STABILIZING POLICY GRADIENTS FOR SAMPLE-EFFICIENT REINFORCEMENT LEARNING IN LLM REASONING

Luckeciano C. Melo[1,2] **Alessandro Abate**[2] **Yarin Gal**[1]
[1]OATML, University of Oxford [2]OXCAV, University of Oxford
luckeciano.carvalho.melo@cs.ox.ac.uk

ABSTRACT

Reinforcement Learning, particularly through policy gradient methods, has played a central role in enabling reasoning capabilities of Large Language Models. However, the optimization stability of policy gradients in this setting remains understudied. As a result, existing implementations often resort to conservative hyperparameter choices to ensure stability, which requires more training samples and increases computational costs. Hence, developing models for reliably tracking the underlying optimization dynamics and leveraging them into training enables more sample-efficient regimes and further unleashes scalable post-training. We address this gap by formalizing the stochastic optimization problem of policy gradients with explicit consideration of second-order geometry. We propose a tractable computational framework that tracks and leverages curvature information during policy updates. We further employ this framework to design interventions in the optimization process through data selection. The resultant algorithm, Curvature-Aware Policy Optimization (CAPO), identifies samples that contribute to unstable updates and masks them out. Theoretically, we establish monotonic improvement guarantees under realistic assumptions. On standard math reasoning benchmarks, we empirically show that CAPO ensures stable updates under aggressive learning regimes where baselines catastrophically fail. With minimal intervention (rejecting fewer than 8% of tokens), CAPO achieves up to $30\times$ improvement in sample efficiency over standard GRPO for LLM reasoning.

1 INTRODUCTION

The emergence of reasoning capabilities in Large Language Models (LLMs) represents a major shift in AI research. Beyond language understanding, reasoning has become a core ingredient of widely deployed systems (OpenAI et al., 2024; Gemini, 2025), enabling applications such as mathematical problem solving (Shao et al., 2024), code generation (Shojaee et al., 2023), and agentic workflows (Yao et al., 2023). This progress is primarily attributed to advances in scaling Reinforcement Learning (RL) techniques for LLM post-training, particularly policy gradient methods such as PPO (Schulman et al., 2017), GRPO (Shao et al., 2024), and variants (Yu et al., 2025; Liu et al., 2025b). These methods enabled LLMs to develop behaviors for autonomous chain-of-thought reasoning (Gandhi et al., 2025) and to effectively scale test-time compute (Setlur et al., 2025).

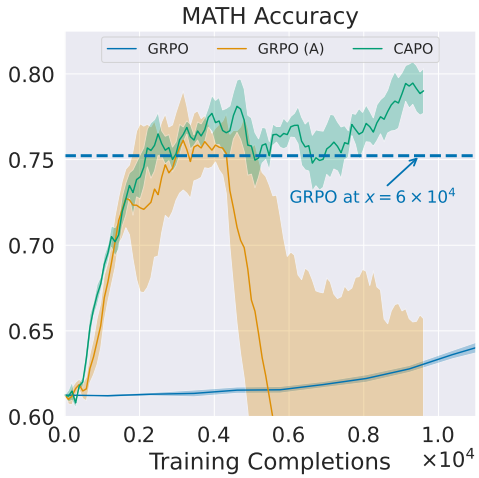

Figure 1: **Accuracy on MATH dataset from different RL methods.** CAPO (ours) achieves $30\times$ greater sample efficiency under an aggressive (A) update regime (higher learning rate, smaller batch size), whereas GRPO suffers policy collapse.

Despite its success in LLM fine-tuning and other decision-making tasks (Bellemare et al., 2020; Mnih et al., 2015), RL still faces fundamental challenges that limit its broader practicality and scalability. In particular, policy gradients suffer from optimization instabilities driven by the non-stationary nature of the RL objective and the high variance of estimates (Castanyer et al., 2025). These problems are further compounded by the known pathologies of training deep networks (Pascanu et al., 2013; Pennington et al., 2017). These factors lead to several undesired consequences, such as catastrophic updates and policy collapse (Dohare et al., 2023), plasticity loss (Juliani & Ash, 2024), sample inefficiency (Kaiser et al., 2020), and hyperparameter sensitivity (Henderson et al., 2018). As a result, the optimization dynamics of RL remain an active area of research from both theoretical and empirical standpoints (Mei et al., 2022; Lyle et al., 2022; Vaswani et al., 2022) .

Perhaps due to the recency of the topic, the optimization dynamics of RL *in the context of LLMs* remains underexplored. These challenges persist in the LLM setting and may be even more pronounced, since training involves billion-parameter models with very deep architectures and sampling horizons that can extend arbitrarily. In practice, current implementations of RL for LLMs typically rely on conservative hyperparameters to ensure stability, such as low learning rates (e.g., 3×10^{-6} or less) and large batch sizes (e.g. thousands of generations per policy update) (Sheng et al., 2024; Hugging Face, 2025; Guo et al., 2025). These choices substantially increase the number of LLM generations required for learning, raising computational costs. Therefore, stabilizing these algorithms in sample-efficient regimes is crucial to further scale RL for LLM reasoning.

One promising direction is to design algorithms that explicitly model second-order geometry in the optimization landscape and incorporate this information into policy updates. In this work, we formalize the RL optimization problem accounting for curvature terms, namely the Hessian of the objective and the Fisher Information Matrix of the policy distribution. Building on this formulation, we introduce a computationally and numerically tractable model of optimization dynamics that approximates this curvature information. This model enables continuous monitoring of gradient and curvature estimates during policy updates, scales to billion-parameter models and provides analytical expressions for these quantities, which facilitate a systematic analysis of the learning dynamics.

We further leverage this optimization model to *plan* the next policy gradient step[1]. It allows *anticipating* policy updates that potentially induce sudden shifts in the objective or policy distribution – often associated with unstable optimization behavior – and intervening before taking the actual step in the LLM. We propose a simple data selection mechanism as intervention: we identify particular samples that heavily contribute to these abrupt shifts and mask them out of the policy gradient estimation. We refer to this method as *Curvature-Aware Policy Optimization* (CAPO).

We theoretically establish monotonic policy improvement guarantees under CAPO with practical assumptions. We then empirically validate CAPO on standard math reasoning benchmarks, showing that it yields stable optimization even in regimes with aggressive updates, where standard RL algorithms suffer catastrophic updates and policy collapse. As a result, CAPO achieves up to $30\times$ improvement in sample efficiency compared to GRPO in the standard regime, as presented in Figure 1. Lastly, we show that its interventions are minimal, typically rejecting fewer than 8% of the tokens, with negligible computational overhead.

2 Related Work

RL & LLMs. The use of RL techniques to optimize LLMs has been an active area of research in recent years. Early work focused on RL from Human Feedback (RLHF), which optimizes policies toward modeled human preferences (Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022). More recently, RL for LLM reasoning has gained significant attention for its effectiveness in enabling autonomous chain-of-thought reasoning (Gandhi et al., 2025) and in scaling test-time compute (Setlur et al., 2025). This breakthrough was initially driven by the seminal works of the OpenAI o-series (OpenAI et al., 2024) and DeepSeek-R1 (Guo et al., 2025), which popularized GRPO (Shao et al., 2024). Since then, the research community has studied several aspects of the training pipeline (Zhang et al., 2025), including alternative objectives (Roux et al., 2025; Hu et al., 2025), sampling mechanisms (Yu et al., 2025), reward shapings (Yang et al., 2024), and different training configura-

[1]In this work, "model" refers to the proposed computational model of curvatures and "policy" to the LLM. "Model gradients" are computed under the former, while "policy gradients" denote the true LLM gradients.

tions (Liu et al., 2025b; Team et al., 2025). Our work fits within this line of research by investigating RL for LLMs from an optimization dynamics perspective, proposing a model of the optimization landscape and using it to design stable policy gradient updates.

Optimization Dynamics in RL. The non-convex and non-stationary nature of RL training has motivated a large body of work on understanding and stabilizing optimization dynamics in RL agents. In the context of policy gradients, prior research has investigated the role of baselines (Mei et al., 2022), variance reduction techniques (Greensmith et al., 2001), and emergent pathologies such as plasticity or capacity loss (Sokar et al., 2023; Klein et al., 2024) and policy collapse (Dohare et al., 2023). Beyond these analyses, past literature has also developed conservative policy optimization methods (Schulman et al., 2015; 2017; Achiam et al., 2017). While this line of work is extensive and evolving, we primarily highlight the recent contribution of Castanyer et al., which, like ours, examines the stabilization of policy gradients through curvature-informed interventions. Their methodology, however, differs: they apply natural gradients with K-FAC (Eschenhagen et al., 2023) in general deep RL environments, whereas our work develops a new approximation of curvature that is tractable at the scale of LLMs and is incorporates it into optimization through data selection.

Improving RL for LLM Reasoning. In the context of LLM research, a nascent but growing literature explores improvements to RL training for reasoning. These works typically propose heuristics that target specific problems observed during training—for example, noisy gradient estimates, limited output diversity, or large policy updates. Common approaches include rethinking advantage estimation (Liu et al., 2025a; Ahmadian et al., 2024), controlling policy entropy (Yu et al., 2025; Cui et al., 2025), and bounding advantage estimates or log-likelihoods Yang et al. (2025a;b). In contrast, our work takes a more principled approach. Rather than introducing heuristics to address isolated issues, we develop a framework based on second-order stochastic optimization that fundamentally explains these instabilities and addresses them in a unified manner.

3 PRELIMINARIES

Problem Statement. We formulate the problem of next-token generation as a Markov Decision Process (MDP), defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \rho_0, \gamma, T)$, in which $\mathcal{S}$ is a state space, $\mathcal{A}$ is an action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ a transition function, $R : \mathcal{S} \times \mathcal{A} \to [-r_{\text{bound}}, +r_{\text{bound}}]$ a bounded reward function, $\rho_0 : \mathcal{S} \to \Delta(\mathcal{S})$ an initial state distribution, $\gamma \in [0, 1]$ a discount factor, and T the length of the horizon. In the LLM setting, let $\mathcal{V}$ be a token vocabulary and $L \in \mathbb{N}$ a maximum sequence length, including both prompt and generated tokens. $\mathcal{S} = \bigcup_{n=0}^{L} \mathcal{V}^n$ is the set of all finite sequences, with each state $s_t \in \mathcal{S}$ representing the concatenation of the prompt and the tokens generated up to time t, with total length at most L. $\mathcal{A}$ is the space spanned by $\mathcal{V}$: at each step, the policy selects a token $a_t \in \mathcal{V}$. $\mathcal{P}$ is governed by autoregressive sampling and takes the form of a trivial deterministic function $s_{t+1} = s_t \circ a_t$, where $\circ$ denotes concatenation. The initial state distribution ρ_0 specifies a distribution over prompts. During policy optimization, one typically optimizes a parameterized LLM $\pi_{\boldsymbol{\theta}} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{A})$, with the objective of maximizing the expected cumulative reward over the generated sequence:

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}} \Big[\sum_{t=0}^{T} \gamma^t R(s_t, a_t) \Big], \tag{1}$$

where τ denotes a trajectory, $s_0 \sim \rho_0(s_0)$, $a_t \sim \pi_{\boldsymbol{\theta}}(a_t \mid s_t)$, and $s_{t+1} = s_t \circ a_t$.

Policy Gradient (PG) methods optimize a stochastic policy by differentiating $J(\boldsymbol{\theta})$ with respect to the policy parameters (Williams, 1992) and can be written as (Sutton et al., 1999):

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}} \left[\sum_{t=0}^{T} \gamma^t \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)\, R(s_t, a_t) \right]. \tag{2}$$

This expectation can be estimated via Monte Carlo sampling under the current policy $\pi_{\boldsymbol{\theta}}$. However, such estimates often have high variance. A standard remedy is to subtract a baseline $b(s_t)$ which leaves the gradient unbiased while reducing variance. In practice, this is typically done by replacing the reward with an estimate of the advantage function $A(s_t, a_t)$. For the rest of this work, we will assume the advantage version of this objective.

Group Relative Policy Optimization (Shao et al., 2024) is a widely used method for RL in LLMs. Akin to PPO (Schulman et al., 2017), it optimizes a surrogate objective that employs off-policy correction Kakade & Langford (2002) with a clipping strategy to prevent large deviations:

$$J_{\text{GRPO}}(\boldsymbol{\theta}) = \mathbb{E}_{\tau\sim\pi_\beta}\Big[\frac{1}{|\tau_i|}\sum_{t=0}^{|\tau_i|}\min\Big(r_\theta(s_t,a_t), \text{clip}(r_{\boldsymbol{\theta}}(s_t,a_t), 1-\epsilon, 1+\epsilon)\Big)A^{\text{GRPO}}(s_t,a_t) \\ -\beta\mathcal{D}_{\text{KL}}(\pi_{\boldsymbol{\theta}}(\cdot \mid s_t) \,\|\, \pi_{\text{base}}(\cdot \mid s_t))\Big], \tag{3}$$

where $r_{\boldsymbol{\theta}}(s_t,a_t) = \frac{\pi_{\boldsymbol{\theta}}(a_t|s_t)}{\pi_\beta(a_t|s_t)}$ and π_β is the sampling policy. The KL divergence term acts as a regularizer that penalizes deviation from π_{base}, the initial LLM. In contrast to standard PG methods, GRPO draws samples in groups: for each prompt $s_0 \sim \rho_0(s_0)$, it generates a group of trajectories $\{\tau_i\}_{i=1}^G \sim \pi_\beta$. Contributions from all state-action pairs of a trajectory are averaged (rather than discounted), which effectively assume $\gamma = 1$ with per-trajectory normalization. Finally, the advantage estimator is defined as:

$$\hat{A}^{\text{GRPO}}(s_t,a_t) \;=\; \frac{\hat{R}(\tau)-\bar{R}}{\hat{\sigma}_R+\varepsilon}, \quad \bar{R} \;=\; \frac{1}{G}\sum_{i=1}^{G}\hat{R}(\tau_i), \quad \hat{\sigma}_R \;=\; \sqrt{\frac{1}{G}\sum_{i=1}^{G}\big(\hat{R}(\tau_i)-\bar{R}\big)^2}, \tag{4}$$

where $\hat{R}(\tau)$ is the return for trajectory τ and ε is a small constant for numerical stability.

4 Modeling the Optimization Landscape with Second-Order Geometry

In this section, we develop a model of the optimization landscape. We formulate the reinforcement learning (RL) optimization problem with policy gradients by explicitly incorporating second-order geometric information. Building on this formulation, we introduce a tractable computational model that approximates the role of curvature during learning. Our hypothesis is that by providing a simple but effective approximation of second-order gradients, one could track sudden shifts in the objective or policy and anticipate potentially unstable updates.

The Higher-Order Objective. Consider the objective function $J(\boldsymbol{\theta})$ as in Equation 1. After an update step $\Delta\boldsymbol{\theta}$, the new objective $J(\boldsymbol{\theta}+\Delta\boldsymbol{\theta})$ is given by the following Taylor expansion:

$$J(\boldsymbol{\theta}+\Delta\boldsymbol{\theta}) = J(\boldsymbol{\theta}) + \underbrace{\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta})^\top\Delta\boldsymbol{\theta} + \tfrac{1}{2}\,\Delta\boldsymbol{\theta}^\top H(\boldsymbol{\theta})\,\Delta\boldsymbol{\theta}}_{m_H(\Delta\boldsymbol{\theta})} + \mathcal{O}(\|\Delta\boldsymbol{\theta}\|^3), \tag{5}$$

where $H(\boldsymbol{\theta})$ denotes the Hessian of the objective. Equation 5 holds under a Lipschitz continuous Hessian (see Assumption A.1), with a detailed proof in Appendix A. As the cubic term may be negative, we can establish a guaranteed lower bound $J(\boldsymbol{\theta}+\Delta\boldsymbol{\theta}) \geq J(\boldsymbol{\theta}) + m_H(\Delta\boldsymbol{\theta}) - \mathcal{O}(\|\Delta\boldsymbol{\theta}\|^3)$. In practice, the cubic term is often negligible, and we approximate the objective change by $m_H(\Delta\boldsymbol{\theta})$. Crucially, standard gradient ascent ignores the Hessian contribution, which can lead to a decrease in the objective for non-convex problems (such as RL) when this contribution is sufficiently negative.

The Fisher Information Matrix. The Hessian captures the local curvature of the objective function. In RL, however, the objective is non-stationary, and what ultimately matters is how updates change the policy distribution. For instance, an update may produce only a small change in the objective while inducing a large shift in the policy. This alters how future trajectories are sampled and may destabilize learning. Therefore, it is necessary to track the geometry of the policy distribution directly, which is what the Fisher Information Matrix (FIM) enables. One can show that the directional curvature under the Fisher geometry approximates the average KL divergence between a policy and before and after a small step $\Delta\boldsymbol{\theta}$:

$$\bar{D}_{\text{KL}}(\pi_{\boldsymbol{\theta}} \,\|\, \pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}) = \underbrace{\tfrac{1}{2}\Delta\boldsymbol{\theta}^\top F(\boldsymbol{\theta})\Delta\boldsymbol{\theta}}_{m_F(\Delta\boldsymbol{\theta})} + \mathcal{O}(\|\Delta\boldsymbol{\theta}\|^3), \tag{6}$$

where $\bar{D}_{\text{KL}}(\pi_{\boldsymbol{\theta}} \,\|\, \pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}) := \mathbb{E}_{s\sim d_\pi}\big[\text{KL}\big(\pi_{\boldsymbol{\theta}}(\cdot \mid s) \,\|\, \pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}(\cdot \mid s)\big)\big]$, and $F(\boldsymbol{\theta}) := \mathbb{E}_{s\sim d_\pi,\, a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}\Big[\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a \mid s)\,\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a \mid s)^\top\Big]$ is the FIM. The proof is in Appendix

B. Similarly to the Hessian case, the cubic term is often negligible and we focus on $m_F(\Delta\boldsymbol{\theta})$. One can further show that enforcing a trust region $\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}} \,\|\, \pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}) \leq \delta$ during policy updates leads to monotonic improvement of the true objective, given sufficiently small δ (Schulman et al., 2015).

