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

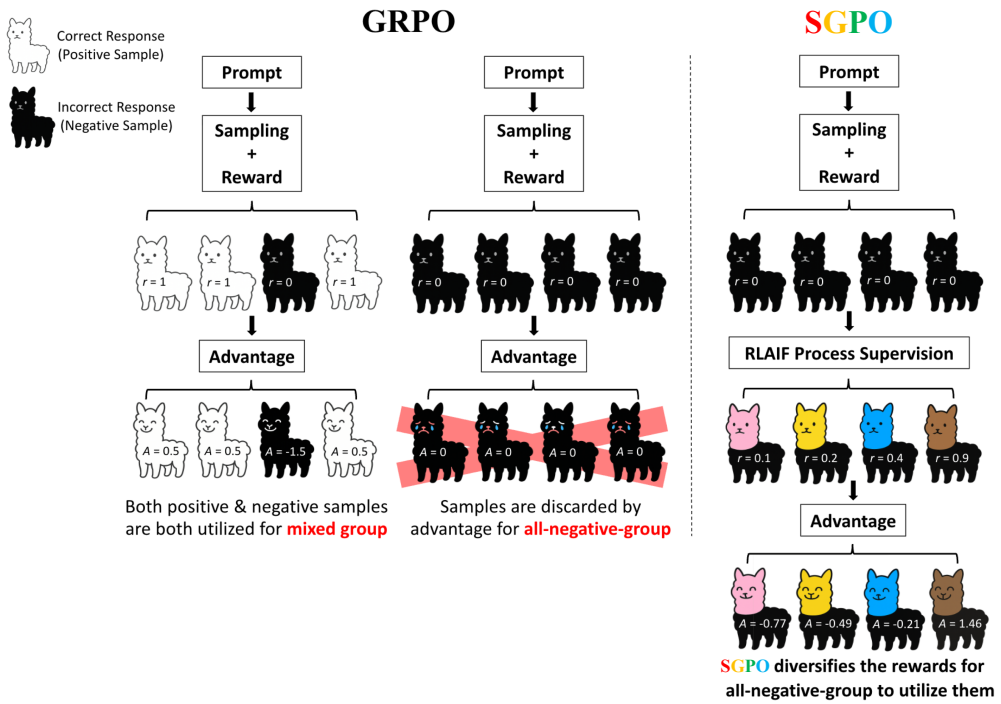

Figure 1: Main pipeline for Stepwise Guided Policy Optimization. On each llama, r indicates the reward of the sampled response and A indicates response's advantage through group relative computation.

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* $p^{(0)}_{\texttt{GRPO}} = q^{(0)}_{\texttt{GRPO}} = p^{(0)}_{\texttt{SGPO}} = q^{(0)}_{\texttt{SGPO}} = \frac{1}{2}$ *and* $\eta = 1$[3] *for GRPO and SGPO. Then, we have that (i) GRPO and SGPO achieve successful learning:* $p^{(k)}_{\texttt{GRPO}}, q^{(k)}_{\texttt{GRPO}}, p^{(k)}_{\texttt{SGPO}}, q^{(k)}_{\texttt{SGPO}} \to 1$ *as* $k \to +\infty$*; (ii) SGPO outperforms GRPO in learning the "good" action in the first step:* $p^{(k)}_{\texttt{SGPO}} > p^{(k)}_{\texttt{GRPO}}$ *for all* $k \geq 1$*; (iii) SGPO outperforms GRPO in learning the optimal policy:* $p^{(k)}_{\texttt{SGPO}} q^{(k)}_{\texttt{SGPO}} > p^{(k)}_{\texttt{GRPO}} q^{(k)}_{\texttt{GRPO}}$ *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 $p_{\texttt{SGPO}}^{(k)} q_{\texttt{SGPO}}^{(k)} > (p_{\texttt{GRPO}}^{(k)})^2$ implies $p_{\texttt{SGPO}}^{(k+1)} q_{\texttt{SGPO}}^{(k+1)} > (p_{\texttt{GRPO}}^{(k+1)})^2$ for $k \geq 1$. By Lemma B.4(iii), we know $p_{\texttt{SGPO}}^{(k)} > q_{\texttt{SGPO}}^{(k)}$ for all $k \geq 1$. Thus, Lemma B.2 implies that

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

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

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

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., $p_{SGPO}^{(k)}$ vs. $p_{GRPO}^{(k)}$), while the right panel shows the probability of learning the optimal policy (i.e., $p_{SGPO}^{(k)} q_{SGPO}^{(k)}$ vs. $p_{GRPO}^{(k)} q_{GRPO}^{(k)}$). 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).

