SERPO: Sample-Efficient Recursive Policy Optimization for Large Language Models

Anonymous ACL submission

Abstract

Group Relative Policy Optimization (GRPO) has emerged as a promising paradigm for Reinforcement Learning (RL), yet it incurs substantial computational overhead. This computational burden arises because GRPO necessitates extensive on-policy sampling to acquire stable reward signals for gradient updates. While existing methods typically enhance learning performance through data selection or reward modeling, few approaches address the problem from an algorithmic perspective to improve training efficiency. In this work, we address this methodological limitation by introducing **S**ample-**E**fficient **R**ecursive **P**olicy **O**ptimization (SERPO). By successfully integrating variance-reduced gradients into the GRPO framework, SERPO achieves comparable estimation precision through iterative small-batch updates. We derive convergence bounds for SERPO and provide the first formal sample complexity analysis for both algorithms, theoretically confirming SERPO's improved efficiency over GRPO. Empirically, experiments on LLM benchmarks demonstrate that SERPO matches the alignment quality of GRPO baselines while reducing the sampling cost.

1 Introduction

Reinforcement Learning (RL) has become the standard approach for aligning Large Language Models (LLMs) with human intent (Ouyang et al., 2022) and complex reasoning objectives (Lightman et al., 2023). Historically, Proximal Policy Optimization (PPO) (Schulman et al., 2017) served as the foundational framework, utilizing a parameterized critic model to reduce gradient variance. However, maintaining this auxiliary network imposes a substantial memory overhead, limited scalability in long-context scenarios. To resolve this memory bottleneck, Group Relative Policy Optimization (GRPO) (Shao et al., 2024) has emerged as a critic-free alternative. By estimating advantages via the reward of the sampled group, GRPO eliminates the critic model entirely, liberating significant memory for larger models and longer reasoning chains.

However, this simplified architecture leads to a statistical bottleneck. Since GRPO relies on group sampling rather than a learned critic for advantage estimate, its gradient variance is sensitive to the sample group size. The design of GRPO necessitates extensive sampling to mitigate the variance inherent in group-based advantage estimate, which necessitates heavy on-policy sampling and results in considerable computational overhead (Sun et al., 2025; Roux et al., 2025).

To reduce this sampling burden, prior works have primarily followed two routes to optimize efficiency. From a data-centric perspective, recent works focus on data selection to filter for information-dense samples (Wang et al., 2025; Li et al., 2025). In parallel, reward-centric strategies aim to optimize the reward mechanism, such as DAPO (Yu et al., 2025), RAFT++ (Xiong et al., 2025) and GSPO (Zheng et al., 2025). While these methods do reduce some computational costs by refining data and reward quality, they fail to target the sampling overhead inherent to stabilizing the group baseline. Consequently, the significant computational burden of extensive rollouts still persists.

Since this reliance on extensive rollouts stems from the need to suppress gradient variance, a theoretical solution lies in stochastic variance reduction techniques. These methods trace their origins to convex optimization (Johnson and Zhang, 2013), where they were pioneered to enhance gradient estimate accuracy using recursive estimators. Subsequently, this paradigm was adapted to reinforcement learning (Nguyen et al., 2017a; Papini et al., 2018) to stabilize policy updates. Recent studies have leveraged these estimators to address sample complexity (Xu et al., 2020b; Zhang et al., 2021; Liang and Yang, 2024), decoupling gradient estimation precision from the batch size. However,

applying these techniques to GRPO framework is not straightforward. GRPO's clipping mechanism introduces update bias that triggers exponential variance accumulation, undermining the theoretical foundations of traditional methods. Additionally, iterative updates using biased gradients can lead to detrimental training drift. Consequently, standard recursive estimators are incompatible with these clipped objectives.

To bridge this theoretical gap, we introduce Sample-Efficient Recursive Policy Optimization (SERPO). Methodologically, SERPO adapts the gradient estimator to the non-smooth nature of GRPO. The algorithm operates on a dual-loop architecture: it first computes a high-accuracy anchor gradient derived from a snapshot model, which captures the global landscape geometry. During the inner iterations, SERPO employs a recursive mechanism that recalibrates the gradient estimate by integrating mini-batch information with the stable anchor gradient. This design ensures that the variance reduction remains empirically robust.

Our contributions are summarized as follows:

- **Enabling Variance Reduction for GRPO.** We propose SERPO, a novel framework that adapts recursive gradient estimate to GRPO. To harmonize variance reduction techniques with the non-smooth clipping, SERPO introduces a semi-stochastic estimator that utilizes global gradient snapshot, thereby ensuring theoretical consistency and effectiveness.

- **Theoretical Guarantee and Improved Complexity.** We provide a rigorous convergence analysis accounting for SERPO and GRPO. We prove that SERPO achieves improved sample complexity compared to the standard GRPO, effectively breaking the coupling between variance reduction and sampling cost.

- **Empirical Verification.** We evaluate SERPO on the mathematical and reasoning benchmarks. The results demonstrate that SERPO matches the performance of the GRPO while requiring fewer sample interactions.

2 Preliminaries

2.1 Formulation of GRPO

GRPO eliminates the parameterized value network by computing relative advantages through group normalization. Let π_θ denote the policy parameterized by θ. For a query $q \sim P(Q)$, the algorithm samples a group of G outputs $\{o_j\}_{j=1}^G$ from $\pi_{\theta_{\text{old}}}$. The relative advantage $A_{i,j}$ for output o_j is computed via group normalization:

$$A_{i,j} = \frac{r_j - \mu}{\sigma + \epsilon}, \quad \text{with } r_j \doteq r(q, o_j),$$
$$\mu = \frac{1}{G}\sum_k r_k, \quad \sigma = \sqrt{\frac{1}{G}\sum_k (r_k - \mu)^2}. \quad (1)$$

The optimization objective for the i-th iteration is given by the surrogate objective:

$$\max_\theta \; J_i(\theta) = \mathbb{E}_{q,\{o_j\}\sim\pi_{\theta_{\text{old}}}} \Bigg[\frac{1}{G}\sum_{j=1}^{G} \min\left(\rho_j(\theta) A_{i,j}, \right.$$
$$\left. \text{clip}\left(\rho_j(\theta), 1-\epsilon, 1+\epsilon\right) A_{i,j}\right)\Bigg], \quad (2)$$

where $\rho_j(\theta) = \frac{\pi_\theta(o_j|q)}{\pi_{\theta_{\text{old}}}(o_j|q)}$. The detailed optimization procedure of GRPO is provided in Algorithm 2. Crucially, GRPO requires a large group size G to maintain estimate accuracy, coupling sampling cost directly with optimization stability.

2.2 Recursive Variance Reduction

To mitigate the sample inefficiency of reinforce learning, we consider recursive variance reduction techniques such as SPIDER (Fang et al., 2018) and SARAH (Nguyen et al., 2017b). These methods construct a semi-stochastic gradient estimator $\mathbf{v}_k$ that utilizes historical gradient information:

$$\mathbf{v}_k = \nabla J_{\mathcal{B}_k}(\theta_k) - \nabla J_{\mathcal{B}_k}(\theta_{k-1}) + \mathbf{v}_{k-1}, \quad (3)$$

where $\mathcal{B}_k$ represents the current mini-batch and $\nabla J_{\mathcal{B}_k}(\theta_{k-1})$ serves as a control variate.

However, variance reduction for iterative gradients relies heavily on unbiased gradient estimate. In GRPO, the clipping mechanism in Eq. (2) yields biased gradients. This bias causes the estimation variance to accumulate exponentially rather than linearly during iterative updates. Consequently, standard recursive gradient methods are incompatible with the GRPO optimization objective.

3 Algorithm

We consider the standard reinforcement learning objective to maximize the expected reward:

$$\max_\theta J(\theta) = \mathbb{E}_{q\sim\mathcal{D}, o\sim\pi_\theta}[r(q, o)]. \quad (4)$$

To resolve the tension between sample efficiency and gradient estimation variance, we introduce

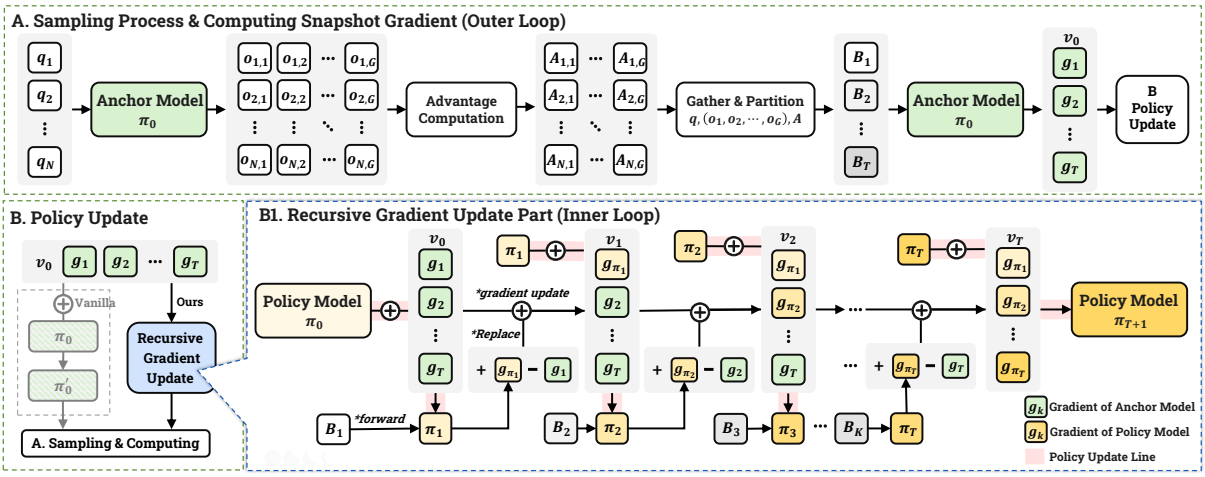

Figure 1: Overview of the optimization process. **Phase A:** The anchor model generates G responses for N queries to construct the trajectory dataset. Following reward computation, these samples are partitioned into mini-batches to calculate the snapshot gradient v_0 by averaging the mini-batch gradients. **Phase B:** While vanilla GRPO utilizes the gradient v_0 for a single update, SERPO leverages mini-batches for iterative updates. First, we initialize π_0 from the snapshot and perform an update using v_0 to obtain π_1. Subsequently, the algorithm enters a recursive loop: the gradient of the current mini-batch $\mathcal{B}_k$ on the updated policy is computed to replace the corresponding anchor gradient g_k to obtain v_k. The policy parameters are updated recursively using these refined gradients until the inner loop concludes, yielding the final model π_{T+1} as the new anchor.

SERPO framework. While traditional methods rely on large batch size to suppress variance, SERPO utilizes a semi-stochastic recursive gradient estimate. The core intuition is to decouple the precision from the batch size. By maintaining a stable global anchor while performing frequent local updates, the algorithm achieves an equilibrium between estimation variance and sampling cost.

As illustrated in Fig.1, the execution flow of SERPO is modeled as a dual-phase iteration. For notational brevity, we suppress the outer iteration index i in the following description. In the Snapshot Phase, the current policy is designated as the anchor model θ_0, from which we sample N outputs to generate a fixed trajectory dataset. This anchor batch is then utilized to compute a high-precision anchor gradient, $v_0 = \nabla J(\theta_0)$. In the Recursive Update Phase, we perform T inner iterations without further sampling. At each step k, we draw a mini-batch $\mathcal{B}_k$ from the anchor batch to construct the semi-stochastic gradient estimator:

$$v_k = \nabla J_{\mathcal{B}_k}(\theta_k) - \nabla J_{\mathcal{B}_k}(\theta_0) + v_{k-1}. \quad (5)$$

By recursively aggregating mini-batch gradients $\nabla J_{\mathcal{B}_k}(\theta_k)$, this estimator v_k is iteratively refined based on the policy state. The model parameters are then updated using the gradient v_k. The detailed procedure is summarized in Algorithm 1.

A structural instability arises when traditional recursive estimators interact with clipping mechanism. Specifically, when the probability ratio exits the trust region, the minibatch gradient $\nabla J_{\mathcal{B}_k}(\theta_k)$ vanishes. Under standard recursive frameworks, the estimator is then biased by the residual negative term $-\nabla J_{\mathcal{B}_k}(\theta_0)$, which misleads the model toward an incorrect optimization trajectory. SERPO circumvents this by retaining the global snapshot gradient v_0, which counterbalances the negative anchor component and maintains alignment with the global optimization.

Finally, to balance efficiency and stability, we adopt a single-pass processing strategy for the inner loop. By partitioning the trajectory dataset, we ensure that specific samples are not repeatedly optimized. This protocol mitigates local overfitting and limits distributional shift to ensure the validity of variance reduction.

4 Theoretical Analysis

4.1 Theoretical Assumptions

To establish the convergence properties of our framework, we adopt the following conditions regarding the policy parameterization, the reward landscape, and the stochastic variance structure.

We begin by formalizing the Lipschitz continuity of the policy gradient.

Assumption 4.1 (Policy Regularity)**.** The output space $\mathcal{O}$ is finite. The policy distribution $\pi_\theta(o|q)$ is

Algorithm 1 Sample-Efficient Recursive Policy Optimization (SERPO)

Input:
Dataset $\mathcal{D}$, Batch size N, Mini-batch size B.

