LEARNING TO EXPLORE A CLASS OF MULTIPLE REWARD-FREE ENVIRONMENTS

Mirco Mutti*
Politecnico di Milano
Università di Bologna

Mattia Mancassola*
Politecnico di Milano

Marcello Restelli
Politecnico di Milano

ABSTRACT

Several recent works have been dedicated to the pure exploration of a single reward-free environment. Along this line, we address the problem of learning to explore a class of multiple reward-free environments with a unique general strategy, which aims to provide a universal initialization to subsequent reinforcement learning problems specified over the same class. Notably, the problem is inherently multi-objective as we can trade off the exploration performance between environments in many ways. In this work, we foster an exploration strategy that is sensitive to the most adverse cases within the class. Hence, we cast the exploration problem as the maximization of the mean of a critical percentile of the state visitation entropy induced by the exploration strategy over the class of environments. Then, we present a policy gradient algorithm, MEMENTO, to optimize the introduced objective through mediated interactions with the class. Finally, we empirically demonstrate the ability of the algorithm in learning to explore challenging classes of continuous environments and we show that reinforcement learning greatly benefits from the pre-trained exploration strategy when compared to learning from scratch.

1 INTRODUCTION

The typical Reinforcement Learning (RL, Sutton & Barto, 2018) setting involves a learning agent interacting with an environment in order to maximize a reward signal. In principle, the reward signal is a given and perfectly encodes the task. In practice, the reward is usually hand-crafted, and designing it to make the agent learn a desirable behavior is often a huge challenge. This poses a serious roadblock on the way of autonomous learning, as any task requires a costly and specific formulation, while the synergy between solving one RL problem and another is very limited. To address this crucial limitation, Jin et al. (2020) have formulated the *reward-free* RL problem. In this setting, the agent is tasked with mastering an environment without rewards, so that the knowledge it acquires can be eventually deployed to solve any RL problem one could specify in this same environment. *Mastering* the environment has been formulated in (i) *learning to model* its transition dynamics (Jin et al., 2020; Tarbouriech et al., 2020b;a; Zhang et al., 2020b), or (ii) *learning to explore* it with a general, task-agnostic, strategy (Hazan et al., 2019). Although they overcome the reliance on a reward function, previous solutions to reward-free RL are still severely environment-specific.

In this work, we aim to push the generality of reward-free learning even further by addressing the problem of *learning to explore multiple environments* with a single exploration strategy. Especially, the agent faces a class of reward-free environments that belong to the same domain but differ in their transition dynamics. At each turn of the learning process, the agent is drawn into an environment within the class, where it can interact for a finite number of steps before facing another turn. The process carries on sequentially for a finite number of turns. The ultimate goal of the agent is to learn an exploration strategy that helps to solve any subsequent RL task specified over the class.

Our contribution to the problem is three-fold: (c1) We frame it into a tractable *formulation* (Section 3), (c2) we propose a *methodology* to address it (Section 4), for which (c3) we provide a thorough *empirical* evaluation (Section 5). First, we extend a reward-free objective meant for environment-specific exploration, which is the *Maximum State Visitation Entropy* (MSVE, Hazan et al., 2019).

*These authors contributed equally. Correspondence to mirco.mutti@polimi.it.

