# Directed Exploration via Uncertainty-Aware Critics

**Anonymous Authors**[1]

## Abstract

The exploration-exploitation dilemma is still an open problem in Reinforcement Learning (RL), especially when coped with deep architectures and in the context of continuous action spaces. Uncertainty quantification has been extensively used as a means to achieve efficient directed exploration. However, state-of-the-art methods for continuous actions still suffer from high sample complexity requirements. Indeed, they either completely lack strategies for propagating the epistemic uncertainty throughout the updates, or they mix it with aleatory uncertainty while learning the full return distribution (e.g., distributional RL). In this paper, we propose Wasserstein Actor-Critic (WAC), an actor-critic architecture inspired by the recent Wasserstein Q-Learning (WQL) (Metelli et al., 2019), that employs approximate Q-posteriors to represent the epistemic uncertainty and Wasserstein barycenters for uncertainty propagation across the state-action space. WAC enforces exploration in a principled way by guiding the policy learning process with the optimization of an upper bound of the Q-value estimates. Furthermore, we study some peculiar issues that arise when using function approximation, coupled with the uncertainty estimation, and propose a regularized loss for the uncertainty estimation. Finally, we evaluate our algorithm on a suite of continuous-actions domains, where exploration is crucial, in comparison with state-of-the-art baselines. Our experiments show a clear benefit of using uncertainty-aware critics for continuous-actions control.

## 1. Introduction

Reinforcement Learning (RL Sutton & Barto, 2018) is one of the most widely used frameworks for solving sequential decision-making problems, especially in model-free settings, where a model of the environment dynamics is not available. When an agent acts in an uncertain environment, it faces the choice between *exploring* with the hope of discovering more profitable behaviors or *exploiting* the current information about the actions' values. This exploration-exploitation dilemma is particularly challenging in *continuous-state* spaces, where *function approximation* is required to generalize across states, and, differently from the tabular case, an accurate estimate of the uncertainty on the value estimates is not available point-wise. *Continuous-action* tasks pose additional challenges since most exploration methods require the maximization of some objective (e.g., upper bound of the Q-value) over the action space. While in the discrete case, this maximization can be performed by enumeration, in the continuous case it requires solving a complex optimization problem, increasing the computational demands.

*Actor-Critic* (AC) methods (Haarnoja et al., 2018; Ciosek et al., 2019; Schulman et al., 2015) represent the current state of the art for continuous control. Despite their widespread adoption, these methods still suffer from high sample complexity. Efficient exploration strategies have been extensively studied in the literature as a means of reducing sample complexity mainly in tabular domains (Auer et al., 2008; Ian et al., 2013; Metelli et al., 2019; O'Donoghue et al., 2018). Classical exploration strategies, like $\epsilon$-greedy or Boltzmann (Sutton & Barto, 2018), inject noise around the current greedy policy to enforce exploration. Although in simple settings this is enough to guarantee convergence (Szepesvári, 1997), this exploration strategy is not efficient in the general case.

Another line of approaches considers the *maximum entropy* setting to improve exploration, and avoids the *deterministic collapse* of the policies, e.g., Soft Actor Critic (SAC, Haarnoja et al., 2018). In this setting, stochastic policies are preferred by optimizing the expected return regularized with an entropy term. This too represents a form of *undirected* exploration since the policies are forced to be stochastic, thanks to the entropy bonus, but the induced noise does not consciously shift its focus towards promising regions of the state space. A common trend in the RL literature consists in endowing existing methods with some form of uncertainty quantification and using it to perform *directed* exploration while focusing on the most promising regions. For instance, *optimistic* approaches have been applied to both Q-learning (Jin et al., 2018) and SAC. In particular, a recent extension of SAC, Optimistic Actor-Critic(OAC, Ciosek et al., 2019), proved to improve sample efficiency

over the standard SAC.

Indeed, uncertainty quantification is a fundamental step to define efficient exploration strategies. The most used exploration strategies, coming from the Multi-Armed Bandit (MAB, Lattimore, 2020 - 2020) literature, use uncertainty estimates to explore either based on optimism (Auer et al., 2002) or *posterior sampling*(PS, Thompson, 1933). These methods have been extended for the RL settings, starting from tabular domains (Auer et al., 2008; Ian et al., 2013; Metelli et al., 2019), with theoretical guarantees on the sample complexity and/or regret. Uncertainty quantification methods have been proposed for the Deep Reinforcement Learning (DRL) settings too, but the guarantees no longer hold up. This is done by means of *posterior* distributions to represent uncertainty. Posterior distributions are either modeled explicitly (Metelli et al., 2019; Lee et al., 2021) or implicitly by using an ensemble of value estimates (Wang et al., 2021). Ensemble methods allow quantifying the uncertainty but do not *propagate* it across the state action-space when performing the critic updates. Uncertainty propagation is a fundamental tool of any principled uncertainty estimation approach since most AC methods rely on bootstrapping when updating the critics. This results in Q-value estimates that also incorporate uncertainty about the bootstrapped values. Distributional RL (O'Donoghue et al., 2018) allows for uncertainty propagation but considers only aleatoric uncertainty, being aimed at estimating the full return distribution. This is not straightforward in practice, and, furthermore, it is not strictly necessary in the classical RL setting where the goal is to maximize the expected return. To the best of our knowledge, the only method capable of propagating the epistemic uncertainty, without the need to learn the return distribution is Wasserstein Q-learning (WQL Metelli et al., 2019), which has only been proposed for the discrete action case.

In this paper, we address the problem of uncertainty estimation and propagation in the context of continuous-action RL. Starting from the methodology introduced in WQL (Metelli et al., 2019), we devise a novel actor-critic algorithm, *Wasserstein Actor-Critic* (WAC), which employs Q-posteriors both to quantify uncertainty on the critic estimates to drive exploration, as well as a tool to propagate it across the state-action space, by means of Wasserstein barycenters (Section 3). The Q-posteriors quantify the epistemic uncertainty of the Q-values and incorporate both the uncertainty due to the empirical estimate of the stochastic transition and immediate reward sample, as well as the Q-value uncertainty of the next states imported during the bootstrapping of the Temporal Difference (TD, Sutton & Barto, 2018) updates. Furthermore, we consider some practical problems that arise while quantifying uncertainty by means of Q-posteriors coped with function approximators, especially neural networks. To this end, we propose a new regularization approach for the uncertainty networks to avoid the collapse of the uncertainty estimates due to uncontrolled generalization (Section 4). WAC uses the Q-posteriors to explore efficiently, by optimizing an upper bound of the Q-values. Unlike OAC (Ciosek et al., 2019), which employs bootstrapped uncertainty estimates from an ensemble of critics (two) to define the upper bound, we employ the Q-posteriors, which will eventually shrink to point estimates. Furthermore, WAC recovers SAC for a specific hyperparameter configuration and, more importantly, is able to explore more efficiently with negligible additional computational costs. After revewing the literature (Section 5), we present a thorough experimental evaluation over some simple 1D navigation domains, as well as some Mujoco (Todorov et al., 2012) tasks designed for exploration to assess the effect of uncertainty estimation and propagation on exploration and sample complexity (Section 6).

## 2. Preliminaries

**Markov Decision Processes** We consider infinite-horizon discounted Markov Decision Processes (MDP, Puterman, 2014). An MDP is a 5-tuple $\mathcal{M}=(\mathcal{S},\mathcal{A},\mathcal{P},\mathcal{R},\gamma)$, defined by the state space $\mathcal{S}$, the action space $\mathcal{A}$, a transition kernel $\mathcal{P}:\mathcal{S}\times\mathcal{A}\to\Delta(\mathcal{S})$, a reward $\mathcal{R}:\mathcal{S}\times\mathcal{A}\to\Delta(\mathbb{R})$ and the discount factor $\gamma\in[0,1)$.[1] Let $r:\mathcal{S}\times\mathcal{A}\to\mathbb{R}$ be the expectation of the reward $\mathcal{R}$ that we assume bounded in $[r_{\min},r_{\max}]$. The behavior of an agent is described by a policy $\pi:\mathcal{S}\to\Delta(\mathcal{A})$. When the agent observes state $s\in\mathcal{S}$, it chooses an action according to policy $a\sim\pi(s)$, executes it and observes a reward, $r\sim\mathcal{R}(s,a)$ and the next state sampled from the transition kernel $s'\sim\mathcal{P}(s,a)$. The performance of a policy $\pi$ is measured by its state-value function $V^\pi(s)=\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t r_t|\, s_0=s\right]$, where the expectation is taken w.r.t. to the stochasticity of the reward, the transition model, and the policy $\pi$. Similarly, the action-value function is defined as $Q^\pi(s,a)=\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t r_t|\, s_0=s,a_0=a\right]$, where we fix the first action and follow policy $\pi$ for the next steps. The value functions satisfy the Bellman equations, $V^\pi(s)=\mathbb{E}\left[r(s,a)+\gamma V^\pi(s')\right]$ for every $s\in\mathcal{S}$, $Q^\pi(s,a)=r(s,a)+\gamma\,\mathbb{E}\left[Q^\pi(s',a')\right]$ for every $(s,a)\in\mathcal{S}\times\mathcal{A}$. The optimal action-value function $Q^*$ is the maximum, over all policies, of the $Q$ function, for all state action pairs, $Q^*(s,a)=\sup_\pi\{Q^\pi(s,a)\}$. $Q^*$ satisfies the Bellman optimality equation $Q^*(s,a)=r(s,a)+\gamma\,\mathbb{E}\left[\sup_{a'\in\mathcal{A}}\{Q^*(s',a')\}\right]$. The Bellman equations form the basis of TD-learning, which updates the estimation of the $V$ or $Q$ functions in the current state using estimates of the next-states $V$ or $Q$ function. The goal of learning algorithms in this setting is to find the optimal policy $\pi^*$, which is defined as the policy that acts greedily w.r.t. $Q^*$, $\pi^*(\cdot|s)\in\Delta\left(\arg\max_{a\in\mathcal{A}}\{Q^*(s,a)\}\right),\forall s\in\mathcal{S}$.

