FERPO: FORWARD ENTROPY-REGULARIZED POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Several state-of-the-art methods for online reinforcement learning in continuous control improve policies using action gradients of a learned critic. However, critics are typically trained to predict returns, and accurate value predictions do not necessarily yield accurate action derivatives, potentially leading to unreliable policy updates. We propose Forward Entropy-Regularized Policy Optimization (FERPO), an on-policy maximum entropy reinforcement learning algorithm that performs policy improvement using critic values without differentiating the critic with respect to actions. FERPO derives an optimal target action distribution from a policy-improvement objective regularized by entropy and Kullback–Leibler (KL) divergence. We then fit the actor to this target by minimizing a forward-KL objective, estimated using self-normalized importance sampling (SNIS) with actions drawn from the rollout policy. By limiting the target distribution's deviation from the rollout policy, the KL regularization helps keep these importance weights well behaved. In contrast to reverse-KL objectives, which can favor a subset of the target distribution's modes, the forward-KL objective encourages coverage of multiple high-value modes and thereby promotes exploration. Experiments and ablations on MuJoCo Playground and ManiSkill show competitive performance and sample-efficiency gains. Computational benchmarks also demonstrate faster actor updates than Relative Entropy Pathwise Policy Optimization (REPPO).

1 INTRODUCTION

Reinforcement learning (RL) learns policies through interaction to maximize cumulative reward. Its successes range from game playing (Silver et al., 2017) to autonomous drone racing (Kaufmann et al., 2023) and robot locomotion (Ha et al., 2025). In robotics, massively parallel simulation has accelerated on-policy learning by collecting trajectories from thousands of environments simultaneously. This has enabled rapid policy training and successful transfer from simulation to physical robots (Rudin et al., 2022; Zakka et al., 2025). These advances make efficient and reliable policy improvement central to scaling online robot learning.

Relative Entropy Pathwise Policy Optimization (REPPO; Voelcker et al., 2026) is a recent state-of-the-art on-policy method for this setting. It learns a soft action-value critic from on-policy trajectories and improves the actor by differentiating the critic with respect to actions, while limiting policy changes relative to the rollout policy. This makes pathwise actor updates compatible with massively parallel on-policy learning. However, standard critic training supervises values, whereas these actor updates require accurate action derivatives. Accurate value predictions do not necessarily imply accurate derivatives: critic-gradient error can bias policy improvement (D'Oro & Jaśkowski, 2020), and learned action gradients can show poor directional agreement with rollout-based estimates (Sarafian et al., 2021). In on-policy learning, biased gradients can also interact with changing data distributions and destabilize training (Gurumurthy et al., 2023). These limitations motivate policy improvement that uses critic values without requiring their action derivatives.

We derive the optimal target distribution for a standard RL policy-improvement objective with entropy and Kullback–Leibler (KL) regularization. Making this target explicit lets us replace reverse-KL projection by forward-KL fitting, which encourages coverage of target modes (Minka, 2005). The KL trust region keeps the target close to the rollout policy, helping control importance-weight concentration, as in recent diffusion and normalizing flow based sampling methods (von Klitzing

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

et al., 2026; Blessing et al., 2025). We introduce Forward Entropy-Regularized Policy Optimization (FERPO), which applies this construction to on-policy learning. Its actor update uses critic values without differentiating the critic with respect to actions. Our contributions are:

- An on-policy actor update that uses critic values to construct a regularized target action distribution and fits the actor to this target using forward KL, without differentiating the critic with respect to actions (Section 3.1).
- A practical estimator using self-normalized importance sampling (SNIS), with Lagrange multipliers that regulate policy entropy and importance-weight concentration through normalized effective sample size (ESS) (Sections 3.1 and 3.2).
- An evaluation on ManiSkill and MuJoCo Playground demonstrating competitive performance and sample-efficiency gains (Section 5.4), supported by ablations (Section 5.3) and computational benchmarks showing faster actor updates than REPPO (Section 5.2).
- Cached actor updates that reduce computation, and a state-value extension based on one-step simulation, with promising results on AcrobotSwingupSparse (Section 5.3).

2 Problem Description

Maximum-entropy reinforcement learning. We consider a discounted Markov decision process with continuous state space $\mathcal{S}$, bounded continuous action space $\mathcal{A}$, initial-state density ρ_0, transition density $P(s' \mid s, a)$, reward $r(s, a)$, and discount factor $\gamma \in [0, 1)$. For a policy π, let ρ_t^π denote the state distribution at time t, generated by $s_0 \sim \rho_0$, $a_t \sim \pi(\cdot \mid s_t)$, and $s_{t+1} \sim P(\cdot \mid s_t, a_t)$. Define its normalized discounted state-visitation distribution as $d^\pi(s) := (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \rho_t^\pi(s)$, with $\int d^\pi(s)\,\mathrm{d}s = 1$. Using this distribution, the maximum-entropy objective can be written as

$$\begin{aligned} \max_\theta \quad J_\tau(\pi_\theta) &:= \frac{1}{1-\gamma}\mathbb{E}_{\substack{s\sim d^{\pi_\theta} \\ a\sim\pi_\theta(\cdot|s)}}\left[r(s,a) - \tau \log \pi_\theta(a \mid s)\right] \\ &= \mathbb{E}_{\rho_0, P, \pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t \big(r(s_t, a_t) - \tau \log \pi_\theta(a_t \mid s_t)\big)\right], \end{aligned} \tag{1}$$

where the first expectation draws a state $s \sim d^{\pi_\theta}$ and then an action $a \sim \pi_\theta(\cdot \mid s)$. The second expectation is over trajectories generated by $s_0 \sim \rho_0$, $a_t \sim \pi_\theta(\cdot \mid s_t)$, and $s_{t+1} \sim P(\cdot \mid s_t, a_t)$. The factor $1/(1-\gamma)$ accounts for the normalization of d^{π_θ}, making the two forms equal. Here π_θ is a neural-network policy (see Appendix A.4 for its parameterization), and $\tau > 0$ controls its entropy. Since $\mathcal{H}(\pi(\cdot \mid s)) = -\mathbb{E}_{a\sim\pi}[\log \pi(a \mid s)]$, this objective balances expected reward and policy entropy (Ziebart, 2010; Haarnoja et al., 2017; 2018a). Its soft value functions satisfy

$$Q_\tau^\pi(s,a) = r(s,a) + \gamma \mathbb{E}_{s'\sim P(\cdot|s,a)}[V_\tau^\pi(s')], \quad V_\tau^\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}[Q_\tau^\pi(s,a) - \tau \log \pi(a \mid s)].$$

Thus, Q_τ^π includes future entropy rewards, while the current action's entropy contribution appears explicitly in V_τ^π.

KL surrogate. Let $\pi_{\mathrm{roll}} := \pi_{\theta_{\mathrm{roll}}}$ be the frozen policy that generated the current rollout, with soft critic $Q_{\mathrm{roll}} := Q_\tau^{\pi_{\mathrm{roll}}}$. Applying the policy-gradient theorem (Sutton et al., 1999) to Eq. 1 yields the local reverse-KL surrogate

$$\mathcal{L}_{\mathrm{ME}}(\theta; \theta_{\mathrm{roll}}) := \frac{\tau}{1-\gamma}\mathbb{E}_{s\sim d_{\mathrm{roll}}}[D_{\mathrm{KL}}(\pi_\theta(\cdot \mid s) \| p_{Q,\tau}(\cdot \mid s))], \tag{2}$$

where $d_{\mathrm{roll}} := d^{\pi_{\theta_{\mathrm{roll}}}}$ is the rollout policy's normalized discounted state-visitation distribution and $p_{Q,\tau}(a \mid s) \propto \exp(Q_{\mathrm{roll}}(s,a)/\tau)$ is its Boltzmann policy-improvement target. The policy π_θ acts as a variational approximation to this target. With Q_{roll} and d_{roll} held fixed, the surrogate satisfies $\nabla_\theta \mathcal{L}_{\mathrm{ME}}(\theta; \theta_{\mathrm{roll}}) = -\nabla_\theta J_\tau(\pi_\theta)$ at $\theta = \theta_{\mathrm{roll}}$. Thus, the surrogate's descent direction matches the return's ascent direction at the rollout parameters; see Appendix B.1 for the derivation.

Trust regions and total variation. Repeated updates on one rollout can change state visitation and reduce surrogate accuracy. Write $\pi_{\mathrm{curr}} = \pi_\theta$ and define $D_{\mathrm{TV}}(\pi_{\mathrm{curr}}, \pi_{\mathrm{roll}}) := \frac{1}{2}\int_{\mathcal{A}} |\pi_{\mathrm{curr}}(a) - \pi_{\mathrm{roll}}(a)|\,\mathrm{d}a$, suppressing state conditioning. Its worst-state value is $D_{\mathrm{TV}}^{\max} := \sup_s D_{\mathrm{TV}}(\pi_{\mathrm{curr}}(\cdot \mid s), \pi_{\mathrm{roll}}(\cdot \mid s))$. Let $\Delta J_\tau := J_\tau(\pi_{\mathrm{curr}}) - J_\tau(\pi_{\mathrm{roll}})$ be the actual return change and $\Delta J_{\mathrm{surrogate}} :=$

$\mathcal{L}_{\mathrm{ME}}(\theta_{\mathrm{roll}};\theta_{\mathrm{roll}}) - \mathcal{L}_{\mathrm{ME}}(\theta;\theta_{\mathrm{roll}})$ the predicted gain. For an exact critic and soft advantage bounded in magnitude by ϵ_A, the performance-difference argument (Schulman et al., 2015; Cayci et al., 2024) gives

$$\Delta J_\tau \geq \Delta J_{\mathrm{surrogate}} - \frac{4\gamma\epsilon_A}{(1-\gamma)^2}(D_{\mathrm{TV}}^{\max})^2 - \frac{\tau}{1-\gamma}D_{\mathrm{KL}}^{\max}, \tag{3}$$

where $D_{\mathrm{KL}}^{\max} := \sup_s D_{\mathrm{KL}}(\pi_{\mathrm{curr}}(\cdot \mid s)\|\pi_{\mathrm{roll}}(\cdot \mid s))$. The TV term bounds the advantage error; the KL term is the entropy correction (Appendix B.6). Pinsker's inequality, $2(D_{\mathrm{TV}}^{\max})^2 \leq D_{\mathrm{KL}}^{\max}$ (van Erven & Harremoës, 2014, Theorem 31), upper-bounds the total penalty by $\big(\frac{2\gamma\epsilon_A}{(1-\gamma)^2} + \frac{\tau}{1-\gamma}\big)D_{\mathrm{KL}}^{\max}$. This motivates minimizing $\mathcal{L}_{\mathrm{ME}}$ subject to $D_{\mathrm{KL}}^{\max} \leq \delta_{\mathrm{KL}}$. For tanh-squashed Gaussians, as frequently used in MaxEnt-RL, uniform forward-KL constraints or TV bounds strictly below one also control the reverse-KL entropy correction (Appendix B.7).

Practical objective. Replacing the supremum by an expectation over rollout states gives the average-KL constraint $\bar{D}_\theta := \mathbb{E}_{s\sim d_{\mathrm{roll}}}[D_{\mathrm{KL}}(\pi_\theta(\cdot \mid s)\|\pi_{\mathrm{roll}}(\cdot \mid s))] \leq \delta_{\mathrm{KL}}$. We also require a minimum average entropy $\bar{\mathcal{H}}_\theta := \mathbb{E}_{s\sim d_{\mathrm{roll}}}[\mathcal{H}(\pi_\theta(\cdot \mid s))] \geq \mathcal{H}_{\mathrm{target}}$, interpreting τ as its Lagrange multiplier, as in automatic entropy tuning (Haarnoja et al., 2018b). With Q_{roll} and d_{roll} held fixed, maximizing expected critic value under these constraints gives the Lagrangian

$$\mathcal{L}(\pi_\theta,\tau,\lambda) := -\mathbb{E}_{\substack{s\sim d_{\mathrm{roll}}\\ a\sim\pi_\theta(\cdot|s)}}[Q_{\mathrm{roll}}(s,a)] + \tau(\mathcal{H}_{\mathrm{target}} - \bar{\mathcal{H}}_\theta) + \lambda(\bar{D}_\theta - \delta_{\mathrm{KL}}), \tag{4}$$

where $\tau,\lambda \geq 0$. The policy minimizes $\mathcal{L}$, while the multipliers maximize it: τ increases when entropy is below target, and λ increases when the KL bound is exceeded. The rollout-average constraint and estimated critic approximate the trust-region problem but do not guarantee its worst-state bound. This motivates a shared design principle for practical on-policy algorithms: control deviation from the rollout policy when making multiple updates on the same data, so that the local surrogate remains informative.

Connection to existing methods. TRPO controls policy changes through an explicit KL constraint (Schulman et al., 2015), while Proximal Policy Optimization (PPO) clips policy likelihood ratios in the actor surrogate to discourage large changes (Schulman et al., 2017). Relative entropy pathwise policy optimization (REPPO) learns a soft critic Q_ϕ from on-policy rollouts. At each state, let $\widehat{k}_\theta(s)$ estimate the forward KL $D_{\mathrm{KL}}(\pi_{\mathrm{roll}}(\cdot \mid s)\|\pi_\theta(\cdot \mid s))$, using actions drawn from π_{roll}. For $\widehat{k}_\theta(s) < \delta_{\mathrm{KL}}$, REPPO minimizes the unconstrained loss $\ell_{\mathrm{ME}}(\theta;s) = \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}[\tau\log\pi_\theta(a \mid s) - Q_\phi(s,a)]$ using pathwise gradients. Otherwise, it replaces this loss by $\lambda\widehat{k}_\theta(s)$, with adaptive $\lambda > 0$, pulling the policy back toward π_{roll} (Voelcker et al., 2026).

3 METHOD: FERPO

Pathwise actor updates require accurate critic action derivatives, although standard critic learning supervises values. Value accuracy alone does not ensure derivative accuracy (D'Oro & Jaśkowski, 2020; Sarafian et al., 2021). This motivates an actor update that uses critic values without requiring $\nabla_a Q_\phi(s,a)$. We introduce *Forward Entropy-Regularized Policy Optimization* (**FERPO**). FERPO first minimizes the Lagrangian in Eq. 4 over unrestricted action distributions $q(\cdot \mid s)$ with fixed multipliers, then fits the parametric policy to the resulting target using the forward KL divergence combined with self-normalized importance sampling (SNIS). The temperature and trust-region multiplier are adapted alongside the actor. The actor differentiates the policy log density with the target weights held fixed; it still depends on critic-value accuracy and adequate sampling of the target.

3.1 TARGET CONSTRUCTION AND ACTOR UPDATES

We define the target as $q^* = \arg\min_q \mathcal{L}(q,\tau,\lambda)$, with $\mathcal{L}$ from Eq. 4 and the minimization over all conditional action densities $q(\cdot \mid s)$, unrestricted by the parametric policy family. With the critic, rollout policy, τ, and λ fixed, this minimization separates across states. Introducing a Lagrange multiplier for the normalization constraint $\int_{\mathcal{A}} q(a \mid s)\,\mathrm{d}a = 1$ and setting the functional derivative with respect to q to zero yields the unique optimal density

$$q^*(a \mid s) = \frac{1}{Z_*(s)}\exp\left(\frac{Q_{\mathrm{roll}}(s,a)}{\tau+\lambda}\right)\pi_{\mathrm{roll}}(a \mid s)^{\lambda/(\tau+\lambda)}. \tag{5}$$

Here $Z_*(s)$ normalizes the density. Thus, q^* is the unrestricted minimizer of $\mathcal{L}(\cdot, \tau, \lambda)$, which we subsequently approximate within the policy family. See Appendix B.2 for a derivation. The target combines high-value actions with proximity to the rollout policy: $\lambda = 0$ recovers $p_{Q,\tau}$, while increasing λ makes the update more conservative. Under the assumptions of Appendix B.4, repeated exact unrestricted updates with a fixed critic and temperature and a refreshed reference converge to $p_{Q,\tau}$. The fixed KL multiplier λ controls the convergence rate without changing the limit.

For fixed multipliers, Eq. 4 equals $(\tau + \lambda)\mathbb{E}_{s\sim d_{\text{roll}}}[D_{\text{KL}}(\pi_\theta \| q^*)]$ plus a constant independent of the policy. FERPO replaces this reverse-KL projection by a forward-KL fit, keeping the critic and rollout parameters fixed during each actor update:

$$\mathcal{L}_{\text{F}}(\theta) := \mathbb{E}_{s\sim d_{\text{roll}}}[D_{\text{KL}}(q^*(\cdot \mid s) \| \pi_\theta(\cdot \mid s))] = -\mathbb{E}_{\substack{s\sim d_{\text{roll}} \\ a\sim q^*(\cdot|s)}}[\log \pi_\theta(a \mid s)] + \text{const.}$$

This is a weighted maximum-likelihood fit to the optimal regularized target. All state expectations below use the frozen distribution d_{roll} and are estimated using rollout batches. To estimate the action expectation, draw K candidate actions $a_i \sim \pi_{\text{roll}}(\cdot \mid s)$ and minimize the SNIS loss

$$\boxed{\widehat{\mathcal{L}}_{\text{F}}(\theta) := -\mathbb{E}_{s\sim d_{\text{roll}}}\left[\sum_{i=1}^{K} w_i(s) \log \pi_\theta(a_i \mid s)\right], \quad \text{where} \quad w_i(s) := \frac{\exp(\ell_i(s))}{\sum_{j=1}^{K} \exp(\ell_j(s))}}, \tag{6}$$

with $\ell_i(s) := [Q_\phi(s, a_i) - \tau \log \pi_{\text{roll}}(a_i \mid s)]/(\tau + \lambda)$. The weights are held fixed during the actor update.

Multiplier losses and complete actor update. As suggested by Eq. 4, we adapt the temperature τ and trust-region coefficient λ as Lagrange multipliers alongside the policy. We parameterize $\tau = \exp(\eta)$ and $\lambda = \exp(\xi)$ to keep both multipliers positive. The temperature update follows automatic entropy tuning in SAC (Haarnoja et al., 2018b). Using fresh actions $\widetilde{a}_s \sim \pi_\theta(\cdot \mid s)$, we estimate the policy entropy as $\widehat{\mathcal{H}}_\theta := -\mathbb{E}_{s\sim d_{\text{roll}}}[\log \pi_\theta(\widetilde{a}_s \mid s)]$ and minimize

$$\mathcal{L}_\tau(\eta) := \tau \ \text{stopgrad}[\widehat{\mathcal{H}}_\theta - \mathcal{H}_{\text{target}}], \tag{7}$$

where stopgrad holds its argument fixed during differentiation. Gradient descent increases τ when entropy is below $\mathcal{H}_{\text{target}}$ and decreases it when entropy exceeds the target.

For λ, however, we replace direct KL control with an update based on the log normalized effective sample size (ESS) of the importance weights. This regulates weight concentration; Section 3.2 gives the detailed motivation and its connection to trust-region control. Using the weights from Eq. 6, define $\widehat{e}(s) := (K \sum_{i=1}^{K} w_i(s)^2)^{-1}$, which ranges from $1/K$ when one action carries all weight to 1 for uniform weights. For $K \geq 2$ and a target $e_{\text{target}} \in (1/K, 1)$, we minimize

$$\mathcal{L}_\lambda(\xi) := \lambda \ \text{stopgrad}[\mathbb{E}_{s\sim d_{\text{roll}}}[\log \widehat{e}(s)] - \log e_{\text{target}}]\,. \tag{8}$$

This increases λ when average log ESS falls below $\log e_{\text{target}}$, flattening the weights, and decreases it when average log ESS exceeds the target. Together with the forward fit, these multiplier losses give the complete actor objective

$$\mathcal{L}_{\text{policy}}(\theta, \eta, \xi) := \widehat{\mathcal{L}}_{\text{F}}(\theta) + \mathcal{L}_\tau(\eta) + \mathcal{L}_\lambda(\xi).$$

The weights are detached from both the policy and multipliers, and the entropy and log-ESS gaps are detached in their respective losses. The critic and rollout policy stay fixed within each actor update; updated τ and λ are used to construct the next target weights. Algorithm 1 in Appendix A.4 summarizes the full training loop.

3.2 Log ESS and trust-region control

Controlling importance-weight concentration. Using the optimal regularized distribution q^* as the forward-KL target tempers the importance weights needed to fit the Boltzmann target $p_{Q,\tau}$ directly. For $\tau > 0$, $q^*(a \mid s)/\pi_{\text{roll}}(a \mid s) \propto [p_{Q,\tau}(a \mid s)/\pi_{\text{roll}}(a \mid s)]^{\tau/(\tau+\lambda)}$. For $\lambda > 0$, the exponent lies below one, reducing log-weight differences and flattening the normalized weights on the same candidate actions. With zero entropy regularization ($\tau = 0$, $\lambda > 0$), the explicit rollout-density factor cancels entirely, leaving $w_i(s) \propto \exp(Q_\phi(s, a_i)/\lambda)$. For positive τ, the residual factor

is $\pi_{\text{roll}}(a_i \mid s)^{-\tau/(\tau+\lambda)}$. For a fixed critic and rollout policy, increasing τ at fixed λ strengthens this inverse-density dependence and can increase weight concentration, although it also softens the value weighting, so the overall effect need not be monotone. The update of the Lagrange multiplier λ in Eq. 8 uses this flattening effect to limit weight concentration.

At the population level, a lower bound on normalized ESS also gives an upper bound on the target's KL divergence from the rollout policy. The following proposition also transfers this control to the fitted actor in TV when its population fitting loss does not increase. This motivates using ESS to adapt λ and connects it to the policy-proximity motivation in Section 2.

Proposition 1 (ESS control of target and actor)**.** *Fix a state and a target $q^* \ll \pi_{\text{roll}}$ with population normalized ESS $e_\infty := 1/\mathbb{E}_{\pi_{\text{roll}}}[(q^*/\pi_{\text{roll}})^2] > 0$. If fitting to this fixed target satisfies $D_{\text{KL}}(q^*\|\pi_{\text{curr}}) \le D_{\text{KL}}(q^*\|\pi_{\text{roll}})$, the triangle inequality and Pinsker's inequality give*

$$\frac{1}{2} D_{\text{TV}}(\pi_{\text{curr}}, \pi_{\text{roll}})^2 \le D_{\text{KL}}(q^*\|\pi_{\text{roll}}) \le D_2(q^*\|\pi_{\text{roll}}) = -\log e_\infty, \tag{9}$$

where D_2 is the order-two Rényi divergence (van Erven & Harremoës, 2014).

The proof is given in Appendix B.3. **What the ESS target controls.** FERPO uses a shared λ to regulate log ESS averaged over rollout states. Under the population fitting condition, higher population ESS tightens the bound on the total variation between π_{curr} and π_{roll}. For Gaussian policies, a uniform TV bound below one also bounds $D_{\text{KL}}(\pi_{\text{curr}}\|\pi_{\text{roll}})$. In practice, ESS is estimated from sampled actions and averaged across states, so the controller does not guarantee this KL bound at every state.