The underlying intuition is that a general exploration strategy has to visit with high probability any state where the agent might be rewarded in a subsequent RL task. When dealing with multiple environments, this becomes a *multi-objective* problem, as one could establish any combination of preferences over the environments. In this work, instead of naïvely optimizing the mean of the state visitation entropy across the class, we consider its Conditional Value-at-Risk (CVaR, Rockafellar et al., 2000) to prioritize performance in particularly rare or adverse environments. We propose a policy gradient algorithm (Deisenroth et al., 2013), *Multiple Environments Maximum ENTropy Optimization* (MEMENTO), to optimize the learning objective via mere interactions with the class of environments. As in recent works (Mutti et al., 2020; Liu & Abbeel, 2021; Seo et al., 2021), the algorithm employs non-parametric methods to deal with state entropy estimation in continuous and high-dimensional environments. Then, it leverages these estimated values to optimize the CVaR of the entropy by following its policy gradient (Tamar et al., 2015). Finally, we provide an experimental analysis of the proposed method in a two-stage setting. First, we show that it is effective in training general exploration strategies over classes of continuous and high-dimensional environments without rewards. Second, we test the obtained exploration strategies as initialization for a number of RL tasks defined over the same class. Notably, the trained exploration strategy allows us to solve sparse-rewards tasks that are impractical to learn from scratch, while being robust to the most unfavorable environment thanks to the CVaR objective. A preliminary theoretical analysis is in Section 6, Related works are in Appendix A, the proofs of the theorems are in Appendix B. The implementation of MEMENTO can be found at `https://github.com/muttimirco/memento`.

2 Preliminaries

In this section we report background and notation. A vector $\boldsymbol{v}$ is in bold, and v_i is its i-th entry.

Probability and Percentiles Let X be a random variable distributed according to a cumulative density function (cdf) $F_X(x) = Pr(X \leq x)$. We denote with $\mathbb{E}[X]$, $\mathbb{V}\mathrm{ar}[X]$ the expected value and the variance of X respectively. Let $\alpha \in (0, 1)$ be a confidence level, we call the α-percentile (shortened to α%) of the variable X its Value-at-Risk (VaR), which is defined as $\mathrm{VaR}_\alpha(X) = \inf \big\{ x \mid F_X(x) \geq \alpha \big\}$. Analogously, we call the mean of this same α-percentile the Conditional Value-at-Risk (CVaR) of X, $\mathrm{CVaR}_\alpha(X) = \mathbb{E}\big[X \mid X \leq \mathrm{VaR}_\alpha(X) \big]$.

Markov Decision Processes A Controlled Markov Process (CMP) is a tuple $\mathcal{M} := (\mathcal{S}, \mathcal{A}, P, D)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, the transition model $P(s'|a, s)$ denotes the conditional probability of reaching state $s' \in \mathcal{S}$ when selecting action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$, and D is the initial state distribution. The behavior of an agent is described by a policy $\pi(a|s)$, which defines the probability of taking acion $a \in \mathcal{A}$ in $s \in \mathcal{S}$. Let Π be the set of all the stationary policies. Executing a policy $\pi \in \Pi$ in a CMP over a time horizon T generates a trajectory $\tau = (s_{0,\tau}, a_{0,\tau}, \dots, a_{T-2,\tau}, s_{T-1,\tau})$ with probability $p_{\pi,\mathcal{M}}(\tau) = D(s_{0,\tau}) \prod_{t=0}^{T-1} \pi(a_{t,\tau}|s_{t,\tau}) P(s_{t+1,\tau}|s_{t,\tau}, a_{t,\tau})$. We denote the state-visitation frequencies induced by τ with $d_\tau(s) = \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{1}(s_{t,\tau} = s)$, and we call $d_\pi^{\mathcal{M}} = \mathbb{E}_{\tau \sim p_{\pi,\mathcal{M}}}[d_\tau]$ the marginal state distribution. We define the differential entropy (Shannon, 1948) of d_τ as $H(d_\tau) = -\int_{\mathcal{S}} d_\tau(s) \log d_\tau(s) \,\mathrm{d}s$. For simplicity, we will write $H(d_\tau)$ as a random variable $H_\tau \sim \delta(h - H(d_\tau)) p_{\pi,\mathcal{M}}(\tau)$, where $\delta(h)$ is a Dirac delta.

