REINFORCE++: STABILIZING CRITIC-FREE POLICY OPTIMIZATION WITH GLOBAL NORMALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Critic-free reinforcement learning reduces the cost of post-training large language models, but its behavior depends strongly on how rewards are converted into advantages. In group-relative methods, normalizing rewards separately for each prompt couples the update scale to statistics estimated from a small set of responses. We introduce REINFORCE++, a critic-free policy optimization framework that instead normalizes advantages across the rollout batch. The basic variant supports one or more responses per prompt and incorporates a token-level KL penalty into Monte Carlo returns. A second variant, REINFORCE++-Baseline, combines a within-prompt reward baseline with global normalization and a separate KL regularizer. Our analysis characterizes finite-sample normalization and its convergence to population standardization under independent sampling. Experiments cover preference-based alignment, mathematical and logical reasoning, and multi-step tool use. With one response per prompt, REINFORCE++ obtains an Arena-Hard score of 46.7, compared with 46.8 for four-response GRPO. In tool-use experiments, REINFORCE++-Baseline achieves an average score of 24.10, compared with 22.58 for GRPO and 21.85 for PPO. These results support global normalization as a practical design choice for critic-free language-model training.

1 INTRODUCTION

Reinforcement Learning from Human Feedback (RLHF) is used to align Large Language Models (LLMs) with human values and preferences (Vemprala et al., 2023; Achiam et al., 2023; Ouyang et al., 2022; Shen & Zhang, 2024; Shen et al., 2025; Hu et al., 2024). Alongside preference-optimization methods such as DPO (Rafailov et al., 2023), systems including ChatGPT/GPT-4 (Vemprala et al., 2023; OpenAI, 2023), Claude (Anthropic, 2023), and Gemini (Team et al., 2023) have incorporated reinforcement learning into post-training.

A widely used RL algorithm in this setting is Proximal Policy Optimization (PPO) (Schulman et al., 2017). In actor–critic implementations of PPO, a learned value function provides a baseline for estimating advantages. Training this value network adds computation and memory costs (Shao et al., 2024).

REINFORCE-based methods replace the learned critic with reward-based baselines. Examples include ReMax (Li et al., 2023), RLOO (Ahmadian et al., 2024), and GRPO (Shao et al., 2024). These algorithms construct reward-based baselines using greedy decoding or multiple sampled responses to the same prompt.

These critic-free methods differ in their approach to *advantage estimation*. GRPO uses **prompt-level (local) normalization**, estimating a baseline and standard deviation from responses to one prompt; RLOO uses a leave-one-out baseline without inherently requiring this scale normalization. First, Appendix B shows that finite-sample self-normalization differs from population standardization. Second, small groups (e.g., $k = 4$ or $k = 8$) make the scale sensitive to sampled outcomes. Homogeneous groups provide no centered reward signal, while different reward spreads induce different prompt-specific scales. Prompt-specific scales also change the relative weighting of reward differences across prompts. Our experiments examine the resulting training dynamics and held-out generalization.

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

We introduce REINFORCE++ , a framework with two variants centered on global advantage normalization. The basic variant supports $k \geq 1$. It samples one response per prompt when $k = 1$ (Algorithm 1), and supports multiple responses when $k > 1$ (Section 4.2). REINFORCE++-Baseline is a *variant* for complex tasks ($k > 1$) that constructs a local reward baseline from **group sampling**. It separates reward centering from scale estimation by first subtracting the *group mean* (for within-prompt comparisons) and then normalizing by the *global standard deviation* (for a shared batch scale). It also employs a k_2 gradient surrogate for separate KL regularization, as detailed in Appendix C. Our contributions are:

- We bound self-normalized rewards and characterize their finite-sample conditional expectation under Gaussian sampling (Appendix B).
- We propose REINFORCE++ , a critic-free framework centered on Global Advantage Normalization, to share normalization statistics across prompts while retaining PPO clipping.
- We present two algorithms: REINFORCE++ supports $k \geq 1$ for general RLHF and reasoning, while REINFORCE++-Baseline combines group-mean reward centering with global normalization for $k > 1$.
- The experiments show that REINFORCE++ achieves competitive general RLHF scores with shorter responses than GRPO and higher held-out reasoning scores. The baseline variant achieves higher mean accuracy than GRPO and PPO across five tool-use benchmarks.

2 Background and Related Work

2.1 PPO and Critic-Free RLHF

PPO optimizes LLMs by maximizing the following surrogate objective:

$$\mathcal{L}_{\text{PPO}}(\theta) = \mathbb{E}_{q\sim P(Q),\, o\sim\pi_{\theta_{\text{old}}}(O|q)} \left[\frac{1}{|o|} \sum_{t=1}^{|o|} \min \Big(s_t(\theta) A_t, \; \text{clip}\big(s_t(\theta),\, 1-\epsilon,\, 1+\epsilon\big) A_t \Big) \right] \tag{1}$$

where $s_t(\theta) = \frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{\theta_{\text{old}}}(o_t|q,o_{<t})}$ is the probability ratio. The advantage A_t is typically calculated using Generalized Advantage Estimation (GAE) (Schulman et al., 2015):

$$A_{q,o_t} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l} \tag{2}$$

where $\delta_{q,o_t} = r_t + \gamma V(o_{t+1}) - V(o_t)$ is the temporal difference error, and $V(\cdot)$ is the critic network.

Critic-free methods in Figure 1 remove the critic $V(\cdot)$ and compute the advantage A_t directly from rewards.