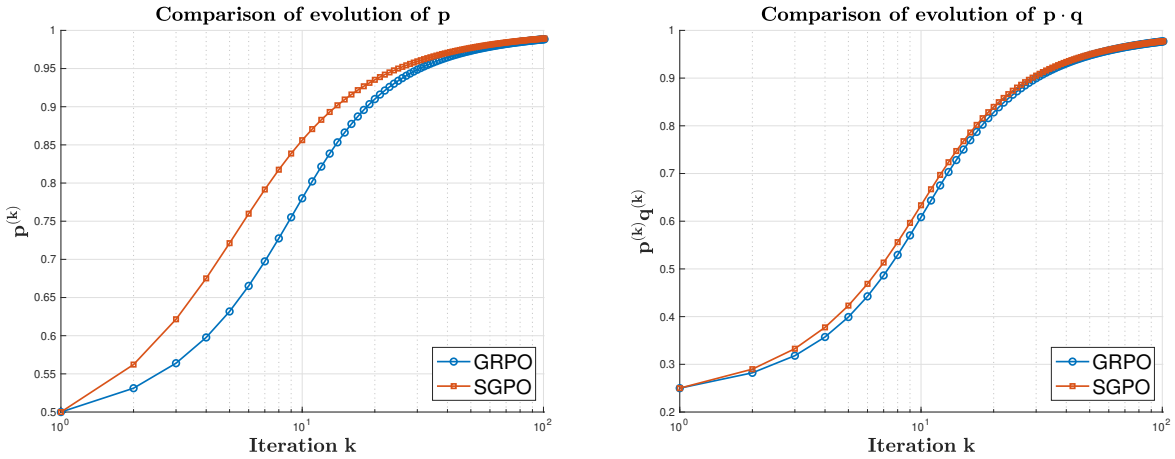

Figure 2: Learning dynamics of GRPO and SGPO in the simplified setting.

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 for which at least one of the k sampled responses is correct, 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.

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**

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 implemented 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-

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

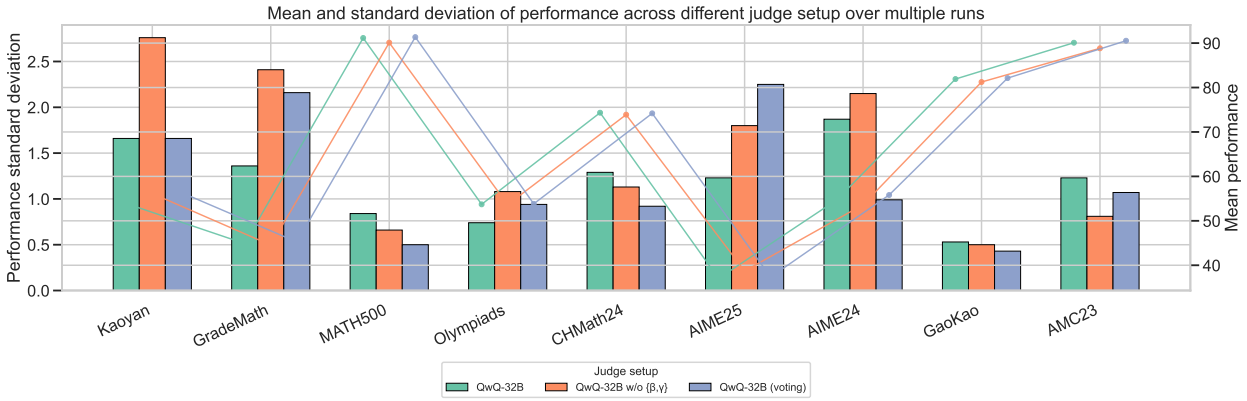

Figure 3: Mean and standard deviation over multiple runs across different judge model setups.

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

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 `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.

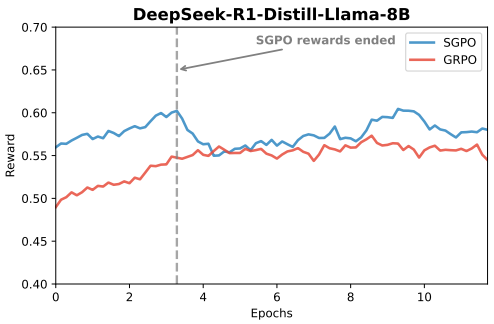

Figure 4: Evaluation results on GRPO and SGPO. SGPO rewards end at epoch 3.

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,

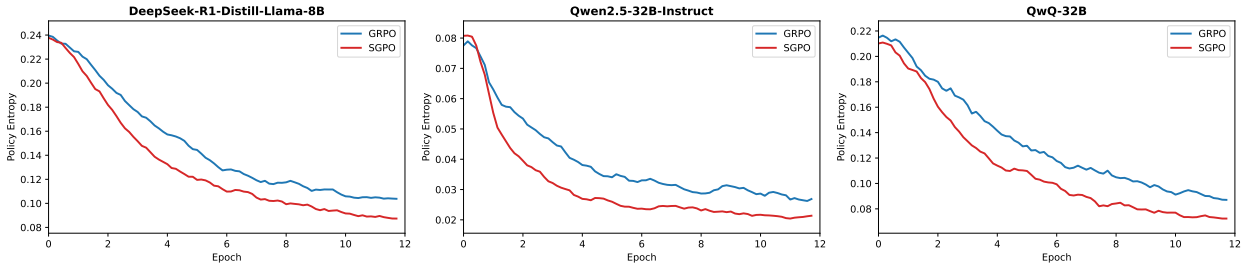

Figure 5: Policy entropy levels during training for GRPO and SGPO across different base models.

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 arXiv: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 arXiv: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.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. VinePPO: Accurate credit assignment in RL for LLM mathematical reasoning. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24*, 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.