Output: Optimized policy parameters θ_S
Initialize: θ_0
Randomly shuffle $\mathcal{D}$ and partition into $I = |\mathcal{D}|/N$ **batches** $\{\mathcal{L}_1, \ldots, \mathcal{L}_I\}$.
for $i = 0$ to $I - 1$ **do**
  Set anchor parameters: $\theta_{\text{anchor}} = \theta_{i,0}$
  Compute anchor gradient on the batch $\mathcal{L}_i$:

$$v_{i,0} = \nabla J(\theta_{\text{anchor}}) = \frac{1}{N} \sum_{j \in \mathcal{L}_i} \nabla J_j(\theta_{\text{anchor}})$$

  Update policy parameters :

$$\theta_{i,1} = \theta_{i,0} + \eta_{i,0} \cdot v_{i,0}$$

  Partition current large batch $\mathcal{L}_i$ into $T = N/B$ **disjoint** mini-batches $\{\mathcal{B}_1, \ldots, \mathcal{B}_T\}$.
  for $k = 1$ to T **do**
    Compute recursive gradients on mini-batch $\mathcal{B}_k$:

$$g_{\text{curr}} = \sum_{j \in \mathcal{B}_k} \nabla J_j(\theta_{i,k})$$
$$g_{\text{anchor}} = \sum_{j \in \mathcal{B}_k} \nabla J_j(\theta_{\text{anchor}})$$

    Recursively update the gradient:

$$v_{i,k} = v_{i,k-1} + \frac{1}{N} (g_{\text{curr}} - g_{\text{anchor}})$$

    Update policy parameters:

$$\theta_{i,k+1} = \theta_{i,k} + \eta_{i,k} \cdot v_{i,k}$$

  end for
  Update parameters: $\theta_{i+1,0} = \theta_{i,T+1}$
end for
Return $\theta_{I,0}$

smooth with respect to θ. Specifically, its gradient is L-Lipschitz continuous:

$$\|\nabla \pi_{\theta_1}(o|q) - \nabla \pi_{\theta_2}(o|q)\| \leq L\|\theta_1 - \theta_2\|, \forall\, \theta_1, \theta_2, o, q. \tag{6}$$

This condition is a common premise in prior works (Zhang et al., 2020; Xu et al., 2020a; Pang and Jin, 2025), which is weaker than assuming the Lipschitz continuity of the score function $\nabla_\theta \log \pi_\theta$. While the score function often exhibits unbounded gradients in softmax parameterizations, Assumption 4.1 remains valid for general bounded policies, offering a more robust theoretical basis.

Next, we impose a reward regularity condition.

Assumption 4.2 (Bounded Reward)**.** There exists a constant $R > 0$ such that the reward is uniformly bounded: $|r(q, o)| \leq R$ for all (q, o).

This condition is inherently satisfied in practical applications where reward models are typically bounded, and aligns with standard theoretical frameworks (Jin et al., 2024; Pang and Jin, 2025). The triangle inequality directly implies a bound on the advantage estimates:

$$|A(q, o)| \leq |r(q, o)| + |\mu_r| \leq R + R = 2R. \tag{7}$$

Finally, to analyze the variance of gradient estimate , we decompose the stochastic variance of the gradient estimator into two hierarchical levels: uncertainty from the data distribution and uncertainty from the policy's generation.

Assumption 4.3 (Decomposed Bounded Variances)**.** We assume the stochastic gradient variance is bounded by two independent sources: inter-query variance σ^2_{query} across the data distribution and intra-query variance σ^2_{output} across generated responses. Formally, for any query q and data distribution $\mathcal{D}$, we assume:

$$\mathbb{E}_{o \sim \pi_\theta(\cdot|q)} \left[\left\| g(\theta; q, o) - \nabla \tilde{L}_q(\theta) \right\|^2 \right] \leq \sigma^2_{output},$$
$$\mathbb{E}_{q \sim \mathcal{D}} \left[\left\| \nabla \tilde{L}_q(\theta) - \nabla J(\theta) \right\|^2 \right] \leq \sigma^2_{query}. \tag{8}$$

Here, $\nabla \tilde{L}_q(\theta)$ and $\nabla J(\theta)$ represent the query-specific and global gradients respectively. This standard assumption in stochastic optimization (Sun et al., 2025) serves as the foundation for analyzing the estimator's total variance

4.2 Convergence Analysis

In this section, we provide the convergence analysis for the proposed algorithm. Detailed proofs are deferred to Appendix B.

A critical component in our theoretical analysis lies in the accumulation of estimator bias. Due to the clipping mechanism, there is a deviation between the estimated gradient and the true gradient. Therefore, we need to estimate the error caused by the clipping, which is defined as

$\|\mathbb{E}[v_{i,k}] - \nabla J(\theta_{i,0})\|$. We decompose this bias into two distinct terms:

$$\|\mathbb{E}[v_{i,k}] - \nabla J(\theta_{i,0})\| \leq \underbrace{\frac{B}{N}\sum_{l=1}^{T}\|\mathcal{E}_{i,l}^{\text{clip}}\|}_{\text{Clipping Bias}} + \underbrace{\frac{B}{N}\sum_{l=1}^{T}\|\mathcal{E}_{i,l}^{\text{drift}}\|}_{\text{Drift Bias}}, \tag{9}$$

where $\mathcal{E}_{i,l}^{\text{clip}}$ accounts for the gradient suppressed by the trust region clipping, and $\mathcal{E}_{i,l}^{\text{drift}}$ represents the error accumulation due to parameter shifts.

For the drift term $\mathcal{E}_{i,l}^{\text{drift}}$, we invoke the L-smoothness assumption on the policy gradient. By applying the triangle inequality to the recursive update rule and leveraging the shift Bound, we show that the drift is controlled by the step sizes.

To bound the gradient clipping term $\mathcal{E}_{i,l}^{\text{clip}}$, we analyze the probability of the clipping event E_1.

$$E_1 := \left\{1 - \epsilon < \frac{\pi_{\theta_{i,k}}(o|q)}{\pi_{\theta_{i,0}}(o|q)} < 1 + \epsilon\right\}. \tag{10}$$

We utilized the proof strategy proposed by Jin et al. (2024), and construct an auxiliary event $E_{2,\lambda}$.

$$E_{2,\lambda} := \left\{\sqrt{\pi_{\theta_{i,0}}(o_j|q)} > \left(\frac{\sqrt{2L}}{2} + \lambda\right) \left(\sum_{m=0}^{k-1}\eta_{i,m}\right)\mathbb{E}_{\pi_{\theta_{i,0}}}\left(\frac{2R\sqrt{2L(1+\epsilon)}}{\sqrt{\pi_{\theta_{i,0}}(o|q)}}\right)\right\}, \tag{11}$$

By applying the Mean Value Theorem to the likelihood ratio, we establish the subset relationship $\Omega \setminus E_1 \subset \Omega \setminus E_{2,\lambda}$. This inclusion allows us to transition the bias bound from the intractable clipping event to the controllable safety event. Thenthe clip term can be controlled by the step size.

Despite the constant-level differences between the drift and clipping bounds, both exhibit the same order of dependence on the step size η. Furthermore, the upper bound for GRPO's bias is a constant multiple of that of SERPO. Therefore, to maintain notational brevity in the subsequent convergence derivation, we encapsulate these error terms into a unified constant K_{clip}:

$$\|\mathbb{E}[v_{i,k}] - \nabla J(\theta_{i,0})\| \leq K_{\text{clip}}\sum_{m=0}^{k-1}\eta_{i,m}. \tag{12}$$

This uniform bound enables a unified convergence framework for both algorithms.

Since Assumption 4.1 implies that the objective J is $\mathcal{L}$-smooth, we can leverage this regularity to perform a second-order expansion of the objective at the next iteration $\theta_{i+1,0} = \theta_{i,0} + \sum_{k=0}^{T}\eta_{i,k}v_{i,k}$:

$$J(\theta_{i+1,0}) \geq J(\theta_{i,0}) + \left\langle \nabla J(\theta_{i,0}), \sum_{k=0}^{T}\eta_{i,k}v_{i,k}\right\rangle - \frac{\mathcal{L}}{2}\left\|\sum_{k=0}^{T}\eta_{i,k}v_{i,k}\right\|^2. \tag{13}$$

To lower-bound the inner product term, we apply the polarization identity to the interaction between the true gradient and the recursive estimator. By incorporating the established cumulative bias bound, we demonstrate that the linear gain remains dominated by the squared gradient norm, adjusted for the residual drift introduced by the clipping mechanism and encapsulated by K_{clip}.

For the quadratic term, we employ the basic inequality to isolate stochastic noise from estimation error. The stochastic component is bounded via the variance properties of the query and output distributions (Lemma A.2), while the bias is controlled by the clipping constant.

Substituting these bounds back into the Taylor expansion allows for a unified convergence characterization. By imposing the condition $\mathcal{L}\tilde{\eta}_i \leq 1/6$ where $\tilde{\eta}_i = \sum_{k=0}^{T}\eta_{i,k}$ denotes the cumulative step size, we ensure that the positive gradient effectively offsets the accumulated variance and the clipping bias. Rearranging the resulting terms establishes the descent inequality:

$$\mathbb{E}\left[J(\theta_{i+1,0}) - J(\theta_{i,0})\right] \geq \underbrace{\frac{\tilde{\eta}_i}{4}\|\nabla J(\theta_{i,0})\|^2}_{\text{Gradient Term}} - \underbrace{\frac{\tilde{\eta}_i\left(\sigma_{query}^2 + \frac{\sigma_{output}^2}{G}\right)}{4N}}_{\text{Variance Penalty}} - \underbrace{\frac{1}{4}K_{\text{clip}}^2\tilde{\eta}_i^3}_{\text{Bias Term}}. \tag{14}$$

The ascent is discounted by the stochastic variance and the cumulative bias induced by clipping. Telescoping this inequality over i yields the theorem:

Theorem 4.4 (Convergence Guarantee)**.** *Suppose Assumptions 4.1, 4.2, and 4.3 hold. Let $\{\theta_{i,0}\}_{i=0}^{I-1}$ be the sequence of parameters generated by Algorithm 1. Provided that the cumulative step size $\tilde{\eta}_i = \sum_{k=0}^{T}\eta_{i,k}$ satisfies the stability condition $\mathcal{L}\tilde{\eta}_i \leq \frac{1}{6}$, the minimum expected gradient norm*

over the optimization trajectory is bounded by:

$$\min_{0\le i<I} \mathbb{E}\|\nabla J(\theta_{i,0})\|^2 \le \frac{4\Delta J}{\sum_{i=0}^{I-1}\tilde{\eta}_i} + \frac{K_{clip}^2 \sum_{i=0}^{I-1}\tilde{\eta}_i^3}{\sum_{i=0}^{I-1}\tilde{\eta}_i} + \frac{1}{N}\left(\sigma_{query}^2 + \frac{\sigma_{output}^2}{G}\right), \quad (15)$$

where $\Delta J = J(\theta^*) - J(\theta_{0,0})$ represents the initial gap. By choosing the constant step size $\tilde{\eta}^* = \left(\frac{2\Delta J}{IK_{clip}^2}\right)^{\frac{1}{3}}$, the algorithm achieves a convergence rate of $\mathcal{O}(I^{-2/3}) + \mathcal{O}(N^{-1})$:

$$\min_{0\le i<I} \mathbb{E}\|\nabla J(\theta_{i,0})\|^2 \le \frac{3(2\Delta J K_{clip})^{2/3}}{I^{2/3}} + \frac{\sigma_{query}^2}{N} + \frac{\sigma_{output}^2}{NG}. \quad (16)$$

Our results are better than those of Pang and Jin (2025), which achieved $\mathcal{O}(I^{-1/2}) + \mathcal{O}(N^{-1/2})$.

4.3 Sample Complexity Analysis

Based on the result of Theorem 4.4, we analyze the sample complexity $\mathcal{T}$ to achieve an ϵ-stationary point such that $\min_{0\le i<I} \mathbb{E}\|\nabla J(\theta_{i,0})\|^2 \le \epsilon$. Detailed proofs are deferred to Appendix C.

Complexity of SERPO. To control the first optimization gap, the number of outer iterations must scale as $I = \mathcal{O}(\epsilon^{-1.5})$ to bridge the optimization gap. For variance reduction, the anchor batch size must scale as $N = \mathcal{O}(\epsilon^{-1})$. Crucially, SERPO derives its mini-batches B_k by partitioning the anchor batch L_i, thereby eliminating the need for extra trajectory sampling in the inner loop. Aggregating the sampling cost across I batches yields:

$$\mathcal{T}_{\text{SERPO}} = IN = \mathcal{O}(\epsilon^{-2.5}). \quad (17)$$

Complexity of GRPO (Baseline). In contrast, GRPO requires a full large batch ($N = \mathcal{O}(\epsilon^{-1})$) for each of the T inner steps to match SERPO's variance. Given the same outer iterations $I = \mathcal{O}(\epsilon^{-1.5})$, the total sample complexity to achieve the same convergence accuracy scales as:

$$\mathcal{T}_{\text{GRPO}} = ITN = \mathcal{O}(T\epsilon^{-2.5}). \quad (18)$$

This comparison implies that SERPO achieves strictly superior sample efficiency over GRPO whenever the number of inner iterations $T > 1$. In practice, $T = 1$ necessitates a costly resampling step for every gradient update. Such frequent resampling might introduce significant generation latency. Our result improves upon the $\mathcal{O}(\epsilon^{-4})$ implicit in Pang and Jin (2025).

