Epistemic Bellman Operators

Pascal R. van der Vaart
Delft University of Technology
2628 CD Delft, The Netherlands
p.r.vandervaart-1@tudelft.nl

Matthijs T. J. Spaan
Delft University of Technology
2628 CD Delft, The Netherlands
m.t.j.spaan@tudelft.nl

Neil Yorke-Smith
Delft University of Technology
2628 CD Delft, The Netherlands
n.yorke-smith@tudelft.nl

Abstract

Uncertainty quantification remains a difficult challenge in reinforcement learning. Several algorithms exist that successfully quantify uncertainty in a practical setting, however it is unclear whether these algorithms are theoretically sound and can be expected to converge. Furthermore, they seem to treat the uncertainty in the target parameters in different ways. In this work, we unify several practical algorithms into one theoretical framework by defining a new Bellman operator on distributions, and show that this Bellman operator is a contraction. Further, building on our theory, we modify PPO, a popular modern model-free algorithm, into an uncertainty-aware variant to showcase the general applicability of our main result.

1 Introduction

Reinforcement learning (RL) algorithms have surpassed humans' ability in many games [1, 2], and have now also found success in real world problems such as controlling plasma in a nuclear fusion reactor [3], video compression [4], large language models [5] and algorithm design [6, 7]. However, even for relatively simple tasks, algorithms still require many simulations or real interactions to learn a strong policy, making them inefficient. One approach to attack this problem is by making algorithms aware of their epistemic uncertainty, which is uncertainty caused by a lack of data. This allows them to explore only parts of the problem that are still uncertain, decreasing the total amount of interactions required.

However, proper uncertainty quantification is still an open problem in reinforcement learning. Many techniques from supervised learning, such as ensembles [8, 9] and Bayesian methods [10, 11, 12, 13], have found success in practice when applied to supervised learning tasks with labelled data. However, in reinforcement learning data is not labelled with a ground truth, and instead the label for the current state is a self-supervised bootstrap from the label of the next state, known as the target value. Uncertainty quantification in RL must consider this sequential nature. At the heart of this problem is the fact that uncertainty in the current state should include the uncertainty in the target values, which is the uncertainty in the future states.

Adaptations of uncertainty quantification methods from supervised learning have been applied to reinforcement learning settings [14, 15, 16, 17, 18, 19, 20, 21] with good practical results, but there is no guarantee that the way these algorithms treat the uncertainty in the successor state leads to a theoretically sound algorithm, in the sense that the uncertainty quantification aspect can be expected to converge to a solution at all. At least guaranteeing that these methods work in potentially simplified scenarios is essential for the adoption of uncertainty quantification in algorithms in the real world.

17th European Workshop on Reinforcement Learning (EWRL 2024).

Furthermore, some algorithms seemingly disagree in their decisions on how to treat the uncertainty in the target values.

When adapting Deep Q-learning (DQN)-style algorithms to uncertainty aware algorithms like Bootstrapped DQN [14], EVE [20], Langevin-DQN [19], LMCDQN [22], SMC-DQN [21] and BDQN [17], there is a decision to be made about how to use and update the target parameters. Generally, these algorithms condition their posterior on a posterior of the target parameters. As a main problem, we highlight that there is no guarantee that the process of repeatedly updating the current distribution, conditioned on the distribution over target parameters, and copying it to the target parameters will converge to a limiting distribution.

Recently, Fellows et al. [23] studied this problem theoretically and contended that Bayesian model-free RL algorithms create a posterior over Bellman operators. They showed that the posterior converges to the true Bellman operator in the limit of infinite data. We instead take an arguably more natural and direct approach, and show that the problem can be formulated as a generic Bellman operator that works on distributions, and converges to a consistent posterior-like distribution even for finite data. This result ensures that it is sensible to talk about a posterior distribution conditioned on a finite data set, and specifies how the distribution over target parameters should be used to develop theoretically sound uncertainty-aware Q- learning algorithms.

Summarising, our contributions over the state-of-the-art are as follows:

- We introduce Epistemic Bellman operators, formalizing the process of conditioning the likelihood on a distribution over target parameters.
- We prove that Epistemic Bellman operators are contractions, proving that the process of interleaving posterior inference and target updates converges to a consistent fixed point for a general class of distributions. Furthermore, we show that the mean of the fixed point is equal to the fixed point of a non-epistemic Bellman operator in the case of policy evaluation.
- We showcase an application of Epistemic Bellman operators to the return estimator of Proximal Policy Optimization (PPO), a popular model-free algorithm, modifying it into an epistemic version that clips less aggressively whenever the agent is certain about its advantages. We test this agent on a variety of tasks and show increased performance across several environments.

2 Background

2.1 Markov Decision Processes

We focus on Markov Decision Processes (MDP) with infinite horizon in the discounted reward setting. Formally, a Markov Decision Process is a tuple $(\mathcal{S}, \mathcal{A}, T, R, \gamma)$ of a state space $\mathcal{S}$, action space $\mathcal{A}$, transition function $T : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and discount factor $0 \leq \gamma < 1$. At each time step t, an agent observes the current state s_t, chooses an action $a_t \sim \pi(s_t)$ according to its policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, and receives reward $r_t = R(s_t, a_t)$. The goal of reinforcement learning is to find a policy π that maximizes the discounted cumulative reward $\mathbb{E}_{T,\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$. Of central importance is the Q-function

$$Q^\pi(s,a) = R(s,a) + \mathbb{E}_{T,\pi}\left[\sum_{t=1}^{\infty} \gamma^t r_t \mid s_0 = s, a_0 = a\right],$$

denoting the expected discounted future reward if the agent executes action a in state s and then follows the policy π.

In a tabular setting, we represent the reward function, transition function and policy as vectors and matrices $R \in \mathbb{R}^{|S||A|}$, $T \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|\times|\mathcal{S}|}$, $\pi \in \mathcal{R}^{|S||A|}$. The Bellman operator for a policy π can then be written as

$$B^\pi_{T,R} Q = R + \gamma T^\pi Q,$$

where $T^\pi \in \mathbb{R}^{|S||A|\times|S||A|}$ is the transition function from state-action to state-action induced by the transition function T and the policy π, defined by

$$\begin{aligned}(T^\pi)_{sas'a'} &= \mathbb{P}(s_{t+1}, a_{t+1} = s', a' \mid s_t, a_t = s, a) \\ &= T_{sas'}\pi_{s'a'}.\end{aligned} \tag{1}$$

Since the transition function T and reward function R are assumed to be unknown to the agent, computing a strong policy requires exploration of the environment to learn which actions result in optimal return.

2.2 Model-free Reinforcement Learning

Typically interesting problems have large states and action spaces, making it difficult to learn the transition and reward functions. Model-free algorithms such as actor-critics [24, 25, 26] and Q-learning [1] bypass this step and instead aim to learn a good policy or the values of a good policy directly, without estimating T and R.

