPO}})) > \exp(f_{11}(p^{(k)}_{\texttt{GRPO}})) = \exp(\log p^{(k)}_{\texttt{GRPO}} + h_{p^{(k)}_{\texttt{GRPO}}}(p^{(k)}_{\texttt{GRPO}}(1-p^{(k)}_{\texttt{GRPO}}))) \\
&> \exp(\log p^{(k)}_{\texttt{GRPO}} + h_{p^{(k)}_{\texttt{GRPO}}}(p^{(k)}_{\texttt{GRPO}}(1-p^{(k)}_{\texttt{GRPO}})q^{(k)}_{\texttt{GRPO}})) = \exp(f_{21}(p^{(k)}_{\texttt{GRPO}}, q^{(k)}_{\texttt{GRPO}})) = p^{(k+1)}_{\texttt{GRPO}},
\end{aligned}$$

where the first inequality uses Lemma B.1(i) and the second one uses Lemma B.1(ii). Thus, $p^{(k+1)}_{\texttt{SGPO}} > p^{(k+1)}_{\texttt{GRPO}}$ and the induction is done. We have proved that $p^{(k)}_{\texttt{SGPO}} > p^{(k)}_{\texttt{GRPO}}$ for all $k \geq 1$.

To show (iii), first notice that we can show $p^{(k)}_{\texttt{GRPO}} = q^{(k)}_{\texttt{GRPO}}$ for all $k \geq 0$ by induction. The base case is trivial by initialization. Suppose $p^{(k)}_{\texttt{GRPO}} = q^{(k)}_{\texttt{GRPO}}$ for some $k \geq 0$, then by noticing that $f_{21}(p,p) = f_{22}(p,p)$, we have

$$\begin{aligned}
p^{(k+1)}_{\texttt{GRPO}} &= \exp(f_{21}(p^{(k)}_{\texttt{GRPO}}, q^{(k)}_{\texttt{GRPO}})) = \exp(f_{21}(p^{(k)}_{\texttt{GRPO}}, p^{(k)}_{\texttt{GRPO}})) \\
&= \exp(f_{22}(p^{(k)}_{\texttt{GRPO}}, p^{(k)}_{\texttt{GRPO}})) = \exp(f_{22}(p^{(k)}_{\texttt{GRPO}}, q^{(k)}_{\texttt{GRPO}})) = q^{(k+1)}_{\texttt{GRPO}}.
\end{aligned}$$

Thus, by induction, $p^{(k)}_{\texttt{GRPO}} = q^{(k)}_{\texttt{GRPO}}$ for all $k \geq 0$. Now, we can reduce the update rule of $p^{(k)}_{\texttt{GRPO}}$ as

$$p^{(k+1)}_{\texttt{GRPO}} = \frac{1}{(1/p^{(k)}_{\texttt{GRPO}}-1)\exp(-(p^{(k)}_{\texttt{GRPO}})^2(1-p^{(k)}_{\texttt{GRPO}}))+1}.$$

In addition, we recall the update rule of $p^{(k)}_{\texttt{SGPO}}$ and $q^{(k)}_{\texttt{SGPO}}$ as

$$\begin{aligned}
p^{(k+1)}_{\texttt{SGPO}} &= \frac{1}{(1/p^{(k)}_{\texttt{SGPO}}-1)\exp(-p^{(k)}_{\texttt{SGPO}}(1-p^{(k)}_{\texttt{SGPO}}))+1} \\
q^{(k+1)}_{\texttt{SGPO}} &= \frac{1}{(1/q^{(k)}_{\texttt{SGPO}}-1)\exp(-(p^{(k)}_{\texttt{SGPO}})^2 q^{(k)}_{\texttt{SGPO}}(1-q^{(k)}_{\texttt{SGPO}}))+1}
\end{aligned}.$$

It suffices to show $p^{(k)}_{\texttt{SGPO}} q^{(k)}_{\texttt{SGPO}} > (p^{(k)}_{\texttt{GRPO}})^2$ for all $k \geq 1$. We prove by induction. For the base case,

$$\sqrt{p^{(1)}_{\texttt{SGPO}} q^{(1)}_{\texttt{SGPO}}} = \sqrt{\frac{1}{1+e^{-1/4}} \cdot \frac{1}{1+e^{-1/16}}} > \frac{1}{1+e^{-1/8}} = p^{(1)}_{\texttt{GRPO}}.$$

[3]We use the unit stepsize for simplicity. Our results are valid for any sufficiently small step size.

Table 1: Evaluation results on offline RL training. For each model, we report the baseline performance before RL training. We then report RL training results that uses only negative samples and positive samples, respectively. Performance across validation and training dataset (`LIMO`) is shown.

	`AMC23` avg@16	`AIME24` avg@16	`MATH500` pass@1	`Olympiads` pass@1	`LIMO` pass@1
Qwen2.5-14B-Instruct					
Baseline	58.59	14.58	**80.40**	41.78	31.70
Negative Samples only	**61.88**	**15.21**	**80.40**	**42.37**	30.11
Positive Samples only	61.72	14.58	79.80	42.07	**38.68**
Qwen2.5-32B-Instruct					
Baseline	64.22	17.08	**83.60**	45.93	34.64
Negative Samples only	**69.53**	**20.42**	83.00	46.37	36.47
Positive Samples only	66.87	18.75	**83.60**	**47.41**	**41.86**

The above inequality holds true since Lemma B.1(iv) implies

$$2\log(1+e^{-1/8}) > \log(1+e^{-1/4}) + \log(1+e^{-1/16}),$$

It remains to show that $p^{(k)}_{\texttt{SGPO}} q^{(k)}_{\texttt{SGPO}} > (p^{(k)}_{\texttt{GRPO}})^2$ implies $p^{(k+1)}_{\texttt{SGPO}} q^{(k+1)}_{\texttt{SGPO}} > (p^{(k+1)}_{\texttt{GRPO}})^2$ for $k \geq 1$. By Lemma B.4(iii), we know $p^{(k)}_{\texttt{SGPO}} > q^{(k)}_{\texttt{SGPO}}$ for all $k \geq 1$. Thus, Lemma B.2 implies that

$$p^{(k+1)}_{\texttt{SGPO}} q^{(k+1)}_{\texttt{SGPO}} = \frac{1}{A(p^{(k)}_{\texttt{SGPO}}) B(p^{(k)}_{\texttt{SGPO}}, q^{(k)}_{\texttt{SGPO}})} > \frac{1}{C\left(\sqrt{p^{(k)}_{\texttt{SGPO}} q^{(k)}_{\texttt{SGPO}}}\right)^2}.$$

Using Lemma B.1(iii), we complete the induction by applying our induction hypothesis:

$$\frac{1}{C\left(\sqrt{p^{(k)}_{\texttt{SGPO}} q^{(k)}_{\texttt{SGPO}}}\right)^2} = \left(\exp\left(f_{21}\left(\sqrt{p^{(k)}_{\texttt{SGPO}} q^{(k)}_{\texttt{SGPO}}}, \sqrt{p^{(k)}_{\texttt{SGPO}} q^{(k)}_{\texttt{SGPO}}}\right)\right)\right)^2 > \left(\exp(f_{21}(p^{(k)}_{\texttt{GRPO}}, p^{(k)}_{\texttt{GRPO}}))\right)^2 = (p^{(k+1)}_{\texttt{GRPO}})^2.$$

This completes the proof. □

Remark 3.4. *Theorem 3.3 presents one of the first theoretical analyses of GRPO with multiple samples and multi-step reasoning in the context of LLM reasoning. The first part establishes that SGPO converges to the optimal policy. The second and third parts demonstrate that SGPO both accelerates the acquisition of partially correct reasoning steps and preserves partial reasoning ability even when the final answer is incorrect. Importantly, the theorem provides a* ***per-iteration*** *comparison of learning under different reward mechanisms – an aspect rarely examined in previous works. The provable improvement in learning the optimal policy is also consistent with our numerical findings. We plot the resulting learning curves of our numerical simulation in Figure 2. The left panel shows the probability of selecting the "good" action in the first step at iteration k (i.e., $p^{(k)}_{SGPO}$ vs. $p^{(k)}_{GRPO}$), while the right panel shows the probability of learning the optimal policy (i.e., $p^{(k)}_{SGPO} q^{(k)}_{SGPO}$ vs. $p^{(k)}_{GRPO} q^{(k)}_{GRPO}$). The results align with the predictions of Theorem 3.3, demonstrating that the likelihood of learning the optimal policy under SGPO consistently exceeds that of GRPO across training.*

4 Experiments

We present the benefits of differentiating negative samples through experiments in both offline and online settings. Offline RL is more computationally efficient, offering faster training, reduced memory consumption, and improved stability. In contrast, online RL provides greater flexibility and learning capacity, and has become the standard approach in large-scale reasoning models such as DeepSeek-R1 (Guo et al., 2025a).

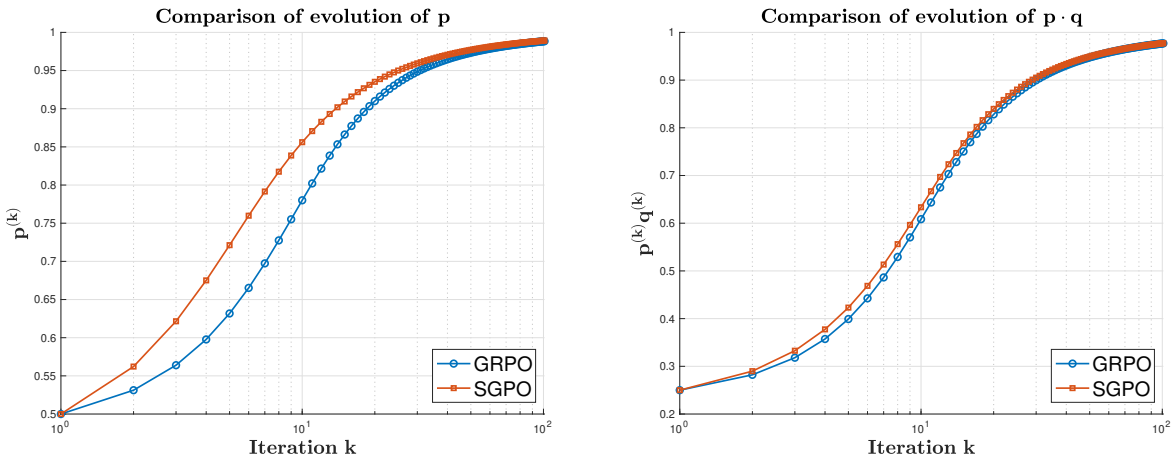

Figure 2: Learning dynamics of GRPO and SGPO in the simplified setting.

4.1 Offline Training

For baselines, we consider strong models without further fine-tuned on math-specific SFT datasets, namely `Qwen2.5-14B-Instruct` and `Qwen2.5-32B-Instruct`. Prior work showed that a small set of carefully curated prompts significantly enhance the reasoning capability. Accordingly, we adopt the `GAIR/LIMO` dataset (Ye et al., 2025) as the training set, which has demonstrated strong potential for improving the reasoning performance of large-scale (32B) models in offline SFT. Evaluation is conducted on four standard math reasoning benchmarks: `AIME24`, `AMC23`, `MATH500` (Hendrycks et al., 2021), and `OlympiadBench` (He et al., 2024). Our aim is to highlight the rich learning signal contained in all-negative-sample groups, showing that training exclusively on them can still yield performance gains. For benchmarks with fewer than 100 questions (`AMC23`, `AIME24`), we report `avg@16` results with a decoding temperature of 0.6 and `Top_P` $= 0.95$. For benchmarks with more than 100 questions, we report `pass@1` results using greedy decoding. Here, pass@k is the percentage of prompts solved by at least one sample, while avg@k is the average percentage of corrected samples among the k samples. The maximum decoding length is set to 32768 tokens.

We conduct all response generation and model updates using offline RL (Peters & Schaal, 2007) with the standard GRPO mechanism. Specifically, the model is updated with advantages estimated from the offline dataset (see, e.g., Peng et al., 2019; Li et al., 2024b). For each prompt, we sample six responses per group and identify all-negative-sample groups in which all responses yield incorrect answers. Within these groups, we apply the step-wise judge model to assign differentiated rewards to negative samples, which are then used for offline RL updates. The model is trained for three epochs with a learning rate of 2×10^{-6}. As a contrastive baseline, we also perform offline RL using only positive rollouts with correct answers. This parallel setup enables a direct comparison between learning from exclusively negative reasoning trajectories and from exclusively positive ones.

We conduct offline RL training to demonstrate that utilizing all-negative-sample groups can enhance the reasoning abilities of LLMs. For comparison, we also include positive-only offline RL training. As shown in Table 1, SGPO with negative samples improves average performance and performs competitively on most benchmarks, and in some cases even surpasses models trained solely on positive samples. In particular, in the 14B model experiment, training on negative samples yields improvements on four benchmarks relative to the positive-sample baseline. These findings underscore the utility of negative samples, which should not be discarded in online GRPO training; see further comments in Section 4.4.

4.2 Online Training

For baselines, we consider applying `Qwen2.5-14B-Instruct`, `Qwen2.5-32B-Instruct`, `QwQ-32B`, `DeepSeek-R1-Distill-Qwen-7B` and `DeepSeek-R1-Distill-Llama-8B`. Online GRPO training is imple-

Table 2: Evaluation results on online RL training. We refer to Baseline as the performance of the original model without RL finetuning. **Overall** is average performance across all the benchmarks. Note that the training dataset is `AIME1997-2023`. For `DeepSeek-R1-Distill-Qwen-7B`, we report additional results, including (i) compatibility with more judge models and (ii) ablation on the stability parameters β and γ.

	Kaoyan pass@1	GradeMath pass@1	MATH500 pass@1	Olympiads pass@1	CHMath24 avg@16	AIME25 avg@16	AIME24 avg@16	GaoKao avg@16	AMC23 avg@16	**Overall** avg
DeepSeek-R1-Distill-Qwen-7B										
Baseline	50.25	41.43	87.00	49.93	73.75	**40.62**	52.92	80.22	89.53	62.85
GRPO	55.78	43.33	89.40	**56.00**	71.04	36.68	52.08	80.30	88.91	63.72
SGPO+o4-mini-0416	57.79	46.19	90.80	54.67	75.00	38.33	54.58	81.33	90.00	65.41
SGPO+DeepSeek-V3-0324	54.77	**47.17**	91.00	55.11	**77.29**	40.42	**56.87**	**82.28**	90.83	**66.19**
SGPO+Qwen3-235B-A22B	56.78	46.67	**92.00**	54.67	73.33	37.92	55.63	81.17	90.63	65.42
SGPO+QwQ-32B	52.26	45.24	**92.00**	53.78	75.00	35.21	56.46	**82.28**	**91.88**	64.91
SGPO+QwQ-32B w/o $\{\beta,\gamma\}$	**58.29**	42.38	90.20	55.11	74.58	38.69	53.63	81.24	88.75	65.08
DeepSeek-R1-Distill-Llama-8B										
Baseline	29.15	23.81	77.40	41.48	**61.46**	27.92	**42.29**	**72.78**	87.97	51.58
GRPO	35.68	28.33	**84.00**	46.32	57.08	**28.33**	42.08	68.99	86.72	53.06
SGPO+Claude-3.7	**39.70**	**29.05**	83.60	**48.44**	58.96	24.58	39.37	71.52	**89.06**	**53.81**
Qwen2.5-14B-Instruct										
Baseline	37.69	49.52	80.40	41.78	21.88	13.13	**14.58**	**41.14**	58.59	39.85
GRPO	**43.22**	47.14	80.20	43.11	21.88	13.13	13.33	39.16	**59.84**	40.11
SGPO+o4-mini-0416	38.69	**53.33**	**81.00**	**44.00**	**22.92**	**16.67**	14.17	39.00	59.22	**41.00**
Qwen2.5-32B-Instruct										
Baseline	45.73	**53.81**	**83.60**	45.93	26.87	12.29	17.08	44.15	64.22	43.74
GRPO	**48.24**	52.86	83.20	45.93	22.50	12.08	**21.67**	**45.73**	67.34	44.39
SGPO+o4-mini-0416	**48.24**	**53.81**	83.00	**46.81**	**29.79**	**14.58**	19.58	45.09	**69.53**	**45.06**
QwQ-32B										
Baseline	64.32	62.38	94.60	68.74	**89.39**	**68.54**	77.71	86.88	97.03	78.84
GRPO	71.36	63.81	94.60	69.48	88.75	64.38	75.83	**87.11**	97.03	79.15
SGPO+DeepSeek-V3-0324	**73.37**	**64.76**	**95.00**	**70.22**	88.33	66.46	**78.33**	**87.11**	**97.97**	**80.17**

mented using the `verl` framework (Sheng et al., 2025). For the step-wise judge model, we adopt a diverse set of LLMs, ranging from closed-source models with strong reasoning capabilities (`o4-mini`, `Claude3.7`) to open-source models that are more accessible to the community, including `DeepSeek-V3-0324`, `Qwen3-235B-A22B`, and `QwQ-32B`.