5 Experiments

5.1 Experimental Setup

Primary Evaluation. To assess the empirical performance of our framework, we conduct evaluations on two reasoning benchmarks using Qwen2.5-0.5B/7B-Instruct models: mathematics via GSM8K (Cobbe et al., 2021) and logical reasoning via a standardized 3-character subset of the Knights and Knaves (KK) puzzle (Pan et al., 2023). Our implementation is built upon Logic-RL (Xie et al., 2025), a reinforcement learning framework developed on VERL (Sheng et al., 2025). We compare the proposed SERPO against the GRPO baseline. Following the RLVR framework (Guo et al., 2025), we employ a reward function balancing structural formatting and solution correctness. To ensure a fair comparison, both the standard GRPO and our SERPO uniformly adopt the RLOO estimator (Ahmadian et al., 2024) for unbiased advantage estimation. For the performance metric, inference is conducted via greedy decoding to minimize sampling-induced variance. All reported results represent the mean and standard deviation across three random seeds to ensure statistical reliability.

Exploratory Scaling Analysis. To investigate the generalizability and performance boundaries of our algorithm at a larger scale, we conduct supplementary scaling experiments using the Qwen3-1.7B and Qwen3-8B architectures. These configurations are evaluated on more challenging benchmarks, namely the MATH dataset (Hendrycks et al., 2021) and the competitive AIME25 benchmark (Contributors, 2023). Due to the computational overhead, these scaled-up experiments are conducted via a single trial per configuration. Consequently, we present these scaling results in Appendix D to demonstrate the consistent efficiency trends of our approach at scale.

Detailed hyperparameter and infrastructure configurations are provided in Appendix E.

5.2 Main Results

For a fair comparison, we tuned the learning rate for each method. We report the representative accuracy from a stable, high-performing phase of the validation trajectory, as illustrated in Figure 2.

As shown in Table 1, our algorithm demonstrates a substantial improvement in sample efficiency, achieving high accuracy with a significantly reduced sample budget compared to GRPO. This

Model	Method	Dataset	Accuracy ↑	Queries (K) ↓	Time (min) ↓
Qwen2.5-0.5B-Instruct	GRPO	GSM8K	0.451 ± 0.024	2.64	118.7
Qwen2.5-0.5B-Instruct	**SERPO**	GSM8K	0.451 ± 0.005	**0.96**	**68.6**
Qwen2.5-0.5B-Instruct	GRPO	KK	0.253 ± 0.025	4.16	97.1
Qwen2.5-0.5B-Instruct	**SERPO**	KK	0.253 ± 0.038	**2.08**	**79.3**
Qwen2.5-7B-Instruct	GRPO	GSM8K	0.925 ± 0.005	7.20	293.3
Qwen2.5-7B-Instruct	**SERPO**	GSM8K	0.928 ± 0.004	**2.40**	**89.3**
Qwen2.5-7B-Instruct	GRPO	KK	0.918 ± 0.025	11.76	454.5
Qwen2.5-7B-Instruct	**SERPO**	KK	0.917 ± 0.049	**5.16**	**318.9**

Table 1: **Main Results.** Comparison of peak mean validation accuracy, cumulative sample queries and end-to-end wall-clock training time across different models.

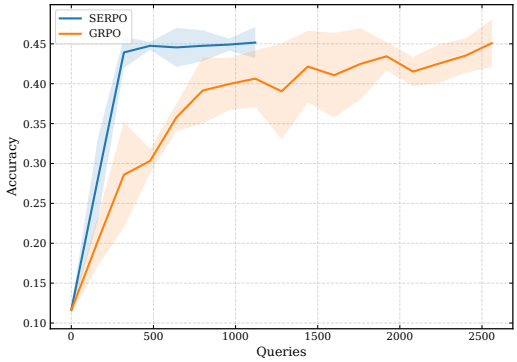

Figure 2: **Validation accuracy dynamics on GSM8K and Qwen2.5-0.5B.** Solid lines denote the mean, and shaded regions represent the standard deviation over three random seeds.

empirical success is theoretically substantiated by Theorem 4.4. The improved sample efficiency also leads to a reduction in total training time.

A natural question arises: is the efficacy of SERPO merely an artifact of simple sample reuse? To investigate this hypothesis, we construct a controlled baseline wherein each sample is utilized for two consecutive policy updates. Empirical evaluations reveal that naive sample reuse severely undermines training stability, frequently culminating in catastrophic policy collapse. Our monitoring indicates that this instability stems from an excessive clip ratio. In naive reuse, as the current policy is updated multiple times against a fixed old policy, importance ratios are more likely to drift beyond the trust region, causing most gradients to be zeroed out by the clipping mechanism. This lack of valid gradient signals eventually triggers policy collapse. In contrast, SERPO's recursive mini-batch updates refine the estimator incrementally, avoiding simultaneous catastrophic clipping and maintaining training stability.

5.3 Ablation Study

All ablation experiments were conducted with all hyperparameters kept identical to those in our main experiments except for the parameter being ablated, and we used the optimal learning rates determined in the main experiments.

Setting	Accuracy ↑	Queries (K) ↓	Time (min) ↓
GSM8K			
$N = 16$	0.924 ± 0.006	**1.92**	119.5
$N = 24$	$\mathbf{0.928 \pm 0.004}$	2.40	89.3
$N = 32$	0.925 ± 0.005	2.40	**79.7**
KK Dataset			
$N = 16$	$\mathbf{0.924 \pm 0.042}$	5.22	386.3
$N = 24$	0.917 ± 0.049	**5.16**	318.9
$N = 32$	0.881 ± 0.033	5.76	**287.8**

Table 2: 7B-Ablation study on Batch Size (N).

Impact of Snapshot Batch Size (N) and Mini-Batch Size (B).

We conduct ablation studies to examine the sensitivity of our framework to the snapshot batch size (N) and mini-batch size (B). As shown in Table 2, increasing N effectively reduces the overall training time. However, its impact on final accuracy highly depends on task complexity. On the relatively simpler GSM8K dataset, the performance remains robust, suggesting that less complex tasks can tolerate aggressive sample reuse. Conversely, on the more challenging KK dataset, a larger N leads to noticeable performance degradation. This decline is an inherent characteristic of the recursive mechanism: larger values of N extend the recursive update chain, thereby amplifying the accumulated estimation bias. Furthermore, Table 3 highlights a standard optimization trade-off regarding the mini-batch size B. While a smaller B yields faster initial convergence, it might compromise training stabil-

ity. A larger B mitigates this noise but slows down the overall learning process. These ablation studies collectively demonstrate that setting the ratio of the snapshot batch size N to the mini-batch size B to approximately 3 yields a robust default configuration, which delivers near-optimal acceleration while preserving comparable final performance.

SETTING	ACCURACY ↑	QUERIES (K) ↓	TIME (MIN) ↓
GSM8K			
$B = 6$	0.917 ± 0.004	**1.32**	**51.2**
$B = 8$	0.928 ± 0.004	2.40	89.3
$B = 12$	$\mathbf{0.930 \pm 0.006}$	4.56	176.8
KK DATASET			
$B = 6$	0.896 ± 0.031	**4.08**	**245.6**
$B = 8$	$\mathbf{0.917 \pm 0.049}$	5.16	318.9
$B = 12$	0.914 ± 0.020	5.64	339.4

Table 3: 7B-Ablation study on Mini-batch Size (B).

METHOD	ACCURACY ↑	QUERIES(K) ↓	TIME(MIN) ↓
GSM8K			
GRPO *w/o* S	0.925 ± 0.005	7.32	293.3
GRPO **+ S**	$\mathbf{0.928 \pm 0.004}$	**2.40**	**89.3**
DAPO *w/o* S	0.922 ± 0.003	6.98	294.2
DAPO **+ S**	$\mathbf{0.925 \pm 0.006}$	**1.92**	**113.7**
KK DATASET			
GRPO *w/o* S	$\mathbf{0.918 \pm 0.025}$	11.76	454.5
GRPO **+ S**	0.917 ± 0.049	**5.16**	**318.9**
DAPO *w/o* S	0.906 ± 0.024	12.96	499.5
DAPO **+ S**	$\mathbf{0.912 \pm 0.029}$	**8.08**	**327.4**

Table 4: Impact of integrating SERPO across DAPO and GRPO on Qwen2.5-7B. Here, **+ S** and **w/o S** denote the configurations with and without SERPO framework, respectively.

Generalization Across RL Paradigms. We further evaluate the modularity of SERPO by applying it to DAPO framework using Qwen2.5-7B-Instruct, with the same hyperparameter configurations as those used in our main experiments. As presented in Table 4, our method consistently improves sample efficiency and speeds up training in the DAPO paradigm. Notably, we observe increased training instability when combining SERPO with GSPO. This phenomenon can be reasonably explained by the distinct design of GSPO. GSPO adopts the sequence-level clipping, which substantially elevates the frequency of clipping operations, as evidenced by the clipping rates reported in its technical report. This higher clip ratio accelerates the accumulation of bias induced by clipping, which contributes to the observed training instability.

5.4 Efficiency-Memory Trade-off.

We systematically profile the supplementary memory overhead of our framework, identifying the retention of additional gradients and intermediate computational variables. This memory acts as an optimization buffer for past gradient states, significantly improving per-sample efficiency. Crucially, whereas simply increasing the batch size suffers from diminishing returns, our framework ensures more effective sample utilization. A detailed quantitative profiling of the peak memory consumption is provided in Appendix E.5.

Notably, the computational demand of rollout generation becomes increasingly pronounced as the response length scales to 4096 tokens. Under this paradigm, our method compresses the iteration frequency through efficient sample reuse, translating into stable efficiency advantages.

Besides, we examine the performance under a strict peak-memory equilibrium on Qwen2.5-0.5B. As reported in Table 5, when memory budgets are perfectly equalized, our method still outperforms GRPO, with a moderated efficiency margin.

METHOD	ACCURACY ↑	QUERIES(K) ↓	TIME(MIN) ↓
GSM8K			
GRPO	0.240 ± 0.010	2.88	41.2
SERPO	0.240 ± 0.056	**1.28**	**36.3**
KK DATASET			
GRPO	0.447 ± 0.015	1.92	74.2
SREPO	0.447 ± 0.005	**0.48**	**28.9**

Table 5: Ablation under Fixed Memory Constraints.

6 Conclusion

This work addresses the prohibitive sampling cost inherent to GRPO. To overcome this, we propose Sample Efficient Recursive Policy Optimization (SERPO), which leverages recursive estimate to maintain estimation accuracy without massive sampling. Theoretically and empirically, we demonstrate that SERPO achieves better data efficiency compared to GRPO baselines. Consequently, our framework achieves comparable performance with a fraction of the sample budget, establishing a data-efficient paradigm for LLM alignment.

Limitations

Bounded Efficacy. Our framework introduces additional reasoning and memory usage overhead, specifically during the backward propagation phase. Although these expenses are well-amortized when the sequence length is large and the generation serves as the primary consumption stage, we recognize that the efficiency gains might diminish in hardware-constrained scenarios where model gradients occupy the majority of the device memory.

Hyperparameter Complexity. Our framework introduces additional hyperparameters, specifically the batch size and mini-batch size, which increase the complexity of the optimization setup compared to standard GRPO. Although we identified a robust default ratio that yields near-optimal convergence in our experiments, we recognize that the optimal configuration may vary across different tasks or model scales.

Idealized Assumption. Our convergence analysis relies on the Lipschitz smoothness of the policy (Assumption 4.1). While this is a standard prerequisite in optimization theory, it remains an idealized condition that may not strictly hold throughout the entire LLM training landscape. The inherent difficulty of empirically verifying such regularity remains a limitation.

Ethical Considerations

While our work focuses on improving algorithmic efficiency, we acknowledge that LLM alignment inherently carries ethical risks. The effectiveness of SERPO depends on the underlying reward signals; thus, any biases present in the reward models or human feedback could be efficiently propagated into the policy. Furthermore, although efficiency reduces the barrier for safety research, it also poses a dual-use risk where it could be utilized to align models with harmful objectives more rapidly. We encourage practitioners to use robust reward modeling and safety evaluation frameworks alongside our algorithm.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. 2024. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12248–12267. ACL.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. 2018. Spider: near-optimal non-convex optimization via stochastic path integrated differential estimator. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 687–697. Curran Associates Inc.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, and 1 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks , virtual*.

Ruinan Jin, Shuai Li, and Baoxiang Wang. 2024. On stationary point convergence of PPO-clip. In *The Twelfth International Conference on Learning Representations*.

Rie Johnson and Tong Zhang. 2013. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems.*, pages 315–323.

Xuefeng Li, Haoyang Zou, and Pengfei Liu. 2025. Limr: Less is more for rl scaling. *arXiv:2502.11886*.

Ling Liang and Haizhao Yang. 2024. On the stochastic (variance-reduced) proximal gradient method for regularized expected reward optimization. *Trans. Mach. Learn. Res.*

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *Preprint*, arXiv:2305.20050.

Lam M. Nguyen, Jie Liu, Katya Scheinberg, and Martin Takác. 2017a. Stochastic recursive gradient algorithm for nonconvex optimization. *CoRR*, arXiv:1705.07261.

Lam M. Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. 2017b. Sarah: a novel method for machine learning problems using stochastic recursive gradient. In *Proceedings of the 34th International Conference on Machine Learning*, ICML'17, page 2613–2621. JMLR.org.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, and 1 others. 2022. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*. Curran Associates Inc.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. 2023. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3806–3824.

Lei Pang and Ruinan Jin. 2025. On the theory and practice of grpo: A trajectory-corrected approach with fast convergence. *Preprint*, arXiv:2508.02833.

Matteo Papini, Damiano Binaghi, Giuseppe Canonaco, Matteo Pirotta, and Marcello Restelli. 2018. Stochastic variance-reduced policy gradient. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4023–4032. PMLR.

Nicolas Le Roux, Marc G Bellemare, Jonathan Lebensold, Arnaud Bergeron, Joshua Greaves, Alexandre Fréchette, Carolyne Pelletier, Eric Thibodeau-Laufer, Sándor Tóth, and Sam Work. 2025. Tapered off-policy REINFORCE - stable and efficient reinforcement learning for large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, page 1279–1297. ACM.