[1] $\Delta(\mathcal{X})$ denotes the set of probability distributions over the set $\mathcal{X}$.

**Actor-Critic Methods** AC methods maintain a parameterized value-function $Q_{\boldsymbol{\omega}}$ (*critic*) to estimate the value of the current (or a given target) policy, and a parameterized policy $\pi_{\boldsymbol{\theta}}$ (*actor*), trained through gradient descent. In particular, SAC (Haarnoja et al., 2018), employs an *entropy-regularized* architecture. It maintains two parameterized action-value functions $\{Q_{\boldsymbol{\omega}_1}, Q_{\boldsymbol{\omega}_2}\}$ to estimate the entropy-regularized value function of policy $\pi_{\boldsymbol{\theta}}$. They are trained on the same samples and differ only on the initialization of $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$. The actor optimizes a "lower bound" of the action-value function, $Q_{LB}(s,a) = \min\{Q_{\boldsymbol{\omega}_1}(s,a), Q_{\boldsymbol{\omega}_2}(s,a)\}$. To update the critic, given a sample $(s,a,r,s')$, SAC uses the SARSA (Sutton & Barto, 2018) update rule, $Q_{\{\boldsymbol{\omega}_1,\boldsymbol{\omega}_2\}}(s,a) \leftarrow r + \gamma Q_{LB}(s',a')$, where $a' \sim \pi_{\boldsymbol{\theta}}(s')$. Specifically, SAC maintains experience collected with previous policies $\pi_{\boldsymbol{\theta}}$ in a *replay buffer* $\mathcal{D}$. The critic is trained to minimize the (entropy regularized) Bellman error over this replay buffer, as follows:

$$J_{\mathrm{C}}(\{\boldsymbol{\omega}_1,\boldsymbol{\omega}_2\}) = \mathop{\mathbb{E}}_{s,a,r,s' \sim \mathcal{D}} \Big[ (Q_{\{\boldsymbol{\omega}_1,\boldsymbol{\omega}_2\}}(s,a) - (r + \gamma \widetilde{Q}(s',a')))^2 \Big], \tag{1}$$

where $\widetilde{Q}(s,a) = \overline{Q}_{LB}(s,a) - \alpha \log \pi_{\boldsymbol{\theta}}(s',a')$, $\overline{Q}_{LB}(s,a) = \min\{Q_{\overline{\boldsymbol{\omega}}_1}(s,a), Q_{\overline{\boldsymbol{\omega}}_2}(s,a)\}$ is the lower bound of the Q-values given from two target networks which are updated slowly to improve stability (Mnih et al., 2015), $a' \sim \pi_{\boldsymbol{\theta}}(s')$, and $\alpha > 0$ specifies the level of entropy regularization. The actor network is trained to optimize an entropy-regularized objective. Since the target $Q$-function is a parameterized function approximator, the policy can directly follow the gradient of the critic:

$$J_A(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{s_t \sim \mathcal{D}, a_t \sim \pi_{\boldsymbol{\theta}}(s_t)} \left[ \log \pi_{\boldsymbol{\theta}}(s_t,a_t) - Q_{LB}(s_t,a_t) \right]. \tag{2}$$

**Wasserstein TD-Learning** Bayesian RL approaches (Dearden et al., 1998; Metelli et al., 2019) maintain, for each state-action pair $(s,a) \in \mathcal{S} \times \mathcal{A}$, a probability distribution $\mathcal{Q}(s,a)$, called a *Q-posterior*, used to represent the *epistemic* uncertainty over the value estimates. In practice, $\mathcal{Q}$ is an approximate distribution in a class $\mathscr{Q}$ (e.g., Gaussians). *V-Posteriors* can be defined, as in the classical way, as an average of the Q-posteriors, by employing the notion of *barycenters* defined in terms of a given divergence. As in (Metelli et al., 2019), we choose Wasserstein (Villani, 2008) divergence since the variance of our Q-posteriors vanishes as the number of samples grows to infinity, and the Wasserstein divergence allows computing the distance between distributions with disjoint support. Given two probability distributions, $\mu$ and $\nu$, the $L^p$-Wasserstein distance between $\mu$ and $\nu$ is defined as: $W_p(\mu,\nu) = (\inf_{\rho \in \Gamma(\mu,\nu)} \mathbb{E}_{X,Y \sim \rho}[d(X,Y)^p])^{1/p}$, where $\Gamma(\mu,\nu)$ is the set of all joint measures with marginals $\mu$ and $\nu$, and $d$ is a metric. Given a class of probability distributions $\mathscr{N}$, a set of probability distributions $\{\mu_i\}_{i=1}^n$, $\mu_i \in \mathscr{N}$ and a set of weights $\{\xi_i\}_{i=1}^n$, $\sum_{i=1}^n \xi_i = 1$ and $\xi_i \geq 0$, the $L^2$-Wasserstein barycenter is defined as (Agueh & Carlier, 2011) $\overline{\mu} \in \arg\min_{\mu \in \mathscr{N}} \{\sum_{i=1}^n \xi_i W_2(\mu,\mu_i)^2\}$.

**Algorithm 1** Wasserstein Actor-Critic.

**Input:** critic parameters $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2$, policy parameters $\boldsymbol{\theta}, \boldsymbol{\theta}^T$
Initialize $\mathcal{Q}_{\{1,2\}}(s,a)$ with the prior $\mathcal{Q}_0$
Initialize replay buffer $\mathcal{D} \leftarrow \emptyset$
**for** epoch $= 1, 2, \ldots$ **do**
  **for** $t = 1, 2, \ldots$ **do**
    Take action $a_t \sim \pi_{\boldsymbol{\theta}}(\cdot|s_t)$
    Observe $s_{t+1}$ and $r_{t+1}$
    $\mathcal{D} \leftarrow \mathcal{D} \bigcup (s_t, a_t, r_{t+1}, s_{t+1})$
  **end for**
  $\sigma_{\text{old}}^{\{1,2\}} \leftarrow \sigma_{\boldsymbol{\omega}_{\{1,2\}}}$
  **for** iteration $= 1, 2, \ldots$ **do**
    Update critic weights $\boldsymbol{\omega}_{\{1,2\}}$ using Equation (8)
    Update actor weights $\boldsymbol{\theta}$ ($\boldsymbol{\theta}^T$) using Equation (6) (Equation (7))
  **end for**
**end for**=0

Using the concept of Wasserstein barycenter, we can also propagate the uncertainty of the value function estimates across the state-action space. Indeed, having observed a transition $(s,a,r,s')$, the *Wasserstein Temporal Difference* (WTD, Metelli et al., 2019) update rule is defined via the computation of the barycenter of the current Q-posterior and the *TD- target posterior*, defined as $\mathcal{T}_t = r + \gamma \mathcal{Q}_t(s',a')$:

$$\mathcal{Q}_{t+1}(s,a) \in \mathop{\arg\min}_{\mathcal{Q} \in \mathscr{Q}} \Big\{ (1-\alpha_t) W_2(\mathcal{Q}, \mathcal{Q}_t(s,a))^2 + \alpha_t W_2(\mathcal{Q}, \mathcal{T}_t)^2 \Big\}, \tag{3}$$

where $\alpha_t$ is the learning rate, and $a'$ is the action taken in the next step. Depending on the policy chosen to take action $a'$ the update can be on-policy or off-policy. The presence of $\gamma$ in the definition of $\mathcal{T}_t$ shrinks the posteriors, vanishing the uncertainty when the number of samples grows to infinity. When the Q-posteriors become point estimates, the update rule reduces to the classic TD update rule. For some choices of distribution classes, Equation (3) can be computed in closed form. We will focus on Gaussian posteriors; this is not a limiting choice, as the sample mean is approximately Gaussian with enough samples, even if the return distribution is not Gaussian in the general case.

## 3. Wasserstein Actor-Critic

In this section, we introduce Wasserstein Actor-Critic (WAC), which extends WQL (Metelli et al., 2019) to handle environments with continuous-action spaces. We present the algorithm, define the update rules, and a regularization for the uncertainty estimates.

**Distributional Critic** Similar to Bayesian approaches,

WAC uses *distributional critics* to represent the epistemic uncertainty on the Q-value estimates. For each state-action pair $(s,a) \in \mathcal{S} \times \mathcal{A}$, we maintain an approximate distribution $Q(s,a) \sim \mathcal{Q}(s,a)$, representing a Q-posterior over the possible values of $Q(s,a)$ to model the uncertainty estimate on the value function. While these distributions will generally depend on the aleatoric uncertainty of the environment (state transition and reward), our updates will vanish the variance as we collect samples. This represents our main difference w.r.t. Distributional RL, as we do not require learning the whole return distribution, while still propagating uncertainty across the state-action space. More specifically, given a replay buffer of past behavior $\mathcal{D}$, our critic minimizes the $L_2$-Wasserstein distance between the Q-posterior $\mathcal{Q}_{\boldsymbol{\omega}}$ and the target posterior $r + \gamma \mathcal{Q}_{\overline{\boldsymbol{\omega}}}$, defined through the target parameters $\overline{\boldsymbol{\omega}}$ and target policy $\pi_{\boldsymbol{\theta}^T}$:

$$J_C(\boldsymbol{\omega}) = \mathop{\mathbb{E}}_{s,a,s',r\sim\mathcal{D}} \left[ W_2\left(\mathcal{Q}_{\boldsymbol{\omega}}(s,a), r + \gamma \mathcal{Q}_{\overline{\boldsymbol{\omega}}}(s', \pi_{\boldsymbol{\theta}^T}(s')\right)^2 \right]. \tag{4}$$

Like in the original WQL paper, different flavors of the algorithm can be proposed, based on the combination of: (i) distribution classes $\mathscr{Q}$, (ii) behavioral $\pi_{\boldsymbol{\theta}}$, and (iii) target $\pi_{\boldsymbol{\theta}^T}$ policies. In the discrete-action case, the exploration policy can follow an optimistic approach (maximizing an upper bound on the Q-values) or a posterior sampling approach (sampling actions with the probability of being optimal). However, when moving to continuous actions, even sampling from the distribution of the maximum requires computing a product integral (D'Eramo et al., 2017). In addition, we could only sample from the distribution approximately (e.g., through a Monte Carlo random walk over the action space). For this reason, we focus on the optimistic exploration, that just requires optimizing upper bounds. Finally, although other distribution classes, like particle-models, could be employed, we limit our discussion to Gaussian posteriors, as their parametrization allows for direct control over the distribution variance.

