EMA Policy Gradient: Taming Reinforcement Learning for LLMs with EMA Anchor and Top-k KL

Lunjun Zhang [1] **Jimmy Ba** [1]

Abstract

Reinforcement Learning (RL) has enabled Large Language Models (LLMs) to acquire increasingly complex reasoning and agentic behaviors. In this work, we propose two simple techniques to improve policy gradient algorithms for LLMs. First, we replace the fixed anchor policy during RL with an Exponential Moving Average (EMA), similar to a target network in deep Q-learning. Second, we introduce Top-k KL estimator, which allows for flexible interpolation between exact KL and sampled KL. We derive the stability conditions for using EMA anchor; moreover, we show that our Top-k KL estimator yields both unbiased KL values and unbiased gradients at any k, while bringing the benefits of exact KL. When combined with GRPO, the two techniques (**EMA-PG**) lead to a significant performance boost. On math reasoning, it allows R1-distilled Qwen-1.5B to reach 53.9% on OlympiadBench compared to 50.8% by GRPO. On agentic RL domains, with Qwen-3B base, EMA-PG improves GRPO by an average of 33.3% across 7 datasets of Q&A with search engines, including 29.7% $\rightarrow$ 44.1% on HotpotQA, 27.4% $\rightarrow$ 40.1% on 2WikiMultiHopQA. Overall, we show that EMA-PG is a simple, principled, and powerful approach to scaling RL for LLMs.

1. Introduction

As the pre-training paradigm began to show diminishing returns for large-language models (LLMs), reinforcement learning (RL) has re-emerged as a powerful mechanism for LLMs to continually self-improve based on self-generated experience. While RL was initially only used for alignment within the RL from Human Feedback (RLHF) framework (Christiano et al., 2017; Ouyang et al., 2022; Bai et al., 2022a;b), it has since then been applied to reasoning models such as o1 (Jaech et al., 2024) and R1 (Guo et al., 2025).

[1]Department of Computer Science, University of Toronto. Correspondence to: Lunjun Zhang <lunjun@cs.toronto.edu>.

Published as a paper at the 1st FoGen workshop, ICML 2026, Seoul, South Korea, 2026. Copyright 2026 by the author(s).

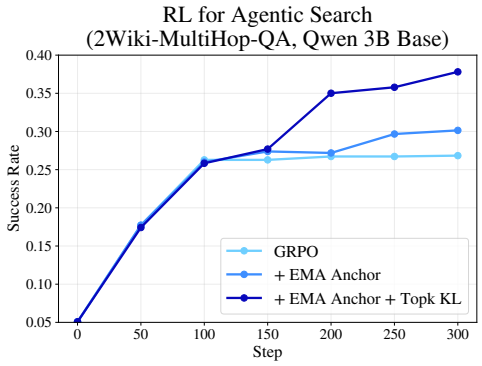

Figure 1. Using EMA anchor policy and Top-k KL estimator significantly improves the performance of RL algorithms like GRPO.

Perhaps surprisingly, the predominant RL algorithm for LLMs today is still REINFORCE (Williams, 1992; Sutton et al., 1999) and its modern off-policy variants like PPO (Schulman et al., 2017b), GRPO (Shao et al., 2024), and CISPO (Chen et al., 2025). This motivates us to study the foundational building blocks of policy gradient algorithms, so that such advancements can be broadly applied to LLMs.

The use of a target network has been the common practice since the dawn of deep Q-learning (Mnih et al., 2015). A target network uses an Exponential Moving Average (EMA) to update its knowledge while producing stable supervision targets. Using EMA has seen success in self-supervised learning as well (Grill et al., 2020b). In RLHF, the idea of an EMA anchor policy has been explored in TR-DPO (Gorbatovski et al., 2024) and WARP (Ramé et al., 2024). A crucial difference between our work and theirs is how the EMA anchor is used: prior work optimizes sequence-level KL, but we optimize token-level KL, which we show makes a crucial difference in reasoning and agentic settings.

KL regularization is a key component of policy optimization. Initially, it was a way to enforce trust-region (Schulman et al., 2015) and ensure training stability. In LLM alignment, it mitigates reward hacking (Gao et al., 2023). In LLM reasoning, it helps preserve base model capabilities (Liu et al., 2025a). However, what KL estimator to use for LLM has long been a curious question. InstructGPT (Ouyang et al., 2022) adds a sequence-level sampled KL to the reward; GRPO (Shao et al., 2024) uses token-

Table 1. **Token-level KL Estimators**. For brevity, we omit the conditioning of π_θ and π_{ref} on past tokens. Notations: sg is stop-gradient, $p \sim \pi_\theta(\cdot)$ is the sampled token, $r = \pi_\theta(p)/\text{sg}(\pi_\theta(p))$, $w = \pi_{\text{ref}}(p)/\pi_\theta(p)$; $\mathcal{V}$ is the vocabulary, $\widehat{w}(j) = \pi_{\text{ref}}(j)/\pi_\theta(j), \forall j \in \mathcal{V}$. The estimators being highlighted in blue are the KL estimators that provide both unbiased KL values and unbiased KL gradients.

$\mathbb{E}_{\pi_\theta}[\hat{D}_f(\pi_\theta, \pi_{\text{ref}})]$	Expression	Value	Unbiased?	Gradient	Unbiased?	Memory	Overall
Reverse KL: exact	$\sum_{j\in\mathcal{V}} \pi_\theta(j)[-\log \widehat{w}(j)]$	$\text{KL}(\pi_\theta, \pi_{\text{ref}})$	✓	$\nabla_\theta \text{KL}(\pi_\theta, \pi_{\text{ref}})$	✓	$O(\lvert\mathcal{V}\rvert)$	×
K1	$-\log w$	$\text{KL}(\pi_\theta, \pi_{\text{ref}})$	✓	$\nabla_\theta 0$	×	$O(1)$	×
K2	$(\log w)^2/2$	$\mathbb{E}_{\pi_\theta}[(\log w)^2/2]$	×	$\nabla_\theta \text{KL}(\pi_\theta, \pi_{\text{ref}})$	✓	$O(1)$	×
K3	$-\log w + w - 1$	$\text{KL}(\pi_\theta, \pi_{\text{ref}})$	✓	$\nabla_\theta \text{KL}(\pi_{\text{ref}}, \pi_\theta)$	×	$O(1)$	×
K3^{++}	$r \cdot (-\log w + w - 1)$	$\text{KL}(\pi_\theta, \pi_{\text{ref}})$	✓	$\nabla_\theta \text{KL}(\pi_\theta, \pi_{\text{ref}})$	✓	$O(1)$	✓
K4	$r \cdot \text{sg}(-\log w)$	$\text{KL}(\pi_\theta, \pi_{\text{ref}})$	✓	$\nabla_\theta \text{KL}(\pi_\theta, \pi_{\text{ref}})$	✓	$O(1)$	✓
Topk**Reverse**KL	Alg 1	$\text{KL}(\pi_\theta, \pi_{\text{ref}})$	✓	$\nabla_\theta \text{KL}(\pi_\theta, \pi_{\text{ref}})$	✓	$O(k)$	✓
Forward KL: exact	$\sum_{j\in\mathcal{V}} \pi_{\text{ref}}(j)[\log \widehat{w}(j)]$	$\text{KL}(\pi_{\text{ref}}, \pi_\theta)$	✓	$\nabla_\theta \text{KL}(\pi_{\text{ref}}, \pi_\theta)$	✓	$O(\lvert\mathcal{V}\rvert)$	×
K5	$\text{sg}(w) \log w + \log r$	$\text{KL}(\pi_{\text{ref}}, \pi_\theta)$	✓	$\nabla_\theta \text{KL}(\pi_{\text{ref}}, \pi_\theta)$	✓	$O(1)$	✓
Topk**Forward**KL	Alg 2	$\text{KL}(\pi_{\text{ref}}, \pi_\theta)$	✓	$\nabla_\theta \text{KL}(\pi_{\text{ref}}, \pi_\theta)$	✓	$O(k)$	✓

Algorithm 1 Topk**Reverse**KL$(\pi_\theta, \pi_{\text{ref}}, \pi_{\text{sampling}}, p, q)$

Require: Policy π_θ, Anchor π_{ref} for $\text{KL}(\pi_\theta, \pi_{\text{ref}})$
Require: Sampling policy π_{sampling}, sampled token p
Require: Set of top-K logit indices q from sampling policy
$\widehat{\text{KL}}_{\text{trun}} = \sum_{j\in q} \pi_\theta(j)[\log \pi_\theta(j) - \log \pi_{\text{ref}}(j)]$
$r_\theta = \pi_\theta(p)/\text{sg}(\pi_\theta(p)), \quad w_\theta = \pi_{\text{ref}}(p)/\pi_\theta(p)$
$\widehat{\text{KL}}_{\text{sampled}} = r_\theta \cdot \text{sg}(-\log w_\theta)$
$s = \text{sg}(\pi_\theta(p)/\pi_{\text{sampling}}(p))$ {clip to $[s_{\min}, s_{\max}]$}
return $\widehat{\text{KL}}_{\text{trun}} + s\, \mathbb{1}(p \notin q)\, \widehat{\text{KL}}_{\text{sampled}}$

Algorithm 2 Topk**Forward**KL$(\pi_\theta, \pi_{\text{ref}}, \pi_{\text{sampling}}, p, q)$

Require: Policy π_θ, Anchor π_{ref} for $\text{KL}(\pi_{\text{ref}}, \pi_\theta)$
Require: Sampling policy π_{sampling}, sampled token p
Require: Set of top-K logit indices q from anchor policy
$\widehat{\text{KL}}_{\text{trun}} = \sum_{j\in q} \pi_{\text{ref}}(j)[\log \pi_{\text{ref}}(j) - \log \pi_\theta(j)]$
$r_\theta = \pi_\theta(p)/\text{sg}(\pi_\theta(p)), \quad w_\theta = \pi_{\text{ref}}(p)/\pi_\theta(p)$
$\widehat{\text{KL}}_{\text{sampled}} = \text{sg}(w_\theta) \log w_\theta + \log r_\theta$
$s = \text{sg}(\pi_\theta(p)/\pi_{\text{sampling}}(p))$ {clip to $[s_{\min}, s_{\max}]$}
return $\widehat{\text{KL}}_{\text{trun}} + s\, \mathbb{1}(p \notin q)\, \widehat{\text{KL}}_{\text{sampled}}$

level sampled KL; the two objectives are different (Tang & Munos, 2025). Moreover, knowledge distillation (Hinton et al., 2015) shows that leveraging logits in KL computation can significantly facilitate learning. We analyze the trade-offs between sequence-level KL vs token-level KL, sampled KL vs exact KL, to identify the best practices in Top-k KL.

We propose **two very simple changes** to policy gradients. Given an RL update for LLM π_θ on prompt x:

$$\theta \leftarrow \theta + \alpha(\underbrace{A(x,y)\nabla_\theta \log \pi_\theta(y|x)}_{\text{Any policy gradient algorithm}} - \beta \underbrace{\nabla_\theta \widehat{\text{KL}}}_{\text{What we change}})$$

where y is the LLM response of length L, and $A(x,y)$ is the advantage function. We only change ***what*** the KL is regularizing against and ***how*** the KL is computed:

$$\boxed{\widehat{\text{KL}} = \sum_{l=0}^{L-1} \text{TopkKL}(\pi_\theta(\cdot|x, y_{0:l-1}), \pi_{\theta_{\text{ema}}}(\cdot|x, y_{0:l-1}))}$$

where TopkKL is an unbiased token-level KL estimator that computes exact KL on the top-k indices of logits and a sampled KL on the rest; θ_{ema} is an EMA target network:

$$\boxed{\theta_{\text{ema}} \leftarrow \eta \cdot \theta_{\text{ema}} + (1-\eta) \cdot \theta}$$

The two changes make a significant difference on RL training of reasoning and agentic models. On math, it allows R1-distilled Qwen-1.5B to reach 53.9% on OlympiadBench (text-only) (He et al., 2024) compared to 50.8% by GRPO. In agentic settings, with Qwen-3B base, EMA-PG improves GRPO by 33.3% across 7 datasets of Q&A with search engine, including 29.7% → 44.1% on HotpotQA (Yang et al., 2018), 27.4% → 40.1% on 2WikiMultiHopQA (Ho et al., 2020b), 12.8% → 33.1% on Bamboogle (Press et al., 2023).

The rest of the paper is organized as the following. **(1)** We derive the stability condition for using an EMA anchor (§4). **(2)** We explain why token-level KL is more suited for reasoning models than sequence-level KL (§5.1). **(3)** For token-level KL, common KL estimators such as K1, K2, K3 are unable to yield unbiased KL estimates and unbiased gradients at the same time, so we construct estimators K3^{++}, K4, and K5 that can (§5.2). **(4)** We propose Top-k KL estimators, which extract dense supervision from logits with low memory footprint (§5.3). Overall, EMA anchor and Top-k KL result in notable sample efficiency gain for RL.

2. Related Work

Improving RL for LLMs Leveraging verifiable domains such as math and coding, lots of effort has gone into scaling RL for reasoning (Jaech et al., 2024; Guo et al., 2025; Chen et al., 2025) and agentic tasks (Kimi et al., 2025a; Comanici et al., 2025). The predominant algorithm for RL in reasoning, GRPO (Shao et al., 2024), has gained widespread adoption due to its effectiveness. Many works have tried

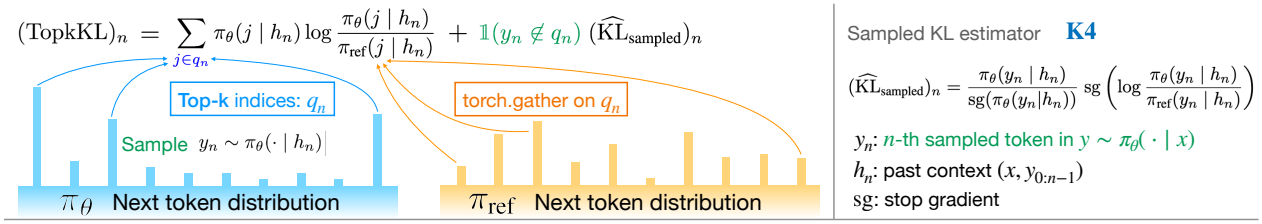

Figure 2. Top-k KL computes exact KL on the top-k indices, and adds a masked sampled KL to keep the estimator *unbiased*.

to further improve it, which include removing bias in policy gradient (Liu et al., 2025b), better importance weight clipping (Yu et al., 2025), using sequence-level objective (Zheng et al., 2025), asynchronous data generation (Piché et al., 2025), better scaling practices (Khatri et al., 2025). Since we exclusively focus on the KL component in RL, our work is largely orthogonal to those techniques above.

KL Estimators for LLMs KL estimation for LLMs is challenging due to autoregressive action space and large vocabulary size. Early work on RLHF adds a sequence-level KL penalty to the sequence-level reward (Ouyang et al., 2022; Bai et al., 2022a). As reasoning models begin to scale the sequence length of thinking traces, token-level KL regularization becomes popular (Guo et al., 2025). Token-level sampled KL estimators are typically K1, K2, and K3 (Schulman, 2020); K3 was used in GRPO. Since then, many works have pointed out that K3 gradients are forward KL gradients (Tang & Munos, 2025; Zhang et al., 2025; Shah et al., 2025). Sampled KL ignores the fact that we have white-box access to logits and next-token distributions; knowledge distillation has shown that such information is very useful to learning (Hinton et al., 2015; Kim & Rush, 2016; Agarwal et al., 2024; Zhou et al., 2024). When the vocabulary is large, storing full logits is too memory-intensive; our insight is to reduce the memory usage of logits via Top-k truncation.

EMA in Deep Learning Deep Q-learning almost universally uses an EMA network to produce stable learning targets (Mnih et al., 2015; Lillicrap et al., 2015; Haarnoja et al., 2018). Self-supervised learning has also used EMA for stable bootstrapping (He et al., 2020; Grill et al., 2020b; Busbridge et al., 2023). EMA has been shown to improve generalization in deep learning (Nichol et al., 2018; Zhang et al., 2019). In the diffusion model literature, EMA is often used to produce the final model (Ho et al., 2020a). For RL on LLMs, prior work has also suggested using an EMA anchor for alignment (Gorbatovski et al., 2024; Ramé et al., 2024); our work differs in how the KL is computed. Elastic reset (Noukhovitch et al., 2023) also keeps an EMA copy during RL, but does not use it for reference policy. The effectiveness of EMA can be partially explained by Linear Mode Connectivity (Frankle et al., 2020) and Monotonic Linear Interpolation (Goodfellow et al., 2014; Lucas et al., 2021) between neural networks that share the same initialization.

3. Background

Reinforcement Learning (RL) Given a pre-trained LLM, RL trains the model to maximize a reward function while remaining close to the original model. Concretely, let π_θ denote a policy (language model) parameterized by $\theta \in \mathbb{R}^d$, and let $R(x, y)$ be a reward function scoring response y to prompt x. The standard objective is to maximize:

$$\mathbb{E}_x\{\mathbb{E}_{\pi_\theta(y|x)}\left[R(x,y)\right] - \beta \text{KL}\big(\pi_\theta(\cdot \mid x) \,\|\, \pi_{\text{ref}}(\cdot \mid x)\big)\} \quad (1)$$

where $x \sim \mathcal{D}$ is a sampled prompt, π_{ref} is the reference policy (also called the **anchor** policy), β controls how much π_θ is allowed to deviate from the anchor. The term $\text{KL}(\pi_\theta \| \pi_{\text{ref}}) = \mathbb{E}_x \mathbb{E}_{y \sim \pi_\theta(\cdot|x)}[\log(\pi_\theta(y|x)/\pi_{\text{ref}}(y|x))]$ is the reverse KL divergence for the policy π_θ.

Exponential Moving Average (EMA) EMA is a technique for maintaining a smoothed version of an iteratively updated quantity. Given a sequence $\{\theta_t\}_{t=0}^{\infty}$ and coefficient $\eta \in (0, 1)$, the EMA sequence $\{\theta_{\text{ema},\,t}\}$ is:

$$\theta_{\text{ema},\,0} = \theta_0 \qquad \theta_{\text{ema},\,t+1} = \eta\, \theta_{\text{ema},\,t} + (1 - \eta)\, \theta_t \quad (2)$$

which is widely used in deep Q-learning as *target network*.

Local Quadratic Approximation of KL The Fisher information matrix of a policy π_θ is defined as:

$$F_\theta \triangleq \mathbb{E}_{x \sim \mathcal{D}} \mathbb{E}_{y \sim \pi_\theta(\cdot|x)}\left[(\nabla_\theta \log \pi_\theta(y|x))(\nabla_\theta \log \pi_\theta(y|x))^\top\right]$$

which captures the local sensitivity of a policy's output distribution to parameter changes. The natural gradient $F_\theta^{-1} g$ is the steepest-ascent direction under a local KL trust-region constraint. For $\theta' = \theta + \delta$ with $\|\delta\| \to 0$,

$$\text{KL}(\pi_\theta \| \pi_{\theta+\delta}) = \delta^\top F_\theta \delta / 2 + O(\|\delta\|^3), \quad (3)$$

and the gradient w.r.t θ is: $\nabla_\theta \text{KL}(\pi_\theta \| \pi_{\theta'}) \approx F_\theta(\theta - \theta')$.

4. Training Dynamics of the EMA Anchor

Consider the policy gradient $g_t = (R\, \nabla_\theta \log \pi_\theta)|_{\theta=\theta_t}$ at t:

$$\theta_{t+1} = \theta_t + \alpha[g_t - \beta\, \nabla_\theta \text{KL}(\pi_\theta || \pi_{\theta_{\text{ema},\,t}})|_{\theta=\theta_t}]$$
$$\theta_{\text{ema},\,t+1} = \eta\, \theta_{\text{ema},\,t} + (1 - \eta)\, \theta_t$$

We define the *lag* between current and EMA weights as:

$$\delta_t := \theta_t - \theta_{\text{ema},\,t} \quad (4)$$

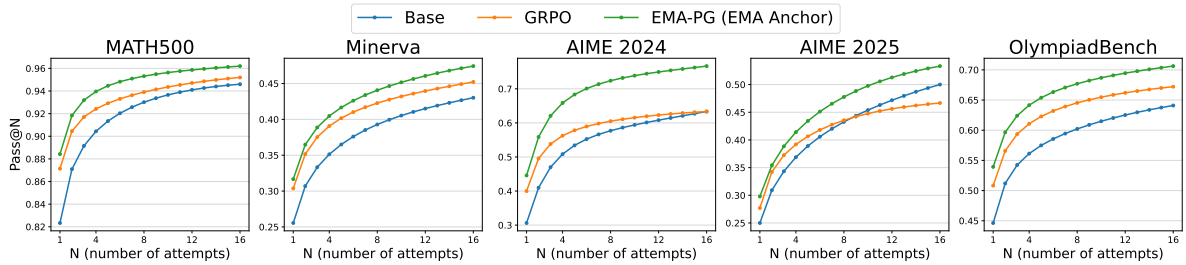

Figure 3. **On math reasoning datasets, using EMA Anchor improves not only Pass@1 but Pass@N as well compared to GRPO**. Results obtained on a 1.5B model DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025).

Using the local approximation (3), with $F_t = F_{\theta_t}$

$$\theta_{t+1} = \theta_t + \alpha g_t - \alpha\beta F_t \delta_t \tag{5}$$

The joint dynamics of θ_t and δ_t becomes:

$$\begin{bmatrix}\theta_{t+1}\\ \delta_{t+1}\end{bmatrix} = \begin{bmatrix} I & -\alpha\beta F_t \\ 0 & \eta I - \alpha\beta F_t\end{bmatrix}\begin{bmatrix}\theta_t\\ \delta_t\end{bmatrix} + \begin{bmatrix}\alpha I\\ \alpha I\end{bmatrix} g_t \tag{6}$$

Lemma 4.1. *Quasi-steady state: If we assume that the gradient and the Fisher matrix change extremely slowly over k steps, meaning that for $\Delta = 0 \cdots k-1$:*

$$g_{t+\Delta} \approx g, \qquad F_{t+\Delta} \approx F \tag{7}$$

then, under the following definitions of matrices D, S_k, M_k:

$$D = \eta I - \alpha\beta F \quad S_k = \sum\nolimits_{\Delta=0}^{k-1} D^{\Delta} \quad M_k = \sum\nolimits_{\Delta=0}^{k-1} S_{\Delta}$$

we have: firstly, $I - D$ is invertible and $S_k = (I - D^k)(I - D)^{-1}$, $M_k = (kI - S_k)(I - D)^{-1}$; secondly, we have a closed-form dynamics of EMA Policy Gradient:

$$\begin{bmatrix}\theta_{t+k}\\ \delta_{t+k}\\ g\end{bmatrix} = \begin{bmatrix} I & -\alpha\beta F S_k & \alpha k I - \alpha^2\beta F M_k \\ 0 & D^k & \alpha S_k \\ 0 & 0 & I\end{bmatrix}\begin{bmatrix}\theta_t\\ \delta_t\\ g\end{bmatrix}$$

Lemma 4.2. *Stability Condition: Let $\{v_i\}_{i=0}^{d-1}$ be the eigenbasis of the Fisher matrix F, and $\{\lambda_i\}_{i=0}^{d-1}$ be its corresponding eigenvalues (with $\lambda_{\max}$ being the largest eigenvalue). The dynamics of the lag δ_t in the eigenbasis of the Fisher, where $\delta_t^{(i)} := \delta_t{}^{\top} v_i$ and $g^{(i)} := g^{\top} v_i$, can be written as:*

$$\delta_{t+k}^{(i)} = \underbrace{(\chi_i)^k \delta_t^{(i)}}_{\textit{transient}} + \underbrace{\alpha \tfrac{1-(\chi_i)^k}{1-\chi_i} g^{(i)}}_{\textit{particular solution}} \qquad \chi_i = \eta - \alpha\beta\lambda_i \tag{8}$$

Thus, the stability condition of δ is:

$$\boxed{\alpha\beta\lambda_{\max} < 1 + \eta} \tag{9}$$

Summary of Stability / Oscillation Conditions:

1. $0 \le \alpha\beta\lambda_{\max} \le \eta$: stable, non-oscillatory.
2. $\eta < \alpha\beta\lambda_{\max} < 1 + \eta$: stable, oscillatory.
3. $\alpha\beta\lambda_{\max} \ge 1 + \eta$: unstable.