- **ReMax** (Li et al., 2023) uses a greedy decoding response $\hat{o}$ as the baseline: $A_{q,o_t} = r(o) - r(\hat{o})$.
- **RLOO** (Ahmadian et al., 2024) samples k responses and uses the mean of others as the baseline: $A_{q,o_t^{(i)}} = r(o^{(i)}) - \frac{1}{k-1} \sum_{j\neq i} r(o^{(j)})$.
- **GRPO** (Shao et al., 2024) also samples k responses but normalizes the advantage using the mean and standard deviation of the *local group*:

$$A_{q,o_t^{(i)}} = \frac{r(o^{(i)}) - \text{mean}\big(\{r(o^{(j)})\}_{j=1}^{k}\big)}{\text{std}\big(\{r(o^{(j)})\}_{j=1}^{k}\big) + \epsilon} \tag{3}$$

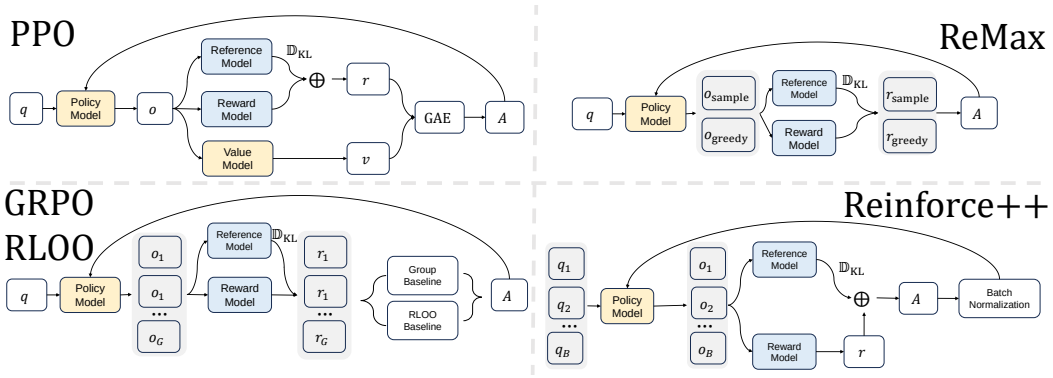


Figure 1: Comparison of PPO, ReMax, GRPO, RLOO, and REINFORCE++ . REINFORCE++ uses global advantage normalization with reward-based returns and no learned critic. REINFORCE++-Baseline uses group sampling, similar to GRPO/RLOO, but combines group-mean subtraction with global batch normalization.

2.2 Properties of Local Normalization

The **prompt-level (local) normalization** in Eq. (3) has three properties relevant to advantage estimation and evaluation:

1. **Finite-Sample Distortion**: Appendix B proves that self-normalization differs from the population-standardized reward in finite samples. The result concerns the conditional expectation of a self-normalized scalar reward under independent Gaussian sampling.
2. **Practical Sensitivity**: Small groups (e.g., 4 or 8 responses) provide noisy estimates of reward spread. Standardized advantages are bounded by $\sqrt{k-1}$ for nonconstant groups using divisor k. Identical rewards produce zero centered signal; mixed outcomes induce prompt-specific update scales.
3. **Prompt Weighting**: Group-relative updates compare responses to the same prompt and rescale each group by its own reward spread. Our small-data experiments evaluate training-set fit and held-out performance separately to examine generalization under this weighting.

3 Method

Our method uses normalization statistics pooled across the rollout batch for critic-free RLHF. The variants differ in reward centering and KL regularization.

3.1 Global Normalization

The primary REINFORCE++ algorithm is designed for general-purpose RLHF. In its $k = 1$ configuration, it allocates each response in the rollout budget to a distinct prompt.

It optimizes the PPO objective in Eq. (1) but redefines the advantage A_{q,o_t}. We combine a terminal task reward with token-level k_1 KL penalties to obtain the return:

$$A_{q,o_t} = r(o_{1:T}, q) - \beta \cdot \sum_{i=t}^{T} \text{KL}(i) \tag{4}$$

where $\text{KL}(t) = \log\left(\frac{\pi_{\theta_{\text{old}}}(o_t|q,o_{<t})}{\pi_{ref}(o_t|q,o_{<t})}\right)$.

We normalize advantages using a shared batch mean and standard deviation, both for single-response and multiple-response sampling (Andrychowicz et al., 2020):

$$A^{\text{norm}}_{q,o_t} = \frac{A_{q,o_t} - \text{mean}\left(A|A \in \mathcal{D}_{\text{batch}}\right)}{\text{std}\left(A|A \in \mathcal{D}_{\text{batch}}\right) + \epsilon} \tag{5}$$

Pooling a rollout batch (e.g., 1024 responses) gives all prompts a shared advantage scale. Appendix B characterizes convergence of the normalization statistics under independent sampling. Valid token positions exclude padding, and the moments remain fixed during rollout reuse. The single-response $k = 1$ case is detailed in Algorithm 1.

3.2 Local Baseline

For multi-step reasoning, we consider multiple responses per prompt ($k > 1$) (Yue et al., 2025). The baseline variant, REINFORCE++-Baseline , centers their rewards within each prompt group. This variant uses group sampling for reward comparisons and global normalization for scale estimation. The advantage calculation is a two-step process:

- **Group Mean Subtraction for Reshaping**: We first subtract the group mean reward. This baseline removes reward offsets shared by responses to the same prompt; the subsequent global normalization sets their common scale.

$$A'_{q,o_t^{(i)}} = r(q, o^{(i)}) - \frac{1}{k}\sum_{j=1}^{k} r(q, o^{(j)}) \tag{6}$$

- **Global Batch Normalization for Scaling**: We then normalize this initial advantage using the **global batch statistics** shared across prompts.

$$A^{\text{norm}}_{q,o_t} = \frac{A'_{q,o_t} - \text{mean}_{\text{batch}}(A')}{\text{std}_{\text{batch}}(A') + \epsilon} \tag{7}$$

This separates reward centering from scale estimation; the group-centered reward equals $(k-1)/k$ times the RLOO advantage (Appendix D).