Similar to WQL, we maintain a parameterized distributional critic using a function approximator (e.g., neural network) that outputs the parameters of the distribution. For the Gaussian case, $Q(s,a) \sim \mathcal{N}(\mu_{\boldsymbol{\omega}}(s,a), \sigma_{\boldsymbol{\omega}}(s,a))$, the Wasserstein distance has a closed form, and the critic objective becomes:

$$J_C(\boldsymbol{\omega}) = \mathop{\mathbb{E}}_{s,a,s',r\sim\mathcal{D}} \Big[ \big(\mu_{\boldsymbol{\omega}}(s,a) - (r + \gamma \widetilde{\mu}_{\overline{\boldsymbol{\omega}}}(s', \pi_{\boldsymbol{\theta}^T}(s')))\big)^2 + \big(\sigma_{\boldsymbol{\omega}}(s,a) - \gamma \sigma_{\overline{\boldsymbol{\omega}}}(s', \pi_{\boldsymbol{\theta}^T}(s'))\big)^2 \Big], \tag{5}$$

where $\widetilde{\mu}_{\overline{\boldsymbol{\omega}}}(s,a) = \mu_{\overline{\boldsymbol{\omega}}}(s,a) - \alpha \log \pi_{\boldsymbol{\theta}^T}(s,a)$. In practice, $\mu_{\boldsymbol{\omega}}$ and $\sigma_{\boldsymbol{\omega}}$ can use either a shared network architecture or two different networks. Similar to WQL, we initialize the posterior networks using the bias of the last layer of the network. If the reward function is limited in the interval $[r_{\min}, r_{\max}]$, the Q values will be in the range $[q_{\min}, q_{\max}]$ with $q_{\min} = r_{\min}/(1-\gamma)$ and $q_{\max} = r_{\max}/(1-\gamma)$. We therefore initialize the uncertainty networks to $\sigma_0 = (q_{\max} - q_{\min})/\sqrt{12}$, i.e. the Gaussian minimizing the KL divergence with the uniform distribution in $[q_{\min}, q_{\max}]$.

**Actor** The actor in WAC is updated by optimizing an *upper bound* $U^{\delta}_{\boldsymbol{\omega}}$ of the estimated Q-value, which we can efficiently compute using Gaussian posterior: $U^{\delta}_{\boldsymbol{\omega}}(s,a) = \mu_{\boldsymbol{\omega}}(s,a) + \sigma_{\boldsymbol{\omega}}(s,a)\Phi^{-1}(\delta)$, where $\Phi^{-1}$ is the quantile function of the standard normal and $\delta \in (0,1)$. When actions are finite, no actor is needed, as we can compute the maximum by enumeration. However, in our case, we need an actor that follows $U^{\delta}_{\boldsymbol{\omega}}(s,a)$, which is differentiable in $\boldsymbol{\omega}$, leading to the minimization of the objective:

$$J_A(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{s_t\sim\mathcal{D}, a_t\sim\pi_{\boldsymbol{\theta}}(s_t)} \left[ \log \pi_{\boldsymbol{\theta}}(s_t,a_t) - U^{\delta}_{\boldsymbol{\omega}}(s_t,a_t) \right], \tag{6}$$

where $\boldsymbol{\theta}$ are the parameters of the behavioral policy. In practice, as discussed in Section 2, we employ a double critic strategy to improve stability, like SAC, i.e., we maintain two distributional critics and use the minimum of the two upper bounds, as target: $U^{\delta}_{\boldsymbol{\omega}}(s,a) = \min\{U^{\delta}_{\boldsymbol{\omega}_1}(s,a), U^{\delta}_{\boldsymbol{\omega}_2}(s,a)\}$.[2]

**Target Policy** We propose two alternatives for the target policy $\pi_{\boldsymbol{\theta}^T}$, corresponding to different estimators for the target posterior $\mathcal{T}_t$. First, we can use the same policy we use for exploration, i.e., $\boldsymbol{\theta} = \boldsymbol{\theta}^T$, like SAC. This has the advantage of not requiring a second parameterized policy. We call this version *Optimistic Estimator-WAC* (OE-WAC), which represents an on-policy algorithm. Alternatively, we can use a *greedy* policy that optimizes the expected value of the Q-posteriors (the mean critic $\mu_{\boldsymbol{\omega}}(s,a)$ in the Gaussian case). In this case, the target policy minimizes:

$$J_T(\boldsymbol{\theta}_T) = \mathop{\mathbb{E}}_{s_t\sim\mathcal{D}, a_t\sim\pi_{\boldsymbol{\theta}^T}(s_t)} \left[ \log \pi_{\boldsymbol{\theta}^T}(s_t,a_t) - \mu_{\boldsymbol{\omega}}(s_t,a_t) \right]. \tag{7}$$

We call this version *Mean Estimator-WAC* (ME-WAC). The best version to use between the two is task-dependent. Generally, OE-WAC is more suitable for environments that require large exploration, whereas ME-WAC is more suitable for simpler environments where OE-WAC might over-explore and might suffer from some instability.

## 4. Regularized Uncertainty Estimation

Our Q-posteriors are initialized to high uncertainty at the beginning of the learning process. Since they represent epistemic uncertainty, their variance will shrink as we observe more samples. This is apparent in Equation (5), where the targets $\sigma_{\boldsymbol{\omega}}$ are multiplied with $\gamma$. In tabular settings, the updates are localized, i.e., they affect a single state-action pair, without interfering with the others. However, when function

[2] It is worth noting that if both $U^{\delta}_{\boldsymbol{\omega}_1}$ and $U^{\delta}_{\boldsymbol{\omega}_2}$ are upper bounds of $Q_{\boldsymbol{\omega}}$, their minimum is still an upper bound.

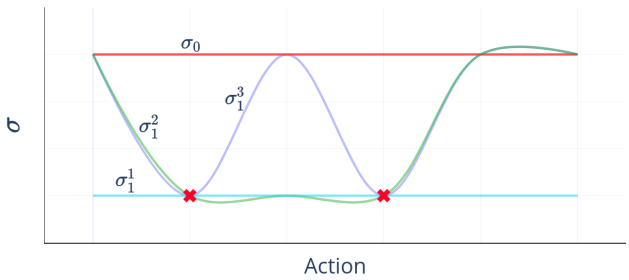



Figure 1: Example of uncertainty estimates. $\sigma_0$ shows the initial constant high value. The other curves show potential estimates after two collected samples (red crosses).

approximators are involved, generalizing uncertainty in an uncontrolled way might cause non-visited areas of the state-action space to take low uncertainty values, which might be undesired.[3] Consider the example in Figure 1. The graph shows the uncertainty estimate as a function of the action, in a fixed state $s$. Starting from an initial high constant estimate of $\sigma_0$, at the beginning of the learning process, we will observe samples like the red crosses in the figure, i.e., with lower uncertainty since it gets shrunk with $\gamma$. Among all the possible fitting lines, we would prefer an estimate like $\sigma_1^3$, which keeps high uncertainty in unseen regions, and would like to avoid failures like $\sigma_1^1$. This requires controlling the "smoothness" properties of the approximator. To avoid the additional computational burden, we propose a simple scheme based on *synthetic* samples. Specifically, we periodically save the weights of the uncertainty network $\sigma_{\text{old}}$ and use it as the target for state-action pairs drawn *uniformly* from the state-action space. More formally, our distributional critic minimizes:

$$J'_C(\boldsymbol{\omega}_{\{1,2\}}) = J_C(\boldsymbol{\omega}_{\{1,2\}}) + \lambda \mathop{\mathbb{E}}_{s,a\sim\mathcal{U}(\mathcal{S}\times\mathcal{A})} \left[ \left( \sigma_{\boldsymbol{\omega}_{\{1,2\}}}(s,a) - \sigma_{\text{old}}(s,a) \right)^2 \right], \tag{8}$$

where $J_C(\boldsymbol{\omega}_{\{1,2\}})$ is defined in Equation (5) and $\lambda \geq 0$ defines the relative weight of the regularization. Furthermore, in practice we add a second parameter, $\rho \in [0,1]$ which represents the fraction of *fake* samples (w.r.t. the samples used for $J_C(\boldsymbol{\omega}_{\{1,2\}})$) drawn for regularization. Specifically, if we estimate $J_C(\boldsymbol{\omega}_{\{1,2\}}$ using $N$ samples from replay buffer $\mathcal{D}$, we will estimate the expectation in Equation (8) with $M = \rho N$ samples from $\mathcal{U}(\mathcal{S}\times\mathcal{A})$. In Section 6, we will see that for simple environments, small values of $\rho$ are enough, while for more complex tasks the regularizer will require larger $\rho$ values. Algorithm 1 reports the pseudocode of WAC, embedding the regularized uncertainty estimation.

[3]This generalization phenomenon happens for the mean too, but, as visible in Equation (5), is particularly critical for the variance that gets updated with the next-state-action variance scaled by $\gamma < 1$.

