Offline RL Policies Should be Trained to be Adaptive

Dibya Ghosh [1] **Anurag Ajay** [2] **Pulkit Agrawal** [2] **Sergey Levine** [1]

Abstract

Offline RL algorithms must account for the fact that the dataset they are provided may leave many facets of the environment unknown. The most common way to approach this challenge is to employ pessimistic or conservative methods, which avoid behaviors that are too dissimilar from those in the training dataset. However, relying exclusively on conservatism has drawbacks: performance is sensitive to the exact degree of conservatism, and conservative objectives can recover highly suboptimal policies. In this work, we propose that offline RL methods should instead be *adaptive* in the presence of uncertainty. We show that acting optimally in offline RL in a Bayesian sense involves solving an implicit POMDP. As a result, optimal policies for offline RL must be adaptive, depending not just on the current state but rather all the transitions seen so far during evaluation. We present a model-free algorithm for approximating this optimal adaptive policy, and demonstrate the efficacy of learning such adaptive policies in offline RL benchmarks.

1. Introduction

Uncertainty about the environment is paramount in offline reinforcement learning (RL), the task of learning control behaviors from a dataset of logged environment interactions. Unlike in traditional RL, where the agent learns about all aspects of the environment through online interaction during training, offline RL methods must rely only on a fixed dataset, which often precludes the agent from identifying the exact dynamics of the environment.

The most common approach for handling the resulting uncertainty is to learn conservative state-based policies incentivized to stay close to dataset behaviors. While these methods have been empirically successful with careful hyperparameter choices, it remains unclear whether conservative objectives are, in general, the best approach for designing offline RL algorithms. To formally understand this, we study the problem of offline RL under a Bayesian perspective. We show that when learning from an offline dataset of experience that does not fully specify the environment, uncertainty manifests as an *implicit partial observability* in the original fully-observable learning problem. This implicit partial observability leads to a surprising consequence: a static state-based policy in offline RL, no matter how conservative, can be arbitrarily sub-optimal for maximizing test-time return, and in general, offline RL agents must *adapt* to changes in the agent's uncertainty to act optimally.

To understand where adaptation can help, consider what happens if an offline RL policy makes a mistake at evaluation time, e.g. taking an action that the agent incorrectly believes to lead to a high-value state. Since the agent observes transitions from the environment in RL, it sees that the action doesn't lead to a high-value state, and has the ability to correct this behavior instead of blindly continuing the original learned strategy. In general, there exist a breadth of strategies for changing behaviors based on witnessed environment transitions that are untapped by contemporary offline RL algorithms, which learn static policies using objectives not designed to incentivize adaptation.

The Bayesian perspective defines an optimal adaptive offline RL policy, which is the solution to a POMDP implicitly induced by the offline dataset (Duff & Barto, 2002; Ghosh et al., 2021). Unfortunately, this optimal policy cannot be easily obtained using standard model-free POMDP algorithms, since the POMDP is only an *implicit construction* defined by the agent's epistemic uncertainty. Rather, we derive an approximate approach to solve the POMDP that uses an ensemble of value functions to estimate the implicit partial observability, and learns an adaptive policy using this ensemble. By optimizing this adaptive policy with the implicit POMDP objective, the policy learns to adapt in a way that helps the agent maximize evaluation performance in the true environment.

The primary contribution of our work is to formally demonstrate the necessity of adaptation in offline RL, and to provide a practical algorithm for learning optimally adaptive policies. We use the Bayesian formalism to show why

[1]UC Berkeley [2]MIT. Correspondence to: Dibya Ghosh <dibya@berkeley.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

adaptability is necessary, and how a policy must adapt to optimally maximize offline RL performance. As a first step towards approximating Bayes-optimal adaptiveness, we propose an ensemble-based offline RL algorithm that imbues policies with the ability to adapt within an episode, and demonstrate its favorable properties in offline RL domains.

2. Problem Setup

The goal in reinforcement learning is to maximize an agent's expected return in a Markov decision process (MDP), written as $\mathcal{M} := (\mathcal{S}, \mathcal{A}, r, T, \rho, \gamma)$. An MDP is specified by a state space $\mathcal{S}$, action space $\mathcal{A}$, Markovian transition kernel $T(s'|s,a)$, reward function $r(s,a)$, initial state distribution $\rho(s_0)$, and discount factor γ. A history is a sequence of witnessed transitions in an MDP $h_t = (s_0, a_0, r_0, s_1, a_1, \ldots, s_t)$; we write $s(h_t) := s_t$ to denote the current state. A policy π maps a history to a distribution over actions, and achieves return in the MDP $J_{\mathcal{M}}(\pi) = \mathbb{E}_{\pi}[\sum_{t \geq 0} \gamma^t r(s_t, a_t)]$. The maximal return in an MDP can be attained by a Markovian policy (one that depends on h_t only through s_t) (Puterman, 1994).

The value function of a policy π is the expected return of the policy having taken an action a after witnessing a history h, $Q^{\pi}_{\mathcal{M}}(h_t, a_t) = \mathbb{E}_{\pi}[\sum_{t'=0}^{\infty} \gamma^{t'} r(s_{t+t'}, a_{t+t'})|h, a]$. The gradient of $J_{\mathcal{M}}(\pi_\theta)$ can be written using Q^{π}:

$$\nabla_\theta J_{\mathcal{M}}(\pi_\theta) = \mathbb{E}_{h \sim \pi}[\nabla_\theta \mathbb{E}_{a \sim \pi_\theta(\cdot|h)}[Q^{\pi}_{\mathcal{M}}(h, a)]]. \quad (1)$$

This expression serves as the basis for the policy objective in many standard actor-critic algorithms for MDPs. While these updates are traditionally written for Markovian policies $\pi_\theta(a|s_t)$, we write the more general form using history-based policies $\pi_\theta(a|h_t)$ since our paper studies the use of history-based policies in an MDP when doing offline RL.

In offline RL, the agent receives a dataset of trajectories $\mathcal{D} = \{\tau_i\}_{i=1}^n$ pre-collected by a behavior policy π_β from some MDP $\mathcal{M}^*$, and must use this dataset to learn a policy that performs well in this MDP. The performance of an offline RL policy is the expected return it achieves in the true MDP: $J_{\mathcal{M}^*}(\pi)$. In this paper, we will show that even though there exist optimal Markov policies in an MDP, when learning from an offline dataset, history-based policies $\pi_\theta(a|h_t)$ can play a role in improving performance. We refer to history-based policies as *adaptive* policies interchangeably throughout the text.

3. When History is Useful for Offline RL

Before formally studying optimality in offline RL, we illustrate example offline RL problems where adaptive policies achieve higher return than pessimistic Markov policies. In the first example, pessimism will provide a sub-optimal solution that can be improved with adaptation; in the second, pessimism will completely fail to solve the problem, making adaptation necessary to achieve high performance.

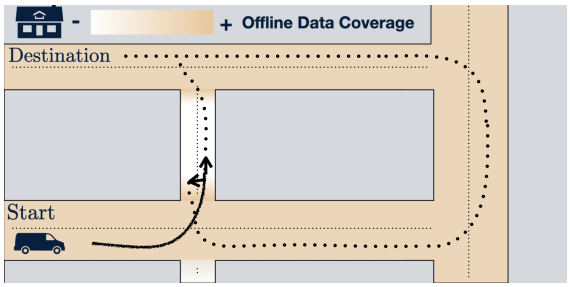

Figure 1. The *City navigation* example, where there is low data coverage on the small side street. An adaptive policy, which first tries the side street and reverts to the large road if unknown circumstances arise, will outperform Markovian alternatives, such as the conservative solution that always takes the larger road.

City navigation: Suppose we wish to learn to navigate in a city to a destination as quickly as possible from an offline dataset of trajectories where data is mostly concentrated on big city roads and highways, but a few trajectories navigate smaller side streets. Given the dataset, the agent has less uncertainty about paths relying on the big roads, but there exist shorter paths using side streets for which the agent has high uncertainty (visualized in Figure 1). An algorithm that trains an RL algorithm using this dataset as a replay buffer would take the side streets, since empirically this leads to the goal fastest. This has a potentially high downside: if the agent encounters unforeseen conditions on the side streets, it may be unable to reach the goal by a reasonable time.

A pessimistic offline RL algorithm sidesteps this failure mode by penalizing the side streets for having high uncertainty, since they have low coverage in the dataset, and learns to navigate only on big roads. While robust, this penalty prevents capitalization on the upside of the side street potentially leading to the destination faster. If we step beyond Markovian policies, we can learn adaptive strategies that can capitalize on this upside without the failure mode of standard RL: for example by starting along the side-street, and doubling back to the main road if it witnesses unexpected conditions along the way. Adaptation improves over a pessimistic solution because at test-time, the environment-agent feedback loop can signal whether the risky behavior (taking the side street) may be successful, enabling revision and consequent improvement within an evaluation episode.

Locked doors: Consider an environment where an agent is asked to exit a room with four doors, one unlocked and three locked. Each episode, a different door is left unlocked, and the agent receives an image whose label indicates the unlocked door (hence the environment is fully observable). Given access to a limited set of training images in the offline dataset, offline RL must learn a policy that can parse a new image at evaluation and go through the correct door.

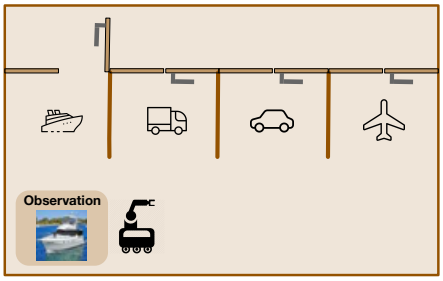

Figure 2. The *locked doors* task with CIFAR-10 images. Adaptive policies, which can change which door to try based on environment feedback, will outperform Markovian policies, which by design, continue trying the same incorrect door for the entire episode.

