Revisiting Dual Policies with Bayesian Optimism in Reinforcement Learning

Anonymous authors
Paper under double-blind review

Abstract

Deep reinforcement learning algorithms for continuous control often rely on conservative Q-value estimates to stabilise actor-critic learning. This conservatism is useful for exploitation, but it can also discourage actions whose value is uncertain yet potentially high. Prior work has addressed exploration by adding intrinsic rewards, explicitly estimating uncertainty, or training separate exploration and exploitation policies. We revisit the dual-policy setting and ask whether optimistic exploration can be obtained without explicit uncertainty heads, learned optimism coefficients, or auxiliary exploration rewards. In this work, we propose BOXD, a simple dual-policy actor-critic method that trains an optimistic exploration policy alongside a conservative task policy. BOXD uses dropout-enabled critic ensembles and trains the exploration policy against the maximum over stochastic critic estimates. Under a local Gaussian approximation, this maximum behaves approximately like a UCB mean-plus-uncertainty objective, providing an optimistic training signal without explicitly estimating uncertainty. The task policy, meanwhile, remains trained with the conservative value targets used by the underlying actor-critic algorithm. During data collection, BOXD stochastically switches between the exploration and task policies, gradually increasing the use of the task policy as learning progresses. The shared replay buffer therefore accumulates both exploratory and task-aligned transitions, allowing each policy to benefit from the data collected by the other. Experiments on DeepMind Control Suite show that BOXD substantially improves SAC- and TD3-based agents, allowing them to remain competitive with heavily engineered state-of-the-art methods while retaining a relatively simple algorithmic structure. We find that BOXD leads to performance improvements even when combined with a strong XQC algorithm, along with gains on several tasks, and across these benchmarks it also compares favourably with related optimistic and dual-policy exploration baselines on several challenging tasks. These results suggest that dual-policy optimistic exploration can strengthen modern actor-critic algorithms and that exploration mechanisms remain a promising route for advancing state-of-the-art continuous-control performance.

1 Introduction

Modern actor-critic reinforcement learning (RL) must balance two competing uses of value estimates: stabilising exploitation and encouraging effective exploration. In continuous control, many state-of-the-art methods deliberately reduce overestimation bias (Thrun & Schwartz, 1993) by using conservative Q-value estimates (Chen et al., 2021; Fujimoto et al., 2018; Hasselt et al., 2016; Hiraoka et al., 2022; Kuznetsov et al., 2020; Lan et al., 2020). This is often beneficial for exploitation, since overestimated actions can be repeatedly reinforced by the policy and destabilise learning. However, the same conservatism can also discourage actions whose value is uncertain but potentially high. In this sense, value overestimation is not only a source of approximation error: it can also be interpreted as a form of *optimism in the face of uncertainty*, which may encourage the agent to explore uncertain regions of the state-action space.

This creates a fundamental tension between exploitation and exploration. Exploitation benefits from conservative and reliable value estimates, whereas exploration benefits from optimistic behaviour that probes actions not yet fully validated by the current critic. Asking a single policy to satisfy both objectives can

lead to a compromise that is neither sufficiently exploratory nor sufficiently exploitative. Prior work has considered separating exploration and exploitation policies (Colas et al., 2018; Li et al., 2022; Schäfer et al., 2022; Whitney et al., 2021). These dual-policy methods correctly recognise that exploration and exploitation are different behaviours. However, they often require auxiliary exploration rewards, intrinsic motivation modules, or additional objectives to train the exploration policy.

Another related line of work addresses exploration through optimistic value estimation. In particular, Upper Confidence Bound (UCB)-style methods train agents to prefer actions with high predicted value and high uncertainty (Ciosek et al., 2019; Moskovitz et al., 2021; O'Donoghue et al., 2018). Such methods provide a principled way to encourage optimism, but they often require explicit uncertainty estimation, additional learned components, or carefully tuned optimism parameters. Thus, prior dual-policy methods provide a useful separation between exploration and exploitation, while prior optimistic methods provide a useful training signal for exploration. The central idea of this work is to connect these two lines: we use a UCB-like target specifically as the training signal for the exploration policy in a dual-policy actor-critic framework. This perspective suggests that even strong modern actor-critic algorithms may still leave performance on the table when exploration and exploitation are handled by the same policy.

We propose **BOXD** (***B**ayesian **O**ptimism e**X**ploration with **D**ual Policies*), a dual-policy actor-critic framework that separates optimistic exploration from conservative task learning. BOXD maintains an exploration policy π^{explore} and a task policy π^{task}, each with its own critics. The task branch retains the original learning objective of the underlying actor-critic algorithm (Chen et al., 2021; Fujimoto et al., 2018; Haarnoja et al., 2018; Palenicek et al., 2026), while the exploration branch uses the maximum over stochastic, dropout-enabled critic estimates in its Bellman target (Gal & Ghahramani, 2016; Hiraoka et al., 2022). Under a local Gaussian approximation, the expected maximum has a mean-plus-uncertainty form analogous to a UCB objective (Auer et al., 2002; Ciosek et al., 2019; O'Donoghue et al., 2018). BOXD produces an optimistic exploration signal without an explicit uncertainty head, auxiliary exploration reward, or learned optimism coefficient, in contrast to existing optimistic or dual-policy approaches (Ciosek et al., 2019; Li et al., 2022; Schäfer et al., 2022). The overall framework is illustrated in Figure 1.

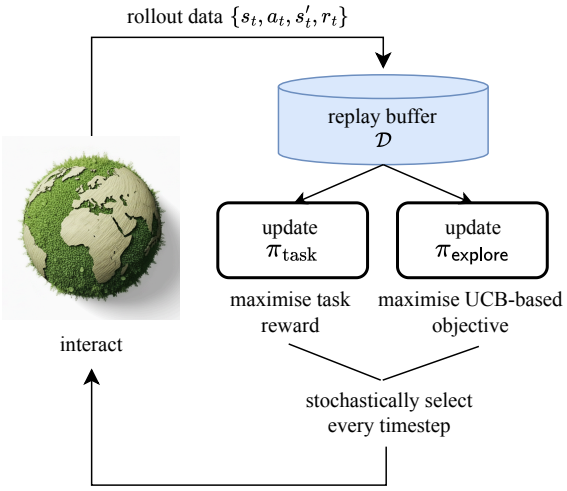

Figure 1: Schematic overview of our proposed method.

During environment interaction, a dynamic switcher selects between the two policies. Early data collection is dominated by the optimistic exploration policy, while the task policy is selected increasingly often as the two policies become aligned on replayed states. Both branches learn from the resulting shared replay buffer, which contains a mixture of exploration-driven and task-aligned transitions. This differs from applying optimistic and conservative estimates within a single policy objective (Ji et al., 2024; Omura et al., 2025): BOXD confines optimism to a dedicated exploration branch while preserving conservative learning for the task branch.

Our main contributions are threefold. Firstly, we introduce a simple max-over-stochastic-critics target that provides a UCB-like optimistic training signal for a dedicated exploration policy. Secondly, we combine this optimistic branch with a conservative task branch and an adaptive policy switcher, yielding a framework that can be added to different actor-critic algorithms. Thirdly, we evaluate BOXD on 14 DeepMind Control Suite tasks (Tunyasuvunakool et al., 2020) using SAC (Haarnoja et al., 2018), TD3 (Fujimoto et al., 2018), and XQC (Palenicek et al., 2026) as base algorithms, and we find that BOXD substantially improves SAC and TD3, provides moderate additional gains on the state-of-the-art XQC algorithm, and remains competitive with related optimistic, dual-policy, and heavily engineered continuous-control methods (Ciosek et al., 2019; Lee et al., 2021; Nauman et al., 2024). These results suggest that dual-policy optimistic exploration can

strengthen modern actor-critic algorithms and that exploration mechanisms remain a promising route for advancing state-of-the-art continuous-control performance.

The rest of the paper is organised as follows: we first motivate the benefit of optimism on a toy MDP (Section 3), then present BOXD (Section 4), and evaluate it on continuous-control benchmarks (Section 5).

2 Preliminaries

A standard RL problem (Bellman, 1957; Sutton & Barto, 2018) is defined as an infinite-horizon Markov Decision Process MDP $= \langle \mathcal{S}, \mathcal{A}, P, \mathcal{R}, \gamma \rangle$, where the RL agent at timestep t observes a state s_t from a set of states $\mathcal{S}$, chooses an action a from a set of actions $\mathcal{A}$, and receives a reward r according to a mapping of the reward function $\mathcal{R}$, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. The environment then transitions into a state s_{t+1} with a transition probability function $P(s_{t+1}|s_t, a_t)$ and the interaction continues. We also define the replay buffer $\mathcal{D}$ containing the state, action, reward, and next state at timestep t as $\mathcal{D} = \{(s_t, a_t, r_t, s_{t+1})\}_t$. The objective of an RL agent is to maximise the discounted expected return $\mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t)]$, which is the expected cumulative sum of rewards when following the policy in the MDP, and the importance of the horizon is determined by a discount factor $\gamma \in [0, 1)$.

Bayesian Optimisation and the Upper Confidence Bound. In Bayesian optimisation, the objective function $f(\boldsymbol{x})$ is assumed to be unknown, and the goal is to identify optimal $\boldsymbol{x}^* \in \mathcal{X}$ that maximises $f(\boldsymbol{x})$, given a set of observations $\{\boldsymbol{x}_i, y_i\}_{i=1}^N$ where $y_i = f(\boldsymbol{x}_i)$. The main challenge in Bayesian optimisation lies in effectively exploring the parameter space $\mathcal{X}$ whilst collecting informative samples. To this end, candidate points are typically selected by maximising an acquisition function $U(\boldsymbol{x})$. A widely used acquisition function is the upper confidence bound (UCB), defined as

$$U(\boldsymbol{x}) = \mu(\boldsymbol{x}) + c\sigma(\boldsymbol{x}) \tag{1}$$

where $\mu(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$ denote the predictive mean and standard deviation of $f(\boldsymbol{x})$, respectively, and c trades off optimism against conservatism.

In classical reinforcement learning, it has been noted that exploration using UCB often performs well in discrete state–action spaces (Auer et al., 2002; Sutton & Barto, 2018). However, in continuous state–action spaces, estimating the mean and standard deviation of the expected return is non-trivial (Ciosek et al., 2019; O'Donoghue et al., 2018), and therefore exploration based on the UCB principles is less straightforward in recent deep RL methods. BOXD provides a simple way to obtain a UCB-like optimistic target without explicitly learning separate uncertainty heads or optimism coefficients.

Using a Critic Ensemble in RL. Contemporary actor-critic reinforcement learning algorithms for continuous control typically train an ensemble of n independently initialised critics Q_j for $j = 1, 2 \ldots n$. For the computation of the target value y, the minimum value amongst these functions is employed: $y = r + \gamma \ \min_{j=1,\ldots,n} Q_j(s', a'), a' \ \sim \pi(\cdot|s')$. This calculation produces more conservative Q-value estimates (An et al., 2021; Chen et al., 2021; Ciosek et al., 2019; Fujimoto et al., 2018; Haarnoja et al., 2018; Hasselt et al., 2016; Hiraoka et al., 2022) in order to mitigate overestimation bias.

Using Model Ensembles for Capturing Uncertainty. Model ensembles can be interpreted as capturing a complementary form of epistemic uncertainty: rather than modelling only local posterior width around one mode, they approximate uncertainty through a mixture over solutions reached by different initialisations (Lakshminarayanan et al., 2017; Wilson & Izmailov, 2020). Recent work suggests that MC-dropout (Gal & Ghahramani, 2016) and ensembles capture complementary aspects of posterior uncertainty: MC-dropout reflects local uncertainty around a mode, while ensembles better capture multi-modality across modes (Fort et al., 2019; Izmailov et al., 2021; Ovadia et al., 2019; Wilson & Izmailov, 2020).

3 Motivating Example: Optimism Speeds Value Discovery

The purpose of this preliminary experiment is to investigate how the underestimation bias induced by the conservative targets commonly used in contemporary actor-critics affects the convergence rate, and how a UCB-style estimate can help mitigate it.

We conduct an experiment in a toy environment MDP as depicted in the left side of Figure 2 with the goal of evaluating the convergence rate of training Q-values towards their optimal values. In this environment, the reward r_k is obtained upon reaching state s_k, whilst the episode terminates when the agent reaches either state s_3 or s_4. For this simplified scenario, the discount rate γ is set to 0.9. Both the Q-values and the policy logits $\theta_{s,a}$ are stored in respective tables without modelling noisy functional approximators, and we use two differently initialised Q-tables.

We estimate the mean and standard deviation from the two Q-tables and compute a UCB-style value estimate, as depicted in Equation (1). The Q-values are updated using a temporal difference rule analogous to that of SARSA: $Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma \mathbb{E}_{a' \sim \pi}[Q(s',a')] - Q(s,a)\right)$, whilst the policy is updated using a policy gradient method ($\theta_{s,a} \leftarrow \theta_{s,a} + \alpha \nabla_\theta \log \pi_\theta(a|s)\ Q(s,a)$), where α represents the step size. Some additional details about this toy example can be found in Appendix A.

To obtain the best of both worlds, we also include a scheme where $Q^{\text{annealing}} = wQ^{\text{UCB}} + (1-w)Q^{\text{SARSA}}$ transitions from UCB to SARSA Q-values as w decays linearly from 1.0 to 0.0.