Compared to offline RL, online RL yields larger improvements in a model's reasoning capabilities. Since our baselines already include strong distillation models, some benchmarks used in offline evaluation are nearing saturation. To provide a better assessment, we expand our evaluation suite beyond `AMC23`, `AIME24`, `MATH500`, and `OlympiadBench` by including `AIME25`, `GradeSchool` (Ye et al., 2025), `CHMath24`, `Kaoyan`, and `Gaokao`. Specifically, `CHMath24` is the benchmark from the 2024 Chinese High School Mathematics League Competition, `Gaokao` from China's 2024 National College Entrance Examination, `Kaoyan` from the Chinese Graduate School Entrance Examinations, and `GradeSchool` targets elementary-level mathematical reasoning. Among these, `CHMath24` and `Gaokao` each contain fewer than 100 questions, for which we apply the temperature-based decoding for evaluation.

For GRPO training, we use the `AIME` collections from 1997 to 2023 provided in DeepScaler (Luo et al., 2025b), training for 12 epochs. All training questions are in English, while evaluation benchmarks include multilingual questions. Notably, negative samples learned during training generalize well to out-of-domain mathematical reasoning tasks. SGPO training follows the same setup. With batch-simultaneous processing, judge model calls take 90 seconds per batch of negatives, adding 10% wall-clock time relative to rollout and update. Step-wise supervision is applied only to all-negative-sample groups during the first three epochs, as we expect this duration to suffice for the model to internalize corrective signals; beyond this point, unresolved examples are more indicative of model capacity limits than learnability. Accordingly, the end-to-end wall-clock overhead is upper bounded by approximately $10\% \times 3/12 = 2.5\%$, and in practice this overhead can be offset by a small reduction of time needed to reach a target performance. For all models, rollout length is fixed at 8192 tokens and group size at 8. Models less than 8B are trained on 8 H100, 14B models on 16 H100, and 32B models on 32 H200. We adopt the default KL coefficient and learning rate from the

Table 3: Evaluation results are reported for `DeepSeek-R1-Distill-Qwen-7B` across four independent runs. First column indicates judge models and its corresponding reward stability setup.

	Kaoyan pass@1	GradeMath pass@1	MATH500 pass@1	Olympiads pass@1	CHMath24 avg@16	AIME25 avg@16	AIME24 avg@16	GaoKao avg@16	AMC23 avg@16	Overall avg
DeepSeek-R1-Distill-Qwen-7B-SGPO										
+Qwen3-235B-A22B	53.90 ± 2.10	46.55 ± 0.24	91.30 ± 0.87	53.45 ± 1.35	74.48 ± 1.10	37.40 ± 1.05	55.73 ± 1.39	81.33 ± 0.18	90.19 ± 0.48	64.92 ± 0.37
+QwQ-32B	53.89 ± 1.66	44.88 ± 1.36	91.15 ± 0.84	53.71 ± 0.74	74.33 ± 1.29	37.03 ± 1.23	54.76 ± 1.87	81.91 ± 0.53	90.08 ± 1.23	64.64 ± 0.41
+QwQ-32B w/o $\{\beta,\gamma\}$	56.14 ± 2.76	44.53 ± 2.41	90.10 ± 0.66	53.64 ± 1.08	73.89 ± 1.13	38.70 ± 1.80	53.63 ± 2.15	81.24 ± 0.50	88.83 ± 0.81	64.52 ± 0.57

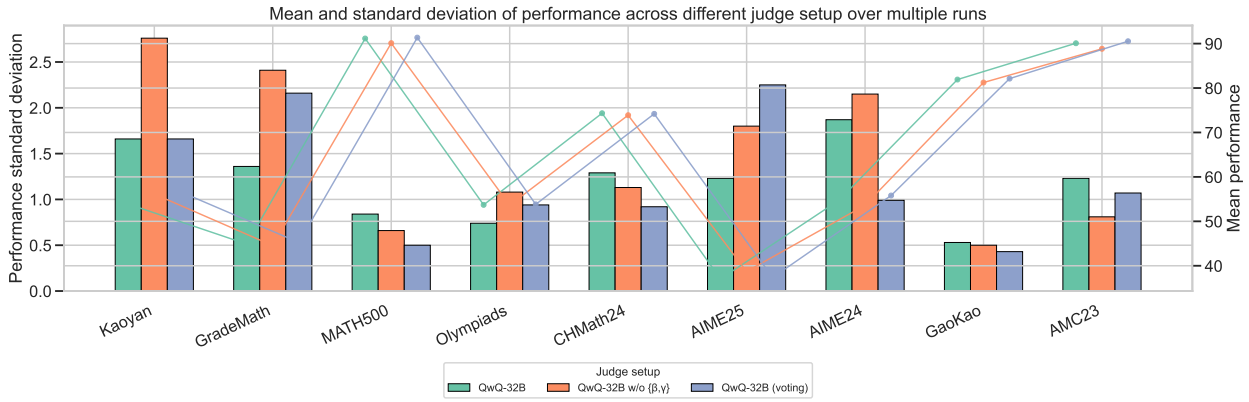

Figure 3: Mean and standard deviation over multiple runs across different judge model setups.

Table 4: Evaluation results are reported for `DeepSeek-R1-Distill-Qwen-7B` as base model and `QwQ-32B` as judge model with and without majority voting.

	Kaoyan pass@1	GradeMath pass@1	MATH500 pass@1	Olympiads pass@1	CHMath24 avg@16	AIME25 avg@16	AIME24 avg@16	GaoKao avg@16	AMC23 avg@16	Overall avg
DeepSeek-R1-Distill-Qwen-7B-SGPO										
+QwQ-32B	53.89 ± 1.66	44.88 ± 1.36	91.15 ± 0.84	53.71 ± 0.74	74.33 ± 1.29	37.03 ± 1.23	54.76 ± 1.87	81.91 ± 0.53	90.08 ± 1.23	64.64 ± 0.41
+QwQ-32B with voting	56.66 ± 1.66	45.24 ± 2.16	91.35 ± 0.50	53.82 ± 0.94	74.19 ± 0.92	37.35 ± 2.25	55.81 ± 0.99	82.12 ± 0.43	90.53 ± 1.07	65.23 ± 0.18

`verl` training script (Sheng et al., 2025), and use the LIMO evaluation script (Ye et al., 2025), both of which are standard practices in the community. That being said, these benefits are not uniform across every benchmark. When negative samples are short, highly noisy, or dominated by early derailments, step-wise judging provides less actionable signal and can even introduce additional variance through judge noise, so SGPO may match or occasionally underperform GRPO or the baseline on specific tasks. This pattern is consistent with our empirical results and motivates a more nuanced view: SGPO is most effective when the model's failures retain informative intermediate structure (e.g., truncated near-correct trajectories or localized errors), rather than unstructured breakdowns.

A key insight from Table 2 is that stronger models generate higher-quality negative samples, which aid learning. As model capability improves, so does the informativeness of its mistakes. Negative samples fall into two categories: (i) correct reasoning trajectories truncated by output length limits, and (ii) trajectories containing logical errors. The first type remains highly valuable – yet discarded in GRPO – since it preserves meaningful reasoning steps, motivating our step-wise judge model. The second type, though incorrect, still provides informative signals, especially when all samples fail on genuinely challenging problems. Notably, stronger distilled models average 6K tokens per response, compared to only 1K tokens for weaker base models, making truncated but informative negative samples more common in the stronger case. Likewise, their erroneous responses tend to be richer and more useful for step-level judgment.

Table 5: Evaluation results are reported in terms of `pass@16` across benchmarks. The first two columns show the total number of questions and the number solved within 16 attempts, while the last two columns report the number of questions solved by SGPO but not by GRPO (SGPO\GRPO), and vice versa (GRPO\SGPO).

	SGPO - `pass@16`	GRPO - `pass@16`	SGPO \ GRPO	GRPO \ SGPO
`AIME24`	23/30	19/30	4	0
`AIME25`	21/30	21/30	1	1
`Gaokao`	70/79	68/79	2	0
`AMC23`	39/40	38/40	1	0
`CHMath24`	27/30	25/30	2	0

4.3 Other Ablation Studies

To assess reliability of judge models, we evaluate our approach not only with strong closed-source reasoning models but also with publicly available models of weaker capacity: `DeepSeek-V3`, `Qwen3-235B` and `QwQ-32B`. As shown in Table 2 (best-tuned results) and Table 3 (multiple runs with weaker judges), performance remains stable, indicating that weaker judges do not significantly degrade outcomes. We attribute this reliability to two design choices: (i) first-step error identification with the reference answer. SGPO requires the judge only to verify each step against the reference, not to solve the problem, thereby reducing task difficulty and avoiding the pitfalls of generic PRMs; (ii) reward stability parameters β and γ, which set the update inertia and reduce sensitivity to rewards from earlier failed rollouts. As confirmed by ablations, removing β and γ increases variance and weakens performance. To improve verification, we incorporate a Grok4−Heavy-inspired strategy: multiple independent evaluations by the judge model, with the error position selected by majority voting. Using `QwQ-32B` as the judge model, `DeepSeek-R1-Distill-Qwen-7B` as the base model, and four rollouts per judgment, we observed noticeable gains in consistency and stability (see Table 4). Figure 3 visualizes the aggregated results under different judge modes from Tables 3 and 4.

Although `avg@16` measures average performance across rollouts, `pass@16` reflects the ability to solve new questions with multiple attempts. As shown in Table 5, SGPO's gains in `pass@16` stem directly from leveraging negative samples. Learning only from solvable problems reinforces existing ability, whereas all-negative-sample groups correspond to genuinely difficult questions where GRPO consistently fails. These are precisely the cases where additional feedback can be most valuable. By providing step-level signals, SGPO rewards near-misses by reinforcing correct reasoning up to the first error, penalizes early failures by discouraging persistent error modes, and exposes blind spots by turning hard cases into informative training signals. In this regard, SGPO can provide additional benefits over GRPO, covering more hard problems and providing sharper credit assignment, which translates to more reliable learning under realistic compute budgets.

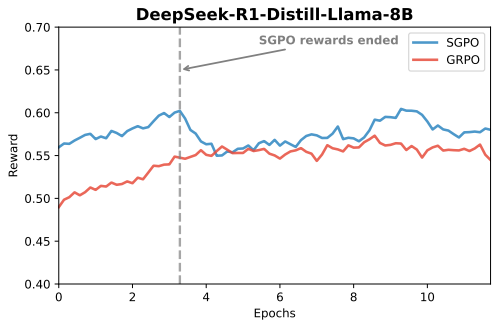

Figure 4: Evaluation results on GRPO and SGPO. SGPO rewards end at epoch 3.

By leveraging richer early-stage signals from negative samples, SGPO can achieve competitive performance with GRPO, and in some cases better performance or improved coverage on hard problems. As shown in Figure 4, SGPO continues improving beyond epoch 5 by solving several additional hard training problems, whereas GRPO plateaus. This improvement stems from informative negative samples that help resolve previously unsolved problems as also shown in Table 5. In line with our theoretical finding that SGPO converges faster than GRPO, empirical metrics offer supporting evidence. Prior work on RLVR entropy highlights its link to performance: Cui et al. (2025) showed that lower policy entropy under correct signals correlates with stronger policies, while Agarwal et al. (2025) demonstrated that directly minimizing entropy can improve performance. As shown in Figure 5, SGPO reduces policy entropy more rapidly than GRPO,

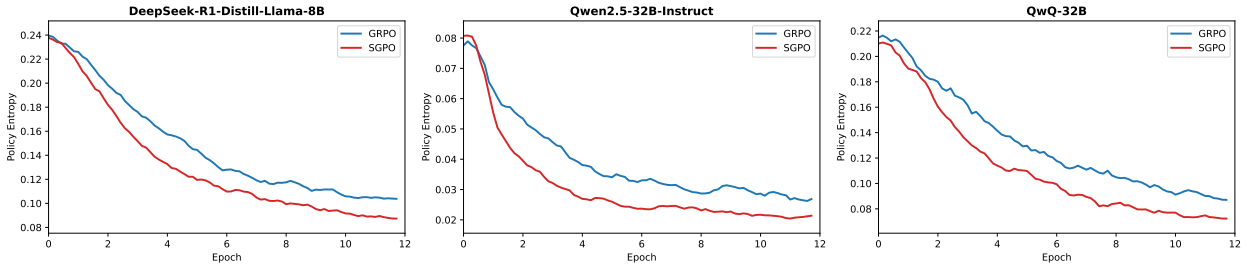

Figure 5: Policy entropy levels during training for GRPO and SGPO across different base models.

indicating faster convergence toward deterministic RLVR behavior with higher rollout confidence. This matches our theoretical results, confirming that step-wise signals accelerate convergence.

4.4 Discussions

We highlight the motivation for evaluating both offline and online RL. In the offline setup, training uses **only** negative samples, allowing us to directly test whether incorrect or incomplete reasoning trajectories can improve performance. In the online setup, we simulate realistic GRPO training, where batches contain a random mix of positive and negative samples. This demonstrates that negative samples are not only effective in isolation but also remain valuable in practical settings with noisier, mixed data. While mixing positives and negatives introduces noise, simply discarding negative samples does not stabilize training; in several cases, the performance of GRPO drops below baseline, as the models overfit to problems they can solve.

This instability arises from limited out-of-domain generalization and catastrophic forgetting. Without exposure to challenging or partially correct reasoning, the model risks overfitting to easy cases, reinforcing shallow heuristics instead of developing robust problem-solving skills. The absence of diverse failure cases can also cause catastrophic forgetting, degrading performance on previously solvable tasks. Incorporating negative samples mitigates these issues, and SGPO shows stronger robustness on the Chinese OOD math benchmarks we evaluate. We emphasize that SGPO does not guarantee uniform improvements on every benchmark: its gains depend on the structure of negative samples. SGPO is most beneficial when failures are due to truncated but largely correct trajectories or localized mistakes, whereas when negative samples fail very early or lack meaningful structure, the step-wise signal can be less informative and gains may diminish. These observations motivate further work on more stable training frameworks, including richer reward diversification mechanisms for handling negative samples and efficient RL methods beyond GRPO.

Benchmarking scope. Our goal is to isolate the benefit of learning from all-negative-sample groups in outcome-based, verifiable-reward post-training with group-relative updates. We therefore benchmark SGPO primarily against GRPO under matched training pipelines (same base models, data, rollout budget, group size, KL control, and optimizer settings), since SGPO is designed as a drop-in modification of GRPO's credit assignment only in all-negative groups. This choice makes the comparison controlled: improvements can be attributed to step-wise differentiation of negative samples rather than to changes in the broader RL recipe.

A natural question is how SGPO compares to methods that rely on process reward models (PRMs) or other step-level scoring mechanisms. Such comparisons would indeed further strengthen the empirical picture, but they introduce additional moving parts (PRM training data/labels, architecture, calibration, and inference overhead) that are orthogonal to our main question: given an outcome-verifiable setting and GRPO-style updates, can we recover learning signal from all-negative groups by cheaply localizing the first error? In this work, we keep the benchmark focused on GRPO and its outcome-based setting, and view PRM-based pipelines as complementary directions for future evaluation, especially when reliable PRMs and their training recipes are available and standardized. That said, we acknowledge that broader benchmarking (e.g., PRM-based pipelines) is an important next step, but given the recency and centrality of GRPO in outcome-based RLVR, a controlled GRPO-centered comparison is sufficient to establish relevance for the target setting.

5 Conclusion

We propose a simple and efficient framework that introduces response diversity within all-negative-sample groups and prove, in a simplified setting, that such diversification can accelerate the learning dynamic of GRPO. Empirically, SGPO improves average performance and improves coverage on harder problems, with the largest benefits in early and mid-training when all-negative groups are prevalent; gains are not uniform across benchmarks and depend on the structure/informativeness of negative samples. Future works include extending theoretical results to broader multi-step reasoning tasks, applying response diversity to accelerate other RL methods, and designing lightweight, task-specific reward models that evaluate reasoning steps correctly even if they cannot solve the full problem.