Which characterizes the exact conditions for stability in EMA-PG's training dynamics under mild assumptions (7).

5. KL Estimators: Which and Why

5.1. Sequence-level KL or Token-level KL?

Because LLMs have an autoregressive action space, it is worth discussing whether we should apply sequence-level KL or token-level KL. For a language model, the KL component in (1) can be rewritten as the loss:

$$\widehat{\mathrm{KL}}_{\text{seq}} \triangleq -\log(\pi_\theta(y|x)/\pi_{\text{ref}}(y|x)) \tag{10}$$

$$\nabla_\theta L_{\text{KL}}^{\text{seq}} \triangleq \mathbb{E}_{\pi_\theta(y|x)}[-\text{sg}(\widehat{\mathrm{KL}}_{\text{seq}})\nabla_\theta \log \pi_\theta(y|x)] \tag{11}$$

where $\text{sg}(\cdot)$ denotes the stop gradient operator. Note that this is an unbiased **sequence-level KL** estimator:

$$\nabla_\theta \mathrm{KL}(\pi_\theta(\cdot \mid x) || \pi_{\text{ref}}(\cdot \mid x)) = \nabla_\theta L_{\text{KL}}^{\text{seq}}(\theta) \tag{12}$$

This KL regularizer was used in both for InstructGPT (Ouyang et al., 2022) and WARP (Ramé et al., 2024). It is simple to implement: just adding a scalar sequence logprob difference to the scalar sequence-level reward would suffice. DPO (Rafailov et al., 2023) also uses sequence-level KL.

In contrast, **token-level KL** regularization, as popularized by GRPO (Shao et al., 2024; Guo et al., 2025; Liu et al., 2025a), implements the following:

$$\widehat{\mathrm{KL}}_n \triangleq \mathrm{KL}(\pi_\theta(\cdot \mid x, y_{0:n-1}) \| \pi_{\text{ref}}(\cdot \mid x, y_{0:n-1})) \tag{13}$$

$$\nabla_\theta L_{\text{KL}}^{\text{token}} \triangleq \mathbb{E}_{\pi_\theta(y|x)}\Big[\sum\nolimits_{n=0}^{L-1} \nabla_\theta \widehat{\mathrm{KL}}_n\Big] \tag{14}$$

where the KL estimator is a design choice (See Tab 1). We assume the use of exact KL rather than sampled KL in our analysis of token-level KL. Token-level KL was traditionally seen as a surrogate objective for sequence-level KL; here we analyze their differences. Rewrite sequence-level KL as:

$$\rho_n \triangleq \log(\pi_\theta(y_n|x, y_{0:n-1})/\pi_{\text{ref}}(y_n|x, y_{0:n-1}))$$

$$\omega_n \triangleq \nabla_\theta \log \pi_\theta(y_n|x, y_{0:n-1})$$

It can be shown that (§D):

$$\nabla_\theta L_{\text{KL}}^{\text{seq}} = \mathbb{E}_{\pi_\theta(y|x)}\Big[\sum_n \Big(\sum_{m \ge n} \rho_m\Big)\omega_n\Big] \tag{15}$$

$$\nabla_\theta L_{\text{KL}}^{\text{token}} = \mathbb{E}_{\pi_\theta(y|x)}\Big[\sum_n (\rho_n)\,\omega_n\Big] \tag{16}$$

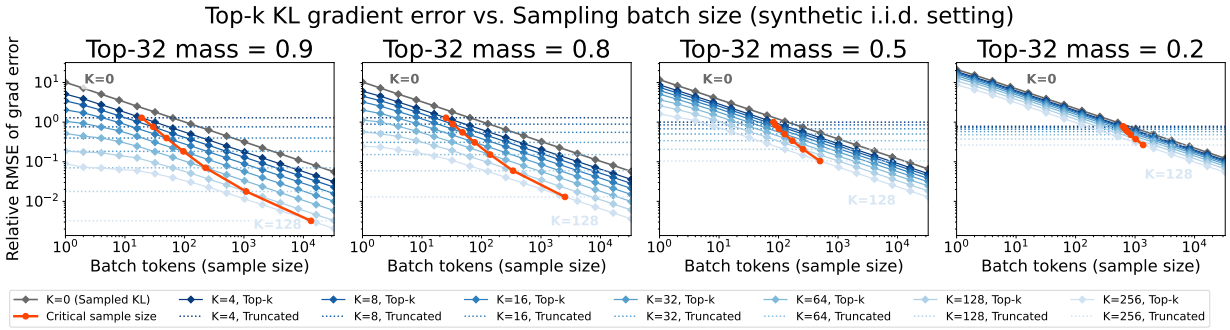

Figure 4. **Bias-variance tradeoff in Top-k KL estimator:** Top-k KL (unbiased) has lower gradient error than sampled KL (unbiased) in all regimes, but only outperforms Truncated KL (biased) beyond a certain *critical sample size*; we illustrate this bias-variance tradeoff in a synthetic setting (§G). Rule of thumb: when using a small k, apply the tail correction (25) (default); for a large k, consider Truncated KL.

The difference is clear: **sequence-level KL punishes each current token for the divergence of all its future tokens until the end of sequence**, while token-level KL does not.

We find that in reasoning and agentic settings, since the sequence lengths of thinking traces are long and the rewards are reliable, sequence-level KL does not help; **token-level KL should be used instead**. In other words, during RL training for reasoning, KL should be token-wise in the loss, not added to the outcome reward, even though sequence-level KL may seem more "correct" (12) for objective (1).

5.2. Exact KL or Sampled KL?

For token-level KL estimation, one can technically implement **exact KL**, which is unbiased and low-variance, but at the cost of memory. We use following notations:

$$h_n \triangleq (x, y_{0:n-1}), \quad \mathbb{E}_n[\cdot] \triangleq \mathbb{E}_{y_n \sim \pi_\theta(\cdot|h_n)}[\cdot] \tag{17}$$

In the case of reverse KL, for each token $y_n \sim \pi_\theta(\cdot \mid h_n)$:

$$\mathrm{KL}_n \triangleq \sum\nolimits_{j \in \mathcal{V}} \pi_\theta(j \mid h_n) \log \frac{\pi_\theta(j \mid h_n)}{\pi_{\text{ref}}(j \mid h_n)} \tag{18}$$

Since this is the ground-truth definition of reverse KL, it naturally yields exact values and exact gradients. The problem with exact KL is that it requires storing full logits for every position in each sequence both for current policy π_θ and reference policy π_{ref}, costing a memory of $O(|\mathcal{V}|)$ per token. The alternative, which has negligible memory footprint $O(1)$, is **sampled KL**. For the n-th position in y, let

$$y_n \sim \pi_\theta(\cdot \mid h_n) \quad w_n \triangleq \pi_{\text{ref}}(y_n \mid h_n)/\pi_\theta(y_n \mid h_n) \tag{19}$$

The K1, K2, K3 estimators (Schulman, 2020) are given by:

$$(\mathrm{K1})_n \triangleq -\log w_n \quad (\mathrm{K2})_n \triangleq (1/2)(\log w_n)^2$$
$$(\mathrm{K3})_n \triangleq -\log w_n + w_n - 1$$

None of the existing sampled KL estimators simultaneously gives <u>both</u> unbiased value and unbiased gradient on the token level. Ideally, we would like an KL estimator $\widehat{\mathrm{KL}}_n$ to suffice both conditions (with notations from (17)):

$$\mathbb{E}_n[\widehat{\mathrm{KL}}_n] = \mathrm{KL}_n \quad \mathbb{E}_n[\nabla_\theta \widehat{\mathrm{KL}}_n] = \nabla_\theta \mathrm{KL}_n \tag{20}$$

K1 and K3 give the correct values but wrong gradients; K2 give the correct gradients but wrong values (see Tab 1). We show that K1 and K3 can be fixed with a simple trick. Let

$$r_n \triangleq \pi_\theta(y_n \mid h_n)/\mathrm{sg}(\pi_\theta(y_n \mid h_n)) \tag{21}$$

where sg means stop gradient. We construct the following unbiased sampled KL estimators to satisfy (20):

$$\boxed{(\mathbf{K3}^{++})_n \triangleq r_n(-\log w_n + w_n - 1)} \tag{22}$$

$$\boxed{(\mathbf{K4})_n \triangleq r_n \cdot \mathrm{sg}(-\log w_n)} \tag{23}$$

Eq (22) is named **K3**$^{++}$ because of its resemblance to the modified K3 in (Liu et al., 2025a), but here we disentangle generic off-policy correction (which can be applied to any sampled KL estimator, not just K3; moreover, importance weights should be applied with $\mathrm{sg}(\cdot)$ on themselves) from an actual change in KL direction:

- **K3**$^{++}$ is obtained by multiplying existing K3 by r_n in (21). **This simple change alone converts the gradient from forward KL to reverse KL.**
- **K4** is obtained by first doing stop gradient on K1 and then multiplying it by r_n in (21). It is simpler, and it is unbiased as well.

K3^{++} and **K4** satisfy the conditions in (20): they yield <u>both</u> unbiased value and unbiased gradient for reverse KL.

In addition, the success of K3's forward KL gradients raises the interesting question of whether forward KL is just as effective as reverse KL in regularizing RL training dynamics. We propose a **forward KL estimator K5**, which produces unbiased forward KL values and unbiased forward KL gradients under $\mathbb{E}_n[\cdot]$ (17) when sampling from (19):

$$\boxed{(\mathbf{K5})_n \triangleq \mathrm{sg}(w_n) \log w_n + \log r_n} \tag{24}$$

When sampling is off-policy ($\pi_{\text{sampling}} \neq \pi_\theta$), any sampled KL estimator $\widehat{\mathrm{KL}}_n$ above simply needs to multiply the (stop-gradient) importance weight $\mathrm{sg}(\frac{\pi_\theta}{\pi_{\text{sampling}}})$ to remain unbiased.

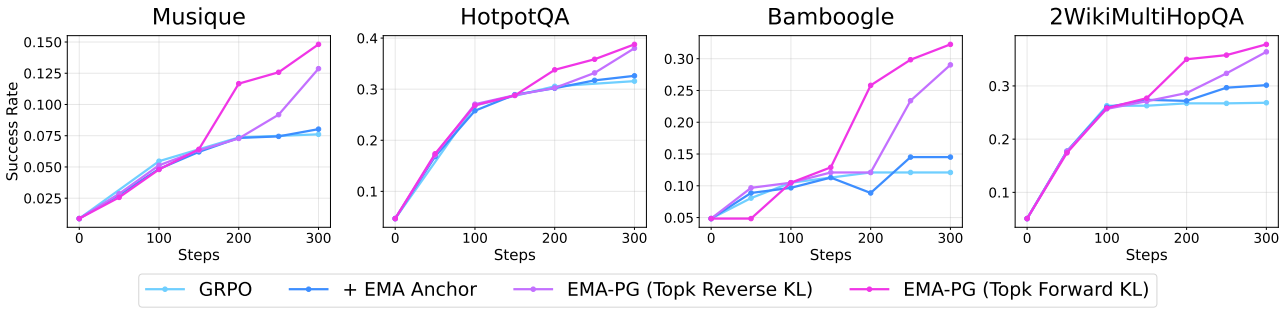

Figure 5. **EMA-PG not only learns faster and but reaches higher asymptotic performance** on agentic tasks of Q&A with search engine. Using Topk KL Estimator, forward KL learns slightly faster than reverse KL. Both forward and reverse Top-k KL significantly outperforms GRPO (and GRPO with EMA Anchor). This shows that the combination of EMA Anchor and Top-k KL is highly effective.

Concretely, when $y_n \sim \pi_{\text{sampling}}$, any previously unbiased $\widehat{\text{KL}}_n$ under $y_n \sim \pi_\theta$ will require an off-policy correction $\text{KL}_n = \mathbb{E}_{y_n}[s_n \widehat{\text{KL}}_n]$ with importance weight (IW) s_n:

$$y_n \sim \pi_{\text{sampling}}(\cdot|h_n) \quad s_n \triangleq \text{sg}(\pi_\theta(y_n|h_n)/\pi_{\text{sampling}}(y_n|h_n))$$

On the other hand, exact KL is always on-policy (but memory-intensive, with $O(|\mathcal{V}|)$ rather than $O(1)$ memory footprint), and thus does not require off-policy correction.

5.3. Top-k KL Estimator

We now aim to combine the best of both worlds of exact KL and sampled KL. We start from the observation that for modern LLMs, the probability mass tends to concentrate on a small number of top tokens at each generation step; it was for this reason that top-k sampling (Fan et al., 2018) and nucleus sampling (Holtzman et al., 2019) were effective. The vocabulary size for modern LMs is very large (typically 100k-200k), so most of the logit indices in $\mathcal{V}$ contribute very minimally to the KL value or KL gradients.

Our key insight is to **only partially compute exact KL on the top-k logit indices** of the policy under which the expectation is taken (which is current policy for reverse KL, reference policy for forward KL), and then **use a partially masked sampled KL to correct for the tail of the distribution**, so that we arrive at an unbiased estimator for KL values and gradients, with lower variance than sampled KL.

First, we consider Reverse KL. Let q_n be the set of top-k logit indices from $\pi_\theta(\cdot \mid h_n)$ at sequence position n,

$$q_n \leftarrow \text{Topk}(\pi_\theta(\cdot \mid h_n)) \qquad q_n \subseteq \mathcal{V} \qquad |q_n| = k$$

we can rewrite reverse KL by partitioning the vocabulary:

$$\underbrace{\sum_{j \in q_n} \pi_\theta(j|h_n) \log \frac{\pi_\theta(j|h_n)}{\pi_{\text{ref}}(j|h_n)}}_{\text{'Truncated' KL: } (\widehat{\text{KL}}_{\text{trun}})_n} + \underbrace{\sum_{j \notin q_n} \pi_\theta(j|h_n) \log \frac{\pi_\theta(j|h_n)}{\pi_{\text{ref}}(j|h_n)}}_{\text{'Tail' correction: } (\widehat{\text{KL}}_{\text{tail}})_n}$$

The second term can be estimated using sampled KL:

$$(\widehat{\text{KL}}_{\text{tail}})_n = \begin{cases} 0 & \text{if } y_n \in q_n \\ (\mathbf{K4})_n & \text{if } y_n \notin q_n \end{cases} \qquad y_n \sim \pi_\theta(\cdot|h_n) \quad (25)$$

The **Top-k reverse KL** estimator is an addition of the two:

$$\boxed{(\text{Topk}\mathbf{Reverse}\text{KL})_n = (\widehat{\text{KL}}_{\text{trun}})_n + (\widehat{\text{KL}}_{\text{tail}})_n} \quad (26)$$

Similarly, we can estimate forward KL by splitting it into a truncated exact KL and a sampled tail term:

$$q_n \leftarrow \text{Topk}(\pi_{\text{ref}}(\cdot \mid h_n))$$

$$(\widehat{\text{FKL}}_{\text{trun}})_n = \sum\nolimits_{j \in q_n} \pi_{\text{ref}}(j \mid h_n) \log \frac{\pi_{\text{ref}}(j \mid h_n)}{\pi_\theta(j \mid h_n)}$$

$$(\widehat{\text{FKL}}_{\text{tail}})_n = \begin{cases} 0 & \text{if } y_n \in q_n \\ (\mathbf{K5})_n & \text{if } y_n \notin q_n \end{cases} \qquad y_n \sim \pi_\theta(\cdot \mid h_n)$$

The **Top-k forward KL** estimator is therefore:

$$\boxed{(\text{Topk}\mathbf{Forward}\text{KL})_n = (\widehat{\text{FKL}}_{\text{trun}})_n + (\widehat{\text{FKL}}_{\text{tail}})_n} \quad (27)$$

Top-k KL estimators are unbiased regardless of whether the chosen logit indices are top-k or not. Moreover, Top-k KL renders both exact KL and sampled KL as special cases. When $k = |\mathcal{V}|$, it is exact KL; when $k = 0$, it is sampled KL. Top-k KL allows for flexible interpolation between the two, remains unbiased at any k, and only requires $O(k)$ memory. We generalize our estimator to any f-divergence in §H.

6. Experiments

We evaluate the efficacy of EMA Policy Gradients in reasoning RL and agentic RL. The base algorithm we use is GRPO (Shao et al., 2024) with clip-higher (Yu et al., 2025), a popular default for LLMs. To show that the two techniques we have introduced (EMA anchor and Topk KL) can benefit the base RL algorithm, we focus on the following tasks:

Math Reasoning. We follow the DeepScaleR setting (Luo et al., 2025) and train DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025) on an RL dataset of 40k question-answer pairs that comprise of: AIME problems from 1984-2023, Omni-MATH dataset (Gao et al., 2024), AMC problems prior to 2023, Still datasets (Jiang et al., 2024; Min et al., 2024). We then evaluate on the datasets of: MATH500 (Hendrycks et al., 2021), Minerva (Lewkowycz et al., 2022), AIME 2024 / 2025, and OlympiadBench (He et al., 2024).

Method	General QA			Multi-Hop QA				
	NQ	TriviaQA	PopQA	HotpotQA	2wiki	Musique	Bamboogle	Avg.
Qwen2.5-**3B-Instruct** (Prompting)								
CoT (no search) (Wei et al., 2022)	0.106	0.288	0.108	0.149	0.244	0.020	0.024	0.134
Search-o1 (Li et al., 2025)	0.238	0.472	0.262	0.221	0.218	0.054	0.320	0.255
RAG (Lewis et al., 2020)	0.348	0.544	0.387	0.255	0.226	0.047	0.080	0.270
Qwen2.5-**7B-Instruct** (Prompting)								
RAG	0.349	0.585	0.392	0.299	0.235	0.058	0.208	0.304
Qwen2.5-**3B-Base** (RL)								
before RL	0.060	0.122	0.076	0.047	0.051	0.009	0.048	0.059
WARP	0.114	0.197	0.116	0.103	0.140	0.014	0.056	0.106
PPO	0.406	0.587	0.435	0.284	0.273	0.049	0.088	0.303
GRPO	0.421	0.583	0.413	0.297	0.274	0.066	0.128	0.312
EMA-PG: *w/* EMA Anchor	0.468	0.624	0.445	0.345	0.277	0.084	0.194	0.348
+ Topk Forward KL	0.469	0.614	0.443	0.425	0.385	0.146	**0.363**	0.406
+ Topk Reverse KL	**0.479**	**0.631**	**0.467**	**0.441**	**0.401**	**0.163**	0.331	**0.416**
Improvement over GRPO	13.8% ↑	8.2% ↑	13.1% ↑	48.5% ↑	46.3% ↑	147.0% ↑	158.6% ↑	33.3% ↑

Table 2. **EMA-PG excels at agentic RL tasks that require using a search engine inside CoTs to reason about questions.**

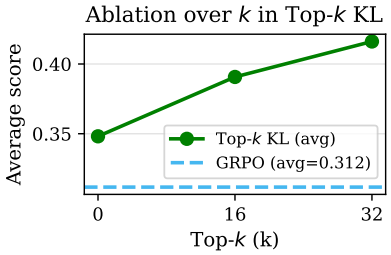

Figure 6. Ablation on k of Top-k (reverse) KL in EMA-PG.

Agentic RL for Search. We follow the Search-R1 setting (Jin et al., 2025) and train a Qwen-2.5 3B (Yang et al., 2024) to use the search engine with RL. GRPO being used here applies the RL loss only on CoT and action tokens, not on environment observation tokens that interleave with CoTs. RL uses the combined training sets of Natural Questions (NQ) (Kwiatkowski et al., 2019) and HotpotQA (Yang et al., 2018). After RL, we run the eval on both General QA and Multi-Hop QA. For General QA, we use the test set of NQ, TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023). For Multi-Hop QA, we use the test set of HotpotQA, 2WikiMultiHopQA (2wiki) (Ho et al., 2020b), Musique (Trivedi et al., 2022), and Bamboogle (Press et al., 2023).

Effect of EMA anchor We empirically demonstrate the benefits of using an EMA target network instead of a fixed anchor for RL on LLMs. Note that using EMA does not require storing an additional set of weights during training, since the anchor (reference) policy weights already need to be stored for standard KL-regularized policy gradients.

First of all, we empirically find that keeping the KL with a small $\beta > 0$ is necessary for preventing training collapse when doing full-finetuning (no LoRA). This is corroborated by with the common practice of still using KL in large-scale RL training runs (Liu et al., 2025a; Kimi et al., 2025a).

Secondly, we find that in math reasoning, using an EMA anchor ($\eta < 1$) improves GRPO across all evaluated datasets, as shown in Tab 3. For instance, using the EMA anchor gives us a boost of 50.8% $\rightarrow$ 53.9% on OlympiadBench. In addition, using an EMA anchor improves not just Pass@1 but Pass@16 as well, as shown in Fig 3; this suggests that target networks can effectively mitigate Pass@N collapse that is common in reasoning RL (Yue et al., 2025).

Using token-level KL instead of sequence-level KL is crucial for good performance. On agentic search, we compare against WARP (Ramé et al., 2024), which uses sequence-level KL (12) and EMA anchor; we find that WARP performs a lot worse than vanilla GRPO (10.6% vs 31.2%). When using token-level KL + EMA anchor, we see a gain from 31.2% $\rightarrow$ 34.8%. This validates our analysis in §5.1.

