Journal of Machine Learning Research 25 (2024) 1-42 Submitted 7/23; Revised 5/24; Published 9/24

Value-Distributional Model-Based Reinforcement Learning

Carlos E. Luis CARLOS@ROBOT-LEARNING.DE
Bosch Corporate Research, TU Darmstadt
Robert-Bosch-Campus 1, 71272 Renningen (Germany)

Alessandro G. Bottero ALESSANDROGIACOMO.BOTTERO@BOSCH.COM
Bosch Corporate Research, TU Darmstadt

Julia Vinogradska JULIA.VINOGRADSKA@BOSCH.COM
Bosch Corporate Research

Felix Berkenkamp FELIX.BERKENKAMP@BOSCH.COM
Bosch Corporate Research

Jan Peters JAN.PETERS@TU-DARMSTADT.DE
TU Darmstadt, German Research Center for AI (DFKI), Hessian.AI

Editor: Marc Bellemare

Abstract

Quantifying uncertainty about a policy's long-term performance is important to solve sequential decision-making tasks. We study the problem from a model-based Bayesian reinforcement learning perspective, where the goal is to learn the posterior distribution over value functions induced by parameter (epistemic) uncertainty of the Markov decision process. Previous work restricts the analysis to a few moments of the distribution over values or imposes a particular distribution shape, e.g., Gaussians. Inspired by distributional reinforcement learning, we introduce a Bellman operator whose fixed-point is the value distribution function. Based on our theory, we propose Epistemic Quantile-Regression (EQR), a model-based algorithm that learns a value distribution function. We combine EQR with soft actor-critic (SAC) for policy optimization with an arbitrary differentiable objective function of the learned value distribution. Evaluation across several continuous-control tasks shows performance benefits with respect to both model-based and model-free algorithms. The code is available at `https://github.com/boschresearch/dist-mbrl`.

Keywords: Model-Based Reinforcement Learning, Bayesian Reinforcement Learning, Distributional Reinforcement Learning, Uncertainty Quantification, Quantile Regresion.

1. Introduction

Reinforcement learning (RL) tackles optimal decision-making in an unknown Markov Decision Process (MDP) (Sutton and Barto, 2018). Uncertainty is at the heart of the RL problem: on one hand, aleatoric uncertainty refers to the stochasticity in the MDP transitions and the RL agent's action selection; on the other hand, *epistemic* uncertainty appears due to lack of knowledge about the MDP. During policy evaluation, both sources of uncertainty induce a distribution of possible returns, which should be considered for policy optimization. For instance, in high-stakes applications like medical treatments, accounting for aleatoric noise is key towards training risk-averse policies (Chow et al., 2015; Keramati et al., 2020).

Similarly, effective exploration can be achieved by proper handling of epistemic uncertainty (Deisenroth and Rasmussen, 2011; Curi et al., 2020).

Two paradigms have emerged to capture uncertainty in the predicted outcomes of a policy. First, *distributional* RL (Bellemare et al., 2017) models the aleatoric uncertainty about returns, due to the inherent noise of the decision process. In contrast, *Bayesian* RL (Ghavamzadeh et al., 2015) captures the epistemic uncertainty about the unknown *expected* return of a policy, denoted as the *value* function, due to incomplete knowledge of the MDP. As such, the distribution over outcomes from each perspective has fundamentally different meaning and utility. If we care about effective exploration of *unknown* (rather than stochastic) outcomes, then Bayesian RL is the appropriate choice of framework (Osband et al., 2019).

In this paper, we focus on the Bayesian RL setting where a posterior distribution over possible MDPs induces a distribution over value functions. The posterior over values naturally models the epistemic uncertainty about the long-term performance of the agent, which is the guiding principle behind provably-efficient exploration (Strehl and Littman, 2008; Jaksch et al., 2010). An open question remains how to effectively model and learn the posterior distribution over value functions. We approach this problem by using tools from distributional RL in the Bayesian framework. The key idea is that, for time-inhomogeneous MDPs with a tabular representation, the value distribution follows a Bellman equation from which we can derive an iterative estimation algorithm that resembles methods from distributional RL. Based on this insight, we present a novel algorithm that uses a learned value distribution for policy optimization.

Our contribution. We introduce the value-distributional Bellman equation that describes the relationship between the value distributions over successive steps. Moreover, we show that the fixed-point of the associated Bellman operator is precisely the posterior value distribution. Then, leveraging tools from distributional RL, we propose a practical algorithm for learning the *quantiles* of the value distribution function. We propose Epistemic Quantile-Regression (EQR), a model-based policy optimization algorithm that learns a distributional value function. Finally, we combine EQR with soft actor-critic (SAC) to optimize a policy for any differentiable objective function of the learned value distribution (e.g., mean, exponential risk, CVaR, etc.)

1.1 Related work

Distributional RL. The treatment of the policy return as a random variable dates back to Sobel (1982), where it is shown that the higher moments of the return obeys a Bellman equation. More recently, distributional RL has emerged as a paradigm for modelling and utilizing the entire distribution of returns (Tamar et al., 2013; Bellemare et al., 2023), with real-world applications including guidance of stratospheric balloons (Bellemare et al., 2020) and super-human race-driving in simulation (Wurman et al., 2022). The distributional RL toolbox has expanded over the years with diverse distribution representations (Bellemare et al., 2017; Dabney et al., 2018b,a; Yang et al., 2019) and deeper theoretical understanding (Bellemare et al., 2019; Rowland et al., 2018; Lyle et al., 2019). In our core algorithm, we use quantile-regression (QR) by Dabney et al. (2018b) as a tool for learning the value, rather than return, distribution. Moreover, QR has been integrated with soft actor-critic (SAC) (Haarnoja et al., 2018) for improved performance (Wurman et al., 2022; Kuznetsov et al.,

2020). At a high level, this paper combines model learning with quantile-regression, which is then integrated with SAC for policy optimization.

Bayesian RL. Model-free approaches to Bayesian RL directly model the distribution over values, e.g., with normal-gamma priors (Dearden et al., 1998), Gaussian Processes (Engel et al., 2003) or ensembles of neural networks (Osband et al., 2016). Instead, model-based Bayesian RL represents uncertainty in the MDP dynamics, which must then be propagated to the value function. For instance, the PILCO algorithm by Deisenroth and Rasmussen (2011) learns a Gaussian Process model of the transition dynamics and integrates over the model's total uncertainty to obtain the expected values. In order to scale to high-dimensional continuous-control problems, Chua et al. (2018) use ensembles of probabilistic neural networks (NNs) to capture both aleatoric and epistemic uncertainty, as first proposed by Lakshminarayanan et al. (2017). Both approaches propagate model uncertainty during policy evaluation and improve the policy via greedy exploitation over this model-generated noise.

Closest to our problem setting are approaches that explicitly model the value distribution function or statistics thereof. The uncertainty Bellman equation (UBE) offers a framework to estimate the *variance* of the value distribution (O'Donoghue et al., 2018; Zhou et al., 2020; Luis et al., 2023). Jorge et al. (2020) propose a principled backwards induction framework to estimate value distributions, with the caveat of assuming a Gaussian parameterization for practical implementation. Perhaps closest to our approach is the work by Dearden et al. (1999), which introduces a local sampling scheme that maintains a sample-based approximation of the value distribution, which is updated using a Bellman equation. While it does not assume a restrictive parametric form for the distribution, it ignores that samples from the value distribution at successive states are correlated through the Bellman equation; we make a similar approximation in our theory, see Section 3. In our work, rather than generating random samples of the value distribution, we keep track of a relevant set of statistics (Rowland et al., 2019), e.g., evenly spread quantiles, that have adequate coverage and representation power of the underlying distribution.

Mixed Approaches. Recent methods have combined distributional and model-based RL methods. Kastner et al. (2023) introduce the distributional model equivalence principle to train models that can plan optimally for risk-sensitive objectives. Moskovitz et al. (2021); Eriksson et al. (2022) aim to capture both sources of uncertainty by training an ensemble of return-distributional critics, where each critic models aleatoric uncertainty and the ensemble recovers epistemic uncertainty. Our approach is fundamentally different: we leverage tools from distributional RL to model the epistemic uncertainty around *expected* returns, i.e., we average over aleatoric noise. Moreover, our experiments show that our value representation with quantiles leads to substantial gains in performance over an ensemble of critics.

Uncertainty-Aware Policy Optimization. There exists a wide variety of policy optimization objectives that leverage epistemic uncertainty. Multi-model MDPs (MMDPs) (Steimle et al., 2021) consider a discrete distribution of MDPs and study the optimization of the average value under the MDP uncertainty. Solving exactly for the optimal policy is only possible for small MMDPs, but more recent methods can scale to larger problems (Su and Petrik, 2023). Robust MDPs optimize for risk-averse objectives, like the percentile criterion (also known as value-at-risk) (Delage and Mannor, 2010; Behzadian et al., 2021). In practice, robust MDP objectives tend to be overly conservative, thus soft-robustness (Derman et al.,

2018) has been proposed as an alternative objective, which is identical to the risk-neutral objective of MMDPs.

In this paper we approach uncertainty-aware optimization from a different perspective. Instead of fixing the policy optimization objective and designing a particular algorithm to solve for that objective, we propose a general-purpose method that aims to optimize *any* differentiable function of a learned distribution of values. The strength of our approach is that it flexibly accomodates an entire family of objectives that might suit different tasks, all within the same algorithm and with minimal changes.

2. Background & Notation

In this section, we provide the relevant background and formally introduce the problem of value distribution estimation. We use upper-case letters to denote random variables and lower-case otherwise. The notation $\mathcal{P}(\mathcal{X})$ refers to the space of probability distributions over the set $\mathcal{X}$, such that $\nu \in \mathcal{P}(\mathcal{X})$ is a probability measure with the usual[1] σ-algebra. We forego measure-theoretic formalisms and further qualifications of measurability will be implied with respect to the usual σ-algebra (cf. Bellemare et al., 2023, Remark 2.1).

2.1 Markov Decision Processes

We consider an agent that acts in an infinite-horizon MDP $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, p, r, \gamma\}$ with finite state space $\mathcal{S}$, finite action space $\mathcal{A}$, unknown transition function $p : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ that maps states and actions to the set of probability distributions over $\mathcal{S}$, a known[2] and bounded reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and a discount factor $\gamma \in [0, 1)$. The agent is equipped with an action-selection stochastic policy $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ that defines the conditional probability distribution $\pi(a \mid s)$, $(s, a) \in \mathcal{S} \times \mathcal{A}$. Given an initial state $s \in \mathcal{S}$ and a policy π, the RL agent interacts with the environment and generates a random *trajectory* $T = \{S_h, A_h, R_h\}_{h=0}^{\infty}$, with $S_0 = s$ and for $h \geq 0$ we have $A_h \sim \pi(\cdot \mid S_h)$, $R_h = r(S_h, A_h)$, $S_{h+1} \sim p(\cdot \mid S_h, A_h)$.

2.2 Return-Distributional Reinforcement Learning

The *return* of a policy, denoted Z^π, is a random variable defined as the discounted sum of rewards along a trajectory, $Z^\pi(s) = \sum_{h=0}^{\infty} \Big[\gamma^h R_h \mid S_0 = s\Big]$. The randomness in trajectories and returns originates from the inherent stochasticity of the environment dynamics and the policy, oftentimes called *aleatoric* uncertainty. A common objective for the RL agent is to maximize the *expected* return, where we average over this aleatoric noise to obtain a deterministic function known as the *value*. The value function of policy π under dynamics p, starting from $s \in \mathcal{S}$ is defined as a map $v^{\pi,p} : \mathcal{S} \to \mathbb{R}$ and is given by

$$v^{\pi,p}(s) = \mathbb{E}_T \left[\sum_{h=0}^{\infty} \gamma^h R_h \middle| S_0 = s, p \right], \tag{1}$$

1. Refers to the power set σ-algebra for finite $\mathcal{X}$, the Borel σ-algebra for infinite $\mathcal{X}$ and the product σ-algebra on products of such spaces (Bellemare et al., 2023).
2. The theory results can be easily extended to unknown reward functions.

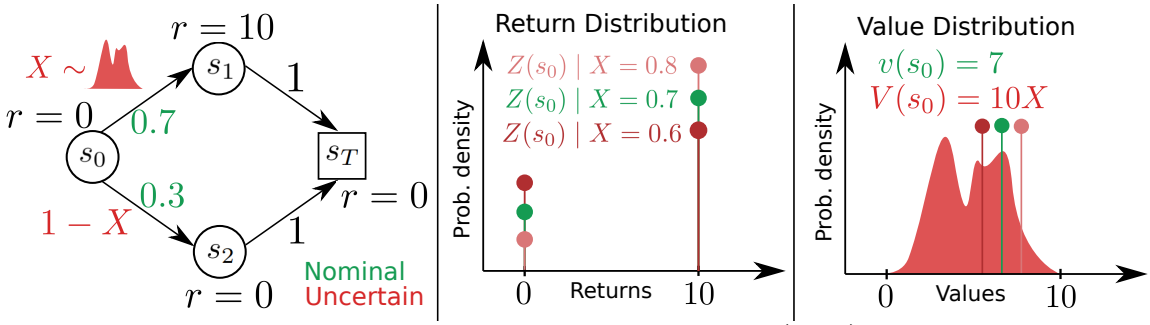

Figure 1: Return and value distributions in Bayesian RL. **(Left)** MDP with uncertain transition probability from s_0 given by a random variable $X \in [0, 1]$. **(Middle)** Return distributions at s_0 for realizations of X, including the nominal dynamics (green). The return distribution captures the *aleatoric* noise under the sampled dynamics. **(Right)** Distribution of values at s_0. In the nominal case, the value $v(s_0)$ is a scalar obtained from averaging the aleatoric uncertainty of the return distribution $Z(s_0)$ under the nominal dynamics. In our setting, $V(s_0)$ is a random variable due to the *epistemic* uncertainty around the MDP dynamics. To sample from $V(s_0)$ is equivalent to first sample $X = \tilde{x}$, compute the *conditional* return distribution $Z(s_0)|X = \tilde{x}$ and finally average over the aleatoric noise.

where we explicitly condition on the dynamics p; although redundant in the standard RL setting, this notation will become convenient later when we consider a distribution over dynamics.

In contrast to learning the value function, *return-distributional* RL aims to learn the entire distribution of returns by leveraging the random variable *return-distributional* Bellman equation (Bellemare et al., 2017)

$$Z^\pi(s) \overset{D}{=} r(s, A) + \gamma Z^\pi(S'), \tag{2}$$

where $A \sim \pi(\cdot \mid s)$, $S' \sim p(\cdot \mid s, A)$ and $(\overset{D}{=})$ denotes equality in distribution, i.e., the random variables in both sides of the equation may have different outcomes, but they share the same probability distribution.

2.3 Bayesian RL

In this paper, we adopt a Bayesian perspective and model the unknown dynamics p as a random transition function P with some prior distribution $\Phi(P)$. As the agent acts in $\mathcal{M}$, it collects data[3] $\mathcal{D}$ and obtains the posterior distribution $\Phi(P \mid \mathcal{D})$ via Bayes' rule. More concretely, for tabular problems we consider priors that admit analytical posterior updates (e.g., Dirichlet, Gaussian) (Dearden et al., 1999), and for continuous state-action spaces we use neural network ensembles (Lakshminarayanan et al., 2017) which have been linked to approximate Bayesian inference (Osband et al., 2018).

In what follows, we will assume $P \sim \Phi(P \mid \mathcal{D})$ and consider trajectories T defined as previously but with next-states as $S_{h+1} \sim P(\cdot \mid S_h, A_h)$. Notably, the sampling process of next states mixes two sources of uncertainty: the aleatoric noise, as with the original

3. We omit time-step subscripts and refer to dataset $\mathcal{D}$ as the collection of all available transition data.

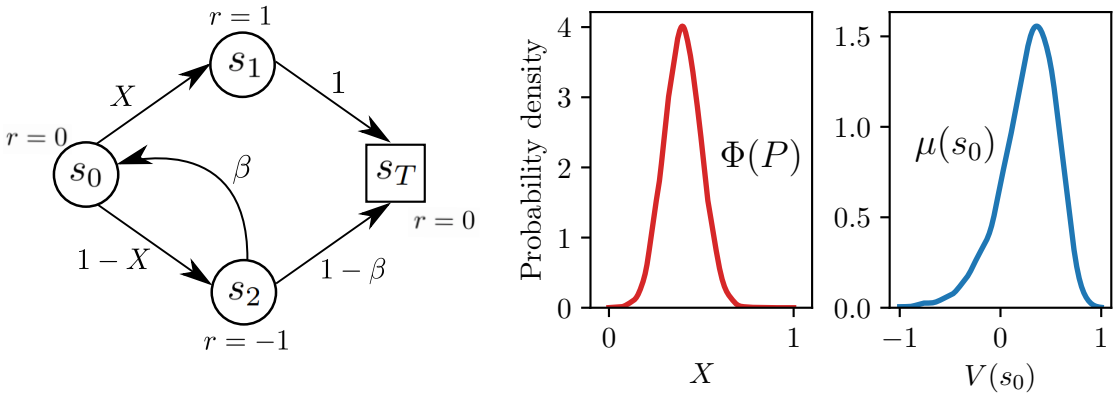

Figure 2: Example value distribution. **(Left)** Uncertain MDP with a truncated Gaussian transition probability $X \sim \bar{\mathcal{N}}(\mu = 0.4, \sigma = 0.1)$ and a scalar (deterministic) $\beta \in [0, 1]$. For this example, we fixed $\beta = 0.9$. **(Middle)** Distribution over MDPs, which corresponds directly to the distribution of X. **(Right)** Corresponding distribution of values for state s_0.