As demonstrated in the right sub-figure in Figure 2, we may observe that SARSA-based Q updates (in grey) are less biased but potentially sacrifice convergence rate. By performing UCB-principled optimistic estimation (in red), we may obtain faster convergence to the optimal Q-value, potentially improving sampling efficiency. This toy example motivates the design principle used later in BOXD: optimistic UCB Q-value estimates can help early value discovery, while a gradual transition towards the standard task objective can help recover a less biased exploitation policy. As depicted in the right sub-figure, the same intuition is behind the dynamic policy switcher we introduce in Subsection 4.2.

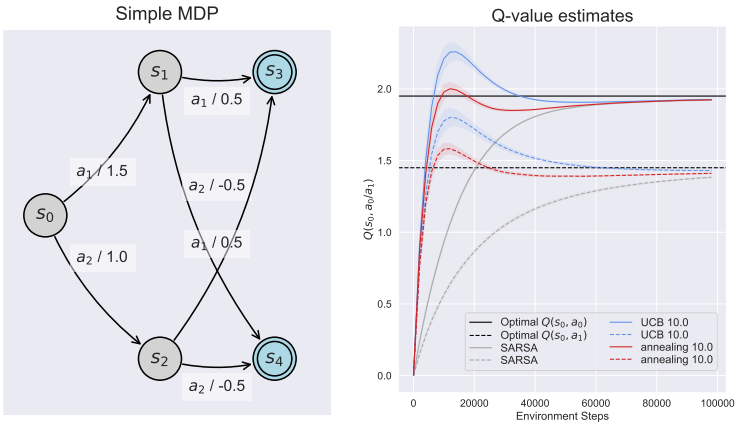

Figure 2: **Left:** A simple toy environment MDP. **Right:** The estimated values of $Q(s_0, a_0)$ and $Q(s_0, a_1)$ when using tabular actor-critic with a SARSA-based critic update, UCB-based update and annealing update. In the right sub-figure, the learning curve that includes the UCB may approximate the optimal value more swiftly.

4 Bayesian Optimism Exploration with Dual Policies

Method overview. BOXD augments a base actor-critic with a conservative task branch $(\pi^{\text{task}}, Q^{\text{task}})$ and an optimistic exploration branch $(\pi^{\text{explore}}, Q^{\text{explore}})$. The task branch retains the base algorithm's objective, whereas the exploration branch replaces conservative critic aggregation with the maximum over stochastic, dropout-enabled estimates. A policy switcher selects the acting branch, and both branches learn from a shared replay buffer. We next define the optimistic target and policy switcher before summarising the full algorithm.

4.1 Optimistically Trained Exploration Policy

Constructing a UCB-like target by directly estimating the mean and standard deviation of the critics typically requires explicit uncertainty models, distributional critics, or learned optimism coefficients. We instead use a simpler mechanism: taking the maximum over stochastic critic estimates.

Specifically, assume two samples are drawn from a Gaussian distribution: $X_1, X_2 \sim \mathcal{N}(\mu, \sigma^2)$. It is known that $\max(X_1, X_2)$ follows an extreme value distribution, and its expectation is given by

$$\mathbb{E}[\max(X_1, X_2)] = \mu + \frac{\sigma}{\sqrt{\pi}}, \tag{2}$$

where π is the circle constant (Arnold et al., 1992). This shows that taking the maximum over stochastic estimates naturally induces a mean-plus-uncertainty form, similar to the UCB acquisition function, without explicitly estimating a separate uncertainty bonus.

This relationship in Equation (2) may be generalised to the maximum of n samples. Let $X_j = \mu + \sigma Z_j$ where $Z_j \sim \mathcal{N}(0, 1)$. The expected value of the *maximum* taken across n Gaussian samples is

$$\begin{aligned}
&\mathbb{E}[\max(X_1, X_2, \ldots, X_n)] && (3)\\
&= \mathbb{E}[\max(\mu + \sigma Z_1, \mu + \sigma Z_2, \ldots, \mu + \sigma Z_n)] && (4)\\
&= \mu + \mathbb{E}[\max(Z_1, \ldots, Z_n)]\sigma && (5)\\
&= \mu + \int_{-\infty}^{\infty} z \cdot n[\Phi(z)]^{n-1}\phi(z)\, dz\, \sigma && (6)\\
&\approx \mu + \Phi^{-1}\left(\frac{n - 0.375}{n + 0.25}\right)\sigma \qquad \text{from (Arnold et al., 1992; Blom, 1958)} && (7)
\end{aligned}$$

where $\Phi(z)$ denotes the standard normal cumulative distribution function (CDF) and $\phi(z)$ denotes the standard normal probability density function (PDF). Under this approximation, taking the *maximum* over Gaussian samples behaves like a UCB-style objective, where the optimism coefficient is determined by the number of samples n. We therefore use Equation (7) as a motivating approximation for the optimistic target.

We next describe how BOXD obtains the stochastic critic estimates. The exploration branch contains n independently initialised critics. Following DroQ (Hiraoka et al., 2022), we add dropout and layernorm to each critic. Let $Q_j(s, a)$ denote a single dropout-sampled evaluation of exploration critic j. BOXD therefore obtains variation from two complementary sources: different dropout masks every forward pass produce local stochasticity within each critic, while independent critic initialisations produce disagreement across learned solutions. These may be viewed as complementary approximate sources of epistemic uncertainty (Durasov et al., 2021; Fort et al., 2019; Gal & Ghahramani, 2016; Izmailov et al., 2021; Ovadia et al., 2019; Wilson & Izmailov, 2020).

Given a next action $a' \sim \pi^{\text{explore}}(\cdot|s')$, BOXD forms the optimistic exploration value by taking the maximum over one stochastic evaluation from each exploration critic:

$$Q^{\text{explore}}_{\max}(s', a') = \max_{j=1,\ldots,n} Q^{\text{explore}}_j(s', a') \tag{8}$$

This maximum is used in the Bellman target of the exploration critics. Omitting base-algorithm-specific entropy, distributional, and target-network terms for clarity, the target is

$$y = r(s, a) + \gamma\, Q^{\text{explore}}_{\max}(s', a') \tag{9}$$

The exploration actor is then updated using the standard actor objective of the chosen base algorithm and its exploration critics. The task branch follows the corresponding base algorithm throughout, including its conservative critic aggregation, by adding the same dropout-and-layer-normalisation architecture as the exploration branch.

In this construction, the number of exploration critics n is the primary optimism-related term, that is, increasing n increases the implicit optimism coefficient. Dropout supplies stochasticity within each critic, while independent initialisation supplies diversity across critics. Detailed base-algorithm-specific targets and implementation settings are provided in Appendix B.

4.2 Dynamic Policy Switcher When Interacting with the Environment

The interacting policy π^{act} is chosen between the optimistic exploration policy π^{explore} and the conservative task policy π^{task} at each timestep. There are several strategies for this policy selection. We propose to use

a dynamic switcher that adapts this stochastic selection according to the current learning state. The goal is to keep using π^{explore} when the task branch is still unreliable, and to increase the probability of using π^{task} once the task policy becomes aligned with the exploration policy on replayed states. Furthermore, we explore another simple timestep based annealing strategy in Equation (13).

Our proposed switcher maintains a scalar mixing coefficient $\lambda^{\text{mix}} \in [0, 1]$. This coefficient controls the probability of selecting the task policy during environment interaction. The acting policy is selected as

$$\pi^{\text{act}} = \begin{cases} \pi^{\text{explore}} & \text{if } u > \lambda^{\text{mix}}, \\ \pi^{\text{task}} & \text{otherwise,} \end{cases} \quad \text{where} \quad u \sim \mathcal{U}(0, 1) \tag{10}$$

Thus, when λ^{mix} is small, BOXD mostly acts with the exploration policy; when λ^{mix} becomes large, BOXD increasingly acts with the task policy.

For the dynamic switcher, let k denote the switcher-update index. To update λ_k^{mix}, we estimate an agreement score from an additional switcher replay batch $\mathcal{D}_{\text{switcher}}$. This $\mathcal{D}_{\text{switcher}}$ contains 50% task samples and 50% exploration samples. The agreement score measures whether the task policy and exploration policy propose similar actions on replayed states. Let $\mu^{\text{task}}(s)$ and $\mu^{\text{explore}}(s)$ denote the mean of the Gaussian policies π^{task} and π^{explore}, respectively. At switcher update k, the agreement score is computed as:

$$S_k^{\text{agreement}} = \mathbb{E}_{s \sim \mathcal{D}_{\text{switcher}}} \left[\exp\left(- \left\| \mu^{\text{task}}(s) - \mu^{\text{explore}}(s) \right\|_2^2 \right) \right] \tag{11}$$

This score is high when the task policy remains close to the exploration policy on replayed states, and low when the two policies propose substantially different actions. Agreement is used as a proxy for when increasing task-policy interaction is less likely to move data collection away from the support of $\mathcal{D}_{\text{switcher}}$. A strong disagreement suggests that switching too quickly to π^{task} may move data collection away from the exploratory behaviour that generated the current replay distribution.

We further smooth this agreement score using a two-stage exponential moving average (EMA), which is initialised from its first observed agreement score as $\lambda_1^{\text{mix}} = \alpha_{\text{switcher}} S_1^{\text{agreement}}$:

$$\lambda_k^{\text{mix}} \leftarrow (2 - \alpha_{\text{switcher}} - \beta_{\text{EMA}})\lambda_{k-1}^{\text{mix}} - (1 - \alpha_{\text{switcher}})(1 - \beta_{\text{EMA}})\lambda_{k-2}^{\text{mix}} + \alpha_{\text{switcher}}\beta_{\text{EMA}} S_k^{\text{agreement}}, \quad k \geq 2 \tag{12}$$

The switcher is updated every 30,000 environment steps, and k counts these switcher updates. The first EMA uses an update rate of $\beta_{\text{EMA}} = 0.005$, and the values of α_{switcher} used in our experiments are reported in Subsection 5.1. The derivation of Equation (12) is given in Appendix B.1.

Conceptually, the dynamic switcher moves towards the task policy once it agrees with the exploration policy on replayed states. As a result, the replay buffer is populated mainly by transitions collected from the exploration policy when the task branch is not yet reliable, and increasingly by task-aligned samples once the task branch becomes aligned with the exploration policy on replayed states. This preserves the core design of BOXD: optimism is used for data collection through π^{explore}, while conservative value learning is preserved for π^{task}.

Non-dynamic alternative. We can also formulate a simpler timestep annealing switcher as:

$$\lambda_t^{\text{mix}} = \left\lfloor \frac{10\, t}{T} \right\rfloor \Big/ 10 \tag{13}$$

where t is the current environment step and T is the total number of training steps. This produces a staircase that raises λ_t^{mix} from 0 to 0.9 over training, and does not depend on the agreement between the two policies. Visualisation of these two switchers' changes during learning is in Figure 7.

Algorithm 1: BOXD

Initialise policy networks π^{explore} and π^{task}, exploration critic parameters ϕ_j, $j = 1, \ldots, n$, task critic parameters ψ_j, $j = 1, 2$, and an empty replay buffer $\mathcal{D}$. Set $\lambda_0^{\text{mix}} = 0$ and switcher-update index $k = 0$. Set target parameters if required by the base algorithm.

while *initial collection steps* $\leq t \leq T$ **do**

  Take action $a_t \sim \pi^{\text{act}}(\cdot|s_t)$ according to the switcher policy selection Equation (10), and record the selected branch $b_t \in \{\text{explore}, \text{task}\}$. Observe reward r_t, next state s_{t+1};
  $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t, r_t, s_{t+1}, b_t)$.

  Sample a mini-batch $\mathcal{B} = \{(s_i, a_i, r_i, s_i')\}_{i=1}^{B}$ from $\mathcal{D}$.

  ✠ Update optimistic critic Q_ϕ^{explore}

  Compute the target value for Q_ϕ^{explore} using one stochastic dropout evaluation per exploration critic (Equation (8)):
  $y_i^{\text{explore}} = r_i + \gamma \max_{j=1,\ldots,n} Q_{\phi_j}^{\text{explore}}(s_i', a_i'), \quad a_i' \sim \pi^{\text{explore}}(\cdot|s_i')$
  Update each $Q_{\phi_j}^{\text{explore}}$ by minimising $B^{-1} \sum_{i=1}^{B} (y_i^{\text{explore}} - Q_{\phi_j}^{\text{explore}}(s_i, a_i))^2$

  ✠ Update conservative critic Q_ψ^{task}

  Compute the target value for Q_ψ^{task}:
  $y_i^{\text{task}} = r_i + \gamma \min_{j=1,2} Q_{\psi_j}^{\text{task}}(s_i', a_i'), \quad a_i' \sim \pi^{\text{task}}(\cdot|s_i')$
  Update each $Q_{\psi_j}^{\text{task}}$ by minimising $B^{-1} \sum_{i=1}^{B} (y_i^{\text{task}} - Q_{\psi_j}^{\text{task}}(s_i, a_i))^2$

  ✠ Update exploration policy π^{explore} via base algorithm using Q^{explore}

  ✠ Update task policy π^{task} via base algorithm using Q^{task}

  ✠ Update target networks if applicable, depending on the base algorithm

  if *using the dynamic switcher & time to update switcher* **then**

    Sample $\mathcal{D}_{\text{switcher}}$ with equal numbers explore/task transitions.
    $k \leftarrow k + 1$.
    Compute $S_k^{\text{agreement}}$ from $\mathcal{D}_{\text{switcher}}$ using Equation (11).
    Update λ_k^{mix} using Equation (12).

  end

end

4.3 Algorithmic Summary

In summary, BOXD adds an exploration actor-critic branch with a max-over-stochastic-critics target alongside the conservative task branch, plus a switcher that alternates between them during interaction. The pseudocode of our proposed algorithm is presented in Algorithm 1.

