Abstract

We consider the problem of quantifying uncertainty over expected cumulative rewards in model-based reinforcement learning. In particular, we focus on characterizing the variance over values induced by a distribution over MDPs. Previous work upper bounds the posterior variance over values by solving a so-called uncertainty Bellman equation, but the over-approximation may result in inefficient exploration. We propose a new uncertainty Bellman equation whose solution converges to the true posterior variance over values and explicitly characterizes the gap in previous work. Moreover, our uncertainty quantification technique is easily integrated into common exploration strategies and scales naturally beyond the tabular setting by using standard deep reinforcement learning architectures. Experiments in difficult exploration tasks, both in tabular and continuous control settings, show that our sharper uncertainty estimates improve sample-efficiency.

1 INTRODUCTION

The goal of reinforcement learning (RL) agents is to maximize the expected return via interactions with an a priori unknown environment (Sutton and Barto, 2018). In model-based RL (MBRL), the agent learns a statistical model of the environment, which can then be used for efficient exploration (Sutton, 1991; Strehl and Littman, 2008; Jaksch et al., 2010). The performance of deep MBRL algorithms was historically lower than that of model-free methods, but the gap has been closing in recent years (Janner et al., 2019). Key to these improvements are models that quantify epistemic and aleatoric uncertainty (Depeweg et al., 2018; Chua et al., 2018) and algorithms that leverage model uncertainty to optimize the policy (Curi et al., 2020). Still, a core challenge in MBRL is to quantify the uncertainty in long-term performance predictions of a policy given a probabilistic model of the dynamics (Deisenroth and Rasmussen, 2011). Leveraging predictive uncertainty of the policy performance during policy optimization facilitates deep exploration — methods that reason about the long-term information gain of rolling out a policy — which has shown promising results in the model-free (Osband et al., 2016; Ciosek et al., 2019) and model-based settings (Deisenroth and Rasmussen, 2011; Fan and Ming, 2021).

We adopt a Bayesian perspective on RL to characterize uncertainty in the decision process via a posterior distribution. This distributional perspective of the RL environment induces distributions over functions of interest for solving the RL problem, e.g., the expected return of a policy, also known as the value function. This perspective differs from distributional RL (Bellemare et al., 2017), whose main object of study is the distribution of the return induced by the inherent stochasticity of the MDP and the policy. As such, distributional RL models aleatoric uncertainty, whereas Bayesian RL focuses on the epistemic uncertainty arising from finite data of the underlying MDP. Recent work by Eriksson et al. (2022) and Moskovitz et al. (2021) combines Bayesian and distributional RL for various risk measures accounting for both sources of uncertainty.

We focus on model-based Bayesian RL, where the value distribution is induced by a posterior over MDPs. In particular, we analyze the variance of such a distribution of values. Schneegass et al. (2010) estimate uncertainty in value functions using statistical uncertainty propagation, with the caveat of assuming the value distribution is Gaussian. Previous results by O’Donoghue et al. (2018); Zhou et al. (2020) establish upper-bounds on the posterior variance of the values by solving a so-called uncertainty Bellman equation (UBE). These results make no assumptions on the value distribution and are amenable for deep RL implementations. However, these bounds over-approximate the variance of the values and thus may lead to inefficient exploration when used for uncertainty-aware optimization (e.g., risk-seeking or risk-averse policies). In principle, tighter uncertainty estimates have the potential to improve data-efficiency, which
is the main motivation behind this paper.

**Our contribution.** We show that, under the same assumptions as previous work, the posterior variance of the value function obeys a Bellman-style recursion exactly. Our theory characterizes the gap in the previously tightest upper-bound by Zhou et al. (2020), which ignores the inherent aleatoric uncertainty of acting in a potentially stochastic MDP. Inspired by this insight, we propose learning the solution to the Bellman recursion prescribed by our theory, as done by O’Donoghue et al. (2018), but integrate it within an actor-critic framework for continuous action problems, rather than using DQN (Mnih et al., 2013) for discrete action selection. Our experiments in tabular and continuous control problems demonstrate that our variance estimation method improves sample efficiency when used for optimistic optimization of the policy. The source code is available\(^1\).

**Related work.** 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). Jorge et al. (2020) estimate value distributions using a backwards induction framework, while Metelli et al. (2019) propagate uncertainty using Wasserstein barycenters. Fellows et al. (2021) showed that, due to bootstrapping, model-free Bayesian methods infer a posterior over Bellman operators rather than values.

Model-based Bayesian RL maintains a posterior over plausible MDPs given the available data, which induces a distribution over values. The MDP uncertainty is typically represented in the one-step transition model as a by-product of model-learning. For instance, the well-known PILCO algorithm by Deisenroth and Rasmussen (2011) learns a Gaussian Process (GP) 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) propose PETS, which uses 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. Dyna-style (Sutton, 1991) actor-critic algorithms have been paired with model-based uncertainty estimates for improved performance in both online (Buckman et al., 2018; Zhou et al., 2019) and offline (Yu et al., 2020; Kidambi et al., 2020) RL.

To balance exploration and exploitation, provably-efficient RL algorithms based on optimism in the face of the uncertainty (OFU) (Auer and Ortner, 2006; Jaksch et al., 2010) rely on building upper-confidence (optimistic) estimates of the true values. These optimistic values correspond to a modified MDP where the rewards are enlarged by an uncertainty bonus, which encourages exploration. In practice, however, the aggregation of optimistic rewards may severely over-estimate the true values, rendering the approach inefficient (Osband and Van Roy, 2017). O’Donoghue et al. (2018) show that methods that approximate the variance of the values can result in much tighter upper-confidence bounds, while Ciosek et al. (2019) demonstrate their use in complex continuous control problems. Similarly, Chen et al. (2017) propose a model-free ensemble-based approach to estimate the variance of values.

Interest about the higher moments of the return of a policy dates back to the work of Sobel (1982), showing these quantities obey a Bellman equation. Methods that leverage these statistics of the return are known as distributional RL (Tamar et al., 2013; Bellemare et al., 2017). Instead, we focus specifically on estimating and using the variance of the expected return for policy optimization. A key difference between the two perspectives is the type of uncertainty they model: distributional RL models the aleatoric uncertainty about the returns, which originates from the aleatoric noise of the MDP transitions and the stochastic policy; our perspective studies the epistemic uncertainty about the value function, due to incomplete knowledge of the MDP. Provably efficient RL algorithms use this isolated epistemic uncertainty as a signal to balance exploring the environment and exploiting the current knowledge.

O’Donoghue et al. (2018) propose a UBE whose fixed-point solution converges to a guaranteed upper-bound on the posterior variance of the value function in the tabular RL setting. This approach was implemented in a model-free fashion using the DQN (Mnih et al., 2013) architecture and showed performance improvements in Atari games. Follow-up work by Markou and Rasmussen (2019) empirically shows that the upper-bound is loose and the resulting over-approximation of the variance impacts negatively the regret in tabular exploration problems. Zhou et al. (2020) propose a modified UBE with a tighter upper-bound on the value function, which is then paired with proximal policy optimization (PPO) (Schulman et al., 2017) in a conservative on-policy model-based approach to solve continuous-control tasks. We propose a new UBE and integrate it within a model-based soft actor-critic (Haarnoja et al., 2018) architecture similar to Janner et al. (2019); Froehlich et al. (2022).

2 PROBLEM STATEMENT

We consider an agent that acts in an infinite-horizon MDP \(\mathcal{M} = \{S, A, p, r, \gamma\}\) with finite state space \(|S| = S\), finite action space \(|A| = A\), unknown transition function \(p : S \times A \rightarrow \Delta(S)\) that maps states and actions to the \(S\)-dimensional probability simplex, an initial state distribution \(p : S \rightarrow [0, 1]\), a known and bounded reward function \(r : S \times A \rightarrow \mathbb{R}\), and a discount factor \(\gamma \in [0, 1]\). Although we consider a known reward function, the main theoretical

\(^1\)https://github.com/boschresearch/ube-mbrl
results can be easily extended to the case where it is learned alongside the transition function (see Appendix B.1). The one-step dynamics $p(s' \mid s, a)$ denote the probability of going from state $s$ to state $s'$ after taking action $a$. In general, the agent selects actions from a stochastic policy $\pi : \mathcal{S} \rightarrow \Delta(\mathcal{A})$ that defines the conditional probability distribution $\pi(a \mid s)$. At each time step of episode $t$ the agent is in some state $s$, selects an action $a \sim \pi(t \mid s)$, receives a reward $r(s, a)$, and transitions to a next state $s' \sim p(\cdot \mid s, a)$. We define the value function $V^{\pi,p} : \mathcal{S} \rightarrow \mathbb{R}$ of a policy $\pi$ and transition function $p$ as the expected sum of discounted rewards under the MDP dynamics,

$$V^{\pi,p}(s) = \mathbb{E}_{r \sim p} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s \right], \quad (1)$$

where the expectation is taken under the random trajectories $\tau$ drawn from the trajectory distribution $P(\tau) = \prod_{h=0}^{\infty} \pi(a_h \mid s_h)p(s_{h+1} \mid s_h, a_h)$.