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*, 2024b. 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 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., 2024b). 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) 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,

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

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

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

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 $\bar{g}_{\mathrm{SGPO}}$ or $\bar{g}_{\mathrm{GRPO}}$. 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 $\bar{g}_{\mathrm{SGPO}}(\theta)$ 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 $\bar{g}_{\mathrm{GRPO}}(\theta)$ 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

$$p^{(k+1)}_{\mathrm{SGPO}} = \frac{p^{(k)}_{\mathrm{SGPO}} e^{p^{(k)}_{\mathrm{SGPO}}(1-p^{(k)}_{\mathrm{SGPO}})}}{1-p^{(k)}_{\mathrm{SGPO}}+p^{(k)}_{\mathrm{SGPO}} e^{p^{(k)}_{\mathrm{SGPO}}(1-p^{(k)}_{\mathrm{SGPO}})}} = \exp(f_{11}(p^{(k)}_{\mathrm{SGPO}})),$$
$$p^{(k+1)}_{\mathrm{GRPO}} = \frac{p^{(k)}_{\mathrm{GRPO}} e^{p^{(k)}_{\mathrm{GRPO}}(1-p^{(k)}_{\mathrm{GRPO}})q^{(k)}_{\mathrm{GRPO}}}}{1-p^{(k)}_{\mathrm{GRPO}}+p^{(k)}_{\mathrm{GRPO}} e^{p^{(k)}_{\mathrm{GRPO}}(1-p^{(k)}_{\mathrm{GRPO}})q^{(k)}_{\mathrm{GRPO}}}} = \exp(f_{21}(p^{(k)}_{\mathrm{GRPO}}, q^{(k)}_{\mathrm{GRPO}})).$$

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

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

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

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

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

Furthermore, we have

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

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

Moreover, we have

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

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

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

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 $u > 0$:

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

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

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

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

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

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 $s > 0$,

$$19471456710454363005152664s^{10} + 96845883777316430719272 36s^8 > 14413823109350224541499726s^9.$$

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

$$\text{LHS} > 1.9\times10^{25}s^{10} + 9.6\times10^{24}s^8 > 2\sqrt{182.4}\times10^{24}s^9 > 2.7\times10^{25}s^9 > \text{RHS}.$$

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:

$$95791062786555508724088742320s^{15} + 57180955054180793753095 2s^5 > 70273595236432368329707716s^{10};$$
$$43785862330162499052209529768s^{14} + 18478934378953446115053 0s^4 > 99854072704322871537392604s^9;$$
$$16326736853527122991715155824s^{13} + 35488375569622472169240s^3 > 35726031377969792088188925s^8;$$
$$46082190500843267907489331 53s^{12} + 43629508588131704492 28s^2 > 60990378953076701422876 08s^7.$$

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

$$\text{LHS} > 9.5\times10^{28}s^{15} + 5.7\times10^{23}s^5 > 2\sqrt{541.5}\times10^{25}s^{10} > 4.6\times10^{26}s^{10} > \text{RHS};$$
$$\text{LHS} > 4.3\times10^{28}s^{14} + 1.8\times10^{23}s^4 > 2\sqrt{77.4}\times10^{25}s^9 > 1.7\times10^{26}s^{10} > \text{RHS};$$
$$\text{LHS} > 1.6\times10^{28}s^{13} + 3.5\times10^{22}s^3 > 2\sqrt{5.6}\times10^{25}s^9 > 4.7\times10^{25}s^8 > \text{RHS};$$
$$\text{LHS} > 4.6\times10^{27}s^{12} + 4.3\times10^{21}s^2 > 2\sqrt{19.7}\times10^{24}s^7 > 8.8\times10^{24}s^7 > \text{RHS}.$$