5 Experiments

In this section, we empirically evaluate the performance of BOXD, comparing it to related online RL approaches on a variety of challenging tasks. We also provide analyses on BOXD's design choices and include ablation studies in the appendices. Detailed hyperparameters and implementation details used in our experiments are in Appendix B, and details about metrics calculations and aggregations are in Appendix C.

5.1 Experimental Setups

BOXD variants and settings. BOXD is built on XQC and SAC in the main experiments, with deterministic-policy results based on TD3 reported in Appendix E. For all BOXD variants, we add dropout and layernorm to all critics, use $p = 0.001$. For SAC+BOXD, the default α_{switcher} is 0.05; we use 0.02 for cheetah-run, hopper-hop, humanoid-run, and quadruped-walk, and 0.06 for hopper-stand and humanoid-

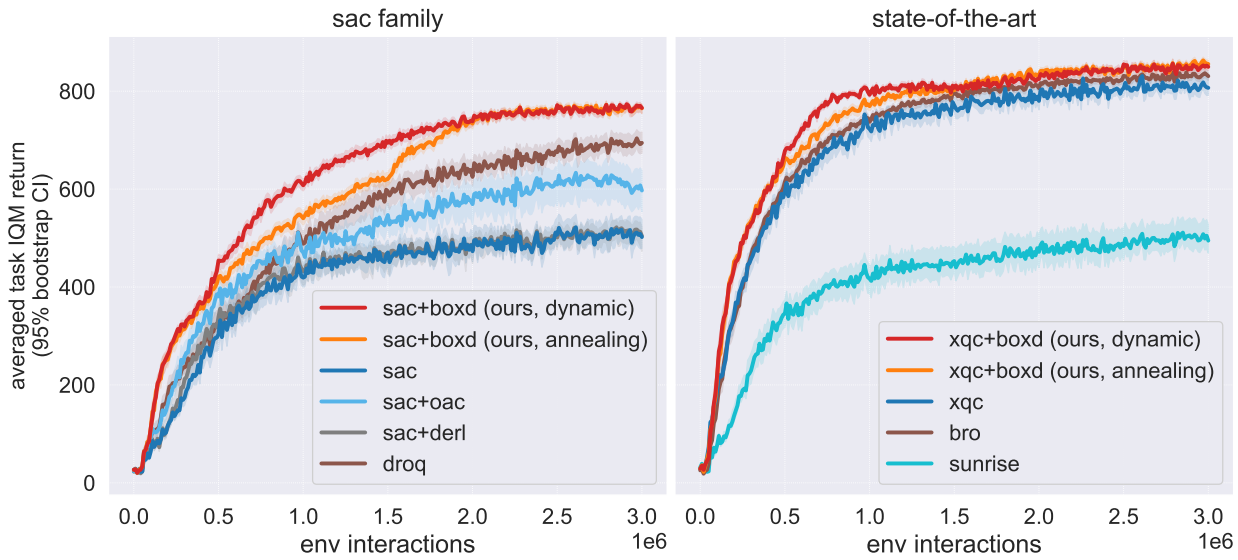

Figure 3: Learning curves of the mean per-task IQM return across 14 DM Control tasks. The shaded areas show 95% percentile-bootstrap confidence intervals for the aggregate statistic, obtained by independently resampling seeds within each task. BOXD improves strong actor-critic baselines and is competitive with related optimistic methods.

walk. For XQC+BOXD, we use $\alpha_{\text{switcher}} = 0.05$ on every task. Further results and ablations are included in Appendix D, Appendix E and Appendix F.

Comparative methods. In addition to comparing with the base algorithms, we provide a comparative performance analysis against five other related methods. The first is an approach analogous to DERL (Schäfer et al., 2022), a framework which is conceptually similar in that it also disentangles exploration and exploitation policies, whilst adding exploration bonuses to the exploration policy to encourage it. We used RND (Burda et al., 2019) for its exploration bonus. The second is OAC (Ciosek et al., 2019), a SAC-based algorithm (therefore single policy) notable for its use of optimistic Q-value estimation by UCB principles whilst addressing the lack of directional information of action sampled from the policy. The third is SUNRISE (Lee et al., 2021), which we include as an ensemble-based exploration baseline that combines uncertainty-weighted Bellman backups with UCB-style action selection. The fourth is DroQ (Hiraoka et al., 2022), a SAC-based method that adds dropout and layernorm to the critics; it corresponds to BOXD's conservative task branch without the optimistic exploration policy, therefore isolates the contribution of the dual-policy mechanism from the dropout-layernorm regularisation. The fifth related method is BRO (Nauman et al., 2024), one of the state-of-the-art algorithms that combines multiple advances.

Benchmark and evaluation method. We evaluate our method on challenging tasks in the commonly used DeepMind Control (DM Control) Suite (Tunyasuvunakool et al., 2020). We train every task for 3 million environment steps using 10 random seeds $\{0, \ldots, 9\}$ and evaluate every 10000 steps using 20 episodes per seed. For each task, we report the interquartile mean (IQM) across the 10 seed-level returns following recent trends and best practices (Agarwal et al., 2021). Aggregate figures report the average of these per-task IQMs. Their shaded bands are 95% percentile-bootstrap confidence intervals for the aggregate statistic, computed from 5000 replicates by independently resampling seeds within each task and recomputing the complete aggregate. During evaluation, Equation (10) is applied at every environment step, so the acting policy is resampled between π^{explore} and π^{task} according to λ^{mix}. Further details of the metric and confidence-interval computation are provided in Appendix C.

5.2 Results and Q&As

Our experiments aim to answer the following questions.

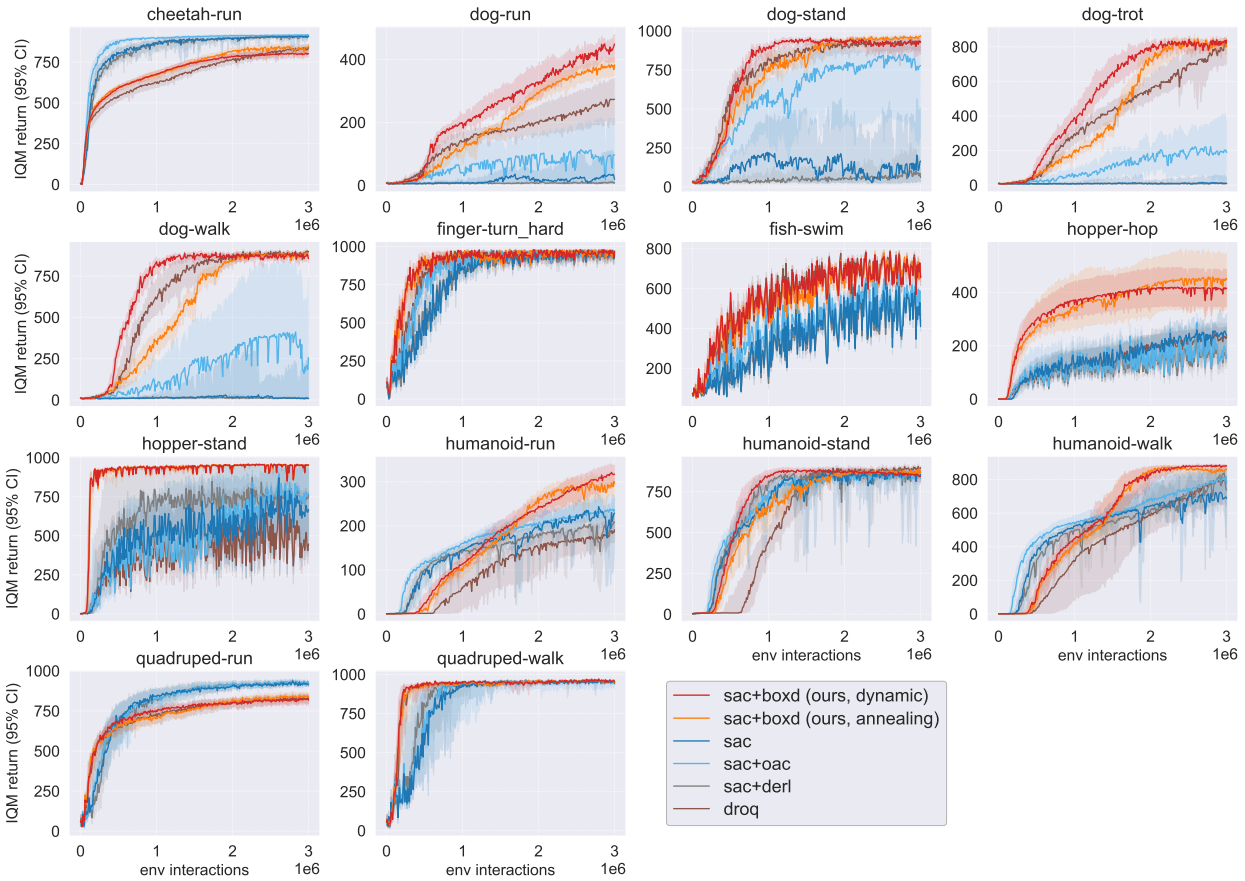

Figure 4: Learning curves of the per-task IQM returns in DM Control of our proposed method versus baselines. Our proposed method (orange, red) generally achieves the best or near-best performance, with particularly large gains over SAC in tasks such as hopper tasks and humanoid tasks.

Q1: What is the performance of BOXD for DM Control benchmarking tasks?
A1: BOXD improves SAC on aggregate.

The mean per-task IQM returns across 14 DM Control tasks are depicted in Figure 3 for the SAC-family algorithms and XQC-based BOXD variants; the corresponding per-task results are shown in Figures 4 and 8.

Compared with baselines, SAC+BOXD generally achieves best or near-best performance among the benchmarked methods, outperforming OAC, DERL, SUNRISE, and DroQ. The hopper and humanoid gains likely arise because these larger-state-space tasks benefit from stronger exploration, where effective data collection is more difficult. Conversely, our proposed BOXD underperforms cheetah-run, we hypothesise that this is because it can be solved relatively easily using single mode, so a stronger exploration mechanism provides little benefit. The delayed humanoid improvement likely arises because early exploration-focused data does not immediately aid exploitation, but becomes valuable once the task policy strengthens.

Q2: How does BOXD compare with state-of-the-art algorithms, such as BRO or XQC?
A2: XQC+BOXD compares favourably with BRO and remains competitive with XQC.

This suggests that improving the exploration mechanism can complement heavily engineered state-of-the-art systems, without relying on all of their additional architectural components. The mean per-task IQM returns are included in Figure 3, with the XQC per-task results in Figure 8.

In the task-stratified probability-of-improvement (POI) plot in Figure 5, values above 0.5 favour the algorithm named in the subplot title. "XQC > X" shows that XQC achieves higher performance than the other baselines. XQC+BOXD remains above this threshold against BRO throughout training. Its advantage over XQC is more pronounced early in training, and while the performance gain slightly diminishes in later stages,

the POI remains consistently above 0.5 throughout most of the training. Based on the two aggregations (Figure 3 and Figure 5), this suggests that, comparing XQC vs XQC+BOXD, the aggregate gain is driven by improvements on some tasks and the proposed exploration method primarily accelerates early-stage performance improvements on state-of-the-art algorithms.

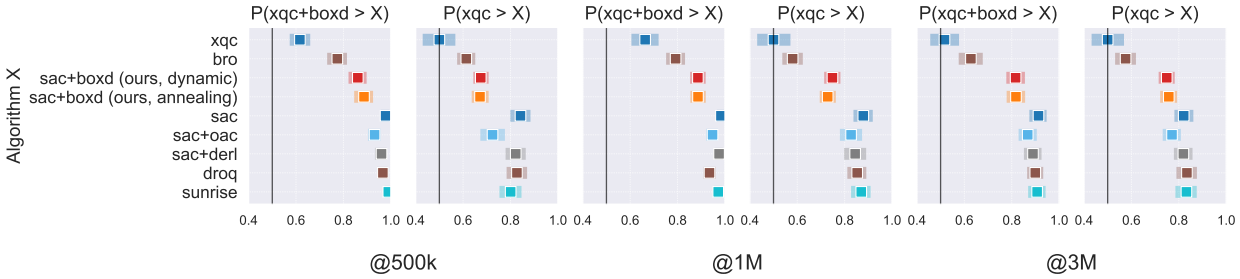

Figure 5: POI across 14 DM Control tasks at 500k, 1M, and 3M steps. Squares show point estimates and bands show 95% bootstrap confidence intervals. XQC+BOXD compares favourably with BRO and SAC-family methods, while approaching parity with XQC at 3M steps.

Q3: Does adaptive switching provide an advantage?

A3: The dynamic switcher improves early sample efficiency. No switching and strongly exploration-biased mixing perform worse, indicating that transitioning towards the task policy is important.

To verify the effectiveness of our proposed dynamic policy switcher, we compare the dynamic switcher with fixed policy-selection probabilities on 10 non-dog tasks: always exploration (100% – 0%), 90% – 10%, and 50% – 50% exploration-task mixing. The mean per-task IQM returns are depicted in Figure 6, and per-task results are provided in Appendix G.

