BEHAVIOR-CONSISTENT DEEP REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) often exhibits high variance across training runs, leading to unreliable performance and posing a major challenge to deployment in real-world domains. In this work, we address the challenge of cross-run policy divergence by formalizing the problem of *behavior-consistent RL*, where the objective is to obtain policies that are both high-performing and distributionally similar across training runs. Our key observation is that maximum-entropy RL provides a direct mechanism for controlling behavioral divergence by anchoring runs to a common (uniform) prior. We prove that, for Boltzmann policies, choosing the temperature proportional to Q-function disagreement bounds the pairwise KL divergence between the induced policies. However, we also show that naïvely increasing entropy might impair policy optimization while amplifying off-policy error. Building upon these observations, we propose Q-value Expectile Disagreement (QED), a state-dependent temperature schedule that uses double-critic disagreement as a single-run proxy for cross-run disagreement. Empirically, we demonstrate that across 18 continuous-control tasks, QED reduces across-run divergence by two orders of magnitude without sacrificing performance, resulting in a considerable reduction in return variance at modest sample-efficiency costs.

1 INTRODUCTION

Reinforcement learning (RL) is a powerful tool for solving control problems. Yet, the lack of consistency across independent training runs remains a persistent and under-addressed challenge (Colas et al., 2018; Henderson et al., 2018; Paleu & Pascal, 2023). Small changes in random initialization, data ordering, or environment stochasticity can yield policies that differ dramatically in performance (Islam et al., 2017; Henderson et al., 2018; Bjorck et al., 2022). Even when final performance returns appear similar, learned policies can exhibit different behaviors (Clary et al., 2018; Flageat et al., 2024; Eaton et al., 2023; 2026), leading to variations in robustness and failure modes.

This variability poses a fundamental obstacle to both scientific progress (Henderson et al., 2018) and practical deployment (Lynnerup et al., 2020; Cavenaghi et al., 2023). From a scientific perspective, high variance across runs complicates evaluation and often requires averaging over numerous random seeds to obtain statistically significant comparisons (Colas et al., 2018; Agarwal et al., 2021). Stochasticity can even cause performance to vary for a fixed policy (Flageat et al., 2024). In practice, the consequences are more severe. When policies trained under identical conditions yield qualitatively different behavior, iterative processes such as reward design (Booth et al., 2023), debugging, and safety validation become unreliable. Here, apparent improvements may reflect stochastic idiosyncrasies rather than the modification. For example, in deployment settings such as large-language-model fine-tuning, policies trained on such rewards might cause regular updates to feel qualitatively different to users in terms of helpfulness, tone, or safety.

While prior work has studied RL instability through statistical metrics such as return variance (Colas et al., 2018; Agarwal et al., 2021), behavioral divergence across runs has received comparatively little algorithmic attention (see Appendix B for related work). Existing algorithmic interventions that reduce variability across RL training runs or during learning remain sparse (Bjorck et al., 2022; Tang & Berseth, 2024). Related theoretical work formalizes replicability as requiring identical outputs across executions (Eaton et al., 2023; 2026). Eaton et al. (2026) also provide an empirical method inspired by their theory that obtains increased cross-execution policy agreement in a small set of

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.

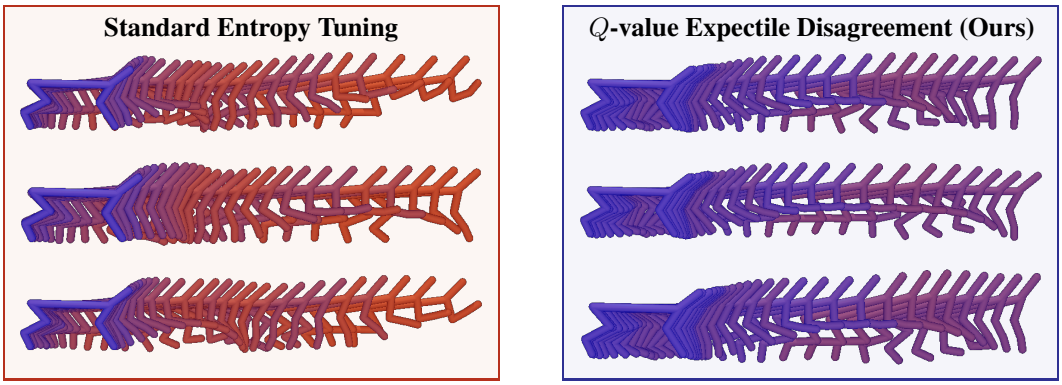


Figure 1: **QED makes independently trained policies visibly behavior-consistent.** Visualization of policies from three different training runs on the `cheetah_run` task, comparing (left) traditional entropy autotuning (Haarnoja et al., 2018b) against (right) our approach (QED). Color shade denotes the mean pairwise L_2 distance between state vectors at each timestep: blue is low, red is high.

Atari experiments. We go beyond this by studying how behavioral consistency can be achieved under the practical requirements of any deep RL setting, including those of continuous control.

In many applications, from robotics to conversational AI systems, the consistency of behavior is often as critical as its performance. In this work, we argue that **behavioral consistency should be treated as a core goal in RL**. We formalize this perspective through the new framework of *behavior-consistent RL*, in which the goal is to learn policies that are both high-performing and distributionally similar across independent runs. This view extends the focus of RL beyond achieving high expected return to also ensuring that learning outcomes are stable and replicable.

Toward this goal, our key insight is that maximum entropy (MaxEnt) RL provides a natural mechanism for inducing behavioral consistency. By regularizing policies toward a common prior (Ziebart et al., 2008; Levine, 2018), entropy shrinks the policy space and restricts the range of behaviors independent runs can realize. Yet, naïvely increasing entropy can amplify off-policy error. These competing effects reveal a tradeoff: maximizing entropy promotes consistency but can degrade performance. To resolve this tradeoff, we establish a theoretical connection between policy divergence and Q-disagreement. We prove that, for Boltzmann policies, scaling the temperature in proportion to Q-disagreement bounds the divergence between the induced policies. This suggests a principled approach where entropy should be high when Q-estimates are uncertain and decrease as they converge.

Based on this insight, we introduce Q-value Expectile Disagreement (QED), a practical algorithm for adaptive entropy tuning in RL. QED increases temperature early to achieve consistency and cools down over time for convergence. We evaluate QED on top of two strong base algorithms and show that it reduces pairwise policy divergence across independent runs by up to two orders of magnitude while maintaining competitive performance. QED produces quantitatively and qualitatively more similar behavior, and can reduce return variance by around 50%. To the best of our knowledge, this is the first algorithmic intervention that directly achieves behavioral similarity in deep RL across independent algorithm executions in highly continuous control tasks. These results show that controlling behavioral consistency in RL is both feasible and beneficial, offering a new perspective on stability and replicability in RL. **Our key contributions include:**

1. We introduce the setting of *behavior-consistent reinforcement learning* (BRL).
2. We prove a relationship between policy divergence and critic disagreement in MaxEnt RL.
3. We illustrate the off-policy challenges in high-entropy MaxEnt RL.
4. We introduce QED, a method that increases behavioral similarity and reduces return variance.

2 Preliminaries

We consider the RL problem (Sutton & Barto, 2018) of finding an optimal policy in a Markov decision process (MDP) (Puterman, 1994) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$, with states $\mathcal{S}$, actions $\mathcal{A}$, transition function $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and discount factor $\gamma \in (0, 1)$. In general, this

paper focuses on continuous state-action spaces but for ease of exposition we will state all definitions and results with discrete action spaces.

A policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ attains $J(\pi) = \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty} \gamma^t\, r(s_t, a_t)]$, and the optimal policy is $\pi^\star = \arg\max_\pi J(\pi)$. In practice, one learns the action-value function

$$Q^\pi(s,a) = \mathbb{E}_{\tau\sim\pi}\Big[\sum_{t=0}^{\infty} \gamma^t\, r(s_t,a_t) \;\Big|\; s_0 = s, a_0 = a\Big]$$

and acquires π via approximate policy improvement with respect to Q^π, noting that the Q^π-greedy policy $\pi_{\text{greedy}}(s) \in \arg\max_{a\in\mathcal{A}} Q^\pi(s,a)$ is optimal when $Q^\pi = Q^\star$.

KL-Constrained RL. KL-regularized RL (Todorov, 2009; Peters et al., 2010) provides a principled surrogate for this idea by constraining each policy improvement step relative to a known *reference* (or *prior*) policy density $\pi_0(\cdot \mid s)$. A standard formulation is the KL-constrained objective

$$\max_\pi \;\; \mathbb{E}_{\tau\sim\pi}\Big[\sum_{t=0}^{\infty} \gamma^t \Big(r(s_t,a_t) \;-\; \alpha\, D_{\text{KL}}\big(\pi(\cdot \mid s_t) \,\|\, \pi_0(\cdot \mid s_t)\big)\Big)\Big],$$

with $\alpha > 0$ controlling the strength of the constraint. This induces a prior-weighted Boltzmann policy: for each state s, the optimal action probabilities are proportional to the reference density

$$\pi^\star(a \mid s) = \frac{\pi_0(a \mid s)\exp(Q^\star(s,a)/\alpha)}{\sum_{a'\in\mathcal{A}} \pi_0(a' \mid s)\exp(Q^\star(s,a')/\alpha)} \quad .$$

Maximum Entropy RL. In this paper, we work with the maximum entropy (MaxEnt) (Ziebart et al., 2008; Haarnoja et al., 2018a) framework as an instantiation of KL-regularized RL (Levine, 2018), obtained by choosing a state-independent base density π_0 on $\mathcal{A}$, so that the KL penalty differs from the entropy penalty $-H(\pi(\cdot \mid s))$ only by an additive constant. The resulting objective is

$$\pi^\star_{\text{ME}} = \arg\max_\pi \; \mathbb{E}_{\tau\sim\pi}\Big[\sum_{t=0}^{\infty} \gamma^t \big(r(s_t,a_t) + \alpha\, H(\pi(\cdot \mid s_t))\big)\Big],$$

where $H(\pi(\cdot \mid s)) = -\mathbb{E}_{a\sim\pi(\cdot|s)}\big[\log \pi(a \mid s)\big]$ is the differential entropy, and the optimal policy has the Boltzmann form $\pi^\star(a \mid s) \propto \exp(Q^\star(s,a)/\alpha)$. Informally, larger α yields a more uniform action distribution, while smaller α concentrates probability on high-value actions. When α is fixed to a specific value C, we will simply denote it α_C. More traditionally, α is tuned such that the action distribution approaches a fixed target entropy H_{target} (Haarnoja et al., 2018a) (often set to the negative of the dimension of the action space) by continually fitting α to find

$$\alpha_{\text{TE}} \in \arg\min_{\alpha\in\mathbb{R}_{>0}} \mathbb{E}_{s\sim\mathcal{D},\, a\sim\pi(\cdot|s)}\Big[\alpha\big(-\log\pi(a \mid s) - H_{\text{target}}\big)\Big]. \tag{1}$$

3 Behavior-consistent RL

Moving beyond the standard RL problem—and towards empirical algorithms that make reward tuning, experimental replication, and debugging more reliable—we study differences between independent executions of the same algorithm in a fixed MDP, which we refer to as *runs*. To capture behavioral similarity between the policies produced by these runs, we introduce the setting of *behavior-consistent RL* (BRL). In BRL, the objective is not only to maximize return but also to obtain policies that are distributionally similar across runs, so that an algorithm is both performant and produces policies that exhibit the same behavior.

A common way to measure the similarity between the two probability distributions—or, in our case, policies—is via a divergence D. Consider the disagreement between any two policies $\mathbb{E}_{s\sim\mu_{\text{ref}}}\left[D\left(\pi^{(i)}(\cdot \mid s), \pi^{(j)}(\cdot \mid s)\right)\right]$, where μ_{ref} is a reference distribution. To turn agreement into a measure of behavioral stability, one can tie the reference distribution μ_{ref} directly to the output of the algorithm. A sensible choice for it is the population state distribution induced by the learned policies themselves, $\mu_{\text{ref}} = \mathbb{E}_{\pi\sim\mathcal{A}}\left[d^\pi\right]$, where $\pi \sim \mathcal{A}$ denotes the policy produced by a random execution of algorithm $\mathcal{A}$, and d^π denotes the state occupancy distribution induced by executing π in the MDP. Then, to measure behavioral similarity, the following criterion can be defined.