Ultimately, we aim to design a model that approximates $m_H(\Delta\boldsymbol{\theta})$ and $m_F(\Delta\boldsymbol{\theta})$ without explicitly computing gradients or curvature terms in the high-dimensional parameter space of the LLM. This approach can be viewed as a form of model-based RL, but from a different perspective: whereas prior work typically models components of the MDP, such as the dynamics or reward function, we instead model the optimization process itself, which allows us to plan gradient estimates.

4.1 Computational Model

For an LLM with d parameters, both Hessian and FIM are $d \times d$ matrices, which is intractable for billion-size parameter spaces. Even approximations such as K-FAC (Eschenhagen et al., 2023) would incur unfeasible memory cost. Therefore, we need to devise a computational model that is scalable and effectively provides curvature information to stabilize policy gradients. Next, we describe our methodology.

Last-Layer Model. Since modeling the full Hessian or Fisher Information Matrix (FIM) is infeasible, we restrict attention to curvature in a parameter subspace. To this end, we adopt a simple last-layer approach. An LLM is a softmax policy over the token vocabulary $\pi_{\boldsymbol{\theta}}(a \mid s) = \frac{\exp(f_\theta(s,a))}{\sum_{a'} \exp(f_\theta(s,a'))}$, where $f_\theta(s,a) \in \mathbb{R}$ are the logits produced by the network. Letting $f_{\boldsymbol{\theta}}(s_t)$ denote the full logits vector, with $\boldsymbol{\theta} = (\bar{\boldsymbol{\theta}}, \boldsymbol{\psi})$, we represent the pre-softmax layer as $f_{\boldsymbol{\theta}}(s_t) = W h_{\bar{\boldsymbol{\theta}}}(s_t)$, where $W \in \mathbb{R}^{K \times d_i}$ is the last-layer weight matrix, $K = \dim(\mathcal{V})$, and $h_{\bar{\boldsymbol{\theta}}}(s_t) \in \mathbb{R}^{d_i}$. We then define $\boldsymbol{\psi} = \mathrm{vec}(W) \in \mathbb{R}^{K \cdot d_i}$. In Appendix C, we show that the last-layer model gradient $\tilde{g}(\boldsymbol{\psi})$ of the objective in Equation 1 is:

$$\tilde{g}(\boldsymbol{\psi}) = \mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}} \left[\sum_{t=0}^{T} \gamma^t A(s_t, a_t)(e_a - \pi_{\boldsymbol{\theta}}(s_t)) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t) \right], \tag{7}$$

where $\otimes$ denotes a Kronecker product, $e_{a_t} \in \mathcal{V}$ is the one-hot action vector $e_{a_t} = \mathbf{1}\{a = a_t\}$, and $\pi_{\boldsymbol{\theta}}(s_t))$ the policy distribution vector. We use the vectorization operation $\mathrm{vec}(\cdot)$ only for convenience and it does not introduce new assumptions. In this work, we use a tilde superscript to denote *model-based* gradients and curvatures, in contrast to the actual *policy* gradient $g(\boldsymbol{\theta}) := \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$.

Under the last-layer model, the Hessian of the objective takes the following form:

$$\tilde{H}(\boldsymbol{\psi}) = \mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}} \left[\sum_{t=0}^{T} \gamma^t A(s,a) \Big((e_a - \pi_{\boldsymbol{\theta}}(s_t))(e_a - \pi_{\boldsymbol{\theta}}(s_t))^\top - F(s_t) \Big) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t) h_{\bar{\boldsymbol{\theta}}}(s_t)^\top \right], \tag{8}$$

where $F(s_t)$ is the FIM for state s_t. In Lemma C.1, we show that this expression can be estimated via samples. Similarly, the last-layer approximation of the FIM is:

$$\tilde{F}(\boldsymbol{\psi}) = \mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}} \left[\big((e_{a_t} - \pi_{\boldsymbol{\theta}}(s_t))(e_{a_t} - \pi_{\boldsymbol{\theta}}(s_t))^\top \big) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t) h_{\bar{\boldsymbol{\theta}}}(s_t)^\top \right]. \tag{9}$$

Computing *Directional* Curvatures. Even with the approximated model, the curvature matrices have dimension $Kd_i \times Kd_i$. For current LLMs, where $K > 10^5$ and $d_i > 10^3$, fully materializing these matrices is computationally infeasible. Fortunately, our goal is to approximate the shifts in the objective and policy, $m_H(\Delta\boldsymbol{\theta})$ and $m_F(\Delta\boldsymbol{\theta})$. Thus, we only need to approximate the *directional* curvatures $\Delta\boldsymbol{\theta}^\top C(\boldsymbol{\theta})\, \Delta\boldsymbol{\theta}$, without explicitly materializing the full Hessian or FIM. In Appendix D, we present a mechanism that enables this computation without constructing large tensors. Our method requires storing only $\mathcal{O}(Kd_i)$ tensors per state–action sample, instead of the $\mathcal{O}((Kd_i)^2)$ entries of the full curvature matrices.

Exploiting Gradient Sparsity. We further reduce complexity by exploiting the structure of gradients arising from LLM generation. Standard LLM decoding relies on selective sampling methods (e.g., top-k, nucleus sampling) Wolf et al. (2020) to improve generation quality, as most of the probability mass is concentrated on a small subset k of the vocabulary (Fan et al., 2018; Holtzman et al., 2020), typically with $k < 100$. Consequently, only k tokens have non-zero probability at each generation step, which implies that only the $k * d_i$ parameters of the last-layer weight matrix W associated

with these logits yield non-zero gradients. We therefore store and operate these gradients in sparse form. This sparsity also applies to the computation of directional curvatures in Equations 58 and 63, as these reduce to dot products involving sparse vectors (e.g., $(e_{a_t} - \pi_{\boldsymbol{\theta}}(s_t)$ and the model-based update step $\Delta\boldsymbol{\theta}$). Naturally, as we estimate gradients with more samples, the representation expands to cover all $\tilde{k}$ tokens generated, but typically $\tilde{k} << K$. For instance, our experiments presented $\tilde{k} < 10^4$. Overall, the memory and dot product complexity reduce to $\mathcal{O}(\tilde{k} \cdot d_i)$.

Modeling the Step $\Delta\boldsymbol{\theta}$. A final design choice concerns how to model the planned update steps, $\Delta\boldsymbol{\theta}$. Under the last-layer model, these steps take the form $\Delta\boldsymbol{\psi}$. This choice essentially determines how we represent the optimizer. A simple option is to model the update as a stochastic gradient descent (SGD) step, $\Delta\boldsymbol{\psi} = \alpha\tilde{g}$, where α is the learning rate. Alternatively, we can match the LLM optimizer, which in our case is Adam (Kingma & Ba, 2015), i.e., $\Delta\boldsymbol{\psi} = \alpha\frac{\hat{p}_t}{\sqrt{\hat{q}_t}+\epsilon}$, where $\hat{p}_t$ and $\hat{q}_t$ are the bias-corrected first and second moment estimates of the gradient.

5 Curvature-Aware Policy Optimization

We may now compute the objective and policy shifts under our model as:

$$m_H(\boldsymbol{\psi}) = \tilde{g}(\boldsymbol{\psi})^\top \Delta\boldsymbol{\psi} + \tfrac{1}{2}\Delta\boldsymbol{\psi}^\top \tilde{H}(\boldsymbol{\psi})\Delta\boldsymbol{\psi}, \quad m_F(\boldsymbol{\psi}) = \tfrac{1}{2}\Delta\boldsymbol{\psi}^\top \tilde{F}(\boldsymbol{\psi})\Delta\boldsymbol{\psi}, \tag{10}$$

and estimate m_H and m_F via samples following the methodology described in the subsection 4.1. We now design an algorithm that intervenes in the optimization of the underlying LLM policy using the model-based updates. Since our objective is to stabilize policy gradients in sample-efficient regimes, a natural choice is to construct an algorithm that follows the principles of trust-region methods (Murphy, 2022). We implement this idea through a rejection sampling mechanism.

Given a batch $\mathcal{B}$ of collected trajectories, we partition it into disjoint subsets $b_i \subset \mathcal{B}$. For each subset, we compute a proposed step $\Delta\boldsymbol{\psi}_i$ and evaluate the shifts defined in Equation 10. We then accept a subset if it satisfies the (local) trust-region constraints δ_F, δ_H, and δ_H^{high}:

$$\delta_H \leq m_H(\Delta\psi_i) \;\leq\; \delta_H^{high}, \qquad m_F(\Delta\psi_i) \;\leq\; \delta_F. \tag{11}$$

The accepted subsets are subsequently used to compute the gradient update of the LLM policy. Conceptually, this mechanism is analogous to token masking. Overall, this data selection mechanism is simple, computationally inexpensive, and flexible, as it can be applied at different granularities, including tokens, sentences, groups, or full batches. The formal pseudocode is provided in Algorithm 1. Next, we establish theoretical results for monotonic policy improvement under CAPO.

Theorem 5.1 (Monotonic improvement under CAPO)**.** *Fix thresholds $\delta_H > 0$ and $\delta_F > 0$. Let $\mathcal{B}$ be a batch of sampled trajectories. Split $\mathcal{B}$ into disjoint N subsets $b_i \subset \mathcal{B}$, and propose candidate subset updates $\{\Delta\theta_i\}_{i:N}$. Retain those satisfying:*

$$m_H(\Delta\theta_i) \;\geq\; \delta_H = \omega + \tfrac{1}{2}Mr^2, \qquad m_F(\Delta\theta_i) \;\leq\; \delta_F, \tag{12}$$

with $\omega > 0$ and M, r defined as in Assumption E.1. Let $\mathcal{B}_{acc}$ denote the superset of the B accepted subsets, and define the aggregated update:$\Delta\theta \;=\; \frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} \Delta\theta_i$. Then, for two policies $\pi_{\boldsymbol{\theta}}$ and $\pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}$, with $|A^\pi(s,a)| \leq \epsilon$, we obtain:

$$J(\pi_{\theta+\Delta\theta}) - J(\pi_\theta) \;\geq\; \omega \;-\; C\sqrt{\delta_F}, \quad C = \frac{2\gamma}{(1-\gamma)^2}\,\epsilon\sqrt{2}. \tag{13}$$

Thus choosing $\omega \geq C\sqrt{\delta_F}$ guarantees monotonic improvement: $J(\pi_{\theta+\Delta\theta}) \geq J(\pi_\theta)$.

The proof is provided in Appendix E. Observe that δ_H^{high} is not required to establish monotonic improvement. Nonetheless, it serves as a safeguard against overly aggressive steps. In practice, introducing this upper cap reduces the observed M and r, which allows the use of smaller δ_H. Finally, we note that Theorem 5.1 relies on the true objective and policy shifts, whereas in practice these quantities are approximated using our model.

6 Experiments and Discussion

In this section, we evaluate (i) how the proposed computational model captures the optimization landscape, and (ii) how this information can be used to stabilize RL optimization dynamics through

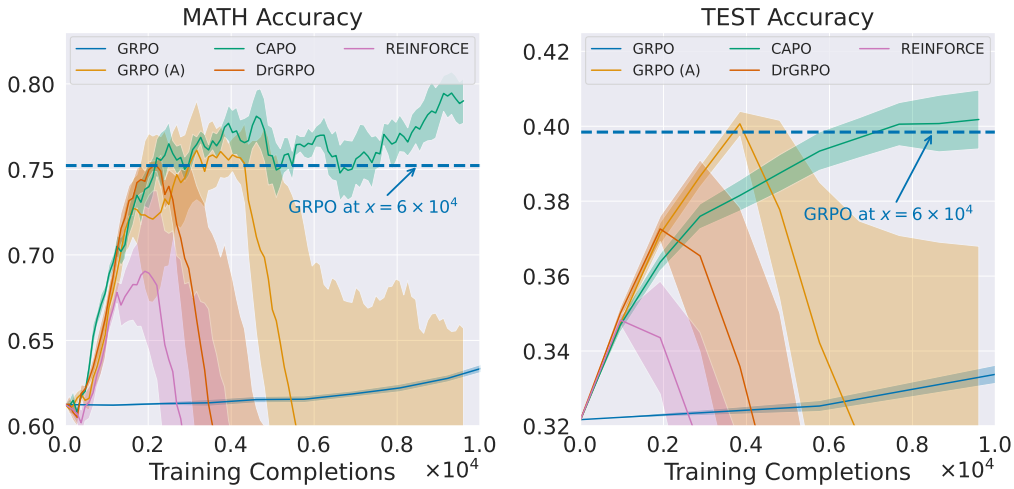

Figure 2: **Comparison with baseline methods on policy gradient stability.** While the setup with more aggressive updates makes all methods more sample-efficient, it also leads the baselines to policy collapse. In contrast, CAPO prevents collapse and achieves up to 30× greater sample efficiency than GRPO under aggressive updates.

CAPO. Our central hypothesis is that an inexpensive yet effective approximation of second-order geometry can track unstable shifts in the objective and policy, and that this information can in turn be used to stabilize aggressive update regimes, leading to more sample-efficient RL in LLMs.

Experimental Setup. We consider a standard RL setup for finetuning LLMs on reasoning tasks. Our implementation builds on the Open-R1 open-source project (Hugging Face, 2025), and we maximize an accuracy-based reward. Following prior work, we fine-tune a Qwen2.5-Math-7B LLM (Qwen et al., 2025) on mathematical reasoning questions. Our primary evaluation metric is accuracy, but we also track optimization-related quantities such as gradient and curvature statistics and token rejection rates. Since our goal is to evaluate sample efficiency, we report all metrics as a function of the number of training completions (i.e., LLM *trajectories* generated). Appendix G provides additional details regarding implementation, hyperparameters, and compute resources[2].

Datasets & Benchmarks. We train our policies on the MATH dataset (Hendrycks et al., 2021). For evaluation, we consider eight benchmarks: GSM8K (Cobbe et al., 2021), MATH500 (Lightman et al., 2023), OlympiadBench (He et al., 2024), MinervaMath (Lewkowycz et al., 2022), GPQA:Diamond (Rein et al., 2023), AMC23, AIME24, and AIME25. Most of these benchmarks contain mathematical questions at varying levels (high school, graduate, and olympiad), while GPQA focuses on general STEM-related problems. For simplicity, we report the average performance across all eight benchmarks, which we refer to as "TEST" in the results.

Comparison Methods. We evaluate our approach against two GRPO variants. The first corresponds to the standard "conservative" update regime implemented in the Open-R1 codebase. The second, which we denote "GRPO (A)," adopts a more aggressive regime intended to improve sample efficiency, with a learning rate $5\times$ higher and a batch size $12\times$ smaller. This matches the configuration used by CAPO. We also evaluate Dr.GRPO (Liu et al., 2025a) and REINFORCE (Williams, 1992), both under the same aggressive regime.

CAPO operationalization. CAPO optimizes the same objective as GRPO, but leverages the data selection mechanism introduced in Section 5. For a fair comparison, we use the same hyperparameters as GRPO (A). We implement CAPO with token-level selection, i.e., proposing steps $\Delta\psi_i$ and rejecting samples on a per-token basis. Finally, we model optimization steps using Adam.

6.1 Experiments

We highlight and analyze the following questions to evaluate our hypothesis and proposed method:

[2]We release our code at `https://github.com/luckeciano/stable-pg-llm`.

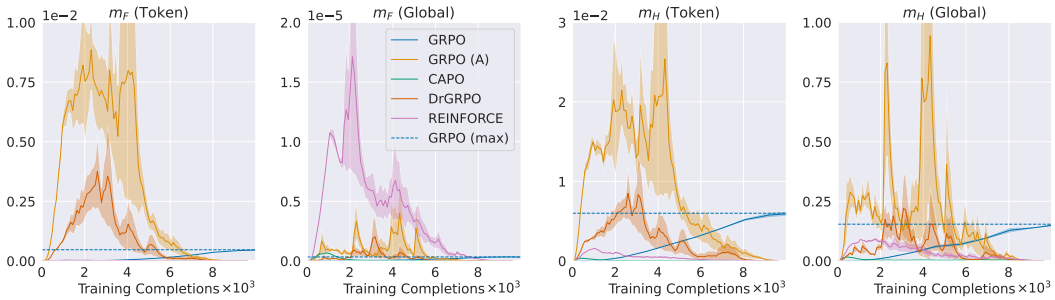

Figure 3: **Evaluation of policy and objective shifts estimates from the proposed computational model during training.** Unstable methods exhibit large and abrupt directional curvatures, while stable ones maintain much smaller and smoother shifts. CAPO, by applying token-level bounds, also ensures well-behaved shifts at the global (batch) level, supporting the rationale of Theorem 5.1.

Does CAPO prevent instability in LLM policy gradients? Does it lead to better sample efficiency? Figure 2 reports accuracy for all methods on MATH and on the TEST benchmark set. First, we observe that the more aggressive setup does lead to more sample-efficient learning than the conservative one across all methods. However, for the baselines, this improvement comes at the cost of stability. Under the aggressive regime, all baseline methods suffer from policy collapse, with performance dropping well below that of the base model and therefore losing the ability to learn further. In contrast, CAPO maintains stable performance throughout training, remaining effective long after all other methods have collapsed. This demonstrates that CAPO effectively prevents instability under aggressive updates. As a result, CAPO requires $30\times$ fewer completions on MATH and $9\times$ fewer completions on TEST compared to standard conservative GRPO.