To investigate the effectiveness of the regularized uncertainty loss, on an illustrative example, we trained two different agents, in a one-dimensional Linear Quadratic Regulator (LQG, Dorato et al., 2000). This task has a one-dimensional state and action spaces, which allow us to visualize the uncertainty estimates. As function approximator, we used a two-layer MLP (Bishop, 2006) of 128 neurons per layer. Figure 2 shows the comparison of the uncertainty estimation with and without the regularized uncertainty loss. The first row depicts the case without regularization (Equation 5) and the second row with regularization (Equation 8). On the left, we show the empirical state-action visitation distribution. The agent starts in one of the borders of the state space and has to reach the center in a few steps while calibrating the actions (there is a punishment for high actions). This is apparent in both left plots, with the highest densities in the borders and the center. We consider desirable to obtain uncertainty estimates that mirror these state-action densities, as the epistemic uncertainty is inversely proportional to the state-action visitation. While in both cases, the state-action densities are similar, the uncertainty estimates are completely different. In the top-right figure, we see the output of the uncertainty critic, without regularization, completely fails to represent the uncertainty. On the bottom-right corner, we can see that the regularized uncertainty critic, almost perfectly matches the state-action densities, with low uncertainty in the center and a gradual increase as we move away and then again a sharp decrease close to the borders. In Section 6 and Appendix B we show a more thorough investigation of the effect of the regularized uncertainty loss.

## 5. Related Works

There is a large body of literature studying efficient exploration techniques in RL. In the tabular settings, *provably efficient* methods have been devised, both in the model-based (Auer et al., 2008; Jaksch et al., 2010; Ian et al., 2013) and model-free (Jin et al., 2018; Strehl et al., 2006; Metelli et al., 2019) settings. These methods cannot be easily extended to the Deep RL setting, or when extensions are proposed, they lose their theoretical guarantees. In this section, we focus on tractable exploration methods proposed for Deep RL for continuous action spaces. Two main exploration frameworks exist: *uncertainty-based* methods and *intrinsic motivation* methods.

**Uncertainty-Based Methods** Classical value-based methods (including ACs) maintain a point estimate of the value functions for each state (or state-action pairs). Exploration policies, like $\epsilon$-greedy or Boltzmann (Sutton & Barto, 2018), add noise around the greedy action derived from these point estimates. These methods are not efficient, mainly because the exploration is not *directed* towards unvisited regions of

(a) **No uncertainty regularization**: state-action density (left) and uncertainty estimate (right).

(b) **With uncertainty regularization**: state-action density (left) and uncertainty estimate (right).

Figure 2: Comparison on the uncertainty estimates after training with and without uncertainty regularization in the LQG illustrative example (action on the $y$ axis and state on the $x$ axis).

the state space. The entropy regularization of SAC is a form of undirected exploration too, as the policies are trained to sacrifice some returns to preserve stochastic behavior. In recent years, several methods that move away from point estimates have been proposed. Ensemble methods (Chen et al., 2021; Wang et al., 2021) implicitly model the epistemic uncertainty of the Q-value estimates by maintaining multiple Q-function approximators. OAC (Ciosek et al., 2019) explicitly models the uncertainty on the value estimates by computing the variance of two critics, and it uses it to compute an exploration policy that optimizes an upper bound of the $Q$-values. Unfortunately, this uncertainty estimate is just heuristic and only stems from the disagreement between the two $Q$-networks with different initialization. Indeed, the networks are also trained with the same samples, and same target $Q$-values, so any disagreement is purely due to the random initialization only. Recently, SUNRISE (Lee et al., 2021) proposes a framework to unify ensemble methods for epistemic uncertainty estimation and shows considerable performance improvements in discrete and continuous action spaces. Distributional RL (Bellemare et al., 2017), on the other hand, models the *aleatoric* uncertainty, as its goal is to estimate the whole return distribution. First proposed for problems with a discrete action space (Bellemare et al., 2017; Dabney et al., 2018; Mavrin et al., 2019), it has been successfully extended also to the AC setting in TOP (Moskovitz et al., 2021). TOP models both aleatoric and epistemic uncertainty and adapts the level of optimism/pessimism by means of a Multi-Armed Bandit (MAB, Lattimore, 2020 - 2020) approach. While TOP deals with uncertainty propagation, it mixes the epistemic and aleatoric uncertainty while estimating the return distribution. To the best of our knowledge, WAC is the first method able to propagate epistemic uncertainty in continuous action spaces, without the need for estimating the full return distributions.

**Intrinsic Motivation** Tractable model-free methods based on intrinsic motivation have been proposed in recent years. Methods based on pseudo-counts (Bellemare et al., 2016; Ostrovski et al., 2017) assign exploration bonuses according to the novelty of the state-action pairs visited. While they have been applied with good results to deep architectures, they generally rely on (often pre-trained) density-models, which are not straightforward to maintain. Other methods apply exploration bonuses based on the state-action visitation density of the policy. MADE (Zhang et al., 2021) adds an exploration bonus, based on the deviation of the state visitation density of the new policy from the last observed policies. While it has been applied to continuous state-action spaces, it comes with a considerable computational cost to estimate the state densities and also requires pre-training of density models. State entropy maximiza-

tion (Mutti et al., 2021; Seo et al., 2021; Yarats et al., 2021) has also been applied as an incentive to explore the whole state-action space, including hard to reach regions. These methods generally scale better to continuous domains, as they do not explicitly need to estimate the state occupancy but only the entropy of this distribution. Numerous methods have also been proposed, with bonuses based on the information-gain (Houthooft et al., 2016; Achiam & Sastry, 2017; Pathak et al., 2019), but come with considerable computational costs to estimate these bonuses.

## 6. Experiments

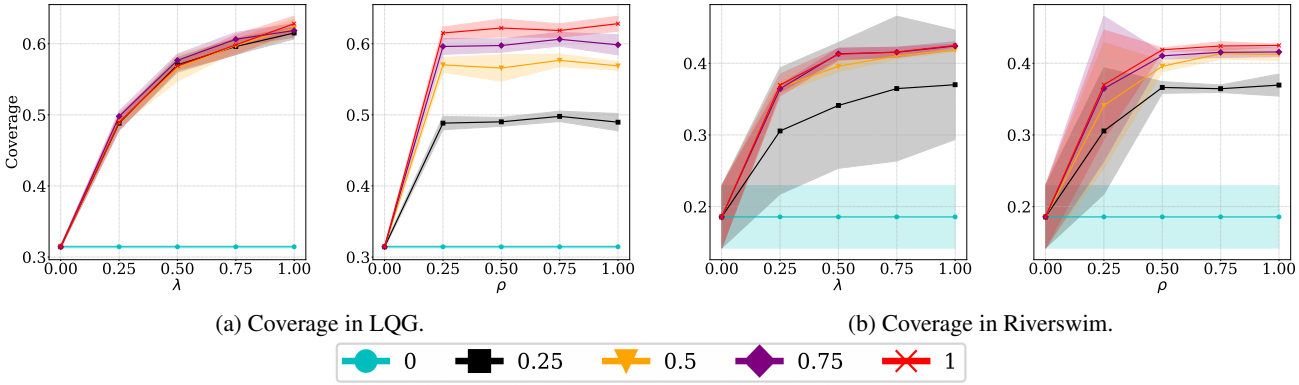
In this section, we present the empirical evaluation of WAC in various continuous control domains. We start from simple domains like LQG and a continuous version of Riverswim (Strehl & Littman, 2008), where we can better visualize the effects of the Q-posteriors in the learning and exploration process. Then we evaluate WAC on a set of Mujoco (Todorov et al., 2012) tasks designed for exploration.

**1D Navigation** The goal of this set of experiments is to measure the effect of uncertainty estimation on exploration. We keep track of the cumulative *coverage* of the state-action space, i.e., the portion of the total volume visited with relative frequency larger than $\epsilon > 0$. For this reason, we perform an empirical evaluation of WAC in two simple 1D navigation tasks. We consider a one-dimensional LQG, an environment with no particular exploration challenges, and a more challenging continuous-action version of the Riverswim chain, where long sequences of rewardless actions are needed to reach high reward states. A full description of the environments is reported in Appendix A.

Figure 3 shows the results of these experiments. For each environment, we train WAC, varying the parameters $\lambda$ and $\rho$ of the regularized uncertainty loss in Equation (8). For each value, we report the coverage averaged over all training epochs. Firstly, we observe that both parameters directly control the amount of exploration. Indeed, the coverage is monotonically increasing with both $\lambda$ and $\rho$. As expected, low values of $\rho$ cause higher variance, as fewer samples are employed to estimate the uncertainty regularization. This can be seen in all the curves of the leftmost plot, as well as in the third plot, where the black curve corresponding to $\rho = 0.25$ suffers from a high variance. In Appendix B we perform a similar study for OAC, varying the parameters $\beta$ and $\delta$ which control exploration, and we observe that the coverage is not so easily controllable with these parameters. We attribute this to the heuristic nature of the uncertainty estimation of OAC, based on the critics' disagreement only.

**2D Navigation** To assess whether a principled uncertainty estimation and propagation translate into lower sample complexity, we perform an empirical evaluation in a set of Mujoco (Todorov et al., 2012) tasks, where the amount of exploration needed to solve the task can be controlled. We start from the 2D navigation task used in (Moro et al., 2022), where the agent has to reach a goal state in a 2D world, by avoiding obstacles. The reward is the negative Euclidean distance from the goal state. While this is a *dense reward*, the obstacle presence generates local optima which the agent needs to overcome by exploring efficiently. We progressively make the task more challenging by adding additional walls and obstacles. A visual representation of the tasks is shown in Figure 4a. We leave the full description of the environments in Appendix A . We name the tasks as "Point $x$" with $x \in \{1, 2, 3, 4\}$, where a higher $x$ means a more difficult exploration challenge. We compare the performance of WAC, in both versions defined in Section 3, with SAC and OAC. In each task, we track the cumulative return, as well as the number of episodes completed in a fixed number of steps (higher is better). We use the implementation of SAC and OAC used in (Ciosek et al., 2019), and extend the repository with our implementation, to guarantee comparable results. The same network architectures are used for all algorithms. For the common hyperparameters, we only tune SAC and use the same values for WAC and OAC, by additionally tuning the algorithm specific parameters ($\delta$ and $\beta$ for OAC and $\lambda$ and $\rho$ for WAC). Details on the hyperparameter tuning are in Appendix A .

