

# Sample More to Think Less: Group Filtered Policy Optimization for Concise Reasoning

Vaishnavi Shrivastava<sup>m</sup> Ahmed Awadallah<sup>m</sup> Vidhisha Balachandran<sup>m</sup>  
 Shivam Garg<sup>m</sup> Harkirat Behl<sup>m</sup> Dimitris Papailiopoulos<sup>m,ω</sup>

<sup>m</sup>Microsoft Research  
<sup>ω</sup>University of Wisconsin-Madison

## Abstract

1 Large language models trained with reinforcement learning on verifiable rewards  
 2 often inflate response length—trading brevity for accuracy. While longer reasoning  
 3 can help on hard problems, many extra tokens are filler: verbose text making little  
 4 progress. We introduce GFPO (*Group Filtered Policy Optimization*), which curbs  
 5 this length explosion by sampling larger groups per problem and only training  
 6 on responses filtered by (1) length and (2) token efficiency (reward per token).  
 7 By sampling *more* during training-time, GFPO teaches models to think *less* at  
 8 inference-time. On Phi-4-reasoning, GFPO cuts GRPO’s length inflation by up to  
 9 85% across STEM and coding benchmarks (AIME 24/25, GPQA, Omni-MATH,  
 10 LiveCodeBench) while preserving accuracy. We further propose Adaptive Diffi-  
 11 culty GFPO, which allocates more training exploration to harder problems using  
 12 real-time difficulty estimates, yielding better efficiency-accuracy trade-offs on  
 13 challenging questions. GFPO demonstrates that modest extra training compute can  
 14 deliver substantial test-time savings—an effective recipe for efficient reasoning.

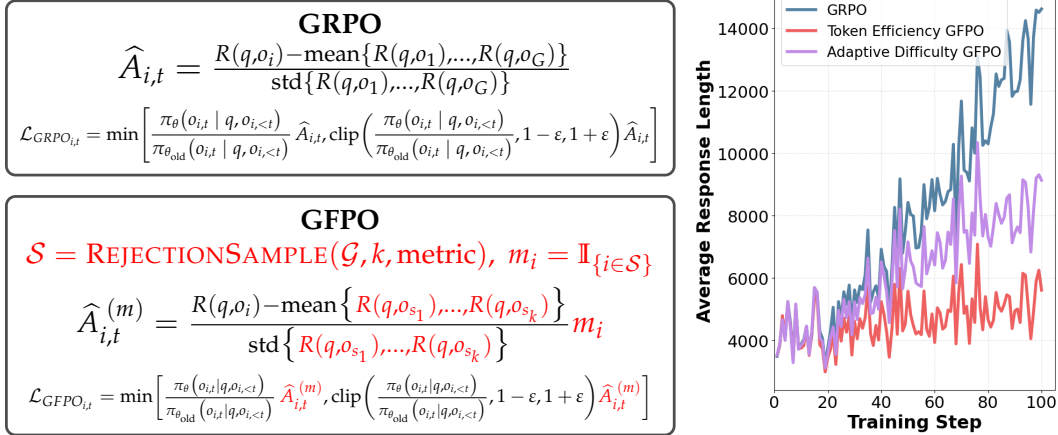


Figure 1: **Left:** GFPO introduces simple yet powerful modifications to GRPO: sample more responses during training ( $\uparrow G$ ), rank them by a target attribute (e.g., length, token efficiency), and learn only from the top- $k$ —setting the advantages of the rest to zero. This selective learning functions as implicit reward shaping, steering the policy toward desired behaviors. **Right:** When optimizing for length or token efficiency, GFPO curbs GRPO’s length inflation—letting the model *think less* at inference-time by *sampling more* at training-time—while maintaining its core reasoning capabilities.

# 1 Introduction

Reinforcement learning from verifier rewards (RLVR) methods such as GRPO (Shao et al., 2024) and PPO (Schulman et al., 2017) have been pivotal for test-time scaling—enabling models like O3 (OpenAI, 2025) and DeepSeek-R1 (Guo et al., 2025) to “think longer” and achieve state-of-the-art results on challenging reasoning tasks such as AIME and IMO. Yet longer chains are not always better: prior work shows that long responses don’t correlate with correct answers—shorter responses can even be more accurate. For instance, Balachandran et al. (2025) report that DeepSeek-R1 produces responses nearly 5× longer than Claude 3.7 Sonnet on AIME 25 with no accuracy gain, while Hassid et al. (2025) find that QwQ-32B’s shortest responses outperform random ones by 2% while using 31% fewer tokens.

One may suspect that longer responses simply reflect harder problems. However, by comparing correct and incorrect responses to the same AIME 25 questions with Phi-4-reasoning-plus (Abdin et al., 2025), we find the opposite: in 72% of cases, the longer responses are more likely to be wrong. This suggests that verbosity is not just a byproduct of difficulty but a distinct failure mode.

Works such as Dr. GRPO (Liu et al., 2025) and DAPO (Yu et al., 2025) apply token-level normalization to address this failure mode. However, even with these methods, response length for Phi-4-reasoning-plus balloons from 4k to 14k tokens within 100 GRPO steps. We hypothesize that while normalization penalizes long incorrect outputs, it also amplifies rewards for long correct ones—reinforcing verbosity in models already SFTed for step-by-step reasoning (Abdin et al., 2025; Guo et al., 2025).

Motivated by these observations, our goal is to train efficient reasoning models: ones that preserve GRPO’s accuracy while producing far shorter reasoning chains. Our contributions are as follows:

- **GFPO (Group Filtered Policy Optimization):** A variant of GRPO that samples larger groups of candidate chains to increase exposure to desirable outputs, filters them based on a target metric, and learns only from the filtered subset. GFPO optimized for response length reduces GRPO’s length inflation by 46–71% across AIME 24/25, GPQA, Omni-MATH, and LiveCodeBench, with no loss in accuracy.
- **Token Efficiency:** Defined as the ratio of reward to response length—allows longer chains when justified by higher rewards. Optimizing for token efficiency cuts length inflation by 71–85%.
- **Adaptive Difficulty GFPO:** A dynamic variant of GFPO that allocates more exploration to hard problems using unsupervised difficulty estimates, striking a better balance between efficiency and accuracy.
- **Out-of-Distribution Generalization:** We demonstrate that GFPO preserves accuracy while curbing response length even for out-of-distribution tasks.

Sampling more at training time enables shorter, more efficient reasoning at deployment—offering a simple, effective solution to the length inflation inherent in RLVR-trained models.

## 2 Group Filtered Policy Optimization

Group Relative Policy Optimization (GRPO; Shao et al. (2024)) simplifies Proximal Policy Optimization (PPO; Schulman et al. (2017)) by removing the value model and instead using the average reward of sampled responses as a baseline, while retaining PPO’s clipped surrogate objective.

We propose *Group Filtered Policy Optimization* (GFPO), a simple yet effective method for targeted policy optimization of desirable response properties. GFPO samples a larger group of candidate responses per question, broadening the response pool to include more candidates with desirable traits, and then explicitly filters for these traits when computing the policy gradient. While it may seem natural to directly encode desirable attributes such as brevity or informativeness into the scalar reward, doing so for multiple traits can be challenging, especially when correctness must already be captured.

Data filtration instead serves as an implicit, flexible form of reward shaping—akin to iterative self-improvement methods that use selective sampling to amplify specific model behaviors (Zelikman et al., 2022). After this explicit filtering step isolates the preferred responses, standard rewards are then used solely to compute relative advantages within the selected group. Thus, GFPO optimizes for multiple desirable properties (e.g., length and accuracy) simultaneously, without requiring complex

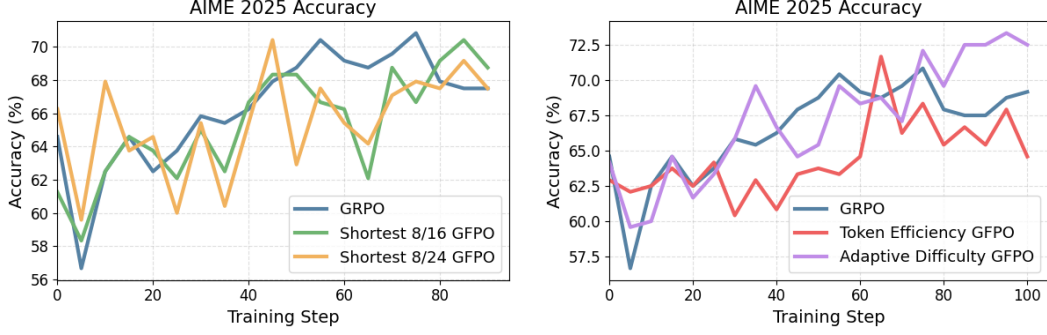


Figure 2: **Comparison of GFPO and GRPO on AIME 25 accuracy** ( $n = 8$  samples) during training. GFPO variants reach the same peak performance as GRPO (blue).

reward engineering. Since our goal is to reduce the response length inflation in RL, we focus on using GFPO to optimize for shorter responses while matching GRPO’s accuracy.

Given a question  $q$ , we sample a large set of responses  $\mathcal{G} = \{o_1, \dots, o_G\}$  from the current policy. Rather than training equally on all responses, GFPO applies a selection step based on a user-specified metric to filter a subset of size  $k$  of the most desirable responses to train on. We compute a metric score for each response and sort accordingly, selecting the top- $k$  responses to form the retained subset  $\mathcal{S} \subseteq \mathcal{G}$ . We define a binary mask  $m \in \{0, 1\}^G$ , where  $m_i = 1$  indicates a selected response and  $m_i = 0$  indicates a rejected response.

Formally, we define the GFPO objective<sup>1</sup> as:

$$\mathcal{J}_{\text{GFPO}}(\theta) = \mathbb{E}_{q \sim P(Q), \{o_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(O|q)} \frac{1}{\sum_{i=1}^G |o_i|} \sum_{i=1}^G \sum_{t=1}^{|o_i|} \min(r_{i,t} \hat{A}_{i,t}^{(m)}, \text{clip}(r_{i,t}, 1 - \varepsilon, 1 + \varepsilon) \hat{A}_{i,t}^{(m)}) - \beta \mathcal{D}_{KL}(\pi_{\theta} \parallel \pi_{\theta_{\text{old}}}) + \gamma \text{Entropy}(\pi_{\theta}) \quad (1)$$

where

$$\mathcal{S}, m = \text{REJECTIONSAMPLE}(\mathcal{G}, k, \text{metric}, \text{order}), m_i = \mathbb{I}_{\{i \in \mathcal{S}\}}$$

$$\hat{A}_{i,t}^{(m)} = \frac{R(q, o_i) - \frac{1}{k} \sum_{j \in \mathcal{S}} R(q, o_j)}{\sqrt{\frac{1}{k} \sum_{j \in \mathcal{S}} (R(q, o_j) - \frac{1}{k} \sum_{p \in \mathcal{S}} R(q, o_p))^2}} m_i, \quad r_{i,t} = \frac{\pi_{\theta}(o_{i,t} | q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} | q, o_{i,<t})}$$

and  $\beta \mathcal{D}_{KL}(\pi_{\theta} \parallel \pi_{\theta_{\text{old}}})$  denotes the KL penalty.