3.3 Critic learning

To improve learning of the soft action-value function Q_ϕ, we follow the critic-training approach of REPPO (Voelcker et al., 2026), combining multi-step temporal-difference (TD) targets with trace parameter λ_{TD}, the histogram loss with Gaussian targets (HL-Gauss) (Farebrother et al., 2024), and auxiliary prediction losses. The targets are computed from on-policy rollouts and include the entropy contributions to the soft return. Here λ_{TD} is the trace-decay parameter, separate from the actor's KL coefficient λ. HL-Gauss encodes each scalar soft-return target as a Gaussian-smoothed histogram over value bins and trains the critic's categorical output using cross-entropy. The scalar $Q_\phi(s, a)$ is recovered as the mean of the predicted histogram. Following REPPO, auxiliary self-prediction trains the critic representation to predict the next transition's latent representation, providing an additional learning signal alongside the value targets (Schwarzer et al., 2021; Voelcker et al., 2024; Fujimoto et al., 2025).

4 Related Work

Constrained measure transport (CMT). Related methods use similar constrained target construction and distribution fitting for sampling with normalizing flows (von Klitzing et al., 2026) and diffusion models (Blessing et al., 2025). Both adapt the trust-region multiplier λ using KL constraints. The normalizing-flow method additionally regulates temperature through a bound on entropy decrease.

Maximum a posteriori policy optimization. MPO is an off-policy method that alternates KL-constrained target construction with forward-KL policy fitting, using replay data and Retrace for critic learning (Abdolmaleki et al., 2018). Using our notation, its variational objective uses the augmented reward $r(s, a) - \lambda \log(q(a \mid s)/\pi_{\text{roll}}(a \mid s))$, with unrestricted improvement distribution q and frozen reference policy π_{roll}. In MPO, this reference can differ from the behavior policies represented in replay. On-policy MPO would set $q = \pi_{\text{roll}}$ where the log ratio vanishes, so its critic estimates unregularized environment returns. MPO uses a KL constraint to limit the improvement of q away from π_{roll} while this critic is held fixed (Abdolmaleki et al., 2018). FERPO also uses forward-KL fitting, but learns from on-policy rollouts with the entropy-augmented return $r(s, a) - \tau \log \pi_\theta(a \mid s)$. Its critic therefore retains future entropy contributions even when evaluating the rollout policy, while a separate λ controls policy deviation during improvement.

Advantage-weighted policy fitting. Advantage-weighted regression (AWR; Peng et al., 2019) and advantage-weighted actor–critic (AWAC; Nair et al., 2020) are off-policy methods that fit policies to

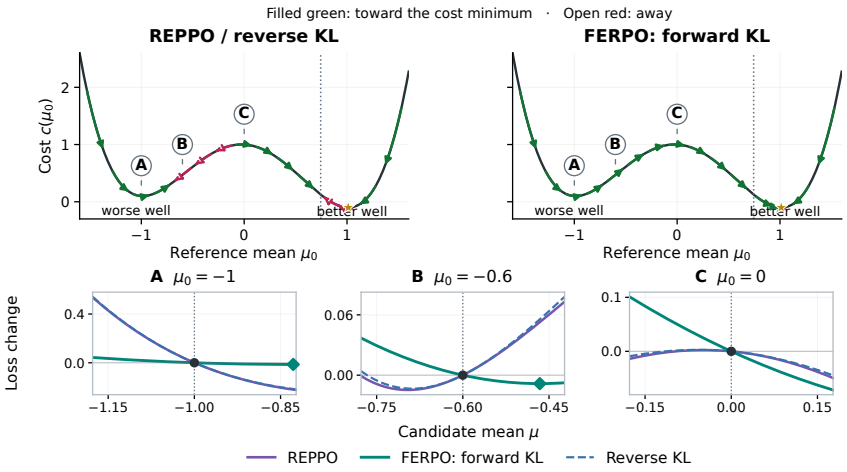


Figure 1: **Actor directions and local losses.** Top: green/red arrows point toward/away from the cost minimum (star); lengths are illustrative, not convergence guarantees. Bottom: frozen-reference losses at A–C, zeroed at the reference; KL losses are scaled by T. Details: Appendix A.1.

actions stored in replay using exponential advantage weights. AWR can learn online from scratch, while AWAC targets offline pretraining followed by online fine-tuning. Their underlying target is $q^*(a \mid s) \propto \pi_{\text{beh}}(a \mid s) \exp(A_{\text{env}}(s,a)/\lambda)$, where π_{beh} is the conditional action distribution represented in replay, potentially combining multiple behavior policies, and A_{env} estimates unregularized advantage. The state-dependent advantage baseline cancels on normalization, so FERPO at $\tau = 0$ has the same target for a common reference and critic. AWR and AWAC omit the state-dependent normalizer, thereby reweighting states in the policy-fitting objective (see Appendix A.5 for details). FERPO instead learns from fresh on-policy rollouts, samples candidate actions from the frozen rollout policy, and uses SNIS to normalize their weights separately within each state. For $\tau > 0$, FERPO also favors less likely actions at equal critic value, providing an explicit exploration incentive.

5 Experiments

5.1 Update directions and sampling

We compare actor updates on the bimodal cost $c(a) = (a^2 - 1)^2 - 0.1a$, whose right well is better. We use an exact critic $Q = -c$ and Gaussian policies $\pi_\mu = \mathcal{N}(\mu, \sigma^2)$, with fixed $\sigma = 0.4$, $\tau = 0.1$, $\lambda = 0.05$, and $T = \tau + \lambda$.

Forward fitting changes the update direction. Let ν denote the candidate mean and $m_* := \mathbb{E}_{q^*_\mu}[a]$. We differentiate with respect to ν, holding the reference π_μ and target q^*_μ fixed. At $\nu = \mu$, REPPO's KL regularization is inactive:

$$\begin{aligned} \nabla_\nu L_{\text{REPPO}}(\nu)\big|_{\nu=\mu} &= T\, \nabla_\nu D_{\text{KL}}(\pi_\nu \| q^*_\mu)\big|_{\nu=\mu} = \mathbb{E}_{\pi_\mu}[c'(a)], \\ \nabla_\nu L_{\text{FERPO}}(\nu;\mu)\big|_{\nu=\mu} &= \nabla_\nu D_{\text{KL}}(q^*_\mu \| \pi_\nu)\big|_{\nu=\mu} = \frac{\mu - m_*}{\sigma^2} = \frac{1}{\lambda}\mathbb{E}_{q^*_\mu}[c'(a)]. \end{aligned} \tag{10}$$

Both methods average the same cost derivative under different distributions (derivation: Appendix B.5). FERPO's value-based target can weight distant low-cost regions: at B ($\mu = -0.6$), $\mathbb{E}_{\pi_\mu}[c'] = 0.284$ and $\mathbb{E}_{q^*_\mu}[c'] = -0.0419$. Gradient descent moves REPPO left and FERPO right toward $m_* \simeq -0.466$ (Figure 1).

Recovering the direction with SNIS. At B, SNIS with $K = 1{,}024$ rollout-policy samples recovers the rightward direction in only 26.4% of batches: the better well is rarely sampled. Keeping the target fixed and sampling from $r_\kappa = \mathcal{N}(\mu, \kappa^2\sigma^2)$ with proposal-corrected importance weights raises recovery to 98.9% for $\kappa = 2$ and 99.9% for $\kappa = 3$. Appendix A.1 gives the estimator and sampling results (Figure 6).

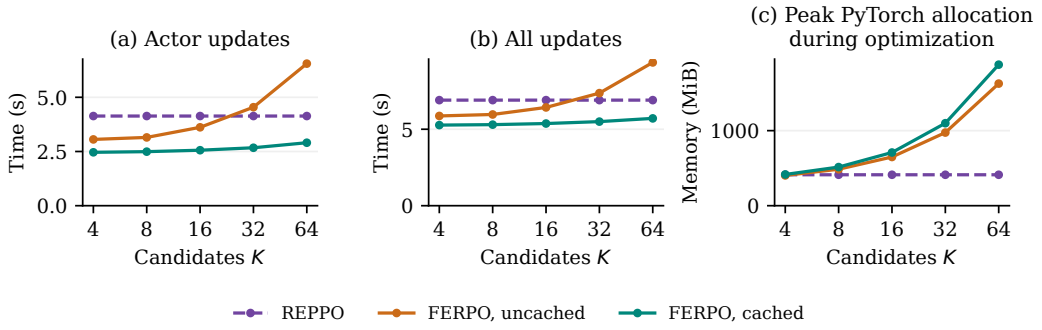


Figure 2: **Update cost on an A100.** Medians and ranges over three repetitions with 131,072 rollout states. Actor time includes cache construction; total updates include critic training. PyTorch allocation excludes simulation and unused reservations.

5.2 Computational cost

We time updates on fixed synthetic rollouts using REPPO's published configuration (Voelcker et al., 2026), including its 16 KL samples per state. Uncached FERPO samples K actions and evaluates Q at every actor epoch; cached FERPO reuses actions, rollout log probabilities, and Q values after critic training, recomputing weights as the multipliers change. Cached actor updates are faster than REPPO for $K = 4$–64; uncached updates become slower at $K \geq 32$ (Figure 2; protocol in Appendix A.2). At $K = 16$, uncached FERPO reduces combined actor-and-critic update time by 6.9% relative to REPPO. Caching provides a further 16.4% reduction, bringing the total saving to 22.2%. A drawback is higher GPU memory usage: on CheetahRun, peak GPU memory usage is 22.4% higher for uncached FERPO and 25.6% higher for cached FERPO than for REPPO, because PyTorch retains additional memory after updates.

5.3 Ablations

Number of SNIS samples. To test whether a larger sampling budget improves learning, we vary the number of rollout-policy action samples per state, $K \in \{2, 4, 8, 16\}$, on G1 flat terrain with adaptive temperature. Mean return improves as K increases, with smaller gains at larger sample counts suggesting diminishing returns. At 50.07M environment steps, increasing K from 8 to 16 raises mean return from 34.32 ± 0.18 to 34.81 ± 0.24 (mean $\pm$ standard error of the mean, SEM; Figure 3, left), while doubling the number of candidate actions evaluated.

Temperature. To assess the contribution of entropy regularization, we compare adaptive temperature with $\tau = 0$ at $K = 16$ using the same G1 evaluation budget. Without entropy regularization, mean return falls to 30.32 ± 0.27 (mean $\pm$ SEM; Figure 3, left), 4.49 points below the adaptive control. FERPO's maximum-entropy formulation therefore achieves higher mean return than the variant without entropy regularization on this task (see Appendix A.3.1 for details).

ESS versus KL control. On HopperHop and AcrobotSwingupSparse, both controllers support learning at suitable settings, with task-dependent best returns. We retain ESS control because it directly regulates SNIS weight concentration and shows lower temporal variation at comparable achieved ESS in these experiments (Appendix A.3.6).

Cached Q values. To test whether caching speeds up training while retaining performance, we complete critic training before the actor phase, freeze the critic, and cache candidate actions, Q values, and rollout log probabilities. Actor updates reuse these quantities while recomputing SNIS weights. On G1 with $K = 8$, cached and uncached updates reach 33.58 ± 0.40 and 34.03 ± 0.21 at 50.07M steps (mean $\pm$ SEM over five seeds; Figure 3, middle). Caching reduces mean time to complete this budget from 68.4 to 55.7 minutes, an 18.6% saving. It therefore trades a modest observed return gap for faster training. Details are provided in Appendix A.3.3.

Cached V extension. A learned Q function must approximate both the immediate reward and future returns, and may therefore fail to preserve the reward's shape across actions. Learning this shape

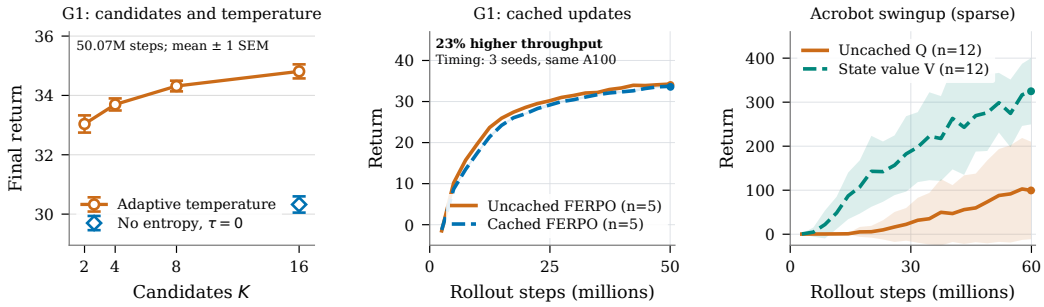


Figure 3: **Ablations.** Left: G1 candidate count and temperature at 50.07M steps (eight seeds for $K = 2, 16$ and $\tau = 0$; ten for $K = 4, 8$). Middle: cached versus uncached G1 updates ($K = 8$). Right: state-value versus Q critics on AcrobotSwingupSparse (sparse rewards, $K = 16$). Means with one SEM (left, middle) or one SD (right); panels use separate controls. The state-value variant uses $17\times$ more simulator transitions per rollout step.

can be particularly difficult in sparse reward settings, where few sampled transitions yield nonzero rewards. To mitigate this source of error, we extend the cached Q variant above by replacing its critic with a soft state-value function. For each rollout state, we simulate K candidate actions and, after fitting and freezing V, cache $r(s, a) + \gamma V(s')$ as a one-sample Monte Carlo estimate of $Q_V(s, a)$ for reuse throughout actor updates. This incorporates the observed immediate reward directly, at the cost of K additional environment calls per rollout state. In massively parallel online RL, these independent one-step simulations can be batched on the GPU (Zakka et al., 2025), and caching amortizes their cost across all actor epochs. With Warp graph execution and batched candidate simulation during rollout, cached V is only 8.4% slower per training iteration than uncached Q on AcrobotSwingupSparse with $K = 16$: 8.51 versus 7.85 seconds on an A100, measured after warmup. The K additional transitions nevertheless increase the total simulation budget by a factor of $K + 1 = 17$. Figure 3 (right) compares returns over 12 seeds. See Appendix A.3.4 for more details.

5.4 Learning benchmarks

We evaluate whether FERPO's policy updates translate into competitive final performance across three benchmark suites: 22 DeepMind Control Suite (DMC) tasks (Tassa et al., 2018) and four G1/T1 locomotion tasks in MuJoCo Playground (Zakka et al., 2025), and eight ManiSkill manipulation tasks (Tao et al., 2025). All baseline results come from the experiments accompanying the REPPO paper (Voelcker et al., 2026). For each benchmark suite, we train FERPO with exactly the same actor and critic architectures as REPPO. Alongside REPPO, we compare with PPO (Schulman et al., 2017) and PPO (Brax) separately on DMC, PPO on ManiSkill, and FastTD3 (Seo et al., 2025), a scalable variant of TD3 (Fujimoto et al., 2018), on DMC. For each of these additional baseline families, we select one configuration per suite using performance at 50M steps and keep that configuration fixed across the suite's tasks. Appendix A.3 provides the selection protocol and all per-environment learning curves. The G1/T1 FERPO results use $K = 16$ and normalized ESS target 0.65, with eight completed seeds per task.

Figure 4 aggregates results separately within each suite using the interquartile mean (IQM). Every environment receives equal total weight, divided equally among its available seeds; returns and success rates retain their original units. We focus the comparison on the final common evaluation budget: 50M interactions for DMC and G1/T1, and 49,971,200 interactions for ManiSkill. Individual IQM uncertainties use pointwise 95% bootstrap confidence intervals from 10,000 seed resamples within environments, matching Figure 4. Below, estimate $\pm$ half the bootstrap interval width gives a symmetric approximation; all significance statements use the original interval bounds.

Final performance. On DMC, FERPO reaches an IQM return of 897.7 ± 6.7. REPPO reaches 891.0 ± 6.4, FastTD3 890.6 ± 12.8, PPO (Brax) 542.8 ± 30.2, and PPO 309.8 ± 9.4. FERPO therefore has the highest point estimate, with a small numerical lead over REPPO and FastTD3 and a much larger lead over both PPO families. On G1/T1, REPPO reaches 34.63 ± 0.17, compared with

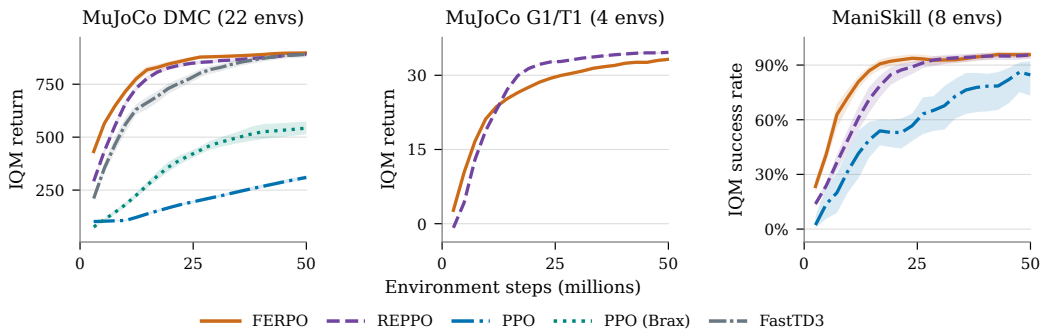


Figure 4: **Benchmark learning curves.** IQM with equal environment weights and equal seed weights within environments. Bands: pointwise 95% confidence intervals from 10,000 within-environment bootstrap resamples. G1/T1 FERPO uses $K = 16$, ESS target 0.65, and eight seeds per task. Details: Appendix A.3.

33.22 ± 0.28 for FERPO. FERPO's endpoint estimate is slightly higher on T1 flat and similar on T1 rough, while REPPO leads on both G1 tasks (Figure 14). On ManiSkill, FERPO's success-rate IQM is $(95.67 \pm 1.83)\%$; REPPO reaches $(95.52 \pm 1.85)\%$, and PPO reaches $(84.61 \pm 9.49)\%$. The two leading ManiSkill point estimates differ by only 0.15 percentage points.

Beyond final performance, FERPO reaches high returns on DMC and high success rates on ManiSkill with fewer environment interactions than REPPO (Figure 4). This sample-efficiency advantage complements its lower optimization cost: at $K = 16$, combined actor-and-critic updates take 6.9% less time in the matched A100 benchmark (Section 5.2). Together, these results suggest improved wall-clock efficiency.

6 Conclusion, Limitations, and Future Work

FERPO performs on-policy maximum-entropy policy improvement by fitting an entropy- and KL-regularized target through forward KL. Critic values weight actions sampled from the rollout policy using self-normalized importance sampling, avoiding critic action derivatives. An ESS controller regulates the concentration of these weights.

On DMC and ManiSkill, FERPO reaches high performance with fewer environment interactions than REPPO and achieves competitive final results. At $K = 16$, combined actor-and-critic update time is 6.9% lower than REPPO for uncached FERPO and we also evaluate cached FERPO with a 22.2% actor update speedup. Ablations on G1 flat terrain highlight the benefit of entropy regularization and gains from larger SNIS candidate sets. Limitations include critic-value errors, finite-sample importance-sampling errors, restricted Gaussian policies, and higher GPU memory use. With eight seeds per task, FERPO remains below REPPO on the G1/T1 suite aggregate at the same environment-interaction budget. The gap is concentrated on the G1 tasks; T1 endpoints are similar or slightly higher for FERPO under the recorded evaluation protocols.

State-value FERPO combines one-step simulation with a learned state-value critic to score candidate actions. Its higher returns on AcrobotSwingupSparse and the modest iteration-time overhead measured for its optimized implementation make it a particularly promising direction for future work. Two further directions are FERPO in discrete action spaces and combining FERPO with diffusion policies. For small discrete action spaces, target normalization and the forward-KL objective could be evaluated by exact summation over actions and larger ones through sampling. Diffusion-augmented MDPs (DA-MDPs) (Sanokowski & Patil, 2026) offer a promising and relatively straightforward route to combining FERPO with expressive diffusion policies: FERPO's forward-KL updates could be applied to individual denoising transitions in the augmented MDP.

ARTIFICIAL INTELLIGENCE (AI) USE STATEMENT

We used generative AI tools, including large language models (LLMs), to assist with writing and revising the manuscript, developing research code, and creating scientific figures. We also used these tools to help identify related literature on Q-function gradients and selected mathematical references relevant to our theoretical analysis. LLMs assisted in proofing the math for the derivation of the KL surrogate in Appendix B.1, the ESS-to-policy total-variation result in Appendix B.3.1, and the Gaussian trust-region comparisons in Appendix B.7. The authors checked AI-generated and AI-assisted text, code, and mathematical arguments for correctness. The authors take responsibility for the final manuscript, code, figures, mathematical claims, and experimental results.

REFERENCES

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1806.06920.

Denis Blessing, Julius Berner, Lorenz Richter, Carles Domingo i Enrich, Yuanqi Du, Arash Vahdat, and Gerhard Neumann. Trust region constrained measure transport in path space for stochastic optimal control and inference. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 165462–165510. Curran Associates, Inc., 2025. doi: 10.52202/085713-5516. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/f1ccd13e51f610ff4a7a08db9a8b458b-Paper-Conference.pdf.

Semih Cayci, Niao He, and R. Srikant. Convergence of entropy-regularized natural policy gradient with linear function approximation. *SIAM Journal on Optimization*, 34(3):2729–2755, 2024. doi: 10.1137/22M1540156. URL https://doi.org/10.1137/22M1540156.

Luc Devroye, Abbas Mehrabian, and Tommy Reddad. The total variation distance between high-dimensional Gaussians with the same mean, 2023. URL https://arxiv.org/abs/1810.08693v7. Version 7.

Pierluca D'Oro and Wojciech Jaśkowski. How to learn a useful critic? model-based action-gradient-estimator policy optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 313–324. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/03255088ed63354a54e0e5ed957e9008-Paper.pdf.

Jesse Farebrother, Jordi Orbay, Quan Vuong, Adrien Ali Taïga, Yevgen Chebotar, Ted Xiao, Alex Irpan, Sergey Levine, Pablo Samuel Castro, Aleksandra Faust, Aviral Kumar, and Rishabh Agarwal. Stop regressing: Training value functions via classification for scalable deep RL. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 13049–13071. PMLR, 2024. URL https://proceedings.mlr.press/v235/farebrother24a.html.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1587–1596. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/fujimoto18a.html.

Scott Fujimoto, Pierluca D'Oro, Amy Zhang, Yuandong Tian, and Michael Rabbat. Towards general-purpose model-free reinforcement learning. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 83509–83552, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/cf6501108fced72ee5c47e2151c4e153-Paper-Conference.pdf.

Swaminathan Gurumurthy, Zachary Manchester, and J Zico Kolter. Practical critic gradient based actor critic for on-policy reinforcement learning. In Nikolai Matni, Manfred Morari, and George J. Pappas (eds.), *Proceedings of The 5th Annual Learning for Dynamics and Control Conference*, volume 211 of *Proceedings of Machine Learning Research*, pp. 625–638. PMLR, 15–16 Jun 2023. URL https://proceedings.mlr.press/v211/gurumurthy23a.html.

Sehoon Ha, Joonho Lee, Michiel van de Panne, Zhaoming Xie, Wenhao Yu, and Majid Khadiv. Learning-based legged locomotion: State of the art and future perspectives. *The International Journal of Robotics Research*, 44(8):1396–1427, 2025.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1352–1361. PMLR, 2017. URL https://proceedings.mlr.press/v70/haarnoja17a.html.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 2018a. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b. URL https://arxiv.org/abs/1812.05905.