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. □

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

Notation	Value
c	437675956526049436836
c_{20}	$2(2s+3)^2(1714774320744848750s^{18}+26610409260691576200s^{17}+19177874646880231718 1s^{16}+85019414985508231922 4s^{15}+25870824342900458060 49s^{14}+57044159060391607318 74s^{13}+93661975819632320544 60s^{12}+11563054951307567026248s^{11}+10670965452123886149660s^{10}+71871767695820752612 92s^{9}+33729721689965794300 17s^{8}+10720828361582207039 52s^{7}+37030209404289077128 5s^{6}+32990167737690242581 8s^{5}+28279998680561626786 2s^{4}+15116827017089336500 8s^{3}+49139849518345513368s^{2}+9090603727935062976s+742484948385838248)$
c_{19}	$8(2s+3)^2(8573871603724243750s^{19}+14118375184879884045 0s^{18}+10850654575356110840 97s^{17}+51599054246626785276 63s^{16}+16962017821014041355285s^{15}+40761515892906930261393s^{14}+73777358861762677983126s^{13}+10196816347629194227764 3s^{12}+10771955018327920294533 6s^{11}+85951871005332247942347s^{10}+50477142420763872747039s^{9}+21228461710484227270812s^{8}+71755762533602862021 93s^{7}+38216421194479088101 38s^{6}+33431406025396150709 82s^{5}+23080587413803109461 44s^{4}+10465286915656214713 44s^{3}+30076928586074414602 8s^{2}+50403014643440592936s+3785769852305984190)$
c_{18}	$2(2s+3)^2(3258071209415212625 00s^{20}+56739873809773123962 00s^{19}+46320143614200183575358s^{18}+23516462486845543431 4740s^{17}+83033078234649987863 1402s^{16}+21591058927666966255 08432s^{15}+42680663647777106288 78112s^{14}+65211777262765861916 28264s^{13}+77429334197200253596 60131s^{12}+71109113618735821090 51992s^{11}+49764748763830706631 95517s^{10}+26002307191355911482 69222s^{9}+10353079281161503861 09695s^{8}+42049522015816578367 2300s^{7}+28743634586216802620 9421s^{6}+22574991211752704712 0354s^{5}+13220555392476575702 3286s^{4}+52546262098747895532 864s^{3}+13596497258861930544 1108s^{2}+20873057772457298889 36s+145293329180967197454)$
c_{17}	$12(2s+3)^2(3258071209415212625 00s^{21}+59829921917002688553 00s^{20}+51700930512613327403406s^{19}+27907978077725947794 4590s^{18}+10531873179450453647679 66s^{17}+29454589028098495868768 10s^{16}+63109003318653630127438 44s^{15}+10554182290179327214273 482s^{14}+13894164004300292404029 789s^{13}+14397511755502695216399 777s^{12}+11648187532971253859583 195s^{11}+72536728432498948752036 21s^{10}+34639340760627119841481 83s^{9}+14013118711246369225106 79s^{8}+70963507345380338433066 3s^{7}+52660130078172271896962 1s^{6}+36641146292090355701412 0s^{5}+18779093836691745063360 6s^{4}+66828265788979238418684s^{3}+15774993964318985462592s^{2}+22381772821590129459 66s+145311275743961970078)$
c_{16}	$3(2s+3)^2(5538721056005861462500s^{22}+10696394904119483034480 0s^{21}+97537241738707509555711 0s^{20}+55777018695676356245163 12s^{19}+22401070138854784562671 602s^{18}+67034725561809321462257 220s^{17}+15468959266006177578068 3034s^{16}+28088953980133276709409 1608s^{15}+40566432899300593609815 8220s^{14}+46743223459745032337765 4624s^{13}+42816291170183647045348 8816s^{12}+30885119317741985964812 0664s^{11}+17392371150762459541255 5792s^{10}+78611043902295277305014 364s^{9}+34502723932108659968068 191s^{8}+20784869678087847011350 512s^{7}+15211070491275270899260 479s^{6}+94397951694788177205573 90s^{5}+43236427772992108674668 40s^{4}+13983362552252256686861 76s^{3}+30416510338341914536668 0s^{2}+40170170984468888396880s+244568879953459209 1926)$

c_{15}	$12(2s+3)^2(4430976844804689170000s^{23}+897736246587880721196 00s^{22}+861449789967478967902968s^{21}+52020523405703639617992 72s^{20}+221509586747221476368878 80s^{19}+706107689774315971385024 16s^{18}+174547747719038741579249148s^{17}+341846798830220759744006802s^{16}+537017897494050473220887475s^{15}+680386317088643909072652357s^{14}+694900917445527709102606203s^{13}+568854294351452352559155357s^{12}+370162500862325208186949407s^{11}+192089934615274810143113637s^{10}+855600003744048710780447 43s^9+421889790693255956904848 94s^8+279864829836354489161494 77s^7+193494730005279484231600 98s^6+108265495224176505823479 03s^5+449911900941991185014790 3s^4+133726808192964610911642 9s^3+270188727447397870150299s^2+334121914487222028717 93s+191648133946778922 7047)$
c_{14}	$3(2s+3)^2(4430976844804689170000 0s^{24}+939760900846202799633600s^{23}+946588223144997927058161 6s^{22}+601894156442872654302402 24s^{21}+270831851136366961169859 120s^{20}+916082243422713193340650 464s^{19}+241460571833561888085397 0032s^{18}+507178480505041003582316 7576s^{17}+860600335178662801913981 1024s^{16}+118820727215592680025785 94336s^{15}+133731357360665889152672 25192s^{14}+122337897530173273813986 56592s^{13}+903808724360213876819507 8704s^{12}+536928655269087344627611 7328s^{11}+262353585257354926709155 5300s^{10}+119607178098293965186514 4096s^9+660078244623496780263588 075s^8+451393195997666208667458 852s^7+290343677073268941864046 305s^6+148042042832629803967321 050s^5+564641933310434041167417 84s^4+155594784301928508298188 24s^3+293917901549477584712119 2s^2+342046929585497061253176s+185579146468002784590 54)$
c_{13}	$6(2s+3)^2(4430976844804689170000 0s^{25}+981785555104524878071200s^{24}+103571323344221695391126 88s^{23}+691661317833842749970623 20s^{22}+327906609514495942625889 552s^{21}+117287293644201929682500 4672s^{20}+328306372162931054591158 9320s^{19}+736033323922078596671432 2212s^{18}+134112380919595198011738 55314s^{17}+200308743312586394476164 05430s^{16}+246126320665844350630456 93896s^{15}+248618079432432478485647 02728s^{14}+205546163099386765640881 93632s^{13}+138287017746444456071984 89296s^{12}+759330701801977694759131 6692s^{11}+357316443768253965114029 8914s^{10}+171120540089863474143258 8709s^9+102643620132548287322773 1181s^8+693346698158130213351442 587s^7+413729468327452341092783 823s^6+194077643830831926535960 674s^5+685619052202317756190510 80s^4+176409705088033325463979 44s^3+313263376945319873280103 2s^2+344560424227000935565860s+177450969254891684236 20)$
c_{12}	$3(2s+3)^2(1440067474561523980250 00s^{26}+332738318042925260865360 0s^{25}+366868066779219215750244 12s^{24}+256712663785782687868607 040s^{23}+127887848844391886971897 7532s^{22}+482250196563585207839970 8760s^{21}+142849371676054076251236 13032s^{20}+340396757340205312187136 39960s^{19}+662691940873478048426418 90936s^{18}+106420251181999059483739 591464s^{17}+141673718347886548611527 896944s^{16}+156512253734744849831503 592064s^{15}+143119458698672790284020 359156s^{14}+107772919457957006535562 903236s^{13}+665835633168189319051172 87625s^{12}+342080279777871014800067 5072s^{11}+158214305102943398914167 81741s^{10}+803091352209266474305708 0482s^9+505052178573414373425714 5460s^8+329237430151157964493910 2872s^7+182721236521121211517132 9068s^6+795146625577647417589837 740s^5+262142182103903879108812 389s^4+633544272142961809377795 84s^3+106256104197218980943202 72s^2+110877007990059372292236 0s+543716225994186505217 07)$