Effect of Top-k KL We find that the use of Top-k KL can make a substantial difference in RL performance; both Topk Forward KL (Alg 2) and Topk Reverse KL (Alg 1) simultaneously improve sample efficiency and asymptotic performance of RL. It is shown in Fig 5 that, compared to sampled KL, Top-k KL enables the learning process to rapidly take off and avoid premature convergence. This is because Top-k KL works like knowledge distillation, which uses logit-level information to provide dense supervision and lower gradient variance. In terms of the final performance (Tab 2), when used with EMA anchor, Top-k forward KL obtains a 34.8% $\rightarrow$ 40.6% gain compared to K3; Top-k reverse KL performs even better, achieving 41.6%.

The benefits of Top-k KL only appear when $k > 0$, but to

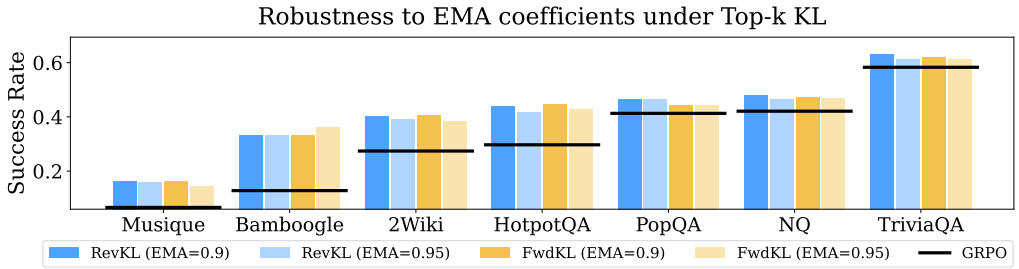

Figure 7. **When Top-k KL is used, EMA-PG is robust to various choices of EMA coefficient and KL direction**. Here we show that EMA-PG can work well under many choices of KL and EMA η (with $k = 32$). We recommend setting η to be in the range of [0.9, 0.95].

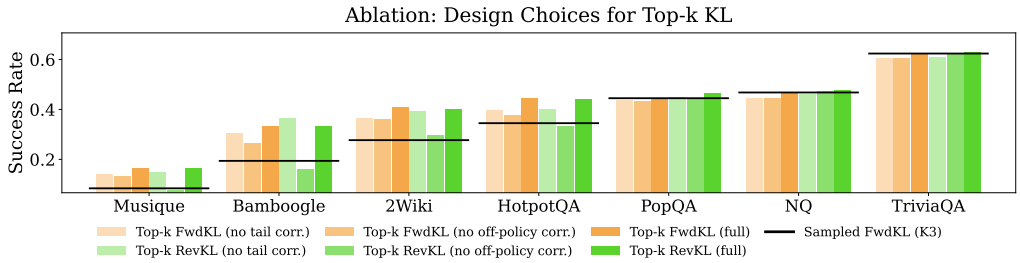

Figure 8. **Top-k KL performs best when tail correction and off-policy correction are jointly applied**. The truncated KL part is always on-policy, and truncated KL alone already performs well; the tail term can either be dropped, or require off-policy correction (especially for reverse KL). Forward KL is not as sensitive to the absence of off-policy correction as reverse KL.

R1-Distill-Qwen-1.5B	**Base**	**GRPO**	**EMA-PG**	Δ (Rel)
MATH500	0.823	0.871	**0.884**	27% ↑
AMC 2024	0.499	0.656	**0.686**	19% ↑
Minerva	0.256	0.304	**0.317**	27% ↑
AIME 2024	0.306	0.400	**0.446**	49% ↑
AIME 2025	0.250	0.277	**0.298**	77% ↑
OlympiadBench (text)	0.446	0.508	**0.539**	50% ↑
Average	0.430	0.503	**0.528**	34% ↑

Table 3. Pass@1 comparison across six math reasoning datasets. Relative Improvement = (EMA-PG − GRPO)/(GRPO − Base).

save memory we need to keep k small. In general, the larger the k the better (with the limit being exact KL), but when k is large, the tail correction term ($\widehat{\mathrm{KL}}_{\mathrm{tail}}$ and $\widehat{\mathrm{FKL}}_{\mathrm{tail}}$) does not necessarily help, meaning that truncated KL alone might perform better. We illustrate this *bias-variance tradeoff* in a synthetic setting in Fig 4. For RL at scale, we empirically validate that increasing k from $0 \to 16 \to 32$ does lead to improved performance (Fig 6). Moreover, Fig 8 shows that, when tail correction and off-policy correction are jointly applied, Top-k KL outperforms truncated KL across datasets. This suggests that in realistic LLM settings, our token-level KL estimator is typically past the critical sample size as illustrated in Fig 4, and we should use unbiased Top-k KL.

We also study what EMA coefficients are effective when using Top-k KL. In Fig 7, we find that $\eta = 0.9$ or $\eta = 0.95$ for EMA are safe choices along with $k = 32$ for Top-k KL.

Other design choices We now study two additional design choices in EMA-PG. **(1)** Firstly, since Fig 8 has found off-policy correction to be important especially for reverse KL, we study the clipping range for the importance weight (IW), which is $[s_{\min}, s_{\max}]$ in Alg 1 and 2. Prior literature has mixed views: (Guo et al., 2025) uses no off-policy correction ($s_{\min} = s_{\max} = 1$); (Liu et al., 2025a) uses no clipping at all; (Zhang et al., 2025) uses aggressive clipping. Fig 9 finds that forward KL benefits from a tighter clipping range close to 1, but reverse KL benefits from more lenient clipping. This observation is consistent with the prevailing common practice that either uses forward KL (K3) without any off-policy correction or uses reverse KL with no clipping on IW. **(2)** Secondly, we also study whether the advantage normalization strategy inside the RL algorithm makes a difference in the presence of Top-k KL. The two choices are either group-wise normalization as done in GRPO or global advantage normalization (Hu et al., 2025) as done in the original PPO. We find that either choice did not make a noticeable difference to EMA-PG performance.

7. Conclusion

In this work, we introduced two simple changes to policy gradients on LLMs: EMA anchor, which changes what KL is regularizing against, and Top-k KL, which changes how KL gradients are computed. We showed that the two changes are highly effective. Future work includes further investigation into Top-k f-divergence estimators (§H.2) beyond just KL, the application of Top-k KL to both offline and on-policy knowledge distillation of LLMs, and the application of our sampled KL (and more broadly, f-divergence) estimators to the policy gradient operator itself (§J– §K).

References

Abdolmaleki, A., Springenberg, J. T., Tassa, Y., Munos, R., Heess, N., and Riedmiller, M. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*, 2018.

Agarwal, R., Vieillard, N., Zhou, Y., Stanczyk, P., Garea, S. R., Geist, M., and Bachem, O. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*, 2024.

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

Busbridge, D., Ramapuram, J., Ablin, P., Likhomanenko, T., Dhekane, E. G., Suau Cuadros, X., and Webb, R. How to scale your ema. *Advances in Neural Information Processing Systems*, 36:73122–73174, 2023.

Chen, A., Li, A., Gong, B., Jiang, B., Fei, B., Yang, B., Shan, B., Yu, C., Wang, C., Zhu, C., et al. Minimax-m1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*, 2025.

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Comanici, G., Bieber, E., Schaekermann, M., Pasupat, I., Sachdeva, N., Dhillon, I., Blistein, M., Ram, O., Zhang, D., Rosen, E., et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Fan, A., Lewis, M., and Dauphin, Y. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

Frankle, J., Dziugaite, G. K., Roy, D., and Carbin, M. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*, pp. 3259–3269. PMLR, 2020.

Gao, B., Song, F., Yang, Z., Cai, Z., Miao, Y., Dong, Q., Li, L., Ma, C., Chen, L., Xu, R., et al. Omni-math: A universal olympiad level mathematic benchmark for large language models. *arXiv preprint arXiv:2410.07985*, 2024.

Gao, L., Schulman, J., and Hilton, J. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pp. 10835–10866. PMLR, 2023.

Ghasemipour, S. K. S., Zemel, R., and Gu, S. A divergence minimization perspective on imitation learning methods. In *Conference on robot learning*, pp. 1259–1277. PMLR, 2020.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Goodfellow, I. J., Vinyals, O., and Saxe, A. M. Qualitatively characterizing neural network optimization problems. *arXiv preprint arXiv:1412.6544*, 2014.

Gorbatovski, A., Shaposhnikov, B., Malakhov, A., Surnachev, N., Aksenov, Y., Maksimov, I., Balagansky, N., and Gavrilov, D. Learn your reference model for real good alignment. *arXiv preprint arXiv:2404.09656*, 2024.

Grill, J.-B., Altché, F., Tang, Y., Hubert, T., Valko, M., Antonoglou, I., and Munos, R. Monte-carlo tree search as regularized policy optimization. In *International Conference on Machine Learning*, pp. 3769–3778. PMLR, 2020a.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020b.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

He, C., Luo, R., Bai, Y., Hu, S., Thai, Z., Shen, J., Hu, J., Han, X., Huang, Y., Zhang, Y., et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on*

computer vision and pattern recognition, pp. 9729–9738, 2020.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Ho, J. and Ermon, S. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020a.

Ho, X., Nguyen, A.-K. D., Sugawara, S., and Aizawa, A. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*, 2020b.

Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Hu, J., Liu, J. K., and Shen, W. Reinforce++: An efficient rlhf algorithm with robustness to both prompt and reward models. *arXiv preprint arXiv:2501.03262*, 2025.

Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Jiang, J., Chen, Z., Min, Y., Chen, J., Cheng, X., Wang, J., Tang, Y., Sun, H., Deng, J., Zhao, W. X., Liu, Z., Yan, D., Xie, J., Wang, Z., and Wen, J.-R. Enhancing llm reasoning with reward-guided tree search. *arXiv preprint arXiv:2411.11694*, 2024.

Jin, B., Zeng, H., Yue, Z., Yoon, J., Arik, S., Wang, D., Zamani, H., and Han, J. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Joshi, M., Choi, E., Weld, D. S., and Zettlemoyer, L. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

Ke, L., Choudhury, S., Barnes, M., Sun, W., Lee, G., and Srinivasa, S. Imitation learning as f-divergence minimization. In *International workshop on the algorithmic foundations of robotics*, pp. 313–329. Springer, 2020.

Khatri, D., Madaan, L., Tiwari, R., Bansal, R., Duvvuri, S. S., Zaheer, M., Dhillon, I. S., Brandfonbrener, D., and Agarwal, R. The art of scaling reinforcement learning compute for llms. *arXiv preprint arXiv:2510.13786*, 2025.

Kim, Y. and Rush, A. M. Sequence-level knowledge distillation. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 1317–1327, 2016.

Kimi, Bai, Y., Bao, Y., Chen, G., Chen, J., Chen, N., Chen, R., Chen, Y., Chen, Y., Chen, Y., et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025a.

Kimi, Du, A., Gao, B., Xing, B., Jiang, C., Chen, C., Li, C., Xiao, C., Du, C., Liao, C., et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025b.

Kingma, D. P. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A., Anil, C., Schlag, I., Gutman-Solo, T., et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Li, X., Dong, G., Jin, J., Zhang, Y., Zhou, Y., Zhu, Y., Zhang, P., and Dou, Z. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*, 2025.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Liu, A., Mei, A., Lin, B., Xue, B., Wang, B., Xu, B., Wu, B., Zhang, B., Lin, C., Dong, C., et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025a.

Liu, Z., Chen, C., Li, W., Qi, P., Pang, T., Du, C., Lee, W. S., and Lin, M. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025b.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Lucas, J., Bae, J., Zhang, M. R., Fort, S., Zemel, R., and Grosse, R. Analyzing monotonic linear interpolation in neural network loss landscapes. *arXiv preprint arXiv:2104.11044*, 1, 2021.

Luo, M., Tan, S., Wong, J., Shi, X., Tang, W. Y., Roongta, M., Cai, C., Luo, J., Zhang, T., Li, L. E., et al. Deepscaler: Surpassing o1-preview with a 1.5 b model by scaling rl. *Notion Blog*, 2025.

Mallen, A., Asai, A., Zhong, V., Das, R., Khashabi, D., and Hajishirzi, H. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9802–9822, 2023.

Min, Y., Chen, Z., Jiang, J., Chen, J., Deng, J., Hu, Y., Tang, Y., Wang, J., Cheng, X., Song, H., Zhao, W. X., Liu, Z., Wang, Z., and Wen, J.-R. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems. *arXiv preprint arXiv:2412.09413*, 2024.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Nichol, A., Achiam, J., and Schulman, J. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Noukhovitch, M., Lavoie, S., Strub, F., and Courville, A. C. Language model alignment with elastic reset. *Advances in Neural Information Processing Systems*, 36:3439–3461, 2023.

Nowozin, S., Cseke, B., and Tomioka, R. f-gan: Training generative neural samplers using variational divergence minimization. *Advances in neural information processing systems*, 29, 2016.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Peters, J. and Schaal, S. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th international conference on Machine learning*, pp. 745–750, 2007.

Peters, J., Mulling, K., and Altun, Y. Relative entropy policy search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, pp. 1607–1612, 2010.

Piché, A., Kamalloo, E., Pardinas, R., Chen, X., and Bahdanau, D. Pipelinerl: Faster on-policy reinforcement learning for long sequence generation. *arXiv preprint arXiv:2509.19128*, 2025.

Press, O., Zhang, M., Min, S., Schmidt, L., Smith, N. A., and Lewis, M. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, 2023.

Rafailov, R., Sharma, A., Mitchell, E., Manning, C. D., Ermon, S., and Finn, C. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36: 53728–53741, 2023.

Ramé, A., Ferret, J., Vieillard, N., Dadashi, R., Hussenot, L., Cedoz, P.-L., Sessa, P. G., Girgin, S., Douillard, A., and Bachem, O. Warp: On the benefits of weight averaged rewarded policies. *arXiv preprint arXiv:2406.16768*, 2024.

Schulman, J. Approximating kl divergence. *John Schulman's Homepage*, 2020.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

Schulman, J., Chen, X., and Abbeel, P. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*, 2017a.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017b.

Shah, V., Obando-Ceron, J., Jain, V., Bartoldson, B., Kailkhura, B., Mittal, S., Berseth, G., Castro, P. S., Bengio, Y., Malkin, N., et al. A comedy of estimators: On kl regularization in rl training of llms. *arXiv preprint arXiv:2512.21852*, 2025.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Tang, Y. and Munos, R. On a few pitfalls in kl divergence gradient estimation for rl. *arXiv preprint arXiv:2506.09477*, 2025.

Trivedi, H., Balasubramanian, N., Khot, T., and Sabharwal, A. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Wang, C., Jiang, Y., Yang, C., Liu, H., and Chen, Y. Beyond reverse kl: Generalizing direct preference optimization with diverse divergence constraints. *arXiv preprint arXiv:2309.16240*, 2023.

Wang, Y., He, H., and Tan, X. Truly proximal policy optimization. In *Uncertainty in artificial intelligence*, pp. 113–122. PMLR, 2020.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W., Salakhutdinov, R., and Manning, C. D. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Dai, W., Fan, T., Liu, G., Liu, L., et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yue, Y., Chen, Z., Lu, R., Zhao, A., Wang, Z., Song, S., and Huang, G. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

Zhang, L., Yang, G., and Stadie, B. C. World model as a graph: Learning latent landmarks for planning. In *International conference on machine learning*, pp. 12611–12620. PMLR, 2021.

Zhang, M., Lucas, J., Ba, J., and Hinton, G. E. Lookahead optimizer: k steps forward, 1 step back. *Advances in neural information processing systems*, 32, 2019.

Zhang, Y., Liu, Y., Yuan, H., Yuan, Y., Gu, Q., and Yao, A. C.-C. On the design of kl-regularized policy gradient algorithms for llm reasoning. *arXiv preprint arXiv:2505.17508*, 2025.

Zheng, C., Liu, S., Li, M., Chen, X.-H., Yu, B., Gao, C., Dang, K., Liu, Y., Men, R., Yang, A., et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

Zhou, Y., Lyu, K., Rawat, A. S., Menon, A. K., Rostamizadeh, A., Kumar, S., Kagy, J.-F., and Agarwal, R. Distillspec: Improving speculative decoding via knowledge distillation. In *The Twelfth International Conference on Learning Representations*, 2024.

Zhu, B., Sharma, H., Frujeri, F. V., Dong, S., Zhu, C., Jordan, M. I., and Jiao, J. Fine-tuning language models with advantage-induced policy alignment. *arXiv preprint arXiv:2306.02231*, 2023.

Ziebart, B. D., Maas, A. L., Bagnell, J. A., Dey, A. K., et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pp. 1433–1438. Chicago, IL, USA, 2008.

Appendix Contents

- A Algorithm Box for EMA Policy Gradient 14
- B Additional Experimental Results 16
- C Derivations for EMA Policy Gradient Dynamics
  - C.1 Policy gradient dynamics under local quadratic KL approximation . . . 17
  - C.2 Quasi-steady state analysis over a k-step window (Proof of Lemma 4.1) . . . 18
  - C.3 Eigen-decomposition of Fisher and mode-wise dynamics (Proof of Lemma 4.2) . . . 19
  - C.4 Stability and oscillations . . . 20
  - C.5 Steady-state lag and bounds . . . 20
- D Sequence-level KL vs. Token-level KL
  - D.1 Sequence-level KL gradients . . . 22
  - D.2 Analysis of sequence-level KL gradients . . . 22
  - D.3 Token-level KL gradients . . . 23
- E Derivation of Sampled Reverse KL Estimators
  - E.1 Sampled KL estimators (K1 / K2 / K3) and their expected values . . . 24
  - E.2 Gradients of K1 / K2 / K3 under autodiff $\mathbb{E}_p[\nabla_\theta(\cdot)]$ . . . 24
  - E.3 The $\boldsymbol{r}$-trick restores $\nabla_\theta \mathbb{E}_p[\cdot]$ from $\mathbb{E}_p[\nabla_\theta(\cdot)]$. . . . 25
  - E.4 **K3**$^{++}$ yields unbiased reverse-KL values and gradients . . . 25
  - E.5 **K4** yields unbiased reverse-KL values and gradients . . . 26
- F Derivation of Sampled Forward KL Estimators
  - F.1 Forward KL and its gradient . . . 27
  - F.2 **K5** yields unbiased forward KL values . . . 27
  - F.3 **K5** yields unbiased forward KL gradients . . . 28
  - F.4 Variance reduction in **K5** gradients . . . 28
- G Bias-Variance Tradeoff in Top-k KL: An Illustrative Synthetic Setup 29
- H Top-k $\boldsymbol{f}$-Divergence Estimator
  - H.1 f-divergence and its gradients . . . 30
  - H.2 A generic, unbiased, sampled f-divergence estimator . . . 31
  - H.3 Top-k f-divergence estimator: A generalization of Top-k KL estimator . . . 31
- I Optimal Policies under Forward vs Reverse KL Regularization
  - I.1 Reward transformation for equivalent optima between Forward KL and Reverse KL . . . 33
  - I.2 Reward transformation for equivalent optima across f-divergences . . . 34
- J Policy Gradient Methods can be Derived from Sampled KL Estimators
  - J.1 K1 estimator leads to on-policy PPO / GRPO loss . . . 36
  - J.2 K2 estimator leads to APA / Kimi loss . . . 37
  - J.3 K5 estimator leads to RWR / MPO loss . . . 38
  - J.4 K3 / K3^{++} / K4 estimators lead to combined losses . . . 38
- K Policy Gradient as $\boldsymbol{f}$-Divergence Minimization 40

A. Algorithm Box for EMA Policy Gradient

Algorithm 3 **EMA** Policy Gradient with TopkKL

Require: Prompt set $\mathcal{D}(x)$, LLM $\pi_\theta(y|x)$, reward $R(x, y)$.
Require: Anchor policy $\pi_{\theta_{\text{ema}}}(y|x)$.

$\theta_{\text{ema}} \leftarrow \theta$
for iteration $t = 1, 2, \ldots$ **do**
  Sample task prompt $x \sim \mathcal{D}$
  $\theta_{\text{old}} \leftarrow \theta$ for sampling
  Sample $y \sim \pi_{\theta_{\text{old}}}(\cdot \mid x)$; let L be the length of y
  Compute $\log \pi_{\theta_{\text{old}}}(y_n \mid x, y_{0:n-1})$ and $\log \pi_{\text{ref}}(y_n \mid x, y_{0:n-1})$ for $n = 0, \ldots, L-1$
  if using *reverse* KL **then**
    Compute $[q_0, \ldots, q_{L-1}]$, where q_n is the top-k logit indices from $\pi_{\theta_{\text{old}}}(\cdot \mid x, y_{0:n-1})$ {In *reverse* KL, the expectation is taken under π_θ}
  else
    Compute $[q_0, \ldots, q_{L-1}]$, where q_n is the top-k logit indices from $\pi_{\theta_{\text{ema}}}(\cdot \mid x, y_{0:n-1})$ {In *forward* KL, the expectation is taken under $\pi_{\theta_{\text{ema}}}$}
  end if
  Use $R(x, y)$ to compute advantage function A
  for $\upsilon = 1, \ldots, \Lambda$ **do**
    Compute $\log \pi_\theta(y_n \mid x, y_{0:n-1})$ for $n = 0, \ldots, L-1$
    Let $s_n = \text{sg}(\pi_\theta(y_n \mid x, y_{0:n-1}) / \pi_{\theta_{\text{old}}}(y_n \mid x, y_{0:n-1}))$
    $\mathcal{J}(\theta) = \sum_{n=0}^{L-1} s_n \, A \, \log \pi_\theta(y_n \mid x, y_{0:n-1})$, clip $\mathcal{J}(\theta)$ in PPO-style
    if using *reverse* KL **then**
      $\widehat{\text{KL}} = \sum_{n=0}^{L-1} \text{Topk}\textbf{Reverse}\text{KL}(\pi_\theta(\cdot|x, y_{0:n-1}), \pi_{\theta_{\text{ema}}}(\cdot|x, y_{0:n-1}), \pi_{\text{old}}(\cdot|x, y_{0:n-1}), y_n, q_n)$
    else
      $\widehat{\text{KL}} = \sum_{n=0}^{L-1} \text{Topk}\textbf{Forward}\text{KL}(\pi_\theta(\cdot|x, y_{0:n-1}), \pi_{\theta_{\text{ema}}}(\cdot|x, y_{0:n-1}), \pi_{\text{old}}(\cdot|x, y_{0:n-1}), y_n, q_n)$
    end if
    $L(\theta) = -\mathcal{J}(\theta) + \beta \, \widehat{\text{KL}}$
    $\theta \leftarrow \theta - \alpha \, \nabla_\theta L(\theta)$
  end for
  if $t \bmod T_{\text{ema}} = 0$ **then**
    $\theta_{\text{ema}} \leftarrow \eta \, \theta_{\text{ema}} + (1 - \eta) \, \theta$
  end if
end for

The changes required to modify any policy gradient method into EMA-PG (with **EMA anchor** and **Top-k KL**) are highlighted above.