We normalize the advantages for responses in the selected subset  $\mathcal{S}$  using the mean and standard deviation of the response-level rewards in  $\mathcal{S}$ . This enables meaningful comparisons among responses already exhibiting the desired property, ensuring GFPO prioritizes the highest-reward responses within the filtered subset. Responses not in  $\mathcal{S}$  receive zero advantage, effectively excluding them from influencing policy updates. Thus, GFPO’s primary intervention is at the level of advantage estimation, making it compatible with any GRPO variant such as DAPO (Yu et al., 2025), Dr. GRPO (Liu et al., 2025), or GRPO with the Dual-Clip PPO loss (Ye et al., 2020). Although GFPO incurs higher training-time compute by sampling more responses, this cost is partially offset as the learned policy produces shorter responses than GRPO.

While GFPO is general-purpose and can accommodate various scoring metrics, our experiments specifically leverage metrics aimed at reducing response length inflation:

- **Response Length:** Training on short responses directly encourages brevity.
- **Token Efficiency** (reward/length): Training on highly token-efficient responses encourages succinctness, but still allows longer responses if sufficiently “justified” by proportionately higher rewards.

<sup>1</sup>Note we use the DAPO token-level loss aggregation for both GFPO and GRPO which is the default choice in ver1. We employ a slightly modified version of the clipped surrogate policy gradient loss introduced in prior work (Li et al., 2025), which reduces training instabilities caused by negative advantages and large policy ratios.

94 Other metrics—such as factuality, diversity, or external quality scores—could also be integrated into  
 95 GFPO to optimize different attributes of interest.

96 **2.1 Adaptive Difficulty GFPO.** We introduce Adaptive Difficulty GFPO, which allocates more  
 97 training signal to harder questions. At each step, we estimate difficulty from the average reward of  
 98 sampled responses—lower averages indicate higher difficulty.

99 To scale the number of retained responses  $k$ , we maintain a streaming summary of prompt difficulties  
 100 using a lightweight t-digest, which approximates quartiles over past rewards. New questions are  
 101 bucketed into difficulty levels, and assigned  $k = 4$  (easy),  $k = 6$  (medium), or  $k = 8$  (hard/very hard)  
 102 out of 16 sampled.<sup>2</sup> The number of buckets and  $k$  per bucket are hyperparameters.

103 This curriculum sharpens filtering on easy prompts while encouraging exploration on harder ones,  
 104 reducing verbosity where correctness is already high and preserving accuracy on challenging cases.  
 105 To our knowledge, this is the first RLVR method that adapts group size based on question difficulty.

### 106 3 Setup

107 **Model.** We build on Phi-4-reasoning (Abdin et al., 2025), a 14B-parameter Phi-4 model (Abdin et al.,  
 108 2024) extensively SFTed on synthetic o3-mini reasoning traces in STEM, but never RL-trained. We  
 109 refer to this as the SFT baseline.

110 **Baseline.** We compare our GFPO trained models with Phi-4-reasoning-plus (Abdin et al., 2025),  
 111 which is trained with GRPO and DAPO’s token-level loss aggregation. We refer to this as the GRPO  
 112 baseline. We match its setup, including a slightly modified clipped surrogate objective for training  
 113 stability (Section 2).

114 **Dataset.** RL training uses 72k math problems from the same corpus as Abdin et al. (2025). With 100  
 115 training steps and batch size 64, models see only 6.4k problems—identical to the GRPO baseline.

116 **Reward Function.** We adopt the GRPO baseline reward: a weighted sum of (i) a length-aware binary  
 117 accuracy reward  $R_{\text{acc}}$  and (ii) a 5-gram repetition penalty  $R_{\text{rep}}$ :

$$R = w_{\text{acc}} \text{LENGTHSCALE}(R_{\text{acc}}) + w_{\text{rep}} R_{\text{rep}}, \quad R \in [-1, 1]. \quad (2)$$

118 Accuracy is 0/1 based on extracted final answers, with GPT-4o fallback if regex extraction fails; long  
 119 correct responses are downweighted, and formatting violations receive the minimum reward. This  
 120 length penalty, however, is insufficient to prevent GRPO’s length inflation—motivating GFPO.

121 **Training Configuration.** We train with ver1 (Sheng et al., 2024) on 32 H100s, global batch size 64,  
 122 for 100 steps. We use Adam with learning rate  $1 \times 10^{-7}$ , cosine warmup (10 steps), temperature  
 123  $T = 1.0$ , KL regularization ( $\beta = 0.001$ ), and entropy coefficient ( $\gamma = 0.001$ ). Models are trained  
 124 with a 32k context, reserving 1k tokens for the prompt. GRPO uses group size  $G = 8$ . GFPO  
 125 increases  $G \in \{8, 16, 24\}$  to expose the model to more candidates, but retains only  $k \leq 8$  responses  
 126 for policy gradients, ensuring a fair comparison.

127 **Evaluation.** We evaluate on: AIME 25/24 (AIME, 2025, 2024) (32 samples), GPQA (Rein et al.,  
 128 2024) (5 samples), Omni-MATH (Gao et al., 2025) (1 sample), and LiveCodeBench (8/24–1/25) (Jain  
 129 et al., 2024) (3 samples). Responses are sampled at  $T = 0.8$  with 32k max length (1k prompt  
 130 tokens). Final answers are extracted via regex, with GPT-4o fallback. LiveCodeBench tests OOD  
 131 generalization to code, which is unseen during RL training.

132 We report **pass@1 accuracy**, raw response length  $L$ , and **excess length reduction (ELR)**:

$$ELR = \frac{L_{\text{GRPO}} - L_{\text{GFPO}}}{L_{\text{GRPO}} - L_{\text{SFT}}}. \quad (3)$$

133 Statistical significance is tested using the Wilcoxon signed-rank test (Wilcoxon, 1992) over per-  
 134 question accuracy differences.

## 135 4 Results

136 **4.1 GFPO Reduces Length by Sampling More and Retaining Less.** An initial question is whether  
 137 rejection sampling alone, without increasing sampled responses, suffices to shorten reasoning chains.

---

<sup>2</sup>During a short warmup period, all questions use  $k = 8$  to avoid unstable estimates.

|                     | AIME 25       | AIME 24       | GPQA          | Omni-MATH     | LiveCodeBench | Average     |              |               |
|---------------------|---------------|---------------|---------------|---------------|---------------|-------------|--------------|---------------|
|                     | % Len Inf (↓) | % Len Inf (↓) | % Len Inf (↓) | % Len Inf (↓) | % Len Inf (↓) | Acc         | Avg Len      | % Len Inf (↓) |
| SFT                 | N/A           | N/A           | N/A           | N/A           | N/A           | 69.2        | 9.5k         | N/A           |
| GRPO                | 0.0           | 0.0           | 0.0           | 0.0           | 0.0           | 72.1        | 13k          | 0.0           |
| 6 of 8              | 1.8           | 9.5           | 11.5          | -5.5          | 7.0           | 72.7        | 12.9k        | 4.8           |
| 8 of 16             | 23.8          | 33.0          | 23.7          | 31.5          | 36.5          | <b>73.4</b> | 12k          | 29.7          |
| 6 of 16             | 25.6          | 35.6          | 38.8          | 43.7          | <b>37.2</b>   | 72.3        | 11.8k        | 36.2          |
| 4 of 16             | <b>38.0</b>   | <b>46.8</b>   | <b>45.7</b>   | <b>47.3</b>   | 43.2          | 72.0        | <b>11.5k</b> | <b>44.2</b>   |
| 8 of 24             | <b>54.4</b>   | 52.7          | 52.2          | 51.9          | <b>59.4</b>   | 71.7        | 11.1k        | 54.1          |
| 6 of 24             | 41.0          | 44.9          | 48.6          | 58.2          | 42.7          | 72.2        | 11.4k        | 47.1          |
| 4 of 24             | 46.1          | <b>59.8</b>   | <b>57.3</b>   | <b>71.0</b>   | 57.0          | <b>72.3</b> | <b>11k</b>   | <b>58.2</b>   |
| Token Efficiency    | <b>70.9</b>   | <b>84.6</b>   | <b>79.7</b>   | <b>82.6</b>   | <b>79.7</b>   | 71.7        | <b>10.2k</b> | <b>79.5</b>   |
| Adaptive Difficulty | 50.8          | 52.9          | 41.7          | 35.1          | 49.4          | <b>72.9</b> | 11.4k        | 46.0          |

Table 1: **Pass@1 Accuracy, Response Lengths, and Length Inflation Reduction.** GFPO matches GRPO accuracy (no statistically significant differences under the Wilcoxon signed-rank test) while cutting length inflation across all benchmarks. Sampling more responses is key, and lowering the  $k/G$  ratio is an effective lever for controlling length. Token Efficiency achieves the largest average reduction (79.5%) at GRPO-level accuracy and Adaptive Difficulty surpasses shortest  $k/G$  at equivalent compute. On LiveCodeBench (out-of-distribution, coding), GRPO inflates length without accuracy gains, whereas GFPO reduces length and sometimes improves accuracy (e.g., 8/16, 4/24). Pass@1 is computed over 32 (AIME 25/24), 5 (GPQA), 1 (Omni-MATH), and 3 (LCB) samples. See Appendix Table 2 for per-dataset pass@1 and response lengths.

138 To examine this, we evaluate Shortest 6/8 GFPO, which retains the six shortest responses from a  
139 group of eight. While accuracy remains comparable to GRPO on AIME 25/24, GPQA, and Omni-  
140 MATH, length reductions are minimal (1.8–11.5%) and even negative on Omni-MATH (+5.5%).  
141 This indicates that subsampling within small response groups offers little efficiency benefit.

142 Substantial improvements emerge once the group size is increased. With Shortest 8/16 GFPO, which  
143 filters the shortest half of 16 candidates, excess length is reduced by 24–37% across benchmarks  
144 without statistically significant accuracy loss. Further decreasing the number of retained responses  
145 strengthens this effect: Shortest 6/16 and 4/16 achieve an additional 2–22% reduction relative to  
146 8/16. Scaling the group size amplifies these gains—for example, increasing from 8/16 to 8/24 yields  
147 20–30% additional reduction, while 4/24 results in an additional 4% reduction over 8/24 (Table 1).

148 Taken together, these results indicate that the decisive factor is the retention fraction  $k/G$  (Figure 7).  
149 Decreasing this fraction—either by reducing  $k$  or increasing  $G$ —consistently shortens reasoning  
150 chains. 4/16 and 6/24 both retain 25% of responses, and their length reductions are nearly identical—confirming that  $k/G$  is the key factor. Sampling from a larger group offers only a slight  
151 additional benefit, as seen with 6/24. Beyond a point, however, returns diminish: moving from 8/24  
152 to 4/24 yields only marginal additional gains. The strongest reductions are observed at retention  
153 fractions of about 25–33%.  
154

155 **4.2 Reinforcing Token Efficiency.** Reducing the retention ratio  $k/G$  alone eventually stalls—beyond  
156 a certain group size, it fails to deliver meaningfully shorter chains. To break this ceiling, we introduce  
157 Token Efficiency GFPO, which ranks responses by reward-per-token ( $R_i/|o_i|$ )—favoring longer  
158 chains only when their rewards justify the added cost. Token Efficiency GFPO filters for high  
159 reward-per-token responses—typically short correct chains, plus some long correct and long incorrect  
160 ones. Within this set, short correct chains receive the strongest positive gradients, long correct ones  
161 are modestly penalized, and long incorrect ones are sharply cut back, providing more direct length  
162 control than shortest- $k$ , which relies on the KL penalty to implicitly suppress late-token probabilities.