Definition 3.1 (Behavior-consistency in RL)**.** Let $\mathcal{A}$ be an RL algorithm that outputs a policy $\pi \in \Pi$ when trained on a fixed MDP. Let $\pi \sim \mathcal{A}$ denote the random policy obtained from one execution of $\mathcal{A}$, and let d^π denote the state occupancy distribution induced by executing π in the MDP. Define the behavioral reference distribution as $\mu_\pi \doteq \mathbb{E}_{\pi\sim\mathcal{A}}\left[d^\pi\right]$. Let $D : \Delta(\mathcal{A}) \times \Delta(\mathcal{A}) \to \mathbb{R}_{\geq 0}$ be a non-negative divergence between action distributions. Let $\pi^{(i)}, \pi^{(j)} \sim \mathcal{A}$ denote two policies produced by independent executions of $\mathcal{A}$. The behavioral consistency of $\mathcal{A}$ is defined as

$$\mathcal{V}(\mathcal{A}) \doteq \mathbb{E}_{\pi^{(i)},\pi^{(j)}\sim\mathcal{A}}\left[\mathbb{E}_{s\sim\mu_\pi}\left[D\left(\pi^{(i)}(\cdot \mid s), \pi^{(j)}(\cdot \mid s)\right)\right]\right]. \tag{2}$$

This notion captures the requirement that all policies produced by the same algorithm be close only on the space said policies will visit. This is sufficient to attain that lower inter-run variability means that different runs produce more behaviorally consistent policies. We note two things. First, the measure is directly dependent on the policies produced by the algorithm and thus two algorithms might measure similarity on different distributions. Second, behavior-consistency should always be tied to a secondary criterion such as return maximization as otherwise the trivial solution of outputting a constant policy will minimize the requirement.

4 High-temperature maximum entropy RL

If the policy of another run were available, we could directly constrain the updates in our current run toward it using KL-constrained RL and achieve behavioral similarity. Yet, in practice, the policies obtained from other runs may not be available, either because obtaining them might require additional expensive training or because they are produced independently by other users. Instead, we use a fixed, common reference policy π_0 as an independent anchor toward which all runs can be regularized. If each policy remains close to this shared reference, then behavior consistency can be controlled without access to other runs' policies; all policies remain close to the reference and therefore to each other. The MaxEnt RL framework offers a natural common prior, namely the uniform policy.

We now prove that the Boltzmann temperature, and therefore the strength of the regularization, directly controls the difference between policies. With an appropriate temperature, the soft Bellman operator produces policies across runs that never differ by more than some constant κ even if their Q-values are not initialized identically. Intuitively, the only way two Boltzmann policies can substantially differ at a fixed state is if the corresponding Q-vectors differ relative to the temperature. When this happens, increasing the entropy regularization reduces the softmax sensitivity to pointwise Q-value discrepancies, preventing the resulting action probabilities from separating too sharply. Setting $\alpha(s)$ proportional to the maximum pointwise disagreement makes these discrepancies small in temperature units, which keeps the two policies close in KL. The following theorem formalizes this.

Theorem 4.1 (Pairwise KL control via disagreement-scaled temperature)**.** *Assume $\mathcal{A}$ is finite and fix a state $s \in \mathcal{S}$. Let $Q^{(1)}(s,\cdot), Q^{(2)}(s,\cdot) \in \mathbb{R}^{|\mathcal{A}|}$ be two action-value vectors, and fix $\kappa > 0$ and $\alpha_{\min} > 0$. Define the shared temperature*

$$\alpha(s) \doteq \max\left\{\alpha_{\min}, \frac{\|Q^{(1)}(s,\cdot) - Q^{(2)}(s,\cdot)\|_\infty}{\kappa}\right\}. \tag{3}$$

Let $\pi^{(1)}(\cdot \mid s)$ and $\pi^{(2)}(\cdot \mid s)$ be the corresponding Boltzmann policies at temperature $\alpha(s)$. Then

$$D_{\mathrm{KL}}(\pi^{(1)}(\cdot \mid s) \| \pi^{(2)}(\cdot \mid s)) \leq 2\kappa \ .$$

Proof Sketch: (Full proof in Appendix A.1) The high-level idea is to write the KL of two Boltzmann policies as an average of the log-ratio between them. This log-ratio has two pieces: how much the two Q-values differ on the chosen action, and how much their normalizing constants differ. The first piece is immediately bounded by the maximum Q-disagreement. The second piece is bounded by observing that every term in one normalizer is within a fixed factor of the corresponding term in the other normalizer. So both pieces are controlled by the maximum disagreement divided by the temperature. Since the temperature was chosen appropriately, the total KL is bounded. □

Next, we show that using the disagreement-scaled temperature can lead to a behaviorally-consistent optimal entropy-regularized policy. In fact, we recover the optimal soft-value iteration result as the disagreement-dependent error vanishes with convergence.

Theorem 4.2 (Convergence under disagreement-scaled temperature)**.** *Assume $\alpha_{\min} > 0$. Let $Q^\star$ denote the optimal Q-function of the unregularized MDP. For a temperature $\alpha : \mathcal{S} \to \mathbb{R}_{>0}$, define*

$$(\mathcal{T}_\alpha Q)(s,a) \doteq r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[\alpha(s') \log \sum_{a' \in \mathcal{A}} \exp\left(\frac{Q(s',a')}{\alpha(s')} \right) \right].$$

Consider the coupled iterates $Q^{(i)}_{t+1} = \mathcal{T}_{\alpha_t} Q^{(i)}_t$, $i \in \{1,2\}$, where $\alpha_t(s)$ is the coupling temperature from equation 3. Let $\Delta_0 \doteq \|Q^{(1)}_0 - Q^{(2)}_0\|_\infty$ be the initial disagreement. Then, for $i \in \{1,2\}$ and $t \geq 0$,

$$\|Q^{(i)}_t - Q^\star\|_\infty \leq \gamma^t \|Q^{(i)}_0 - Q^\star\|_\infty + \frac{\gamma \alpha_{\min} \log |\mathcal{A}|}{1-\gamma}(1-\gamma^t) + \frac{\Delta_0 \log |\mathcal{A}|}{\kappa} t\gamma^t \ .$$

Proof Sketch: The proof (Appendix A.2) follows the standard approximate value-iteration template for the soft Bellman operator (Ziebart et al., 2008; Haarnoja et al., 2018a; Levine, 2018). The key step is to relate the approximation error to our adaptive temperature. At iteration t, the soft backup differs from the unregularized Bellman backup by at most a term proportional to $\alpha_{\min} + \|Q^{(1)}_t - Q^{(2)}_t\|_\infty / \kappa$. As the two coupled runs use the same temperature, their disagreement contracts as $\|Q^{(1)}_t - Q^{(2)}_t\|_\infty \leq \gamma^t \Delta_0$. Thus, the disagreement-dependent error vanishes after unrolling the recursion, leaving only the persistent entropy-regularization bias induced by the temperature floor $\alpha_{\min}$. □

Together, Theorems 4.1 and 4.2 show that the disagreement-scaled soft Bellman updates approach a neighborhood of the unregularized optimum whose radius is controlled by $\alpha_{\min}$, while the policies induced by the coupled Q-functions remain uniformly close in KL divergence at every iteration t.

5 Empirical consistency via Q-value expectile disagreement

Theorem 4.1 suggests that we can achieve similar policies across different runs by tuning the temperature of the Boltzmann policy. A high temperature should lead to increased consistency across runs. However, two crucial issues arise that we need to address. Theorem 4.2 relies on the difference in Q-values between different runs, but we do not have access to these in practice. Therefore, we need to develop a practical estimator for $\alpha(s)$ with the information available in a *single* run to achieve high consistency without collapsing to a trivial solution. In addition, high temperature introduces optimization difficulties in MaxEnt RL by amplifying off-policy extrapolation error. We discuss how this can be mitigated by leveraging solutions from the literature on critic overestimation.

5.1 An empirical proxy for setting $\alpha(s)$

We now introduce *Q-value Expectile Disagreement* (QED), which optimizes a proxy of Theorem 4.2 that can be plugged into any MaxEnt deep RL algorithm. To obtain a practical instantiation of a behavioral-consistency regularization, we require two components: a disagreement heuristic and a practical approximation of the $\max$ operator in Equation 3.

To address the fact that each run only knows its own Q-function, we exploit the double Q trick (Fujimoto et al., 2018), where two neural network critics, $Q_{\theta_1}(s,a)$ and $Q_{\theta_2}(s,a)$, are trained in parallel from shared data. We use double-critic disagreement as a single-run proxy for cross-run disagreement. The underlying hypothesis is that the two critics are representative of the critics that would arise in two independent runs. We provide empirical evidence for this hypothesis in Appendix E.1 by directly measuring the correlation between early double-critic disagreement and cross-run Q-function disagreement. If the two critics are representative, then using its disagreement to regulate policy improvement also regulates the disagreement between policies that independent runs would induce. In particular, when the critics disagree, the scaled temperature makes policy improvement sufficiently high-entropy that the resulting policies remain close. Since close policies induce similar data distributions and hence similar critic updates, this closeness persists throughout training.

Next, we need to approximate the $\max$ over all actions in Equation 3. Optimizing this maximum directly can induce an adversarial search over actions, as it highlights whichever action yields the largest critic discrepancy. This is undesirable, since Q-functions based on function approximation can be unreliable on unseen actions (Thrun & Schwartz, 1993; Fujimoto et al., 2019). Thus, the maximizer

may be driven by extrapolation error rather than meaningful uncertainty, which would unnecessarily inflate $\alpha(s)$ and inject high-variance updates. Instead, we compute an upper expectile of the double-critic difference under the current policy only. This yields a high-quantile summary of disagreement over actions the policy actually considers, capturing large but representative discrepancies without being dominated by a single worst-case action that lies far outside the support.

In summary, at state s, QED draws N actions $\{a_i\}_{i=1}^N$ from the current policy, $a_i \sim \pi(\cdot \mid s)$. It then evaluates both critics and aggregates the absolute differences with an expectile at level $\tau \in (0.5, 1)$:

$$\widetilde{\Delta}(s; Q_{\theta_1}, Q_{\theta_2}) = \text{Expectile}_\tau\Big(\big\{|Q_{\theta_1}(s, a_i) - Q_{\theta_2}(s, a_i)|\big\}_{i=1}^N \,\big|\, a_i \sim \pi(\cdot \mid s)\Big).$$

Then, we can implement Equation 3 with our proxy by setting the adaptive temperature to:

$$\alpha_{\text{QED}}(s) = \text{clip}\left(\frac{\widetilde{\Delta}(s; Q_{\theta_1}, Q_{\theta_2})}{kd}, \alpha_{\min}, \alpha_{\max}\right), \tag{4}$$

where $k > 0$ is an algorithm-dependent hyperparameter that controls how aggressively disagreement increases temperature, analogous to κ in Theorem 4.1, d is the action dimension, used to scale constraints across tasks, and $\alpha_{\min}, \alpha_{\max}$ ensure numerical stability. Note that, if the two critics agree closely at a state, $\alpha_{\text{QED}}(s)$ can become small, which can be problematic in sparse reward settings. Early in training, both critics may be near zero, causing entropy to collapse and exploration to vanish. To prevent this failure mode, we tune $\alpha_{\min} = \alpha_{\text{TE}}$ using the standard SAC temperature objective in Equation 1. $\alpha_{\max}$ is treated as a hyperparameter. With this, QED reduces to SAC with regular entropy tuning when the disagreement term vanishes or the divergence constraint is loose.

5.2 Off-policy instability

The derived α_{QED} setting from Equation 4 can lead to relatively large entropy penalties early in training where the two Q estimates differ substantially. The resulting increased entropy widens the policy's action support, which increases how often the critic is bootstrapped on actions that are weakly covered by the replay distribution. To show that this can be an issue, we consider a toy example.