Details of Top-k KL: For completeness, we write down the full expression of Top-k KL terms. Using common notations:

$$y \sim \pi_{\text{old}}(\cdot \mid x) \quad h_n = (x, y_{0:n-1}) \quad s_n = \text{sg}\left(\frac{\pi_\theta(y_n \mid h_n)}{\pi_{\theta_{\text{old}}}(y_n \mid h_n)}\right) \quad s_n^{\text{clip}} = \text{clip}(s_n, s_{\min}, s_{\max})$$

The Top-k Reverse KL can be written as:

$$q_n \leftarrow \text{Topk}(\pi_{\theta_{\text{old}}}(\cdot \mid h_n)) \qquad (\widehat{\text{KL}}_{\text{sampled}})_n = \frac{\pi_\theta(y_n \mid h_n)}{\text{sg}(\pi_\theta(y_n \mid h_n))} \, \text{sg}\left(\log \frac{\pi_{\text{ref}}(y_n \mid h_n)}{\pi_\theta(y_n \mid h_n)}\right)$$

$$(\text{Topk}\textbf{Reverse}\text{KL})_n = \sum_{j \in q_n} \pi_\theta(j \mid h_n) \log \frac{\pi_\theta(j \mid h_n)}{\pi_{\text{ref}}(j \mid h_n)} + \mathbb{1}(y_n \notin q_n) \cdot s_n^{\text{clip}} \cdot (\widehat{\text{KL}}_{\text{sampled}})_n$$

And the Top-k Forward KL can be written as:

$$q_n \leftarrow \mathrm{Topk}(\pi_{\text{ref}}(\cdot \mid h_n)) \qquad (\widehat{\mathrm{FKL}}_{\text{sampled}})_n = \mathrm{sg}\left(\frac{\pi_{\text{ref}}(y_n \mid h_n)}{\pi_\theta(y_n \mid h_n)}\right) \log \frac{\pi_{\text{ref}}(y_n \mid h_n)}{\pi_\theta(y_n \mid h_n)} + \log \frac{\pi_\theta(y_n|h_n)}{\mathrm{sg}(\pi_\theta(y_n|h_n))}$$

$$(\mathrm{Topk}\textbf{Forward}\mathrm{KL})_n = \sum_{j \in q_n} \pi_{\text{ref}}(j \mid h_n) \log \frac{\pi_{\text{ref}}(j \mid h_n)}{\pi_\theta(j \mid h_n)} + \mathbb{1}(y_n \notin q_n) \cdot s_n^{\text{clip}} \cdot (\widehat{\mathrm{FKL}}_{\text{sampled}})_n$$

Then we sum up the token-wise KL along the sequence length:

$$\widehat{\mathrm{KL}} = \sum\nolimits_{n=0}^{L-1} (\mathrm{Topk}\textbf{Reverse}\mathrm{KL})_n \qquad \widehat{\mathrm{FKL}} = \sum\nolimits_{n=0}^{L-1} (\mathrm{Topk}\textbf{Forward}\mathrm{KL})_n$$

Details of the Base RL Algorithm We discuss some additional implementation details below.

- For the policy gradient loss function $\mathcal{J}(\theta)$, we use a standard version of GRPO: PPO-style clipping (Schulman et al., 2017b), Monte-Carlo estimation of advantages (Shao et al., 2024; Liu et al., 2025a), with clip-higher (Yu et al., 2025):

$$\begin{aligned}
&y^{(i)} \sim \log \pi_{\theta_{\text{old}}}(\cdot \mid x), \quad i = 0, \ldots, N-1 \\
&V(x) = \tfrac{1}{N} \sum\nolimits_{i=0}^{N-1} R(x, y^{(i)}) \\
&A(x, y^{(i)}) = R(x, y^{(i)}) - V(x) \qquad \mathrm{std}(x) = \sqrt{\tfrac{1}{N-1} \sum\nolimits_{i=0}^{N-1} \left(R(x, y^{(i)}) - V(x)\right)^2} \\
&s_n^{(i)} = \mathrm{sg}\left(\frac{\pi_\theta(y_n^{(i)} \mid x, y_{0:n-1}^{(i)})}{\pi_{\theta_{\text{old}}}(y_n^{(i)} \mid x, y_{0:n-1}^{(i)})}\right) \qquad M_n^{(i)} = \begin{cases} 0 & \text{if } A(x, y^{(i)}) > 0 \text{ and } s_l^{(i)} > 1 + \epsilon_{\text{high}}, \\ 0 & \text{if } A(x, y^{(i)}) < 0 \text{ and } s_l^{(i)} < 1 - \epsilon_{\text{low}}, \\ 1 & \text{otherwise.} \end{cases} \\
&\mathcal{J}^{\text{clip}}(x) = \tfrac{1}{N} \tfrac{1}{L} \sum\nolimits_{i=0}^{N-1} \left(\sum\nolimits_{n=0}^{L-1} s_n^{(i)} M_n^{(i)} \log \pi_\theta(y_n^{(i)} \mid x, y_{0:n-1}^{(i)})\right) A(x, y^{(i)}) \\
&\mathcal{J}(\theta) = \mathbb{E}_{x \sim \mathcal{D}}\left[\mathcal{J}^{\text{clip}}(x) \,/\, \mathrm{std}(x)\right]
\end{aligned}$$

  where $\epsilon_{\text{high}} = 0.28$ and $\epsilon_{\text{low}} = 0.2$. $\mathcal{J}^{\text{clip}}$ applies the PPO-clip objective written in an equivalent form using binary masks (Wang et al., 2020; Chen et al., 2025). Note that the two techniques of EMA-PG are orthogonal to how we compute $\mathcal{J}(\theta)$; we can in principle also choose to use CISPO (Chen et al., 2025) or GSPO (Zheng et al., 2025) as the base algorithm.

- For the optimizer, we use Adam (Kingma, 2014) with decoupled weight decay (Loshchilov & Hutter, 2017), with a learning rate of 1e-6 for *full finetuning*. Since the focus of our study is to device a better algorithm for full-scale post-training, we do not use any low-rank adaptation (LoRA) (Hu et al., 2022) in our experiments.

- $\beta = 0.001$ for KL regularization.

- $T_{\text{ema}} = 10$, following prior empirical practice among practitioners of Deep Q-learning that target networks (even though under EMA soft update) still only get updated every T_{ema} iterations; (Zhang et al., 2021) set the value to 10 (see `target_update_freq` $= 10$ in their implementation).

B. Additional Results

Method	NQ	TriviaQA	PopQA	HotpotQA	2wiki	Musique	Bamboogle	Avg.
Qwen2.5-**3B-Instruct** (RL)								
before RL	0.205	0.427	0.277	0.199	0.200	0.044	0.153	0.215
PPO	0.341	0.545	0.378	0.324	0.319	0.103	0.264	0.325
GRPO	0.397	0.565	0.391	0.331	0.310	0.124	0.232	0.336
EMA-PG (Reverse KL)	**0.411**	**0.581**	**0.405**	**0.360**	**0.337**	**0.144**	**0.355**	**0.370**
Improvement over GRPO	3.5% ↑	2.8% ↑	3.6% ↑	8.8% ↑	8.7% ↑	16.1 % ↑	53.0 % ↑	10.1 % ↑

Table 4. **EMA-PG outperforms GRPO on Instruct model as well, but interestingly, the gain is not as significant as on base model.** This empirically shows that the *cold-start* strategy (Guo et al., 2025) is a valid and well-justified strategy.

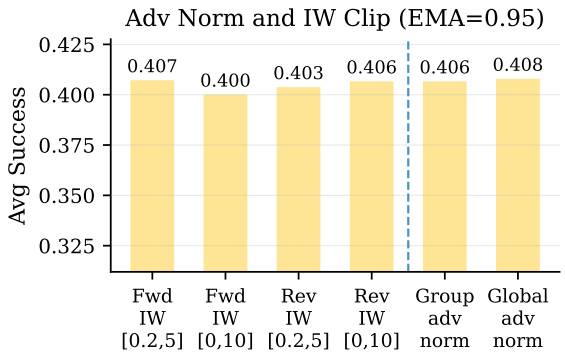

Figure 9. Ablation on importance weight (IW) clip range and advantage normalization strategies (group-wise vs global).

The off-policy importance weight (IW) clipping range is the one specified in

$$s_n^{\text{clip}} = \text{clip}(s_n, s_{\min}, s_{\max})$$

which is used in both Reverse KL (Rev) and Forward KL (Fwd); we study the IW clip range in both situations. For Forward KL, using a tighter clipping range $[0, 2.5]$ helps; for reverse KL, using a more lenient clipping range $[0, 10]$ seems to perform better. Note that ignoring off-policy correction is equivalent to setting the clip range to $s_{\min} = s_{\max} = 1$.

In addition, we also study the advantage normalization strategies in the $\mathcal{J}(\theta)$, in particular global advantage normalization in the original PPO (Schulman et al., 2017b). This has been shown to slightly improve performance in (Hu et al., 2025). It essentially adds additional normalization on advantages $\boldsymbol{A} \in \mathbb{R}^{B\times N}$ across an entire batch of B problems $\{x^{(1)}, \ldots, x^{(B)}\}$:

$$(\boldsymbol{A} - \text{mean}(\boldsymbol{A}))/\text{std}(\boldsymbol{A})$$

which does a z-score normalization on the **flattened** $\boldsymbol{A}$ of shape $(B \times N)$, and discards per-problem normalization based on $\text{std}(x)$. We find that this helps slightly but did not make a significant difference on performance.

C. Derivations for EMA Policy Gradient Dynamics

Fisher matrix The Fisher information matrix of a policy π_θ is:

$$F_\theta \triangleq \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[(\nabla_\theta \log \pi_\theta(y|x))(\nabla_\theta \log \pi_\theta(y|x))^\top\right]$$

The Fisher matrix is positive semidefinite (PSD), so all eigenvalues satisfy $\lambda_i \geq 0$. Consequently,

$$\lambda_{\max}(F_\theta) \leq \mathrm{tr}(F_\theta) = \mathbb{E}_{\substack{x\sim\mathcal{D}\\ y\sim\pi_\theta(\cdot|x)}}\left[\|\nabla_\theta \log \pi_\theta(y|x)\|_2^2\right] \tag{28}$$

giving us a practical way of estimating an upper bound on the largest Fisher eigenvalue of a policy under $\mathcal{D}$.

Setup. Let $J(\theta) \coloneqq R(\pi_\theta)$ denote the (expected) RL objective. Define the EMA parameters and the *lag*

$$\theta_{\text{ema},t+1} = \eta\,\theta_{\text{ema},t} + (1-\eta)\,\theta_t, \qquad \eta \in [0,1),$$
$$\delta_t \coloneqq \theta_t - \theta_{\text{ema},t}.$$

Let $g_t \coloneqq \nabla_\theta J(\theta)|_{\theta=\theta_t}$ be the policy-gradient (or any ascent direction). We consider the update

$$\theta_{t+1} = \theta_t + \alpha\Big[g_t - \beta\,\nabla_\theta \mathrm{KL}(\pi_\theta \,\|\, \pi_{\theta_{\text{ema},t}})\Big]_{\theta=\theta_t}, \tag{29}$$

with step size $\alpha > 0$ and regularization coefficient $\beta \geq 0$.

C.1. Policy Gradient Dynamics under Local Quadratic KL Approximation

We assume a local quadratic approximation of the KL around θ_t:

$$\mathrm{KL}(\pi_{\theta_t} \,\|\, \pi_{\theta_{\text{ema},t}}) \;\approx\; \frac{1}{2}(\theta_t - \theta_{\text{ema},t})^\top F_t(\theta_t - \theta_{\text{ema},t}) \;=\; \frac{1}{2}\delta_t^\top F_t \delta_t, \tag{30}$$

where $F_t \coloneqq F_{\theta_t}$ is the (empirical) Fisher information matrix at θ_t. Differentiating the RHS of (30) w.r.t. θ_t yields

$$\nabla_{\theta_t}\,\mathrm{KL}(\pi_{\theta_t} \,\|\, \pi_{\theta_{\text{ema},t}}) \;\approx\; \nabla_{\theta_t}\left(\tfrac{1}{2}\delta_t^\top F_t \delta_t\right) \;=\; F_t\delta_t, \tag{31}$$

where we treat F_t as locally constant under this approximation. Substituting (31) into (29) gives

$$\theta_{t+1} = \theta_t + \alpha g_t - \alpha\beta F_t \delta_t. \tag{32}$$

Lemma C.1 (Lag dynamics)**.** *Under* (32) *and the EMA recursion, the lag obeys*

$$\delta_{t+1} = (\eta I - \alpha\beta F_t)\,\delta_t + \alpha g_t. \tag{33}$$

Proof. By definition $\delta_{t+1} = \theta_{t+1} - \theta_{\text{ema},t+1}$, and by the EMA update,

$$\theta_{\text{ema},t+1} = \eta\theta_{\text{ema},t} + (1-\eta)\theta_t \;\Rightarrow\; \theta_t - \theta_{\text{ema},t+1} = \theta_t - \eta\theta_{\text{ema},t} - (1-\eta)\theta_t = \eta(\theta_t - \theta_{\text{ema},t}) = \eta\,\delta_t. \tag{34}$$

Therefore,

$$\delta_{t+1} = (\theta_{t+1} - \theta_t) + (\theta_t - \theta_{\text{ema},t+1}) = (\theta_{t+1} - \theta_t) + \eta\,\delta_t. \tag{35}$$

Using (32), $\theta_{t+1} - \theta_t = \alpha g_t - \alpha\beta F_t\delta_t$, so

$$\delta_{t+1} = \eta\delta_t + \alpha g_t - \alpha\beta F_t\delta_t = (\eta I - \alpha\beta F_t)\delta_t + \alpha g_t. \tag{36}$$

□

Define the (time-varying) linear operator

$$D_t \coloneqq \eta I - \alpha\beta F_t. \tag{37}$$

Collecting (32) and (33) gives the affine joint update

$$\begin{bmatrix}\theta_{t+1}\\ \delta_{t+1}\end{bmatrix} = \begin{bmatrix} I & -\alpha\beta F_t \\ 0 & \eta I - \alpha\beta F_t\end{bmatrix}\begin{bmatrix}\theta_t\\ \delta_t\end{bmatrix} + \begin{bmatrix}\alpha I\\ \alpha I\end{bmatrix} g_t. \tag{38}$$

C.2. Quasi-steady state analysis over a k-step window (Proof of Lemma 4.1)

Assumption (frozen gradient and Fisher). Fix an integer $k \geq 1$ and assume that over the window $\{t, \dots, t+k-1\}$,

$$g_{t+j} \approx g, \qquad F_{t+j} \approx F \qquad \text{for } j = 0, \dots, k-1, \tag{39}$$

for some constant vectors/matrices g, F. Define the corresponding constant operator

$$D \coloneqq \eta I - \alpha\beta F. \tag{40}$$

C.2.1. Closed form for δ_{t+k}

Unrolling (33) with constant D and g yields

$$\begin{aligned}
\delta_{t+1} &= D\delta_t + \alpha g, \\
\delta_{t+2} &= D\delta_{t+1} + \alpha g = D^2\delta_t + \alpha(D+I)g, \\
&\vdots \\
\delta_{t+k} &= D^k\delta_t + \alpha \sum_{j=0}^{k-1} D^j g.
\end{aligned} \tag{41}$$

Define the geometric-series matrix

$$S_k \coloneqq \sum_{j=0}^{k-1} D^j. \tag{42}$$

Then (41) becomes

$$\delta_{t+k} = D^k\delta_t + \alpha S_k\, g. \tag{43}$$

C.2.2. Invertibility of $I - D$ and closed form for S_k

Let v_i be an eigenvector of F, and λ_i be its corresponding eigenvalue. Using $D = \eta I - \alpha\beta F$,

$$Dv_i = (\eta I - \alpha\beta F)v_i = (\eta - \alpha\beta\lambda_i)v_i. \tag{44}$$

Define

$$\chi_i \coloneqq \eta - \alpha\beta\lambda_i. \tag{45}$$

Then $Dv_i = \chi_i v_i$. Since $\lambda_i \geq 0$ ($F \succeq 0$), $\eta < 1$, and $\beta, \alpha \geq 0$, we have

$$\chi_i \leq \eta < 1, \tag{46}$$

Hence $1 - \chi_i > 0$ for all i, so $I - D$ is invertible. Consequently,

$$S_k = \sum_{j=0}^{k-1} D^j = (I - D^k)(I - D)^{-1}. \tag{47}$$

C.2.3. Closed form for θ_{t+k} and the M_k matrix

From (32) with frozen F, g,

$$\theta_{t+1} = \theta_t + \alpha g - \alpha\beta F\delta_t, \tag{48}$$

$$\theta_{t+k} = \theta_t + \sum_{j=0}^{k-1} \big(\alpha g - \alpha\beta F\delta_{t+j}\big) = \theta_t + \alpha k\, g - \alpha\beta F \sum_{j=0}^{k-1} \delta_{t+j}. \tag{49}$$

Using (43) at times $t+j$,

$$\delta_{t+j} = D^j \delta_t + \alpha S_j g, \tag{50}$$

so

$$\sum_{j=0}^{k-1} \delta_{t+j} = \sum_{j=0}^{k-1} D^j \delta_t + \alpha \sum_{j=0}^{k-1} S_j g = S_k \delta_t + \alpha \Big(\sum_{j=0}^{k-1} S_j \Big) g. \tag{51}$$

Define

$$M_k \coloneqq \sum_{j=0}^{k-1} S_j. \tag{52}$$

Then

$$\theta_{t+k} = \theta_t + \alpha k\, g - \alpha\beta F S_k \delta_t - \alpha^2 \beta F M_k\, g. \tag{53}$$

Closed form for M_k. Using $S_j = (I - D^j)(I - D)^{-1}$ from (47),

$$M_k = \sum_{j=0}^{k-1} (I - D^j)(I - D)^{-1} = \Big(kI - \sum_{j=0}^{k-1} D^j \Big)(I - D)^{-1} = (kI - S_k)(I - D)^{-1}. \tag{54}$$

Collecting (53) and (43), one may the dynamics as a homogeneous linear map in (θ, δ, g):

$$\begin{bmatrix} \theta_{t+k} \\ \delta_{t+k} \\ g \end{bmatrix} = \begin{bmatrix} I & -\alpha\beta F S_k & \alpha k I - \alpha^2 \beta F M_k \\ 0 & D^k & \alpha S_k \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \theta_t \\ \delta_t \\ g \end{bmatrix}$$

C.3. Eigen-decomposition of Fisher and mode-wise dynamics (Proof of Lemma 4.2)

Since F is a Fisher information matrix (or symmetric empirical approximation), we assume F is symmetric PSD:

$$F \succeq 0.$$

Therefore, there exists an orthonormal eigenbasis $\{v_i\}_{i=0}^{d-1}$ and eigenvalues $\lambda_i \geq 0$ such that

$$F v_i = \lambda_i v_i, \qquad \lambda_i \geq 0.$$

We refer to v_i as the i-th Fisher eigenvector (eigen-direction). "High curvature" corresponds to large λ_i. Previously we have already shown that

$$\chi_i \coloneqq \eta - \alpha\beta\lambda_i \qquad \chi_i \leq \eta < 1,$$

and χ_i may be negative if $\alpha\beta\lambda_i > \eta$. Now we define the scalar projections

$$\delta_t^{(i)} \coloneqq v_i^\top \delta_t, \qquad g^{(i)} \coloneqq v_i^\top g. \tag{55}$$

Projecting (43) onto v_i yields

$$\delta_{t+k}^{(i)} = v_i^\top D^k \delta_t + \alpha\, v_i^\top S_k g = \chi_i^k \delta_t^{(i)} + \alpha \sum_{j=0}^{k-1} \chi_i^j\, g^{(i)} \tag{56}$$

$$= \chi_i^k \delta_t^{(i)} + \alpha\, \frac{1 - \chi_i^k}{1 - \chi_i}\, g^{(i)}, \tag{57}$$

This yields the decomposition into a transient term and a particular solution.

C.4. Stability and oscillations

C.4.1. Stability condition

The transient part of the lag dynamics is $\delta_{t+1} = D\delta_t$. Stability (transient $\to 0$) requires $\rho(D) < 1$, equivalently

$$\max_i |\chi_i| < 1. \tag{58}$$

Since $\chi_i \le \eta < 1$, the upper inequality $|\chi_i| < 1$ is only violated if $\chi_i \le -1$ for some i. The most negative eigenvalue occurs at $\lambda_{\max} := \max_i \lambda_i$, so stability is equivalent to

$$\eta - \alpha\beta\lambda_{\max} > -1 \quad \Longleftrightarrow \quad \alpha\beta\lambda_{\max} < 1 + \eta. \tag{59}$$

This proves (9).

C.4.2. Non-oscillatory vs. oscillatory transients

From (57), the sign of the transient in mode i is governed by χ_i^k. If $\chi_i \in (0, 1)$, the transient decays monotonically. If $\chi_i \in (-1, 0)$, the transient alternates sign each step (damped overshoot/undershoot), which we call "oscillatory". Thus:

- **Non-oscillatory in all modes** iff $\chi_i \ge 0$ for all i, i.e.

$$\eta - \alpha\beta\lambda_{\max} \ge 0 \quad \Longleftrightarrow \quad \alpha\beta\lambda_{\max} \le \eta. \tag{60}$$

- **Oscillatory in some high-curvature directions** if there exists i with $\chi_i < 0$, which can occur when $\alpha\beta\lambda_{\max} > \eta$.

Combined with stability (59) yields the three regimes summarized in the main text:

- If $0 \le \alpha\beta\lambda_{\max} \le \eta$, then $\chi_i \in [0, \eta] \subset [0, 1)$ for all i, so the dynamics is stable and all transients decay monotonically (non-oscillatory).
- If $\eta < \alpha\beta\lambda_{\max} < 1 + \eta$, then stability holds ($|\chi_i| < 1$ for all i) but there exist high-curvature directions with $\chi_i < 0$, producing damped sign-flipping transients (oscillations).
- If $\alpha\beta\lambda_{\max} \ge 1 + \eta$, then $\chi_{\max} \le -1$ for the top-curvature direction, so $\max_i |\chi_i| \ge 1$ and the lag dynamics is unstable.