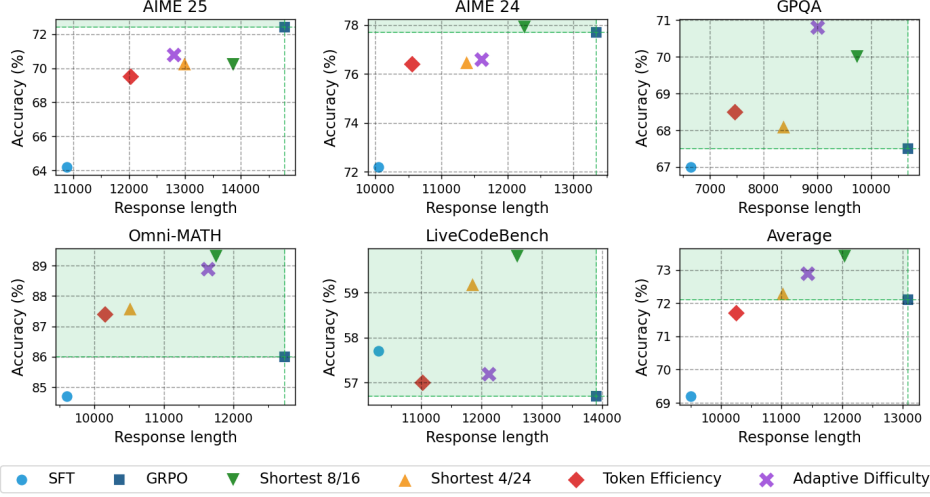


Figure 3: **Pareto Trade-off Between Accuracy and Response Length.** For all benchmarks except AIME 25, at least one GFPO variant strictly dominates GRPO—achieving both higher accuracy and shorter responses (green region above and to the left of GRPO). For AIME 25, GRPO attains the highest accuracy, but several GFPO variants, while taking non-significant accuracy dips, remain Pareto-optimal because their responses are shorter, and no other method is simultaneously more accurate and more concise. On average, Shortest 4/24, Adaptive Difficulty, and Shortest 8/16 are strictly Pareto-superior to GRPO with Token Efficiency close behind.

With  $k = 8, G = 16$ , this method yields the largest length reductions across tasks—70.9% (AIME 25), 84.6% (AIME 24), 79.7% (GPQA), 82.6% (Omni-MATH), and 79.7% (LiveCodeBench)—outperforming shortest- $k$  at similar or smaller  $G$  (Table 1). These gains come with minor tradeoffs: higher variance in training curves (Figure 2) and small, non-significant drops in accuracy. Still, Token Efficiency GFPO consistently delivers the sharpest token savings without compromising accuracy—showing reward-per-token to be a powerful proxy for concise reasoning.

**4.3 Adaptive Difficulty GFPO.** Beyond intelligent sampling through improved rejection metrics, we introduce Adaptive Difficulty GFPO, which adjusts the retained group size  $k$  by question difficulty—allocating more training resources to harder questions. We estimate question difficulty using the average reward of responses per question, efficiently compute problem difficulty quartiles at each training step, and categorize questions into four difficulty buckets: very hard (bottom 25%), hard (25–50%), medium (50–75%), and easy (top 25%). For these categories, we retain 8, 8, 6, and 4 shortest responses (from  $G=16$  samples), respectively. This yields an average  $k = 6.5$ , making Shortest 6/16 GFPO a natural baseline for comparison.

Adaptive Difficulty GFPO achieves stronger excess length reductions than Shortest 6/16 on AIME 25 (51% vs. 26%), AIME 24 (53% vs. 36%), GPQA (42% vs. 39%), and LiveCodeBench (46% vs. 36%) though Shortest 6/16 is more effective on Omni-MATH (44% vs. 35%). Even against the more aggressive Shortest 4/16, it performs better on AIME 25 (51% vs. 38%), AIME 24 (53% vs. 47%), and LiveCodeBench (49% vs. 43%) (Table 1). It also delivers the highest accuracy on GPQA (71%) and on the hardest AIME 25 quartile (27%) compared to GRPO and other GFPO variants (Figure 4b).

**4.4 Out-of-Distribution Effects of GFPO.** Our RL training recipe is geared towards enhancing mathematical reasoning performance. To investigate potential adverse effects of GFPO’s bias toward shorter responses, we assess out-of-distribution generalization on the LiveCodeBench coding benchmark. Note that coding is not a part of our RL training set.

GRPO inflates response length even out-of-distribution—outputs grow from 10.3k tokens (SFT) to 13.9k, while accuracy stagnates (57% vs. 58%) (Table 2). This verbosity is undesirable, especially without accuracy gains. GFPO counters this: Token Efficiency reduces excess length by 80%, and Shortest 8/24 trims 57% while modestly improving accuracy to 59% (vs. 58% SFT, 57% GRPO). GFPO not only reins in unnecessary length but can also enhance out-of-distribution generalization.

**4.5 Accuracy-Length Pareto Comparison.** Figure 3 shows the accuracy–length frontier. On four of five benchmarks, at least one GFPO variant is strictly *Pareto-superior* to GRPO (green

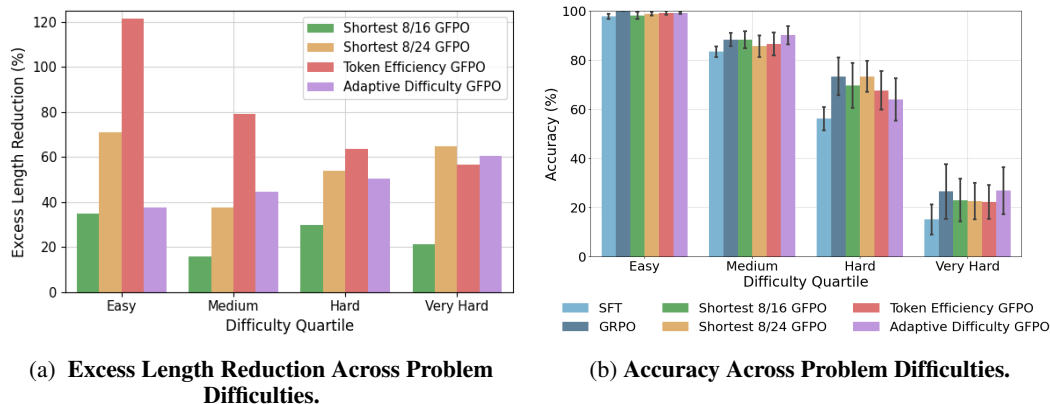


Figure 4: **Excess Length Reductions and Accuracy Across AIME 25 Problem Difficulties.** (a) GFPO reduces excess length across all difficulties. Token efficiency has the strongest overall reductions—with outputs more brief than the SFT model on easy questions. Shortest 8/24 has the best reductions on very hard questions. (b) Adaptive Difficulty and Shortest 8/24 have the best accuracies.

region), demonstrating that GFPO can yield both shorter and more accurate answers. Even on AIME 25, where GRPO is slightly more accurate, GFPO variants remain on the Pareto front by offering meaningful length reductions without significant accuracy loss. Aggregated results (bottom-right) highlight Shortest 4/24, Adaptive Difficulty, and Shortest 8/16 as the most consistently concise and accurate, with Token Efficiency trailing in accuracy by a narrow margin.

## 5 Analysis

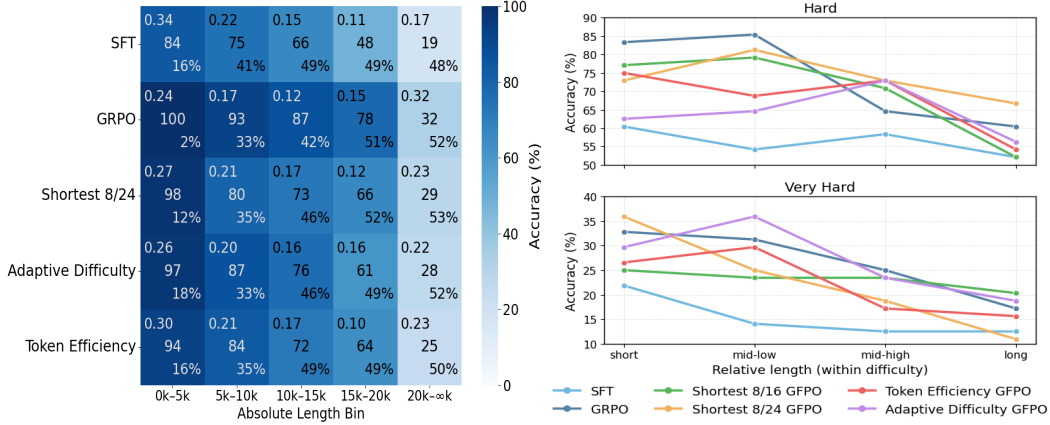
We analyze performance of GFPO tuned models on AIME 2025 by measuring question difficulty as  $1 - \text{SFT accuracy}$ , which captures how challenging each problem is for the base SFT model prior to RL. Problems are partitioned into quartiles (easy–very hard) to study how GFPO affects length and accuracy across difficulty and the accuracy of long responses at fixed difficulty. We also examine which parts of responses GFPO trims, with qualitative comparisons to GRPO for AIME 25 and GPQA in Appendix A.

**5.1 Length and Accuracy Across Problem Difficulty.** On AIME 2025, response lengths grow steeply with problem difficulty—from roughly 4k tokens on easy questions to over 20k on very hard ones (Figure 8). GFPO consistently reduces this verbosity across all quartiles (Figure 4a). Token Efficiency GFPO delivers the largest overall reductions, exceeding 120% excess length reduction on easy problems—producing shorter outputs than the SFT baseline, while maintaining accuracy. Its impact diminishes on harder problems (56–79% reduction) because the token efficiency criterion permits longer responses when justified by higher rewards. Adaptive Difficulty GFPO follows the opposite trend: modest gains on easy questions (38%) but substantially stronger reductions on very hard ones (60%), effectively suppressing the “long tail” of overly verbose outputs. Shortest 8/24 also consistently surpasses Shortest 8/16, achieving the strongest reductions on very hard problems.