The baseline variant also uses a separate k_2 KL regularizer. At a fixed context with on-policy actions held fixed during differentiation, its expected pointwise gradient equals the reverse-KL gradient (Liu et al., 2025a). During old-policy rollout reuse, the same term serves as an off-policy surrogate (Appendix C). The maximized objective is:

$$\mathcal{L} = \mathcal{L}_{\text{PPO}}(A^{\text{norm}}) - \lambda\, \mathbb{E}_{o\sim\pi_{\text{old}}}\left[\frac{1}{|o|}\sum_t \frac{1}{2}\left(\log\frac{\pi_\theta(o_t \mid q, o_{<t})}{\pi_{ref}(o_t \mid q, o_{<t})}\right)^2\right]. \tag{8}$$

3.3 Relationship with PPO

Both variants retain PPO clipping without a learned critic. Setting $V = 0$ and $\gamma = \lambda_{\text{GAE}} = 1$ reduces GAE to Monte Carlo returns, the structure used by the basic variant before normalization. REINFORCE++-Baseline additionally uses prompt-specific reward centering and a separate KL loss; these changes distinguish it from standard actor–critic PPO.

3.4 Implementation Details

Token-level Advantage The task's immediate reward is zero before termination, but its undiscounted return contributes to every preceding response token in Eq. (4). REINFORCE++-Baseline broadcasts its centered terminal reward to valid response positions. The objective averages token losses within each response and then across responses, as specified in Appendix A.

Batch Construction For REINFORCE++ ($k = 1$), a global batch of size N comprises N distinct prompts. For REINFORCE++-Baseline ($k = 4$), a global batch of $N = 1024$ responses contains $1024/4 = 256$ prompts. The global moments pool valid response positions across the rollout batch.

Mini-Batch Updates We partition each rollout batch into mini-batches for policy optimization:

- **Batch Processing:** Each update evaluates the clipped objective on a subset of the rollout batch.
- **Multiple Updates:** Reuse detached advantages and old-policy probabilities over E optimization epochs.

- **Stochastic Optimization:** Mini-batch sampling determines which responses contribute to each update.

Reward Normalization and Clipping Reward preprocessing controls the location, range, and scale of reward values:

- **Normalization:** Centers rewards and divides by a shared batch standard deviation.
- **Clipping:** Bounds reward values before advantage computation.
- **Scaling:** Sets the magnitude of the task reward relative to the KL penalty.

3.5 Statistical Motivation

Omitting the numerical stabilizer, under independent sampling with nonzero population variance, normalization approaches the population-standardized target as $N \to \infty$ (Appendix B). This motivates global normalization in both REINFORCE++ and REINFORCE++-Baseline : a batch of 1024 responses pools more observations than groups of 4 or 8. Within-response token dependence affects the effective sample size of the pooled statistics. The scalar convergence analysis describes scale estimation; clipping and rollout reuse enter separately through the optimization objective.

3.6 Summary

The two variants share **Global Advantage Normalization** and a critic-free clipped objective. Instead of normalizing within a prompt's local group, REINFORCE++ normalizes the advantage function across the *entire global training batch*. The method has the following properties:

- **Shared Statistics**: Sample moments converge under independent sampling. For token-level statistics, within-trajectory dependence determines the effective sample size (Appendix B).
- **Critic-Free Optimization**: Both variants avoid a learned value network and its associated forward passes, gradient updates, and optimizer state.
- **Held-Out Evaluation**: The reasoning experiments report higher AIME and AMC scores than GRPO with local normalization.

Algorithm 1 REINFORCE++ when $k = 1$

Require: Initial policy model π_{ref}, reward models R, task prompts $\mathcal{D}$

policy model $\pi_\theta \leftarrow \pi_{ref}$
for step $= 1, \ldots, M$ **do**
  Sample a batch $\mathcal{D}_{batch}$ from $\mathcal{D}$
  Update the old policy model $\pi_{\text{old}} \leftarrow \pi_\theta$
  Sample output ($k = 1$) $o \sim \pi_{\text{old}}(\cdot \mid q)$ for each question $q \in \mathcal{D}_{batch}$
  Compute rewards r_i for each sampled o_i using R
  Compute advantage A_{q,o_t} via Eq. (4)
  Normalize advantages globally across $\mathcal{D}_{batch}$ to get A^{norm}_{q,o_t} via Eq. (5)
  for iteration $= 1, \ldots, E$ **do**
    Update π_θ by maximizing the objective in Eq. (1) using A^{norm}_{q,o_t}
  end for
end for

Ensure: π_θ

4 Experiments

We evaluate our two algorithms, REINFORCE++ and REINFORCE++-Baseline , in their respective target domains using the OpenRLHF framework (Hu et al., 2024).

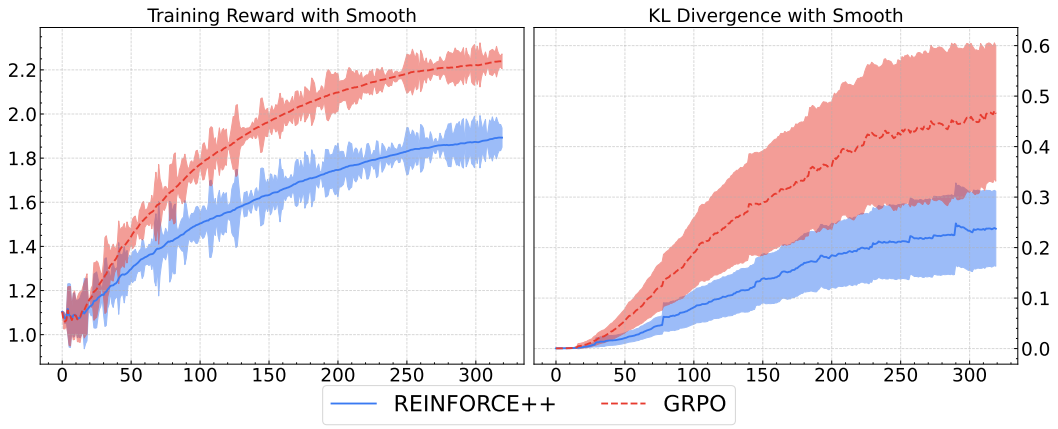


Figure 2: Smoothed training reward and KL divergence. REINFORCE++ ($k = 1$) shows a more gradual reward increase and lower KL divergence than GRPO over the plotted training interval.