What does the proposed computational model reveal about the optimization landscape? To analyze this question, we examine the policy shift m_F and the objective shift m_H at both the token level and the global (batch) level over the course of training, presented in Figure 3. For m_F, we find that unstable methods (GRPO (A), DrGRPO, REINFORCE) exhibit very high global directional curvatures during training, whereas stable methods (CAPO, standard GRPO) maintain much smaller shifts. In particular, the global m_F correlates closely with the instability observed in Figure 1, showing that the model, despite its simplicity, remains informative about optimization dynamics.

For m_H, we observe similar trends: unstable methods show abrupt shifts, while stable ones produce smoother, better-behaved curves. Note that, while a higher m_F directly signals instability since it tracks policy shifts, a higher m_H does not necessarily directly imply instability. This is because m_H depends on the adopted advantage function (Equation 37) and the normalization strategy of each method. Still, sharp peaks in the m_H curves also correlate with training instabilities. Lastly, we highlight that CAPO, by applying a local bound per token, also ensures well-behaved shifts at the global level, which supports the rationale of Theorem 5.1. Overall, these results highlight that the computational model provides meaningful information about the optimization landscape, and that CAPO effectively leverages this information to stabilize training.

Can we extend curvature-aware selection to other RL methods? To test this, we extend Dr.GRPO and REINFORCE by incorporating our proposed curvature-aware selection, resulting in Dr.CAPO and ReinCAPO, respectively. Figure 4 reports the evaluation results for these methods. In all cases, incorporating the selection strategy improves upon the base method and prevents policy collapse. These findings suggest that the proposed computational model and intervention mechanism are broadly applicable across different policy optimization objectives.

How aggressive is CAPO's intervention to ensure stability? We analyze the extent of token rejection required by CAPO to maintain stable gradients, measured by the token rejection rate during training (Figure 5). The rejection rate peaks at about 8% in the early stages of opti-

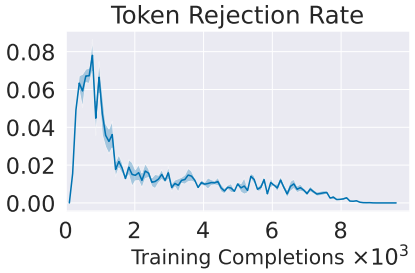

Figure 5: **Token rejection rate under CAPO.** It maintains a low rejection rate over training, stabilizing learning with minimal intervention.

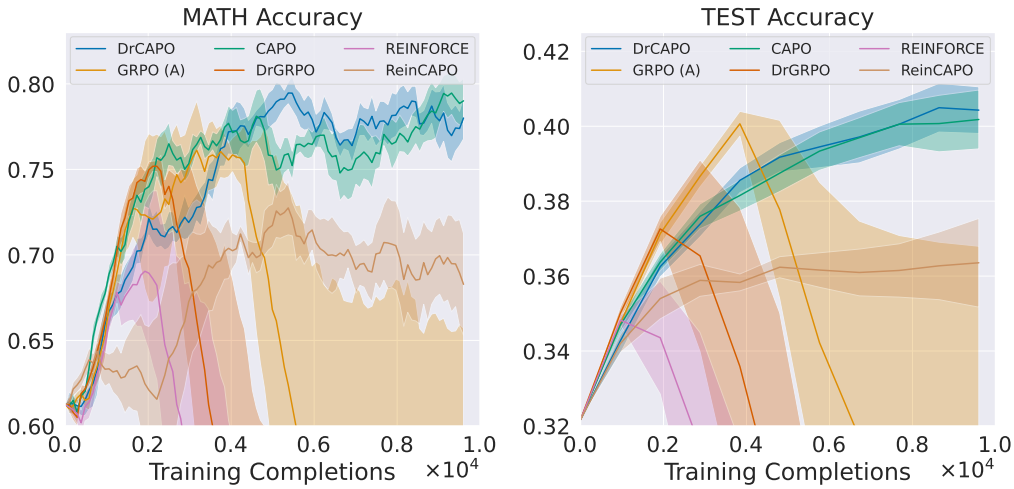

Figure 4: **Evaluation of extended versions of RL methods with curvature-aware selection.** Incorporating curvature-aware selection consistently improves the base methods, preventing policy collapse and demonstrating the broader applicability of our approach across different policy optimization objectives.

mization, when higher learning rates produce more aggressive updates, but quickly decreases and remains below 2% for the remainder of training. Overall, this shows that CAPO guides optimization toward stable curvature regions while keeping its intervention minimal, allowing the LLM to continue leveraging the vast majority of samples.

Additional Experiments. We provide a computational cost analysis of CAPO in Appendix H, where we show that the additional components incur minor overhead. Additionally, we present further experiments in Appendix I, including an ablation study on the optimizer model and a detailed evaluation of other heuristics traditionally used to ensure stability (e.g., PPO clipping and KL regularization), highlighting their limitations in the LLM setup.

7 Final Remarks

In this work, we propose a computational framework that models curvature information and integrates it into policy updates through CAPO. We provide theoretical guarantees for CAPO and show that it is effective at identifying samples that contribute to unstable updates, preventing policy collapse in aggressive training regimes where standard RL methods for LLM reasoning fail. As a result, CAPO achieves up to a $30\times$ improvement in sample efficiency compared to widely used training setups, while requiring only minimal intervention and computational overhead. Overall, it enables more sample-efficient learning regimes, supporting further scalability post-training scalability.

Limitations. Despite the encouraging results, we acknowledge some limitations of our work. First, due to compute budget constraints, we focused on experiments at a smaller, academic scale. While we demonstrated the effectiveness of CAPO against commonly used RL methods, future work could extend these results to distinct problem settings and longer training schedules. Second, the choice of CAPO thresholds depends on the problem setting (MDP, objective function, base policy) and may require tuning across different scenarios. Nonetheless, this is not a major concern, as the thresholds can be tuned solely on the training distribution.

Future Work. Beyond scalability, future research may explore different parametrizations of the computational model (for instance, by extending it to deeper layers) and investigate their impact on computational tractability and curvature estimates. In addition, future work may evaluate CAPO extensions to other intervention mechanisms, such as soft masking or regularization methods.

ACKNOWLEDGMENTS

We thank Shreshth Malik, Clare Lyle, and Yoav Gelberg for the insightful discussions in the early stages of this project. Luckeciano C. Melo acknowledges funding from the Air Force Office of Scientific Research (AFOSR) European Office of Aerospace Research & Development (EOARD) under grant number FA8655-21-1-7017. Yarin Gal is supported by a Turing AI Fellowship financed by the UK government's Office for Artificial Intelligence, through UK Research and Innovation (grant reference EP/V030302/1) and delivered by the Alan Turing Institute. We gratefully acknowledge compute resources provided by the Alan Turing Institute. This work was also funded under the Horizon Europe grant 101213369 DVPS.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 22–31. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/achiam17a.html.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.662. URL https://aclanthology.org/2024.acl-long.662/.

S. Amari, A. Cichocki, and H. H. Yang. A new learning algorithm for blind signal separation. In *Proceedings of the 9th International Conference on Neural Information Processing Systems*, NIPS'95, pp. 757–763, Cambridge, MA, USA, 1995. MIT Press.

Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998. doi: 10.1162/089976698300017746.

Marc Bellemare, Salvatore Candido, Pablo Castro, Jun Gong, Marlos Machado, Subhodeep Moitra, Sameera Ponda, and Ziyu Wang. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588:77–82, 12 2020. doi: 10.1038/s41586-020-2939-8.

Lukas Biewald. Experiment tracking with weights and biases, 2020. URL https://www.wandb.com/. Software available from wandb.com.

Roger Creus Castanyer, Johan Obando-Ceron, Lu Li, Pierre-Luc Bacon, Glen Berseth, Aaron Courville, and Pablo Samuel Castro. Stable gradients for stable learning at scale in deep reinforcement learning, 2025. URL https://arxiv.org/abs/2506.15544.

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

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. The entropy mechanism of reinforcement learning for reasoning language models, 2025. URL https://arxiv.org/abs/2505.22617.

Shibhansh Dohare, Qingfeng Lan, and A. Rupam Mahmood. Overcoming policy collapse in deep reinforcement learning. In *Sixteenth European Workshop on Reinforcement Learning*, 2023. URL https://openreview.net/forum?id=m9Jfdz4ymO.

Runa Eschenhagen, Alexander Immer, Richard Turner, Frank Schneider, and Philipp Hennig. Kronecker-factored approximate curvature for modern neural network architectures. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 33624–33655. Curran Associates, Inc.,

2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/6a6679e3d5b9f7d5f09cdb79a5fc3fd8-Paper-Conference.pdf.

Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1082. URL https://aclanthology.org/P18-1082/.

Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, Nathan Lile, and Noah D. Goodman. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars, 2025. URL https://arxiv.org/abs/2503.01307.

Gemini. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities, 2025. URL https://arxiv.org/abs/2507.06261.

Evan Greensmith, Peter L. Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. In *Proceedings of the 15th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, pp. 1507–1514, Cambridge, MA, USA, 2001. MIT Press.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL https://doi.org/10.1038/s41586-025-09422-z.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems, 2024.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018. doi: 10.1609/aaai.v32i1.11694. URL https://ojs.aaai.org/index.php/AAAI/article/view/11694.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. Reinforce++: An efficient rlhf algorithm with robustness to both prompt and reward models, 2025. URL https://arxiv.org/abs/2501.03262.

Hugging Face. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL https://github.com/huggingface/open-r1.

Arthur Juliani and Jordan T. Ash. A study of plasticity loss in on-policy deep reinforcement learning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 113884–113910. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/ce7984e36d58659211a8dc7d5457cd6f-Paper-Conference.pdf.

Kaiser, Mohammad Babaeizadeh, Piotr Miłos, Błażej Osiński, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model based reinforcement learning for atari. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1xCPJHtDB.

Sham M. Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, 2002. URL https://api.semanticscholar.org/CorpusID:31442909.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

Timo Klein, Lukas Miklautz, Kevin Sidak, Claudia Plant, and Sebastian Tschiatschek. Plasticity loss in deep reinforcement learning: A survey, 2024. URL https://arxiv.org/abs/2411.04832.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models, 2022. URL https://arxiv.org/abs/2206.14858.

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

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. In *2nd AI for Math Workshop @ ICML 2025*, 2025a. URL https://openreview.net/forum?id=jLpC1zavzn.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective, 2025b. URL https://arxiv.org/abs/2503.20783.

Clare Lyle, Mark Rowland, Will Dabney, Marta Kwiatkowska, and Yarin Gal. Learning dynamics and generalization in deep reinforcement learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 14560–14581. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/lyle22a.html.

Jincheng Mei, Wesley Chung, Valentin Thomas, Bo Dai, Csaba Szepesvari, and Dale Schuurmans. The role of baselines in policy gradient optimization. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 17818–17830. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/718d02a76d69686a36eccc8cde3e6a41-Paper-Conference.pdf.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 00280836. URL http://dx.doi.org/10.1038/nature14236.

Kevin P. Murphy. *Probabilistic Machine Learning: An introduction*. MIT Press, 2022. URL http://probml.github.io/book1.

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrey Mishchenko, Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghorbani, Ben Rossen, Benjamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao Hao, Bowen Baker, Brandon Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lugaresi, Cary Bassin, Cary Hudson, Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen, Chong Zhang, Chris Koch, Chris Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan Roberts, Daniel Kappler, Daniel Levy, Daniel Selsam, David Dohan, David Farhi, David Mely, David Robinson, Dimitris Tsipras, Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Edmund Wong, Elizabeth Proehl, Enoch Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan Mays, Fan Wang, Felipe Petroski Such, Filippo Raso, Florencia Leoni, Foivos Tsimpourlas, Francis Song, Fred von Lohmann, Freddie Sulit, Geoff Salmon, Giambattista Parascandolo, Gildas Chabot, Grace Zhao, Greg Brockman, Guillaume Leclerc, Hadi Salman, Haiming Bao, Hao Sheng, Hart Andrin, Hessam Bagherinezhad, Hongyu Ren, Hunter Lightman, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian Osband, Ignasi Clavera Gilaberte, Ilge Akkaya, Ilya Kostrikov, Ilya Sutskever, Irina Kofman, Jakub Pachocki, James Lennon, Jason Wei, Jean Harb, Jerry Twore, Jiacheng Feng, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joaquin Quiñonero Candela, Joe Palermo, Joel Parish, Johannes Heidecke, John Hallman, John Rizzo, Jonathan Gordon, Jonathan Uesato, Jonathan Ward, Joost Huizinga, Julie Wang, Kai Chen, Kai Xiao, Karan Singhal, Karina Nguyen, Karl Cobbe, Katy Shi, Kayla Wood, Kendra Rimbach, Keren Gu-Lemberg, Kevin Liu, Kevin Lu, Kevin Stone, Kevin Yu, Lama Ahmad, Lauren Yang, Leo Liu, Leon Maksin, Leyton Ho, Liam Fedus, Lilian Weng, Linden Li, Lindsay McCallum, Lindsey Held, Lorenz Kuhn, Lukas Kondraciuk, Lukasz Kaiser, Luke Metz, Madelaine Boyd, Maja Trebacz, Manas Joglekar, Mark Chen, Marko Tintor, Mason Meyer, Matt Jones, Matt Kaufer, Max Schwarzer, Meghan Shah, Mehmet Yatbaz, Melody Y. Guan, Mengyuan Xu, Mengyuan Yan, Mia Glaese, Mianna Chen, Michael Lampe, Michael Malek, Michele Wang, Michelle Fradin, Mike McClay, Mikhail Pavlov, Miles Wang, Mingxuan Wang, Mira Murati, Mo Bavarian, Mostafa Rohaninejad, Nat McAleese, Neil Chowdhury, Neil Chowdhury, Nick Ryder, Nikolas Tezak, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Patrick Chao, Paul Ashbourne, Pavel Izmailov, Peter Zhokhov, Rachel Dias, Rahul Arora, Randall Lin, Rapha Gontijo Lopes, Raz Gaon, Reah Miyara, Reimar Leike, Renny Hwang, Rhythm Garg, Robin Brown, Roshan James, Rui Shu, Ryan Cheu, Ryan Greene, Saachi Jain, Sam Altman, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Santiago Hernandez, Sasha Baker, Scott McKinney, Scottie Yan, Shengjia Zhao, Shengli Hu, Shibani Santurkar, Shraman Ray Chaudhuri, Shuyuan Zhang, Siyuan Fu, Spencer Papay, Steph Lin, Suchir Balaji, Suvansh Sanjeev, Szymon Sidor, Tal Broda, Aidan Clark, Tao Wang, Taylor Gordon, Ted Sanders, Tejal Patwardhan, Thibault Sottiaux, Thomas Degry, Thomas Dimson, Tianhao Zheng, Timur Garipov, Tom Stasi, Trapit Bansal, Trevor Creech, Troy Peterson, Tyna Eloundou, Valerie Qi, Vineet Kosaraju, Vinnie Monaco, Vitchyr Pong, Vlad Fomenko, Weiyi Zheng, Wenda Zhou, Wes McCabe, Wojciech Zaremba, Yann Dubois, Yinghai Lu, Yining Chen, Young Cha, Yu Bai, Yuchen He, Yuchen Zhang, Yunyun Wang, Zheng Shao, and Zhuohan Li. Openai o1 system card, 2024. URL https://arxiv.org/abs/2412.16720.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In Sanjoy Dasgupta and David McAllester (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 1310–1318, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL https://proceedings.mlr.press/v28/pascanu13.html.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/d9fc0cdb67638d50f411432d0d41d0ba-Paper.pdf.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

Nicolas Le Roux, Marc G. Bellemare, Jonathan Lebensold, Arnaud Bergeron, Joshua Greaves, Alex Fréchette, Carolyne Pelletier, Eric Thibodeau-Laufer, Sándor Toth, and Sam Work. Tapered off-policy reinforce: Stable and efficient reinforcement learning for llms, 2025. URL https://arxiv.org/abs/2503.14286.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/schulman15.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Amrith Setlur, Nived Rajaraman, Sergey Levine, and Aviral Kumar. Scaling test-time compute without verification or rl is suboptimal, 2025. URL https://arxiv.org/abs/2502.12118.

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

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

Parshin Shojaee, Aneesh Jain, Sindhu Tipirneni, and Chandan K. Reddy. Execution-based code generation using deep reinforcement learning. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=0XBuaxqEcG.

Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32145–32168. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/sokar23a.html.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1f89885d556929e98d3ef9b86448f951-Paper.pdf.

Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'99, pp. 1057–1063, Cambridge, MA, USA, 1999. MIT Press.

Yunhao Tang and Rémi Munos. On a few pitfalls in kl divergence gradient estimation for rl, 2025. URL https://arxiv.org/abs/2506.09477.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, Haiqing Guo, Han Zhu, Hao Ding, Hao Hu, Hao Yang, Hao Zhang, Haotian Yao, Haotian Zhao, Haoyu Lu, Haoze Li, Haozhen Yu, Hongcheng Gao, Huabin Zheng, Huan Yuan, Jia Chen, Jianhang Guo, Jianlin Su, Jianzhou Wang, Jie Zhao, Jin Zhang, Jingyuan Liu, Junjie Yan, Junyan Wu, Lidong Shi, Ling Ye, Longhui Yu, Mengnan Dong, Neo Zhang, Ningchen Ma, Qiwei Pan, Qucheng Gong, Shaowei Liu, Shengling Ma, Shupeng Wei, Sihan Cao, Siying Huang, Tao Jiang, Weihao Gao, Weimin Xiong, Weiran He, Weixiao Huang, Weixin Xu, Wenhao Wu, Wenyang He, Xianghui Wei, Xianqing Jia, Xingzhe Wu, Xinran Xu, Xinxing Zu, Xinyu Zhou, Xuehai Pan, Y. Charles, Yang Li, Yangyang Hu, Yangyang Liu, Yanru Chen, Yejie Wang, Yibo Liu, Yidao Qin, Yifeng Liu, Ying Yang, Yiping Bao, Yulun Du, Yuxin Wu, Yuzhi Wang, Zaida Zhou, Zhaoji Wang, Zhaowei Li, Zhen Zhu, Zheng Zhang, Zhexu Wang, Zhilin Yang, Zhiqi Huang, Zihao Huang, Ziyao Xu, Zonghan Yang, and Zongyu Lin. Kimi k1.5: Scaling reinforcement learning with llms, 2025. URL https://arxiv.org/abs/2501.12599.