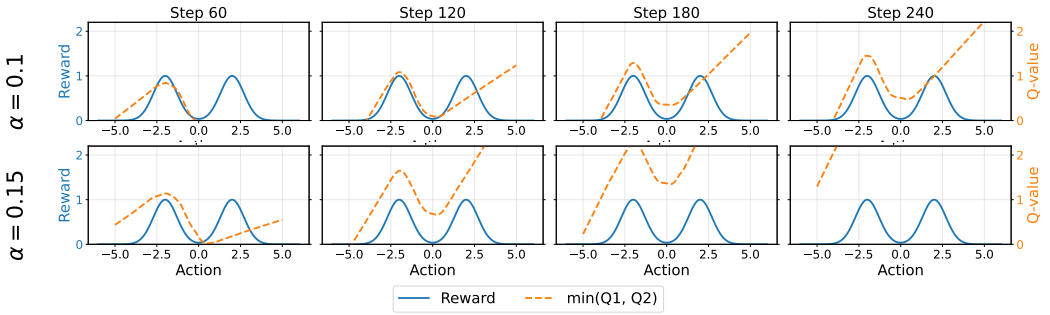
Toy example setup: To study how increased entropy affects the behavioral consistency of an empirical algorithm with function approximation, we study an MDP with a single state, continuous action space $\mathcal{A} = (-5, 5)$, and no transition dynamics. For any action a, the agent receives a reward $r(a)$ given by a bimodal Gaussian density, with modes centered at -2 and 2, each with variance 0.5. The problem has horizon $H = 2$ and discount factor $\gamma = 0.9$. For visualization, we use a standard SAC-style implementation with a unimodal Gaussian policy over the action interval and fixed α, allowing us to directly compare the learned Q-landscape and the induced policy as α varies.

High entropy training collapse: To showcase what happens during high-entropy training, we pre-fill the buffer of SAC with actions that lie only in $(-3, 0)$; thus, the right reward peak is outside of the data support. The agent is not allowed to collect new data to illustrate the effects of missing counterfactuals. We record both the critic networks' and the policy networks' predictions.

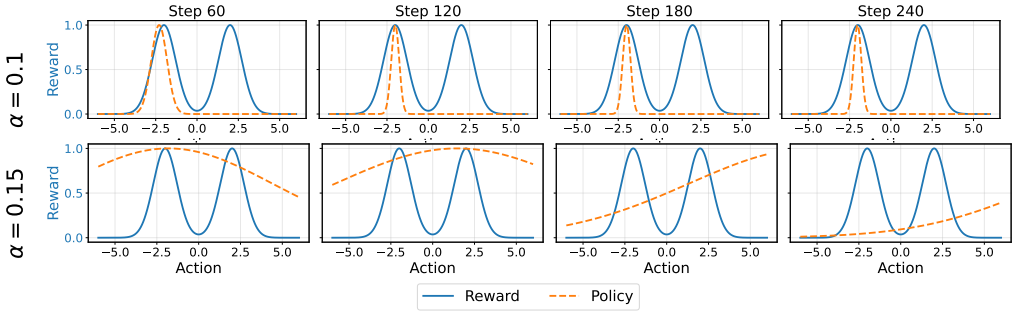
The results in Figure 2 demonstrate two key behaviors. First, even for small entropy coefficients ($\alpha = 0.1$), the Q-function overestimates values for out-of-distribution (OOD) actions. Still, the policy learns the correct behavior because its narrow distribution prevents it from sampling and evaluating those poorly estimated OOD regions during the actor update. In contrast, when we increase α to 0.15, the policy is forced to maximize a higher entropy bonus, resulting in an initially broad distribution that spans the entire action space (visible at Step 60). This wider sampling forces the critic to evaluate OOD actions where it lacks data, triggering the classical Q-value divergence spiral.

The policy then updates to exploit these hallucinated Q-values rather than the true reward. Notably, the policy does not collapse into a single sharp peak over the highest Q-values; instead, it drifts to the right. The agent attempts to shift its probability mass toward the runaway linear extrapolation of the Q-values, but the strong entropy penalty forces the distribution to remain wide. Ultimately, the agent completely abandons the true reward structure, resulting in functional policy collapse. This illustrates why behavioral consistency is not easily achieved by simply maximizing entropy; high entropy broadens action support, which can amplify extrapolation errors and destabilize learning.

Empirical solutions: The off-policy instability of high entropy RL suggests that we cannot naïvely apply our adaptive temperature scheduling to any double-critic RL algorithm and expect stable



(a) Q plots demonstrate that limited action coverage induces extrapolation error outside replay support.



(b) Policy plots show that, for $\alpha = 0.1$, the policy is able to consistently learn one of the modes. However, for the larger $\alpha = 0.15$, the policy learns erroneous Q values, pulling behavior away from the true reward modes.

Figure 2: **High entropy can amplify off-policy extrapolation error.** We consider the toy MDP and prefill the replay buffer with actions from only part of the action space, leaving one reward mode outside the data support. The learned Q-functions exhibit extrapolation error, and the policy accentuates this problem as it predicts value outside the support, particularly at larger α values.

training. To highlight this, we compare a traditional continuous control approach with an approach explicitly designed to handle distribution shifts. Concretely, we use:

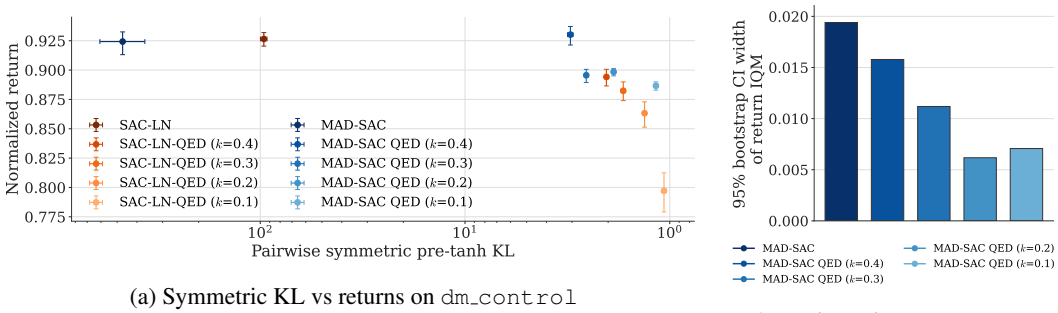
1. **Soft Actor-Critic with Layer Normalization (SAC-LN)**: The first algorithm we use is the traditional Soft Actor-Critic algorithm (Haarnoja et al., 2018a). Motivated by recent work on stabilizing critic bootstrapping in off-policy reinforcement learning (Hussing et al., 2024; Nauman et al., 2024a), we use layer normalization (Ba et al., 2016) in our SAC baseline. Layer normalization prohibits the Q-function from diverging when frequently queried on out-of-distribution data; however it does not fully remedy the out-of-distribution problem (Hussing et al., 2024).

2. **Model-Augmented Data Soft-Actor Critic (MAD-SAC)**: The second algorithm is a MaxEnt version of MAD-TD (Voelcker et al., 2025). MAD-TD builds on top of TD-MPC2 (Hansen et al., 2024) and augments critic training with model-generated data to explicitly address the off-policy bootstrapping problem. For stability, it also uses layer normalization, as well as HL-Gauss and a self-prediction loss (Li et al., 2023; Voelcker et al., 2024b; Fujimoto et al., 2025). For computational simplicity, we run the algorithm at a low update-to-data ratio and omit MPC.

It is important to stress that, while our experimental evaluation uses these approaches, our method is agnostic to the exact base algorithm and can be applied to a wide variety of other stable off-policy RL algorithms (Nauman et al., 2024b; Fujimoto et al., 2025; Palenicek et al., 2025; Lee et al., 2025).

6 Experiments

We conduct experimental evaluation of QED and its ability to increase behavioral similarity across runs using the 18 base tasks from the `dm_control` suite (Tunyasuvunakool et al., 2020). The MDPs are designed with a frame skip of 2, which is common on this benchmark (Hansen et al., 2024;



(a) Symmetric KL vs returns on `dm_control`

(b) Variance in `dm_control`

Figure 3: **QED reduces inter-run policy divergence while preserving returns.** (a) Final IQM of normalized return vs pairwise symmetric KL across independent training runs averaged over 18-tasks. Lower KL indicates more behaviorally consistent policies. Applying QED to both SAC-LN and MAD-SAC decreases pairwise KL by about two orders of magnitude, while retaining comparable normalized return. (b) Width of the 95% bootstrap CI of the evaluation IQM estimates under MAD-SAC. QED reduces return variance, suggesting that behavioral consistency implies stable returns.

Voelcker et al., 2025; Palenicek et al., 2026); we set $\gamma = 0.99$. All methods are run over 10 random seeds, and we report 95% bootstrapped confidence intervals (Agarwal et al., 2021).

We first evaluate whether QED improves behavioral consistency across runs. We then study its relationship with variance and the performance-consistency tradeoff controlled by k. Finally, we examine similarity in action choices and visual behavior. Appendix E.2 compares QED against fixed high-entropy baselines, while Appendix E.3 studies QED as a heuristic for encouraging exploration.

6.1 Main experiments

First, we evaluate how QED reduces behavioral differences by applying it to the two algorithms described above across values of k. We collect 20 evaluation rollouts with the final trained policies on all control tasks. Then, we report the cumulative return and measure similarity $\mathcal{V}$ using the symmetric KL divergence described in Appendix F. The results are shown in Figure 3a. For an easier comparison, we also report the variances in Figure 3b.

QED increases behavioral similarity: Our first observation is that, across the board, both baseline algorithms achieve high returns; however, the KL differences between runs are very large. This makes the setting a suitable candidate for evaluating our intervention. As we apply QED to both algorithms, we observe a drastic increase in behavioral similarity by two orders of magnitude and, with a well-chosen value of k, performance does not decrease. However, to get to very low KL divergences (close to a value of 1) requires small values of k, and as a result both algorithms see a reduction in return. However, MAD-SAC (which is made to deal with out-of-distribution prediction explicitly) Pareto dominates SAC as predicted and only incurs a reduction in performance of of roughly 5% at the lowest KL level while SAC loses more than 10% in performance. Appendix G shows that this is in part due to a predicted loss in sample efficiency as the algorithm needs to explore until it can reduce its Q-function uncertainty.

Relationship between variance and similarity: Using MAD-SAC, our results explicitly show that as k decreases, the return variance shrinks as well by up to 50%. This validates the hypothesis that behavioral similarity can be a proxy for return variance. Yet, we find that SAC itself does not exhibit this effect. This can be attributed to the fact that SAC is not designed to handle the bootstrapping problem in the same way. Runs that lose performance often do so because they become unstable, rather than because performance decreases consistently across seeds. In other words, the reduction in performance is driven by individual runs failing to learn correctly. MAD-SAC was designed to mitigate this failure mode, which is why its variance shrinks gracefully to a fixed point as k decreases.

Ablating k: As we increase k in both algorithms, the trend is to move back toward the original performance of the base algorithms. This is again to be expected as our approach directly generalizes traditional entropy tuning. On the other hand, as we decrease k, the KL-divergence decreases,

mapping out a Pareto frontier induced by the performance-vs-similarity trade-off. As the parameter k controls this trade-off, we recommend starting with the largest k that meaningfully reduces divergence relative to the base algorithm and lower it when consistency is prioritized over sample efficiency. In our experiments, $k \in [0.1, 0.4]$ was stable for MAD-SAC, while SAC starts to slowly deteriorate in that range. Thus, k is algorithm- and domain-dependent, and less-stable algorithms may need weaker constraints to avoid amplifying critic error.

6.2 Evaluation of behavior rollout effects

Next, we are interested in ablating the behavioral changes that we can observe due to QED to verify that we are obtaining more consistent long-term behavior and not just pointwise improvements. To test this hypothesis, we run a quantitative and a qualitative analysis of behavior rollouts.

Action distance: We take the trained MAD-SAC policies and roll them out for 20 evaluation trajectories of 100 steps. In each task, we compute the pairwise (L_2) action distance between all policies trained on that task at every step. We report the cumulative distance over all steps and take the mean over all environments. Due to this evaluation's structure, it is difficult to construct a sensible measure of variance; we omit it, as the sample size is very large and confidence intervals would be tight.