A common component in these algorithms is to learn the values or Q-values by representing them by a neural network and minimizing the squared temporal difference loss on a data set of transitions $\mathcal{D}$:

$$L_{TD}(\theta, \theta', \mathcal{D}) = \sum_{(s,a,r,s') \in \mathcal{D}} TD(\theta, \theta', (s,a,r,s'))^2 = \sum_{(s,a,r,s') \in \mathcal{D}} \left[Q_\theta(s,a) - r - \gamma G(\theta', s')\right]^2,$$

where $G(\theta', s')$ is some return estimator usually depending on a bootstrap from a target network θ' [1]. Examples are $G(\theta', a') = \max_{a'} Q_{\theta'}(s', a')$ in the case of one step Q-learning, or $G(\theta', a') = \sum_{a' \in A} \pi(a'|s') Q_{\theta'}(s', a')$ in the case of policy evaluation in actor-critics.

Agents use empirically observed transitions (s, a, r, s') to learn these models, requiring exploration to sufficiently cover the environment to achieve accurate values. Quantifying uncertainty in the value models can greatly improve the exploration capability of reinforcement learning algorithms through Thompson Sampling [14, 15, 27, 16, 20, 17, 19] or exploration bonuses [28, 29, 18]. Furthermore, uncertainty quantification can also aid in general stability of algorithms by reweighting Bellman errors [30].

2.3 Bayesian Value Learning

One method to quantify uncertainty is through Bayesian algorithms. Generally, a Bayesian neural network is any neural network parameterized by $\theta \in \Theta$ where one attempts to model the posterior distribution

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d(\theta)},$$

where $p(\mathcal{D}|\theta)$ is the likelihood, $p(\theta)$ is a prior and $\mathcal{D}$ is some data set. The posterior density $p(\theta|\mathcal{D})$ signifies how likely values of θ are, and is a natural method to model uncertainty as a distribution.

To equip an agent with uncertainty quantification, a posterior distribution over the parameters of a Q-function can be constructed $p(\theta|\mathcal{D}, \theta') \propto p(\mathcal{D}|\theta, \theta')p(\theta)$. Since the squared error loss is proportional to the log-density of a normal distribution, defining

$$p(\mathcal{D}|\theta, \theta') = \exp\left(-\sum_{(s,a,r,s') \in \mathcal{D}} \left[Q_\theta(s,a) - r - \gamma G(\theta', s')\right]^2\right) \tag{2}$$

is a natural candidate for the likelihood when extending value learning algorithms to a Bayesian paradigm. This corresponds to the assumption that the temporal difference errors are normally distributed:

$$TD(\theta, \theta', (s,a,r,s')) \sim \mathcal{N}(0, \sigma). \tag{3}$$

While this assumption is in general not correct for every MDP, it is a convenient design choice and it should come as no surprise that several previous works have used this likelihood before [15, 20, 19, 17, 22].

The likelihood $p(\mathcal{D}|\theta, \theta')$ and therefore also the posterior density $p(\theta|\mathcal{D}, \theta')$ does not only depend on the data, i.e. the observed transitions, it is also conditioned on the target values θ'. Handling this dependency is crucial for a theoretically sound algorithm that handles the sequential nature of uncertainty in this setting. Furthermore, posterior distributions are generally difficult to compute in practice, requiring approximate models. For example, BootDQN [14, 15] uses ensembles, Langevin-DQN, LMCDQN and SMC-DQN [19, 22, 21] use Monte Carlo methods, EVE [20] uses a Laplace approximation and BDQN [17] performs inference over only the final layer of the Q-network.

3 Problem Statement

In this section, we identify a key problem with model-free Bayesian Reinforcement Learning algorithms and motivate the value of our main contribution.

3.1 Problems with Target Updates

Roughly speaking, algorithms such as BootDQN, Langevin-DQN, LMCDQN, SMC-DQN, BDQN and EVE operate by interleaving steps

1. Infer a posterior given the current targets, $p_{\texttt{main}}(\theta|\mathcal{D}) = p(\theta|\mathcal{D}', \theta')$, where the targets are drawn or assumed to be from some distribution over targets $p_{\texttt{target}}(\theta')$.
2. Update the distribution over targets: $p_{\texttt{target}}(\theta) \leftarrow p_{\texttt{main}}(\theta|\mathcal{D}) = p(\theta|\mathcal{D}', \theta')$ to the current distribution over the main parameters θ.

This is analogous to the target update in many non-probabilistic algorithms that use temporal difference learning, and may seem like a reasonable adaptation to the Bayesian setting. However, for distributions there is no guarantee that this scheme converges, or is in fact well defined, since setting $p_{\texttt{target}}(\theta) \leftarrow p_{\texttt{main}}(\theta)$ is mathematically unsupported when $p_{\texttt{main}}(\theta)$ is a distribution that was conditioned on the target parameters. Furthermore, if this scheme does not converge to the same $p_{\texttt{main}}(\theta|\mathcal{D})$ for a fixed data set $\mathcal{D}$ and every starting distribution, it is not sensible to define a posterior $p_{\texttt{main}}(\theta|\mathcal{D})$ that is only conditioned on the data.

Fellows et al. [23] propose interpreting the problem as inferring a posterior distribution over Bellman operators, and show convergence of the posterior to the true Bellman operator as more data is collected. Instead, we propose a new Bellman operator that operates on posterior-like distributions, and prove that this operator is a contraction and has a fixed point. Roughly speaking, we show that an algorithm that alternates between updating a distribution conditioned on the targets, and updating the distribution over targets converges to a limiting distribution, proving that several common Bayesian algorithms which are special cases of our operator can be expected to converge, independent of the starting the starting distribution.

3.2 What Do These Distributions Look Like?

Before we introduce Epistemic Bellman Operators, we analyze which distributions Bayesian Q-learning algorithms actually attempt to approximate. To this end, we study BootDQN and EVE in a tabular setting, and assume there exists some idealized distribution over targets $Q' \sim p_{\texttt{target}}(Q)$ that our agent currently has. Furthermore, as in BootDQN and EVE, we are equipped with a likelihood

$$p(\mathcal{D}|Q, Q') \propto \exp\left(-\frac{1}{2\sigma^2}\sum_{s,a,r,s'\in\mathcal{D}} \text{TD}(Q, Q', s, a, r, s')^2\right),$$

also conditioned on a set of target values Q'. This results in a posterior distribution

$$p(Q|\mathcal{D}, Q') \propto p(\mathcal{D}|Q, Q')p(Q).$$

However, this distribution is conditioned on a single value for the targets and does not yet incorporate the fact that $Q' \sim p_{\texttt{target}}(Q)$, i.e. the uncertainty over the targets.

In the case of BootDQN, $p_{\texttt{target}}(Q)$ is modelled by the ensemble of target networks $\theta'_1, \ldots, \theta'_n$, and to approximate the posterior each ensemble member optimizes for its own loss $Q_i^* = \arg\max p(Q_i|\mathcal{D}, Q'_i)$. On the other hand, EVE has a Laplace approximation for $p_{\texttt{target}}(Q)$, and updates the main distribution by sampling one $\tilde{Q}' \sim p_{\texttt{target}}(Q)$, maximizing $Q = \arg\max p(Q|\mathcal{D}, \tilde{Q}')$ and also updating the Fisher information. .