Sharan Vaswani, Olivier Bachem, Simone Totaro, Robert Müller, Shivam Garg, Matthieu Geist, Marlos C. Machado, Pablo Samuel Castro, and Nicolas Le Roux. A general class of surrogate functions for stable and efficient reinforcement learning. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera (eds.), *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 8619–8649. PMLR, 28–30 Mar 2022. URL https://proceedings.mlr.press/v151/vaswani22a.html.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2020. URL https://arxiv.org/abs/1910.03771.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu,

Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024. URL https://arxiv.org/abs/2409.12122.

Shihui Yang, Chengfeng Dou, Peidong Guo, Kai Lu, Qiang Ju, Fei Deng, and Rihui Xin. Dcpo: Dynamic clipping policy optimization, 2025a. URL https://arxiv.org/abs/2509.02333.

Zhihe Yang, Xufang Luo, Zilong Wang, Dongqi Han, Zhiyuan He, Dongsheng Li, and Yunjian Xu. Do not let low-probability tokens over-dominate in rl for llms. 2025b. URL https://arxiv.org/abs/2505.12929.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, Yu Fu, Xingtai Lv, Yuchen Zhang, Sihang Zeng, Shang Qu, Haozhan Li, Shijie Wang, Yuru Wang, Xinwei Long, Fangfu Liu, Xiang Xu, Jiaze Ma, Xuekai Zhu, Ermo Hua, Yihao Liu, Zonglin Li, Huayu Chen, Xiaoye Qu, Yafu Li, Weize Chen, Zhenzhao Yuan, Junqi Gao, Dong Li, Zhiyuan Ma, Ganqu Cui, Zhiyuan Liu, Biqing Qi, Ning Ding, and Bowen Zhou. A survey of reinforcement learning for large reasoning models, 2025. URL https://arxiv.org/abs/2509.08827.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019. URL https://arxiv.org/abs/1909.08593.

A Derivation of the Second-Order Optimization Objective

In this section, we formally derive the higher-order expansion of the objective function around a given parameter vector, and present conditions for monotonic improvement. We start by highlighting a smoothness assumption required for our analysis.

Assumption A.1 (Lipschitz continuity of the Hessian)**.** There exists a constant $L_2 \geq 0$ such that, for all $\tau \in [0, 1]$ and all $\Delta\boldsymbol{\theta} \in \mathbb{R}^d$,

$$\left\|\nabla^2 J(\boldsymbol{\theta} + \tau\Delta\boldsymbol{\theta}) - \nabla^2 J(\boldsymbol{\theta})\right\|_{\mathrm{op}} \;\leq\; L_2\,\tau\,\|\Delta\boldsymbol{\theta}\|\,. \tag{14}$$

Assumption A.1 is standard in the analysis of trust-region and cubic-regularized methods, and holds locally for smooth policy parameterizations.

Proposition A.1 (Second-order expansion with integral remainder)**.** *Let $J : \mathbb{R}^d \to \mathbb{R}$ be three times differentiable, and denote $g \triangleq \nabla J(\boldsymbol{\theta})$ and $H \triangleq \nabla^2 J(\boldsymbol{\theta})$. For any update direction $\Delta\boldsymbol{\theta} \in \mathbb{R}^d$, the objective value at the perturbed parameter $\boldsymbol{\theta} + \Delta\boldsymbol{\theta}$ admits the expansion*

$$J(\boldsymbol{\theta}+\Delta\boldsymbol{\theta}) = J(\boldsymbol{\theta})+g^\top\Delta\boldsymbol{\theta}+\tfrac{1}{2}\,\Delta\boldsymbol{\theta}^\top H\,\Delta\boldsymbol{\theta}+\int_0^1 (1-\tau)\,\Delta\boldsymbol{\theta}^\top\big(\nabla^2 J(\boldsymbol{\theta}+\tau\Delta\boldsymbol{\theta})-H\big)\Delta\boldsymbol{\theta}\,d\tau. \tag{15}$$

Under Assumption A.1, the following lower-bound holds

$$J(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) \;\geq\; J(\boldsymbol{\theta}) + g^\top\Delta\boldsymbol{\theta} + \tfrac{1}{2}\,\Delta\boldsymbol{\theta}^\top H\,\Delta\boldsymbol{\theta} - \tfrac{L_2}{6}\,\|\Delta\boldsymbol{\theta}\|^3\,. \tag{16}$$

Proof. Let $\phi(\tau) = J(\boldsymbol{\theta} + \tau\Delta\boldsymbol{\theta})$ for $\tau \in [0, 1]$. Then $\phi'(0) = g^\top\Delta\boldsymbol{\theta}$ and $\phi''(0) = \Delta\boldsymbol{\theta}^\top H\Delta\boldsymbol{\theta}$. The (one-dimensional) Taylor formula with integral remainder gives

$$\phi(1) = \phi(0) + \phi'(0) + \tfrac{1}{2}\phi''(0) + \int_0^1 (1 - \tau)\,(\phi''(\tau) - \phi''(0))\;d\tau. \tag{17}$$

Since $\phi''(\tau) - \phi''(0) = \Delta\boldsymbol{\theta}^\top(\nabla^2 J(\boldsymbol{\theta} + \tau\Delta\boldsymbol{\theta}) - H)\Delta\boldsymbol{\theta}$, we obtain equation 15. Assumption A.1 implies $\left|\Delta\boldsymbol{\theta}^\top(\nabla^2 J(\boldsymbol{\theta} + \tau\Delta\boldsymbol{\theta}) - H)\Delta\boldsymbol{\theta}\right| \leq \|\Delta\boldsymbol{\theta}\|^2 \left\|\nabla^2 J(\boldsymbol{\theta} + \tau\Delta\boldsymbol{\theta}) - H\right\|_{\mathrm{op}} \leq L_2\tau\,\|\Delta\boldsymbol{\theta}\|^3$. Solving the integral gives $\int_0^1 (1 - \tau)\,L_2\tau\,d\tau = L_2/6$. Since this term can be negative, a worst-case bound yields the inequality 16. □

B Derivation of the Policy Divergence Quadratic Approximation

In this section, we formally derive the higher-order expansion of the KL term around a small step $\Delta\boldsymbol{\theta}$. Throughout this derivation, we assume standard regularity assumptions hold (e.g., parameter-independent support, differentiability of $\log \pi_{\boldsymbol{\theta}}$, and dominated convergence so that differentiation may pass under the expectation). The state averaging distribution d_π is fixed.

Assumption B.1 (Lipschitz continuity of the Fisher curvature)**.** Let $F(\boldsymbol{\theta}) := \mathbb{E}_{s\sim d_\pi,\, a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}\Big[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)\, \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)^\top\Big]$.

There exists a constant $L_F \geq 0$ such that, for all $\tau \in [0, 1]$ and all $\Delta\boldsymbol{\theta} \in \mathbb{R}^d$,

$$\big\|F(\boldsymbol{\theta} + \tau\Delta\boldsymbol{\theta}) - F(\boldsymbol{\theta})\big\|_{\mathrm{op}} \;\leq\; L_F\, \tau\, \|\Delta\boldsymbol{\theta}\|. \tag{18}$$

Assumption B.1 is analogous to the Assumption A.1 applied to the Fisher geometry.

Lemma B.1 (The grad-log-prob identity)**.** *Under regularity assumptions, the following identity holds:*

$$\mathbb{E}_{s\sim d_\pi,\, a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}\big[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)\big] = 0. \tag{19}$$

Proof. Fix s. By normalization, $\sum_a \pi_{\boldsymbol{\theta}}(a \mid s) = 1$. Differentiating, $\sum_a \nabla_{\boldsymbol{\theta}}\pi_{\boldsymbol{\theta}}(a \mid s) = 0$. Since $\nabla_{\boldsymbol{\theta}}\pi_{\boldsymbol{\theta}} = \pi_{\boldsymbol{\theta}}\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}$, we obtain

$$\sum_a \pi_{\boldsymbol{\theta}}(a \mid s)\, \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s) = 0, \tag{20}$$

i.e. $\mathbb{E}_{a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)] = 0$. Averaging over $s \sim d_\pi$ preserves zero. □

Lemma B.2 (Fisher identity)**.** *Under regularity assumptions, the following identity holds:*

$$-\mathbb{E}\big[\nabla^2_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)\big] = \mathbb{E}\big[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)\, \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)^\top\big] =: F(\boldsymbol{\theta}). \tag{21}$$

Proof. Fix s. Twice differentiating normalization gives $\nabla^2_{\boldsymbol{\theta}} \sum_a \pi_{\boldsymbol{\theta}}(a \mid s) = \sum_a \nabla^2_{\boldsymbol{\theta}}\pi_{\boldsymbol{\theta}}(a \mid s) = 0$. Using $\nabla^2_{\boldsymbol{\theta}}\pi_{\boldsymbol{\theta}} = \pi_{\boldsymbol{\theta}}\big(\nabla^2_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}} + \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}\, \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}^\top\big)$, we obtain

$$0 = \sum_a \pi_{\boldsymbol{\theta}}(a \mid s)\, \nabla^2_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s) + \sum_a \pi_{\boldsymbol{\theta}}(a \mid s)\, \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)\, \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)^\top. \tag{22}$$

Recognizing expectations over $a \sim \pi_{\boldsymbol{\theta}}(\cdot \mid s)$ and multiplying by -1 yields

$$-\mathbb{E}_{a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}[\nabla^2_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)] = \mathbb{E}_{a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)\, \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)^\top]. \tag{23}$$

Averaging over $s \sim d_\pi$ gives the result. □

Proposition B.1 (Second-order expansion with integral remainder)**.** *Define the average forward KL as*

$$\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}} \,\|\, \pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}) := \mathbb{E}_{s\sim d_\pi}\big[\mathrm{KL}\big(\pi_{\boldsymbol{\theta}}(\cdot \mid s) \,\|\, \pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}(\cdot \mid s)\big)\big]. \tag{24}$$

Then, for any update $\Delta\boldsymbol{\theta}$,

$$\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}} \,\|\, \pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}) = \tfrac{1}{2}\Delta\boldsymbol{\theta}^\top F(\boldsymbol{\theta})\Delta\boldsymbol{\theta} + \int_0^1 (1-\tau)\, \Delta\boldsymbol{\theta}^\top\big(F(\boldsymbol{\theta} + \tau\Delta\boldsymbol{\theta}) - F(\boldsymbol{\theta})\big)\Delta\boldsymbol{\theta}\, d\tau. \tag{25}$$

And, under Assumption B.1, the following holds:

$$\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}} \,\|\, \pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}) = \tfrac{1}{2}\Delta\boldsymbol{\theta}^\top F(\boldsymbol{\theta})\Delta\boldsymbol{\theta} + \mathcal{O}(\|\Delta\boldsymbol{\theta}\|^3). \tag{26}$$

Proof. Let $\phi(\tau) := \bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}} \,\|\, \pi_{\boldsymbol{\theta}+\tau\Delta\boldsymbol{\theta}})$. By the Taylor expansion with integral remainder,

$$\phi(1) = \phi(0) + \phi'(0) + \tfrac{1}{2}\phi''(0) + \int_0^1 (1-\tau)\big(\phi''(\tau) - \phi''(0)\big)\, d\tau. \tag{27}$$

Then $\phi(0) = 0$, and $\phi'(\tau) = -\mathbb{E}[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}+\tau\Delta\boldsymbol{\theta}}(a \mid s)]^\top \Delta\boldsymbol{\theta}$, so by Lemma B.1, $\phi'(0) = 0$. Differentiating again and applying Lemma B.2,

$$\phi''(\tau) = \Delta\boldsymbol{\theta}^\top F(\boldsymbol{\theta} + \tau\Delta\boldsymbol{\theta})\Delta\boldsymbol{\theta}, \qquad \phi''(0) = \Delta\boldsymbol{\theta}^\top F(\boldsymbol{\theta})\Delta\boldsymbol{\theta}. \tag{28}$$

Substituting the evaluated terms yields the expansion.

Finally, Assumption B.1 implies

$$\left|\Delta\boldsymbol{\theta}^\top \big(F(\boldsymbol{\theta} + \tau\Delta\boldsymbol{\theta}) - F(\boldsymbol{\theta})\big)\Delta\boldsymbol{\theta}\right| \le L_F \tau \|\Delta\boldsymbol{\theta}\|^3. \tag{29}$$

Integrating $\int_0^1 (1-\tau)\tau \, d\tau = 1/6$, so the remainder term is $\mathcal{O}(\|\Delta\boldsymbol{\theta}\|^3)$. □

C Derivation of gradients and curvatures under last-layer model

In this section, we formally derive the gradient and curvature expressions assuming the last-layer model.

Proposition C.1 (Gradient w.r.t.last-layer model of a softmax policy)**.** *Let us consider a softmax policy* $\pi_{\boldsymbol{\theta}}(a \mid s) = \frac{\exp\left(f_{\boldsymbol{\theta}}(s,a)\right)}{\sum_{a'} \exp\left(f_{\boldsymbol{\theta}}(s,a')\right)}$*. Let us also denote the pre-softmax layer by* $f_{\boldsymbol{\theta}}(s_t) = Wh_{\bar{\boldsymbol{\theta}}}(s_t), W \in \mathbb{R}^{K \times d_i}$, $h_{\bar{\boldsymbol{\theta}}}(s_t) \in \mathbb{R}^{d_i}$*. Define* $\boldsymbol{\psi} \coloneqq \mathrm{vec}(W) \in \mathbb{R}^{Kd}$*, with* $\boldsymbol{\theta} = (\bar{\boldsymbol{\theta}}, \boldsymbol{\psi})$*,* $K = dim(\mathcal{V})$*. Then the policy gradient with respect to* $\boldsymbol{\psi}$ *of the PG objective:*

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\left[\sum_{t=0}^{T} \gamma^t A(s_t, a_t) \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)\right] \tag{30}$$

is given by:

$$\tilde{g}(\boldsymbol{\psi}) = \mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T} \gamma^t \, A(s_t, a_t) \left(e_a - \pi_{\boldsymbol{\theta}}(s_t)\right) \otimes h(s_t)\right], \tag{31}$$

where $e_a \in \mathbb{R}^K$*,* $K = dim(\mathcal{V})$*, denotes the one-hot vector of the realized action* a_t *at time* t *(i.e.,* $e_a = e_{a_t}$*),* $\pi_{\boldsymbol{\theta}}(s_t) \in \mathbb{R}^K$ *is the vector of action probabilities at* s_t*, and* $\otimes$ *denotes the Kronecker product.*

Proof. Starting from the advantage version of Equation 1, the policy gradient with respect to $\boldsymbol{\psi}$ is given by

$$\tilde{g}(\boldsymbol{\psi}) = \mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T} \gamma^t \, A(s_t, a_t) \, \nabla_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)\right]. \tag{32}$$

With logits $f(s_t) = Wh_{\bar{\boldsymbol{\theta}}}(s_t)$, the Jacobian of the log-softmax with respect to $f(s_t)$ is:

$$\frac{\partial \log \pi_{\boldsymbol{\theta}}(a \mid s)}{\partial f(s_t)} = e_a - \pi_{\boldsymbol{\theta}}(s_t) \; \in \; \mathbb{R}^K. \tag{33}$$

Vectorizing W gives:

$$\frac{\partial f(s_t)}{\partial \boldsymbol{\psi}} = I_K \otimes h_{\bar{\boldsymbol{\theta}}}(s_t)^\top \; \in \; \mathbb{R}^{K \times Kd}. \tag{34}$$

By the chain rule,

$$\nabla_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a \mid s) = \Big(e_a - \pi_{\boldsymbol{\theta}}(s_t)\Big)^\top \big(I_K \otimes h_{\bar{\boldsymbol{\theta}}}(s_t)^\top\big) = \big(e_a - \pi_{\boldsymbol{\theta}}(s_t)\big) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t),$$

where we used standard Kronecker product identities to obtain a vector in $\mathbb{R}^{Kd}$. Plugging the expression for $\nabla_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)$ into Equation 32 yields

$$\tilde{g}(\boldsymbol{\psi}) = \mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T} \gamma^t \, A(s_t, a_t) \left(e_a - \pi_{\boldsymbol{\theta}}(s_t)\right) \otimes h(s_t)\right]. \tag{35}$$

□

The Hessian of the Objective. For the Hessian, we start by extending the PG Theorem for Hessians:

Lemma C.1 (Hessian of the Policy Gradient)**.** *Let* $\pi_{\boldsymbol{\theta}}(a \mid s)$ *be a differentiable stochastic policy and consider the discounted policy gradient objective*

$$J(\boldsymbol{\theta}) \; = \; \mathbb{E}_{\tau \sim \pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T} \gamma^t A(s_t, a_t) \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)\right], \tag{36}$$

where $A(s_t, a_t)$ *is the advantage function at time* t*. Then, the Hessian of* $J(\boldsymbol{\theta})$ *is given by*

$$\nabla^2_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \;=\; \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T}\gamma^t A(s_t,a_t)\left(\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a_t\mid s_t)\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a_t\mid s_t)^\top+\nabla^2_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a_t\mid s_t)\right)\right]. \tag{37}$$

Proof. Taking the first derivative of $J(\boldsymbol{\theta})$, we obtain

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \;=\; \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T}\gamma^t A(s_t,a_t)\,\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a_t\mid s_t)\right]. \tag{38}$$