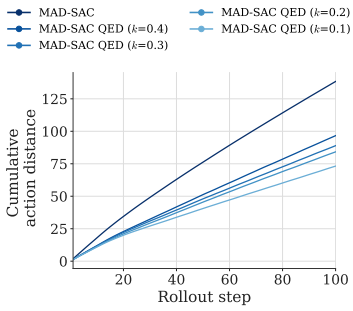


Figure 4: **QED produces more consistent rollout-level behavior across independently trained policies.** Measuring pairwise L_2 action distances across policies at each step, we find that QED reduces cumulative action distance.

Figure 4 shows that QED produces a large drop in action distance, providing quantitative evidence that lower KL is not merely due to higher-variance action distributions but rather because the *distribution means* are closer in aggregate. Increasing the behavioral constraint by decreasing k is correlated with a decrease in action distance. This provides quantitative evidence that QED is learning policies that behave more similarly at test time.

Visual similarity: Next, we examine qualitative rollouts of our policies. We choose the `cheetah_run` environment where both policies perform well and we see a reasonably low KL divergence using QED (see Appendix G). We initialize environments at a fixed state. Then, we execute policies from both the target-entropy version of MAD-SAC and the QED version. We visualize the behavior of each agent as it walks for 25 steps in Figure 1.

The policies trained with target entropy tuning learn behaviors that appear qualitatively distinct. One starts with a small hop, another initially walks, and a third begins with a medium-sized hop before transitioning into a larger jump. These are three distinct behaviors obtained by regular training. In contrast, all policies trained with QED exhibit consistent behavioral pattern that immediately starts the task with a wide jump. This illustrates that QED improves behavioral similarity not only quantitatively but visibly, a key requirement for making it useful for reward function tuning.

7 Conclusion

As RL moves toward larger models, longer training horizons, and real-world deployment, variability across runs becomes more than an evaluation nuisance: it limits our ability to predict, debug, and systematically improve learning systems. Behavior-consistent RL offers a framework to address this problem, and we show in this paper that enforcing consistent policies is both feasible and beneficial. Our results suggest that behavioral consistency may be a missing ingredient in scalable and reliable RL. If independent runs of the same RL algorithm can converge to different behavior, then evaluation, reward design, safety validation, and scaling-law estimation of performance all become more difficult.

Our results also expose new research questions. Many tasks admit multiple valid behavior strategies, raising the question of when diversity across runs should be preserved versus eliminated. Similarly, the tradeoff between consistency and sample complexity suggests deeper connections between uncertainty, replicability, and scalable policy optimization. Controlling behavioral variability opens a new direction for RL research, enabling the design of algorithms whose results are not only performant but also stable, predictable, and scientifically reproducible.

AI USE STATEMENT

In this work, we used generative AI tools for assisting in the writing of proofs, implementing methods. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, proposing or refining hypothesis, designing or providing feedback on research methodology or experiments, or interpreting results and generating synthetic datasets, assisting with translation, cleaning or reformatting datasets, supporting qualitative or thematic data analysis are not applicable to this work. Additionally, we used generative AI tools for create or edit software code, summarizing or analyzing existing literature, identifying relevant literature and sourcing information, editing a research paper to improve readability. We have reviewed all AI-assisted work. For coding, AI was used largely for code completion and all code was seen by a human and is humanly verified. For proof writing, AI was used to generate latex code in some cases. However, all claims were formulated by humans and derived from human intuition. All mathematical claims are humanly verified. During writing, AI was used to make text edits, substitute words, or shorten text. All text was actively judged by human writers. In the process of paper writing, we also asked AI for completeness of the already comprehensive related work that was put together by humans. AI suggested a few key papers that were then read by humans and only some of the papers were included as relevant literature. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

For theoretical results, all theorems provide intuition text, brief proof sketches, as well as full proofs. The work was largely developed on the foundations of open existing code and benchmarks. The experiments use publicly available `dm_control` environments described in Section 6. The code builds on the original publicly available MAD-TD codebase by Voelcker et al. (2025). For reproducibility, the main hyperparameters are documented in Appendix C. We will make our own version publicly available upon acceptance.

ACKNOWLEDGMENTS

Omitted for blind review.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, 2021.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

Philip J. Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, 2023.

Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Aditya Bhatt, Daniel Palenicek, Boris Belousov, Max Argus, Artemij Amiranashvili, Thomas Brox, and Jan Peters. Cross$q$: Batch normalization in deep reinforcement learning for greater sample efficiency and simplicity. In *The Twelfth International Conference on Learning Representations*, 2024.

Johan Bjorck, Carla P Gomes, and Kilian Q Weinberger. Is high variance unavoidable in RL? a case study in continuous control. In *International Conference on Learning Representations*, 2022.

Serena Booth, W. Bradley Knox, Julie Shah, Scott Niekum, Peter Stone, and Alessandro Allievi. The perils of trial-and-error reward design: Misdesign through overfitting and invalid task specifications. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37, 2023.

Emanuele Cavenaghi, Gabriele Sottocornola, Fabio Stella, and Markus Zanker. A systematic study on reproducibility of reinforcement learning in recommendation systems. *ACM Trans. Recomm. Syst.*, 2023.

Stephanie Chan, Sam Fishman, John Canny, Anoop Korattikara, and Sergio Guadarrama. Measuring the reliability of reinforcement learning algorithms. In *International Conference on Learning Representations*, 2020.

Xinyue Chen, Che Wang, Zijian Zhou, and Keith W. Ross. Randomized ensembled double q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021.

Richard Cheng, Abhinav Verma, Gabor Orosz, Swarat Chaudhuri, Yisong Yue, and Joel Burdick. Control regularization for reduced variance reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

Kamil Ciosek, Quan Vuong, Robert Loftin, and Katja Hofmann. Better exploration with optimistic actor critic. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Kaleigh Clary, Emma Tosch, John Foley, and David Jensen. Let's Play Again: Variability of Deep Reinforcement Learning Agents in Atari Environments. In *Critiquing and Correcting Trends in Machine Learning Workshop at Neural Information Processing Systems*, 2018.

Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. How many random seeds? statistical power analysis in deep reinforcement learning experiments. *arXiv preprint arXiv:1806.08295*, 2018.

Pierluca D'Oro, Max Schwarzer, Evgenii Nikishin, Pierre-Luc Bacon, Marc G Bellemare, and Aaron Courville. Sample-efficient reinforcement learning by breaking the replay ratio barrier. In *International Conference on Learning Representations*, 2023.

Eric Eaton, Marcel Hussing, Michael Kearns, and Jessica Sorrell. Replicable reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Eric Eaton, Marcel Hussing, Michael Kearns, Aaron Roth, Sikata Bela Sengupta, and Jessica Sorrell. Replicable reinforcement learning with linear function approximation. In *The Fourteenth International Conference on Learning Representations*, 2026.

Jesse Farebrother, Jordi Orbay, Quan Vuong, Adrien Ali Taiga, 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 *International Conference on Machine Learning*, 2024.

Manon Flageat, Bryan Lim, and Antoine Cully. Beyond expected return: Accounting for policy reproducibility when evaluating reinforcement learning algorithms. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Roy Fox, Ari Pakman, and Naftali Tishby. Taming the noise in reinforcement learning via soft updates. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pp. 202–211, Arlington, Virginia, USA, 2016. AUAI Press. ISBN 9780996643115.

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.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pp. 2052–2062, 2019.

Scott Fujimoto, Pierluca D'Oro, Amy Zhang, Yuandong Tian, and Michael Rabbat. Towards general-purpose model-free reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized Markov decision processes. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 2160–2169, 2019.

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*, 2017.

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 (ICML-18)*, pp. 1861–1870, 2018a.

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.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *The Twelfth International Conference on Learning Representations*, 2024.

Hado van Hasselt. Double q-learning. In *Advances in Neural Information Processing Systems*, 2010.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pp. 2094–2100. AAAI Press, 2016.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, New Orleans, Louisiana, USA, 2018. AAAI Press.

Takuya Hiraoka, Takahisa Imagawa, Taisei Hashimoto, Takashi Onishi, and Yoshimasa Tsuruoka. Dropout q-functions for doubly efficient reinforcement learning. In *International Conference on Learning Representations*, 2022.

Max Hopkins, Sihan Liu, Christopher Ye, and Yuichi Yoshida. From generative to episodic: Sample-efficient replicable reinforcement learning. *arXiv preprint arXiv:2507.11926*, 2025.

Marcel Hussing, Claas A Voelcker, Igor Gilitschenski, Amir-massoud Farahmand, and Eric Eaton. Dissecting deep rl with high update ratios: Combatting value divergence. In *Reinforcement Learning Conference*, 2024.

Russell Impagliazzo, Rex Lei, Toniann Pitassi, and Jessica Sorrell. Reproducibility in learning. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, 2022.

Riashat Islam, Peter Henderson, Maziar Gomrokchi, and Doina Precup. Reproducibility of benchmarked deep reinforcement learning tasks for continuous control. In *Reproducibility in Machine Learning Workshop (ICML)*, 2017.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. In *2019 International Conference on Robotics and Automation (ICRA)*, 2019.

Scott Jordan, Adam White, Bruno Castro da Silva, Martha White, and Philip S. Thomas. Position: Benchmarking is limited in reinforcement learning research. In *Forty-first International Conference on Machine Learning*, 2024.

Amin Karbasi, Grigoris Velegkas, Lin F. Yang, and Felix Zhou. Replicability in reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Woojun Kim, Yongjae Shin, Jongeui Park, and Youngchul Sung. Sample-efficient and safe deep reinforcement learning via reset deep ensemble agents. In *Advances in Neural Information Processing Systems*, 2023.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Hojoon Lee, Youngdo Lee, Takuma Seno, Donghu Kim, Peter Stone, and Jaegul Choo. Hyperspherical normalization for scalable deep reinforcement learning. In *Forty-second International Conference on Machine Learning*, 2025.

Kimin Lee, Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning. In *International Conference on Machine Learning*. PMLR, 2021.

Felix Leibfried and Jordi Grau-Moya. Mutual-information regularization in markov decision processes and actor-critic learning. In *Proceedings of the Conference on Robot Learning*, 2019.

Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Qiyang Li, Aviral Kumar, Ilya Kostrikov, and Sergey Levine. Efficient deep reinforcement learning requires regulating overfitting. In *International Conference on Learning Representations*, 2023.

Zhuang Liu, Xuanlin Li, Bingyi Kang, and Trevor Darrell. Regularization matters in policy optimization - an empirical study on continuous control. In *International Conference on Learning Representations*, 2021.

Clare Lyle, Zeyu Zheng, Khimya Khetarpal, Hado van Hasselt, Razvan Pascanu, James Martens, and Will Dabney. Disentangling the causes of plasticity loss in neural networks. In *Proceedings of The 3rd Conference on Lifelong Learning Agents*, 2025.

Nicolai A. Lynnerup, Laura Nolling, Rasmus Hasle, and John Hallam. A survey on reproducibility by evaluating deep reinforcement learning algorithms on real-world robots. In *Proceedings of the Conference on Robot Learning*, 2020.

Ted Moskovitz, Jack Parker-Holder, Aldo Pacchiano, Michael Arbel, and Michael Jordan. Tactical optimism and pessimism for deep reinforcement learning. *Advances in Neural Information Processing Systems*, 2021.

Michal Nauman, Michał Bortkiewicz, Piotr Miłoś, Tomasz Trzcinski, Mateusz Ostaszewski, and Marek Cygan. Overestimation, overfitting, and plasticity in actor-critic: the bitter lesson of reinforcement learning. In *Forty-first International Conference on Machine Learning*, 2024a.

Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, regularized, optimistic: scaling for compute and sample-efficient continuous control. *Advances in Neural Information Processing Systems*, 2024b.

Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning. In *International Conference on Machine Learning*, 2022.

Brendan O'Donoghue, Ian Osband, Remi Munos, and Vlad Mnih. The uncertainty Bellman equation and exploration. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

Ian Osband, Benjamin Van Roy, Daniel Russo, and Zheng Wen. Deep exploration via randomized value functions, 2019. URL `https://arxiv.org/abs/1703.07608`.

Daniel Palenicek, Florian Vogt, Joe Watson, and Jan Peters. Scaling off-policy reinforcement learning with batch and weight normalization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Daniel Palenicek, Florian Vogt, Joe Watson, Ingmar Posner, and Jan Peters. XQC: Well-conditioned optimization accelerates deep reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

Tudor-Andrei Paleu and Carlos Pascal. Reproducibility in deep reinforcement learning with maximum entropy. In *2023 27th International Conference on System Theory, Control and Computing (ICSTCC)*, 2023.