Elia Kaufmann, Leonard Bauersfeld, Antonio Loquercio, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, August 2023. ISSN 1476-4687. doi: 10.1038/s41586-023-06419-4. URL https://doi.org/10.1038/s41586-023-06419-4.

Mohammad Emtiyaz Khan and Håvard Rue. The Bayesian learning rule. *Journal of Machine Learning Research*, 24(281):1–46, 2023. URL https://jmlr.org/papers/v24/22-0291.html.

Tom Minka. Divergence measures and message passing. Technical Report MSR-TR-2005-173, Microsoft Research, 2005. URL https://www.microsoft.com/en-us/research/publication/divergence-measures-and-message-passing/.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. AWAC: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020. URL https://arxiv.org/abs/2006.09359.

Art B. Owen. *Monte Carlo Theory, Methods and Examples*. Online book, 2013. URL https://artowen.su.domains/mc/.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019. URL https://arxiv.org/abs/1910.00177.

Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In Aleksandra Faust, David Hsu, and Gerhard Neumann (eds.), *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 91–100. PMLR, 08–11 Nov 2022. URL https://proceedings.mlr.press/v164/rudin22a.html.

Sebastian Sanokowski and Kaustubh Patil. Diffusion-augmented Markov decision processes for maximum entropy reinforcement learning, 2026. URL https://arxiv.org/abs/2512.02019.

Elad Sarafian, Shai Keynan, and Sarit Kraus. Recomposing the reinforcement learning building blocks with hypernetworks. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9301–9312. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/sarafian21a.html.

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

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

Max Schwarzer, Ankesh Anand, Rishab Goel, R. Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=uCQfPZwRaUu.

Younggyo Seo, Carmelo Sferrazza, Haoran Geng, Michal Nauman, Zhao-Heng Yin, and Pieter Abbeel. FastTD3: Simple, fast, and capable reinforcement learning for humanoid control. *arXiv preprint arXiv:2505.22642*, 2025. URL https://arxiv.org/abs/2505.22642.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, October 2017. ISSN 1476-4687. doi: 10.1038/nature24270. URL https://doi.org/10.1038/nature24270.

Richard S. Sutton, David A. McAllester, Satinder P. Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 12, 1999. URL https://proceedings.neurips.cc/paper/1999/hash/464d828b85b0bed98e80ade0a5c43b0f-Abstract.html.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-Kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Demonstrating GPU parallelized robot simulation and rendering for generalizable embodied AI with ManiSkill3. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.021. URL https://www.roboticsproceedings.org/rss21/p021.html.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. DeepMind Control Suite, 2018. URL https://arxiv.org/abs/1801.00690.

Tim van Erven and Peter Harremoës. Rényi divergence and Kullback–Leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014. doi: 10.1109/TIT.2014.2320500. URL https://arxiv.org/abs/1206.2459.

Claas A. Voelcker, Tyler Kastner, Igor Gilitschenski, and Amir-massoud Farahmand. When does self-prediction help? understanding auxiliary tasks in reinforcement learning. *Reinforcement Learning Journal*, 4:1567–1597, 2024. URL https://rlj.cs.umass.edu/2024/papers/Paper197.html.

Claas A. Voelcker, Axel Brunnbauer, Marcel Hussing, Michal Nauman, Pieter Abbeel, Eric Eaton, Radu Grosu, Amir-massoud Farahmand, and Igor Gilitschenski. Relative entropy pathwise policy optimization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/3c2fd72a28fb98facf98546727320249-Abstract-Conference.html.

Christopher von Klitzing, Denis Blessing, Henrik Schopmans, Pascal Friederich, and Gerhard Neumann. Learning Boltzmann generators via constrained mass transport. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 88771–88810, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/8f38c9d65d89ce60a630b2619b383e55-Paper-Conference.pdf.

Kevin Zakka, Baruch Tabanpour, Qiayuan Liao, Mustafa Haiderbhai, Samuel Holt, Jing Yuan Luo, Arthur Allshire, Erik Frey, Koushil Sreenath, Lueder A. Kahrs, Carmelo Sferrazza, Yuval Tassa, and Pieter Abbeel. MuJoCo playground, 2025. URL https://arxiv.org/abs/2502.08844.

Yufeng Zhang, Jialu Pan, Kenli Li, Wanwei Liu, Zhenbang Chen, Xinwang Liu, and Ji Wang. On the properties of Kullback–Leibler divergence between multivariate Gaussian distributions. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 58152–58165. Curran Associates, Inc., 2023. doi: 10.52202/075280-2535.

Brian D. Ziebart. *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. PhD thesis, Carnegie Mellon University, December 2010. URL https://www.cs.cmu.edu/~bziebart/publications/thesis-bziebart.pdf.

APPENDIX CONTENTS

A Extended paper sections **15**

A.1 Motivating experiment details 15

A.2 Computation and memory on the A100 17

A.2.1 Update-cost protocol 17

A.2.2 Memory cost 17

A.2.3 Memory across rollout and update phases 17

A.3 Learning ablations and benchmarks 18

A.3.1 Ablation details 18

A.3.2 Critic-width protocol 19

A.3.3 Cached-Q timing protocol and elapsed-time curves 20

A.3.4 State-value critic with candidate simulation 21

A.3.5 Additional planned ablation 21

A.3.6 ESS versus policy-KL control 22

A.3.7 Data and comparison protocol 24

A.3.8 Pairwise comparisons of final benchmark performance 26

A.3.9 DMC tasks 27

A.3.10 G1/T1 locomotion and ManiSkill manipulation 29

A.4 FERPO pseudocode 31

A.5 State-dependent normalization in policy fitting 32

B Proofs **32**

B.1 Derivation of the KL surrogate 32

B.2 Derivation of the optimal regularized target 33

B.3 Log ESS and divergence control 34

B.3.1 From target ESS to policy total variation 34

B.4 Convergence under exact fitting 35

B.5 Update directions and importance sampling 36

B.5.1 Cost gradients under the policy and the target 36

B.5.2 Recovering the target-mean direction from samples 38

B.6 From surrogate improvement to return improvement 39

B.7 Trust regions for Gaussian policies 45

B.7.1 Forward KL bounds reverse KL 45

B.7.2 TV bounds both KL directions 46

B.7.3 Implications for the return bound 47

A Extended paper sections

A.1 Motivating experiment details

We expand on the bimodal experiment in Section 5.1. The cost is $c(a) = (a^2 - 1)^2 - 0.1a$, and the critic is exact, $Q = -c$. We use Gaussian policies with fixed standard deviation $\sigma = 0.4$, temperature $\tau = 0.1$, and FERPO multiplier $\lambda = 0.05$. The reference means are $\mu_0 = -1, -0.6, 0$ (A–C). Each loss curve keeps the reference policy and its target fixed while varying the candidate mean ν.

In Figure 1, each direction uses a newly constructed target, frozen for differentiation. Filled green and open red arrows indicate geometric direction toward and away from the global cost minimum, not immediate cost decrease or eventual convergence. Dotted lines mark the best fixed-width Gaussian mean; black dots mark the references and diamonds visible FERPO loss minima. All candidates in the local loss windows satisfy REPPO's KL bound 0.1.

Exact actor losses. We compute REPPO's expected cost analytically and FERPO's target mean by numerical quadrature. REPPO uses its cost-and-entropy loss while $k(\nu) = D_{\mathrm{KL}}(\pi_{\mu_0} \| \pi_\nu) < 0.1$, and the loss $0.1k(\nu)$ otherwise. Figure 5 shows this full loss at B, together with forward and reverse KL to the same FERPO target. We scale both KL losses by $T = \tau + \lambda = 0.15$ to express them in cost units. At B, the target mean is $m_* \simeq -0.466$, so forward fitting moves right, whereas REPPO and reverse KL initially move left.

Estimating the target from samples. At B, sampling from the rollout policy recovers FERPO's rightward direction in only 26.4% of batches with $K = 1{,}024$. To study sampling coverage, we draw actions from $r_\kappa = \mathcal{N}(\mu_0, \kappa^2\sigma^2)$ with $\kappa \in \{1, 2, 3\}$, keeping the rollout policy and target fixed. The proposal-corrected weights and estimated target mean are

$$u_i = \frac{e^{-c(a_i)/T}\pi_{\mu_0}(a_i)^{\lambda/T}}{r_\kappa(a_i)}, \qquad w_i = \frac{u_i}{\sum_j u_j}, \qquad \widehat{m} = \sum_i w_i a_i. \tag{11}$$

We use 5,000 batches for each $K \in \{64, 1{,}024\}$, with paired standard-normal draws across proposal widths. An update points right when $\widehat{m} > \mu_0$. Table 1 reports the resulting direction frequencies and target-mean estimates. For each batch, we use the SNIS weights to estimate FERPO's fitting loss as a function of the candidate policy mean. Figure 6 shows the median and pointwise 5th–95th percentiles of these estimated loss curves across 5,000 batches, together with the exact population loss. Wider proposals sample the better right well more often and recover the rightward direction in 98.9% and 99.9% of batches at $K = 1{,}024$.

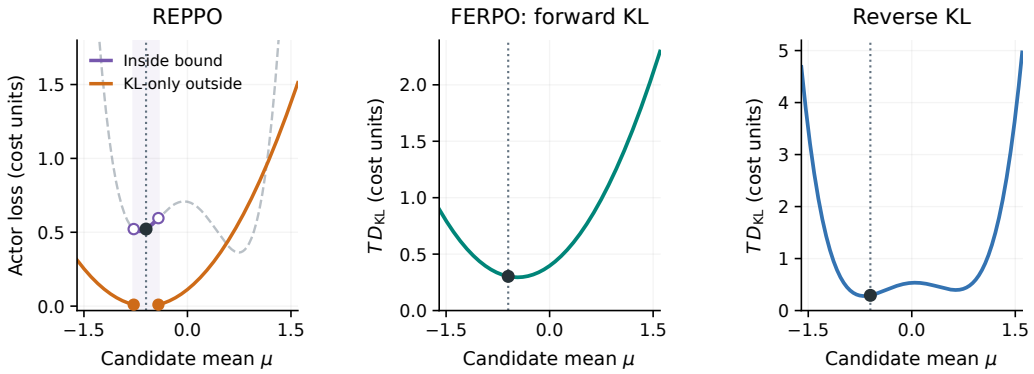


Figure 5: **Frozen-reference losses at B.** Left: REPPO's active expected-cost/entropy branch (purple) and KL-only branch (orange); the pale dashed curve extends the inactive cost branch. Open and filled endpoints show the loss switch. Middle: forward KL from the fixed FERPO target to the candidate Gaussian. Right: reverse KL to that same target. Both KL curves are scaled by $T = 0.15$; this positive scaling preserves directions. Vertical dotted lines and black dots mark the reference $\mu_0 = -0.6$.

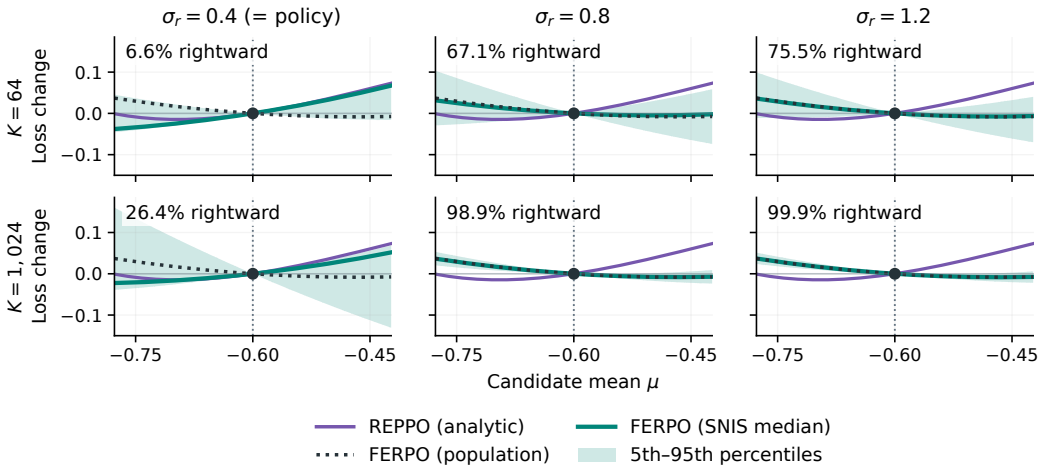


Figure 6: **Wider proposals recover the population fitting loss.** Losses at B with the same reference policy and target throughout. Teal: SNIS median and pointwise 5th–95th percentiles across 5,000 batches; dotted black: population FERPO; purple: analytic REPPO. Percentages give the frequency of rightward SNIS directions. Shading shows batch variation, not confidence intervals for the mean. Losses are zeroed at the reference, and forward losses are scaled by T.

Proposal σ_r	K	Median $\widehat{m}$	5th–95th percentiles	Rightward
0.4	64	-0.916	$[-0.963, -0.416]$	6.6%
0.8	64	-0.498	$[-0.863, -0.066]$	67.1%
1.2	64	-0.472	$[-0.753, -0.091]$	75.5%
0.4	1,024	-0.824	$[-0.923, 0.275]$	26.4%
0.8	1,024	-0.468	$[-0.565, -0.369]$	98.9%
1.2	1,024	-0.466	$[-0.546, -0.381]$	99.9%

Table 1: **SNIS target-mean estimates at B.** The population target mean is $-0.465943\ldots$ and rightward movement means $\widehat{m} > -0.6$. Intervals are the empirical 5th–95th percentiles across 5,000 batches, not confidence intervals. Policy standard deviation remains 0.4 in every row.

A.2 COMPUTATION AND MEMORY ON THE A100

A.2.1 UPDATE-COST PROTOCOL

The update-cost measurements use one NVIDIA A100-SXM4-40GB with AMD EPYC 7H12 host CPUs, PyTorch 2.9.1+cu128, FP32 parameters, matmul precision `medium`, and one PyTorch CPU thread, without AMP or compilation. We use the official REPPO PyTorch implementation at commit `05a5776` and the published large-data MuJoCo Playground configuration described in Section 5.2.[1] Each rollout has 131,072 states, with eight epochs of 64 minibatches of 2048. The shared baseline has 17 observation and six action dimensions, a three-layer width-512 actor, and two-layer width-512 critic encoder, Q head, and auxiliary predictor. FERPO evaluates all $2{,}048K$ candidate state-action pairs in each minibatch together.

FERPO completes critic training before its actor phase; REPPO retains its interleaved update schedule. Three warmup minibatches precede each trial. We use seeds 1–3 and randomize method order within each repetition. CUDA events measure actor time; synchronized wall time measures the complete update. Peak allocated PyTorch memory covers the complete update, including rollout tensors and optimizer state, but excludes simulator memory and allocator reservation.

A.2.2 MEMORY COST

REPPO backpropagates through Q for one action per state; its 16 KL samples require policy evaluations only. FERPO instead evaluates many actions together through Q. Even without retaining a backward graph, these larger batches require larger temporary tensors. FERPO's peak allocated memory therefore increases with K (Figure 2).

Cached FERPO additionally stores K actions, reference log probabilities, and Q values for each of S rollout states. With action dimension d and FP32 storage, this requires

$$M_{\text{cache}} = 4SK(d+2) \text{ bytes.} \tag{12}$$

For the measured rollout size at $K = 16$, the cache occupies 64 MiB for $d = 6$ and 528 MiB for $d = 64$.

A.2.3 MEMORY ACROSS ROLLOUT AND UPDATE PHASES

We separately profile CheetahRun with the same network and rollout configuration at $K = 16$. The matched comparison uses seeds 2 and 3 on one A100 node, with a fresh process per method and seed, two warmup iterations, and three measured iterations. The simulator and a 32-environment evaluation instance remain resident; initialization and evaluation are excluded from the measurement windows. NVML samples total process memory every 5 ms, including runtime and allocator reservations; these sampled peaks are lower bounds. PyTorch records allocated and reserved peaks separately for each phase. We exclude seed 1 from this comparison because uncached FERPO ran on a second A100 with a different driver.

	Process peak		PyTorch allocated peak		PyTorch reserved
Method	Rollout	Update	Rollout	Update	Rollout
REPPO	1,438	1,408	601	422	614
FERPO	1,760	1,728	600	658	936
Cached FERPO	1,806	1,774	600	723	982

Table 2: **Memory across rollout and update phases (MiB).** Peaks are identical across the two matched seeds. Reserved memory includes active tensors.

All three methods reach their process peak during rollouts (Table 2). Active PyTorch rollout allocations are nearly equal, while larger pools remain reserved after FERPO's updates. These process footprints therefore reflect allocator retention as well as live tensors, rather than minimum GPU capacity requirements.

[1] Pinned REPPO source; the PyTorch and JAX implementations both use 16 samples for the KL estimate.

A.3 Learning ablations and benchmarks

A.3.1 Ablation details

The candidate-count ablation uses per-seed G1JoystickFlatTerrain evaluation logs for $K = 2, 4, 8, 16$. All four settings use actor/critic and temperature learning rates of 0.0003, entropy target factor -0.5, and normalized ESS target 0.85. The replacements correct the historical $K = 2$ entropy target of -1.0 and $K = 16$ temperature learning rate of 0.003; those superseded observations are excluded from the plots. We report arithmetic means with one-SEM error bars and bands. The temperature comparison reuses the corrected $K = 16$ control. These ablation cohorts remain separate from the G1/T1 benchmark cohort below, which uses normalized ESS target 0.65.

The candidate-count plot in Figure 3 (left) uses the final common evaluation at 50,069,504 environment steps, rather than selecting a best checkpoint or averaging a trailing window. There are eight seeds at this endpoint for $K = 2, 16$, ten for $K = 4, 8$, and eight for the additional $\tau = 0$ marker at $K = 16$. The no-entropy learning curve in Figure 7 uses the same eight completed seeds at every evaluation. The corrected adaptive-temperature control uses seeds 1–8. Both temperature curves and all candidate-count endpoints show mean $\pm$ one SEM. All eight no-entropy runs reach 50,069,504 steps, with final mean 30.32 ± 0.27. They disable entropy regularization and its temperature update. The entropy target changed from -1.0 in the historical runs to -0.5 in the new runs, but is inactive when $\tau = 0$. For all G1 ablations, including cached-Q and critic-width comparisons, we compute $\mathrm{SEM} = s/\sqrt{n}$, where s is the sample standard deviation (with $n - 1$ in its variance denominator) and n is the number of contributing seeds. These are standard errors of the mean, not 95% confidence intervals. Benchmark curves use IQM and bootstrap intervals as specified below.

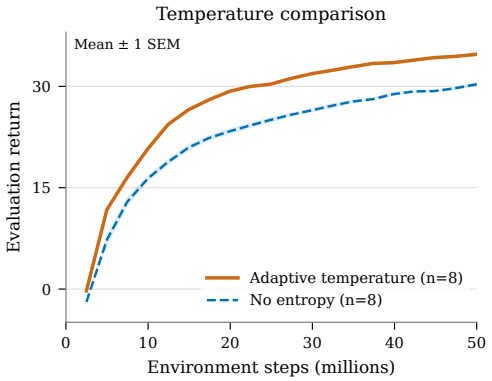


Figure 7: **Temperature learning curves.** Adaptive temperature versus no entropy regularization at $K = 16$. Both curves show seed means with one-SEM bands. Both curves use eight completed seeds at every evaluation; the adaptive control uses corrected temperature learning rate 0.0003. Final returns are 34.81 ± 0.24 with adaptive temperature and 30.32 ± 0.27 without entropy regularization. The final evaluations also appear in Figure 3 (left).

A.3.2 CRITIC-WIDTH PROTOCOL

The width comparison uses the uploaded G1 preset on G1JoystickFlatTerrain with learning rate 3×10^{-4}, target entropy factor -0.5, eight critic epochs, HL-Gauss loss, and auxiliary weight one. Both methods use the full rollout. FERPO uses $K = 8$; REPPO retains its pathwise actor update and KL bound 0.1. We vary hidden width in the critic encoder, Q head, and auxiliary predictor together; depth and the 512-wide actor are fixed. Each endpoint is the final deterministic evaluation at 50,069,504 rollout interactions. Seeds 1–2 contribute at each width except REPPO width 1,024, for which only seed 1 is available. These are dedicated critic-width runs, separate from the published REPPO benchmark results. We have not measured action-gradient approximation error in these runs.

At 50.07M interactions, reducing critic width from 124 to 62 lowers FERPO's mean return from 23.04 to 6.29 and REPPO's from 22.12 to 4.19. Both methods therefore rely on sufficient critic capacity.

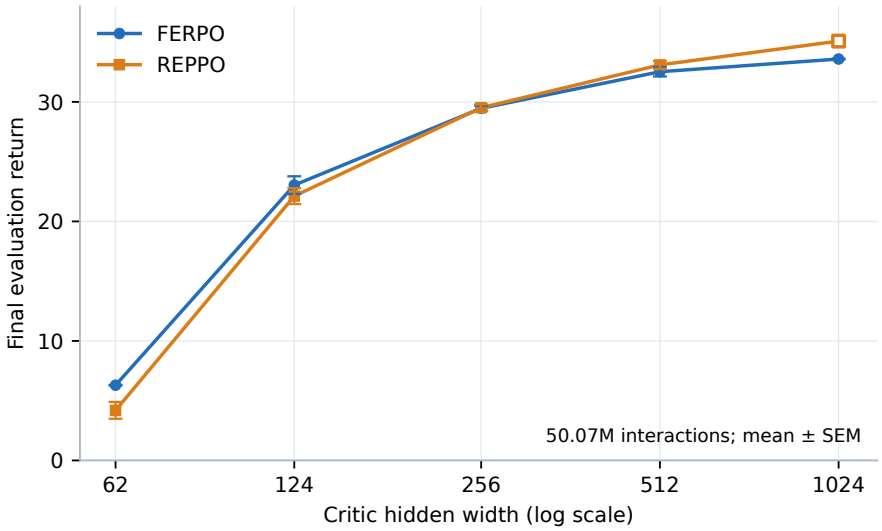


Figure 8: **Final return versus critic width on G1.** Points show seed means and error bars show one SEM. The open square marks the single-seed REPPO reference at width 1,024, for which no error bar is reported. Both methods lose substantial return at width 62; FERPO reaches 6.29 and REPPO 4.19. These endpoints do not identify the source of critic approximation error.

A.3.3 CACHED-Q TIMING PROTOCOL AND ELAPSED-TIME CURVES

We compare interleaved critic/actor updates with a staged variant that first fits the critic, then freezes it and caches $K = 8$ actions, their Q values, and rollout log probabilities. Actor training recomputes SNIS weights as the multipliers change. Both methods use eight full-rollout critic epochs, 1,024-wide critics, learning rate 3×10^{-4}, and target entropy factor -0.5. The five-seed rollout curves are unsmoothed. Final mean returns at 50,069,504 steps are 34.03 ± 0.21 uncached and 33.58 ± 0.40 cached (mean $\pm$ SEM).

The timing annotation in Figure 3 (middle) and elapsed-time curves in Figure 9 use seeds 3–5 for each method. These six runs shared the same Vega allocation and node, with one NVIDIA A100-SXM4-40GB and a separate CPU set per run. Seeds 1–2 remain in the five-seed rollout comparison in Figure 3 (middle), but are excluded from timing because their cached and uncached runs used different clusters. The per-run clock starts before environment construction and includes simulation, learning, evaluation, and checkpointing; scheduler waiting time is excluded. Mean completion times at 50,069,504 rollout steps are 55.7 minutes cached and 68.4 minutes uncached. The time reduction is $1 - T_{\text{cached}}/T_{\text{uncached}} = 18.6\%$; the throughput increase is $T_{\text{uncached}}/T_{\text{cached}} - 1 = 22.9\%$. These are measured runtimes for this allocation.