C.5. Steady-state lag and bounds

Assume stability (59), $D^k \to 0$ as $k \to \infty$. Taking $k \to \infty$ in (43) gives the steady lag

$$\delta_* = \lim_{k\to\infty} \delta_{t+k} = \alpha\Big(\sum_{j=0}^{\infty} D^j\Big) g = \alpha(I - D)^{-1} g. \tag{61}$$

Using $I - D = I - (\eta I - \alpha\beta F) = (1 - \eta)I + \alpha\beta F$, we obtain

$$\delta_* = \alpha\big((1 - \eta)I + \alpha\beta F\big)^{-1} g, \tag{62}$$

In each eigen-direction,

$$\delta_*^{(i)} = \frac{\alpha}{(1 - \eta) + \alpha\beta\lambda_i}\, g^{(i)}. \tag{63}$$

C.5.1. Euclidean norm upper bound

Since $F \succeq 0$, we have the PSD ordering

$$(1 - \eta)I + \alpha\beta F \;\succeq\; (1 - \eta)I.$$

For positive definite matrices, PSD ordering reverses under inversion, hence

$$\big((1-\eta)I + \alpha\beta F\big)^{-1} \;\preceq\; \frac{1}{1-\eta} I.$$

Therefore,

$$\|\delta_*\|_2 = \alpha \Big\| \big((1-\eta)I + \alpha\beta F\big)^{-1} g \Big\|_2 \le \alpha \Big\| \big((1-\eta)I + \alpha\beta F\big)^{-1} \Big\|_2 \|g\|_2 \;\le\; \alpha \cdot \frac{1}{1-\eta} \|g\|_2, \tag{64}$$

C.5.2. Quasi-steady KL magnitude

Under the quadratic KL approximation (30), the quasi-steady KL is

$$\widehat{\mathrm{KL}}_* \;\approx\; \frac{1}{2} \delta_*^\top F \delta_*. \tag{65}$$

Expanding in the eigenbasis, $F = \sum_i \lambda_i v_i v_i^\top$ and $\delta_* = \sum_i \delta_*^{(i)} v_i$, gives

$$\delta_*^\top F \delta_* = \sum_{i=0}^{d-1} \lambda_i \big(\delta_*^{(i)}\big)^2. \tag{66}$$

Substituting (63) yields

$$\widehat{\mathrm{KL}}_* \;\approx\; \frac{\alpha^2}{2} \sum_{i=0}^{d-1} \frac{\lambda_i \big(g^{(i)}\big)^2}{\big((1-\eta) + \alpha\beta\lambda_i\big)^2}, \tag{67}$$

Thus, within the quasi-steady model, increasing η increases $\widehat{\mathrm{KL}}_*$ unless β is increased or α is decreased.

D. Sequence-level KL vs. Token-level KL

Notation and setup For prompt x, consider the conditional distribution over output sequence $y = (y_0, \ldots, y_{L-1})$.

$$\pi_\theta(y \mid x) = \prod_{n=0}^{L-1} \pi_\theta(y_n \mid x, y_{0:n-1}).$$

For each position l, define the prefix (history)

$$h_n \triangleq (x, y_{0:n-1}),$$

and the per-token log-ratio and score term

$$\rho_n \triangleq \log \frac{\pi_\theta(y_n \mid h_n)}{\pi_{\text{ref}}(y_n \mid h_n)}, \qquad \omega_n \triangleq \nabla_\theta \log \pi_\theta(y_n \mid h_n).$$

Note that $\log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} = \sum_{n=0}^{L-1} \rho_n$ and $\nabla_\theta \log \pi_\theta(y \mid x) = \sum_{n=0}^{L-1} \omega_n$.

D.1. Sequence-level KL gradients

Define the (sequence-level) conditional KL:

$$\mathrm{KL}\big(\pi_\theta(\cdot \mid x) \,\|\, \pi_{\text{ref}}(\cdot \mid x)\big) \triangleq \mathbb{E}_{y \sim \pi_\theta(\cdot|x)} \left[\log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \right].$$

Differentiate it:

$$\begin{aligned} \nabla_\theta \mathrm{KL} &= \nabla_\theta \int \pi_\theta(y \mid x) \, \log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \, dy \\ &= \int \nabla_\theta \pi_\theta(y \mid x) \, \log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \, dy + \int \pi_\theta(y \mid x) \, \nabla_\theta \log \pi_\theta(y \mid x) \, dy. \end{aligned}$$

Using $\nabla_\theta \pi_\theta = \pi_\theta \nabla_\theta \log \pi_\theta$ and $\int \pi_\theta \nabla_\theta \log \pi_\theta \, dy = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta] = 0$, we obtain the standard score-function form:

$$\nabla_\theta \mathrm{KL}\big(\pi_\theta(\cdot \mid x) \,\|\, \pi_{\text{ref}}(\cdot \mid x)\big) = \mathbb{E}_{y \sim \pi_\theta(\cdot|x)} \left[\log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \, \nabla_\theta \log \pi_\theta(y \mid x) \right]. \tag{68}$$

D.2. Analysis of sequence-level KL gradients

Substitute the autoregressive decompositions:

$$\nabla_\theta \mathrm{KL}\big(\pi_\theta(\cdot \mid x) \,\|\, \pi_{\text{ref}}(\cdot \mid x)\big) = \mathbb{E}_{y \sim \pi_\theta(\cdot|x)} \left[\left(\sum_{m=0}^{L-1} \rho_m \right) \left(\sum_{n=0}^{L-1} \omega_n \right) \right] = \mathbb{E}_{y \sim \pi_\theta(\cdot|x)} \left[\sum_{n=0}^{L-1} \sum_{m=0}^{L-1} \rho_m \omega_n \right]. \tag{69}$$

Split the inner sum into "future" and "past" relative to n:

$$\sum_{m=0}^{L-1} \rho_m \omega_n = \left(\sum_{m \geq n} \rho_m \right) \omega_n + \left(\sum_{m < n} \rho_m \right) \omega_n. \tag{70}$$

Why the "past" terms vanish Recall the conditional expectation notation

$$\mathbb{E}_n[\cdot] \triangleq \mathbb{E}_{y_n \sim \pi_\theta(\cdot|h_n)}[\cdot], \qquad h_n \triangleq (x, y_{0:n-1}).$$

We will use the standard Law of total expectation in the following explicit form: for any random variable $Z = Z(x, y_{0:n})$,

$$\mathbb{E}_{y \sim \pi_\theta(\cdot|x)}[Z] = \mathbb{E}_{y_{0:n-1} \sim \pi_\theta(\cdot|x)} \Big[\mathbb{E}_{y_n \sim \pi_\theta(\cdot|x, y_{0:n-1})}[Z] \Big] = \mathbb{E}_{y_{0:n-1} \sim \pi_\theta(\cdot|x)} \big[\mathbb{E}_n[Z] \big]. \tag{71}$$

Step 1: the score has zero mean. Conditioning on a fixed prefix h_n, we have

$$\mathbb{E}_n[\omega_n] = \sum_a \pi_\theta(a \mid h_n)\, \nabla_\theta \log \pi_\theta(a \mid h_n) = \sum_a \nabla_\theta \pi_\theta(a \mid h_n) = \nabla_\theta \sum_a \pi_\theta(a \mid h_n) = \nabla_\theta 1 = 0.$$

Step 2: past log-ratios are constant under $\mathbb{E}_n[\cdot]$. Since $\sum_{m<n} \rho_m$ depends only on the prefix $h_n = (x, y_{0:n-1})$, it does not depend on y_n and can be pulled out of $\mathbb{E}_n[\cdot]$:

$$\mathbb{E}_n\left[\left(\sum_{m<n} \rho_m\right) \omega_n\right] = \left(\sum_{m<n} \rho_m\right) \mathbb{E}_n[\omega_n] = 0.$$

Step 3: unconditional past terms also vanish. Apply (71) with $Z = \left(\sum_{m<n} \rho_m\right) \omega_n$:

$$\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\left(\sum_{m<n} \rho_m\right) \omega_n\right] = \mathbb{E}_{y_{0:n-1}\sim\pi_\theta(\cdot|x)}\left[\mathbb{E}_n\left[\left(\sum_{m<n} \rho_m\right) \omega_n\right]\right] = 0, \tag{72}$$

which justifies dropping the "past tokens" contribution in the main derivation.

D.3. Token-level KL gradients

Define the token-level conditional KL at position n:

$$\widehat{\mathrm{KL}}_n \triangleq \mathrm{KL}\big(\pi_\theta(\cdot \mid h_n) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid h_n)\big) = \mathbb{E}_{y_n\sim\pi_\theta(\cdot|h_n)}\left[\log \frac{\pi_\theta(y_n \mid h_n)}{\pi_{\mathrm{ref}}(y_n \mid h_n)}\right].$$

By the same argument as (68), its gradient can be written as

$$\nabla_\theta \widehat{\mathrm{KL}}_n = \mathbb{E}_{y_n\sim\pi_\theta(\cdot|h_n)}\left[\log \frac{\pi_\theta(y_n \mid h_n)}{\pi_{\mathrm{ref}}(y_n \mid h_n)}\, \nabla_\theta \log \pi_\theta(y_n \mid h_n)\right] = \mathbb{E}_{y_n\sim\pi_\theta(\cdot|h_n)}[\rho_n\, \omega_n]. \tag{73}$$

Hence the token-level KL objective yields the per-token weighting:

$$\nabla_\theta L_{\mathrm{kl}}^{\mathrm{token}} = \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\sum_{n=0}^{L-1} \nabla_\theta \widehat{\mathrm{KL}}_n\right] = \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\sum_{n=0}^{L-1} \rho_n \omega_n\right]. \tag{74}$$

Comparison Sequence-level KL weights token n by the cumulative *future* log-ratio $\sum_{m\geq n} \rho_m$, while token-level KL weights token n only by the *local* log-ratio ρ_n.

$$\nabla_\theta L_{\mathrm{KL}}^{\mathrm{seq}} = \mathbb{E}_{\pi_\theta(y|x)}\Big[\sum_n \Big(\sum_{m\geq n} \rho_m\Big) \omega_n\Big] \qquad \nabla_\theta L_{\mathrm{KL}}^{\mathrm{token}} = \mathbb{E}_{\pi_\theta(y|x)}\Big[\sum_n \big(\rho_n\big) \omega_n\Big]$$

In other words, **sequence-level KL punishes each current token for the divergence of all its future tokens**; token-level KL is only concerned with the current token.

E. Derivation of Sampled Reverse KL Estimator

We focus on a fixed token position n with prefix $h_n = (x, y_{0:n-1})$. Let $p \sim \pi_\theta(\cdot \mid h_n)$ be the sampled token, and define

$$w(p) \triangleq \frac{\pi_{\text{ref}}(p \mid h_n)}{\pi_\theta(p \mid h_n)}, \qquad \ell(p) \triangleq -\log w(p) = \log \frac{\pi_\theta(p \mid h_n)}{\pi_{\text{ref}}(p \mid h_n)}.$$

To avoid ambiguity, throughout this subsection we use the shorthand

$$\mathbb{E}_p\big[\,\cdot\,\big] \;\triangleq\; \mathbb{E}_{p\sim\pi_\theta(\cdot\mid h_n)}\big[\,\cdot\,\big]$$

Reverse KL at a fixed prefix. The token-level reverse KL at history h_n is (with $\mathcal{V}$ being the vocabulary):

$$\text{KL}_n(\theta) \;\triangleq\; \text{KL}\big(\pi_\theta(\cdot \mid h_n) \,\|\, \pi_{\text{ref}}(\cdot \mid h_n)\big) = \sum_{j\in\mathcal{V}} \pi_\theta(j \mid h_n) \log \frac{\pi_\theta(j \mid h_n)}{\pi_{\text{ref}}(j \mid h_n)} = \mathbb{E}_p\big[\ell(p)\big].$$

Exact reverse-KL gradient.

$$\nabla_\theta \text{KL}_n(\theta) = \sum_{j\in\mathcal{V}} \pi_\theta(j) \log \frac{\pi_\theta(j)}{\pi_{\text{ref}}(j)} \nabla_\theta \log \pi_\theta(j) = \mathbb{E}_{p\sim\pi_\theta(\cdot\mid h_n)}\Big[\ell(p)\, \nabla_\theta \log \pi_\theta(p \mid h_n)\Big].$$

$\mathbb{E}_p[\nabla_\theta(\cdot)]$ **versus** $\nabla_\theta \mathbb{E}_p[\cdot]$ Since $\pi_\theta(\cdot \mid h_n)$ depends on θ, $\mathbb{E}_{p\sim\pi_\theta(\cdot\mid h_n)}[\cdot]$ is itself a function of θ; we write it as $\mathbb{E}_p[\cdot]$. For any scalar function $f(p, \theta)$, the gradient $\nabla_\theta \mathbb{E}_p[\cdot]$ is:

$$\mathbb{E}_p\big[f(p,\theta)\big] = \sum_{j\in\mathcal{V}} \pi_\theta(j \mid h_n)\, f(j,\theta) \qquad \nabla_\theta \mathbb{E}_p\big[f(p,\theta)\big] = \mathbb{E}_p\Big[f(p,\theta)\, \nabla_\theta \log \pi_\theta(p \mid h_n) + \nabla_\theta f(p,\theta)\Big]. \tag{75}$$

In contrast, $\mathbb{E}_p[\nabla_\theta f(p, \theta)]$ alone is in general is not equal to $\nabla_\theta \mathbb{E}_p[f]$, as it has only one of the two terms in the full gradient.

E.1. Sampled KL estimators (K1 / K2 / K3) and their expected values

Define the single-sample estimators

$$(\text{K1})_n \triangleq \ell(p), \qquad (\text{K2})_n \triangleq \frac{1}{2}\,\ell(p)^2, \qquad (\text{K3})_n \triangleq \ell(p) + w(p) - 1. \tag{76}$$

Then

$$\mathbb{E}_p\big[(\text{K1})_n\big] = \mathbb{E}_p\big[\ell(p)\big] = \text{KL}_n(\theta), \tag{77}$$

$$\mathbb{E}_p\big[w(p)\big] = \sum_{j\in\mathcal{V}} \pi_\theta(j \mid h_n) \frac{\pi_{\text{ref}}(j \mid h_n)}{\pi_\theta(j \mid h_n)} = \sum_{j\in\mathcal{V}} \pi_{\text{ref}}(j \mid h_n) = 1, \tag{78}$$

$$\mathbb{E}_p\big[(\text{K3})_n\big] = \mathbb{E}_p\big[\ell(p)\big] + \mathbb{E}_p\big[w(p)\big] - 1 = \text{KL}_n(\theta), \tag{79}$$

while in general $\mathbb{E}_p\big[(\text{K2})_n\big] = \frac{1}{2}\mathbb{E}_p[\ell(p)^2] \neq \mathbb{E}_p[\ell(p)]$, so K2 has incorrect expected values.

E.2. Gradients of K1 / K2 / K3 under autodiff $\mathbb{E}_p[\nabla_\theta(\cdot)]$

Here $\nabla_\theta(\cdot)$ denotes the autodiff gradient of the sampled scalar while treating the sampled token p as fixed.

K1: correct value but wrong gradient. Since $(\text{K1})_n = \ell(p) = \log \pi_\theta(p \mid h_n) - \log \pi_{\text{ref}}(p \mid h_n)$ and π_{ref} is fixed,

$$\nabla_\theta (\text{K1})_n = \nabla_\theta \log \pi_\theta(p \mid h_n).$$

Averaging over $p \sim \pi_\theta(\cdot \mid h_n)$ gives

$$\mathbb{E}_p\big[\nabla_\theta (\text{K1})_n\big] = \sum_{j\in\mathcal{V}} \pi_\theta(j \mid h_n) \nabla_\theta \log \pi_\theta(j \mid h_n) = \sum_{j\in\mathcal{V}} \nabla_\theta \pi_\theta(j \mid h_n) = \nabla_\theta 1 = 0,$$

which is zero.

K2: correct reverse-KL gradient but wrong value. We have $(\mathrm{K2})_n = \frac{1}{2}\ell(p)^2$, so

$$\nabla_\theta(\mathrm{K2})_n = \ell(p)\,\nabla_\theta \ell(p) = \ell(p)\,\nabla_\theta \log \pi_\theta(p \mid h_n).$$

Therefore,

$$\mathbb{E}_p\big[\nabla_\theta(\mathrm{K2})_n\big] = \mathbb{E}_p\Big[\ell(p)\,\nabla_\theta \log \pi_\theta(p \mid h_n)\Big]. \tag{80}$$

K3: yields the forward-KL gradient under $\mathbb{E}_p[\nabla_\theta(\cdot)]$. Since $(\mathrm{K3})_n = \ell(p) + w(p) - 1$ and,

$$w(p) = \pi_{\mathrm{ref}}(p \mid h_n)/\pi_\theta(p \mid h_n) \quad \nabla_\theta w(p) = -w(p)\,\nabla_\theta \log \pi_\theta(p \mid h_n),$$
$$\nabla_\theta(\mathrm{K3})_n = \nabla_\theta \ell(p) + \nabla_\theta w(p) = \big(1 - w(p)\big)\,\nabla_\theta \log \pi_\theta(p \mid h_n).$$

Averaging over $p \sim \pi_\theta(\cdot \mid h_n)$ gives

$$\begin{aligned}
\mathbb{E}_p\big[\nabla_\theta(\mathrm{K3})_n\big] &= \mathbb{E}_p\big[\nabla_\theta \log \pi_\theta(p \mid h_n)\big] - \mathbb{E}_p\big[w(p)\,\nabla_\theta \log \pi_\theta(p \mid h_n)\big] \\
&= 0 - \sum_{j \in \mathcal{V}} \pi_\theta(j \mid h_n) \frac{\pi_{\mathrm{ref}}(j \mid h_n)}{\pi_\theta(j \mid h_n)} \nabla_\theta \log \pi_\theta(j \mid h_n) \\
&= -\sum_{j \in \mathcal{V}} \pi_{\mathrm{ref}}(j \mid h_n)\,\nabla_\theta \log \pi_\theta(j \mid h_n) = \nabla_\theta \mathrm{KL}\big(\pi_{\mathrm{ref}}(\cdot \mid h_n) \,\|\, \pi_\theta(\cdot \mid h_n)\big),
\end{aligned} \tag{81}$$

which is the forward-KL gradient, not the reverse-KL gradient.

E.3. The r-trick restores $\nabla_\theta \mathbb{E}_p[\cdot]$ from $\mathbb{E}_p[\nabla_\theta(\cdot)]$.

Define $r(\cdot)$ as:

$$r(p, \theta) \triangleq \frac{\pi_\theta(p \mid h_n)}{\mathrm{sg}\big(\pi_\theta(p \mid h_n)\big)}. \tag{82}$$

In the forward pass $r(p, \theta) = 1$, but in backpropagation the denominator is treated as constant, hence

$$\nabla_\theta r(p, \theta) = r(p, \theta)\,\nabla_\theta \log \pi_\theta(p \mid h_n) = \nabla_\theta \log \pi_\theta(p \mid h_n).$$

For any scalar $f(p, \theta)$, differentiating $r(p, \theta) f(p, \theta)$ gives

$$\nabla_\theta\big(rf\big) = f\,\nabla_\theta r + r\,\nabla_\theta f = f(p, \theta)\,\nabla_\theta \log \pi_\theta(p \mid h_n) + \nabla_\theta f(p, \theta),$$

where we used $r = 1$ in the forward pass so $r\,\nabla_\theta f = \nabla_\theta f$. Taking $\mathbb{E}_p[\cdot]$ on both sides and comparing with (75) yields

$$\mathbb{E}_p\big[\nabla_\theta\big(r(p, \theta) f(p, \theta)\big)\big] = \nabla_\theta \mathbb{E}_p\big[f(p, \theta)\big]. \tag{83}$$

This identity holds even when f depends on θ (via the $\nabla_\theta f$ term).

E.4. K3^{++} yields unbiased reverse-KL values and gradients

Define

$$(\mathrm{K3}^{++})_n \triangleq r(p, \theta)\,(\mathrm{K3})_n = r(p, \theta)\big(\ell(p) + w(p) - 1\big). \tag{84}$$

Value: since $r = 1$ in the forward pass,

$$\mathbb{E}_p\big[(\mathrm{K3}^{++})_n\big] = \mathbb{E}_p\big[(\mathrm{K3})_n\big] = \mathrm{KL}_n(\theta) \quad \text{by (79).} \tag{85}$$

Gradient: by (83) with $f = (\mathrm{K3})_n$,

$$\mathbb{E}_p\big[\nabla_\theta(\mathrm{K3}^{++})_n\big] = \nabla_\theta \mathbb{E}_p\big[(\mathrm{K3})_n\big] = \nabla_\theta \mathrm{KL}_n(\theta), \tag{86}$$

because $\mathbb{E}_p[(\mathrm{K3})_n] = \mathrm{KL}_n(\theta)$.