Jan Peters, Katharina Mülling, and Yasemin Altün. Relative entropy policy search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2010.

Doina Precup, Richard S. Sutton, and Sanjoy Dasgupta. Off-policy temporal difference learning with function approximation. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pp. 417–424, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1558607781.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., USA, 1st edition, 1994. ISBN 0471619779.

Carlo Romeo, Girolamo Macaluso, Alessandro Sestini, and Andrew D. Bagdanov. SPEQ: Offline stabilization phases for efficient q-learning in high update-to-data ratio reinforcement learning. In *Reinforcement Learning Conference*, 2025.

Tom Schaul, Andre Barreto, John Quan, and Georg Ostrovski. The phenomenon of policy churn. In *Advances in Neural Information Processing Systems*, 2022.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

Hongyao Tang and Glen Berseth. Improving deep reinforcement learning by reducing the chain effect of value and policy churn. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Sebastian Thrun and Anton Schwartz. Issues in using function approximation for reinforcement learning. In *Proceedings of the 1993 Connectionist Models Summer School*, 1993.

Emanuel Todorov. Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences*, 106(28):11478–11483, 2009.

John Tsitsiklis and Benjamin Van Roy. Analysis of temporal-diffference learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 9, 1996.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638.

Hado van Hasselt, Yotam Doron, Florian Strub, Matteo Hessel, Nicolas Sonnerat, and Joseph Modayil. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.

Claas A Voelcker, Marcel Hussing, and Eric Eaton. Can we hop in general? a discussion of benchmark selection and design using the hopper environment. In *Finding the Frame: An RLC Workshop for Examining Conceptual Frameworks*, 2024a.

Claas A Voelcker, Tyler Kastner, Igor Gilitschenski, and Amir-massoud Farahmand. When does self-prediction help? understanding auxiliary tasks in reinforcement learning. In *Reinforcement Learning Conference*, 2024b.

Claas A Voelcker, Marcel Hussing, Eric Eaton, Amir massoud Farahmand, and Igor Gilitschenski. MAD-TD: Model-augmented data stabilizes high update ratio RL. In *The Thirteenth International Conference on Learning Representations*, 2025.

Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the National Conference on Artificial Intelligence*, 2008.

A PROOFS

We provide the proofs for the two theoretical results, Theorems 4.1 and 4.2, that motivate QED. We prove that, for two Boltzmann policies induced by different Q-functions at a fixed state, setting the shared temperature proportional to their pointwise Q-disagreement bounds the KL divergence between the resulting policies. We then prove that using this disagreement-scaled temperature inside soft-value iteration still approaches a neighborhood of the unregularized optimum: the only persistent error comes from the entropy floor $\alpha_{\min}$, while the disagreement-dependent term vanishes over time. Together, these arguments formalize the principle behind QED: entropy should be high when value estimates disagree, but should shrink as those disagreements contract.

A.1 PROOF OF THEOREM 4.1

Theorem (Pairwise KL control via disagreement-scaled temperature)**.** *Assume $\mathcal{A}$ is finite and fix a state $s \in \mathcal{S}$. Let $Q^{(1)}(s,\cdot), Q^{(2)}(s,\cdot) \in \mathbb{R}^{|\mathcal{A}|}$ be two action-value vectors, and fix $\kappa > 0$ and $\alpha_{\min} > 0$. Define the shared temperature*

$$\alpha(s) \doteq \max\left\{\alpha_{\min}, \frac{\|Q^{(1)}(s,\cdot) - Q^{(2)}(s,\cdot)\|_\infty}{\kappa}\right\}. \tag{5}$$

Let $\pi^{(1)}(\cdot \mid s)$ and $\pi^{(2)}(\cdot \mid s)$ be the Boltzmann policies at temperature $\alpha(s)$. Then

$$D_{\mathrm{KL}}(\pi^{(1)}(\cdot \mid s) \| \pi^{(2)}(\cdot \mid s)) \le 2\kappa.$$

Proof. Fix some $s \in S$. Let $Z^{(i)} \doteq \sum_{b\in\mathcal{A}} \exp(Q^{(i)}(s,b)/\alpha(s))$, $i \in \{1,2\}$. For any action a, we can write out the log ratio between our two policies as

$$\log \frac{\pi^{(1)}(a \mid s)}{\pi^{(2)}(a \mid s)} = \frac{Q^{(1)}(s,a) - Q^{(2)}(s,a)}{\alpha(s)} + \log \frac{Z^{(2)}}{Z^{(1)}}.$$

Taking expectation with respect to $a \sim \pi^{(1)}(\cdot \mid s)$ gives the KL divergence

$$D_{\mathrm{KL}}(\pi^{(1)} \| \pi^{(2)}) = \frac{\mathbb{E}_{a\sim\pi^{(1)}}[Q^{(1)}(s,a) - Q^{(2)}(s,a)]}{\alpha(s)} + \log \frac{Z^{(2)}}{Z^{(1)}}.$$

We bound these two terms independently. For the first term, it is easy to see that

$$\frac{\mathbb{E}_{a\sim\pi^{(1)}}[Q^{(1)}(s,a) - Q^{(2)}(s,a)]}{\alpha(s)} \le \frac{\mathbb{E}_{a\sim\pi^{(1)}}[|Q^{(1)}(s,a) - Q^{(2)}(s,a)|]}{\alpha(s)} \le \frac{\|Q^{(1)}(s,\cdot) - Q^{(2)}(s,\cdot)\|_\infty}{\alpha(s)}.$$

For the second term, for every $b \in \mathcal{A}$,

$$\begin{aligned}\exp(Q^{(2)}(s,b)/\alpha(s)) &= \exp((Q^{(1)}(s,b) - (Q^{(1)}(s,b) - Q^{(2)}(s,b)))/\alpha(s)) \\ &\le \exp(Q^{(1)}(s,b)/\alpha(s)) \exp(\|Q^{(1)}(s,\cdot) - Q^{(2)}(s,\cdot)\|_\infty/\alpha(s)).\end{aligned}$$

Summing over b yields $Z^{(2)} \le Z^{(1)} \exp(\|Q^{(1)}(s,\cdot) - Q^{(2)}(s,\cdot)\|_\infty/\alpha(s))$, and therefore

$$\log \frac{Z^{(2)}}{Z^{(1)}} \le \frac{\|Q^{(1)}(s,\cdot) - Q^{(2)}(s,\cdot)\|_\infty}{\alpha(s)}.$$

Combining the bounds,

$$D_{\mathrm{KL}}(\pi^{(1)} \| \pi^{(2)}) \le \frac{2\|Q^{(1)}(s,\cdot) - Q^{(2)}(s,\cdot)\|_\infty}{\alpha(s)}.$$

The result follows from the construction of $\alpha(s)$. □

Symmetric KL. Although Theorem 4.1 is stated for directed KL, the same bound immediately applies to the symmetric KL used in our empirical metric. Indeed, applying the theorem once to $D_{\mathrm{KL}}(\pi^{(1)}\|\pi^{(2)})$ and once with the two Q-functions reversed gives

$$D_{\mathrm{KL}}(\pi^{(1)}\|\pi^{(2)}) \le 2\kappa, \qquad D_{\mathrm{KL}}(\pi^{(2)}\|\pi^{(1)}) \le 2\kappa.$$

Therefore,

$$\frac{1}{2}\left[D_{\mathrm{KL}}(\pi^{(1)}\|\pi^{(2)}) + D_{\mathrm{KL}}(\pi^{(2)}\|\pi^{(1)})\right] \le 2\kappa.$$

Thus the idealized Boltzmann analysis controls the same symmetrized divergence form used in the experiments.

A.2 Proof of Theorem 4.2

Theorem (Convergence under disagreement-scaled temperature)**.** *Assume $\alpha_{\min} > 0$. Let $Q^\star$ denote the optimal Q-function of the unregularized MDP. For a temperature $\alpha : \mathcal{S} \to \mathbb{R}_{>0}$, define*

$$(\mathcal{T}_\alpha Q)(s,a) \doteq r(s,a) + \gamma \mathbb{E}_{s'\sim P(\cdot|s,a)}\left[\alpha(s')\log \sum_{a'\in\mathcal{A}} \exp\left(\frac{Q(s',a')}{\alpha(s')}\right)\right].$$

Consider the coupled iterates $Q_{t+1}^{(i)} = \mathcal{T}_{\alpha_t} Q_t^{(i)}$, $i \in \{1,2\}$, where $\alpha_t(s)$ is the shared temperature from equation 3. Let $\Delta_0 \doteq \|Q_0^{(1)} - Q_0^{(2)}\|_\infty$ be the initial disagreement. Then, for $i \in \{1,2\}$ and $t \ge 0$,

$$\|Q_t^{(i)} - Q^\star\|_\infty \le \gamma^t \|Q_0^{(i)} - Q^\star\|_\infty + \frac{\gamma \alpha_{\min} \log|\mathcal{A}|}{1-\gamma}(1-\gamma^t) + \frac{\Delta_0 \log|\mathcal{A}|}{\kappa}\, t\gamma^t.$$

Proof. Let $\mathcal{T}$ denote the unregularized Bellman optimality operator and let $e_t^{(i)} \doteq \|Q_t^{(i)} - Q^\star\|_\infty$. We use two standard facts about soft Bellman backups from maximum-entropy value iteration (Ziebart et al., 2008; Haarnoja et al., 2018a; Levine, 2018). First, for any fixed positive temperature α, $\mathcal{T}_\alpha$ is a γ-contraction in Q. Second, the soft value is a one-sided approximation to the hard maximum:

$$0 \le (\mathcal{T}_\alpha Q)(s,a) - (\mathcal{T}Q)(s,a) \le \gamma \log|\mathcal{A}|\, \mathbb{E}_{s'\sim P(\cdot|s,a)}[\alpha(s')]. \tag{6}$$

Hence,

$$\|\mathcal{T}_{\alpha_t} Q - \mathcal{T}Q\|_\infty \le \gamma \log|\mathcal{A}|\, \|\alpha_t\|_\infty. \tag{7}$$

The schedule-specific step is to bound $\|\alpha_t\|_\infty$. Since both runs use the same α_t, the standard contraction argument applies conditionally on α_t:

$$\|Q_{t+1}^{(1)} - Q_{t+1}^{(2)}\|_\infty = \|\mathcal{T}_{\alpha_t} Q_t^{(1)} - \mathcal{T}_{\alpha_t} Q_t^{(2)}\|_\infty \le \gamma \|Q_t^{(1)} - Q_t^{(2)}\|_\infty. \tag{8}$$

Iterating gives

$$\|Q_t^{(1)} - Q_t^{(2)}\|_\infty \le \gamma^t \Delta_0. \tag{9}$$

Therefore, by the disagreement-scaled definition of α_t,

$$\|\alpha_t\|_\infty \le \alpha_{\min} + \frac{\|Q_t^{(1)} - Q_t^{(2)}\|_\infty}{\kappa} \le \alpha_{\min} + \frac{\gamma^t \Delta_0}{\kappa}. \tag{10}$$

Now compare each iterate to $Q^\star$. Since $\mathcal{T}Q^\star = Q^\star$ and $\mathcal{T}$ is a γ-contraction,

$$\begin{aligned} e_{t+1}^{(i)} &= \|\mathcal{T}_{\alpha_t} Q_t^{(i)} - \mathcal{T}Q^\star\|_\infty \\ &\le \|\mathcal{T}Q_t^{(i)} - \mathcal{T}Q^\star\|_\infty + \|\mathcal{T}_{\alpha_t} Q_t^{(i)} - \mathcal{T}Q_t^{(i)}\|_\infty \\ &\le \gamma e_t^{(i)} + \gamma \log|\mathcal{A}| \left(\alpha_{\min} + \frac{\gamma^t \Delta_0}{\kappa}\right). \end{aligned} \tag{11}$$

Unrolling this approximate-contraction recursion gives

$$\begin{aligned} e_t^{(i)} &\le \gamma^t e_0^{(i)} + \gamma \alpha_{\min} \log|\mathcal{A}| \sum_{j=0}^{t-1} \gamma^{t-1-j} + \frac{\gamma \Delta_0 \log|\mathcal{A}|}{\kappa} \sum_{j=0}^{t-1} \gamma^{t-1-j}\gamma^j \\ &= \gamma^t e_0^{(i)} + \frac{\gamma \alpha_{\min} \log|\mathcal{A}|}{1-\gamma}(1-\gamma^t) + \frac{\Delta_0 \log|\mathcal{A}|}{\kappa}\, t\gamma^t. \end{aligned} \tag{12}$$

Substituting back $e_t^{(i)} = \|Q_t^{(i)} - Q^\star\|_\infty$ proves the stated bound. □

B Related work

Reproducibility, replicability, and run-to-run variability in RL. Deep RL is well known to be sensitive to random seeds, implementation choices, environment stochasticity, and evaluation protocol (Islam et al., 2017; Henderson et al., 2018; Colas et al., 2018). This sensitivity makes empirical conclusions statistically fragile, especially when comparisons rely on small numbers of runs, motivating robust aggregate metrics such as the interquartile mean and bootstrap confidence intervals (Agarwal et al., 2021). At the same time, benchmarking alone can be limited as a scientific instrument for understanding the behavior learned by RL algorithms (Jordan et al., 2024).