In our idealized setting, we can directly consider the marginalization of the conditioned posterior over the targets Q':

$$p_{\texttt{main}}(Q|\mathcal{D}) = \int p(Q|\mathcal{D}, q')dp_{\texttt{target}}(q').$$

Figure 1 shows a graphical presentation of this marginalization, together with BootDQN and EVE, in a simplified setting with an MDP with one state and one action. The top row is the idealized version

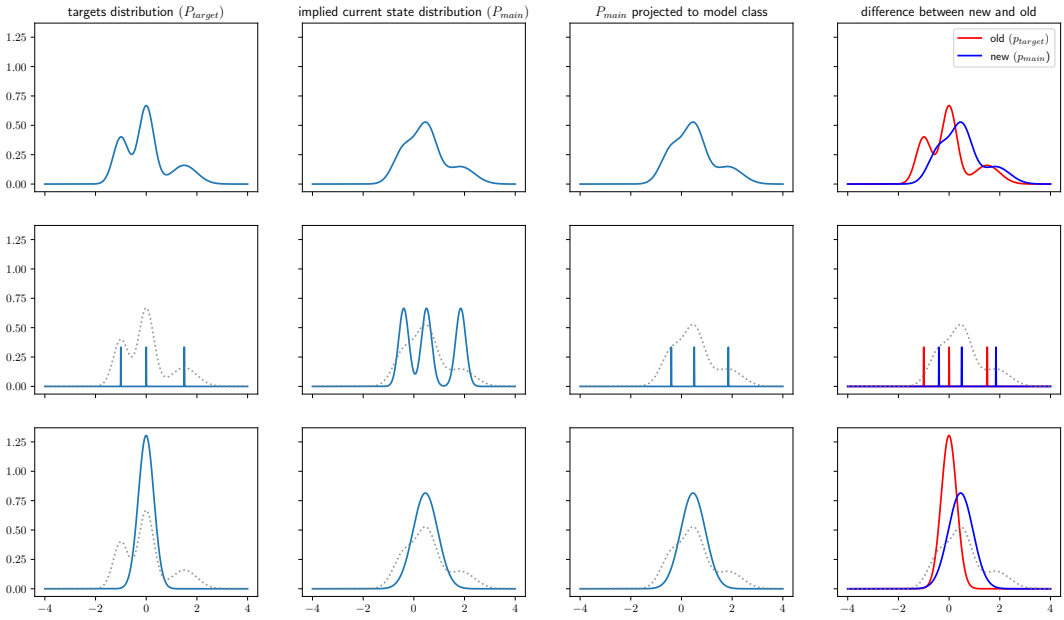

Figure 1: Plots of the distribution over the Q-value of a single state-action. The final column shows the difference between the target distribution (red) and the current distribution (blue). Rows are (1) idealized model class, (2) ensemble approximation (BootDQN), (3) Laplace approximation (EVE).

of Bayesian model-free reinforcement learning algorithms. A distribution over the targets defines a distribution over the main values, which can exactly be inferred by a fully expressive model class. The second row contains a sketch of the situation with ensembles. The distribution $p_{\texttt{target}}$ is an ensemble, which together with the normal distribution likelihoods makes a mixture distribution for the main values. Estimating this distribution with an ensemble ideally returns an ensemble containing the modes of the new distribution.

For EVE, the target distribution is a normal distribution. The distribution for the current state is therefore also a normal distribution, and representing it in the model class of normal distributions returns a normal distribution.

Both BootDQN and EVE can be considered as approximations to this marginalization, approximating the integral with an ensemble in the case of BootDQN and a single sample from $p_{\texttt{target}}(q')$ in the case of EVE. After constructing an approximate $\tilde{p}_{\texttt{main}}(Q|\mathcal{D})$ each method then attempts to represent this distribution in their model class.

Considering this marginalization process, we can now define what it means for a well-defined posterior to exist. If the process of

$$p^{(k)}_{\texttt{main}}(Q|\mathcal{D}) = \int p(Q|\mathcal{D}, q')dp^{(k)}_{\texttt{target}}(q') \tag{4}$$

$$p^{(k+1)}_{\texttt{target}}(q'|\mathcal{D}) = p^{(k)}_{\texttt{main}}(Q|\mathcal{D}) \tag{5}$$

$$k = k+1 \tag{6}$$

converges to the same limiting distribution $p(Q|\mathcal{D})^*$ for every starting $p^{(0)}_{\texttt{target}}(q')$, the posterior-like distribution $p(Q|\mathcal{D})$ is well-defined. We formalize this process with the Epistemic Bellman Operator.

4 Epistemic Bellman Operators

For any Bellman operator or contraction $B_{\mathcal{D}}$, perhaps depending on some data set $\mathcal{D}$, we can define a pushforward distribution with additive noise as

$$p(Q|\mathcal{D}, Q') = \text{Law}\left(B_{\mathcal{D}}(Q') + \epsilon_{\mathcal{D}}\right), \tag{7}$$

where Law (X) denotes the probability density of X. This is equivalent to the notion that the Q-values are distributed around the target values Q' with some *local* uncertainty $\epsilon_{\mathcal{D}}$, independent of Q'. This is a naturally occurring distribution in literature, since the posterior distribution of a normal likelihood with a normal prior takes this shape, which is commonly used in model-free deep RL literature [14, 15, 20, 16, 17, 19, 22]. The Epistemic Bellman Operator for this distribution marginalizes the distribution over Q', and returns a new distribution.

Definition 4.1 (EBO)**.** For any measurable set A, let $\mathcal{P}(A)$ denote the set of probability distributions over A. Let $p(q|q')$ be a distribution over Q-values conditioned on target Q-values, e.g. Equation 7. We define the corresponding Epistemic Bellman Operator (EBO), as an operator $\mathcal{B}_p : \mathcal{P}(\mathbb{R}^{|S||A|}) \to \mathcal{P}(\mathbb{R}^{|S||A|})$, mapping distributions over Q-values to another distribution over Q-values by

$$\mathcal{B}_p P_Q(q) = \int p(q|q') dP_Q(q'). \tag{8}$$

When $p(q|q')$ is of the form Law $(B_{\mathcal{D}}(q') + \epsilon_{\mathcal{D}})$, we can equivalently write Equation 8 as

$$\mathcal{B}_p P_Q = \text{Law}\left(B_{\mathcal{D}}(Q) + \epsilon_{\mathcal{D}}, Q \sim P_Q\right). \tag{9}$$

If the distribution $p(q|q') = \text{Law}\left(B_{\mathcal{D}}(Q) + \epsilon_{\mathcal{D}}\right)$ has contracting properties, for example when $B_{\mathcal{D}}$ is a Bellman operator, it can be shown that the respective EBO is also a contraction. This is formalized in Theorem 4.2, whose proof is provided in Appendix A.

Theorem 4.2 (Contraction)**.** *Let $\mathcal{Q} = (\mathbb{R}^{|S||A|}, \|.\|_\infty)$ be a metric space, $B_{\mathcal{D}}$ be a contraction on $\mathcal{Q}$, and let $p_B(q|q') = Law\left(B_{\mathcal{D}}(q') + \epsilon_{\mathcal{D}}\right)$ be a distribution over $\mathcal{Q}$ conditioned on target values in $\mathcal{Q}$.*

Then the corresponding Epistemic Bellman Operator $\mathcal{B}_p : \mathcal{P}(\mathcal{Q}) \to \mathcal{P}(\mathcal{Q})$ defined by Equation 9, where ϵ_D is independent of Q, is a W_p-contraction on $\mathcal{P}(\mathcal{Q})$ for any $p \in [1, \infty)$.

This theorem states that even with finite data, repeatedly applying an EBO to any starting distribution will converge to a fixed point. A consequence is that algorithms which interleave posterior inference with target distribution updates are theoretically sound in the sense that they converge to a solution.