Differentiating once more yields

$$\nabla^2_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}}\,\mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T}\gamma^t A(s_t,a_t)\,\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a_t\mid s_t)\right]. \tag{39}$$

Expanding the expectation explicitly over state–action pairs weighted by the discounted state distribution $d^\pi_\gamma(s_t)$ gives

$$\nabla^2_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_s d^\pi_\gamma(s_t)\sum_a \nabla_{\boldsymbol{\theta}}\Big[\pi_{\boldsymbol{\theta}}(a\mid s)\,A(s,a)\,\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a\mid s)\Big]. \tag{40}$$

Applying the product rule, we obtain

$$\nabla^2_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_s d^\pi_\gamma(s_t)\sum_a \pi_{\boldsymbol{\theta}}(a\mid s)A(s,a)\Big(\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a\mid s)\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a\mid s)^\top+\nabla^2_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a\mid s)\Big). \tag{41}$$

Rewriting in expectation form gives the final result:

$$\nabla^2_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \;=\; \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T}\gamma^t A(s_t,a_t)\left(\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a_t\mid s_t)\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a_t\mid s_t)^\top+\nabla^2_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a_t\mid s_t)\right)\right]. \tag{42}$$

□

Now, we can state the Hessian form under the last-layer model:

Proposition C.2 (Hessian under Last-Layer Model)**.** *Let us consider a softmax policy* $\pi_{\boldsymbol{\theta}}(a\mid s) = \frac{\exp\left(f_{\boldsymbol{\theta}}(s,a)\right)}{\sum_{a'}\exp\left(f_{\boldsymbol{\theta}}(s,a')\right)}$*. Let us also denote the pre-softmax layer by* $f(s_t) = Wh_{\bar{\boldsymbol{\theta}}}(s_t), W \in \mathbb{R}^{K\times d},\ h_{\bar{\boldsymbol{\theta}}}(s_t)\in\mathbb{R}^d$*. Define* $\boldsymbol{\psi} \coloneqq \mathrm{vec}(W)\in\mathbb{R}^{Kd}$*, with* $\boldsymbol{\theta} = (\bar{\boldsymbol{\theta}},\boldsymbol{\psi})$*,* $K = dim(\mathcal{V})$*. Then, the Hessian of the discounted policy gradient objective*

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T}\gamma^t A(s_t,a_t)\log\pi_{\boldsymbol{\theta}}(a_t\mid s_t)\right] \tag{43}$$

is given by

$$\tilde{H}(\boldsymbol{\psi}) = \nabla^2_{\boldsymbol{\psi}} J(\theta) = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T}\gamma^t A(s,a)\Big((e_a-\pi_{\boldsymbol{\theta}}(s_t))(e_a-\pi_{\boldsymbol{\theta}}(s_t))^\top - F(s_t)\Big)\otimes h_{\bar{\boldsymbol{\theta}}}(s_t)h_{\bar{\boldsymbol{\theta}}}(s_t)^\top\right], \tag{44}$$

where $e_a\in\mathbb{R}^K$ *is the one-hot vector of action* a*,* $\pi_{\boldsymbol{\theta}}(s_t)\in\mathbb{R}^K$ *is the vector of action probabilities, and* $F(s_t) := \mathrm{diag}(\pi_{\boldsymbol{\theta}}(s_t)) - \pi_{\boldsymbol{\theta}}(s_t)\pi_{\boldsymbol{\theta}}(s_t)^\top$ *is the Fisher information matrix at state* s_t*.*

Proof. From Proposition C.1,

$$\nabla_{\boldsymbol{\psi}}\log\pi_{\boldsymbol{\theta}}(a_t\mid s_t) = (e_a-\pi_{\boldsymbol{\theta}}(s_t))\otimes h_{\bar{\boldsymbol{\theta}}}(s_t). \tag{45}$$

Hence, the outer product is

$$\begin{aligned}\nabla_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)\, \nabla_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)^\top = \quad (46)\\ = \big((e_a - \pi_{\boldsymbol{\theta}}(s_t)) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t)\big)\big((e_a - \pi_{\boldsymbol{\theta}}(s_t)) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t)\big)^\top \\ = (e_a - \pi_{\boldsymbol{\theta}}(s_t))(e_a - \pi_{\boldsymbol{\theta}}(s_t))^\top \otimes h_{\bar{\boldsymbol{\theta}}}(s_t) h_{\bar{\boldsymbol{\theta}}}(s_t)^\top,\end{aligned}$$

where we applied the identity $(u \otimes v)(u \otimes v)^\top = (uu^\top) \otimes (vv^\top)$. Next, we compute the second derivative. Since $\nabla_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t) = (e_a - \pi_{\boldsymbol{\theta}}(s_t)) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t)$, it follows that

$$\nabla^2_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t) = -\nabla_{\boldsymbol{\psi}} \pi_{\boldsymbol{\theta}}(s_t) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t). \quad (47)$$

Using $\nabla \pi_{\boldsymbol{\theta}}(s_t) = \big(\mathrm{diag}(\pi_{\boldsymbol{\theta}}(s_t)) - \pi_{\boldsymbol{\theta}}(s_t)\pi_{\boldsymbol{\theta}}(s_t)^\top\big) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t)$, we obtain

$$\begin{aligned}\nabla^2_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t) = \quad (48)\\ = -\big(\mathrm{diag}(\pi_{\boldsymbol{\theta}}(s_t)) - \pi_{\boldsymbol{\theta}}(s_t)\pi_{\boldsymbol{\theta}}(s_t)^\top\big) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t) h_{\bar{\boldsymbol{\theta}}}(s_t)^\top \\ = -F(s_t) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t) h_{\bar{\boldsymbol{\theta}}}(s_t)^\top. \quad (49)\end{aligned}$$

Finally, substituting both terms into the general Hessian expression from Lemma C.1,

$$\nabla^2_{\boldsymbol{\psi}} J(\boldsymbol{\psi}) = \mathbb{E}_{s,a\sim\pi_{\boldsymbol{\psi}}}\big[A(s,a)\,\big(\nabla_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)\, \nabla_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)^\top + \nabla^2_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)\big)\big],$$

yields:

$$\tilde{H}(\boldsymbol{\psi}) = \nabla^2_{\boldsymbol{\psi}} J(\theta) = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T} \gamma^t A(s,a)\Big((e_a - \pi_{\boldsymbol{\theta}}(s_t))(e_a - \pi_{\boldsymbol{\theta}}(s_t))^\top - F(s_t)\Big) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t) h_{\bar{\boldsymbol{\theta}}}(s_t)^\top\right], \quad (50)$$

□

Proposition C.3 (Fisher Information under the Last-Layer Model)**.** *Let us consider a softmax policy* $\pi_{\boldsymbol{\theta}}(a \mid s) = \frac{\exp\big(f_{\boldsymbol{\theta}}(s,a)\big)}{\sum_{a'} \exp\big(f_{\boldsymbol{\theta}}(s,a')\big)}$*. Let us also denote the pre-softmax layer by* $f(s_t) = W h_{\bar{\boldsymbol{\theta}}}(s_t)$*,* $W \in \mathbb{R}^{K\times d}$*,*$h_{\bar{\boldsymbol{\theta}}}(s_t) \in \mathbb{R}^d$*. Define* $\boldsymbol{\psi} \coloneqq \mathrm{vec}(W) \in \mathbb{R}^{Kd}$*, with* $\boldsymbol{\theta} = (\bar{\boldsymbol{\theta}}, \boldsymbol{\psi})$*,* $K = dim(\mathcal{V})$*. Then, the Fisher information matrix with respect to* $\boldsymbol{\psi}$ *is*

$$\tilde{F}(\boldsymbol{\psi}) = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\big[\big((e_{a_t} - \pi_{\boldsymbol{\theta}}(s_t))(e_{a_t} - \pi_{\boldsymbol{\theta}}(s_t))^\top\big) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t) h_{\bar{\boldsymbol{\theta}}}(s_t)^\top\big], \quad (51)$$

where $e_{a_t} \in \mathbb{R}^K$ *is the one-hot vector of the realized action* a_t*, and* $\pi_{\boldsymbol{\theta}}(s_t) \in \mathbb{R}^K$ *is the vector of action probabilities at state* s_t*.*

Proof. From Proposition C.1,

$$\nabla_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t) = (e_{a_t} - \pi_{\boldsymbol{\theta}}(s_t)) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t). \quad (52)$$

Therefore,

$$\nabla_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)\, \nabla_{\boldsymbol{\psi}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)^\top = \big((e_{a_t} - \pi_{\boldsymbol{\theta}}(s_t))(e_{a_t} - \pi_{\boldsymbol{\theta}}(s_t))^\top\big) \otimes h_{\bar{\boldsymbol{\theta}}}(s_t) h_{\bar{\boldsymbol{\theta}}}(s_t)^\top. \quad (53)$$

where the last step follows from the Kronecker identity $(u \otimes x)(v \otimes x)^\top = (uv^\top) \otimes (xx^\top)$. Substituting this into the definition of the discounted Fisher information matrix yields the result. □

D DIRECTIONAL CURVATURES COMPUTATION

In this section, we present our mechanisms to compute Hessian and Fisher directional curvatures.

D.1 DIRECTIONAL FISHER CURVATURE

For the last-layer parameters $\boldsymbol{\psi} = \mathrm{vec}(W)$ with $W \in \mathbb{R}^{K\times d_i}$, $K = \dim(\mathcal{V})$, denote by $U := \mathrm{unvec}(\Delta\boldsymbol{\psi}) \in \mathbb{R}^{K\times d_i}$ the corresponding matrix form of the direction. We aim to compute the curvature of the Fisher information matrix along a direction $\Delta\boldsymbol{\psi}$ in parameter space. Recall the Fisher information matrix under the Last-Layer Model (Equation 9):

$$\tilde{F}(\boldsymbol{\psi}) = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\left(u_t u_t^\top\right) \otimes \left(h_t h_t^\top\right)\right], \tag{54}$$

where $u_t := e_{a_t} - \pi_{\boldsymbol{\theta}}(s_t) \in \mathbb{R}^K$ is the policy error vector and $h_t := h_{\bar{\boldsymbol{\theta}}}(s_t) \in \mathbb{R}^{d_i}$ is the feature vector. Using the Kronecker Vector identity $\mathrm{vec}(X)^\top(A\otimes B)\,\mathrm{vec}(X) = \mathrm{Tr}(AXBX^\top)$:

$$\Delta\boldsymbol{\psi}^\top\tilde{F}(\boldsymbol{\psi})\Delta\boldsymbol{\psi} = \mathbb{E}_\tau\left[\mathrm{vec}(U)^\top\left(u_t u_t^\top \otimes h_t h_t^\top\right)\mathrm{vec}(U)\right] \tag{55}$$

$$= \mathbb{E}_\tau\left[\mathrm{Tr}\left(u_t u_t^\top\, U\, h_t h_t^\top U^\top\right)\right]. \tag{56}$$

Let $v_t := Uh_t \in \mathbb{R}^K$. Then $\mathrm{Tr}(u_t u_t^\top v_t v_t^\top) = (u_t^\top v_t)^2$. And we obtain:

$$\Delta\boldsymbol{\psi}^\top\tilde{F}(\boldsymbol{\psi})\Delta\boldsymbol{\psi} = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[(u_t^\top v_t)^2\right]. \tag{57}$$

We can estimate the Equation above with samples. Given a batch of N state–action–time samples $\{(s_i, a_i, t_i)\}_{i=1}^N$, an estimator of the curvature is:

$$\widehat{\Delta\boldsymbol{\psi}^\top\tilde{F}\Delta\boldsymbol{\psi}} = \frac{1}{N}\sum_{i=1}^{N}\left(u_i^\top(\widehat{U}h_i)\right)^2, \tag{58}$$

with $u_i = e_{a_i} - \pi_{\boldsymbol{\theta}}(s_i)$ and $h_i = h_{\bar{\boldsymbol{\theta}}}(s_i)$. In practice, $\Delta\boldsymbol{\psi}$ itself is typically estimated from data (e.g., as a stochastic gradient direction), hence not strictly deterministic. Therefore, estimating Equation 58 introduces a mild bias as u_t and h_t are statistically dependent.

Cost Analysis. The computation requires only vector and matrix–vector operations. Per sample, we compute Uh_i at cost $\mathcal{O}(Kd)$ and the dot product $u_i^\top v_i$ at cost $\mathcal{O}(K)$, followed by a scalar square. In memory, we only store U (Kd parameters) and the per-sample vectors u_i and h_i. This is dramatically cheaper than materializing the full Fisher matrix $\tilde{F} \in \mathbb{R}^{Kd\times Kd}$, which would require $(Kd)^2$ entries.

D.2 DIRECTIONAL HESSIAN CURVATURE

We now consider the curvature of the Hessian along a direction $\Delta\boldsymbol{\psi}$. We also assume the same notation as in subsection D.1 Recall the Hessian under the Last-Layer model (Equation 8):

$$\tilde{H}(\boldsymbol{\psi}) = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[A(s,a)\left(u_t u_t^\top - F(s)\right) \otimes h_{\bar{\boldsymbol{\theta}}}(s) h_t h_t^\top\right], \tag{59}$$

where $F(s) = \mathrm{diag}(\pi_{\boldsymbol{\theta}}(s)) - \pi_{\boldsymbol{\theta}}(s)\pi_{\boldsymbol{\theta}}(s)^\top$ is the Fisher matrix at state s, $u_t := e_{a_t} - \pi_{\boldsymbol{\theta}}(s_t) \in \mathbb{R}^K$ is the policy error vector and $h_t := h_{\bar{\boldsymbol{\theta}}}(s_t) \in \mathbb{R}^{d_i}$ is the feature vector.

The directional curvature along $\Delta\boldsymbol{\psi}$ is

$$\Delta\boldsymbol{\psi}^\top\tilde{H}(\boldsymbol{\psi})\Delta\boldsymbol{\psi} = \mathbb{E}_\tau\left[\sum_{t=0}^{T}\gamma^t\, A(s_t,a_t)\,\mathrm{vec}(U)^\top\left(\left(u_t u_t^\top - F(s_t)\right)\otimes h_t h_t^\top\right)\mathrm{vec}(U)\right]. \tag{60}$$

Using the Kronecker–Vector identity $\mathrm{vec}(X)^\top (A \otimes B)\,\mathrm{vec}(X) = \mathrm{Tr}(AXBX^\top)$, we obtain:

$$\Delta_{\boldsymbol{\psi}}^\top \tilde{H}(\boldsymbol{\psi})\Delta_{\boldsymbol{\psi}} = \mathbb{E}_\tau\left[\sum_{t=0}^{T} \gamma^t\, A(s_t,a_t)\left(\mathrm{Tr}(u_t u_t^\top\, U h_t h_t^\top U^\top) - \mathrm{Tr}(F(s_t)\, U h_t h_t^\top U^\top)\right)\right]. \quad (61)$$

Let $v_t := U h_t$. Then the two traces simplify via

$$\mathrm{Tr}(u_t u_t^\top v_t v_t^\top) = (u_t^\top v_t)^2, \qquad \mathrm{Tr}(F(s_t)\, v_t v_t^\top) = v_t^\top F(s_t)\, v_t,$$

where the first equality uses $uu^\top vv^\top = (u^\top v)\, uv^\top$ and $\mathrm{Tr}(ab^\top) = b^\top a$, and the second uses $\mathrm{Tr}(A\, xx^\top) = x^\top Ax$. Hence,

$$\Delta_{\boldsymbol{\psi}}^\top \tilde{H}(\boldsymbol{\psi})\Delta_{\boldsymbol{\psi}} = \mathbb{E}_{\tau\sim\pi_{\boldsymbol{\theta}}}\left[\sum_{t=0}^{T} \gamma^t\, A(s_t,a_t)\left((u_t^\top v_t)^2 \;-\; v_t^\top F(s_t)\, v_t\right)\right]. \quad (62)$$

We can estimate the Equation above via samples, noting the same remarks as in subsection D.1. The sample-based estimator is

$$\widehat{\Delta\boldsymbol{\psi}^\top \tilde{H} \Delta\boldsymbol{\psi}} = \frac{1}{N}\sum_{i=1}^{N} \gamma^{t_i} A(s_i,a_i)\left((u_i^\top \widehat{v_i})^2 - \widehat{v_i}^{\,\top} F(s_i)\, \widehat{v_i}\right), \quad u_i = e_{a_i} - \pi_{\boldsymbol{\theta}}(s_i), \;\; \widehat{v_i} = \widehat{U} h_{\widehat{\boldsymbol{\theta}}}(s_i). \quad (63)$$

Cost Analysis. The computation again only involves vectors and matrix–vector operations. Per sample, we compute $v_t = U h_t$ at cost $\mathcal{O}(Kd)$, then $(u_t^\top v_t)^2$ at cost $O(K)$. The second term requires an analogous computation to the Fisher case in subsection D.1. Hence, the complexity remains $O(Kd)$ per sample, and the memory cost is linear in K and d, avoiding materialization of the full Hessian $\tilde{H} \in \mathbb{R}^{Kd\times Kd}$.

E MONOTONIC POLICY IMPROVEMENT UNDER CAPO

In this section, we formalize the conditions of monotonic improvement under CAPO.