Any standard offline RL method (including pessimistic approaches) will learn a Markovian policy that goes towards the door that the agent believes has the highest chance of being unlocked, but such strategies are sub-optimal for this setting. When such a policy encounters an image for which its original prediction is incorrect, the agent will try to open a locked door. Since the agent's policy does not change and the agent remains in the same state, it will simply attempt the same action again, and become stuck perpetually trying to open the wrong door. The only way to avoid getting stuck is to try a different door, whether by policy adaptation or some stochastic posterior sampling scheme.

In both examples, the offline dataset left uncertainty about how to act optimally, and adaptability allowed the agent to outperform than the fixed strategy learned by pessimism. In the next section, we will study optimality in offline RL to formalize this intuition, towards understanding why adaptability can improve the performance of offline RL agents.

4. Adaptive Policies are Optimal for Offline RL

We now analyze offline RL from a Bayesian perspective, to examine *why* adaptive behavior might be useful in offline RL, and *how* an offline RL agent should adapt to be maximally performant.

During training, an offline RL agent only receives information about the true environment $\mathcal{M}^*$ via the dataset $\mathcal{D}$. In general, this dataset does not uniquely identify $\mathcal{M}^*$: if the data-collection policy only visits a limited subset of the state space or only takes certain actions, there may be many potential MDPs that behave identically on the states and actions in the dataset, but differ in their transitions and rewards on out-of-sample states and actions. As a result, the dataset induces epistemic uncertainty about the identity of the MDP. Formally, the dataset $\mathcal{D}$ and a prior distribution over MDPs $P(\mathcal{M})$ define a *posterior distribution* over MDPs, $P(\mathcal{M}|\mathcal{D}) \propto P(\mathcal{M})P(\mathcal{D}|\mathcal{M})$.

After training, the policy learned via offline RL is deployed into the true MDP, and must act to maximize return in this environment. The Bayesian objective for policy learning under this model is to maximize return in expectation over MDPs from this posterior distribution,

$$J_{\text{Bayes}}(\pi) := \mathbb{E}_{\mathcal{M}\sim P(\mathcal{M}|\mathcal{D})}[J_{\mathcal{M}}(\pi)], \tag{2}$$

since this objective defines the agent's expected performance in the true environment under its prior beliefs. As is common in Bayesian treatments of RL (Ghavamzadeh et al., 2015), we call Equation 2 the Bayesian offline RL objective, and define its maximizers as *Bayes-optimal policies* for offline RL.

To shed light on what maximizing the Bayesian objective requires, we note that Equation 2 corresponds to the expected return of a policy in a partially observable environment defined by the agent's epistemic uncertainty called the **epistemic POMDP**, following a construction in a Bayesian treatment of generalization in RL (Ghosh et al., 2021).

The epistemic POMDP $\mathcal{M}_{po}$ may be described as follows: in each episode, the agent is placed in a new MDP $\mathcal{M} \sim P(\mathcal{M}|\mathcal{D})$ and asked to maximize reward, but is not told the identity of this MDP. As the MDP $\mathcal{M}$ dictates the environment dynamics for the agent, but its identity omitted from the observation vector, the agent faces a partially observable problem. This construction mirrors that of the epistemic POMDP in Ghosh et al. (2021) and Bayes-adaptive MDPs (Duff & Barto, 2002), so we leave the complete technical specification for Appendix A. Reflecting on the city navigation example, we might imagine that the posterior distribution consists of two MDPs, one where the side street has low traffic and one where it has high traffic. Acting optimally without knowing which of these MDPs the agent was placed in requires adaptation, since midway through an episode, the agent will learn whether the side street has high or low traffic and thereby resolve its partial observability.

We can use the epistemic POMDP perspective on Equation 2 to show that acting Bayes-optimally for offline RL in an MDP *requires* adaptation, and that in general, pessimistic Markov policies may do an arbitrarily poor job of optimizing the offline RL objective.

Theorem 4.1 (informal)**.** *The optimal solution to the Bayesian offline RL objective is, in general, adaptive. Further, there are offline RL problem instances* $(\mathcal{D}, p(\mathcal{M}))$ *where the best Markovian policy far underperforms the optimal adaptive solution.*

These claims are formally stated and proven in Appendix A, but we provide intuition here. Since Equation 2 can be interpreted as the expected return in a POMDP, Bayes-optimality for offline RL is equivalent to acting optimally under partial observability, which necessitates adaptivity (Monahan, 2007). This partial observability also explains why pessimism cannot by itself lead to optimal performance for

offline RL. An agent that cannot change its behavior on receiving new information that resolves its partial observability, even if pessimistic, will be unable to keep up with an adaptive agent that reacts and adapts to this signal.

It is important to note that the epistemic POMDP is not a physical environment that the agent will explicitly ever act in, but rather only an interpretation of the Bayesian offline RL objective. That is, the agent never explicitly encounters partial observability (it always acts in an MDP); rather, we are using the parlance of partial observability to describe how uncertainty induced by the offline dataset affects policy learning and evaluation process under a Bayesian viewpoint.

The POMDP reinterpretation of the Bayesian offline RL objective provides concrete benefits for designing offline RL algorithms. First, it demonstrates the necessity of adaptation in offline RL, since non-Markovianity is well-known to be required for handling partial observability. It also provides post-facto justification for stochastic policies and stochastic regularization, which are common algorithmic design choices in offline RL (Fujimoto et al., 2019; Kostrikov et al., 2021a), since stochastic policies are better than deterministic counterparts under partial observability (Eysenbach & Levine, 2019; Singh et al., 1994). We will see in the next section that although it may be infeasible to directly apply POMDP algorithms to learn Bayes-optimal offline RL policies, they can nonetheless serve as inspiration for designing adaptive offline RL methods.

5. Optimizing for Adaptation in Offline RL

Our analysis establishes the need for adaptive policies in offline RL, and prescribes performance in the epistemic POMDP (Equation 2) as an objective to learn optimal test-time adaptation mechanisms. Unfortunately, the straightforward approach to learning such policies, explicitly modeling the epistemic POMDP and running a generic POMDP algorithm, is unscalable: it requires modeling the posterior distribution over environment dynamics and reward $P(\mathcal{M}|\mathcal{D})$, which cannot be obtained with high fidelity in most applications of interest.

We derive a policy update that avoids the burden of learning a posterior over environment dynamics by instead modeling the simpler posterior over value functions $P(Q^\pi_\mathcal{M}|\mathcal{D})$, using the fact that the value function $Q^\pi_\mathcal{M}$ entangles the necessary information about both dynamics and rewards for a given policy. This reduces the burden of uncertainty estimation, as we no longer need to learn an uncertainty-aware dynamics model, and also allows us to leverage recent progress in learning value function posteriors in offline RL (An et al., 2021; Ghasemipour et al., 2022).In this section, we define a suitable class of adaptive policies and derive an exact policy update for the objective, which will serve as the backbone for the practical actor-critic algorithm that we will introduce in Section 6.

5.1. Uncertainty-Adaptive Policies

In the previous section, we showed the sub-optimality of Markovian policies $\pi(\cdot|s)$ for offline RL since they lack the ability to change behavior on receiving new information that resolves its epistemic uncertainty. The following proposition characterizes precisely what Markovian policies are missing: a measure of how the agent's uncertainty has changed since the beginning of the episode.

Definition 5.1. The relative MDP belief $\boldsymbol{b}(h)$ for a history h is the relative change in the posterior distribution over MDPs after incorporating the history:

$$\boldsymbol{b}(h)(\mathcal{M}) = \frac{P(\mathcal{M}|h, \mathcal{D})}{P(\mathcal{M}|\mathcal{D})}. \tag{3}$$

Proposition 5.1. *The Bayesian offline RL objective is maximized by a policy depending only on the current state and relative MDP belief:* $\pi(\cdot|h_t) = \pi(\cdot|s_t, \boldsymbol{b}(h_t))$.

The relative MDP belief $\boldsymbol{b}(h)$ should be interpreted as a *weighting* that indicates which of the original hypotheses generated during offline training are consistent with the trajectory seen so far during evaluation. In the epistemic POMDP, it serves as the belief state, serving as a compressed statistic of all the information in the history before the current state that is relevant to maximizing future return.

This proposition implies that when optimizing the offline RL objective, it is sufficient to consider only **uncertainty-adaptive** policies $\pi(\cdot|s, \boldsymbol{b})$, those that conditions their behavior not only on the current state (as a Markovian policy does), but also on a sufficient statistic of the relative MDP belief. The simplicity of uncertainty-adaptive policies is particularly appealing: in structure, it is a Markovian policy that takes in an additional input $\boldsymbol{b}$, but because this new input $\boldsymbol{b}$ changes throughout the episode, the policy is no longer static but rather adaptive with history. Another useful property of uncertainty-adaptive policies is that they are allowed to adapt and change behavior only if the agent's epistemic uncertainty about the environment changes.

5.2. The Bayesian offline RL policy gradient

With an appropriate class of adaptive policies, we turn to optimizing the Bayesian objective. We consider a standard first-order optimization approach, whereby a parametric stochastic uncertainty-adaptive policy $\pi_\theta(\cdot|s, \boldsymbol{b})$ updates the parameters θ according to the gradient of the Bayesian objective (equivalently, the epistemic POMDP objective). The following proposition shows that this policy gradient can be written in terms of the posterior distribution of the current policy's value function.