Broader Impacts

Although large reasoning models can enable substantial social and economic benefits, a growing literature also highlights potential negative impacts, including environmental costs from training and inference, concentration of power and access, labor displacement, and downstream misuse (e.g., for manipulation or disinformation). Our work improves the post-training procedure for such models; as a result, it may contribute to both beneficial applications and accelerated adoption, which can amplify these broader concerns. We emphasize that SGPO is a technical contribution and does not address these systemic risks directly. Responsible deployment should consider energy and computing reporting, access and governance practices, and safeguards against misuse. More broadly, we view a rigorous assessment of societal impacts alongside technical progress as essential.

Acknowledgment

We sincerely appreciate Buzz High Performance Computing (https://www.buzzhpc.ai, info@buzzhpc.ai) for providing computational resources and support for this work.

References

J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. GPT-4 technical report. *ArXiv Preprint: 2303.08774*, 2023.

A. Agarwal, S. M. Kakade, J. D. Lee, and G. Mahajan. Optimality and approximation with policy gradient methods in markov decision processes. In *COLT*, pp. 64–66, 2020.

S. Agarwal, Z. Zhang, L. Yuan, J. Han, and H. Peng. The unreasonable effectiveness of entropy minimization in LLM reasoning. *ArXiv Preprint: 2505.15134*, 2025.

A. Ahmadian, C. Cremer, M. Gallé, M. Fadaee, J. Kreutzer, O. Pietquin, A. Üstün, and S. Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *ACL*, pp. 12248–12267, 2024.

D. Arora and A. Zanette. Training language models to reason efficiently. *ArXiv Preprint: 2502.04463*, 2025.

M. G. Azar, Z. Guo, B. Piot, R. Munos, M. Rowland, M. Valko, and D. Calandriello. A general theoretical paradigm to understand learning from human preferences. In *AISTATS*, pp. 4447–4455, 2024.

Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, et al. Constitutional AI: Harmlessness from AI feedback. *ArXiv Preprint: 2212.08073*, 2022.

M. Besta, N. Blach, A. Kubicek, R. Gerstenberger, M. Podstawski, L. Gianinazzi, J. Gajda, T. Lehmann, H. Niewiadomski, P. Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *AAAI*, pp. 17682–17690, 2024.

X. Bi, D. Chen, G. Chen, S. Chen, D. Dai, C. Deng, H. Ding, K. Dong, Q. Du, Z. Fu, et al. Deepseek LLM: Scaling open-source language models with longtermism. *ArXiv Preprint: 2401.02954*, 2024.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. In *NeurIPS*, pp. 1877–1901, 2020.

H. Chen, G. He, L. Yuan, G. Cui, H. Su, and J. Zhu. Noise contrastive alignment of language models with explicit rewards. In *NeurIPS*, pp. 117784–117812, 2024a.

H. Chen, H. Zhao, H. Lam, D. Yao, and W. Tang. MallowsPO: Fine-tune your LLM with preference dispersions. In *ICLR*, 2025a. URL https://openreview.net/forum?id=d8cnezVcaW.

P. Chen, X. Chen, W. Yin, and T. Lin. ComPO: Preference alignment via comparison oracles. *ArXiv Preprint: 2505.05465*, 2025b.

P. Chen, X. Li, X. Chen, and T. Lin. Reward-free alignment for conflicting objectives. *ArXiv Preprint: 2602.02495*, 2026a.

P. Chen, X. Li, Z. Li, W. Yin, X. Chen, and T. Lin. Exploration vs exploitation: Rethinking RLVR through clipping, entropy, and spurious reward. In *ICLR*, 2026b. URL https://openreview.net/forum?id=sE8DCSJTzd.

Q. Chen, L. Qin, J. Liu, D. Peng, J. Guan, P. Wang, M. Hu, Y. Zhou, T. Gao, and W. Che. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *ArXiv Preprint: 2503.09567*, 2025c.

X. Chen, J. Xu, T. Liang, Z. He, J. Pang, D. Yu, L. Song, Q. Liu, M. Zhou, Z. Zhang, et al. Do not think that much for 2+3 = ? on the overthinking of o1-like LLMs. *ArXiv Preprint: 2412.21187*, 2024b.

J. Cheng and B. Van Durme. Compressed chain of thought: Efficient reasoning through dense representations. *ArXiv Preprint: 2412.13171*, 2024.

D. R. Chialvo and P. Bak. Learning from mistakes. *Neuroscience*, 90(4):1137–1148, 1999.

A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. In *NeurIPS*, pp. 4302–4310, 2017.

X. Chu, H. Huang, X. Zhang, F. Wei, and Y. Wang. GPG: A simple and strong reinforcement learning baseline for model reasoning. *ArXiv Preprint: 2504.02546*, 2025.

K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al. Training verifiers to solve math word problems. *ArXiv Preprint: 2110.14168*, 2021.

G. Cui, Y. Zhang, J. Chen, L. Yuan, Z. Wang, Y. Zuo, H. Li, Y. Fan, H. Chen, W. Chen, et al. The entropy mechanism of reinforcement learning for reasoning language models. *ArXiv Preprint: 2505.22617*, 2025.

P. Dayan. Reinforcement comparison. In *Connectionist Models*, pp. 45–51. Elsevier, 1991.

B. Ding, C. Qin, L. Liu, Y. K. Chia, B. Li, S. Joty, and L. Bing. Is GPT-3 a good data annotator? In *ACL*, pp. 11173–11195, 2023.

H. Dong, W. Xiong, D. Goyal, Y. Zhang, W. Chow, R. Pan, S. Diao, J. Zhang, K. Shum, and T. Zhang. RAFT: Reward ranked fine-tuning for generative foundation model alignment. *Transactions on Machine Learning Research*, 2023. URL https://openreview.net/forum?id=m7p5O7zblY.

H. Dong, W. Xiong, B. Pang, H. Wang, H. Zhao, Y. Zhou, N. Jiang, D. Sahoo, C. Xiong, and T. Zhang. RLHF workflow: From reward modeling to online RLHF. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=a13aYUU9eU.

K. Ethayarajh, W. Xu, N. Muennighoff, D. Jurafsky, and D. Kiela. Model alignment as prospect theoretic optimization. In *ICML*, pp. 12634–12651, 2024.

G. Feng, B. Zhang, Y. Gu, H. Ye, D. He, and L. Wang. Towards revealing the mystery behind chain of thought: A theoretical perspective. In *NeurIPS*, pp. 70757–70798, 2023.

K. Gandhi, D. Lee, G. Grand, M. Liu, W. Cheng, A. Sharma, and N. D. Goodman. Stream of search (SOS): Learning to search in language. *ArXiv Preprint: 2404.03683*, 2024.

L. Gao, J. Schulman, and J. Hilton. Scaling laws for reward model overoptimization. In *ICML*, pp. 10835–10866, 2023.

F. Gilardi, M. Alizadeh, and M. Kubli. ChatGPT outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120, 2023.

Z. Gou, Z. Shao, Y. Gong, Y. Shen, Y. Yang, M. Huang, N. Duan, and W. Chen. ToRA: A tool-integrated reasoning agent for mathematical problem solving. In *ICLR*, 2024. URL https://openreview.net/forum?id=Ep0TtjVoap.

Y. Gu, L. Dong, F. Wei, and M. Huang. MiniLLM: Knowledge distillation of large language models. In *ICLR*, 2024. URL https://openreview.net/forum?id=5h0qf7IBZZ.

D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *ArXiv Preprint: 2501.12948*, 2025a.

J. Guo, Y. Wu, J. Qiu, K. Huang, X. Juan, L. Yang, and M. Wang. Temporal consistency for LLM reasoning process error identification. *ArXiv Preprint: 2503.14495*, 2025b.

J. Guo, L. Yang, P. Chen, Q. Xiao, Y. Wang, X. Juan, J. Qiu, K. Shen, and W. Wang. GenEnv: Difficulty-aligned co-evolution between LLM agents and environment simulators. *ArXiv Preprint: 2512.19682*, 2025c.

S. Hao, Y. Gu, H. Ma, J. Hong, Z. Wang, D. Wang, and Z. Hu. Reasoning with language model is planning with world model. In *EMNLP*, pp. 8154–8173, 2023.

S. Hao, Y. Gu, H. Luo, T. Liu, X. Shao, X. Wang, S. Xie, H. Ma, A. Samavedhi, Q. Gao, Z. Wang, and Z. Hu. LLM reasoners: New evaluation, library, and analysis of step-by-step reasoning with large language models. In *COLM*, 2024a. URL https://openreview.net/forum?id=b0y6fbSUG0.

S. Hao, S. Sukhbaatar, D. Su, X. Li, Z. Hu, J. Weston, and Y. Tian. Training large language models to reason in a continuous latent space. *ArXiv Preprint: 2412.06769*, 2024b.

A. Havrilla, Y. Du, S. C. Raparthy, C. Nalmpantis, J. Dwivedi-Yu, M. Zhuravinskyi, E. Hambro, S. Sukhbaatar, and R. Raileanu. Teaching large language models to reason with reinforcement learning. *ArXiv Preprint: 2403.04642*, 2024.

C. He, R. Luo, Y. Bai, S. Hu, Z. Thai, J. Shen, J. Hu, X. Han, Y. Huang, Y. Zhang, et al. OlympiadBench: A challenging benchmark for promoting AGI with Olympiad-level bilingual multimodal scientific problems. In *ACL*, pp. 3828–3850, 2024.

T. He, R. Mu, L. Liao, Y. Cao, M. Liu, and B. Qin. Good learners think their thinking: Generative PRM makes large reasoning model more efficient math learner. *ArXiv Preprint: 2507.23317*, 2025.

D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS Datasets and Benchmarks Track*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

J. Hong, N. Lee, and J. Thorne. ORPO: Monolithic preference optimization without reference model. In *EMNLP*, pp. 11170–11189, 2024.

A. Hosseini, X. Yuan, N. Malkin, A. Courville, A. Sordoni, and R. Agarwal. V-STar: Training verifiers for self-taught reasoners. In *COLM*, 2024. URL https://openreview.net/forum?id=stmqBSW2dV.

J. Huang and K. C-C. Chang. Towards reasoning in large language models: A survey. In *ACL*, pp. 1049–1065, 2023.

K. Huang, J. Guo, Z. Li, X. Ji, J. Ge, W. Li, Y. Guo, T. Cai, H. Yuan, R. Wang, Y. Wu, M. Yin, S. Tang, Y. Huang, C. Jin, X. Chen, C. Zhang, and M. Wang. MATH-Perturb: Benchmarking LLMs' math reasoning abilities against hard perturbations. *ArXiv Preprint: 2502.06453*, 2025.

S. Huang, Z. Ma, J. Du, C. Meng, W. Wang, and Z. Lin. Mirror-consistency: Harnessing inconsistency in majority voting. In *EMNLP*, pp. 2408–2420, 2024.

H. Hwang, D. Kim, S. Kim, S. Ye, and M. Seo. Self-explore: Enhancing mathematical reasoning in language models with fine-grained rewards. In *EMNLP*, pp. 1444–1466, 2024.

A. Jaech, A. Kalai, A. Lerer, A. Richardson, A. El-Kishky, A. Low, A. Helyar, A. Madry, A. Beutel, A. Carney, et al. OpenAI o1 system card. *ArXiv Preprint: 2412.16720*, 2024.

N. Jain, K. Han, A. Gu, W-D. Li, F. Yan, T. Zhang, S. Wang, A. Solar-Lezama, K. Sen, and I. Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *ICLR*, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

M. Kang, S. Lee, J. Baek, K. Kawaguchi, and S. J. Hwang. Knowledge-augmented reasoning distillation for small language models in knowledge-intensive tasks. In *NeurIPS*, pp. 48573–48602, 2023.

A. Kazemnejad, M. Aghajohari, E. Portelance, A. Sordoni, S. Reddy, A. Courville, and N. Le Roux. VinePPO: Accurate credit assignment in RL for LLM mathematical reasoning. In *The NeurIPS Workshop on Mathematical Reasoning and AI*, 2024. URL https://openreview.net/forum?id=KqALqWJSbF.

T. Khot, H. Trivedi, M. Finlayson, Y. Fu, K. Richardson, P. Clark, and A. Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. In *ICLR*, 2023. URL https://openreview.net/forum?id=_nGgzQjzaRy.

M. Kwon, S. M. Xie, K. Bullard, and D. Sadigh. Reward design with language models. In *ICLR*, 2023. URL https://openreview.net/forum?id=10uNUgI5Kl.

A. K. Lampinen, I. Dasgupta, S. C. Y. Chan, H. R. Sheahan, A. Creswell, D. Kumaran, J. L. McClelland, and F. Hill. Language models, like humans, show content effects on reasoning tasks. *PNAS Nexus*, 3(7): pgae233, 2024.

Y. LeCun. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1): 1–62, 2022.

H. Lee, S. Phatale, H. Mansoor, T. Mesnard, J. Ferret, K. Lu, C. Bishop, E. Hall, V. Carbune, A. Rastogi, et al. RLAIF vs. RLHF: Scaling reinforcement learning from human feedback with AI feedback. In *ICML*, pp. 26874–26901, 2024.

L. Lehnert, S. Sukhbaatar, D. Su, Q. Zheng, P. McVay, M. Rabbat, and Y. Tian. Beyond $A^\star$: Better planning with transformers via search dynamics bootstrapping. In *COLM*, 2024. URL https://openreview.net/forum?id=SGoVIC0u0f.

Z. Li, H. Liu, D. Zhou, and T. Ma. Chain of thought empowers transformers to solve inherently serial problems. In *ICLR*, 2024a. URL https://openreview.net/forum?id=3EWTEy9MTM.

Z. Li, T. Xu, Y. Zhang, Z. Lin, Y. Yu, R. Sun, and Z-Q. Luo. ReMax: A simple, effective, and efficient reinforcement learning method for aligning large language models. In *ICML*, pp. 29128–29163, 2024b.

Z. Li, C. Chen, T. Xu, Z. Qin, J. Xiao, Z-Q. Luo, and R. Sun. Preserving diversity in supervised fine-tuning of large language models. In *ICLR*, 2025. URL https://openreview.net/forum?id=NQEe7B7bSw.

H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe. Let's verify step by step. In *ICLR*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

T. Liu, Y. Zhao, R. Joshi, M. Khalman, M. Saleh, P. J. Liu, and J. Liu. Statistical rejection sampling improves preference optimization. In *ICLR*, 2024a. URL https://openreview.net/forum?id=xbjSwwrQOe.

T. Liu, Z. Qin, J. Wu, J. Shen, M. Khalman, R. Joshi, Y. Zhao, M. Saleh, S. Baumgartner, J. Liu, et al. LiPO: Listwise preference optimization through learning-to-rank. In *NAACL*, pp. To appear, 2025a.

Z. Liu, M. Lu, S. Zhang, B. Liu, H. Guo, Y. Yang, J. Blanchet, and Z. Wang. Provably mitigating overoptimization in RLHF: Your SFT loss is implicitly an adversarial regularizer. In *NeurIPS*, pp. 138663–138697, 2024b.

Z. Liu, C. Chen, W. Li, P. Qi, T. Pang, C. Du, W. S. Lee, and M. Lin. Understanding R1-zero-like training: A critical perspective. *ArXiv Preprint: 2503.20783*, 2025b.

H. Luo, L. Shen, H. He, Y. Wang, S. Liu, W. Li, N. Tan, X. Cao, and D. Tao. o1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning. *ArXiv Preprint: 2501.12570*, 2025a.

L. Luo, Y. Liu, R. Liu, S. Phatale, M. Guo, H. Lara, Y. Li, L. Shu, Y. Zhu, L. Meng, et al. Improve mathematical reasoning in language models by automated process supervision. *ArXiv Preprint: 2406.06592*, 2024.

M. Luo, S. Tan, J. Wong, X. Shi, W. Y. Tang, M. Roongta, C. Cai, J. Luo, L. E. Li, R. A. Popa, and I. Stoica. DeepScaleR: Surpassing o1-preview with a 1.5B model by scaling RL. https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a4e2, 2025b. Notion Blog.