A related theoretical line studies replicability (Impagliazzo et al., 2022), asking when repeated executions of a learning algorithm produce identical outputs. Recent work has begun to formalize this question for RL (Eaton et al., 2023; Karbasi et al., 2023; Hopkins et al., 2025; Eaton et al., 2026), but existing results primarily focus on tabular or linear-function-approximation settings. Closer to ours, some empirical work has studied the variability beyond aggregate return. Clary et al. (2018) report post-training performance variation among Atari agents. Chan et al. (2020) propose reliability metrics and tests for learned policies. Schaul et al. (2022) analyze policy churn during value-based deep RL training while Tang & Berseth (2024) propose methods to reduce policy churn in deep RL. In contrast, we directly target cross-run behavioral consistency in deep continuous-control RL.

Off-policy error, value divergence, and stabilizing deep actor-critic learning. Our analysis is also connected to the broader literature on instability in off-policy RL with function approximation. The classical "deadly triad" of bootstrapping, off-policy data, and function approximation has long been understood as a source of divergence (Thrun & Schwartz, 1993; Tsitsiklis & Van Roy, 1996; Precup et al., 2001; van Hasselt et al., 2018). One of the most popular interventions in modern actor-critic algorithms to deal with overestimation are double critics (Hasselt, 2010; Hasselt et al., 2016; Fujimoto et al., 2018). This idea has been pushed to the limit in ensemble critics that help mitigate divergence at the computational expense of having to query a large amount of neural networks (Chen et al., 2021; Hiraoka et al., 2022). Alternative interventions that stabilize learning include conservative value updates (Fujimoto et al., 2019; Kumar et al., 2020), categorical value function representations (Bellemare et al., 2017; Farebrother et al., 2024) and normalization (Ball et al., 2023; Bhatt et al., 2024; Hussing et al., 2024; Nauman et al., 2024a; Lyle et al., 2025). Recent work has shown that high update-to-data ratios can improve sample efficiency (Nikishin et al., 2022; D'Oro et al., 2023) but exacerbate value divergence (Hussing et al., 2024) unless stabilized by regularization (Palenicek et al., 2025), model-augmented data (Voelcker et al., 2025), architectural changes (Hussing et al., 2024; Fujimoto et al., 2025; Palenicek et al., 2026), or algorithmic interventions (Romeo et al., 2025). Our toy examples show that high entropy can interact with these same off-policy errors by widening policy support and forcing actor updates to query poorly supported regions of the critic. QED includes a built-in mechanism to avoid querying actions with spuriously high Q-value disagreement, but it is still most effective when paired with actor-critic methods that already control overestimation well.

Policy reguarlization in RL. MaxEnt RL augments reward maximization with an entropy bonus, producing stochastic policies that encourage exploration and avoid premature collapse to deterministic actions (Ziebart et al., 2008; Haarnoja et al., 2017; 2018a). This framework is closely connected to KL-regularized control and probabilistic inference views of RL, where policy improvement is regularized toward a prior or reference distribution (Todorov, 2009; Peters et al., 2010; Levine, 2018; Geist et al., 2019). Soft Actor-Critic instantiates this idea in deep off-policy actor-critic learning, using entropy regularization and automatic temperature tuning to target a desired policy entropy (Haarnoja et al., 2018a;b). Other forms of regularization, including mutual information (Leibfried & Grau-Moya, 2019), control priors (Johannink et al., 2019), and policy regularization (Fox et al., 2016; Liu et al., 2021), have been used to improve exploration, robustness, and optimization stability. Closest in spirit, Cheng et al. (2019) regularize policies toward a control-theoretic prior to reduce variance in reinforcement learning. In contrast, we reduce cross-run behavioral divergence by adapting the entropy temperature from critic disagreement, without requiring an external controller prior.

Uncertainty-driven exploration and critic disagreement. A large body of work uses uncertainty estimates to guide exploration or reduce value-estimation error, including ensembles, bootstrapping, optimistic objectives, and epistemic bonuses (O'Donoghue et al., 2018; Osband et al., 2019; Ciosek et al., 2019; Lee et al., 2021; Moskovitz et al., 2021; Kim et al., 2023). QED also uses critic disagreement, but for a different role: we treat disagreement as a local proxy for policy instability.

C HYPERPARAMETERS

All experiments use the 18 `dm_control` tasks listed in Table 1. We use the same task set, training budget, evaluation protocol, and random seeds for SAC-LN, MAD-SAC, and their QED variants. For our implementation, we use the original MAD-TD codebase. The only three hyperparameters that we change are: proportion real to 0.5 (for MAD-SAC), no actor layer normalization, and gradient clipping set to 20. The shared experimental settings are summarized in Table 2. All environments use frame skip 2 and discount factor $\gamma = 0.99$. Each method is trained with 10 independent random seeds for 2M environment steps. Final performance is computed from 20 evaluation rollouts per trained policy, and aggregate results are reported using IQM with 95% bootstrap confidence intervals.

Table 1: The 18 `dm_control` tasks used in the main experiments.

Tasks			
`acrobot_swingup`	`cartpole_balance`	`cartpole_swingup`	`cheetah_run`
`finger_spin`	`finger_turn_easy`	`finger_turn_hard`	`fish_swim`
`hopper_hop`	`hopper_stand`	`pendulum_swingup`	`quadruped_run`
`quadruped_walk`	`reacher_easy`	`reacher_hard`	`walker_run`
`walker_stand`	`walker_walk`		

Table 2: Core experimental settings used across the main `dm_control` experiments.

Setting	Value
Number of tasks	18
Number of seeds	10
Training budget	2M environment steps
Frame skip	2
Discount factor	$\gamma = 0.99$
Final evaluation	20 rollouts per trained policy
Aggregate return metric	IQM with 95% bootstrap CI
Policy-divergence metric	Pairwise symmetric pre-tanh KL
Rollout behavior metric	Cumulative pairwise ℓ_2 action distance
Action-distance rollout length	100 steps

For the baseline methods, we evaluate SAC-LN and MAD-SAC with standard target-entropy tuning. SAC-LN uses a SAC-style actor-critic with double critics and layer normalization. MAD-SAC uses the same maximum-entropy actor-critic structure, but augments critic training with model-generated data following the MAD-TD family of methods. For computational simplicity, MAD-SAC is evaluated without MPC at action-selection time.

For QED variants, we replace the scalar entropy temperature with the state-dependent disagreement-scaled temperature described in Section 5.1. The target-entropy temperature α_{TE} is retained as a learned temperature floor, so QED reduces to ordinary target-entropy tuning when the disagreement term is inactive. We compute disagreement using $N = 8$ actions sampled from the current policy and aggregate the double-critic differences with expectile level $\tau = 0.9$. The disagreement is normalized by the action dimension, and the maximum temperature is clipped at $\alpha_{\max} = 0.2$.

D COMPUTE RESOURCES

All experiments were run on a shared academic GPU cluster. Each training run used a single GPU and 8 CPU cores, with 80G of system memory requested per job. The main `dm_control` experiments consist of 18 tasks, 10 random seeds, and the method variants listed with all seeds run simultaneously on the same GPU and trained for 2M environment steps. Wall-clock time varied by task and method, with typical runs taking approximately `8–48` hours. The total reported experimental suite required approximately 4,000 GPU-hours, excluding preliminary debugging and failed runs.

For reproducibility, the reported experiments use publicly available simulated environments from `dm_control` and do not require specialized hardware beyond standard CUDA-enabled GPUs. The

QED variants add only a small amount of computation relative to their corresponding base algorithms: at each actor update, QED samples $N = 8$ actions from the current policy and evaluates both critics to compute the expectile disagreement statistic. This overhead is minor compared with the cost of the usual actor–critic updates and model-augmented critic training in MAD-SAC.

E Additional experiments

E.1 Ablating the double critic proxy

QED uses disagreement between two critics in a single training run as a practical proxy for the disagreement that would be observed between independently trained runs. In particular, we argue that if the disagreement between the two critics is representative of cross-run disagreement initially, the distribution from which we sample our data stays continuously close as we ensure our policies stay close. We evaluate this hypothesis directly by measuring whether early within-run critic disagreement predicts cross-seed Q-function disagreement across our `dm_control` 18 task suite.

For each task, we take the MAD-SAC runs (10 seeds) trained with standard target-entropy tuning and evaluate the critics over the early-training interval (first 15 rollouts). For each rollout, we compute the mean difference between the double critic over a sampled batch and compare it to the pairwise difference in mean Q-values of the first Q-function across all seeds. Figure 5 shows a strong positive relationship between the within-run double-critic disagreement and the cross-seed Q-function disagreement across tasks. This suggests that tasks where the two critics disagree (early in training) strongly correlates ($R^2 = 0.77$) with evaluation Q_1 dispersion after training, which supports our use of in-run Q-function disagreement as a practical proxy for cross-seed variability.

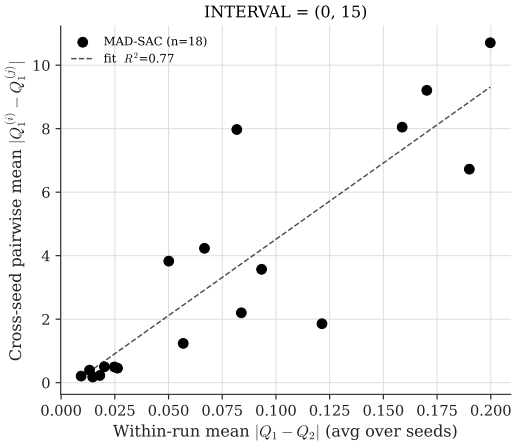


Figure 5: Early double-critic disagreement predicts cross-seed Q-function disagreement.

E.2 Ablating fixed α

The guarantees for policy similarity without considering return are naturally also achievable by simply choosing a relatively high static α. To illustrate the benefits of tying α directly to the Q-function disagreement, we compare MAD-SAC with QED to a variety of fixed alpha values. In particular, we measure the mean return and pairwise KL-divergence across policies. Here, we report the mean rather than the IQM used previously because the mean is more sensitive to the low-return runs induced by large fixed temperatures. The results are shown in Figure 6.

As expected, working with a large fixed α reduces the pairwise KL divergence between policies. However, this reduction comes with a marked decrease in mean return. In contrast, QED maintains relatively stable performance across the tested values of k, yielding a Pareto frontier that dominates that of fixed α. The decline under fixed temperatures is driven by an increasing number of low-return seeds, which lowers the mean and increases the variation across runs. At the lowest measured symmetric KL level, a fixed α incurs an approximately 7% lower mean performance than QED.

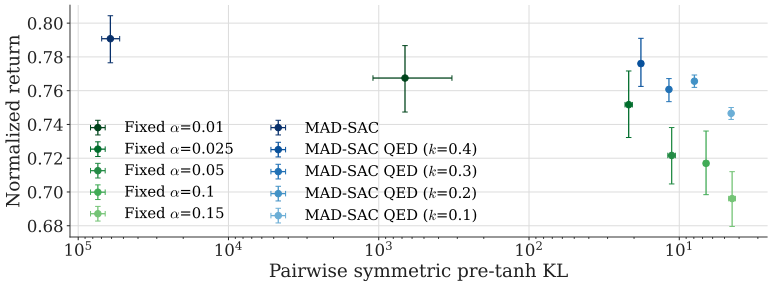


Figure 6: **Adaptively choosing α via QED leads to higher performance at the same KL-distance compared to static values of α.** Mean normalized return vs pairwise symmetric KL across independent runs on the 18-task `dm_control` suite comparing QED vs static values of α. Fixed values of α lead to decrease in pairwise KL but also yield strictly lower performance than using QED.