Proposition 5.2. *The gradient of the Bayesian offline RL objective for an uncertainty-adaptive policy $\pi_\theta(\cdot|s, \boldsymbol{b})$ can be written in terms of the MDP value functions $Q^\pi_\mathcal{M}$ from the posterior distribution:*

$$\nabla_\theta J_{Bayes}(\pi_\theta) = \mathbb{E}_{h\sim\pi}[\nabla_\theta \mathbb{E}_{a\sim\pi_\theta(\cdot|s(h),\boldsymbol{b}(h))}[\mathbb{E}_{\mathcal{M}\sim P(\mathcal{M}|\mathcal{D})}[\boldsymbol{b}(h)(\mathcal{M}) Q^\pi_\mathcal{M}(h, a)]]. \quad (4)$$

It is instructive to compare the policy gradient of the Bayesian offline RL objective to a standard MDP policy gradient (Equation 1), accented in color above. First, since the exact MDP is not known from the dataset (only a posterior), we must average the value functions across the induced posterior distribution $P(\mathcal{M}|\mathcal{D})$. Second, since the agent's uncertainty changes during an episode, this average is re-weighted by $\boldsymbol{b}(h)(\mathcal{M})$ to account for how the agent's uncertainty has changed from the original dataset posterior after witnessing history h. To complete the specification of the policy update, we describe the value function for an uncertainty-adaptive policy π in an MDP $\mathcal{M}$.

Proposition 5.3. *The value function for a uncertainty-adaptive policy in an MDP $Q^\pi_\mathcal{M}(h, a)$ depends on h_t only through $(s_t, \boldsymbol{b}(h_t))$ and satisfies the following Bellman recursion:*

$$Q^\pi_\mathcal{M}(s, \boldsymbol{b}, a) = r(s, a) + \gamma \mathbb{E}_{\substack{s'\sim\mathcal{M} \\ a\sim\pi}} [Q^\pi_\mathcal{M}(s', \boldsymbol{b}', a)] \quad (5)$$

where $\boldsymbol{b}' \coloneqq \text{BeliefUpdate}(\boldsymbol{b}, (s, a, r, s))$ is the new relative MDP belief after witnessing (s, a, r, s'),

$$\text{BeliefUpdate}(\boldsymbol{b}, (s, a, r, s))(\mathcal{M}) \propto p_\mathcal{M}(r, s'|s, a)\boldsymbol{b}(\mathcal{M}) \quad (6)$$

The value function for an uncertainty-adaptive policy is similar to that of a Markovian policy, the main change being the need to take into account how the relative MDP belief changes over time ($\boldsymbol{b}$ on the left, $\boldsymbol{b}'$ on the right). Since the value function understands how values change when the agent's uncertainty changes and the policy adapts, optimizing the policy using these value functions provides the learning signal for the policy to adapt in a way most conducive to maximizing test-time return.

Together, the propositions describe policy and value function learning rules needed to learn Bayes-optimal uncertainty-adaptive policies for offline RL. Holistically, three main characteristics distinguishes this Bayesian offline RL policy update from policy optimization in a known MDP: 1) we must learn a posterior distribution over value functions instead of a single value function, 2) the policy conditioned on a belief $\boldsymbol{b}$ must maximize the $\boldsymbol{b}$-reweighted average across the value function posterior, and 3) the value function posterior must be trained to account for how the agent's uncertainty changes when a new transition is seen.

6. Practical Algorithm

In order to practically implement the policy update dictated in the previous section, we must approximate the posterior distribution over value functions for our current policy, and choose a representation for the relative MDP belief to pass into our adaptive policy. This section tackles these issues; the result is an actor-critic algorithm, APE-V, for learning adaptive policies for offline RL in an MDP.

6.1. A finite approximation to the posterior

We choose to approximate the posterior distribution over value functions given the offline dataset $P(Q^\pi_\mathcal{M}|\mathcal{D})$ with a finite ensemble of value functions $\{\hat{Q}_1, \dots \hat{Q}_n\}$, where $\hat{Q}_k$ is the value function of π in $\mathcal{M}_k \sim p(\mathcal{M}|\mathcal{D})$. For this finite posterior, the relative MDP belief $\boldsymbol{b}$ is a probability distribution over the n MDPs corresponding to our value function ensemble, starting as a uniform distribution $\boldsymbol{b}_0 = [\frac{1}{n} \dots \frac{1}{n}]^\top$ at the beginning of the episode, and skewing towards the MDP most consistent with the witnessed trajectory as the episode continues. In the remainder of this section, we will write $\boldsymbol{b} \in \mathbb{R}^n$ on the n-dimensional probability simplex, with $\boldsymbol{b}_k \coloneqq \boldsymbol{b}(\mathcal{M}_k)$.

6.2. Training value functions

We now discuss how each value function $\hat{Q}_k$ in the posterior should be learned. As described by Proposition 5.3, learning the value function for a uncertainty-adaptive policy $\pi(a|s, \boldsymbol{b})$ requires two changes from that for Markov policies: 1) the value function must take in both the state s and relative belief $\boldsymbol{b}$, since the policy π depends on both; 2) the Bellman consistency equation must incorporate how the relative belief shifts $\boldsymbol{b} \to \boldsymbol{b}'$ when the new transition (s, a, r, s') is witnessed.

Ensuring that the value function accounts for how new transition (s, a, r, s') changes the relative belief is what provides the policy the signal for adjusting its adaptation mechanism towards higher performance. One problem with implementing this scheme is that $\text{BeliefUpdate}(\boldsymbol{b}, (s, a, r, s))$ depends on transition log-likelihoods $\log P_\mathcal{M}(s', r|s, a)$, which we do not model, and so we must replace it with an approximation. Recognizing that $-\log P_\mathcal{M}(s', r|s, a)$ represents the information-theoretic surprise of witnessing the transition (s, a, r, s') in $\mathcal{M}$, we choose to replace it with a surrogate surprise $\log \hat{P}_\mathcal{M}(\cdot)$ defined by our value model:

$$\log \hat{P}_{\mathcal{M}_k}(s', r|s, a) = -\big|\hat{Q}_k(s, \boldsymbol{b}, a) - (r + \gamma \mathbb{E}_{a'\sim\pi}[\hat{Q}_k(s', \boldsymbol{b}, a')])\big|^2 \quad (9)$$

$\log \hat{P}_{\mathcal{M}_k}$ provides a measure of how likely we are to see the tuple $(V(s'), r)$ in $\mathcal{M}_k$ (as a proxy for the likelihood of (s', r)) and remains aligned with the state dynamics model $P_{\mathcal{M}_k}$, in that both are high if (s', r) truly comes from $\mathcal{M}_k$.

Algorithm 1 Adaptive Policies with Ensembles of Value Functions (APE-V)

Receive input: dataset $\mathcal{D}$, number of ensemble members n
Initialize policy $\pi(\cdot|s, \boldsymbol{b}) : \mathcal{S} \times \Delta_n \to \Delta(\mathcal{A})$
Initialize ensemble of value functions $\{\hat{Q}_1, \dots \hat{Q}_n\}$, where $\hat{Q}_k(s, \boldsymbol{b}, a) : \mathcal{S} \times \Delta_n \times \mathcal{A} \to \mathbb{R}$
while π has not converged **do**
  Sample transition $(s, a, r, s') \sim \mathcal{D}$ from dataset and possible belief $\boldsymbol{b} \sim p(\boldsymbol{b})$
  Approximate next-step belief $\boldsymbol{b}' = \text{BeliefUpdate}(\boldsymbol{b}, (s, a, r, s'))$ using Equation 9
  Optimize value functions to minimize TD error taking into account the updated belief $\boldsymbol{b} \to \boldsymbol{b}'$

$$\min \mathcal{L}(\hat{Q}_k) \coloneqq (\hat{Q}_k(s, \boldsymbol{b}, a) - (r + \gamma \mathbb{E}_{a' \sim \pi(\cdot|s', \boldsymbol{b}')}[\hat{Q}_k(s', \boldsymbol{b}', a')]|))^2 \quad \forall k \in \{1, \dots n\} \tag{7}$$

  Optimize adaptive policy $\pi(\cdot|s, \boldsymbol{b})$ to maximize $\boldsymbol{b}$-weighted average of value functions

$$\max_{\pi(\cdot|s, \boldsymbol{b})} \mathbb{E}_{a_\pi \sim \pi}[\sum_k \boldsymbol{b}_k \hat{Q}_k(s, \boldsymbol{b}, a_\pi)] \tag{8}$$

end while

With these pieces in place, to update $\hat{Q}_k(s, \boldsymbol{b}, a)$ using the transition (s, a, r, s'), we first estimate the new belief vector $\boldsymbol{b}' = \text{BeliefUpdate}(\boldsymbol{b}, (s, a, r, s))$ using our surrogate model, and then optimize to achieve the consistency in Equation 5 by minimizing TD error. This leads to the following objective for a single value function in our ensemble:

$$\mathcal{L}_{\text{critic}}(\hat{Q}_k) = \mathbb{E}_{\substack{(s,a,r,s') \sim \mathcal{D} \\ \boldsymbol{b} \sim p(\boldsymbol{b})}}\big[\big(\hat{Q}_k(s, \boldsymbol{b}, a) - (r + \gamma \mathbb{E}_{a' \sim \pi(\cdot|s', \boldsymbol{b}')}[\hat{Q}_k(s', \boldsymbol{b}', a')]|)\big)^2\big].$$

Sampling the relative weighting $\boldsymbol{b} \sim p(\boldsymbol{b})$ independently of the transition allows us to estimate values for all possible belief weightings that we may encounter at any state.

6.3. Training the policy

The policy we learn takes in a state s and current belief vector $\boldsymbol{b}$ and outputs a distribution over actions. As discussed in Proposition 5.2, for any state s and belief $\boldsymbol{b}$, the optimal ascent direction maximizes the average over the value functions in our ensemble *weighted by the belief*, leading to the following policy loss for an offline actor-critic method:

$$\mathcal{L}_{\text{actor}}(\pi) = -\mathbb{E}_{\substack{s \sim \mathcal{D} \\ \boldsymbol{b} \sim p(\boldsymbol{b})}}\big[\mathbb{E}_{a \sim \pi_\theta(\cdot|s, \boldsymbol{b})}[\sum_k \boldsymbol{b}_k \hat{Q}_k(s, \boldsymbol{b}, a)]\big].$$

This policy objective has a simple interpretation: since $\boldsymbol{b}$ defines our current belief over which MDP we are actually in, when choosing an action to take at a particular state s, we should place greater consideration on the value function for the MDPs that we consider to be more likely. As in the value function objective, we train the policy for many different choices of belief so that at test-time, our policy can act appropriately for whatever belief we may have at that particular moment.