We consider a Bayesian setting similar to previous work by O’Donoghue et al. (2018); O’Donoghue (2021); Zhou et al. (2020), in which the transition function $p$ is a random variable with some known prior distribution $\Phi_0$. Define the transition data observed up to episode $t$ as $\mathcal{D}_t$, then we update our belief about the random variable $p$ by applying Bayes’ rule to obtain the posterior distribution conditioned on $\mathcal{D}_t$, which we denote as $\Phi_t$. The distribution of transition functions naturally induces a distribution over value functions. The main focus of this paper is to study methods that estimate the variance of the value function $V^{\pi,p}$ under the posterior distribution $\Phi_t$, namely $V_{p \sim \Phi_t} [V^{\pi,p}(s)]$.

Our theoretical results extend to state-action value functions (see Appendix B.2). The motivation behind studying this quantity is its potential use for exploring the environment.

Zhou et al. (2020) introduce a method to upper-bound the variance of Q-values by solving a UBE. Their theory holds for a class of MDPs where the value functions and transition functions are uncorrelated. This family of MDPs is characterized by the following assumptions:

**Assumption 1** (Independent transitions), $p(s' \mid x, a)$ and $p(s' \mid y, a)$ are independent random variables if $x \neq y$.

**Assumption 2** (Acyclic MDP (O’Donoghue et al., 2018)). The MDP $M$ is a directed acyclic graph, i.e., states are not visited more than once in any given episode.

Assumption 1 holds naturally in the case of discrete state-action spaces with a tabular transition function, where there is no generalization. Assumption 2 is non-restrictive as any finite-horizon MDP with cycles can be transformed into an equivalent time-inhomogeneous MDP without cycles by adding a time-step variable $h$ to the state-space. Similarly, for infinite-horizon MDPs we can consider an effective horizon $H = 1/1 - \gamma$ and apply the same logic. The direct consequence of these assumptions is that the random variables $V^{\pi,p}(s')$ and $p(s' \mid s, a)$ are uncorrelated (see Lemmas 2 and 3 in Appendix A.1 for a formal proof).

Other quantities of interest are the posterior mean transition function starting from the current state-action pair $(s, a)$,

$$\mathbb{E}_{p \sim \Phi_t}[p(\cdot \mid s, a)], \quad (2)$$

and the posterior mean value function for any $s \in \mathcal{S}$,

$$V_t^{\pi}(s) = \mathbb{E}_{p \sim \Phi_t} [V^{\pi,p}(s)], \quad (3)$$

where the subscript $t$ represents the dependency on $\mathcal{D}_t$ of both quantities. Note that $\mathbb{E}_t$ is a transition function that combines both aleatoric and epistemic uncertainty. Even if we limit the posterior $\Phi_t$ to only include deterministic transition functions, $\mathbb{E}_t$ remains a stochastic transition function due to the epistemic uncertainty.

Zhou et al. (2020) define the local uncertainty

$$w_t(s) = \mathbb{V}_{p \sim \Phi_t} \left[ \sum_{a,s'} \pi(a \mid s)p(s' \mid s, a)\mathbb{V}_t^{\pi}(s') \right], \quad (4)$$

and solve the UBE

$$W_t^{\pi}(s) = \gamma^2 w_t(s) + \gamma^2 \sum_{a,s'} \pi(a \mid s)\mathbb{E}_t(s')W_t^{\pi}(s'), \quad (5)$$

whose unique solution satisfies $W_t^{\pi} \geq \mathbb{V}_{p \sim \Phi_t} [V^{\pi,p}(s)]$.

### 3 Uncertainty Bellman Equation

In this section, we build a new UBE whose fixed-point solution is equal to the variance of the value function and we explicitly gap between (5) and $\mathbb{V}_{p \sim \Phi_t} [V^{\pi,p}(s)]$.

The values $V^{\pi,p}$ are the fixed-point solution to the Bellman expectation equation, which relates the value of the current state $s$ with the value of the next state $s'$. Further, under Assumptions 1 and 2, applying the expectation operator to the Bellman recursion results in $\bar{V}_t^{\pi}(s) = V^{\pi,\bar{p}_t}(s)$. The Bellman recursion propagates knowledge about the local rewards $r(s, a)$ over multiple steps, so that the value function encodes the long-term value of states if we follow policy $\pi$. Similarly, a UBE is a recursive formula that propagates a notion of local uncertainty, $u_t(s)$, over multiple steps. The fixed-point solution to the UBE, which we call the $U$-values, encodes the long-term epistemic uncertainty about the values of a given state.

Previous formulations by O’Donoghue et al. (2018); Zhou et al. (2020) differ only on their definition of the local uncertainty and result on $U$-values that upper-bound the posterior variance of the values. The first key insight of our paper is that we can define $u_t$ such that the $U$-values converge exactly to the variance of values. This result is summarized in the following theorem:

**Theorem 1.** Under Assumptions 1 and 2, for any $s \in \mathcal{S}$ and policy $\pi$, the posterior variance of the value function,
Model-Based Uncertainty in Value Functions

\[ U^\pi_t = \mathbb{V}_{p \sim \Phi_t}[V^\pi \cdot p] \text{ obeys the uncertainty Bellman equation} \]

\[ U^\pi_t(s) = \gamma^2 u_t(s) + \gamma^2 \sum_{a', s'} \pi(a \mid s) \hat{p}_t(s' \mid s, a) U^\pi_t(s'), \]

where \( u_t(s) \) is the local uncertainty defined as

\[ u_t(s) = \mathbb{V}_{a, s' \sim \pi_t, \hat{p}_t} [V^\pi_t(s')] - \mathbb{E}_{p \sim \Phi_t} [\mathbb{V}_{a, s' \sim \pi_t, p} [V^\pi_t(s')]]. \] \tag{6}

Theorem 2. Under Assumptions 1 and 2, for any \( s \in S \) and policy \( \pi \), it holds that \( u_t(s) = w_t(s) - g_t(s) \), where \( g_t(s) = \mathbb{E}_{p \sim \Phi_t} [\mathbb{V}_{a, s' \sim \pi_t, p} [V^\pi_t(s')] - \mathbb{V}_{a, s' \sim \pi_t, \hat{p}_t} [V^\pi_t(s')]] \). Furthermore, we have that the gap \( g_t(s) \) is non-negative, thus \( u_t(s) \leq w_t(s) \).

Proof. See Appendix A.2. \qed

The gap \( g_t(s) \) of Theorem 2 can be interpreted as the average difference of aleatoric uncertainty about the next values with respect to the mean values. The gap vanishes only if the epistemic uncertainty goes to zero, or if the MDP and policy are both deterministic.

We directly connect Theorems 1 and 2 via the equality

\[ \mathbb{V}_{a, s' \sim \pi_t, \hat{p}_t} [V^\pi_t(s')] = u_t(s) + \mathbb{E}_{p \sim \Phi_t} [\mathbb{V}_{a, s' \sim \pi_t, p} [V^\pi_t(s')]] \]

which helps us analyze our theoretical results. The uncertainty reward defined in (7) has two components: the first term corresponds to the total uncertainty about the mean values of the next state, which is further decomposed in (8) into an epistemic and aleatoric components. When the epistemic uncertainty about the MDP vanishes, then \( w_t(s) \to 0 \) and only the aleatoric component remains. Similarly, when the MDP and policy are both deterministic, the aleatoric uncertainty vanishes and we have \( \mathbb{V}_{a, s' \sim \pi_t, \hat{p}_t} [V^\pi_t(s')] = u_t(s) \).

Corollary 1. Under Assumptions 1 and 2, for any \( s \in S \) and policy \( \pi \), it holds that the solution to the uncertainty Bellman equation (5) satisfies \( W^\pi_t(s) = \geq U^\pi_t(s) \).

Proof. The solution to the Bellman equations (5) and (6) are the value functions under some policy \( \pi \) of identical MDPs except for their reward functions. Given two identical MDPs \( M_1 \) and \( M_2 \) differing only on their corresponding reward functions \( r_1 \) and \( r_2 \), if \( r_1 \leq r_2 \) for any input value, then for any trajectory \( \tau \) we have that the returns (sum of discounted rewards) must obey \( R_1(\tau) \leq R_2(\tau) \). Lastly, since the value functions \( V^\pi_1, V^\pi_2 \) are defined as the expected returns under the same trajectory distribution, and the expectation operator preserves inequalities, then we have that \( R_1(\tau) \leq R_2(\tau) \implies V^\pi_1 \leq V^\pi_2 \).