4.1 General RLHF

We evaluate REINFORCE++ in its $k = 1$ configuration for general-domain RLHF, allocating one sampled response to each prompt in the rollout batch.

Experimental Setup We use an instruction-following policy model (Llama-3-8B-SFT) and refine it using a Bradley-Terry reward model (Bai et al., 2022) trained on ∼700K human preference pairs. The policy is trained on 20,000 diverse prompts. We compare REINFORCE++ ($k = 1$) with other critic-free methods: GRPO ($k = 4$), RLOO ($k = 4$), and ReMax ($k = 1 + 1$).

Table 1: Comparison on Arena-Hard (Li et al., 2024). The single-sample REINFORCE++ ($k = 1$) scores 46.7 compared with 46.8 for GRPO, with shorter mean responses. Score/length divides the evaluation score by mean response length.

	Score	Length	Score/length
REINFORCE++ (k=1)	46.7	832	**0.0561**
GRPO (k=4)	**46.8**	860	0.0544
RLOO (k=4)	44.6	866	0.0515
ReMax (k=1+1)	45.1	**805**	0.0560

Experimental Results As shown in Table 1, the single-sample **REINFORCE++** ($k = 1$) achieves a score of 46.7, close to the group-sampling GRPO (46.8). However, it produces shorter responses (832 tokens vs. 860), yielding a higher score/length ratio (0.0561). The single-response configuration therefore achieves a similar evaluation score with shorter mean responses.

Results Analysis Figure 2 shows the training dynamics. GRPO's reward and KL divergence rise rapidly, while REINFORCE++ exhibits a more gradual reward increase and smaller divergence. These curves describe the reward–KL trade-offs of the compared training configurations. Response length can also influence preference-based evaluation (Dubois et al., 2024). Table 1 reports both quantities for each method.

4.2 Reasoning Experiments

We evaluate REINFORCE++ with multiple responses per prompt ($k > 1$) on reasoning tasks with rule-based rewards. The comparison with GRPO examines global and local normalization in this grouped-sampling setting.

4.2.1 LONG REASONING TASK

We use rule-based rewards (RLVR) in mathematical reasoning settings (Guo et al., 2025; Seed et al., 2025).

Analysis on Small-Scale Datasets To examine generalization from a small training set, we trained on 30 questions from AIME-24 and evaluated on AIME-25. As shown in Table 2, GRPO (local norm) achieves 95.0% on the training set and 0.0 Pass@1 and 0.4 Pass@16 on the test set, showing a large train–test performance gap. In contrast, REINFORCE++ with global norm, while scoring lower on the training set (71.0), shows higher held-out scores (2.5 Pass@1, 40.0 Pass@16).

Table 2: Comparison on a small training dataset. Training on AIME-24 and evaluation on AIME-25. REINFORCE++ has lower training accuracy and higher held-out scores than GRPO.

	AIME-24 (Train)	AIME-25 (Test)	
Pass@N	N = 1	N = 1	N = 16
GRPO	**95.0**	0.0	0.4
REINFORCE++	71.0	**2.5**	**40.0**

The training curves in Figure 3 show faster fitting of the training questions by GRPO and a more gradual increase for REINFORCE++ .

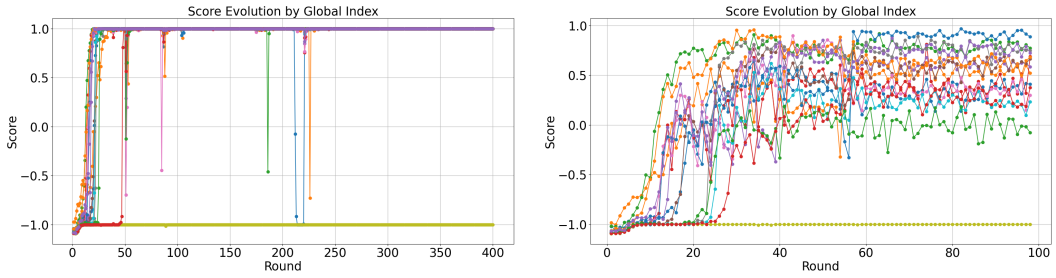


Figure 3: Training curves on a small prompt dataset. Left: GRPO with local normalization. Right: REINFORCE++ with global normalization. The curves show training dynamics; held-out scores appear in Table 2.

Logical Reasoning (K&K Puzzles) We also tested on the Knights and Knaves (K&K) puzzles (Xie et al., 2025; 2024), where difficulty increases with the number of characters in each puzzle. Figure 4 shows that while GRPO is competitive on two- and three-character puzzles, it has lower accuracy on the harder eight-character puzzles. REINFORCE++ outperforms GRPO on the evaluated tasks with four or more characters and achieves a higher average score (62.1 vs. 55.7).

RL from Zero Setting Finally, we applied reinforcement learning directly to a Qwen2.5-Math-Base model on MATH dataset splits (Guo et al., 2025). Table 3 shows that REINFORCE++ achieves higher Pass@8 on AIME-24 and AMC-23, with gains of 2.08 and 1.25 percentage points, respectively. On MATH-500, its Pass@1 is 72.00 compared with 73.00 for GRPO.

Table 3: Comparison on RL from Zero. REINFORCE++ achieves higher Pass@8 on AIME-24 and AMC-23; GRPO achieves higher Pass@1 on MATH-500.