E.4.1. ALTERNATIVE PROOF

Alternatively, we compute the $\mathrm{K3}^{++}$ gradients in a brute force manner by directly differentiating using the product rule:

$$\nabla_\theta(\mathrm{K3}^{++})_n = \big(\ell + w - 1\big)\,\nabla_\theta r + r\big(\nabla_\theta \ell + \nabla_\theta w\big). \tag{87}$$

Now compute each piece (all conditioned on the sampled p and fixed h_n):

$$\nabla_\theta r = \nabla_\theta \log \pi_\theta(p \mid h_n), \tag{88}$$

$$\nabla_\theta \ell = \nabla_\theta \log \pi_\theta(p \mid h_n), \tag{89}$$

$$\nabla_\theta w = \nabla_\theta \left(\frac{\pi_{\mathrm{ref}}(p \mid h_n)}{\pi_\theta(p \mid h_n)} \right) = -w\,\nabla_\theta \log \pi_\theta(p \mid h_n). \tag{90}$$

Substitute (88)–(90) into (87). Using that $r = 1$ in the forward pass,

$$\nabla_\theta(\mathrm{K3}^{++})_n = \big(\ell + w - 1\big)\,\nabla_\theta \log \pi_\theta(p \mid h_n) + \big(1 - w\big)\,\nabla_\theta \log \pi_\theta(p \mid h_n) = \ell(p)\,\nabla_\theta \log \pi_\theta(p \mid h_n) \tag{91}$$

Taking expectation over the sampled token $p \sim \pi_\theta(\cdot \mid h_n)$ on both sides of (91) yields

$$\mathbb{E}_{p\sim\pi_\theta(\cdot|h_n)}\big[\nabla_\theta(\mathrm{K3}^{++})_n\big] = \mathbb{E}_{p\sim\pi_\theta(\cdot|h_n)}\Big[\ell(p)\,\nabla_\theta \log \pi_\theta(p \mid h_n)\Big] = \nabla_\theta \mathrm{KL}_n(\theta),$$

E.5. K4 yields unbiased reverse-KL values and gradients

Define

$$(\mathrm{K4})_n \triangleq r(p,\theta) \cdot \mathrm{sg}\big((\mathrm{K1})_n\big) = r(p,\theta) \cdot \mathrm{sg}\big(\ell(p)\big). \tag{92}$$

Value: since $r = 1$ in the forward pass,

$$\mathbb{E}_p\big[(\mathrm{K4})_n\big] = \mathbb{E}_p\big[\ell(p)\big] = \mathrm{KL}_n(\theta). \tag{93}$$

Gradient: $\mathrm{sg}(\ell(p))$ is treated as constant in backprop, so differentiating yields

$$\nabla_\theta(\mathrm{K4})_n = \mathrm{sg}\big(\ell(p)\big)\,\nabla_\theta r(p,\theta) = \mathrm{sg}\big(\ell(p)\big)\,\nabla_\theta \log \pi_\theta(p \mid h_n). \tag{94}$$

Taking expectation over $p \sim \pi_\theta(\cdot \mid h_n)$ gives

$$\mathbb{E}_p\big[\nabla_\theta(\mathrm{K4})_n\big] = \mathbb{E}_p\Big[\ell(p)\,\nabla_\theta \log \pi_\theta(p \mid h_n)\Big] = \nabla_\theta \mathrm{KL}_n(\theta), \tag{95}$$

F. Derivation of Sampled Forward KL Estimator

Similar to before, we focus on a fixed token position n with prefix $h_n = (x, y_{0:n-1})$. For brevity, write

$$\pi_\theta(j) \triangleq \pi_\theta(j \mid h_n), \qquad \pi_{\text{ref}}(j) \triangleq \pi_{\text{ref}}(j \mid h_n).$$

At this position we sample a token

$$p \sim \pi_\theta(\cdot \mid h_n),$$

and define the importance ratio and stop-gradient ratio

$$w(p) \triangleq \frac{\pi_{\text{ref}}(p)}{\pi_\theta(p)}, \qquad r(p,\theta) \triangleq \frac{\pi_\theta(p)}{\text{sg}(\pi_\theta(p))}.$$

The expectation over p is also same as before:

$$\mathbb{E}_p\big[\,\cdot\,\big] \triangleq \mathbb{E}_{p\sim\pi_\theta(\cdot\mid h_n)}\big[\,\cdot\,\big]$$

F.1. Forward KL and its gradient

The token-wise forward KL is

$$\text{FKL}_n(\theta) \triangleq \text{KL}\big(\pi_{\text{ref}}(\cdot \mid h_n) \,\|\, \pi_\theta(\cdot \mid h_n)\big) = \sum_{j\in\mathcal{V}} \pi_{\text{ref}}(j) \log \frac{\pi_{\text{ref}}(j)}{\pi_\theta(j)}.$$

Differentiating w.r.t. θ (note π_{ref} does not depend on θ),

$$\nabla_\theta \text{FKL}_n(\theta) = \sum_{j\in\mathcal{V}} \pi_{\text{ref}}(j)\, \nabla_\theta(\log \pi_{\text{ref}}(j) - \log \pi_\theta(j)) = -\sum_{j\in\mathcal{V}} \pi_{\text{ref}}(j)\, \nabla_\theta \log \pi_\theta(j). \tag{96}$$

We can rewrite (96) as an expectation over $p \sim \pi_\theta(\cdot \mid h_n)$ using $w(p)$:

$$-\sum_{j\in\mathcal{V}} \pi_{\text{ref}}(j)\, \nabla_\theta \log \pi_\theta(j) = -\sum_{j\in\mathcal{V}} \pi_\theta(j) \frac{\pi_{\text{ref}}(j)}{\pi_\theta(j)}\, \nabla_\theta \log \pi_\theta(j) = -\mathbb{E}_p\Big[w(p)\, \nabla_\theta \log \pi_\theta(p)\Big]. \tag{97}$$

Finally, we use the score-mean identity $\mathbb{E}_p[\nabla_\theta \log \pi_\theta(p)] = 0$ to obtain an equivalent form:

$$\nabla_\theta \text{FKL}_n(\theta) = -\mathbb{E}_p\Big[w(p)\, \nabla_\theta \log \pi_\theta(p)\Big] = \mathbb{E}_p\Big[(1 - w(p))\, \nabla_\theta \log \pi_\theta(p)\Big]. \tag{98}$$

F.2. The K5 estimator has unbiased value

We propose the forward-KL estimator

$$(\mathbf{K5})_n \triangleq \text{sg}(w(p)) \log w(p) + \log r(p,\theta). \tag{99}$$

For the KL value estimation, note that $r(p,\theta) = 1$ in the forward pass, so $\log r(p,\theta) = 0$ in value. Therefore

$$\begin{aligned}\mathbb{E}_p\big[(\mathbf{K5})_n\big] &= \mathbb{E}_p\big[w(p) \log w(p)\big] = \sum_{j\in\mathcal{V}} \pi_\theta(j) \frac{\pi_{\text{ref}}(j)}{\pi_\theta(j)} \log \frac{\pi_{\text{ref}}(j)}{\pi_\theta(j)} \\ &= \sum_{j\in\mathcal{V}} \pi_{\text{ref}}(j) \log \frac{\pi_{\text{ref}}(j)}{\pi_\theta(j)} = \text{FKL}_n(\theta),\end{aligned} \tag{100}$$

which is the token-wise forward KL.

F.3. The K5 estimator has unbiased gradient

We next show that $\mathbb{E}_p[\nabla_\theta(\mathbf{K5})_n] = \nabla_\theta \mathrm{FKL}_n(\theta)$.

First, compute the gradient of each term in (99). Since $\mathrm{sg}(w(p))$ is treated as a constant in backprop,

$$\nabla_\theta\big(\mathrm{sg}(w(p))\log w(p)\big) = \mathrm{sg}(w(p))\,\nabla_\theta \log w(p).$$

Because $\log w(p) = \log \pi_{\mathrm{ref}}(p) - \log \pi_\theta(p)$ and π_{ref} is fixed,

$$\nabla_\theta \log w(p) = -\nabla_\theta \log \pi_\theta(p) \qquad \nabla_\theta \log r(p,\theta) = \nabla_\theta \log \pi_\theta(p).$$

Combining them yields the per-sample gradient

$$\nabla_\theta(\mathbf{K5})_n = -\mathrm{sg}(w(p))\,\nabla_\theta \log \pi_\theta(p) + \nabla_\theta \log \pi_\theta(p) = (1 - w(p))\,\nabla_\theta \log \pi_\theta(p). \tag{101}$$

Taking expectation over $p \sim \pi_\theta(\cdot \mid h_n)$ gives

$$\mathbb{E}_p\big[\nabla_\theta(\mathbf{K5})_n\big] = \mathbb{E}_p\Big[(1 - w(p))\,\nabla_\theta \log \pi_\theta(p)\Big] = \nabla_\theta \mathrm{FKL}_n(\theta), \tag{102}$$

where the last equality follows from (98).

F.4. Variance reduction in K5 gradients

We start from the exact forward-KL gradient written in the importance-weighted form (Eq. (97)):

$$\nabla_\theta \mathrm{FKL}_n(\theta) = -\mathbb{E}_{p\sim\pi_\theta(\cdot|h_n)}\Big[w(p)\,\nabla_\theta \log \pi_\theta(p \mid h_n)\Big], \qquad w(p) = \frac{\pi_{\mathrm{ref}}(p \mid h_n)}{\pi_\theta(p \mid h_n)}.$$

A key observation is that under the same sampling distribution $p \sim \pi_\theta(\cdot \mid h_n)$, the weight $w(p)$ has mean 1:

$$\mathbb{E}_{p\sim\pi_\theta(\cdot|h_n)}[w(p)] = \sum_{j\in\mathcal{V}} \pi_\theta(j \mid h_n)\frac{\pi_{\mathrm{ref}}(j \mid h_n)}{\pi_\theta(j \mid h_n)} = \sum_{j\in\mathcal{V}} \pi_{\mathrm{ref}}(j \mid h_n) = 1.$$

Therefore, we can subtract this constant baseline from $w(p)$ without changing the expectation of the gradient estimator. Indeed, using the score-mean identity $\mathbb{E}_{p\sim\pi_\theta(\cdot|h_n)}[\nabla_\theta \log \pi_\theta(p \mid h_n)] = 0$, we have

$$\begin{aligned}
-\mathbb{E}_p\Big[w(p)\,\nabla_\theta \log \pi_\theta(p \mid h_n)\Big] &= -\mathbb{E}_p\Big[(w(p) - 1)\,\nabla_\theta \log \pi_\theta(p \mid h_n)\Big] - \underbrace{\mathbb{E}_p\Big[1 \cdot \nabla_\theta \log \pi_\theta(p \mid h_n)\Big]}_{=0} \\
&= \mathbb{E}_p\Big[(1 - w(p))\,\nabla_\theta \log \pi_\theta(p \mid h_n)\Big].
\end{aligned}$$

Thus, replacing $-w(p)$ with $(1 - w(p))$ centered around 0 preserves an unbiased gradient while typically reducing variance, known as the use of a *baseline* in policy gradient methods.

G. Bias-Variance Tradeoff in Top-k KL: An Illustrative Synthetic Setup

This appendix specifies the synthetic setting used to produce Fig. 4. The goal is to isolate the *bias–variance* behavior of the **Top-k Reverse KL** estimator in a controlled regime that reflects how KL penalties are applied to LLM policies, while removing confounders such as long-horizon autoregressive dependencies.

G.1. Token-wise i.i.d. simplification

Let $\pi_\theta(\cdot \mid h_n)$ be an LLM policy over the vocabulary $\mathcal{V}$ at token position n given history $h_n = (x, y_{<n})$, and let $\pi_{\text{ref}}(\cdot \mid h_n)$ be the reference policy. The token-level reverse KL is:

$$\mathrm{KL}\big(\pi_\theta(\cdot \mid h_n) \,\|\, \pi_{\text{ref}}(\cdot \mid h_n)\big) = \sum_{j \in \mathcal{V}} \pi_\theta(j \mid h_n) \log \frac{\pi_\theta(j \mid h_n)}{\pi_{\text{ref}}(j \mid h_n)}. \tag{103}$$

To focus purely on the estimator (rather than the dynamics that produce h_n), we adopt the following simplification:

> **(i.i.d. token-position assumption)** Each token position contributes an i.i.d. draw from a common distribution over (π, π_{ref}), and we analyze a representative position whose ground-truth objective is the reverse KL in (103). In other words, we ignore autoregressive coupling across positions when constructing synthetic tasks.

G.2. Synthetic task and metric

Under this assumption, each token position is treated as a draw of a single-step categorical policy pair (π, π_{ref}) over a fixed vocabulary $\mathcal{V}$. We fix $|\mathcal{V}| = 32000$ and sample

$$\pi_{\text{ref}} = \mathrm{softmax}(\tilde{z}_{\text{ref}}), \qquad \tilde{z}_{\text{ref}} \sim \mathcal{N}(0, I_V), \tag{104}$$

$$\pi = \mathrm{softmax}(s \cdot \tilde{z}), \qquad \tilde{z} \sim \mathcal{N}(0, I_V),\ s > 0. \tag{105}$$

Controlling concentration via top-32 mass. Define the top-32 probability mass

$$M_{32}(\pi) \triangleq \sum_{j \in \mathrm{TopK}(\pi, 32)} \pi(j). \tag{106}$$

We tune the logit scale s so that $M_{32}(\pi)$ matches one of the target values

$$M_{32}(\pi) \approx m, \qquad m \in \{0.2,\ 0.5,\ 0.8,\ 0.9\}, \tag{107}$$

thereby probing a range of entropy regimes from diffuse to highly concentrated policies.

Estimators and error metric. We compare three per-token gradient estimators for the reverse-KL penalty: (i) *sampled-only* reverse KL (fully Monte Carlo), (ii) *truncated-only* (exact top-k head without tail correction; biased), and (iii) *Top-k* reverse KL (exact head plus masked sampled tail; unbiased). We sweep

$$K \in \{4, 8, 16, 32, 64, 128, 256\}, \qquad B \in \{1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768\},$$

where B denotes the number of i.i.d. samples drawn from π ("batch tokens") used by the sampled components of the estimators. For each estimator, we measure the relative root-mean-squared error (RMSE) of the per-token gradient:

$$\mathrm{RelRMSE} \triangleq \frac{\sqrt{\mathbb{E}\big[\|\hat{g} - g_{\text{true}}\|_2^2\big]}}{\|g_{\text{true}}\|_2}, \tag{108}$$

where g_{true} is the exact gradient of (103) with respect to the token logits at that position, and $\hat{g}$ is the corresponding estimator. The expectation in (108) is approximated by Monte Carlo sampling and averaging over many independently generated synthetic tasks.

Critical sample size. For each K, the truncated-only estimator induces a bias floor (its error does not vanish with more samples since it omits the tail). The Top-k estimator is unbiased but has sampling variance from the tail correction. We define the *critical sample size* as the smallest B beyond which Top-k KL achieves lower gradient error than Truncated KL, i.e., the intersection point between the Top-k RMSE curve and the truncated-only bias floor.

H. Top-k f-Divergence Estimator

We generalize the Top-k "truncated exact term + masked sampled tail" construction from KL to a broad class of f-divergences, while retaining (i) unbiased value estimates and (ii) unbiased gradients under the convention $\mathbb{E}[\nabla_\theta(\cdot)]$ (i.e., the expectation of autodiff gradients of sampled scalars).

Setup and notation. Fix a token position n and prefix $h_n = (x, y_{0:n-1})$. Let the current policy at this position be

$$P_\theta(j) \triangleq \pi_\theta(j \mid h_n), \qquad j \in \mathcal{V},$$

and let the reference policy be

$$Q(j) \triangleq \pi_{\text{ref}}(j \mid h_n),$$

which is treated as independent of θ. We use an explicit expectation shorthand:

$$\mathbb{E}_p\big[\,\cdot\,\big] \triangleq \mathbb{E}_{p\sim P_\theta(\cdot)}\big[\,\cdot\,\big]$$

H.1. f-divergence and its gradients

Let $f : \mathbb{R}_+ \to \mathbb{R}$ be convex with $f(1) = 0$. The (discrete) f-divergence is

$$D_f(P_\theta \| Q) \triangleq \sum_{j\in\mathcal{V}} Q(j)\, f\!\left(\frac{P_\theta(j)}{Q(j)}\right). \tag{109}$$

Define the likelihood ratio and its inverse

$$t(j) \triangleq \frac{P_\theta(j)}{Q(j)}, \qquad w(j) \triangleq \frac{Q(j)}{P_\theta(j)} = \frac{1}{t(j)}. \tag{110}$$

Eq. (109) can be written as an expectation under Q:

$$D_f(P_\theta \| Q) = \mathbb{E}_{j\sim Q}\big[f(t(j))\big]. \tag{111}$$

It also admits an equivalent importance-weighted expectation under P_θ. Define

$$g_f(w) \triangleq w\, f(1/w). \tag{112}$$

Then

$$\begin{aligned} D_f(P_\theta \| Q) &= \sum_{j\in\mathcal{V}} Q(j)\, f\!\left(\frac{P_\theta(j)}{Q(j)}\right) = \sum_{j\in\mathcal{V}} P_\theta(j)\, \frac{Q(j)}{P_\theta(j)}\, f\!\left(\frac{P_\theta(j)}{Q(j)}\right) \\ &= \sum_{j\in\mathcal{V}} P_\theta(j)\, g_f(w(j)) = \mathbb{E}_p\big[g_f(w(p))\big], \end{aligned} \tag{113}$$

where $p \sim P_\theta(\cdot)$ and $w(p) = Q(p)/P_\theta(p)$.

H.1.1. Exact gradient w.r.t. θ (reference policy fixed)

Since Q is fixed, differentiating (109) yields

$$\begin{aligned} \nabla_\theta D_f(P_\theta \| Q) &= \sum_{j\in\mathcal{V}} Q(j)\, f'(t(j))\, \nabla_\theta\!\left(\frac{P_\theta(j)}{Q(j)}\right) = \sum_{j\in\mathcal{V}} f'(t(j))\, \nabla_\theta P_\theta(j) \\ &= \sum_{j\in\mathcal{V}} P_\theta(j)\, f'(t(j))\, \nabla_\theta \log P_\theta(j) = \mathbb{E}_p\Big[f'(t(p))\, \nabla_\theta \log P_\theta(p)\Big]. \end{aligned} \tag{114}$$

Eq. (114) is the policy-gradient-shaped form used throughout this appendix.

H.2. A generic, unbiased, sampled f-divergence estimator

We define the following quantities as before:

$$r(p) \triangleq \frac{P_\theta(p)}{\mathrm{sg}\big(P_\theta(p)\big)} \qquad g_f(w(p)) \triangleq w(p)\, f(1/w(p)) \qquad w(p) = \frac{Q(p)}{P_\theta(p)}$$

Then we have our sampled f-divergence estimator, $\mathbf{K}f^{++}$:

$$(\mathbf{K}f^{++})_n \triangleq r(p)\, g_f(w(p)), \qquad p \sim P_\theta(\cdot). \tag{115}$$

Which in effect becomes:

$$\boxed{(\mathbf{K}f^{++})_n = \mathrm{sg}\left(\frac{Q(p)}{P_\theta(p)}\right) f\left(\frac{P_\theta(p)}{Q(p)}\right) = \mathrm{sg}(w)\, f(1/w)} \tag{116}$$

Another way to write this expression is $(\mathbf{K}f^{++})_n = \mathrm{sg}\left(\frac{\pi_{\text{ref}}(p|h_n)}{\pi_\theta(p|h_n)}\right) f\left(\frac{\pi_\theta(p|h_n)}{\pi_{\text{ref}}(p|h_n)}\right)$.

Unbiased value. Since $r = 1$ in the forward pass, from (113) we have

$$\mathbb{E}_p\big[(\mathbf{K}f^{++})_n\big] = \mathbb{E}_p\big[g_f(w(p))\big] = D_f(P_\theta \| Q). \tag{117}$$

Unbiased gradient under $\mathbb{E}[\nabla_\theta(\cdot)]$. Differentiate (115) using product rule:

$$\nabla_\theta(\mathbf{K}f^{++})_n = g_f(w(p))\, \nabla_\theta r(p) + r(p)\, \nabla_\theta g_f(w(p)) = g_f(w(p))\, \nabla_\theta \log P_\theta(p) + \nabla_\theta g_f(w(p)) \tag{118}$$

Taking expectation over $p \sim P_\theta(\cdot)$ gives

$$\mathbb{E}_p\big[\nabla_\theta(\mathbf{K}f^{++})_n\big] = \nabla_\theta \mathbb{E}_p\big[g_f(w(p))\big] = \nabla_\theta D_f(P_\theta \| Q), \tag{119}$$

where the last equality follows from (113). Thus $(\mathbf{K}f^{++})_n$ provides both unbiased values and unbiased gradients for any f-divergence.

H.3. Top-k f-divergence estimator: A generalization of Top-k KL estimator

Let $S \subseteq \mathcal{V}$ be any index set of size K (e.g., $S = \mathrm{Topk}(P_\theta)$ or $S = \mathrm{Topk}(Q)$). Being an unbiased estimator does not require S to be Top-K; Top-K is a variance/compute heuristic.

Truncated exact term. Define the truncated f-divergence (computed exactly given logits on S):

$$\boxed{(\widehat{D}_{f,\text{trun}})_n \triangleq \sum_{j \in S} Q(j)\, f\left(\frac{P_\theta(j)}{Q(j)}\right)} \tag{120}$$

Masked sampled tail term. Sample $p \sim P_\theta(\cdot)$ and define the masked tail estimator

$$\boxed{(\widehat{D}_{f,\text{tail}})_n \triangleq \mathbf{1}\{p \notin S\}\, (\mathbf{K}f^{++})_n} = \mathbf{1}\{p \notin S\}\, r(p)\, g_f\left(\frac{Q(p)}{P_\theta(p)}\right) \tag{121}$$

Top-k f-divergence estimator. We define the Top-K f-divergence estimator

$$\boxed{(\text{TopK-}D_f)_n \triangleq (\widehat{D}_{f,\text{trun}})_n + (\widehat{D}_{f,\text{tail}})_n} \tag{122}$$

Unbiased value. Using (113) and the definition of g_f,

$$\mathbb{E}_p\big[(\widehat{D}_{f,\text{tail}})_n\big] = \sum_{j \in \mathcal{V}} P_\theta(j)\, \mathbf{1}\{j \notin S\}\, g_f\left(\frac{Q(j)}{P_\theta(j)}\right) = \sum_{j \notin S} Q(j)\, f\left(\frac{P_\theta(j)}{Q(j)}\right), \tag{123}$$

so

$$\mathbb{E}_p\big[(\text{TopK-}D_f)_n\big] = (\widehat{D}_{f,\text{trun}})_n + \mathbb{E}_p\big[(\widehat{D}_{f,\text{tail}})_n\big] = D_f(P_\theta \| Q). \tag{124}$$

Unbiased gradient. Since $(\widehat{D}_{f,\mathrm{trun}})_n$ is computed exactly on S, it has the correct gradient contribution from indices in S. For the tail term,

$$\mathbb{E}_p\big[\nabla_\theta(\widehat{D}_{f,\mathrm{tail}})_n\big] = \mathbb{E}_p\big[\mathbf{1}\{p \notin S\}\,\nabla_\theta(\mathbf{K}f^{++})_n\big] = \nabla_\theta \sum_{j\notin S} Q(j)\, f\left(\frac{P_\theta(j)}{Q(j)}\right), \tag{125}$$

where the last step follows by expanding the expectation as a sum over $j \notin S$ and using the property (119) of our sampled f-divergence estimator $\mathbf{K}f^{++}$ restricted to the tail indices. Therefore,

$$\mathbb{E}_p\big[\nabla_\theta(\mathrm{TopK}\text{-}D_f)_n\big] = \nabla_\theta D_f(P_\theta \| Q). \tag{126}$$

A note on variance reduction in Forward and Reverse KL gradient estimators.

- The Forward KL estimator derived from the standard f-divergence formula in Eq (116) is $\mathrm{sg}(w)\log w$, which differs from our **K5** estimator $\mathrm{sg}(w)\log w + \log r$. This is explained in §F.4: the additional term $\log r$ is added as a *baseline* to reduce gradient variance, where the baseline is $\mathbb{E}[\mathrm{sg}(w)] = 1$.
- The Reverse KL estimator derived from Eq (116) is $r \cdot (-\log w)$; this also differs from our **K4** estimator $r \cdot \mathrm{sg}(-\log w)$. The reason is similar: for reverse KL, the baseline is $\mathbb{E}[-\log w] = 0$, and $\mathbb{E}[-\nabla_\theta \log w] = \mathbb{E}[\nabla_\theta \log \pi_\theta(y|x)] = 0$. As a result, doing a stop gradient operation on $(-\log w)$ keeps the gradients unbiased while reducing variance, so we do a stop gradient on the $(-\log w)$ term in **K4**.

Table 5. Top-k f-divergence estimators. **Let** $P_\theta(j) = \pi_\theta(j|h_n),\ Q(j) = \pi_{\mathrm{ref}}(j|h_n),\ t_j = P_\theta(j)/Q(j)$. Sample $p \sim P_\theta(\cdot)$. We compute $r = P_\theta(p)/\mathrm{sg}(P_\theta(p))$, $w = Q(p)/P_\theta(p)$. Note that $\mathrm{sg}(w)/w = r$.