Corollary 1 reaches the same conclusions as Zhou et al. (2020), but it brings important explanations about their upper bound on the variance of the value function. First, by Theorem 2 the upper bound is a consequence of the over approximation of the reward function used to solve the UBE. Second, the gap between the exact reward function \( u_t(s) \) and the approximation \( w_t(s) \) is fully characterized by \( g_t(s) \) and brings interesting insights. In particular, the influence of the gap term depends on the stochasticity of the dynamics and the policy. In the limit, the term vanishes under deterministic transitions and action selection. In this scenario, the upper-bound found by Zhou et al. (2020) becomes tight.

Our method returns the exact epistemic uncertainty about the values by considering the inherent aleatoric uncertainty of the MDP and the policy. In a practical RL setting, disentangling the two sources of uncertainty is key for effective exploration. We are interested in exploring regions of high epistemic uncertainty, where new knowledge can be obtained. If the variance estimate fuses both sources of uncertainty, then we may be guided to regions of high uncertainty but with little information to be gained.

3.1 Toy Example

To illustrate the theoretical findings of this paper, consider the simple Markov reward process (MRP) of Figure 1. Assume \( \delta \) and \( \beta \) to be random variables drawn from a discrete uniform distribution \( \delta \sim \text{Unif}(0.7, 0.6) \) and
we update the policy by solving the upper-confidence bound (UCB) (Auer and Ortner, 2006) optimization problem.

\[ \pi_t = \arg\max_{\pi} Q_t^\pi + \lambda \sqrt{\tilde{U}_t^\pi}, \quad (9) \]

where \( Q_t^\pi \) is the posterior mean value function and \( \lambda \) is a parameter that trades off exploration and exploitation. We use Algorithm 1 to estimate \( Q_t^\pi \) and \( U_t^\pi \): we sample an ensemble of \( N \) MDPs from the current posterior \( \Gamma_t \) in Line 2 and use it to solve the Bellman expectation equation in Line 3, resulting in an ensemble of \( N \) corresponding \( Q \) functions and the posterior mean \( Q_t^* \). Lastly, \( \tilde{U}_t^\pi \) is estimated in Line 4 via a generic variance estimation method \( \text{qvariance} \) for which we consider three implementations: \text{ensemble-var} computes a sample-based approximation of the variance given by \( \sqrt{Q_t} \), which is a model-based version of the estimate from Chen et al. (2017); \text{pombu} uses the solution to the UBE (5); and \text{exact-ube} uses the solution to our proposed UBE (6). For the UBE-based methods we use the equivalent equations for \( Q \)-functions, see Appendix B.3 for details.

**Algorithm 1 Model-based \( Q \)-variance estimation**

1: **Input:** Posterior MDP \( \Gamma_t \), policy \( \pi \).
2: \{\( p_i, r_i \)\}_{i=1}^N \leftarrow \text{sample_mdp}(\Gamma_t)
3: \( \hat{Q}_t, \{Q_t\}_{i=1}^N \leftarrow \text{solve_bellman}(\{p_i, r_i\}_{i=1}^N, \pi) \)
4: \( \tilde{U}_t^\pi \leftarrow \text{qvariance}(\{p_i, r_i, Q_t\}_{i=1}^N, \hat{Q}_t, \pi) \)

**Practical bound.** In practice, typical RL techniques for model learning violate our theoretical assumptions. For tabular implementations, flat prior choices like a Dirichlet distribution violate Assumption 2 while function approximations introduce correlations between states and thus violate Assumption 1. A challenge arises in this practical setting: \text{exact-ube} may result in negative \( U \)-values, as a combination of (i) the assumptions not holding and (ii) the possibility of negative uncertainty rewards. While (i) cannot be easily resolved, we propose a practical upper-bound on the solution of (6) such that the resulting \( U \)-values are non-negative and hence interpretable as variance estimates. We consider the clipped uncertainty rewards \( \tilde{u}_t = \max(u_{\text{min}}, u_t(s)) \) with corresponding \( U \)-values \( \tilde{U}_t^\pi \). It is straightforward to prove that, if \( u_{\text{min}} = 0 \), then \( W_t^\pi(s) \geq \tilde{U}_t^\pi(s) \geq U_t^\pi(s) \), which means that using \( \tilde{U}_t^\pi \) still results in a tighter upper-bound on the variance than \( W_t^\pi \), while preventing non-positive solutions to the UBE. In what follows, we drop this notation and assume all \( U \)-values are computed from clipped uncertainty rewards. Also note that \text{pombu} does not have this problem, since \( u_t(s) \) is already non-negative.

**Tabular implementation.** For model learning, we impose a Dirichlet prior on the transition function and a standard Normal prior for the rewards (O’Donoghue et al., 2019), which leads to closed-form posterior updates. After sampling \( N \) times from the MDP posterior (Line 2), we obtain the \( Q \)-functions (Line 3) in closed-form by solving the corresponding Bellman equation. For the UBE-based approaches, we estimate uncertainty rewards via approximations of the expectations/variances therein. Lastly, we solve
We adopt as a baseline architecture MBPO by Janner et al. (2019) and the implementation from Pineda et al. (2021). In contrast to the tabular implementation, maintaining an explicit distribution over MDPs from which we can sample is intractable. Instead, we consider $\Gamma_t$ to be a discrete uniform distribution of $N$ probabilistic neural networks, denoted $p_0$, that output the mean and covariance of a Gaussian distribution over next states and rewards (Chua et al., 2018). In this case, the output of Line 2 in Algorithm 1 is precisely the ensemble of neural networks.

The original MBPO trains $Q$-functions represented as neural networks via TD-learning on data generated via model-randomized $k$-step rollouts from initial states that are sampled from $D_t$. Each forward prediction of the rollout comes from a randomly selected model of the ensemble and the transitions are stored in a single replay buffer $D_{\text{model}}$ which is then fed into a model-free optimizer like soft actor-critic (SAC) (Haarnoja et al., 2018). SAC trains a stochastic policy represented as a neural network with parameters $\phi$, denoted by $\pi_\phi$. The policy’s objective function is similar to (9) but with entropy regularization instead of the uncertainty term. In practice, the argmax is replaced by $G$ steps of stochastic gradient ascent, where the policy gradient is estimated via mini-batches drawn from $D_{\text{model}}$.

Algorithm 1 requires a few modifications from the MBPO methodology. To implement Line 3, in addition to $D_{\text{model}}$, we create $N$ new buffers $\{D_{\text{model}}^{(i)}\}_{i=1}^N$ filled with model-consistent rollouts, where each $k$-step rollout is generated under a single model of the ensemble, starting from initial states sampled from $D_t$. We train an ensemble of $N$ value functions $\{Q_i\}_{i=1}^N$, parameterized by $\{\psi_i\}_{i=1}^N$, and minimize the residual Bellman error with entropy regularization instead of the uncertainty term.

To approximate the solution to the UBE, we train a neural network parameterized by a vector $\varphi$, denoted $U_\varphi$ (informally, the $U$-net). Since we interpret the output of the network as predictive variances, we (i) regularize the output to be positive by penalizing negative values and (ii) use a softplus output layer to guarantee non-negative values. For regularization, let $f_\varphi$ be the network output before the softplus operation, then we define the regularization loss

$$\mathcal{L}_{\text{reg}}(\varphi) = \mathbb{E}_{(s,a,r,s') \sim D_{\text{model}}}[\text{ReLU}(-f_\varphi(s,a) - \epsilon)^2],$$

such that $\mathcal{L}_{\text{reg}}(\varphi) \geq 0$ iff $f_\varphi(s,a) < \epsilon$ for some small $\epsilon > 0$. Otherwise, for $f_\varphi(s,a) > \epsilon$ the loss is zero and regularization is inactive. In practice, we found that regularization is key to avoid network collapse in sparse reward problems, while it is typically not required if rewards are dense. Training of the $U$-net is carried out by minimizing the uncertainty Bellman error with regularization:

$$\mathcal{L}(\varphi) = \mathbb{E}_{(s,a,r,s') \sim D_{\text{model}}}[\left(z - U(s,a;\varphi)\right)^2] + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}}(\varphi),$$

with targets $z = \gamma^2 u(s,a) + \gamma^2 U(s',a';\bar{\varphi})$ and target parameters $\bar{\varphi}$ updated like in regular critics. Lastly, we optimize $\pi_\varphi$ as in MBPO via SGD on the SAC policy loss, but also adding the uncertainty term from (9). A detailed algorithm of our approach is included in Appendix D.1.