In the case of policy evaluation with a one step Bellman Operator

$$BQ = R + \gamma T^\pi Q,$$

the fixed point of the EBO $\mathcal{B}$ is easy to characterize. This can be extended to any affine B. Notably, the following theorem states that the mean of the fixed point is equal to the fixed point of the non-epistemic Bellman operator in $p(q|q')$ when it is affine. We refer to Appendix A for the proof.

Theorem 4.3 (Mean of $\mathcal{B}$)**.** *Let $\mathcal{B}$ be the EBO corresponding to $p_B(q|q') = Law\left(B(q') + \epsilon\right)$. Let $P_B(Q)$ be the fixed point of $\mathcal{B}$, and Q_B be the fixed point of B. If B is an affine contraction, then $\mathbb{E}_{P_B}[Q] = Q_B$. Furthermore, writing $B(Q) = AQ + b$, the covariance $\Sigma_Q = \mathbb{E}_{P_B}\left[QQ^\top - Q_B Q_B^\top\right]$ is given by*

$$Vec(\Sigma_Q) = (I - A \otimes A)^{-1} Vec(\Sigma_\epsilon)$$

where Vec(X) denotes the vectorization of X and $\otimes$ is the Kronecker product.

To showcase what our theorems state, we conduct an experiment in an MDP with one state and two actions so that the distributions are easy to visualize. We initialize a multivariate normal distribution, and iteratively apply the EBO. Figure 2 displays the density of the distribution over time, with the fixed point of the non-epistemic Bellman equation Q^π marked in red. It can be seen that the distributions converge to a normal distribution centered around Q^π, where the Q-values are strongly correlated. This correlation is expected, since both actions transition to the same state. Furthermore, Figure 3 shows the Wasserstein distance between the distribution at each iteration and the fixed point. It very closely matches the theoretical contraction rate of γ.

5 Epistemic Clipping PPO

Since Theorem 4.2 holds for any contraction, it is applicable to a wide range of return estimators used in practice. To showcase the generality of our results, we modify Proximal Policy Optimization

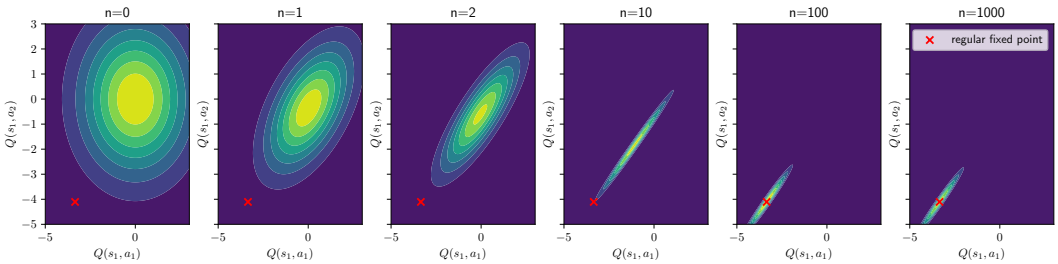

Figure 2: The Epistemic Bellman Operator applied iteratively to an initial distribution with a fixed policy in a single state, two actions MDP. The fixed point of the regular Bellman operator is in red.

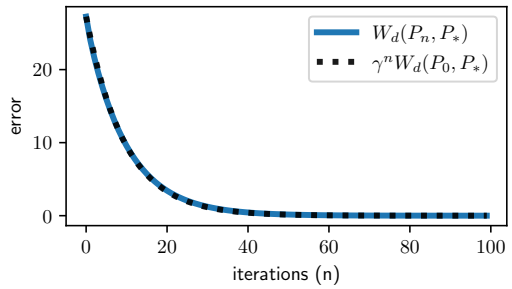

Figure 3: The Wasserstein Distance between distributions over Q-values and the fixed point of the EBO when iteratively applying the EBO (blue). The rate of contraction matches the predicted contraction rate γ (black, dashed)

(PPO) [25] into Epistemic Clipping PPO (ECPPO) by replacing the value models with an ensemble. PPO estimates the advantages of its policy with the following return estimator:

$$A_t = \delta_t + \gamma\lambda\delta_{t+1} + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1} \tag{10}$$

$$\text{where } \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t). \tag{11}$$

An approximation of the posterior over $V(s)$ provides the agent with uncertainty quantification on the advantages, which we use to clip less aggressively in the policy loss whenever we are certain about the advantages. To this end, the typical policy loss in PPO

$$L^{PPO}(\theta) = \mathbb{E}[\min(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t)], \tag{12}$$

is modified to

$$L^{ECPPO}(\theta) = \mathbb{E}[\min(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1-\epsilon\phi(U_t), 1+\epsilon\phi(U_t))A_t)],$$

where U_t is an estimate of the uncertainty in A_t and ϕ is a monotonically decreasing function, so that the clipping range expands whenever U_t is low.

In our implementation of ECPPO, the value network of PPO is replaced by an ensemble $V_1(s), \ldots, V_n(s)$, and the advantages are computed according to each ensemble member k independently:

$$A_t^{(k)} = \delta_t^{(k)} + \gamma\lambda\delta_{t+1}^{(k)} + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1}^{(k)}, \tag{13}$$

$$\text{where } \delta_t^{(k)} = r_t + \gamma V_k(s_{t+1}) - V_k(s_t). \tag{14}$$

As in standard PPO, the advantages are then normalized $\tilde{A}_t^{(k)} = \frac{A_t^{(k)} - \mu}{\sigma}$ using statistics μ, σ estimated from the minibatch, and the uncertainty is defined as $U_t = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(\tilde{A}_t^{(k)})^2 - (\frac{1}{n}\sum_{k=1}^{n}\tilde{A}_t^{(k)})^2}$, which is the empirical standard deviation of the ensemble. The clipping range is modified by a function $\phi(U_t)$ such that $0.5 \leq \phi(U_t) \leq 2$. For exact specifications we refer to Appendix B.2.