A. Madaan, K. Hermann, and A. Yazdanbakhsh. What makes chain-of-thought prompting effective? a counterfactual study. In *EMNLP*, pp. 1448–1535, 2023.

J. Mei, Y. Gao, B. Dai, C. Szepesvari, and D. Schuurmans. Leveraging non-uniformity in first-order non-convex optimization. In *ICML*, pp. 7555–7564, 2021.

Y. Meng, M. Xia, and D. Chen. SimPO: Simple preference optimization with a reference-free reward. In *NeurIPS*, pp. 124198–124235, 2024.

W. Merrill and A. Sabharwal. The expressive power of transformers with chain of thought. In *ICLR*, 2024. URL https://openreview.net/forum?id=NjNGlPh8Wh.

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. In *NeurIPS*, pp. 27730–27744, 2022.

A. Pal, D. Karkhanis, S. Dooley, M. Roberts, S. Naidu, and C. White. Smaug: Fixing failure modes of preference optimisation with DPO-positive. *ArXiv Preprint: 2402.13228*, 2024.

R. Y. Pang, W. Yuan, H. He, K. Cho, S. Sukhbaatar, and J. Weston. Iterative reasoning preference optimization. In *NeurIPS*, pp. 116617–116637, 2024.

R. Park, R. Rafailov, S. Ermon, and C. Finn. Disentangling length from quality in direct preference optimization. In *ACL*, pp. 4998–5017, 2024.

X. Peng, A. Kumar, G. Zhang, and S. Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *ArXiv Preprint: 1910.00177*, 2019.

J. Peters and S. Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *ICML*, pp. 745–750, 2007.

R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, pp. 53728–53741, 2023.

R. Rafailov, J. Hejna, R. Park, and C. Finn. From r to q^*: Your language model is secretly a Q-function. In *COLM*, 2024. URL https://openreview.net/forum?id=kEVcNxtqXk.

N. Razin, S. Malladi, A. Bhaskar, D. Chen, S. Arora, and B. Hanin. Unintentional unalignment: Likelihood displacement in direct preference optimization. In *ICLR*, 2025. URL https://openreview.net/forum?id=uaMSBJDnRv.

P. Roit, J. Ferret, L. Shani, R. Aharoni, G. Cideron, R. Dadashi, M. Geist, S. Girgin, L. Hussenot, O. Keller, et al. Factually consistent summarization via reinforcement learning with textual entailment feedback. In *ACL*, pp. 6252–6272, 2023.

J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel. Trust region policy optimization. In *ICML*, pp. 1889–1897, 2015.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *ArXiv Preprint: 1707.06347*, 2017.

A. Setlur, C. Nagpal, A. Fisch, X. Geng, J. Eisenstein, R. Agarwal, A. Agarwal, J. Berant, and A. Kumar. Rewarding progress: Scaling automated process verifiers for LLM reasoning. In *ICLR*, 2025. URL https://openreview.net/forum?id=A6Y7AqlzLW.

Amrith Setlur, Saurabh Garg, Xinyang Geng, Naman Garg, Virginia Smith, and Aviral Kumar. RL on incorrect synthetic data scales the efficiency of LLM math reasoning by eight-fold. In *NeurIPS*, pp. 43000–43031, 2024.

Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. K. Li, Y. Wu, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *ArXiv Preprint: 2402.03300*, 2024.

G. Sheng, C. Zhang, Z. Ye, X. Wu, W. Zhang, R. Zhang, Y. Peng, H. Lin, and C. Wu. HybridFlow: A flexible and efficient RLHF framework. In *EuroSys*, pp. 1279–1297. ACM, 2025.

B. Shi, M. Tang, K. R. Narasimhan, and S. Yao. Can language models solve Olympiad programming? In *COLM*, 2024. URL https://openreview.net/forum?id=kGa4fMtP9l.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

A. Singh, J. D. Co-Reyes, R. Agarwal, A. Anand, P. Patil, X. Garcia, P. J. Liu, J. Harrison, J. Lee, K. Xu, A. T. Parisi, A. Kumar, A. A. Alemi, A. Rizkowsky, A. Nova, B. Adlam, B. Bohnet, G. F. Elsayed, H. Sedghi, I. Mordatch, I. Simpson, I. Gur, J. Snoek, J. Pennington, J. Hron, K. Kenealy, K. Swersky, K. Mahajan, L. A. Culp, L. Xiao, M. Bileschi, N. Constant, R. Novak, R. Liu, T. Warkentin, Y. Bansal, E. Dyer, B. Neyshabur, J. Sohl-Dickstein, and N. Fiedel. Beyond human data: Scaling self-training for problem-solving with language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=lNAyUngGFK. Expert Certification.

I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg. ProgPrompt: Generating situated robot task plans using large language models. In *ICRA*, pp. 11523–11530. IEEE, 2023.

J. Skalse, N. H. R. Howe, D. Krasheninnikov, and D. Krueger. Defining and characterizing reward hacking. In *NeurIPS*, pp. 9460–9471, 2022.

C. V. Snell, J. Lee, K. Xu, and A. Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *ICLR*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

F. Song, B. Yu, M. Li, H. Yu, F. Huang, Y. Li, and H. Wang. Preference ranking optimization for human alignment. In *AAAI*, pp. 18990–18998, 2024.

S. Srivastava, A. PV, S. Menon, A. Sukumar, A. Philipose, S. Prince, S. Thomas, et al. Functional benchmarks for robust evaluation of reasoning performance, and the reasoning gap. *ArXiv Preprint: 2402.19450*, 2024.

N. Stiennon, L. Ouyang, J. Wu, D. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano. Learning to summarize with human feedback. In *NeurIPS*, pp. 3008–3021, 2020.

D. Su, S. Sukhbaatar, M. Rabbat, Y. Tian, and Q. Zheng. Dualformer: Controllable fast and slow thinking by learning with randomized reasoning traces. In *ICLR*, 2025. URL https://openreview.net/forum?id=bmbRCRiNDu.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*, volume 1. MIT Press, 1998.

F. Tajwar, A. Singh, A. Sharma, R. Rafailov, J. Schneider, T. Xie, S. Ermon, C. Finn, and A. Kumar. Preference fine-tuning of LLMs should leverage suboptimal, on-policy data. In *ICML*, pp. 47441–47474, 2024.

Y. Tang, Z. Guo, Z. Zheng, D. Calandriello, R. Munos, M. Rowland, P. H. Richemond, M. Valko, B. Pires, and B. Piot. Generalized preference optimization: A unified approach to offline alignment. In *ICML*, pp. 47725–47742, 2024.

K. Team, A. Du, B. Gao, B. Xing, C. Jiang, C. Chen, C. Li, C. Xiao, C. Du, C. Liao, et al. Kimi k1.5: Scaling reinforcement learning with LLMs. *ArXiv Preprint: 2501.12599*, 2025.

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *ArXiv Preprint: 2302.13971*, 2023.

J. Uesato, N. Kushman, R. Kumar, F. Song, N. Siegel, L. Wang, A. Creswell, G. Irving, and I. Higgins. Solving math word problems with process-and outcome-based feedback. *ArXiv Preprint: 2211.14275*, 2022.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, pp. 6000–6010, 2017.

H. Wang, C. Qian, W. Zhong, X. Chen, J. Qiu, S. Huang, B. Jin, M. Wang, K-F. Wong, and H. Ji. OTC: Optimal tool calls via reinforcement learning. *ArXiv Preprint: 2504.14870*, 2025.

P. Wang, L. Li, Z. Shao, R. Xu, D. Dai, Y. Li, D. Chen, Y. Wu, and Z. Sui. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *ACL*, pp. 9426–9439, 2024.

X. Wang, J. Wei, D. Schuurmans, Q. V. Le, E. H. Chi, S. Narang, A. Chowdhery, and D. Zhou. Self-consistency improves chain of thought reasoning in language models. In *ICLR*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, pp. 24824–24837, 2022.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

Y. Wu, Z. Sun, S. Li, S. Welleck, and Y. Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for LLM problem-solving. In *ICLR*, 2025. URL https://openreview.net/forum?id=VNckp7JEHn.

V. Xiang, C. Snell, K. Gandhi, A. Albalak, A. Singh, C. Blagden, D. Phung, R. Rafailov, N. Lile, D. Mahan, et al. Towards system 2 reasoning in LLMs: Learning how to think with meta chain-of-though. *ArXiv Preprint: 2501.04682*, 2025.

J. Xiao, Z. Li, X. Xie, E. Getzen, C. Fang, Q. Long, and W. J. Su. On the algorithmic bias of aligning large language models with RLHF: Preference collapse and matching regularization. *ArXiv Preprint: 2405.16455*, 2024.

Y. Xie, K. Kawaguchi, Y. Zhao, J. X. Zhao, M-Y. Kan, J. He, and M. Q. Xie. Self-evaluation guided beam search for reasoning. In *NeurIPS*, pp. 41618–41650, 2023.

W. Xiong, H. Dong, C. Ye, Z. Wang, H. Zhong, H. Ji, N. Jiang, and T. Zhang. Iterative preference learning from human feedback: Bridging theory and practice for RLHF under KL-constraint. In *ICML*, pp. 54715–54754, 2024.

W. Xiong, J. Yao, Y. Xu, B. Pang, L. Wang, D. Sahoo, J. Li, N. Jiang, T. Zhang, C. Xiong, and H. Dong. A minimalist approach to LLM reasoning: From rejection sampling to reinforce. *ArXiv Preprint: 2504.11343*, 2025a.

W. Xiong, W. Zhao, W. Yuan, O. Golovneva, T. Zhang, J. Weston, and S. Sukhbaatar. StepWiser: Stepwise generative judges for wiser reasoning. *ArXiv Preprint: 2508.19229*, 2025b.

H. Xu, A. Sharaf, Y. Chen, W. Tan, L. Shen, B. Van Durme, K. Murray, and Y. J. Kim. Contrastive preference optimization: Pushing the boundaries of LLM performance in machine translation. In *ICML*, pp. 55204–55224, 2024.

Y. Yang, D. Campbell, K. Huang, M. Wang, J. Cohen, and T. Webb. Emergent symbolic mechanisms support abstract reasoning in large language models. *ArXiv Preprint: 2502.20332*, 2025.

Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov, and C. D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, 2018.

S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *NeurIPS*, pp. 11809–11822, 2023.

Y. Ye, Z. Huang, Y. Xiao, E. Chern, S. Xia, and P. Liu. LIMO: Less is more for reasoning. *ArXiv Preprint: 2502.03387*, 2025.

F. Yu, A. Gao, and B. Wang. OVM, Outcome-supervised value models for planning in mathematical reasoning. In *NAACL*, pp. 858–875, 2024a.

F. Yu, L. Jiang, H. Kang, S. Hao, and L. Qin. Flow of reasoning: Efficient training of LLM policy with divergent thinking. In *ICML*, pp. To appear, 2025a.

L. Yu, W. Jiang, H. Shi, J. Yu, Z. Liu, Y. Zhang, J. Kwok, Z. Li, A. Weller, and W. Liu. MetaMath: Bootstrap your own mathematical questions for large language models. In *ICLR*, 2024b. URL https://openreview.net/forum?id=N8N0hgNDRt.

Q. Yu, Z. Zhang, R. Zhu, Y. Yuan, X. Zuo, Y. Yue, T. Fan, G. Liu, L. Liu, X. Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *ArXiv Preprint: 2503.14476*, 2025b.

H. Yuan, Z. Yuan, C. Tan, W. Wang, S. Huang, and F. Huang. RRHF: Rank responses to align language models with human feedback. In *NeurIPS*, pp. 10935–10950, 2023a.

L. Yuan, G. Cui, H. Wang, N. Ding, X. Wang, B. Shan, Z. Liu, J. Deng, H. Chen, R. Xie, Y. Lin, Z. Liu, B. Zhou, H. Peng, Z. Liu, and M. Sun. Advancing LLM reasoning generalists with preference trees. In *ICLR*, 2025. URL https://openreview.net/forum?id=2ea5TNVR0c.

Z. Yuan, H. Yuan, C. Li, G. Dong, K. Lu, C. Tan, C. Zhou, and J. Zhou. Scaling relationship on learning mathematical reasoning with large language models. *ArXiv Preprint: 2308.01825*, 2023b.

X. Yue, X. Qu, G. Zhang, Y. Fu, W. Huang, H. Sun, Y. Su, and W. Chen. MAmmoTH: Building math generalist models through hybrid instruction tuning. In *ICLR*, 2024. URL https://openreview.net/forum?id=yLClGs770I.

E. Zelikman, Y. Wu, J. Mu, and N. D. Goodman. STaR: self-taught reasoner bootstrapping reasoning with reasoning. In *NeurIPS*, pp. 15476–15488, 2022.

K. Zha, Z. Gao, M. Shen, Z-W. Hong, D. S. Boning, and D. Katabi. RL Tango: Reinforcing generator and verifier together for language reasoning. *ArXiv Preprint: 2505.15034*, 2025.

D. Zhang, S. Zhoubian, Z. Hu, Y. Yue, Y. Dong, and J. Tang. ReST-MCTS*: LLM self-training via process reward guided tree search. In *NeurIPS*, pp. 64735–64772, 2024a.

H. Zhang, J. He, J. Xu, J. Wang, and Z. Zheng. Enhancing language model with both human and artificial intelligence feedback data. In *WSC*, pp. 299–310, 2024b.

K. Zhang, Y. Hong, J. Bao, H. Jiang, Y. Song, D. Hong, and H. Xiong. GVPO: Group variance policy optimization for large language model post-training. *ArXiv Preprint: 2504.19599*, 2025a.

L. Zhang, A. Hosseini, H. Bansal, M. Kazemi, A. Kumar, and R. Agarwal. Generative verifiers: Reward modeling as next-token prediction. In *The NeurIPS Workshop on System-2 Reasoning at Scale*, 2024c. URL https://openreview.net/forum?id=aLgXy8A7k7.

Z. Zhang, C. Zheng, Y. Wu, B. Zhang, R. Lin, B. Yu, D. Liu, J. Zhou, and J. Lin. The lessons of developing process reward models in mathematical reasoning. *ArXiv Preprint: 2501.07301*, 2025b.

H. Zhao, G. I. Winata, A. Das, S-X. Zhang, D. Yao, W. Tang, and S. Sahu. RainbowPO: A unified framework for combining improvements in preference optimization. In *ICLR*, 2025. URL https://openreview.net/forum?id=trKee5pIFv.

Y. Zhao, R. Joshi, T. Liu, M. Khalman, M. Saleh, and P. J. Liu. SLiC-HF: Sequence likelihood calibration with human feedback. *ArXiv Preprint: 2305.10425*, 2023.

D. Zhou, N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, C. Cui, O. Bousquet, Q. V. Le, and E. H. Chi. Least-to-most prompting enables complex reasoning in large language models. In *ICLR*, 2023. URL https://openreview.net/forum?id=WZH7099tgfM.

D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving. Fine-tuning language models from human preferences. *ArXiv Preprint: 1909.08593*, 2019.

A Related Works

We comment on all related topics, including reasoning through chain-of-thought and its variants, test-time compute, direct preference alignment methods, reward models and reinforcement learning from AI feedback. For an overview of more reasoning models and methods, we refer to two recent surveys (Huang & Chang, 2023; Chen et al., 2025c).

Chain-of-Thought and its variants. Chain-of-thought (CoT) refers to as a broad class of methods that generate an intermediate reasoning process before arriving at a final answer. These approaches either prompt LLMs (Wei et al., 2022; Khot et al., 2023; Zhou et al., 2023) or train LLMs to generate reasoning chains through supervised fine-tuning (SFT) (Yue et al., 2024; Yu et al., 2024b; Li et al., 2025) and/or RL (Wang et al., 2024; Shao et al., 2024; Havrilla et al., 2024; Yu et al., 2025a). While CoT has proven effective for certain tasks, its auto-regressive generation nature makes it challenging to mimic human reasoning on more complex problems (LeCun, 2022; Hao et al., 2023), which require planning and search. Recent efforts were devoted to equipping LLMs with tree search methods (Xie et al., 2023; Yao et al., 2023; Hao et al., 2024a) or training LLMs on search trajectories (Lehnert et al., 2024; Gandhi et al., 2024; Su et al., 2025). Several other works have investigated why CoT is effective. For example, (Madaan et al., 2023) used a counterfactual prompting approach to examine the relative contributions of prompt elements, including symbols (digits, entities) and patterns (equations). (Feng et al., 2023; Merrill & Sabharwal, 2024; Li et al., 2024a) analyzed CoT from the perspective of model expressivity, and (Feng et al., 2023) showed that employing CoT increases the effective depth of a transformer since the generated outputs are looped back to the input. This insight motivated the chain-of-continuous-thought paradigm (Hao et al., 2024b), and a related approach has been proposed in (Cheng & Van Durme, 2024).