Algorithm 2 APE-V Test-Time Adaptation

$s_0 = \text{ENV.RESET}()$
Initialize belief vector to uniform: $\boldsymbol{b}_0 = [\frac{1}{n}, \dots, \frac{1}{n}]^\top$
for environment step $t = 0, 1, \dots$ **do**
  Sample action: $a_t \sim \pi(\cdot|s_t, \boldsymbol{b}_t)$
  Act in environment: $r_t, s_{t+1} \leftarrow \text{ENV.STEP}(a_t)$
  Update belief vector using new transition (Eq 9)
    $\boldsymbol{b}_{t+1} = \text{BeliefUpdate}(\boldsymbol{b}_t, (s_t, a_t, r_t, s_{t+1}))$
end for

Summary

Our actor-critic algorithm (outlined in Algorithm 1) has three core components: an ensemble of state-action value functions $\{\hat{Q}_1, \dots \hat{Q}_n\}$, a weighting $\boldsymbol{b}(h)$ over the ensemble that represents how well each value function describes the seen history, and an adaptive policy $\pi(\cdot|s, \boldsymbol{b})$ trained against the ensemble. For any belief weighting $\boldsymbol{b}$, the policy is trained to maximize the $\boldsymbol{b}$-weighted average of the value function ensemble, which prioritizes the value functions most consistent with the seen history. The ensemble of value functions is trained using standard TD methods, but modified to account for how the belief weighting changes $\boldsymbol{b} \to \boldsymbol{b}'$ with new transitions. During evaluation, the belief state $\boldsymbol{b}$ is updated online using Equation 9, which down-weights ensemble members whose value predictions are inconsistent with the new transitions; this, in turn, causes the policy to output actions focused on maximizing only the consistent value functions in the ensemble.

7. Related Work

Offline RL: Policy learning challenges in offline RL (Lange et al., 2012; Levine et al., 2020) arise when the environment is underspecified, for instance when the dataset is collected

with a narrow behavior policy or collected only in part of the state spaces (Mandlekar et al., 2021; Fu et al., 2020). A commonly used principle to handle this is pessimism: biasing the policy learning objective towards the data-collection policy, and disincentivizing behaviors that may take the agent away from the distribution of states seen in the dataset. Pessimism can be incorporated in many ways, such as policy regularization (Fujimoto et al., 2019; Kumar et al., 2019; Ghasemipour et al., 2021; Kostrikov et al., 2021b) or forming conservative value estimates (Kumar et al., 2020; An et al., 2021; Luo et al., 2021; Jin et al., 2021). In practice, these methods can suffer from over-conservatism without careful tuning (Kumar et al., 2021).

Ensembles in RL: Ensembles of value functions have been used in the online RL setting for a variety of purposes: to avoid over-estimation (Hasselt et al., 2016; Fujimoto et al., 2018), to enable faster exploration (Osband et al., 2013), and to stabilize the training objective (Chen et al., 2021; Lee et al., 2021). In offline RL, value ensembles have been deployed primarily to provide a source of pessimism into the policy optimization objective: Agarwal et al. (2020) averages an ensemble of value functions to attain more stable target values in TD-learning, Ghasemipour et al. (2022) form LCB estimates of expected return in the MDP using an ensemble of value functions; An et al. (2021) optimizes policies against the most pessimistic value function in the ensemble. In contrast to these static methods that learn a Markovian policy optimizing some ensemble statistic (e.g. min, LCB, median). our adaptive method learns an weighting-conditioned policy that optimizes all possible re-weightings of the ensemble, and uses Bayes filtering at test-time to recover the best re-weighting.

Bayesian RL: We study offline RL within the Bayesian RL framework (see (Ghavamzadeh et al., 2015) for a survey), which has been studied in many other sub-fields of RL (Ramachandran & Amir, 2007; Lazaric & Ghavamzadeh, 2010; Jeon et al., 2018; Zintgraf et al., 2020), most prominently for deriving exploration strategies in online RL. The epistemic POMDP interpretation of the Bayesian offline RL objective we introduce is a specific instantiation of the Bayes-adaptive MDP (Duff & Barto, 2002) and related to the construction of Ghosh et al. (2021). Being exactly Bayes-optimal is known to be intractable, so algorithms have been developed to learn approximations of Bayes-optimality that leverage either explicit access to the posterior over MDPs or trajectories collected in the posterior (Zintgraf et al., 2020; Dorfman & Tamar, 2020). This prevents their immediate application in offline RL, since this posterior distribution cannot be constructed with high fidelity, motivating our approach of avoiding explicitly modelling this posterior over dynamics.

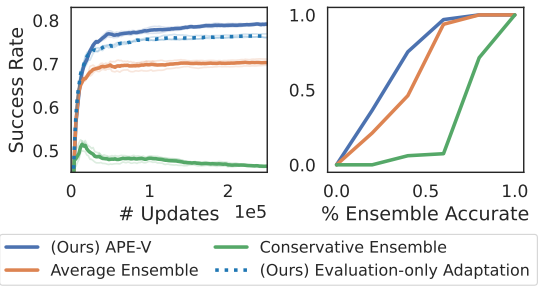

Figure 3. **Locked Doors with CIFAR-10:** (left) Success rate over training for various ensemble-based offline RL schemes (right) Success rate conditioned on the number of ensemble members with successful classification. Adaptive policies outperform stochastic and pessimistic alternatives because they succeed with higher probability if at least one ensemble member outputs accurate values.

8. Experiments

The primary aim of our experiments is to ascertain whether adaptability leads to improved performance in offline RL. Thus, we provide an evaluation on standard D4RL benchmark tasks (Fu et al., 2020) and two offline RL tasks that require handling ambiguity and generalization, *Locked Doors* and *Procgen Mazes*. We do not expect adaptability to uniformly improve across these domains, since some datasets and environments do not lead to multiple hypotheses that an agent may adapt between (e.g., in the city navigation example, if the dataset contained *no* trajectories from small roads, then no improvement can be expected over the conservative strategy using big roads). Implementation details and hyperparameters in Appendix C.

8.1. Locked Doors with CIFAR10

We first study the *Locked Doors* domain from Section 3, where an agent seeks to exit a room, but must infer which door it can leave out of by parsing a CIFAR-10 image (visualized in Figure 4, details in Appendix C). Since CIFAR-10 is a well-studied problem in supervised classification where the training set does not fully eliminate uncertainty about test image labels, embedding CIFAR-10 into an offline RL navigation problem provides us a controlled way of studying the effect of a limited offline dataset within an RL domain with a challenging perception component.

Figure 3 displays the success rate of policies learned by various offline RL procedures, where we see that policies that are adapted at test-time outperform non-adaptive baselines. Using APE-V, which trains the value ensemble and policy to approximate Bayes-optimal behavior, leads to the highest performance and robust adaptation. Using Equation 9 to adapt a policy from an ensemble trained agnostic of adaptation also leads to large improvements, although less than when trained explicitly for adaptation with APE-V. We observe qualitatively that the learned APE-V policy better takes advantage of the spatial layout of the room than an

Table 1. Normalized average returns on D4RL suite, averaged over 4 seeds. Full results in Appendix E.

Task Name	BC	SAC (Haarnoja et al., 2018)	REM (Agarwal et al., 2020)	CQL (Kumar et al., 2020)	IQL (Kostrikov et al., 2021b)	SAC-N (An et al., 2021)	APE-V
halfcheetah-medium	43.2±0.6	55.2±27.8	-0.8±1.3	44.4	47.4±0.2	67.5±1.2	**69.1 ± 0.4**
halfcheetah-medium-replay	37.6±2.1	0.8±1.0	6.6±11.0	46.2	44.2±1.2	**63.9±0.8**	**64.6 ± 0.9**
hopper-medium-replay	16.6±4.8	7.4±0.5	27.5±15.2	48.6	94.7±8.6	**101.8±0.5**	98.5 ± 0.5
walker2d-medium	70.9±11.0	-0.3±0.2	0.2±0.7	74.5	78.3±8.7	87.9±0.2	**90.3 ± 1.6**
walker2d-medium-replay	20.3±9.8	-0.4±0.3	12.5±6.2	32.6	73.8±7.1	78.7±0.7	**82.9 ± 0.4**

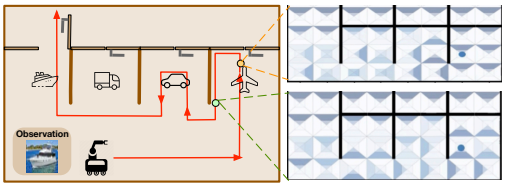

Figure 4. **Adaptation in Locked Doors:** (left) A trajectory within the *Locked Doors* domain; (right): visualization of adaptive policy before and after trying the locked door on the far right.

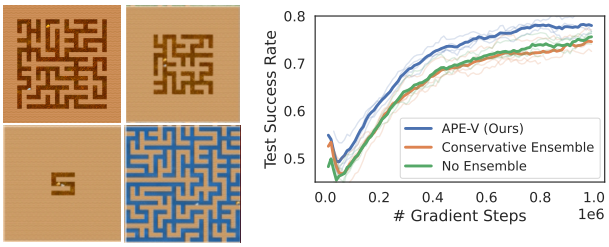

Figure 5. **Procgen Mazes:** (left) 4 sample mazes visualized in Procgen. (right) Evaluation success rates on held-out mazes when trained on offline dataset of 1000 mazes. APE-V, which adaptively chooses between candidate value functions, performs better than acting conservatively with respect to the value ensemble.

adaptive agent not trained for adaptation; the agent tries and eliminates doors physically closer, rather than in order of most-likely to least-likely, leading to more efficient pathing.