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}	$(2s+3)^2(63362968880707055131000 0s^{28}+1584239110567672292139120 0s^{27}+1897621049684466450141042 96s^{26}+1448899233419905865696230 704s^{25}+7915017749037631812296989 272s^{24}+3291137409296388117112885 1808s^{23}+1081841136804198369570625 71120s^{22}+2881814574059132039155601 47656s^{21}+6325630775471257682897241 75852s^{20}+1156966248842167228077853 758672s^{19}+1775610230389152091840398 326292s^{18}+2294661769917892997575576 903488s^{17}+2498192965867701162268688 835924s^{16}+2285659112305489254434518 711416s^{15}+1748991971403770816761602 941934s^{14}+1113797684163785846264967 323376s^{13}+5924913951597435743900967 42111s^{12}+2744766007155415656150112 13796s^{11}+1270446195852713506501888 45452s^{10}+7053858232614772386540776 5680s^9+4488968110254001706707249846 5s^8+2660289888716075956783633452 0s^7+1296776974656174947900170128 0s^6+4960789745148078555411299844s^5+1450732098731590049595423084s^4+3139308944683290821452849 56s^3+4752567062061261105861801 9s^2+4506795195444098216749962s+2019810664389200880741 21)$
c_9	$2(2s+3)^2(28801349491230479605000 0s^{29}+7474247118895785746840400s^{28}+9308513797882609289684184s^{27}+7404022441838906136102386 16s^{26}+4222492016436341447317422840s^{25}+1837347458773775714357659 7976s^{24}+6337431126498352669577742938 4s^{23}+1776892961989847704086466054 92s^{22}+4119921298476838644538559778 90s^{21}+7992705700875121584791677780 14s^{20}+1307454563265970530661662310 488s^{19}+1811440906673507850196226957 292s^{18}+2129008833631551803404401966 618s^{17}+2120318902997465121470552812 398s^{16}+1782752578142774040882126045 846s^{15}+1258348636755929955587480574 282s^{14}+7422229506057785006481351208 08s^{13}+3687595326777177603238716204 48s^{12}+1632923401776166653452978780 34s^{11}+7562813554420816525428037104 0s^{10}+4285873595461389310248944256 9s^9+2681560986049355543901710696 7s^8+1515406630759595340156585468 0s^7+6986502898234763627448498006s^6+2529774938694747662152377363s^5+7023305018106747704878627 23s^4+1447326550278639488278622 61s^3+2092454720837126248556029 5s^2+1899529777608715114521399s+8166786538840395351817 5)$