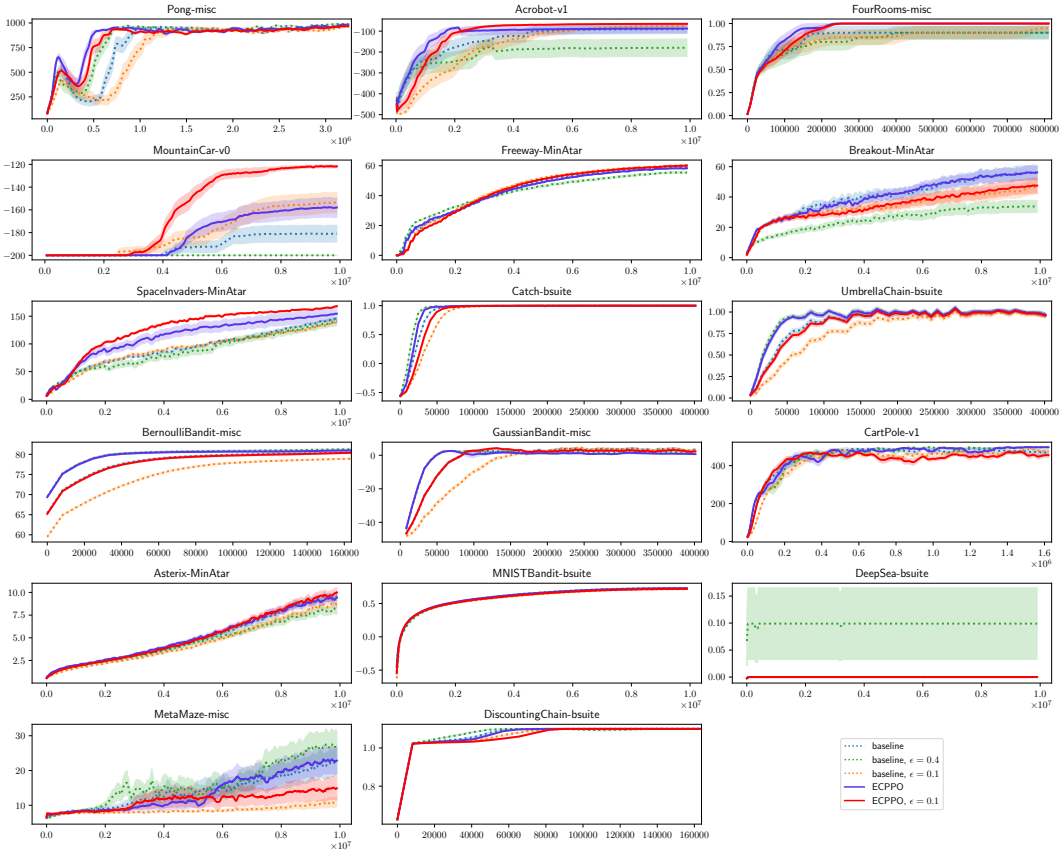

Figure 4: Mean learning curves of ECPPO and baseline PPO algorithms on 17 environments from Gymnax. Shaded areas denote the standard error of the mean, based on 20 seeds per agent per environment.

6 Empirical Study

Having introduced ECPPO as an example of how Epistemic Bellman Operators can be integrated into an algorithm such as PPO, we now study the benefits experimentally. We test the RL agent with base clipping ratio parameters $\epsilon = 0.2$ and $\epsilon = 0.1$ on all discrete state environments in Gymnax [31], which includes environments from OpenAI Gym [32], BSuite [33], MinAtar [34], and several miscellaneous environments [35, 36, 37, 38], but excluding SimpleBandit-bsuite, which is non-sequential and trivial, and MemoryChain-bsuite, which requires memory.

We compare results against the baseline version of PPO in PureJaxRL [39], which is tuned for these environments by the authors. Furthermore, since ECPPO is a modification to the clipping behaviour, we also compare against modified baselines with different clipping parameters ϵ. The clipping parameter of the additional baselines are set to $\epsilon = 0.1$ and $\epsilon = 0.4$, which is the minimum and maximum of the range in which ECPPO can adapt its clipping ratio parameter.

Figure 4 shows the learning curves on each environment. It can be seen that in Pong, Acrobot, and FourRooms, ECPPO outperforms all baselines. Furthermore, on MountainCar, Freeway, Breakout and Space Invaders, lower ϵ is optimal in the baselines, and ECPPO matches the best baseline. In UmbrellaChain, BernoulliBandit, and Gaussianbandit and Cartpole, higher ϵ lets PPO learn faster, and ECPPO again closely matches the best baseline. In the rest of the environments the relationship between ϵ and the performance of the baselines is not immediately clear, but ECPPO still matches or outperforms the strongest baseline, Except for DiscountingChain, where ECPPO is slightly behind the baseline with the respective ϵ. We note that none of our agents solve DeepSea reliably.

To further highlight how ECPPO improves over the baseline PPO with fixed ϵ, Figure 5 shows the regret of ECPPO with $\epsilon = 0.2$ relative to the regret of the baseline in 0.2. It is immediately visible that

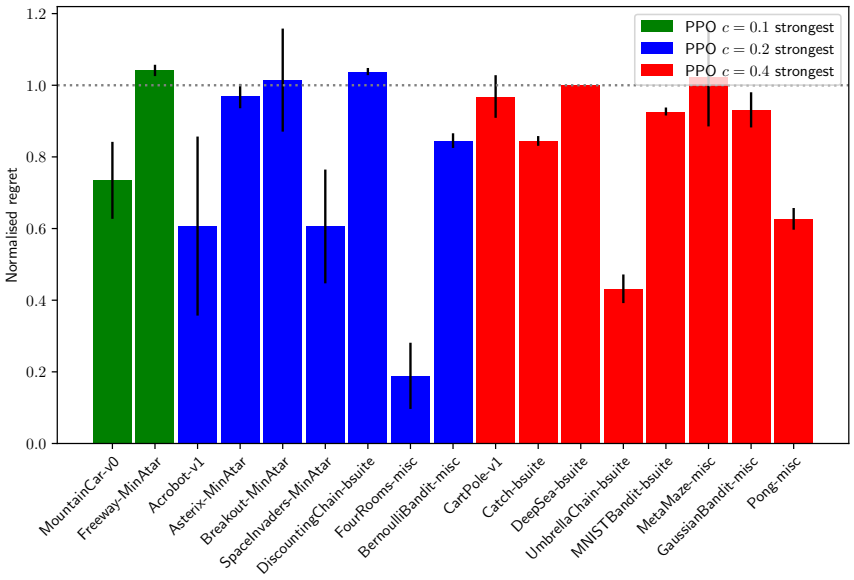

Figure 5: Regret of ECPPO with $\epsilon = 0.2$ relative to the regret of the baseline with $\epsilon = 0.2$ (lower is better). Environments are grouped by whether more or less clipping is beneficial to the baseline.

ECPPO improves performance across many environments, independent of whether high or low ϵ is optimal in the specific environment. Finally, we observe in Figure 6 that the uncertainty quantification make sense in a qualitative manner in the FourRooms environment, where uncertainty is high where the current policy has low support.

7 Related work

There is a large body of research for Bayesian methods in RL. On the practical side, there are algorithms such as BootDQN [14, 15], EVE [20], BDQN [17], Langevin-DQN [19], LMCDQN [22] and SMC-DQN [21]. Our main theoretical result aims to theoretically ground these methods within a general framework by interpreting them as special cases of an EBO, which works on distributions, and prove that this is a contraction.

Operators that work on distributions are also a main focus in Distributional RL [40]. The goal in distributional RL is to model the distribution of returns, as opposed to learning only the mean. Distributional methods model the aleatoric uncertainty, which is the inherent randomness of returns due to the randomness in the policy and MDP. Instead, we focus on learning the mean of the returns, and compute a distribution over possible means given our observations to model the epistemic uncertainty on the mean. Furthermore, our operator naturally takes into account the dependency and covariance of the Q-values.

Operators that propagate epistemic uncertainty as a scalar have previously been studied [27, 41], whereas we model the epistemic uncertainty as a full distribution. Dearden [42] discusses a similar operator on distributions, also providing convergence guarantees with a contraction argument. This result can be interpreted as a special case of our results with a specific return estimator and specific approximation class. Our main theorem instead applies to any return estimator with contractive properties.

Furthermore, Luis et al. [43] study Bayesian value models in a model-based setting, also introducing an operator on value functions with a distributional fixed point representing the epistemic uncertainty. We focus on the model-free setting, losing accuracy in the posterior but more closely resembling current Bayesian model-free algorithms.