By coupling a CMP $\mathcal{M}$ with a reward function R we obtain a Markov Decision Process (MDP, Puterman, 2014) $\mathcal{M}^R := \mathcal{M} \cup R$. Let $R(s, a)$ be the expected immediate reward when taking $a \in \mathcal{A}$ in $s \in \mathcal{S}$ and let $R(\tau) = \sum_{t=0}^{T-1} R(s_{t,\tau})$, the *performance* of a policy π over the MDP $\mathcal{M}^R$ is defined as

$$\mathcal{J}_{\mathcal{M}^R}(\pi) = \mathop{\mathbb{E}}_{\tau \sim p_{\pi,\mathcal{M}}} \big[R(\tau) \big]. \tag{1}$$

The goal of reinforcement learning Sutton & Barto (2018) is to find an optimal policy $\pi^*_{\mathcal{J}} \in \arg\max \mathcal{J}_{\mathcal{M}^R}(\pi)$ through sampled interactions with an unknown MDP $\mathcal{M}^R$.

3 Learning to Explore Multiple Environments

Let $\boldsymbol{\mathcal{M}} = \{\mathcal{M}_1, \dots, \mathcal{M}_I\}$ be a class of unknown CMPs, in which every element $\mathcal{M}_i = (\mathcal{S}, \mathcal{A}, P_i, D)$ has a specific transition model P_i, while $\mathcal{S}, \mathcal{A}, D$ are homogeneous across the class.

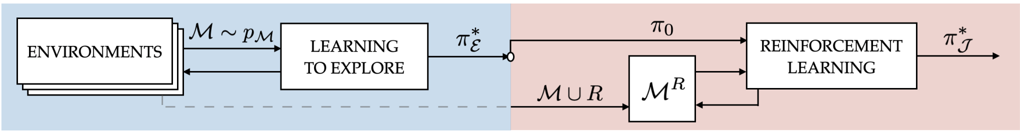


Figure 1: Illustration of the two-phase learning problem. On the left, we highlight the process of learning to explore multiple environments. The obtained exploration policy $\pi^*_{\mathcal{E}}$ conveys a pre-trained initialization to the subsequent RL process (right), which outputs a reward maximizing policy $\pi^*_{\mathcal{J}}$ for any MDP $\mathcal{M}^R$.

At each turn, the agent is able to interact with a single environment $\mathcal{M} \in \boldsymbol{\mathcal{M}}$. The selection of the environment to interact with is mediated by a distribution $p_{\boldsymbol{\mathcal{M}}}$ over $\boldsymbol{\mathcal{M}}$, outside the control of the agent. The aim of the agent is to learn an exploration strategy that is general across all the MDPs $\mathcal{M}^R$ one can build upon $\boldsymbol{\mathcal{M}}$. In a single-environment setting, this problem has been assimilated to learning a policy that maximizes the entropy of the induced state visitation frequencies (Hazan et al., 2019; Lee et al., 2019; Mutti et al., 2020). One can straightforwardly extend the objective to multiple environments by considering the expectation over the class of CMPs, $\mathcal{E}_{\boldsymbol{\mathcal{M}}}(\pi) = \mathbb{E}_{\substack{\mathcal{M} \sim p_{\boldsymbol{\mathcal{M}}} \\ \tau \sim p_{\pi,\mathcal{M}}}} \big[H_\tau \big]$, where the usual entropy objective over the single environment $\mathcal{M}_i$ can be easily recovered by setting $p_{\mathcal{M}_i} = 1$. However, this objective function does not account for the tail behavior of H_τ, i.e., for the exploration performance in environments of $\boldsymbol{\mathcal{M}}$ that are rare or particularly unfavorable. This is decidedly undesirable as the agent may be tasked with an MDP built upon one of these adverse environments in the subsequent RL phase, where even an optimal strategy w.r.t. $\mathcal{E}_{\boldsymbol{\mathcal{M}}}(\pi)$ may fail to provide sufficient exploration. To overcome this limitation, we look for a more nuanced exploration objective that balances the expected performance with the sensitivity to the tail behavior. By taking inspiration from the risk-averse optimization literature (Rockafellar et al., 2000), we consider the CVaR of the state visitation entropy induced by π over $\boldsymbol{\mathcal{M}}$,