MDP, but also the uncertainty in P due to finite data, often called *epistemic* uncertainty. Consequently, the aleatoric and epistemic noise in trajectories propagates to the returns. We define the value function of policy π as a random variable under the random dynamics P as

$$V^\pi(s) = v^{\pi,P}(s). \tag{3}$$

According to the value function definition in (1), V^π is an expectation over the trajectories T *conditioned* on the random variable P, which means the aleatoric noise of trajectories is averaged out, but the epistemic uncertainty (due to the conditioning on P) remains and is propagated to V^π. Intuitively, to obtain a sample from V^π is equivalent to sample from the posterior $\Phi(P \mid \mathcal{D})$ and calculate the corresponding expected return, i.e., the value. As such, the stochasticity of V^π vanishes as we gather data and $\Phi(P \mid \mathcal{D})$ concentrates around the true transition function p.

The main focus of this paper is to study the *value-distribution*[4] function $\mu^\pi : \mathcal{S} \to \mathcal{P}(\mathbb{R})$, such that $V^\pi(s) \sim \mu^\pi(s)$, $\forall s \in \mathcal{S}$. As such, μ^π represents the distribution of the *epistemic* noise around the *expected* return of a policy. In Figure 1, we illustrate in a simple MDP the fundamental difference between return distributions and value distributions: the former captures aleatoric noise from the decision process, while the latter models *epistemic* uncertainty stemming from uncertain MDP dynamics. Refer to Figure 2 for another example of an uncertain transition probability and the value distribution it induces. While both value and return distributions aim to obtain a richer representation of complex random variables, only the former characterizes the type of uncertainty that is valuable for effective exploration of the environment.

4. We focus on state-value functions for simplicity, but the results have a straightforward extension for state-action-value functions.

3. The Value-Distributional Bellman Equation

In this section, we establish the theoretical backbone of iterative algorithms for learning the value-distribution function μ^π. We include formal proofs in Appendix B.

For the nominal transition kernel p, we can relate the values at subsequent time steps using the well-known Bellman equation

$$v^{\pi,p}(s) = \sum_a \pi(a \mid s) r(s,a) + \gamma \sum_{s',a} \pi(a \mid s) p(s' \mid s,a) v^\pi(s'), \tag{4}$$

which holds for any policy π and state $s \in \mathcal{S}$. The following statement is the straightforward extension of (4) in our Bayesian setting. While the result is not novel, it serves as a building block towards our main theoretical contribution.

Proposition 1 (Random Variable Value-Distribution Bellman Equation) *Let V^π be the random value function defined in* (3)*. Then, it holds that*

$$V^\pi(s) = \sum_a \pi(a \mid s) r(s,a) + \gamma \sum_{s',a} \pi(a \mid s) P(s' \mid s,a) V^\pi(s'), \tag{5}$$

for any policy π and initial state $s \in \mathcal{S}$.

Note that the random variable value-distribution Bellman equation in (5) differs from the random variable return-distribution Bellman equation in (2) in that the former holds in strict equality, while the latter holds in the weaker notion of equality in distribution. The main caveat of (5) with respect to model-free distributional RL is that, in general, $P(s' \mid s,a)$ and $V^\pi(s')$ are correlated.

We now shift from discussing the random value function to focus on the value distribution function. First, we provide a definition for μ^π that holds in the general case. Intuitively, we can think of μ^π as the result of *pushing* the probability mass of the posterior distribution Φ through the map defined by the value function (1). To formalize our statement, we leverage the notion of pushforward measures akin to Rowland et al. (2018).

Definition 1 *Given measurable spaces $\mathcal{X}$ and $\mathcal{Y}$, a measurable function $f : \mathcal{X} \to \mathcal{Y}$ and a measure $\nu \in \mathcal{P}(\mathcal{X})$, the pushforward measure $f_\# \nu \in \mathcal{P}(\mathcal{Y})$ is defined by $f_\# \nu(B) = \nu(f^{-1}(B))$, for all Borel sets $B \subseteq \mathcal{Y}$.*

Informally, given a random variable $X \sim \nu$ and the map f as in Definition 1, the pushforward of ν by f, denoted $f_\# \nu$, is defined as the distribution of the random variable $f(X)$ (Bellemare et al., 2023).

With slight abuse of notation, define the map $v^\pi : \mathcal{S} \times \mathscr{P} \to \mathbb{R}$ where $\mathscr{P}$ denotes the set of all transition functions[5] for $(\mathcal{S}, \mathcal{A})$, such that $v^\pi(s,p) = v^{\pi,p}(s)$ for any $p \in \mathscr{P}$, $s \in \mathcal{S}$. Then, the value distribution function μ^π is simply the pushforward measure of Φ by v^π.

Definition 2 *The value distribution function is defined as*

$$\mu^\pi = v^\pi_\# \Phi. \tag{6}$$

5. The set of all transition functions can also be written as $\mathcal{P}(\mathcal{S})^{\mathcal{S}\times\mathcal{A}}$ in standard set theory notation.

From Definition 2 we can already derive a simple (albeit, inefficient and computationally expensive) sample-based algorithm to estimate μ^π: sample from the posterior Φ and compute the value function (1) for each sample, which results in samples from μ^π. However, our main goal in this paper is to find a recursive definition of μ^π such that we can introduce a simple, yet efficient estimation algorithm.

The main challenge in establishing a recursion for learning μ^π is the dependency between $P(s' \mid s, a)$ and $V^\pi(s')$ in (5). We side-step this issue by restricting our study to a family of MDPs under which these random variables are independent, similar to previous work (O'Donoghue et al., 2018; Luis et al., 2023). All the results that follow in this section hold under:

Assumption 1 (Parameter Independence (Dearden et al., 1999)) *The posterior over the random vector $P(\cdot \mid s, a)$ is independent for each pair $(s, a) \in \mathcal{S} \times \mathcal{A}$.*

Assumption 2 (Directed Acyclic MDP (O'Donoghue et al., 2018)) *Let $\tilde{p} \in \mathscr{P}$ be a realization of the random variable P. Then, the MDP $\tilde{\mathcal{M}}$ with transition function $\tilde{p}$ is a directed acyclic graph, i.e., states are not visited more than once in any finite trajectory.*

Assumption 3 (Terminal State) *Define a terminal (absorbing) state s_T such that $r(s_T, a) = 0$ and $p(s_T \mid s_T, a) = 1$ for any $a \in \mathcal{A}$ and $p \in \mathscr{P}$. Let $\tilde{p} \in \mathscr{P}$ be a realization of the random variable P. Then, the MDP $\tilde{\mathcal{M}}$ with transition function $\tilde{p}$ deterministically transitions to s_T after a finite horizon $H \leq |\mathcal{S}|$, i.e., $\tilde{p}(s_T \mid s_H, a) = 1$ for any $a \in \mathcal{A}$.*

The consequence of these assumptions is that $P(s' \mid s, a)$ and $V^\pi(s')$ are conditionally independent for all triplets (s, a, s') (see Lemma 2). Assumption 1 is satisfied when modelling state transitions as independent categorical random variables for every pair (s, a), with the unknown parameter vector $P(\cdot \mid s, a)$ under a Dirichlet prior (Dearden et al., 1999). Assumptions 2 and 3 imply that any infinite trajectory is composed of *distinct* states for the first H steps and then remains at the terminal state s_T indefinitely. Our theoretical results do not hold in the general case of MDPs with cycles. However, one may still obtain reasonable approximations by considering an equivalent time-inhomogeneous MDP without cycles (also known as the "unrolled" MDP (O'Donoghue et al., 2018)) by forcing a transition to a terminal state after H steps (see Appendix A for an example). The approximation improves as $H \to \infty$, but in the limit implies an infinite state space, which would then require additional measure-theoretic considerations outside the scope of this work (cf. Bellemare et al., 2023, Remark 2.3). Despite these limitations, in Section 4 we empirically show that the algorithm stemming from our theory yields reasonable esimates of the value distribution μ^π in MDPs *with* cycles.

Beyond tabular representations of the transition function, introducing function approximation violates Assumption 1 due to generalization of the model (O'Donoghue et al., 2018; Zhou et al., 2020; Derman et al., 2020). However, in Section 6 our approach demonstrates strong empirical performance when paired with neural networks for function approximation.

We want to highlight that, under our assumptions, the *mean* value function $\mathbb{E}[V^\pi]$ corresponds exactly to the value function under the mean of the posterior Φ, denoted $\bar{p}$. That is, $\mathbb{E}[V^\pi] = v^{\pi,\bar{p}}$ (Luis et al., 2023). If our ultimate goal is to estimate the mean of μ^π, then standard approaches to approximately solve the Bellman expectation equation suffice.

However, in this paper we motivate the need for the distributional approach as it allows to flexibly specify policy objectives *beyond* maximizing the mean of the values. For instance, in Section 6 we explore the performance of optimistic value objectives.

To establish a Bellman-style recursion defining the value distribution function, we use the notion of pushforward measure from Definition 1. In particular, we are interested in the pushforward of the value distribution by the bootstrap function in (5). First, for any MDP with transition function $p \in \mathscr{P}$, we denote by $p^\pi : \mathcal{S} \to \mathcal{P}(\mathcal{S})$ the transition function of the Markov Reward Process (MRP) *induced* by policy π, defined by $p^\pi(s' \mid s) = \sum_a \pi(a \mid s) p(s' \mid s, a)$. Further, it is convenient to adopt the matrix-vector notation of the standard Bellman equation: $\mathbf{v}^{\pi,p} = \mathbf{r}^\pi + \gamma \mathbf{p}^\pi \mathbf{v}^{\pi,p}$, where $\mathbf{r}^\pi \in \mathbb{R}^{\mathcal{S}}$, $\mathbf{v}^{\pi,p} \in \mathbb{R}^{\mathcal{S}}$ are vectors and $\mathbf{p}^\pi \in \mathbb{R}^{\mathcal{S}\times\mathcal{S}}_{[0,1]}$ is a so-called stochastic matrix whose entries are restricted to $[0, 1]$ and whose rows sum up to 1, i.e., such that it represents the transition function p^π. Then, we define the bootstrap function $b_{r,p,\gamma} : \mathbb{R}^{\mathcal{S}} \to \mathbb{R}^{\mathcal{S}}$ applied to value vectors:

$$b_{r,p,\gamma} : \mathbf{v} \to \mathbf{r} + \gamma \mathbf{p} \mathbf{v}, \tag{7}$$

for an arbitrary $\mathbf{r} \in \mathbb{R}^{\mathcal{S}}$, $\mathbf{p} \in \mathbb{R}^{\mathcal{S}\times\mathcal{S}}_{[0,1]}$ and $\gamma \in [0, 1)$. Applying $b_{r,p,\gamma}$ is a combination of adding $\mathbf{r}$ to a γ-scaled linear transformation of the input vector. Further, we express mixtures with weights given by the posterior $\Phi(P \mid \mathcal{D})$ more compactly with the notation[6] $\mathbb{E}_P[\cdot]$, where the argument is a probability distribution depending on P. Given the pushforward and mixture operations, we can now propose a Bellman equation for the value distribution function μ^π.

Lemma 1 (Value-Distribution Bellman Equation) *The value distribution function μ^π obeys the Bellman equation.*

$$\mu^\pi = \mathbb{E}_P\left[(b_{r^\pi, P^\pi, \gamma})_{\#}\mu^\pi\right] \tag{8}$$

for any policy π.

Lemma 1 provides the theoretical backbone towards designing an iterative algorithm for learning the value distribution function. In particular, the recursive definition for μ^π, which corresponds to a mixture of pushforwards of itself, leads to efficient estimation methods by dynamic programming. Alternatively, we can also write the value-distributional Bellman equation for each state $s \in \mathcal{S}$. With slight abuse of notation, define $b_{r,\gamma} : \mathbb{R} \to \mathbb{R}$ as the map $v \to r + \gamma v$, then

$$\mu^\pi(s) = \mathbb{E}_P\left[\sum_{s'} P^\pi(s' \mid s)(b_{r^\pi(s),\gamma})_{\#}\mu^\pi(s')\right]. \tag{9}$$

Note that (6) holds generally, while (8) and (9) only hold under Assumptions 1–3. Moreover, the operator $\mathbb{E}_P[\cdot]$ is well-defined in (8) and (9) since $P^\pi(s' \mid s)$ is bounded in $[0, 1]$ for all $s, s' \in \mathcal{S}$ (Billingsley, 1995).

In Figure 3 we illustrate the core operations involved in the value-distributional Bellman recursion prescribed by (9).

From (8) we can extract an operator that acts on arbitrary value distribution functions.

6. Adapted from Bellemare et al. (2023). It refers to a mixture distribution and must not be mistaken by an expected value, which is a scalar.
7. The pushforward operator $(b_{r,\gamma})_{\#}$ is linear (cf. Bellemare et al., 2023, Exercise 2.13), so it can be moved outside the next-state mixture operation as depicted in the diagram.

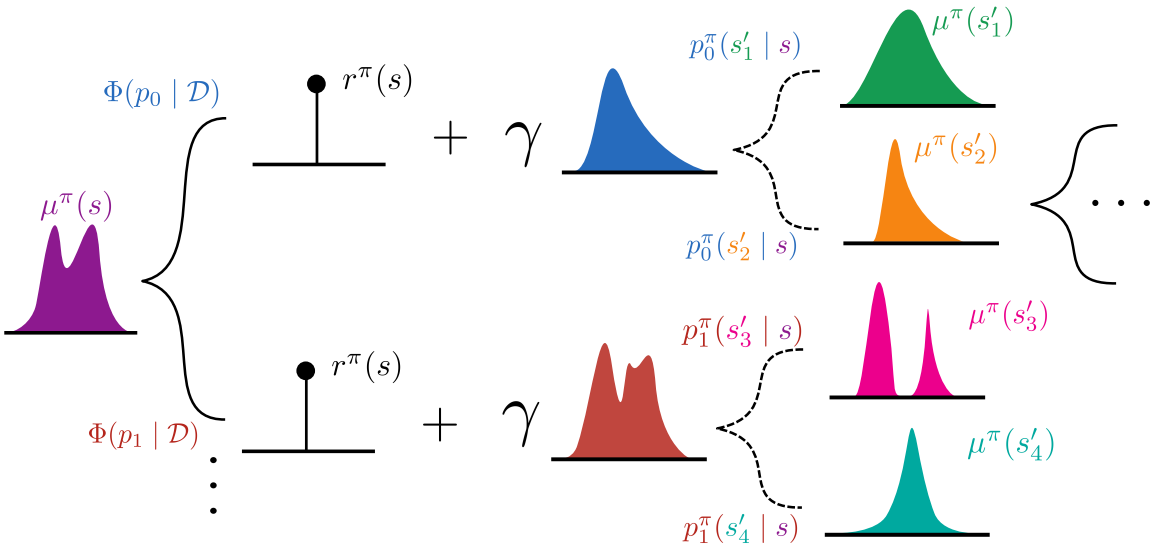

Figure 3: Visualization of the value-distributional Bellman backups, as prescribed by (9). We identify four operations on distributions: infinite mixture over posterior transition functions (solid braces), shift by reward, scale by discount factor and mixture over next states (broken line braces)[7]. The main difference w.r.t the return-distributional backup (cf. Bellemare et al., 2023, Figure 2.6) is the presence of the two distinct mixture operations.

Definition 3 *The value-distributional Bellman operator* $\mathcal{T}^\pi : \mathcal{P}(\mathbb{R})^{\mathcal{S}} \to \mathcal{P}(\mathbb{R})^{\mathcal{S}}$ *is defined by*

$$\mathcal{T}^\pi \mu = \mathbb{E}_P\left[(b_{r^\pi, P^\pi, \gamma})_{\#}\mu\right] \tag{10}$$

Intuitively, the operator $\mathcal{T}^\pi$ corresponds to mixing pushforward distributions, where the pushforward itself involves shifting, scaling and linearly transforming the probability mass. The natural question that follows is whether we can establish convergence to μ^π by repeated applications of $\mathcal{T}^\pi$ starting from an arbitrary initial guess μ_0.

Our convergence result is an adaptation of the standard distributional RL analysis by Bellemare et al. (2023). With some abuse of notation, we adopt the supremum p-Wasserstein distance $\bar{w}_p$ to establish contractivity of the operator $\mathcal{T}^\pi$ (see Definition 4 in Appendix B).

Theorem 1 *The operator* $\mathcal{T}^\pi$ *is a* γ*-contraction with respect to* $\bar{w}_p$ *for all* $p \in [1, \infty)$*. That is,* $\bar{w}_p(\mathcal{T}^\pi \mu, \mathcal{T}^\pi \mu') \leq \gamma \bar{w}_p(\mu, \mu')$ *for all* $\mu, \mu' \in \mathcal{P}(\mathbb{R})^{\mathcal{S}}$ *such that* $V(s') \sim \mu(s')$*,* $V'(s') \sim \mu'(s')$ *are conditionally independent of* $P^\pi(s' \mid s)$ *given* $s' \in \mathcal{S}$*.*

Theorem 1 parallels similar results in standard RL and model-free distributional RL, in that it allows us to establish the convergence of iterated applications of $\mathcal{T}^\pi$ and characterize the operator's fixed-point.