Yifan Sun, Jingyan Shen, Yibin Wang, Tianyu Chen, Zhendong Wang, Mingyuan Zhou, and Huan Zhang. 2025. Improving data efficiency for LLM reinforcement fine-tuning through difficulty-targeted online data selection and rollout replay. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Jiachen T. Wang, Prateek Mittal, Dawn Song, and Ruoxi Jia. 2025. Data shapley in one training run. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. 2025. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning. *Preprint*, arXiv:2502.14768.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, and Hanze Dong. 2025. A minimalist approach to LLM reasoning: from rejection sampling to reinforce. *CoRR*, arXiv:2504.11343.

Pan Xu, Felicia Gao, and Quanquan Gu. 2020a. An improved convergence analysis of stochastic variance-reduced policy gradient. In *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, volume 115 of *Proceedings of Machine Learning Research*, pages 541–551. PMLR.

Pan Xu, Felicia Gao, and Quanquan Gu. 2020b. Sample efficient policy gradient methods with recursive variance reduction. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *Preprint*, arXiv:2503.14476.

Junyu Zhang, Chengzhuo Ni, Zheng Yu, Csaba Szepesvári, and Mengdi Wang. 2021. On the convergence and sample efficiency of variance-reduced policy gradient method. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021*, pages 2228–2240.

Kaiqing Zhang, Alec Koppel, Hao Zhu, and Tamer Başar. 2020. Global convergence of policy gradient methods to (almost) locally optimal policies. *SIAM J. Control Optim.*, 58(6):3586–3612.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. 2025. Group sequence policy optimization. *Preprint*, arXiv:2507.18071.

A Lemmas

Lemma A.1. *lemma A1 of Jin et al. (2024). If Assumption 1 holds, we have*

$$\|\nabla \pi_\theta(o|q)\| \leq \sqrt{2L \cdot \pi_\theta(o|q)} \quad (\forall q, o). \tag{19}$$

Lemma A.2. *(Variance Decomposition). If Assumption 4.3 holds, the anchor gradient $v_{i,0}$ in Algorithm 1 acts as an unbiased estimator of the global objective $J(\theta)$. Its variance is bounded by the superposition of the query distribution variance and the output response variance, scaled by their respective batch sizes. i.e.,*

$$\mathbb{E}[v(\theta_{i,0})] = \nabla J(\theta_{i,0}), ; \mathbb{E}\Big[\big|v(\theta_{i,0}) - \nabla J(\theta_{i,0})\big|^2\Big] \leq \frac{1}{N}\left(\sigma^2_{query} + \frac{\sigma^2_{output}}{G}\right). \tag{20}$$

Proof. First, we establish the unbiasedness. For a specific query , let $\mathbb{E}_{\mathbf{o}|q}[g(\theta; q)]$ denote the gradient estimator computed within groups. For RLOO advantage, the conditional expectation satisfies:

$$\mathbb{E}_{\mathbf{o}|q}[g(\theta; q)] = \mathbb{E}_{\mathbf{o}\sim\pi(\cdot|q)}\left[\frac{1}{G}\sum_{j=1}^{G}\nabla_\theta \log \pi_\theta(o_j \mid q)\left(r_j - \frac{1}{G-1}\sum_{i\neq j} r_i\right)\right] \tag{21}$$

$$= \frac{1}{G}\sum_{j=1}^{G}\mathbb{E}_{o_j|q}\left[\nabla_\theta \log \pi_\theta(o_j \mid q) r_j\right] - \underbrace{\frac{1}{G(G-1)}\sum_{j=1}^{G}\sum_{i\neq j}\mathbb{E}_{\mathbf{o}|q}\left[\nabla_\theta \log \pi_\theta(o_j \mid q) r_i\right]}_{=0} \tag{22}$$

$$= \nabla \tilde{L}_q(\theta). \tag{23}$$

The second equality holds because $\mathbb{E}[\nabla_\theta \log \pi_\theta(o_j \mid q)] = 0$ and responses o_i, o_j are conditionally independent given q.

For normalized advantage, the conditional expectation aligns with the previous gradient, up to a scaling factor:

$$\mathbb{E}_{\mathbf{o}|q}[g(\theta; q)] = \mathbb{E}_{\mathbf{o}\sim\pi(\cdot|q)}\left[\frac{1}{\sigma G}\sum_{j=1}^{G}\nabla_\theta \log \pi_\theta(o_j \mid q)\left(r_j - \frac{1}{G}\sum_{i} r_i\right)\right] \tag{24}$$

$$= \frac{1}{\sigma G}\sum_{j=1}^{G}\mathbb{E}_{o_j|q}\left[\nabla_\theta \log \pi_\theta(o_j \mid q) r_j\right] - \underbrace{\frac{1}{\sigma G^2}\sum_{j=1}^{G}\mathbb{E}_{\mathbf{o}|q}\left[\nabla_\theta \log \pi_\theta(o_j \mid q) r_j\right]}_{=0} \tag{25}$$

$$= \frac{G-1}{\sigma G}\nabla \tilde{L}_q(\theta). \tag{26}$$

Applying the law of total expectation over the query distribution $\mathcal{D}$:

$$\mathbb{E}[v(\theta_{i,0})] = \mathbb{E}_{q\sim\mathcal{D}}\left[\frac{1}{N}\sum_{j=1}^{N}\mathbb{E}_{\mathbf{o}|q_j}[g(\theta_{i,0}; q_j)]\right] = \mathbb{E}_q[\nabla \tilde{L}_q(\theta_{i,0})] = \nabla J(\theta_{i,0}). \tag{27}$$

Further, to analyze the variance, we leverage the independence of queries within the anchor batch. The variance of the averaged estimator scales linearly with the inverse of the batch size N:

$$\mathbb{E}\left[\|v(\theta_{i,0}) - \nabla J(\theta_{i,0})\|^2\right] = \frac{1}{N}\mathrm{Var}(g(\theta_{i,0}; q)). \tag{28}$$

By the Law of Total Variance, we decompose the total variance of a single query gradient estimator $g(\theta_{i,0}; q)$ into inter-query and intra-query components:

$$\mathrm{Var}(g(\theta_{i,0}; q)) = \underbrace{\mathrm{Var}_q\left(\mathbb{E}_{\mathbf{o}|q}[g(\theta_{i,0}; q)]\right)}_{\text{Inter-query Variance}} + \underbrace{\mathbb{E}_q\left[\mathrm{Var}_{\mathbf{o}|q}(g(\theta_{i,0}; q))\right]}_{\text{Intra-query Variance}}. \tag{29}$$

Algorithm 2 Group Relative Policy Optimization (GRPO)

Input:
Full dataset $\mathcal{D}$, batch size N.
Output: Optimized policy parameters $\theta_{S,1}$
Initialize: $\theta_{0,1}$
for $i = 0$ to $I - 1$ **do**
  Randomly shuffle dataset $\mathcal{D}$ and partition into I mini-batches $\{\mathcal{B}_1, \ldots, \mathcal{B}_I\}$.
  For each query, sample G outputs $\{o_j\}_{j=1}^{G} \sim \pi_{\theta_{i,k}}(\cdot|q)$.
  for $i = 1$ to T **do**
    Compute gradients on the mini-batch $\mathcal{B}_i$:

$$g_{i,k} = \frac{1}{N} \sum_{j \in \mathcal{B}_k} \nabla J_j(\theta_{i,k})$$

    Update policy parameters:

$$\theta_{i,k+1} = \theta_{i,k} + \eta_{i,k} \cdot g_{i,k}$$

  end for
  Update outer loop parameters: $\theta_{i+1,1} = \theta_{i,T+1}$
end for
Return $\theta_{I,0}$

From the unbiasedness result established above, we have $\mathbb{E}_{\mathbf{o}|q}[g(\theta_{i,0}; q)] = \nabla \tilde{L}_q(\theta_{i,0})$. Thus, the first term corresponds to the variance of the true gradients across the query distribution, which is bounded by σ^2_{query} according to Assumption 4.3:

$$\mathrm{Var}_q\left(\nabla \tilde{L}_q(\theta_{i,0})\right) \leq \sigma^2_{query}. \tag{30}$$

For the second term, utilizing the conditional independence of responses within the group (as shown in the standard GRPO derivation), the variance of the group estimator reduces the sample variance by a factor of G:

$$\mathbb{E}_q\left[\mathrm{Var}_{\mathbf{o}|q}(g(\theta_{i,0}; q))\right] \leq \frac{\sigma^2_{output}}{G}. \tag{31}$$

Combining these results, we obtain the final variance bound:

$$\mathbb{E}\left[\|v(\theta_{i,0}) - \nabla J(\theta_{i,0})\|^2\right] \leq \frac{1}{N}\left(\sigma^2_{query} + \frac{\sigma^2_{output}}{G}\right). \tag{32}$$

Notice that the constant terms and scaling factors can be absorbed into the bounds σ^2_{query} and σ^2_{output} for notational convenience in subsequent convergence analyses. □

To estimate the upper bound of the GRPO gradient norm, we first establish a rigorous mathematical definition of the algorithm.

Lemma A.3. *(Parameter Shift Bound for GRPO). Under Assumptions 4.1 and 4.2, for the estimator $g_{i,k}$ in Algorithm 2, the parameter shift at iteration k is bounded by:*

$$\|g_{i,k}\| \leq 2R \cdot \mathbb{E}_{\pi_{\theta_{i,0}}}\left(\frac{\sqrt{2L(1+\epsilon)}}{\sqrt{\pi_{\theta_{i,0}}(o|q)}}\right). \tag{33}$$

Proof. Let N be the anchor batch size. We define the clipping event for the k-th inner iteration as:

$$E_1^{i,j,k} := \left\{1 - \epsilon < \frac{\pi_{\theta_{i,k}}(o_j|q)}{\pi_{\theta_{i,0}}(o_j|q)} < 1 + \epsilon\right\}. \tag{34}$$

and its indicator function as $\mathbf{1}_{E_1^{i,j,k}}$. The estimator $g_{i,k}$ can be formulated as:

$$\|g_{i,k}\| = \frac{1}{N}\sum_{l=1}^{N}\left\|\mathbb{E}_{\pi_{\theta_{i,0}}}\left(\frac{1}{G}\sum_{j=1}^{G}\mathbf{1}_{E_1^{i,j,k}}\frac{\nabla\pi_{\theta_{i,k}}(o_j|q_l)}{\pi_{\theta_{i,1}}(o_j|q_l)}A_{i,j}\right)\right\| \tag{35}$$

$$\leq \frac{1}{N}\sum_{l=1}^{N}\mathbb{E}_{\pi_{\theta_{i,0}}}\left(\mathbf{1}_{E_{i,k}^{(o,q_l)}}\frac{\|\nabla\pi_{\theta_{i,k}}(o|q_l)\|}{\pi_{\theta_{i,0}}(o|q_l)}|A_{1,j}|\right). \tag{36}$$

Assumption 4.2 provides an upper bound for $|A_{1,j}|$. By Lemma 1

$$\|\nabla\pi_{\theta_{i,k}}(o|q)\| \leq \sqrt{2L\pi_{\theta_{i,k}}(o|q)}. \tag{37}$$

Combining the above inequalities, we obtain the Lemma 3. □

Lemma A.4. *(Parameter Shift Bound for SERPO). Under Assumptions 4.1 and 4.2, for the variance-reduced estimator $v_{i,k}$ in Algorithm 1, the parameter shift at iteration k is bounded by:*

$$\|v_{i,k}\| \leq 2R\cdot\mathbb{E}_{\pi_{\theta_{i,0}}}\left(\frac{\sqrt{2L(1+\epsilon)}}{\sqrt{\pi_{\theta_{i,0}}(o|q)}}\right). \tag{38}$$

Proof. We analyze the variance-reduced estimator $v_{i,k}$. Based on the recursive update rule $v_{i,k} = v_{i,k-1} + \frac{1}{N}(g_{\text{curr}} - g_{\text{anchor}})$, the estimator at step k can be explicitly decomposed into two components: the gradients from the anchor policy for unvisited samples, and the accumulated gradients from the current policies for visited mini-batches.

Using the notation $\mathbf{1}_{E_1^{i,j,k}}$. The estimator $v_{i,k}$ can be expanded as:

$$\|v_{i,k}\| = \left\|\mathbb{E}_{\pi_{\theta_{i,0}}}\left(\underbrace{\frac{1}{N}\sum_{j=Bk+1}^{N}\nabla J_j(\theta_{i,0})}_{\text{Unvisited Samples (Anchor)}} + \underbrace{\frac{1}{N}\sum_{l=1}^{k}\sum_{j\in\mathcal{B}_l}\mathbf{1}_{E_1^{i,j,k}}\nabla J_j(\theta_{i,t})}_{\text{Visited Samples (Updated)}}\right)\right\| \tag{39}$$

$$\leq \frac{1}{N}\sum_{j=Bk+1}^{N}\left\|\mathbb{E}_{\pi_{\theta_{i,0}}}\left(\frac{\nabla\pi_{\theta_{i,0}}(o|q)}{\pi_{\theta_{i,0}}(o|q)}A_{i,j}\right)\right\|$$

$$+ \frac{1}{N}\sum_{l=1}^{k}\sum_{j\in\mathcal{B}_l}\left\|\mathbb{E}_{\pi_{\theta_{i,0}}}\left(\mathbf{1}_{E_1^{i,j,k}}\frac{\nabla\pi_{\theta_{i,t}}(o|q)}{\pi_{\theta_{i,0}}(o|q)}A_{i,j}\right)\right\|. \tag{40}$$

Applying Assumption 4.2 which bounds the advantage $|A_{i,j}| \leq 2R$, and Lemma 1 which provides the gradient bound $\|\nabla\pi_\theta\| \leq \sqrt{2L\pi_\theta}$, we analyze the terms separately.