Accuracy patterns mirror these differences (Figure 4b). All methods perform near-perfectly on easy problems, while GRPO and GFPO both improve over the SFT baseline on harder ones. Token Efficiency’s reductions come with small, statistically insignificant accuracy dips. Adaptive Difficulty, by contrast, matches or exceeds GRPO accuracy across easy, medium, and very hard questions (e.g., 90% vs. 88% on medium; 27% vs. 27% on very hard) while simultaneously reducing length by up to 60%. Its only shortcoming appears on “hard” questions, where filtering occasionally removes useful longer responses. This can be mitigated by increasing the group size: for example, Shortest 8/24 fully recovers GRPO’s 73% accuracy on hard questions while producing substantially shorter outputs.

**5.2 Accuracy of Long Responses under GFPO.** Reasoning models often produce less accurate answers as response length grows, but this effect is entangled with problem difficulty—harder questions naturally elicit longer chains. To isolate verbosity, we fix difficulty and examine accuracy by response length on AIME 2025. Using SFT per-question accuracy as a difficulty proxy, we partition responses to hard and very hard problems into length quartiles and plot accuracy (Figure 5b).





(a) **Accuracy, Response Share, and Prompt Difficulty by Response Length.** Each cell shows accuracy (center), response share (top left), and prompt difficulty (bottom right; avg difficulty  $1 - SFT_{acc}$ ) of prompts corresponding to responses in cell, for a fixed response length range.

(b) **Accuracy vs Relative Length for Hard and Very Hard Problems.** On very hard problems, Adaptive Difficulty is most robust. Token efficiency and Shortest 8/24 drop in the longer bins, likely due to aggressive filtering.

**Figure 5: Accuracy Across Response Lengths for AIME 25.** (a) GFPO cuts long-tail verbosity (32% to 22% outputs  $\geq 20k$  tokens) and solves hard problems with shorter responses ( $\sim 9\times$  harder prompts solved with  $\leq 5k$  tokens). (b) Accuracy declines with increasing response length even at fixed difficulty. On hard problems, most models peak at 12k-16k tokens, while GFPO variants outperform GRPO in the longest bin by producing shorter, more accurate long responses.

Accuracy falls steadily with length even under fixed difficulty. On hard problems, most models peak in the mid-length range (12k–16k tokens, Table 3), suggesting a sweet spot: long enough for reasoning but short enough to avoid over-thinking. Beyond this, accuracy drops consistently. GFPO variants outperform GRPO in the longest bin (67% vs 52% on Hard; 20% vs 17% on Very Hard, Table 4), as their longest responses are both shorter (20k vs 24k on Hard; 27k vs 28k on Very Hard) and more accurate. On very hard problems, degradation is sharper. Adaptive Difficulty and Token Efficiency briefly improve from short to mid-low bins, but all methods decline at longer lengths. Token Efficiency and Shortest 8/24 show the steepest drops, likely from reduced exposure to long chains. Adaptive Difficulty is the most robust, maintaining stable accuracy across bins. By contrast, SFT degrades little with length but rarely solves hard problems, producing a flat yet low curve.

We complement this with an absolute-length analysis across models (Figure 5a). GFPO shifts substantial mass away from the long tail ( $\geq 20k$  tokens), cutting it from 32% under GRPO to 22%, and increasing the share of  $<15k$  responses. These shorter chains often solve harder problems: in the  $\leq 5k$  bin, GFPO’s prompt difficulty is  $\sim 9\times$  higher than GRPO’s (16–18% vs 2%) with only minor accuracy loss (100%  $\rightarrow$  97%). Lower accuracy in GFPO’s longest bins reflects that most solvable prompts are already handled at shorter lengths; the remaining long chains correspond to the hardest, out-of-distribution cases.

Together, the relative- and absolute-length analyses show verbosity—not difficulty—is the main driver of GRPO’s long-chain errors. GFPO mitigates this by solving harder problems more succinctly while maintaining or improving accuracy. Among variants, Shortest 8/24 and Adaptive Difficulty achieve the best balance—substantially shortening responses while preserving performance. Further gains may be possible by tuning the  $k/G$  ratio for Token Efficiency and Adaptive Difficulty.

**5.3 What is GFPO trimming?** To understand the source of GFPO’s length reductions, we annotate reasoning traces from five models—SFT, GRPO, Shortest 8/24 GFPO, Token Efficiency GFPO, and Adaptive Difficulty GFPO—on AIME 25 using GPT-4o. Each trace is segmented into four roles: *Problem* (problem setup), *Solution* (developing candidate solutions), *Verification* (checking intermediate results), and *Final* (answer statements).

Figure 6 reports average token counts per section. GRPO inflates the mid-trace reasoning compared to SFT—for example, on AIME 25 the Solution segment expands from 6.5k to 8.3k to-



258 kens, and Verification from 1.9k to 3.1k. GFPO reverses this trend by compressing Solution and  
 259 Verification. On AIME 25, Shortest 8/24 GFPO shrinks the Solution phase from 8.3k to 6.6k  
 260 tokens (94.4% of excess length removed), trimming many digressive or incorrect solution attempts.  
 261 It also reduces Verification from 3.1k to 2.3k  
 262 tokens (66.7% of excess length removed), cutting  
 263 redundant, circular checks characteristic of  
 264 GRPO.

## 265 6 Related Work

266 **GRPO Loss Modifications.** Several works re-  
 267 fine GRPO’s loss normalization to better handle  
 268 token efficiency and stability. Dr. GRPO (Liu  
 269 et al., 2025) normalizes by the longest chain in  
 270 the batch, and DAPO (Yu et al., 2025) by the  
 271 total token count—both amplifying penalties on  
 272 long incorrect outputs. GFPO adopts DAPO’s  
 273 normalization (as in ver1 (Sheng et al., 2024),  
 274 TRL (von Werra et al., 2020)) but shows that  
 275 normalization alone cannot prevent verbosity:  
 276 it penalizes long failures yet also rewards long  
 277 successes. GFPO instead modifies the advan-  
 278 tage function—filtering which chains count for  
 279 learning—orthogonal to normalization and com-  
 280 patible with variants like Dr. GRPO.

281 **Length-Aware Penalties.** Another line of work directly penalizes verbosity, e.g., capping rewards  
 282 beyond a token limit (Hou et al., 2025), applying adaptive or solve-rate scaled penalties (Su & Cardie,  
 283 2025; Xiang et al., 2025), or optimizing toward target lengths (Aggarwal & Welleck, 2025). Such  
 284 reward engineering can reduce length but often harms accuracy or requires careful tuning. GFPO  
 285 sidesteps explicit penalties: its rejection step implicitly shapes which outputs drive learning, enabling  
 286 multi-objective control (length, safety, etc.) with simpler machinery.

287 **Inference-Time Interventions.** Reasoning length can also be managed without retraining. Prior  
 288 work includes voting over the shortest  $m$  of  $k$  responses (Hassid et al., 2025), using “budget forcing”  
 289 phrases to stop generation (Muennighoff et al., 2025), or halting once answers stabilize (Liu & Wang,  
 290 2025; Yang et al., 2025). These approaches are complementary to GFPO, offering inference-time  
 291 levers for further cost savings.

292 **Rejection Sampling Methods.** Rejection sampling has been applied post-training for length control,  
 293 e.g., fine-tuning or DPO on shortest correct outputs (Kim et al., 2024), or stabilizing RL updates via  
 294 prompt filtering (Yu et al., 2025; Xiong et al., 2025). GFPO integrates rejection within RL itself:  
 295 sampling larger groups and updating only on top-ranked chains by length or reward efficiency. Related  
 296 methods use rejection for contrastive pair selection (Khaki et al., 2024), constrained decoding (Lipkin  
 297 et al., 2025), or efficient Best-of- $N$  search (Sun et al., 2024). GFPO differs in embedding this  
 298 principle directly into the policy update, avoiding extra distillation or explicit penalties.

## 299 7 Conclusion

300 We present Group-Filtered Policy Optimization (GFPO), a simple extension of GRPO that samples  
 301 more responses and selectively trains on those aligned with desirable attributes. This filtering acts  
 302 as implicit reward shaping, improving output quality and curbing length inflation without complex  
 303 reward design. Across benchmarks, GFPO maintains GRPO-level accuracy while substantially  
 304 boosting efficiency: Token Efficiency GFPO achieves the largest reductions overall, and Adaptive  
 305 Difficulty GFPO delivers the best trade-offs on the hardest problems. By investing extra compute at  
 306 training time, GFPO yields lasting gains in inference-time reasoning efficiency.

## 307 8 Acknowledgments

308 We would like to thank Guoqing Zheng, Suriya Gunasekar, Piero Kauffmann, Mojan Javaheripi,  
 309 Safoora Yousefi, and the AI Frontiers team at Microsoft Research for valuable discussions and  
 310 feedback throughout this project.

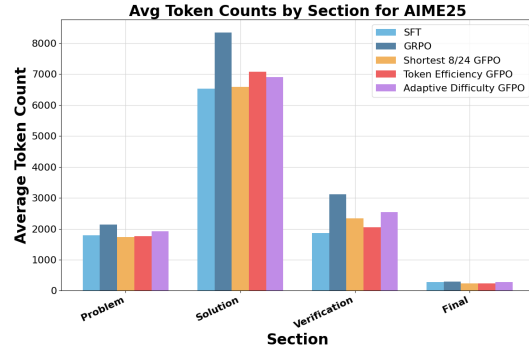


Figure 6: **Average Token Counts by Reasoning-Trace.** GRPO inflates the Solution and Verification phases relative to SFT. GFPO variants markedly reduce this excess—on AIME 25, Shortest 8/24 cuts Solution length by 94.4% and Verification length by 66.7%.

## References

- Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Xin Wang, Rachel Ward, Yue Wu, Dingli Yu, Cyril Zhang, and Yi Zhang. Phi-4 technical report, 2024. URL <https://arxiv.org/abs/2412.08905>.
- Marah Abdin, Sahaj Agarwal, Ahmed Awadallah, Vidhisha Balachandran, Harkirat Behl, Lingjiao Chen, Gustavo de Rosa, Suriya Gunasekar, Mojan Javaheripi, Neel Joshi, Piero Kauffmann, Yash Lara, Caio César Teodoro Mendes, Arindam Mitra, Besmira Nushi, Dimitris Papailiopoulos, Olli Saarikivi, Shital Shah, Vaishnavi Shrivastava, Vibhav Vineet, Yue Wu, Safoora Yousefi, and Guoqing Zheng. Phi-4-reasoning technical report, 2025. URL <https://arxiv.org/abs/2504.21318>.
- Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning, 2025. URL <https://arxiv.org/abs/2503.04697>, 2025.
- AIME. Aime 83-24. [https://huggingface.co/datasets/di-zhang-fdu/AIME\\_1983\\_2024](https://huggingface.co/datasets/di-zhang-fdu/AIME_1983_2024), 2024. Accessed: 2025-03-17.
- AIME. Aime 2025. <https://huggingface.co/datasets/lchen001/AIME2025>, 2025. Accessed: 2025-03-17.
- Vidhisha Balachandran, Jingya Chen, Lingjiao Chen, Shivam Garg, Neel Joshi, Yash Lara, John Langford, Besmira Nushi, Vibhav Vineet, Yue Wu, et al. Inference-time scaling for complex tasks: Where we stand and what lies ahead. *arXiv preprint arXiv:2504.00294*, 2025.
- Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, et al. Omni-math: A universal olympiad level mathematic benchmark for large language models. *ICLR*, 2025.
- 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.
- Michael Hassid, Gabriel Synnaeve, Yossi Adi, and Roy Schwartz. Don’t overthink it. preferring shorter thinking chains for improved llm reasoning. *arXiv preprint arXiv:2505.17813*, 2025.
- Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning. *arXiv preprint arXiv:2504.01296*, 2025.
- Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.
- Saeed Khaki, JinJin Li, Lan Ma, Liu Yang, and Prathap Ramachandra. Rs-dpo: A hybrid rejection sampling and direct preference optimization method for alignment of large language models. *arXiv preprint arXiv:2402.10038*, 2024.
- Sunnie SY Kim, Q Vera Liao, Mihaela Vorvoreanu, Stephanie Ballard, and Jennifer Wortman Vaughan. "i’m not sure, but...": Examining the impact of large language models’ uncertainty expression on user reliance and trust. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, pp. 822–835, 2024.
- Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, et al. Minimax-01: Scaling foundation models with lightning attention. *arXiv preprint arXiv:2501.08313*, 2025.
- Benjamin Lipkin, Benjamin LeBrun, Jacob Hoover Vigly, João Loula, David R MacIver, Li Du, Jason Eisner, Ryan Cotterell, Vikash Mansinghka, Timothy J O’Donnell, et al. Fast controlled generation from language models with adaptive weighted rejection sampling. *arXiv preprint arXiv:2504.05410*, 2025.