We use recorded evaluation timestamps for the time curves. Within each method, we linearly interpolate each seed's return to common elapsed times, then compute the mean and SEM over all three seeds. The bold mean curve ends when the first run in that method finishes; faint individual curves continue to their own final evaluation. We do not extrapolate or carry completed returns forward. The arrow separately marks the mean completion times from the run summaries, including the final checkpoint. The five-seed curves in Figure 3 (middle) use recorded rollout-step coordinates directly, without interpolation.

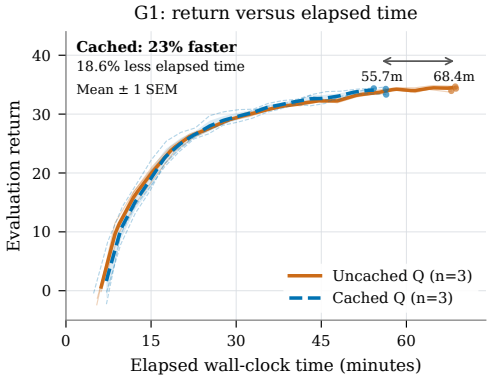


Figure 9: **Cached versus uncached Q against elapsed time.** G1 deterministic evaluation return for seeds 3–5 per method on the same Vega A100 node. Both use $K = 8$ and eight critic epochs on the full rollout. Bold curves and one-SEM bands use linear interpolation at common times; faint curves show individual runs. The arrow marks mean completion times at 50,069,504 interactions: 55.7 versus 68.4 minutes. At 50 minutes, interpolated mean returns are 33.55 cached and 32.86 uncached. Figure 3 (middle) shows all five seeds against environment interactions.

A.3.4 STATE-VALUE CRITIC WITH CANDIDATE SIMULATION

We compare standard uncached FERPO with its cached V extension on AcrobotSwingupSparse. For a fixed fitted critic, the ideal candidate score is $Q_V(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[V_\psi(s')]$. We approximate the transition expectation with one simulated successor per candidate, giving $\widehat{Q}_V(s,a) = r(s,a) + \gamma V_\psi(s')$. This incorporates the observed immediate reward directly while using a one-sample Monte Carlo estimate of the future value. Scores are computed after fitting V_ψ and held fixed throughout actor training. The bootstrap is zero at true terminals; time limits retain the value of the final observation.

Both methods use $K = 16$, 512-wide critics, eight critic epochs, HL-Gauss loss, auxiliary weight one, learning rate 10^{-3}, and target entropy factor -1. Across 12 matched seed identifiers, cached V reaches 324.79 ± 75.28 at 59.90 million rollout steps, compared with 99.46 ± 110.75 for uncached Q (mean $\pm$ one SD; Figure 3, right). The deterministic evaluation curves are unsmoothed. Each rollout step requires K additional candidate transitions, so cached V uses 17 times as many simulator transitions at this rollout budget.

Efficient candidate simulation. We combine Warp graph execution with parallel simulation of all K candidates across environments. Candidates are sampled and simulated during rollout, branching from each state without altering the rollout trajectory. Their rewards and next observations remain on the GPU until the critic has been fitted, then serve to construct the score cache. This avoids repeated transfers of simulator states between CPU and GPU. Actor epochs reuse the candidates and scores while updating their SNIS weights. Since observation normalization can change during rollout, we retain the sampling densities and correct these weights for any difference from the frozen rollout reference.

Timing protocol. We benchmark both methods on the same NVIDIA A100-SXM4-40GB with Warp graphs enabled. Both use 1,024 environments, 128 rollout steps per iteration, actor and critic widths of 512, eight actor and critic epochs, 64 minibatches, and one PyTorch CPU thread. We run three repetitions in separate processes, randomizing method order and averaging five iterations after two warmup iterations. GPU synchronization at iteration boundaries ensures that timings include all simulation and learning work; startup, compilation warmup, evaluation, external logging, and checkpointing are excluded. Cached V takes 8.508 seconds per iteration versus 7.846 seconds for uncached Q, an 8.4% overhead.

A.3.5 ADDITIONAL PLANNED ABLATION

The following additional G1 comparison remains planned. It will use matched seeds, interaction budgets, and evaluation protocols, keeping other settings fixed.

Forward KL instead of ESS regularization. We will replace the log-ESS constraint with $\mathbb{E}_{s \sim d_{\text{roll}}}[D_{\text{KL}}(\pi_{\text{roll}}(\cdot \mid s) \| \pi_\theta(\cdot \mid s))] \leq 0.1$, matching REPPO's bound. The action expectation in this forward KL is under the old rollout policy. We will adapt λ using the KL constraint gap and compare return, achieved KL, and normalized ESS against the ESS-controlled baseline.

A.3.6 ESS VERSUS POLICY-KL CONTROL

We compare log-ESS control (Eq. 8) at $e_{\text{target}} \in \{0.40, 0.50, 0.65, 0.85\}$ with control of the rollout-state mean $D_{\text{KL}}(\pi_{\text{roll}} \| \pi_{\text{curr}})$ at $\epsilon \in \{0.03, 0.10, 0.20, 0.30\}$ nats. Both adapt λ with the same forward-KL actor objective. ESS controls SNIS weight concentration; policy KL measures policy displacement.

We use uncached FERPO ($K = 16$) on HopperHop and AcrobotSwingupSparse with 12 seeds per setting. All runs use width-512 critics, eight epochs, 64 minibatches, HL-Gauss loss, auxiliary weight one, actor/critic and temperature learning rates 10^{-3}, and entropy target factor -1. Rollouts comprise 1,024 environments and 128 steps. Final returns average the first episode from each of 128 deterministic evaluation environments, with frozen normalizers and fixed initial states within each run.

Both controllers support learning at suitable settings, with task-dependent preferred regularization strengths and no consistent return advantage for either controller (Figure 10).

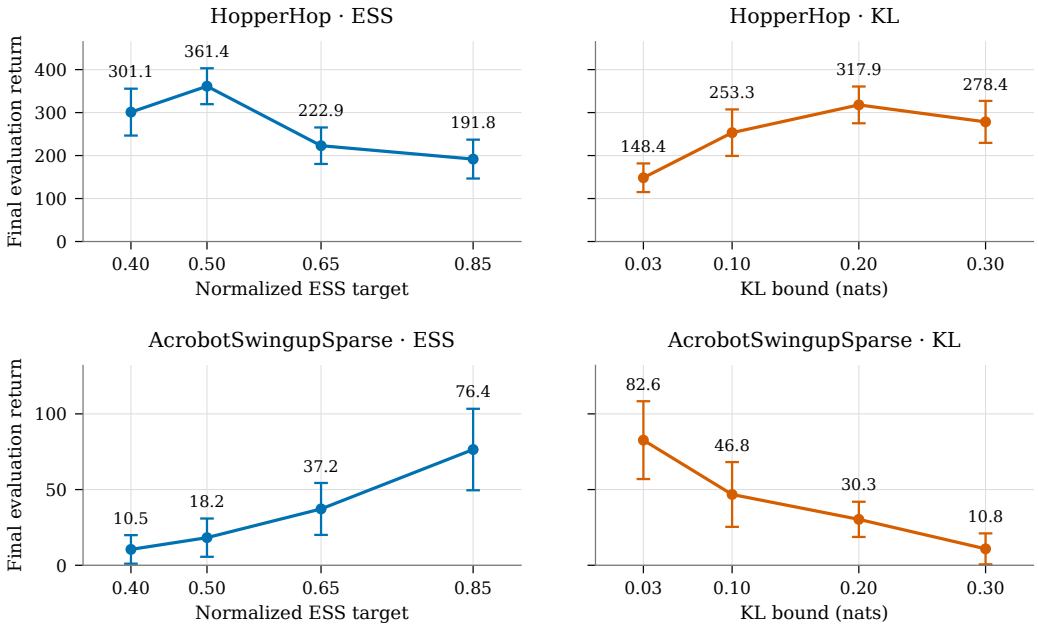


Figure 10: **Final return versus controller target.** HopperHop (top) and AcrobotSwingupSparse (bottom), with ESS targets (left) and policy-KL bounds (right). Points show mean final return at 50.07M interactions with one SEM over 12 seeds.

Direct ESS control generally tracks its target, whereas fixed policy-KL bounds produce ESS that varies across tasks and training (Figure 11). At comparable mean achieved ESS, direct control shows less temporal variation, although the aggressive ESS target 0.40 remains variable on Acrobot. These observations motivate ESS as a direct, interpretable control of sampling-weight concentration.

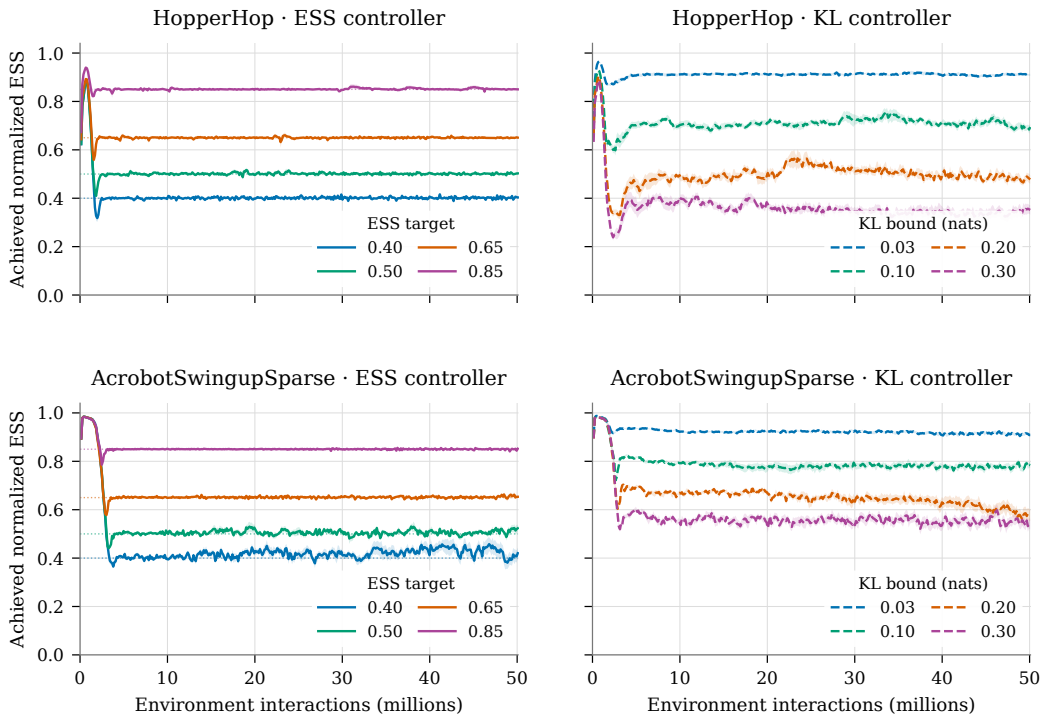


Figure 11: **Achieved ESS under both controllers.** Per-iteration geometric mean of normalized SNIS ESS under direct ESS control (left) and policy-KL control (right). Curves show unsmoothed means with one-SEM bands across 12 seeds. Dotted lines mark ESS targets; right-panel legends give policy-KL bounds in nats.

A.3.7 DATA AND COMPARISON PROTOCOL

The benchmark comprises 22 DMC, four G1/T1, and eight ManiSkill tasks. FERPO uses training-log return evaluations for 19 DMC tasks and success rates for ManiSkill. The three Walker tasks use separate policy-evaluation CSVs, whose task metadata and reconstructed means and counts match the original benchmark summaries. Walker has ten evaluated seeds per task beyond 50M steps, replacing the older partial training-log evaluations. For Walker, we display the available positive checkpoint evaluations from 3,014,656 through 50,987,008 interactions; seeds 11–12 have no exported evaluation curves and are not counted. The other 19 DMC tasks retain their own recorded evaluation schedules. These existing evaluations use deterministic actions and report 1,000-step episode lengths, but sampling settings differ: the older training evaluator uses 1,024 environments, the newer one uses 128, and the Walker checkpoint evaluator uses 128 episodes. Initial evaluation states are fixed within a run rather than refreshed at every checkpoint. Thus these are existing evaluation exports, not a reevaluation under one identical sampling protocol. The G1/T1 benchmark uses the completed $K = 16$, normalized ESS target 0.65 runs, with seeds 1–8 for each task. Actor/critic and temperature learning rates are 0.0003, and the entropy target factor is -0.5. We use all 21 recorded deterministic training evaluations through 50,069,504 interactions, averaging the first episode from each of 128 parallel environments. The ManiSkill source label "FEPPO architecture" denotes FERPO. We retain all available evaluated seeds within their observed ranges, so counts may vary across checkpoints; training-only runs contribute no observations. Each DMC panel labels the number of contributing seeds for every method; a range indicates changing availability along the displayed curve. We retain the common 50M budget for numerical method comparisons and do not extrapolate baselines to the last Walker checkpoint.

Baseline sources and configuration selection. All baseline results are drawn from the experiments accompanying the REPPO paper (Voelcker et al., 2026). REPPO uses the official per-task trial CSVs at commit `05a5776`.[2] The local alternative REPPO configurations and baselines embedded in FERPO's summary files are excluded. Following Figure 15 of Voelcker et al. (2026), we distinguish PPO and PPO (Brax) on DMC. We use the available CSV exports at their recorded step coordinates; these are not verified as the exact trials behind that published figure. We neither rescale their training horizons nor combine the two PPO families. For the remaining methods, we compare configurations separately by method family and suite (DMC, G1/T1, or ManiSkill). A candidate must cover every task available for its method in that group and span the 50-million-step selection budget. We linearly interpolate each task's seed-mean return to that budget. For DMC, we min–max normalize each task's score across eligible configurations of the same method, assign 0.5 when all candidates tie, and average over tasks. For ManiSkill, we average success rates directly. The highest-scoring configuration is then fixed across all tasks in its group, with alphabetical tie-breaking. This is retrospective selection on the displayed runs, rather than selection on an independent validation set.

Group	Method	Selected configuration or source
DMC	REPPO	Official published trials
DMC	PPO	`ppo_4`
DMC	PPO (Brax)	`ppo_brax`
DMC	FastTD3	`fasttd3_small`
G1/T1	REPPO	Official published trials
ManiSkill	REPPO	Official published trials
ManiSkill	PPO	`maniskill_ppo_short`

Table 3: Baseline sources, fixed across tasks within each group. The archive contains no PPO or FastTD3 G1/T1 curves, and no FastTD3 ManiSkill curves.

Statistics and display. All benchmark methods show IQM over the available independent seeds. We compute IQM by integrating the middle 50% of empirical probability mass, including fractional observations at quartile boundaries. FERPO runs are aligned on the union of their recorded evaluation steps, using linear interpolation only between observed evaluations of the same run. We do not extrapolate or fill in missing run endpoints. At each checkpoint, we resample the available seeds with replacement 10,000 times and report the 2.5th and 97.5th percentiles, using fixed random seeds. Checkpoints with the same available seeds share the same trajectory resampling; a changed

[2] Official REPPO result CSVs at the pinned commit.

seed cohort is resampled separately. At least two observed seeds are required. These are pointwise 95% intervals conditional on the observed seeds and selected configuration; they do not correct for missing runs or configuration search. Baselines have complete trial grids and use whole-trajectory resampling. FERPO uses exactly the same actor and critic architectures as REPPO for each benchmark suite.

REPPO evaluations labeled step zero are placed at 2,490,368 environment steps; nonzero REPPO coordinates and all other methods' coordinates remain as recorded. Figures display up to 50 million steps without smoothing or extrapolation, except that the DMC task panels extend to the recorded Walker checkpoint at 50,987,008. FERPO's ManiSkill curves end at their last recorded evaluation, 49,971,200 steps. Bootstrap success-rate intervals lie within $[0, 1]$ without clipping.

Aggregation across environments and seeds. Figure 4 pools scores within each suite. For T environments and n_t available seeds in environment t, each seed receives weight $1/(Tn_t)$ before IQM trimming. Scores retain their original units without cross-task normalization. We use 21 shared checkpoints, requiring at least two seeds per environment and method. The common ranges are 3.01–50M steps for DMC, 2.49–50M for G1/T1, and 2.49–49.97M for ManiSkill. The 10,000-resample seed bootstrap is stratified by environment, keeping the environment set and its equal weights fixed. In particular, each of the three Walker tasks contributes $1/22$ of the DMC probability mass before trimming, split among its ten evaluated seeds: $1/220$ per seed. A task with twenty evaluated seeds still contributes $1/22$, with $1/440$ per seed. We do not duplicate Walker seeds or discard seeds from other tasks to equalize counts. At 50M, FERPO contributes 372 task–seed observations across the 22 tasks (7–20 per task). The aggregate uses the same positive FERPO evaluations as the task panels; its first checkpoint is 3,014,656 steps, and it ends at the common 50M budget without extrapolating baselines to Walker's last checkpoint.

A.3.8 PAIRWISE COMPARISONS OF FINAL BENCHMARK PERFORMANCE

The endpoint comparison uses the same tasks, available seeds, configuration choices, and weighting as Figure 4. All methods within a suite are compared at its final common budget: 50,000,000 steps for DMC and G1/T1, and 49,971,200 steps for ManiSkill. Each seed contributes its linearly interpolated evaluation score only when that budget lies within its observed range. Individual IQMs and their marginal 95% intervals in Section 5.4 are the existing 10,000-resample Figure 4 estimates.

For the pairwise analysis, we independently resample seeds with replacement within each fixed environment and method, preserving each environment's seed count and total weight. Numerical seed labels do not establish paired runs across methods, so we do not use a paired-method bootstrap. For each of 100,000 resamples, we compute the suite IQM for each method and subtract the two IQMs of interest. We retain the environment set rather than resampling environments. The procedure uses deterministic method-specific random seeds and reproduces the plotted endpoint IQMs before computing differences.

There are fourteen comparisons: ten on DMC, one on G1/T1, and three on ManiSkill. We use quantiles $0.025/14$ and $1 - 0.025/14$ of each bootstrap difference distribution. These are approximately 99.643% per-comparison intervals, corresponding to a Bonferroni correction that targets at least 95% coverage for the family of fourteen comparisons, subject to the bootstrap approximation. A difference interval entirely above zero supports higher IQM for method A; an interval entirely below zero supports method B. An interval spanning zero is inconclusive and does not establish equal performance. The intervals condition on the observed seeds and selected configurations; they do not account for configuration search or informative missingness.

Suite	Method A − method B	IQM difference	Adjusted interval
MuJoCo DMC	FERPO − REPPO	6.67	$[-6.89, 20.65]$
MuJoCo DMC	FERPO − FastTD3	7.10	$[-11.82, 30.90]$
MuJoCo DMC	FERPO − PPO	587.89	$[570.19, 604.70]$
MuJoCo DMC	FERPO − PPO (Brax)	354.86	$[307.06, 398.14]$
MuJoCo DMC	REPPO − FastTD3	0.44	$[-18.30, 23.87]$
MuJoCo DMC	REPPO − PPO	581.22	$[563.44, 597.76]$
MuJoCo DMC	REPPO − PPO (Brax)	348.19	$[300.83, 391.24]$
MuJoCo DMC	FastTD3 − PPO	580.78	$[554.61, 602.24]$
MuJoCo DMC	FastTD3 − PPO (Brax)	347.76	$[297.07, 393.03]$
MuJoCo DMC	PPO − PPO (Brax)	-233.02	$[-281.80, -188.12]$
MuJoCo G1/T1	FERPO − REPPO	-1.41	$[-1.94, -0.94]$
ManiSkill (pp)	FERPO − REPPO	0.15	$[-3.75, 4.31]$
ManiSkill (pp)	FERPO − PPO	11.07	$[0.67, 27.83]$
ManiSkill (pp)	REPPO − PPO	10.91	$[0.40, 27.78]$

Table 4: **Pairwise differences in final suite IQM.** Intervals use 100,000 seed bootstrap resamples and a Bonferroni correction across all fourteen comparisons (95% family-wise target). MuJoCo differences are in return units; ManiSkill differences are in percentage points (pp). Positive values favor method A. The G1/T1 comparison uses eight FERPO seeds per task.

A.3.9 DMC TASKS

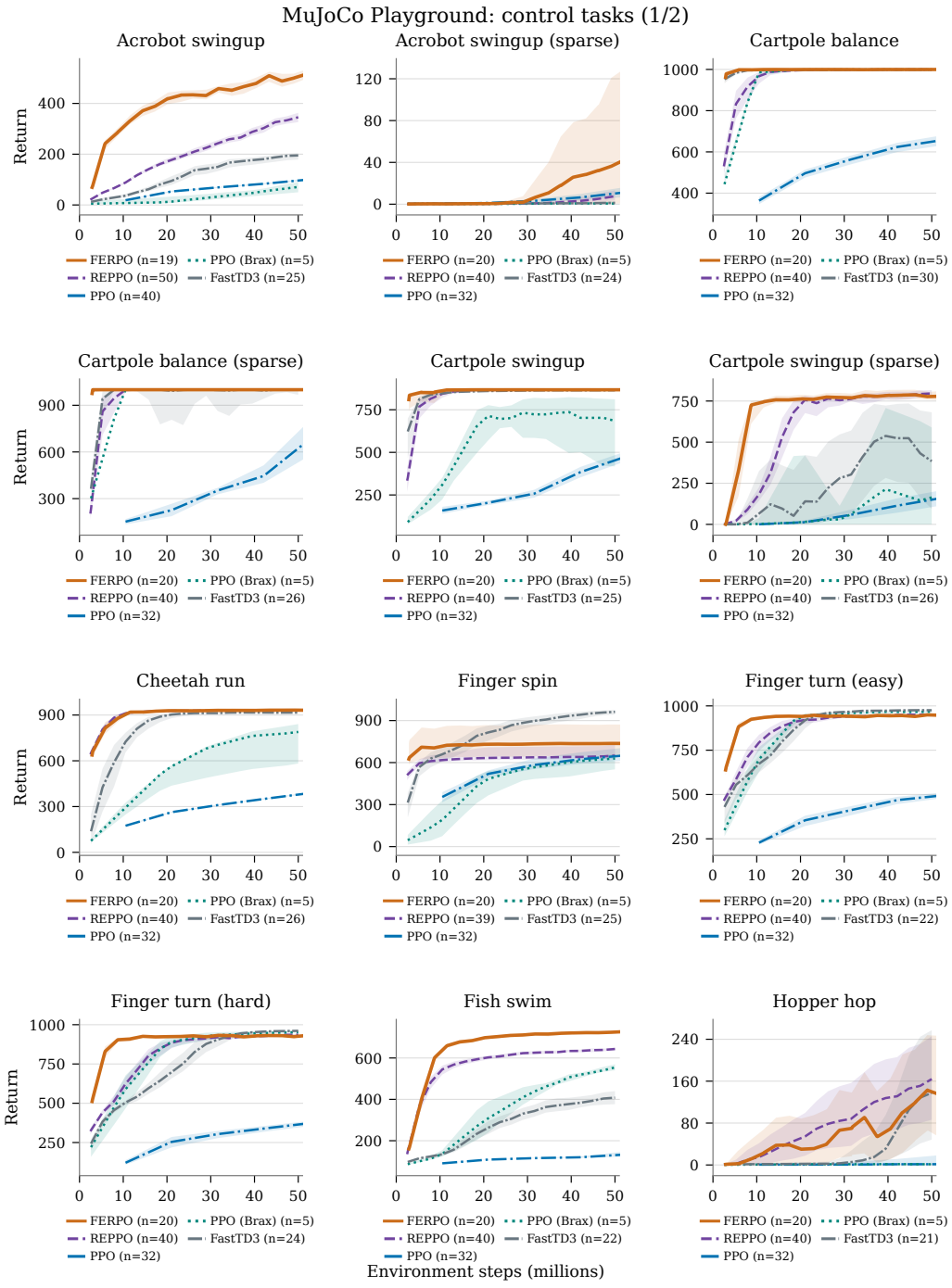


Figure 12: **DMC learning curves (1/2).** The first 12 of 22 tasks. All methods show IQM with pointwise 95% bootstrap confidence intervals. Legends report contributing seeds n; ranges indicate changes across the displayed evaluations. Curves retain their source evaluation schedules. Baseline configurations are fixed across all DMC tasks as specified in Table 3.