	AIME-24 (OOD)	AMC-23 (OOD)	MATH-500 (ID)
Pass@N	N = 8	N = 8	N = 1
GRPO	18.96	59.22	**73.00**
REINFORCE++	**21.04**	**60.47**	72.00

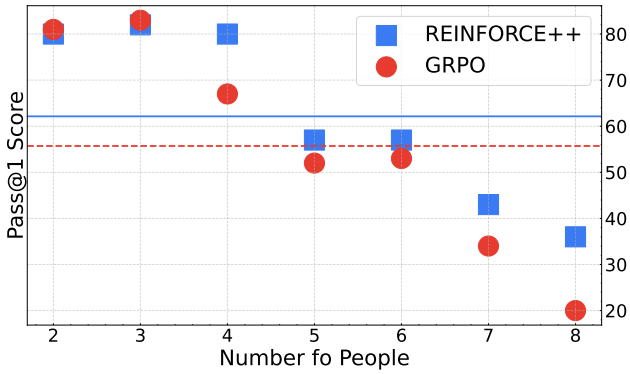


Figure 4: Comparison on logic benchmarks. REINFORCE++ achieves higher accuracy than GRPO on puzzles with four or more characters, including the eight-character evaluation.

4.3 Multi-step Reinforcement Learning

For multi-step reasoning, we use the zero-shot agent setup of Mai et al. (2025), in which Qwen2.5-Base-7B learns to use Python tools to solve mathematical problems. We evaluate REINFORCE++-Baseline with group sampling ($k > 1$) and group-mean reward centering in this setting (see Section 5).

Experimental Setup We adhere to the training and evaluation protocols of the ZeroTIR environment Mai et al. (2025). The backbone model is Qwen2.5-Base-7B, trained using OpenRLHF Hu et al. (2024) with datasets from ORZ Hu et al. (2025) and DAPO Yu et al. (2025). We assess performance on benchmarks including AIME 2024 Jia (2024), AIME-2025 math ai (2025), HMMT FEB-2024/2025, and CMIMC, using the `average@32` metric.

Experimental Result As shown in Table 4, REINFORCE++-Baseline achieves the highest five-benchmark mean accuracy (24.10) among the three methods. It outperforms both GRPO (22.58) and actor–critic PPO (21.85). The improvements in mean accuracy are 1.52 and 2.25 percentage points, respectively. The baseline variant leads on four of the five benchmarks, while GRPO leads on AIME-24. The evaluated baseline variant combines reward centering, global normalization, and separate KL regularization.

Table 4: Tool-use benchmark accuracy (average@32). REINFORCE++-Baseline achieves the highest five-benchmark mean and leads on four benchmarks.

Algorithm	AIME 24	AIME 25	HMMT 2025	HMMT 2024	CMIMC	**Avg**
GRPO	**31.66**	21.87	16.97	17.70	24.68	22.58
PPO	30.20	21.66	15.00	18.43	23.95	21.85
REINFORCE++-Baseline	30.83	**27.18**	**17.91**	**18.95**	**25.62**	**24.10**

5 Practical Considerations

5.1 Sampling and Reward Design

The two variants differ in how they allocate a response budget and construct the reward baseline. With B responses per rollout batch, single-response sampling can cover B prompts, whereas sampling k responses per prompt covers B/k prompts. The latter allocation provides within-prompt reward comparisons for the baseline variant.

For multi-turn tool use, group-mean centering compares each response with alternatives to the same prompt. A group with identical rewards has zero centered task reward before batch normalization, while a mixed-outcome group retains positive and negative reward differences. All responses remain

in the training batch after centering. Subtracting the group mean also removes any reward offset common to the group. For example, changing binary rewards from $0/1$ to $-1/1$ multiplies the centered rewards by two; subsequent global standardization cancels this factor when the numerical stabilizer is omitted. With token-weighted global moments, the batch mean of the centered rewards can be nonzero when response lengths differ.

For general-domain RLHF, the $k = 1$ configuration allocates the rollout budget to broader prompt coverage. Section 4.1 reports competitive preference-alignment scores and shorter responses than GRPO under this configuration. For tasks with multiple sampled responses, the baseline variant provides prompt-specific comparisons together with a scale shared across the batch. Reward design, response length, and the relative weight of KL regularization jointly determine the resulting training signal.

5.2 Evidence from Related Systems

Several subsequent studies examine global normalization in reasoning systems. Their experiments provide comparisons across reward schemes, model scales, and output-length constraints.

LitePPO (Liu et al., 2025c) combines group-mean centering, batch-level standard deviation, and token-level loss aggregation. Its normalization experiments favor the global standard deviation when paired with a group-level mean, while comparisons of full batch normalization vary with the reward scheme. These results support treating reward centering and scale estimation as separate design choices.

ScaleRL (Khatri et al., 2025) compares prompt-level, batch-level, and no variance normalization, reporting similar performance with a marginal advantage for batch-level normalization. Its larger-scale leave-one-out study includes runs with a 16,000 GPU-hour budget. ScaleRL uses batch-level normalization in its final training configuration.

DLER (Liu et al., 2025b) evaluates normalization under output truncation. In its DeepSeek-R1-7B experiment, batch-wise normalization preserves AIME-24 accuracy as response length decreases, whereas group-wise normalization shows declining accuracy. The reported difference occurs during training with reduced output lengths.

6 Conclusion

We present REINFORCE++, a critic-free framework that combines PPO clipping with global advantage normalization. The framework separates the choice of reward baseline from the scale used across prompts. Our scalar analysis characterizes finite-sample self-normalization and its convergence to population standardization under independent Gaussian sampling with a vanishing numerical stabilizer.

The two variants implement this framework for different sampling configurations:

1. REINFORCE++ uses globally normalized returns with a KL penalty incorporated into the reward. The $k = 1$ configuration provides one response per prompt, while $k > 1$ supports multiple sampled responses in reasoning tasks.
2. REINFORCE++-Baseline combines group-mean reward centering, global normalization, and a separate k_2 KL surrogate. It uses within-prompt reward comparisons while sharing normalization statistics across the rollout batch.

Experiments cover preference alignment, mathematical and logical reasoning, and multi-step tool use. The single-response variant scores 46.7 on Arena-Hard, compared with 46.8 for GRPO, with shorter mean responses. The reasoning experiments show higher held-out scores than GRPO across the reported AIME and AMC comparisons. In tool use, the baseline variant achieves the highest mean accuracy across five benchmarks, exceeding GRPO and PPO by 1.52 and 2.25 percentage points. Together, these results support global advantage normalization as a practical component of critic-free policy optimization across the evaluated tasks.