Corollary 1 *Denote the space of value distribution functions with bounded support*[8] *by* $\mathcal{P}_B(\mathbb{R})^{\mathcal{S}}$*. Given an arbitrary value distribution function* $\mu_0 \in \mathcal{P}_B(\mathbb{R})^{\mathcal{S}}$*, the sequence* $\{\mu_k\}_{k=0}^{\infty}$

8. Under bounded reward functions, the corresponding value distributions have bounded support. The corollary can be relaxed to distributions with bounded moments (see Proposition 4.30 in Bellemare et al. (2023).)

defined by $\mu_{k+1} = \mathcal{T}^\pi \mu_k$ *for all* $k \geq 0$ *is such that* $\bar{w}_p(\mu_k, \mu^\pi) \leq \gamma^k \bar{w}_p(\mu_0, \mu^\pi) \to 0$ *as* $k \to \infty$ *for* $p \in [1, \infty)$. *That is,* μ^π *is the unique fixed-point of the operator* $\mathcal{T}^\pi$.

Proof We wish to apply Theorem 1 on the sequence of pairs $\{(\mu_k, \mu^\pi)\}_{k=0}^\infty$. The conditional independence assumption required to apply Theorem 1 holds for μ^π (see Lemma 2), but it is straightforward to show it also holds (under Assumptions 1–3) for all the elements of the sequence $\{\mu_k\}_{k=0}^\infty$ (see Lemma 3). Further, since we consider bounded rewards, it follows immediately that $\mu^\pi \in \mathcal{P}_B(\mathbb{R})^\mathcal{S}$. Moreover, it can be shown that the operator $\mathcal{T}^\pi$ maps $\mathcal{P}_B(\mathbb{R})^\mathcal{S}$ onto itself, such that for arbitrary $\mu \in \mathcal{P}_B(\mathbb{R})^\mathcal{S}$ then $\mathcal{T}^\pi \mu \in \mathcal{P}_B(\mathbb{R})^\mathcal{S}$ (see Lemma 4). By Theorem 1, $\mathcal{T}^\pi$ is then a *contraction mapping* and by Banach's fixed-point theorem $\mathcal{T}^\pi$ admits a unique fixed-point which is the limiting value of the sequence $\{\mu_k\}_{k=0}^\infty$. Since $\mu^\pi = \mathcal{T}^\pi \mu^\pi$ holds by Lemma 1, then μ^π must be the unique fixed-point of $\mathcal{T}^\pi$. ■

In summary, Corollary 1 establishes that repeated applications of $\mathcal{T}^\pi$ from an arbitrary initial guess converges to the value distribution function μ^π. Inspired by this theoretical result, in the remaining sections we introduce and evaluate a practical algorithm for learning the value distribution function.

4. Quantile-Regression for Value-Distribution Learning

In the previous section we described an iterative process that converges to μ^π starting from an arbitrary value distribution with bounded support. In practice, however, to implement such a recursion we must project the value distributions onto some finite-dimensional parameter space. Following the success of quantile distributional RL (Dabney et al., 2018b), we adopt the quantile parameterization. Let $\mathcal{V}_m$ be the space of quantile distributions with m quantiles and corresponding quantile levels $\tau_i = 1/m$ for $i = \{1, \ldots, m\}$ and $\tau_0 = 0$. Define a parametric model $q : \mathcal{S} \to \mathbb{R}^m$, then the quantile distribution $\mu_q \in \mathcal{V}_m$ maps states to a uniform probability distribution supported on $q_i(s)$. That is, $\mu_q(s) = \frac{1}{m} \sum_{i=1}^m \delta_{q_i(s)}$, where δ_x denotes the Dirac delta distribution centered at $x \in \mathbb{R}$, such that $\mu_q(s)$ is a uniform mixture of Dirac deltas where the particle $q_i(s)$ corresponds to the τ_i-quantile at state s. With this parameterization, our aim now becomes to compute the so-called quantile projection of μ^π onto $\mathcal{V}_m$, given by

$$\Pi_{w_1} \mu^\pi := \operatorname*{argmin}_{\mu_q \in \mathcal{V}_m} w_1(\mu^\pi, \mu_q), \tag{11}$$

where w_1 is the 1-Wasserstein distance. Define $F^{-1}_{\mu^\pi}$ as the inverse cumulative distribution function of μ^π, then the distance metric becomes

$$w_1(\mu^\pi, \mu_q) = \sum_{i=1}^m \int_{\tau_{i-1}}^{\tau_i} \left| F^{-1}_{\mu^\pi}(\omega) - q_i \right| d\omega, \tag{12}$$

since μ_q is a uniform distribution over m Dirac deltas with support $\{q_1, \ldots, q_m\}$. Let $\hat{\tau}_i = (2i-1)/2m$, then a valid minimizer of (11) exists and is achieved by selecting $q_i = F^{-1}_{\mu^\pi}(\hat{\tau}_i)$ (cf. Dabney et al., 2018b, Lemma 2). In summary, quantile projection as defined in (11) is equivalent to estimating each $\hat{\tau}_i$-quantile of μ^π.

We follow closely the treatment by Rowland et al. (2023) of quantile-regression temporal-difference learning for return-distributions and adapt it to instead work on value-distributions.

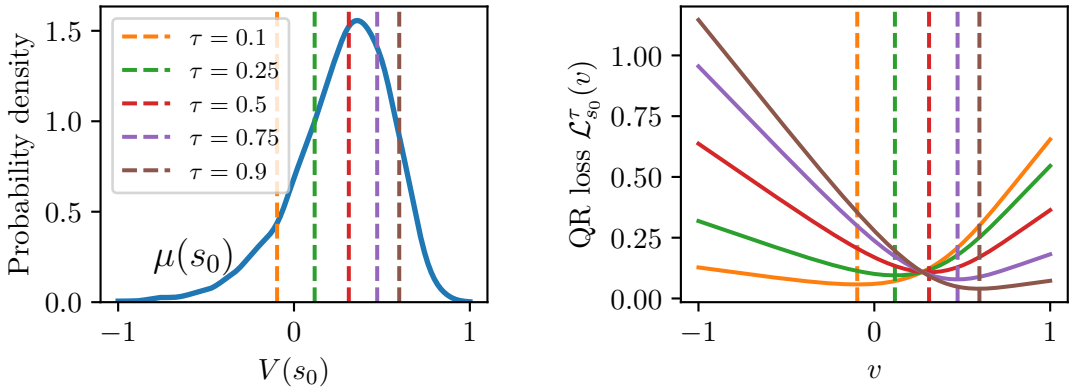

Figure 4: Quantile-regression loss for the example MDP of Figure 2. **(Left)** Probability density of values for state s_0, with five quantile levels in colored vertical lines. **(Right)** The quantile regression loss (13) for the five quantile levels; the vertical lines correspond to the minimum of the color-matching loss. The vertical lines on both plots match upto numerical precision, meaning that following the gradient of such a convex loss function would indeed converge to the quantile projection $\Pi_{w_1}\mu$.

The following loss function corresponds to the quantile-regression problem of estimating the τ-quantile of the value distribution μ^π:

$$\mathcal{L}_s^{\tau,\pi}(v) = \mathbb{E}_P\Big[\big(\tau \mathbb{1}\{V^\pi(s) > v\} + (1-\tau)\mathbb{1}\{V^\pi(s) < v\}\big)\big|V^\pi(s) - v\big|\Big]. \tag{13}$$

It is an asymmetric convex loss function, where quantile overestimation and underestimation errors are weighted by τ and $1-\tau$, respectively. The unique minimizer of this loss is the τ-quantile of μ^π, which we illustrate with an example in Figure 4.

Our goal is to propose a practical algorithm to learn the value distribution function based on the quantile-regression loss (13). If we have access to samples of $V^\pi(s)$, denoted $\tilde{v}^\pi(s)$, then we can derive an unbiased estimate of the negative gradient of (13) and obtain the update rule

$$q_i(s) \leftarrow q_i(s) + \alpha\big(\tau_i - \mathbb{1}\{\tilde{v}^\pi(s) < q_i(s)\}\big), \tag{14}$$

where α is some scalar step size. One option to sample $V^\pi = \tilde{v}^\pi$ would be to first sample a model $P = \tilde{p}$ and then solve the corresponding Bellman equation. Instead, we use a computationally cheaper alternative (albeit biased) and bootstrap like in temporal-difference learning, so that the samples are defined as

$$\tilde{v}^\pi(s) = r^\pi(s) + \gamma \sum_{s'} \tilde{p}^\pi(s' \mid s) q_J(s'), \tag{15}$$

where $J \sim \text{Uniform}(1, \dots, m)$. Lastly, we reduce the variance of the gradient estimate by averaging over the values of J, which results in the update

$$q_i(s) \leftarrow q_i(s) + \frac{\alpha}{m}\left(\tau_i - \sum_{j=1}^{m} \mathbb{1}\left\{r^\pi(s) + \gamma \sum_{s'} \tilde{p}^\pi(s' \mid s) q_j(s') < q_i(s)\right\}\right). \tag{16}$$

Algorithm 1 Epistemic Quantile-Regression (EQR)

Input: Posterior MDP Φ, policy π, number of quantiles m.
Randomly initialize estimates $\{q_i(s)\}_{i=1}^m$ for all $s \in \mathcal{S}$
repeat
  Sample $\tilde{p}$ from posterior Φ
  for $i = 1, \dots, m$ **do**
    Update $q_i(s)$ with (16) for all $s \in \mathcal{S}$
until convergence
return $\{q_i(s)\}_{i=1}^m$

We introduce EQR in Algorithm 1 to iteratively learn the value distribution function μ^π. From an arbitrary initial guess of quantiles, we sample an MDP from the posterior and update the quantiles using (16) for all states until convergence. The following examples illustrate the performance of EQR in tabular problems.

Example 1 (Toy MDP) *Consider once more the tabular MDP of Figure 2. Our goal is to assess the convergence properties of EQR both when Assumptions 1–3 hold, but also when they are violated* [9]*. We control the degree of violation of Assumptions 2 and 3 via the parameter β of the MDP. If $\beta = 0$, the assumptions hold and $V(s_0)$ and $P(s_2|s_0)$ are decorrelated. As β goes from zero to one, the covariance between these two random variables increases monotonically. We manually design three MDP posterior distributions that result in diverse distributions for $V(s_0)$. The value distributions shown in the top row of Figure 5 are the result of modelling the MDP parameter X as the following mixtures of truncated Gaussian distributions: $X_{left} \sim \bar{\mathcal{N}}(\mu = 0.5, \sigma = 0.1)$, $X_{middle} \sim 0.5\bar{\mathcal{N}}(\mu = 0.3, \sigma = 0.03) + 0.5\bar{\mathcal{N}}(\mu = 0.6, \sigma = 0.05)$ and $X_{right} \sim 0.5\bar{\mathcal{N}}(\mu = 0.3, \sigma = 0.03) + 0.5\bar{\mathcal{N}}(\mu = 0.5, \sigma = 0.15)$. For this selection of value distributions, we run EQR to estimate $m = 10$ quantiles.*

The middle row of Figure 5 shows that, for $\beta = 0$, the prediction error oscillates close to zero for every quantile, thus validating the result of Corollary 1. To test the prediction quality when the assumptions are violated, we generate different values for $\beta \in [0, 1]$ and run EQR for the same three MDP posterior distributions. The bottom plots in Figure 5 show the 1-Wasserstein metric between the true and predicted quantile distributions after 10^4 gradient steps; the error, like the covariance between $V(s_0)$ and $P(s_2|s_0)$, increases monotonically with β. The prediction quality thus degrades depending on the magnitude of the covariance between the transition kernel and the values.

Example 1 is mostly pedagogical and serves the purpose of validating our theoretical result, but it remains a contrived example with limited scope. The next example analyzes the performance of EQR in a standard tabular problem.

Example 2 (Gridworld) *We consider a modification of the N-room gridworld environment by Domingues et al. (2021), consisting of three connected rooms of size 5×5. The task for this example is to predict $m = 100$ quantiles of the value distribution under the optimal policy*

9. In order to be closer to standard settings, when the assumptions are violated (i.e., MDP contains cycles) we do not unroll the MDP as described in Appendix A.

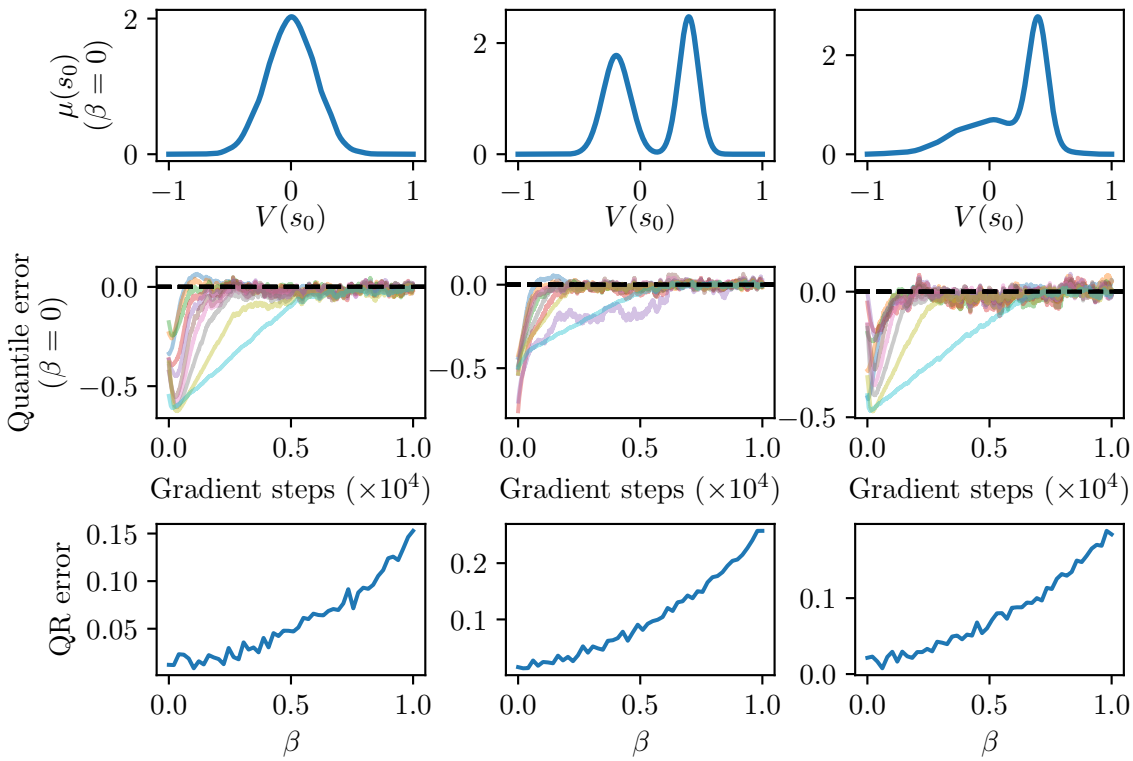

Figure 5: Performance of quantile-regression for value-distribution learning in the example MDP of Figure 2. The parameter β controls the covariance between $V(s_0)$ and $P(s_2|s_0)$; the covariance increases with β and is zero for $\beta = 0$. **(Top)** Value distributions (Gaussian, bi-modal and heavy-tailed) generated by different prior distributions of the parameter δ. **(Middle)** Evolution of the per-quantile estimation error $(\Pi_{w_1}\mu(s_0) - \mu_q(s_0))$ between the true quantile projection and the prediction; for $\beta = 0$, our algorithm oscillates around the true quantile projection. **(Bottom)** 1-Wasserstein metric between the true quantile projection and the estimate μ_q after 10^4 gradient steps, as a function of the correlation parameter β. As β moves from zero to one, the regression error increases and the algorithm no longer converges to the true quantiles, although the error is relatively small.

$\pi^\star$ (obtained by running any standard tabular exploration algorithm, like PSRL (Osband et al., 2013)). We use a Dirichlet prior for the transition kernel and a standard Gaussian for the rewards. We collect data using $\pi^\star$, update the posterior MDP and run EQR to predict the value distribution.

In Figure 6, we summarize the results at three different points during data collection. As more data is collected, the corresponding MDP posterior shrinks and we observe the value distribution concentrates around the value under the true dynamics (dotted vertical line). For both wide (episode 1) and narrow (episode 100) posteriors, EQR is able to accurately predict the distribution of values. The impact of violating Assumption 2 is the non-zero steady-state quantile-regression error. We observe the bias of the predicted quantiles is typically lowest (near zero) close to the median and highest at the extrema.