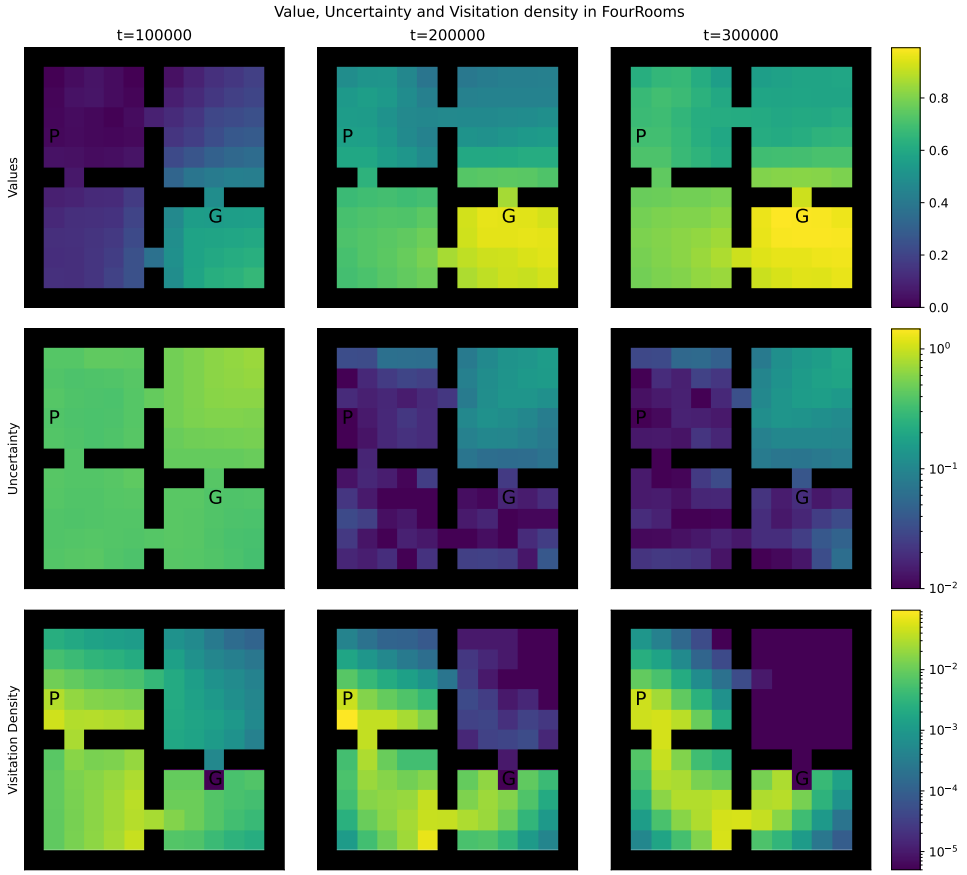

Figure 6: Value, uncertainty and state visitation density on FourRooms of ECPPO at several points during training. The starting position and goal position are denoted by P and G respectively.

8 Conclusion

We have introduced Epistemic Bellman Operators, which are operators that map a distribution over Q-values to the pushforward of regular Bellman operators with additive noise. We have shown that our operator generalizes several probabilistic reinforcement learning algorithms, unifying practical algorithms that appear to have dissimilar architectures. Furthermore, we have proven that Epistemic Bellman Operators are contractions, which implies that interleaving posterior inference and target updates converges to a fixed distribution and motivates these practical algorithms by showing consistency in tabular settings. We showed that the fixed point of an EBO is sensible when doing policy evaluation. Finally, we showcased the generality of our operators by modifying PPO into an uncertainty-aware variant that outperforms the original PPO in several environments.

In the future, we aim to design new uncertainty-aware algorithms using the insights from our main theorem to guide practical design choices regarding the treatment of target parameters. Furthermore, we aim to investigate the influence of priors and likelihoods and study more suitable distributions than normal distributions for model-free reinforcement learning.

Acknowledgments and Disclosure of Funding

We thank the reviewers for their helpful comments. This work has received funding from the European Union's Horizon 2020 research and innovation programme, under grant agreements 964505 (E-pi) and 952215 (TAILOR).

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015.

[2] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[3] Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de Las Casas, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022.

[4] Amol Mandhane, Anton Zhernov, Maribeth Rauh, Chenjie Gu, Miaosen Wang, Flora Xue, Wendy Shang, Derek Pang, Rene Claus, Ching-Han Chiang, Cheng Chen, Jingning Han, Angie Chen, Daniel Jaymin Mankowitz, Jackson Broshear, Julian Schrittwieser, Thomas Hubert, Oriol Vinyals, and Timothy A. Mann. MuZero with self-competition for rate control in VP9 video compression, 2022.

[5] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in neural information processing systems*, volume 35, pages 27730–27744, 2022.

[6] Alhussein Fawzi, Matej Balog, Aja Huang, Thomas Hubert, Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Francisco J. R. Ruiz, Julian Schrittwieser, Grzegorz Swirszcz, David Silver, Demis Hassabis, and Pushmeet Kohli. Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610(7930):47–53, 2022.

[7] Daniel J. Mankowitz, Andrea Michi, Anton Zhernov, Marco Gelmi, Marco Selvi, Cosmin Paduraru, Edouard Leurent, Shariq Iqbal, Jean-Baptiste Lespiau, Alex Ahern, Thomas Koppe, Kevin Millikin, Stephen Gaffney, Sophie Elster, Jackson Broshear, Chris Gamble, Kieran Milan, Robert Tung, Minjae Hwang, Taylan Cemgil, Mohammadamin Barekatain, Yujia Li, Amol Mandhane, Thomas Hubert, Julian Schrittwieser, Demis Hassabis, Pushmeet Kohli, Martin Riedmiller, Oriol Vinyals, and David Silver. Faster sorting algorithms discovered using deep reinforcement learning. *Nature*, 618(7964):257–263, 2023.

[8] Thomas G. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems*, pages 1–15. Springer Berlin Heidelberg, 2000.

[9] 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, 2017.

[10] Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International Conference on Machine Learning*, 2014.

[11] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[12] Francesco D'Angelo and Vincent Fortuin. Repulsive deep ensembles are bayesian. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[13] Florian Wenzel, Kevin Roth, Bastiaan Veeling, Jakub Swiatkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. How good is the bayes posterior in deep neural networks really? In *International Conference on Machine Learning*, 2020.

[14] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[15] Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[16] Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, et al. Noisy networks for exploration, 2017.

[17] Kamyar Azizzadenesheli, Emma Brunskill, and Animashree Anandkumar. Efficient exploration through bayesian deep q-networks. In *2018 Information Theory and Applications Workshop (ITA)*. IEEE, 2018.

[18] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2018.

[19] Vikranth Dwaracherla and Benjamin Van Roy. Langevin DQN, 2021.

[20] Simon Schmitt, John Shawe-Taylor, and Hado van Hasselt. Exploration via epistemic value estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 2023.

[21] Pascal Van der Vaart, Neil Yorke-Smith, and Matthijs Spaan. Bayesian ensembles for exploration in deep reinforcement learning. In *Proceedings of the 2024 International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '24, Richland, SC, 2024. International Foundation for Autonomous Agents and Multiagent Systems. To appear.