To test the resiliency of these ensemble-based agents, we plot their success rate conditioned on how many value functions provide correct predictions for the given image (Figure 3 (right)). This scales poorly for a conservative policy trained with LCB values, since an incorrect value function causes actions towards the correct door to be excessively penalized. Averaging models performs better, but degrades once the majority of the ensemble members are incorrect. In comparison, the adaptive policy can often succeed in an environment even when only one ensemble member offers correct predictions for the current image. We provide implementation details for all comparisons in Appendix C.

8.2. Procgen Mazes

We next investigate the performance of APE-V on an offline variant of the Maze task from the Procgen benchmark (Cobbe et al., 2020), a challenging image-based benchmark. An agent, which receives a top-down 64×64 image of the maze (visualized in Figure 5), must navigate to the specified exit before the episode ends. During training, the agent receives an offline dataset of transitions from N_{train} training mazes, each with differing layout and visual textures, and is evaluated on new unseen mazes during deployment. We construct the offline dataset to contain transitions of the agent at all locations within the training mazes – note that despite this uniform coverage, the agent faces epistemic uncertainty at test-time since it is evaluated on new levels never seen in the offline dataset. The full experimental setup is described in Appendix D.

We compare the performance of APE-V to an ablation that doesn't account for epistemic uncertainty and one that learns conservative value functions (using LCB) rather than being

Table 2. Maze-solving success rates, averaged over 4 seeds.

	200 Train Levels		1000 Train Levels	
	Train	Test	Train	Test
No Ensemble	0.96 ±0.02	0.24 ±.02	0.93 ±0.01	0.75 ±0.03
Conservative	0.96 ±0.02	0.23 ±0.02	0.93 ±0.01	0.75 ±0.04
APE-V	0.97 ±0.00	**0.31** ±0.04	0.92 ±0.01	**0.79** ±0.04

adaptive.When provided a training dataset of 1000 training mazes ($\approx 5 \times 10^5$ transitions), APE-V is able to reliably solve 79% of new mazes at test-time, higher than that achieved by conservatism or ignoring uncertainty (Figure 5). In a more data-limited setting with only 200 training mazes, APE-V is also able to outperform the conservative approach, although all methods succeed far less frequently on held-out mazes than the previous setting (Table 2).

8.3. D4RL

To complement our analysis in the CIFAR-10 Locked Rooms domain and Procgen Mazes, we also investigate the performance of APE-V on the D4RL benchmark (Fu et al., 2020). We instantiate our method using SAC-n (An et al., 2021) as the base procedure to learn each value function in our ensemble (details in Appendix E).

Within-episode adaptation with APE-V leads to higher evaluation performance than SAC-n, which learns a pessimistic Markov policy, and other pessimism-focused methods like CQL (Kumar et al., 2020) when trained on data from a medium-performance agent (`xxxx-medium`), or data from the buffer of an RL agent (`xxxx-medium-replay`). When the data is collected from a random policy (`xxxx-random`) or contains expert demonstrations

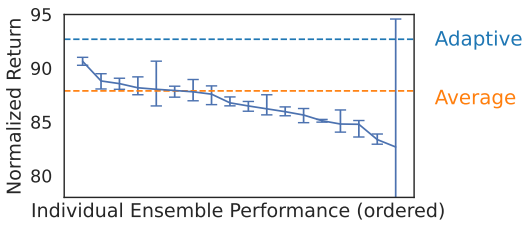

Figure 6. **Adaptation in D4RL:** Performance of ensemble members vs adaptive policy on `walker2d-medium`. Adapting the belief $\boldsymbol{b}$ within an episode outperforms any static choice of $\boldsymbol{b}$.

(`xxxx-medium-expert`), APE-V performs equivalently to the conservative baseline We do note that the improvement from adaptivity on D4RL is relatively minor; we believe this happens because these tasks generally do not have data distributions that lead to multiple salient hypotheses, so adapting amongst these strategies leads to marginal gain.

To better understand adaptation in this domain, we compare the performance of the adaptive policy to non-adaptive policies that holds the belief state static to $\boldsymbol{b} = \boldsymbol{e}_k$ (i.e., following only the k-th ensemble member $\hat{Q}_k$), visualized for `walker2d-medium` in Figure 8.3. For this task, we see that the different members of the ensemble lead to slightly differing performances, and that APE-V in fact receives average return above all these individual strategies, indicating that adaptation within the episode indeed allows the policy to adapt to a better strategy than it may have started with.

9. Discussion

In this paper, we discussed how the Bayesian perspective on offline RL naturally leads to adaptive policies. We showed that the epistemic uncertainty an offline RL agent faces due to the limited dataset manifests as an implicit partial observability, and therefore necessitates adaptive behaviors to act optimally. We then derived a policy gradient formulation in this setting that allows us to express the gradient of an adaptive policy in terms of a distribution over Q-functions and a posterior over MDPs that is induced by the uncertainty inherent in any finite-data offline RL problem. We instantiated an offline RL algorithm based on this principle called APE-V, and showed how an ensemble of value functions can be used to approximate the theoretically motivated adaptive policy update. Our algorithm is only a first foray in optimizing for adaptation in offline RL. Methods with more expressive posterior distributions over value functions, or those with more scalable adaptation mechanisms have the potential for greater benefits in adaptation. Moving forward, understanding how we may infuse existing pessimistic perspectives in offline RL with the benefits conferred by adaptivity is likely to be an exciting direction, one which we hope will lead to more powerful algorithms for learning behaviors from offline sources.

Acknowledgements

The authors thank Abhishek Gupta, Aviral Kumar, Colin Li, Katie Kang, Laura Smith, Young Geng, the members of RAIL & Improbable AI Lab, and the anonymous reviewers for discussions and helpful feedback. We thank MIT Supercloud and the Lincoln Laboratory Supercomputing Center for providing compute resources. This research was supported by an NSF graduate fellowship, a DARPA Machine Common Sense grant, an MIT-IBM grant, the Office of Naval Research, ARL W911NF-21-1-0097, and Intel.

References

Discrete-Time Birth-Death Chains, aug 10 2020. [Online; accessed 2022-01-28].

Agarwal, R., Schuurmans, D., and Norouzi, M. An optimistic perspective on offline reinforcement learning. In *ICML*, 2020.

An, G., Moon, S., Kim, J.-H., and Song, H. O. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in Neural Information Processing Systems*, 34, 2021.

Bellemare, M. G., Dabney, W., and Munos, R. A distributional perspective on reinforcement learning. In *ICML*, 2017.

Chen, X., Wang, C., Zhou, Z., and Ross, K. W. Randomized ensembled double q-learning: Learning fast without a model. *ArXiv*, abs/2101.05982, 2021.

Cobbe, K., Hesse, C., Hilton, J., and Schulman, J. Leveraging procedural generation to benchmark reinforcement learning. *ArXiv*, abs/1912.01588, 2020.

Dorfman, R. and Tamar, A. Offline meta learning of exploration. *arXiv: Learning*, 2020.

Duff, M. and Barto, A. Optimal learning: computational procedures for bayes-adaptive markov decision processes. 2002.

Eysenbach, B. and Levine, S. If maxent rl is the answer, what is the question? *arXiv preprint arXiv:1910.01913*, 2019.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4rl: Datasets for deep data-driven reinforcement learning. *ArXiv*, abs/2004.07219, 2020.

Fujimoto, S., Hoof, H. V., and Meger, D. Addressing function approximation error in actor-critic methods. *ArXiv*, abs/1802.09477, 2018.

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

Ghasemipour, S. K. S., Schuurmans, D., and Gu, S. S. Emaq: Expected-max q-learning operator for simple yet effective offline and online rl. In *International Conference on Machine Learning*, pp. 3682–3691. PMLR, 2021.

Ghasemipour, S. K. S., Gu, S. S., and Nachum, O. Why so pessimistic? estimating uncertainties for offline rl through ensembles, and why their independence matters, 2022. URL https://arxiv.org/abs/2205.13703.

Ghavamzadeh, M., Mannor, S., Pineau, J., and Tamar, A. Bayesian reinforcement learning: A survey. *Found. Trends Mach. Learn.*, 8:359–483, 2015.

Ghosh, D., Rahme, J., Kumar, A., Zhang, A., Adams, R. P., and Levine, S. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability. *ArXiv*, abs/2107.06277, 2021.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Hasselt, H. V., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. *ArXiv*, abs/1509.06461, 2016.

Jeon, W., Seo, S., and Kim, K.-E. A bayesian approach to generative adversarial imitation learning. In *NeurIPS*, 2018.

Jin, Y., Yang, Z., and Wang, Z. Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*, pp. 5084–5096. PMLR, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kostrikov, I., Fergus, R., Tompson, J., and Nachum, O. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, pp. 5774–5783. PMLR, 2021a.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021b.

Kumar, A., Fu, J., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. *arXiv preprint arXiv:1906.00949*, 2019.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.

Kumar, A., Singh, A., Tian, S., Finn, C., and Levine, S. A workflow for offline model-free robotic reinforcement learning. *arXiv preprint arXiv:2109.10813*, 2021.

Lange, S., Gabel, T., and Riedmiller, M. A. Batch reinforcement learning. In *Reinforcement Learning*, 2012.

Lazaric, A. and Ghavamzadeh, M. Bayesian multi-task reinforcement learning. In *ICML*, 2010.

Lee, K., Laskin, M., Srinivas, A., and Abbeel, P. Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning. In *ICML*, 2021.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Luo, M., Balakrishna, A., Thananjeyan, B., Nair, S., Ibarz, J., Tan, J., Finn, C., Stoica, I., and Goldberg, K. Mesa: Offline meta-rl for safe adaptation and fault tolerance. *arXiv preprint arXiv:2112.03575*, 2021.

Mandlekar, A., Xu, D., Wong, J., Nasiriany, S., Wang, C., Kulkarni, R., Fei-Fei, L., Savarese, S., Zhu, Y., and Martín-Martín, R. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Monahan, G. E. A survey of partially observable markov decision processes: Theory, models, and algorithms. 2007.