Assumption E.1 (Bounded curvature and step norms)**.** Let $\pi_{\boldsymbol{\theta}}$ be a differentiable policy with objective $J(\boldsymbol{\theta})$. Write $g(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$, $H(\boldsymbol{\theta}) = \nabla^2_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$, and $F(\boldsymbol{\theta}) = \mathbb{E}_{s\sim d_\pi,\, a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}\big[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)\, \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)^\top\big]$. For $\Delta\boldsymbol{\theta} \in \mathbb{R}^d$ define the quadratic diagnostics

$$m_H(\Delta\boldsymbol{\theta}) \;:=\; g(\boldsymbol{\theta})^\top \Delta\boldsymbol{\theta} + \tfrac{1}{2}\,\Delta\boldsymbol{\theta}^\top H(\boldsymbol{\theta}) \Delta\boldsymbol{\theta}, \qquad m_F(\Delta\boldsymbol{\theta}) \;:=\; \tfrac{1}{2}\,\Delta\boldsymbol{\theta}^\top F(\boldsymbol{\theta}) \Delta\boldsymbol{\theta}. \tag{64}$$

Assume:

(i) (**Hessian operator norm bound**) $\|H(\boldsymbol{\theta})\|_{\mathrm{op}} \le M$ for some finite $M > 0$, where $\|H(\boldsymbol{\theta})\|_{\mathrm{op}} := \sup_{x\neq 0} \frac{\|H(\boldsymbol{\theta})x\|}{\|x\|}$.

(ii) (**Per-candidate step bound**) Every candidate update considered by the algorithm satisfies $\|\Delta\boldsymbol{\theta}\| \le r$ for some $r > 0$.

Remarks. The step norm bound is standard in practice, since learning rates, clipping, or trust-region constraints ensure $\|\Delta\boldsymbol{\theta}\| \le r$. The Hessian bound $\|H(\boldsymbol{\theta})\|_{\mathrm{op}} \le M$ is more restrictive globally, but over any compact region of parameter space visited by the algorithm, continuity of $H(\boldsymbol{\theta})$ implies a finite M.

Lemma E.1 (Surrogate–true performance gap)**.** *For any policies π and π', with $D_{\mathrm{KL}}(\pi\|\pi')$ the average forward KL under d_π,*

$$J(\pi') \;\ge\; L_\pi(\pi') \;-\; C\,\sqrt{D_{\mathrm{KL}}(\pi\|\pi')}, \qquad C = \frac{2\gamma}{(1-\gamma)^2}\,\epsilon\,\sqrt{2}, \tag{65}$$

where $|A^\pi(s,a)| \le \epsilon$ with ϵ finite, and $L_\pi(\pi') := J(\pi) + \mathbb{E}_{s\sim d_\pi,\, a\sim\pi'(\cdot|s)}[A_\pi(s,a)]$. Moreover, writing $\pi = \pi_{\boldsymbol{\theta}}$ and $\pi' = \pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}$ for a parameter step $\Delta\boldsymbol{\theta}$,

$$L_{\pi_{\boldsymbol{\theta}}}(\pi_{\boldsymbol{\theta}'}) - J(\pi_{\boldsymbol{\theta}}) \;=\; g(\boldsymbol{\theta})^\top \Delta\boldsymbol{\theta} \;+\; \tfrac{1}{2}\,\Delta\boldsymbol{\theta}^\top H(\boldsymbol{\theta})\,\Delta\boldsymbol{\theta} \;+\; o(\|\Delta\boldsymbol{\theta}\|^2). \tag{66}$$

Proof. The proof of equation 65 is in Achiam et al. (2017). For Equation 66, we define $\Psi(\boldsymbol{\theta}') := L_{\pi_{\boldsymbol{\theta}}}(\pi_{\boldsymbol{\theta}'})$. Note that $\Psi(\boldsymbol{\theta}) = J(\pi_{\boldsymbol{\theta}})$. Now compute the gradient of Ψ at $\boldsymbol{\theta}' = \boldsymbol{\theta}$:

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}'}\Psi(\boldsymbol{\theta}')\big|_{\boldsymbol{\theta}'=\boldsymbol{\theta}} &= \nabla_{\boldsymbol{\theta}'}\mathbb{E}_{s\sim d_\pi,\, a\sim\pi_{\boldsymbol{\theta}'}(\cdot|s)}[A_\pi(s,a)]\Big|_{\boldsymbol{\theta}'=\boldsymbol{\theta}} \\
&= \mathbb{E}_{s\sim d_\pi,\, a\sim\pi_{\boldsymbol{\theta}}}[A_\pi(s,a)\,\nabla_{\boldsymbol{\theta}'}\log\pi_{\boldsymbol{\theta}'}(a|s)]_{\boldsymbol{\theta}'=\boldsymbol{\theta}} \\
&= \mathbb{E}_{s\sim d_\pi,\, a\sim\pi}[A_\pi(s,a)\,\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a|s)] =: g(\boldsymbol{\theta}),
\end{aligned} \tag{67}$$

where $g(\boldsymbol{\theta})$ is precisely the policy gradient. Differentiate once more:

$$\begin{aligned}
\nabla^2_{\boldsymbol{\theta}'}\Psi(\boldsymbol{\theta}')\big|_{\boldsymbol{\theta}'=\boldsymbol{\theta}} &= \mathbb{E}_{s\sim d_{\pi_{\boldsymbol{\theta}}},\, a\sim\pi_{\boldsymbol{\theta}'}(\cdot|s)}\Big[A_{\pi_{\boldsymbol{\theta}}}(s,a)\,\nabla^2_{\boldsymbol{\theta}'}\log\pi_{\boldsymbol{\theta}'}(a\mid s)\Big]_{\boldsymbol{\theta}'=\boldsymbol{\theta}} \\
&\quad + \mathbb{E}_{s\sim d_{\pi_{\boldsymbol{\theta}}},\, a\sim\pi_{\boldsymbol{\theta}'}(\cdot|s)}\Big[A_{\pi_{\boldsymbol{\theta}}}(s,a)\,\nabla_{\boldsymbol{\theta}'}\log\pi_{\boldsymbol{\theta}'}(a\mid s)\,\nabla_{\boldsymbol{\theta}'}\log\pi_{\boldsymbol{\theta}'}(a\mid s)^\top\Big]_{\boldsymbol{\theta}'=\boldsymbol{\theta}}. \\
&:= H(\boldsymbol{\theta}).
\end{aligned}$$

By the second-order Taylor expansion,

$$\Psi(\boldsymbol{\theta}+\Delta\boldsymbol{\theta}) = \Psi(\boldsymbol{\theta}) \;+\; g(\boldsymbol{\theta})^\top\Delta\boldsymbol{\theta} \;+\; \tfrac{1}{2}\,\Delta\boldsymbol{\theta}^\top H(\boldsymbol{\theta})\,\Delta\boldsymbol{\theta} \;+\; o(\|\Delta\boldsymbol{\theta}\|^2), \tag{68}$$

which is exactly equation 66. □

Theorem E.1 (Monotonic improvement under CAPO, restated)**.** *Fix thresholds $\delta_H > 0$ and $\delta_F > 0$. Let $\mathcal{B}$ be a batch of sampled trajectories. Split $\mathcal{B}$ into disjoint N subsets $b_i \subset \mathcal{B}$, and propose candidate subset updates $\{\Delta\theta_i\}_{i:N}$. Retain those satisfying:*

$$m_H(\Delta\theta_i) \;\ge\; \delta_H = \omega + \tfrac{1}{2}Mr^2, \qquad m_F(\Delta\theta_i) \;\le\; \delta_F, \tag{69}$$

with $\omega > 0$ and M, r defined as in Assumption E.1. Let $\mathcal{B}_{acc}$ denote the superset of the B accepted subsets, and define the aggregated update:$\Delta\theta = \frac{1}{B}\sum_{i\in\mathcal{B}_{acc}}\Delta\theta_i$. Then, for two policies $\pi_{\boldsymbol{\theta}}$ and $\pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}$, we obtain:

$$J(\pi_{\theta+\Delta\theta}) - J(\pi_\theta) \;\ge\; \omega \;-\; C\,\sqrt{\delta_F}. \tag{70}$$

Thus choosing $\omega \ge C\sqrt{\delta_F}$ guarantees monotonic improvement: $J(\pi_{\theta+\Delta\theta}) \ge J(\pi_\theta)$.

Proof. We first establish bounds in the global Fisher and Hessian directional curvatures.

Fisher global bound. Since $F \succeq 0$, the quadratic form $\phi(u) := u^\top F u$ is convex. Thus:

$$\Delta\theta^\top F \Delta\theta = \Big(\frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} \Delta\theta_i\Big)^\top F\Big(\frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} \Delta\theta_i\Big) \;\le\; \frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} \Delta\theta_i^\top F \Delta\theta_i. \tag{71}$$

The inequality above follows from:

$$\frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} \Delta\theta_i^\top F \Delta\theta_i \;-\; \Big(\frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} \Delta\theta_i\Big)^\top F\Big(\frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} \Delta\theta_i\Big) \tag{72}$$

$$= \frac{1}{2B^2}\sum_{i,j\in\mathcal{B}_{acc}} (\Delta\theta_i - \Delta\theta_j)^\top F(\Delta\theta_i - \Delta\theta_j) \ge 0, \tag{73}$$

because $F \succeq 0$ implies each summand is nonnegative. Hence:

$$\Delta\theta^\top F \Delta\theta \;\le\; \frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} \Delta\theta_i^\top F \Delta\theta_i \;\le\; \frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} 2\, m_F(\Delta\theta_i) \;\le\; 2\,\delta_F. \tag{74}$$

Hessian global bound. Expanding $m_H(\Delta\boldsymbol{\theta})$:

$$\begin{aligned} m_H(\Delta\boldsymbol{\theta}) &= g(\boldsymbol{\theta})^\top \Delta\boldsymbol{\theta} + \tfrac{1}{2}\,\Delta\boldsymbol{\theta}^\top H \Delta\boldsymbol{\theta} \\ &= g(\boldsymbol{\theta})^\top\Big(\tfrac{1}{B}\sum_{i\in\mathcal{B}_{acc}} \Delta\boldsymbol{\theta}_i\Big) + \frac{1}{2}\Big(\tfrac{1}{B}\sum_{i\in\mathcal{B}_{acc}} \Delta\boldsymbol{\theta}_i\Big)^\top H\Big(\tfrac{1}{B}\sum_{j\in\mathcal{B}_{acc}} \Delta\boldsymbol{\theta}_j\Big) \\ &= \frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} g(\boldsymbol{\theta})^\top \Delta\boldsymbol{\theta}_i + \frac{1}{2B^2}\sum_{i,j\in\mathcal{B}_{acc}} \Delta\boldsymbol{\theta}_i^\top H \Delta\boldsymbol{\theta}_j. \end{aligned} \tag{75}$$

We can decompose the quadratic form:

$$\sum_{i,j\in\mathcal{B}_{acc}} \Delta\boldsymbol{\theta}_i^\top H \Delta\boldsymbol{\theta}_j = \sum_{i\in\mathcal{B}_{acc}} \Delta\boldsymbol{\theta}_i^\top H \Delta\boldsymbol{\theta}_i + \sum_{\substack{i,j\in\mathcal{B}_{acc}\\ i\neq j}} \Delta\boldsymbol{\theta}_i^\top H \Delta\boldsymbol{\theta}_j. \tag{76}$$

Substituting equation 76 into equation 75 and grouping yields

$$m_H(\Delta\boldsymbol{\theta}) = \frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} m_H(\Delta\boldsymbol{\theta}_i) - \frac{B-1}{2B^2}\sum_{i\in\mathcal{B}_{acc}} \Delta\boldsymbol{\theta}_i^\top H \Delta\boldsymbol{\theta}_i + \frac{1}{2B^2}\sum_{\substack{i,j\in\mathcal{B}_{acc}\\ i\neq j}} \Delta\boldsymbol{\theta}_i^\top H \Delta\boldsymbol{\theta}_j. \tag{77}$$

By the operator norm bound $\|H\|_{\mathrm{op}} \le M$ and Cauchy–Schwarz,

$$|\Delta\boldsymbol{\theta}_i^\top H \Delta\boldsymbol{\theta}_j| \;\le\; M\,\|\Delta\boldsymbol{\theta}_i\|\,\|\Delta\boldsymbol{\theta}_j\|.$$

Hence, using $\|\Delta\boldsymbol{\theta}_i\| \le r$ for all i,

$$\sum_{i\in\mathcal{B}_{acc}} \Delta\boldsymbol{\theta}_i^\top H \Delta\boldsymbol{\theta}_i \;\le\; MBr^2, \qquad \sum_{\substack{i,j\in\mathcal{B}_{acc}\\ i\neq j}} \Delta\boldsymbol{\theta}_i^\top H \Delta\boldsymbol{\theta}_j \;\ge\; -MB(B-1)r^2. \tag{78}$$

Substituting into equation 77,

$$m_H(\Delta\boldsymbol{\theta}) \;\ge\; \frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} m_H(\Delta\boldsymbol{\theta}_i) - Mr^2\Big(1 - \tfrac{1}{B}\Big). \tag{79}$$

If each accepted subset satisfies $m_H(\Delta\boldsymbol{\theta}_i) \;\ge\; \omega + Mr^2$, then averaging gives $\frac{1}{B}\sum_{i\in\mathcal{B}_{acc}} m_H(\Delta\boldsymbol{\theta}_i) \ge \omega + Mr^2$. Plugging into equation 79 yields

$$m_H(\Delta\boldsymbol{\theta}) \;\ge\; \omega + Mr^2 - Mr^2\Big(1 - \tfrac{1}{B}\Big) = \omega + \frac{Mr^2}{B} \;\ge\; \omega. \tag{80}$$

From Equations 65 and 66 of Lemma E.1, we have that:

$$J(\pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}) - J(\pi_{\boldsymbol{\theta}}) \;\ge\; \underbrace{g(\boldsymbol{\theta})^\top \Delta\boldsymbol{\theta} \;+\; \tfrac{1}{2}\,\Delta\boldsymbol{\theta}^\top H(\boldsymbol{\theta})\,\Delta\boldsymbol{\theta}}_{m_H(\Delta\boldsymbol{\theta})} + o(\|\Delta\boldsymbol{\theta}\|^2) - C\,\underbrace{\sqrt{D_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}\|\pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}})}}_{m_F(\Delta\boldsymbol{\theta})+o(\|\Delta\boldsymbol{\theta}\|^2)} \tag{81}$$

Then, using $m_F(\Delta\boldsymbol{\theta}) < \delta_F$, $m_H(\Delta\boldsymbol{\theta}) > \omega$, and assuming the cubic terms negligible,

$$J(\pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}) - J(\pi_{\boldsymbol{\theta}}) \;\ge\; \omega - C\sqrt{\delta_F}. \tag{82}$$

Thus choosing $\omega \ge C\sqrt{\delta_F}$ guarantees monotonic improvement: $J(\pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}) \ge J(\pi_{\boldsymbol{\theta}})$. □

F PSEUDOCODE OF CAPO

In this Appendix, we present CAPO's algorithm.

Algorithm 1: Curvature-Aware Policy Optimization (CAPO)

Input : Policy π_θ; batch $\mathcal{B}$ of sampled trajectories;
thresholds $(\delta_F,\ \delta_H,\ \delta_H^{high})$;
optimizer for the last-layer model (e.g., SGD or Adam).
Output: Updated policy parameters θ

while *not done* **do**
  `// Collect data with the current policy`
  Sample a batch $\mathcal{B} = \{\tau\}_i^N$ of trajectories, $\tau \sim \pi_\theta$.
  Partition $\mathcal{B}$ into disjoint subsets $\{b_i\}_{i=1}^N$.
  for $i = 1, \ldots, N$ ***in parallel*** **do**
    `// Build last-layer meta-model stats on subset` b_i
    Estimate model-based gradient $\tilde{g}(\psi)$ using Equation 7;
    Propose $\Delta\psi_i$ with the optimizer model (e.g., $\Delta\psi_i = \alpha\,\tilde{g}(\psi)$ for SGD, or Adam's rule)
    Compute directional curvatures $\frac{1}{2}\,\Delta\psi^\top \tilde{H}(\psi)\,\Delta\psi$, $\Delta\psi^\top \tilde{F}(\psi)\,\Delta\psi$ as in Appendix D;
    Compute objective and policy shifts under the last-layer model:
      $m_H(\Delta\psi) \leftarrow \tilde{g}(\psi)^\top \Delta\psi + \frac{1}{2}\,\Delta\psi^\top \tilde{H}(\psi)\,\Delta\psi$, $m_F(\Delta\psi) \leftarrow \frac{1}{2}\,\Delta\psi^\top \tilde{F}(\psi)\,\Delta\psi$.
    `// Local trust-region acceptance test`
    if $\delta_H \le m_H(\Delta\psi_i) \le \delta_H^{high}$ ***and*** $m_F(\Delta\psi_i) \le \delta_F$ **then**
      Mark subset b_i as ACCEPT; add to $\mathcal{B}_{acc}$.
    else
      REJECT b_i.
  `// Compute the actual policy update on accepted data`
  if $\mathcal{B}_{acc} \neq \varnothing$ **then**
    Estimate the objective on accepted samples (e.g., GRPO/PPO surrogate):
      $\widehat{J}(\theta)$ = `pg-objective`$(\pi_\theta;\ \bigcup_{b_i \in \mathcal{B}_{acc}} b_i)$.
    `// Policy Gradient and parameter update`
    $\theta \leftarrow \theta + \alpha\,\widehat{\nabla}_\theta J$
return θ

G REPRODUCIBILITY STATEMENT

Code Release. To ensure the reproducibility of our research findings, we release our code at `https://github.com/luckeciano/stable-pg-llm`. Our implementation is based on PyTorch (Paszke et al., 2017) and HuggingFace (Wolf et al., 2020). All baselines are available in the released code. We also plan to publish all the experiments logs in WandB (Biewald, 2020).

Reproducibility. We detail our methodology in Sections 4.1 and 5 and our experimental setup in Section 6. We provide all hyperparameters used in this work in Appendix J. For all experiments in this paper, we report the results over five seeds with standard errors. For the MATH benchmark, we report in-training performance every step, while for the TEST benchmark set we evaluate checkpoints every 10 learning steps. For better visualization, we applied smoothing with exponential moving average on the curves. All datasets are open-source and available online for academic use.