Figure 6 shows that with no switching (100-0) or a poorly chosen fixed mixing (90-10), the performance stagnates or decays at the later stages of training. We hypothesise that this is because the replay buffer then contains too many exploration-aligned samples and too few that are aligned with the exploitation objective. Some form of switching from π^{explore} to π^{task} is therefore beneficial, as also suggested by the toy example Section 3. Perhaps surprisingly, a balanced 50-50 probability sampling between the exploration and exploitation policies performs remarkably well.

Although the dynamic switcher does not improve final performance over annealing, it improves sample efficiency earlier in training, so returns begin to rise sooner. This indicates that the dual-policy exploration signal itself, rather than the precise switching schedule, is the main driver of BOXD's gains.

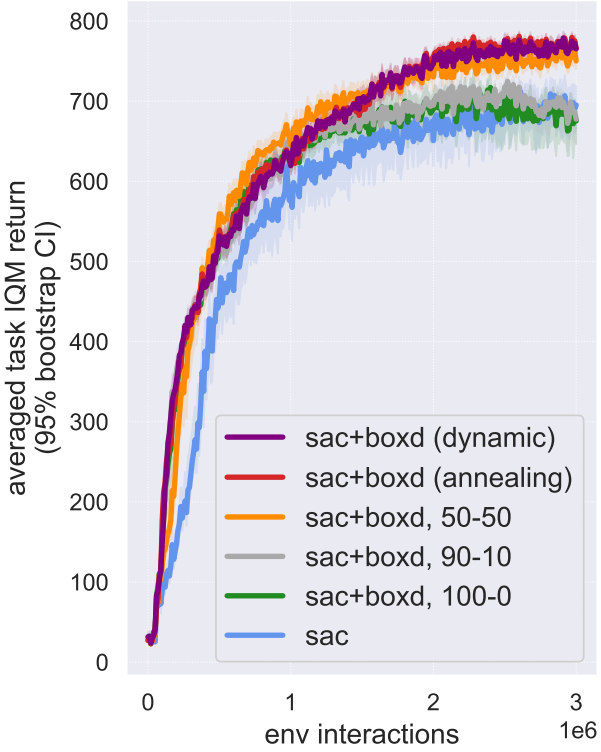

Figure 6: Learning curves of the mean per-task IQM return across 10 DM Control tasks (excluding dog tasks) for SAC+BOXD with different switching strategies. The shaded areas show 95% percentile-bootstrap confidence intervals for the aggregate statistic, obtained by independently resampling seeds within each task. No switching and poorly chosen fixed switching show signs of performance saturation.

Q4: How does λ^{mix} evolve during training?

A4: The dynamic λ^{mix} rises rapidly and settles at a task-dependent level, typically between 0.5 and 0.8. It can decrease when the policies diverge, unlike fixed timestep annealing (Figure 7).

Both strategies start at zero, so early data collection is dominated by the exploration policy. The dynamic switcher then raises λ^{mix} as the task policy comes to agree with the exploration policy on replayed states, whereas the annealing schedule raises it on a fixed timetable. This adaptation is task-specific: as shown in Figure 7, on tasks such as dog-stand, finger-turn_hard and humanoid-stand, λ^{mix} even falls back once the two policies diverge, rather than following the fixed annealing schedule. This confirms that the switcher adapts the exploration-exploitation balance per task.

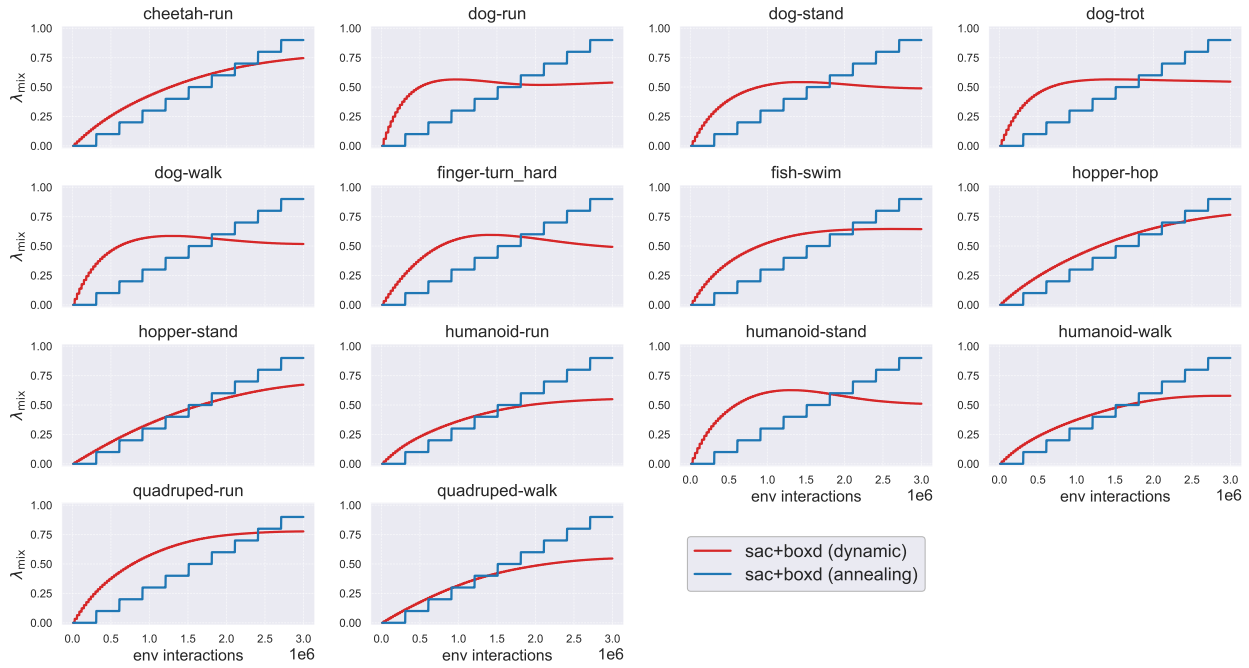

Figure 7: Evolution of the switcher mixing coefficient λ^{mix} during training for SAC+BOXD.

Q5: What are the important hyperparameters of BOXD?

A5: The number of exploration critics n controls the strength of optimism, while α_{switcher} controls the switching rate.

Increasing n raises the expected maximum over exploration-critic estimates and therefore strengthens the implicit optimism of the exploration target. However, more optimism is not uniformly beneficial: larger ensembles improve some exploration-sensitive tasks but can degrade others. We therefore use $n = 2$, matching the number of task critics, as a simple and robust default.

The switching rate α_{switcher} determines how quickly λ^{mix} follows the smoothed policy-agreement signal. Larger values shift data collection towards the task policy more rapidly, whereas smaller values retain exploration-policy interaction for longer. Its effect is mainly task-dependent timing rather than a uniform change in final performance. Ablations of both hyperparameters are reported in Appendix F.

6 Related Works

Separating exploration and exploitation. One line of work trains distinct behaviours or policies for exploration and exploitation. GEP-PG (Colas et al., 2018) separates a goal-directed exploration phase from subsequent policy optimisation, while DEEP (Whitney et al., 2021) and DERL (Schäfer et al., 2022) train exploration and exploitation objectives separately, with DERL using intrinsic rewards to drive its exploration policy. MULEX (Beyer et al., 2019) maintains multiple policies with different exploration-exploitation trade-offs and selects among them during interaction. Most closely, Li et al. (2022) jointly train an optimistic exploration policy and a target policy using critic UCBs and DICE-based distribution correction. BOXD similarly separates exploratory data collection from conservative task learning, but derives optimism from

the maximum over dropout-enabled critics and switches directly between its two policies, without intrinsic rewards or distribution correction.

Related forms of separation also appear in other settings. In meta-RL, exploration may be learned as a distinct information-gathering phase before exploitation (Liu et al., 2021; Norman & Clune, 2024); offline-to-online RL can separate offline policy learning from subsequent online exploration (Mark et al., 2023); and decoupled exploration and exploitation have also been studied in multi-armed bandits (Avner et al., 2012). BOXD instead addresses a purely online, concurrent actor-critic setting in which both policies learn from a shared replay buffer.

Optimistic value estimation for exploration. A second line of work encourages exploration by acting according to optimistic value estimates. UCB-based approaches construct actions from a combination of predicted value and uncertainty (Ciosek et al., 2019; Moskovitz et al., 2021; O'Donoghue et al., 2018). OAC (Ciosek et al., 2019), for example, derives an optimistic action distribution from critic uncertainty, while Tactical Optimism and Pessimism (TOP) (Moskovitz et al., 2021) adaptively alternates between optimistic and pessimistic value estimates within a single policy. Most closely related, SUNRISE (Lee et al., 2021) trains a critic ensemble and uses ensemble disagreement both to weight Bellman backups and to select actions through a UCB-style rule; like BOXD it derives optimism from a critic ensemble, but applies it within a single acting policy rather than routing it to a dedicated exploration policy. BRO (Nauman et al., 2024) similarly incorporates optimistic value estimates as one component of a larger actor-critic system. These methods apply optimism within a single acting objective or a broader ensemble framework, whereas BOXD assigns optimistic and conservative objectives to separate exploration and task branches and switches between their policies during data collection.

Another related direction obtains more optimistic targets by moving towards the Bellman optimality operator. Methods based on IQL-style objectives or annealed Bellman operators (Ji et al., 2024; Kostrikov et al., 2022; Luo et al., 2024; Omura et al., 2025) blend optimistic and standard value estimates within the learning objective of one policy. BOXD instead keeps the two objectives separated: the exploration policy is trained from an optimistic max-over-stochastic-critics target, while the task policy continues to use the conservative target of the chosen base algorithm. Taken together, BOXD combines the behavioural separation of dual-policy methods with the training signal emphasised by optimistic value-estimation methods.

7 Limitations and Conclusion

This work introduces BOXD, an algorithm built on the principle of disentangling exploration and exploitation. BOXD separates optimistic data collection from conservative task learning. Its exploration branch uses the maximum over stochastic critic estimates as an implicit optimistic target, while its task branch retains the objective of the underlying actor-critic algorithm. An agreement-based switcher controls which policy interacts with the environment. Across 14 DM Control tasks, BOXD substantially improves SAC and TD3. When added to XQC, it compares favourably with BRO and remains competitive with XQC at 3 million steps. The ablations further indicate that maximum aggregation can benefit exploration-sensitive tasks, while the adaptive switcher primarily improves early sample efficiency rather than consistently increasing final performance.

Several limitations remain. First, BOXD requires an additional policy and critic ensemble, increasing memory use and computation. Second, the UCB interpretation is a motivating approximation rather than a calibrated uncertainty guarantee, thus the effects of the critic count and dropout rate are task-dependent. Third, our evidence is restricted to online continuous control in the DM Control Suite. The reported confidence intervals quantify seed-level uncertainty for these fixed tasks and do not establish generalisation to other task distributions. The adaptive switcher also warrants further study. It does not consistently outperform balanced fixed mixing at convergence, and uses task-specific switching rates for some SAC experiments. Future work should therefore examine a learned or self-tuning switcher, broader and sparser-reward benchmarks, compute-efficient parameter sharing, and replay sampling that accounts for whether transitions were collected by the exploration or task policy. Despite these limitations, the results show that isolating optimism within a dedicated data-collection policy is a practical way to strengthen modern actor-critic algorithms without changing their conservative task-learning objectives.

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 (NeurIPS)*, 2021.

Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-Based Offline Reinforcement Learning with Diversified Q-Ensemble. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Barry C. Arnold, N. Balakrishnan, and Hari Nagaraja. *A First Course in Order Statistics*. Wiley, New York, 1992.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time Analysis of the Multiarmed Bandit Problem. In *Machine Learning*, 2002.

Orly Avner, Shie Mannor, and Ohad Shamir. Decoupling Exploration and Exploitation in Multi-Armed Bandits. In *International Conference on Machine Learning (ICML)*, 2012.

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. *arXiv preprint arXiv:1607.06450*, 2016. URL `http://arxiv.org/abs/1607.06450`.

Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.

Lucas Beyer, Damien Vincent, Olivier Teboul, Sylvain Gelly, Matthieu Geist, and Olivier Pietquin. MULEX: Disentangling Exploitation from Exploration in Deep RL. *arXiv preprint arXiv:1907.00868*, 2019. URL `https://arxiv.org/abs/1907.00868`.

Gunnar Blom. *Statistical Estimates and Transformed Beta-Variables*. John Wiley & Sons, New York, 1958. ISBN 9780471080157.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/jax-ml/jax`.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. *arXiv preprint 1606.01540*, 2016.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations (ICLR)*, 2019.

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 (ICLR)*, 2021.

Kamil Ciosek, Quan Vuong, Robert Loftin, and Katja Hofmann. Better Exploration with Optimistic Actor-Critic. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. GEP-PG: Decoupling Exploration and Exploitation in Deep Reinforcement Learning Algorithms. In *International Conference on Machine Learning (ICML)*, 2018.

Rodrigo de Lazcano, Kallinteris Andreas, Jun Jet Tai, Seungjae Ryan Lee, and Jordan Terry. Gymnasium Robotics, 2024. URL `http://github.com/Farama-Foundation/Gymnasium-Robotics`.

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 (ICLR)*, 2023.