**Runtime complexity.** In tabular RL, exact-ube solves $N+2$ Bellman equations ($Q^*_t, Q_t, U^*_t$), pombu solves two ($Q^*_t, U^*_t$) and ensemble-var solves $N+1$ ($Q^*_t, Q_t$). In deep RL, UBE-based methods have the added complexity of training the $U$-net, but it can be parallelized with the $Q$-ensemble training. Despite the increased complexity, we show in Section 5.3 that our method performs well for small $N$, which reduces the computational burden.

## 5 EXPERIMENTS

In this section, we empirically evaluate the performance of the policy optimization scheme (9) for the different variance estimates that we introduced in Section 4.

### 5.1 Tabular Environments

We evaluate the tabular implementation in grid-world environments. We include PSRL by Osband et al. (2013) as a baseline since it typically outperforms recent OFU-based methods (O’Donoghue, 2021; Tiapkin et al., 2022).

**DeepSea.** First proposed by Osband et al. (2019), this environment tests the agent’s ability to explore over multiple time steps in the presence of a deterrent. It consists of an $L \times L$ grid-world MDP, where the agent starts at the top-left cell and must reach the lower-right cell. The agent decides to move left or right, while always descending to the row below. We consider the deterministic version of the problem, so the agent always transitions according to the chosen action. Going left yields no reward, while going right incurs an action cost (negative reward) of 0.01/L. The bottom-right cell yields a reward of 1, so that the optimal policy is to always go right. As the size of the environment...
Figure 2: Performance in the DeepSea benchmark. Lower values in plots indicate better performance. (Left) Learning time is measured as the first episode where the sparse reward has been found at least in 10% of episodes so far. (Right) Total regret is approximately equal to the number of episodes where the sparse reward was not found. Results represent the average over 5 random seeds, and vertical bars on total regret indicate the standard error. Our variance estimate achieves the lowest regret and best scaling with problem size.

Figure 3: Total regret curve for the 7-room environment. Lower regret is better. Results are the average (solid lines) and standard error (shaded regions) over 10 random seeds. Our method achieves the lowest regret, significantly outperforming PSRL.

As implemented by Domingues et al. (2021), the 7-room environment consists of seven connected rooms of size $5 \times 5$. The agent starts in the center of the middle room and an episode lasts 40 steps. The possible actions are up-down-left-right and the agent transitions according to the selected action with probability $0.95$, otherwise it lands in a random neighboring cell. The environment has zero reward everywhere except two small rewards at the start position and in the left-most room, and one large reward in the right-most room. Unlike DeepSea, the underlying MDP for this environment contains cycles, so it evaluates our method beyond the theoretical assumptions. In Figure 3, we show the regret curves over 5000 episodes. Our method achieves the lowest regret, which is remarkable considering recent empirical evidence favoring PSRL over OFU-based methods in these type of environments (Tiapkin et al., 2022). The large gap between ensemble-var and the UBE-based methods is due to overall larger variance estimates from the former, which consequently requires more episodes to reduce the value uncertainty.
5.2 Continuous Control Environments

In this section, we evaluate the performance of the deep RL implementation in environments with continuous state-action spaces. Implementation details and hyperparameters are included in Appendix D.1.

Sparse Inverted Pendulum. As proposed by Curi et al. (2020), non-zero rewards only exist close to the upward position. The pendulum is always initialized in the downward position with zero velocity and one episode lasts 400 steps. Stochasticity is introduced via zero-mean Gaussian noise in the pendulum’s angle. We complicate the problem further by adding an action cost, which directly counteracts the effect of exploration signals. The combination of sparse rewards and action costs represent a failure case for model-free approaches relying on the stochasticity of the policy to explore (e.g. SAC). While noisy transitions may actually help solve these problems by increasing the random chance of encountering the sparse rewards, they also motivate the need for proper filtering of aleatoric noise when estimating the epistemic uncertainty.

The benchmark includes two additional baselines: ensemble-mean, which uses no optimism and only averages over the epistemic uncertainty of the Q-ensemble, and SAC. Figure 4 shows the learning curves over 75 episodes for three different noise levels. SAC quickly converges to the suboptimal solution of not applying any torque to the pendulum, while all model-based approaches avoid this pitfall. Overall, our exact-ube method has the most robust performance across the different noise levels, in most cases improving sample-efficiency and achieving comparable or higher final return. Importantly, exact-ube outperforms pombu in all scenarios, which is consistent with our theoretical insights about our method better handling aleatoric uncertainty. Perhaps surprisingly, greedily averaging over the epistemic uncertainty (ensemble-mean) is a strong baseline. Meanwhile, the ensemble-var method tends to over-explore due to higher variance estimates than the UBE-based methods, leading to more erratic learning curves and lower sample-efficiency (see Appendix D.4 for a visualization).

PyBullet Locomotion. We evaluate performance on three locomotion tasks from the PyBullet suite (Coumans and Bai, 2016), which have increased dimensionality compared to the simple pendulum environment. Although these environments have dense rewards, thus arguably less need for deep exploration, the results in Figure 5 demonstrate some performance improvement using exact-ube compared to the baselines. Similar to the pendulum task, ensemble-var affords higher variance estimates which severely hinders performance, while ensemble-mean is a strong baseline upon which some improvements can be afforded with UBE-based optimism.

While we cannot make broad claims based on these results, they provide supporting evidence that: (1) UBE-based methods can be scaled to continuous-control problems using U-nets and (2) our UBE formulation provides benefits in solving RL tasks with respect to prior work.

5.3 Ensemble Size Ablation

The ensemble size $N$ represents a critical hyperparameter for ensemble-based methods, balancing compute and sample diversity. The work by An et al. (2021) suggests that classical ensemble methods may require large $N$ to achieve good performance, which is computationally expensive. We evaluate this hypothesis through an ablation study over $N$ across different exploration tasks. The results in Figure 6 show that our method achieves the best or comparable performance across all environments and values of $N$. The ensemble-var estimate is more sensitive to $N$ and its performance increases for larger ensembles, matching
Figure 5: Learning curves in Pybullet locomotion environments. Returns are smoothened by a moving average and we report the mean (solid lines) and standard error (shaded regions) over 10 random seeds. While these environments have dense rewards, our UBE-based exploration method shows improvements in terms of learning speed and final performance.

Figure 6: Ablation study for the ensemble size $N$. We report the mean/standard error of the final total regret for DeepSea ($L = 30$) and 7-room across five and ten seeds, respectively. For the sparse pendulum, we set the angle noise standard deviation to 0 and show the mean/standard error of the final return after 75 episodes across ten seeds. All methods improve performance for larger $N$, but our method is able to achieve the best overall performance.

the observations from An et al. (2021). We hypothesize that sample-based approximations of the local uncertainty rewards, which typically have small magnitude, are less sensitive to sample size than directly estimating variance from the ensemble members. Further experiments in the pendulum environment (included in Appendix D.3) suggest that larger ensembles may not always lead to better performance in the presence of sparse rewards; in the absence of a strong reward signal, most ensemble members will agree on predicting close-to-zero values which may then lead to premature convergence of the policy. We hypothesize that for larger ensembles it is key to promote sufficient diversity to avoid variance collapse and solve the task.

6 CONCLUSIONS

In this paper, we derived an uncertainty Bellman equation whose fixed-point solution converges to the variance of values given a posterior distribution over MDPs. Our theory brings new understanding by characterizing the gap in previous UBE formulations that upper-bound the variance of values. We showed that this gap is the consequence of an over-approximation of the uncertainty rewards being propagated through the Bellman recursion, which ignore the inherent aleatoric uncertainty from acting in an MDP. Instead, our theory recovers exclusively the epistemic uncertainty due to limited environment data, thus serving as an effective exploration signal.

We proposed a practical method to estimate the solution of the UBE, scalable beyond tabular problems with standard deep RL practices. Our variance estimation was integrated into a model-based approach using the principle of optimism in the face of uncertainty to explore effectively. Experimental results showed that our method improves sample efficiency in hard exploration problems and without requiring large ensembles.
References


Erwin Coumans and Yunfei Bai. PyBullet, a Python module for physics simulation for games, robotics and machine learning, 2016.


Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOReL: Model-Based Offline


### Supplementary Material:
Model-Based Uncertainty in Value Functions

**Table of Contents**

<table>
<thead>
<tr>
<th>A THEORY PROOFS</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.1 Proof of Theorem 1</td>
<td></td>
</tr>
<tr>
<td>A.2 Proof of Theorem 2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B THEORY EXTENSIONS</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.1 Unknown Reward Function</td>
<td></td>
</tr>
<tr>
<td>B.2 Extension to $Q$-values</td>
<td></td>
</tr>
<tr>
<td>B.3 State-Action Uncertainty Rewards</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C TABULAR ENVIRONMENTS EXPERIMENTS</th>
<th>18</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.1 Implementation Details</td>
<td></td>
</tr>
<tr>
<td>C.2 Environment Details</td>
<td></td>
</tr>
<tr>
<td>C.3 DeepSea Additional Experiments</td>
<td></td>
</tr>
<tr>
<td>C.3.1 Extended Results</td>
<td></td>
</tr>
<tr>
<td>C.3.2 Uncertainty Rewards Ablation</td>
<td></td>
</tr>
<tr>
<td>C.3.3 Ensemble Size Ablation</td>
<td></td>
</tr>
<tr>
<td>C.3.4 Exploration Gain Ablation</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D CONTINUOUS CONTROL EXPERIMENTS</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>D.1 Implementation Details</td>
<td></td>
</tr>
<tr>
<td>D.2 Environment Details</td>
<td></td>
</tr>
<tr>
<td>D.3 Ensemble Size Ablation</td>
<td></td>
</tr>
<tr>
<td>D.4 Visualization of Variance Estimates</td>
<td></td>
</tr>
</tbody>
</table>
A THEORY PROOFS

A.1 Proof of Theorem 1

In this section, we provide the formal proof of Theorem 1. We begin by showing an expression for the posterior variance of the value function without assumptions on the MDP. We define the joint distribution $p^\pi(a, s' \mid s) = \pi(a \mid s)p(s' \mid s, a)$ for a generic transition function $p$. To ease notation, since $\pi$ is fixed, we will simply denote the joint distribution as $p(a, s' \mid s)$.

**Lemma 1.** For any $s \in S$ and any policy $\pi$, it holds that

$$\nabla_{p \sim \Phi_t}[V^\pi,p(s)] = \gamma^2 \mathbb{E}_{p \sim \Phi_t} \left[ \left( \sum_{a, s'} p(a, s' \mid s)V^\pi,p(s') \right)^2 \right] - \gamma^2 \left( \mathbb{E}_{p \sim \Phi_t} \left[ \sum_{a, s'} p(a, s' \mid s)V^\pi,p(s') \right] \right)^2. \quad (13)$$

**Proof.** Using the Bellman expectation equation

$$V^\pi,p(s) = \sum_a \pi(a \mid s)r(s, a) + \gamma \sum_{a, s'} p(a, s' \mid s)V^\pi,p(s'), \quad (14)$$

we have

$$\nabla_{p \sim \Phi_t}[V^\pi,p(s)] = \nabla_{p \sim \Phi_t} \left[ \sum_a \pi(a \mid s)r(s, a) + \gamma \sum_{a, s'} p(a, s' \mid s)V^\pi,p(s') \right] \quad (15)$$

$$= \nabla_{p \sim \Phi_t} \left[ \gamma \sum_{a, s'} p(a, s' \mid s)V^\pi,p(s') \right], \quad (16)$$

where (16) holds since $r(s, a)$ is deterministic. Using the identity $\nabla[Y] = \mathbb{E}[Y^2] - (\mathbb{E}[Y])^2$ on (16) concludes the proof. $\square$

The next result is the direct consequence of our set of assumptions.

**Lemma 2.** Under Assumptions 1 and 2, for any $s \in S$, any policy $\pi$, $\text{Cov}[p(s' \mid s, a), V^\pi,p(s')] = 0$.

**Proof.** Define $\tau$ to be any trajectory starting from state $s'$, $\tau = \{s', a_0, s_1, a_1, \ldots \}$. First, by Assumption 1, if $s_i \neq s'$ for some $i \in \{1, 2, \ldots \}$, then $p(s' \mid s, a)$ is independent of $p(s' \mid s_i, a)$. However, by Assumption 2, $s_i \neq s'$ for all $i > 0$, which implies that the trajectory distribution $P(\tau)$ is independent of the transition $p(s' \mid s, a)$. Lastly, since $V^\pi,p(s')$ is an expectation under $P(\tau)$, and independence implies zero correlation, the lemma holds. $\square$

Using the previous result yields the following lemma.

**Lemma 3.** Under Assumptions 1 and 2, it holds that

$$\sum_{a, s'} \mathbb{E}_{p \sim \Phi_t}[p(a, s' \mid s)V^\pi,p(s')] = \sum_{a, s'} \bar{p}_t(a, s' \mid s) \mathbb{E}_{p \sim \Phi_t}[V^\pi,p(s')]. \quad (17)$$

**Proof.** For any pair of random variables $X$ and $Y$ on the same probability space, by definition of covariance it holds that $\mathbb{E}[XY] = \text{Cov}[X, Y] + \mathbb{E}[X] \mathbb{E}[Y]$. Using this identity with Lemma 2 and the definition of posterior mean transition (2) yields the result. $\square$

Now we are ready to prove the main theorem.

**Theorem 1.** Under Assumptions 1 and 2, for any $s \in S$ and policy $\pi$, the posterior variance of the value function, $U^\pi_t = \nabla_{p \sim \Phi_t}[V^\pi,p]$ obeys the uncertainty Bellman equation

$$U^\pi_t(s) = \gamma^2 u_t(s) + \gamma^2 \sum_{a, s'} \pi(a \mid s)\bar{p}_t(s' \mid s, a)U^\pi_t(s'), \quad (6)$$

where $u_t(s)$ is the local uncertainty defined as

$$u_t(s) = \nabla_{a, s' \sim \pi, \bar{p}_t}[\tilde{V}^\pi_t(s')] - \mathbb{E}_{p \sim \Phi_t} \left[ \nabla_{a, s' \sim \pi, p}[V^\pi,p(s')] \right]. \quad (7)$$
Applying linearity of expectation to bring it inside the sum and an application of Lemma 3 (note that the lemma applies for
Under Assumptions 1 and 2, it holds that
Lemma 4.
Φ
distribution
p
conditioning variable
X
V

In this section, we provide the supporting theory and the proof of Theorem 2. First, we will use the identity
Proof. For the second term of the r.h.s of (13) we apply again Lemma 3 and under definition of variance
Treating the inner expectation,
Due to the conditioning, \( p(s' \mid s, a) \) is deterministic within the expectation
By Lemma 2, \( V^{π,p}(s') \) is independent of \( P_s \), so we can drop the conditioning

Lastly, since drawing samples from a marginal distribution is equivalent to drawing samples from the joint, i.e., \( \mathbb{V}_x[f(x)] = \mathbb{V}(x,y) \{ f(x) \} \), then:

\[
\mathbb{V}_{p \sim \Phi_{s,t}} \left[ \sum_{a,s'} p(a, s' \mid s) \bar{V}_t^\pi (s') \right] = \mathbb{V}_{p \sim \Phi_t} \left[ \sum_{a,s'} p(a, s' \mid s) \bar{V}_t^\pi (s') \right], 
\]

(27)

completing the proof.

The next lemma establishes the result for the expression \( \mathbb{E}[(\mathbb{E}[Y \mid X])^2] \).

**Lemma 5.** Under Assumptions 1 and 2, it holds that

\[
\mathbb{E}_{p \sim \Phi_{s,t}} \left[ \left( \mathbb{E}_{p \sim \Phi_t} \left[ \sum_{a,s'} p(a, s' \mid s) V^{\pi,p}(s') \mid P_s \right] \right)^2 \right] = \sum_{a,s'} \bar{p}_t(a, s' \mid s) (\bar{V}_t^\pi (s'))^2 - \mathbb{E}_{p \sim \Phi_t} \left[ \mathbb{V}_{a,s' \sim \pi,p} [\bar{V}_t^\pi (s')] \right]. 
\]

(28)

**Proof.** The inner expectation is equal to the one in Lemma 4, so we have that

\[
\left( \mathbb{E}_{p \sim \Phi_t} \left[ \sum_{a,s'} p(a, s' \mid s) V^{\pi,p}(s') \mid P_s \right] \right)^2 = \left( \sum_{a,s'} p(a, s' \mid s) \bar{V}_t^\pi (s') \right)^2
\]

(29)

\[
= \sum_{a,s'} p(a, s' \mid s) (\bar{V}_t^\pi (s'))^2 - \mathbb{V}_{a,s' \sim \pi,p} [\bar{V}_t^\pi (s')]. 
\]

(30)

Finally, applying expectation on both sides of (30) yields the result.

Similarly, the next lemma establishes the result for the expression \( \mathbb{E}[\mathbb{E}[Y \mid X]]^2 \).

**Lemma 6.** Under Assumptions 1 and 2, it holds that

\[
\left( \mathbb{E}_{p \sim \Phi_{s,t}} \left[ \mathbb{E}_{p \sim \Phi_t} \left[ \sum_{a,s'} p(a, s' \mid s) V^{\pi,p}(s') \mid P_s \right] \right] \right)^2 = \sum_{a,s'} \bar{p}_t(a, s' \mid s) (\bar{V}_t^\pi (s'))^2 - \mathbb{E}_{a,s' \sim \pi,p} [\bar{V}_t^\pi (s')]. 
\]

(31)

**Proof.** By the tower property of expectations, \( \mathbb{E}[\mathbb{E}[Y \mid X]]^2 = \mathbb{E}[\mathbb{E}[Y]]^2 \). Then, the result follows directly from (20) and (21).