For the unvisited samples (Anchor term), we have $\pi_\theta = \pi_{\theta_{i,0}}$:

$$\frac{\|\nabla\pi_{\theta_{i,0}}(o|q)\|}{\pi_{\theta_{i,0}}(o|q)} \leq \frac{\sqrt{2L\pi_{\theta_{i,0}}(o|q)}}{\pi_{\theta_{i,0}}(o|q)} = \frac{\sqrt{2L}}{\sqrt{\pi_{\theta_{i,0}}(o|q)}}. \tag{41}$$

For the visited samples, under the event $E_1^{i,j,k}$, we have $\pi_{\theta_{i,k}} \leq (1+\epsilon)\pi_{\theta_{i,0}}$. Thus:

$$\frac{\|\nabla\pi_{\theta_{i,t}}(o|q)\|}{\pi_{\theta_{i,0}}(o|q)} \leq \frac{\sqrt{2L\pi_{\theta_{i,t}}(o|q)}}{\pi_{\theta_{i,0}}(o|q)} \leq \frac{\sqrt{2L(1+\epsilon)\pi_{\theta_{i,0}}(o|q)}}{\pi_{\theta_{i,0}}(o|q)} = \frac{\sqrt{2L(1+\epsilon)}}{\sqrt{\pi_{\theta_{i,0}}(o|q)}}. \tag{42}$$

Substituting these bounds back into the norm inequality:

$$\|v_{i,k}\| \leq \frac{N - Bk}{N} \mathbb{E}_{\pi_{\theta_{i,0}}} \left(\frac{\sqrt{2L}}{\sqrt{\pi_{\theta_{i,0}}(o|q)}} \cdot 2R \right) + \frac{Bk}{N} \mathbb{E}_{\pi_{\theta_{i,0}}} \left(\frac{\sqrt{2L(1+\epsilon)}}{\sqrt{\pi_{\theta_{i,0}}(o|q)}} \cdot 2R \right). \tag{43}$$

Since $\sqrt{2L} < \sqrt{2L(1+\epsilon)}$, the convex combination is upper-bounded by the larger term:

$$\|v_{i,k}\| \leq 2R \mathbb{E}_{\pi_{\theta_{i,0}}} \left(\frac{\sqrt{2L(1+\epsilon)}}{\sqrt{\pi_{\theta_{i,0}}(o|q)}} \right). \tag{44}$$

This completes the proof. □

B Proof of Theorem 4.4.

We start from the -smoothness of the objective function . By invoking the $\mathcal{L}$-smoothness property of the objective function J (Assumption 4.1), we establish a quadratic lower bound on the objective improvement between consecutive batches. Substituting the cumulative parameter update $\theta_{i+1,0} - \theta_{i,0} = \sum_{k=0}^{T} \eta_{i,k} v_{i,k}$ derived from the inner-loop aggregation yields:

$$\begin{aligned} J(\theta_{i+1,0}) &\geq J(\theta_{i,0}) + \langle \nabla J(\theta_{i,0}), \theta_{i+1,0} - \theta_{i,0} \rangle - \frac{\mathcal{L}}{2} \|\theta_{i+1,0} - \theta_{i,0}\|^2 \\ &= J(\theta_{i,0}) + \left\langle \nabla J(\theta_{i,0}), \sum_{k=0}^{T} \eta_{i,k} v_{i,k} \right\rangle - \frac{\mathcal{L}}{2} \left\| \sum_{k=0}^{T} \eta_{i,k} v_{i,k} \right\|^2. \end{aligned} \tag{45}$$

For GRPO and SERPO, we can have the following forms of gradient :

$$\begin{aligned} \mathbb{E}_{\pi_{\theta_{i,0}}} (v_{i,k} - \nabla J(\theta_{i,0})) &= \frac{1}{N} \sum_{j=1}^{B} \sum_{l=1}^{T} \left(\mathbb{I}_{E_1^{i,j,k}} \nabla Jj(\theta_{i,l}) - \mathbb{I}_{E_1^{i,j,k}} \nabla Jj(\theta_{i,0}) - \mathbb{I}_{B_l / E_1^{i,j,l}} \nabla Jj(\theta_{i,0}) \right) \\ &= \frac{B}{N} \sum_{l=1}^{T} \mathbb{E}_{\pi_{\theta_{i,0}}} \left(\mathbb{I}_{E_1^{i,j,k}} \frac{\nabla \pi \theta_{i,l}(o|q) - \nabla \pi_{\theta_{i,0}}(o|q)}{\pi_{\theta_{i,0}}(o|q)} \cdot A(q,o) \right) \\ &\quad - \frac{B}{N} \sum_{l=1}^{T} \mathbb{E}_{\pi_{\theta_{i,0}}} \left(\mathbb{I}_{B_l / E_1^{i,j,l}} \frac{\nabla \pi_{\theta_{i,0}}(o|q)}{\pi_{\theta_{i,0}}(o|q)} \cdot A(q,o) \right) \\ &\triangleq \frac{B}{N} \sum_{l=1}^{T} \mathcal{E}_{i,l}^{\text{drift}} - \frac{B}{N} \sum_{l=1}^{T} \mathcal{E}_{i,l}^{\text{clip}}. \end{aligned} \tag{46}$$

The Anchor Boundary Term $\mathcal{E}_{i,l}^{\text{drift}}$ captures the anchor gradient mass excluded by the clipping constraint. This term quantifies the specific bias introduced by the non-linear clipping operator acting on the anchor samples.

We invoke the L-smoothness property of the policy π (Assumption 4.1), which implies that the policy gradient $\nabla \pi$ satisfies the L-Lipschitz continuity condition with respect to the parameters θ:

$$\|\nabla \pi_{\theta_{i,l}}(o|q) - \nabla \pi_{\theta_{i,0}}(o|q)\| \leq L \cdot \|\theta_{i,l} - \theta_{i,0}\|. \tag{47}$$

Applying the norm inequality to the term $\mathcal{E}_{i,l}^{\text{drift}}$ and substituting the Lipschitz condition derived above, we obtain the following upper bound:

$$\begin{aligned}\|\mathcal{E}_{i,l}^{\text{drift}}\| &\leq \mathbb{E}_{\pi_{\theta_{i,0}}}\left(\mathbf{I}_{E_1^{i,j,l}} \cdot \frac{\|\nabla\pi_{\theta_{i,l}}(o|q) - \nabla\pi_{\theta_{i,0}}(o|q)\|}{\pi_{\theta_{i,0}}(o|q)} \cdot |A(q,o)|\right) \\ &\leq \mathbb{E}_{\pi_{\theta_{i,0}}}\left(\mathbf{I}_{E_1^{i,j,l}} \cdot \frac{L \cdot \|\theta_{i,l} - \theta_{i,0}\|}{\pi_{\theta_{i,0}}(o|q)} \cdot |A(q,o)|\right). \end{aligned} \tag{48}$$

Next, we seek to bound the cumulative parameter drift,$\|\theta_{i,l} - \theta_{i,0}\|$. By applying the triangle inequality to the recursive sequence of updates, this drift is bounded by the summation of single-step deviations:

$$\|\theta_{i,l} - \theta_{i,0}\| = \left\|\sum_{k=0}^{l-1}(\theta_{i,k+1} - \theta_{i,k})\right\| \leq \sum_{k=0}^{l-1}\|\theta_{i,k+1} - \theta_{i,k}\|. \tag{49}$$

For notational brevity and to facilitate subsequent analysis, we define the constant C_{step} to represent the step-size-independent upper bound derived in Lemma 4:

$$C_{\text{step}} := \mathbb{E}_{\pi_{\theta_{i,0}}}\left(2\frac{\sqrt{2L(1+\epsilon)}}{\sqrt{\pi_{\theta_{i,0}}(o_j|q)}}R\right). \tag{50}$$

Consequently, the cumulative parameter drift is uniformly bounded by the accumulated step sizes:

$$\|\theta_{i,l} - \theta_{i,0}\| \leq \sum_{m=0}^{l-1}\eta_{i,m}C_{\text{step}}. \tag{51}$$

Substituting this upper bound back into the inequality for $\|\mathcal{E}_{i,l}^{\text{drift}}\|$ yields the final bound for the boundary term:

$$\begin{aligned}\|\mathcal{E}_{i,l}^{\text{drift}}\| &\leq \mathbb{E}_{\pi_{\theta_{i,0}}}\left(\mathbf{I}_{E_1^{i,j,l}} \cdot \frac{L\left(\sum_{m=0}^{l-1}\eta_{i,m}C_{\text{step}}\right)}{\pi_{\theta_{i,0}}(o|q)} \cdot |A(q,o)|\right) \\ &\leq \mathbb{E}_{\pi_{\theta_{i,0}}}\left(\frac{L\left(\sum_{m=0}^{l-1}\eta_{i,m}C_{\text{step}}\right)}{\pi_{\theta_{i,0}}(o|q)} \cdot |A(q,o)|\right) \\ &\leq 2LR|\mathcal{O}|\left(\sum_{m=0}^{l-1}\eta_{i,m}C_{\text{step}}\right). \end{aligned} \tag{52}$$

Here, the final inequality follows from the bounded advantage assumption ($|A| \leq 2R$) and the property of the expectation over the finite output space $\mathcal{O}$, where $\mathbb{E}_{\pi_{\theta_{i,0}}}[1/\pi_{\theta_{i,0}}(o|q)] = |\mathcal{O}|$.

The Continuity-Controlled Term $\mathcal{E}_{i,l}^{\text{clip}}$ represents the recursive gradient correction within the trust region. Governed by the Lipschitz continuity of the policy gradient, this term diminishes as $\theta_{i,l}$ approaches the anchor $\theta_{i,0}$, thereby preserving the variance reduction property.

We invoke the Mean Value Theorem to estimate the likelihood ratio between $\pi_{\theta_{i,k}}$ and $\pi_{\theta_{i,0}}$:

$$\frac{\pi_{\theta_{i,k}}(o|q)}{\pi_{\theta_{i,0}}(o|q)} = \exp\left(\ln\pi_{\theta_{i,k}}(o|q) - \ln\pi_{\theta_{i,0}}(o|q)\right) = \exp\left(\frac{\nabla_\theta\pi_{\tilde{\theta}_{i,k}}(o|q)}{\pi_{\tilde{\theta}_{i,k}}(o|q)}\right)^\top(\theta_{i,k} - \theta_{i,0}), \tag{53}$$

where $\tilde{\theta}_{i,k}$ lies on the line segment connecting $\theta_{i,k}$ and $\theta_{i,0}$. Utilizing Lemma 1 (Lipschitz continuity of the policy gradient), which implies $\|\nabla\pi\| \leq \sqrt{2L\pi}$, we obtain the following bounds for the ratio:

$$\frac{\pi_{\theta_{i,k}}(o|q)}{\pi_{\theta_{i,0}}(o|q)} \in \left[\exp\left\{-\frac{\sqrt{2L}}{\sqrt{\pi_{\tilde{\theta}_{i,k}}(o|q)}}\|\theta_{i,k} - \theta_{i,0}\|\right\}, \quad \exp\left\{\frac{\sqrt{2L}}{\sqrt{\pi_{\tilde{\theta}_{i,k}}(o|q)}}\|\theta_{i,k} - \theta_{i,0}\|\right\}\right]. \tag{54}$$

To facilitate the analysis of the clipping event $E_1^{i,j,k}$, we construct an auxiliary event $E_{2,\lambda}^{i,j,k}$ based on the probability mass of the anchor policy. Using the parameter shift upper bound derived in Lemma 4, we define:

$$E_{2,\lambda}^{i,j,k} := \left\{ \sqrt{\pi_{\theta_{i,0}}(o_j|q)} > \left(\frac{\sqrt{2L}}{2} + \lambda \right) \left(\sum_{m=0}^{k-1} \eta_{i,m} \right) \mathbb{E}_{\pi_{\theta_{i,0}}} \left(\frac{2R\sqrt{2L(1+\epsilon)}}{\sqrt{\pi_{\theta_{i,0}}(o|q)}} \right) \right\}, \tag{55}$$

where $\lambda > 0$ is a control coefficient to be determined.

Next, we inspect the properties of the square-root density function $\sqrt{\pi_\theta(o|q)}$. Its gradient norm is bounded by:

$$\left\| \nabla \sqrt{\pi_\theta(o|q)} \right\| = \frac{\|\nabla \pi_\theta(o|q)\|}{2\sqrt{\pi_\theta(o|q)}} \leq \frac{\sqrt{2L}}{2}. \tag{56}$$

This implies that $\sqrt{\pi_\theta}$ is $\frac{\sqrt{2L}}{2}$-Lipschitz. Consequently, for the intermediate point $\tilde{\theta}_{i,k}$:

$$\left| \sqrt{\pi_{\tilde{\theta}_{i,k}}(o|q)} - \sqrt{\pi_{\theta_{i,0}}(o|q)} \right| \leq \frac{\sqrt{2L}}{2} \|\tilde{\theta}_{i,k} - \theta_{i,0}\| \leq \frac{\sqrt{2L}}{2} \|\theta_{i,k} - \theta_{i,0}\|. \tag{57}$$

Conditioned on the occurrence of the safety event $E_{2,\lambda}^{i,j,k}$, we can lower-bound the density at the intermediate point:

$$\begin{aligned}
\sqrt{\pi_{\tilde{\theta}_{i,k}}(o|q)} &\geq \sqrt{\pi_{\theta_{i,0}}(o|q)} - \left\| \sqrt{\pi_{\tilde{\theta}_{i,k}}(o|q)} - \sqrt{\pi_{\theta_{i,0}}(o|q)} \right\| \\
&\geq \sqrt{\pi_{\theta_{i,0}}(o|q)} - \frac{\sqrt{2L}}{2} \|\theta_{i,k} - \theta_{i,0}\| \\
&> \left(\frac{\sqrt{2L}}{2} + \lambda \right) \|\theta_{i,k} - \theta_{i,0}\| - \frac{\sqrt{2L}}{2} \|\theta_{i,k} - \theta_{i,0}\| \\
&= \lambda \|\theta_{i,k} - \theta_{i,0}\|.
\end{aligned} \tag{58}$$

Note that in the third inequality, we utilized the fact that the term defining $E_{2,\lambda}^{i,j,k}$ is an upper bound on the parameter drift $\|\theta_{i,k} - \theta_{i,0}\|$ (from Lemma 4).