$$\mathcal{E}^\alpha_{\boldsymbol{\mathcal{M}}}(\pi) = \mathrm{CVaR}_\alpha(H_\tau) = \mathop{\mathbb{E}}_{\substack{\mathcal{M} \sim p_{\boldsymbol{\mathcal{M}}} \\ \tau \sim p_{\pi,\mathcal{M}}}} \big[H_\tau \mid H_\tau \leq \mathrm{VaR}_\alpha(H_\tau) \big], \tag{2}$$

where α is a confidence level and $\mathcal{E}^1_{\boldsymbol{\mathcal{M}}}(\pi) := \mathcal{E}_{\boldsymbol{\mathcal{M}}}(\pi)$. The lower we set the value of α, the more we hedge against the possibility of a bad exploration outcome in some $\mathcal{M} \in \boldsymbol{\mathcal{M}}$. In the following sections, we propose a method to effectively learn a policy $\pi^*_{\mathcal{E}} \in \arg\max \mathcal{E}^\alpha_{\boldsymbol{\mathcal{M}}}(\pi)$ through mere interactions with $\boldsymbol{\mathcal{M}}$, and we show how this serves as a pre-training for RL (Figure 1).

4 A Policy Gradient Approach

In this section, we present an algorithm, called *Multiple Environments Maximum ENTropy Optimization* (MEMENTO), to optimize the exploration objective in (2) through mediated interactions with a class of continuous environments.

MEMENTO operates as a typical policy gradient approach (Deisenroth et al., 2013). It directly searches for an optimal policy by navigating a set of parametric differentiable policies $\Pi_\Theta := \{\pi_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^n\}$. It does so by repeatedly updating the policy parameters $\boldsymbol{\theta}$ in the gradient direction, until a stationary point is reached. This update has the form

$$\boldsymbol{\theta}' = \boldsymbol{\theta} + \beta \nabla_{\boldsymbol{\theta}} \mathcal{E}^\alpha_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}),$$

where β is a learning rate, and $\nabla_{\boldsymbol{\theta}} \mathcal{E}^\alpha_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})$ is the gradient of (2) w.r.t. $\boldsymbol{\theta}$. The following proposition provides the formula of $\nabla_{\boldsymbol{\theta}} \mathcal{E}^\alpha_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})$. The derivation follows closely the one in (Tamar et al., 2015, Proposition 1), which we have adapted to our objective function of interest (2).

Algorithm 1 MEMENTO

```
Input: percentile $\alpha$, learning rate $\beta$
Output: policy $\pi_{\boldsymbol{\theta}}$
initialize $\boldsymbol{\theta}$
for epoch $= 0, 1, \dots,$ until convergence do
    for $i = 1, 2, \dots, N$ do
        sample an environment $\mathcal{M}_i \sim p_{\boldsymbol{\mathcal{M}}}$
        sample a trajectory $\tau_i \sim p_{\pi_{\boldsymbol{\theta}}, \mathcal{M}_i}$
        estimate $H_{\tau_i}$ with (3)
    end for
    estimate $\mathrm{VaR}_\alpha(H_\tau)$ with (4)
    estimate $\nabla_{\boldsymbol{\theta}} \mathcal{E}^\alpha_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})$ with (5)
    update parameters $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \beta \widehat{\nabla}_{\boldsymbol{\theta}} \mathcal{E}^\alpha_{\mathcal{M}}(\pi_{\boldsymbol{\theta}})$
end for
```

Proposition 4.1. *The policy gradient of the exploration objective $\mathcal{E}^\alpha_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})$ w.r.t. $\boldsymbol{\theta}$ is given by*

$$\nabla_{\boldsymbol{\theta}} \mathcal{E}^\alpha_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) = \mathop{\mathbb{E}}_{\substack{\mathcal{M} \sim p_{\boldsymbol{\mathcal{M}}} \\ \tau \sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}}}} \Bigg[\bigg(\sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_{t,\tau} | s_{t,\tau}) \bigg) \Big(H_\tau - \mathrm{VaR}_\alpha(H_\tau) \Big) \Big| H_\tau \leq \mathrm{VaR}_\alpha(H_\tau) \Bigg].$$