Reasoning through test-time compute. OpenAI-o1 (Jaech et al., 2024) is among the first large-scale applications of RL to reasoning, and achieved state-of-the-art performance upon release. Following this trend, DeepSeek-R1 (Guo et al., 2025a) is the first open-weight model to match or exceed OpenAI-o1. Their real-world success stories have involved several simple yet novel techniques that enhance LLM reasoning through more test-time compute, including chain-of-thought (Wei et al., 2022), self-consistency (Wang et al., 2023), best-of-N sampling (Snell et al., 2025), process reward models (Lightman et al., 2024), exploration-exploitation mechanism (Chen et al., 2026b), Monte Carlo tree search (Silver et al., 2016; Hao et al., 2023), tree-of-thought (Yao et al., 2023), and recent works on preventing overthinking (Chen et al., 2024b; Team et al., 2025; Luo et al., 2025a; Arora & Zanette, 2025) and compressing chain-of-thought (Hao et al., 2024b; Cheng & Van Durme, 2024). More specifically, *chain-of-thought* is a reasoning approach where intermediate steps are explicitly written to make complex problem-solving processes more transparent and logical. *Self-consistency* suggests generating multiple final answers and returning the mode of an empirical distribution, enhancing test-time performance when test-time verifiers are unavailable. Unfortunately, it is computationally expensive and effective only when answers can be clustered. *Best-of-N sampling* resolves this issue by sampling answers from the model and selecting the best at test time according to the scoring function; however, it is sensitive to the accuracy of test-time scoring functions (Gao et al., 2023). *Process reward models* offer fine-grained supervision of chain-of-thought reasoning, but they might be vulnerable to reward hacking and introduce computation overhead. *Monte Carlo tree search* is a generic technique that allocates computational resources toward the most promising regions of the search space, and *tree-of-thought* and its extension (Besta et al., 2024; Gandhi et al., 2024) simplified this idea by exploring multiple reasoning paths in a specific structure, allowing language models to select the most promising line of thought for complex problem-solving. Both *length regularization* and *compressed chain-of-thought* are developed to reduce inference costs for reasoning, which is crucial for the economic feasibility, user experience and environmental sustainability of LLMs. In addition, several works have focused on specific reasoning tasks (Lampinen et al., 2024; Yang et al., 2025; Srivastava et al., 2024; Huang et al., 2025; 2024; Guo et al., 2025b; Gou et al., 2024; Wang et al., 2025; Guo et al., 2025c), demonstrating promising performance. The recent findings of Xiong et al. (2025a) have shown that the REINFORCE-type methods (including GRPO (Shao et al., 2024)) can not effectively learn from all-negative-sample groups. Our work alleviates this issue by leveraging AI feedback to differentiate negative samples. We also provide a theoretical analysis through a stylized model, explaining why such diversification improves GRPO's learning dynamics.

Direct preference alignment methods. These methods (e.g., DPO (Rafailov et al., 2023)) are simple and stable offline alternatives to online RLHF. Various DPO variants with other objectives have been proposed, including ranking ones beyond pairwise preference data (Dong et al., 2023; Yuan et al., 2023a; Song et al., 2024; Chen et al., 2024a; Liu et al., 2025a) and simple ones that do not rely on a reference model (Hong et al., 2024; Meng et al., 2024). Since DPO does not train a reward model, the limited size of human labels becomes a bottleneck. To alleviate this limitation, subsequent works proposed to augment preference data using a trained SFT policy (Zhao et al., 2023) or a refined SFT policy with rejection sampling (Liu et al., 2024a). The DPO loss was recently rederived and extended to a token-level MDP view (Rafailov et al., 2024), where the state is the token prefix and the transition is deterministic – which has covered the fine-tuning of LLMs – and more general RL problems (Azar et al., 2024). There are other DPO variants (Ethayarajh et al., 2024; Park et al., 2024; Xu et al., 2024; Tang et al., 2024; Meng et al., 2024; Chen et al., 2025a; Zhao et al., 2025; Chen et al., 2026a). For example, Ethayarajh et al. (2024) designed a DPO-style loss variant using a prospect theory, Tang et al. (2024) optimized a general preference loss instead of the log-likelihood loss, and Meng et al. (2024) aligned the reward function in the preference optimization objective with the generation metric. Dong et al. (2024) and Xiong et al. (2024) proposed to generate human feedback in an online fashion to mitigate the distribution-shift and over-parameterization phenomenon. This improves DPO for complex reasoning tasks (Pang et al., 2024). Several other works focus on *unintentional alignment* of DPO and developing new methods (Pal et al., 2024; Tajwar et al., 2024; Liu et al., 2024b; Xiao et al., 2024; Yuan et al., 2025; Razin et al., 2025; Chen et al., 2025b). Among these works, Razin et al. (2025) proposed to measure the similarity between preferred and dispreferred responses using the centered hidden embedding similarity (CHES) score and showed that filtering out preference pairs with small CHES score improves DPO, while (Chen et al., 2025b) proposed to use comparison oracles, and showed that combining it with DPO effectively alleviated the issue of unintentional alignment.

Reward models. For the prompt $\mathbf{x}$ with a ground-truth response $\mathbf{y}^\star_{\mathbf{x}}$, we evaluate by implementing a regular expression match on the final answer (Hendrycks et al., 2021): $r(\mathbf{x}, \mathbf{y}) = 1$ if $\mathbf{y}$ matches $\mathbf{y}^\star_{\mathbf{x}}$ on the *final answer* and $r(\mathbf{x}, \mathbf{y}) = 0$ otherwise. An *outcome reward* model (ORM) (Cobbe et al., 2021; Uesato et al., 2022) is trained for estimating $r(\mathbf{x}, \mathbf{y})$. In particular, we first choose $\mathbf{x} \in \mathcal{D}$ and collect training samples $(\mathbf{x}, \mathbf{y} \sim \pi_\theta(\cdot|\mathbf{x}), r(\mathbf{x}, \mathbf{y}))$. Then, we take $(\mathbf{x}, \mathbf{y})$ as input and train an ORM to predict $r(\mathbf{x}, \mathbf{y})$. This can be done using binary classification (Cobbe et al., 2021; Yu et al., 2024a), direct preference optimization (Hosseini et al., 2024) or next-token prediction (Zhang et al., 2024c). Previous works also train LLMs on self-generated data using the ground-truth outcome reward model with either supervised fine-tuning (Singh et al., 2024; Yuan et al., 2023b; Zelikman et al., 2022) or online RL (Bi et al., 2024; Guo et al., 2025a). A *process reward* model (PRM) is trained to score a_h at $\mathbf{s}_h = (\mathbf{x}, a_1, \ldots, a_{h-1})$ either using human annotations (Lightman et al., 2024) or the value functions based on LLM-generated data (Wang et al., 2024; Luo et al., 2024; Setlur et al., 2025); indeed, PRMs estimate either the likelihood of future success or the change in the likelihood of future success before and after taking a_h. In addition, PRMs were also developed to improve search methods (Snell et al., 2025; Wu et al., 2025), and to identify the "first pit" in an incorrect reasoning trajectory to construct preference pairs for direct preference alignment (Hwang et al., 2024; Setlur et al., 2024). Related work such as VinePPO (Kazemnejad et al., 2024) refines process-level credit assignment in PPO without explicitly training a PRM and avoids a learned value network by using Monte Carlo rollouts from intermediate prefixes; this is complementary to our GRPO-oriented setting.

Reinforcement learning from AI feedback. Reinforcement learning from human feedback (RLHF) uses human-preference-aligned reward models to evaluate response quality (Christiano et al., 2017; Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022). A key barrier to scale RLHF is the need for high-quality human labels. Previous studies (Gilardi et al., 2023; Ding et al., 2023; Zhang et al., 2024b) have shown that modern LLMs exhibit strong alignment with human judgments, suggesting that AI-generated labels can serve as a viable alternative. In this context, (Bai et al., 2022) was the first to explore RLAIF, jointly optimizing helpfulness and harmlessness using both human and AI-generated labels, and (Roit et al., 2023; Kwon et al., 2023; Lee et al., 2024) showed that LLMs can produce informative reward signals for RL post-training. Our approach can leverage AI feedback to introduce response diversity within all-negative-sample groups by assigning intermediate binary rewards to reasoning steps. Indeed, one identifies the proportion of correct steps in the reasoning trajectory and use it to compute a reward $r_i \in [0, 1)$.

B Missing Proofs

In this section, we present the derivations underlying the update rules in Section 3.2. In particular, we derive $g_{\texttt{GRPO}}(\theta)$ and $g_{\texttt{SGPO}}(\theta)$, and obtain the corresponding update forms in Eqs. (3) and (4).

To derive $g_{\texttt{GRPO}}(\theta)$ and $g_{\texttt{SGPO}}(\theta)$, we start by restating the score functions:

$$s(a_1 = 1 \mid x) = \begin{bmatrix} p \\ -p \\ 0 \\ 0 \end{bmatrix}, \quad s(a_1 = 2 \mid x) = \begin{bmatrix} p-1 \\ 1-p \\ 0 \\ 0 \end{bmatrix},$$

$$s(a_2 = 1 \mid x, a_1 = 2) = \begin{bmatrix} 0 \\ 0 \\ q \\ -q \end{bmatrix}, \quad s(a_2 = 2 \mid x, a_1 = 2) = \begin{bmatrix} 0 \\ 0 \\ q-1 \\ 1-q \end{bmatrix}.$$

Let $S(y) := \sum_{h=1}^{H} s(a_h \mid x, a_{1:h-1})$ denote the score-sum for trajectory $y = (a_1, a_2)$. Under the restricted trajectory space $\{(1,1),(2,1),(2,2)\}$, we have

$$S(1,1) = \begin{bmatrix} p \\ -p \\ 0 \\ 0 \end{bmatrix}, \quad S(2,1) = \begin{bmatrix} p-1 \\ 1-p \\ q \\ -q \end{bmatrix}, \quad S(2,2) = \begin{bmatrix} p-1 \\ 1-p \\ q-1 \\ 1-q \end{bmatrix},$$

and the trajectory probabilities are

$$\mathbb{P}(1,1) = 1-p, \quad \mathbb{P}(2,1) = p(1-q), \quad \mathbb{P}(2,2) = pq.$$

Recall $G = 2$ and $H = 2$. For a single prompt, our estimator averages over both samples and steps:

$$g(\theta) = \tfrac{1}{GH}\left(\sum_{k=1}^{G}\sum_{h=1}^{H} s\left(a_h^{(k)} \mid x, a_{1:h-1}^{(k)}\right) A_k\right) = \tfrac{1}{GH}\left(\sum_{k=1}^{G} S(y^{(k)}) A_k\right),$$

where A_k is the within-group standardized advantage computed from the two rewards. For $G = 2$, whenever the two rewards are distinct, standardization yields $A_{\text{high}} = +1$ and $A_{\text{low}} = -1$; if the rewards are equal we set $A_1 = A_2 = 0$. Thus, for any ordered pair $(y^{(1)}, y^{(2)})$ with distinct rewards,

$$g(\theta) = \tfrac{1}{GH}\left(S(y^{(1)}) - S(y^{(2)}))\cdot \operatorname{sign}(r(y^{(1)}) - r(y^{(2)})\right).$$

GRPO. In GRPO, $r(2,2) = 1$ and $r(1,1) = r(2,1) = 0$. Thus, the only nonzero contributions come from mixed pairs $\{(2,2),(1,1)\}$ and $\{(2,2),(2,1)\}$. Using independence of the two samples and summing both orderings,

$$\bar{g}_{\text{GRPO}}(\theta) = \mathbb{E}[g(\theta)] = \tfrac{2}{GH}\left(\mathbb{P}(2,2)\mathbb{P}(1,1)(S(2,2) - S(1,1)) + \mathbb{P}(2,2)\mathbb{P}(2,1)(S(2,2) - S(2,1))\right).$$

With $G = H = 2$, $\frac{2}{GH} = \frac{1}{2}$. Substituting the probabilities and $S(\cdot)$ yields

$$\bar{g}_{\text{GRPO}}(\theta) = \tfrac{1}{2}\begin{bmatrix} p(p-1)q \\ p(1-p)q \\ pq(q-1) \\ pq(1-q) \end{bmatrix}.$$

SGPO. In SGPO, $r_{\text{SGPO}}(2,2) = 1$, $r_{\text{SGPO}}(2,1) = \frac{1}{2}$, and $r_{\text{SGPO}}(1,1) = 0$. Therefore, all three unordered mixed pairs contribute: $\{(2,2),(2,1)\}$, $\{(2,2),(1,1)\}$, and $\{(2,1),(1,1)\}$. Summing both orderings, we have

$$\bar{g}_{\text{SGPO}}(\theta) = \tfrac{2}{GH}\left(\mathbb{P}(2,2)\mathbb{P}(2,1)(S(2,2) - S(2,1)) + \mathbb{P}(2,2)\mathbb{P}(1,1)(S(2,2) - S(1,1)) + \mathbb{P}(2,1)\mathbb{P}(1,1)(S(2,1) - S(1,1))\right)).$$

Again, we have $\frac{2}{GH} = \frac{1}{2}$ for $G = H = 2$, and substituting gives

$$\bar{g}_{\text{SGPO}}(\theta) = \frac{1}{2}\begin{bmatrix} p(p-1) \\ p(1-p) \\ p^2q(q-1) \\ p^2q(1-q) \end{bmatrix}.$$

Now, we derive the update rules in Eq. (3) and the functions in Eq. (4) from the population gradients. Let $\theta^{(k+1)} = \theta^{(k)} + \eta\,\bar{g}(\theta^{(k)})$, where $\bar{g}$ is either $\bar{g}_{\text{SGPO}}$ or $\bar{g}_{\text{GRPO}}$. For notational brevity, define the step-1 logits $(\theta_1, \theta_2) = (\theta^1_{x,1}, \theta^1_{x,2})$ and step-2 logits $(\theta_3, \theta_4) = (\theta^2_{x,2,1}, \theta^2_{x,2,2})$. In addition, we have

$$p = \frac{e^{\theta_2}}{e^{\theta_1}+e^{\theta_2}}, \quad q = \frac{e^{\theta_4}}{e^{\theta_3}+e^{\theta_4}}.$$

Update of p. Let g_1, g_2 denote the first two coordinates of $\bar{g}(\theta)$. Then, we have

$$p^{(k+1)} = \frac{e^{\theta_2^{(k)}+\eta g_2}}{e^{\theta_1^{(k)}+\eta g_1}+e^{\theta_2^{(k)}+\eta g_2}} = \frac{e^{\theta_2^{(k)}}e^{\eta g_2}}{e^{\theta_1^{(k)}}e^{\eta g_1}+e^{\theta_2^{(k)}}e^{\eta g_2}}.$$

Using $p^{(k)} = \frac{e^{\theta_2^{(k)}}}{e^{\theta_1^{(k)}}+e^{\theta_2^{(k)}}}$ and $1 - p^{(k)} = \frac{e^{\theta_1^{(k)}}}{e^{\theta_1^{(k)}}+e^{\theta_2^{(k)}}}$, we can factor out $e^{\theta_1^{(k)}} + e^{\theta_2^{(k)}}$ to obtain

$$p^{(k+1)} = \frac{p^{(k)}e^{\eta(g_2-g_1)}}{(1-p^{(k)})+p^{(k)}e^{\eta(g_2-g_1)}}.$$