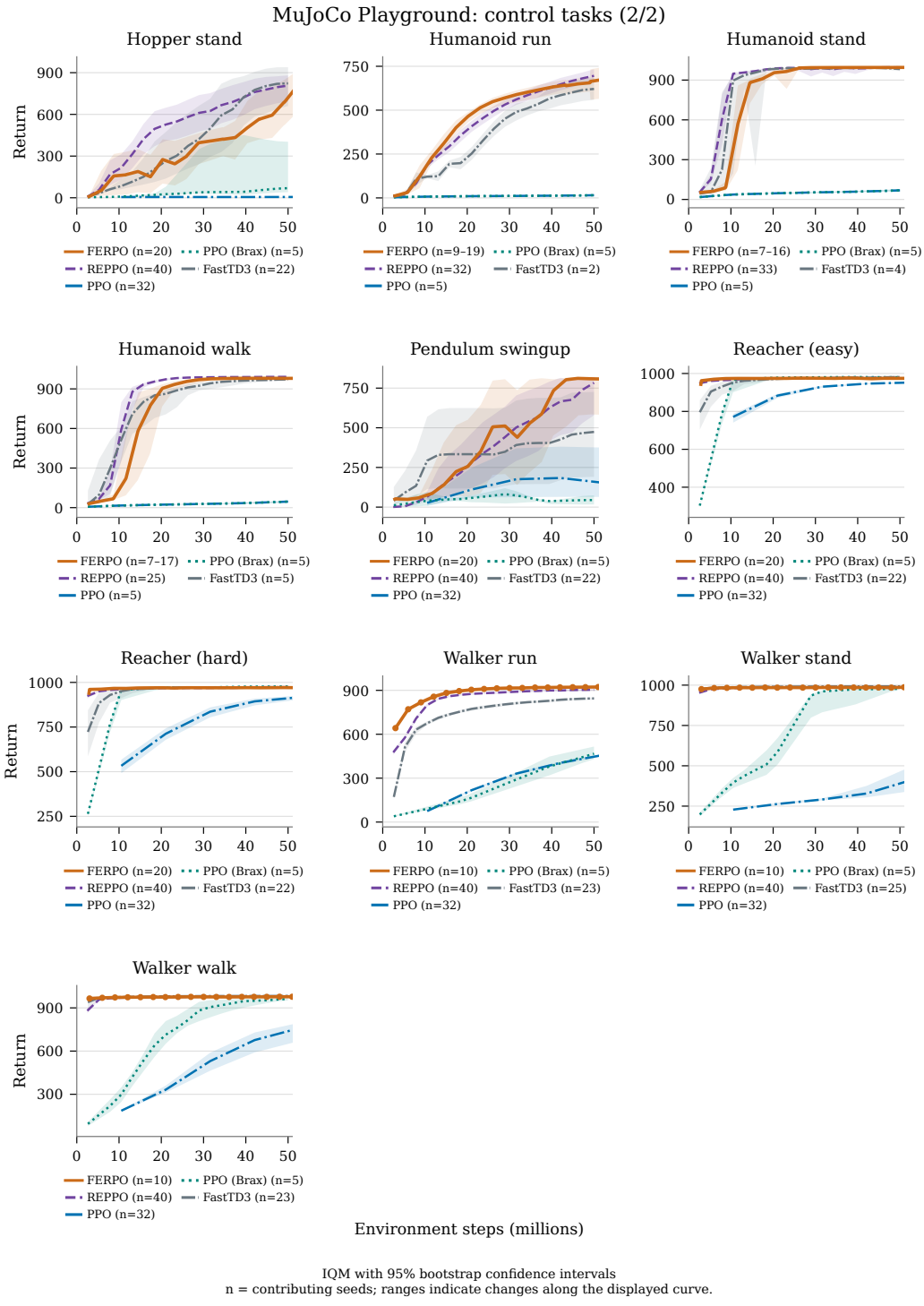


Figure 13: **DMC learning curves (2/2).** The remaining ten tasks, under the same protocol as Figure 12. All methods show IQM with pointwise 95% bootstrap confidence intervals. Walker markers denote the available checkpoint evaluations from 3.01M to 50.99M steps, with ten seeds per task. Legends report contributing seed counts.

A.3.10 G1/T1 LOCOMOTION AND MANISKILL MANIPULATION

At 50M interactions, FERPO's IQM returns are 35.56 and 30.38 on G1 flat and rough terrain, compared with REPPO's 37.60 and 32.89. On T1 flat and rough terrain, FERPO reaches 37.05 and 30.68, compared with REPPO's 36.48 and 30.73. Thus FERPO has a slightly higher point estimate on T1 flat and a similar endpoint on T1 rough, while REPPO retains a larger lead on both G1 tasks.

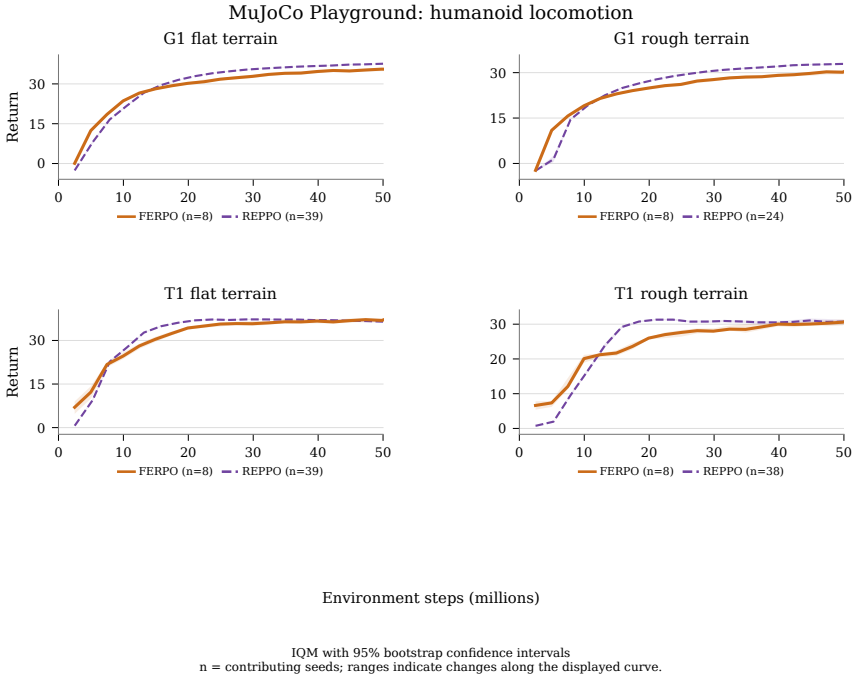


Figure 14: **All four G1/T1 locomotion tasks.** FERPO and official REPPO trials show IQM with pointwise 95% bootstrap confidence intervals. FERPO uses $K = 16$, normalized ESS target 0.65, and eight completed seeds per task; legends report each method's seed count. Results use the recorded evaluation protocols described in Appendix A.3.7.

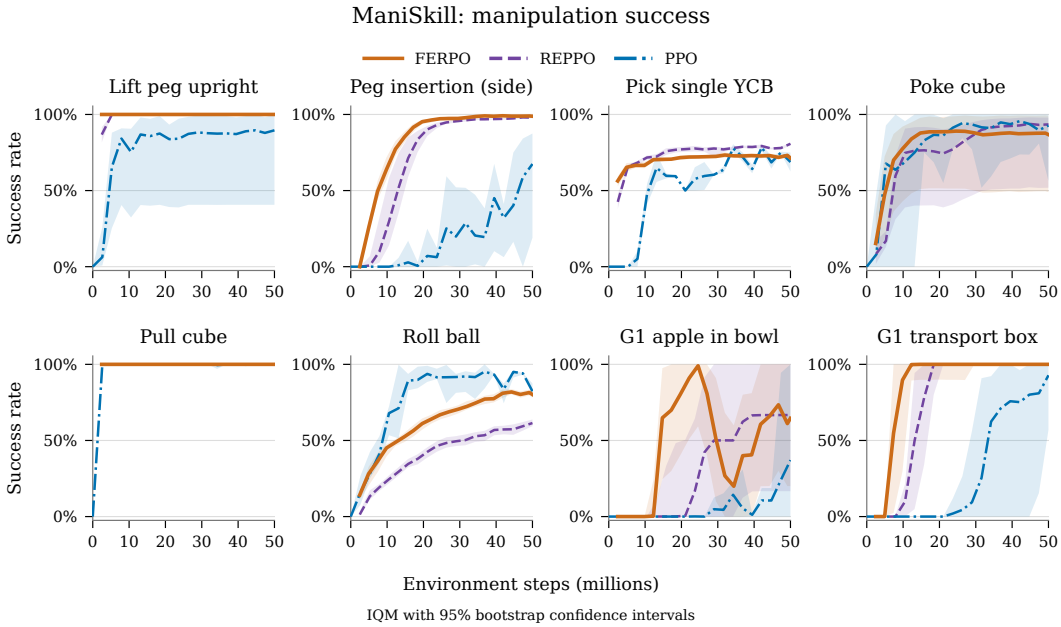


Figure 15: **All eight ManiSkill tasks.** All methods show success-rate IQM with pointwise 95% bootstrap confidence intervals. FERPO uses 20 seeds per task except G1 transport box (ten). PPO uses one configuration across all eight tasks; REPPO uses the official published trial CSVs.

A.4 FERPO PSEUDOCODE

Algorithm 1 shows the interleaved critic and actor updates used in the main learning setup. Critic targets are computed once per rollout; candidate actions and their Q values are refreshed for each actor update. We write $\langle\cdot\rangle_{\mathcal{B}}$ for a minibatch average and sg for stop-gradient. Bootstrap states use the final observation before any environment reset. The frozen parameters $\bar{\phi}$ are a snapshot for target construction.

Algorithm 1 FERPO with interleaved critic and actor updates

Require: Environment, rollout length T, discount γ, trace parameter λ_{TD}, auxiliary weight c_{aux}, candidate count $K \geq 2$, epochs E, minibatch size, entropy target $\mathcal{H}_{\mathrm{target}}$, ESS target $e_{\mathrm{target}} \in (1/K, 1)$, critic settings from Section 3.3.

Initialize policy π_θ, critic Q_ϕ, log multipliers η, ξ, and their optimizers.
while the environment interaction budget remains **do**
  Freeze $\pi_{\mathrm{roll}} \leftarrow \pi_\theta$ and $\bar{\phi} \leftarrow \phi$, $\tau_{\mathrm{roll}} \leftarrow e^{\eta}$.
  Collect rollout transitions $\mathcal{D}$ using $a_t \sim \pi_{\mathrm{roll}}(\cdot \mid s_t)$, including terminal and truncation flags.
  For each transition t, set $m_t \leftarrow 0$ at a true terminal and $m_t \leftarrow 1$ otherwise, including truncations.
  Set $c_t \leftarrow 1$ if $t < T-1$ and transition t is neither terminal nor truncated; otherwise set $c_t \leftarrow 0$.
  At each nonterminal successor, use $a_t^+ \leftarrow a_{t+1}$ if $c_t = 1$; otherwise draw $a_t^+ \sim \pi_{\mathrm{roll}}(\cdot \mid s_{t+1})$.
  Store $v_t^+ \leftarrow Q_{\bar{\phi}}(s_{t+1}, a_t^+)$ and $b_t^+ \leftarrow \log \pi_{\mathrm{roll}}(a_t^+ \mid s_{t+1})$ without gradients; set both to zero at true terminals.
  Set $G_T \leftarrow 0$ at the rollout end; $c_{T-1} = 0$ ensures bootstrapping uses v_{T-1}^+.
  for $t = T-1, \ldots, 0$ in each environment **do**
    $G_t \leftarrow r_t + \gamma m_t[(1 - \lambda_{\mathrm{TD}} c_t) v_t^+ + \lambda_{\mathrm{TD}} c_t G_{t+1} - \tau_{\mathrm{roll}} b_t^+]$.
  end for
  Store detached G_t, next-feature targets from the frozen encoder $\bar{\phi}$, and reward targets r_t in $\mathcal{D}$.
  for epoch $= 1, \ldots, E$ **do**
    for each shuffled minibatch $\mathcal{B} \subset \mathcal{D}$ **do**
      $\mathcal{L}_Q \leftarrow \big\langle \mathbf{1}_{\{\text{not truncated}\}}[\mathcal{L}_{\text{HL-Gauss}}(\phi; s_t, a_t, G_t) + c_{\mathrm{aux}} \mathcal{L}_{\mathrm{aux}}(\phi; t)] \big\rangle_{\mathcal{B}}$.
      Take an optimizer step to minimize $\mathcal{L}_Q$ with respect to ϕ, holding the critic targets fixed.
      $\tau \leftarrow e^{\eta}$, $\lambda \leftarrow e^{\xi}$.
      For each $s \in \mathcal{B}$, draw $a_i(s) \sim \pi_{\mathrm{roll}}(\cdot \mid s)$, $i = 1, \ldots, K$, without gradients.
      Evaluate $v_i(s) \leftarrow Q_\phi(s, a_i(s))$ and $b_i(s) \leftarrow \log \pi_{\mathrm{roll}}(a_i(s) \mid s)$ without gradients.
      $\ell_i(s) \leftarrow [v_i(s) - \tau b_i(s)]/(\tau + \lambda)$.
      $w_i(s) \leftarrow \mathrm{sg}\big[\mathrm{softmax}_i(\ell(s))\big]$ {normalize over the K actions at each state}
      $\widehat{\mathcal{L}}_{\mathrm{F}} \leftarrow -\big\langle \sum_{i=1}^K w_i(s) \log \pi_\theta(a_i(s) \mid s) \big\rangle_{\mathcal{B}}$.
      Draw fresh $\widetilde{a}_s \sim \pi_\theta(\cdot \mid s)$ and estimate $\widehat{\mathcal{H}} \leftarrow -\langle \log \pi_\theta(\widetilde{a}_s \mid s) \rangle_{\mathcal{B}}$.
      $\widehat{e}(s) \leftarrow \big(K \sum_{i=1}^K w_i(s)^2\big)^{-1}$.
      $\mathcal{L}_\tau \leftarrow \tau\ \mathrm{sg}[\widehat{\mathcal{H}} - \mathcal{H}_{\mathrm{target}}]$.
      $\mathcal{L}_\lambda \leftarrow \lambda\ \mathrm{sg}[\langle \log \widehat{e}(s) \rangle_{\mathcal{B}} - \log e_{\mathrm{target}}]$.
      Take an optimizer step to minimize $\widehat{\mathcal{L}}_{\mathrm{F}} + \mathcal{L}_\tau + \mathcal{L}_\lambda$ with respect to (θ, η, ξ), holding ϕ and π_{roll} fixed.
    end for
  end for
  Discard $\mathcal{D}$ before collecting the next rollout.
end while
return π_θ.

$\mathcal{L}_{\text{HL-Gauss}}$ matches the critic's categorical prediction to a Gaussian-smoothed histogram of the scalar target G_t; $Q_\phi(s, a)$ is the mean of this prediction. $\mathcal{L}_{\mathrm{aux}}$ predicts the next critic representation and reward, with fixed targets and zero loss at episode ends (Section 3.3). The target recursion adds the next action's entropy contribution once, outside the TD mixture. Actor updates differentiate only policy log probabilities and multiplier losses; their gradients do not change the critic or rollout policy.

Policy parameterization and action bounds. For bounded-action environments, we use a tanh-squashed Gaussian policy π_θ, following REPPO (Voelcker et al., 2026) and SAC (Haarnoja et al., 2018a): $z \sim \mathcal{N}(\mu_\theta(s), \mathrm{diag}(\sigma_\theta^2(s)))$, $a = \tanh(z)$, where a neural network parameterizes μ_θ and σ_θ. Actions are rescaled to the environment's action bounds where needed. Policy log densities include the corresponding change-of-variables correction.

For cached FERPO, complete critic training first, freeze Q_ϕ, and sample and store $a_i(s)$, $v_i(s)$, and $b_i(s)$ once for every rollout state. Reuse these tensors throughout actor training, recomputing $\ell_i(s)$ and $w_i(s)$ as τ and λ change; release the cache before the next rollout. Appendix A.2 reports the runtime and memory costs of this schedule.

A.5 State-dependent normalization in policy fitting

AWR's practical regression objective omits the per-state normalizer (Peng et al., 2019, Eqs. 8–10); AWAC explicitly discusses the same omission (Nair et al., 2020, Appendix A-B). To isolate its effect, fix a state distribution d, the rollout policy $\pi_{\mathrm{roll}}(\cdot \mid s)$, advantage estimates $A(s,a)$, and $\lambda > 0$. We consider the exponential weights before clipping and assume the expectations and gradients below exist. Define

$$\begin{aligned} u(s,a) &:= \exp(A(s,a)/\lambda), \qquad Z(s) := \mathbb{E}_{a\sim\pi_{\mathrm{roll}}(\cdot|s)}[u(s,a)], \\ q_A(a \mid s) &:= \frac{\pi_{\mathrm{roll}}(a \mid s)u(s,a)}{Z(s)}, \qquad 0 < Z(s) < \infty. \end{aligned} \tag{13}$$

Let $\ell_\theta(s) := -\mathbb{E}_{a\sim q_A(\cdot|s)}[\log \pi_\theta(a \mid s)]$. The normalized forward-KL fit minimizes $L_{\mathrm{norm}}(\theta) := \mathbb{E}_{s\sim d}[\ell_\theta(s)]$, up to a constant independent of θ. Omitting $Z(s)$ instead gives

$$\begin{aligned} L_{\mathrm{omit}}(\theta) &:= -\mathbb{E}_{s\sim d}\mathbb{E}_{a\sim\pi_{\mathrm{roll}}(\cdot|s)}[u(s,a)\log \pi_\theta(a \mid s)] \\ &= \mathbb{E}_{s\sim d}[Z(s)\ell_\theta(s)]. \end{aligned} \tag{14}$$

Thus, holding the target weights fixed during actor fitting and writing $g_\theta(s) := \nabla_\theta \ell_\theta(s)$,

$$\nabla_\theta L_{\mathrm{norm}} = \mathbb{E}_{s\sim d}[g_\theta(s)], \qquad \nabla_\theta L_{\mathrm{omit}} = \mathbb{E}_{s\sim d}[Z(s)g_\theta(s)]. \tag{15}$$

The omission therefore changes the relative influence of states even with unlimited data. A constant $Z(s)$ only rescales the gradient. With independent, unrestricted fitting at each state, both objectives recover q_A. With a shared neural policy, varying $Z(s)$ can change the overall gradient direction and, when the target cannot be fitted exactly, the optimal approximation. The size of this effect depends on the variation in $Z(s)$ and the alignment of gradients across states.

An advantage baseline does not generally remove this reweighting: for $A(s,a) = Q(s,a) - V(s)$, $Z(s) = e^{-V(s)/\lambda}\mathbb{E}_{a\sim\pi_{\mathrm{roll}}(\cdot|s)}[e^{Q(s,a)/\lambda}]$, which need not equal one. The baseline cancels from q_A after normalization, but affects state weights when $Z(s)$ is omitted. AWR's additional weight clipping changes the action weights as well (Peng et al., 2019, Section 3.3).

FERPO estimates normalized weights separately within each state using Eq. 6. SNIS has finite-sample bias but is consistent under the usual integrability and sampling assumptions (Owen, 2013, Section 9.2); see also Appendix B.5.2. This establishes a difference in the fitted objective, rather than a general performance ordering.

B Proofs

B.1 Derivation of the KL surrogate

State distribution and expectation in Eq. 1. For a fixed policy π, let ρ_t^π denote the state density at time t in trajectories generated by $s_0 \sim \rho_0$, $a_t \sim \pi(\cdot \mid s_t)$, and $s_{t+1} \sim P(\cdot \mid s_t, a_t)$. Thus $\rho_0^\pi = \rho_0$. As in Section 2, define the normalized discounted state-visitation distribution

$$d^\pi(s) := (1-\gamma)\sum_{t=0}^{\infty}\gamma^t\rho_t^\pi(s), \qquad d_{\mathrm{roll}} := d^{\pi_{\mathrm{roll}}}.$$

Whereas ρ_t^π describes the states at a single time step, d^π averages these distributions over time with normalized discount weights. For any function $f(s,a)$ with finite absolute discounted expectation, the definition of d^π gives

$$\frac{1}{1-\gamma}\mathbb{E}_{\substack{s\sim d^\pi \\ a\sim\pi(\cdot|s)}}[f(s,a)] = \sum_{t=0}^{\infty}\gamma^t\mathbb{E}_{\substack{s\sim\rho_t^\pi \\ a\sim\pi(\cdot|s)}}[f(s,a)] = \mathbb{E}_{\rho_0,P,\pi}\left[\sum_{t=0}^{\infty}\gamma^t f(s_t,a_t)\right].$$

Taking $f(s,a) = r(s,a) - \tau \log \pi(a \mid s)$ shows that Eq. 1 is exactly the discounted maximum-entropy return.

To describe one step of state evolution under the policy, define the transition kernel

$$P_\pi(s' \mid s) := \int \pi(a \mid s) P(s' \mid s, a)\, \mathrm{d}a.$$

The state distribution evolves by averaging the next-state distribution over the current states:

$$\rho^\pi_{t+1}(s') = \int \rho^\pi_t(s) P_\pi(s' \mid s)\, \mathrm{d}s =: (\rho^\pi_t P_\pi)(s').$$

Thus $\rho^\pi_{t+1} = \rho^\pi_t P_\pi$ is shorthand for this integral; it describes one step under the environment dynamics. Using this recursion and shifting the index in the discounted sum yields $d^\pi = (1-\gamma)\rho_0 + \gamma d^\pi P_\pi$.

Gradient of the KL surrogate. Applying the policy-gradient theorem (Sutton et al., 1999) to the entropy-regularized objective gives

$$\nabla_\theta J_\tau(\pi_\theta) = \frac{1}{1-\gamma} \mathbb{E}_{\substack{s\sim d^{\pi_\theta} \\ a\sim\pi_\theta(\cdot|s)}} \left[\nabla_\theta \log \pi_\theta(a \mid s) \big(Q^{\pi_\theta}_\tau(s,a) - \tau \log \pi_\theta(a \mid s) \big) \right]. \tag{16}$$

The additional derivative of the entropy term has zero expectation by $\mathbb{E}_{a\sim\pi_\theta}[\nabla_\theta \log \pi_\theta(a \mid s)] = 0$.