However, in this work we do not assume full knowledge of the class of CMPs $\boldsymbol{\mathcal{M}}$, and the expected value in Proposition 4.1 cannot be computed without having access to $p_{\boldsymbol{\mathcal{M}}}$ and $p_{\pi_{\boldsymbol{\theta}},\mathcal{M}}$. Instead, MEMENTO computes the policy update via a Monte Carlo estimation of $\nabla_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}$ from the sampled interactions $\{(\mathcal{M}_i, \tau_i)\}_{i=1}^N$ with the class of environments $\boldsymbol{\mathcal{M}}$. The policy gradient estimate itself relies on a Monte Carlo estimate of each entropy value H_{τ_i} from τ_i, and a Monte Carlo estimate of $\mathrm{VaR}_\alpha(H_\tau)$ given the estimated $\{H_{\tau_i}\}_{i=1}^N$. The following paragraphs describe how these estimates are carried out, while Algorithm 1 provides the pseudocode of MEMENTO. Additional details and implementation choices can be found in Appendix C.

Entropy Estimation We would like to compute the entropy H_{τ_i} of the state visitation frequencies d_{τ_i} from a single realization $\{s_{t,\tau_i}\}_{t=0}^{T-1} \subset \tau_i$. This estimation is notoriously challenging when the state space is continuous and high-dimensional $\mathcal{S} \subseteq \mathbb{R}^p$. Taking inspiration from recent works pursuing the MSVE objective (Mutti et al., 2020; Liu & Abbeel, 2021; Seo et al., 2021), we employ a principled k-Nearest Neighbors (k-NN) entropy estimator (Singh et al., 2003) of the form

$$\widehat{H}_{\tau_i} \propto -\frac{1}{T} \sum_{t=0}^{T-1} \log \frac{k\,\Gamma(\frac{p}{2}+1)}{T\,\left\|s_{t,\tau_i} - s_{t,\tau_i}^{k\text{-NN}}\right\|^p \pi^{\frac{p}{2}}}, \tag{3}$$

where Γ is the Gamma function, $\|\cdot\|$ is the Euclidean distance, and $s_{t,\tau_i}^{k\text{-NN}} \in \tau_i$ is the k-nearest neighbor of s_{t,τ_i}. The intuition behind the estimator in (3) is straightforward: We can suppose the state visitation frequencies d_{τ_i} to have a high entropy as long as the average distance between any encountered state and its k-NN is large. Despite its simplicity, a Euclidean metric suffices to get reliable entropy estimates in continuous control domains (Mutti et al., 2020).

VaR Estimation and Baseline The last missing piece to get a Monte Carlo estimate of the policy gradient $\nabla_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}$ is the value of $\mathrm{VaR}_\alpha(H_\tau)$. Being $H_{[1]}, \dots, H_{[N]}$ the order statistics out of the estimated values $\{\widehat{H}_{\tau_i}\}_{i=1}^N$, we can naïvely estimate the VaR as

$$\widehat{\mathrm{VaR}}_\alpha(H_\tau) = H_{[\lceil \alpha N \rceil]}. \tag{4}$$

Albeit asymptotically unbiased, the VaR estimator in (4) is known to suffer from a large variance in finite sample regimes (Kolla et al., 2019), which is aggravated by the error in the upstream entropy estimates, which provide the order statistics. This variance is mostly harmless when we use the estimate to filter out entropy values beyond the $\alpha\%$, i.e., the condition $H_\tau \leq \mathrm{VaR}_\alpha(H_\tau)$ in Proposition 4.1. Instead, its impact is significant when we subtract it from the values within the $\alpha\%$, i.e., the term $H_\tau - \mathrm{VaR}_\alpha(H_\tau)$ in Proposition 4.1. To mitigate this issue, we consider a convenient baseline $b = -\mathrm{VaR}_\alpha(H_\tau)$ to be subtracted from the latter, which gives the Monte Carlo policy gradient estimator

$$\widehat{\nabla}_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) = \sum_{i=1}^N f_{\tau_i}\ \widehat{H}_{\tau_i}\ \mathbb{1}(\widehat{H}_{\tau_i} \leq \widehat{\mathrm{VaR}}_\alpha(H_\tau)), \tag{5}$$

where $f_{\tau_i} = \sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_{t,\tau_i}|s_{t,\tau_i})$. Notably, the baseline b trades off a lower estimation error for a slight additional bias in the estimation (5). We found that this baseline leads to empirically good results and we provide some theoretical corroboration over its benefits in Appendix C.1.