E.3 Tight KL control over high-variance tasks

Finally, to demonstrate the exploration benefits of QED, we highlight a challenging task in the `dm_control` suite: the `hopper_hop` task. Across various state-of-the-art algorithms, the return variance on this task is very high and reported mean performance ranges from 250-500 with confidence intervals often half as large as the mean returns over a few trials (D'Oro et al., 2023; Lee et al., 2025; Palenicek et al., 2026). This can be partially explained by the difficult initial state distribution of the task and the unstable dynamics of the single-legged hopper (Voelcker et al., 2024a). Both lead to strong dissimilarity in behavior across runs. Figure 7 shows the reward curves during training of this task. Relatedly, Hussing et al. (2024) argue that this task likely requires strong exploration.

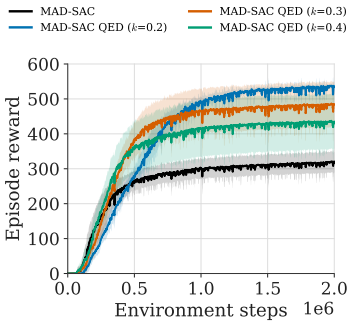


Figure 7: **QED reduces training-time variance and policy divergence on a high-variance control task.** Return over training steps. QED improves performance and reduces dispersion across seeds.

As expected, the base MAD-SAC algorithm has high variance during training but adding QED improves performance. A complementary interpretation of QED is that it increases policy entropy and thus exploration. An interesting phenomenon occurs when the QED constraint is strong enough at $k = 0.2$, as all runs converge to an identical performance. This suggests that QED shapes exploration, with stronger constraints aligning executions in policy space.

E.4 Limitations

A key limitation of QED is that it relies on pointwise Q-estimates and their agreement. In these experiments, we find that disagreement can often take a long time to vanish, and it is not currently clear when vanishing disagreement is an appropriate criterion in practice. Furthermore, while QED provides a consistent boost in behavioral similarity, higher-dimensional tasks reveal a more pronounced trade-off between performance and similarity. KL values can often remain above 1, which we find to be an important threshold for multi-step similarity. We hope that future work can improve these issues while ensuring consistent behavior over very long horizons.

F Policy-divergence evaluation protocol

Definition 3.1 defines inter-run variability $\mathcal{V}$ using a generic divergence D between policies evaluated under a fixed protocol. In our experiments, we instantiate D as the symmetric KL divergence between the pre-tanh Gaussian action distributions produced by the actor.

On-policy inter-run evaluation. For each task, method, and checkpoint, we construct an evaluation state set $\mathcal{S}_{\text{eval}}^{m,t}$ from evaluation rollouts generated by the 10 independently trained seeds of method m

at checkpoint t. We then evaluate all pairwise policy divergences among those 10 seeds on this shared within-method state set. For a fixed task, method, and checkpoint, every policy pair is compared on the same collection of states, and the resulting quantity measures variability across independent executions of the same learning algorithm.

This protocol is intentionally on-policy. Behavior-consistency concerns the stability of the behaviors that a learning algorithm actually produces under its own training and evaluation distribution. Evaluating each method on the states induced by its own independently trained policies measures the realized behavioral variability of that method: whether different random seeds lead to different action distributions in the regions of the state space that the method reaches at evaluation time.

Given two policies π_i and π_j from method m, let their pre-tanh Gaussian distributions at state s be

$$\tilde{\pi}_i(\cdot \mid s) = \mathcal{N}(\mu_i(s), \mathrm{diag}(\sigma_i^2(s))), \qquad \tilde{\pi}_j(\cdot \mid s) = \mathcal{N}(\mu_j(s), \mathrm{diag}(\sigma_j^2(s))).$$

We define the pairwise policy divergence as

$$D_m(\pi_i, \pi_j) = \frac{1}{|\mathcal{S}_{\mathrm{eval}}^{m,t}|} \sum_{s \in \mathcal{S}_{\mathrm{eval}}^{m,t}} \frac{1}{2} \left[D_{\mathrm{KL}}\big(\tilde{\pi}_i(\cdot \mid s) \,\|\, \tilde{\pi}_j(\cdot \mid s)\big) + D_{\mathrm{KL}}\big(\tilde{\pi}_j(\cdot \mid s) \,\|\, \tilde{\pi}_i(\cdot \mid s)\big) \right].$$

The inter-run variability of method m at checkpoint t is then

$$\mathcal{V}_m(t) = \frac{1}{I(I-1)} \sum_{i \neq j} D_m(\pi_i, \pi_j),$$

where $I = 10$ is the number of independent training seeds.

Why evaluate on each method's induced states? A fixed exogenous state set can be useful for pointwise policy comparison, but it does not directly measure the behavioral variability realized by an RL algorithm at deployment time. In continuous-control tasks, the states visited by a policy are part of the learned behavior: different gaits, balance strategies, or recovery maneuvers induce different state distributions. Since our goal is to measure whether independent runs of the same method produce consistent behaviors, we evaluate each method on the states reached by that method's own evaluation rollouts. This gives a method-level behavioral variability estimate that includes the regions of the state space actually occupied by the learned policies.

This choice also avoids evaluating policies primarily on states that are irrelevant or rarely visited under that method's learned behavior. For example, a policy may be highly variable on off-distribution states but consistent on the trajectory manifold it actually follows; conversely, a method may produce different trajectory manifolds across seeds, which will be reflected in the states collected from its own rollouts and in the resulting inter-run KL. The metric therefore captures realized behavioral consistency rather than pointwise discrepancy detached from the method's induced occupancy measure.

Cross-method interpretation. When comparing methods, $\mathcal{V}_m(t)$ should be interpreted as each method's on-policy inter-run variability: the average pairwise divergence among independent runs of that method on the states induced by that method. It does not require all methods to be evaluated on an identical exogenous state set, because the object of comparison is the variability of each learning procedure under its own induced evaluation distribution.

Relation to the Boltzmann-policy analysis. Theorem 4.1 and our discussion of symmetric KL show that, for exact Boltzmann policy improvement in finite action spaces, disagreement-scaled temperature controls both directed and symmetric KL between the induced policies. Our continuous-control experiments use tanh-squashed Gaussian actors, so the empirical divergence is computed between the actual actor distributions used by SAC-LN and MAD-SAC. We compute this KL in the pre-tanh Gaussian space for numerical stability. Thus, the empirical metric is the natural continuous-control analogue of the symmetrized policy divergence controlled in the idealized Boltzmann setting: both measure how sensitive the policy-improvement step is to run-specific value estimates, but they are instantiated for different policy classes.

G FULL EXPERIMENTAL RESULTS

We report the per-task reward and policy-divergence curves underlying the aggregate results in Section 6. Figures 8 and 9 show how QED affects learning performance across all 18 `dm_control` tasks for SAC-LN and MAD-SAC, respectively. Figures 10 and 11 show the corresponding inter-run policy-divergence curves.

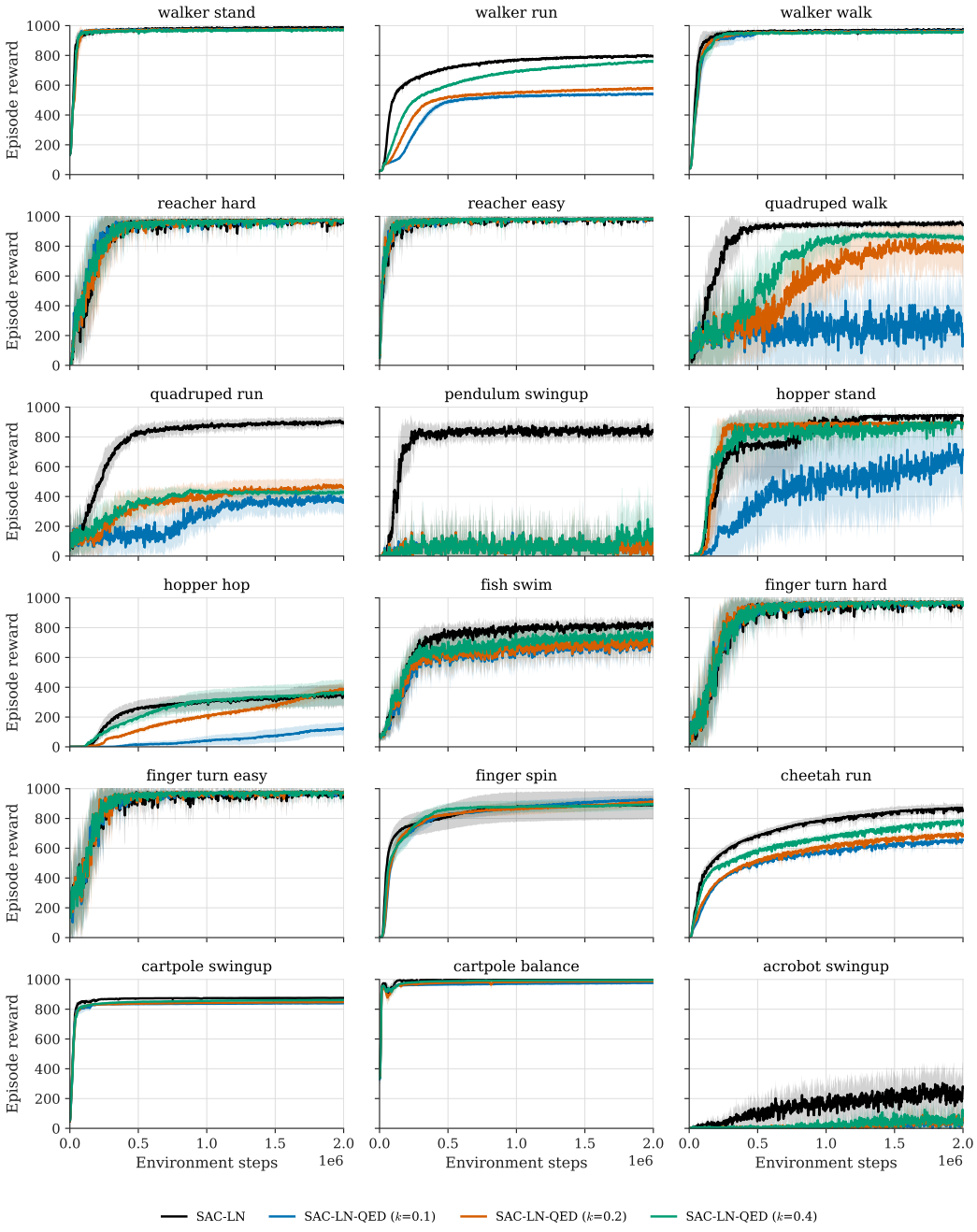


Figure 8: **Per-task learning curves for SAC-LN and SAC-QED on the 18-task `dm_control` suite.** Episode reward over environment steps for the SAC-LN baseline and QED variants with $k \in \{0.1, 0.2, 0.4\}$, matching the SAC-LN conditions used in the aggregate results in Figure 3a. QED generally preserves strong performance on easier tasks while introducing a performance-consistency trade-off on harder locomotion tasks, especially when the behavioral-consistency constraint is strongest.