Compute Resources. We execute all RL experiments using 4 NVIDIA H100 GPUs. Each seed in the regime with aggressive updates takes approximately 4 hours, while the standard regime takes approximately one day. Evaluation is done separately in the same hardware, taking approximately 90 minutes per seed.

LLM Usage Details. We use LLMs for paper writing to improve grammar, enhance clarity and writing flow, and assist with code and mathematical iterations. All outputs generated by the LLMs were thoroughly reviewed and verified by the authors to ensure factual accuracy and correctness.

H COMPUTATIONAL COST ANALYSIS

Execution Time. Table 1 reports a breakdown of CAPO's execution time, including both the model estimations and the masking process. The table shows the average time (in seconds) of each operation, averaged over all learning iterations, measured on our NVIDIA 4×H100 hardware. The total learning iteration time include LLM generations and forward and backward passes. We find that CAPO contributes less than 3% of the total step time in a learning iteration, resulting in minimal training overhead. Most of the cost arises from computing the Adam gradient and updating its moments, since this also requires computing batch gradients on sparse representations. Lastly, the cost of computing the mask is minimal, below 0.01 seconds.

Memory cost. CAPO uses only volatile GPU memory, since all operations are transient and tensors are discarded after the masking generation. The main memory usage comes from maintaining token-level gradient tensors, which have shape (N, T, K, D), corresponding to batch size, completion length, top-K probabilities, and the number of parameters in the last-layer model. In our experiments, with $N = 24$, $T = 1024$, $K = 50$, and $D = 896$, this amounts to a volatile memory footprint of approximately 2 GB, which is minimal given the scale of LLM training. For comparison, this is significantly less expensive than performing KL regularization, which requires storing an additional copy of the LLM in memory for the reference policy.

Step	**Avg. Time (s)**	**% of Total**
Learning Iteration (Total)	135.84	100.00%
LLM Generations	55.50	40.85%
Total CAPO time	**3.99**	**2.94%**
Compute token-level gradients	0.04	0.03%
Compute Adam token gradients	0.51	0.38%
Compute & log m_H	0.09	0.07%
Compute & log m_F	0.01	0.01%
Update Adam Moments	3.34	2.46%
Compute Hessian Mask	0.00	0.00%
Compute Fisher Mask	0.00	0.00%

Table 1: **Breakdown of the execution time of CAPO**. CAPO contributes less than 3% of the total step time, resulting in minimal overhead relative to standard training.

I ADDITIONAL EXPERIMENTS

Ablation of the Optimizer Model. We conducted an ablation study on the impact of the optimizer representation. This choice reflects a trade-off between step accuracy and computational cost: SGD is cheaper, but the LLM policy is optimized with Adam. Figure 6 shows the results on the MATH dataset. For CAPO, representing the optimizer with either SGD or Adam yields similar performance. However, for Dr.CAPO and ReinCAPO, the SGD variant is insufficient to prevent policy collapse. This suggests that matching the optimizer representation provides a more robust choice across different setups.

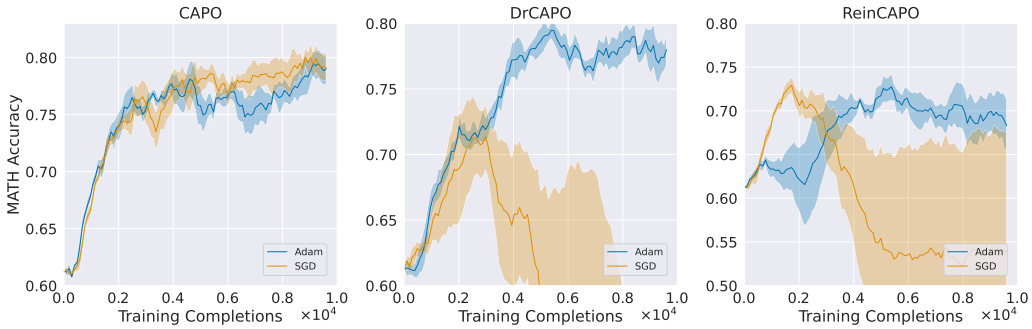

Figure 6: **Ablation study of the optimizer model.** For CAPO, both representations yield similar performance, whereas for Dr.CAPO and ReinCAPO, only the Adam-based representation prevents policy collapse, indicating that matching the optimizer provides a more robust choice across setups.

Is PPO clipping enough to ensure stability? PPO clipping (Schulman et al., 2017) is a heuristic designed to prevent large updates by clipping the probability ratio between the current policy and the old policy that collected the data. This raises the question of whether clipping alone is sufficient to avoid policy collapse in our LLM setup. We note that clipping is primarily intended to facilitate off-policy updates, whereas our experiments with on-policy data already reveal instability in current RL methods. Nevertheless, we conducted additional experiments using off-policy data reused for t iterations under different clipping ratios. Figure 7 shows results for two setups: $t = 2$ (minimal off-policy shift) and $t = 5$ (moderate shift). We find that the standard clipping ratio ($\epsilon = 0.2$) does not prevent collapse. More aggressive ratios alleviate instabilities but reduce performance, likely due to the strong bias introduced in the gradients. This trade-off becomes more pronounced as t increases.

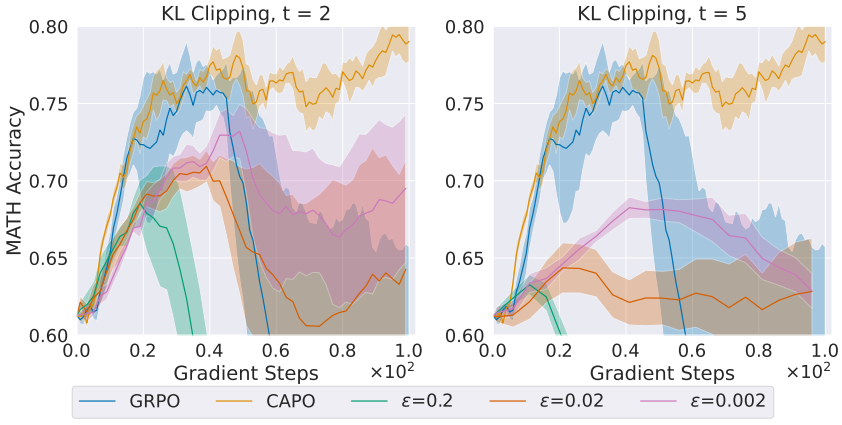

Figure 7: **Effect of "PPO clipping" on GRPO stability.** Standard clipping ($\epsilon = 0.2$) fails to prevent collapse, while more aggressive ratios improve stability but reduce overall performance, with the trade-off becoming more severe as t increases.

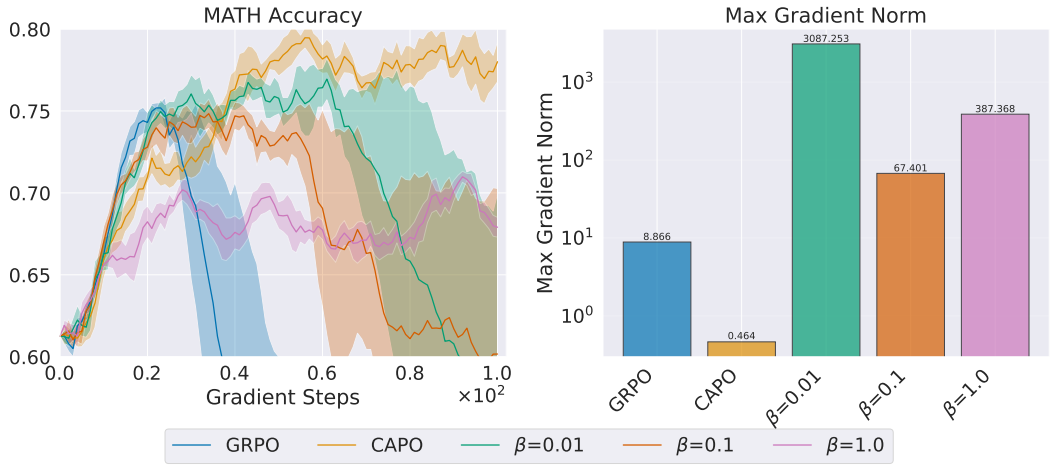

Figure 8: **Effect of KL regularization on GRPO stability.** (Left) Accuracy on the MATH dataset under different levels of KL regularization. Stronger regularization ($\beta = 1.0$) reduces instability but degrades performance. (Right) Maximum gradient norms (before clipping), averaged across seeds. KL regularization produces unbounded gradients that may drive the optimization into unstable regions.

Is KL regularization enough to ensure stability? Another common strategy to mitigate instabilities is to add a KL regularizer that penalizes deviations from the base policy (see Equation 3). The rationale is that keeping the policy close to the base model may prevent large distributional shifts, such as those associated with policy collapse. In Figure 8 (left), we test different levels of regularization. We observe a trend similar to clipping: only stronger regularization ($\beta = 1.0$) helps prevent catastrophic updates, but at the cost of performance.

A more fundamental limitation of KL regularization becomes evident when examining its gradient:

$$\nabla_\theta \mathcal{D}_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{base}}) = \mathbb{E}_{s\sim d^\pi,\, a\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a \mid s)\left(\log \frac{\pi_\theta(a \mid s)}{\pi_{\text{base}}(a \mid s)} + 1\right)\right]. \tag{83}$$

Differentiating through the KL term introduces a multiplicative $\log$ factor, which can produce unbounded gradients. More concretely, as $\pi_{\text{base}}(a \mid s) \to 0$, the gradient magnitude diverges, effectively "exploding" the LLM policy gradient. We observe this empirically in Figure 8 (right), which shows the maximum gradient norms (before gradient clipping) over training, averaged across seeds. While gradient clipping can reduce the gradient's magnitude, it does not alter its direction, which may still drive the optimization into unstable regions.

Finally, there are also practical drawbacks to KL regularization. First, it requires storing a full copy of the base model in memory, which has led prior work to abandon the technique (Liu et al., 2025b). Second, differentiating KL estimates as loss functions typically yields biased approximations of the true KL gradient (Tang & Munos, 2025).

J HYPERPARAMETERS

In this section, we present the hyperparameters used in our experiments. Table 2 lists the hyperparameters common to all training configurations and algorithms. Table 3 specifies the learning rate and batch size for the conservative and aggressive setups. Finally, Table 4 reports the hyperparameters specific to curvature-aware masking, along with their values for each method. Due to compute budget constraints, we performed manual hyperparameter tuning, primarily searching across different orders of magnitude of both δ_H and δ_F. For simplicity, we implemented a single symmetric threshold for the Hessian, i.e., rejecting samples outside the interval $-\delta_H < m_H < \delta_H$.

Hyperparameter	**Value**
LLM Generation	
Max Prompt Length	512
Max Completion Length	1024
Num Generations per Prompt	8
Temperature	0.9
Training	
Gradient Steps	100
Warmup Ratio	0.1
Iterations per Batch	1
Optimizer	Adam
LR Scheduler	Cosine
KL β	0.0

Table 2: **Training Hyperparameters.**

Hyperparameter	**Standard Setup**	**Aggressive Setup**
Learning Rate	3×10^{-6}	1.5×10^{-5}
Total Batch Size	1152	96

Table 3: **Hyperparameters for the standard (conservative) and aggressive regimes.**

Hyperparameter	**CAPO**	**Dr.CAPO**	**ReinCAPO**
Hessian δ_H	10^{-2}	5×10^{-4}	10^{-1}
Fisher δ_F	10^{-4}	10^{-3}	10^{-5}

Table 4: **Curvature-aware masking thresholds for CAPO, Dr.CAPO and ReinCAPO.**

K MONOTONIC POLICY IMPROVEMENT UNDER CAPO IN THE UNDISCOUNTED, FINITE-HORIZON SETTING

Appendix E formalizes the conditions under which CAPO guarantees monotonic improvement in the standard discounted, infinite-horizon setting. Although this formulation is general and aligned with prior RL literature, this section extends the analysis to the undiscounted, finite-horizon setting, which better reflects the LLM reasoning setup and is more consistent with the assumptions underlying practical algorithms such as GRPO.

For this analysis, we consider a finite-horizon Markov decision process (MDP) with horizon $T \in \mathbb{N}$, state space $\mathcal{S}$, action space $\mathcal{A}$, transition kernel $P(s' \mid s, a)$, reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and initial state distribution ρ_o. A (stochastic) policy π is a conditional distribution $\pi(a \mid s)$ over actions given states. The return of a policy π is given by:

$$J(\pi) := \mathbb{E}_\pi \left[\sum_{t=0}^{T-1} R(s_t, a_t) \right]. \tag{84}$$

Furthermore, we define the advantage function as $A_\pi(s, a) := Q_\pi(s, a) - V_\pi(s)$. For a second policy π', we also define the π'-averaged advantage of π at state s: $\bar{A}_\pi^{\pi'}(s) := \mathbb{E}_{a \sim \pi'(\cdot|s)}[A_\pi(s, a)]$.

Lemma K.1 (Performance Difference Lemma, Finite Horizon, $\gamma = 1$)**.** *Let π and π' be two policies. Then*

$$J(\pi') - J(\pi) = \sum_{t=0}^{T-1} \mathbb{E}_{s \sim d_{\pi',t}} \big[\bar{A}_\pi^{\pi'}(s) \big], \tag{85}$$

where $d_{\pi,t}(s) := \Pr_\pi(s_t = s)$ denotes the time-t state-marginal under π.

Proof. We start from the identity $Q_\pi(s, a) = r(s, a) + \mathbb{E}_{s' \sim P(\cdot|s,a)}[V_\pi(s')]$. Rearranging,

$$r(s, a) = Q_\pi(s, a) - \mathbb{E}_{s'}[V_\pi(s')] = A_\pi(s, a) + V_\pi(s) - \mathbb{E}_{s'}[V_\pi(s')]. \tag{86}$$

Consider a trajectory $(s_0, a_0, \ldots, s_{T-1}, a_{T-1})$ generated by policy π'. Then

$$\sum_{t=0}^{T-1} r(s_t, a_t) = \sum_{t=0}^{T-1} \Big(A_\pi(s_t, a_t) + V_\pi(s_t) - \mathbb{E}[V_\pi(s_{t+1}) \mid s_t, a_t] \Big). \tag{87}$$

Taking expectation under π' and using the law of total expectation,

$$J(\pi') = \mathbb{E}_{\pi'} \left[\sum_{t=0}^{T-1} A_\pi(s_t, a_t) \right] + \mathbb{E}_{\pi'} \left[\sum_{t=0}^{T-1} V_\pi(s_t) - V_\pi(s_{t+1}) \right], \tag{88}$$

where $V_\pi(s_T) := 0$ by definition. The second sum telescopes:

$$\sum_{t=0}^{T-1} V_\pi(s_t) - V_\pi(s_{t+1}) = V_\pi(s_0) - V_\pi(s_T) = V_\pi(s_0). \tag{89}$$

Thus,

$$J(\pi') = \mathbb{E}_{\pi'} \left[\sum_{t=0}^{T-1} A_\pi(s_t, a_t) \right] + \underbrace{\mathbb{E}_{s_0 \sim \rho_o}[V_\pi(s_0)]}_{J(\pi)}. \tag{90}$$

Therefore,

$$J(\pi') - J(\pi) = \sum_{t=0}^{T-1} \mathbb{E}_{s_t, a_t \sim \pi'}[A_\pi(s_t, a_t)]. \tag{91}$$

We can rewrite each term as

$$\mathbb{E}_{s_t, a_t \sim \pi'}[A_\pi(s_t, a_t)] = \mathbb{E}_{s \sim d_{\pi',t}} \Big[\mathbb{E}_{a \sim \pi'(\cdot|s)}[A_\pi(s, a)] \Big] = \mathbb{E}_{s \sim d_{\pi',t}}[\bar{A}_\pi^{\pi'}(s)], \tag{92}$$

which proves the claimed identity. □

We now bound the difference between the state marginals $d_{\pi',t}$ and $d_{\pi,t}$ in terms of how different the policies are. For $t \geq 0$, we first define the policy-induced transition kernels:

$$P_\pi(s' \mid s) := \sum_a \pi(a \mid s) P(s' \mid s, a), \qquad P_{\pi'}(s' \mid s) := \sum_a \pi'(a \mid s) P(s' \mid s, a). \tag{93}$$

Then $d_{\pi,t+1}^\top = d_{\pi,t}^\top P_\pi$ and $d_{\pi',t+1}^\top = d_{\pi',t}^\top P_{\pi'}$.