The second part of Theorem 2 is a corollary of the next lemma.

**Lemma 7.** Under Assumptions 1 and 2, it holds that

\[
\mathbb{E}_{p \sim \Phi_t} \left[ \mathbb{V}_{a,s' \sim \pi,p} [V^{\pi,p}(s')] - \mathbb{V}_{a,s' \sim \pi,p} [\bar{V}_t^\pi (s')] \right] 
\]

is non-negative.

**Proof.** We will prove the lemma by showing (32) is equal to \( \mathbb{E}_{p \sim \Phi_t} \left[ \mathbb{V}_{a,s' \sim \pi,p} [V^{\pi,p}(s') - \bar{V}_t^\pi (s')] \right] \), which is a non-negative quantity by definition of variance. The idea is to derive two expressions for \( \mathbb{V}[\mathbb{E}[Y \mid X]] \) and compare them. First, we will use the identity \( \mathbb{E}[\mathbb{V}[Y \mid X]] = \mathbb{E}[\mathbb{E}[(Y - \mathbb{E}[Y \mid X])^2 \mid X]] \). The outer expectation is w.r.t the marginal distribution
\( \Phi_{s,t} \) while the inner expectations are w.r.t \( \Phi_t \). For the inner expectation we have

\[
\mathbb{E}_{p \sim \pi_t} \left[ \left( \sum_{a,s'} p(a, s' | s) V^\pi(p(s')) - \mathbb{E}_{p \sim \pi_t} \left[ \sum_{a,s'} p(a, s' | s) V^\pi(p(s')) \right] p_s \right)^2 \right] p_s
\]

(33)

\[
= \mathbb{E}_{p \sim \pi_t} \left[ \left( \sum_{a,s'} p(a, s' | s) \left( V^\pi(p(s')) - \mathbb{E}_{p \sim \pi_t}[V^\pi(p(s'))] \right) \right)^2 \right] p_s
\]

(34)

\[
= \mathbb{E}_{p \sim \pi_t} \left[ \sum_{a,s'} p(a, s' | s) \left( V^\pi(p(s')) - \bar{V}_t^\pi(s') \right)^2 \right] p_s
\]

(35)

\[
= \sum_{a,s'} p(a, s' | s) \mathbb{V}_{p \sim \pi_t} \left[ V^\pi(p(s')) - \bar{V}_t^\pi(s') \right] p_s
\]

(36)

Applying the outer expectation to the last equation, along with Lemma 2 and the tower property of expectations yields:

\[
\mathbb{E}[V(Y | X)] = \sum_{a,s'} \bar{p}_t(a, s' | s) \mathbb{V}_{p \sim \pi_t} \left[ V^\pi(p(s')) - \bar{V}_t^\pi(s') \right].
\]

(37)

Now we repeat the derivation but using \( \mathbb{E}[V(Y | X)] = \mathbb{E}[E[Y^2 | X]) - (\mathbb{E}[Y | X]^2) \). For the inner expectation of the first term we have:

\[
\mathbb{E}_{p \sim \pi_t} \left[ \left( \sum_{a,s'} p(a, s' | s) V^\pi(p(s')) \right)^2 \right] p_s
\]

(39)

\[
= \mathbb{E}_{p \sim \pi_t} \left[ \sum_{a,s'} p(a, s' | s) \left( V^\pi(p(s')) \right)^2 \right] p_s
\]

(40)

Applying the outer expectation:

\[
\mathbb{E}[E[Y^2 | X]] = \sum_{a,s'} \bar{p}_t(a, s' | s) \mathbb{E}_{p \sim \pi_t} \left[ \left( V^\pi(p(s')) \right)^2 \right] - \mathbb{E}_{p \sim \pi_t} \left[ \mathbb{V}_{a,s' \sim \pi_t} [V^\pi(p(s'))] \right].
\]

(41)

Lastly, for the inner expectation of \( \mathbb{E}[(\mathbb{E}[Y | X])^2] \):

\[
\left( \mathbb{E}_{p \sim \pi_t} \left[ \sum_{a,s'} p(a, s' | s) V^\pi(p(s')) \right] p_s \right)^2 = \left( \sum_{a,s'} p(a, s' | s) \bar{V}_t^\pi(s') \right)^2
\]

(42)

\[
= \sum_{a,s'} p(a, s' | s) \left( \bar{V}_t^\pi(s') \right)^2 - \mathbb{V}_{a,s' \sim \pi_t} [\bar{V}_t^\pi(s')].
\]

(43)

Applying the outer expectation:

\[
\mathbb{E}[(\mathbb{E}[Y | X])^2] = \sum_{a,s'} \bar{p}_t(a, s' | s) \left( \bar{V}_t^\pi(s') \right)^2 - \mathbb{E}_{p \sim \pi_t} \left[ \mathbb{V}_{a,s' \sim \pi_t} [\bar{V}_t^\pi(s')] \right].
\]

(44)

Finally, by properties of variance, (38) = (41) - (44) which gives the desired result.

**Theorem 2.** Under Assumptions 1 and 2, for any \( s \in S \) and policy \( \pi \), it holds that \( u_t(s) = w_t(s) - g_t(s) \), where \( g_t(s) = \mathbb{E}_{p \sim \pi_t} \left[ \mathbb{V}_{a,s' \sim \pi_t} [V^\pi(p(s'))] - \mathbb{V}_{a,s' \sim \pi_t} [\bar{V}_t^\pi(s')] \right] \). Furthermore, we have that the gap \( g_t(s) \) is non-negative, thus \( u_t(s) \leq w_t(s) \).
Proof. By definition of \( u_t(s) \) in (7), proving the claim is equivalent to showing

\[
V_{a,s' \sim \pi, \bar{\pi}} [\bar{V}^\pi_t(s')] = w_t(s) + \mathbb{E}_{p \sim \Phi_t} \left[ V_{a,s' \sim \pi, p} [\bar{V}^\pi_t(s')] \right],
\]

which holds by combining Lemmas 4–6. Lastly, \( u_t(s) \leq w_t(s) \) holds by Lemma 7. \( \square \)

B THEORY EXTENSIONS

B.1 Unknown Reward Function

We can easily extend the derivations on Appendix A.1 to include the additional uncertainty coming from an unknown reward function. Similarly, we assume the reward function is a random variable \( r \) drawn from a prior distribution \( \Psi_0 \), and whose belief will be updated via Bayes rule. In this new setting, we now consider the variance of the values under the distribution of MDPs, represented by the random variable \( M \). We need the following additional assumptions to extend our theory.

Assumption 3 (Independent rewards). \( r(x, a) \) and \( r(y, a) \) are independent random variables if \( x \neq y \).

Assumption 4 (Independent transitions and rewards). The random variables \( p(\cdot \mid s, a) \) and \( r(s, a) \) are independent for any \((s, a)\).

With Assumption 3 we have that the value function of next states is independent of the transition function and reward function at the current state. Assumption 4 means that sampling \( M \sim \Gamma_t \) is equivalent as independently sampling \( p \sim \Phi_t \) and \( r \sim \Psi_t \).

Theorem 3. Under Assumptions 1–4, for any \( s \in S \) and policy \( \pi \), the posterior variance of the value function, \( U_t^\pi = \mathbb{V}_{M \sim \Gamma_t} [V^\pi,M] \) obeys the uncertainty Bellman equation

\[
U_t^\pi(s) = \mathbb{V}_{r \sim \Psi_t} \left[ \sum_a \pi(a \mid s) r(s, a) \right] + \gamma^2 u_t(s) + \gamma^2 \sum_{a,s'} \pi(a \mid s) \bar{p}_t(s' \mid s, a) U_t^\pi(s'),
\]

where \( u_t(s) \) is defined in (7).

Proof. By Assumptions 3 and 4 and following the derivation of Lemma 1 we have

\[
\mathbb{V}_{M \sim \Gamma_t} \left[ V^\pi,M(s) \right] = \mathbb{V}_{M \sim \Gamma_t} \left[ \sum_a \pi(a \mid s) r(s, a) + \gamma \sum_{a,s'} p(a, s' \mid s) V^\pi,M(s') \right]
= \mathbb{V}_{r \sim \Psi_t} \left[ \sum_a \pi(a \mid s) r(s, a) \right] + \mathbb{V}_{M \sim \Gamma_t} \left[ \gamma \sum_{a,s'} p(a, s' \mid s) V^\pi,M(s') \right].
\]

Then following the same derivations as Appendix A.1 completes the proof. \( \square \)

B.2 Extension to Q-values

Our theoretical results naturally extend to action-value functions. The following result is analogous to Theorem 1.