360 Xin Liu and Lu Wang. Answer convergence as a signal for early stopping in reasoning. *arXiv preprint*  
361 *arXiv:2506.02536*, 2025.

362 Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min  
363 Lin. Understanding rl-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*,  
364 2025.

365 Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke  
366 Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. sl: Simple test-time  
367 scaling. *arXiv preprint arXiv:2501.19393*, 2025.

368 OpenAI. Introducing openai o3 and o4-mini. [https://openai.com/index/  
369 introducing-o3-and-o4-mini/](https://openai.com/index/introducing-o3-and-o4-mini/), April 2025.

370 David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani,  
371 Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In  
372 *First Conference on Language Modeling*, 2024.

373 John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal pol-  
374 icy optimization algorithms. In *Proceedings of the 5th International Conference on Learning*  
375 *Representations (ICLR)*, 2017. URL <https://openreview.net/forum?id=r1etN1rtPB>.

376 Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang,  
377 Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathemat-  
378 ical reasoning in open language models, 2024. URL <https://arxiv.org/abs/2402.03300>.

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

382 Jinyan Su and Claire Cardie. Thinking fast and right: Balancing accuracy and reasoning length with  
383 adaptive rewards. *arXiv preprint arXiv:2505.18298*, 2025.

384 Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang, Jiahao Qiu, Ming Yin, Mengdi Wang, Peter  
385 Bartlett, and Andrea Zanette. Fast best-of-n decoding via speculative rejection. *Advances in Neural*  
386 *Information Processing Systems*, 37:32630–32652, 2024.

387 Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan  
388 Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement  
389 learning. <https://github.com/huggingface/trl>, 2020.

390 Frank Wilcoxon. Individual comparisons by ranking methods. In *Breakthroughs in statistics:*  
391 *Methodology and distribution*, pp. 196–202. Springer, 1992.

392 Violet Xiang, Chase Blagden, Rafael Rafailov, Nathan Lile, Sang Truong, Chelsea Finn, and Nick  
393 Haber. Just enough thinking: Efficient reasoning with adaptive length penalties reinforcement  
394 learning. *arXiv preprint arXiv:2506.05256*, 2025.

395 Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong  
396 Zhang, Caiming Xiong, et al. A minimalist approach to llm reasoning: from rejection sampling to  
397 reinforce. *arXiv preprint arXiv:2504.11343*, 2025.

398 Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Zheng Lin, Li Cao,  
399 and Weiping Wang. Dynamic early exit in reasoning models. *arXiv preprint arXiv:2504.15895*,  
400 2025.

401 Deheng Ye, Zhao Liu, Mingfei Sun, Bei Shi, Peilin Zhao, Hao Wu, Hongsheng Yu, Shaojie Yang,  
402 Xipeng Wu, Qingwei Guo, et al. Mastering complex control in moba games with deep reinforce-  
403 ment learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp.  
404 6672–6679, 2020.

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

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

### AIME 25 II Problem 1: GRPO v/s Token Efficiency GFPO

#### User:

Six points  $A, B, C, D, E$  and  $F$  lie in a straight line in that order. Suppose that  $G$  is a point not on the line and that  $AC = 26$ ,  $BD = 22$ ,  $CE = 31$ ,  $DF = 33$ ,  $AF = 73$ ,  $CG = 40$ , and  $DG = 30$ . Find the area of  $\triangle BGE$ .

#### GRPO

<think>... So now triangle  $BGE$ :  $B = (18, 0)$ ,  $G = (58, 24)$ ,  $E = (57, 0)$ . Its base  $BE$  = distance from 18 to 57 = 39. But then what is the height? The triangle's base is along x-axis and  $G$  has y coordinate 24, but note that  $G$  is not vertically above the segment? Actually, base  $BE$ : endpoints  $(18, 0)$  and  $(57, 0)$ . The distance from  $G$  to line  $BE$  is simply the absolute y coordinate if  $G$ 's x coordinate lies between 18 and 57? But here  $x_G = 58$ , so  $G$  is to the right of  $E$ . But the formula for area using base and height with base along line is the distance from  $G$  to the line containing  $BE$ . But the line containing  $BE$  is horizontal ( $y = 0$ ). So the distance is  $|24| = 24$ , regardless of the x-coordinate. So area =  $\frac{1}{2} \cdot \text{base} \cdot \text{height} = \frac{1}{2} \cdot 39 \cdot 24 = 468$ .

Wait,  $39 \cdot 24 = 936$ , half is 468. But wait, is that the answer? Let's check: base  $BE$  length =  $57 - 18 = 39$ , height = 24. So area =  $\frac{39 \cdot 24}{2} = 468$ .

But the problem's answer is expected to be an integer? Possibly yes. But let's check step 1: "Six points  $A, B, C, D, E, F$  lie in a straight line in that order." Our computed positions are consistent:  $A = 0$ ,  $B = 18$ ,  $C = 26$ ,  $D = 40$ ,  $E = 57$ ,  $F = 73$ . Check distances:  $AC = 26$ ,  $BD = 40 - 18 = 22$ ,  $CE = 57 - 26 = 31$ ,  $DF = 73 - 40 = 33$ ,  $AF = 73$ , correct. Now then triangles:  $CG =$  distance from  $C = (26, 0)$  to  $G = (58, 24)$  equals  $\sqrt{(32)^2 + 24^2} = \sqrt{1024 + 576} = \sqrt{1600} = 40$ , and  $DG = \sqrt{(58 - 40)^2 + 24^2} = \sqrt{18^2 + 576} = \sqrt{324 + 576} = \sqrt{900} = 30$ , so correct. Now triangle  $BGE$ : vertices  $B = (18, 0)$ ,  $G = (58, 24)$ ,  $E = (57, 0)$ . The base  $BE$  has length  $57 - 18 = 39$ , and the height is the vertical distance from  $G$  to the x-axis is 24, so area =  $\frac{39 \cdot 24}{2} = 468$ . So answer is 468.

However, double-check: It might be that  $G$  could have negative  $y$ . But then area is same magnitude, absolute value. Therefore answer: 468.

But wait, check problem statement: "Find the area of triangle  $BGE$ ." Not necessarily with base  $BE$  horizontal? But we computed that. I want to check if there's any trick: Possibly the triangle is not oriented with base along the line containing  $B$  and  $E$  because then the altitude would be the distance from  $G$  to that line. And indeed the line  $BE$  is horizontal, so the altitude is the absolute value of  $G$ 's  $y$ -coordinate. We computed  $G$ 's  $y$ -coordinate as 24. So final answer: 468.

Wait, but what about possibility  $x = 58$ ? Let's double-check the  $x$ -coordinate from difference of circle equations: We had circles:  $(x - 26)^2 + y^2 = 40^2 = 1600$ ,  $(x - 40)^2 + y^2 = 30^2 = 900$ . Subtracting:  $(x - 26)^2 - (x - 40)^2 = 700$ . Expand:  $(x^2 - 52x + 676) - (x^2 - 80x + 1600) = -52x + 676 + 80x - 1600 = 28x - 924 = 700$ , so  $28x = 1624$ ,  $x = 58$ . So that's consistent. So yes. Thus final answer: 468. .... </think>

#### Token Efficiency GFPO

<think>... Now we want the area of triangle  $BGE$ . The vertices are  $B = (18, 0)$ ,  $G = (58, 24)$  or  $(58, -24)$ , and  $E = (57, 0)$ . The area is given by  $\frac{1}{2}$  absolute value of cross product of  $\vec{BG}$  and  $\vec{BE}$  etc. Let's compute vectors:  $\vec{BG} = G - B = (58 - 18, 24 - 0) = (40, 24)$ .  $\vec{BE} = E - B = (57 - 18, 0 - 0) = (39, 0)$ . Area =  $\frac{1}{2} |\det(\vec{BG}, \vec{BE})| = \frac{1}{2} |(40 \cdot 0 - 24 \cdot 39)| = \frac{1}{2} |-936| = 468$ . Thus answer: 468. So final answer: 468. .... </think>

#### Comparison:

The GRPO response recomputes the area of  $\triangle BGE$  six times, while Token Efficiency GFPO computes it only once.

# AIME 25 I Problem 8: GRPO v/s Token Efficiency GFPO

## User:

Let  $k$  be real numbers such that the system  $|25 + 20i - z| = 5$  and  $|z - 4 - k| = |z - 3i - k|$  has exactly one complex solution  $z$ . The sum of all possible values of  $k$  can be written as  $\frac{m}{n}$ , where  $m$  and  $n$  are relatively prime positive integers. Find  $m + n$ . Here  $i = \sqrt{-1}$ .

## GRPO

<think>... Equate squares:  $(x - (4 + k))^2 + y^2 = (x - k)^2 + (y - 3)^2$ . Expand:  $(x - (4 + k))^2 = (x - k - 4)^2 = (x - k)^2 - 2 \cdot 4 \cdot (x - k) + 16$ , wait check: actually, let's expand properly. Let  $A = 4 + k$ ,  $B = k$ , then equation becomes  $(x - A)^2 + y^2 = (x - B)^2 + (y - 3)^2$ . Expand LHS:  $(x - A)^2 = x^2 - 2Ax + A^2$  and then plus  $y^2$ , so LHS =  $x^2 - 2Ax + A^2 + y^2$ . RHS:  $(x - B)^2 + (y - 3)^2 = (x^2 - 2Bx + B^2) + (y^2 - 6y + 9) = x^2 - 2Bx + B^2 + y^2 - 6y + 9$ . Cancel  $x^2$  and  $y^2$  yields  $-2Ax + A^2 = -2Bx + B^2 - 6y + 9$ . Rearranging:  $-2Ax + A^2 + 2Bx - B^2 + 6y - 9 = 0$ , or  $2(B - A)x + 6y + (A^2 - B^2 - 9) = 0$ . But note that  $A^2 - B^2 = (A + B)(A - B)$  and  $A + B = (4 + k) + k = 4 + 2k$ , and  $A - B = (4 + k) - k = 4$ , so  $A^2 - B^2 = 4(4 + 2k) = 16 + 8k$ . Then our equation is  $2(B - A)x + 6y + (16 + 8k - 9) = 0$ , i.e.  $2(B - A)x + 6y + (7 + 8k) = 0$ , but note that  $B - A = k - (4 + k) = -4$ . Then  $2(B - A)x = 2(-4)x = -8x$ , so the equation becomes  $-8x + 6y + 7 + 8k = 0$  or  $8x - 6y = 7 + 8k$ .