AI USE STATEMENT

Generative AI tools were used to assist manuscript writing and polish the language and presentation, including formatting, mathematical exposition, and proof revision. Experimental data and results were produced by the authors. The authors reviewed the AI-assisted revisions and take responsibility for the final manuscript and its accompanying artifacts.

ETHICS STATEMENT

Reinforcement learning is increasingly used to steer large language models by optimizing them against a reward signal, including verifiable, programmatic rewards (RLVR) for domains such as math, code, and tool use. The main potential positive impact of this line of work is improved scalability and reproducibility of post-training: verifiable rewards can reduce reliance on human labeling and make optimization targets easier to audit and replicate. The critic-free methods studied here aim to support this use of reinforcement learning through shared advantage scaling and prompt-specific reward centering. Potential risks include reward gaming when verifiers are incomplete or mis-specified, and dual-use concerns if improved tool-use or problem-solving capabilities are deployed without safeguards. We recommend evaluating beyond verifier scores (e.g., robustness and red-teaming), stress-testing verifiers for exploitability, and using conservative deployment controls (e.g., access restrictions and monitoring) for systems trained with RL.

REPRODUCIBILITY STATEMENT

Section 3 specifies the algorithms, and Section 4 describes the experimental settings. Appendix A specifies the response-wise loss reduction of the policy objective. Appendices B and C provide assumptions and derivations for normalization and KL regularization.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet stün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint*, 2024.

Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphael Marinier, Léonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, et al. What matters in on-policy reinforcement learning? a large-scale empirical study. *arXiv preprint arXiv:2006.05990*, 2020.

AI Anthropic. Introducing claude, 2023. URL `https://www.anthropic.com/news/introducing-claude`.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.

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

Jian Hu, Xibin Wu, Zilin Zhu, Weixun Wang, Dehao Zhang, Yu Cao, et al. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*, 2024.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

Maxwell Jia. Aime 2024 dataset card. https://huggingface.co/datasets/Maxwell-Jia/AIME_2024, 2024. Version 2024-03.

Devvrit Khatri, Lovish Madaan, Rishabh Tiwari, Rachit Bansal, Sai Surya Duvvuri, Manzil Zaheer, Inderjit S Dhillon, David Brandfonbrener, and Rishabh Agarwal. The art of scaling reinforcement learning compute for llms. *arXiv preprint arXiv:2510.13786*, 2025.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*, 2024.

Ziniu Li, Tian Xu, Yushun Zhang, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. Remax: A simple, effective, and efficient method for aligning large language models. *arXiv preprint arXiv:2310.10505*, 2023.

Kezhao Liu, Jason Klein Liu, Mingtao Chen, and Yiming Liu. Rethinking kl regularization in rlhf: From value estimation to gradient optimization. *arXiv preprint arXiv:2510.01555*, 2025a.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, Yejin Choi, et al. Dler: Doing length penalty right-incentivizing more intelligence per token via reinforcement learning. *arXiv preprint arXiv:2510.15110*, 2025b.

Zihe Liu, Jiashun Liu, Yancheng He, Weixun Wang, Jiaheng Liu, Ling Pan, Xinyu Hu, Shaopan Xiong, Ju Huang, Jian Hu, et al. Part i: Tricks or traps? a deep dive into rl for llm reasoning. *arXiv preprint arXiv:2508.08221*, 2025c.

Xinji Mai, Haotian Xu, Zhong-Zhi Li, Weinong Wang, Jian Hu, Yingying Zhang, Wenqiang Zhang, et al. Agent rl scaling law: Agent rl with spontaneous code execution for mathematical problem solving. *arXiv preprint arXiv:2505.07773*, 2025.

math ai. Aime 2025 dataset card. https://huggingface.co/datasets/math-ai/aime25, 2025. Version 2024-03.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.18290.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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

ByteDance Seed, Jiaze Chen, Tiantian Fan, Xin Liu, Lingjun Liu, Zhiqi Lin, Mingxuan Wang, Chengyi Wang, Xiangpeng Wei, Wenyuan Xu, et al. Seed1. 5-thinking: Advancing superb reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.13914*, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Yu Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Wei Shen and Chuheng Zhang. Policy filtration in rlhf to fine-tune llm for code generation. *arXiv preprint arXiv:2409.06957*, 2024.

Wei Shen, Guanlin Liu, Zheng Wu, Ruofei Zhu, Qingping Yang, Chao Xin, Yu Yue, and Lin Yan. Exploring data scaling trends and effects in reinforcement learning from human feedback. *arXiv preprint arXiv:2503.22230*, 2025.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Sai Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. Chatgpt for robotics: Design principles and model abilities. *Microsoft Auton. Syst. Robot. Res*, 2:20, 2023.

Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih Ghazi, and Ravi Kumar. On memorization of large language models in logical reasoning. *arXiv preprint arXiv:2410.23123*, 2024.

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

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025.

A Response-wise loss reduction and implementation

The objective in Eq. (1) uses a mean over valid tokens within each response, followed by a mean over responses in the batch. Response averaging assigns equal weight to each sampled response.

Let $m_{i,t} \in \{0, 1\}$ indicate a valid response token included in the policy loss, and let $n_i = \sum_t m_{i,t} > 0$. For B responses, the policy loss is

$$\boxed{L_{\mathrm{PG}}(\theta) = \frac{1}{B} \sum_{i=1}^{B} \left[\frac{1}{n_i} \sum_t m_{i,t}\, \ell_{i,t}(\theta) \right], \qquad \ell_{i,t}(\theta) = -f_{i,t}(\theta, \mathrm{sg}[\widehat{A}_{i,t}]).} \tag{9}$$

Here $\widehat{A}$ denotes A^{norm}, f is the clipped surrogate in Eq. (1), and sg denotes stop-gradient. Padding and positions outside the response loss mask do not enter the numerator or token count. The separate k_2 term in Eq. (8) uses the same reduction.