For SGPO, from $\bar{g}_{\text{SGPO}}(\theta)$ we have $g_1 = \frac{1}{2}p(p-1)$ and $g_2 = \frac{1}{2}p(1-p)$, hence $g_2 - g_1 = p(1-p)$. For GRPO, from $\bar{g}_{\text{GRPO}}(\theta)$ we have $g_1 = \frac{1}{2}p(p-1)q$ and $g_2 = \frac{1}{2}p(1-p)q$, hence $g_2 - g_1 = p(1-p)q$. Setting $\eta = 1$ gives the claimed update

$$p^{(k+1)}_{\text{SGPO}} = \frac{p^{(k)}_{\text{SGPO}}e^{p^{(k)}_{\text{SGPO}}(1-p^{(k)}_{\text{SGPO}})}}{1-p^{(k)}_{\text{SGPO}}+p^{(k)}_{\text{SGPO}}e^{p^{(k)}_{\text{SGPO}}(1-p^{(k)}_{\text{SGPO}})}} = \exp(f_{11}(p^{(k)}_{\text{SGPO}})),$$

$$p^{(k+1)}_{\text{GRPO}} = \frac{p^{(k)}_{\text{GRPO}}e^{p^{(k)}_{\text{GRPO}}(1-p^{(k)}_{\text{GRPO}})q^{(k)}_{\text{GRPO}}}}{1-p^{(k)}_{\text{GRPO}}+p^{(k)}_{\text{GRPO}}e^{p^{(k)}_{\text{GRPO}}(1-p^{(k)}_{\text{GRPO}})q^{(k)}_{\text{GRPO}}}} = \exp(f_{21}(p^{(k)}_{\text{GRPO}}, q^{(k)}_{\text{GRPO}})).$$

Taking log on both sides yields

$$f_{11}(p) = \log p + p(1-p) - \log\left(1 - p + pe^{p(1-p)}\right),$$
$$f_{21}(p,q) = \log p + p(1-p)q - \log\left(1 - p + pe^{p(1-p)q}\right).$$

Update of q. An identical argument applies to q using the last two coordinates g_3, g_4 of $\bar{g}(\theta)$:

$$q^{(k+1)} = \frac{q^{(k)}e^{\eta(g_4-g_3)}}{(1-q^{(k)})+q^{(k)}e^{\eta(g_4-g_3)}}.$$

For SGPO, $g_3 = \frac{1}{2}p^2q(q-1)$ and $g_4 = \frac{1}{2}p^2q(1-q)$, so $g_4 - g_3 = p^2q(1-q)$. For GRPO, $g_3 = \frac{1}{2}pq(q-1)$ and $g_4 = \frac{1}{2}pq(1-q)$, so $g_4 - g_3 = pq(1-q)$. With $\eta = 1$ this gives

$$q^{(k+1)}_{\text{SGPO}} = \exp(f_{12}(p^{(k)}_{\text{SGPO}}, q^{(k)}_{\text{SGPO}})), \quad q^{(k+1)}_{\text{GRPO}} = \exp(f_{22}(p^{(k)}_{\text{GRPO}}, q^{(k)}_{\text{GRPO}})),$$

where by taking log on both sides, we have

$$f_{12}(p,q) = \log q + p^2q(1-q) - \log\left(1 - q + qe^{p^2q(1-q)}\right),$$
$$f_{22}(p,q) = \log q + pq(1-q) - \log\left(1 - q + qe^{pq(1-q)}\right).$$

We then provide several technical lemmas that are important to the proof of Theorem 3.3. Indeed, the first lemma summarizes the properties of particular functions related to the aforementioned functions f_{11}, f_{21}, f_{12} and f_{22} from Eq. (4).

Lemma B.1. *The following statements hold true,*

(i) The function f_{11} is strictly increasing on $(0,1)$.
(ii) The function $h_p(x) := x - \log(1-p+pe^x)$ is strictly increasing for any fixed $p \in (0,1)$.
(iii) The function f_{21} is strictly increasing in either p for any fixed q or q for any fixed p on $(0,1)$.
(iv) The function $\varphi(x) := \log(1+e^{-e^x})$ is strictly concave on $(-\infty, 0)$.

Proof. First of all, we have

$$f_{11}'(p) = \frac{1+(1-2p)p(1-p)}{p(1-p+pe^{p(1-p)})} > \frac{3}{4p(1-p+pe^{p(1-p)})} > 0.$$

Thus, the function f_{11} is strictly increasing on $(0,1)$.

Furthermore, we have

$$h_p'(x) = 1 - \frac{pe^x}{1-p+pe^x} = \frac{1-p}{1-p+pe^x} \overset{0<p<1}{>} 0.$$

Thus, the function $h_p(x)$ is strictly increasing.

Moreover, we have

$$\begin{aligned} \frac{\partial f_{21}(p,q)}{\partial p} &= \frac{1+q(1-2p)p(1-p)}{p(1-p+pe^{qp(1-p)})} > \frac{3}{4p(1-p+pe^{p(1-p)})} > 0, \\ \frac{\partial f_{21}(p,q)}{\partial q} &= \frac{p(1-p)^2}{1-p+pe^{qp(1-p)}} > 0. \end{aligned}$$

Thus, the function f_{21} is strictly increasing in either p for any fixed q or q for any fixed p on $(0,1)$.

Finally, we have

$$\varphi''(x) = \frac{(e^{x+e^x} - e^{e^x} - 1)e^x}{e^{2e^x} + 2e^{e^x} + 1}.$$

Since $u = e^x \in (0,1)$ for $x < 0$, we have

$$(ue^u - e^u - 1)u = (e^u(u-1) - 1)u < -u < 0.$$

Thus, $\varphi''(x) < 0$ for all $x < 0$ which shows that φ is strictly concave on $(-\infty, 0)$. □

The second lemma presents an inequality which plays a key role in the proof of Theorem 3.3.

Lemma B.2. *We define the auxiliary functions as follows,*

$$A(x) = 1 + \left(\tfrac{1}{x} - 1\right) e^{-x(1-x)},\ B(x,y) = 1 + \left(\tfrac{1}{y} - 1\right) e^{-x^2 y(1-y)},$$
$$C(z) = 1 + \left(\tfrac{1}{z} - 1\right) e^{-z^2(1-z)}.$$

Then, we have $C(\sqrt{xy})^2 > A(x)B(x,y)$ for all x and y satisfying $1/2 < y < x < 1$.

Proof. We consider the lower and upper bound of e^{-u} when $u > 0$:

$$1 - u + \frac{u^2}{2} - \frac{u^3}{6} < e^{-u} < 1 - u + \frac{u^2}{2}.$$

Since $1/x - 1$, $1/y - 1$ and $1/\sqrt{xy} - 1$ are all positive, we have

$$A(x) \le 1 + \frac{1-x}{x}\left(1 - x(1-x) + \frac{x^2(1-x)^2}{2}\right) = \frac{1}{x} - (1-x)^2 + \frac{x(1-x)^3}{2}.$$

$$B(x,y) \le 1 + \frac{1-y}{y}\left(1 - x^2 y(1-y) + \frac{x^4 y^2 (1-y)^2}{2}\right) = \frac{1}{y} - x^2(1-y)^2 + \frac{x^4 y(1-y)^3}{2}.$$

$$\begin{aligned} C(z) &\ge 1 + \frac{1-z}{z}\left(1 - z^2(1-z) + \frac{z^4(1-z)^2}{2} - \frac{z^6(1-z)^3}{6}\right) \\ &= \frac{1}{z} - z(1-z)^2 + \frac{z^3(1-z)^3}{2} - \frac{z^5(1-z)^4}{6}. \end{aligned}$$

Set $z^2 = xy$, the original statement is equivalent to $(zC(z))^2 > (xA(x))(yB(x,y))$. Using the above upper and lower bound, it suffices to show $C_1(\sqrt{xy})^2 > A_1(x)B_1(x,y)$ where

$$A_1(x) = 1 - x(1-x)^2 + x^2(1-x)^3/2,\ B_1(x,y) = 1 - x^2y(1-y)^2 + x^4y^2(1-y)^3/2,$$
$$C_1(z) = 1 - z^2(1-z)^2 + z^4(1-z)^3/2 - z^6(1-z)^4/6.$$

By Lemma B.3, this is indeed true. This completes the proof. □

Lemma B.3. *Define functions*

$$A_1(x) = 1 - x(1-x)^2 + x^2(1-x)^3/2,\ B_1(x,y) = 1 - x^2y(1-y)^2 + x^4y^2(1-y)^3/2,$$
$$C_1(z) = 1 - z^2(1-z)^2 + z^4(1-z)^3/2 - z^6(1-z)^4/6.$$

Then, $C_1(\sqrt{xy})^2 > A_1(x)B_1(x,y)$ *for all* $1/2 < y < x < 1$.

Proof. Let $x = u^2$ and $y = v^2$, then $1 > u > v > 1/\sqrt{2}$ and $z = uv$. We next show the desired inequality holds on a larger region, i.e., $1 > u > v > 2/3$. On this larger region, we have the reparameterization as follows:

$$u = \tfrac{2s+3}{3s+3}, \quad v = \tfrac{2r+2s+3}{3r+3s+3}, \quad s, r \in (0, +\infty),$$

or equivalently,

$$s = \tfrac{3(1-u)}{3u-2}, \quad r = \tfrac{3(u-v)}{(3u-2)(3v-2)}, \quad 1 > u > v > \tfrac{2}{3}.$$

It is easy to see this defines a one-to-one correspondence from (u,v)-space to (s,r)-space. Thus, we aim to prove the following function f is positive:

$$F(s,r) := C_1\left(\tfrac{2s+3}{3s+3} \cdot \tfrac{2r+2s+3}{3r+3s+3}\right)^2 - A_1\left(\left(\tfrac{2s+3}{3s+3}\right)^2\right) B_1\left(\left(\tfrac{2s+3}{3s+3}\right)^2, \left(\tfrac{2r+2s+3}{3r+3s+3}\right)^2\right).$$

By leveraging Sympy's symbolic engine, the function expands and simplifies to:

$$F(s,r) = \tfrac{f(s,r)}{c(s+1)^{20}(r+s+1)^{20}}, \text{ where } f(s,r) := \sum_{k=0}^{20} c_{20-k}(s) r^{20-k},$$

where $c > 0$ is a universal constant, and single-variable polynomials $c_{20}(s), \ldots, c_2(s), c_0(s) > 0$ and $\Delta_2 := c_1(s)^2 - 4c_2(s)c_0(s) < 0$, for all $s > 0$ (see Table 6 for details). Notice that from the table, we can see the only nontrivial parts are $c_3(s) > 0$ and $c_2(s) > 0$ because only these two contain negative coefficients. The positivity of $c_3(s)$ is simple because there is only one term (s^9) with negative coefficient and for all $s > 0$,

$$194714567104543630051526 64s^{10} + 968458837773164307192723 6s^8 > 144138231093502245414997 26s^9.$$

To see this, simple estimation and AM-GM inequality yield

$$\text{LHS} > 1.9 \times 10^{25}s^{10} + 9.6 \times 10^{24}s^8 > 2\sqrt{182.4} \times 10^{24}s^9 > 2.7 \times 10^{25}s^9 > \text{RHS}.$$

The positivity of $c_2(s)$ is more complicated because it has 4 negative terms s^{10}, s^9, s^8, s^7. However, we can use similar idea, i.e., choosing a pair of positive terms to bound a negative term:

$$957910627865555087240887423 20s^{15} + 571809550541807937530952s^5 > 702735952364323683297077 16s^{10};$$
$$437858623301624990522095297 68s^{14} + 184789343789534461150530s^4 > 998540727043228715373926 04s^9;$$
$$163267368535271229917151558 24s^{13} + 354883755696224721692 40s^3 > 357260313779697920881889 25s^8;$$
$$460821905008432679074893315 3s^{12} + 436295085881317044922 8s^2 > 609903789530767014228760 8s^7.$$

To see this, simple estimation and AM-GM inequality yield

$$\text{LHS} > 9.5 \times 10^{28}s^{15} + 5.7 \times 10^{23}s^5 > 2\sqrt{541.5} \times 10^{25}s^{10} > 4.6 \times 10^{26}s^{10} > \text{RHS};$$
$$\text{LHS} > 4.3 \times 10^{28}s^{14} + 1.8 \times 10^{23}s^4 > 2\sqrt{77.4} \times 10^{25}s^9 > 1.7 \times 10^{26}s^{10} > \text{RHS};$$
$$\text{LHS} > 1.6 \times 10^{28}s^{13} + 3.5 \times 10^{22}s^3 > 2\sqrt{5.6} \times 10^{25}s^9 > 4.7 \times 10^{25}s^8 > \text{RHS};$$
$$\text{LHS} > 4.6 \times 10^{27}s^{12} + 4.3 \times 10^{21}s^2 > 2\sqrt{19.7} \times 10^{24}s^7 > 8.8 \times 10^{24}s^7 > \text{RHS}.$$

In conclusion, we have all coefficient $c_i(s)$ positive except $c_1(s)$, but it doesn't affect the positivity of f because $\Delta_2 < 0$. This completes the proof. $\square$

Table 6: Coefficient Lists of $F(s,r)$

Notation	Value
c	437675956526049436836
c_{20}	$2(2s+3)^2(1714774320744848750s^{18}+26610409260691576200s^{17}+191778746468802317181s^{16}+850194149855082319224s^{15}+2587082434290045806049s^{14}+5704415906039160731874s^{13}+9366197581963232054460s^{12}+11563054951307567026248s^{11}+10670965452123886149660s^{10}+7187176769582075261292s^{9}+3372972168996579430017s^{8}+1072082836158220703952s^{7}+370302094042890771285s^{6}+329901677376902425818s^{5}+282799986805616267862s^{4}+151168270170893365008s^{3}+49139849518345513368s^{2}+9090603727935062976s+742484948385838248)$
c_{19}	$8(2s+3)^2(8573871603724243750s^{19}+14118375184879884045 0s^{18}+10850654575356110840 97s^{17}+5159905424662678527663s^{16}+1696201782101404135528 5s^{15}+407615158929069302613 93s^{14}+73777358861762677983126s^{13}+10196816347629194227764 3s^{12}+1077195501832792029453 36s^{11}+85951871005332247942347s^{10}+5047714242076387274703 9s^{9}+212284617104842272708 12s^{8}+71755762533602862021 93s^{7}+3821642119447908810138s^{6}+3343140602539615070982s^{5}+2308058741380310946144s^{4}+1046528691565621471344s^{3}+300769285860744146028s^{2}+50403014643440592936s+3785769852305984190)$
c_{18}	$2(2s+3)^2(32580712094152126250 0s^{20}+5673987380977312396200s^{19}+46320143614200183575358s^{18}+23516462486845543431474 0s^{17}+83033078234649987863140 2s^{16}+21591058927666966255084 32s^{15}+4268066364777710628878112s^{14}+652117772627658619162826 4s^{13}+77429334197200253596601 31s^{12}+71109113618735821090519 92s^{11}+4976474876383070663195517s^{10}+2600230719135591148269222s^{9}+1035307928116150386109695s^{8}+420495220158165783672300s^{7}+287436345862168026209421s^{6}+225749912117527047120354s^{5}+132205553924765757023286s^{4}+52546262098747895532864s^{3}+13596497258861930544108s^{2}+2087305777245729888936s+145293329180967197454)$
c_{17}	$12(2s+3)^2(32580712094152126250 0s^{21}+5982992191700268855300s^{20}+5170093051261332740 3406s^{19}+2790797807772594779445 90s^{18}+1053187317945045364767966s^{17}+2945458902809849586876810s^{16}+631090033186536301274384 4s^{15}+1055418229017932721427348 2s^{14}+1389416400430029240402978 9s^{13}+1439751175550269521639977 7s^{12}+1164818753297125385958319 5s^{11}+725367284324989487520362 1s^{10}+346393407606271198414818 3s^{9}+140131187112463692251067 9s^{8}+709635073453803384330663s^{7}+526601300781722718969621s^{6}+366411462920903557014120s^{5}+187790938366917450633606s^{4}+66828265788979238418684s^{3}+15774993964318985462592s^{2}+2238177282159012945966s+145311275743961970078)$
c_{16}	$3(2s+3)^2(5538721056005861462500s^{22}+10696394904119483034480 0s^{21}+9753724173870750955571 10s^{20}+5577701869567635624516312s^{19}+2240107013885478456267160 2s^{18}+6703472556180932146225722 0s^{17}+1546895926600617757806830 34s^{16}+2808895398013327676940916 08s^{15}+4056643289930059360981582 20s^{14}+4674322345974503233776546 24s^{13}+4281629117018364704534888 16s^{12}+3088511931774198596481206 64s^{11}+1739237115076245954125557 92s^{10}+7861104390229527730501436 4s^{9}+3450272393210865996806819 1s^{8}+2078486967808784701135051 2s^{7}+1521107049127527089926047 9s^{6}+9439795169478817720557390s^{5}+4323642777299210867466840s^{4}+1398336255225225668686176s^{3}+3041651033834191453666 80s^{2}+4017017098446888839688 0s+2445688799534592091926)$