In Figure 4b, we present the average return as a function of the training epochs, whereas in Figure 4c we present the number of episodes completed in 3000 steps of interaction. Starting from left to right, we increase the difficulty of the task. We can see that for the easiest task, all algorithms are able to find the optimal policy of quickly avoiding an obstacle in the middle to reach the goal state, even though SAC learns slower compared to the others. Being a simple exploration task, ME-WAC performs better and is more stable than OE-WAC. OAC is also able to quickly solve the task. While the difference in return is negligible, the number of completed episodes shows an advantage for ME-WAC, which completes more episodes faster. Finally, we underline that even though the task does not require particular exploration, WAC does not over-explore, but rather solves with a speed comparable with the other baselines. The clear advantages of WAC in terms of exploration can be seen starting from the second task, where the exploration requirements are increased. Both versions of WAC learn faster and with less variance compared with both SAC and OAC. The difference is even more apparent in the number of episodes completed, where SAC and WAC have disjoint confidence intervals. In the third task, SAC completely fails in learning to reach the goal, while OAC succeeds in some of the seeds only, showing a high variance. WAC, on the other hand, outperforms them both in terms of return and




(a) Coverage in LQG.

(b) Coverage in Riverswim.

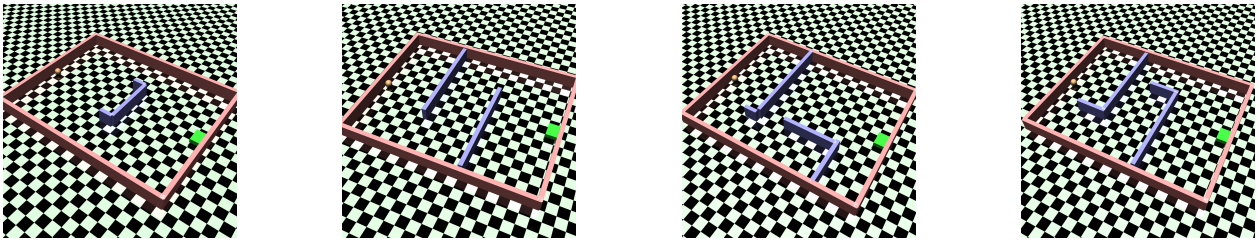
Figure 3: Coverage in LQG and Riverswim as function of $\lambda$ and $\rho$; average of 5 seeds, 95% c.i..

completed episodes. ME-WAC performs better, even though the task requires a good amount of exploration. Compared to ME-WAC, OE-WAC over-explores and it shows a slower learning curve. The last task is solved by the WAC agents only. SAC and OAC never reach the goal state. WAC outperforms them, in both versions, with statistical significance. We also see the need for larger exploration, apparent from the difference in performance between OE-WAC and ME-WAC.

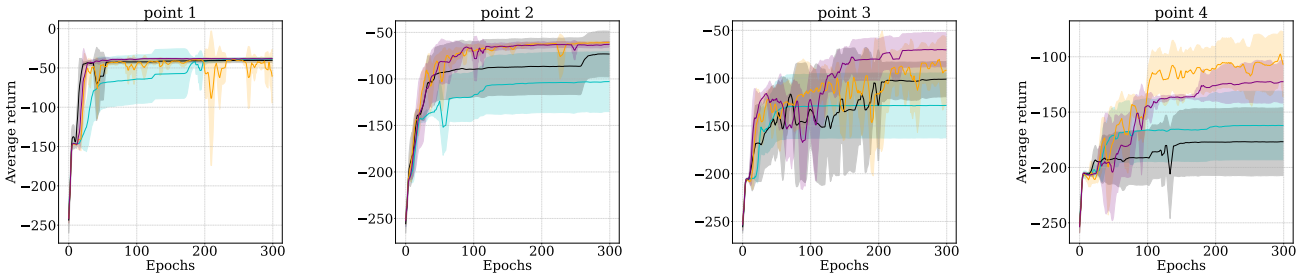
Finally, Figure 4d presents the number of episodes completed in a *sparse reward* version of the same tasks. In this scenario, we do not show the return as it is proportional to the number of completed episodes. We only trained OE-WAC agents in these tasks, as they present a substantial exploration challenge. The advantage of WAC is extremely evident in these tasks. SAC and OAC are only able to solve the simplest task. In a sparse reward setting, SAC and OAC will only explore randomly so they fully rely on the chance of reaching the goal state with random actions. OAC explores more compared to SAC, but since the exploration does not depend on the state-action visitations, but only on the disagreement between the critics, sparse reward tasks are a great challenge. WAC, instead, will still explore, even when facing sparse rewards since the uncertainty will gradually decline in visited regions, so the upper bounds will favor reaching unvisited ones. Indeed, OE-WAC outperforms both baselines in all the tasks with sparse rewards.

## 7. Conclusions

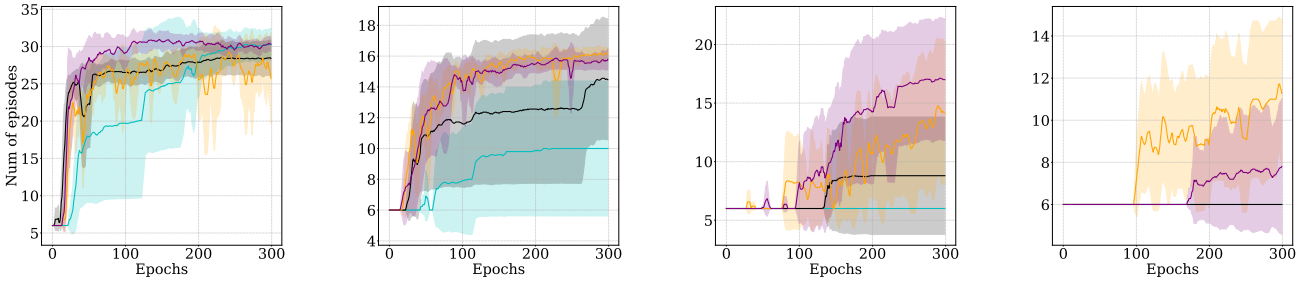
In this paper, we presented a novel AC algorithm to perform directed exploration. We presented WAC, which extends the recently proposed WQL to the continuous-actions case. Furthermore, we addressed a problem of uncertainty estimation that arises when using function approximation, related to the generalization of the uncertainty estimates. We proposed a simple, yet effective, regularization method, based on synthetic samples that allowed us to better generalize the uncertainty across the state-action space. Finally, we performed a thorough empirical evaluation to investigate the advantages of performing a principled uncertainty estimation and propagation in continuous-action domains. We observed that the uncertainty estimates of WAC can effectively steer exploration towards promising regions of the state-action space, even under sparse rewards, especially when comparing it with heuristic uncertainty estimation based on ensemble methods. Future work includes extending the current method to posterior sampling exploration strategies.



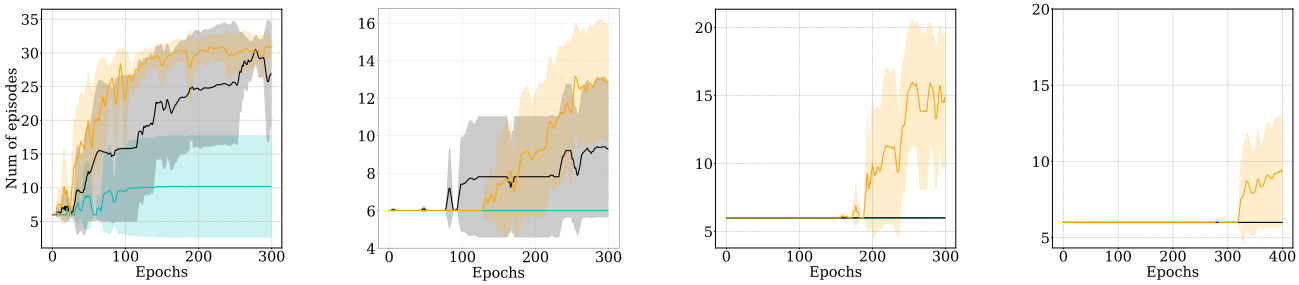
(a) Visual representation of the 4 2D navigation tasks.




(b) Average return in 4 2D navigation tasks.




(c) Number of episodes completed in 3000 steps in 4 2D navigation tasks.




(d) Number of episodes completed in 3000 steps in 4 2D navigation tasks with sparse reward function.

SAC OAC OE-WAC ME-WAC

Figure 4: Experimental results in 4 2D navigation tasks starting from the easiest (left) to the hardest (right); average of 5 seeds, 95% c.i..

## References


Achiam, J. and Sastry, S. Surprise-based intrinsic motivation for deep reinforcement learning. *CoRR*, abs/1703.01732, 2017. URL http://arxiv.org/abs/1703.01732.

Agueh, M. and Carlier, G. Barycenters in the wasserstein space. *SIAM Journal on Mathematical Analysis*, 43(2): 904–924, 2011.

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

Auer, P., Jaksch, T., and Ortner, R. Near-optimal regret bounds for reinforcement learning. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L. (eds.), *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2008. URL https://proceedings.neurips.cc/paper/2008/file/e4a6222cdb5b34375400904f03d8e6a5-Paper.pdf.

Bellemare, M. G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 1479–1487, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

Bellemare, M. G., Dabney, W., and Munos, R. A distributional perspective on reinforcement learning. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 449–458. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/bellemare17a.html.

Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

Chen, X., Wang, C., Zhou, Z., and Ross, K. W. Randomized ensembled double q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=AY8zfZm0tDd.

Ciosek, K., Vuong, Q., Loftin, R., and Hofmann, K. Better exploration with optimistic actor critic. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/a34bacf839b923770b2c360eefa26748-Paper.pdf.

Dabney, W., Rowland, M., Bellemare, M. G., and Munos, R. Distributional reinforcement learning with quantile regression. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018. ISBN 978-1-57735-800-8.