	$\boldsymbol{f(t)}$ (so $D_f = \sum_j Q(j)\, f(t_j)$)	**Sampled f-divergence estimator** $= \mathrm{sg}(w)\, f(1/w)$
Forward KL	$-\log t$	$\mathrm{sg}(w) \cdot \log w$
Reverse KL	$t \log t$	$r \cdot (-\log w)$
Pearson χ^2	$(t-1)^2$	$r \cdot \frac{(1-w)^2}{w}$
Neyman χ^2	$\frac{(t-1)^2}{t}$	$r \cdot (1-w)^2$
Squared Hellinger	$\frac{1}{2}(\sqrt{t}-1)^2$	$r \cdot \frac{1}{2}(1-\sqrt{w})^2$
Jensen–Shannon (JS)	$\frac{1}{2}\left(t\log t - (t+1)\log\left(\frac{t+1}{2}\right)\right)$	$r \cdot \frac{1}{2}\left[-\log w - (1+w)\log\left(\frac{1+w}{2w}\right)\right]$
Total variation (TV)	$\frac{1}{2}\lvert t-1\rvert$	$r \cdot \frac{1}{2}\lvert 1-w(p)\rvert$
α-divergence	$\frac{t^\alpha - 1 - \alpha(t-1)}{\alpha(\alpha-1)},\ \alpha \neq 0, 1$	$r \cdot \frac{w^{1-\alpha} + (\alpha-1)w - \alpha}{\alpha(\alpha-1)}$

I. Optimal Policies under Forward vs Reverse KL Regularization

We now study the optimal policies under different types of regularizers and see if the optimal policies are interchangeable.

The RL objectives under reverse and forward KL regularization are:

$$J = \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[R(x,y)] - \beta\,\mathrm{KL}(\pi_\theta(\cdot \mid x)\|\pi_{\text{ref}}(\cdot \mid x)) \tag{127}$$

$$J_f = \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[R(x,y)] - \beta\mathrm{KL}(\pi_{\text{ref}}(\cdot \mid x)\|\pi_\theta(\cdot \mid x)) \tag{128}$$

Optimal Policy under Reverse KL It is well known that the optimal policy for objective (127) is (Peters et al., 2010; Schulman et al., 2017a; Abdolmaleki et al., 2018):

$$\boxed{\pi^*(y|x) = \pi_{\text{ref}}(y|x)\,\frac{\exp(R(x,y)/\beta)}{Z(x)}} \qquad Z(x) = \sum\nolimits_y \pi_{\text{ref}}(y|x)\exp(R(x,y)/\beta) \tag{129}$$

Optimal Policy under Forward KL We derive the optimal policy for objective (128). Expanding the forward KL:

$$J_f = \sum\nolimits_y \pi_\theta(y|x)R(x,y) + \beta\sum\nolimits_y \pi_{\text{ref}}(y|x)\log\pi_\theta(y|x) + \text{const} \tag{130}$$

where the constant absorbs terms independent of π_θ. This objective decomposes into an RL term (first sum) and a behavioral cloning (BC) term (second sum). Forming the Lagrangian:

$$\mathcal{L} = \sum\nolimits_y \pi_\theta(y|x)R(x,y) + \beta\sum\nolimits_y \pi_{\text{ref}}(y|x)\log\pi_\theta(y|x) - \lambda\left(\sum\nolimits_y \pi_\theta(y|x) - 1\right)$$

$$\frac{\partial\mathcal{L}}{\partial\pi_\theta(y|x)} = R(x,y) + \beta\,\frac{\pi_{\text{ref}}(y|x)}{\pi_\theta(y|x)} - \lambda = 0$$

$$\pi_\theta(y \mid x) = \frac{\beta\,\pi_{\text{ref}}(y \mid x)}{\lambda - R(x,y)}$$

For this to be a valid probability distribution, we need $\lambda > \max_y R(x,y)$. Denote $Z^*(x) := \lambda$, we require $Z^*(x)$ to satisfy:

$$\sum\nolimits_y \frac{\pi_{\text{ref}}(y \mid x)}{Z^*(x) - R(x,y)} = \frac{1}{\beta} \tag{131}$$

Equation (131) implicitly defines $Z^*(x)$ as a function of the reward distribution as well. The optimal policy is:

$$\boxed{\pi_f^*(y \mid x) = \pi_{\text{ref}}(y \mid x)\,\frac{\beta}{Z^*(x) - R(x,y)}} \tag{132}$$

A similar result was obtained in (Grill et al., 2020a) under a different context.

I.1. Reward transformation for equivalent optima between forward KL and reverse KL

We now derive the reward transformation that makes the two objectives yield the same optimal policy.

$$\pi^*(y \mid x) = \pi_f^*(y \mid x) \qquad \frac{\pi_{\text{ref}}(y|x)\exp(\tilde{R}/\beta)}{Z(x)} = \frac{\beta\,\pi_{\text{ref}}(y \mid x)}{Z^*(x) - R(x,y)} \tag{133}$$

Therefore, the optimal policy under reward function R and forward KL regularizer is equivalent to the counterpart under reward function $\tilde{R}$ and reverse KL regularizer:

$$\text{Forward KL (with } R) \to \text{Reverse KL (with } \tilde{R}) \qquad \boxed{\tilde{R} = -\beta\log(Z^* - R) + \beta\log(\beta Z)} \tag{134}$$

Proposition I.1. *The function $\tilde{R} = -\beta\log(Z^* - R) + \text{const}$ is strictly monotonically increasing and strictly convex in R.*

Proof. Taking the derivative with respect to R (treating Z^* as constant): $\frac{\partial\tilde{R}}{\partial R} = -\beta\cdot\frac{-1}{Z^*-R} = \frac{\beta}{Z^*-R} > 0$. The last step was due to $Z^* > \max_y R(x,y)$ by construction. Thus $\tilde{R}$ is strictly monotonically increasing in R.

Taking the second derivative: $\frac{\partial^2\tilde{R}}{\partial R^2} = \frac{\partial}{\partial R}\left(\frac{\beta}{Z^*-R}\right) = \beta\cdot\frac{1}{(Z^*-R)^2} = \frac{\beta}{(Z^*-R)^2} > 0$. Thus $\tilde{R}$ is strictly convex in R. □

Inverse transformation: Given a reward R used in the reverse KL objective J, what reward R_f should be used in the forward KL objective J_f to obtain the same optimal policy?

Setting $\pi^*(y|x) = \pi_f^*(y|x)$, we have $\exp(R/\beta)/Z = \beta/(Z^* - R_f)$, which simplifies to:

$$\text{Reverse KL (with } R) \to \text{Forward KL (with } R_f) \qquad \boxed{R_f = Z^* - \beta Z \exp(-R/\beta)} \tag{135}$$

Proposition I.2. *The function $R_f = Z^* - \beta Z \exp(-R/\beta)$ is strictly monotonically increasing and strictly concave in R.*

Proof. Taking first-order derivative $\frac{\partial R_f}{\partial R} = -\beta Z \cdot \left(-\frac{1}{\beta}\right) \exp(-R/\beta) = Z \exp(-R/\beta) > 0$; therefore, R_f is strictly monotonically increasing (preserving reward ordering).

The second-order derivative is $\frac{\partial^2 R_f}{\partial R^2} = Z \cdot \left(-\frac{1}{\beta}\right) \exp(-R/\beta) = -\frac{Z}{\beta} \exp(-R/\beta) < 0$, we find that R_f is strictly concave (compressing high rewards). □

Direction	Transformation	Monotonic?	Curvature	Effect
Forward KL → Reverse KL	$-\beta \log(Z^* - R) + \text{const}$	Yes	Convex	Amplify high rewards
Reverse KL → Forward KL	$Z^* - \beta Z e^{-R/\beta}$	Yes	Concave	Compress high rewards

I.2. Reward transformation for equivalent optima across f-divergences

We now generalize beyond KL divergence to arbitrary f-divergence regularization. Consider the objective:

$$J_f = \mathbb{E}_{y \sim \pi_\theta(\cdot|x)}[R(x,y)] - \beta\, D_f(\pi_\theta(\cdot|x) \| \pi_{\text{ref}}(\cdot|x)) \tag{136}$$

where the f-divergence is defined as $D_f(P\|Q) = \sum_y Q(y)\, f\left(\frac{P(y)}{Q(y)}\right)$ where $f : \mathbb{R}^+ \to \mathbb{R}$ is a convex function with $f(1) = 0$. Standard KKT conditions yield the optimal policy:

$$\boxed{\pi_f^*(y \mid x) = \pi_{\text{ref}}(y \mid x) \cdot (f')^{-1}\left(\frac{R(x,y) - \lambda(x)}{\beta}\right)} \tag{137}$$

where $\lambda(x)$ is determined by the normalization constraint:

$$\sum_y \pi_{\text{ref}}(y \mid x) \cdot (f')^{-1}\left(\frac{R(x,y) - \lambda(x)}{\beta}\right) = 1 \tag{138}$$

which was previously derived in (Wang et al., 2023).

Given two f-divergences with f_1 and f_2, the optimal policies $\pi_{f_1}^*$ and $\pi_{f_2}^*$ coincide when:

$$(f_1')^{-1}\left(\frac{R_1 - \lambda_1}{\beta}\right) = (f_2')^{-1}\left(\frac{R_2 - \lambda_2}{\beta}\right) \tag{139}$$

This yields the general reward transformation:

$$\boxed{R_2 = \beta \cdot f_2'\left((f_1')^{-1}\left(\frac{R_1 - \lambda_1}{\beta}\right)\right) + \lambda_2} \tag{140}$$

The curvature of this transformation is determined by the composition $(f_2') \circ (f_1')^{-1}$.

Table 6. **Optimal Policy form** under the each f-divergence regularization.

Divergence	$f(t)$	$f'(t)$	$(f')^{-1}(s)$	Optimal Policy Form
Reverse KL	$t\log t - t + 1$	$\log t$	e^s	$\pi_{\text{ref}} \cdot \exp\big((R-\lambda)/\beta\big)$
Forward KL	$-\log t + t - 1$	$1 - \frac{1}{t}$	$\frac{1}{1-s}$	$\pi_{\text{ref}} \cdot \frac{1}{1-(R-\lambda)/\beta}$
Jensen-Shannon	$\frac{1}{2}\Big[t\log t - (t+1)\log\Big(\frac{t+1}{2}\Big)\Big]$	$\frac{1}{2}\log\Big(\frac{2t}{t+1}\Big)$	$\frac{1}{2e^{-2s}-1}$	$\pi_{\text{ref}} \cdot \frac{\exp\big(2(R-\lambda)/\beta\big)}{2-\exp\big(2(R-\lambda)/\beta\big)}$
Squared Hellinger	$(\sqrt{t}-1)^2 = t - 2\sqrt{t} + 1$	$1 - \frac{1}{\sqrt{t}}$	$\frac{1}{(1-s)^2}$	$\pi_{\text{ref}} \cdot \frac{1}{\big(1-(R-\lambda)/\beta\big)^2}$
Pearson χ^2	$\frac{1}{2}(t-1)^2$	$t-1$	$s+1$	$\pi_{\text{ref}} \cdot \Big(1 + \frac{R-\lambda}{\beta}\Big)$
Neyman χ^2	$\frac{(t-1)^2}{t} = t - 2 + \frac{1}{t}$	$1 - \frac{1}{t^2}$	$\frac{1}{\sqrt{1-s}}$	$\pi_{\text{ref}} \cdot \frac{1}{\sqrt{1-(R-\lambda)/\beta}}$
α-divergence	$\frac{4}{1-\alpha^2}\Big(1 - t^{\frac{1+\alpha}{2}}\Big)$	$-\frac{2}{1-\alpha}\, t^{\frac{\alpha-1}{2}}$	$\big(-\frac{1-\alpha}{2}\, s\big)^{\frac{2}{\alpha-1}}$	$\pi_{\text{ref}} \cdot \Big(-\frac{1-\alpha}{2} \cdot \frac{R-\lambda}{\beta}\Big)^{\frac{2}{\alpha-1}}$

Table 7. **Reward transformations that produce the *same optimal policy* as each f-divergence under Reverse KL.** Let λ and $\widetilde{\lambda}$ be normalization constants enforcing $\sum_y \pi^*(y \mid x) = 1$ for the f-divergence and Reverse KL objectives, respectively. Then the equivalence is achieved by $\widetilde{R}(x,y) = \beta\log\big((f')^{-1}((R-\lambda)/\beta)\big) + \widetilde{\lambda}$, for each $f(\cdot)$ in f-divergence.

Divergence f (target)	$(f')^{-1}(s)$	**Reverse-KL reward transform** $\widetilde{R}(x,y)$	Ranges of $\lambda(x)$
Forward KL	$\dfrac{1}{1-s}$	$-\beta\log\Big(1 - \dfrac{R-\lambda}{\beta}\Big) + \widetilde{\lambda}$	$\lambda > R - \beta$
Jensen–Shannon	$\dfrac{1}{2e^{-2s}-1}$	$-\beta\log\Big(2\exp\big(-2\dfrac{R-\lambda}{\beta}\big) - 1\Big) + \widetilde{\lambda}$	$\lambda > R - \frac{\beta}{2}\log 2$
Squared Hellinger	$\dfrac{1}{(1-s)^2}$	$-2\beta\log\Big(1 - \dfrac{R-\lambda}{\beta}\Big) + \widetilde{\lambda}$	$\lambda > R - \beta$
Pearson χ^2	$1+s$	$\beta\log\Big(1 + \dfrac{R-\lambda}{\beta}\Big) + \widetilde{\lambda}$	$\lambda < R + \beta$
Neyman χ^2	$\dfrac{1}{\sqrt{1-s}}$	$-\dfrac{\beta}{2}\log\Big(1 - \dfrac{R-\lambda}{\beta}\Big) + \widetilde{\lambda}$	$\lambda > R - \beta$
α-divergence	$\Big(-\dfrac{1-\alpha}{2}\, s\Big)^{\frac{2}{\alpha-1}}$	$\dfrac{2\beta}{\alpha-1}\log\Big(-\dfrac{1-\alpha}{2} \cdot \dfrac{R-\lambda}{\beta}\Big) + \widetilde{\lambda}$	$-(1-\alpha)(R-\lambda) > 0$

J. Policy Gradient Methods can be Derived from Sampled KL Estimators

If RL optimization is simply written as closing the reverse KL between current policy π_θ and the reference policy distribution π_{ref} transformed by the return R (Peters & Schaal, 2007; Schulman et al., 2017a; Haarnoja et al., 2018), we have:

$$\min_\theta \mathrm{KL}\left(\pi_\theta(\cdot|x) \middle|\middle| \frac{1}{Z(x)}\pi_{\text{ref}}(\cdot|x)\exp\Big(R(x,\cdot)\Big)\right) \tag{141}$$

to make the connection with sampled KL estimators clear, we define the ratio between the optimal policy π^* and current policy π_θ as w, same as before:

$$w = \frac{\pi_{\text{ref}}(y|x)\,\exp R(x,y)}{\pi_\theta(y|x)\,Z(x)} \tag{142}$$

where the partition function $Z(x)$ is approximated via a group of N trajectories per prompt $y^{(i)} \sim \pi_\theta(\cdot \mid x)$ for $i = 1, \ldots, N$:

$$\log Z(x) \triangleq \log \mathbb{E}_{\pi_{\text{ref}}(y|x)}[\exp R(x,y)] \approx \log \frac{1}{N}\sum_{i=1}^{N} \mathrm{sg}\left(\frac{\pi_{\text{ref}}(y^{(i)} \mid x)}{\pi_\theta(y^{(i)} \mid x)}\right)\exp\Big(R(x,y^{(i)})\Big) \tag{143}$$

$$\geq \frac{1}{N}\sum_{i=1}^{N}\left(R(x,y^{(i)}) - \mathrm{sg}\left(\log\frac{\pi_\theta(y^{(i)} \mid x)}{\pi_{\text{ref}}(y^{(i)} \mid x)}\right)\right) \tag{144}$$

where the last step uses Jensen's inequality. We split it into two terms, value function ($V(x)$) and aggregate KL ($\widehat{\mathrm{KL}}$):

$$V(x) \triangleq \mathbb{E}_{\substack{y^{(i)}\sim\pi_\theta(\cdot|x)\\ i=1,\ldots,N}}\left[\frac{1}{N}\sum_{i=1}^{N} R(x,y^{(i)})\right] \qquad \widehat{\mathrm{KL}} \triangleq \mathbb{E}_{\substack{y^{(i)}\sim\pi_\theta(\cdot|x)\\ i=1,\ldots,N}}\left[\frac{1}{N}\sum_{i=1}^{N}\mathrm{sg}\left(\log\frac{\pi_\theta(y^{(i)} \mid x)}{\pi_{\text{ref}}(y^{(i)} \mid x)}\right)\right] \tag{145}$$

As a side note, *to sharpen or flatten the target distribution, we simply need to multiply all rewards by* $1/\beta$; as a result, we omit β in our derivation and let it be absorbed by $R(x,y)$.

Next, we show that **every sampled KL estimator in Tab 1 can be turned into a policy gradient algorithm**.

J.1. K1 estimator leads to on-policy PPO / GRPO loss

We start with $\mathbf{K1} = -\log w$:

$$\text{K1:} \quad \min_\theta -\log w \qquad \nabla_\theta L_1 = -\nabla_\theta \log \pi_\theta(y|x)\Big(R(x,y) - \log Z(x)\Big)$$

$$= -\nabla_\theta \log \pi_\theta(y|x)\left(R(x,y) - V(x) + \frac{1}{N}\sum_{i=1}^{N}\mathrm{sg}\left(\log\frac{\pi_\theta(y^{(i)} \mid x)}{\pi_{\text{ref}}(y^{(i)} \mid x)}\right)\right)$$

Summing over all N rollouts, we get

$$\underbrace{-\sum_{j=1}^{N}\nabla_\theta \log \pi_\theta(y^{(j)} \mid x)\Big(R(x,y^{(j)}) - V(x)\Big)}_{\text{REINFORCE}} + \underbrace{\sum_{j=1}^{N}\nabla_\theta \log \pi_\theta(y^{(j)} \mid x)\left(\frac{1}{N}\sum_{i=1}^{N}\mathrm{sg}\left(\log\frac{\pi_\theta(y^{(i)} \mid x)}{\pi_{\text{ref}}(y^{(i)} \mid x)}\right)\right)}_{\text{The regularizer gradient } \mathbb{E}_{y^{(j)}\sim\pi_\theta}[\sum_{j=1}^{N}\nabla_\theta \log \pi_\theta(y^{(j)} \mid x)\widehat{\mathrm{KL}}]}$$

Lemma J.1. *The regularizer gradient* $\mathbb{E}_{y^{(j)}\sim\pi_\theta}[\sum_{j=1}^{N}\nabla_\theta \log \pi_\theta(y^{(j)} \mid x)\widehat{\mathrm{KL}}]$ *is equivalent to the expected KL gradient:*

$$\mathbb{E}_{\substack{y^{(j)}\sim\pi_\theta(\cdot|x)\\ j=1,\ldots,N}}\left[\sum_{j=1}^{N}\nabla_\theta \log \pi_\theta(y^{(j)} \mid x)\,\widehat{\mathrm{KL}}\right] = \mathbb{E}_{\substack{y^{(j)}\sim\pi_\theta(\cdot|x)\\ j=1,\ldots,N}}\left[\frac{1}{N}\sum_{j=1}^{N}\nabla_\theta \log \pi_\theta(y^{(j)} \mid x)\,\mathrm{sg}\left(\log\frac{\pi_\theta(y^{(j)}|x)}{\pi_{ref}(y^{(j)}|x)}\right)\right] \tag{146}$$

$$= \nabla_\theta \mathrm{KL}(\pi_\theta(\cdot \mid x) \parallel \pi_{ref}(\cdot \mid x))$$

Proof. Expanding the product of sums gives

$$\mathbb{E}_{\substack{y^{(j)}\sim\pi_\theta(\cdot|x)\\ j=1,\dots,N}}\left[\sum_{j=1}^{N}\nabla_\theta\log\pi_\theta(y^{(j)}\mid x)\,\widehat{\mathrm{KL}}\right]=\mathbb{E}_{\substack{y^{(j)}\sim\pi_\theta(\cdot|x)\\ j=1,\dots,N}}\left[\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N}\nabla_\theta\log\pi_\theta(y^{(j)}\mid x)\,\mathrm{sg}\left(\log\frac{\pi_\theta(y^{(i)}\mid x)}{\pi_{\text{ref}}(y^{(i)}\mid x)}\right)\right]$$

which can be written as two components:

$$\begin{aligned}&\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N}\nabla_\theta\log\pi_\theta(y^{(j)}\mid x)\,\mathrm{sg}\left(\log\frac{\pi_\theta(y^{(i)}\mid x)}{\pi_{\text{ref}}(y^{(i)}\mid x)}\right)\\ &=\underbrace{\frac{1}{N}\sum_{j=1}^{N}\nabla_\theta\log\pi_\theta(y^{(j)}\mid x)\,\mathrm{sg}\left(\log\frac{\pi_\theta(y^{(j)}\mid x)}{\pi_{\text{ref}}(y^{(j)}\mid x)}\right)}_{=\,\mathbb{E}_{y\sim\pi_\theta}\left[\nabla_\theta\log\pi_\theta(y\mid x)\,\mathrm{sg}\left(\log\frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}\right)\right]}+\underbrace{\frac{1}{N}\left(\sum_{j=1}^{N}\nabla_\theta\log\pi_\theta(y^{(j)}\mid x)\right)\left(\sum_{i\neq j}^{N}\mathrm{sg}\left(\log\frac{\pi_\theta(y^{(i)}\mid x)}{\pi_{\text{ref}}(y^{(i)}\mid x)}\right)\right)}_{\text{has an expected value }=0\text{, since }\mathbb{E}_{y\sim\pi_\theta}[\nabla_\theta\log\pi_\theta(y\mid x)]=0}\end{aligned}$$

For $i\neq j$, independence of samples and the score-function identity $\mathbb{E}_{y\sim\pi_\theta}[\nabla_\theta\log\pi_\theta(y\mid x)]=0$ imply that these off-diagonal terms have zero expectation. The only non-vanishing contribution comes from the diagonal $i=j$, yielding

$$\mathbb{E}_{y\sim\pi_\theta}\left[\nabla_\theta\log\pi_\theta(y\mid x)\,\mathrm{sg}\left(\log\frac{\pi_\theta(y\mid x)}{\pi_{\text{ref}}(y\mid x)}\right)\right]=\nabla_\theta\mathrm{KL}(\pi_\theta(\cdot\mid x)\parallel\pi_{\text{ref}}(\cdot\mid x)).$$

Thus, the regularizer gradient is equivalent to the common KL gradient. □

Therefore, L_1 is simply a typical KL-regularized policy gradient objective:

$$\text{K1:}\quad \min_\theta -\log w \qquad \boxed{\nabla_\theta L_1=-\nabla_\theta\log\pi_\theta(y\mid x)\Big(R(x,y)-V(x)\Big)+\nabla_\theta\mathrm{KL}(\pi_\theta(\cdot\mid x)\parallel\pi_{\text{ref}}(\cdot\mid x))} \tag{147}$$

Which is what typical policy gradient algorithms like PPO (Schulman et al., 2017b) and GRPO (Shao et al., 2024) optimize in the on-policy case.