c_{15}	$12(2s+3)^2(4430976844804689170000s^{23}+8977362465878807211960 0s^{22}+861449789967478967902968s^{21}+52020523405703639617992 72s^{20}+2215095867472214763688788 0s^{19}+7061076897743159713850241 6s^{18}+17454774771903874157924914 8s^{17}+34184679883022075974400680 2s^{16}+53701789749405047322088747 5s^{15}+68038631708864390907265235 7s^{14}+69490091744552770910260620 3s^{13}+56885429435145235255915535 7s^{12}+37016250086232520818694940 7s^{11}+19208993461527481014311363 7s^{10}+8556000037440487107804474 3s^{9}+4218897906932559569048489 4s^{8}+2798648298363544891614947 7s^{7}+1934947300052794842316009 8s^{6}+1082654952241765058234790 3s^{5}+4499119009419911850147903s^{4}+1337268081929646109116429s^{3}+2701887274473978701502 99s^{2}+3341219144872220287179 3s+191648133946778922704 7)$
c_{14}	$3(2s+3)^2(4430976844804689170000 0s^{24}+9397609008462027996336 00s^{23}+946588223144997927058161 6s^{22}+6018941564428726543024022 4s^{21}+2708318511363669611698591 20s^{20}+9160822434227131933406504 64s^{19}+2414605718335618880853970 032s^{18}+5071784805050410035823167 576s^{17}+8606003351786628019139811 024s^{16}+1188207272155926800257859 4336s^{15}+1337313573606658891526722 5192s^{14}+1223378975301732738139865 6592s^{13}+9038087243602138768195078 704s^{12}+5369286552690873446276117 328s^{11}+2623535852573549267091555 300s^{10}+1196071780982939651865144 096s^{9}+6600782446234967802635880 75s^{8}+4513931959976662086674588 52s^{7}+2903436770732689418640463 05s^{6}+1480420428326298039673210 50s^{5}+5646419333104340411674178 4s^{4}+1555947843019285082981882 4s^{3}+2939179015494775847121192s^{2}+3420469295854970612531 76s+1855791464680027845905 4)$
c_{13}	$6(2s+3)^2(4430976844804689170000 0s^{25}+9817855551045248780712 00s^{24}+1035713233442216953911268 8s^{23}+6916613178338427499706232 0s^{22}+3279066095144959426258895 52s^{21}+1172872936442019296825004 672s^{20}+3283063721629310545911589 320s^{19}+7360333239220785966714322 212s^{18}+1341123809195951980117385 5314s^{17}+2003087433125863944761640 5430s^{16}+2461263206658443506304569 3896s^{15}+2486180794324324784856470 2728s^{14}+2055461630993867656408819 3632s^{13}+1382870177464444560719848 9296s^{12}+7593307018019776947591316 692s^{11}+3573164437682539651140298 914s^{10}+1711205400898634741432588 709s^{9}+1026436201325482873227731 181s^{8}+6933466981581302133514425 87s^{7}+4137294683274523410927838 23s^{6}+1940776438308319265359606 74s^{5}+6856190522023177561905108 0s^{4}+1764097050880333254639794 4s^{3}+3132633769453198732801032s^{2}+3445604242270009355658 60s+1774509692548916842362 0)$
c_{12}	$3(2s+3)^2(1440067474561523980250 00s^{26}+3327383180429252608653 600s^{25}+3668680667792192157502 4412s^{24}+2567126637857826878686 07040s^{23}+1278878488443918869718 977532s^{22}+4822501965635852078399 708760s^{21}+1428493716760540762512 3613032s^{20}+3403967573402053121871 3639960s^{19}+6626919408734780484264 1890936s^{18}+1064202511819990594837 39591464s^{17}+1416737183478865486115 27896944s^{16}+1565122537347448498315 03592064s^{15}+1431194586986727902840 20359156s^{14}+1077729194579570065355 62903236s^{13}+6658356331681893190511 7287625s^{12}+3420802797778710148000 6575072s^{11}+1582143051029433989141 6781741s^{10}+8030913522092664743057 080482s^{9}+5050521785734143734257 145460s^{8}+3292374301511579644939 102872s^{7}+1827212365211212115171 329068s^{6}+7951466255776474175898 37740s^{5}+2621421821039038791088 12389s^{4}+6335442721429618093777 9584s^{3}+1062561041972189809432 0272s^{2}+1108770079900593722922 360s+5437162259941865052170 7)$

c_{11}	$2(2s+3)^2(28801349491230479605000 0s^{27}+6927926613537598727151600s^{26}+$ $7968499405072408869622956 0s^{25}+5830135753486700217327189 84s^{24}+$ $3044719667796555717818159544s^{23}+1207115288169085268602840692 0s^{22}+$ $3771975180228306456186022831 2s^{21}+9518657075468497895431568072 4s^{20}+$ $1971414432505091371211002980 18s^{19}+3386213982946173069363008884 86s^{18}+$ $4853203941721254940783622901 42s^{17}+5817879064516111538357630359 26s^{16}+$ $5828027860833244086757468021 92s^{15}+4859938328494178086778700964 80s^{14}+$ $3355944446899925164463735414 70s^{13}+1918186782349513132753181871 66s^{12}+$ $9321550814168794157984604359 1s^{11}+4299548855661907296542112784 5s^{10}+$ $2299333420544570449863855165 9s^{9}+1470200287638349161916712529 3s^{8}+$ $9151802478157350485470780638s^{7}+4746495171599998113140409294s^{6}+$ $1930238883493800552821467836s^{5}+5976617058265502770521785 82s^{4}+$ $1363694210951181524092878 75s^{3}+2169005140767851682417301 5s^{2}+$ $2154400447443907595487183s+1008731317586940460287 45)$
c_{10}	$(2s+3)^2(6336296888070705513100 00s^{28}+1584239110567672292139120 0s^{27}+$ $1897621049684466450141042 96s^{26}+1448899233419905865696230704s^{25}+$ $7915017749037631812296989272s^{24}+3291137409296388117112885 1808s^{23}+$ $1081841136804198369570625711 20s^{22}+2881814574059132039155601476 56s^{21}+$ $6325630775471257682897241758 52s^{20}+1156966248842167228077853758672s^{19}+$ $1775610230389152091840398326292s^{18}+2294661769917892997575576903488s^{17}+$ $2498192965867701162268688835924s^{16}+2285659112305489254434518711416s^{15}+$ $1748991971403770816761602941934s^{14}+1113797684163785846264967323376s^{13}+$ $5924913951597435743900967421 11s^{12}+2744766007155415656150112137 96s^{11}+$ $1270446195852713506501888454 52s^{10}+705385823261477238654077656 80s^{9}+$ $4488968110254001706707249846 5s^{8}+2660289888716075956783633452 0s^{7}+$ $1296776974656174947900170128 0s^{6}+4960789745148078555411299844s^{5}+$ $1450732098731590049595423084s^{4}+3139308944683290821452849 56s^{3}+$ $4752567062061261105861801 9s^{2}+4506795195444098216749962s+2019810664389200880741 21)$
c_{9}	$2(2s+3)^2(28801349491230479605000 00s^{29}+7474247118895785746840400s^{28}+$ $9308513797882609298968418 4s^{27}+7404022441838906136102386 16s^{26}+$ $4222492016436341447317422840s^{25}+1837374587737775714357659 7976s^{24}+$ $6337431126498352669577742938 4s^{23}+1776892961989847704086466054 92s^{22}+$ $4119921298476838644538559778 90s^{21}+7992705700875121584791677780 14s^{20}+$ $1307454563265970530661662310 488s^{19}+1811440906673507850196226957292s^{18}+$ $2129008833631551803404401966618s^{17}+2120318902997465121470552812398s^{16}+$ $1782752578142774040882126045846s^{15}+1258348636755929955587480574282s^{14}+$ $7422229506057785006481351208 08s^{13}+3687595326777177603238716204 48s^{12}+$ $1632923401776166653452978780 34s^{11}+7562813554420816525428037104 0s^{10}+$ $4285873595461389310248944256 9s^{9}+2681560986049355543901710696 7s^{8}+$ $1515406630759595340156585468 0s^{7}+6986502898234763627448498006s^{6}+$ $2529774938694747662152377363s^{5}+7023305018106747704878627 23s^{4}+$ $1447326550278639488278622 61s^{3}+2092454720837126248556029 5s^{2}+$ $1899529777608715114521399s+8166786538840395351817 5)$

c_8	$3(2s+3)^2(1440067474561523980250 00s^{30}+387370368578743962834 2400s^{29}+500869506060239258684790 36s^{28}+414347293343103311413357200s^{27}+246245571641789555346681440 4s^{26}+111903118390663442595367140 24s^{25}+404092959727731770501045665 56s^{24}+118947134811306669490272616200s^{23}+290460448001449937986609641984s^{22}+595650982876961700444247916136s^{21}+103439450333283704483287197366 0s^{20}+152912252239834145223774826432 8s^{19}+192917011214440853129524724035 2s^{18}+207727572983577989862797245256 4s^{17}+190440009258484914739880325785 7s^{16}+147929501924676524442161796040 8s^{15}+967232587581523134128767690737s^{14}+529827149456294513579866857762s^{13}+245850149371151220247602350838s^{12}+103586181981415962365079350136s^{11}+474419698310052107658566514 90s^{10}+270173703286137312429887261 16s^{9}+165436040746213423165348338 96s^{8}+895980810481550396812090116 0s^{7}+393414848050341758787717765 3s^{6}+135670850192660322074408069 4s^{5}+359356010050812330787473279s^{4}+708003980253654289019192 52s^{3}+980570563944186001099307 4s^{2}+854294427989006197205052s+353061860756115402430 56)$
c_7	$12(2s+3)^2(2215488422402344585000 0s^{31}+616966740327228674348400s^{30}+827090707369616268343048 8s^{29}+710548883749584474193314 00s^{28}+439319157110546803811896968s^{27}+208115303570456665813062480 0s^{26}+785169940845868080224263448 4s^{25}+242075167606254230722070734 14s^{24}+620925174288058758667946748 81s^{23}+134191349159021239688414857431s^{22}+246518378732311775029464265197s^{21}+387229048800653531738114739999s^{20}+521849077561894657636398449655s^{19}+604010499659026951460897607741s^{18}+599706193952136964321127694249s^{17}+508932600092700362308014694866s^{16}+366923845830106735087199088303s^{15}+222970678878271885535159028828s^{14}+113510751404816691589761355815s^{13}+489596215801597338452898846 33s^{12}+193438197331709555443195521 63s^{11}+857096764824341929866037721 1s^{10}+484228170521352710841651327 9s^{9}+290913837456231799036003139 4s^{8}+152292130359617399613857590 5s^{7}+642212965528640572310436879s^{6}+212347408792609836644815359s^{5}+539405048691381659833555 56s^{4}+102009814215488662722720 21s^{3}+135760942374822642715777 8s^{2}+113782587792457476788865s+452847621089613513418 2)$
c_6	$3(2s+3)^2(4430976844804689170000 0s^{32}+127595813491277942713440 0s^{31}+177121246487435203742460 00s^{30}+157800783303192342484807776s^{29}+101347600209682265393424753 6s^{28}+499634681147199443824097040 0s^{27}+196568910490037599977382196 64s^{26}+633430185859868230826197292 88s^{25}+170258537043632779229473723584s^{24}+386719255970188993917224574336s^{23}+749195973037122202127202135768s^{22}+124595278986516384215992550025 6s^{21}+178599096936071504600011736356 8s^{20}+221091189708515243073782231548 8s^{19}+236332565694983363186993047423 2s^{18}+217638653447586854325017845202 4s^{17}+171851926959773076904621993192 5s^{16}+115489899765681461703830942492 4s^{15}+653993799682370716585040976045s^{14}+309097812387589842994049088882s^{13}+122649986514805927975811013678s^{12}+441393215933864864375413042 32s^{11}+181946005120731311796524742 18s^{10}+100807294552471771494663601 08s^{9}+601215449500983286314386070 3s^{8}+308671390190448530207742391 2s^{7}+126453651300264489155357353 2s^{6}+404269004313535997709064188s^{5}+990853185541373958934672 65s^{4}+180668695929658455737317 68s^{3}+231803755921968445453389 3s^{2}+187335877784959577298978s+719206981536479606622 9)$

c_5	$6(2s+3)^2(8861953689609378340000s^{33}+2635965578342203011144 00s^{32}+3784460184258343211722032s^{31}+3492046692914393487980836 8s^{30}+2326421301668771204257529 76s^{29}+1191697800945367086458055072s^{28}+4880694751880441559218476632s^{27}+1640695082446954768056077876 4s^{26}+4611286528041885148932207413 4s^{25}+1098122749449603511155991786018s^{24}+2237258564335167139988498056 32s^{23}+3926601409170982278839410198 80s^{22}+5964503328992760850067540004 36s^{21}+7862415591394253061677474373 24s^{20}+9001624724457062708272057354 08s^{19}+8941184561988137623747448748 42s^{18}+7680197657003568215410480642 11s^{17}+5671889195150294047951246398 15s^{16}+3568928491736905887842798659 23s^{15}+1888877264400580909181235698 07s^{14}+8275851967757393779780165867 8s^{13}+2975584196829515518093837716 0s^{12}+9265655123985315388229427762s^{11}+3269804705124270960946179492s^{10}+1745378750437674672254525343s^{9}+1067124171571276165079329161s^{8}+5531923274479354261466695 44s^{7}+2244718815472323109755889 2s^{6}+7027128711382933309284923 4s^{5}+1675985651729512089493665 6s^{4}+2963427571961631174417201s^{3}+3679880939623824678753 21s^{2}+2875049397368658429499 8s+1066338997876680421860)$
c_4	$3(2s+3)^2(5538721056005861462500s^{34}+1700009304286779480012 00s^{33}+2521542870629614052494182s^{32}+2406903386107390039319428 8s^{31}+1661121402394717302241279 50s^{30}+8828619847432481952202270 68s^{29}+3758135559481861075860190560s^{28}+1315578765825874123589075580 0s^{27}+3858713497344878636633024 8032s^{26}+9612910934558376588663832 1232s^{25}+2054471477463803356647687 54954s^{24}+3794494533759267788710286 92368s^{23}+6087767809804353811451779 31646s^{22}+8512518397318974970676005 41072s^{21}+1039128387112340409627618 422211s^{20}+1107338907683241287910817 574616s^{19}+1028236425910911889110211 542327s^{18}+8286533887195509978413345 99814s^{17}+5756798600992144609913367 67122s^{16}+3411139802508743857151954 98752s^{15}+1696021121609668732446725 54562s^{14}+6901912188052417810839515 2092s^{13}+2219227596550968165776317 2138s^{12}+5513039324469607430622101 184s^{11}+1308552733582056616685717 799s^{10}+6228224269682729356107505 62s^{9}+4412082186155643660214786 80s^{8}+2510866267449979188062218 32s^{7}+1055427263431979624476683 30s^{6}+3318271204175103819268490 4s^{5}+7819078036046387444720541s^{4}+1353372939412637489000388s^{3}+1635933722331722818732 02s^{2}+1239672729966108251209 2s+4448137902935067283 68)$
c_3	$6(2s+3)^2(6516142418830425250 00s^{35}+2061811908364331856540 0s^{34}+3156213349996927861511 16s^{33}+3113065178572047666007860s^{32}+2222979754518395268052028 4s^{31}+1224225617399414815998354 84s^{30}+5408405145287421117617769 12s^{29}+1968384017248063183323104 976s^{28}+6014334916396782599305697 730s^{27}+1564273350905910799494783 0402s^{26}+3499106342792033984966407 0098s^{25}+6783451224245674486841205 3510s^{24}+1146107356416358232326805 58264s^{23}+1694202892194124436822450 06436s^{22}+2196301294579059436764574 75806s^{21}+2499148884589175758632804 85878s^{20}+2494580325094377405252407 77731s^{19}+2179203808681064934815157 56421s^{18}+1658682998392915454101611 33771s^{17}+1091968041316217564562484 26789s^{16}+6145574397104443845110768 3232s^{15}+2901082085745016093381214 0136s^{14}+1111172476982043580911263 9700s^{13}+3230978841571830756780993 930s^{12}+5977340713166300300960714 52s^{11}+1947145671045436300515266 4s^{10}-1441382310935022454149972 6s^{9}+9684588377731643071927236s^{8}+1256021958303918550403991 0s^{7}+6629813221106696409536742s^{6}+2267528918316638233964400s^{5}+5495108516124471979461 00s^{4}+9516959371683531754669 8s^{3}+1133337907636920896160 6s^{2}+8377615455010504271 46s+2911835924761782947 4)$