Dearden, R., Friedman, N., and Russell, S. J. Bayesian q-learning. In Mostow, J. and Rich, C. (eds.), *Proceedings of the Fifteenth National Conference on Artificial Intelligence and Tenth Innovative Applications of Artificial Intelligence Conference, AAAI 98, IAAI 98, July 26-30, 1998, Madison, Wisconsin, USA.*, pp. 761–768. AAAI Press / The MIT Press, 1998.

Dorato, P., Cerone, V., and Abdallah, C. *Linear quadratic control: an introduction*. Krieger Publishing Co., Inc., 2000.

D'Eramo, C., Nuara, A., Pirotta, M., and Restelli, M. Estimating the maximum expected value in continuous reinforcement learning problems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), Feb. 2017. URL https://ojs.aaai.org/index.php/AAAI/article/view/10887.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

Houthooft, R., Chen, X., Duan, Y., Schulman, J., De Turck, F., and Abbeel, P. Vime: Variational information maximizing exploration. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 1117–1125, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

Ian, O., Benjamin, V. R., and Daniel, R. (more) efficient reinforcement learning via posterior sampling. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pp. 3003–3011, Red Hook, NY, USA, 2013. Curran Associates Inc.

Jaksch, T., Ortner, R., and Auer, P. Near-optimal regret bounds for reinforcement learning. *J. Mach. Learn. Res.*, 11:1563–1600, aug 2010. ISSN 1532-4435.

Jin, C., Allen-Zhu, Z., Bubeck, S., and Jordan, M. I. Is q-learning provably efficient? In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 4868–4878. Curran Associates, Inc., 2018.

Lattimore, T. *Bandit algorithms / Tor Lattimore (deepMind) and Csaba Szepesvari (University of Alberta).* Cambridge University Press, Cambridge, United Kingdom ;, 2020 - 2020. ISBN 9781108486828.

Lee, K., Laskin, M., Srinivas, A., and Abbeel, P. Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6131–6141. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/lee21g.html.

Mavrin, B., Yao, H., Kong, L., Wu, K., and Yu, Y. Distributional reinforcement learning for efficient exploration. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4424–4434. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/mavrin19a.html.

Metelli, A. M., Likmeta, A., and Restelli, M. Propagating uncertainty in reinforcement learning via wasserstein barycenters. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/f83630579d055dc5843ae693e7cdafe0-Paper.pdf.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518 (7540):529–533, February 2015. ISSN 00280836. URL http://dx.doi.org/10.1038/nature14236.

Moro, L., Likmeta, A., Prati, E., and Restelli, M. Goal-directed planning via hindsight experience replay. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=6NePxZwfae.

Moskovitz, T., Parker-Holder, J., Pacchiano, A., Arbel, M., and Jordan, M. Tactical optimism and pessimism for deep reinforcement learning. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 12849–12863. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/6abcc8f24321d1eb8c95855eab78ee95-Paper.pdf.

Mutti, M., Pratissoli, L., and Restelli, M. Task-agnostic exploration via policy gradient of a non-parametric state entropy estimate. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 9028–9036, 2021.

O'Donoghue, B., Osband, I., Munos, R., and Mnih, V. The uncertainty Bellman equation and exploration. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3839–3848, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Ostrovski, G., Bellemare, M. G., van den Oord, A., and Munos, R. Count-based exploration with neural density models. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 2721–2730. JMLR.org, 2017.

Pathak, D., Gandhi, D., and Gupta, A. Self-supervised exploration via disagreement. *CoRR*, abs/1906.04161, 2019. URL http://arxiv.org/abs/1906.04161.

Puterman, M. L. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/schulman15.html.

Seo, Y., Chen, L., Shin, J., Lee, H., Abbeel, P., and Lee, K. State entropy maximization with random encoders for efficient exploration. In *International Conference on Machine Learning*, pp. 9443–9454. PMLR, 2021.

Strehl, A. L. and Littman, M. L. An analysis of model-based interval estimation for markov decision processes. *J. Comput. Syst. Sci.*, 74(8):1309–1331, 2008. doi: 10.1016/j.jcss.2007.08.009.

Strehl, A. L., Li, L., Wiewiora, E., Langford, J., and Littman, M. L. PAC model-free reinforcement learning. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, pp. 881–888, 2006.

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

Szepesvári, C. The asymptotic convergence-rate of q-learning. In *Proceedings of the 10th International Conference on Neural Information Processing Systems*, NIPS'97, pp. 1064–1070, Cambridge, MA, USA, 1997. MIT Press.

Thompson, W. R. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *IROS*, pp. 5026–5033. IEEE, 2012. ISBN 978-1-4673-1737-5. URL http://dblp.uni-trier.de/db/conf/iros/iros2012.html#TodorovET12.

Villani, C. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Wang, H., Lin, S., and Zhang, J. Adaptive ensemble q-learning: Minimizing estimation bias via error feedback. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 24778–24790. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/cfa45151ccad6bf11ea146ed563f2119-Paper.pdf.

Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. Reinforcement learning with prototypical representations. In *International Conference on Machine Learning*, pp. 11920–11931. PMLR, 2021.

Zhang, T., Rashidinejad, P., Jiao, J., Tian, Y., Gonzalez, J. E., and Russell, S. MADE: Exploration via maximizing deviation from explored regions. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=DTVfEJIL3DB.

# A. Experimental Details

## A.1. Environments Description

**RiverSwim** We extend the classical Riverswim domain (Strehl & Littman, 2008) to a continuous setting. In this environment, the agent has to navigate a 1 dimensional state space, ranging from 0 to $max_state$, by applying a 1 dimensional action, representing the intended movement $a \in [-1 \quad 1]$. The initial state is a uniformly distributed in $[0 \quad 0.5]$. When an action is chosen the agent moves left or right on the state space. The distance of the movement is equal to the absolute value of the action (if the result is outside the state space a clip operations brings it back inside). The direction $d \in \{-1, 0, 1\}$ of the movement is stochastic, according to the following probabilities:

$$P(d=-1|a) = \begin{cases} 1-0.9\cdot(a+1) & \text{if } a \leq 0 \\ 0.1 & \text{if } a > 0 \end{cases}$$

$$P(d=0|a) = \begin{cases} 0.9\cdot(a+1) & \text{if } a \leq 0 \\ 0.9-0.3a & \text{if } a > 0 \end{cases}$$

$$P(d=1|a) = \begin{cases} 0 & \text{if } a \leq 0 \\ 0.3a & \text{if } a > 0 \end{cases}$$

Given the current state $s_t$, the action $a_t$ and the direction $d_t$ sampled according the previous probabilities, the next state is $s_{t+1} = \text{clip}(s_t + d_t|a_t|)$, where clip clips the state in the range $[0 \quad max_state]$ The reward depends on the starting state $s$ and the action sign:

$$r = \begin{cases} 5\cdot 10^{-4} & \text{if } s \leq 1 \\ 1 & \text{if } s \geq (max_state-1) \text{ and } a > 0 \\ 0 & \text{otherwise} \end{cases}$$

The optimal policy is to always perform $a=1$ which gives the agent the best chance of moving toward high reward states.

In our experiments:

- $max_state = 25$

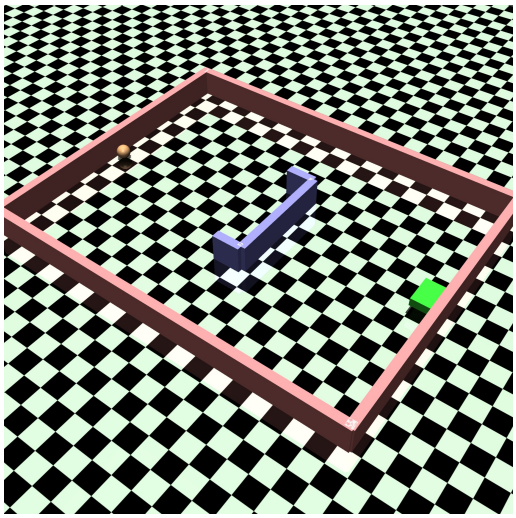
**LQG** We test our agents also on an instance of a Linear Quadratic Gaussian control. Given a state $x$, an action $a$, and $v \sim \mathcal{N}(0, 0.5)$ the next state and the cost $c$ $(=-r)$ are defined as:

$$x' = Ax + Ba + v$$
$$c = Qx^2 + Ra^2$$

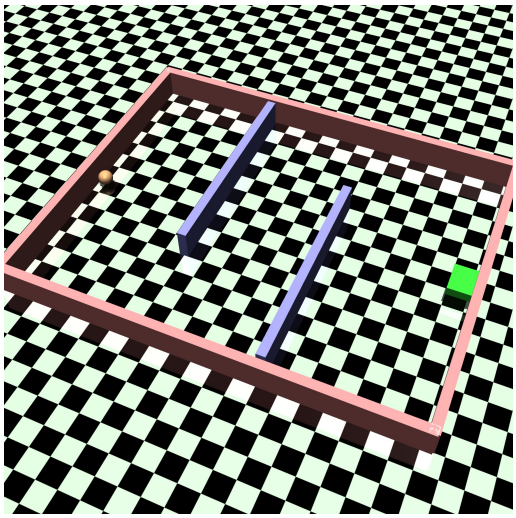
In our experiments, we use $A=1, B=1, Q=0.9, R=0.9$. The agents starts from the borders of the state space and, with this configuration, the goal is to reach the origin of the state space while balancing the actions.

**Point** This environment models a sphere moving inside a two-dimensional maze. The goal of the agent is to get close enough to a goal state on the right side of the maze, while avoiding obstacles. Once the agent gets close enough to the goal (Euclidean distance $<2$) the episode ends. The state space includes the agent position in the 2-dimensional space, as well as the velocities. The action space is also 2-dimensional, controlling the actuators in both directions. We use 2 reward functions for this task. In the dense reward version of the environment, the reward is the negative Euclidean distance of the sphere from the center of the goal. This represents a dense reward signal, which makes optimization easier but also introduces local maximums due to the presence of the obstacles. In the sparse version the reward is always $-1$, so the optimal policy is to reach the goal as quickly as possible so that the episodes ends. We devise 4 environment configuarations, with different levels of difficulty.