For the frozen rollout critic, define the normalized Boltzmann target

$$p_{Q,\tau}(a \mid s) := \frac{\exp(Q_{\mathrm{roll}}(s,a)/\tau)}{Z_Q(s)}, \qquad Z_Q(s) := \int_{\mathcal{A}} \exp(Q_{\mathrm{roll}}(s,a)/\tau)\, \mathrm{d}a. \tag{17}$$

Expanding the reverse KL gives the per-state identity

$$\tau D_{\mathrm{KL}}(\pi_\theta \| p_{Q,\tau}) = -\mathbb{E}_{a\sim\pi_\theta}[Q_{\mathrm{roll}}(s,a)] - \tau \mathcal{H}(\pi_\theta(\cdot \mid s)) + \tau \log Z_Q(s). \tag{18}$$

Differentiating this identity while holding the rollout critic and state distribution fixed and comparing with Eq. 16 yields the gradient-matching condition stated in Section 2. Thus, minimizing Eq. 2 matches the local return-ascent direction at the rollout parameters. The equivalence is local because the frozen critic and state distribution do not track subsequent policy changes.

B.2 Derivation of the optimal regularized target

Fix a state s and write $Q(a) := Q_{\mathrm{roll}}(s,a)$ and $\mu(a) := \pi_{\mathrm{roll}}(a \mid s)$. Assume $\mu > 0$ almost everywhere on $\mathcal{A}$ and that the integrals below are finite. For $\tau > 0$ and $\lambda \geq 0$, the per-state contribution to Eq. 4 is

$$\begin{aligned} \mathcal{L}_s(q;\tau,\lambda) &:= -\mathbb{E}_{a\sim q}[Q(a)] + \tau(\mathcal{H}_{\mathrm{target}} - \mathcal{H}(q)) + \lambda(D_{\mathrm{KL}}(q\|\mu) - \delta_{\mathrm{KL}}) \\ &= \int_{\mathcal{A}} q(a) \big[(\tau+\lambda)\log q(a) - Q(a) - \lambda \log \mu(a) \big]\, \mathrm{d}a + \tau \mathcal{H}_{\mathrm{target}} - \lambda \delta_{\mathrm{KL}}. \end{aligned} \tag{19}$$

Introduce a multiplier ν for the normalization constraint: $\widetilde{\mathcal{L}}_s(q,\nu;\tau,\lambda) := \mathcal{L}_s(q;\tau,\lambda) + \nu(\int_{\mathcal{A}} q(a)\,\mathrm{d}a - 1)$. The stationarity condition is

$$\frac{\delta \widetilde{\mathcal{L}}_s}{\delta q(a)} = (\tau+\lambda)(\log q(a) + 1) - Q(a) - \lambda \log \mu(a) + \nu = 0. \tag{20}$$

Solving for q and normalizing gives

$$\begin{aligned} q^*(a) &= \frac{1}{Z_*(s)} \exp\left(\frac{Q(a)}{\tau+\lambda} \right) \mu(a)^{\lambda/(\tau+\lambda)}, \\ Z_*(s) &:= \int_{\mathcal{A}} \exp\left(\frac{Q(a)}{\tau+\lambda} \right) \mu(a)^{\lambda/(\tau+\lambda)}\, \mathrm{d}a, \end{aligned} \tag{21}$$

which is Eq. 5. Substituting $\log q^*$ into Eq. 19 yields the exact identity

$$\mathcal{L}_s(q;\tau,\lambda) = (\tau+\lambda) D_{\mathrm{KL}}(q\|q^*) - (\tau+\lambda)\log Z_*(s) + \tau \mathcal{H}_{\mathrm{target}} - \lambda \delta_{\mathrm{KL}}. \tag{22}$$

Since KL is nonnegative and vanishes only when its arguments agree almost everywhere, q^* is the unique global minimizer. Averaging this identity over d_{roll} also proves the reverse-KL projection form of Eq. 4 used in Section 3.1.

For the forward-KL fit, the target-to-proposal density ratio is

$$\frac{q^*(a)}{\mu(a)} = \frac{1}{Z_*(s)} \exp\left(\frac{Q(a) - \tau \log \mu(a)}{\tau + \lambda}\right). \tag{23}$$

Self-normalizing these ratios across sampled actions cancels $Z_*(s)$ and gives Eq. 6, with the exact critic replaced by Q_ϕ in practice.

B.3 Log ESS and divergence control

Target-divergence bound in Proposition 1 and ESS identities. Fix a state s, and write $q(a) = q^*_\lambda(a \mid s)$ for the target density and $\mu(a) = \pi_{\text{roll}}(a \mid s)$ for the rollout-policy density. Define the importance ratio $r(a) := q(a)/\mu(a)$ wherever $\mu(a) > 0$. As assumed in the proposition, $q \ll \mu$ and $\mathbb{E}_\mu[r^2] < \infty$. The population normalized ESS is $e_\infty := 1/\mathbb{E}_\mu[r^2]$. Because q and μ are probability densities and $q \ll \mu$, $\mathbb{E}_\mu[r] = \int q(a)\,\mathrm{d}a = 1$. Therefore,

$$\chi^2(q\|\mu) = \mathbb{E}_\mu[(r-1)^2] = \mathbb{E}_\mu[r^2] - 1, \qquad e_\infty = \frac{1}{1 + \chi^2(q\|\mu)}.$$

Jensen's inequality for the concave logarithm gives

$$D_{\text{KL}}(q\|\mu) = \mathbb{E}_q[\log r] \le \log \mathbb{E}_q[r] = \log \mathbb{E}_\mu[r^2] = D_2(q\|\mu) = -\log e_\infty.$$

Since $-\log x$ is decreasing, $e_\infty \ge e_{\text{target}}$ implies $D_{\text{KL}}(q\|\mu) \le -\log e_{\text{target}}$. For independent samples $a_1, \dots, a_K \sim \mu$, define the normalized weights $w_i = r(a_i)/\sum_j r(a_j)$ and empirical normalized ESS $\widehat{e} := 1/(K\sum_i w_i^2)$. Substituting the weights gives

$$\widehat{e} = \frac{\left(K^{-1}\sum_{i=1}^K r(a_i)\right)^2}{K^{-1}\sum_{i=1}^K r(a_i)^2} \xrightarrow[K\to\infty]{\text{a.s.}} \frac{(\mathbb{E}_\mu[r])^2}{\mathbb{E}_\mu[r^2]} = e_\infty.$$

The convergence follows from the strong law of large numbers and the finite second-moment assumption. If an initial batch has zero total weight, the normalized estimate is undefined; the sample average of r converges to one, so the denominator for weight normalization is positive for all sufficiently large K almost surely. □

Finite-batch interpretation. There is an exact identity for any finite batch of normalized weights $w = (w_1, \dots, w_K)$. With $u_K(i) = 1/K$ on sample indices,

$$\chi^2(w\|u_K) = K\sum_i w_i^2 - 1, \qquad -\log \widehat{e} = \log\big(1 + \chi^2(w\|u_K)\big) = D_2(w\|u_K).$$

This finite-batch equality concerns probability vectors on sampled indices. It does not identify $-\log \widehat{e}$ with the divergence between the full action distributions q and μ.

B.3.1 From target ESS to policy total variation

A decrease in the population forward-KL fitting loss suffices to transfer the target's divergence control to the actor in total variation. We use the half-L^1 convention in Section 2 and natural logarithms throughout.

We restate the actor-control assumption of Proposition 1. Fix a state and a target q^*, suppressing state conditioning. Assume $q^* \ll \pi_{\text{roll}}$ and $\mathbb{E}_{\pi_{\text{roll}}}[(q^*/\pi_{\text{roll}})^2] < \infty$, and let $e_\infty := 1/\mathbb{E}_{\pi_{\text{roll}}}[(q^*/\pi_{\text{roll}})^2]$. Suppose the actor update does not increase the population forward-KL fitting objective:

$$D_{\text{KL}}(q^*\|\pi_{\text{curr}}) \le D_{\text{KL}}(q^*\|\pi_{\text{roll}}). \tag{24}$$

Then Eq. 9 holds:

$$\begin{aligned}\frac{1}{2} D_{\text{TV}}(\pi_{\text{curr}}, \pi_{\text{roll}})^2 &\le D_{\text{KL}}(q^*\|\pi_{\text{roll}}) \\ &\le D_2(q^*\|\pi_{\text{roll}}) = -\log e_\infty.\end{aligned}$$

No exact minimization or Gaussian policy assumption is required.

Proof of the actor-TV bound in Proposition 1. The triangle inequality, obtained by adding and subtracting q^* inside the absolute value in the TV definition, gives

$$D_{\mathrm{TV}}(\pi_{\mathrm{curr}}, \pi_{\mathrm{roll}}) \le D_{\mathrm{TV}}(\pi_{\mathrm{curr}}, q^*) + D_{\mathrm{TV}}(q^*, \pi_{\mathrm{roll}}).$$

TV is symmetric, so Pinsker's inequality (van Erven & Harremoës, 2014, Theorem 31) can bound both terms using KL divergences with q^* as their first argument. Applying Eq. 24 then yields

$$\begin{aligned} D_{\mathrm{TV}}(\pi_{\mathrm{curr}}, \pi_{\mathrm{roll}}) &\le \sqrt{\tfrac{1}{2} D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{curr}})} + \sqrt{\tfrac{1}{2} D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{roll}})} \\ &\le 2\sqrt{\tfrac{1}{2} D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{roll}})} = \sqrt{2 D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{roll}})}. \end{aligned}$$

Squaring both sides and dividing by two gives $\frac{1}{2} D_{\mathrm{TV}}(\pi_{\mathrm{curr}}, \pi_{\mathrm{roll}})^2 \le D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{roll}})$. The target-divergence result proved above supplies the remaining inequality and identity, $D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{roll}}) \le D_2(q^* \| \pi_{\mathrm{roll}}) = -\log e_\infty$. □

Meaning of fitting improvement. With q^* fixed, the difference of the fitting objectives is

$$D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{curr}}) - D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{roll}}) = -\mathbb{E}_{q^*}\left[\log \frac{\pi_{\mathrm{curr}}}{\pi_{\mathrm{roll}}}\right].$$

Thus Eq. 24 requires only that the actor increase or preserve the target-weighted population log likelihood. This is the objective approximated by the SNIS actor loss; it is distinct from an assumption of higher environment return.

Averaging over rollout states. The result also holds on average without per-state fitting improvement. All distributions below are conditioned on s. Squaring the first Pinsker bound above and using $(\sqrt{x} + \sqrt{y})^2 \le 2(x + y)$ gives the pointwise inequality

$$D_{\mathrm{TV}}(\pi_{\mathrm{curr}}, \pi_{\mathrm{roll}})^2 \le D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{curr}}) + D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{roll}}).$$

Assume the relevant expectations are finite and that

$$\mathbb{E}_{s \sim d_{\mathrm{roll}}}[D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{curr}})] \le \mathbb{E}_{s \sim d_{\mathrm{roll}}}[D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{roll}})].$$

Averaging the pointwise inequality and applying the target-divergence bound in Eq. 9 yields

$$\begin{aligned} \frac{1}{2}\mathbb{E}_{s \sim d_{\mathrm{roll}}}[D_{\mathrm{TV}}(\pi_{\mathrm{curr}}, \pi_{\mathrm{roll}})^2] &\le \mathbb{E}_{s \sim d_{\mathrm{roll}}}[D_{\mathrm{KL}}(q^* \| \pi_{\mathrm{roll}})] \\ &\le -\mathbb{E}_{s \sim d_{\mathrm{roll}}}[\log e_\infty(s)]. \end{aligned}$$

If the geometric mean of population ESS meets the controller target, $\mathbb{E}_{s \sim d_{\mathrm{roll}}}[\log e_\infty(s)] \ge \log e_{\mathrm{target}}$, the right-hand side is at most $-\log e_{\mathrm{target}}$. This controls average squared TV, rather than the largest deviation across states. Decreases in finite-sample SNIS loss and empirical ESS control do not by themselves guarantee these population conditions. Nor does the TV bound imply an upper bound on the fitted actor's KL; it cannot replace the KL penalty in Eq. 3 without further assumptions.

B.4 Convergence under exact fitting

The regularized target in Eq. 5 interpolates between the reference policy and the Boltzmann distribution. Repeated exact updates reach the Boltzmann distribution when the reference is replaced by the fitted policy after every update. The following result quantifies how λ controls this convergence for a fixed critic.

Proposition 2 (Convergence of exact FERPO updates for a fixed critic)**.** *Fix a state s, a critic $Q(s, \cdot)$, and a temperature $\tau > 0$. Suppress state conditioning and let $p(a) := \exp(Q(s, a)/\tau)/Z(s)$ be normalizable. Assume $D_{\mathrm{KL}}(p \| \pi_0) < \infty$ and fix $0 < \lambda < \infty$. At each iteration, compute the unrestricted target and fit it exactly:*

$$q_n^* := \arg\min_q \left\{ \tau D_{\mathrm{KL}}(q \| p) + \lambda D_{\mathrm{KL}}(q \| \pi_n) \right\}, \qquad \pi_{n+1} = q_n^*.$$

The policy after $n \geq 1$ exact updates is

$$\pi_n(a) = \frac{1}{Z_n(s)} \pi_0(a)^{\left(\frac{\lambda}{\tau+\lambda}\right)^n} \exp\left[\frac{1 - \left(\frac{\lambda}{\tau+\lambda}\right)^n}{\tau} Q(s,a)\right], \tag{25}$$

where $Z_n(s)$ normalizes the density. Since $(\lambda/(\tau+\lambda))^n \to 0$ as $n \to \infty$, the initial policy's exponent vanishes and the coefficient of $Q(s,a)$ approaches $1/\tau$. Consequently,

$$\lim_{n\to\infty} \pi_n(a) = \frac{\exp(Q(s,a)/\tau)}{\int_{\mathcal{A}} \exp(Q(s,b)/\tau)\,\mathrm{d}b} = p(a) \quad \textit{almost everywhere},$$

and convergence also holds in total variation. The limit is the unique maximizer of $\mathbb{E}_{a\sim q}[Q(s,a) - \tau \log q(a)]$, independently of the fixed finite value of λ. For $\lambda = 0$, the target is reached in one exact update.

The role of λ. Equation 25 separates the influence of the initial policy from the Boltzmann target. Larger λ makes $\lambda/(\tau+\lambda)$ closer to one, so the initial policy's influence decays more slowly. Smaller λ gives faster progress toward the same target. Thus, λ controls the rate of approach, while τ determines the limiting Boltzmann distribution.

The proposition assumes exact forward-KL fits and a refreshed reference policy at every iteration. Repeated fitting to a single frozen target only recovers that target. It establishes convergence to the Boltzmann distribution of the fixed critic; convergence of full RL training, where the critic is re-estimated, requires a separate policy-iteration argument.

Proof of Proposition 2. Write $\beta := \lambda/(\tau+\lambda) \in (0,1)$. Equation 5 gives $\pi_{n+1} = p^{1-\beta}\pi_n^\beta / C_n$, where $C_n := \int p^{1-\beta}\pi_n^\beta \,\mathrm{d}a$. Induction gives $\pi_n \propto p^{1-\beta^n}\pi_0^{\beta^n}$. Substituting $p \propto \exp(Q/\tau)$ and $\beta = \lambda/(\tau+\lambda)$ yields Eq. 25. The finite-KL assumption ensures that π_0 covers the support of p; this property is preserved by the update. As $n \to \infty$, $p^{1-\beta^n}\pi_0^{\beta^n} \to p$ almost everywhere. The bound $p^{1-t}\pi_0^t \leq p + \pi_0$ for $0 < t < 1$ permits dominated convergence, so the normalizing integral of this geometric mixture tends to one. This proves the stated almost-everywhere limit.

To quantify convergence, Hölder's inequality and normalization imply $0 < C_n \leq 1$. Therefore,

$$D_{\mathrm{KL}}(p\|\pi_{n+1}) = \beta D_{\mathrm{KL}}(p\|\pi_n) + \log C_n \leq \beta D_{\mathrm{KL}}(p\|\pi_n). \tag{26}$$

Iterating gives the KL contraction bound

$$D_{\mathrm{KL}}(p\|\pi_{n+1}) \leq \beta D_{\mathrm{KL}}(p\|\pi_n), \qquad D_{\mathrm{KL}}(p\|\pi_n) \leq \beta^n D_{\mathrm{KL}}(p\|\pi_0). \tag{27}$$

Pinsker's inequality then implies convergence in total variation. Optimality of p follows from Eq. 18. For $\lambda = 0$, the update minimizes $D_{\mathrm{KL}}(q\|p)$ and reaches p immediately. □

Finite-step accuracy and varying multipliers. If $D_0 := D_{\mathrm{KL}}(p\|\pi_0) > \varepsilon > 0$, then $n \geq \lceil \log(D_0/\varepsilon)/\log(1+\tau/\lambda) \rceil$ exact updates suffice for $D_{\mathrm{KL}}(p\|\pi_n) \leq \varepsilon$. For varying coefficients $\lambda_k > 0$ at fixed Q and τ, β^n is replaced by $\prod_{k=0}^{n-1} \lambda_k/(\tau+\lambda_k)$; convergence follows if this product tends to zero, as it does for uniformly bounded λ_k.

B.5 Update directions and importance sampling

For a Gaussian policy with fixed covariance, both REPPO and FERPO have mean gradients that can be expressed as averages of cost gradients. REPPO averages under the current policy; FERPO averages under its cost-reweighted target, with a factor $1/\lambda$. We derive these identities and explain how the different distributions produce the opposing directions in Figure 1. We then show why recovering the target direction from samples requires adequate proposal coverage.

B.5.1 Cost gradients under the policy and the target

Fix a state and a cost $c(a) = -Q_\phi(s,a)$ on $\mathbb{R}^d$. Let $\pi_m = \mathcal{N}(m,\Sigma)$ be the reference and initial actor, with fixed covariance $\Sigma \succ 0$. The candidate actor is $\pi_n = \mathcal{N}(n,\Sigma)$. Fix $\tau, \lambda > 0$ and write $T = \tau + \lambda$. FERPO constructs

$$q_m^*(a) = \frac{1}{Z_m} e^{-c(a)/T} \pi_m(a)^{\lambda/T}, \tag{28}$$

and holds q_m^* fixed while fitting π_n. Let $m^*(m) := \mathbb{E}_{a\sim q_m^*}[a]$ denote the target mean. Assume c is continuously differentiable and bounded below, and that c and ∇c grow at most polynomially. These conditions ensure finite target moments and justify the differentiation below.

Proposition 3 (Initial mean-descent directions)**.** *At $n = m$, REPPO's KL regularization is inactive. Its mean loss is $L_{\mathrm{R}}(n) = \mathbb{E}_{\pi_n}[c] - \tau H(\pi_n)$, while FERPO fits $L_{\mathrm{F}}(n; m) = D_{\mathrm{KL}}(q_m^* \| \pi_n)$. Their negative gradients are*

$$\begin{aligned} d_{\mathrm{R}}(m) &:= -\nabla_n L_{\mathrm{R}}(n)|_{n=m} = -\mathbb{E}_{a\sim\pi_m}[\nabla c(a)], \\ d_{\mathrm{F}}(m) &:= -\nabla_n L_{\mathrm{F}}(n; m)|_{n=m} = \Sigma^{-1}\big(m^*(m) - m\big) = -\frac{1}{\lambda}\mathbb{E}_{a\sim q_m^*}[\nabla c(a)]. \end{aligned} \tag{29}$$

The exact forward-KL mean fit sets $n = m^*(m)$. Reverse KL to the same frozen target has the same initial descent direction *as REPPO, with its negative gradient equal to $d_{\mathrm{R}}(m)/T$.*

Proof. Using the reparameterization $a = n + \Sigma^{1/2} z$, where $z \sim \mathcal{N}(0, I)$, we obtain

$$\nabla_n \mathbb{E}_{a\sim\pi_n}[c(a)] = \mathbb{E}_{a\sim\pi_n}[\nabla_a c(a)].$$

Since the covariance is fixed, the entropy does not depend on n. This gives REPPO's mean-descent direction. For FERPO, expansion of the Gaussian log density gives

$$L_{\mathrm{F}}(n; m) = \frac{1}{2}\big(n - m^*(m)\big)^\top \Sigma^{-1}\big(n - m^*(m)\big) + \mathrm{const}(m). \tag{30}$$

Differentiating with respect to n gives the FERPO gradient, and the quadratic is minimized at $n = m^*(m)$.

To express FERPO's gradient in terms of cost gradients, differentiate the target log density with respect to the action:

$$\nabla_a \log q_m^*(a) = -\frac{1}{T}\big(\nabla c(a) + \lambda\Sigma^{-1}(a - m)\big). \tag{31}$$

The Gaussian factor and the lower bound on c make the target density vanish at infinity; the polynomial growth assumptions ensure integrability. Integrating its derivative over $\mathbb{R}^d$ therefore gives $\mathbb{E}_{q_m^*}[\nabla_a \log q_m^*] = \int \nabla_a q_m^*(a)\,\mathrm{d}a = 0$. Taking expectations in Eq. 31 yields

$$\mathbb{E}_{q_m^*}[\nabla c(a)] = \lambda\Sigma^{-1}\big(m - m^*(m)\big), \tag{32}$$

which proves the final equality for d_{F}.

Connection to REPPO. We now consider fitting the same frozen target using reverse KL, $D_{\mathrm{KL}}(\pi_n \| q_m^*)$. FERPO instead minimizes the forward KL, $D_{\mathrm{KL}}(q_m^* \| \pi_n)$. The following decomposition shows that reverse-KL fitting gives the same initial descent direction as REPPO, up to a positive scaling factor. Substituting the target density in Eq. 28 yields

$$T D_{\mathrm{KL}}(\pi_n \| q_m^*) = \mathbb{E}_{\pi_n}[c] - \tau H(\pi_n) + \lambda D_{\mathrm{KL}}(\pi_n \| \pi_m) + T \log Z_m. \tag{33}$$

At $n = m$, the KL penalty contributes no gradient, and Z_m is independent of the candidate mean n. Differentiating the right-hand side therefore gives REPPO's loss gradient. Consequently, minimizing $D_{\mathrm{KL}}(\pi_n \| q_m^*)$ gives the initial descent direction $d_{\mathrm{R}}(m)/T$: the same direction as REPPO, scaled by $1/T$. □

Shared small-variance limit. Assume additionally that c is three-times continuously differentiable and its derivatives through order three grow at most polynomially. Fix the mean m and positive constants τ and λ. Let $\Sigma = \sigma^2 I$ with $\sigma \to 0$. The Taylor expansion underlying the delta method (Khan & Rue, 2023, Appendix C) applies to both averages. Under π_m, $\mathbb{E}[a - m] = 0$ and $\mathbb{E}\|a - m\|^2 = O(\sigma^2)$. The target is proportional to $e^{-c(a)/T}$ times the density of $\mathcal{N}(m, (T/\lambda)\sigma^2 I)$. The lower bound on c and the growth assumptions imply $\mathbb{E}_{q_m^*}\|a - m\|^2 = O(\sigma^2)$ and $\mathbb{E}_{q_m^*}[\nabla c(a)] = O(1)$. Equation 32 then gives $\mathbb{E}_{q_m^*}[a - m] = O(\sigma^2)$. Expanding $\nabla c(a)$ around m therefore yields, for the initial loss gradients $g_{\mathrm{R}} := -d_{\mathrm{R}}(m)$ and $g_{\mathrm{F}} := -d_{\mathrm{F}}(m)$,

$$g_{\mathrm{R}} = \nabla c(m) + O(\sigma^2), \qquad \lambda g_{\mathrm{F}} = \nabla c(m) + O(\sigma^2).$$

Thus both methods approach the same descent direction wherever $\nabla c(m) \neq 0$; in one dimension, their signs agree for sufficiently small σ. This shared limit does not imply agreement at finite width, where the target can be broad or multimodal, or uniformly near stationary points. Figure 1 uses the exact expectations at finite width.