Response weighting. Each response has outer weight $1/B$, and each valid token within response i has weight $1/(Bn_i)$. This differs from dividing the sum of all token losses by the total number of valid tokens:

$$L_{\mathrm{token}} = \frac{\sum_{i,t} m_{i,t} \ell_{i,t}}{\sum_i n_i}. \tag{10}$$

The latter weights each response in proportion to its length. The two reductions agree when all responses have equal length. With a fixed number k of responses per prompt, batch averaging is also equivalent to averaging the per-response losses within each prompt and then across prompts.

Minibatches and accumulation. For a minibatch $\mathcal{M}$, replace B by $|\mathcal{M}|$ in Eq. (9). To recover the same mean when combining uneven microbatches or distributed shards, their loss sums must be weighted by their response counts. The normalization moments are associated with the rollout batch; changing the partition into minibatches does not change the definition of the already computed advantages.

Terminal reward and token returns. For the basic variant, write $d_{i,t} = \log(\pi_{\text{old}}/\pi_{ref})$ at the sampled token. The immediate reward can be written as

$$\widetilde{r}_{i,t} = \mathbf{1}\{t = T_i\} r(q_i, o_i) - \beta d_{i,t}. \tag{11}$$

With unit discount, its return is $\sum_{u=t}^{T_i} \widetilde{r}_{i,u} = A_{q_i,o_{i,t}}$ from Eq. (4). Thus, a terminal task reward contributes to the return at every preceding response position. For REINFORCE++-Baseline , the group-centered terminal reward in Eq. (6) is broadcast to response positions before global normalization.

B FINITE-SAMPLE PROPERTIES OF NORMALIZATION

We study rewards standardized by the mean and variance of the same sample. Let

$$r_i = \mu + \epsilon_i, \qquad \epsilon_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2), \qquad \sigma > 0, \tag{12}$$

and define $\bar{\epsilon} = N^{-1} \sum_i \epsilon_i$, $D_N^2 = N^{-1} \sum_i (\epsilon_i - \bar{\epsilon})^2$, and $Z_{i,N} = (\epsilon_i - \bar{\epsilon})/D_N$ for $N \geq 2$. The population-standardized target is ϵ_i/σ. We omit the numerical stabilizer for this analysis; $D_N > 0$ almost surely under the stated model.

Proposition B.1 (Finite-sample bound)**.** *For any nonconstant sample of size $N \geq 2$, $|Z_{i,N}| \leq \sqrt{N-1}$. Consequently, under the Gaussian model, $\mathbb{E}[Z_{i,N} \mid \epsilon_i]$ cannot equal ϵ_i/σ almost surely for finite N.*

Proof. Set $x_j = \epsilon_j - \bar{\epsilon}$, so $\sum_j x_j = 0$. By Cauchy–Schwarz,

$$x_i^2 = \left(\sum_{j \neq i} x_j \right)^2 \leq (N-1) \sum_{j \neq i} x_j^2. \tag{13}$$

Therefore $\sum_j x_j^2 \geq N x_i^2/(N-1)$, giving $Z_{i,N}^2 = N x_i^2 / \sum_j x_j^2 \leq N - 1$. The conditional expectation obeys the same bound, whereas $|\epsilon_i|/\sigma > \sqrt{N-1}$ has positive probability for a Gaussian variable. Equality with the standardized target cannot hold almost surely. □

Conditional moments of the numerator and denominator. The centered numerator satisfies

$$\mathbb{E}[\epsilon_i - \bar{\epsilon} \mid \epsilon_i] = \left(1 - \frac{1}{N}\right) \epsilon_i. \tag{14}$$

Using $D_N^2 = N^{-1} \sum_j \epsilon_j^2 - \bar{\epsilon}^2$ and independence,

$$\mathbb{E}[\bar{\epsilon}^2 \mid \epsilon_i] = \frac{\epsilon_i^2 + (N-1)\sigma^2}{N^2}, \tag{15}$$

$$\mathbb{E}[D_N^2 \mid \epsilon_i] = \frac{(N-1)^2}{N^2} \sigma^2 + \frac{N-1}{N^2} \epsilon_i^2. \tag{16}$$

These identities describe how the included observation affects the normalization statistics. Proposition B.1 establishes the finite-sample discrepancy through a direct bound on the ratio.

An exact two-response example. For $N = 2$, $D_2 = |\epsilon_1 - \epsilon_2|/2$ and $Z_{1,2} = \text{sign}(\epsilon_1 - \epsilon_2)$ almost surely. Thus,

$$\mathbb{E}[Z_{1,2} \mid \epsilon_1 = x] = 2\Phi(x/\sigma) - 1, \tag{17}$$

where Φ is the standard normal cumulative distribution function. This bounded, nonlinear function differs from the linear population-standardized value x/σ. It illustrates how a small group changes the reward scale through self-normalization.

Proposition B.2 (Large-sample limit)**.** *Under the Gaussian model, for any fixed i, $Z_{i,N} \to \epsilon_i/\sigma$ almost surely and in L^1 as $N \to \infty$. In particular,*

$$\mathbb{E}\left[|\mathbb{E}[Z_{i,N} \mid \epsilon_i] - \epsilon_i/\sigma| \right] \longrightarrow 0. \tag{18}$$

Proof. The strong law gives $\bar{\epsilon} \to 0$ and $D_N^2 \to \sigma^2$ almost surely, establishing the first limit. By exchangeability and $\sum_{j=1}^{N} Z_{j,N}^2 = N$, we have $\mathbb{E}[Z_{i,N}^2] = 1$ for every N. This uniform second-moment bound implies uniform integrability, so the almost-sure convergence also holds in L^1. Conditional Jensen's inequality gives

$$\mathbb{E}\left[|\mathbb{E}[Z_{i,N} - \epsilon_i/\sigma \mid \epsilon_i]|\right] \leq \mathbb{E}\left[|Z_{i,N} - \epsilon_i/\sigma|\right] \to 0. \tag{19}$$

□

Numerical stabilizer. With a fixed stabilizer $\eta > 0$, $(\epsilon_i - \bar{\epsilon})/(D_N + \eta)$ converges instead to $\epsilon_i/(\sigma + \eta)$. The unstabilized population-standardization limit applies when the stabilizer is omitted or tends to zero. The finite-sample bound remains valid with a nonnegative stabilizer for nonconstant samples; constant samples have zero numerator and return zero when the stabilizer is positive.