Theorem 4. Under Assumptions 1 and 2, for any \((s, a) \in S \times A\) and policy \( \pi \), the posterior variance of the Q-function, \( U_t^\pi = \mathbb{V}_{p \sim \Phi_t} [Q^\pi,p] \) obeys the uncertainty Bellman equation

\[
U_t^\pi(s, a) = \gamma^2 u_t(s, a) + \gamma^2 \sum_{a',s'} \pi(a' \mid s') \bar{p}_t(s' \mid s, a) U_t^\pi(s', a'),
\]

where \( u_t(s, a) \) is the local uncertainty defined as

\[
u_t(s, a) = \mathbb{V}_{a',s' \sim \pi, \bar{\pi}} \left[ \bar{Q}^\pi_t(s', a') \right] - \mathbb{E}_{p \sim \Phi_t} \left[ \mathbb{V}_{a',s' \sim \pi, p} \left[ Q^\pi,p(s', a') \right] \right]
\]

Proof. Follows the same derivation as Appendix A.1 \( \square \)
Similarly, we can connect to the upper-bound found by Zhou et al. (2020) with the following theorem.

**Theorem 5.** Under Assumptions 1 and 2, for any \((s, a) \in S \times A\) and policy \(\pi\), it holds that

\[
u_t(s, a) = w_t(s, a) - g_t(s, a),
\]

where \(w_t(s, a) = \mathbb{V}_{p \sim \Phi_t} \left[ \sum_{a', s'} \pi(a' | s') p(s' | s, a) \bar{Q}_t(s', a') \right]\) and \(g_t(s, a) = \mathbb{E}_{p \sim \Phi_t} \left[ \mathbb{V}_{a', s' \sim \pi, p} \left[ Q^\pi(s', a') \right] - \mathbb{V}_{a', s' \sim \pi, p} \left[ \bar{Q}_t(s', a') \right] \right].\)

Furthermore, we have that the gap \(g_t(s, a) \geq 0\) is non-negative, thus \(\nu_t(s, a) \leq w_t(s, a)\).

**Proof.** Follows the same derivation as Appendix A.2. Similarly, we can prove that the gap \(g_t(s, a)\) is non-negative by showing it is equal to \(\mathbb{E}_{p \sim \Phi_t} \left[ \mathbb{V}_{a', s' \sim \pi, p} \left[ Q^\pi(s', a') \right] - \mathbb{V}_{a', s' \sim \pi, p} \left[ \bar{Q}_t(s', a') \right] \right].\)

### B.3 State-Action Uncertainty Rewards

In our practical experiments, we use the results of both Appendices B.1 and B.2 to compose the uncertainty rewards propagated via the UBE. Concretely, we consider the following two approaches for computing state-action uncertainty rewards:

- **pombu:**
  \[
u_t(s, a) = \mathbb{V}_{p \sim \Phi_t} \left[ \sum_{a', s'} \pi(a' | s') p(s' | s, a) \bar{Q}_t(s', a') \right], \tag{51}\]

- **exact-ube:**
  \[
u_t(s, a) = w_t(s, a) - \mathbb{E}_{p \sim \Phi_t} \left[ \mathbb{V}_{a', s' \sim \pi, p} \left[ Q^\pi(s', a') \right] - \mathbb{V}_{a', s' \sim \pi, p} \left[ \bar{Q}_t(s', a') \right] \right]. \tag{52}\]

Additionally, since we also learn the reward function, we add to the above the uncertainty term generated by the reward function posterior, as shown in Appendix B.1: \(\mathbb{V}_{r \sim \Psi_t} [r(s, a)]\).

### C. TABULAR ENVIRONMENTS EXPERIMENTS

In this section, we provide more details about the tabular implementation of Algorithm 1, environment details and extended results.

#### C.1 Implementation Details

**Model learning.** For the transition function we use a prior Dirichlet\((1/\sqrt{S})\) and for rewards a standard normal \(\mathcal{N}(0, 1)\), as done by O’Donoghue et al. (2019). The choice of priors leads to closed-form posterior updates based on state-visitation counts and accumulated rewards. We add a terminal state to our modeled MDP in order to compute the values in closed-form via linear algebra.

**Accelerating learning.** For the DeepSea benchmark we accelerate learning by imagining each experienced transition \((s, a, s', r)\) is repeated \(L\) times, as initially suggested in Osband et al. (2019) (see footnote 9), although we scale the number of repeats with the size of the MDP. Effectively, this strategy forces the MDP posterior to shrink faster, thus making all algorithms converge in fewer episodes. The same strategy was used for all the methods evaluated in the benchmark.

**Policy optimization.** All tested algorithms (PSRL and OFU variants) optimize the policy via policy iteration, where we break ties at random when computing the argmax, and limit the number of policy iteration steps to 40.

**Hyperparameters.** Unless noted otherwise, all tabular RL experiments use a discount factor \(\gamma = 0.99\), an exploration gain \(\lambda = 1.0\) and an ensemble size \(N = 5\).

**Uncertainty reward clipping.** For DeepSea we clip uncertainty rewards with \(u_{\min} = -0.05\) and for the 7-room environment we keep \(u_{\min} = 0.0\).

#### C.2 Environment Details

**DeepSea.** As proposed by Osband et al. (2019), DeepSea is a grid-world environment of size \(L \times L\), with \(S = L^2\) and \(A = 2\).
Figure 7: Extended results for the DeepSea experiments shown in Figure 2. We report the average (solid line) and standard error (shaded region) over 5 random seeds.

7-room. As implemented by Domingues et al. (2021), the 7-room environment consists of seven connected rooms of size $5 \times 5$, represented as an MDP of size $S = 181$ and discrete action space with size $A = 4$. The starting state is always the center cell of the middle room, which yields a reward of 0.01. The center cell of the left-most room gives a reward of 0.1 and the center cell of the right-most room gives a large reward of 1. The episode terminates after 40 steps and the state with large reward is absorbing (i.e., once it reaches the rewarding state, the agent remains there until the end of the episode). The agent transitions according to the selected action with probability 0.95 and moves to a randomly selected neighboring cell with probability 0.05.

C.3 DeepSea Additional Experiments

C.3.1 Extended Results

Figure 7 shows the total regret in intervals of 50 episodes for all the different DeepSea sizes considered. Our method consistently achieves the lowest total regret.

C.3.2 Uncertainty Rewards Ablation

Our theory prescribes equivalent expressions for the uncertainty rewards under the assumptions. However, since it practice the assumptions do not generally hold, the expressions are no longer equivalent. In this section we evaluate the performance in the DeepSea benchmark for these different definitions of the uncertainty rewards:
Model-Based Uncertainty in Value Functions

Figure 8: Ablation study on DeepSea exploration for different estimates of exact-ube. Results represent the average over 5 random seeds, and vertical bars on total regret indicate the standard error.

- **exact-ube_1**: 
  \[ u_t(s,a) = \mathbb{V}_{a',s' \sim \pi, \bar{p}} \left[ \bar{Q}_{t}^r(s', a') \right] - \mathbb{E}_{p \sim \Phi_t} \left[ \mathbb{V}_{a',s' \sim \pi, p} \left[ Q^p(s', a') \right] - V_{a',s' \sim \pi, p} \left[ \bar{Q}_{t}^{p}(s', a') \right] \right] \]

- **exact-ube_2**: 
  \[ u_t(s,a) = \mathbb{V}_{p \sim \Phi_t} \left[ \sum_{a',s'} \pi(a' | s') p(s' | s,a) \bar{Q}_t(s', a') \right] - \mathbb{E}_{p \sim \Phi_t} \left[ \mathbb{V}_{a',s' \sim \pi, p} \left[ Q^p(s', a') \right] - V_{a',s' \sim \pi, p} \left[ \bar{Q}_{t}^{p}(s', a') \right] \right] \]

- **exact-ube_3** (labeled exact-ube in all other plots): 
  \[ u_t(s,a) = \mathbb{V}_{p \sim \Phi_t} \left[ \sum_{a',s'} \pi(a' | s') p(s' | s,a) \bar{Q}_t(s', a') \right] - \mathbb{E}_{p \sim \Phi_t} \left[ \mathbb{V}_{a',s' \sim \pi, p} \left[ Q^p(s', a') \right] - V_{a',s' \sim \pi, p} \left[ \bar{Q}_{t}^{p}(s', a') \right] \right] \]

Recall that, since we consider an unknown reward function, we add the uncertainty about rewards to the above when solving the UBE. Figure 8 shows the results for the DeepSea benchmark comparing the three uncertainty signals. Since the assumptions are violated in the practical setting, the three signals are no longer equivalent and result in slightly different uncertainty rewards. Still, when integrated into Algorithm 1, the performance in terms of learning time and total regret is quite similar. We select **exact-ube_3** as the default estimate for all other experiments.