At B in Figure 1, $m = -0.6$ and $\sigma = 0.4$. The expected cost derivative is 0.284, while $m^*(m) \simeq -0.466$, giving

$$d_{\mathrm{R}} = -0.284, \qquad d_{\mathrm{F}} \simeq \frac{-0.466 + 0.6}{0.4^2} = 0.838.$$

Thus FERPO initially moves toward the better right well while REPPO moves toward the worse left well. This is the potential benefit illustrated by the experiment: fitting the reweighted mean can give an update toward a better region even when the expected-cost gradient points elsewhere. The comparison concerns directions, not matched step sizes or eventual optimization success. It applies at the reference policy with fixed covariance.

B.5.2 Recovering the target-mean direction from samples

The update-direction comparison in Appendix B.5.1 uses the exact target mean. With SNIS, FERPO instead uses $\widehat{m}^* = \sum_i w_i a_i$ and the estimated negative gradient is $\widehat{d_{\mathrm{F}}} = \Sigma^{-1}(\widehat{m}^* - m)$. The integration-by-parts identity holds for the population target; it need not hold for a finite weighted sample. In particular, SNIS is generally biased at finite K. For a proposal r that is positive wherever q_m^* is positive, independent samples $a_i \sim r$ receive weights

$$u_i = \frac{e^{-c(a_i)/T} \pi_m(a_i)^{\lambda/T}}{r(a_i)}, \qquad w_i = \frac{u_i}{\sum_j u_j}. \tag{34}$$

Changing r changes the estimate's accuracy while preserving the same target, provided the denominator uses the actual proposal.

Proposition 4 (Accuracy of the sampled target mean)**.** *For fixed q_m^* and r, $\widehat{m}^* \to m^*(m)$ almost surely as $K \to \infty$. If the matrix*

$$\Omega_r := \int \frac{q_m^*(a)^2}{r(a)} (a - m^*(m))(a - m^*(m))^\top \, \mathrm{d}a \tag{35}$$

has finite trace, then $\sqrt{K}\big(\widehat{m}^* - m^*(m)\big) \Longrightarrow \mathcal{N}(0, \Omega_r)$. Here, $\Longrightarrow$ denotes convergence in distribution. For large K, the estimated target mean is approximately Gaussian around the exact mean, with covariance Ω_r/K. Thus, the typical estimation error decreases as $1/\sqrt{K}$, while its magnitude depends on how well the proposal covers the target distribution.

Proof. This is the standard SNIS argument (Owen, 2013, Section 9.2). Let $W = q_m^*/r$. Then $\mathbb{E}_r W = 1$, $\mathbb{E}_r[Wa] = m^*(m)$, and $\widehat{m}^* = (\sum_i W(a_i) a_i)/(\sum_i W(a_i))$. The strong law of large numbers proves consistency, since the target has a finite first moment. For the limiting distribution, write

$$\sqrt{K}(\widehat{m}^* - m^*(m)) = \frac{K^{-1/2} \sum_i W(a_i)(a_i - m^*(m))}{K^{-1} \sum_i W(a_i)}.$$

The numerator summands have mean zero and covariance Ω_r. The multivariate central limit theorem and convergence of the denominator to one give the result. □

Why a wider proposal can help. The covariance in Eq. 35 is large when the target places mass far from its mean in a region that the proposal rarely samples. Such a region contributes to the exact mean but is often absent from a finite batch. Increasing proposal density there can reduce this source of error. This explains the improvement from wider proposals in Figure 6; increasing width is useful only insofar as it improves coverage of influential target regions.

Although SNIS is consistent, finite-sample error can place the estimated target mean on the wrong side of the current policy mean, reversing the update direction.

B.6 From surrogate improvement to return improvement

Statement to be proved. We restate Eq. 3 from the main text. For a fixed temperature τ, an exact rollout critic, and bounded soft advantage, define the actual return change, the surrogate loss change, and the predicted return gain as

$$\begin{aligned}\Delta J_\tau &:= J_\tau(\pi_\theta) - J_\tau(\pi_{\mathrm{roll}}),\\ \Delta\mathcal{L}_{\mathrm{ME}} &:= \mathcal{L}_{\mathrm{ME}}(\theta;\theta_{\mathrm{roll}}) - \mathcal{L}_{\mathrm{ME}}(\theta_{\mathrm{roll}};\theta_{\mathrm{roll}}),\\ \Delta J_{\mathrm{surrogate}} &:= -\Delta\mathcal{L}_{\mathrm{ME}}.\end{aligned}$$

Both ΔJ_τ and $\Delta\mathcal{L}_{\mathrm{ME}}$ are defined as candidate minus rollout. Since J_τ is maximized but $\mathcal{L}_{\mathrm{ME}}$ is minimized, a negative loss change gives a positive predicted return gain $\Delta J_{\mathrm{surrogate}}$. Under the assumptions detailed below, we prove the bound with a TV advantage penalty and a KL entropy correction:

$$\begin{aligned}\Delta J_\tau &\geq \Delta J_{\mathrm{surrogate}} - \frac{4\gamma\epsilon_A}{(1-\gamma)^2}(D_{\mathrm{TV}}^{\max})^2 - \frac{\tau}{1-\gamma}D_{\mathrm{KL}}^{\max},\\ D_{\mathrm{TV}}^{\max} &:= \sup_s D_{\mathrm{TV}}(\pi_\theta(\cdot\mid s), \pi_{\mathrm{roll}}(\cdot\mid s))\,,\\ D_{\mathrm{KL}}^{\max} &:= \sup_s D_{\mathrm{KL}}(\pi_\theta(\cdot\mid s)\|\pi_{\mathrm{roll}}(\cdot\mid s))\,.\end{aligned} \tag{3}$$

Here $\epsilon_A = \sup_{s,a}|A_{\mathrm{roll}}(s,a)|$ bounds the soft advantage, defined below. If $\Delta J_{\mathrm{surrogate}}$ exceeds the sum of the two penalties, then $J_\tau(\pi_\theta) > J_\tau(\pi_{\mathrm{roll}})$.

Proof roadmap. The surrogate uses rollout states, whereas the return depends on states visited by the candidate policy. To bound the resulting prediction error, we proceed as follows.

1. **Express the surrogate gain.** Expand the KL losses in Substep 1.2, then average the resulting advantage and entropy correction over rollout states in Substep 1.4.
2. **Express the return gain in the same form.** Telescope the rollout value terms in Substep 2.2. In Substep 2.4, show that the two gains differ only in their state distributions.
3. **Bound the advantage error.** Bound the expected advantage in Substep 3.1 and the change in visited states in Substep 3.5. Combine these bounds into a TV penalty in Substep 3.6.
4. **Complete the return-improvement bound.** Bound the entropy correction in Substep 4.1, combine the two penalties in Substep 4.2, and state the sufficient condition for improvement in Substep 4.3.

We adapt the entropy-regularized performance-difference identity of Cayci et al. (2024, Lemma B.1) (Substep 2.4) and combine it with the TRPO state-distribution mismatch argument (Schulman et al., 2015) (Substep 3.6).

Notation and assumptions. Keep τ and π_{roll} fixed throughout the update, and let π_θ be the candidate policy. We use the main text's ΔJ_τ and $\Delta J_{\mathrm{surrogate}}$, with a positive sign denoting improvement in both cases. The exact rollout critic defines

$$A_{\mathrm{roll}}(s,a) = Q_{\mathrm{roll}}(s,a) - \tau\log\pi_{\mathrm{roll}}(a\mid s) - V_\tau^{\pi_{\mathrm{roll}}}(s), \qquad \epsilon_A := \sup_{s,a}|A_{\mathrm{roll}}(s,a)| < \infty.$$

Its expectation under the rollout policy is zero at every state. Define

$$\begin{aligned}\bar{A}_\theta(s) &:= \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}[A_{\mathrm{roll}}(s,a)],\\ k(s) &:= D_{\mathrm{KL}}(\pi_\theta(\cdot\mid s)\|\pi_{\mathrm{roll}}(\cdot\mid s))\,, \qquad D := \sup_s k(s) = D_{\mathrm{KL}}^{\max}(\pi_\theta\|\pi_{\mathrm{roll}}) < \infty.\end{aligned} \tag{36}$$

Here $\bar{A}_\theta(s)$ averages the rollout advantage over the candidate's actions. The divergence $k(s)$ measures change from the rollout policy; the KL in Eq. 2 instead measures distance to the Boltzmann target. We assume the relevant discounted expectations are finite and $\gamma^t\mathbb{E}_{\rho_0,P,\pi_\theta}[|V_\tau^{\pi_{\mathrm{roll}}}(s_t)|] \to 0$. The latter condition, satisfied for bounded soft values, removes the terminal term in Substep 2.2.

Step 1: express the surrogate improvement

We show that the surrogate gain is the rollout-state average of $\bar{A}_\theta(s) - \tau k(s)$, scaled by $1/(1-\gamma)$. This form will allow a direct comparison with the return gain.

Substep 1.1: Write the difference of the two losses. First, negate the candidate-minus-rollout loss change to obtain the predicted return gain, then expand it as a difference of KL divergences to the same Boltzmann target. Both surrogate losses use the same frozen critic Q_{roll}, Boltzmann target $p_{Q,\tau}$, and state distribution d_{roll}. Only the policy inside the KL changes. Starting from the definition of improvement in the main text and substituting Eq. 2,

$$\begin{aligned}\Delta J_{\mathrm{surrogate}} &= -\Delta\mathcal{L}_{\mathrm{ME}} = \mathcal{L}_{\mathrm{ME}}(\theta_{\mathrm{roll}};\theta_{\mathrm{roll}}) - \mathcal{L}_{\mathrm{ME}}(\theta;\theta_{\mathrm{roll}}) \\ &= \frac{\tau}{1-\gamma}\mathbb{E}_{s\sim d_{\mathrm{roll}}}\left[D_{\mathrm{KL}}(\pi_{\mathrm{roll}}\|p_{Q,\tau}) - D_{\mathrm{KL}}(\pi_\theta\|p_{Q,\tau})\right].\end{aligned}$$

In the following per-state calculations, all action distributions and action expectations are conditional on the same state s.

Substep 1.2: Expand the KL losses and cancel the normalizer. We turn the KL difference into a difference of entropy-augmented critic values. The target's normalizing constant cancels because it is identical in both losses. From Eq. 17, $\log p_{Q,\tau}(a \mid s) = Q_{\mathrm{roll}}(s,a)/\tau - \log Z_Q(s)$. For either $\pi = \pi_{\mathrm{roll}}$ or $\pi = \pi_\theta$, this gives

$$\begin{aligned}\tau D_{\mathrm{KL}}(\pi\|p_{Q,\tau}) &= \mathbb{E}_{a\sim\pi}[\tau\log\pi(a\mid s) - \tau\log p_{Q,\tau}(a\mid s)] \\ &= \mathbb{E}_{a\sim\pi}[\tau\log\pi(a\mid s) - Q_{\mathrm{roll}}(s,a)] + \tau\log Z_Q(s).\end{aligned}$$

The last term is outside the action expectation because it depends only on s. Subtracting the candidate's KL from the rollout policy's KL now makes the cancellation explicit:

$$\begin{aligned}&\tau\big[D_{\mathrm{KL}}(\pi_{\mathrm{roll}}\|p_{Q,\tau}) - D_{\mathrm{KL}}(\pi_\theta\|p_{Q,\tau})\big] \\ &\quad= \mathbb{E}_{a\sim\pi_{\mathrm{roll}}}[\tau\log\pi_{\mathrm{roll}}(a\mid s) - Q_{\mathrm{roll}}(s,a)] + \tau\log Z_Q(s) \\ &\qquad- \mathbb{E}_{a\sim\pi_\theta}[\tau\log\pi_\theta(a\mid s) - Q_{\mathrm{roll}}(s,a)] - \tau\log Z_Q(s) \\ &\quad= \mathbb{E}_{a\sim\pi_\theta}[Q_{\mathrm{roll}}(s,a) - \tau\log\pi_\theta(a\mid s)] - V_\tau^{\pi_{\mathrm{roll}}}(s).\end{aligned}$$

Here we used the soft-value definition from Section 2,

$$V_\tau^{\pi_{\mathrm{roll}}}(s) = \mathbb{E}_{a\sim\pi_{\mathrm{roll}}}[Q_{\mathrm{roll}}(s,a) - \tau\log\pi_{\mathrm{roll}}(a\mid s)].$$

Thus the rollout expectation in the subtraction equals $-V_\tau^{\pi_{\mathrm{roll}}}(s)$.

Substep 1.3: Identify the advantage and entropy correction. We now show that the remaining candidate expectation equals $\bar{A}_\theta(s) - \tau k(s)$. Add and subtract $\tau\log\pi_{\mathrm{roll}}(a\mid s)$ for each action:

$$\begin{aligned}&Q_{\mathrm{roll}}(s,a) - \tau\log\pi_\theta(a\mid s) - V_\tau^{\pi_{\mathrm{roll}}}(s) \\ &\quad= \underbrace{Q_{\mathrm{roll}}(s,a) - \tau\log\pi_{\mathrm{roll}}(a\mid s) - V_\tau^{\pi_{\mathrm{roll}}}(s)}_{A_{\mathrm{roll}}(s,a)} - \tau\big[\log\pi_\theta(a\mid s) - \log\pi_{\mathrm{roll}}(a\mid s)\big] \\ &\quad= A_{\mathrm{roll}}(s,a) - \tau\log\frac{\pi_\theta(a\mid s)}{\pi_{\mathrm{roll}}(a\mid s)}.\end{aligned}$$

Taking the action expectation under π_θ yields

$$\begin{aligned}&\mathbb{E}_{a\sim\pi_\theta}[Q_{\mathrm{roll}}(s,a) - \tau\log\pi_\theta(a\mid s)] - V_\tau^{\pi_{\mathrm{roll}}}(s) \\ &\quad= \underbrace{\mathbb{E}_{a\sim\pi_\theta}[A_{\mathrm{roll}}(s,a)]}_{\bar{A}_\theta(s)} - \tau\underbrace{\mathbb{E}_{a\sim\pi_\theta}\left[\log\frac{\pi_\theta(a\mid s)}{\pi_{\mathrm{roll}}(a\mid s)}\right]}_{D_{\mathrm{KL}}(\pi_\theta\|\pi_{\mathrm{roll}})=k(s)}.\end{aligned}$$

The action distribution in this expectation determines the direction of the KL: its first argument is the candidate policy.

Substep 1.4: Average over rollout states. We complete the surrogate identity by inserting the per-state result from Substep 1.3 into the loss difference from Substep 1.1:

$$\begin{aligned}\Delta J_{\mathrm{surrogate}} &= \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_{\mathrm{roll}}}[\mathbb{E}_{a\sim\pi_\theta}[Q_{\mathrm{roll}}(s,a) - \tau\log\pi_\theta(a\mid s)] - V_\tau^{\pi_{\mathrm{roll}}}(s)] \\ &= \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_{\mathrm{roll}}}[\bar{A}_\theta(s) - \tau k(s)].\end{aligned} \tag{37}$$

Thus, the surrogate gain is the candidate's expected rollout advantage, minus the correction needed to evaluate entropy under the candidate policy.

STEP 2: EXPRESS THE ACTUAL RETURN IMPROVEMENT

We show that ΔJ_τ averages the same quantity $\bar{A}_\theta(s) - \tau k(s)$ as the surrogate, but over the candidate's state distribution d^{π_θ}. We derive this identity directly from the return definition.

Substep 2.1: Subtract the returns using a common initial distribution. We first write the return difference as an expectation over candidate trajectories. Restating Eq. 1,

$$\begin{aligned} J_\tau(\pi_\theta) &= \frac{1}{1-\gamma}\mathbb{E}_{\substack{s\sim d^{\pi_\theta} \\ a\sim\pi_\theta(\cdot|s)}}[r(s,a) - \tau \log \pi_\theta(a \mid s)] \\ &= \mathbb{E}_{\rho_0,P,\pi_\theta}\left[\sum_{t=0}^{\infty}\gamma^t\big(r(s_t,a_t) - \tau \log \pi_\theta(a_t \mid s_t)\big)\right]. \end{aligned}$$

By definition, the rollout policy's soft value is its expected entropy-augmented return from a given state. Hence $J_\tau(\pi_{\text{roll}}) = \mathbb{E}_{s_0\sim\rho_0}[V_\tau^{\pi_{\text{roll}}}(s_0)]$. Both policies start from the same ρ_0, so this subtraction can be placed inside the expectation over candidate trajectories:

$$\Delta J_\tau = \mathbb{E}_{\rho_0,P,\pi_\theta}\left[\sum_{t=0}^{\infty}\gamma^t\big(r(s_t,a_t) - \tau \log \pi_\theta(a_t \mid s_t)\big) - V_\tau^{\pi_{\text{roll}}}(s_0)\right].$$

Substep 2.2: Telescope the rollout values. We replace the subtracted initial value by a sum of differences between successive values. This puts every contribution to ΔJ_τ inside the same discounted sum. For any trajectory and finite horizon N, the intermediate value terms cancel:

$$\begin{aligned} &\sum_{t=0}^{N}\gamma^t\big[\gamma V_\tau^{\pi_{\text{roll}}}(s_{t+1}) - V_\tau^{\pi_{\text{roll}}}(s_t)\big] = \sum_{t=1}^{N+1}\gamma^t V_\tau^{\pi_{\text{roll}}}(s_t) - \sum_{t=0}^{N}\gamma^t V_\tau^{\pi_{\text{roll}}}(s_t) \\ &\quad = -V_\tau^{\pi_{\text{roll}}}(s_0) + \gamma^{N+1}V_\tau^{\pi_{\text{roll}}}(s_{N+1}). \end{aligned}$$

This cancellation also holds along candidate trajectories: it only uses successive evaluations of the same rollout value function. Taking expectations and letting $N \to \infty$, the terminal term vanishes by the assumption stated above. Substituting this sum for the initial-value subtraction therefore gives

$$\Delta J_\tau = \mathbb{E}_{\rho_0,P,\pi_\theta}\left[\sum_{t=0}^{\infty}\gamma^t\Big(r(s_t,a_t) - \tau \log \pi_\theta(a_t \mid s_t) + \gamma V_\tau^{\pi_{\text{roll}}}(s_{t+1}) - V_\tau^{\pi_{\text{roll}}}(s_t)\Big)\right].$$

Each summand is the reward plus entropy contribution at the current step, plus the discounted next-state value, minus the current-state value.

Substep 2.3: Evaluate the conditional one-step contribution. We show that each summand, averaged over the candidate's action and the next state, equals $\bar{A}_\theta(s) - \tau k(s)$. For fixed s and a, the rollout Bellman equation is

$$Q_{\text{roll}}(s,a) = r(s,a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}[V_\tau^{\pi_{\text{roll}}}(s')].$$

Thus, averaging over the next state replaces $r(s,a) + \gamma V_\tau^{\pi_{\text{roll}}}(s')$ by $Q_{\text{roll}}(s,a)$. Averaging next over the candidate's action and using Substep 1.3 gives

$$\begin{aligned} &\mathbb{E}_{\substack{a\sim\pi_\theta(\cdot|s) \\ s'\sim P(\cdot|s,a)}}[r(s,a) - \tau \log \pi_\theta(a \mid s) + \gamma V_\tau^{\pi_{\text{roll}}}(s') - V_\tau^{\pi_{\text{roll}}}(s)] \\ &\quad = \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}[Q_{\text{roll}}(s,a) - \tau \log \pi_\theta(a \mid s) - V_\tau^{\pi_{\text{roll}}}(s)] \\ &\quad = \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[A_{\text{roll}}(s,a) - \tau \log \frac{\pi_\theta(a \mid s)}{\pi_{\text{roll}}(a \mid s)}\right] = \bar{A}_\theta(s) - \tau k(s). \end{aligned}$$

These nested averages are the law of total expectation applied to the candidate trajectories. The value functions remain those of the rollout policy throughout.

Substep 2.4: Average over candidate states and compare the gains. We now obtain the return-improvement identity and compare it with Eq. 37. At time t, these states follow $\rho_t^{\pi_\theta}$. Using $d^{\pi_\theta} = (1-\gamma)\sum_{t\geq 0}\gamma^t\rho_t^{\pi_\theta}$, we obtain

$$\Delta J_\tau = \sum_{t=0}^{\infty}\gamma^t\mathbb{E}_{s\sim\rho_t^{\pi_\theta}}[\bar{A}_\theta(s) - \tau k(s)] = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_\theta}}[\bar{A}_\theta(s) - \tau k(s)]. \tag{38}$$

For a direct comparison, restate the actual return improvement and the surrogate improvement from Eq. 37:

$$\Delta J_\tau = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_\theta}}[\bar{A}_\theta(s) - \tau k(s)], \qquad \Delta J_{\text{surrogate}} = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_{\text{roll}}}[\bar{A}_\theta(s) - \tau k(s)].$$

With an exact rollout critic and fixed temperature, the two improvements differ only in how states are weighted. Both use the candidate policy's actions inside $\bar{A}_\theta(s)$ and $k(s)$. The surrogate evaluates the candidate policy on states visited by the rollout policy, whereas the actual return reflects states visited by the candidate itself. If these state distributions coincide, the two improvements are exactly equal. The remaining proof bounds the discrepancy caused by changing the visited states.

STEP 3: BOUND THE ERROR IN THE ADVANTAGE TERM

We bound the error from averaging $\bar{A}_\theta$ over rollout states instead of candidate states. The argument has two parts: a small policy change limits both the expected rollout advantage and the change in visited states. Their product gives the advantage penalty in Eq. 3.