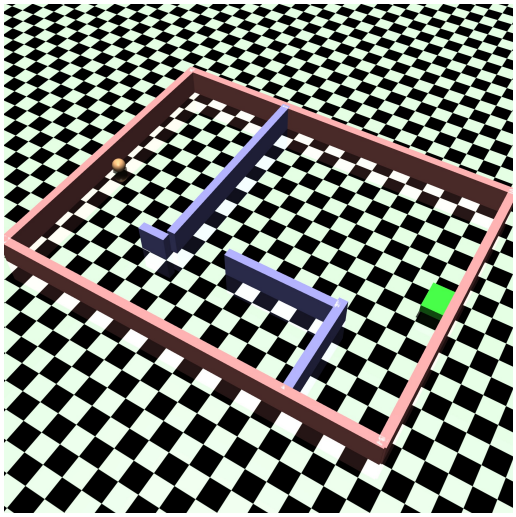
The first environment (Figure 5a) has a U-shaped wall in the middle. The agent has to overcome it by either running into it and then moving back to outflank it, so it can escape the local maximum, or preferably, it has to go around it without



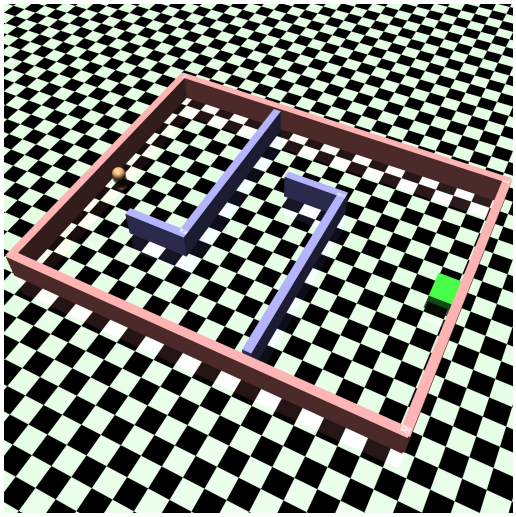
(a) Point 1



(b) Point 2



(c) Point 3



(d) Point 4

touching it. The U-shaped of the wall makes the task more difficult since the agent is unlikely to be able to escape the local maximum once it has hit the wall by simply playing random actions.

The second environment (Figure 5b) has 2 simpler obstacles to overcome compared to first environment, yet the agent now has to learn how to overcome two obstacles which overall makes the second task more difficult than the first one.

To overcome the obstacle of the third environment (Figure 5c) the agent has to perform a rather complete exploration of the space since the gateway to the second room is quite narrow and it is easy to be stuck in a local maximum at the border due to the reward function based on the Euclidean Distance.

The forth and last environment shown in Figure 5d, puts together the challenges of the previous two, it has 2 obstacles to overcome and it also requires a rather complete exploration of the space to advance to the goal.

### A.2. Tuning procedures

In this appendix, we present the hyperparameter tuning employed, as well as the final values used in our experiments. For all the approximators employed, including critics and actors, we use 2 layer MLPs. For what concerns LQG all algorithms could solve it easily independently from the hyper-parameters chosen. The same can be said for RiverSwim, except for SAC which could not solve for any set of hyper-parameters contained in the grid search we performed.

In point environment we performed a grid search on all environments with dense rewards starting with SAC (3 runs with 3

different seeds for each node of the grid). We choose the set of hyper-parameters that could solve the most runs for the two most difficult environments in which at least one run could learn a policy that reached the goal. The best recovered values are reported in Table 1.

Table 1: SAC parameters

| Parameter | best value |
|---|---|
| networks' number of layers | 2 |
| layers' size | 256 |
| replay buffer size | $10^6$ |
| number of train steps per train loop | 1000 |
| number of exploration steps per train loop | 1000 |
| batch size | 256 |
| learning rates | $10^{-3}$ |

Afterwards, we performed a grid search on OAC and WAC, where we fixed all the hyper-parameters they share with SAC to the best values we found on SAC hyper-parameter tuning and we tuned only on their additional hyper-parameters. Once again, we choose the hyper-parameters sets that allow each algorithm to perform best the 2 most difficult environment it could solve. The final values are reported in Table 2 and Table 3.

Table 2: OAC parameters

| Parameter | best value |
|---|---|
| $\delta$ | 18 |
| $\beta_{UB}$ | 6.5 |

Table 3: WAC parameters

| Parameter | best value |
|---|---|
| $\delta$ | 0.95 |
| $\lambda$ | 0.6 |
| $\rho$ | 0.6 |

# B. Additional Experiments

## B.1. Tuning on $\delta$

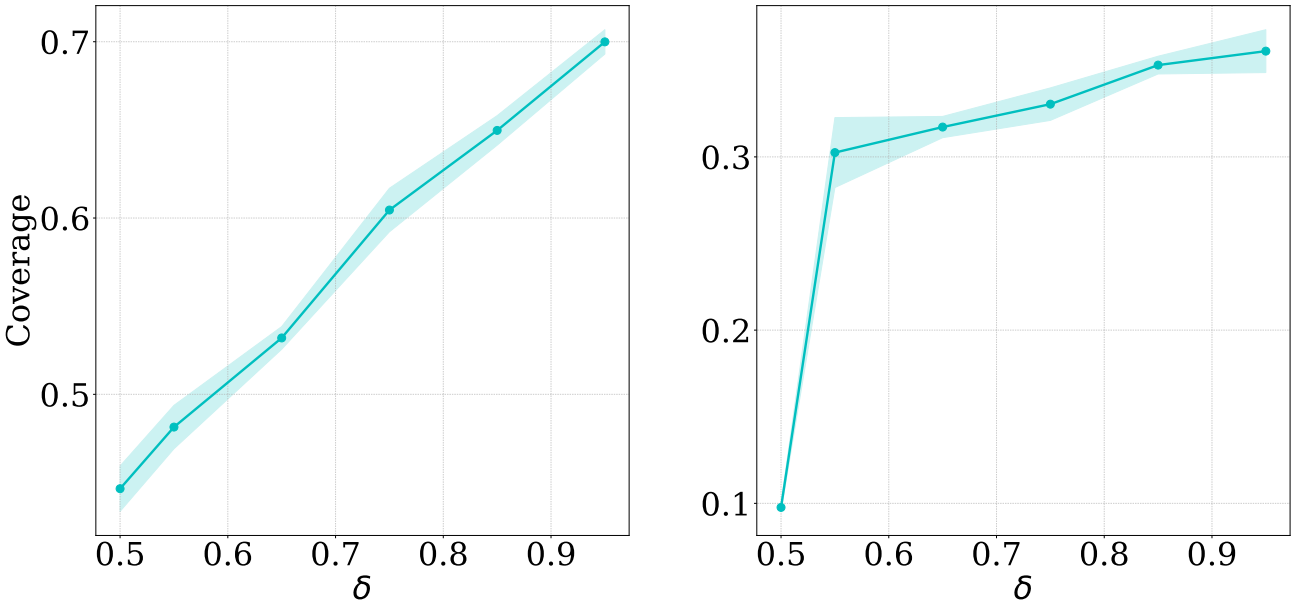
In Section 6 we have shown how by tuning $\lambda$ and $\rho$ we can control the amount of exploration. We now show some experiment that illustrate how the hyper parameter $\delta$ can also be use to control exploration, since it defines what percentile of the estimated Gaussian distribution we use as upper bound. The results are reported in Figure 6. We observe that also $\delta$ is directly related with the coverage. Indeed by increasing $\delta$, we employ larger upper bounds for the value estimates, and this directly translates to larger coverage of the state-action space.

## B.2. Coverage in OAC

We performed a similar study to the one presented in Section 6 on WAC for OAC, to investigate whether we could control how much the algorithm explores based on the values of the hyper-parameters. However, what we found is that is hard to predict OAC's exploration based on its hyper parameters $\delta$ and $\beta_{UB}$. In OAC, $\delta$ controls how much the exploration policy differs from the target policy. From Figure 7, we can see that $\delta$ can even negatively affect exploration, if we allow the exploration policy to differ too much from the target policy. The dependence on $\beta$, which controls the definition of the upper bound (similar to our $\delta$ in OAC), suggests that the uncertainty estimate of OAC is not directly related to exploration either. We attribute both these results to the heuristic estimation of uncertainty that OAC employees, based only on the disagreement between the two critics. We argue that this uncertainty estimation is not enough to direct exploration meaningfully.

## B.3. Exploration Heatmaps in Point

In this section, we show some additional heatmaps which represents the visited states (we have ignored velocities so we could visualize the location of the agent) over 300 epochs of all algorithms we have tested throughout the paper. Figure 8 shows the heatmaps from runs on Point 3 environment, with dense rewards. In Figure 8a we see that SAC cannot get past the first wall and does not explore the space around the local maximum enough to reach other maxima. In Figure 8b we




(a) Coverage in LQG.

(b) Coverage in Riverswim.

Figure 6: Coverage in LQG and Riverswim as function of $\delta$; average of 5 seeds, 95% c.i..