5 Empirical Evaluation

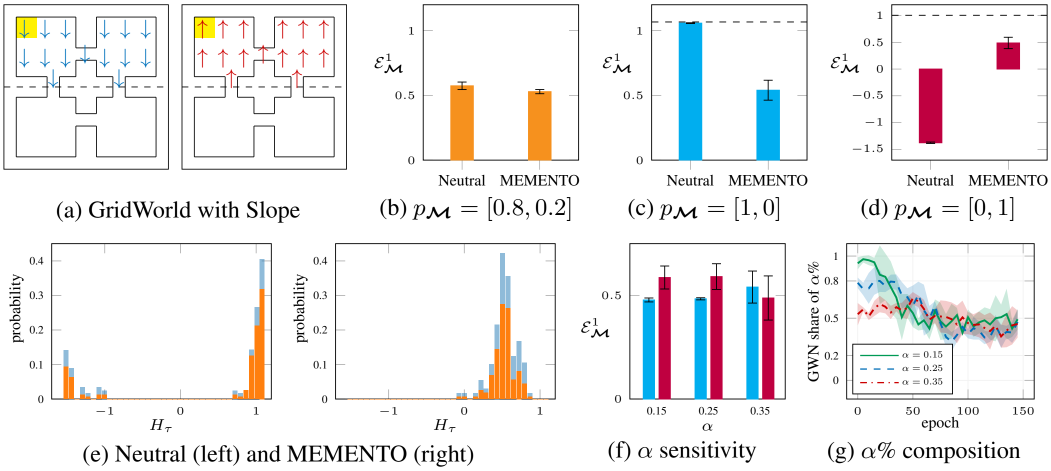
In this section, we provide an extensive empirical evaluation of the proposed methodology over the two-phase learning process described in Figure 1, which is organized as follows:

5.1 We show the ability of our method in learning to explore a class of illustrative continuous gridworlds, emphasizing the importance of the percentile sensitivity;

5.2 We discuss how the choice of the percentile of interest affects the exploration strategy;

5.3 We highlight the benefit that the exploration strategy provides to RL on the same class;

5.4 We verify the ability of our method to scale with the size of the class of environments, by considering a class of 10 continuous gridworlds;



(a) GridWorld with Slope
(b) $p_{\mathcal{M}} = [0.8, 0.2]$
(c) $p_{\mathcal{M}} = [1, 0]$
(d) $p_{\mathcal{M}} = [0, 1]$
(e) Neutral (left) and MEMENTO (right)
(f) α sensitivity
(g) $\alpha\%$ composition

Figure 2: Exploration performance $\mathcal{E}^1_{\mathcal{M}}$ obtained by MEMENTO ($\alpha = 0.35$) and Neutral ($\alpha = 1$) in the *GridWorld with Slope* domain **(a)**. The polices are trained on **(b)** and tested on **(b, c, d)**. The dashed lines in **(c, d)** represent the optimal performance. The empirical distribution having mean in **(b)** is reported in **(e)**. The behaviour of MEMENTO with different α is reported in **(f, g)**, where **(f)** reports the exploration performance and **(g)** the share of GWN trajectories in the $\alpha\%$. For every plot, we provide 95% c.i. over 4 runs.