Osband, I., Russo, D., and Roy, B. V. (more) efficient reinforcement learning via posterior sampling. In *NIPS*, 2013.

Puterman, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

Ramachandran, D. and Amir, E. Bayesian inverse reinforcement learning. In *IJCAI*, 2007.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay. *CoRR*, abs/1511.05952, 2016.

Singh, S. P., Jaakkola, T. S., and Jordan, M. I. Learning without state-estimation in partially observable markovian decision processes. In *Proceedings of the Eleventh International Conference on International Conference*

on Machine Learning, pp. 284–292, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc. ISBN 1558603352.

Zintgraf, L. M., Shiarlis, K., Igl, M., Schulze, S., Gal, Y., Hofmann, K., and Whiteson, S. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. *ArXiv*, abs/1910.08348, 2020.

A. Supplementary for Section 4

A.1. The Epistemic POMDP Formalism

POMDPs: Before we outline the epistemic POMDP, we first provide a quick introduction to partially observable MDPs (POMDPs). A POMDP is defined by the tuple $(\overline{\mathcal{S}}, \mathcal{A}, \mathcal{O}, \overline{P}, O, r, \rho, \gamma)$, where $\overline{\mathcal{S}}$ is the (hidden) state space of the POMDP, $\mathcal{A}$ the action space, $\mathcal{O}$ is the observation space for the agent, $\overline{P}(\overline{s}'|\overline{s}, a)$ is a Markovian transition function between hidden states, O is the emission function that maps a hidden state to the agent's observation $o_t = O(\overline{s}_t)$, $r(\overline{s}, a)$, $\rho(\overline{s})$ the initial hidden state distribution, and γ discount factor. The belief state for a POMDP is given by $p(\overline{s}|h)$, the mapping from a history to the conditional distribution over hidden states having seen this history. It is well-known that the optimal policy in a POMDP is in general history-dependent, and that there is always an optimal policy that depends on history only through the belief state.

The Epistemic POMDP: The epistemic POMDP for offline RL $\mathcal{M}_{po}$ is a POMDP that satisfies the following property: for any policy π, $J_{\mathcal{M}_{po}}(\pi) = J_{\text{Bayes}}(\pi)$. This property means that optimizing the Bayesian offline RL objective is equivalent to optimization in the epistemic POMDP, a useful equivalence for deriving properties of optimality in offline RL and designing new offline RL algorithms.

Formally, given a posterior distribution $P(\mathcal{M}|\mathcal{D})$ over MDPs with shared state space $\mathcal{S}$ and action space $\mathcal{A}$, $\mathcal{M}_{po}$ is defined as following. The hidden state space is $\overline{\mathcal{S}} = \mathcal{S} \times \mathbf{M}$ and a state represented as $\overline{s} := (s, \mathcal{M})$ and the action space is still $\mathcal{A}$, the transition function extended as $\overline{P}((s', \mathcal{M}')|(s, \mathcal{M}), a) = \delta(\mathcal{M} = \mathcal{M}')P_{\mathcal{M}}(s'|s, a)$. The observation function omits the identity of $\mathcal{M}$ from the agent observation: $O((s, \mathcal{M})) = s$. The reward function is $r((s, \mathcal{M}), a) = r_{\mathcal{M}}(s, a)$ and initial state distribution $\rho((s, \mathcal{M})) = P(\mathcal{M}|\mathcal{D})\rho_{\mathcal{M}}(s)$.

A.2. Proof of Theorem 4.1

We note that many prior works in POMDPs (Singh et al., 1994; Duff & Barto, 2002) that show that the optimal solution for POMDPs in general, and for Bayesian objectives is adaptive. In the following theorem, we construct an instance of the epistemic POMDP in which the adaptive policy significantly outperforms all Markovian policies, addressing both desired components.

Proposition A.1 (Sub-optimality of Markovian policies and optimality of adaptiveness)**.** *Let $n \in \mathbb{N}$. There are offline RL problem instances $(\mathcal{D}, p(\mathcal{M}))$ with n-state MDPs where the adaptive Bayes-optimal policy achieves $J_{Bayes}(\pi^*_{adaptive}) = -2n$ but the highest performing Markovian policy achieves return of a magnitude worse: $J_{Bayes}(\pi^*_{markov}) \leq -\frac{1}{2}n^2$.*

Summary and Intuition: Adaptivity is particularly useful when an agent needs to try a sequence of actions, and if it fails, then to backtrack on its steps and return to try something else – behavior that a Markovian policy cannot well imitate. We capture this intuition in our construction, a chain environment where the agent is uncertain if the goal is on the left or the right, so performing well under the Bayesian objective requires being able to reach both the left-hand side of the environment and the right-hand side of the environment. We will show that an adaptive agent can do this in $O(n)$ timesteps but an Markovian policy will need at least $O(n^2)$ timesteps.

Proof.
MDP construction: Our construction uses two standard chain MDPs $\mathcal{M}_1, \mathcal{M}_2$ supported on the states $\{-n, \ldots, 0, \ldots, n\}$ where the agent starts at state 0; at all states k, the agent can go left (to $k-1$) or right (to $k+1$), as long as these states exist and receives -1 reward at every timestep. We modify these MDPs as follows: in the first MDP $\mathcal{M}_1$, entering state n leads to immediate termination; in the second MDP $\mathcal{M}_2$, entering state $-n$ leads to immediate termination. Let $\gamma = 1$.

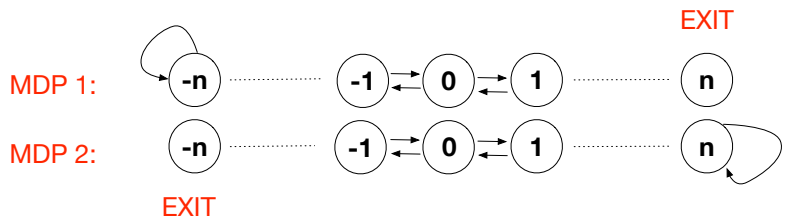

Figure 7. Visualization of MDP construction described in Appendix A.2

Construction of Posterior Distribution: Suppose that after having received an offline dataset that the posterior distribution was uniform on these MDPs: $p(\mathcal{M}_1|\mathcal{D}) = p(\mathcal{M}_2|\mathcal{D}) = \frac{1}{2}$. One way that this could happen in practice is if the prior $p(\mathcal{M})$ had $p(\mathcal{M}_1) = p(\mathcal{M}_2) = \frac{1}{2}$, and no trajectories in the dataset actually reached the ends of the chain (n or $-n$) so it received no signal about this ambiguity. The value of the Bayesian offline RL objective for this problem for a policy is given by $J_{Bayes}(\pi) = -\frac{1}{2}(T_\pi(n) + T_\pi(-n)$, where $T_\pi(n)$ is the expected amount of time it takes π to first reach state n and $T_\pi(-n)$ defined similarly.

Performance of Bayes-optimal adaptive policy: The optimal adaptive policy for the Bayesian offline RL objective defined by $p(\mathcal{M}|\mathcal{D})$ is simple: it first goes right for n timesteps, and then left for $2n$ timesteps (or vice-versa). If it is in $\mathcal{M}_1$, then it exits in n timesteps, otherwise in $3n$ timesteps. This leads to a return of

$$J_{Bayes}(\pi^*_{\text{adaptive}}) = \frac{1}{2}(-n) + \frac{1}{2}(-3n) = -2n$$

Upper bound on performance of Markovian policy: Consider a Markovian policy $\pi(a|n)$ acting in this environment and define $p_n = \pi(\text{right}|n)$, $q_n = 1 - p_n$, and $\ell_n = \frac{p_n}{q_n}$. Notice that $T_\pi(n) + T_\pi(-n)$ can be lower-bounded by $T_{n,0}$ and $T_{-n,0}$, where $T_{n,0}$ is the expected time of π to first return to 0 after visiting n, and $T_{-n,0}$ equivalently. We suppose without loss of generality that $T_{n,0} < T_{-n,0}$. We can write

$$T_{n,0} = \underbrace{\mathbb{E}[\text{Time to reach n from 0}]}_{T_{0\to n}} + \underbrace{\mathbb{E}[\text{Time to reach 0 from n}]}_{T_{n\to 0}}.$$

Noticing that the agent's movement is described by a birth-death chain with birth rate p_n and death rate q_n, we can use results about mean absorbtion time in birth-death chains from the stochastic processes literature (202, 2020) to receive lower-bounds on $T_{0\to n}$ and $T_{n\to 0}$:

$$T_{0\to n} \geq \sum_{j=0}^{n-1} \sum_{k=j+1}^{n-1} \prod_{i=j}^{k} \ell_i \tag{10}$$

$$T_{n\to 0} = \sum_{j=0}^{n-1} \sum_{k=j+1}^{n-1} \prod_{i=j}^{k} \ell_i^{-1} \tag{11}$$

We can combine the two to get a lower bound on $T_{n,0} = T_{n\to 0} + T_{0\to n}$

$$T_{n,0} = T_{n\to 0} + T_{0\to n} \geq \sum_{j=0}^{n-1} \sum_{k=j+1}^{n-1} \left(\prod_{i=j}^{k} \ell_i\right) + \left(\prod_{i=j}^{k} \ell_i\right)^{-1} \tag{12}$$

$$\tag{13}$$

We see that this lower bound on $T_{0\to n} + T_{n\to 0}$ is minimized when $\ell_i = 1$ (that $p_i = q_i = \frac{1}{2}$) for all states i, and takes value n^2. This provides a lower bound on $T_\pi(n) + T_\pi(-n)$ and therefore, the following upper bound on the performance of the Markovian policy:

$$J_{Bayes}(\pi^*_{\text{markov}}) \leq -\frac{1}{2}n^2$$

□

B. Derivations for Section 5

Proposition 5.1. *The Bayesian offline RL objective is maximized by a policy depending only on the current state and relative MDP belief:* $\pi(\cdot|h_t) = \pi(\cdot|s_t, \boldsymbol{b}(h_t))$.