Rearranging this inequality yields:

$$\frac{\|\theta_{i,k} - \theta_{i,0}\|}{\sqrt{\pi_{\tilde{\theta}_{i,k}}(o|q)}} < \frac{1}{\lambda}. \tag{59}$$

Substituting this back into the exponential bounds, we find that whenever $E_{2,\lambda}^{i,j,k}$ occurs:

$$\frac{\pi_{\theta_{i,k}}(o|q)}{\pi_{\theta_{i,0}}(o|q)} \in \left(\exp\left\{ -\frac{\sqrt{2L}}{\lambda} \right\}, \quad \exp\left\{ \frac{\sqrt{2L}}{\lambda} \right\} \right). \tag{60}$$

Finally, we select the coefficient λ_0 to align with the clipping threshold ϵ:

$$\lambda_0 = \min\left\{ -\frac{\sqrt{2L}}{\ln(1-\epsilon)}, \frac{\sqrt{2L}}{\ln(1+\epsilon)} \right\}. \tag{61}$$

With this choice, the occurrence of $E_{2,\lambda_0}^{i,j,k}$ implies:

$$\frac{\pi_{\theta_{i,k}}(o|q)}{\pi_{\theta_{i,0}}(o|q)} \in (1-\epsilon, 1+\epsilon). \tag{62}$$

Thus, $E_{2,\lambda_0}^{i,j,k} \subset E_1^{i,j,k}$. By set complement, $\Omega \setminus E_1^{i,j,k} \subset \Omega \setminus E_{2,\lambda_0}^{i,j,k}$, which establishes the indicator inequality:

$$\mathbf{1}_{\Omega/E_1^{i,j,k}} \leq \mathbf{1}_{\Omega/E_{2,\lambda_0}^{i,j,k}}. \tag{63}$$

We now analyze the norm of the boundary term $\mathcal{E}_{i,l}^{\text{clip}}$. Applying the norm inequality to the expectation:

$$\|\mathcal{E}_{i,l}^{\text{clip}}\| \leq \mathbb{E}_{\pi_{\theta_{i,0}}}\left(\mathbf{I}_{B_l/E_1^{i,j,l}} \cdot \frac{\|\nabla \pi_{\theta_{i,0}}(o|q)\|}{\pi_{\theta_{i,0}}(o|q)} \cdot |A(q,o)|\right). \tag{64}$$

Using the L-smooth assumption on the policy, we have the property $\|\nabla \pi_\theta(o|q)\| \leq \sqrt{2L \cdot \pi_\theta(o|q)}$. Therefore, at the anchor point $\theta_{i,1}$ (noting $\theta_{i,1} \approx \theta_{i,0}$ in context):

$$\frac{\|\nabla \pi_{\theta_{i,1}}(o|q)\|}{\pi_{\theta_{i,1}}(o|q)} \leq \frac{\sqrt{2L \cdot \pi_{\theta_{i,1}}(o|q)}}{\pi_{\theta_{i,1}}(o|q)} = \frac{\sqrt{2L}}{\sqrt{\pi_{\theta_{i,1}}(o|q)}}. \tag{65}$$

The term $\mathcal{E}_{i,l}^{\text{clip}}$ is bounded by:

$$\|\mathcal{E}_{i,l}^{\text{clip}}\| \leq 2\sqrt{2L}R \cdot \mathbb{E}_{\pi_{\theta_{i,0}}}\left(\mathbf{I}_{\Omega/E_2^{i,\lambda,l}} \cdot \frac{1}{\sqrt{\pi_{\theta_{i,0}}(o|q)}}\right). \tag{66}$$

On the event $\mathbf{I}_{\Omega/E_{2,\lambda_0}^{i,j,k}}$ (the complement of the safety set), the probability mass is bounded by the drift magnitude:

$$\sqrt{\pi_{\theta_{i,0}}(o|q)} \leq \left(\frac{\sqrt{2L}}{2} + \lambda_0\right)\left(\sum_{m=1}^{l-1} \eta_{i,m}\right) \cdot C_{\text{step}}. \tag{67}$$

With the indicator inequality, we derive the final bound:

$$\|\mathcal{E}_{i,l}^{\text{clip}}\| \leq 2\sqrt{2L}R \cdot \left[|\mathcal{O}|\left(\frac{\sqrt{2L}}{2} + \lambda_0\right)\left(\sum_{m=1}^{l-1} \eta_{i,m}\right) C_{\text{step}}\right]. \tag{68}$$

Finally, we consolidate the bounds derived for the boundary term $\mathcal{E}_{i,l}^{\text{clip}}$ and the continuity term $\mathcal{E}_{i,l}^{\text{drift}}$ to establish a uniform upper bound on the cumulative bias introduced by the gradient clipping mechanism. By summing their contributions over the update trajectory, we obtain the aggregate bias bound:

$$\begin{aligned}
\left\|\sum_{l=1}^{T} \mathcal{E}_{i,l}^{\text{clip}} + \sum_{l=1}^{T} \mathcal{E}_{i,l}^{\text{drift}}\right\| &\leq \sum_{l=1}^{T}\left(2\sqrt{2L}R \cdot \left[|\mathcal{O}|\left(\frac{\sqrt{2L}}{2} + \lambda_0\right)\left(\sum_{m=1}^{l-1} \eta_{i,m}\right) C_{\text{step}}\right] + 2LR|\mathcal{O}|\left(\sum_{m=1}^{l-1} \eta_{i,m} C_{\text{step}}\right)\right) \\
&\leq 2R|\mathcal{O}|\frac{B}{N}\sum_{l=1}^{T}\sum_{m=1}^{l-1} \eta_{i,m} C_{\text{step}}\left(2L + \sqrt{2L}\lambda_0\right) \\
&\leq 2R|\mathcal{O}|\sum_{m=1}^{T-1} \eta_{i,m} C_{\text{step}}\left(2L + \sqrt{2L}\lambda_0\right) \\
&\triangleq \sum_{m=1}^{T-1} \eta_{i,m} K_{\text{clip}}.
\end{aligned} \tag{69}$$

Here, the constant K_{clip} encapsulates the step-size-independent magnitude of the clipping bias, demonstrating that the cumulative deviation scales linearly with the effective step size.

It is crucial to underscore that, while the parameter shifts for both SERPO and GRPO are governed by a unified theoretical upper bound, the underlying mechanisms driving these updates diverge significantly.

- The SERPO algorithm constructs its estimator via a recursive linear combination of historical anchor gradients and current mini-batch gradients. This design effectively decouples variance reduction from the immediate sample size, ensuring low-variance estimate while reducing the per-iteration sampling burden.

- In contrast, GRPO mandates a full-batch computation at every optimization step to suppress estimator variance. Consequently, by amortizing the cost of variance reduction over the optimization trajectory, SERPO demonstrates a rigorous and substantial improvement in sample efficiency relative to the GRPO baseline.

Equipped with the above inequalities, we can start the proof of the theorem. Using the identity $2\langle a, b\rangle = \|a\|^2 + \|b\|^2 - \|a-b\|^2$, we lower-bound the update. Using the notation $\tilde{\eta}_i = \sum_{k=0}^{T} \eta_{i,k}$, we decompose the aggregated estimator $\sum \eta_{i,k} v_{i,k}$ into the true gradient direction plus the accumulated bias:

$$\begin{aligned}\mathbb{E}\left[\left\langle \nabla J(\theta_{i,0}), \sum_{k=0}^{T} \eta_{i,k} v_{i,k} \right\rangle\right] &= \sum_{k=0}^{T} \eta_{i,k} \mathbb{E}\|\nabla J(\theta_{i,0})\|^2 - \left\langle \nabla J(\theta_{i,0}), \sum_{k=0}^{T} \eta_{i,k} \left(\sum_{l=1}^{T} \|\mathcal{E}_{i,l}^{\text{clip}}\| + \sum_{l=1}^{T} \|\mathcal{E}_{i,l}^{\text{drift}}\| \right) \right\rangle \\ &\geq \frac{\tilde{\eta}_i}{2} \left(\|\nabla J(\theta_{i,0})\|^2 - \left\|\tilde{\eta}_i K_{\text{clip}}\right\|^2 \right). \end{aligned} \tag{70}$$

Here, we utilized the Cauchy-Schwarz inequality and the definition of the clipping bias bound derived earlier.

To bound the second moment of the accumulated updates without relying on the independence of the noise terms to eliminate cross-products, we employ the algebraic inequality $\|\mathbf{a}+\mathbf{b}+\mathbf{c}\|^2 \leq 3(\|\mathbf{a}\|^2 + \|\mathbf{b}\|^2 + \|\mathbf{c}\|^2)$. We explicitly decompose the estimator $v_{i,k}$ into three components: the deterministic true gradient $\nabla J(\theta_{i,0})$, the systematic clipping bias $\Delta_{i,k}$, and the stochastic noise $\xi_{i,k}$.

$$\begin{aligned}\mathbb{E}\left[\left\|\sum_{k=0}^{T} \eta_{i,k} v_{i,k}\right\|^2\right] &= \mathbb{E}\left[\left\|\sum_{k=0}^{T} \eta_{i,k} \left(\nabla J(\theta_{i,0}) + \underbrace{(\mathbb{E}[v_{i,k}] - \nabla J(\theta_{i,0}))}_{\text{Bias}} + \underbrace{(v_{i,k} - \mathbb{E}[v_{i,k}])}_{\text{Variance}} \right)\right\|^2\right] \\ &\leq \mathbb{E}\left[\underbrace{3\tilde{\eta}_i^2 \|\nabla J(\theta_{i,0})\|^2}_{\text{I. Gradient Component}} + \underbrace{3\left(\tilde{\eta}_i^2 K_{\text{clip}}\right)^2}_{\text{II. Bias Component}} + \underbrace{3\tilde{\eta}_i^2 \left[\frac{1}{N} \left(\sigma_{query}^2 + \frac{\sigma_{output}^2}{G} \right) \right]}_{\text{III. Variance Component (Lemma 2)}} \right].\end{aligned} \tag{71}$$

The first inequality results from applying the relaxation to these summed components. For the final bound, we factor out the scalar step sizes, substitute the uniform upper bound, and invoke **Lemma 2** to bound the expectation of the stochastic noise norm, thereby scaling the variance term by the cumulative step mass to account for potential correlations within the batch.

Substituting the relaxed expectation bounds back into Eq. (B) yields the detailed descent inequality:

$$\begin{aligned}\mathbb{E}\left[J(\theta_{i+1,0}) - J(\theta_{i,0})\right] \geq \mathbb{E}\Bigg[ ]\frac{\tilde{\eta}_i}{2} \|\nabla J(\theta_{i,0})\|^2 - \frac{\tilde{\eta}_i}{2} \left\|\tilde{\eta}_i K_{\text{clip}}\right\|^2 \\ - \frac{3\mathcal{L}}{2} \left(\tilde{\eta}_i^2 \|\nabla J(\theta_{i,0})\|^2 + \frac{\tilde{\eta}_i^2}{N} \left(\sigma_{query}^2 + \frac{\sigma_{output}^2}{G} \right) + \left\|\tilde{\eta}_i^2 K_{\text{clip}}\right\|^2 \right) \Bigg].\end{aligned} \tag{72}$$

To tractably analyze this lower bound, we impose the refined step-size stability condition $\mathcal{L}\tilde{\eta}_i \leq 1/6$, allowing us to simplify the inequality:

$$\mathbb{E}\left[J(\theta_{i+1,0}) - J(\theta_{i,0})\right] \geq \mathbb{E}\left[\frac{\tilde{\eta}_i}{4} \|\nabla J(\theta_{i,0})\|^2 - \frac{\tilde{\eta}_i \left(\sigma_{\sigma_{query}^2 + \sigma_{output}^2/G} \right)}{4N} - \frac{1}{4} \tilde{\eta}_i^3 K_{\text{clip}}^2 \right]. \tag{73}$$

Finally, we extend this analysis to the entire training horizon to derive the global convergence rate. Assuming the objective function is bounded above by $J^* = \max_\theta J(\theta)$, we sum the inequality over all

minibatches $i = 0, \ldots, I-1$ and normalize by the total step mass $\frac{1}{4}\sum_{i=0}^{I-1}\tilde{\eta}_i$. This yields the following global guarantee:

$$\mathbb{E}\left[\frac{\sum_{i=0}^{I-1}\tilde{\eta}_i\|\nabla J(\theta_{i,0})\|^2}{\sum_{i=0}^{I-1}\tilde{\eta}_i}\right] \leq \frac{4(J^* - J(\theta_{0,0}))}{\sum_{i=0}^{I-1}\tilde{\eta}_i} + \frac{\sum_{i=0}^{I-1}\left(\tilde{\eta}_i^3 K_{\text{clip}}^2 + \tilde{\eta}_i\frac{\left(\sigma_{query}^2+\sigma_{output}^2/G\right)}{N}\right)}{\sum_{i=0}^{I-1}\tilde{\eta}_i}. \tag{74}$$

Consequently, utilizing the property that the minimum gradient norm is bounded by the weighted average, we conclude with:

$$\min_{i=0..S-1}\mathbb{E}\|\nabla J(\theta_{i,0})\|^2 \leq \frac{4(J^* - J(\theta_{0,0}))}{\sum_{i=0}^{I-1}\tilde{\eta}_i} + \frac{\sum_{i=0}^{I-1}\left(\tilde{\eta}_i^3 K_{\text{clip}}^2 + \tilde{\eta}_i\frac{\left(\sigma_{query}^2+\sigma_{output}^2/G\right)}{N}\right)}{\sum_{i=0}^{I-1}\tilde{\eta}_i}. \tag{75}$$

This result confirms that the algorithm converges to a stationary point, with the residual error term strictly controlled by the estimator variance $\left(\sigma_{query}^2 + \sigma_{output}^2/G\right)/N$ and the clipping bias K_{clip}.