[22] Haque Ishfaq, Qingfeng Lan, Pan Xu, A. Rupam Mahmood, Doina Precup, Anima Anandkumar, and Kamyar Azizzadenesheli. Provable and practical: Efficient exploration in reinforcement learning via langevin monte carlo, 2023.

[23] Matthew Fellows, Kristian Hartikainen, and Shimon Whiteson. Bayesian bellman operators. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[24] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

[25] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[26] 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*, pages 1861–1870. PMLR, 2018.

[27] 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.

[28] Georg Ostrovski, Marc G Bellemare, Aäron Oord, and Rémi Munos. Count-based exploration with neural density models. In *International Conference on Machine Learning*, 2017.

[29] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

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

[31] Robert Tjarko Lange. gymnax: A JAX-based reinforcement learning environment library, 2022. `http://github.com/RobertTLange/gymnax`.

[32] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI gym, 2016.

[33] Ian Osband, Yotam Doron, Matteo Hessel, John Aslanides, Eren Sezener, Andre Saraiva, Katrina McKinney, Tor Lattimore, Csaba Szepesvari, Satinder Singh, Benjamin Van Roy, Richard Sutton, David Silver, and Hado Van Hasselt. Behaviour suite for reinforcement learning. In *International Conference on Learning Representations*, 2020.

[34] Kenny Young and Tian Tian. Minatar: An atari-inspired testbed for thorough and reproducible reinforcement learning experiments, 2019.

[35] Robert Tjarko Lange and Henning Sprekeler. Learning not to learn: Nature versus nurture in silico. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI-22)*, volume 36, pages 7290–7299, 2022.

[36] Thomas Miconi, Aditya Rawal, Jeff Clune, and Kenneth O Stanley. Backpropamine: training self-modifying neural networks with differentiable neuromodulated plasticity. In *International Conference on Learning Representations*, 2018.

[37] Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

[38] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn, 2016.

[39] Chris Lu, Jakub Kuba, Alistair Letcher, Luke Metz, Christian Schroeder de Witt, and Jakob Foerster. Discovered policy optimisation. In *Advances in Neural Information Processing Systems*, volume 35, pages 16455–16468, 2022.

[40] Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International conference on machine learning*, pages 449–458. PMLR, 2017.

[41] 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*, pages 8029–8052. PMLR, 2023.

[42] Richard Dearden, Nir Friedman, and Stuart Russell. Bayesian Q-learning. In Jack Mostow and Chuck Rich, editors, *Proceedings of the Fifteenth National Conference on Artificial Intelligence and Tenth Innovative Applications of Artificial Intelligence Conference, AAAI 98, IAAI 98, July 26-30, 1998, Madison, Wisconsin, USA*, pages 761–768. AAAI Press / The MIT Press, 1998.

[43] Carlos E Luis, Alessandro G Bottero, Julia Vinogradska, Felix Berkenkamp, and Jan Peters. Value-distributional model-based reinforcement learning. *arXiv preprint arXiv:2308.06590*, 2023.

A Proofs

Proof of theorem 4.2. Recall that the Wasserstein distance can be written as

$$W_p(P_X, P_Y) = \left(\inf_{R \in \mathcal{R}_{XY}} \mathbb{E}_{(X,Y)\sim R}(\|X - Y\|^p) \right)^{\frac{1}{p}},$$

where $\mathcal{R}_{XY}$ is the set of joint probability measures that has marginals P_X and P_Y.

We can define an operator analogous to the EBO that works on the joint space instead:

$$\hat{\mathcal{B}}R(X, Y) = \text{Law}\left((B_{\mathcal{D}}(X) + \epsilon_{\mathcal{D}},\ B_{\mathcal{D}}(Y) + \epsilon_{\mathcal{D}})\right),$$

where crucially both $\epsilon_{\mathcal{D}}$ are the **same** variable.

Because $\hat{\mathcal{B}}\bar{R}$ has marginals $\mathcal{B}P_X$ and $\mathcal{B}P_Y$ if $\bar{R}$ has marginals P_X and P_Y, we have

$$W_p(\mathcal{B}P_X, \mathcal{B}P_Y)^p = \inf_{R \in \mathcal{R}_{\mathcal{B}P_X, \mathcal{B}P_Y}} \mathbb{E}_R\left[\|X - Y\|^p\right] \leq \inf_{\bar{R} \in \mathcal{R}_{P_X, P_Y}} \mathbb{E}_{\hat{\mathcal{B}}\bar{R}}\left[\|X - Y\|^p\right]. \tag{15}$$

The inequality holds because the first infimum is over all R with marginals $\mathcal{B}P_X$ and $\mathcal{B}P_Y$, while the second infimum is restricted to the image of $\hat{\mathcal{B}}$.

Finally, using Equation 15, we can conclude that

$$W_p(\mathcal{B}P_X, \mathcal{B}P_Y)^p \leq \inf_{\bar{R} \in \mathcal{R}_{P_X, P_Y}} \mathbb{E}_{\hat{\mathcal{B}}\bar{R}}\left[\|X - Y\|^p\right] = \inf_{\bar{R}} E_{\bar{R}}\|B_{\mathcal{D}}(X) + \epsilon_{\mathcal{D}} - B_{\mathcal{D}}(Y) - \epsilon_{\mathcal{D}}\|^p \tag{16}$$

$$= \inf_{\bar{R}} E_{\bar{R}}\|B_{\mathcal{D}}(X) - B_{\mathcal{D}}(Y)\|^p \leq \inf_{\bar{R}} E_{\bar{R}}\gamma^p\|X - Y\|^p \leq \gamma^p \inf_{\bar{R}} E_{\bar{R}}\|X - Y\|^p \tag{17}$$

$$= (\gamma W_p(P_X, P_Y))^p, \tag{18}$$

where in Equation 17 we use the fact that $B_{\mathcal{D}}$ is a γ-contraction in $\|.\|$. □

Proof of theorem 4.3. Since P_B is a fixed point of $\mathcal{B}$, we have

$$\mathbb{E}_{P_B}[Q] = \mathbb{E}_{\mathcal{B}P_B}[Q] = \mathbb{E}_{P_\epsilon}\mathbb{E}_{P_B}[BQ + \epsilon] = \mathbb{E}_{P_B}[BQ] = B[\mathbb{E}_{P_B}[Q]], \tag{19}$$

where the second equality is the definition of the EBO, the third equality follows from independence of ϵ and Q, and the last equality uses the fact that B is an affine function, proving that $\mathbb{E}_{P_B}[Q]$ is a fixed point of B.

Since B is affine we define A and b such that $BQ = AQ + b$.