Implications for global normalization. The propositions isolate the effect of normalization sample size: estimated moments approach their population values as the number of independent observations grows. A shared batch scale can use observations across many prompt groups, rather than estimating a separate scale from each small group. For grouped or token-level data, the effective sample size depends on the number of independent trajectories and their dependence structure.

Relationship to baselines and gradients. For an unscaled group mean, $\mathbb{E}[r_i - \bar{r}_q \mid r_i, q] = (1 - 1/k)(r_i - \mu_q)$ when responses are conditionally independent with mean reward μ_q. This is the same factor appearing in Eq. (24). The propositions describe convergence to a standardized scalar target. Policy-gradient expectations additionally involve the joint distribution of these normalized rewards and the policy score function.

C KL REGULARIZATION AS A GRADIENT SURROGATE

For a fixed context, write $p_\theta(y) = \pi_\theta(y \mid h)$, $q(y) = \pi_{\text{ref}}(y \mid h)$, and $z(y) = \log[p_\theta(y)/q(y)]$. Assume common support and sufficient regularity to interchange differentiation and expectation. The reverse KL is

$$D_{\text{KL}}(p_\theta \| q) = \mathbb{E}_{y \sim p_\theta}[z(y)], \qquad \nabla_\theta D_{\text{KL}}(p_\theta \| q) = \mathbb{E}_{p_\theta}[z(y) \nabla_\theta \log p_\theta(y)], \tag{20}$$

using $\mathbb{E}_{p_\theta}[\nabla_\theta \log p_\theta] = 0$.

Value estimators and differentiated losses. The common expressions are

$$k_1(z) = z, \qquad k_2(z) = \tfrac{1}{2} z^2, \qquad k_3(z) = e^{-z} - 1 + z. \tag{21}$$

Both k_1 and k_3 have expectation $D_{\text{KL}}(p_\theta \| q)$ under on-policy sampling, since $\mathbb{E}_{p_\theta}[e^{-z}] = 1$. When sampled actions are treated as fixed during automatic differentiation, their pointwise gradients differ:

$$\begin{aligned} \nabla_\theta k_1 &= \nabla_\theta \log p_\theta, \\ \nabla_\theta k_2 &= z \nabla_\theta \log p_\theta, \\ \nabla_\theta k_3 &= (1 - q/p_\theta) \nabla_\theta \log p_\theta. \end{aligned} \tag{22}$$

The on-policy expectation of the k_2 pointwise gradient is exactly Eq. (20). This identity motivates the separate KL term in REINFORCE++-Baseline (Liu et al., 2025a). It is distinct from differentiating the entire expectation $\mathbb{E}_{p_\theta}[k_2]$, which also differentiates the sampling distribution.

Under the same fixed-context assumptions, the expected pointwise gradient of k_1 is zero. The k_1 expression can instead enter the reward and act as a detached score-function coefficient, as in the returns used by the basic variant. The expected pointwise gradient of k_3 equals $-\mathbb{E}_q[\nabla_\theta \log p_\theta]$, the forward-KL gradient. Thus, its on-policy mean estimates reverse KL, while its mean pointwise gradient follows forward KL.

Numerical sensitivity. The k_3 pointwise gradient contains the importance ratio q/p_θ, which can become large when the current policy assigns low probability to actions favored by the reference. Computing this ratio as $\exp(-z)$ can also overflow for sufficiently negative z. The k_2 surrogate uses the log ratio in place of this exponential ratio. Its gradient magnitude and variance depend on the joint distribution of the log ratio and the score function.

Rollout reuse. The on-policy identities apply when the sampled actions follow the current policy. For actions sampled from p_{old}, a fixed-context importance-weighted gradient satisfies

$$\mathbb{E}_{p_{\mathrm{old}}}\left[\frac{p_\theta(y)}{p_{\mathrm{old}}(y)} z(y) \nabla_\theta \log p_\theta(y)\right] = \nabla_\theta D_{\mathrm{KL}}(p_\theta \| q). \tag{23}$$

The practical separate loss in Eq. (8) is the unweighted k_2 surrogate on the rollout data. Its expected pointwise gradient equals the reverse-KL gradient at the on-policy point and changes as the policy moves away from the rollout policy. For autoregressive policies, the fixed-context statement concerns the conditional next-token distribution.

D Design implications and sampling budgets

Local baseline and global scale. For a prompt with $k > 1$ responses, let $C_{q,i} = r_i - k^{-1} \sum_j r_j$. Then

$$C_{q,i} = \frac{k-1}{k}\left(r_i - \frac{1}{k-1}\sum_{j \neq i} r_j\right) = \frac{k-1}{k} A_i^{\mathrm{RLOO}}. \tag{24}$$

The group baseline compares a response with alternatives to the same prompt; global scaling supplies a shared scale across prompts. Homogeneous groups have zero centered task-reward signal. This separates reward centering from prompt-specific standard-deviation normalization.

Response budget. At fixed response count B, single-response sampling covers up to B prompts; grouped sampling covers B/k. Larger groups trade prompt coverage for exploration of each prompt. Runtime and memory also depend on response length, batching, optimizer state, and rollout implementation.

Evaluation and attribution. The tool-use metric average@32 averages success across 32 sampled attempts; it differs from pass@32, which counts success if at least one attempt succeeds. The reported tool-use mean is an unweighted average across five benchmarks. The improvements of 1.52 and 2.25 percentage points over GRPO and PPO are measured for the complete training configuration. The evaluated configuration combines group-mean reward centering, global advantage normalization, a separate KL surrogate, and response-wise loss averaging.