5.5 We verify the ability of our method to scale with the dimensionality of the environments in the class, by considering a class of 29D continuous control Ant domains;

5.6 We verify the ability of our method to scale with visual inputs, by considering a class of 147D MiniGrid (Chevalier-Boisvert et al., 2018) domains;

5.7 We show that the exploration strategy learned with our approach is superior for RL w.r.t. a policy meta-trained with MAML (Finn et al., 2017; Gupta et al., 2018a) on the same class.

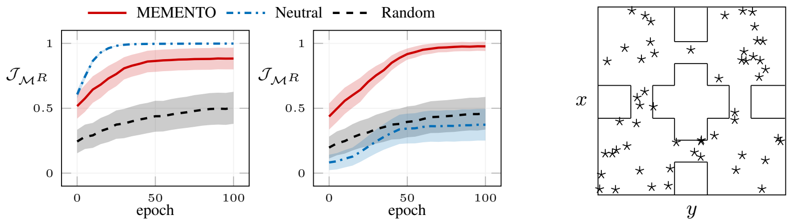
A thorough description of the experimental setting is provided in Appendix D.

5.1 Learning to Explore Multiple Environments with Percentile Sensitivity

In this section, we consider a class $\mathcal{M}$ composed of two different configurations of a continuous gridworld domain with 2D states and 2D actions, which we call the *GridWorld with Slope*. In each configuration, the agent navigates through four rooms connected by narrow hallways, by choosing a (bounded) increment along the coordinate directions. A visual representation of the setting can be found in Figure 2a, where the shaded areas denote the initial state distribution and the arrows render a slope that favors or contrasts the agent's movement. The configuration on the left has a south-facing slope, and thus it is called GridWorld with South slope (GWS). Instead, the one on the right is called GridWorld with North slope (GWN) as it has a north-facing slope. This class of environments is unbalanced (and thus interesting to our purpose) for two reasons: First, the GWN configuration is more challenging from a pure exploration standpoint, since the slope prevents the agent from easily reaching the two bottom rooms; secondly, the distribution over the class is also unbalanced, as it is $p_{\mathcal{M}} = [Pr(\text{GWS}), Pr(\text{GWN})] = [0.8, 0.2]$. In this setting, we compare MEMENTO against *Neutral*, which is a simplified version of MEMENTO with $\alpha = 1$,[1] to highlight the importance of percentile sensitivity w.r.t. a naïve approach to the multiple environments scenario. The methods are evaluated in terms of the state visitation entropy $\mathcal{E}^1_{\mathcal{M}}$ induced by the exploration strategies they learn.

In Figure 2, we compare the performance of the optimal exploration strategy obtained by running MEMENTO ($\alpha = 0.35$) and Neutral ($\alpha = 1$) for 150 epochs on the GridWorld with Slope class ($p_{\mathcal{M}} = [0.8, 0.2]$). We show that the two methods achieve a very similar expected performance over the class (Figure 2b). However, this expected performance is the result of a (weighted) average of very different contributions. As anticipated, Neutral has a strong performance in GWS ($p_{\mathcal{M}} = [1, 0]$, Figure 2c), which is close to the configuration-specific optimum (dashed line), but it displays a bad showing in the adverse GWN ($p_{\mathcal{M}} = [0, 1]$, Figure 2d). Conversely, MEMENTO learns a strategy that is much more robust to the configuration, showing a similar performance in GWS and GWN, as

[1] The pseudocode is identical to Algorithm 1 except that all trajectories affect the gradient estimate in (5).



(a) RL on GWS (left) and GWN (right)

(b) Sampled goals

Figure 3: Average return $\mathcal{J}_{\mathcal{M}^R}$ as a function of learning epochs achieved by TRPO initialized with MEMENTO ($\alpha = 0.35$), Neutral ($\alpha = 1$), and random exploration strategies, when dealing with a set of RL tasks specified on the *GridWorld with Slope* domain **(a)**. We provide 95% c.i. over 50 randomly sampled goal locations **(b)**.