c_2	$(2s+3)^2(651614241883042525000s^{36}+2123612870508923148360 0s^{35}+335176828913504517258492s^{34}+3412480272349060314590760s^{33}+25184187621847674304218612s^{32}+143532477912633204107111040s^{31}+657199533413487737746005840s^{30}+2483057802762141981372604872s^{29}+7890444256414221204984097182s^{28}+21386706626264438997499257504s^{27}+49968626609865032054943532722s^{26}+101443944207534124952038283748s^{25}+180022708498134425424774257358s^{24}+280473316548653330703764939232s^{23}+384770188717561290211106882724s^{22}+465568061409025346140795055268s^{21}+497070528817890647576496874221s^{20}+467853455550722338016560291068s^{19}+387255091639245361342752551670s^{18}+280658492377779215502417761676s^{17}+176840580743027422215976620477s^{16}+957910627865555087240887423 20s^{15}+437858623301624990522095297 68s^{14}+163267368535271229917151558 24s^{13}+460821905008432679074893315 3s^{12}+763090880285610579007863108s^{11}-702735952364323683297077 16s^{10}-998540727043228715373926 04s^{9}-357260313779697920881889 25s^{8}-609903789530767014228760 8s^{7}+385997000711223832356168s^{6}+571809550541807937530952s^{5}+184789343789534461150530s^{4}+354883755696224721692 40s^{3}+436295085881317044922 8s^{2}+320602140994390122456s+10806813741383936712)$
c_1	$2s^2(2s+3)^2(5s+6)^2(1371819456595879000s^{33}+4271634557055966808 8s^{32}+6435444751783137019 08s^{31}+6247566555702580389060s^{30}+4391676556526924901666 0s^{29}+2381294370524934521705 40s^{28}+1036074226442125183419168s^{27}+3714930106381256786671824s^{26}+1118772868707257505376 3704s^{25}+2869670825417555723588 6484s^{24}+6335289140657084244295 6878s^{23}+1213301616269188486575 58398s^{22}+2027652551311758119054 86752s^{21}+2969521561596848429994 76188s^{20}+3821939060079245330600 66196s^{19}+4329771063731550080610 33636s^{18}+4318829425555373349451 82376s^{17}+3789227783217399510763 19160s^{16}+2916937858513690287806 62644s^{15}+1961472241580118928394 33394s^{14}+1144103496246446264232 12729s^{13}+5724888433243097639645 1627s^{12}+2413061795050375698405 1008s^{11}+8287189086050003227856 022s^{10}+2152391916458370195915 867s^{9}+3251846145974111403146 01s^{8}-3300797235187890347540 4s^{7}-4179251093868663889730 4s^{6}-1583118597299644973035 8s^{5}-3877006197061115690130s^{4}-6655060240921758556 80s^{3}-7841981439220991112 0s^{2}-5759186746951521828s-200126180395998828)$
c_0	$s^4(2s+3)^2(5s+6)^4(5487277826383516s^{30}+1629002070479364 48s^{29}+2337377142714373098s^{28}+2158816572989759829 6s^{27}+1442107043736372374 22s^{26}+7422068524214498072 76s^{25}+3061285798160471289822s^{24}+1039189563664458602011 2s^{23}+2958836372773561206903 6s^{22}+7165140410814613889709 6s^{21}+1491176200730274615474 36s^{20}+2688067051789587271872 48s^{19}+4221859922152866254547 36s^{18}+5801917523864989863237 12s^{17}+6996947243102316240642 72s^{16}+7417572618016561112259 84s^{15}+6916662471035836623516 12s^{14}+5670330877797167100233 52s^{13}+4080471316446565805592 96s^{12}+2570366447945656343830 08s^{11}+1411452641418268040735 76s^{10}+6717846053228888490951 6s^{9}+2749966305794295114104 1s^{8}+9582491278489242855672s^{7}+2803365139419823150782s^{6}+6756829533138365528 76s^{5}+1306594986712056470 52s^{4}+1948956305690965449 6s^{3}+2105305456045150908s^{2}+1465944860513900 88s+4941387170271576)$

Δ_2	$-36s^4(2s+3)^4(5s+6)^4(188188862149501274759871978264100000s^{66}$ $+1171982279367338935485204891772187040 0s^{65}$ $+3590340595158437649151724049466274089 92s^{64}$ $+7212088571266452516322419090023965301952s^{63}$ $+10683913202536878409344821445611408008 9896s^{62}$ $+12446539307453434624006123411233664305561 12s^{61}$ $+11874531272586759157048871425273776419646480s^{60}$ $+9539717463978084911740021792529086079754307 2s^{59}$ $+6586014211961623946750060924092706557833870 96s^{58}$ $+3968022355733696091765263130542489256493297296s^{57}$ $+2111718215787453698016965546515609174427692038 0s^{56}$ $+1002346955322263064366492155231201633504320407 84s^{55}$ $+427723545408455739277440670239201960498527445168s^{54}$ $+165169320489858311000198383167538370383880311128 0s^{53}$ $+58038813051285159280644727598895039061348407197 88s^{52}$ $+18645262399111732404927179918528703191829668776944s^{51}$ $+5498216092397086290715585398715769282805100212946 8s^{50}$ $+1493403271312119095060211916193032794760693070014 80s^{49}$ $+374738908504925756556865067810806785110272348870998s^{48}$ $+87095917136338876712545518393847454868443303273397 6s^{47}$ $+18791254231056775235326951135911186532510305432196 50s^{46}$ $+3770893425502604495219029177077658520798571516698604s^{45}$ $+7050015221784174490794108388251813073259643663091008s^{44}$ $+1229751697814540088737743012540597239130953523640481 6s^{43}$ $+2003816448416423665928915846433322332297986145636955 8s^{42}$ $+3053220493065830192217996030789798685096097311040106 0s^{41}$ $+4353974016564230774744937671615227473899797195238128 8s^{40}$ $+5814794272234047468845528446358546692682944712469086 4s^{39}$ $+7276560981967024598164222622942133354281616530078647 8s^{38}$ $+8535222627194411740352205447227499061598741207765802 0s^{37}$ $+9386118512358018098652883224139505412412521033278797 8s^{36}$ $+9677367622870745479418594983628077002220070263778913 6s^{35}$ $+9353550534056682983528096959246190247987486973707032 4s^{34}$ $+8472657307033122222401048138408764617479499951727572 0s^{33}$ $+7189292302539646724238556713520429645936760693117107 2s^{32}$ $+5710750538582167076586109442116935829639912622263872 0s^{31}$ $+4242947557936920029498039681532283983986502171148171 2s^{30}$ $+2945340224867131137091897558637195040010087585918442 4s^{29}$ $+1907688741881547559139122534482867333198543394985974 5s^{28}$ $+1150963090261374243005671011302597994636892936359156 0s^{27}$ $+6455314570589262935425947995355398219841083619460275s^{26}$ $+3357432561061020603580382785922585900536396393910862s^{25}$ $+1614483581569691108434974127018929362203153022365980s^{24}$ $+7151843921239467644322741543961403561291367742781 52s^{23}$ $+2905615937078824111549712703038991345504578369200 80s^{22}$ $+1076866147250386418961409060935085408978065543348 84s^{21}$ $+3617512376782787657128716857177209606664184218971 5s^{20}$ $+1093629541362141784014724465858457912720717438935 2s^{19}$ $+2956231446951945885452453177600650222360031507853s^{18}$ $+7143283154950873347674525118298108303417045322 94s^{17}$ $+1578622563973412013157962625289520289452101601 78s^{16}$ $+3466918481540893224348429068849868462291600055 2s^{15}$ $+8792208859688249070338942374434094956929968434s^{14}$ $+2728958913497904679605629304348283067662149156s^{13}$ $+9085144773106795189912395340847764985695491 30s^{12}$ $+2819973846515718244314169254440158181016966 00s^{11}$ $+7659571256132994444113894245418887273849699 2s^{10}$

	$+1782323536880591650543354401619541599668158 4s^9$ $+3522507407940359387775769801382214828420516s^8$ $+5867820050646080008320434741079193955195 52s^7$ $+8148452315276322918847755584680364066167 2s^6$ $+9275689993243887053899463828498929963344s^5$ $+8438081673219230573799541298816126340 96s^4$ $+5898808758941212962363346647613729785 6s^3$ $+2972581287433071568118115850779633072s^2$ $+959233912036916287714016721581540 16s$ $+148335141036636539337219080656939 2)$

The last lemma presents some properties for the population-level SGPO and GRPO dynamics.

Lemma B.4. *Under the assumptions from Theorem 3.3, the following statements hold true,*

(i) $p^{(k)}_{SGPO}, q^{(k)}_{SGPO}, p^{(k)}_{GRPO}, q^{(k)}_{GRPO} \in (0,1)$ *for all* $k \geq 0$.
(ii) $p^{(k)}_{SGPO}, q^{(k)}_{SGPO}, p^{(k)}_{GRPO}, q^{(k)}_{GRPO}$ *are strictly increasing in* k *and lie in* $(\frac{1}{2}, 1)$ *for all* $k \geq 1$.
(iii) $p^{(k)}_{SGPO} > q^{(k)}_{SGPO}$ *for all* $k \geq 1$.

Proof. We first rewrite the update rule in Eq. (3) as follows,

$$
\begin{aligned}
p^{(k+1)}_{\texttt{SGPO}} &= p^{(k)}_{\texttt{SGPO}} \frac{e^{\Delta^{(k)}_{\texttt{SGPO},p}}}{1-p^{(k)}_{\texttt{SGPO}}+p^{(k)}_{\texttt{SGPO}} e^{\Delta^{(k)}_{\texttt{SGPO},p}}}, \quad \text{where} \quad \Delta^{(k)}_{\texttt{SGPO},p} = p^{(k)}_{\texttt{SGPO}}(1-p^{(k)}_{\texttt{SGPO}}), \\
q^{(k+1)}_{\texttt{SGPO}} &= q^{(k)}_{\texttt{SGPO}} \frac{e^{\Delta^{(k)}_{\texttt{SGPO},q}}}{1-q^{(k)}_{\texttt{SGPO}}+q^{(k)}_{\texttt{SGPO}} e^{\Delta^{(k)}_{\texttt{SGPO},q}}}, \quad \text{where} \quad \Delta^{(k)}_{\texttt{SGPO},q} = (p^{(k)}_{\texttt{SGPO}})^2 q^{(k)}_{\texttt{SGPO}}(1-q^{(k)}_{\texttt{SGPO}}), \\
p^{(k+1)}_{\texttt{GRPO}} &= p^{(k)}_{\texttt{GRPO}} \frac{e^{\Delta^{(k)}_{\texttt{GRPO},p}}}{1-p^{(k)}_{\texttt{GRPO}}+p^{(k)}_{\texttt{GRPO}} e^{\Delta^{(k)}_{\texttt{GRPO},p}}}, \quad \text{where} \quad \Delta^{(k)}_{\texttt{GRPO},p} = p^{(k)}_{\texttt{GRPO}}(1-p^{(k)}_{\texttt{GRPO}}) q^{(k)}_{\texttt{GRPO}}, \\
q^{(k+1)}_{\texttt{GRPO}} &= q^{(k)}_{\texttt{GRPO}} \frac{e^{\Delta^{(k)}_{\texttt{GRPO},q}}}{1-q^{(k)}_{\texttt{GRPO}}+q^{(k)}_{\texttt{GRPO}} e^{\Delta^{(k)}_{\texttt{GRPO},q}}}, \quad \text{where} \quad \Delta^{(k)}_{\texttt{GRPO},q} = p^{(k)}_{\texttt{GRPO}} q^{(k)}_{\texttt{GRPO}}(1-q^{(k)}_{\texttt{GRPO}}).
\end{aligned}
$$

First of all, the uniform initialization yields the desired result for $k = 0$. Suppose $p^{(k)}_{\texttt{SGPO}} \in (0,1)$ for some $k \geq 0$. Then, we have

$$
1 - p^{(k)}_{\texttt{SGPO}} + p^{(k)}_{\texttt{SGPO}} e^{\Delta^{(k)}_{\texttt{SGPO},p}} > p^{(k)}_{\texttt{SGPO}} e^{\Delta^{(k)}_{\texttt{SGPO},p}} > 0,
$$

which implies $p^{(k+1)}_{\texttt{SGPO}} \in (0,1)$. By induction, we have $p^{(k)}_{\texttt{SGPO}} \in (0,1)$ for all $k \geq 0$. Similarly, we can show that $q^{(k)}_{\texttt{SGPO}}, p^{(k)}_{\texttt{GRPO}}, q^{(k)}_{\texttt{GRPO}} \in (0,1)$ for all $k \geq 0$.

Furthermore, we have $\Delta^{(k)}_{\texttt{SGPO},p} > 0$ since $p^{(k)}_{\texttt{SGPO}} \in (0,1)$. This implies

$$
\frac{p^{(k+1)}_{\texttt{SGPO}}}{p^{(k)}_{\texttt{SGPO}}} = \frac{1}{(1-p^{(k)}_{\texttt{SGPO}}) e^{-\Delta^{(k)}_{\texttt{SGPO},p}} + p^{(k)}_{\texttt{SGPO}}} > \frac{1}{1-p^{(k)}_{\texttt{SGPO}}+p^{(k)}_{\texttt{SGPO}}} = 1.
$$

Since $p^{(0)}_{\texttt{SGPO}} = \frac{1}{2}$, we have $p^{(k)}_{\texttt{SGPO}} \in (\frac{1}{2}, 1)$ for all $k \geq 1$. Similarly, we can show that $q^{(k)}_{\texttt{SGPO}}, p^{(k)}_{\texttt{GRPO}}, q^{(k)}_{\texttt{GRPO}}$ are strictly increasing and lie in $(\frac{1}{2}, 1)$.

Finally, we have $p^{(0)}_{\texttt{SGPO}} \geq q^{(0)}_{\texttt{SGPO}}$. Thus, it suffices to show that $p^{(k)}_{\texttt{SGPO}} \geq q^{(k)}_{\texttt{SGPO}}$ implies $p^{(k+1)}_{\texttt{SGPO}} > q^{(k+1)}_{\texttt{SGPO}}$ for all $k \geq 0$. Indeed, Lemma B.1(i) and $p^{(k)}_{\texttt{SGPO}} \geq q^{(k)}_{\texttt{SGPO}}$ yield

$$
p^{(k+1)}_{\texttt{SGPO}} = \exp(f_{11}(p^{(k)}_{\texttt{SGPO}})) \geq \exp(f_{11}(q^{(k)}_{\texttt{SGPO}})) = \exp(\log(q^{(k)}_{\texttt{SGPO}}) + h_{q^{(k)}_{\texttt{SGPO}}}(q^{(k)}_{\texttt{SGPO}}(1-q^{(k)}_{\texttt{SGPO}}))).
$$

Then, Lemma B.1(ii) and $p^{(k)}_{\texttt{SGPO}} \in (0,1)$ yield

$$
\exp(\log(q^{(k)}_{\texttt{SGPO}}) + h_{q^{(k)}_{\texttt{SGPO}}}(q^{(k)}_{\texttt{SGPO}}(1-q^{(k)}_{\texttt{SGPO}}))) > \exp(\log q^{(k)}_{\texttt{SGPO}} + h_{q^{(k)}_{\texttt{SGPO}}}((p^{(k)}_{\texttt{SGPO}})^2 q^{(k)}_{\texttt{SGPO}}(1-q^{(k)}_{\texttt{SGPO}}))).
$$

In addition, we have

$$q_{\texttt{SGPO}}^{(k+1)} = \exp(f_{12}(p_{\texttt{SGPO}}^{(k)}, q_{\texttt{SGPO}}^{(k)})) = \exp(\log q_{\texttt{SGPO}}^{(k)} + h_{q_{\texttt{SGPO}}^{(k)}}((p_{\texttt{SGPO}}^{(k)})^2 q_{\texttt{SGPO}}^{(k)}(1 - q_{\texttt{SGPO}}^{(k)}))).$$

Putting these pieces together yields $p_{\texttt{SGPO}}^{(k+1)} > q_{\texttt{SGPO}}^{(k+1)}$. □