Nikita Durasov, Timur Bagautdinov, Pierre Baque, and Pascal Fua. Masksembles for Uncertainty Estimation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep Ensembles: A Loss Landscape Perspective. *arXiv preprint arXiv:1912.02757*, 2019.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning (ICML)*, 2018.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: representing model uncertainty in deep learning. In *International Conference on Machine Learning (ICML)*, 2016.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *International Conference on Machine Learning (ICML)*, 2018.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-Learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2016.

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 (ICLR)*, 2022.

Pavel Izmailov, Sharad Vikram, Matthew D. Hoffman, and Andrew Gordon Wilson. What Are Bayesian Neural Network Posteriors Really Like? In *International Conference on Machine Learning (ICML)*, 2021.

Tianying Ji, Yu Luo, Fuchun Sun, Xianyuan Zhan, Jianwei Zhang, and Huazhe Xu. Seizing Serendipity: Exploiting the Value of Past Success in Off-Policy Actor-Critic. In *International Conference on Machine Learning (ICML)*, 2024.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline Reinforcement Learning with Implicit Q-Learning. In *International Conference on Learning Representations (ICLR)*, 2022.

Arsenii Kuznetsov, Pavel Shvechikov, Alexander Grishin, and Dmitry Vetrov. Controlling Overestimation Bias with Truncated Mixture of Continuous Distributional Quantile Critics. In *International Conference on Machine Learning (ICML)*, 2020.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Qingfeng Lan, Yangchen Pan, Alona Fyshe, and Martha White. Maxmin Q-learning: Controlling the Estimation Bias of Q-learning. In *International Conference on Learning Representations (ICLR)*, 2020.

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 (ICML)*, 2021.

Jiachen Li, Shuo Cheng, Zhenyu Liao, Huayan Wang, William Yang Wang, and Qinxun Bai. Off-policy Reinforcement Learning with Optimistic Exploration and Distribution Correction. *arXiv preprint 2110.12081*, 2022.

Evan Zheran Liu, Aditi Raghunathan, Percy Liang, and Chelsea Finn. Decoupling Exploration and Exploitation for Meta-Reinforcement Learning without Sacrifices. In *International Conference on Machine Learning (ICML)*, 2021.

Yu Luo, Tianying Ji, Fuchun Sun, Jianwei Zhang, Huazhe Xu, and Xianyuan Zhan. Offline-Boosted Actor-Critic: Adaptively Blending Optimal Historical Behaviors in Deep Off-Policy RL. In *International Conference on Machine Learning (ICML)*, 2024.

Max Sobol Mark, Archit Sharma, Fahim Tajwar, Rafael Rafailov, Sergey Levine, and Chelsea Finn. Offline Retraining for Online RL: Decoupled Policy Learning to Mitigate Exploration Bias. *arXiv preprint arXiv:2310.08558*, 2023. URL `https://arxiv.org/abs/2310.08558`.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous Methods for Deep Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 2016.

Ted Moskovitz, Jack Parker-Holder, Aldo Pacchiano, Michael Arbel, and Michael I. Jordan. Tactical Optimism and Pessimism for Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Vinod Nair and Geoffrey E. Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. In *International Conference on Machine Learning (ICML)*, 2010.

Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, Regularized, Optimistic: scaling for compute and sample-efficient continuous control. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Ben Norman and Jeff Clune. First-Explore, then Exploit: Meta-Learning to Solve Hard Exploration-Exploitation Trade-Offs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Brendan O'Donoghue, Ian Osband, Rémi Munos, and Volodymyr Mnih. The Uncertainty Bellman Equation and Exploration. In *International Conference on Machine Learning (ICML)*, 2018.

Motoki Omura, Kazuki Ota, Takayuki Osa, Yusuke Mukuta, and Tatsuya Harada. Gradual Transition from Bellman Optimality Operator to Bellman Operator in Online Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 2025.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

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

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Lukas Schäfer, Filippos Christianos, Josiah P. Hanna, and Stefano V. Albrecht. Decoupled Reinforcement Learning to Stabilise Intrinsically-Motivated Exploration. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2022.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. In *Journal of Machine Learning Research (JMLR)*, 2014.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL `http://incompleteideas.net/book/the-book-2nd.html`.

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, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

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. In *Software Impacts*, 2020.

William F. Whitney, Michael Bloesch, Jost Tobias Springenberg, Abbas Abdolmaleki, Kyunghyun Cho, and Martin Riedmiller. Decoupled Exploration and Exploitation Policies for Sample-Efficient Reinforcement Learning. *arXiv preprint 2101.09458*, 2021. URL `http://arxiv.org/abs/2101.09458`.

Andrew Gordon Wilson and Pavel Izmailov. Bayesian Deep Learning and a Probabilistic Perspective of Generalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

A Preliminary Experiment Details

In the preliminary toy experiment in Section 3, we compared SARSA-based updates in actor-critic models using the environment depicted in left sub-figure of Figure 2. We compared the optimistic UCB Q-values and the annealing counterpart to the default SARSA-based update. The critic stores the estimated Q-values for each state-action pair in a table and updates them based on either UCB or SARSA. The policy manages logits for each state-action pair in a table, calculates the distribution using the softmax function, and samples actions from this distribution. The policy may be expressed as follows:

$$\pi_\theta(a \mid s) = \frac{\exp\big(\theta_{s,a}\big)}{\sum_b \exp\big(\theta_{s,b}\big)} \tag{14}$$

The update of these logits is performed using the policy gradient method, with the update equation given as follows:

$$\begin{aligned} \theta \;\leftarrow\; \theta \;+\; \alpha\, \nabla_\theta \log \pi_\theta(a_t \mid s_t)\, Q(s_t, a_t), \\ \nabla_{\theta_{s,a'}} \log \pi_\theta(a \mid s) = \delta_{a,a'} \;-\; \pi_\theta(a' \mid s), \end{aligned} \tag{15}$$

where $\delta_{a,a'}$ is the Kronecker delta. The step size α used for updates in both the critic and the policy was set to 5e-4. This step size was chosen because it yields smoother learning curves. Whilst increasing the step size accelerates learning, the observation that UCB-based updates can converge faster than SARSA-based updates remained consistent. The initial state was randomly selected from s_0, s_1, and s_2 with equal probability. Additionally, a probability of 10% of taking random actions, akin to an ϵ-greedy policy, was introduced. We ran the toy example 20 times, and plotted the mean and the standard deviation of Q-tables.

B Experiments Implementation Details & Hyperparameters

Our implementation and experiments were implemented in JAX (Bradbury et al., 2018). The important libraries we used in our experiments were: JAX (Bradbury et al., 2018), MuJoCo (Todorov et al., 2012), DeepMind Control Suite (Tunyasuvunakool et al., 2020), gym (Brockman et al., 2016) and Gymnasium Robotics (de Lazcano et al., 2024). We plan to release source code upon acceptance.

Shared across all algorithms. The replay buffer size is set to 10^6, and the discount factor γ is set to 0.99. The target update rate τ for target network(s) is 0.005. The initial random-data collection phase lasts 10,000 environment steps. To ensure a fair comparison, all methods, unless specified otherwise, employ a batch size of 256, and all neural networks used two hidden layers consisting of 256 units each. All methods use ReLU (Nair & Hinton, 2010) as the activation function. We use Adam (Kingma & Ba, 2015) as optimiser for all neural networks with the learning rate set to 0.0003. Unless otherwise stated, all methods, including BOXD and BRO, use an update-to-data (UTD) ratio of 1 (one gradient update per environment step), so that all algorithms are compared under a comparable update budget.

XQC, SAC and TD3. For baselines, XQC (Palenicek et al., 2026), SAC (Haarnoja et al., 2018) and TD3 (Fujimoto et al., 2018) are all implemented closely following public repositories such as JAXRL `https://github.com/ikostrikov/jaxrl`, high-replay-ratio (D'Oro et al., 2023) `https://github.com/proceduralia/high_replay_ratio_continuous_control` and annealing-q-learning (Omura et al., 2025) (`https://github.com/motokiomura/annealed-q-learning`) and XQC official source code (`https://github.com/danielpalenicek/xqc`). XQC is a recent stochastic actor-critic algorithm that improves deep RL optimisation by making the critic update better conditioned, and we use it as the strongest base algorithm in our experiments. We use the default hyperparameters provided in these algorithms.

BOXD. Our implementation closely follows DroQ, using XQC, SAC and TD3 as base algorithms. For all Q-functions we add dropout (Srivastava et al., 2014) and layernorm (Ba et al., 2016) sequentially after each linear layer (i.e. linear $\rightarrow$ dropout $\rightarrow$ layernorm $\rightarrow$ activation), except for the last linear layer. Thus, "preserving the task objective" refers to preserving its learning objective and target aggregation, only changing its critic architecture. For our proposed method, in all experiments the dropout probability rate is set to

$p = 0.001$ and is the same across all tasks, unless specified otherwise. Unless otherwise stated, we use the default number of exploration critics $n = 2$ for all tasks and base algorithms, and vary n only in the ablation study (Appendix F). The dynamic-switcher mechanism is defined in Subsection 4.2, and its experimental settings are reported in Subsection 5.1.

XQC+BOXD. The task branch π^{task} follows the original XQC objectives but add dropout and layernorm, preserving its exploitation objective and optimisation procedure. The exploration branch π^{explore} retains the original XQC categorical distributional critic, fixed return support, entropy-regularised Bellman transform, projection onto the fixed support, categorical cross-entropy loss, and actor update. BOXD changes only the selection of the categorical distribution used to construct the exploration-critic target.

Specifically, each of the n exploration critics predicts a categorical return distribution over a fixed support $\{z_m\}_{m=1}^{M}$. We first convert each distribution to its scalar expected return,

$$Q_i(s', a') = \sum_{m=1}^{M} z_m \, p_{i,m}(s', a'), \qquad \boldsymbol{p}_i(s', a') = \text{softmax}(\boldsymbol{\ell}_i(s', a')) \,, \tag{16}$$

where $\boldsymbol{\ell}_i$ and $\boldsymbol{p}_i$ denote the predicted logits and categorical probabilities of critic i, respectively. For each transition in a mini-batch, BOXD selects the exploration critic with the largest expected return,

$$i^\star = \arg\max_{i \in \{1,\dots,n\}} Q_i(s', a'). \tag{17}$$

The complete categorical distribution $\boldsymbol{p}_{i^\star}$ predicted by the selected critic is then passed to the otherwise unchanged XQC target-construction procedure. Thus, the maximum is taken only over the scalar expected returns used for critic selection, rather than element-wise over atoms, logits, or probabilities.

Given $a' \sim \pi^{\text{explore}}(\cdot \mid s')$, each support atom of the selected distribution is transformed using the XQC entropy-regularised Bellman operator,

$$\widetilde{z}_m = \text{clip}\big(r + \gamma d \left[z_m - \alpha \log \pi^{\text{explore}}(a' \mid s')\right], v_{\min}, v_{\max}\big) \,, \tag{18}$$

where d is the continuation indicator. As in XQC, the probability mass associated with each transformed atom is linearly projected onto the two neighbouring atoms of the fixed support, and every exploration critic is trained against this shared target distribution using categorical cross-entropy. When multiple dropout samples are used for one critic, their logits are averaged before converting them to probabilities, computing the expected returns, and selecting the maximising critic. During data collection, the dynamic switcher selects between the XQC task policy and the optimistic exploration policy. We use the single fixed XQC switcher configuration reported in Subsection 5.1, selected from the SAC+BOXD settings rather than through a separate XQC-specific search.

TD3+BOXD. For the deterministic-policy variant, the task branch retains the standard TD3 actor update, target-policy smoothing, delayed policy updates, and conservative clipped double-Q target. The exploration branch uses the same TD3 learning procedure except that its Bellman target takes the maximum over the dropout-enabled exploration critics, as defined in Equation (8). Dropout and layer normalisation are added to all critics as in the other BOXD variants.

All TD3+BOXD results, including the main TD3 comparison and the TD3-based ablations, use the non-adaptive time-step annealing strategy in Equation (13), not the agreement-based dynamic switcher. At every environment step, the acting policy is sampled using Equation (10) with the scheduled value of λ_t^{mix}; the same selection procedure is used during evaluation. The schedule starts with $\lambda_0^{\text{mix}} = 0$, increases by 0.1 after each tenth of the training horizon, and reaches 0.9 in the final tenth. Consequently, the TD3 experiments test whether the optimistic dual-policy branch transfers to a deterministic base algorithm under a fixed policy-selection schedule; they do not evaluate the adaptive-switcher contribution.

OAC. For OAC (Ciosek et al., 2019), which was officially implemented in PyTorch (Paszke et al., 2019), we reimplemented into JAX. For hyperparameters, we follow the publicly released official implementation (`https://github.com/microsoft/oac-explore/`) and use $\beta_{ub} = 4.66$.

DERL framework based algorithm. For DERL, the original work experimented on discrete tasks with A2C (Mnih et al., 2016) as the base algorithm. We adapt their framework of disentangling exploration and exploitation policies to continuous control, and add intrinsic reward bonuses to the exploration policy. The policies are trained with TD3 and SAC as base algorithms. Specifically, in our experiments we add RND (Burda et al., 2019) as the intrinsic reward, and experiment with various coefficients $= \{0.1, 1.0, 5.0\}$ when adding the bonus to the extrinsic reward (from the environment). Only the best result is presented.