c_8	$3(2s+3)^2(1440067474561523980250000s^{30}+3873703685787439628342400s^{29}+5008695060602392586847903 6s^{28}+41434729334310331141335720 0s^{27}+24624557164178955534668144 04s^{26}+11190311839066344259536714 024s^{25}+40409295972773177050104566 556s^{24}+11894713481130666949027261 6200s^{23}+29046044800144993798660964 1984s^{22}+59565098287696170044424791 6136s^{21}+10343945033328370448328719 73660s^{20}+15291225223983414522377482 64328s^{19}+19291701121444085312952472 40352s^{18}+20772757298357798986279724 52564s^{17}+19044000925848491473988032 57857s^{16}+14792950192467652444216179 60408s^{15}+96723258758152313412876769 0737s^{14}+52982714945629451357986685 7762s^{13}+24585014937115122024760235 0838s^{12}+10358618198141596236507935 0136s^{11}+47441969831005210765856651 490s^{10}+27017370328613731242988726 116s^9+16543604074621342316534833896s^8+89598081048155039681209011 60s^7+39341484805034175878771776 53s^6+13567085019266032207440806 94s^5+35935601005081233078747327 9s^4+70800398025365428901919252s^3+98057056394418600109930 74s^2+85429442798900619720505 2s+3530618607561154024305 6)$
c_7	$12(2s+3)^2(2215488422402344585000 0s^{31}+6169667403272286743484 00s^{30}+82709070736961626834304 88s^{29}+71054888374958447419331 400s^{28}+43931915711054680381189 6968s^{27}+20811530357045666581306 24800s^{26}+78516994084586808022426 34484s^{25}+24207516760625423072207 073414s^{24}+62092517428805875866794 674881s^{23}+13419134915902123968841 4857431s^{22}+24651837873231177502946 4265197s^{21}+38722904880065353173811 4739999s^{20}+52184907756189465763639 8449655s^{19}+60401049965902695146089 7607741s^{18}+59970619395213696432112 7694249s^{17}+50893260009270036230801 4694866s^{16}+36692384583010673508719 9088303s^{15}+22297067887827188553515 9028828s^{14}+11351075140481669158976 1355815s^{13}+48959621580159733845289 884633s^{12}+19343819733170955544319 552163s^{11}+85709676482434192986603 77211s^{10}+48422817052135271084165 13279s^9+29091383745623179903600 31394s^8+15229213035961739961385 75905s^7+64221296552864057231043 6879s^6+21234740879260983664481 5359s^5+53940504869138165983355 556s^4+10200981421548866272272 021s^3+13576094237482264271577 78s^2+11378258779245747678886 5s+4528476210896135134182)$
c_6	$3(2s+3)^2(4430976844804689170000 0s^{32}+1275958134912779427134 400s^{31}+17712124648743520374246 000s^{30}+15780078330319234248480 7776s^{29}+10134760020968226539342 47536s^{28}+49963468114719944382409 70400s^{27}+19656891049003759997738 219664s^{26}+63343018585986823082619 729288s^{25}+17025853704363277922947 3723584s^{24}+38671925597018899391722 4574336s^{23}+74919597303712220212720 2135768s^{22}+12459527898651638421599 25500256s^{21}+17859909693607150460001 17363568s^{20}+22109118970851524307378 22315488s^{19}+23633256569498336318699 30474232s^{18}+21763865344758685432501 78452024s^{17}+17185192695977307690462 19931925s^{16}+11548989976568146170383 09424924s^{15}+65399379968237071658504 0976045s^{14}+30909781238758984299404 9088882s^{13}+12264998651480592797581 1013678s^{12}+44139321593386486437541 304232s^{11}+18194600512073131179652 474218s^{10}+10080729455247177149466 360108s^9+60121544950098328631438 60703s^8+30867139019044853020774 23912s^7+12645365130026448915535 73532s^6+40426900431353599770906 4188s^5+99085318554137395893467 265s^4+18066869592965845573731 768s^3+23180375592196844545338 93s^2+18733587778459577298978s+71920698153647960662 29)$

c_5	$6(2s+3)^2(8861953689609378340000s^{33}+263596557834220301114400s^{32}+3784460184258343211722032s^{31}+34920466929143934879808368s^{30}+232642130166877120425752976s^{29}+11916978009453670864580550 72s^{28}+48806947518804415592184766 32s^{27}+16406950824469547680560778764s^{26}+46112865280418851489322074134s^{25}+10981227494496035115599178601 8s^{24}+22372585643351671399884980 5632s^{23}+39266014091709822788394101 9880s^{22}+59645033289927608500675400 0436s^{21}+78624155913942530616774743 7324s^{20}+90016247244570627082720573 5408s^{19}+89411845619881376237474487 4842s^{18}+76801976570035682154104806 4211s^{17}+56718891951502940479512463 9815s^{16}+35689284917369058878427986 5923s^{15}+18888772644005809091812356 9807s^{14}+82758519677573937797801658678s^{13}+29755841968295155180938377160s^{12}+92656551239853153882294277 62s^{11}+32698047051242709609461794 92s^{10}+17453787504376746722545253 43s^9+10671241715712761650793291 61s^8+55319232744793542614666954 4s^7+22447188154723231009755889 2s^6+70271287113829333092849234s^5+16759856517295120894936656s^4+29634275719616311744172 01s^3+367988093962382467875321s^2+28750493973686584294998s+106633899787668042 1860)$
c_4	$3(2s+3)^2(5538721056005861462500s^{34}+17000093042867794800 1200s^{33}+25215428706296140524941 82s^{32}+24069033861073900393194 288s^{31}+16611214023947173022412 7950s^{30}+88286198474324819522022 7068s^{29}+37581355594818610758601 90560s^{28}+13155787658258741235890 755800s^{27}+38587134973448786366330 248032s^{26}+96129109345583765886638 321232s^{25}+20544714774638033566476 8754954s^{24}+37944945337592677887102 8692368s^{23}+60877678098043538114517 7931646s^{22}+85125183973189749706760 0541072s^{21}+10391283871123404096276 18422211s^{20}+11073389076832412879108 17574616s^{19}+10282364259109118891102 11542327s^{18}+82865338871955099784133 4599814s^{17}+57567986009921446099133 6767122s^{16}+34111398025087438571519 5498752s^{15}+16960211216096687324467 2554562s^{14}+69019121880524178103951 52092s^{13}+22192275955095081657763 172138s^{12}+55130393244696074306221 01184s^{11}+13085527335820566166857 17799s^{10}+62282242696827293561075 0562s^9+44120821861556436602147 8680s^8+25108662674499791880622 1832s^7+10554272634319796244766 8330s^6+33182712041751038192684 904s^5+78190780360463874447205 41s^4+13533729394126374890003 88s^3+16359337223317228187320 2s^2+12396727299661082512092s+44481379029350672 8368)$
c_3	$6(2s+3)^2(6516142418830425250 00s^{35}+20618119083643318565400s^{34}+31562133499969278615111 6s^{33}+31130651785720476660078 60s^{32}+22229797545183952680520 284s^{31}+12242256173994148159983 5484s^{30}+54084051452874211176177 6912s^{29}+19683840172480631833231 04976s^{28}+60143349163967825993056 97730s^{27}+15642733509059107994947 830402s^{26}+34991063427920339849664 070098s^{25}+67834512242456744868412 053510s^{24}+11461073564163582323268 0558264s^{23}+16942028921941244368224 5006436s^{22}+21963012945790594367645 7475806s^{21}+24991488845891757586328 0485878s^{20}+24945803250943774052524 0777731s^{19}+21792038086810649348151 5756421s^{18}+16586829983929154541016 1133771s^{17}+10919680413162175645624 8426789s^{16}+61455743971044438451107 683232s^{15}+29010820857450160933812 140136s^{14}+11111724769820435809112 639700s^{13}+32309788415718307567809 93930s^{12}+59773407131663003009607 1452s^{11}+19471456710454363005152 664s^{10}+14413823109350224541499 726s^9+96845883777316430719272 36s^8+12560219583039185504039910s^7+66298132211066964095367 42s^6+22675289183166382339644 00s^5+54951085161244719794610 0s^4+95169593716835317546698s^3+11333379076369208961606s^2+83776154550105042714 6s+29118359247617829474)$