Substep 3.1: Bound the expected rollout advantage. We show that $|\bar{A}_\theta(s)| \le 2\epsilon_A\alpha$, where α is the largest policy change in total variation. Recall ϵ_A and define α by

$$\epsilon_A := \sup_{s,a} |A_{\text{roll}}(s,a)|,$$
$$\alpha := \sup_s D_{\text{TV}}(\pi_\theta(\cdot \mid s), \pi_{\text{roll}}(\cdot \mid s)) = \frac{1}{2}\sup_s \int_{\mathcal{A}} |\pi_\theta(a \mid s) - \pi_{\text{roll}}(a \mid s)|\,\mathrm{d}a.$$

Here ϵ_A bounds the advantage magnitude, while $\alpha \in [0,1]$ measures the largest policy change across states in total variation (TV). By the soft-value definition,

$$\int_{\mathcal{A}} \pi_{\text{roll}}(a \mid s) A_{\text{roll}}(s,a)\,\mathrm{d}a = \mathbb{E}_{a\sim\pi_{\text{roll}}}[Q_{\text{roll}}(s,a) - \tau \log \pi_{\text{roll}}(a \mid s)] - V_\tau^{\pi_{\text{roll}}}(s) = 0.$$

Subtracting this zero from $\bar{A}_\theta(s) = \int_{\mathcal{A}} \pi_\theta(a \mid s) A_{\text{roll}}(s,a)\,\mathrm{d}a$ introduces the difference between the two policies. The triangle inequality and the definitions above give

$$\begin{aligned} |\bar{A}_\theta(s)| &= \left| \int_{\mathcal{A}} [\pi_\theta(a \mid s) - \pi_{\text{roll}}(a \mid s)] A_{\text{roll}}(s,a)\,\mathrm{d}a \right| \\ &\le \int_{\mathcal{A}} |\pi_\theta(a \mid s) - \pi_{\text{roll}}(a \mid s)||A_{\text{roll}}(s,a)|\,\mathrm{d}a \\ &\le \epsilon_A \int_{\mathcal{A}} |\pi_\theta(a \mid s) - \pi_{\text{roll}}(a \mid s)|\,\mathrm{d}a \le 2\epsilon_A\alpha. \end{aligned} \tag{39}$$

Substep 3.2: Derive the balance equation for discounted visits. To compare state distributions, we first derive $d^\pi = (1-\gamma)\rho_0 + \gamma d^\pi P_\pi$ from the definition of discounted visitation. This identity separates the initial visit from all later visits. We begin by defining the one-step transition operator P_π. Starting at state s, first draw an action from $\pi(\cdot \mid s)$, then draw the next state from $P(\cdot \mid s,a)$. Averaging over the action gives the next-state distribution under the policy:

$$P_\pi(s' \mid s) := \int_{\mathcal{A}} \pi(a \mid s) P(s' \mid s,a)\,\mathrm{d}a.$$

This conditional distribution is called the policy's transition kernel. If the current state has distribution ρ, the next-state distribution is written ρP_π, meaning

$$(\rho P_\pi)(s') := \int \rho(s) P_\pi(s' \mid s)\,\mathrm{d}s.$$

Thus multiplication by P_π means advancing a state distribution by one time step. In particular, $\rho^\pi_{t+1} = \rho^\pi_t P_\pi$.

Next, separate the initial visit from the later visits. The definition of d^π includes every time step, starting at $t=0$. Separating out its first term gives an equivalent expression:

$$d^\pi := (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \rho^\pi_t = (1-\gamma)\rho_0 + (1-\gamma)\sum_{t=1}^{\infty} \gamma^t \rho^\pi_t.$$

The $t = 0$ term is $(1-\gamma)\gamma^0\rho_0^\pi = (1-\gamma)\rho_0$. Both expressions describe the same distribution; the second writes its initial-state contribution separately.

Now rewrite the remaining sum. Set $u = t-1$ and use $\rho_{u+1}^\pi = \rho_u^\pi P_\pi$:

$$(1-\gamma)\sum_{t=1}^{\infty}\gamma^t\rho_t^\pi = \gamma(1-\gamma)\sum_{u=0}^{\infty}\gamma^u\rho_{u+1}^\pi$$
$$= \gamma(1-\gamma)\sum_{u=0}^{\infty}\gamma^u(\rho_u^\pi P_\pi) = \gamma\underbrace{\left[(1-\gamma)\sum_{u=0}^{\infty}\gamma^u\rho_u^\pi\right]}_{d^\pi}P_\pi = \gamma d^\pi P_\pi.$$

The same transition operator P_π acts linearly on every term, so we can first form the weighted sum in brackets. This bracket is exactly the original definition of d^π, which explains why it reappears. Applying P_π advances every visit by one step, and the factor γ adds the corresponding discount: the first term of $\gamma d^\pi P_\pi$ is $(1-\gamma)\gamma\rho_1^\pi$. Thus it collects visits from time 1 onward. Adding the initial visit gives

$$d^\pi = (1-\gamma)\rho_0 + \gamma d^\pi P_\pi.$$

Substep 3.3: Separate propagation from policy change. We express the difference in visited states as two contributions: propagation of an existing state difference, and the effect of choosing different actions. Apply the balance equation from Substep 3.2 to each policy. Both policies use the same ρ_0, so their initial-state terms cancel. We then add and subtract $\gamma d_{\text{roll}}P_{\pi_\theta}$:

$$d^{\pi_\theta} - d_{\text{roll}} = \gamma\big[d^{\pi_\theta}P_{\pi_\theta} - d_{\text{roll}}P_{\pi_{\text{roll}}}\big] = \gamma\big[d^{\pi_\theta}P_{\pi_\theta} - d_{\text{roll}}P_{\pi_\theta} + d_{\text{roll}}P_{\pi_\theta} - d_{\text{roll}}P_{\pi_{\text{roll}}}\big]$$
$$= \gamma(d^{\pi_\theta} - d_{\text{roll}})P_{\pi_\theta} + \gamma d_{\text{roll}}(P_{\pi_\theta} - P_{\pi_{\text{roll}}}).$$

The first term propagates the difference in current-state distributions through the same candidate transition kernel. The second starts from the same rollout state distribution and measures the effect of changing the policy's action probabilities. We bound these two contributions separately below.

Substep 3.4: Bound propagation through a common transition kernel. We show that one step under the same policy cannot increase the L^1 distance between state distributions. This bounds the first contribution from Substep 3.3. Write $\delta d := d^{\pi_\theta} - d_{\text{roll}}$ for the signed difference of the two state distributions, with $\|\delta d\|_1 := \int |\delta d(s)|\,\mathrm{d}s$. Taking norms in the preceding identity and applying the triangle inequality gives

$$\|\delta d\|_1 \le \gamma\|\delta d P_{\pi_\theta}\|_1 + \gamma\|d_{\text{roll}}(P_{\pi_\theta} - P_{\pi_{\text{roll}}})\|_1.$$

For the first term, use the nonnegativity and normalization of the transition probabilities:

$$\|\delta d P_{\pi_\theta}\|_1 = \int\left|\int \delta d(s)P_{\pi_\theta}(s' \mid s)\,\mathrm{d}s\right|\mathrm{d}s' \le \int|\delta d(s)|\underbrace{\left[\int P_{\pi_\theta}(s' \mid s)\,\mathrm{d}s'\right]}_{1}\mathrm{d}s = \|\delta d\|_1.$$

Thus applying the same transition kernel cannot increase the L^1 distance between state distributions.

Substep 3.5: Bound policy change and the resulting state shift. We bound the second contribution by 2α. Combining it with Substep 3.4 then gives $\|d^{\pi_\theta} - d_{\text{roll}}\|_1 \le 2\gamma\alpha/(1-\gamma)$. First fix a state s and expand the two policy-averaged transition kernels:

$$\int|P_{\pi_\theta}(s' \mid s) - P_{\pi_{\text{roll}}}(s' \mid s)|\,\mathrm{d}s' = \int\left|\int_{\mathcal{A}}[\pi_\theta(a \mid s) - \pi_{\text{roll}}(a \mid s)]P(s' \mid s,a)\,\mathrm{d}a\right|\mathrm{d}s'$$
$$\le \int_{\mathcal{A}}|\pi_\theta(a \mid s) - \pi_{\text{roll}}(a \mid s)|\underbrace{\left[\int P(s' \mid s,a)\,\mathrm{d}s'\right]}_{1}\mathrm{d}a \le 2\alpha.$$

The last inequality is the definition of α: at every state, the L^1 distance between the action distributions is at most 2α. Averaging over the normalized rollout state distribution preserves this bound:

$$\|d_{\text{roll}}(P_{\pi_\theta} - P_{\pi_{\text{roll}}})\|_1 \le \int d_{\text{roll}}(s)\left[\int|P_{\pi_\theta}(s' \mid s) - P_{\pi_{\text{roll}}}(s' \mid s)|\,\mathrm{d}s'\right]\mathrm{d}s$$
$$\le 2\alpha\int d_{\text{roll}}(s)\,\mathrm{d}s = 2\alpha.$$

Substituting these two bounds into the triangle inequality gives $\|\delta d\|_1 \le \gamma\|\delta d\|_1 + 2\gamma\alpha$. Move the term $\gamma\|\delta d\|_1$ to the left and divide by $1-\gamma > 0$. This gives

$$(1-\gamma)\|d^{\pi_\theta} - d_{\text{roll}}\|_1 \le 2\gamma\alpha, \qquad \|d^{\pi_\theta} - d_{\text{roll}}\|_1 \le \frac{2\gamma\alpha}{1-\gamma}. \tag{40}$$

Substep 3.6: Combine the advantage bounds in TV. We now bound the advantage-expectation error by multiplying $\sup_s |\bar{A}_\theta(s)|$ by the state-distribution distance. Both bounds are proportional to α, so their product is proportional to α^2. Combining Eqs. 39 and 40 gives

$$\begin{aligned}\frac{1}{1-\gamma}\left|\mathbb{E}_{d^{\pi_\theta}}[\bar{A}_\theta] - \mathbb{E}_{d_{\text{roll}}}[\bar{A}_\theta]\right| &\le \frac{\|d^{\pi_\theta} - d_{\text{roll}}\|_1 \sup_s |\bar{A}_\theta(s)|}{1-\gamma} \\ &\le \frac{4\gamma\epsilon_A}{(1-\gamma)^2}\alpha^2.\end{aligned} \tag{41}$$

This step bounds the contribution of the advantage term to the discrepancy between ΔJ_τ and $\Delta J_{\text{surrogate}}$. Small policy changes limit both the change in visited states and the expected rollout advantage, producing the quadratic total-variation bound above. Step 4 accounts for the remaining entropy correction.

STEP 4: COMPLETE THE RETURN-IMPROVEMENT BOUND

We bound the remaining entropy correction and add it to the advantage error. This proves Eq. 3 and gives a sufficient condition for J_τ to improve.

Substep 4.1: Bound the entropy correction. Subtracting the two gain identities from Substep 2.4, the contribution from $-\tau k(s)$ is $\frac{\tau}{1-\gamma}(\mathbb{E}_{d_{\text{roll}}}[k] - \mathbb{E}_{d^{\pi_\theta}}[k])$. We show that this contribution is at least $-\tau D/(1-\gamma)$. Since $0 \le k(s) \le D$, both state averages lie in $[0, D]$. Hence

$$\frac{\tau}{1-\gamma}\left(\mathbb{E}_{s\sim d_{\text{roll}}}[k(s)] - \mathbb{E}_{s\sim d^{\pi_\theta}}[k(s)]\right) \ge -\frac{\tau}{1-\gamma}D.$$

Substep 4.2: Combine the advantage and entropy bounds. We obtain the claimed lower bound on ΔJ_τ by subtracting the two gain identities and applying Substep 3.6 to the advantage term, then Substep 4.1 to the entropy term:

$$\begin{aligned}\Delta J_\tau &\ge \Delta J_{\text{surrogate}} - \frac{4\gamma\epsilon_A}{(1-\gamma)^2}\alpha^2 + \frac{\tau}{1-\gamma}\left(\mathbb{E}_{s\sim d_{\text{roll}}}[k(s)] - \mathbb{E}_{s\sim d^{\pi_\theta}}[k(s)]\right) \\ &\ge \Delta J_{\text{surrogate}} - \frac{4\gamma\epsilon_A}{(1-\gamma)^2}\alpha^2 - \frac{\tau}{1-\gamma}D.\end{aligned} \tag{42}$$

This proves Eq. 3. The first penalty bounds the error in the advantage prediction; the second bounds the error in its entropy correction. Both arise from evaluating the update on rollout states instead of the states visited by the candidate.

Substep 4.3: State the improvement condition and its practical meaning. The bound guarantees an increase in J_τ when the surrogate gain exceeds the two error penalties. Explicitly,

$$\Delta J_{\text{surrogate}} > \frac{4\gamma\epsilon_A}{(1-\gamma)^2}\alpha^2 + \frac{\tau}{1-\gamma}D \quad \Longrightarrow \quad J_\tau(\pi_\theta) > J_\tau(\pi_{\text{roll}}).$$

To express a sufficient condition using only KL, apply Pinsker's inequality, $2\alpha^2 \le D$:

$$\frac{4\gamma\epsilon_A}{(1-\gamma)^2}\alpha^2 + \frac{\tau}{1-\gamma}D \le \left(\frac{2\gamma\epsilon_A}{(1-\gamma)^2} + \frac{\tau}{1-\gamma}\right)D.$$

The surrogate gain exceeding this KL upper bound is therefore also sufficient for return improvement. In words, the predicted improvement must exceed the bound on the prediction error. A small KL alone does not imply improvement, and failing this sufficient condition does not imply that the update is harmful.

This motivates the KL term in Eq. 4: minimize the surrogate while limiting policy change. Replacing the worst-state KL constraint by a rollout-average constraint gives the average penalty. With the critic and rollout distribution fixed, the full Lagrangian satisfies

$$\begin{aligned}\mathcal{L}(\pi_\theta, \tau, \lambda) = (1-\gamma)\mathcal{L}_{\text{ME}}(\theta; \theta_{\text{roll}}) + \lambda\mathbb{E}_{s\sim d_{\text{roll}}}[k(s)] \\ + \tau\mathcal{H}_{\text{target}} - \lambda\delta_{\text{KL}} - \tau\mathbb{E}_{s\sim d_{\text{roll}}}[\log Z_Q(s)].\end{aligned}$$

The second line is independent of the policy when the multipliers are fixed. The average constraint is a practical relaxation. With this relaxation and an estimated critic, minimizing $\mathcal{L}$ over the policy does not by itself guarantee the sufficient condition above.

B.7 Trust regions for Gaussian policies

For Gaussian policies, a bound on either KL direction also bounds the other. A TV bound strictly below one bounds both KL directions. The short proofs below give conservative constants sufficient to control the entropy correction in Eq. 3.

Setup and notation. Fix a state and suppress the conditioning on s. Assume positive-definite covariances, meaning every variance is strictly positive for our diagonal Gaussian policies, and use natural logarithms. KL and TV are unchanged by a common invertible transformation: its Jacobian cancels in the density ratio and the integral. We can therefore undo the common tanh squashing and action rescaling, then subtract the rollout mean and divide by the rollout standard deviation in each coordinate. In these standardized coordinates,

$$\pi_{\mathrm{roll}} = \mathcal{N}(0, I), \qquad \pi_{\mathrm{curr}} = \mathcal{N}\big(m, \mathrm{diag}(\sigma_1^2, \dots, \sigma_d^2)\big)\,.$$

Here m_i is the standardized mean change, and σ_i^2 is the current variance relative to the rollout variance. Full covariance matrices reduce to the same form by whitening and rotation. The Gaussian KL formulas separate variance changes from mean changes:

$$\begin{aligned} D_{\mathrm{KL}}(\pi_{\mathrm{roll}} \| \pi_{\mathrm{curr}}) &= \frac{1}{2} \sum_i \left[\log \sigma_i^2 + \frac{1}{\sigma_i^2} - 1 + \frac{m_i^2}{\sigma_i^2} \right], \\ D_{\mathrm{KL}}(\pi_{\mathrm{curr}} \| \pi_{\mathrm{roll}}) &= \frac{1}{2} \sum_i \left[\sigma_i^2 - 1 - \log \sigma_i^2 + m_i^2 \right]. \end{aligned} \tag{43}$$

Each variance term and each mean term is nonnegative.

B.7.1 Forward KL bounds reverse KL

Suppose $D_{\mathrm{KL}}(\pi_{\mathrm{roll}} \| \pi_{\mathrm{curr}}) \le \delta$. We prove the explicit bound

$$\boxed{D_{\mathrm{KL}}(\pi_{\mathrm{curr}} \| \pi_{\mathrm{roll}}) \le \delta e^{1+2\delta}.} \tag{44}$$

This bound tends to zero with δ and does not depend on the action dimension. A sharper bound is given by Zhang et al. (2023, Theorem 1).

Step 1: bound the variance ratios. Since every contribution to forward KL is nonnegative,

$$\log \sigma_i^2 + \frac{1}{\sigma_i^2} - 1 \le 2\delta.$$

Rearranging and using $1/\sigma_i^2 > 0$ gives

$$\log \sigma_i^2 \le 1 + 2\delta - \frac{1}{\sigma_i^2} \le 1 + 2\delta.$$

Exponentiating preserves the inequality, so

$$\sigma_i^2 \le e^{1+2\delta} =: M_\delta.$$

This is a conservative upper bound on the variance; the corresponding standard deviation satisfies $\sigma_i \le e^{1/2+\delta}$.

Step 2: compare the KL terms. For any $x > 0$,

$$x - 1 - \log x \le \max\{1, x\}\big(x^{-1} - 1 + \log x\big). \tag{45}$$

To see this, both scalar expressions vanish at $x = 1$, and the derivative of the left expression is x times the derivative of $x^{-1} - 1 + \log x$. Integrating between 1 and x bounds this factor by $\max\{1, x\}$.

Apply Eq. 45 with $x = \sigma_i^2$. Step 1 gives both $M_\delta \geq 1$ and $\sigma_i^2 \leq M_\delta$, so $\max\{1, \sigma_i^2\} \leq M_\delta$. Thus the variance contribution in each coordinate satisfies

$$\begin{aligned}\sigma_i^2 - 1 - \log \sigma_i^2 &\leq \max\{1, \sigma_i^2\} \left(\frac{1}{\sigma_i^2} - 1 + \log \sigma_i^2 \right) \\ &\leq M_\delta \left(\frac{1}{\sigma_i^2} - 1 + \log \sigma_i^2 \right).\end{aligned}$$

The mean contribution satisfies the same comparison:

$$m_i^2 = \sigma_i^2 \frac{m_i^2}{\sigma_i^2} \leq M_\delta \frac{m_i^2}{\sigma_i^2}.$$

Add these two inequalities in each coordinate, then sum over i. The same M_δ applies to every coordinate, so it factors out:

$$\begin{aligned}D_{\mathrm{KL}}(\pi_{\mathrm{curr}} \| \pi_{\mathrm{roll}}) &= \frac{1}{2} \sum_i \left[\sigma_i^2 - 1 - \log \sigma_i^2 + m_i^2 \right] \\ &\leq \frac{1}{2} \sum_i \left[M_\delta \left(\frac{1}{\sigma_i^2} - 1 + \log \sigma_i^2 \right) + M_\delta \frac{m_i^2}{\sigma_i^2} \right] \\ &= M_\delta \frac{1}{2} \sum_i \left[\log \sigma_i^2 + \frac{1}{\sigma_i^2} - 1 + \frac{m_i^2}{\sigma_i^2} \right] \\ &= M_\delta D_{\mathrm{KL}}(\pi_{\mathrm{roll}} \| \pi_{\mathrm{curr}}) \leq M_\delta \delta.\end{aligned}$$

The last inequality uses the assumed forward-KL budget $D_{\mathrm{KL}}(\pi_{\mathrm{roll}} \| \pi_{\mathrm{curr}}) \leq \delta$, proving Eq. 44. Exchanging the policies proves the reverse implication. If their covariances are equal, the two KL divergences are exactly equal.

B.7.2 TV BOUNDS BOTH KL DIRECTIONS

At a fixed state, Gaussians with positive-definite covariances satisfy $\rho := D_{\mathrm{TV}}(\pi_{\mathrm{roll}}, \pi_{\mathrm{curr}}) < 1$. Let $B_{\mathrm{G}}(\delta)$ denote the dimension-independent reverse-KL bound given a forward-KL budget δ, established by Zhang et al. (2023, Theorem 1). This bound is finite and tends to zero with δ. Writing $t_\rho = -2 \log(1 - \rho)$, we obtain the following corollary:

$$\boxed{\max\{D_{\mathrm{KL}}(\pi_{\mathrm{curr}} \| \pi_{\mathrm{roll}}), D_{\mathrm{KL}}(\pi_{\mathrm{roll}} \| \pi_{\mathrm{curr}})\} \leq \underbrace{2B_{\mathrm{G}}(t_\rho) + t_\rho}_{C_{\mathrm{TV}}(\rho)}.} \tag{46}$$

Proof. Define the affinity and normalized geometric mean

$$A = \int \sqrt{\pi_{\mathrm{roll}}(a) \pi_{\mathrm{curr}}(a)} \, \mathrm{d}a, \qquad \pi_{\mathrm{mid}}(a) = \frac{\sqrt{\pi_{\mathrm{roll}}(a) \pi_{\mathrm{curr}}(a)}}{A}.$$

The distribution π_{mid} is Gaussian because its log density is quadratic. The TV–Hellinger inequality gives $A \geq 1 - \rho$ (Devroye et al., 2023, Section 2). Substituting the definition of π_{mid} yields

$$D_{\mathrm{KL}}(\pi_{\mathrm{mid}} \| \pi_{\mathrm{roll}}) + D_{\mathrm{KL}}(\pi_{\mathrm{mid}} \| \pi_{\mathrm{curr}}) = -2 \log A \leq t_\rho.$$

Both terms are nonnegative, so each is at most t_ρ. Zhang et al.'s theorem therefore gives $D_{\mathrm{KL}}(\pi_{\mathrm{curr}} \| \pi_{\mathrm{mid}}) \leq B_{\mathrm{G}}(t_\rho)$. Finally, the same density identity implies

$$\begin{aligned}D_{\mathrm{KL}}(\pi_{\mathrm{curr}} \| \pi_{\mathrm{roll}}) &= 2 D_{\mathrm{KL}}(\pi_{\mathrm{curr}} \| \pi_{\mathrm{mid}}) - 2 \log A \\ &\leq 2 B_{\mathrm{G}}(t_\rho) + t_\rho.\end{aligned}$$

Exchanging the policies proves the other direction. □

Thus $C_{\mathrm{TV}}(\rho)$ is finite for every $\rho < 1$ and tends to zero as $\rho \to 0$. A common invertible tanh transformation preserves both divergences. Applying this result uniformly across states requires a common TV bound strictly below one.

B.7.3 IMPLICATIONS FOR THE RETURN BOUND

If the forward-KL budget δ holds at every state, Pinsker and Eq. 44 give $(D_{\mathrm{TV}}^{\max})^2 \leq \delta/2$ and $D_{\mathrm{KL}}^{\max} \leq \delta e^{1+2\delta}$. A uniform TV budget $\rho < 1$ instead gives $D_{\mathrm{KL}}^{\max} \leq C_{\mathrm{TV}}(\rho)$. Thus either constraint controls both penalties in Eq. 3. Small temperature further reduces the entropy penalty. The exact-critic and bounded-soft-advantage assumptions still apply, and the surrogate gain must exceed the penalties to guarantee improvement.

These are uniform, population-level statements. REPPO's sampled forward-KL constraint (Voelcker et al., 2026) and FERPO's state-average ESS objective do not establish worst-state control. For FERPO, the fitting condition of Proposition 1 and a population ESS floor $e_\infty(s) \geq e_0 > e^{-1/2}$ at every state would give $D_{\mathrm{TV}}^{\max} \leq \sqrt{-2\log e_0} < 1$; the TV result then also bounds the entropy-related KL. These comparisons concern the two Gaussian actors; the unrestricted target q^* need not be Gaussian.