SUNRISE. SUNRISE (Lee et al., 2021) is an ensemble-based framework built on SAC (Haarnoja et al., 2018) that combines uncertainty-weighted Bellman backups with UCB-style action selection. We re-implement it in JAX following the official implementation (`https://github.com/pokaxpoka/sunrise`). All shared network and optimisation hyperparameters follow the common setup described above. Following the original work, we train an ensemble of $N = 5$ SAC agents with bootstrapped masking, weight the Bellman backup/critic loss using a confidence weight computed from ensemble disagreement using a temperature of $T = 20$, and select exploratory actions with a UCB weight of $\lambda = 1$. All hyperparameters are consistent with the original work.

BRO. We use the official implementation of BRO (Nauman et al., 2024), using mostly default hyperparameters except a batch size of 256. BRO also includes multiple techniques, such as distributional RL by quantile regression and resets. We retain all these.

B.1 Double-EMA Dynamic Switcher

Algorithm 1 implements the dynamic switcher directly using the two-stage EMA in Equation (12). At switcher-update index k, let R_k denote the smoothed readiness score. The code initialises this EMA from its first observation:

$$\begin{aligned} R_1 &= S_1^{\text{agreement}}, \\ R_k &= (1-\beta_{\text{EMA}})R_{k-1} + \beta_{\text{EMA}} S_k^{\text{agreement}}, \qquad k \geq 2 \end{aligned} \tag{19}$$

The mixing coefficient is smoothed separately using α_{switcher}:

$$\lambda_k^{\text{mix}} = (1-\alpha_{\text{switcher}})\lambda_{k-1}^{\text{mix}} + \alpha_{\text{switcher}} R_k, \qquad \lambda_0^{\text{mix}} = 0 \tag{20}$$

The first stage suppresses short-term variation in the agreement score, while the second stage controls how quickly the acting-policy probability follows the readiness score. Thus, β_{EMA} controls agreement-score smoothing and α_{switcher} controls the switching rate.

Substituting this identity and Equation (19) into Equation (20) yields the Equation (12). Its first nonzero value follows directly from the code initialisation:

$$\lambda_1^{\text{mix}} = \alpha_{\text{switcher}} S_1^{\text{agreement}} \tag{21}$$

Assume under maximal readiness, $S_k^{\text{agreement}} = 1$ for every update. Because $R_1 = 1$, the readiness EMA remains at one and the mixing coefficient has the upper envelope

$$\lambda_K^{\text{mix}} = 1 - (1-\alpha_{\text{switcher}})^K \tag{22}$$

For $\alpha_{\text{switcher}} = 0.05$, this envelope is approximately 0.60 after 18 switcher updates and 0.81 after 32 updates, showing that the mixing-coefficient values in Figure 7 are feasible under the implemented initialisation.

C Evaluation Metric Computation

For each task and algorithm, we evaluate 10 random seeds, with the return for one seed defined as its average over 20 evaluation episodes. For algorithm A, let $R_{t,s}^A(x)$ denote the return for task t, seed s, at environment interaction step x, and let m_t^A be the number of available seeds ($m_t^A = 10$ in our experiments). When considering one fixed algorithm, we suppress the superscript and write $R_{t,s}(x)$ and m_t. The main

benchmark contains $T = 14$ tasks, while the switcher comparison over the non-dog tasks uses $T = 10$. All confidence intervals use a percentile bootstrap with $B = 5000$ replicates. Only seeds are resampled within each task and algorithm; the set of tasks is held fixed. We write Q_α for the empirical α-quantile, implemented with `numpy.percentile`.

C.1 Learning-Curve Aggregation

We first trim once across the observed seeds within each task to obtain the per-task point estimate

$$I_t(x) = \mathrm{IQM}\left(\{R_{t,s}(x)\}_{s=1}^{m_t}\right) \tag{23}$$

The plotted aggregate gives equal weight to each of the T tasks:

$$\widehat{M}(x) = \frac{1}{T}\sum_{t=1}^{T} I_t(x) \tag{24}$$

For each bootstrap replicate $b \in \{1, \ldots, B\}$ and each task, we independently draw m_t seed indices with replacement:

$$S_{t,j}^{(b)} \overset{\text{iid}}{\sim} \mathrm{Uniform}\{1, \ldots, m_t\}, \qquad j = 1, \ldots, m_t \tag{25}$$

We then re-sort and re-trim each bootstrapped task sample. With $k_t = \lfloor 0.25 m_t \rfloor$, this gives

$$I_t^{(b)}(x) = \mathrm{IQM}\left(\left\{R_{t,S_{t,j}^{(b)}}(x)\right\}_{j=1}^{m_t}\right) = \frac{1}{m_t - 2k_t}\sum_{i=k_t+1}^{m_t-k_t}\left(R_t^{(b)}(x)\right)_{(i)} \tag{26}$$

We average across tasks before taking percentiles, producing one aggregate value for each bootstrap replicate:

$$M^{(b)}(x) = \frac{1}{T}\sum_{t=1}^{T} I_t^{(b)}(x) \tag{27}$$

The shaded 95% percentile-bootstrap confidence interval is

$$\mathrm{CI}_{95\%}(x) = \left[Q_{0.025}\left(\left\{M^{(b)}(x)\right\}_{b=1}^{B}\right), Q_{0.975}\left(\left\{M^{(b)}(x)\right\}_{b=1}^{B}\right)\right] \tag{28}$$

The trim count k_t is computed when the IQM is re-evaluated in each replicate. Seed indices are sampled independently for each task and each environment-interaction step x, rather than reusing one bootstrap draw across the whole learning curve. Thus, the shaded interval represents seed-level uncertainty for the evaluated tasks, not uncertainty from generalising to other tasks.

C.2 Probability of Improvement

For a comparison between algorithms X and Y, we compute the task-stratified probability of improvement (POI) as

$$\widehat{P}_{X>Y}(x) = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{m_t^X m_t^Y}\sum_{i=1}^{m_t^X}\sum_{j=1}^{m_t^Y}\left[\mathbf{1}\big(R_{t,i}^X(x) > R_{t,j}^Y(x)\big) + \frac{1}{2}\mathbf{1}\big(R_{t,i}^X(x) = R_{t,j}^Y(x)\big)\right]. \tag{29}$$

The POI can be understood with an intuitive explanation as: "If one grab one random run of X and one random run of Y, how often does X win?" A score of 0.5 means a coin flip (no difference), above 0.5 means X is usually better, below 0.5 means Y is.

Each task receives equal weight, and ties contribute one half. For bootstrap replicate b, runs are independently resampled within each task and algorithm:

$$S_{t,i}^{X,(b)} \overset{\text{iid}}{\sim} \text{Uniform}\{1,\dots,m_t^X\}, \qquad S_{t,j}^{Y,(b)} \overset{\text{iid}}{\sim} \text{Uniform}\{1,\dots,m_t^Y\}. \tag{30}$$

The corresponding bootstrap replicate is

$$\widehat{P}_{X>Y}^{(b)}(x) = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{m_t^X m_t^Y}\sum_{i=1}^{m_t^X}\sum_{j=1}^{m_t^Y}\left[\mathbf{1}\left(R^X_{t,S^{X,(b)}_{t,i}}(x) > R^Y_{t,S^{Y,(b)}_{t,j}}(x)\right) + \frac{1}{2}\mathbf{1}\left(R^X_{t,S^{X,(b)}_{t,i}}(x) = R^Y_{t,S^{Y,(b)}_{t,j}}(x)\right)\right]. \tag{31}$$

The 95% confidence interval is

$$\text{CI}^{\text{POI}}_{95\%}(x) = \left[Q_{0.025}\left(\left\{\widehat{P}_{X>Y}^{(b)}(x)\right\}_{b=1}^{B}\right), Q_{0.975}\left(\left\{\widehat{P}_{X>Y}^{(b)}(x)\right\}_{b=1}^{B}\right)\right]. \tag{32}$$

D BOXD with XQC

XQC (Palenicek et al., 2026) is a recent state-of-the-art actor-critic algorithm. In our experiments, XQC serves as the strongest base algorithm considered in the main text, because unlike SAC or TD3, it already incorporates recent algorithmic improvements. This setting is especially important, since improvements over SAC and TD3 show that BOXD can strengthen standard baselines, whereas the XQC experiments test whether dual-policy optimistic exploration can still provide complementary gains when the underlying learner is already strong.

The XQC-specific implementation of the optimistic categorical target is described in Appendix B.

The results in Figure 8 show that XQC+BOXD provides moderate gains over XQC, the strongest base algorithm we tested, on several tasks and is competitive with BRO on aggregate in Figure 3, with higher IQM returns on several hopper and humanoid tasks. The improvement is smaller than the gains obtained when BOXD is added to SAC or TD3, which is expected because XQC is already a substantially stronger base algorithm. Nevertheless, these additional gains suggest that the benefit of dual-policy optimistic exploration is not limited to weaker baselines. Instead, it can complement recent advances in actor-critic optimisation by improving the data collected for learning.

We also include the detailed POI plots of Figure 5 in Figure 9. Squares show point estimates and horizontal bands show 95% percentile-bootstrap confidence intervals using within-task seed resamples. The vertical line at 0.5 denotes parity; larger values favour the reference algorithm.

E BOXD with a Deterministic-Policy Base Algorithm (TD3)

The experiments in the main text build BOXD on stochastic-policy base algorithms (SAC and XQC). Because BOXD only changes how the exploration-critic target is formed and how the acting policy is selected, the mechanism is agnostic to whether the underlying actor-critic uses a stochastic or a deterministic policy. To demonstrate this, we additionally apply BOXD to TD3 (Fujimoto et al., 2018), a deterministic-policy algorithm, which we denote TD3+BOXD.

The TD3-specific implementation and policy-selection strategy are described in Appendix B.

Per-task performance. The per-task IQM returns for TD3+BOXD across the 14 DM Control tasks are reported in Figure 10. TD3+BOXD improves over the TD3 baseline and remains competitive with the heavily engineered BRO, mirroring the gains observed for SAC+BOXD in the main text. As with the stochastic variant, the improvements are most pronounced on tasks with larger state dimensions that benefit

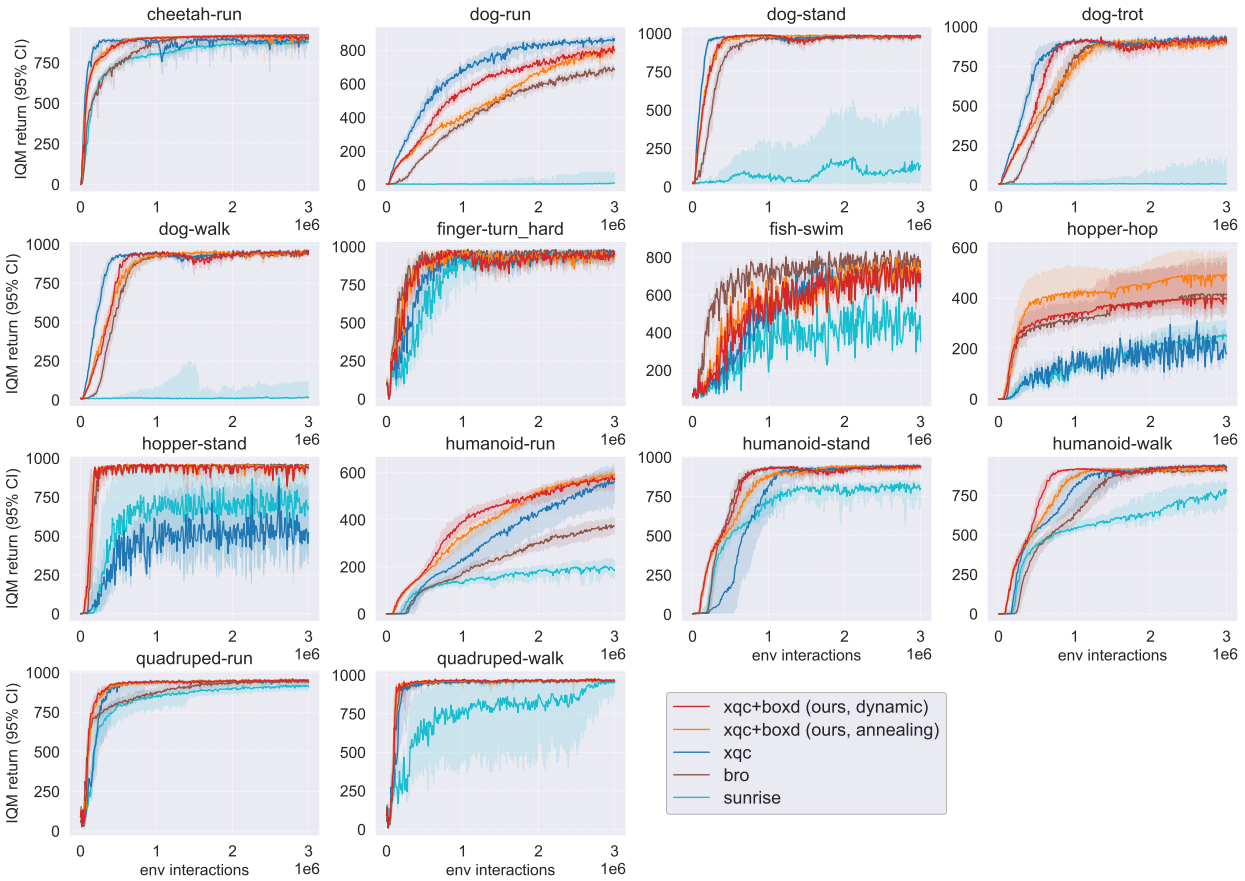

Figure 8: Learning curves of the per-task IQM returns comparing XQC+BOXD (red, orange) with XQC (blue) and other state-of-the-art baselines. Adding BOXD to XQC provides additional gains on several tasks, indicating that optimistic dual-policy exploration can still complement an already strong actor-critic base algorithm.

from stronger exploration, such as hopper-hop and the humanoid tasks, whereas on cheetah-run BOXD slightly underperforms, as additional optimism provides little benefit there.