c_2	$(2s+3)^2(651614241883042525000s^{36} + 212361287050892314836000s^{35} + 335176828913504517258492s^{34} + 341248027234906031459076 0s^{33} + 251841876218476743042186 12s^{32} + 143532477912633204107111040s^{31} + 657199533413487737746005840s^{30} + 248305780276214198137260 4872s^{29} + 789044425641422120498409 7182s^{28} + 213867066262644389974992575 04s^{27} + 499686266098650320549435327 22s^{26} + 101443944207534124952038283748s^{25} + 180022708498134425424774257358s^{24} + 280473316548653330703764939232s^{23} + 384770188717561290211106882724s^{22} + 465568061409025346140795055268s^{21} + 497070528817890647576496874221s^{20} + 467853455550722338016560291068s^{19} + 387255091639245361342752551670s^{18} + 280658492377779215502417761676s^{17} + 176840580743027422215976620477s^{16} + 957910627865555087240887423 20s^{15} + 437858623301624990522095297 68s^{14} + 163267368535271229917151558 24s^{13} + 460821905008432679074893315 3s^{12} + 763090880285610579007863108s^{11} - 702735952364323683297077 16s^{10} - 998540727043228715373926 04s^{9} - 357260313779697920881889 25s^{8} - 609903789530767014228760 8s^{7} + 385997000711223832356168s^{6} + 571809550541807937530952s^{5} + 184789343789534461150530s^{4} + 354883755696224721692 40s^{3} + 436295085881317044922 8s^{2} + 320602140994390122456s + 10806813741383936712)$
c_1	$2s^2(2s+3)^2(5s+6)^2(137181945659587900 0s^{33} + 427163455705596680 88s^{32} + 643544475178313701908s^{31} + 624756655570258038906 0s^{30} + 439167655652692490166 60s^{29} + 238129437052493452170540s^{28} + 103607422644212518341916 8s^{27} + 371493010638125678667182 4s^{26} + 111877286870725750537637 04s^{25} + 286967082541755572358864 84s^{24} + 633528914065708424429568 78s^{23} + 121330161626918848657558398s^{22} + 202765255131175811905486752s^{21} + 296952156159684842999476188s^{20} + 382193906007924533060066196s^{19} + 432977106373155008061033636s^{18} + 431882942555537334945182376s^{17} + 378922778321739951076319160s^{16} + 291693785851369028780662644s^{15} + 196147224158011892839433394s^{14} + 114410349624644626423212729s^{13} + 572488843324309763964516 27s^{12} + 241306179505037569840510 08s^{11} + 828718908605000322785602 2s^{10} + 215239191645837019591586 7s^{9} + 325184614597411140314601s^{8} - 330079723518789034754 04s^{7} - 417925109386866388973 04s^{6} - 158311859729964497303 58s^{5} - 387700619706111569013 0s^{4} - 665506024092175855680s^{3} - 784198143922099111 20s^{2} - 5759186746951521828s - 200126180395998828)$
c_0	$s^4(2s+3)^2(5s+6)^4(5487277826383516s^{30}+162900207047936448s^{29}+233737714271437309 8s^{28}+ 215881657298975982 96s^{27} + 144210704373637237422s^{26} + 742206852421449807276s^{25} + 306128579816047128 9822s^{24} + 103918956366445860201 12s^{23} + 295883637277356120690 36s^{22} + 716514041081461388970 96s^{21}+149117620073027461547436s^{20}+26880670517895872718 7248s^{19}+ 422185992215286625454736s^{18} + 580191752386498986323712s^{17} + 699694724310231624064272s^{16} + 741757261801656111225984s^{15} + 691666247103583662351612s^{14} + 567033087779716710023352s^{13} + 408047131644656580559296s^{12} + 257036644794565634383008s^{11} + 141145264141826804073576s^{10} + 671784605322888849095 16s^{9} + 274996630579429511410 41s^{8} + 958249127848924285567 2s^{7} + 280336513941982315078 2s^{6} + 675682953313836552876s^{5} + 130659498671205647052s^{4} + 194895630569096544 96s^{3} + 21053054560451509 08s^{2} + 146594486051390088s + 4941387170271576)$