Proof. We will show that there is an optimal policy that takes form $\pi(\cdot|s, \boldsymbol{b})$ by showing that $(s, \boldsymbol{b})$ forms a belief state for the epistemic POMDP, and use the well-known fact that there always exists an optimal policy depending only on the belief state of the POMDP (Monahan, 2007).

Recall that the true state in the epistemic POMDP $\overline{s}$ is given by the tuple $\overline{s} := (s, \mathcal{M})$, where s is the current MDP state and $\mathcal{M}$ the MDP currently being acted in. By definition, a belief state for the POMDP is one that is isomorphic to the distribution $P(\overline{s}|h)$.

We first show that $P(\overline{s}|h)$ can be recovered from $(s(h), \boldsymbol{b}(h))$:

$$\begin{aligned} P_{\mathcal{M}_{po}}(\overline{s}|h) = P_{\mathcal{M}_{po}}(s, \mathcal{M}|h) &= P_{\mathcal{M}_{po}}(s|h) P_{\mathcal{M}_{po}}(\mathcal{M}|h, s) \\ &= \delta(s = s(h)) P(\mathcal{M}|h, \mathcal{D}) \\ &= \delta(s = s(h)) \boldsymbol{b}(h)(\mathcal{M}) P(\mathcal{M}|\mathcal{D}). \end{aligned}$$

That $(s(h), \boldsymbol{b}(h))$ can be recovered from $P(\overline{s}|h)$ follows immediately from the definition of $\boldsymbol{b}(h)$ (since it is defined in terms of $P(\mathcal{M}|h, \mathcal{D})$). Therefore, $(s(h), \boldsymbol{b}(h))$ is a belief state for the epistemic POMDP, and as immediate corollary, there exists an optimal policy for the epistemic POMDP that depends only on this belief state. □

Proposition 5.2. *The gradient of the Bayesian offline RL objective for an uncertainty-adaptive policy* $\pi_\theta(\cdot|s, \boldsymbol{b})$ *can be written in terms of the MDP value functions* $Q^\pi_{\mathcal{M}}$ *from the posterior distribution:*

$$\begin{aligned} \nabla_\theta J_{\textit{Bayes}}(\pi_\theta) = \mathbb{E}_{h\sim\pi}[\nabla_\theta \mathbb{E}_{a\sim\pi_\theta(\cdot|s(h),\boldsymbol{b}(h))}[\\ \mathbb{E}_{\mathcal{M}\sim P(\mathcal{M}|\mathcal{D})}[\boldsymbol{b}(h)(\mathcal{M}) Q^\pi_{\mathcal{M}}(h, a)]]. \end{aligned} \tag{4}$$

Proof. From the equivalence $J_{\text{Bayes}}(\pi_\theta) = J_{\mathcal{M}_{po}}(\pi_\theta)$ and the policy gradient of a history-based policy in a POMDP (Monahan, 2007), we have that

$$\nabla_\theta J_{\text{Bayes}}(\pi_\theta) = \mathbb{E}_{h\sim\pi}[\nabla_\theta \mathbb{E}_{a\sim\pi_\theta(\cdot|h)}[Q^\pi_{\mathcal{M}_{po}}(h, a)]], \tag{14}$$

where $Q^\pi_{\mathcal{M}_{po}}(h, a)$ is the value function of π in the epistemic POMDP. To prove the desired statement, we show that $Q^\pi_{\mathcal{M}_{po}}(h, a) = \mathbb{E}_{\mathcal{M}\sim P(\mathcal{M}|\mathcal{D})}[\boldsymbol{b}(h)(\mathcal{M}) Q^\pi_{\mathcal{M}}(h, a)]$.

Let $p^\pi_{\mathcal{M}}(\tau)$ be the distribution over trajectories $(s_0, a_0, r_0, s_1, a_1, \dots)$ for the belief-based policy π in MDP $\mathcal{M}$. From construction of the epistemic POMDP, we can see that $p^\pi_{\mathcal{M}_{po}}(\tau) = \mathbb{E}_{\mathcal{M}\sim P(\mathcal{M}|\mathcal{D})}[p_{\mathcal{M}}(\tau)]$. For a T-step history $h :=$

$(s_0, a_0, r_0, s_1, \dots s_T)$, the value function of π, $Q^\pi_{\mathcal{M}_{po}}(h, a)$ is given by

$$Q^\pi_{\mathcal{M}_{po}}(h,a) = \mathbb{E}_{\tau \sim p^\pi_{\mathcal{M}_{po}}}\left[\sum_{t=T}^{\infty} \gamma^{t-T} r_t | h = h, a = a\right] \tag{15}$$

$$= \int \left(\sum_{t=T}^{\infty} \gamma^{t-T} r_t\right) p^\pi_{\mathcal{M}_{po}}(\tau|h) d\tau \tag{16}$$

$$= \int\int \left(\sum_{t=T}^{\infty} \gamma^{t-T} r_t\right) p^\pi_{\mathcal{M}}(\tau|h) P(\mathcal{M}|\mathcal{D}, h) d\mathcal{M}\, d\tau \tag{17}$$

$$= \int\int \left(\sum_{t=T}^{\infty} \gamma^{t-T} r_t\right) p^\pi_{\mathcal{M}}(\tau|h) \boldsymbol{b}(h)(\mathcal{M}) P(\mathcal{M}|\mathcal{D}) d\mathcal{M}\, d\tau \tag{18}$$

$$= \int\int \left(\sum_{t=T}^{\infty} \gamma^{t-T} r_t\right) p^\pi_{\mathcal{M}}(\tau|h) \boldsymbol{b}(h)(\mathcal{M}) P(\mathcal{M}|\mathcal{D}) d\mathcal{M}\, d\tau \tag{19}$$

$$= \int \underbrace{\mathbb{E}_{\tau \sim p^\pi_{\mathcal{M}}}\left[\left(\sum_{t=T}^{\infty} \gamma^{t-T} r_t\right) | h = h, a = a\right]}_{Q^\pi_{\mathcal{M}}(h,a)} \boldsymbol{b}(h)(\mathcal{M}) P(\mathcal{M}|\mathcal{D}) d\mathcal{M}\, d\tau \tag{20}$$

$$= \mathbb{E}_{\mathcal{M} \sim P(\mathcal{M}|\mathcal{D})}[\boldsymbol{b}(h) Q^\pi_{\mathcal{M}}(h,a)] \tag{21}$$

□

Proposition 5.3. *The value function for a uncertainty-adaptive policy in an MDP $Q^\pi_{\mathcal{M}}(h, a)$ depends on h_t only through $(s_t, \boldsymbol{b}(h_t))$ and satisfies the following Bellman recursion:*

$$Q^\pi_{\mathcal{M}}(s, \boldsymbol{b}, a) = r(s, a) + \gamma \mathbb{E}_{\substack{s' \sim \mathcal{M} \\ a \sim \pi}}[Q^\pi_{\mathcal{M}}(s', \boldsymbol{b}', a)] \tag{5}$$

where $\boldsymbol{b}' \coloneqq \mathrm{BeliefUpdate}(\boldsymbol{b}, (s, a, r, s))$ is the new relative MDP belief after witnessing (s, a, r, s'),

$$\mathrm{BeliefUpdate}(\boldsymbol{b}, (s, a, r, s))(\mathcal{M}) \propto p_{\mathcal{M}}(r, s'|s, a) \boldsymbol{b}(\mathcal{M}) \tag{6}$$

Proof. Consider an arbitrary T-step history $h \coloneqq (s_0, a_0, r_0, s_1, \dots, s_T)$; we write $\boldsymbol{b}_t$ to be the relative MDP belief after t steps. The value of any policy π in $\mathcal{M}$ after taking action a is defined as

$$Q^\pi_{\mathcal{M}}(h, a) = \mathbb{E}_{\pi, \mathcal{M}}[\sum_{t=T}^{\infty} \gamma^{t-T} r_t | h = h, a_T = a].$$

We note the following conditional independences: $s_{T+1} \perp h \mid (s_T, a_T)$ because $\mathcal{M}$ has Markovian transition dynamics and that $a_{T+1} \perp h \mid (s_T, \boldsymbol{b}_T)$ as a_{T+1} depends only on s_{T+1} and $\boldsymbol{b}_{T+1}$ (since π is belief-based) and $\boldsymbol{b}_{T+1}$ depends only on s_T, a_T, and $\boldsymbol{b}_T$. Iterating the argument over, we see that the future trajectory depends on h only through s_T and $\boldsymbol{b}_T$, and therefore the expected discounted return $Q^\pi_{\mathcal{M}}(h, a)$ also only depends on h through s_T and $\boldsymbol{b}_T$.

$$Q^\pi_{\mathcal{M}}(h, a) = Q^\pi_{\mathcal{M}}(s(h), \boldsymbol{b}(h), a).$$

To derive the recursion, we simply notice that by definition, $Q^\pi_{\mathcal{M}}(h, a)$ must satisfy

$$Q^\pi_{\mathcal{M}}(h, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P_{\mathcal{M}}(\cdot|s,a)}[\mathbb{E}_{a' \sim \pi(\cdot|h \cup (s,a,r,s'))}[Q^\pi_{\mathcal{M}}(h \cup (s, a, r, s'), a')]] \tag{22}$$

Injecting the fact that $Q^\pi_{\mathcal{M}}$ and π depend only on s and $\boldsymbol{b}$, we get the desired consistency equation

$$Q^\pi_{\mathcal{M}}(s, \boldsymbol{b}, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P_{\mathcal{M}}(\cdot|s,a)}[\mathbb{E}_{a' \sim \pi(\cdot|s', \boldsymbol{b}')}[Q^\pi_{\mathcal{M}}(s', \boldsymbol{b}', a')]] \tag{23}$$

□

Table 3. Hyperparameters used for training Q learning based agents in Locked Doors domain

Hyperparameter	Value
γ	0.98
batch size	256
learning rate	1e-3
Optimizer	Adam (Kingma & Ba, 2014)
Training steps	250k
Number of ensembles	5
$p(\boldsymbol{b})$	SymmetricDirichlet(0.1)

C. Details for *Locked Doors* Domain

C.1. Task Description

At beginning of every episode, an image is uniformly sampled from a reduced CIFAR10 dataset consisting only of classes airplane, automobile, ship and truck. The agent receives this $32 \times 32 \times 3$ image in its observation, alongside its current (x, y) position in the room (see figure 7); success requires the agent to go through the door corresponding to the right image class within $T = 50$ steps; it physically is unable to go through any of the other doors during the episode. The action space is discrete corresponding to the four cardinal directions.