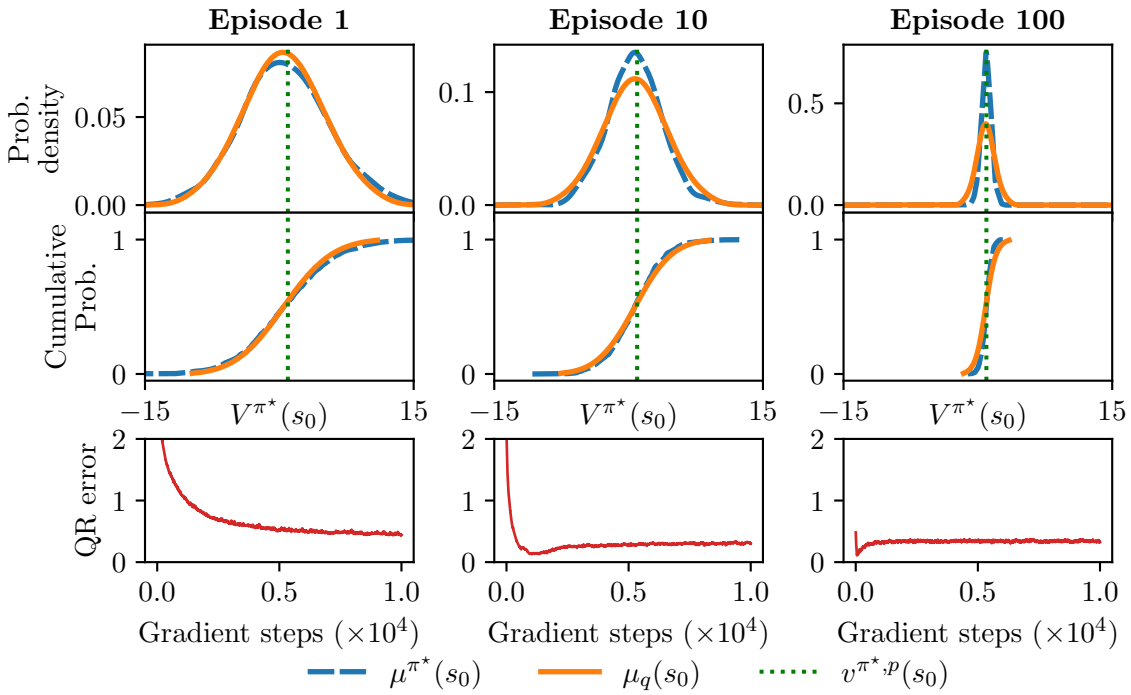

Figure 6: Performance of EQR in a Gridworld environment. We train the optimal policy $\pi^\star$ using PSRL (Osband et al., 2013) and then use it for data collection. At different points during data collection, we run EQR to estimate $m = 100$ quantiles of the value distribution for the initial state under $\pi^\star$, given the current posterior MDP. **(Top-Middle)** PDF and CDF of the true (dashed blue) and predicted (solid orange) value distribution, with the true optimal value (dotted green) as vertical reference. **(Bottom)** 1-Wasserstein distance between the true quantile projection and the prediction.

5. Policy Optimization with Value Distributions

In this section, we propose an algorithm to optimize a policy given some differentiable utility function f of the learned quantile distribution μ_q (which is implicitly parameterized by π). Namely, we define the optimal policy

$$\pi^\star = \operatorname*{argmax}_\pi f(\mu_q; \pi). \tag{17}$$

To approximately solve (17), we combine EQR with SAC (EQR-SAC) to obtain a model-based reinforcement learning algorithm that leverages a value-distributional critic for policy optimization. Our algorithm is agnostic to f as long as it is differentiable and thus can backpropagate gradients through it. The key ingredients of our method are: (1) an ensemble-based posterior over MDPs, (2) a quantile-distributional critic network that models the m-quantile function $q(s, a)$ and (3) a policy network π_ϕ trained to optimize (17). A full algorithmic description of EQR-SAC is included in Appendix C.

Posterior Dynamics. We adopt the baseline architecture from MBPO (Janner et al., 2019) and the implementation from Pineda et al. (2021), where the posterior MDP, denoted

Γ_ψ, is represented as an ensemble of n neural networks trained via supervised learning on the environment dataset $\mathcal{D}$ to predict the mean and variance of a Gaussian distribution over next states and rewards. We use Γ_ψ to populate an experience replay buffer $\mathcal{D}_{\text{model}}$ with model-consistent k-step rollouts; that is, we use a consistent ensemble member throughout a rollout, rather than randomizing the model per-step like in MBPO.

Critic. We train the critic on mini-batches drawn from $\mathcal{D}_{\text{model}}$, use the entropy regularization loss from SAC with temperature α and replace the quantile regression loss with the quantile Huber loss $\rho_\tau(u)$ from Dabney et al. (2018b) (see Appendix C.1)

$$\mathcal{L}_{\text{critic}}(q) = \mathbb{E}_{(S,A)\sim\mathcal{D}_{\text{model}}}\left[\mathbb{E}_{(\hat{R},\hat{P})\sim\Gamma_\psi}\left[\sum_{i=1}^{m}\mathbb{E}_J\left[\rho_{\tau_i}\big(\mathcal{T}q_J(S,A) - q_i(S,A)\big)\right]\right]\right], \tag{18}$$

where the target quantiles $\mathcal{T}q_j$ are defined as

$$\mathcal{T}q_j(s,a) = \hat{R}(s,a) + \gamma\,\mathbb{E}_{(S',A')\sim\hat{P}(\cdot|s,a),\pi_\phi}\big[q_j(S',A') - \alpha\log\pi_\phi(A'\mid S')\big]. \tag{19}$$

The expectation in (19) is approximated by generating transition tuples (s',a') using the policy and the sampled dynamics from Γ_ψ. Typically, model-based algorithms like MBPO only use data in the mini-batch to compose critic targets, rather than leveraging the learned dynamics model for better approximation of expectations.

Actor. The policy is trained to maximize the objective in (17), in addition to the entropy regularization term from SAC,

$$\mathcal{L}_{\text{actor}}(\phi) = \mathbb{E}_{S\sim\mathcal{D}_{\text{model}}}\Big[\mathbb{E}_{A\sim\pi_\phi}\big[f(q(S,A)) - \alpha\log\pi_\phi(A\mid S)\big]\Big]. \tag{20}$$

Let $\bar{q}(s,a)$ and $\sigma_q(s,a)$ be the mean and standard deviations of the quantiles, respectively. Then, we consider two concrete utility functions: the classical mean objective $f_{\text{mean}}(q(s,a)) = \bar{q}(s,a)$ and an objective based on optimism in the face of uncertainty $f_{\text{ofu}} = \bar{q}(s,a) + \sigma_q(s,a)$.

6. Experiments

In this section, we evaluate EQR-SAC in environments with continuous state-action spaces. Implementation details and hyperparameters are included in Appendices C and D, respectively. Unless noted otherwise, all training curves are smoothened by a moving average filter and we report the mean and standard error over 10 random seeds.

6.1 Baselines

We consider the following baselines, which all share a common codebase and hyperparameters.

SAC with typical design choices like target networks (Mnih et al., 2013), clipped double Q-learning (Fujimoto et al., 2018) and automatic entropy tuning (Haarnoja et al., 2019).

MBPO with slight modifications from Janner et al. (2019): (1) it only uses $\mathcal{D}_{\text{model}}$ to update the actor and critic, rather than mixing in data from $\mathcal{D}$; (2) it uses a fixed rollout length k, instead of an adaptive scheme. With respect to EQR-SAC, MBPO collects data differently: instead of collecting k-step rollouts under each model of the ensemble, it does so by uniformly sampling a new model per step of the rollout.

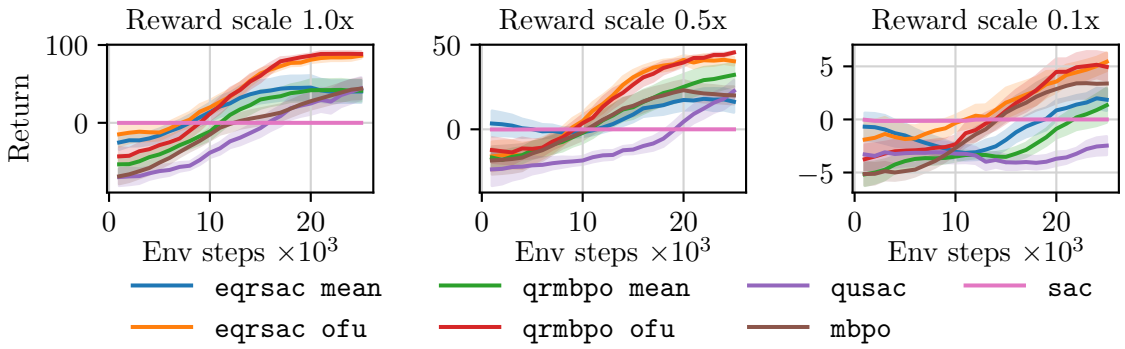

Figure 7: Performance in the Mountain Car environment. We consider the original version of the environment (left) and two variants (middle, right) that scale down the rewards by some factor (0.5 and 0.1, respectively).

QR-MBPO, which replaces the critic in MBPO with a quantile-distributional critic, trained on the standard quantile-regression loss from Wurman et al. (2022), but using data from $\mathcal{D}_{\text{model}}$,

$$\mathcal{L}_{\text{critic}}^{\text{qrmbpo}}(q) = \mathbb{E}_{(S,A,S',R)\sim\mathcal{D}_{\text{model}}}\left[\left[\sum_{i=1}^{M}\mathbb{E}_J\left[\rho_{\tau_i}\left(\mathcal{T}q_J^{\text{qrmbpo}}(R,S,A) - q_i(S,A)\right)\right]\right]\right], \tag{21}$$

where the target quantile is defined as

$$\mathcal{T}q_j^{\text{qrmbpo}}(r,s',a) = r + \gamma\big(q_j(s',a') - \alpha\log\pi_\phi(a' \mid s')\big), \tag{22}$$

and $a' \sim \phi_\phi(a' \mid s')$. Importantly, (21) differs from (18) in that the former captures both the aleatoric and epistemic uncertainty present in $\mathcal{D}_{\text{model}}$, while the latter aims to average out the aleatoric noise from the target quantiles. The objective function for the actor is the same as EQR-SAC.

QU-SAC, as proposed by Luis et al. (2023). It collects data as in EQR-SAC, but stores the n model-consistent rollouts in n separate buffers (while EQR-SAC uses a single buffer). Then it trains an ensemble of n standard critics on the corresponding n model-buffers. As such, it interprets the ensemble of critics as samples from the value distribution. The actor is optimized to maximize the mean prediction of the critic ensemble.

6.2 Case Study - Mountain Car

We motivate the importance of learning a distribution of values with a simple environment, the Mountain Car (Sutton and Barto, 2018) with continuous action space, as implemented in the `gym` library (Brockman et al., 2016). The environment's rewards are composed of a small action penalty and a large sparse reward once the car goes up the mountain, defined by a horizontal position $x > 0.45$ meters. We consider three versions of the problem: the original one, and two variants where all the rewards are scaled by a constant factor of 0.5 and 0.1, respectively.

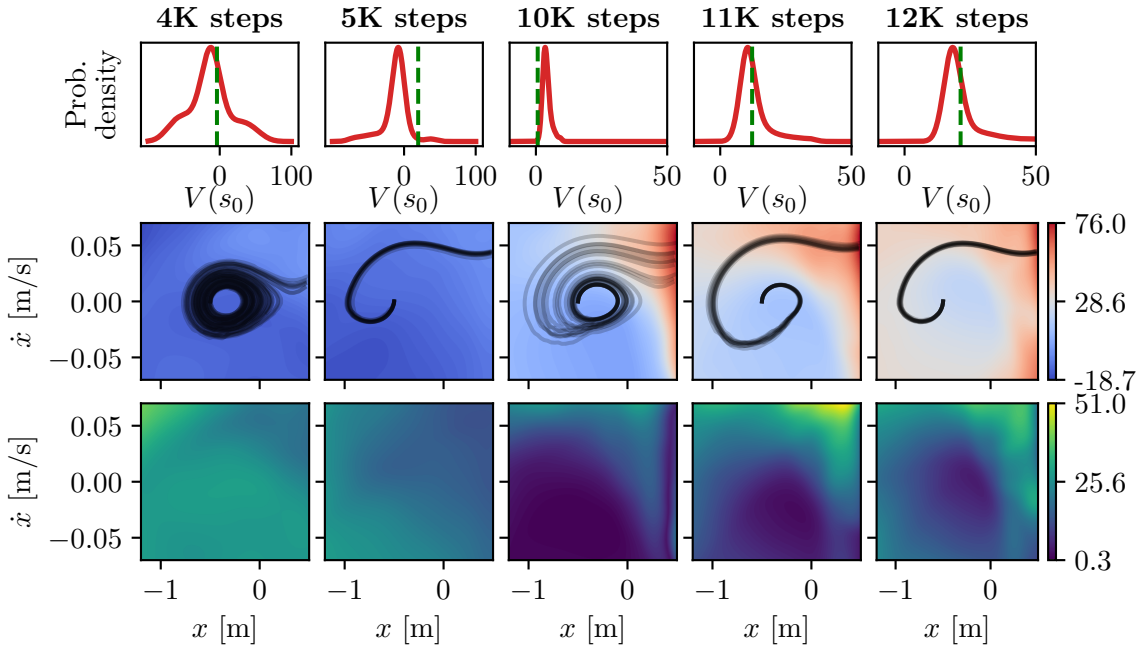

Figure 8: Visualization of the learned value distribution of EQR-SAC at different points during training in the Mountain Car environment (with reward scale of 0.5x). **(Top)** The predicted value distribution at the initial state. The dotted line is the empirical value of s_0 based on ten trajectories (black lines of middle row). **(Mid-Bottom)** The mean (mid) and standard deviation (bottom) of the value distribution across the state space.

While it is low-dimensional and has simple dynamics, many RL algorithms fail to solve the Mountain Car problem due to its combination of action penalties and sparse rewards. Naive exploration strategies based on injecting unstructured noise, like SAC, typically fail to solve such tasks (Raffin et al., 2021). We plot the performance of EQR-SAC and all baselines in Figure 7. EQR-SAC and QR-MBPO have the best overall performance, both using the optimistic objective function f_{ofu} (as previously defined after (20)). These results highlight the need to model uncertainty *and* leverage it during optimization; optimizing the mean values significantly degraded performance of the distributional approaches.

In Figure 8, we inspect further the distribution of values learned by EQR-SAC during a training run. The value distribution is initially wide and heavy-tailed, as the agent rarely visits goal states. At 5K steps, the policy is close-to-optimal but the predicted distribution underestimates the true values. In subsequent steps, the algorithm explores other policies while reducing uncertainty and calibrating the predicted value distribution. At 12K steps, the algorithm stabilizes again at the optimized policy, but with a calibrated value distribution whose mean is close to the empirical value. We notice the large uncertainty in the top-right corner of the state space remains (and typically does not vanish if we run the algorithm for longer); we hypothesize this is mainly an artifact of the discontinuous reward function, which is smoothened out differently by each member of the ensemble of dynamics, such that epistemic uncertainty stays high.

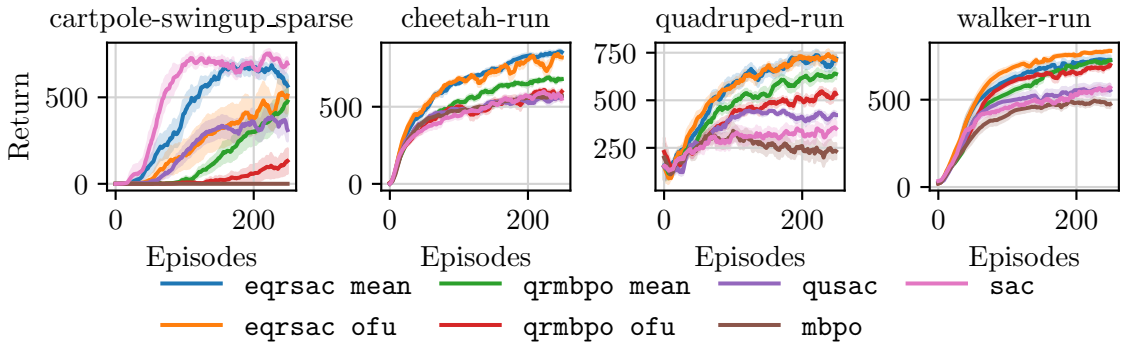

Figure 9: Performance in four DeepMind Control tasks. Cartpole swing-up has sparse rewards, while Cheetah, Quadruped and Walker have dense rewards. EQR-SAC significantly improves performance with respect to the model-based baselines.

6.3 DM Control Benchmark

In order to evaluate EQR-SAC more broadly, we conduct an experiment in a subset of 16 continuous-control tasks from the DeepMind Control Suite (Tunyasuvunakool et al., 2020). The chosen environments include both dense/sparse rewards and smooth/discontinuous dynamics. In Figure 9, we plot the results for four environments ranging from small (cartpole) to mid/large (quadruped) observation spaces. Our method significantly improves performance over previous model-based algorithms in these environments. Moreover, in the full benchmark, EQR-SAC achieves the best (or comparable) final performance in 13 out of 16 tasks (see Appendix F). We summarize the results of the DMC benchmark in Figure 10, following the guidelines by Agarwal et al. (2021). At 250 episodes of training, EQR-SAC OFU achieves the highest normalized IQM score, which is $\sim 17\%$ higher than QR-MBPO mean (see Table 3 for numerical scores). However, there exists some overlap between the 95% confidence intervals, which tend to be large in our benchmark due to a wide range of normalized scores across different environments. In this scenario, the recommendation in Agarwal et al. (2021) is to analyze score distribution performance profiles, as presented in Figure 10, which provide a more complete overview of the results. We observe the EQR-SAC OFU performance profile tends to dominate over the baselines, especially for normalized scores between $[0.5, 0.8]$.

We observe a clear gap in performance between MBPO and QR-MBPO, which supports the observations from Lyle et al. (2019) and reinforces their hypothesis that distributional critics boost performance under non-linear function approximation. The gap between QU-SAC and the distributional methods (QR-MBPO / EQR-SAC) indicates that the quantile representation of values leads to more sample-efficient learning compared to the ensemble-based approach. Moreover, training one distributional critic is typically less computationally intensive than training an ensemble of standard critics. In the next section, we investigate more deeply the performance difference between EQR-SAC and QR-MBPO.