Δ_2	$-36s^4(2s+3)^4(5s+6)^4(1881888621495012747598719782641000 00s^{66}$ $+1171982279367338935485204891772187040 0s^{65}$ $+3590340595158437649151724049466274089 92s^{64}$ $+7212088571266452516322419090023965301952s^{63}$ $+1068391320253687840934482144561140800898 96s^{62}$ $+1244653930745343462400612341123366430556112s^{61}$ $+1187453127258675915704887142527377641964648 0s^{60}$ $+9539717463978084911740021792529086079754307 2s^{59}$ $+6586014211961623946750060924092706557833870 96s^{58}$ $+3968022355733696091765263130542489256493297296s^{57}$ $+2111718215787453698016965546515609174427692038 0s^{56}$ $+1002346955322630643664921552312016335043204078 4s^{55}$ $+4277235454084557392774406702392019604985274451 68s^{54}$ $+1651693204898583110001983831675383703838803111280s^{53}$ $+5803881305128515928064472759889503906134840719788s^{52}$ $+1864526239911173240492717991852870319182966877694 4s^{51}$ $+5498216092397086290715585398715769282805100212946 8s^{50}$ $+1493403271312119095060211916193032794760693070014 80s^{49}$ $+3747389085049257565568650678108067851102723488709 98s^{48}$ $+8709591713633887671254551839384745486844330327339 76s^{47}$ $+1879125423105677523532695113591118653251030543219650s^{46}$ $+3770893425502604495219029177077658520798571516698604s^{45}$ $+7050015221784174490794108388251813073259643663091008s^{44}$ $+1229751697814540088737743012540597239130953523640 4816s^{43}$ $+2003816448416423665928915846433322332297986145636955 8s^{42}$ $+3053220493065830192217996030789798685096097311040106 0s^{41}$ $+4353974016564230774744937671615227473899797195238128 8s^{40}$ $+5814794272234047468845528446358546692682944712469086 4s^{39}$ $+7276560981967024598164222622942133354281616530078647 8s^{38}$ $+8535222627194411740352205447227499061598741207765802 0s^{37}$ $+9386118512358018098652883224139505412412521033278797 8s^{36}$ $+9677367622870745479418594983628077002220070263778913 6s^{35}$ $+9353550534056682983528096959246190247987486973707032 4s^{34}$ $+8472657307033122222401048138408764617479499951727572 0s^{33}$ $+7189292302539646724238556713520429645936760693117107 2s^{32}$ $+5710750538582167076586109442116935829639912622263872 0s^{31}$ $+4242947557936920029498039681532283983986502171148171 2s^{30}$ $+2945340224867131137091897558637195040010087585918442 4s^{29}$ $+1907688741881547559139122534482867333198543394985974 5s^{28}$ $+1150963090261374243005671011302597994636892936359156 0s^{27}$ $+6455314570589262935425947995355398219841083619460275s^{26}$ $+3357432561061020603580382785922585900536396393910862s^{25}$ $+1614483581569691108434974127018929362203153022365980s^{24}$ $+7151843921239467644322741543961403561291367742781 52s^{23}$ $+2905615937078824111549712703038991345504578369200 80s^{22}$ $+1076866147250386418961409060935085408978065543348 84s^{21}$ $+3617512376782787657128716857177209606664184218971 5s^{20}$ $+1093629541362141784014724465858457912720717438935 2s^{19}$ $+2956231446951945885452453177600650222360031507853s^{18}$ $+7143283154950873347674525118298108303417045322 94s^{17}$ $+1578622563973412013157962625289520289452101601 78s^{16}$ $+3466918481540893224348429068849868462291600055 2s^{15}$ $+8792208859688249070338942374434094956929968434s^{14}$ $+2728958913497904679605629304348283067662149156s^{13}$ $+9085144773106795189912395340847764985695491 30s^{12}$ $+2819973846515718244314169254440158181016966 00s^{11}$ $+7659571256132994444113894245418887273849699 2s^{10}$

$$
\begin{aligned}
&+1782323536880591650543354401619541599668158 4s^9\\
&+3522507407940359387775769801382214828420516s^8\\
&+5867820050646080008320434741079193955195 52s^7\\
&+8148452315276322918847755584680364066167 2s^6\\
&+9275689993243887053899463828498929963344s^5\\
&+8438081673219230573799541298816126340 96s^4\\
&+5898808758941212962363346647613729785 6s^3\\
&+2972581287433071568118115850779633072s^2\\
&+9592339120369162877140167215815401 6s\\
&+1483351410366365393372190806569392)
\end{aligned}
$$

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 $p_{\texttt{SGPO}}^{(k+1)} > q_{\texttt{SGPO}}^{(k+1)}$. □