J.2. K2 estimator leads to APA / Kimi loss

Using the K2 estimator:

$$\text{K2:}\qquad \min_\theta\frac{1}{2}(\log w)^2 \qquad\qquad \nabla_\theta L_2=\nabla_\theta\left(\log\pi_\theta(y|x)-\log\pi_{\text{ref}}(y|x)-R(x,y)+\log Z(x)\right)^2$$

Expanding the partition function:

$$\begin{aligned}&\nabla_\theta\left(\log\pi_\theta(y|x)-\log\pi_{\text{ref}}(y|x)-(R(x,y)-V(x))+\widehat{\mathrm{KL}}\right)^2\\ &=\underbrace{\nabla_\theta\left(\log\pi_\theta(y|x)-\log\pi_{\text{ref}}(y|x)-(R(x,y)-V(x))\right)^2}_{\text{APA Loss}}+\underbrace{\nabla_\theta\log\pi_\theta(y\mid x)\,\widehat{\mathrm{KL}}}_{\substack{\text{Becomes }\nabla_\theta\mathrm{KL}(\pi_\theta(\cdot\mid x)\parallel\pi_{\text{ref}}(\cdot\mid x))\\ \text{when summed over }N\text{ trajectories (Lemma J.1)}}}\end{aligned}$$

The loss L_2 is the Advantage-induced Policy Alignment (**APA**) (Zhu et al., 2023), which was also used by Kimi models (Kimi et al., 2025b). The second term was often ignored, since prior works often approximate $\log Z(x)\approx V(x)$ rather than $\log Z(x)\approx V(x)-\widehat{\mathrm{KL}}$. The APA loss already incorporates the same KL term:

$$\nabla_\theta(\log\pi_\theta(y|x)-\log\pi_{\text{ref}}(y|x)-(R(x,y)-V(x)))^2=\nabla_\theta\log\pi_\theta(y|x)(R(x,y)-V(x))+\nabla_\theta\mathrm{KL}(\pi_\theta(\cdot\mid x)\|\pi_{\text{ref}}(\cdot\mid x))$$

Which means that the additional $\nabla_\theta\log\pi_\theta(y\mid x)\,\widehat{\mathrm{KL}}$ term just upweights the KL. Both APA and Kimi showed that this loss is effective in the off-policy regime without necessarily adding off-policy correction:

$$\text{K2:}\qquad \min_\theta\frac{1}{2}(\log w)^2 \qquad \boxed{\nabla_\theta L_2=\nabla_\theta\Big(\log\pi_\theta(y|x)-\log\pi_{\text{ref}}(y|x)-(R(x,y)-V(x))\Big)^2} \tag{148}$$

J.3. K5 estimator leads to RWR / MPO loss

Under the K5 estimator, the divergence minimization in policy optimization becomes:

$$\text{K5:} \quad \min_\theta \operatorname{sg}(w)\log w + \log r \qquad \nabla_\theta L_5 = -\nabla_\theta \log \pi_\theta(y|x)\left(\frac{\pi_{\text{ref}}(y|x)\,\exp R(x,y)}{\pi_\theta(y|x)\,Z(x)}\right)$$

Averaging over all N rollouts per x to simplify the partition function:

$$\begin{aligned}
&-\frac{1}{N}\sum_{i=1}^{N}\nabla_\theta \log \pi_\theta(y^{(i)} \mid x)\left(\frac{\pi_{\text{ref}}(y^{(i)} \mid x)\,\exp R(x,y^{(i)})}{\pi_\theta(y^{(i)} \mid x)\,Z(x)}\right)\\
=&-\frac{1}{N}\sum_{i=1}^{N}\nabla_\theta \log \pi_\theta(y^{(i)} \mid x)\left(\frac{\operatorname{sg}\left(\frac{\pi_{\text{ref}}(y^{(i)}|x)}{\pi_\theta(y^{(i)}|x)}\right)\exp R(x,y^{(i)})}{\frac{1}{N}\sum_{j=1}^{N}\operatorname{sg}\left(\frac{\pi_{\text{ref}}(y^{(j)}|x)}{\pi_\theta(y^{(j)}|x)}\right)\exp R(x,y^{(j)})}\right)\\
=&-\sum_{i=1}^{N}\nabla_\theta \log \pi_\theta(y^{(i)} \mid x)\cdot \operatorname{sg}\underbrace{\left(\frac{\exp\left(R(x,y^{(i)})-\log\frac{\pi_\theta(y^{(i)}|x)}{\pi_{\text{ref}}(y^{(i)}|x)}\right)}{\sum_{j=1}^{N}\exp\left(R(x,y^{(j)})-\log\frac{\pi_\theta(y^{(j)}|x)}{\pi_{\text{ref}}(y^{(j)}|x)}\right)}\right)}_{\text{softmax over } [y^{(1)},\ldots,y^{(N)}]}
\end{aligned}$$

Which can be written in a batched manner over a group of N rollouts per prompt $\boldsymbol{y} = [y^{(1)},\ldots,y^{(N)}]$, where mean and $\mathrm{softmax}$ are taken along the axis of the group dimension $i = 1,\ldots,N$ (which we denote as mean_y and $\mathrm{softmax}_y$):

$$\mathrm{mean}_y\left(\nabla_\theta \log \pi_\theta(\boldsymbol{y} \mid x)\cdot \mathrm{softmax}_y\left(R(x,\boldsymbol{y}) - \log\frac{\pi_\theta(\boldsymbol{y} \mid x)}{\pi_{\text{ref}}(\boldsymbol{y} \mid x)}\right)\right)$$

This loss closely resembles Reward-Weighted Regression (**RWR**) (Peters & Schaal, 2007) and Maximum a Posteriori Policy Optimisation (**MPO**) (Abdolmaleki et al., 2018).

$$\text{K5:} \quad \min_\theta \operatorname{sg}(w)\log w + \log r \qquad \boxed{\nabla_\theta L_5 = -\nabla_\theta \log \pi_\theta(y|x)\cdot \mathrm{softmax}_y\Big(R(x,y) - \log(\pi_\theta(y|x)/\pi_{\text{ref}}(y|x))\Big)} \tag{149}$$

J.4. K3 / K3^{++} / K4 estimators lead to combined losses

The K3 estimator leads to a combined L_1 and L_5 (GRPO + MPO):

$$\text{K3:} \quad \min_\theta(-\log w + w - 1)$$

$$\nabla_\theta L_3 = \underbrace{-\nabla_\theta \log \pi_\theta(y|x)(R(x,y) - \log Z(x))}_{\text{Typical policy gradient, } L_1} - \underbrace{\Big(\nabla_\theta \log \pi_\theta(y|x)\Big)\left(\frac{\pi_{\text{ref}}(y|x)\,\exp R(x,y)}{\pi_\theta(y|x)\,Z(x)}\right)}_{\text{Same term as the one in } L_5} = \nabla_\theta L_1 + \nabla_\theta L_5 \tag{150}$$

The K3^{++} estimator leads to a combined L_1 and L_2 (GRPO + APA):

$$\text{K3}^{++}\text{:} \quad \min_\theta r\cdot(-\log w + w - 1) \qquad r = \frac{\pi_\theta(y|x)}{\operatorname{sg}(\pi_\theta(y|x))}$$

$$\nabla_\theta L_{3++} = \nabla_\theta L_1 + \Big(\nabla_\theta \log \pi_\theta(y|x)\Big)\Big(\log \pi_\theta(y|x) - \log \pi_{\text{ref}}(y|x) - R(x,y) + \log Z(x)\Big) = \nabla_\theta L_1 + \nabla_\theta L_2 \tag{151}$$

The K4 estimator leads to L_2 (same as the APA loss):

$$\text{K4:} \quad \min_\theta r\cdot \operatorname{sg}(-\log w)$$

$$\nabla_\theta L_4 = \Big(\nabla_\theta \log \pi_\theta(y|x)\Big)\Big(\log \pi_\theta(y|x) - \log \pi_{\text{ref}}(y|x) - R(x,y) + \log Z(x)\Big) = \nabla_\theta L_2 \tag{152}$$

A Note on Off-policy Correction Just like the off-policy correction for any sampled KL, any policy loss derived above can simply multiply a stop gradient version of the importance weight to remain unbiased when sampling is not on-policy:

$$y \sim \pi_{\text{sampling}}(\cdot \mid x) \qquad s = \text{sg}\left(\frac{\pi_\theta(y \mid x)}{\pi_{\text{sampling}}(y \mid x)}\right)$$

With the option of using additional PPO-like binary masks on the loss

$$M(x,y) = \begin{cases} 0 & \text{if } R(x,y) - V(x) > 0 \text{ and } s > 1 + \epsilon_{\text{high}} \\ 0 & \text{if } R(x,y) - V(x) < 0 \text{ and } s < 1 - \epsilon_{\text{low}} \\ 1 & \text{otherwise} \end{cases}$$

Leading us to conclude that the off-policy gradient under any loss function L above is

$$\mathbb{E}_{y \sim \pi_{\text{sampling}}(\cdot|x)}[s(x,y)\, M(x,y)\, \nabla_\theta L(x,y)]$$

Though it is worth noting that many prior works find those losses to be effective in the absence of off-policy correction.

Summary of the Equivalence between Policy Gradient and Divergence Minimization We have shown that many existing policy gradient methods can be derived from sampled KL estimators. In other words, better techniques in sampled KL estimation may directly lead to better policy gradient algorithms under such an equivalence.

Estimator	Expression	Corresponding policy gradient	Prior literature?
K1	$-\log w$	$\nabla_\theta L_1 = -\nabla_\theta \log \pi_\theta(y\|x)\Big(R(x,y) - V(x)\Big) + \nabla_\theta \text{KL}(\pi_\theta(\cdot\|x)\|\pi_{\text{ref}}(\cdot\|x))$	**PPO**, **GRPO**, etc
K2	$\frac{1}{2}(\log w)^2$	$\nabla_\theta L_2 = \nabla_\theta\Big(\log \pi_\theta(y\|x) - \log \pi_{\text{ref}}(y\|x) - R(x,y) + V(x)\Big)^2$	**APA**, **Kimi**
K4	$r \cdot \text{sg}(-\log w)$	$\nabla_\theta L_4 = \nabla_\theta L_2$	APA, Kimi
K5	$\text{sg}(w)\log w + \log r$	$\nabla_\theta L_5 = -\nabla_\theta \log \pi_\theta(y\|x)\ \text{softmax}_y\Big(R(x,y) - \log\big(\pi_\theta(y\|x)/\pi_{\text{ref}}(y\|x)\big)\Big)$	**RWR**, **MPO**
K3	$-\log w + w - 1$	$\nabla_\theta L_3 = \nabla_\theta L_1 + \nabla_\theta L_5$	GRPO + MPO
K3^{++}	$r(-\log w + w - 1)$	$\nabla_\theta L_{3++} = \nabla_\theta L_1 + \nabla_\theta L_2$	GRPO + APA

Table 8. **Each KL estimator directly corresponds to a policy gradient loss** under the divergence minimization view of RL.

K. Policy Gradient as f-Divergence Minimization

Once a problem is framed as minimizing a KL, it can typically be recast into the general framework of f-divergence minimization, leading to a family of loss functions. We have seen this in a variety of other optimization problems: the generalization of GAN (Goodfellow et al., 2020) to f-GAN (Nowozin et al., 2016), GAIL (Ho & Ermon, 2016) to f-MAX (Ghasemipour et al., 2020) and f-VIM (Ke et al., 2020), DPO (Rafailov et al., 2023) to f-DPO (Wang et al., 2023). In this section, we still assume that the optimal policy takes the form of a Boltzmann policy (Ziebart et al., 2008; Schulman et al., 2017a) due to its partition function having an intuitive interpretation (which says that the log partition function is equivalent to the value function), but explores using a sampled f-divergence estimator for policy optimization.

We now extend the results in §J to generic f-divergence estimators. Let the optimal policy π^* be:

$$\pi^*(y|x) = \frac{1}{Z(x)}\pi_{\text{ref}}(y|x)\exp(R(x,y)) \qquad w = \frac{\pi^*(y|x)}{\pi_\theta(y|x)} = \frac{\pi_{\text{ref}}(y|x)\,\exp R(x,y)}{\pi_\theta(y|x)\,Z(x)}$$

Re-iterating the definition of f-divergence, with $f(\cdot)$ being convex and $f(1) = 0$:

$$D_f(p||q) = \mathbb{E}_{y\sim q}\left[f\left(\frac{p(y)}{q(y)}\right)\right]$$

We now study two directions of f-divergence minimization: $D_f(\pi_\theta(\cdot|x)||\pi^*(\cdot|x))$ and $D_f(\pi^*(\cdot|x)||\pi_\theta(\cdot|x))$.

K.1. Minimizing $D_f(\pi_\theta, \pi^*)$

If we define function $g_f(\cdot)$ as:

$$g_f(w) \triangleq w\, f\left(\frac{1}{w}\right) \tag{153}$$

Then, using the definition of f-divergence:

$$D_f(\pi_\theta(\cdot|x)||\pi^*(\cdot|x)) = \mathbb{E}_{y\sim\pi^*(\cdot|x)}\left[f\left(\frac{1}{w}\right)\right] = \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[w\, f\left(\frac{1}{w}\right)\right] = \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[g_f(w(x,y))\right] \tag{154}$$

Note that $w(x,y)$ has the following properties:

$$\nabla_\theta \log w(x,y) = -\nabla_\theta \log \pi_\theta(y|x), \qquad \nabla_\theta w(x,y) = -w(x,y)\,\nabla_\theta \log \pi_\theta(y|x). \tag{155}$$

Therefore, using the score-function identity, we arrive at:

$$\begin{aligned}\nabla_\theta D_f(\pi_\theta\|\pi^*) = \nabla_\theta \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[g_f(w)\right] &= \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\Big[\nabla_\theta \log \pi_\theta(y|x)\, g_f(w) + \nabla_\theta g_f(w)\Big] \\ &= \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\Big[\nabla_\theta \log \pi_\theta(y|x)\,\big(g_f(w) - w\, g_f'(w)\big)\Big].\end{aligned} \tag{156}$$

Hence, minimizing $D_f(\pi_\theta\|\pi^*)$ is equivalent to a policy gradient update with scalar weight $-\phi_f(w)$ where $\phi_f(w)$ is

$$\phi_f(w) \triangleq g_f(w) - w\, g_f'(w) = f'(1/w). \tag{157}$$

K.2. Minimizing $D_f(\pi^*, \pi_\theta)$

We now consider the reverse direction:

$$D_f(\pi^*(\cdot|x)\,\|\,\pi_\theta(\cdot|x)) = \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[f\left(\frac{\pi^*(y|x)}{\pi_\theta(y|x)}\right)\right] = \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[f(w(x,y))\right]. \tag{158}$$

By the same derivation as above,

$$\nabla_\theta D_f(\pi^*\|\pi_\theta) = \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\Big[\nabla_\theta \log \pi_\theta(y|x)\,\big(f(w) - w\, f'(w)\big)\Big]. \tag{159}$$

Thus the induced policy-gradient weight for the reverse direction is

$$\psi_f(w) \triangleq f(w) - w\, f'(w). \tag{160}$$

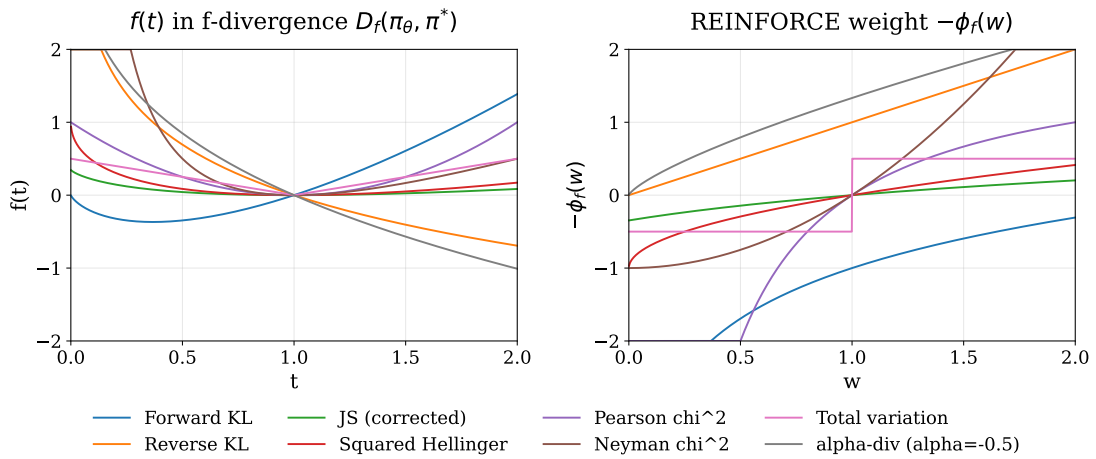

Figure 10. Policy gradient based on $\nabla_\theta D_f(\pi_\theta, \pi^*)$, where the REINFORCE weight for $\nabla_\theta \log \pi_\theta(y|x)$ is $-\phi_f(w) = -f'(1/w)$.

Summary of f-divergence loss. The two directions of f-divergence minimization are reduced to the following:

$$\nabla_\theta D_f(\pi_\theta, \pi^*) = \mathbb{E}_{y\sim\pi_\theta}\Big[\nabla_\theta \log \pi_\theta(y|x)\, \phi_f(w)\Big], \qquad \phi_f(w) = f'(1/w), \tag{161}$$

$$\nabla_\theta D_f(\pi^*, \pi_\theta) = \mathbb{E}_{y\sim\pi_\theta}\Big[\nabla_\theta \log \pi_\theta(y|x)\, \psi_f(w)\Big], \qquad \psi_f(w) = f(w) - w f'(w). \tag{162}$$

The EM Algorithm The expectation-maximization (EM) process can be summarized as the following. For each prompt or task x, we sample N rollouts $[y^{(1)}, \ldots, y^{(N)}]$ from $\pi_\theta(\cdot \mid x)$ to estimate the partition function $Z(x)$ and the ratio w as:

$$\textbf{E-Step} \qquad Z(x) \approx \frac{1}{N}\sum_{i=1}^{N} \mathrm{sg}\left(\frac{\pi_{\text{ref}}(y^{(i)} \mid x)}{\pi_\theta(y^{(i)} \mid x)}\right) \exp\Big(R(x, y^{(i)})\Big) \qquad w = \frac{\pi_{\text{ref}}(y \mid x)}{\pi_\theta(y \mid x)} \cdot \frac{\exp R(x,y)}{Z(x)}$$

where the returns $R(x, y^{(i)})$ are (pre-) scaled by $1/\beta$ to control the regularization effect of the anchor π_{ref}. After the E-Step, M-Step performs gradient updates based on f-divergence minimization; the following two tables, Table 9 and Table 10, summarize what to minimize under each f-divergence (**M-Step**).

Table 9. **M-Step:** Minimizing $D_f(\pi_\theta, \pi^*)$ results in the gradient update $\mathbb{E}_{y\sim\pi_\theta}[\nabla_\theta \log \pi_\theta(y|x)\, \phi_f(w)]$

f-divergence	$f(t)$	$\phi_f(w) = f'(1/w)$
Forward KL	$t \log t$	$-\log w + 1$
Reverse KL	$-\log t$	$-w$
Jensen–Shannon (JS)	$\frac{1}{2}\left(t \log t - (t+1)\log\left(\frac{t+1}{2}\right)\right)$	$\frac{1}{2}\log\left(\frac{2}{1+w}\right)$
Squared Hellinger	$(\sqrt{t}-1)^2$	$1-\sqrt{w}$
Pearson χ^2	$(t-1)^2$	$-2+2/w$
Neyman χ^2	$\frac{(t-1)^2}{t}$	$1-w^2$
Total variation (TV)	$\frac{1}{2}\lvert t-1\rvert$	$-\frac{1}{2}\,\mathrm{sgn}(w-1)$
α-divergence ($\alpha \neq \pm 1$)	$\frac{4}{1-\alpha^2}\left(1-t^{\frac{1+\alpha}{2}}\right)$	$\frac{2}{\alpha-1} w^{\frac{1-\alpha}{2}}$

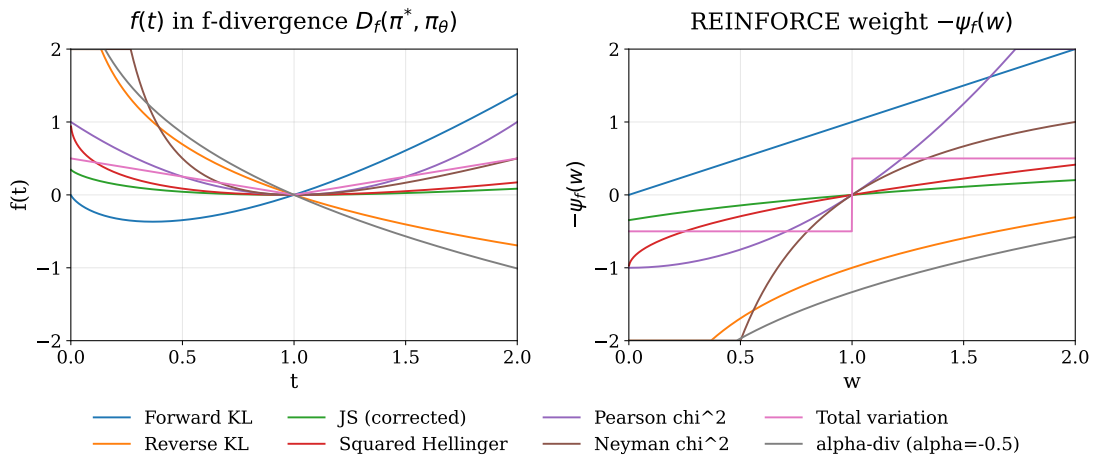

Figure 11. Policy gradient based on $\nabla_\theta D_f(\pi^*, \pi_\theta)$, where the REINFORCE weight for $\nabla_\theta \log \pi_\theta(y|x)$ is $-\psi_f(w) = -f(w) + wf'(w)$.

Table 10. **M-Step:** Minimizing $D_f(\pi^*, \pi_\theta)$ results in the gradient update $\mathbb{E}_{y\sim\pi_\theta}[\nabla_\theta \log \pi_\theta(y|x)\, \psi_f(w)]$

f**-divergence**	$f(t)$	$\psi_f(w) = f(w) - wf'(w)$
Forward KL	$t \log t$	$-w$
Reverse KL	$-\log t$	$-\log w + 1$
Jensen–Shannon (JS)	$\frac{1}{2}\left(t \log t - (t+1)\log\left(\frac{t+1}{2}\right)\right)$	$\frac{1}{2}\log\left(\frac{2}{1+w}\right)$
Squared Hellinger	$(\sqrt{t}-1)^2$	$1-\sqrt{w}$
Pearson χ^2	$(t-1)^2$	$1-w^2$
Neyman χ^2	$\frac{(t-1)^2}{t}$	$-2+2/w$
Total variation (TV)	$\frac{1}{2}\lvert t-1\rvert$	$-\frac{1}{2}\operatorname{sgn}(w-1)$
α-divergence ($\alpha \neq \pm 1$)	$\frac{4}{1-\alpha^2}\left(1-t^{\frac{1+\alpha}{2}}\right)$	$-\frac{2}{\alpha+1}w^{\frac{\alpha+1}{2}} - \frac{4}{(\alpha-1)(\alpha+1)}$