...

Alternatively, we can get the perpendicular bisector of the segment connecting  $(4 + k, 0)$  and  $(k, 3)$ . Let's check the midpoint:  $M = (\frac{4+k+k}{2}, \frac{0+3}{2}) = (\frac{4+2k}{2}, \frac{3}{2}) = (2 + k, 3/2)$ . The slope of the line joining these two points from  $(4 + k, 0)$  to  $(k, 3)$  equals  $\frac{3-0}{k-(4+k)} = \frac{3}{-4} = -\frac{3}{4}$ . So the perpendicular bisector slope is the positive reciprocal:  $\frac{4}{3}$ , and it passes through  $M$ . Its equation is  $y - 3/2 = \frac{4}{3}(x - (2 + k))$ . Multiply both sides by 3:  $3y - 9/2 = 4(x - 2 - k)$ . Multiply both sides by 2:  $6y - 9 = 8(x - 2 - k) = 8x - 16 - 8k$ . Then rearranging:  $6y - 9 = 8x - 16 - 8k$ , so  $8x - 6y = 16 + 8k + 9 = 25 + 8k$ .

...

Let's re-check... So the equation is  $(x - (4 + k))^2 + (y - 0)^2 = (x - k)^2 + (y - 3)^2$ . Expand LHS:  $(x - 4 - k)^2 + y^2 = (x - k - 4)^2 + y^2 = (x^2 - 2x(4 + k) + (4 + k)^2) + y^2$ . The RHS is  $(x - k)^2 + (y - 3)^2 = x^2 - 2kx + k^2 + y^2 - 6y + 9$ . Cancel  $x^2$  and  $y^2$  from both sides to get  $-2(4 + k)x + (4 + k)^2 = -2kx + k^2 - 6y + 9$ . Bring terms to one side:  $-2(4 + k)x + (4 + k)^2 + 2kx - k^2 + 6y - 9 = 0$ . Combine the  $x$  terms:  $-2(4 + k)x + 2kx = -8x - 2kx + 2kx = -8x$ . Then the constant terms:  $(4 + k)^2 - k^2 - 9$ ; expanding  $(4 + k)^2 = k^2 + 8k + 16$ , subtracting  $k^2$  gives  $8k + 16$ , and subtracting 9 gives  $8k + 7$ . So the equation becomes  $-8x + 6y + (8k + 7) = 0$ , or equivalently  $8x - 6y = 8k + 7$ .

..... </think>

## Token Efficiency GFPO

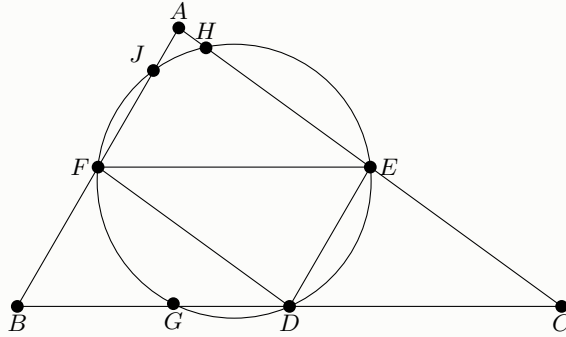
<think>... Squaring both sides yields:  $(x - (4 + k))^2 + y^2 = (x - k)^2 + (y - 3)^2$ . Expand:  $(x^2 - 2(4 + k)x + (4 + k)^2 + y^2) = (x^2 - 2kx + k^2 + y^2 - 6y + 9)$ . Cancel  $x^2$  and  $y^2$ , leaving:  $-2(4 + k)x + (4 + k)^2 = -2kx + k^2 - 6y + 9$ . Simplify: Bring terms:  $-2(4 + k)x + (4 + k)^2 + 2kx - k^2 + 6y - 9 = 0$ . Combine  $x$  terms:  $-2(4 + k)x + 2kx = -2(4 + k - k)x = -2 \cdot 4x = -8x$ . So equation becomes:  $-8x + (4 + k)^2 - k^2 + 6y - 9 = 0$ . Simplify constant:  $(4 + k)^2 - k^2 = (16 + 8k + k^2) - k^2 = 16 + 8k$ . So the equation is:  $-8x + (16 + 8k) + 6y - 9 = 0$ , or  $-8x + 6y + (8k + 7) = 0$ , but check:  $16 + 8k - 9 = 7 + 8k$ . Multiply by  $-1$ :  $8x - 6y - (8k + 7) = 0$ , or in standard form:  $8x - 6y = 8k + 7$ . ..... </think>

## Comparison:

GRPO response recomputes and checks the perpendicular bisector equation 5 times (only 3 instances shown), while Token Efficiency GFPO computes it only once.

User:

Suppose  $\triangle ABC$  has angles  $\angle BAC = 84^\circ$ ,  $\angle ABC = 60^\circ$ , and  $\angle ACB = 36^\circ$ . Let  $D$ ,  $E$ , and  $F$  be the midpoints of sides  $BC$ ,  $AC$ , and  $AB$ , respectively. The circumcircle of  $\triangle DEF$  intersects  $BD$ ,  $AE$ , and  $AF$  at points  $G$ ,  $H$ , and  $J$ , respectively. The points  $G$ ,  $D$ ,  $E$ ,  $H$ ,  $J$ , and  $F$  divide the circumcircle of  $\triangle DEF$  into six minor arcs, as shown. Find  $\widehat{DE} + 2 \cdot \widehat{HJ} + 3 \cdot \widehat{FG}$ , where the arcs are measured in degrees.



GRPO

<think> ..... We need to compute arcs on the circumcircle of triangle  $DEF$ . Let's denote the vertices of triangle  $DEF$  in order: It says circle of triangle  $DEF$ , and then the points  $G, D, E, H, J, F$  are arranged in order along circle. So the vertices of triangle  $DEF$  are  $D, E, F$ . But note the circle has six points in order, and the six arcs are (arc  $DE$ ), then arc  $EH$ ? Wait, but check the given: They mention "the points  $G, D, E, H, J$ , and  $F$  divide the circumcircle of triangle  $DEF$  into six minor arcs, as shown." That means the circle order is:  $G$ , then  $D$ , then  $E$ , then  $H$ , then  $J$ , then  $F$ , and then back to  $G$ . But we need to be careful: The circumcircle of triangle  $DEF$  has vertices  $D, E, F$  as well. So the arcs are between adjacent points in the circle order. The six arcs are: arc  $GD$ , arc  $DE$ , arc  $EH$ , arc  $HJ$ , arc  $JF$ , arc  $FG$ . But the problem statement says "the points  $G, D, E, H, J$ , and  $F$  divide the circumcircle of triangle  $DEF$  into six minor arcs, as shown." But then they ask "find arc  $DE + 2 \cdot \text{arc } HJ + 3 \cdot \text{arc } FG$ ." But wait, which arc is arc  $FG$ ? The arcs, I need to check which arcs are which from the order: They are given in the diagram: The points in order are:  $G$ , then  $D$ , then  $E$ , then  $H$ , then  $J$ , then  $F$ , then back to  $G$ . So the arcs are: arc  $GD$ , arc  $DE$ , arc  $EH$ , arc  $HJ$ , arc  $JF$ , arc  $FG$ . But the problem asks for arc  $DE$ , then  $2 \cdot (\text{arc } HJ)$ , then  $3 \cdot (\text{arc } FG)$ . But note that arc  $FG$  is the arc from  $F$  to  $G$ , but in our ordering, the arc from  $F$  to  $G$  is the arc  $FG$ , but careful: the order is  $G-D-E-H-J-F$ , so arc  $FG$  is the arc from  $F$  to  $G$ , which is indeed the last arc in the sequence. So we need to compute these arc measures, but they are not trivial. We need to find angles related to triangle  $ABC$  and midpoints  $D, E, F$ . Let coordinates:  $A, B, C$ . But maybe we can use properties: The circle through midpoints of sides of a triangle is the medial circle, but it's not necessarily the circumcircle of the medial triangle, but that's exactly what it is: The circumcircle of triangle  $DEF$ , where  $D, E, F$  are midpoints of  $BC, AC, AB$  respectively, is the nine-point circle of triangle  $ABC$ . ..... </think>

Shortest 8/24 GFPO

<think> ..... We need to compute arcs on the circumcircle of triangle  $DEF$ . Let's denote the vertices of triangle  $DEF$  in order: It says circle of triangle  $DEF$ , and then the points  $G, D, E, H, J, F$  are arranged in order along circle. So the vertices of triangle  $DEF$  are  $D, E, F$ . But note the circle has intersections with lines  $BD, AE, AF$ . Let's re-read the problem statement. Given triangle  $ABC$  with angles  $A = 84^\circ, B = 60^\circ, C = 36^\circ$ .  $D, E, F$  are midpoints of sides  $BC, AC, AB$  respectively. The circumcircle of triangle  $DEF$  intersects  $BD, AE, AF$  at points  $G, H$ , and  $J$  respectively. The points  $G, D, E, H, J, F$  divide the circumcircle of triangle  $DEF$  into six minor arcs, as shown. I want to label positions: We have triangle  $ABC$ , but then midpoints  $D, E, F$  are on sides  $BC, AC, AB$  respectively. Now, note that triangle  $DEF$  is the medial triangle of  $ABC$ . And the circumcircle of the medial triangle is known to be the nine-point circle of triangle  $ABC$ . .... </think>

414

**Comparison:**

The Shortest 8/24 GFPO response quickly recognizes that  $\triangle DEF$  is the medial triangle and its circumcircle is the nine-point circle, while the GRPO response gets bogged down in arc bookkeeping before arriving at the same conclusion.

**AIME 25 I Problem 9: GRPO v/s Adaptive Difficulty GFPO****User:**

The parabola with equation  $y = x^2 - 4$  is rotated 60 degrees counterclockwise around the origin. The unique point in the fourth quadrant where the original parabola and its image intersect has  $y$ -coordinate  $\frac{a-\sqrt{b}}{c}$ . Find  $a + b + c$ .

**GRPO**

<think> .....