For the covariance we have

$$\begin{aligned}
\mathbb{E}_{P_B}\left[QQ^\top\right] &= \mathbb{E}_{P_\epsilon}\left[\mathbb{E}_{P_B}\left[(BQ + \epsilon)(BQ + \epsilon)^\top\right]\right] \\
&= \mathbb{E}_{P_\epsilon}\left[\mathbb{E}_{P_B}\left[(BQ)(BQ)^\top + BQ\epsilon^\top + \epsilon(BQ)^\top + \epsilon\epsilon^\top\right]\right] \\
&= \mathbb{E}_{P_B}\left[(BQ)(BQ)^\top\right] + \mathbb{E}_{P_\epsilon}\mathbb{E}_{P_B}\left[BQ\epsilon^\top\right] + \mathbb{E}_{P_\epsilon}\mathbb{E}_{P_B}\left[\epsilon(BQ)^\top\right] + \mathbb{E}_{P_\epsilon}\left[\epsilon\epsilon^\top\right] \\
&= \mathbb{E}_{P_B}\left[(BQ)(BQ)^\top\right] + \mathbb{E}_{P_\epsilon}\left[\epsilon^\top\right]\mathbb{E}_{P_B}\left[BQ\right] + \mathbb{E}_{P_\epsilon}\left[\epsilon\right]\mathbb{E}_{P_B}\left[(BQ)^\top\right] + \mathbb{E}_{P_\epsilon}\left[\epsilon\epsilon^\top\right] \\
&= \mathbb{E}_{P_B}\left[(BQ)(BQ)^\top\right] + \mathbb{E}_{P_\epsilon}\left[\epsilon\epsilon^\top\right] \\
&= \mathbb{E}_{P_B}\left[(BQ)(BQ)^\top\right] + \Sigma_\epsilon \\
&= \mathbb{E}_{P_B}\left[(AQ + b)(Q^\top A^\top + b^\top\right] + \Sigma_\epsilon \\
&= A\mathbb{E}_{P_B}\left[QQ^\top\right]A^\top + b\mathbb{E}_{P_B}\left[Q^\top\right] + \mathbb{E}_{P_B}\left[Q\right]b^\top + bb^\top + \Sigma_\epsilon \\
&= A\mathbb{E}_{P_B}\left[QQ^\top\right]A^\top + b\mathbb{E}_{P_B}\left[Q^\top\right] + \mathbb{E}_{P_B}\left[Q\right]b^\top + bb^\top + \Sigma_\epsilon \\
&= A\mathbb{E}_{P_B}\left[QQ^\top\right]A^\top + bQ_B^\top + Q_Bb^\top + bb^\top + \Sigma_\epsilon,
\end{aligned} \tag{20}$$

and

$$Q_BQ_B^\top = (BQ_B)(BQ_B)^\top = AQ_BQ_B^\top A^\top + bQ_B^\top + Q_Bb^\top + bb^\top,$$

so

$$\begin{aligned}\mathbb{E}_{P_B}\left[QQ^\top - Q_B Q_B^\top\right] &= A\mathbb{E}_{P_B}\left[QQ^\top\right]A^\top - AQ_BQ_B^\top A^\top + \Sigma_\epsilon \\ &= A\mathbb{E}_{P_B}\left[QQ^\top - Q_BQ_B^\top\right]A^\top + \Sigma_\epsilon.\end{aligned} \tag{21}$$

Vectorizing both sides yields

$$\text{Vec}(\mathbb{E}_{P_B}\left[QQ^\top - Q_BQ_B^\top\right]) - \text{Vec}(A\mathbb{E}_{P_B}\left[QQ^\top - Q_BQ_B^\top\right]A^\top) = \tag{22}$$

$$\text{Vec}(\mathbb{E}_{P_B}\left[QQ^\top - Q_BQ_B^\top\right]) - (A\otimes A)\text{Vec}(\mathbb{E}_{P_B}\left[QQ^\top - Q_BQ_B^\top\right]) = \tag{23}$$

$$\text{Vec}(\Sigma_\epsilon). \tag{24}$$

Finally, since B is a contraction, the absolute eigenvalues of A must be strictly smaller than 1. By basic properties of the Kronecker product, $A \otimes A$ then also has absolute eigenvalues strictly smaller than one. Therefore, $I - (A \otimes A)$ has only non-zero eigenvalues, and hence is invertible.

We conclude that

$$\text{Vec}(\mathbb{E}_{P_B}\left[QQ^\top - Q_BQ_B^\top\right]) = (I - A\otimes A)^{-1}\text{Vec}(\Sigma_\epsilon)$$

to finish the proof. □

B Experiment Details

Code is available at `github.com/pascal314`.

B.1 Tabular

The MDP in this experiment has reward function $R = [0.05192758, -0.7084503]$, and the evaluated policy is $\pi = [0.4352794, 0.5647206]$. We use a standard 1 step Bellman operator B such that

$$BQ = R + \gamma T^\pi Q,$$

and the likelihood is given by $p(Q|q') = Bq' + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \texttt{diag}(0.2, 0.2))$. The initial distribution is a normal distribution with mean 0 and covariance $\texttt{diag}(2, 2)$.

These experiments were completed on a single CPU within a few minutes.

B.2 ECPPO

We modify PureJaxRL's implementation of PPO to have an ensemble of value models of size 5. For each time step t, the clipping parameter of the policy ratio is modified to $\epsilon\phi(U_t)$, where

$$\phi(u) = \frac{1}{2} + \frac{3}{2}\sigma\left(15 \cdot (-u + 0.3)\right),$$

and σ is the sigmoid function.

We hand-picked this function as a simple candidate that maps $[0, \infty) \to [0.5, 2.0]$, so that ECPPO can either halve or double the clipping range based on the uncertainty. To find reasonable values for scaling and shifting the uncertainty, we conducted an initial experiment on Acrobot-v1 where we empirically evaluated the expected modification to the clipping $\mathbb{E}_U[\phi(U)]$. We then picked the values 15 and 0.3 so that $\mathbb{E}_U[\phi(U)] \approx 1$ on this environment. We did not conduct any other hyperparameter optimization based on algorithm performance to obtain $\phi(u)$. While we used a specifically shaped function, we note that any positive and monotonically decreasing function ϕ is a valid choice.

We left all further hyperparameters unmodified from the baseline. For completion, we list the hyperparemeters of PureJaxRL's implementation in Table 1.

The learning rate starts at 0.005 and follows a linear schedule down to 0 at the final episode.

The network architectures for both the actor and the value is a fully connected network with hidden sizes 64 and 64, and relu activations. The actor and value networks share no parameters. Epistemic

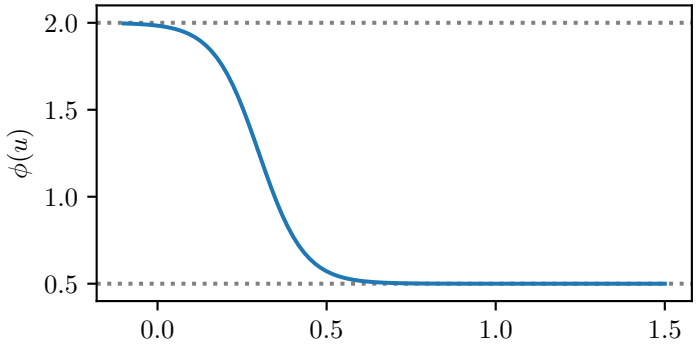

Figure 7: A plot of the function $\phi(u)$ used in our ECPPO experiments.

Parallel environments	64
Rollout length	128
Epochs	4
Batch size	1024
γ	0.99
λ	0.95
ϵ	[0.1, 0.2, 0.4]
Entropy loss factor	0.01
Value loss factor	0.5
Max gradient norm	0.5

Table 1: Hyperparameters of the PPO Baseline

Clipping PPO makes no modification to the actor network, and only replaces the value network with an ensemble with randomized prior functions of size 5. The randomized prior functions are scaled by $\beta = 1$.

Each baseline and variant of ECPPO ran for 20 seeds on each environment. All experiments were completed on a single GPU, taking around one minute per agent per environment for all 20 seeds in parallel.