Let $\Delta J = J^* - J(\theta_{0,0})$ denote the initial objective value gap. For the clarity of the theoretical derivation, we assume a constant cumulative step size per batch, denoted by $\tilde{\eta}_i = \eta$ for all $i = 0, \ldots, I-1$. Consequently, the total accumulated step mass over I batches is $I\eta$.

Substituting $\tilde{\eta}_i = \eta$ into the general bound yields:

$$\begin{aligned}\min_{0\leq i<I}\mathbb{E}\|\nabla J(\theta_{i,0})\|^2 &\leq \frac{4\Delta J}{I\eta} + \frac{\sum_{i=0}^{I-1}\left(\eta^3 K_{\text{clip}}^2 + \eta\frac{\sigma_{query}^2+\sigma_{output}^2/G}{N}\right)}{I\eta} \\ &= \frac{4\Delta J}{I\eta} + \eta^2 K_{\text{clip}}^2 + \frac{\sigma_{query}^2 + \sigma_{output}^2/G}{N}.\end{aligned} \tag{76}$$

The upper bound consists of a variance term constant in I, and an optimization term dependent on η. To find the optimal step size that minimizes the upper bound , we define the error function as $f(\eta) = \frac{4\Delta J}{I\eta} + K_{\text{clip}}^2\eta^2$.

Minimizing $f(\eta)$ with respect to η:

$$\frac{df(\eta)}{d\eta} = -\frac{4\Delta J}{I\eta^2} + 2K_{\text{clip}}^2\eta = 0 \quad \Longrightarrow \quad \eta^3 = \frac{2\Delta J}{IK_{\text{clip}}^2}. \tag{77}$$

Solving for η, we obtain the optimal step size scaling:

$$\eta^* = \left(\frac{2\Delta J}{IK_{\text{clip}}^2}\right)^{\frac{1}{3}} \propto \mathcal{O}(I^{-1/3}). \tag{78}$$

Substituting η^* back into the bound:

$$\begin{aligned}f(\eta^*) &= \frac{4\Delta J}{I}\left(\frac{IK_{\text{clip}}^2}{2\Delta J}\right)^{\frac{1}{3}} + K_{\text{clip}}^2\left(\frac{2\Delta J}{IK_{\text{clip}}^2}\right)^{\frac{2}{3}} \\ &= \frac{4\Delta J}{(2\Delta J)^{1/3}I^{2/3}K_{\text{clip}}^{-2/3}} + \frac{K_{\text{clip}}^2(2\Delta J)^{2/3}}{I^{2/3}K_{\text{clip}}^{4/3}} \\ &= 2\cdot(2\Delta J K_{\text{clip}}^2)^{2/3}I^{-2/3} + (2\Delta J K_{\text{clip}}^2)^{2/3}I^{-2/3} \\ &= \frac{3(2\Delta J K_{\text{clip}}^2)^{2/3}}{I^{2/3}}.\end{aligned} \tag{79}$$

This derivation leads to the the final convergence rate.

$$\min_{0\leq i<I}\mathbb{E}\|\nabla J(\theta_{i,0})\|^2 \leq \frac{3\left[2(J^* - J(\theta_{0,0}))K_{\text{clip}}^2\right]^{2/3}}{I^{2/3}} + \frac{1}{N}\left(\sigma_{query}^2 + \frac{\sigma_{output}^2}{G}\right). \tag{80}$$

C Sample Complexity Analysis

In this section, we contrast the fundamental sampling structures of GRPO and SERPO. This comparison highlights the theoretical advantage of SERPO in terms of total sample complexity. Recall the convetgence result:

$$\mathbb{E}\left[\frac{\sum_{i=0}^{I-1}\tilde{\eta}_i\|\nabla J(\theta_{i,0})\|^2}{\sum_{i=0}^{I-1}\tilde{\eta}_i}\right] \leq \frac{4(J^* - J(\theta_{0,0}))}{\sum_{i=0}^{I-1}\tilde{\eta}_i} + \frac{\sum_{i=0}^{I-1}\left(\tilde{\eta}_i^3 K_{\text{clip}}^2 + \tilde{\eta}_i\frac{(\sigma_{query}^2+\sigma_{output}^2/G)}{N}\right)}{\sum_{i=0}^{I-1}\tilde{\eta}_i}. \tag{81}$$

To achieve an ϵ-stationary point, each term on the right-hand side must be bounded by $\mathcal{O}(\epsilon)$. This imposes the following constraints on the hyperparameters:

- **Total Step Mass:** $\sum_{i=0}^{I-1}\tilde{\eta}_i = \mathcal{O}(\epsilon^{-1})$.
- **Clipping Bias Control:** $\tilde{\eta}_i^2 K_{\text{clip}}^2 \leq \mathcal{O}(\epsilon) \implies \tilde{\eta}_i = \mathcal{O}(\epsilon^{1/2})$.
- **Variance Control:** $\frac{(\sigma_{query}^2+\sigma_{output}^2/G)}{N} \leq \mathcal{O}(\epsilon) \implies N = \mathcal{O}(\epsilon^{-1})$.

We define the **Gradient Sample Complexity** $\mathcal{T}_g$ as the total number of sample trajectories generated during training.

Analysis of SERPO (Ours):

We now derive the total gradient sample complexity $\mathcal{T}_{\text{SERPO}}$ required to achieve an ϵ-stationary point. The complexity is determined by aggregating the sampling cost of the snapshot phase and the recursive update phase over the necessary number of batches.

1. Batch Count (I) Let I denote the total number of batches and $\Psi = \sum_{i=0}^{I-1}\tilde{\eta}_i$ be the cumulative step mass required to bridge the optimization gap. From the convergence inequality, minimizing the optimization gap term requires:

$$\Psi = \mathcal{O}(\epsilon^{-1}). \tag{82}$$

Simultaneously, the clipping bias constraint dictates that the cumulative step size per batch, $\tilde{\eta}_i$, must decay to suppress the bias term proportional to $\tilde{\eta}_i^2 \leq \mathcal{O}(\epsilon)$. This imposes the upper bound:

$$\tilde{\eta}_i = \mathcal{O}(\epsilon^{1/2}). \tag{83}$$

Consequently, the total number of batches I scales as the ratio of the total required mass to the permissible mass:

$$I = \frac{\Psi}{\tilde{\eta}_i} = \frac{\mathcal{O}(\epsilon^{-1})}{\mathcal{O}(\epsilon^{1/2})} = \mathcal{O}(\epsilon^{-1.5}). \tag{84}$$

2. Complexity Per Batch In each batch i, the algorithm's sampling operations are structured as follows:

- **Snapshot Step:** A large anchor batch of size N is sampled to compute the anchor gradient $v_{i,0}$. The variance control constraint $\frac{\sigma^2}{N} \leq \mathcal{O}(\epsilon)$ necessitates:

$$N = \mathcal{O}(\epsilon^{-1}). \tag{85}$$

- **Recursive Update Steps:** The algorithm iterates through inner steps using minibatches $\mathcal{B}_k$. Crucially, in the SERPO framework, these minibatches are **partitioned** directly from the initial anchor batch $\mathcal{L}_i$. Therefore, the inner loop incurs no additional trajectory sampling costs.

Thus, the total number of trajectory sample queries generated per batch is determined by the anchor batch: $N = \mathcal{O}(\epsilon^{-1})$.

3. Total Gradient Sample Complexity The total sample complexity $\mathcal{T}_{\text{SERPO}}$ is the product of the number of batches and the sampling cost per batch. Substituting the derived scalings:

$$\mathcal{T}_{\text{SERPO}} = \sum_{i=0}^{I-1} N = I \times N = \mathcal{O}(\epsilon^{-2.5}). \tag{86}$$

Analysis of GRPO (Baseline):

We analyze the gradient sample complexity for Group Relative Policy Optimization (GRPO). Unlike SERPO, GRPO lacks the recursive correction mechanism and must sample a full batch of size N at every inner iteration to maintain the same variance control guarantees. This fundamental difference drastically alters the total sampling burden.

1. Iteration and Hyperparameter Constraints The theoretical constraints on the step mass and batch size remain consistent with the SERPO analysis to ensure the optimization gap and clipping bias are bounded. However, the sampling frequency changes:

- **Total Batches (I):** Dictated by the ratio of total required step mass (ϵ^{-1}) to the permissible batch step mass ($\epsilon^{1/2}$), the number of outer iterations scales as $I = \mathcal{O}(\epsilon^{-1.5})$.

- **Batch Size (N):** To bound the variance term $\frac{\sigma^2}{N} \leq \mathcal{O}(\epsilon)$, the sample size must scale as $N = \mathcal{O}(\epsilon^{-1})$.

- **Inner Loop Length (T):** We assume the number of inner loop iterations as T. This parameter T dictates the extent of off-policy updates.

2. Total Gradient Sample Complexity The total sample complexity $\mathcal{T}_{\text{GRPO}}$ is calculated by summing the sample cost over all inner iterations across all batches. Since every step requires a fresh batch of N samples:

$$\mathcal{T}_{\text{GRPO}} = \sum_{i=0}^{I-1} \sum_{k=1}^{T} N = I \times T \times N = \mathcal{O}(T\epsilon^{-2.5}). \tag{87}$$

The SERPO algorithm maintains controlled variance with a large-batch requirement N at the anchor points. In comparison, the GRPO algorithm exhibits a total sample complexity of $\mathcal{O}(T\epsilon^{-2.5})$, which is strictly higher than SERPO's $\mathcal{O}(\epsilon^{-2.5})$ whenever $T > 1$. The difference reflects the cost of sampling the full batch N at every step to suppress variance, whereas SERPO efficiently reuses the anchor gradient. SERPO provides a structural advantage by decoupling the variance reduction from the parameter update frequency, allowing for higher data efficiency.

D Extended Empirical Evaluations

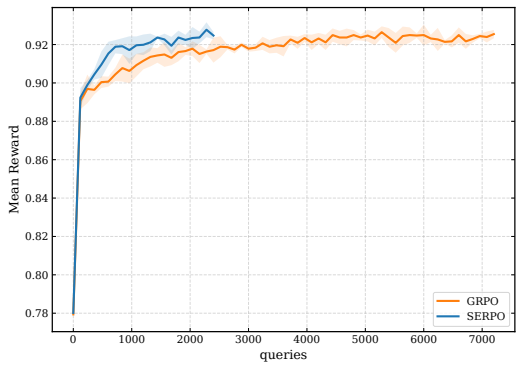

Figure 3: **Validation accuracy dynamics on GSM8K and Qwen2.5-7B.** Solid lines denote the mean, and shaded regions represent the standard deviation over three random seeds.

This section presents the training results and training curve, complementing the empirical findings in Section 5.

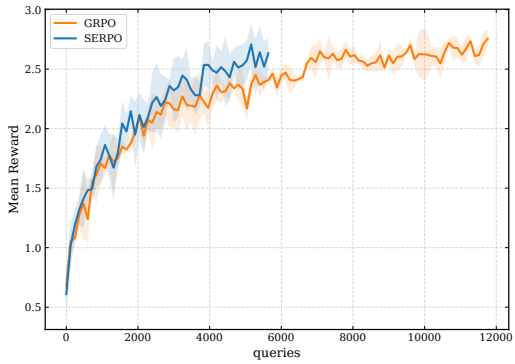

Figure 4: **Validation accuracy dynamics on KK and Qwen2.5-7B.** Solid lines denote the mean, and shaded regions represent the standard deviation over three random seeds.

SETTING	ACCURACY ↑	QUERIES (K) ↓	TIME (MIN) ↓
GSM8K			
$B = 2$	0.425 ± 0.023	**0.32**	**22.7**
$B = 4$	0.451 ± 0.006	0.96	68.6
$B = 8$	$\mathbf{0.477 \pm 0.030}$	1.44	100.7
KK DATASET			
$B = 2$	0.213 ± 0.038	**1.28**	**46.7**
$B = 4$	0.253 ± 0.038	2.08	79.3
$B = 8$	$\mathbf{0.277 \pm 0.021}$	2.56	95.0

Table 6: 0.5B-Ablation study on Mini-batch Size (B).

Table 6 and Table 7 report the ablation results of batch size and mini-batch size on the Qwen2.5-0.5B-Instruct model. All hyperparameters except for the batch and mini-batch configurations remain identical to the main experiments, with the learning rate fixed to the optimal value identified in Section E.2.

To further verify the scalability of our approach, we extend our evaluation to larger model capacities and more rigorous benchmarks by incorporating results for the Qwen3-1.7B and Qwen3-8B models. These scaled-up models are trained on the challenging MATH dataset and evaluated across the MATH and AIME25 benchmarks.