Lemma K.2 (State-Distribution Shift Bound, Finite Horizon)**.** *Let π, π' be two policies with the same initial state distribution $d_{\pi,0} = d_{\pi',0} = \rho_o$. Then, for all $t = 0, \ldots, T-1$,*

$$\|d_{\pi',t} - d_{\pi,t}\|_1 \;\leq\; 2 \sum_{k=0}^{t-1} \mathbb{E}_{s \sim d_{\pi,k}} \Big[D_{\mathrm{TV}}\big(\pi(\cdot \mid s), \pi'(\cdot \mid s)\big) \Big]. \tag{94}$$

Proof. Define the difference vector $\delta_t := d_{\pi',t} - d_{\pi,t}$. Then:

$$\begin{aligned}
\delta_{t+1} &= d_{\pi',t+1} - d_{\pi,t+1} \\
&= d_{\pi',t} P_{\pi'} - d_{\pi,t} P_\pi \\
&= (d_{\pi',t} - d_{\pi,t}) P_{\pi'} + d_{\pi,t}(P_{\pi'} - P_\pi) \\
&= \delta_t P_{\pi'} + d_{\pi,t}(P_{\pi'} - P_\pi).
\end{aligned} \tag{95}$$

Since $P_{\pi'}$ is row-stochastic, $\|\delta_t P_{\pi'}\|_1 \leq \|\delta_t\|_1$. Next, we bound the term $d_{\pi,t}(P_{\pi'} - P_\pi)$. Let $w := d_{\pi,t}(P_{\pi'} - P_\pi)$, so $w(s') = \sum_s d_{\pi,t}(s)\big(P_{\pi'}(s' \mid s) - P_\pi(s' \mid s)\big)$. Then:

$$\begin{aligned}
\|w\|_1 = \sum_{s'} |w(s')| &= \sum_{s'} \Big| \sum_s d_{\pi,t}(s)\big(P_{\pi'}(s' \mid s) - P_\pi(s' \mid s)\big) \Big| \\
&\leq \sum_{s'} \sum_s d_{\pi,t}(s) \big| P_{\pi'}(s' \mid s) - P_\pi(s' \mid s) \big| \\
&= \sum_s d_{\pi,t}(s) \sum_{s'} \big| P_{\pi'}(s' \mid s) - P_\pi(s' \mid s) \big| \\
&= \sum_s d_{\pi,t}(s) \, \|P_{\pi'}(\cdot \mid s) - P_\pi(\cdot \mid s)\|_1.
\end{aligned} \tag{96}$$

For each fixed s, using $P_{\pi'}(s' \mid s) - P_\pi(s' \mid s) = \sum_a (\pi'(a \mid s) - \pi(a \mid s)) P(s' \mid s, a)$ and the fact that $\sum_{s'} P(s' \mid s, a) = 1$, we obtain:

$$\begin{aligned}
\|P_{\pi'}(\cdot \mid s) - P_\pi(\cdot \mid s)\|_1 &= \sum_{s'} \Big| \sum_a (\pi'(a \mid s) - \pi(a \mid s)) P(s' \mid s, a) \Big| \\
&\leq \sum_{s'} \sum_a |\pi'(a \mid s) - \pi(a \mid s)| P(s' \mid s, a) \\
&= \sum_a |\pi'(a \mid s) - \pi(a \mid s)| \\
&= 2 D_{\mathrm{TV}}(\pi(\cdot \mid s), \pi'(\cdot \mid s)).
\end{aligned} \tag{97}$$

Hence $\|w\|_1 \leq 2 \sum_s d_{\pi,t}(s) D_{\mathrm{TV}}(\pi(\cdot \mid s), \pi'(\cdot \mid s))$. Combining these two bounds and using the triangle inequality,

$$\begin{aligned}
\|\delta_{t+1}\|_1 &= \|\delta_t P_{\pi'} + d_{\pi,t}(P_{\pi'} - P_\pi)\|_1 \\
&\leq \|\delta_t P_{\pi'}\|_1 + \|d_{\pi,t}(P_{\pi'} - P_\pi)\|_1 \\
&\leq \|\delta_t\|_1 + 2\alpha_t.
\end{aligned} \tag{98}$$

By definition, $d_{\pi',0} = d_{\pi,0}$, so $\delta_0 = 0$ and $\|\delta_0\|_1 = 0$. Unrolling the recursion:

$$\|\delta_t\|_1 \leq 2 \sum_{k=0}^{t-1} \mathbb{E}_{s \sim d_{\pi,k}} \Big[D_{\mathrm{TV}}\big(\pi(\cdot \mid s), \pi'(\cdot \mid s)\big) \Big]. \tag{99}$$

□

We now define a surrogate objective based on the reference policy π and the state distributions $d_{\pi,t}$.

Lemma K.3 (Surrogate–True Performance Gap, Finite Horizon)**.** *For any policies π and π', with $D_{\mathrm{KL}}(\pi\|\pi')$ the average forward KL under d_π,*

$$J(\pi') \;\geq\; L_\pi(\pi') \;-\; C\,\sqrt{D_{\mathrm{KL}}(\pi\|\pi')}, \qquad C := T\sqrt{\frac{(T-1)(2T-1)}{3}}\epsilon, \tag{100}$$

where $|A^\pi(s,a)| \leq \epsilon$ with ϵ finite, and $L_\pi(\pi') := J(\pi) + \sum_{t=0}^{T-1} \mathbb{E}_{s\sim d_{\pi,t}}\big[\bar{A}_\pi^{\pi'}(s)\big]$.

Proof. By Lemma K.1, $J(\pi') - J(\pi) = \sum_{t=0}^{T-1} \mathbb{E}_{s\sim d_{\pi',t}}[\bar{A}_\pi^{\pi'}(s)]$. Subtracting the surrogate:

$$\begin{aligned} J(\pi') - L_\pi(\pi') &= \sum_{t=0}^{T-1} \Big(\mathbb{E}_{s\sim d_{\pi',t}} \bar{A}_\pi^{\pi'}(s) - \mathbb{E}_{s\sim d_{\pi,t}} \bar{A}_\pi^{\pi'}(s)\Big) \\ &= \sum_{t=0}^{T-1} \sum_s \big(d_{\pi',t}(s) - d_{\pi,t}(s)\big) \bar{A}_\pi^{\pi'}(s). \end{aligned} \tag{101}$$

Taking absolute values and using $|\bar{A}_\pi^{\pi'}(s)| \leq \epsilon$ and applying Lemma K.2:

$$\begin{aligned} \big|J(\pi') - L_\pi(\pi')\big| \leq \sum_{t=0}^{T-1} \epsilon \|d_{\pi',t} - d_{\pi,t}\|_1 &\leq \epsilon \sum_{t=0}^{T-1} 2 \sum_{k=0}^{t-1} \mathbb{E}_{s\sim d_{\pi,k}} \Big[D_{\mathrm{TV}}\big(\pi(\cdot \mid s), \pi'(\cdot \mid s)\big)\Big] \\ &= 2\epsilon \sum_{k=0}^{T-1} \mathbb{E}_{s\sim d_{\pi,k}} \Big[D_{\mathrm{TV}}\big(\pi(\cdot \mid s), \pi'(\cdot \mid s)\big)\Big] \sum_{t=k+1}^{T-1} 1 \\ &= 2\epsilon \sum_{k=0}^{T-1} (T-1-k)\, \mathbb{E}_{s\sim d_{\pi,k}} \Big[D_{\mathrm{TV}}\big(\pi(\cdot \mid s), \pi'(\cdot \mid s)\big)\Big]. \end{aligned} \tag{102}$$

For the KL-based bound, we use Pinsker's inequality and Jensen's inequality. For each t:

$$\begin{aligned} \mathbb{E}_{s\sim d_{\pi,t}} D_{\mathrm{TV}}(\pi(\cdot \mid s), \pi'(\cdot \mid s)) &\leq \mathbb{E}_{s\sim d_{\pi,t}} \sqrt{\tfrac{1}{2} D_{\mathrm{KL}}(\pi(\cdot \mid s) \,\|\, \pi'(\cdot \mid s))} \\ &\leq \sqrt{\tfrac{1}{2} \mathbb{E}_{s\sim d_{\pi,t}} \Big[D_{\mathrm{KL}}(\pi(\cdot \mid s) \,\|\, \pi'(\cdot \mid s))\Big]}. \end{aligned} \tag{103}$$

For conciseness, we define $D_k := D_{\mathrm{KL}}(\pi(\cdot \mid s) \,\|\, \pi'(\cdot \mid s))$. Then:

$$\big|J(\pi') - L_\pi(\pi')\big| \leq 2\epsilon \sum_{k=0}^{T-1} (T-1-k) \sqrt{\tfrac{1}{2} D_k} = \sqrt{2}\,\epsilon \sum_{k=0}^{T-1} b_k \sqrt{D_k}, \tag{104}$$

where we have set $b_k := T - 1 - k$. By Cauchy–Schwarz,

$$\sum_{k=0}^{T-1} b_k \sqrt{D_k} \leq \sqrt{\sum_{k=0}^{T-1} b_k^2} \sqrt{\sum_{k=0}^{T-1} D_k}. \tag{105}$$

We note that

$$\sum_{k=0}^{T-1} b_k^2 = \sum_{j=0}^{T-1} j^2 = \frac{(T-1)T(2T-1)}{6}, \qquad \sum_{k=0}^{T-1} D_k = T\,\bar{D}_{\mathrm{KL}}. \tag{106}$$

Therefore

$$\begin{aligned} \big|J(\pi') - L_\pi(\pi')\big| &\leq \sqrt{2}\,\epsilon \sqrt{\frac{(T-1)T(2T-1)}{6}} \sqrt{T\,\bar{D}_{\mathrm{KL}}} \\ &= T\sqrt{\frac{(T-1)(2T-1)}{3}}\,\epsilon\,\sqrt{\bar{D}_{\mathrm{KL}}}. \end{aligned} \tag{107}$$

□

The proof of Theorem 5.1 for the finite-horizon setting follows exactly the one in Appendix E, but applying Lemma K.3 instead of Lemma E.1.

Infinite-Horizon vs. Finite-Horizon bounds. We highlight that, in both settings, the final guarantee takes the same form $J(\pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}) - J(\pi_{\boldsymbol{\theta}}) \;\geq\; \omega - C\sqrt{\delta_F}$, where $C = \frac{2\gamma}{(1-\gamma)^2}\,\epsilon\,\sqrt{2}$ for the infinite-horizon case, and $C = T\sqrt{\frac{(T-1)(2T-1)}{3}}\,\epsilon$ for the finite-horizon case. In both cases, the constant C scales as $\mathcal{O}(H_{\text{eff}}^2)$, where H_{eff} denotes the effective horizon: $H_{\text{eff}} = T$ in the finite-horizon setting, and $H_{\text{eff}} = \frac{1}{1-\gamma}$ in the infinite-horizon setting. Practically, this implies that both bounds are equally tight within their respective regimes.

L A CLOSER LOOK AT MODEL ESTIMATES $\hat{m}_F$ AND THE KL POLICY SHIFT

In this section, we analyze the relationship between the model's estimate of directional Fisher curvature, $\hat{m}_F$, and the actual policy shift induced by an update, measured by $D_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}||\pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}})$. Our goals are two-fold: (i) to clarify what CAPO requires from the underlying model in order to approximate a trust-region and to assess how well this approximation holds, and (ii) to examine the impact of CAPO's updates on the true change in policy.

Does CAPO require a fully calibrated model? Although well-calibrated estimates are a *sufficient* condition for CAPO's data-selection mechanism to function effectively, they are not *necessary*. To illustrate this, consider a simple case where the estimated directional Fisher curvature satisfies $\hat{m}_F = \alpha\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}||\pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}})$, $\alpha > 0$, where $\alpha >> 1$ or $\alpha << 1$. Such a model is clearly miscalibrated, yet it preserves a strong correlation with the true policy shift. In CAPO, if we aim to enforce the trust-region condition $\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}, |, \pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}) < \delta$, we can simply set the Fisher-threshold to $\delta_F = \alpha\delta$, which recovers the desired constraint. More generally, CAPO only requires that the estimates be *monotonically correlated* with the true policy change, so that large prospective shifts (those most likely to trigger instability or collapse) are reliably identified.

A natural way to evaluate the quality of the model's estimates is to measure their correlation with the true policy changes. Although we do not have direct access to this quantity, we can estimate it via samples. In particular, the KL divergence can be reliably estimated using a standard Monte Carlo estimator, which has manageable variance and leverages token-level information. We therefore compute these estimates and report the resulting Spearman correlations in the Table 5, where $\hat{m}_F$ is evaluated under both GRPO and CAPO updates at both token and global level. We find that the model estimates exhibit a moderately strong correlation with the actual policy change, indicating a consistent monotonic relationship. Notably, this correlation remains high under both GRPO and CAPO, suggesting that the estimates are meaningful even when they are not used to intervene in the update.

Estimate	ρ **(GRPO)**	ρ **(CAPO)**
$\hat{m}_F$ (Token)	0.622	0.459
$\hat{m}_F$ (Global)	0.596	0.498

Table 5: **Spearman correlations ρ between Fisher directional curvature estimates $\hat{m}_F$ and the estimated policy change $\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}||\pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}})$.** We report correlations for both GRPO and CAPO updates. The results indicate that the estimates $\hat{m}_F$ maintain a consistent monotonic relationship with the true policy shift across algorithms, reliable identifying the scale of the policy shifts

.

Ultimately, does CAPO induce a bound on the true $D_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}||\pi_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}})$? In Figure 9, we present the policy shifts over the course of training for both algorithms. GRPO frequently presents peaked shifts, which are often associated with unstable or overly aggressive updates. In contrast, CAPO generally maintains stable, small shifts, suggesting that it is effective in practically implementing a trust-region behavior throughout training.

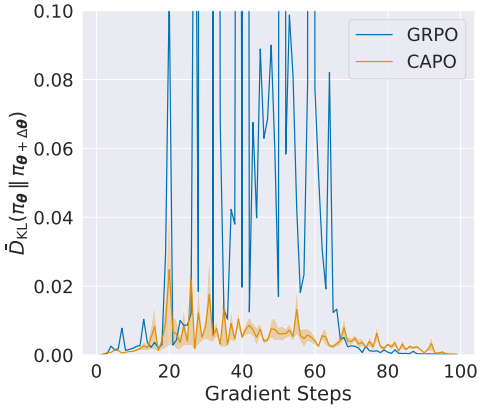

Figure 9: **Estimated policy KL shifts during training.** GRPO exhibits frequent sharp spikes in policy divergence, indicative of unstable updates, whereas CAPO maintains consistently small shifts, reflecting its ability to enforce trust-region–like behavior throughout training.

M FURTHER QUESTIONS

This Appendix presents additional clarification questions aimed at improving the understanding of the proposed method and experiments. These questions were raised during the peer-review process, and we refer to the OpenReview page for the full discussion.

What is the effect of token selection in the sample efficiency evaluation? In Figure 10, we plot the accuracy curves (analogous to Figs. 1 and 2) as a function of the accepted tokens. We observe that these curves closely resemble those obtained when accuracy is plotted against the number of completions. This suggests that the effect of masking on the total number of generated (and accepted) tokens is small, consistent with the rejection rates reported in Figure 5. It also indicates that the learned policies behave similarly in terms of token generation, showing that CAPO improves training sample efficiency without incurring additional inference-time costs.

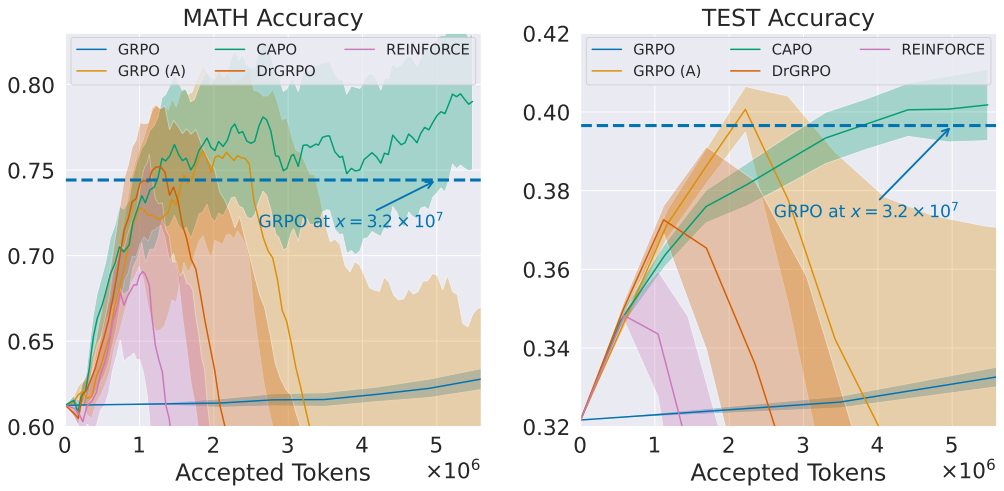

Figure 10: **Sample efficiency curves as a function of the number of accepted tokens**. The trends closely match those obtained when using the number of completions, indicating that masking has minimal impact on token generation and that CAPO improves sample efficiency without added inference cost.

What are the similarities between CAPO and TRPO? What are the differences? In terms of similarities, both CAPO and TRPO share the same motivation: devise a conservative optimization procedure that implements a safe optimization region, typically expressed as a KL ball constraint. This idea predates TRPO, with its roots in natural gradient methods from optimization literature Amari (1998); Amari et al. (1995). What both CAPO and TRPO do is to devise practical instantiations of the natural gradient that is suitable for their respective problem settings.

Methodologically, TRPO incorporates *only* the Fisher matrix in its updates, relying on a first-order approximation of the objective. In contrast, CAPO additionally leverages second-order curvature information of the objective through its Hessian, as shown in Equation 5 and further incorporated in the theoretical development in Equation 68. The main difference, however, lies in the implementation, which crucially leads to different scalability properties.

TRPO incorporates the Fisher matrix by employing a Conjugate-Gradient (CG) algorithm to approximate the natural gradient step without fully materializing the Fisher matrix. Then, TRPO employs a line search algorithm to solve the constrained optimization problem. The CG algorithm involves maintaining five vectors of size d (the gradient, current iterate, the residual, the search direction, and the matrix-vector buffer), where d is the number of parameters in the policy. While this memory cost is feasible for small deep networks (as usual in traditional Deep RL research), it is prohibitive for LLM scale, where d is in the billions.

Furthermore, the CG algorithm is iterative, and each iteration costs roughly the same as a backward pass, unless you sacrifice your Fisher matrix estimation by subsampling data. TRPO uses ten itera-

tions. Considering the execution time in our setup (Appendix H), this overhead is also prohibitive. Lastly, the line search algorithm requires M additional forward passes in the whole batch (M is the number of search trials), which is also a substantial cost in our setup (also illustrated in Appendix H). Overall, TRPO's memory and execution costs are prohibitive to LLM scale. CAPO, in contrast, leverages the last layer model and the optimizations described in Section 4.1, resulting in much lower costs, as evaluated in Table 1 of Appendix H.

In summary, while TRPO and CAPO share the same motivation and draw from the same seminal work on natural gradients, CAPO offers a formulation that scales to the memory and compute demands of LLM policies.