But before that, we can attempt to factor the quartic if possible. It is a quartic in  $u$ . But perhaps a substitution might simplify it? Alternatively, note that  $u$  might be a "nice" value? Alternatively, we can try to find the intersection by considering symmetry. Alternatively, consider the fact that the intersection point is the unique one in the fourth quadrant. Maybe we can try to solve directly by eliminating  $u$ . Alternatively, consider using rotation in reverse: For an intersection point  $(x, y)$ , we have  $y = x^2 - 4$  and also  $(y + 4)???$  Not sure.

..... </think>

**Adaptive Difficulty GFPO**

<think> .....

I think we need to solve for  $x$  explicitly from the quartic. Alternatively, we can use the approach: Let  $(x, y)$  be the intersection point. Then  $(x, y)$  satisfies:  $y = x^2 - 4$ , and also  $(x, y)$  satisfies the equation of the rotated parabola, i.e. if we rotate  $(x, y)$  by  $-60$  degrees, we get a point on the original parabola: So let  $(u, v) = (\frac{1}{2}x + \frac{3}{2}y, -\frac{3}{2}x + \frac{1}{2}y)$ . Then  $y = u^2 - 4$ .

..... </think>

**Comparison:**

The GRPO response jumps between several different alternatives—factoring the quartic, trying substitutions, symmetry arguments, and vague rotation ideas without committing to a plan and doesn't reduce uncertainty, whereas the Adaptive Difficulty GFPO defines a single clear approach to take.

415



### GPQA: GRPO v/s Token Efficiency GFPO

User:

Consider this annihilation process  $p + \bar{p} \rightarrow 2A^+ + 2A^-$   
The antiproton is slowly moving and  $m_A c^2 = 300 \text{ MeV}$ .  
What is the velocity of particle A?

GRPO

<think>... Actually, maybe the process is in the lab frame where the antiproton is slow relative to the proton. But wait, which is slowly moving: "The antiproton is slowly moving" means that maybe the proton is moving? Wait check: " $p + \bar{p}$ " one of them is slow, but which one? "The antiproton is slowly moving" means that the proton is moving fast? But then the question states: "Consider this annihilation process:  $p + \bar{p} \rightarrow 2A^+ + 2A^-$ ". The wording "The antiproton is slowly moving" implies that the initial momentum is coming from the proton, which is fast. But then the question is asking: "What is the velocity of particle A?" Possibly meaning the outgoing particles have velocity determined by energy-momentum conservation in the center-of-mass frame.

...

Alternatively, the problem might be set in the center-of-mass frame. Let me think: "Annihilation process" usually is considered in the center-of-mass frame. But then "The antiproton is slowly moving" might be a hint that the proton is at rest? Because usually one would take  $p$  is at rest, and the anti-proton is slow relative to  $p$ ? Actually, "slowly moving" means small velocity. ... But if the antiproton is slow, then maybe the proton has kinetic energy? But then the process might be not exactly symmetric. Alternatively, maybe the assumption is that the antiproton is slow relative to the lab frame. That means the proton is at rest? Because then the antiproton is slow? That is contradictory: if the proton is at rest, then the anti-proton must be moving if we are in the lab frame?

...

Maybe the intended scenario is that the annihilation takes place in the lab frame where the antiproton is almost at rest, and the proton is moving with a velocity such that the center-of-mass energy is just above threshold. But then threshold for production of  $4A$  is  $4m_A = 1200 \text{ MeV}$ .

...

That doesn't work. Maybe the intended scenario is that both the proton and antiproton are nearly at rest in the center-of-mass frame...

..... </think>

Token Efficiency GFPO

<think>... Possibly the antiproton is slowly moving, so the reaction is in the lab frame, and one particle is at rest, but then the total kinetic energy available is the difference of the rest masses? But careful: if the antiproton is "slowly moving" that means it is non-relativistic? But then the proton might be at rest or antiproton at rest? Let me re-read: "The antiproton is slowly moving." So perhaps the proton is at rest? Actually, which one is at rest? The phrase "slowly moving" might indicate that the antiproton is nearly at rest relative to the proton? But then the total energy available is the mass difference, but then momentum conservation forces the final products to share momentum such that they have equal velocity magnitude. But which velocity? Let me analyze the reaction in the center-of-mass frame. .... </think>

Comparison:

The GRPO response repeatedly rehashes whether the reaction is in the lab frame or center-of-mass frame, while Token Efficiency GFPO response correctly makes the assumption of center-of-mass frame once.

416

### GPQA: GRPO v/s Adaptive Difficulty GFPO

User:

A quantum mechanical particle of mass  $m$  moves in two dimensions in the following potential, as a function of  $(r, \theta)$ :  $V(r, \theta) = \frac{1}{2}kr^2 + \frac{3}{2}kr^2 \cos^2(\theta)$ . Find the energy spectrum.

417

### GRPO

<think>... Let  $x = r \cos \theta$ ,  $y = r \sin \theta$ . Then  $r^2 = x^2 + y^2$ ,  $\cos^2 \theta = \frac{x^2}{r^2}$  if  $r \neq 0$ . So potential becomes:  $V(x, y) = \frac{1}{2}k(x^2 + y^2) + \frac{3}{2}k(x^2 + y^2) \left( \frac{x^2}{x^2 + y^2} \right) = \frac{1}{2}k(x^2 + y^2) + \frac{3}{2}kx^2 = \frac{1}{2}kx^2 + \frac{1}{2}ky^2 + \frac{3}{2}kx^2 = (2k)x^2$ ? Let's check:  $\frac{1}{2}kx^2 + \frac{3}{2}kx^2 = 2kx^2$ , plus  $\frac{1}{2}ky^2$  remains. So potential becomes:  $V(x, y) = 2kx^2 + \frac{1}{2}ky^2$ . That is a two-dimensional harmonic oscillator with different frequencies along  $x$  and  $y$ . But wait, does that match? Let's recalculate: Given  $V(r, \theta) = \frac{1}{2}kr^2 + \frac{3}{2}kr^2 \cos^2 \theta = \frac{1}{2}k(x^2 + y^2) + \frac{3}{2}k \cos^2 \theta (x^2 + y^2)$ . But  $\cos \theta = x/r$  so  $\cos^2 \theta = \frac{x^2}{r^2}$ . So then  $V = \frac{1}{2}k(x^2 + y^2) + \frac{3}{2}k(x^2 + y^2) \cdot \frac{x^2}{x^2 + y^2} = \frac{1}{2}k(x^2 + y^2) + \frac{3}{2}kx^2$ . So indeed  $V = \frac{1}{2}ky^2 + \left(\frac{1}{2} + \frac{3}{2}\right)kx^2 = \frac{1}{2}ky^2 + 2kx^2$ . That is a harmonic oscillator with frequencies:  $\omega_x = \sqrt{\frac{4k}{m}} = 2\sqrt{\frac{k}{m}}$  and  $\omega_y = \sqrt{\frac{k}{m}}$ .

Wait, but our derivation assumed that the potential is separable in Cartesian coordinates, which is valid if we perform transformation  $x = r \cos \theta$ ,  $y = r \sin \theta$ . But is that always valid? Let's double-check: The given potential is  $V(r, \theta) = \frac{1}{2}kr^2 + \frac{3}{2}kr^2 \cos^2 \theta = \frac{1}{2}k(x^2 + y^2) + \frac{3}{2}k \left( \frac{x^2}{x^2 + y^2} \right) (x^2 + y^2) = \frac{1}{2}k(x^2 + y^2) + \frac{3}{2}kx^2 = \frac{1}{2}ky^2 + 2kx^2$ . Yes that is indeed a separable potential in Cartesian coordinates with  $x$  and  $y$  degrees of freedom. So the energy eigenvalues are those of two independent harmonic oscillators with frequencies  $\omega_x = \sqrt{\frac{4k}{m}} = 2\sqrt{\frac{k}{m}}$  and  $\omega_y = \sqrt{\frac{k}{m}}$ .  
..... </think>

### Adaptive Difficulty GFPO

<think>... Let  $x = r \cos \theta$ ,  $y = r \sin \theta$ . Then  $\cos^2 \theta = \frac{x^2}{r^2}$  if  $r \neq 0$ . So then potential becomes:  $V(x, y) = \frac{1}{2}k(x^2 + y^2) + \frac{3}{2}k(x^2 + y^2) \left( \frac{x^2}{x^2 + y^2} \right) = \frac{1}{2}k(x^2 + y^2) + \frac{3}{2}kx^2$ . So rewriting:  $V(x, y) = \frac{1}{2}ky^2 + \left(\frac{1}{2}k + \frac{3}{2}k\right)x^2 = \frac{1}{2}ky^2 + 2kx^2$ . Wait, check:  $x^2 + y^2$  multiplied by factor  $\frac{1}{2}k$  gives  $\frac{1}{2}kx^2 + \frac{1}{2}ky^2$  and then add  $\frac{3}{2}kx^2$  gives total potential:  $V(x, y) = \left(\frac{1}{2}k + \frac{3}{2}k\right)x^2 + \frac{1}{2}ky^2 = (2k)x^2 + \frac{1}{2}ky^2$ . So then the potential is anisotropic harmonic oscillator in Cartesian coordinates with frequencies: for  $x$ :  $\omega_x = \sqrt{\frac{4k}{m}} = 2\sqrt{\frac{k}{m}}$  and for  $y$ :  $\omega_y = \sqrt{\frac{k}{m}}$ .  
..... </think>

### Comparison:

The GRPO response recalculates the harmonic oscillator frequencies multiple times while the Adaptive Difficulty GFPO response only calculates this once.

# GPQA: GRPO v/s Shortest 8/24 GFPO

## User:

A light beam is propagating through a glass with index of refraction  $n$ . The glass is moving at constant velocity  $v$  in the same direction as the beam and toward the observer in laboratory. What is the speed of light in glass relative to the observer in laboratory? Take the speed of light in vacuum  $c = 1$ .

## GRPO

<think>... Let me derive from special relativity: The speed of light in a medium is  $c/n$  in the medium's rest frame. But when the medium is moving with velocity  $v$ , the speed of light in the lab frame is given by the relativistic velocity addition formula:  $u' = \frac{u+v}{1+(uv)/c^2}$ . Here,  $u = c/n$  and  $v$  is the velocity of the medium in lab frame. But wait, is it  $v$ ? But careful: The medium is moving with velocity  $v$  in lab frame and the light moves in the medium frame with speed  $c/n$ . But the medium's velocity relative to lab is  $v$ , so we can use velocity addition. But then the effective speed in lab frame for light traveling in the same direction as medium is:  $u = \frac{c/n+v}{1+(vc/(nc^2))}$ ? Wait, let me recalc:

The formula is:  $u = \frac{u'+v}{1+u'v/c^2}$ . But  $c = 1$ , so  $u = \frac{1/n+v}{1+v/(n)}$ . Actually, check: If  $u' = 1/n$  then  $u = \frac{1/n+v}{(n+v)/n} = \frac{1/n+v}{(n+v)/n} = (\frac{1}{n} + v) \frac{n}{n+v} = \frac{1+nv}{n+v}$ . But wait, then answer is  $\frac{1+nv}{n+v}$ ?