C.3.3 Ensemble Size Ablation

The ensemble size \( N \) is one important hyperparameter for all the OFU-based methods. We perform additional experiments in DeepSea for different values of \( N \), keeping all other hyperparameters fixed and with sizes \( L = \{20, 30\} \). The results in Figure 9 show that our method achieves lower total regret across the different ensemble sizes. For ensemble-var, performance increases for larger ensembles. These results suggest that the sample-based approximation of our uncertainty rewards is not very sensitive to the number of samples and achieve good performance even for \( N = 2 \).

C.3.4 Exploration Gain Ablation

Another important hyperparameter for OFU-based methods is the exploration gain \( \lambda \), controlling the magnitude of the optimistic values optimized via policy iteration. We perform an ablation study over \( \lambda \), keeping all other hyperparameters fixed and testing for DeepSea sizes \( L = \{20, 30\} \). Figure 10 shows the total regret for OFU methods over increasing gain. Unsurprisingly, as we increase \( \lambda \), the total regret of all the methods increases, but overall **exact-ube** achieves the best performance.

D CONTINOUS CONTROL EXPERIMENTS

In this section, we provide details regarding the deep RL implementation of the optimistic, variance-driven policy optimization. Also, we include relevant hyperparameters, environment details and additional results.
D.1 Implementation Details

The optimistic approach on top of MBPO (Janner et al., 2019) is presented in Algorithm 2. The main differences with the original implementation are as follows:

- In Line 8, we perform a total of \( N + 1 \) \( k \)-step rollouts corresponding to both the model-randomized and model-consistent rollout modalities. The original MBPO only executes the former to fill up \( D_{\text{model}} \).

- In Line 11, we update the ensemble of \( Q \)-functions on the corresponding model-consistent buffer. MBPO trains twin critics (as in SAC) on mini-batches from \( D_{\text{model}} \).

- In Line 12, we update the \( U \)-net for the UBE-based variance estimation methods.

- In Line 13, we update \( \pi_\phi \) by maximizing the optimistic \( Q \)-values. MBPO maximizes the minimum of the twin critics (as in SAC). Both approaches include an entropy maximization term.

The main hyperparameters for our experiments are included in Table 2. Further implementation details are now provided.
Algorithm 2 MBPO-style optimistic learning

1: Initialize policy $\pi_\phi$, predictive model $p_\theta$, critic ensemble $\{Q_i\}_{i=1}^N$, uncertainty net $U_\psi$ (optional), environment dataset $D_t$, model datasets $D_{\text{model}}$ and $\{D^i_{\text{model}}\}_{i=1}^N$.
2: global step $\leftarrow 0$
3: for episode $t = 0, \ldots, T - 1$ do
4:   for $E$ steps do
5:     if global step $\% F == 0$ then
6:       Train model $p_\theta$ on $D_t$ via maximum likelihood
7:     for $M$ model rollouts do
8:       Perform $k$-step model rollouts starting from $s \sim D_t$; add to $D_{\text{model}}$ and $\{D^i_{\text{model}}\}_{i=1}^N$
9:     Take action in environment according to $\pi_\phi$; add to $D_t$
10:    for $G$ gradient updates do
11:       Update $\{Q_i\}_{i=1}^N$ with mini-batches from $\{D^i_{\text{model}}\}_{i=1}^N$, via SGD on (10)
12:       (Optional) Update $U_\psi$ with mini-batches from $D_{\text{model}}$, via SGD on (12)
13:       Update $\pi_\phi$ with mini-batches from $D_{\text{model}}$, via stochastic gradient ascent on the optimistic values of (9)
14:   global step $\leftarrow$ global step + 1

Table 2: Hyperparameter settings for continuous control experiments.

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Sparse Pendulum</th>
<th>HalfCheetah</th>
<th>Walker2D</th>
<th>Ant</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T$ - # episodes</td>
<td>75</td>
<td>200</td>
<td>300</td>
<td></td>
</tr>
<tr>
<td>$E$ - # steps per episode</td>
<td>400</td>
<td>1000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$G$ - policy updates per step</td>
<td>20</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M$ - # model rollouts per step</td>
<td>400</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$F$ - frequency of model retraining (# steps)</td>
<td>400</td>
<td>250</td>
<td></td>
<td></td>
</tr>
<tr>
<td>retain updates</td>
<td>1</td>
<td></td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>$N$ - ensemble size</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\lambda$ - exploration gain</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\lambda_{\text{reg}}$ - UBE regularization gain</td>
<td>5.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$k$ - rollout length</td>
<td>10</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Model network</td>
<td>4 layers, 200 units, SiLU activations</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Policy network</td>
<td>2 layers, 64 units, Tanh activation</td>
<td>2 layers, 128 units, Tanh activations</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q$ and $U$ networks</td>
<td>2 layers, 256 units, Tanh activations</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Model learning. We leverage the mbrl-lib Python library from Pineda et al. (2021) and train an ensemble of $N$ probabilistic neural networks. We use the default MLP architecture with four layers of size 200 and SiLU activations. The networks predict delta states, $\Delta = s' - s$, and receive as input normalized state-action pairs. The normalization statistics are updated each time we train the model, and are based on the dataset $D_t$. We use the default initialization of the network provided by the library, which samples weights from a truncated Gaussian distribution, however we found it helpful to increase by a factor of 2.0 the standard deviation of the truncated Gaussian for the sparse pendulum task; a wider distribution of weights allows for more diverse dynamic models at the beginning of training and thus a stronger uncertainty signal to guide exploration.

Model-generated buffers. The capacity of the model-generated buffers $D_{\text{model}}$ and $\{D^i_{\text{model}}\}_{i=1}^N$ is computed as $k \times M \times F \times \text{retain updates}$, where $\text{retain updates}$ is the number of model updates before entirely overwriting the buffers. Larger values of this parameter allows for more off-policy (old) data to be stored and sampled for training.
Uncertainty reward estimation. We estimate the uncertainty rewards (51) and (52) using a finite-sample approximation. For \( w_t(s, a) \), the inner expectation is estimated using a single action \( a' \sim \pi(\cdot | s') \), where we take \( s' \) to be the mean of the Gaussian distribution parameterized by each ensemble member. For the gap term in \( u_t(s, a) \), we sample 10 actions from the current policy to estimate the aleatoric variance term \( V_{a', s'}[\cdot] \). We clip the uncertainty rewards with \( u_{\text{min}} = 0.0 \).

SAC specifics. Our SAC implementation is based on the open-source repository https://github.com/pranz24/pytorch-soft-actor-critic, as done by mbrl-lib. For all our experiments, we use the automatic entropy tuning flag that adaptively modifies the entropy gain \( \alpha \) based on the stochasticity of the policy.

D.2 Environment Details

Sparse Pendulum. The implementation is taken from https://github.com/sebascuri/hucrl/blob/4b4446e54a7269366eeafabd90f91fbe466d8b15/exps/inverted_pendulum/util.py and adapted to the OpenAI Gym (Brockman et al., 2016) convention for RL environments. We use an action cost multiplier \( \rho = 0.2 \) for all our experiments.

Pybullet environments. We use the default Pybullet locomotion environments but remove the observations related to feet contact, which are represented as binary variables, as these can pose challenges to model learning.

D.3 Ensemble Size Ablation

We repeat the experiments for the sparse pendulum task for different ensemble sizes, and summarize the results in Figure 11. In most cases, performance increases with \( N \), although there exists some outliers. In some specific cases, we observed larger ensembles could be detrimental to learning with sparse rewards: if most members of the ensemble converge to similar values then the policy might prematurely converge to a suboptimal policy. We believe network initialization and regularization may play a critical role in maintaining sufficient ensemble diversity to drive exploration in sparse reward settings.

D.4 Visualization of Variance Estimates

In this section, we visualize the evolution of the value function and variance estimates during training in the sparse pendulum problem using optimistic values estimated with the exact-ube method. In Figure 12, we plot the mean \( Q \)-values and the standard deviations corresponding to the exact-ube and ensemble-var estimates. While both exact-ube and ensemble-var have higher variance in regions of interest for exploration, the latter outputs much larger estimates, which may lead to over-exploration.
Figure 12: Visualization of training in sparse pendulum swing-up task using optimistic values estimated with the \textit{exact-ube} method. (Left column) The mean values correspond to $\bar{Q}_\pi^T(s, \bar{a})$, where $\bar{a}$ is the mean of the Gaussian policy $\pi$ at the corresponding episode. (Center and right columns) The posterior standard deviation of $Q$-values, computed as $\sqrt{\hat{U}_\pi^T(s, \bar{a})}$ for the \textit{exact-ube} and \textit{ensemble-var} variance estimates.