Owing to computational constraints, the supplementary scaling experiments on Qwen3 were conducted using a single trial per configuration. Nevertheless, the highly consistent empirical trends observed across these setups firmly demonstrate that our algorithm maintains robust efficiency gains at larger training scales. The comprehensive hyperparameter configurations for these additional experiments are detailed in Appendix 5.

Setting	Accuracy ↑	Queries (K) ↓	Time (min) ↓
GSM8K			
$N = 8$	0.460 ± 0.002	**0.88**	90.0
$N = 12$	$\mathbf{0.464 \pm 0.032}$	0.96	78.6
$N = 16$	0.451 ± 0.006	0.96	68.6
$N = 20$	0.464 ± 0.001	1.00	**63.5**
KK Dataset			
$N = 8$	$\mathbf{0.293 \pm 0.023}$	2.32	98.5
$N = 12$	0.280 ± 0.017	2.4	92.0
$N = 16$	0.253 ± 0.038	**2.08**	**79.3**
$N = 20$	0.193 ± 0.064	2.4	79.6

Table 7: 0.5B-Ablation study on Batch Size (N).

Method	Math ↑	AIME ↑	Queries (K) ↓	Time (M) ↓
SERPO	0.851	0.233	**4.16**	**658.1**
GRPO	0.852	0.200	9.92	1067.8

Table 8: Evaluation of Qwen3-8B on MATH and AIME datasets with a response length of 4096 tokens.

E Experimental details

E.1 Data and Documentation Statement

We utilize four publicly available, standard benchmarks for reasoning: GSM8K, Knights and Knaves, MATH, and AIME25. These datasets primarily cover English-language mathematical reasoning, ranging from grade-school arithmetic to advanced competition-level mathematics. All datasets are used in strict accordance with their original intended purpose: academic research in reasoning. We confirm that our usage is consistent with the access conditions stipulated by the original providers.

Licensing and Ethical Compliance. All artifacts are utilized under their respective open-source licenses . These datasets do not contain personally identifiable information or offensive content. As our research does not involve human subjects, annotators, or the collection of private data, no additional anonymization protocols or ethics review board approvals were required. We assume no ownership of these public artifacts and adhere to all terms of use and distribution.

E.2 Hyperparameters

Tables 10 ,11, 12, 13 present the full hyperparameter configurations employed to obtain our final experimental results. For the Qwen2.5-0.5B-Instruct, we performed learning rate searches over the range [1e-6,5e-6] for the GSM8K dataset and [1e-7,1e-6] for the KK Dataset; for the Qwen2.5-7B-Instruct, the learning rate search range was [1e-6,5e-6] for both the GSM8K and KK Datasets. The final learning rate values for each model-dataset combination were selected based on the stability of the reward curve on the validation set.

Method	N	Accuracy ↑	Queries (K) ↓	Time (M) ↓
SERPO	32	0.720	**6.97**	834.9
SERPO	64	0.721	8.82	**564.9**
GRPO	32	0.714	9.60	1137.9
GRPO	64	0.718	15.08	736.6

Table 9: Ablation study on Global Batch Size (N) for Qwen3-1.7B on the MATH dataset (4096 tokens).

Hyperparameter	**Value (GSM8K)**	**Value (KK)**
Optimizer	AdamW	
SERPO LR	2.0×10^{-6}	5.0×10^{-7}
GRPO LR	4.0×10^{-6}	8.0×10^{-7}
Grad. Accum. Steps	4	
Max Grad. Norm	1.0	
GRPO Config		
Clip Ratio (ϵ)	0.2	
KL Coefficient (β)	0.0	
Epochs	1	
Global Batch Size	16	
PPO Mini-Batch Size	16	
Micro-Batch Size	4	
Generation (Rollout)		
Rollouts (G)	8	
Temp. (Val)	1.0	
Temp. (Val)	0.0	
Resp. Length	512	

Table 10: Complete hyperparameter settings for Qwen2.5-0.5B-Instruct experiments.

Hyperparameter	**Value (GSM8K)**	**Value (KK)**
Optimizer	AdamW	
SERPO LR	1.0×10^{-6}	2.0×10^{-6}
GRPO LR	3.0×10^{-6}	5.0×10^{-6}
Grad. Accum.	4	
Max Grad. Norm	1.0	
GRPO Config		
Clip Ratio (ϵ)	0.2	
KL Coefficient (β)	0.001	
Epochs	10	
Global Batch Size	48	
PPO Mini-Batch Size	24	
Micro-Batch Size	8	
Generation (Rollout)		
Rollouts (G)	8	
Temp. (Train)	1.0	
Temp. (Val)	0.0	
Max Resp. Length	1024	

Table 11: Complete hyperparameter settings for Qwen2.5-7B-Instruct experiments.

Hyperparameter	Value / Configuration
Optimizer	AdamW
SERPO LR	2.0×10^{-6}
GRPO LR	3.0×10^{-6}
Grad. Accumulation	4
Max Grad. Norm	1.0
GRPO Config	
Clip Ratio (ϵ)	0.2
KL Coefficient (β)	0.0
Epochs	15
Global Batch Size	128
PPO Mini-Batch Size	$\{32, 64\}$
Micro-Batch Size	$\{8, 16\}$
Generation (Rollout)	
Rollouts (G)	$\{4, 8\}$
Temperature (Training)	1.0
Temperature (Validation)	0.0
Max Prompt Length	1024
Max Response Length	4096

Table 12: Hyperparameter settings for Qwen3-1.7B scaling experiments on the MATH dataset.

Hyperparameter	Value / Configuration
Optimizer	AdamW
SERPO LR	2.0×10^{-6}
GRPO LR	3.0×10^{-6}
Grad. Accumulation	4
Max Grad. Norm	1.0
GRPO Config	
Clip Ratio (ϵ)	0.2
KL Coefficient (β)	0.0
Epochs	15
Global Batch Size	256
PPO Mini-Batch Size	64
Micro-Batch Size (per GPU)	16
Generation (Rollout)	
Rollouts (G)	8
Temperature (Training)	1.0
Temperature (Validation)	0.0
Max Prompt Length	1024
Max Response Length	4096

Table 13: Hyperparameter settings for Qwen3-8B scaling experiments on the MATH and AIME25 dataset.

E.3 Prompt Engineering

We employ task-specific prompting strategies tailored to the requirements of each benchmark. The exact prompt templates used in our experiments are described below.

GSM8K (Mathematical Reasoning). For the GSM8K dataset, we adopt a standard zero-shot strategy. The instruction triggering step-by-step reasoning is appended directly to the user query. The input format is as follows:

```
[Question] Let's think step by step and output the final answer after "####".
```

The model is expected to generate a reasoning chain followed by the final numerical result delimited by the "####" token.

Knights and Knaves (Logical Constraint Satisfaction). For the logic puzzles, we utilize a structured ChatML-style prompt containing a specific system instruction and a pre-filled assistant response.

System Message:

```
You are a helpful assistant. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within <think> </think> and<answer> </answer> tags, respectively, i.e., <think> reasoning process here </think><answer> answer here </answer>. Now the user asks you to solve a logical reasoning problem. After thinking, when you finally reach a conclusion, clearly state the identity of each character within <answer> </answer> tags. i.e., <answer> (1) Zoey is a knight
(2) ... </answer>.
```

User Message:

```
A very special island is inhabited only by knights and knaves. Knights always tell the truth, and knaves always lie. You meet 3 inhabitants: [Names]. [Statements]. So who is a knight and who is a knave?
```

Assistant Message (Pre-filled):

```
<think>
```

We strictly enforce the output format using the system instruction and facilitate immediate reasoning generation by pre-filling the start token `<think>`.

MATH Dataset. For the MATH dataset, the problem statements are wrapped in a standard conversational structure designed to trigger rigorous mathematical breakdown. The input layout consists of a systemic formatting constraint, the raw math puzzle, and a pre-filled thinking token:

System Message:

```
You are a helpful assistant. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within and tags, respectively, i.e., reasoning process here answer here
```

```
. Now the user asks you to solve a mathematical reasoning problem. After exploring multiple trajectories, please output your final numerical or algebraic conclusion inside the tags.
```

User Message:

```
[MATH Problem Statement]
```

AIME Dataset (Advanced Mathematics Entrance Awards). Given that AIME consists of highly complex, competition-level problems that typically require integer outputs between 000 and 999, we incorporate an explicit constraint within the prompt to enforce integer-focused tracking alongside the standard formatting tags:

System Message:

```
You are a helpful assistant specialized in competitive mathematics. The assistant first executes an intensive step-by-step mathematical derivation and then provides the final answer. The reasoning process and answer must be strictly enclosed within and tags, respectively. Since this is an AIME-level problem, verify your derivation steps carefully and place your final non-negative integer solution within the tags.
```

User Message:

```
[AIME Problem Statement]
```

E.4 Reward Function Design

To rigorously evaluate the model's reasoning capabilities, we implemented task-specific rule-based reward functions. The reward structure $R = r_{\text{fmt}} + r_{\text{ans}}$ balances syntactic correctness (r_{fmt}) with semantic accuracy (r_{ans}).

GSM8K. We employ strict pattern matching to validate numerical answers.

- **Extraction Strategy:** The final answer is extracted via regular expressions targeting the specific delimiter pattern . Formatting artifacts such as commas or currency symbols are normalized prior to comparison.
- **Scoring Protocol:**
  - For Qwen2.5-0.5B-Instruct, we adopt a dense reward scheme: a format reward $r_{\text{fmt}} = 0.2$ is granted for adhering to the required output structure, and an answer reward $r_{\text{ans}} = 0.8$ is awarded for numerical exact matches with the ground truth.
  - For Qwen2.5-7B-Instruct, we transition to a sparse, binary reward signal ($R \in \{0, 1\}$), where credit is assigned solely for correct final answers, serving as a strict verification of reasoning fidelity.

Knights and Knaves. We utilize a comprehensive parsing pipeline to verify logic puzzle solutions.

- **Extraction Strategy:** The model's output is parsed for definitive identity assignments following the template "`<Name> is a <Role>`" (where Role $\in \{\text{Knight}, \text{Knave}\}$).
- **Scoring Protocol:**
  - **Format Reward (r_{fmt}):** A score of $+1$ is awarded if the model successfully provides identity predictions for the complete set of characters mentioned in the puzzle. Failure to adhere to the required XML-style tagging structure results in a penalty of -1.
  - **Answer Reward (r_{ans}):** Based on the parsed identities, we apply a tiered scoring system: a full correct deduction yields $+2$, while any discrepancy with the ground truth logic constraints incurs a penalty of -1.5. Parsing failures or incomplete answers result in a penalty of -2.

MATH. The verification system checks the generated reasoning trajectory based on exact mathematical equivalence:

- **Extraction Strategy:** A regular expression scans the response to extract the string sequence encapsulated between the and tags.
- **Scoring Rules:**
  - **Perfect Match ($R = 1.0$):** Granted if the XML block is correctly closed *and* the extracted expression matches the ground-truth solution after algebraic normalization (e.g., fraction reduction, spacing removal).
  - **Failure ($R = 0.0$):** If the text fails to form a proper XML block or the extracted final result is mathematically incorrect, the reward is zero.

2. Reward Protocol for AIME. Considering the unique mathematical distribution of the AIME competition, the reward function applies a specialized numerical casting protocol:

- **Extraction Strategy:** The parsing engine isolates the text within the tags and applies an additional filtering layer to strip away leading zeros or residual units, normalizing the text into a clean numeric string.
- **Scoring Rules:**
  - **Perfect Match ($R = 1.0$):** Awarded if the model successfully preserves the XML container structure *and* the extracted integer strictly matches the specific target integer from the AIME ground truth.
  - **Failure ($R = 0.0$):** Any syntax distortion (missing or broken tags), fractional outputs where an integer was expected, or numerical mismatches automatically result in a total score of 0.

E.5 Computational Resources and Footprint

Experiments for Qwen2.5-0.5B-Instruct were conducted on a single NVIDIA RTX 4090 (24GB) GPU. Experiments for Qwen2.5-7B-Instruct were conducted on 4 NVIDIA H800(80GB) GPU. Experiments for Qwen3-1.7B were conducted on 8 NVIDIA H800(80GB) GPU. Experiments for Qwen3-8B were conducted on 16 NVIDIA H800(80GB) GPU. To accommodate the memory requirements of reinforcement learning with LLMs, we utilized the following optimization techniques provided by the VeRL framework:

- **Hybrid Engine:** We used vLLM for high-throughput rollout generation and PyTorch FSDP for model training.
- **Memory Optimization:** We enabled gradient checkpointing and offloaded optimizer states to the CPU.
- **Precision:** All training was performed using mixed precision with bfloat16 for forward and backward computation and float32 for gradient reduction, following the FSDP mixed precision configuration.

Empirically, our framework introduces non-trivial peak VRAM overhead, scaling at approximately $1.3\times$ during Qwen2.5-7B-Instruct training, and stabilizing at around $1.2\times$ for Qwen2.5-0.5B-Instruct, Qwen3-1.7B and Qwen3-8B. Mechanistically, this supplementary memory footprint originates from two distinct primary sources: the retention of auxiliary model states along with their corresponding gradient tensors, and the caching of generated token logits.The dominant factor governing this overhead exhibits a dynamic trade-off controlled by the interplay between sequence length and model capacity.

F AI-Assisted Writing Statement

We acknowledge the use of AI-assisted tools to refine the clarity, grammar, and flow of the manuscript during the writing process. After using these tools, the authors carefully reviewed, edited, and validated all generated text to ensure the accuracy and integrity of the technical contributions. All scientific content, data analysis, and conclusions are original work by the authors.