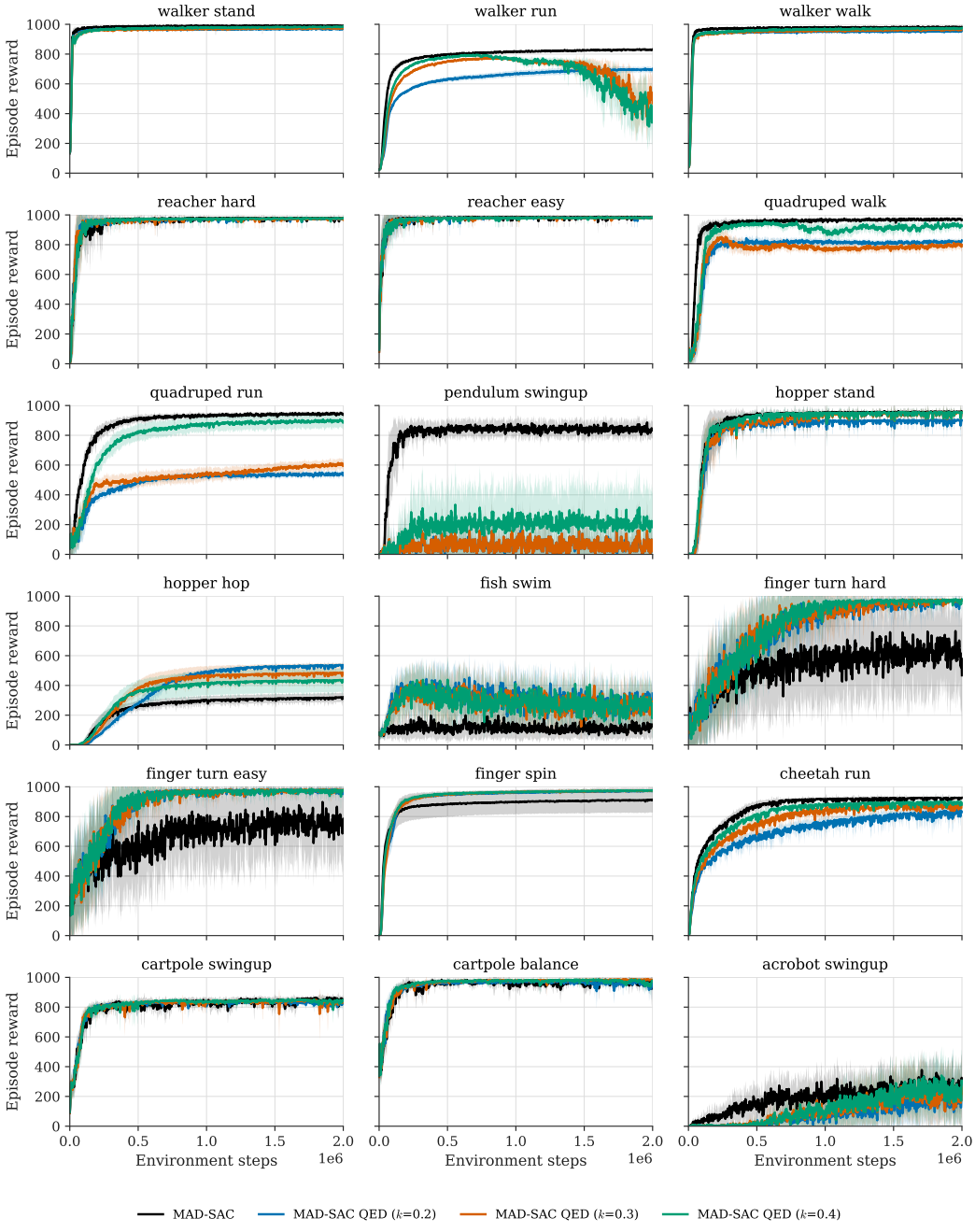


Figure 9: **Per-task learning curves for MAD-SAC and MAD-SAC-QED on the 18-task `dm_control` suite.** Episode reward over environment steps for the MAD-SAC baseline and QED variants with $k \in \{0.2, 0.3, 0.4\}$, matching the MAD-SAC conditions used in the aggregate results in Figure 3a. Compared with SAC-LN, MAD-SAC is more robust to the additional entropy induced by QED, and QED often preserves or improves learning while reducing seed-level dispersion.

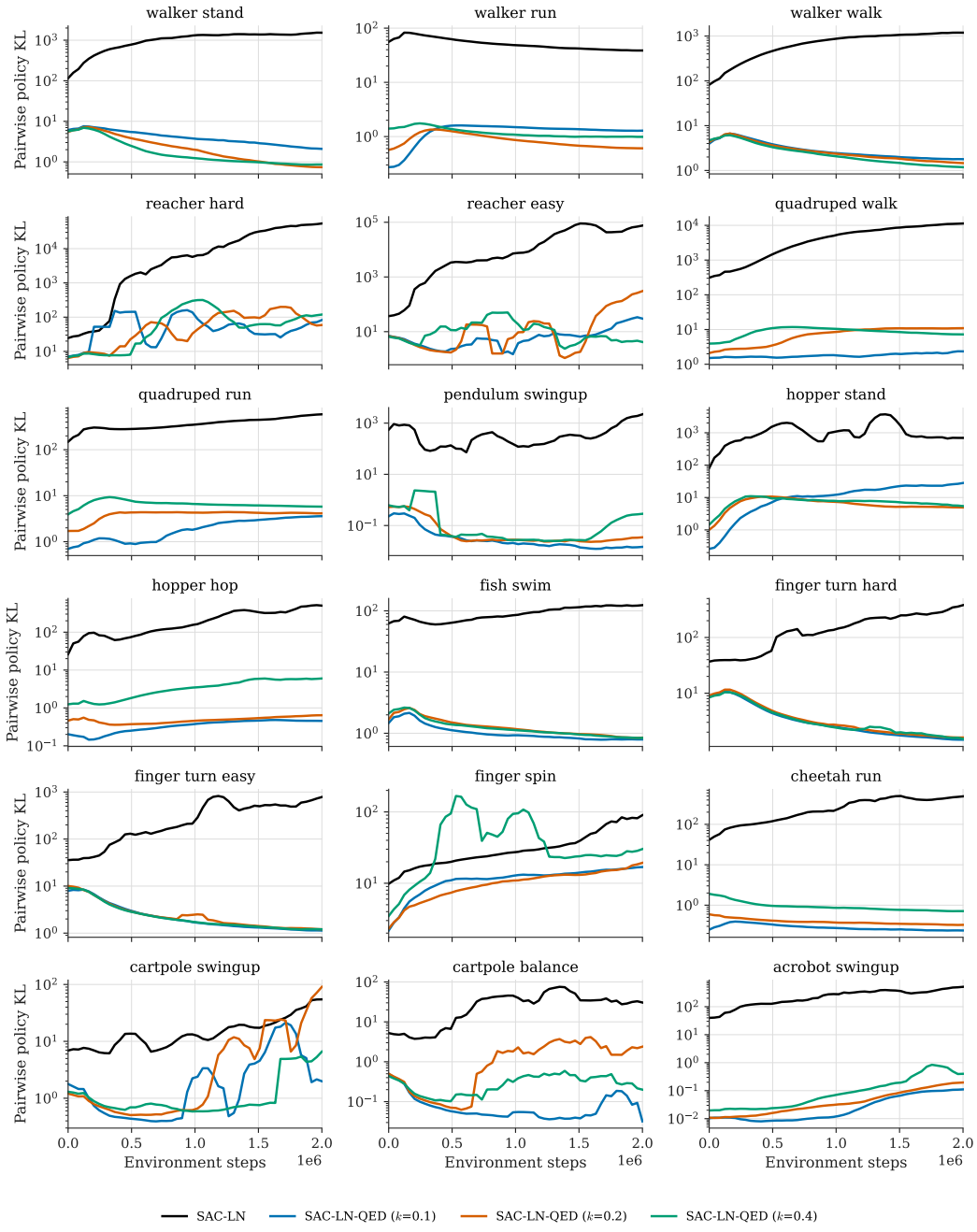


Figure 10: **Per-task inter-run policy divergence for SAC-LN and SAC-QED on the 18-task `dm_control` suite.** Pairwise symmetric pre-tanh policy KL over environment steps for the SAC-LN baseline and QED variants with $k \in \{0.1, 0.2, 0.4\}$, matching the SAC-LN conditions used in the aggregate results in Figure 3a. Across most tasks, QED substantially lowers inter-run policy divergence relative to standard target-entropy tuning, with stronger behavioral-consistency constraints producing lower KL but sometimes larger performance costs.

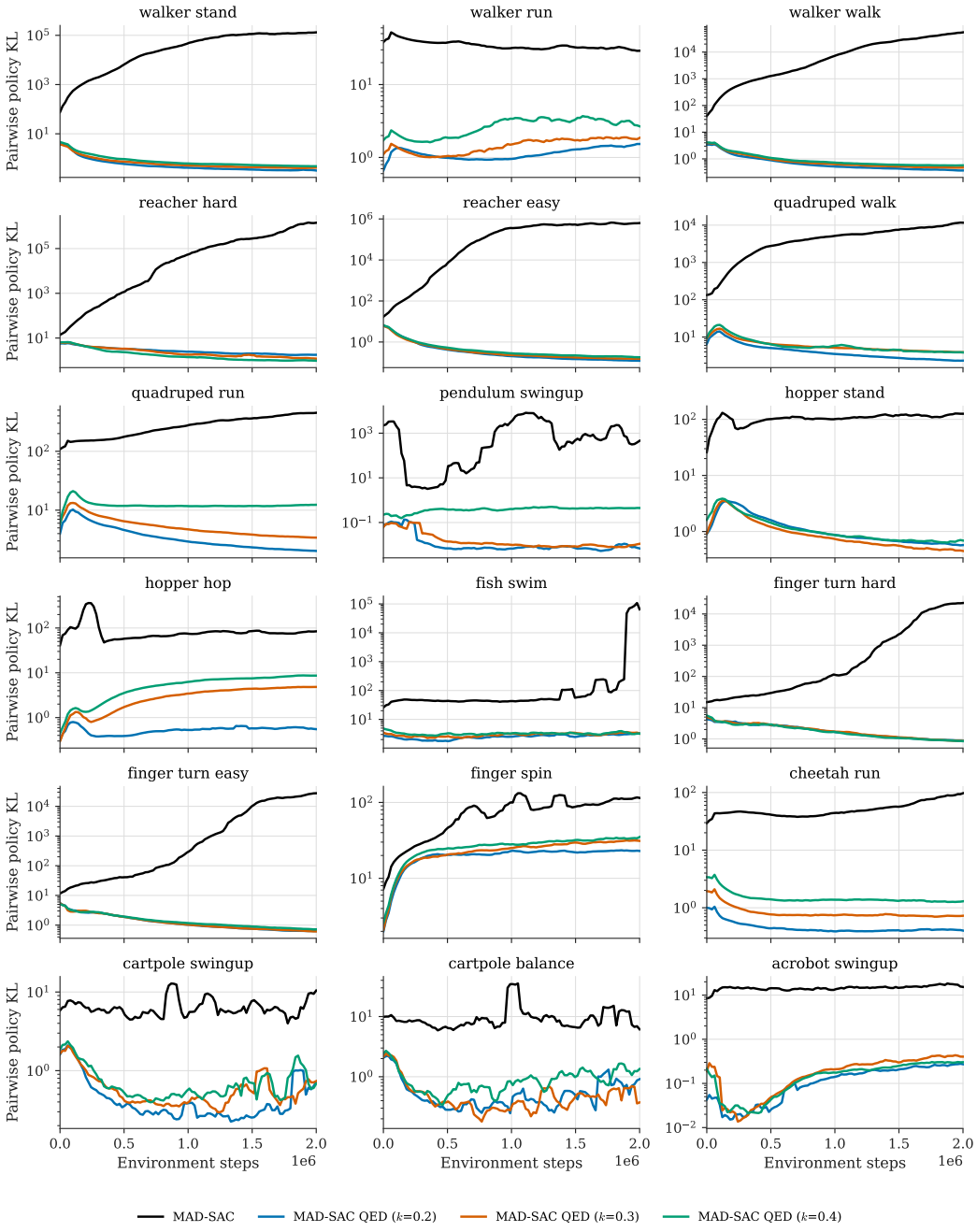


Figure 11: **Per-task inter-run policy divergence for MAD-SAC and MAD-SAC-QED on the 18-task `dm_control` suite.** Pairwise symmetric pre-tanh policy KL over environment steps for the MAD-SAC baseline and QED variants with $k \in \{0.2, 0.3, 0.4\}$, matching the MAD-SAC conditions used in the aggregate results in Figure 3a. QED consistently suppresses the growth of cross-seed policy divergence, showing that the behavioral-consistency effect holds across tasks and is not driven only by the aggregate results.