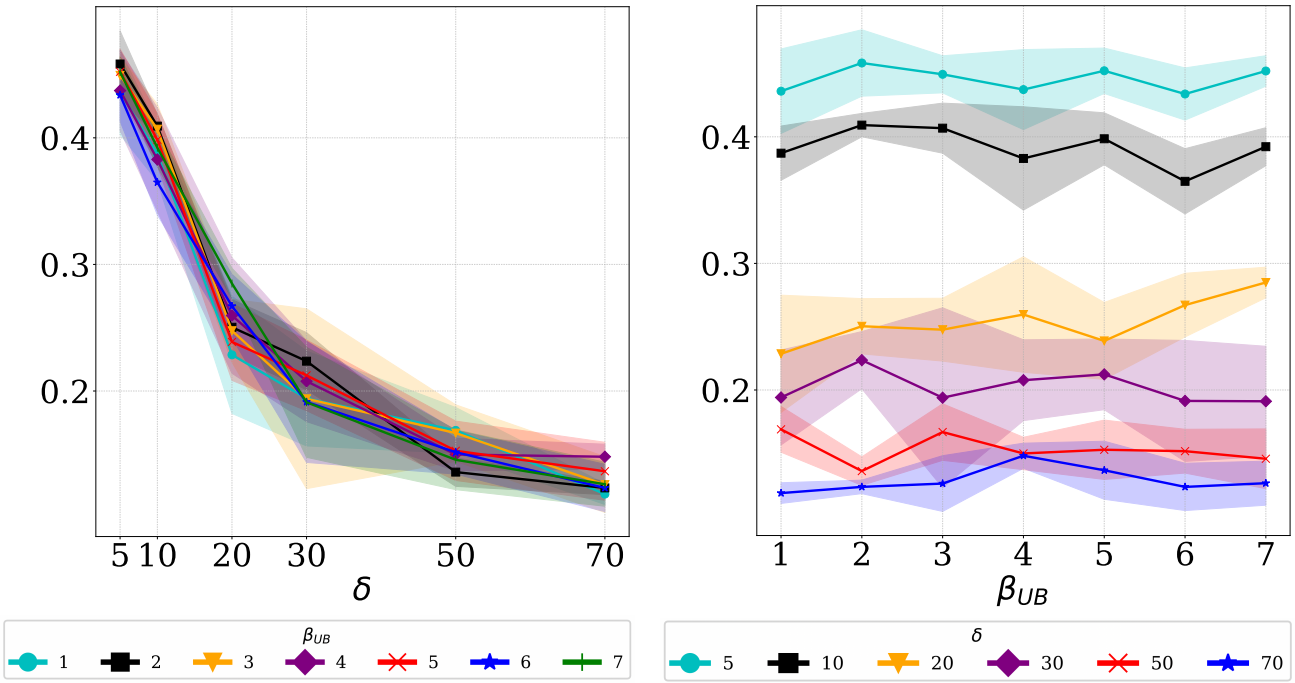
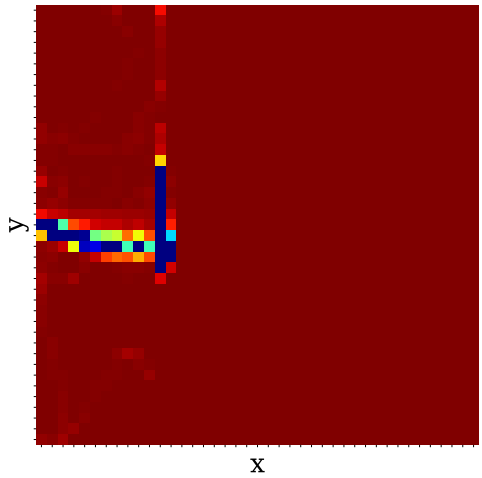


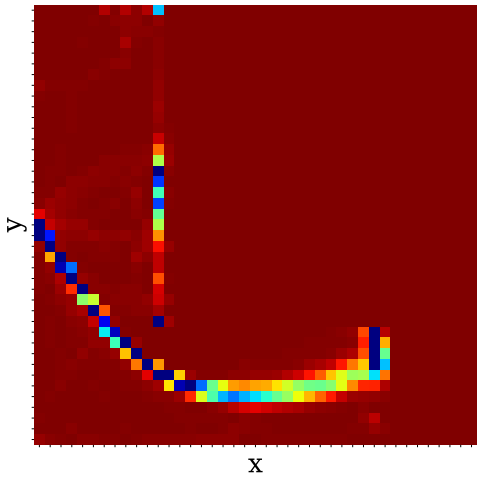

Figure 7: Coverage in LQG as function of $\delta$ and $\beta_{UB}$; average of 5 seeds, 95% c.i..

see that OAC finds a better maximum but still a local one. WAC does find the same local maximum as OAC, in fact we can see in Figure 8c it visits it many times, yet once the uncertainty estimate is low enough it is able to keep exploring and ultimately reach the goal. We have also reported in Figure 8d the same heatmap created by the target policy which follows the critic of the mean instead of the upper bound.

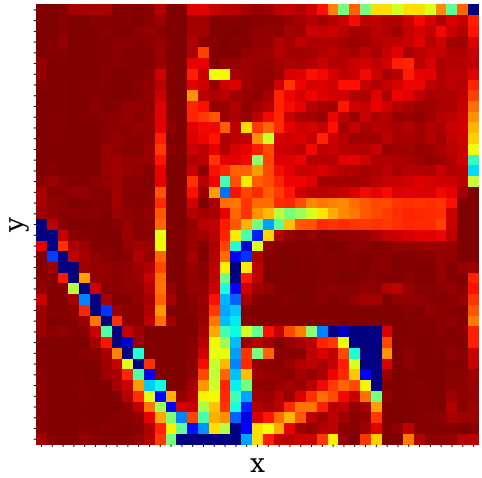
Figure 9 shows the cumulative visited states in the Point 2 environment with sparse reward. We can observe that SAC, having no uncertainty estimate and no informative rewards, mostly explores around the starting states with very simple policies that follow straight lines. OAC is able to reach areas of the maze which are further away from the starting point but it still can't reach the goal. Finally, WAC manages to reach the goal of the maze and explores almost every area of the it.



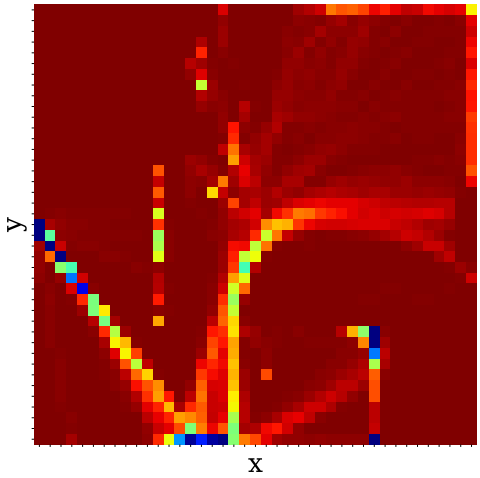

(a) SAC




(b) OAC exploration policy

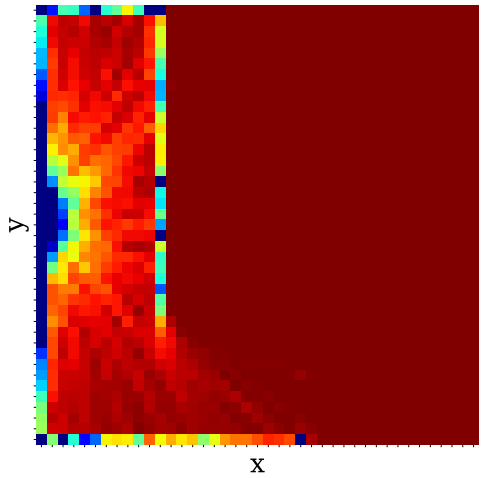



(c) WAC exploration policy

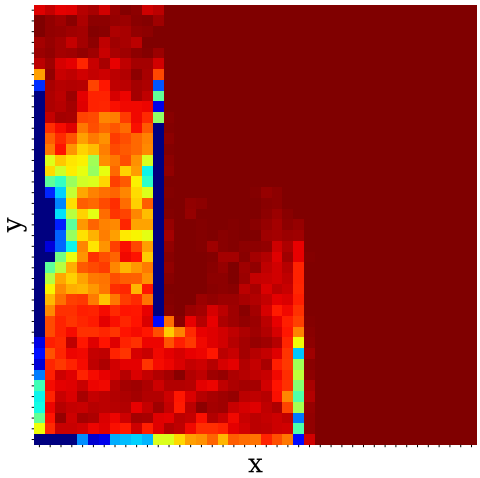



(d) WAC evaluation policy

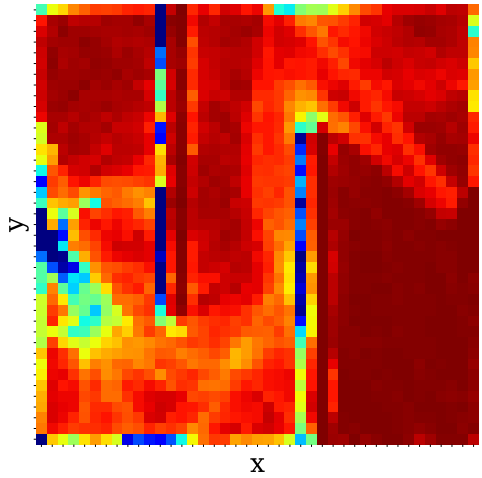
Figure 8: Cumulative visited states in 300 epochs in Point 3 environment (Dense Reward)



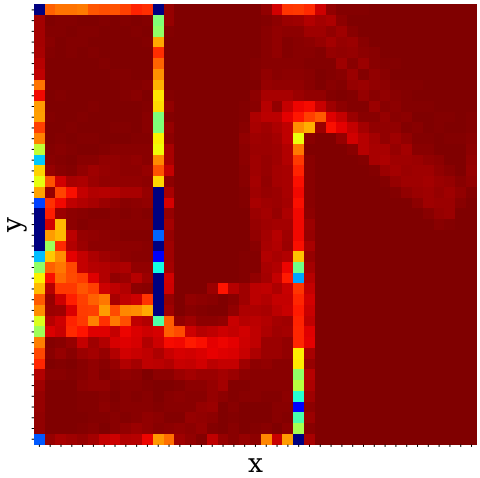

(a) SAC




(b) OAC exploration policy




(c) WAC exploration policy




(d) WAC target policy

Figure 9: Cumulative visited states in 300 epochs in Point 2 environment (Sparse Reward)