6.4 Why Does EQR-SAC Outperform QR-MBPO?

We conduct an additional experiment to determine what component(s) of EQR-SAC are responsible for the increased performance with respect to QR-MBPO. There is three differences

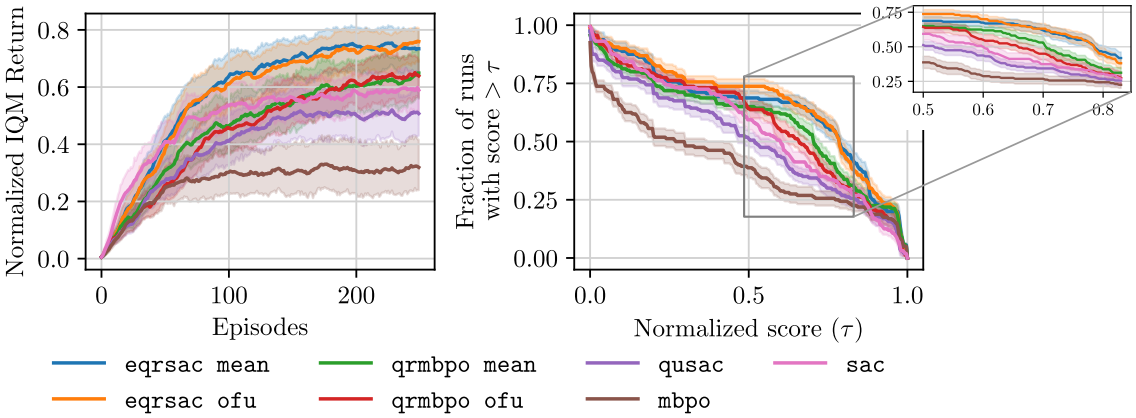

Figure 10: Aggregated performance in DMC benchmark with 95% bootstrap confidence intervals. **(Left)** Inter-quartile mean returns normalized by the maximum achievable score of 1000. **(Right)** Performance profile at 250 episodes of training, with zoom in region with the most spread in results. In both cases, higher curves correspond to better performance.

between EQR-SAC and QR-MBPO: (i) EQR-SAC has a buffer n times bigger than QR-MBPO, and correspondingly scales up the amount of data collected under the ensemble, (ii) EQR-SAC uses consistent rollouts, while QR-MBPO randomizes the model per-step, and (iii), EQR-SAC's critic is trained on the loss (18), while QR-MBPO's critic uses (21). In order to test the impact of each component in isolation, we add three additional baselines: QR-MBPO-big, which uses the same buffer size and collects the same amount of data as EQR-SAC; QR-MBPO-consistent, that replicates how EQR-SAC collects data under the model; and QR-MBPO-loss, that uses (18) to train its critic. Figure 11 shows the performance of EQR-SAC and all QR-MBPO variants (all methods optimize the actor using f_{mean}). The main observation is that QR-MBPO-loss matches closely the performance of EQR-SAC, while all other QR-MBPO variants share similar (lower) performance. The key insight from these results is that our proposed critic loss function (18) is instrumental towards sample-efficient learning, especially in environments with sparse rewards like cartpole swing-up (see also fish-swim and finger-spin in Appendix E). As such, our theory provides a solid guideline on how to integrate model-based RL architectures with distributional RL tools, which goes beyond simply using a distributional critic with established algorithms like MBPO.

6.5 Dynamics Sampling for Target Quantiles

The analysis in Section 6.4 points to the loss function (18) as being the key component of our proposed approach. The main feature of our loss function is how it utilizes the generative model to produce the target quantiles (19). In this experiment, we investigate the effect of the amount of next state-action samples (s', a') drawn from the ensemble of dynamics when estimating the target quantiles. The hypothesis is that a larger sample size would result in a lower-variance estimate of the expectation in (19), which could then lead to better sample-efficiency. Figure 12 shows the performance of EQR-SAC, for both f_{mean} and f_{ofu}, under different sampling regimes. For the cartpole task, we observe a clear progression in

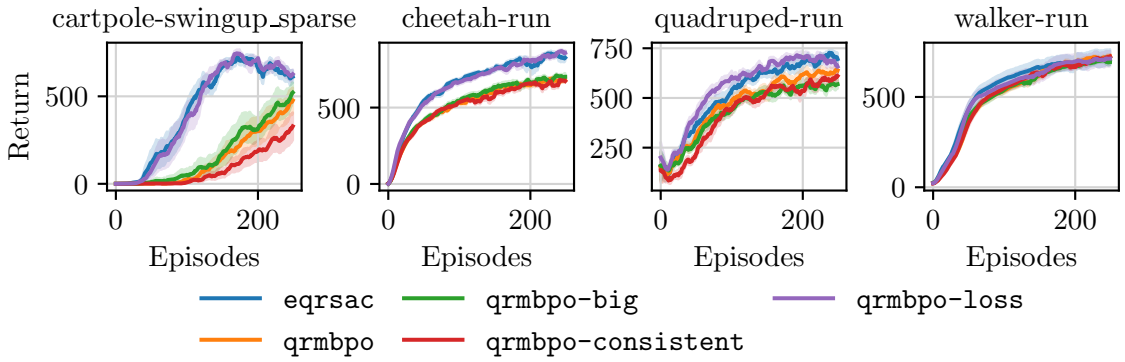

Figure 11: Comparison of EQR-SAC and QR-MBPO baselines in selected DeepMind Control tasks. The results suggest that the biggest contributing factor for increased performance of EQR-SAC w.r.t QR-MBPO is the critic's loss function (18).

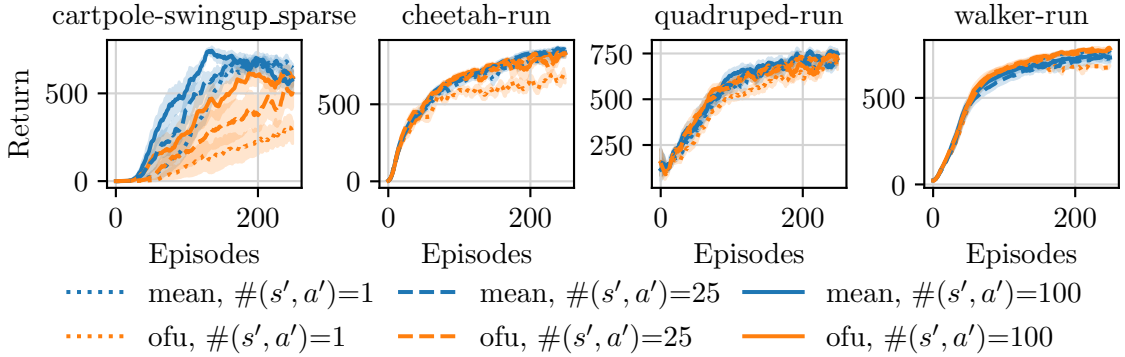

Figure 12: Ablation study on the amount of next state-action samples drawn to approximate the target quantiles (19). Larger sample sizes perform more robustly across all environments.

sample-efficiency as the amount of samples increases. For other environments the differences are less noticeable, but using a sample size of 1 with the optimistic objective leads to worse performance in all cases. Since the sample size might have large effects in performance and its runtime impact is greatly amortized by GPU parallelization, the overall recommendation is to use larger sample sizes (25-100) by default.

6.6 Additional Experiments

In Appendix G, we include additional ablation studies on three hyperparameters of our algorithm: the number of quantiles (m), the rollout length (k) and the size of $\mathcal{D}_{\text{model}}$ (which controls the amount of off-policy data in the buffer). The general observation from these experiments is that EQR-SAC's performance is robust for a wide range of values. Performance typically degrades only for extreme values of the parameters, for example $m = 1$ (only estimate median of value distribution) or $k = 1$ (only generate 1-step rollouts with the model).

Furthermore, in Appendix H we conduct experiments to investigate the utility of optimistic values in policy optimization. The empirical results suggest that optimism has little effect under dense rewards, while under sparse rewards higher optimism can be benefitial in some tasks and detrimental in others. We also study the effect of optimism in tasks with sparse rewards *and* action costs, similar to the MountainCar problem of Section 6.2. For the tested environments, higher optimism does not improve final performance and is typically less sample efficient, unlike the MountainCar results. Overall, all these results indicate that the benefits of optimistic optimization might be environment-dependent. We believe an interesting avenue for future work is to more broadly analyze this phenomenon and reconsider our design choices (ensemble as posterior MDP, quantile representation for values, policy objectives, etc.).

7. Conclusions

We investigated the problem of estimating the distribution of values, given parameter uncertainty of the MDP. We proposed the value-distributional Bellman equation and extracted an operator whose fixed-point is precisely the distribution of values. Leveraging tools from return-distributional RL, we designed Epistemic Quantile-Regression, an iterative procedure for estimating quantiles of the value distribution. We applied our algorithm in small MDPs, validated the convergence properties prescribed by our theory and assessed its limitations once the main assumptions are violated. Lastly, we introduced EQR-SAC, a novel model-based deep RL algorithm that scales up EQR with neural network function approximation and combines it with SAC for policy optimization. We benchmarked our approach in several continuous-control tasks from the DeepMind Control suite and showed improved sample-efficiency and final performance compared to various baselines.

Appendix A. Unrolling MDP with Cycles

In Figure 13 we show the unrolling procedure of an MDP that contains cycles. The two MDPs are *not* equivalent, but the unrolled approximation improves as H grows.

While unrolling the MDP is rarely done in practice, it serves as a reasonable approximation to extend our theoretical results to the general setting of MDPs that contain cycles.

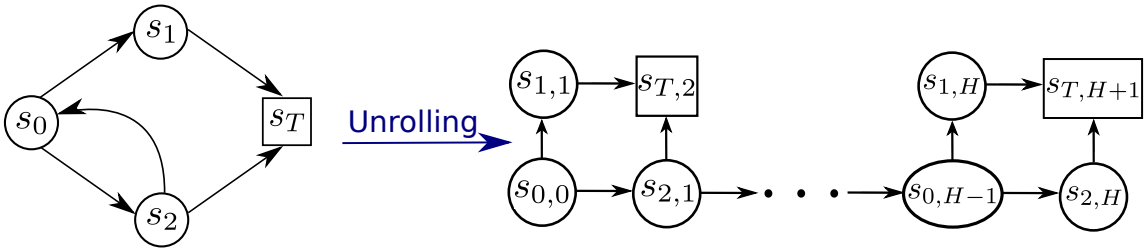

Figure 13: Procedure of unrolling an MDP with cycles. We denote by $s_{i,k}$ the unrolled state which represents being in state s_i of the original MDP at time step k. The unrolled MDP is only an *approximation* of the original version due to truncation after finite H steps.

Appendix B. Theory Proofs

Proposition 1 (Random Variable Value-Distribution Bellman Equation) *Let V^π be the random value function defined in* (3)*. Then, it holds that*

$$V^\pi(s) = \sum_a \pi(a \mid s) r(s,a) + \gamma \sum_{s',a} \pi(a \mid s) P(s' \mid s,a) V^\pi(s'), \tag{5}$$

for any policy π and initial state $s \in \mathcal{S}$.

Proof We proceed similarly as the standard Bellman equation proof shown by Bellemare et al. (2023). First, the random trajectories $\tilde{T}$ have two properties endowed by the Markov decision process: time homogeneity and the Markov property. Informally speaking, time homogeneity states that the trajectory from a given state s is independent of the time k at which the state is visited, while the Markov property states that trajectories starting from s are independent of states, actions or rewards encountered before s (c.f. Bellemare et al. (2023) Lemmas 2.13, 2.14 for a formal definition). In the domain of random variables, these properties imply that two trajectories starting from the same initial state s are equally distributed regardless of past history.

From the definition (3) we decompose the random value into the immediate reward and the value at the next state:

$$V^\pi(s) = \mathbb{E}_{\tilde{T}}[R_0 | S_0 = s,\, P] + \gamma\, \mathbb{E}_{\tilde{T}}\left[\sum_{h=1}^{\infty} \gamma^{h-1} R_h \middle| S_0 = s,\, P\right]. \tag{23}$$

For the first term, the only random variable remaining is A_0, so we rewrite it as

$$= \sum_a \pi(a \mid s) r(s,a) + \gamma\, \mathbb{E}_{\tilde{T}}\left[\sum_{h=1}^{\infty} \gamma^{h-1} R_h \middle| S_0 = s,\, P\right]. \tag{24}$$

For the second term, we apply the tower property of expectations

$$= \sum_a \pi(a \mid s) r(s,a) + \gamma \, \mathbb{E}_{\tilde{T}} \left[\mathbb{E}_{\tilde{T}} \left[\sum_{h=1}^{\infty} \gamma^{h-1} R_h \middle| S_0 = s,\, A_0,\, S_1,\, P \right] \middle| S_0 = s,\, P \right]. \quad (25)$$

By the Markov property,

$$= \sum_a \pi(a \mid s) r(s,a) + \gamma \, \mathbb{E}_{\tilde{T}} \left[\mathbb{E}_{\tilde{T}} \left[\sum_{h=1}^{\infty} \gamma^{h-1} R_h \middle| S_1,\, P \right] \middle| S_0 = s,\, P \right]. \quad (26)$$

By time homogeneity, the inner expectation is exactly equal to the random variable $V^\pi(S_1)$, after a change of variable in the infinite sum index

$$= \sum_a \pi(a \mid s) r(s,a) + \gamma \, \mathbb{E}_{\tilde{T}} \big[V^\pi(S_1) \big| S_0 = s,\, P \big]. \quad (27)$$

Lastly, the remaining random variable is S_1, for which we can explicitly write its probability distribution, concluding the proof

$$= \sum_a \pi(a \mid s) r(s,a) + \gamma \sum_{a,s'} \pi(a \mid s) P(s' \mid s,a) V^\pi(s'). \quad (28)$$

■

Notation: for the following Lemma, we use the notation $\mathfrak{D}(X)$ to denote the distribution of the random variable $X \in \mathcal{X}$, as done in Bellemare et al. (2023). In particular, we use $\mathfrak{D}_P$ to denote the distribution of a random variable belonging to the probability space of P, i.e., random variables derived from the posterior distribution $\Phi(P \mid \mathcal{D})$.

Lemma 1 (Value-Distribution Bellman Equation) *The value distribution function μ^π obeys the Bellman equation.*

$$\mu^\pi = \mathbb{E}_P \big[(b_{r^\pi, P^\pi, \gamma})_\# \mu^\pi \big] \quad (8)$$

for any policy π.

Proof In matrix-vector format, the random-variable value-distributional Bellman equation is expressed as

$$\mathbf{V}^\pi = \mathbf{r}^\pi + \gamma \mathbf{P}^\pi \mathbf{V}^\pi. \quad (29)$$

Since μ^π refers to the distribution of the random variable $\mathbf{V}^\pi$, which belongs to the probability space of the random transition function P, then we use our notation to write $\mu^\pi = \mathfrak{D}_P(\mathbf{V}^\pi)$. Further, using the notation $\mathfrak{D}_P(\cdot)$ on the r.h.s of (29) yields

$$\mu^\pi = \mathfrak{D}_P(\mathbf{r}^\pi + \gamma \mathbf{P}^\pi \mathbf{V}^\pi). \quad (30)$$

For any two random variables X, Y in the same probability space, it holds that the marginal distribution over X can be written as the expected value over Y of the conditional distribution.

That is, $\mathfrak{D}(X) = \mathbb{E}_Y\big[\mathfrak{D}(X \mid Y)\big]$, which follows from standard probability theory (Wasserman, 2013). Applying this property to the r.h.s of (30) results in

$$\mu^\pi = \mathbb{E}_P\big[\mathfrak{D}_P(\mathbf{r}^\pi + \gamma \mathbf{P}^\pi \mathbf{V}^\pi \mid \mathbf{P}^\pi)\big]. \tag{31}$$

A similar derivation can be found in related prior work studying the variance of μ^π (O'Donoghue et al., 2018; Zhou et al., 2019; Luis et al., 2023).

Given that $P(s' \mid s, a)$ and $V^\pi(s')$ are independent under our assumptions, then conditioning on $\mathbf{P}^\pi$ means that the distribution of the matrix-vector product $\mathbf{P}^\pi \mathbf{V}^\pi$ is simply the distribution of applying a linear transformation on $\mathbf{V}^\pi$. The result is that the conditional distribution can be interpreted as the pushforward

$$\mathfrak{D}_P(\mathbf{r}^\pi + \gamma \mathbf{P}^\pi \mathbf{V}^\pi \mid \mathbf{P}^\pi) = (b_{r^\pi, P^\pi, \gamma})_\# \mu^\pi, \tag{32}$$

which completes the proof. ■

We adopt the supremum p-Wasserstein distance to establish contractivity of the operator $\mathcal{T}^\pi$.

Definition 4 *For $p \in [1, \infty)$, the p-Wasserstein distance between two distributions ν, ν' is a metric $w_p : \mathcal{P}(\mathbb{R}) \times \mathcal{P}(\mathbb{R}) \to [0, \infty]$ defined by*

$$w_p(\nu, \nu') = \left(\int_0^1 \left| F_\nu^{-1}(\tau) - F_{\nu'}^{-1}(\tau) \right|^p d\tau \right)^{1/p}, \tag{33}$$

where $F_{(\cdot)}^{-1}$ is the inverse cumulative distribution function. Furthermore, the supremum p-Wasserstein distance $\bar{w}_p$ between two value distribution functions $\mu, \mu' \in \mathcal{P}(\mathbb{R})^{\mathcal{S}}$ is defined by $\bar{w}_p(\mu, \mu') = \sup_{s \in \mathcal{S}} w_p(\mu(s), \mu'(s))$.