... But is that the Fizeau drag effect? The Fizeau drag coefficient is  $1 - 1/n^2$ . Wait, I recall the formula for the speed of light in a moving medium in the direction of flow is:  $u = 1/n + (1 - 1/n^2)v$ . Let me check with our expression:  $\frac{1/n+v}{1+v/n} = \frac{1/n+v}{(n+v)/n} = (\frac{1}{n} + v) \frac{n}{n+v} = \frac{1+nv}{n+v}$ . Now expand that in  $v$ : Let  $v$  be small. Write  $\frac{1+nv}{n+v} = \frac{1+nv}{n(1+v/n)} = \frac{1}{n} \frac{1+nv}{1+v/n}$ . Now expand denominator:  $\frac{1}{1+v/n} = 1 - v/n + (v/n)^2 + \dots$ . So we get  $= \frac{1}{n} (1 + nv)(1 - v/n) = \frac{1}{n} [1 + nv - v/n - nv \cdot (v/n)] = \frac{1}{n} [1 + nv - v/n - v^2]$ . Actually, let me do it properly:  $(1 + nv)(1 - v/n) = 1 - v/n + nv - v^2$ . So that equals  $1 + nv - v/n - v^2$ . Dividing by  $n$  gives  $= 1/n + v - v/n^2 - v^2/n$ . So the coefficient of  $v$  is  $v(1 - 1/n^2)$ . So indeed, to first order in  $v$ ,  $u = 1/n + (1 - 1/n^2)v$ . That is the Fizeau drag formula. So our expression is consistent with the known formula for the speed of light in a moving medium. But wait, are we using the correct sign? The medium is moving toward the observer, so the effective speed of light relative to the observer is increased relative to the medium's rest frame speed. So the answer is  $u_{\text{lab}} = \frac{1/n+v}{1+v/n}$  which simplifies to  $\frac{1+nv}{n+v}$ .

..... </think>

## Shortest 8/24 GFPO

<think>...

The transformation from glass frame to lab frame, where the speed of light is  $c/n$  in the glass frame and then add the velocity of glass, which is  $v$ . But note that the direction is the same as the beam. But the lab observer sees the beam speed as  $v + (1/n)$ ? But wait not exactly, because the addition formula is:  $u' = \frac{u+v}{1+uv}$ . But careful: Actually, we have an object moving at speed  $u$  (with respect to the glass) and the glass moving at speed  $v$  (with respect to lab) along same direction. The lab speed  $u_{\text{lab}}$  of the light beam should be computed using relativistic addition formula:  $u_{\text{lab}} = \frac{u_{\text{glass}}+v}{1+u_{\text{glass}} \cdot v}$ . But note that  $u_{\text{glass}}$  is  $1/n$ . So lab speed  $= \frac{1/n+v}{1+(1/n)v}$ . Multiply numerator and denominator by  $n$ :  $= \frac{1+nv}{n+v}$ ... But in lab frame, the medium is moving, so the speed of light in lab frame is:  $u = \frac{1/n+v}{1+v/n} = \frac{1+nv}{n+v}$  as computed. Wait check: Multiply numerator and denominator by  $n$ , get  $\frac{1+nv}{n+v}$ . Yes.

..... </think>

## Comparison:

The GRPO response circles back and re-derives the velocity addition with  $u_{\text{glass}} = 1/n$  multiple times, while the Shortest 8/24 GFPO response does this once with a small recheck.

## B Extended Accuracy and Response Length Analysis

|                     | AIME 25     |              | AIME 24     |              | GPQA        |             | Omni-MATH   |              | LiveCodeBench |              |
|---------------------|-------------|--------------|-------------|--------------|-------------|-------------|-------------|--------------|---------------|--------------|
|                     | Acc         | Avg Len      | Acc         | Avg Len      | Acc         | Avg Len     | Acc         | Avg Len      | Acc           | Avg Len      |
| SFT                 | 64.2        | 10.9k        | 72.2        | 10.1k        | 67.0        | 6.6k        | 84.7        | 9.6k         | 57.7          | 10.3k        |
| GRPO                | 72.4        | 14.8k        | 77.7        | 13.3k        | 67.5        | 10.7k       | 86.0        | 12.7k        | 56.7          | 13.9k        |
| 6 of 8              | 69.2        | 14.7k        | 79.6        | 13k          | 70.2        | 10.2k       | 88.3        | 12.9k        | 56.4          | 13.6k        |
| 8 of 16             | <b>70.2</b> | 13.9k        | <b>77.9</b> | 12.3k        | <b>70.0</b> | 9.7k        | <b>89.3</b> | 11.8k        | <b>59.8</b>   | 12.6k        |
| 6 of 16             | 70.1        | 13.8k        | 76.9        | 12.2k        | 68.3        | 9.1k        | 87.8        | 11.4k        | 58.3          | <b>12.6k</b> |
| 4 of 16             | 69.7        | <b>13.3k</b> | 76.6        | <b>11.8k</b> | 68.6        | <b>8.8k</b> | 88.0        | <b>11.3k</b> | 57.2          | 12.3k        |
| 8 of 24             | <b>70.4</b> | <b>12.6k</b> | 75.1        | 11.6k        | 68.9        | 8.6k        | 87.5        | 11.1k        | 56.5          | <b>11.8k</b> |
| 6 of 24             | 68.5        | 13.1k        | 75.6        | 11.9k        | <b>70.2</b> | 8.7k        | <b>88.1</b> | 10.9k        | 58.7          | 12.4k        |
| 4 of 24             | 70.3        | 13k          | <b>76.5</b> | <b>11.3k</b> | 68.1        | <b>8.3k</b> | 87.6        | <b>10.5k</b> | <b>59.2</b>   | 11.8k        |
| Token Efficiency    | 69.5        | <b>12k</b>   | 76.4        | <b>10.6k</b> | 68.5        | <b>7.5k</b> | 87.4        | <b>10.1k</b> | 57.0          | <b>11k</b>   |
| Adaptive Difficulty | <b>70.8</b> | 12.8k        | <b>76.6</b> | 11.6k        | <b>70.8</b> | 9k          | <b>88.9</b> | 11.6k        | <b>57.2</b>   | 12.1k        |

Table 2: **Pass@1 Accuracy and Average Response Lengths on AIME 25, AIME 24, GPQA, Omni-MATH, and LiveCodeBench.** GFPO variants substantially reduce response lengths while matching GRPO accuracy. We find no statistically significant differences in accuracy under the Wilcoxon signed-rank test for any dataset.

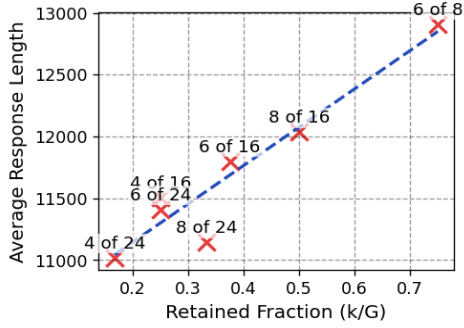


Figure 7: **Average Response Length vs  $k/G$ .** Reducing  $k/G$ , reduces average response length but beyond a point leads to diminishing returns.

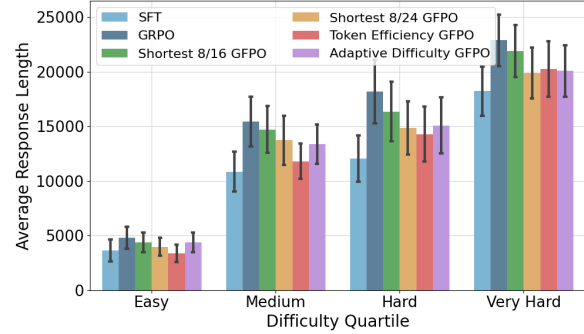


Figure 8: **Average Response Length Across Problem Difficulties.** Response lengths rise with problem difficulty for all methods, but GFPO reduces response length over all difficulty levels.

| Difficulty Bin | Method              | Short        | Mid-Low      | Mid-High     | Long         |
|----------------|---------------------|--------------|--------------|--------------|--------------|
| Hard           | SFT                 | 7298         | 9949         | 12576        | 18349        |
| Hard           | GRPO                | 12719        | 16292        | 19846        | 23834        |
| Hard           | Shortest 8/16       | 11292        | 13948        | 17897        | 22211        |
| Hard           | Shortest 8/24       | 10087        | 12839        | 15711        | 20837        |
| Hard           | Token Efficiency    | <b>8918</b>  | <b>12044</b> | <b>15337</b> | <b>20815</b> |
| Hard           | Adaptive Difficulty | 9593         | 12959        | 16126        | 21677        |
| Very Hard      | SFT                 | 10707        | 15630        | 20875        | 25666        |
| Very Hard      | GRPO                | 16728        | 22026        | 25309        | 27462        |
| Very Hard      | Shortest 8/16       | 15768        | 20786        | 24051        | 26935        |
| Very Hard      | Shortest 8/24       | <b>12657</b> | <b>18219</b> | 22671        | <b>25911</b> |
| Very Hard      | Token Efficiency    | 13034        | 18633        | 23223        | 26109        |
| Very Hard      | Adaptive Difficulty | 13096        | 18625        | <b>22276</b> | 26279        |

Table 3: **Average Response Length by Difficulty and Length Bins for AIME 25.** We bin each model’s responses to hard and very hard problems into length quartiles (short, mid-low, mid-high, long) and report the average response lengths across length bins. We **highlight** the shortest average response length per response length quartile across the different RL methods.

| Difficulty Bin | Method              | Short        | Mid-Low      | Mid-High     | Long         |
|----------------|---------------------|--------------|--------------|--------------|--------------|
| Hard           | SFT                 | 60.42        | 54.17        | 58.33        | 52.08        |
| Hard           | GRPO                | <b>83.33</b> | <b>85.42</b> | 64.58        | 60.42        |
| Hard           | Shortest 8/16       | 77.08        | 79.17        | 70.83        | 52.08        |
| Hard           | Shortest 8/24       | 72.92        | 81.25        | <b>72.92</b> | <b>66.67</b> |
| Hard           | Token Efficiency    | 75.00        | 68.75        | <b>72.92</b> | 54.17        |
| Hard           | Adaptive Difficulty | 62.50        | 64.58        | <b>72.92</b> | 56.25        |
| Very Hard      | SFT                 | 21.88        | 14.06        | 12.50        | 12.50        |
| Very Hard      | GRPO                | 32.81        | 31.25        | <b>25.00</b> | 17.19        |
| Very Hard      | Shortest 8/16       | 25.00        | 23.44        | 23.44        | <b>20.31</b> |
| Very Hard      | Shortest 8/24       | <b>35.94</b> | 25.00        | 18.75        | 10.94        |
| Very Hard      | Token Efficiency    | 26.56        | 29.69        | 17.19        | 15.63        |
| Very Hard      | Adaptive Difficulty | 29.69        | <b>35.94</b> | 23.44        | 18.75        |

Table 4: **Accuracy (%) by Difficulty and Length Bins for AIME 25.** We bin each model’s responses to hard and very hard problems into length quartiles (short, mid-low, mid-high, long) and report the accuracies across length bins. We **highlight** the highest accuracy per response length quartile across the different RL methods.