Sensitivity to design choices. The ablations in Appendix F separate the effects of the optimistic target from those of policy selection. For TD3+BOXD, we vary the number of exploration critics n (Figure 11, top) and compare maximum with mean aggregation (Figure 13), while retaining the fixed time-step annealing schedule. Increasing n has a task-dependent effect, and maximum aggregation provides large gains on hopper-hop and humanoid-run but is not uniformly superior on every task. Taken together, these results show that the optimistic dual-policy branch transfers to the deterministic TD3 base algorithm under a fixed policy-selection schedule, while the adaptive switcher is evaluated only with stochastic-policy variants.

F Ablation Studies

Unless specified otherwise, the ablation studies below are conducted with TD3+BOXD (introduced in Appendix E) to avoid hyperparameter tuning, with several results also reported for SAC+BOXD.

We organise the ablations around three components of the exploration target. First, we vary the number of exploration critics n to study how the strength of optimism affects performance. Second, we vary the dropout rate to examine sensitivity to the stochasticity within each critic. Third, we replace the maximum with the mean to isolate whether optimistic aggregation, rather than the critic ensemble alone, is responsible

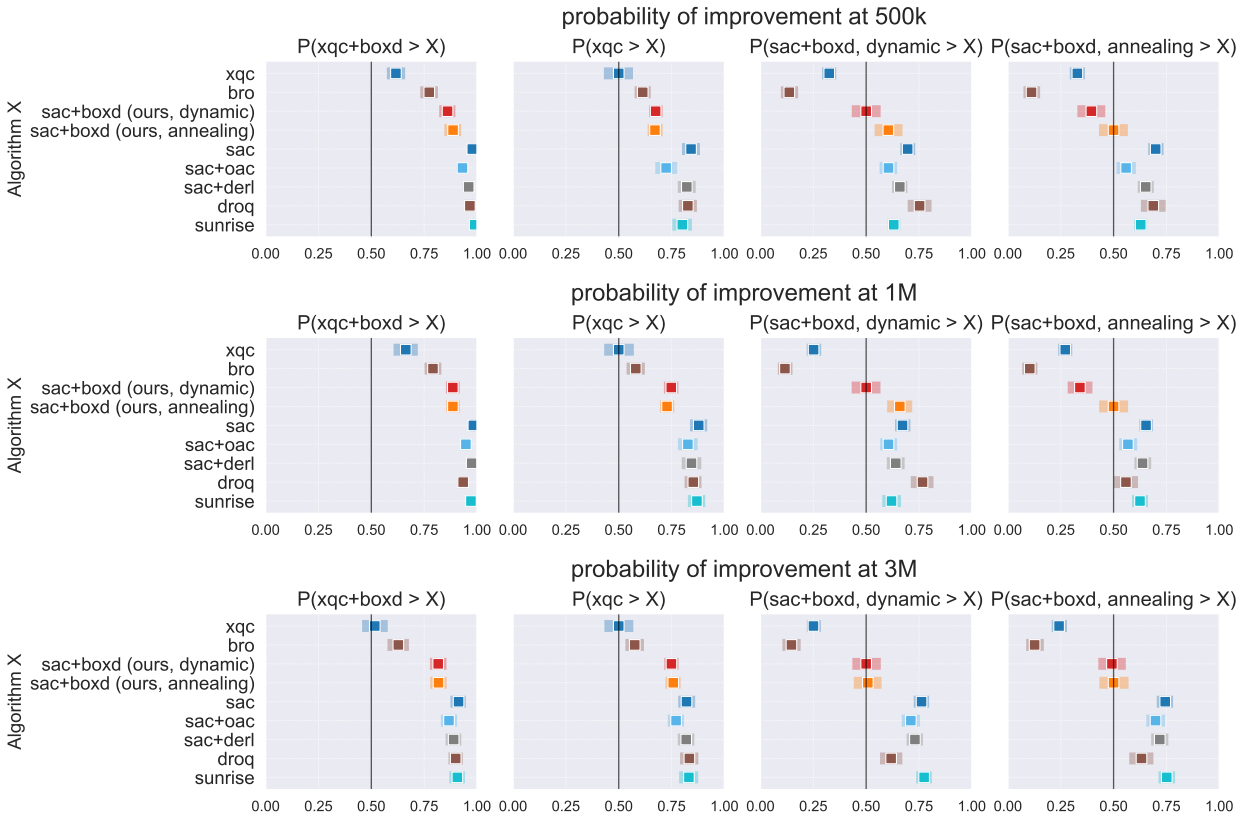

Figure 9: Probability of improvement across the 14 DM Control tasks at 500k (top), 1M (middle), and 3M (bottom) environment steps. From left to right, the panels use XQC+BOXD with dynamic switcher, XQC, SAC+BOXD with the dynamic switcher, and SAC+BOXD with annealing as the reference algorithm, and compare it against algorithm X on the vertical axis. At 3M steps, XQC is relatively close but still compares favourably to BRO. The XQC+BOXD point estimate is above 0.5 against both, but is near parity with XQC. Both SAC+BOXD variants outperform SAC-family baselines, and remain competitive with XQC and BRO.

for the improvement. Together, these experiments test the strength, stochastic source, and aggregation rule of BOXD's optimistic target.

Q: Ablation: How does the number of Q-functions n affect performance?

A: It depends on the tasks. As is standard in contemporary actor-critic algorithms, using $n = 2$ is a good starting point.

The number of critics greatly affects the UCB trade-off parameter c_n, as depicted in Equation (7). We include in Table 1 the approximate values of c_n for $n = \{2, 3, 4, 10\}$. These values follow Blom's approximation (Equation (7)); for $n = 2$ this yields $c_2 \approx 0.589$, close to the exact value $1/\sqrt{\pi} \approx 0.564$ given in Equation (2). We additionally train $n = \{3, 4, 10\}$ for our algorithm based on TD3 and SAC, using the simpler annealing strategy for comparison. For hopper-hop, more optimism may be beneficial. However, for humanoid tasks, too much exploration is disadvantageous in both TD3-based and SAC-based results. The results for TD3-based and SAC-based BOXD are depicted in Figure 11. These experiments were performed using the simpler annealing switching strategy to isolate the effects more precisely.

Q: How does the dropout rate affect performance?

A: In the main experiments, we use a consistent dropout rate of $p = 0.001$. This value follows the small dropout rates investigated by DroQ (Hiraoka et al., 2022), where dropout is used for critic regularisation.

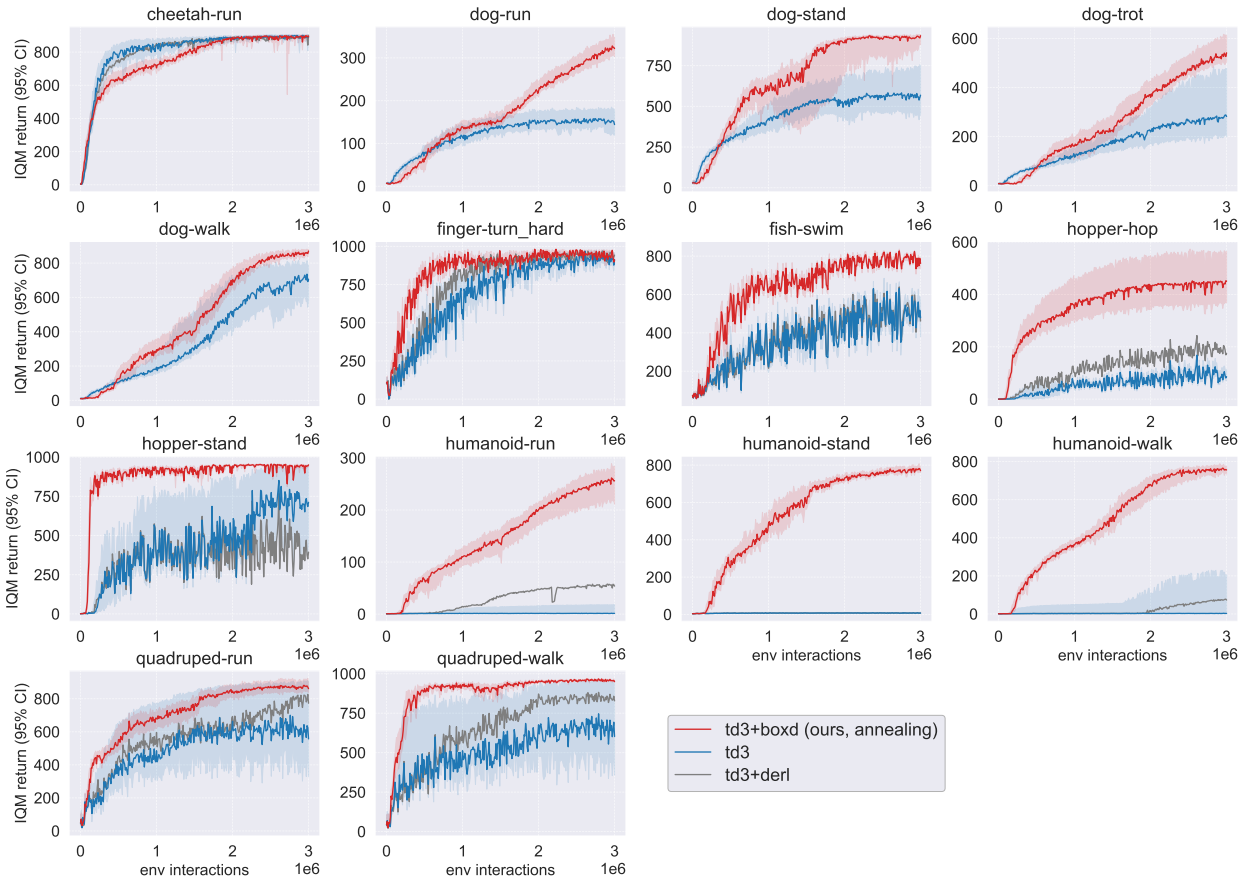

Figure 10: Learning curves of the per-task IQM returns comparing BOXD built on TD3. Adding BOXD to TD3 provides additional gains on several tasks, indicating that optimistic dual-policy exploration can also improve deterministic policy based algorithms.

Table 1: Approximate values for UCB trade-off parameter c_n, depending on the number of critics n.

n	2	3	4	10
c_n	0.589	0.869	1.049	1.547

To evaluate whether BOXD is sensitive to this choice, we compare SAC+BOXD with the dynamic switcher using $p \in \{0, 0.001, 0.005, 0.01\}$ in Figure 12.

Performance is generally stable for small dropout rates, although the effect is task-dependent. The default $p = 0.001$ is competitive across all five tasks, and $p = 0.005$ usually produces similar learning curves. Increasing the rate to $p = 0.01$ can be detrimental, most clearly on hopper-hop, suggesting that excessive stochasticity can make the optimistic critic target too noisy. At the same time, removing dropout entirely remains competitive and performs best on cheetah-run. This indicates that dropout is not the only source of useful variation in BOXD: independently initialised exploration critics already provide cross-critic diversity, while a small dropout rate adds within-critic stochasticity and regularisation. We therefore use $p = 0.001$ as a conservative default that performs robustly across tasks without injecting excessive noise.

Q: How important is maximum aggregation?

A: To isolate the contribution of our optimistic aggregation, we replace the maximum with the mean while keeping the critic ensemble and all other training settings unchanged.