The supremum p-Wasserstein distance was proven to be a metric in $\mathcal{P}(\mathbb{R})^{\mathcal{S}}$ by Bellemare et al. (2017).

To prove that $\mathcal{T}^\pi$ is a contraction, we adopt the technique from Bellemare et al. (2023) that relies on the alternative definition of the p-Wasserstein distance in terms of couplings.

Definition 5 (Coupling (Villani, 2008) Definition 1.1 (adapted)) *Let $\nu, \nu' \in \mathcal{P}(\mathbb{R})$ be two probability distributions over the reals. A coupling between ν and ν' is a joint probability distribution $\upsilon \in \mathcal{P}(\mathbb{R}^2)$ whose marginals are ν and ν'. That is, given random variables $(V, V') \sim \upsilon$, we have $V \sim \nu$ and $V' \sim \nu'$. Further, we denote $\Gamma(\nu, \nu') \subseteq \mathcal{P}(\mathbb{R}^2)$ the set[10] of all couplings between ν and ν'.*

Intuitively, the coupling υ can be interpreted as a transport plan to move probability mass from one distribution to another. The p-Wasserstein distance can also be defined as the cost of the optimal transport plan

$$w_p(\nu, \nu') = \min_{\upsilon \in \Gamma(\nu, \nu')} \mathbb{E}_{(V, V') \sim \upsilon}\big[|V - V|^p\big]^{1/p}. \tag{34}$$

10. This set is non-empty: there exists a trivial coupling in which the variables V, V' are independent (Villani, 2008)

The existence of an optimal coupling $\upsilon^\star$ that minimizes (34) is guaranteed since ν, ν' are measures defined on a complete, separable metric space ($\mathbb{R}$ with the usual metric) and equipped with the corresponding Borel σ-algebra (i.e., ν, ν' are measures on a Polish space) (cf. Villani, 2008, Theorem 4.1).

With these definitions, we now proceed to prove the contraction of the Bellman operator.

Theorem 1 *The operator $\mathcal{T}^\pi$ is a γ-contraction with respect to $\bar{w}_p$ for all $p \in [1, \infty)$. That is, $\bar{w}_p(\mathcal{T}^\pi \mu, \mathcal{T}^\pi \mu') \leq \gamma \bar{w}_p(\mu, \mu')$ for all $\mu, \mu' \in \mathcal{P}(\mathbb{R})^{\mathcal{S}}$ such that $V(s') \sim \mu(s')$, $V'(s') \sim \mu'(s')$ are conditionally independent of $P^\pi(s' \mid s)$ given $s' \in \mathcal{S}$.*

Proof We follow closely the proofs of Proposition 4.1 by Amortila et al. (2020) and Proposition 4.15 by Bellemare et al. (2023). For each $s \in \mathcal{S}$, let $\upsilon^\star$ denote the optimal coupling that minimizes the p-Wasserstein metric from Definition 4 between some arbitrary pair of value distributions $\mu(s), \mu'(s) \in \mathcal{P}(\mathbb{R})$, so that $(V(s), V'(s)) \sim \upsilon^\star$.

Define new random variables $\tilde{V}(s) = r^\pi(s) + \gamma \sum_{s'} P^\pi(s' \mid s) V(s')$, $\tilde{V}'(s) = r^\pi(s) + \gamma \sum_{s'} P^\pi(s' \mid s) V'(s')$. By definition of the operator $\mathcal{T}^\pi$, we have that $\tilde{V}(s) \sim (\mathcal{T}^\pi \mu)(s)$ and $\tilde{V}'(s) \sim (\mathcal{T}^\pi \mu')(s)$, which means that the pair $(\tilde{V}(s), \tilde{V}'(s)) \sim \tilde{\upsilon}$ is a coupling between $(\mathcal{T}^\pi \mu)(s)$ and $(\mathcal{T}^\pi \mu')(s)$.

Starting from Definition 5 and since the p-Wasserstein distance is a minimum over couplings, then

$$w_p^p\big((\mathcal{T}\mu)(s), (\mathcal{T}\mu')(s)\big) \leq \mathbb{E}_P\left[\left|\tilde{V}(s) - \tilde{V}'(s)\right|^p\right]. \tag{35}$$

Plugging the definition of the random variables,

$$= \mathbb{E}_P\left[\left|r^\pi(s) + \gamma \sum_{s'} P^\pi(s' \mid s) V(s') - r^\pi(s) - \gamma \sum_{s'} P^\pi(s' \mid s) V'(s')\right|^p\right] \tag{36}$$

By re-arrangement of terms

$$= \gamma^p \, \mathbb{E}_P\left[\left|\sum_{s'} P^\pi(s' \mid s)(V(s') - V'(s'))\right|^p\right]. \tag{37}$$

Since $f(x) = |x|^p$ is convex for $p \geq 1$, then by Jensen's inequality

$$\leq \gamma^p \, \mathbb{E}_P\left[\sum_{s'} P^\pi(s' \mid s)\big|(V(s') - V'(s'))\big|^p\right]. \tag{38}$$

By linearity of expectation

$$= \gamma^p \sum_{s'} \mathbb{E}_P\Big[P^\pi(s' \mid s)\big|(V(s') - V'(s'))\big|^p\Big]. \tag{39}$$

By the independence assumption on $P^\pi(s' \mid s)$, the expectation of the product becomes the product of expectations

$$= \gamma^p \sum_{s'} \mathbb{E}_P\big[P^\pi(s' \mid s)\big]\, \mathbb{E}_P\Big[\big|(V(s') - V'(s'))\big|^p\Big]. \tag{40}$$

Since the supremum of non-negative values is greater or equal than any convex combination of them

$$\leq \gamma^p \sup_{s'} \mathbb{E}_P\Big[\big|(V(s') - V'(s'))\big|^p\Big]. \tag{41}$$

By definition of the supremum p-Wasserstein distance

$$= \gamma^p \bar{w}_p^p(\mu, \mu'). \tag{42}$$

Taking supremum on the left-hand side and taking the p-th root on both sides completes the proof. ■

Theorem 1 parallels similar results in standard RL and model-free distributional RL, in that it allows us to establish the convergence of iterated applications of $\mathcal{T}^\pi$ (Corollary 1).

B.1 Supporting Lemmas

Lemma 2 *Under Assumptions 1–3, $P(s' \mid s, a)$ and $V^\pi(s')$ are conditionally independent random variables for all triplets $(s, a, s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$.*

Proof Let $T_{0:\infty}$ be a random trajectory under the random transition dynamics P. Under Assumptions 2 and 3, $T_{0:\infty}$ is a sequence of H random, but *unique* states followed by the terminal (absorbing) state $\{S_0, S_1, \ldots, S_H, s_T, s_T, \ldots\}$, i.e., we have $S_i \neq S_j$ for all $i \neq j$. We prove the Lemma by dividing the random trajectory into two segments: the random (finite) segment for step $h \leq H$ and the deterministic (infinite) segment for $h > H$.

Case $T_{0:H}$. Under Assumption 1, the conditioned trajectory probability $\mathbb{P}(T_{0:H} \mid P)$, which is itself a random variable through conditioning on P, is a product of independent random variables defined by

$$\mathbb{P}(T_{0:H} \mid P) = \prod_{h=0}^{H-1} \pi(A_h \mid S_h) P(S_{h+1} \mid S_h, A_h) \tag{43}$$

$$= P(S_1 \mid S_0, A_0)\pi(A_0 \mid S_0) \prod_{h=1}^{H-1} \pi(A_h \mid S_h) P(S_{h+1} \mid S_h, A_h). \tag{44}$$

$$= P(S_1 \mid S_0, A_0)\pi(A_0 \mid S_0)\, \mathbb{P}(T_{1:H} \mid P). \tag{45}$$

Note that each transition probability in $\mathbb{P}(T_{0:H} \mid P)$ is distinct by Assumption 2. Additionally, for any $h > 0$ the action probability $\pi(A_h \mid S_h)$ is *conditionally independent* of $P(S_h \mid S_{h-1}, A_{h-1})$ given S_h. Then, for arbitrary $S_0 = s$, $A_0 = a$ and $S_1 = s'$, we have that $P(s' \mid s, a)$ is conditionally independent of $\mathbb{P}(T_{1:H-1} \mid P)$. Since $V^\pi(S_1 \mid S_1 = s')$ is a function of $\mathbb{P}(T_{1:H} \mid P)$, then it is also conditionally independent of $P(s' \mid s, a)$.

Case $T_{H:\infty}$. The Lemma is trivially satisfied since both the transition probability and the values become constants: we have $P(s_T \mid s_T, a) = 1$ and $V^\pi(s_T) = 0$.

Combining both results, we have that P and V^π are conditionally independent for any arbitrary infinite trajectory, which completes the proof. ■

Lemma 3 *Define an arbitrary $\mu_0 \in \mathcal{P}_B(\mathbb{R})^{\mathcal{S}}$. Then, under Assumptions 1–3, the sequence $\{\mu_k\}_{k=0}^{\infty}$ defined by $\mu_{k+1} = \mathcal{T}^\pi \mu_k$ is such that for all $k \geq 0$ the random variable $V_k(s') \sim \mu_k(s')$ is conditionally independent of $P^\pi(s' \mid s)$ given $s' \in \mathcal{S}$.*

Proof Intuitively, by definition of the operator $\mathcal{T}^\pi$, the random variable $V_k(s')$ is the result of summing rewards starting from state s', following the random dynamics P^π for k steps and then bootstraping with V_0. That is, applying $\mathcal{T}^\pi$ k times results in the random variable

$$V_k(s') = \mathbb{E}_{T_{0:k}}\left[\sum_{h=0}^{k-1} \gamma^h R_h + \gamma^k V_0(S_k) \middle| S_0 = s', P\right], \tag{46}$$

where $T_{0:k}$ is a random state trajectory $\{S_0, S_1, ..., S_k\}$ starting from $S_0 = s'$ and following the random dynamics P. Under Assumptions 1–3, $T_{0:k}$ is a sequence of unique states (with the exception of the absorbing terminal state), which means that the probability of returning to s' are zero. Finally, since $V_k(s')$ is an expectation conditioned on the initial state being s', it follows that $V_k(s')$ is conditionally independent of $P^\pi(s' \mid s)$ given s'. ■

Lemma 4 *If the value distribution function μ has bounded support, then $\mathcal{T}^\pi \mu$ also has bounded support.*

Proof From bounded rewards on $[r_{\min}, r_{\max}]$, then we denote by $\mathcal{P}_B(\mathbb{R})^{\mathcal{S}}$ the space of value distributions bounded on $[v_{\min}, v_{\max}]$, where $v_{\min} = r_{\min}/(1-\gamma)$ and $v_{\max} = r_{\max}/(1-\gamma)$.

Given arbitrary $\mu \in \mathcal{P}_B(\mathbb{R})^{\mathcal{S}}$, let $v(s)$ be a realization of $\mu(s)$ for any $s \in \mathcal{S}$. Then, $\sum_a \pi(a \mid s) r(s,a) + \gamma \sum_{a,s'} \pi(a \mid s) P(s' \mid s, a) v(s')$ is an instantiation of $(\mathcal{T}^\pi \mu)(s)$ for any $s \in \mathcal{S}$. We have:

$$\mathbb{P}\big((\mathcal{T}^\pi \mu)(s) \leq v_{\max}\big) = \mathbb{P}\left(\sum_a \pi(a \mid s) r(s,a) + \gamma \sum_{a,s'} \pi(a \mid s) P(s' \mid s, a) v(s') \leq v_{\max}\right), \tag{47}$$

$$= \mathbb{P}\left(\gamma \sum_{a,s'} \pi(a \mid s) P(s' \mid s, a) v(s') \leq v_{\max} - \sum_a \pi(a \mid s) r(s,a)\right). \tag{48}$$

Since $\sum_a \pi(a \mid s) r(s,a) \leq r_{\max}$, then

$$\geq \mathbb{P}\left(\gamma \sum_{a,s'} \pi(a \mid s) P(s' \mid s, a) v(s') \leq v_{\max} - r_{\max}\right). \tag{49}$$

By definition of $v_{\max}$

$$\geq \mathbb{P}\left(\sum_{a,s'} \pi(a \mid s) P(s' \mid s, a) v(s') \leq v_{\max}\right). \tag{50}$$

Finally, since $v(s') \leq v_{\max}$ for any $s' \in \mathcal{S}$, then

$$= 1. \tag{51}$$

Under the same logic, we can similarly show that $\mathbb{P}\big((\mathcal{T}^\pi \mu)(s) \geq v_{\min}\big) = 1$, such that $\mathbb{P}\big((\mathcal{T}^\pi \mu)(s) \in [v_{\min}, v_{\max}]\big) = 1$ for any $s \in \mathcal{S}$. ■

Appendix C. Implementation Details

C.1 Quantile Huber Loss

We adopt the quantile Huber loss from Dabney et al. (2018b) in order to train the distributional critic. The Huber loss is given by

$$\mathcal{L}_\kappa(u) = \begin{cases} \frac{1}{2}u^2, & \text{if } |u| \leq \kappa \\ \kappa(|u| - \frac{1}{2}\kappa) & \text{otherwise} \end{cases}, \tag{52}$$

and the quantile Huber loss is defined by

$$\rho_\tau^\kappa(u) = \big|\tau - \delta(u < 0)\big| \mathcal{L}_\kappa(u). \tag{53}$$

For $\kappa = 0$, we recover the standard quantile regression loss, which is not smooth as $u \to 0$. In all our experiments we fix $\kappa = 1$ and to simplify notation define $\rho_\tau^1 = \rho_\tau$.

C.2 EQR-SAC Algorithm

A detailed execution flow for training an EQR-SAC agent is presented in Algorithm 2. Further implementation details are now provided.

Model learning. We use the `mbrl-lib` Python library from Pineda et al. (2021) to train N neural networks (Line 7). Our default architecture consists of four fully-connected layers with 200 neurons each (for the Quadruped environments we use 400 neurons to accomodate the larger state space). The networks predict delta states, $(s' - s)$, and receives as inputs normalizes state-action pairs. The normalization statistics are updated each time we train the model and are based on the training dataset $\mathcal{D}$. We use the default initialization that samples weights from a truncated Gaussian distribution, but we increase by a factor of 2 the standard deviation of the sampling distribution.

Capacity of $\mathcal{D}_{\text{model}}$. The capacity of the model buffer is computed as $k \times L \times F \times N \times \Delta$, where Δ is the number of model updates we want to retain data in the buffer. That is, the buffer is filled only with data from the latest Δ rounds of model training and data collection (Lines 6-10).

Algorithm 2 Epistemic Quantile-Regression with Soft Actor-Critic (EQR-SAC)

1: Initialize policy π_ϕ, MDP ensemble Γ_ψ, quantile critic q, environment dataset $\mathcal{D}$, model dataset $\mathcal{D}_{\text{model}}$, utility function f.
2: Warm-up $\mathcal{D}$ with rollouts under π_ϕ
3: global step $\leftarrow 0$
4: **for** episode $t = 0, \dots, T-1$ **do**
5: **for** E steps **do**
6: **if** global step $\%$ $F == 0$ **then**
7: Train Γ_ψ on $\mathcal{D}$ via maximum likelihood
8: **for** each MDP dynamics in Γ_ψ **do**
9: **for** L model rollouts **do**
10: Perform k-step rollouts starting from $s \sim \mathcal{D}$; add to $\mathcal{D}_{\text{model}}$
11: Take action in environment according to π_ϕ; add to $\mathcal{D}$
12: **for** G gradient updates **do**
13: Update $\{q_i\}_{i=1}^m$ with mini-batches from $\mathcal{D}_{\text{model}}$, via SGD on (18)
14: Update π_ϕ with mini-batches from $\mathcal{D}_{\text{model}}$, via SGD on (20)
15: global step $\leftarrow$ global step $+1$

Critic Loss. The distributional critic is updated in Line 13, for which we use the loss function (18). To approximate the target quantiles (19), we use the learned generative model and the policy to generate transition tuples (r, s', a'). More specifically, each (s, a) pair in a mini-batch from $\mathcal{D}_{\text{model}}$ is repeated X times and passed through every member of the ensemble of dynamics, thus generating n batches of X predictions (r, s'). Then, every s' prediction is repeated Y times and passed through π_ϕ, thus obtaining XY next state-action pairs (s', a'). This generated data is finally used in (19) to better approximate the expectation. In our experiments we use $X = Y$ and keep their product as a hyperparameter controlling the total amount of samples we use to approximate the expectation.

Reference Implementation. We use a single codebase for all experiments and share architecture components amongst baselines whenever possible. The execution of experiments follows the workflow of Algorithm 2. The SAC base implementation follows the open-source repository `https://github.com/pranz24/pytorch-soft-actor-critic` and we allow for either model-free or model-based data buffers for the agent's updates, as done in `mbrl-lib`.

Appendix D. Hyperparameters

Table 1: Hyperparameters for DeepMind Control Suite. In red, we highlight the only deviations of the base hyperparameters across all environments and baselines.