C.2. Implementation details

For the locked doors domain, we instantiate APE-V using deep Q learning (Mnih et al., 2013) as the base learning algorithm to train the ensemble of Q networks. Since the action-space is discrete, we do not maintain a separate actor network, instead directly computing the optimal policy given the value functions. For APE-V, this corresponds to using $\pi(a|s, \boldsymbol{b}) = \arg\max_a \sum_k \boldsymbol{b}_k \hat{Q}_k(s, \boldsymbol{b}, a)$ throughout training and test-time. For the average ensemble baseline, we train the Q networks separately using deep Q learning, and only combine the value functions together at test-time. For APE-V (evaluation only) baselines, we follow the same training procedure as for the average ensemble baseline, but choose actions according to the adaptive policy $\arg\max_a \boldsymbol{b}_k \hat{Q}_k(s, \boldsymbol{b}, a)$ at test-time. Our conservative ensemble baseline uses an LCB estimate of Q-values to define the policy ($\pi(a|s) = \arg\max \text{mean}(\{\hat{Q}_k(s, a)\}_k) - \beta \text{std}(\{\hat{Q}_k(s, a)\}_k)$) during training and evaluation.

Hyperparameters for training APE-V and the accompanying baselines are presented in table 3. The neural network architecture we use for our value functions has a CNN head used for processing the CIFAR image, with 3 convolution layers with output channel of 32 and kernel size of 3 and a dense layer with output dimension of 10. Each of the convolution layer is followed by ReLU activation and Avg pooling with a stride of 2. The output of the CNN head, the current belief vector $\boldsymbol{b}$ and the (x,y) position of the agent are concatenated and passed to a fully connected network, with 2 hidden layers of size 256 and ReLU activation, to get Q values for all the 4 actions.

C.3. Additional Analysis

In our experiments, we found that the conservative ensemble significantly underperforms, receiving lower return than even a single ensemble member. When visualizing the behavior of the conservative baseline, we found that it has a tendency to get stuck in place even before trying any of the doors. As an example of this failure mode, visualized in the figure, on a new test image, the *right* action has a high Q-value under the first ensemble member but low under the second, and the *up* action has a low Q-value under the first ensemble member but high in the second. The stay action is neutral (neither helpful nor harmful) under both value functions in the ensemble, and is chosen by the LCB statistic since it appears to have the same net benefit (although this is not true in practice).

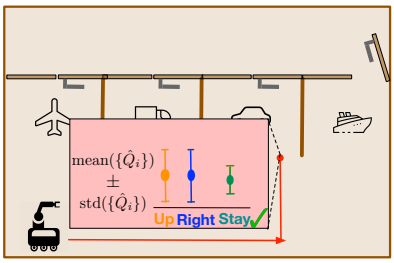

Table 4. Hyperparameters used for training Q learning based agents in Procgen Mazes domain

Hyperparameter	Value
γ	0.99
Reward shift	-1.0
Distributional support	LINSPACE(-31, 9, 81)
Batch size	256
Learning rate	6.25e-5
Optimizer	Adam (Kingma & Ba, 2014)
Training steps	10^6
Number of ensembles	2
$p(\boldsymbol{b})$	SymmetricDirichlet(1.0)

D. Details for *Procgen Mazes* Domain

D.1. Environment and Dataset Details

The Procgen Maze task (Cobbe et al., 2020) requires an agent to control a mouse to reach a block of cheese (the goal) in some maze layout within 500 environment steps. Each time-step, the agent receives a render of the full environment as a $64 \times 64 \times 3$ image (which contains the mouse, the full maze layout, and the goal), and must take one of 15 actions (the standardized Procgen interface). Since the whole maze is visible to the agent at each time-step, there is no partial observability and the environment is an MDP. We procedurally generate a dataset for the Procgen task from a set of training levels in the following way. For each maze, we enumerate the list of valid positions in the maze, then manually reset the agent to each position and take the $\{$Up, Down, Left, Right$\}$ actions, logging the ensuing transitions. These per-maze datasets are concatenated together to form the full offline dataset (we create two offline datasets, one with 200 training mazes, and another with 1000 mazes).

D.2. Implementation details

For all of our comparisons, we train value functions using the C51 (Bellemare et al., 2017) algorithm for discrete max-Q learning, using prioritized experience replay (Schaul et al., 2016) to sample from the offline dataset. We parameterize the Q-function using the same Impala encoder as Cobbe et al. (2020), with a linear readout for the logits of the distributional value function. As in *Locked Doors*, since APE-V additionally takes in a belief vector, we concatenate the belief vector to the output of the visual Impala encoder, and pass this on to the rest of the network. Aside from these environment-specific details, training is the same as in the Locked Doors domain – exact hyperparameter details are provided in Table 4.

Table 5. Normalized average returns on D4RL suite, averaged over 4 random seeds.

Task Name	BC	SAC (Haarnoja et al., 2018)	REM (Agarwal et al., 2020)	CQL (Kumar et al., 2020)	IQL (Kostrikov et al., 2021b)	SAC-N (An et al., 2021)	APE-V
halfcheetah-random	2.2±0.0	29.7±1.4	-0.8±1.1	**35.4**	31.3±3.5	29.8±1.6	29.9±1.1
halfcheetah-medium	43.2±0.6	55.2±27.8	-0.8±1.3	44.4	47.4±0.2	67.5±1.2	**69.1 ± 0.4**
halfcheetah-medium-expert	44.0±1.6	28.4±19.4	0.7±3.7	62.4	95.0±1.4	**102.7±1.5**	**101.4 ± 1.4**
halfcheetah-medium-replay	37.6±2.1	0.8±1.0	6.6±11.0	46.2	44.2±1.2	**63.9±0.8**	**64.6 ± 0.9**
hopper-random	3.7±0.6	9.9±1.5	3.4±2.2	10.8	5.3±0.6	**31.3±0.0**	**31.3±0.2x**
hopper-medium-expert	53.9±4.7	0.7±0.0	0.8±0.0	**111.0**	96.9±15.1	110.1±0.3	105.72 ± 3.7
hopper-medium-replay	16.6±4.8	7.4±0.5	27.5±15.2	48.6	94.7±8.6	**101.8±0.5**	98.5 ± 0.5
walker2d-random	1.3±0.1	0.9±0.8	6.9±8.3	7.0	5.4±1.7	**16.3±9.4**	**15.5±8.5**
walker2d-medium	70.9±11.0	-0.3±0.2	0.2±0.7	74.5	78.3±8.7	87.9±0.2	**90.3 ± 1.6**
walker2d-medium-expert	90.1±13.2	1.9±3.9	-0.1±0.0	98.7	109.1±0.2	**116.0±6.3**	110.0 ± 1.5
walker2d-medium-replay	20.3±9.8	-0.4±0.3	12.5±6.2	32.6	73.8±7.1	78.7±0.7	**82.9 ± 0.4**

E. Details for *D4RL* Benchmark

E.1. Implementation Details

For the D4RL benchmark, we instantiate APE-V using SAC-n (An et al., 2021) as our base value learning method since it has fewer issues of optimization than standard SAC on the D4RL tasks. SAC-n parameterizes a Q-function as the minimum of n independent value functions (standard SAC is SAC-n with $n = 2$) APE-V learns an ensemble of K SAC-n agents, each with different values of n and maintains a belief over them for test time adaptation. To better capture the uncertainty in the environment, we promote diversity amongst the SAC-n agents in the ensemble by choosing a different value of n for each of them. We parameterize the actor using a set of K independent networks as well to avoid potential challenges in optimizing different combinations of our Q-functions simultaneously: $\pi(\cdot|s, \boldsymbol{b}) = \sum_{i=1}^{K} \boldsymbol{b}_i \pi_i(\cdot|s)$ and use $p(\boldsymbol{b}) = \text{SymmetricDirichlet}(.01)$. Since APE-V contains an ensemble of K SAC-n agents, each of which contains n Q functions, learning APE-V can become computationally intensive for large values of n. To remain computationally tractable, we implement weight sharing between Q functions of different SAC-n agents. Specifically, let $\{n_1, \dots n_K\}$ be the values of n we wish to use for each of our ensemble members; we train $\max_i n_i$ ensembles $\{Q_1, \dots Q_{\max_i n_i}\}$, each with K heads. Using this, we define the i-th SAC-n agent $Q_i^{\text{SAC-n}}$ as $Q_i^{\text{SAC-n}} = \min\{Q_1^i, \dots, Q_{n_i}^i\}$. This construction ensures that all the networks within each SAC-n agent be independent, while avoiding creating an excessive number of ensembles.

We use hyperparameters from An et al. (2021) (https://github.com/snu-mllab/EDAC.git). The original implementation of SAC-n uses separate values of n for each domain: 10 for half-cheetah, 20 for walker, and 500 for hopper. We train $N - 2$ SAC-n agents, with values of n in $\{2, \dots, N\}$, where N is the number of ensembles in the original implementation of SAC-n. With the exception of Half-Cheetah (which has the lowest number of ensembles), we implement weight sharing among SAC-n agents contained in APE-V to reduce ensemble training costs.