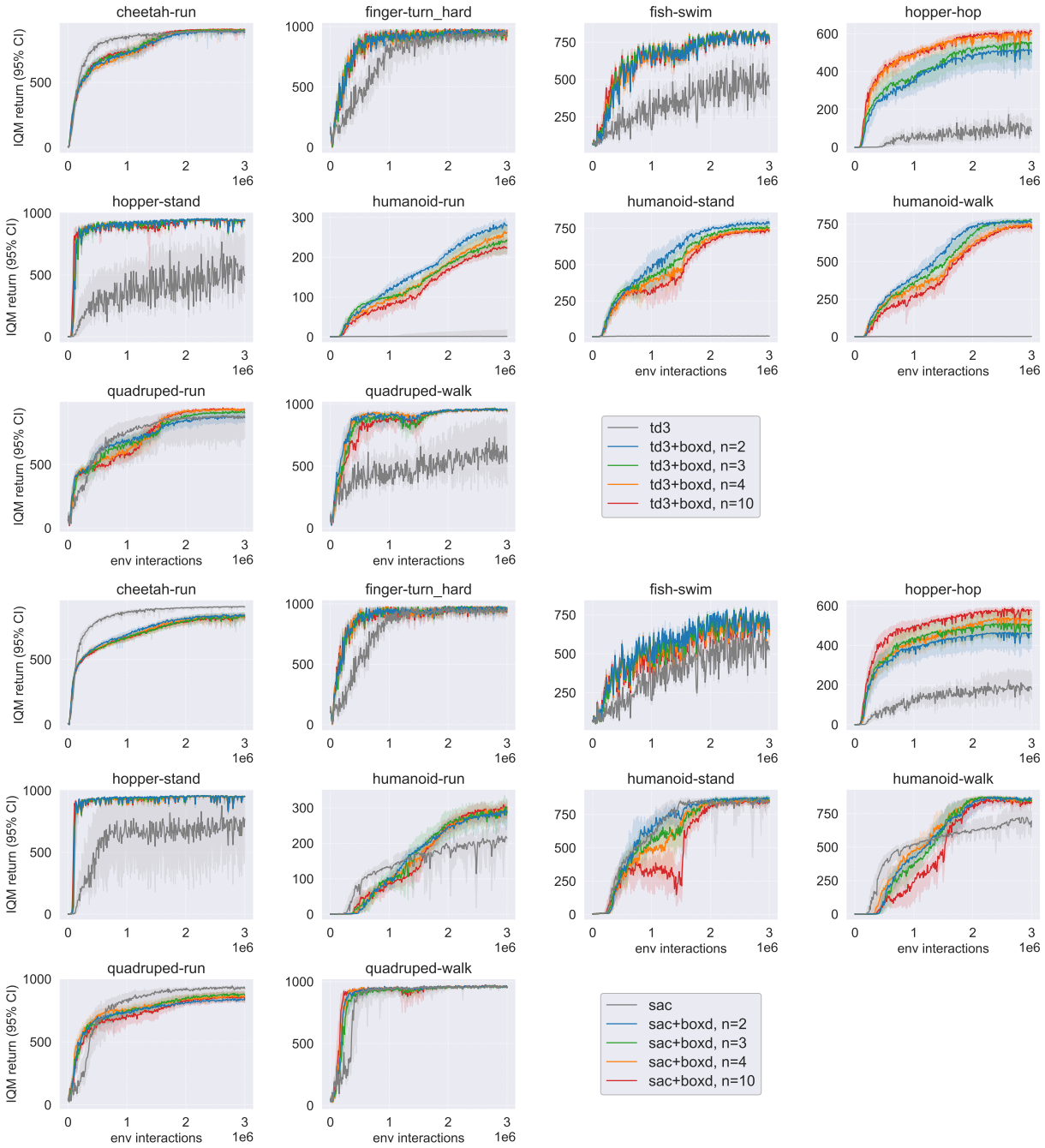

Figure 11: **Top:** Ablation results based on TD3 of the number of Q-functions n. **Bottom:** Ablation results based on SAC of the number of Q-functions n. Increasing n may induce more optimism, which may be beneficial in some tasks (namely hopper-hop) but disadvantageous for humanoid tasks.

As shown in Figure 13, maximum aggregation provides substantial gains on hopper-hop and humanoid-run, where the mean variant saturates at a lower return. The two aggregation rules perform similarly on cheetah-run and humanoid-walk, while the mean is moderately better on humanoid-stand. Thus, the maximum is not uniformly superior on every task, but its large gains on the more exploration-sensitive tasks show that the critic ensemble alone is not sufficient: using its stochastic estimates optimistically is an important part of BOXD's exploration mechanism. This comparison uses TD3+BOXD with the fixed time-step annealing policy-selection schedule in Equation (13), rather than the adaptive switcher.

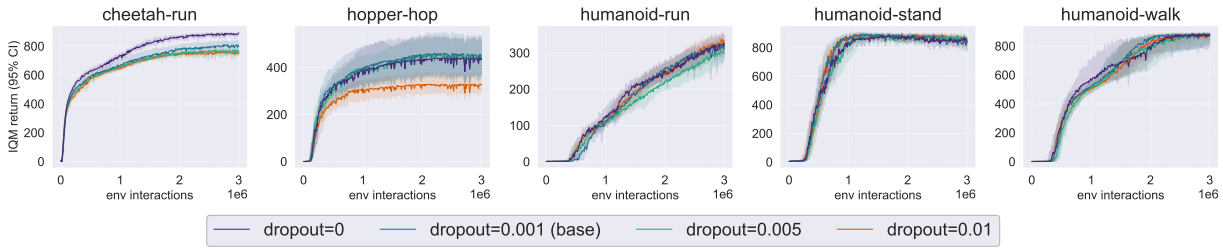

Figure 12: Ablation of the dropout rate p for SAC+BOXD with the dynamic switcher on five representative DM Control tasks. Small rates ($p = 0.001$ and $p = 0.005$) generally produce similar performance, while $p = 0.01$ can degrade learning, most clearly on hopper-hop. The no-dropout variant remains competitive and performs best on cheetah-run, indicating that independent critic initialisation already provides substantial diversity. We use $p = 0.001$ as a robust default across tasks.

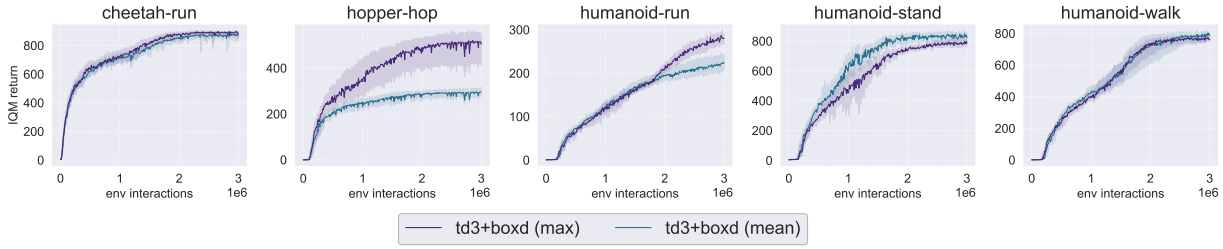

Figure 13: Comparison of maximum and mean aggregation for TD3+BOXD using the fixed time-step annealing policy-selection schedule. All other settings are held constant. Maximum aggregation substantially improves performance on hopper-hop and humanoid-run, performs similarly to the mean on cheetah-run and humanoid-walk, and is moderately worse on humanoid-stand. These results indicate that optimistic aggregation is particularly beneficial on tasks that require stronger exploration.

Q: How does α_{switcher} affect performance?

A: The switching rate α_{switcher} controls how quickly λ^{mix} follows the smoothed agreement target in Equation (12); it changes the speed of the transition from the exploration policy to the task policy, but not the definition of the target itself. We vary $\alpha_{\text{switcher}} \in \{0.02, 0.03, 0.04, 0.05, 0.06, 0.07\}$ on five representative tasks with SAC+BOXD, holding the other settings fixed. Figure 14 reports both the resulting per-task IQM returns and the corresponding λ^{mix} trajectories.

As expected, larger values make λ^{mix} rise more rapidly, whereas smaller values retain exploration-policy data collection for longer. The return curves show that no single switching rate dominates every task. Slower switching ($\alpha_{\text{switcher}} = 0.02$ or 0.03) gives the strongest late-training performance hopper-hop, and humanoid-run, suggesting that switching too rapidly can be harmful on tasks that benefit from prolonged optimistic exploration. In contrast, humanoid-stand and humanoid-walk reach similar final performance across most of the tested range, with larger values mainly affecting the timing of the transition. These results motivate using a moderate default while allowing a smaller switching rate for exploration-sensitive tasks.

G Policy Switcher Results

In the main manuscript, we proposed a dynamic switcher to select the acting policy π^{act} during environment interaction and reported its mean per-task IQM return. Here we provide results for each of the 10 non-dog tasks. Our proposed dynamic switcher mainly improves early-training sample efficiency rather than final performance, performing on par with a well-chosen fixed mixing such as 50-50, whereas no switching or strongly exploration-biased mixing tends to saturate the acting policy.

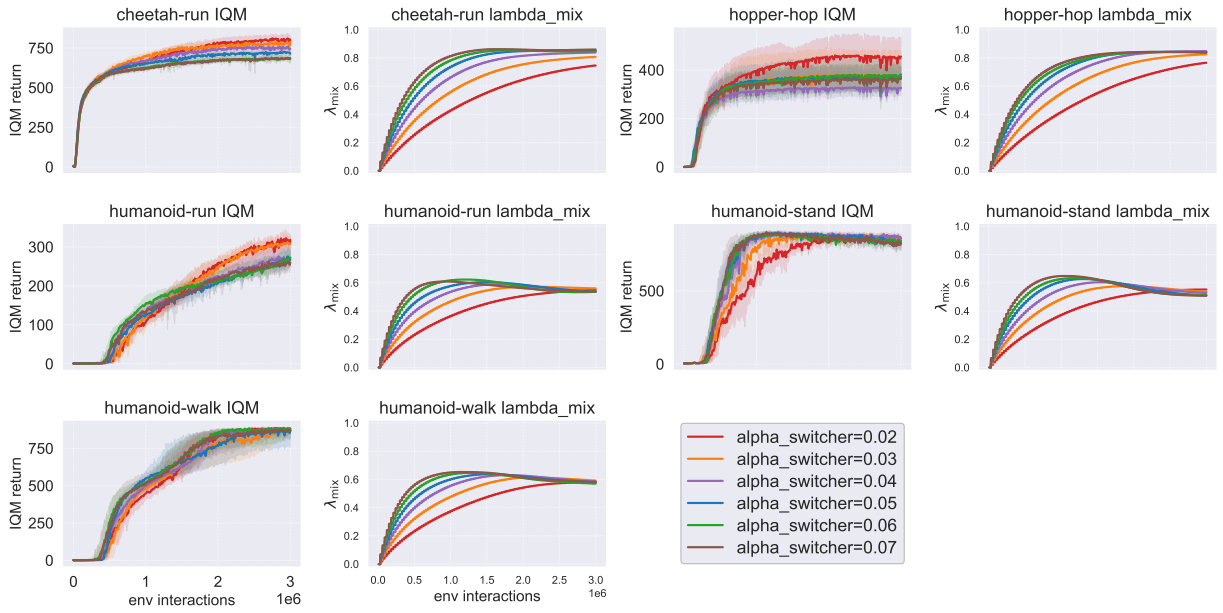

Figure 14: Ablation of the dynamic-switcher switching rate α_{switcher} for SAC+BOXD on five representative DM Control tasks. For each task, the return panel shows the per-task IQM learning curve and the adjacent panel shows the corresponding task-policy mixing coefficient λ^{mix}. Increasing α_{switcher} makes λ^{mix} track its agreement-based target more quickly and therefore switches data collection towards π^{task} earlier. The final returns are comparatively insensitive on humanoid-stand and humanoid-walk, whereas smaller switching rates perform better hopper-hop, and humanoid-run, indicating that these tasks benefit from a longer exploration-dominated phase.

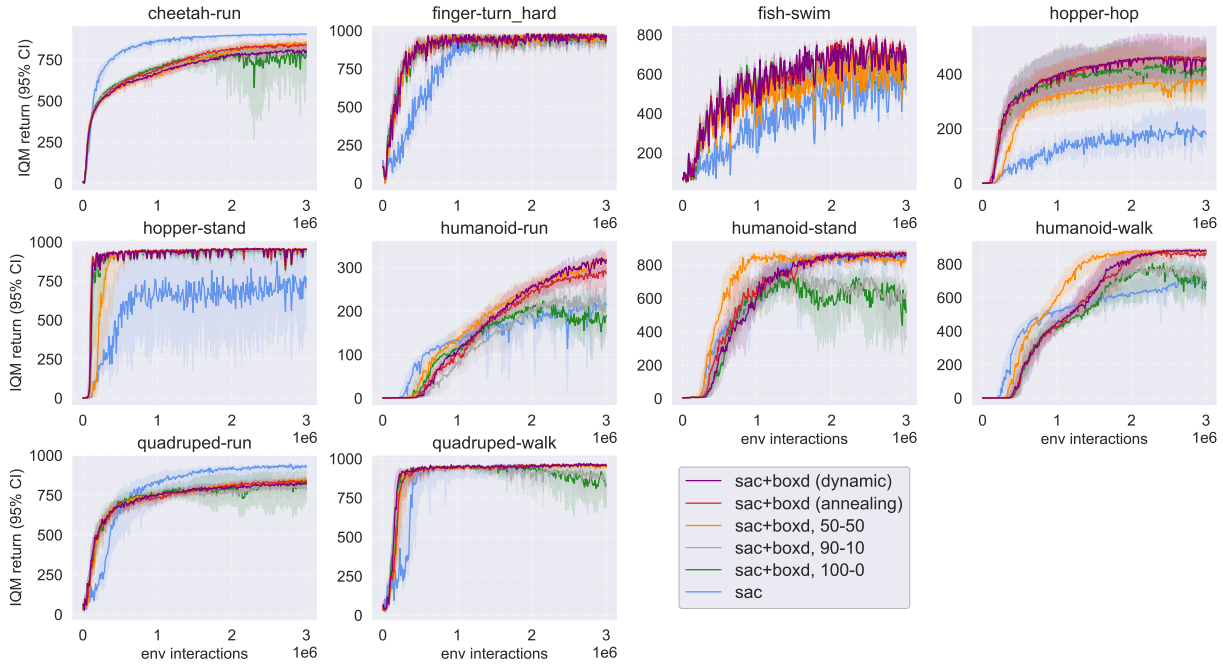

Figure 15: Learning curves of the per-task IQM returns for 10 DM Control tasks (excluding dog tasks) with SAC as the base algorithm.