Name	Value
General	
T - # episodes	250
E - steps per episode	10^3
Replay buffer $\mathcal{D}$ capacity	10^5
Warm-up steps (under initial policy)	5×10^3
SAC	
G - # gradient steps	10
Batch size	256
Auto-tuning of entropy coefficient α?	Yes
Target entropy	$-\dim(\mathcal{A})$
Actor MLP network	2 hidden layers - 128 neurons - Tanh activations
Critic MLP network	2 hidden layers - 256 neurons - Tanh activations
Actor/Critic learning rate	3×10^{-4}
Dynamics Model	
n - ensemble size	5
F - frequency of model training (# steps)	250
L - # model rollouts per step	400
k - rollout length	5
Δ - # Model updates to retain data	10
Model buffer $\mathcal{D}_{\text{model}}$ capacity (EQR-SAC)	$L \times F \times k \times \Delta (\times n) = 5 \times 10^6 (25 \times 10^6)$
Model MLP network (quadruped)	4 layers - 200 (400) neurons - SiLU activations
Learning rate	1×10^{-3}
Quantile Network	
m - # quantiles	51
# (s', a') samples (EQR-SAC only)	25

Appendix E. DM Control Learning Curves

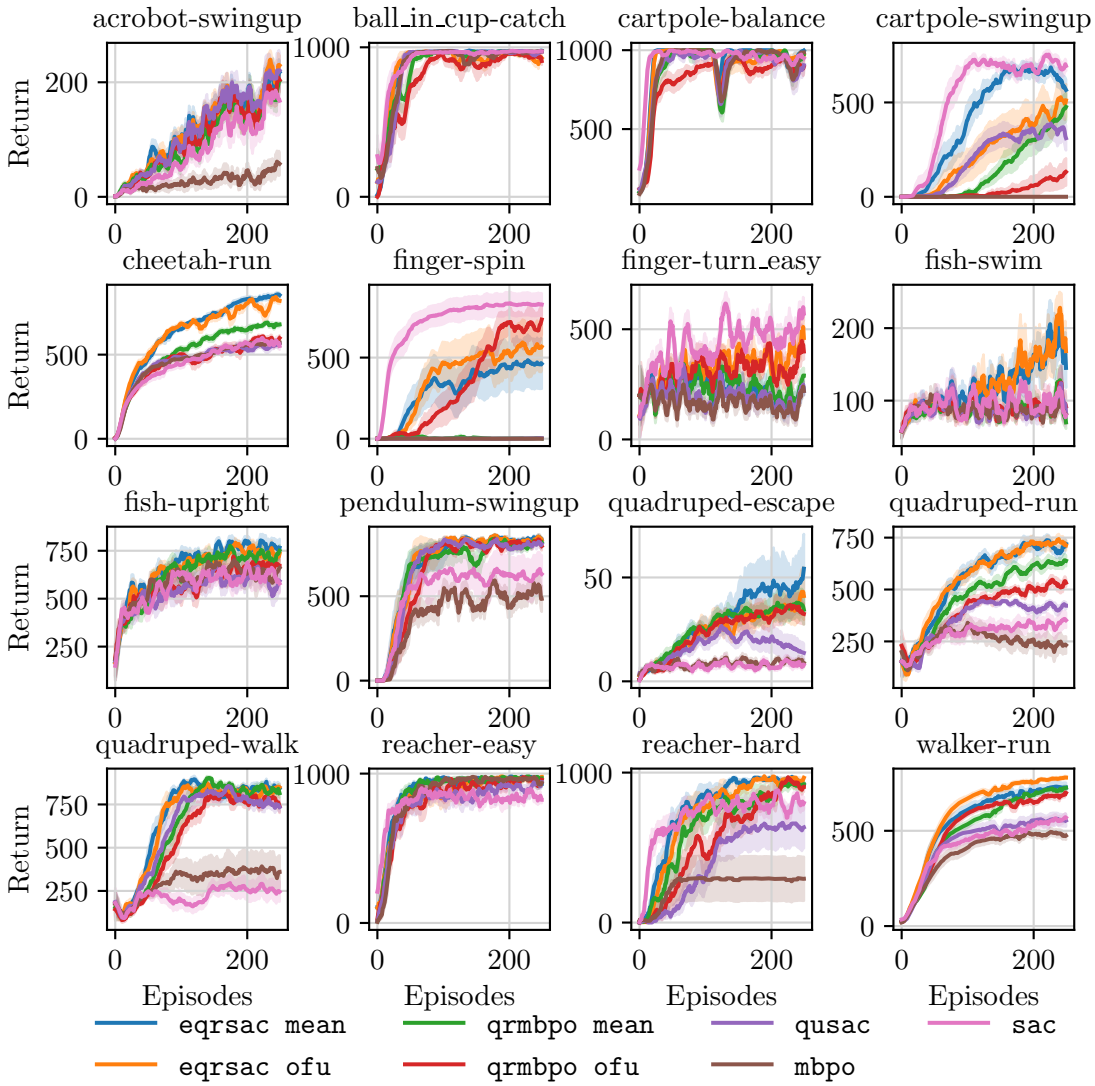

Figure 14: Individual learning curves of DMC benchmark.

Appendix F. DM Control Final Scores

Table 2: Scores in DMC benchmark after 250 episodes (or 250K environment steps). For each environment, we report the mean and standard error scores over 10 random seeds. We **bold** the highest final mean score per environment.

Environment	sac	mbpo	qrmbpo mean	qrmbpo ofu	qusac	eqrsac mean	eqrsac ofu
acrobot-swingup	167.7 ± 22.4	57.7 ± 19.9	202.3 ± 18.7	204.0 ± 21.4	220.3 ± 30.8	217.8 ± 18.5	**229.4** ± 17.4
ball-in-cup-catch	972.6 ± 3.3	972.5 ± 1.7	974.4 ± 2.3	908.3 ± 25.1	972.8 ± 2.8	**977.2** ± 1.4	928.7 ± 18.5
cartpole-balance-sparse	949.5 ± 18.7	985.6 ± 9.7	977.0 ± 22.2	894.9 ± 42.8	904.5 ± 37.3	**997.8** ± 2.1	968.0 ± 15.2
cartpole-swingup-sparse	**693.8** ± 27.2	0.1 ± 0.1	476.2 ± 72.6	131.8 ± 72.1	310.6 ± 64.3	566.4 ± 54.4	510.6 ± 86.9
cheetah-run	551.6 ± 22.6	571.4 ± 19.3	679.1 ± 10.7	598.4 ± 16.2	567.4 ± 16.8	**854.0** ± 11.7	820.0 ± 18.4
finger-spin	**827.5** ± 68.1	0.1 ± 0.1	1.2 ± 1.1	734.4 ± 89.7	3.2 ± 2.6	567.1 ± 146.7	461.9 ± 154.1
finger-turn-easy	**571.3** ± 31.3	220.0 ± 20.0	289.8 ± 34.1	399.0 ± 35.7	220.0 ± 20.0	221.3 ± 19.9	460.5 ± 58.3
fish-swim	79.9 ± 10.9	80.6 ± 10.0	70.0 ± 8.7	91.9 ± 13.3	83.8 ± 10.0	145.1 ± 27.3	**168.3** ± 20.4
fish-upright	579.4 ± 50.8	660.5 ± 59.5	749.9 ± 29.1	671.4 ± 32.2	591.9 ± 53.5	**766.2** ± 45.3	735.0 ± 23.5
pendulum-swingup	631.0 ± 111.7	484.5 ± 76.4	819.2 ± 17.2	796.9 ± 25.2	808.6 ± 19.7	**834.4** ± 15.7	833.7 ± 16.0
quadruped-escape	8.0 ± 1.2	8.8 ± 1.8	34.5 ± 7.1	32.4 ± 5.0	13.7 ± 4.3	**54.2** ± 16.7	41.1 ± 11.4
quadruped-run	352.0 ± 36.4	232.3 ± 41.2	638.5 ± 26.0	532.6 ± 19.0	421.9 ± 16.8	**719.8** ± 19.0	712.5 ± 23.6
quadruped-walk	245.5 ± 57.4	360.1 ± 95.1	815.1 ± 25.4	739.4 ± 38.2	734.8 ± 26.5	844.3 ± 26.8	**849.2** ± 17.1
reacher-easy	824.6 ± 21.9	474.3 ± 20.8	968.6 ± 9.8	959.1 ± 13.2	943.1 ± 13.8	931.2 ± 21.5	**977.9** ± 2.5
reacher-hard	797.5 ± 38.8	291.9 ± 146.3	921.8 ± 22.2	905.0 ± 31.6	635.0 ± 139.5	919.6 ± 15.7	**965.3** ± 9.8
walker-run	568.9 ± 19.1	474.3 ± 20.8	725.5 ± 10.8	698.9 ± 13.7	553.8 ± 30.9	727.4 ± 24.3	**779.3** ± 7.9

Table 3: Normalized inter-quartile mean scores in DMC benchmark after 100 and 250 episodes. For each aggregation metric, we report the point estimate and the 95% bootstrap confidence interval within parentheses, following the methodology in Agarwal et al. (2021). We **bold** the highest mean scores in each case.

Method	IQM-100	IQM-250
● eqrsac mean	**0.63** (0.55, 0.72)	0.73 (0.65, 0.81)
● eqrsac ofu	0.61 (0.53, 0.69)	**0.76** (0.69, 0.81)
● qrmbpo mean	0.46 (0.37, 0.56)	0.65 (0.56, 0.73)
● qrmbpo ofu	0.46 (0.38, 0.55)	0.64 (0.56, 0.69)
● qusac	0.41 (0.33, 0.50)	0.51 (0.43, 0.58)
● mbpo	0.30 (0.23, 0.39)	0.32 (0.24, 0.42)
● sac	0.54 (0.46, 0.61)	0.59 (0.52, 0.66)

Appendix G. Ablations

In this section, we present additional ablation studies on three salient hyperparameters of EQR-SAC: the number of quantiles (m), the rollout length (k) and the number of model updates to retain data (Δ). For all the experiments, we use the default hyperparameters in Table 1 and only vary the hyperparameter of the corresponding ablation study.

G.1 Number of Quantiles

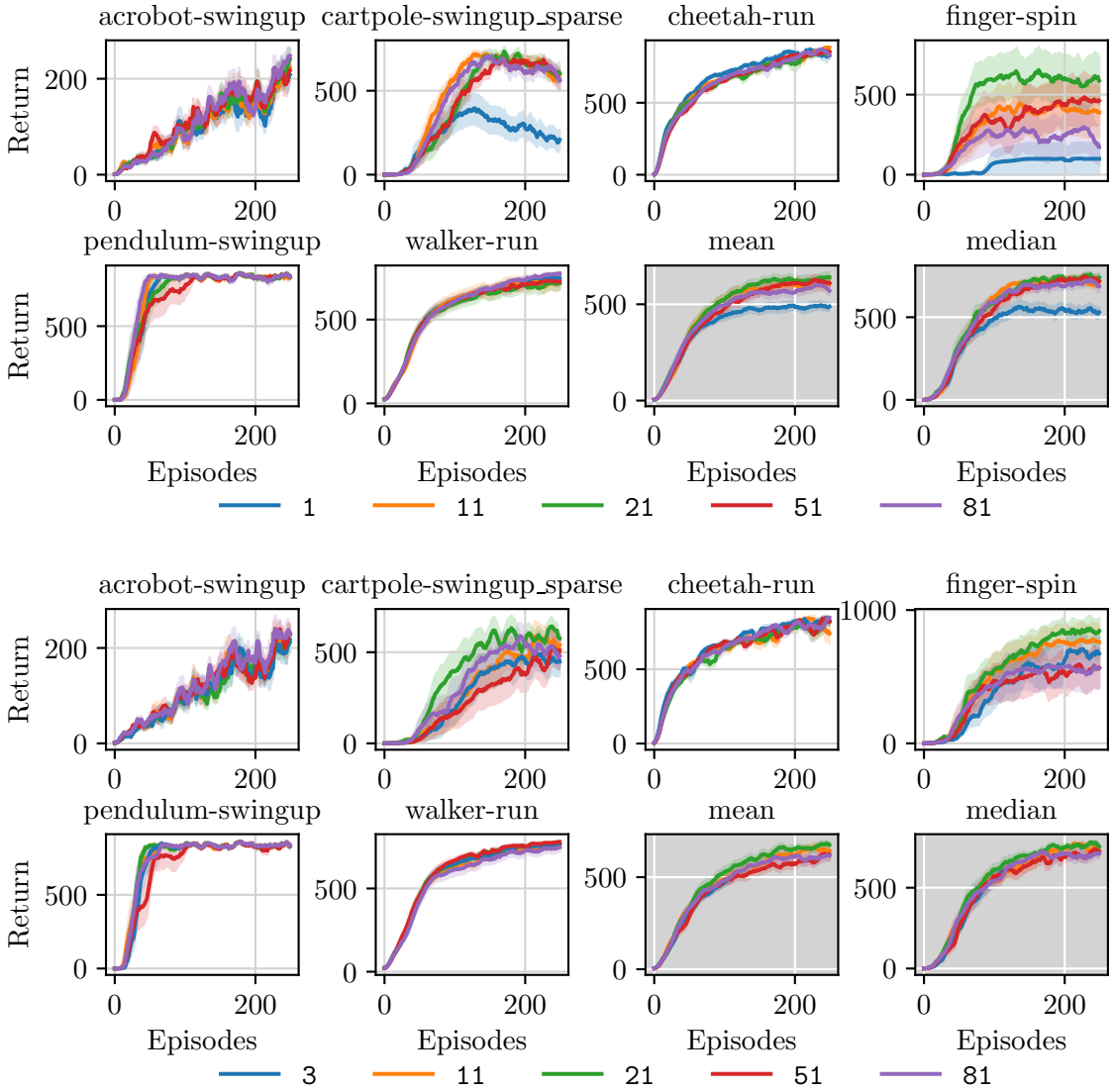

Figure 15: Number of quantiles (m) ablation study. **(Top)** EQR-SAC-mean. **(Bottom)** EQR-SAC-ofu. Note that for EQR-SAC-ofu we require $m > 1$ in order to estimate the standard deviation of quantiles for the optimistic objective function of the actor, thus we select a minimum value of $m = 3$ for this study.

G.2 Rollout Length

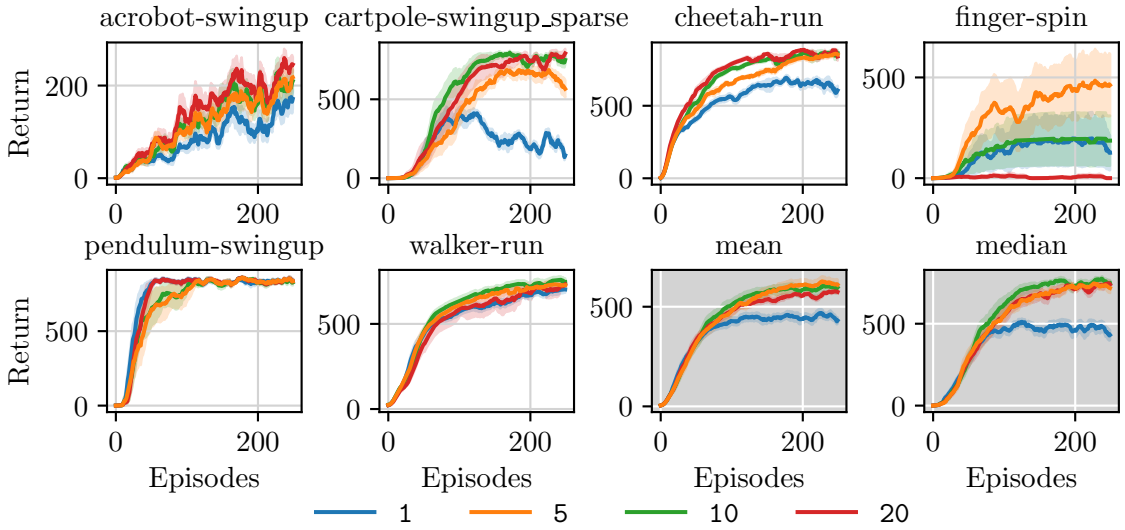

Figure 16: Rollout length (k) ablation study for EQR-SAC-mean.

G.3 Number of Model Updates to Retain Data

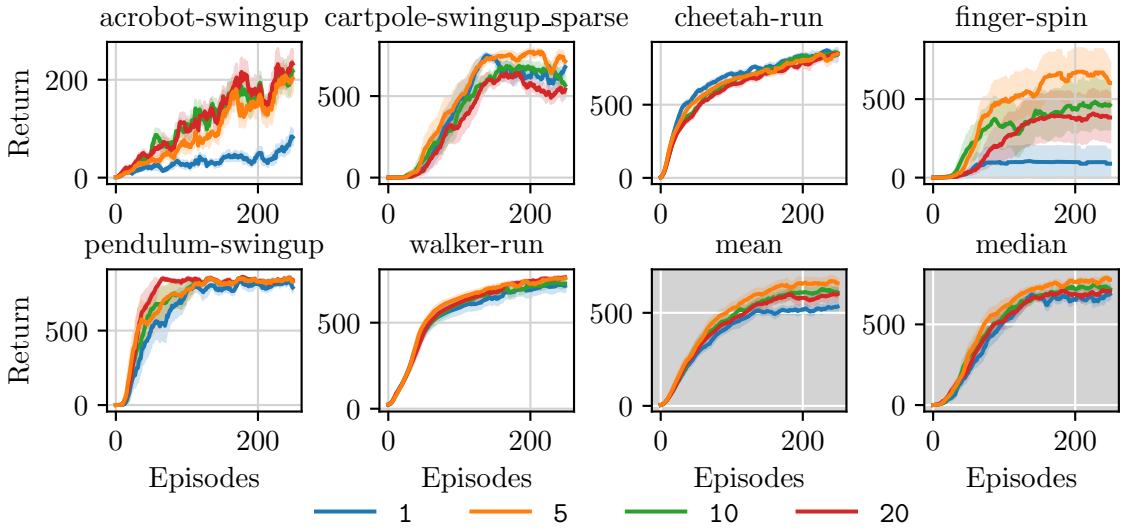

Figure 17: Number of model updates to retain data (Δ) ablation study for EQR-SAC-mean.

Appendix H. Optimistic Policy Optimization

In this section, we investigate the effect of using optimistic value estimates for policy optimization. To conduct the study, we propose a simple variant of EQR-SAC, named EQR-SAC-τ, which uses as the actor's objective function the closest quantile level to a given

target τ. For instance, in our experiments we use $m = 21$ and target levels $\{0.5, 0.7, 0.9\}$, which correspond to actual levels $\{0.5, 0.69, 0.88\}$.

Dense versus sparse rewards. We first investigate how optimism affects performance in environments with dense versus sparse rewards and present the results in Figure 18. For environments with dense rewards (cheetah, walker) optimism has little to no effect, while it results in largely different performance in envionments with sparse rewards (reacher-hard, finger-spin). Even though we would expect optimism to be generally helpful in all exploration tasks, our results indicate its effect is environment-dependent: the most optimistic objective ($\tau = 0.9$) performed worst in reacher-hard but obtained the best performance in finger-spin; inversely, the least optimistic objective ($\tau = 0.5$) performed the best in reacher-hard, but worst in finger-spin.

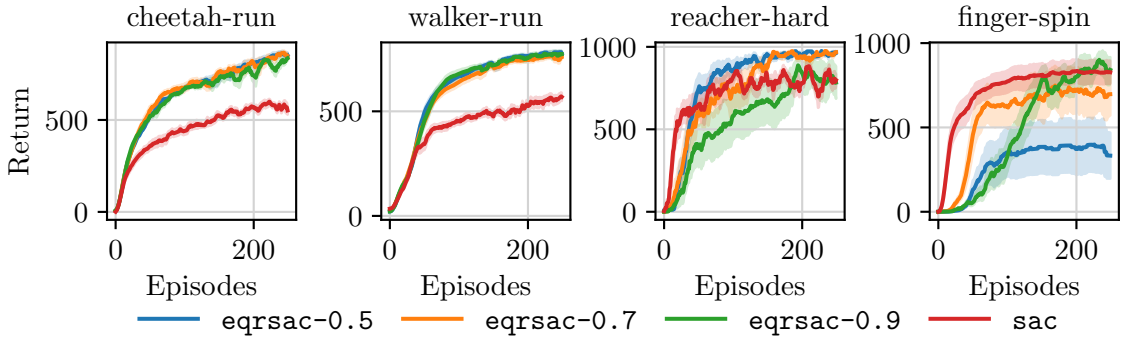

Figure 18: Evaluation of EQR-SAC-τ for different quantile levels. The two tasks on the left have dense rewards, while the other two have sparse rewards.

Action costs and sparse rewards. In Section 6.2, we observe that the combination of action costs and sparse rewards represents a pitfall for methods like SAC, especially since the optimal policy must issue large actions to observe the reward. Meanwhile, the quantile-based optimistic approaches performed best. In this experiment, we test the same setting in two tasks with sparse rewards from the DeepMind Control suite, where we add an action cost proportional to the squared norm of the action taken by the agent. Namely,

$$\texttt{action\_cost} = \rho \sum_{i=1}^{|\mathcal{A}|} a_i^2 \tag{54}$$

where ρ is an environment specific base multiplier, a_i is the i-th component of the action vector and $|\mathcal{A}|$ is the size of the action space. For `cartpole-swingup` we use $\rho = 0.001$ and for `pendulum` we use $\rho = 0.01$. The resuls in Figure 19 show a similar degradation of performance for SAC. Unlike the MountainCar experiments of Section 6.2, higher levels of optimism mostly resulted in less sample-efficient learning.

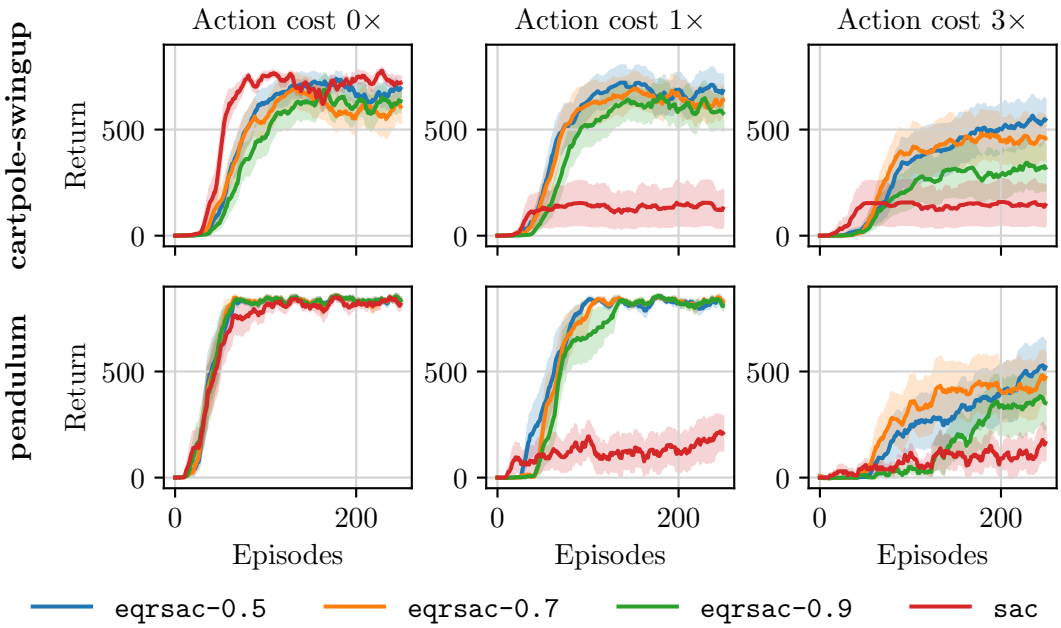

Figure 19: Evaluation of EQR-SAC-τ for different quantile levels and increasing action costs. The top row corresponds to the cartpole swingup task and the bottom row to the pendulum swingup. The action costs range from zero (left column), to a 3× multiplier on (54).

References

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep Reinforcement Learning at the Edge of the Statistical Precipice. In *Advances in Neural Information Processing Systems*, volume 34, pages 29304–29320. Curran Associates, Inc., 2021.

Philip Amortila, Doina Precup, Prakash Panangaden, and Marc G. Bellemare. A Distributional Analysis of Sampling-Based Reinforcement Learning Algorithms. In *International Conference on Artificial Intelligence and Statistics*, volume 108, pages 4357–4366. PMLR, August 2020.

Bahram Behzadian, Reazul Hasan Russel, Marek Petrik, and Chin Pang Ho. Optimizing Percentile Criterion using Robust MDPs. In *International Conference on Artificial Intelligence and Statistics*, volume 130, pages 1009–1017. PMLR, April 2021.

Marc G. Bellemare, Will Dabney, and Rémi Munos. A Distributional Perspective on Reinforcement Learning. In *International Conference on Machine Learning*, volume 70, pages 449–458. PMLR, August 2017.

Marc G. Bellemare, Nicolas Le Roux, Pablo Samuel Castro, and Subhodeep Moitra. Distributional Reinforcement Learning with Linear Function Approximation. In *International Conference on Artificial Intelligence and Statistics*, volume 89, pages 2203–2211. PMLR, April 2019.

Marc G Bellemare, Salvatore Candido, Pablo Samuel Castro, Jun Gong, Marlos C Machado, Subhodeep Moitra, Sameera S Ponda, and Ziyu Wang. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836):77–82, 2020.

Marc G. Bellemare, Will Dabney, and Mark Rowland. *Distributional Reinforcement Learning*. MIT Press, 2023.

Patrick Billingsley. *Probability and Measure*. John Wiley & Sons, third edition edition, 1995.

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

Yinlam Chow, Aviv Tamar, Shie Mannor, and Marco Pavone. Risk-Sensitive and Robust Decision-Making: a CVaR Optimization Approach. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Sebastian Curi, Felix Berkenkamp, and Andreas Krause. Efficient Model-Based Reinforcement Learning through Optimistic Policy Search and Planning. In *Advances in Neural Information Processing Systems*, volume 33, pages 14156–14170. Curran Associates, Inc., 2020.

Will Dabney, Georg Ostrovski, David Silver, and Remi Munos. Implicit Quantile Networks for Distributional Reinforcement Learning. In *International Conference on Machine Learning*, volume 80, pages 1096–1105. PMLR, July 2018a.

Will Dabney, Mark Rowland, Marc G. Bellemare, and Rémi Munos. Distributional Reinforcement Learning with Quantile Regression. In *AAAI Conference on Artificial Intelligence*, pages 2892–2901. AAAI Press, 2018b.

Richard Dearden, Nir Friedman, and Stuart Russell. Bayesian Q-Learning. In *AAAI Conference on Artificial Intelligence*, pages 761–768, 1998. American Association for Artificial Intelligence.

Richard Dearden, Nir Friedman, and David Andre. Model Based Bayesian Exploration. In *Conference on Uncertainty in Artificial Intelligence*, pages 150–159, 1999. Morgan Kaufmann Publishers Inc.

Marc Peter Deisenroth and Carl Edward Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In *International Conference on Machine Learning*, pages 465–472, 2011.

Erick Delage and Shie Mannor. Percentile Optimization for Markov Decision Processes with Parameter Uncertainty. *Operations Research*, 58(1):203–213, February 2010.

Esther Derman, Daniel J Mankowitz, Timothy A Mann, and Shie Mannor. Soft-Robust Actor-Critic Policy-Gradient. In *Uncertainty in Artificial Intelligence Conference*, pages 208–218, 2018.

Esther Derman, Daniel Mankowitz, Timothy Mann, and Shie Mannor. A Bayesian Approach to Robust Reinforcement Learning. In *Uncertainty in Artificial Intelligence Conference*, volume 115, pages 648–658. PMLR, July 2020.

Omar Darwiche Domingues, Yannis Flet-Berliac, Edouard Leurent, Pierre Ménard, Xuedong Shang, and Michal Valko. rlberry - A Reinforcement Learning Library for Research and Education, October 2021.

Yaakov Engel, Shie Mannor, and Ron Meir. Bayes Meets Bellman: The Gaussian Process Approach to Temporal Difference Learning. In *International Conference on Machine Learning*, pages 154–161. AAAI Press, 2003.

Hannes Eriksson, Debabrota Basu, Mina Alibeigi, and Christos Dimitrakakis. SENTINEL: taming uncertainty with ensemble based distributional reinforcement learning. In *Conference on Uncertainty in Artificial Intelligence*, volume 180, pages 631–640. PMLR, August 2022.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing Function Approximation Error in Actor-Critic Methods. In *International Conference on Machine Learning*, volume 80, pages 1587–1596. PMLR, July 2018.

Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, and Aviv Tamar. Bayesian Reinforcement Learning: A Survey. *Foundations and Trends® in Machine Learning*, 8(5-6): 359–483, 2015.

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*, volume 80, pages 1861–1870. PMLR, July 2018.

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:1812.05905*, January 2019.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal Regret Bounds for Reinforcement Learning. *Journal of Machine Learning Research*, 11(4), 2010.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to Trust Your Model: Model-Based Policy Optimization. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Emilio Jorge, Hannes Eriksson, Christos Dimitrakakis, Debabrota Basu, and Divya Grover. Inferential Induction: A Novel Framework for Bayesian Reinforcement Learning. In *Proceedings on "I Can't Believe It's Not Better!" at NeurIPS Workshops*, volume 137, pages 43–52. PMLR, December 2020.

Tyler Kastner, Murat A Erdogdu, and Amir-massoud Farahmand. Distributional Model Equivalence for Risk-Sensitive Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 36, pages 56531–56552. Curran Associates, Inc., 2023.

Ramtin Keramati, Christoph Dann, Alex Tamkin, and Emma Brunskill. Being Optimistic to Be Conservative: Quickly Learning a CVaR Policy. In *AAAI Conference on Artificial Intelligence*, volume 34, pages 4436–4443, April 2020.

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*, volume 119, pages 5556–5566. PMLR, July 2020.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Carlos E. Luis, Alessandro G. Bottero, Julia Vinogradska, Felix Berkenkamp, and Jan Peters. Model-Based Uncertainty in Value Functions. In *International Conference on Artificial Intelligence and Statistics*, volume 206, pages 8029–8052. PMLR, April 2023.

Clare Lyle, Marc G. Bellemare, and Pablo Samuel Castro. A Comparative Analysis of Expected and Distributional Reinforcement Learning. In *AAAI Conference on Artificial Intelligence*, volume 33, pages 4504–4511, July 2019.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing Atari with Deep Reinforcement Learning. In *NIPS Deep Learning Workshop*, December 2013.

Ted Moskovitz, Jack Parker-Holder, Aldo Pacchiano, Michael Arbel, and Michael Jordan. Tactical Optimism and Pessimism for Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 12849–12863. Curran Associates, Inc., 2021.

Brendan O'Donoghue, Ian Osband, Remi Munos, and Volodymyr Mnih. The Uncertainty Bellman Equation and Exploration. In *International Conference on Machine Learning*, pages 3836–3845, 2018.

Ian Osband, Daniel Russo, and Benjamin Van Roy. (More) Efficient Reinforcement Learning via Posterior Sampling. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep Exploration via Bootstrapped DQN. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Ian Osband, John Aslanides, and Albin Cassirer. Randomized Prior Functions for Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Ian Osband, Benjamin Van Roy, Daniel J Russo, and Zheng Wen. Deep Exploration via Randomized Value Functions. *Journal of Machine Learning Research*, 20:1–62, 2019.

Luis Pineda, Brandon Amos, Amy Zhang, Nathan O. Lambert, and Roberto Calandra. MBRL-Lib: A Modular Library for Model-based Reinforcement Learning. *arXiv:2104.10159 [cs, eess]*, April 2021.

Antonin Raffin, Jens Kober, and Freek Stulp. Smooth Exploration for Robotic Reinforcement Learning, June 2021.

Mark Rowland, Marc Bellemare, Will Dabney, Remi Munos, and Yee Whye Teh. An Analysis of Categorical Distributional Reinforcement Learning. In *International Conference on Artificial Intelligence and Statistics*, volume 84, pages 29–37. PMLR, April 2018.

Mark Rowland, Robert Dadashi, Saurabh Kumar, Remi Munos, Marc G. Bellemare, and Will Dabney. Statistics and Samples in Distributional Reinforcement Learning. In *International Conference on Machine Learning*, volume 97, pages 5528–5536. PMLR, June 2019.

Mark Rowland, Rémi Munos, Mohammad Gheshlaghi Azar, Yunhao Tang, Georg Ostrovski, Anna Harutyunyan, Karl Tuyls, Marc G. Bellemare, and Will Dabney. An Analysis of Quantile Temporal-Difference Learning, January 2023.

Matthew J Sobel. The Variance of Discounted Markov Decision Processes. *Journal of Applied Probability*, 19(4):794–802, 1982.

Lauren N. Steimle, David L. Kaufman, and Brian T. Denton. Multi-model Markov decision processes. *IISE Transactions*, pages 1–16, May 2021.

Alexander L Strehl and Michael L Littman. An Analysis of Model-Based Interval Estimation for Markov Decision Processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

Xihong Su and Marek Petrik. Solving multi-model MDPs by coordinate ascent and dynamic programming. In *Conference on Uncertainty in Artificial Intelligence*, volume 216, pages 2016–2025. PMLR, August 2023.

Richard Sutton and Andrew Barto. *Reinforcement Learning: An Introduction*, volume 7. MIT Press, 2018.

Aviv Tamar, Dotan Di Castro, and Shie Mannor. Temporal Difference Methods for the Variance of the Reward To Go. In *International Conference on Machine Learning*, pages 495–503. PMLR, 2013.

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.

Cédric Villani. *Optimal Transport: Old and New*, volume 338. Springer, 2008.

Larry Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer Science & Business Media, 2013.

Peter R. Wurman, Samuel Barrett, Kenta Kawamoto, James MacGlashan, Kaushik Subramanian, Thomas J. Walsh, Roberto Capobianco, Alisa Devlic, Franziska Eckert, Florian Fuchs, Leilani Gilpin, Piyush Khandelwal, Varun Kompella, HaoChih Lin, Patrick MacAlpine, Declan Oller, Takuma Seno, Craig Sherstan, Michael D. Thomure, Houmehr Aghabozorgi, Leon Barrett, Rory Douglas, Dion Whitehead, Peter Dürr, Peter Stone, Michael Spranger, and Hiroaki Kitano. Outracing champion Gran Turismo drivers with deep reinforcement learning. *Nature*, 602(7896):223–228, February 2022.

Derek Yang, Li Zhao, Zichuan Lin, Tao Qin, Jiang Bian, and Tie-Yan Liu. Fully Parameterized Quantile Function for Distributional Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Bo Zhou, Hongsheng Zeng, Fan Wang, Yunxiang Li, and Hao Tian. Efficient and Robust Reinforcement Learning with Uncertainty-based Value Expansion. *arXiv:1912.05328 [cs]*, December 2019.

Qi Zhou, HouQiang Li, and Jie Wang. Deep Model-Based Reinforcement Learning via Estimated Uncertainty and Conservative Policy Optimization. In *AAAI Conference on Artificial Intelligence*, volume 34, pages 6941–6948, April 2020.