the percentile sensitivity prioritizes the worst case during training. To confirm this conclusion, we look at the actual distribution that is generating the expected performance in Figure 2b. In Figure 2e, we provide the empirical distribution of the trajectory-wise performance (H_τ), considering a batch of 200 trajectories with $p_{\boldsymbol{\mathcal{M}}} = [0.8, 0.2]$. It clearly shows that Neutral is heavy-tailed towards lower outcomes, whereas MEMENTO concentrates around the mean. This suggests that with a conservative choice of α we can induce a good exploration outcome for every trajectory (and any configuration), while without percentile sensitivity we cannot hedge against the risk of particularly bad outcomes. However, let us point out that not all classes of environments would expose such an issue for a naïve, risk-neutral approach (see Appendix D.4 for a counterexample), but it is fair to assume that this would arguably generalize to any setting where there is an imbalance (either in the hardness of the configurations, or in their sampling probability) in the class. These are the settings we care about, as they require nuanced solutions (e.g., MEMENTO) for scenarios with multiple environments.

5.2 On the Value of the Percentile

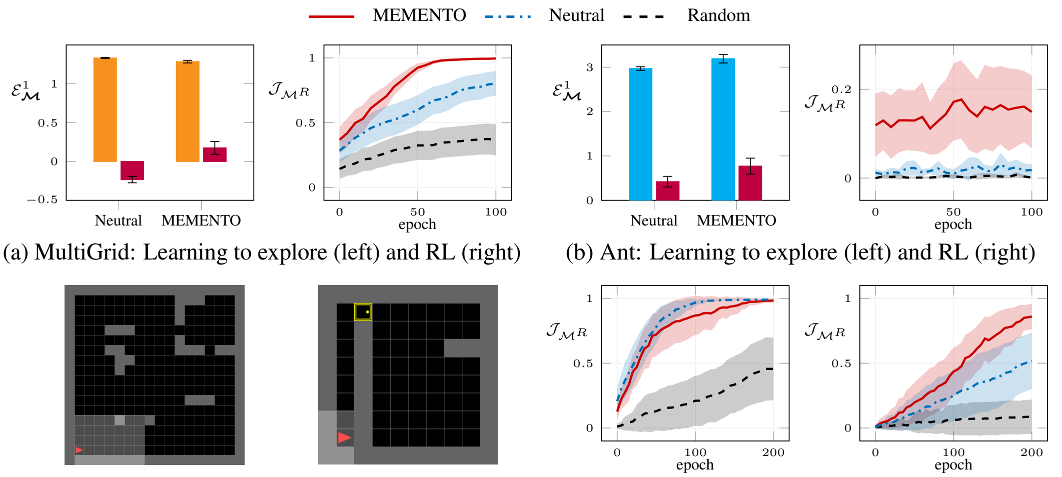
In this section, we consider repeatedly training MEMENTO with different values of α in the GridWorld with Slope domain, and we compare the resulting exploration performance $\mathcal{E}^1_{\boldsymbol{\mathcal{M}}}$ as before. In Figure 2f, we can see that the lower α we choose, the more we prioritize GWN (right bar for every α) at the expense of GWS (left bar). Note that this trend carries on with increasing α, ending in the values of Figures 2c, 2d. The reason for this behavior is quite straightforward, the smaller is α, the larger is the share of trajectories from the adverse configuration (GWN) ending up in the percentile at first (Figure 2g), and thus the more GWN affects the policy update (see the gradient in (5)).

5.3 RL with a General Exploration Strategy

To assess the benefit of the pre-trained strategy, we design a family of MDPs $\mathcal{M}^R$, where $\mathcal{M} \in \{\text{GWS}, \text{GWN}\}$, and R is any sparse reward function that gives 1 when the agent reaches the area nearby a random goal location and 0 otherwise. On this family, we compare the performance achieved by TRPO (Schulman et al., 2015) with different initializations: The exploration strategies learned (as in Section 5.1) by MEMENTO ($\alpha = 0.35$) and Neutral ($\alpha = 1$), or a randomly initialized policy (Random). These three variations are evaluated in terms of their average return $\mathcal{J}_{\mathcal{M}^R}$, which is defined in (1), over 50 randomly generated goal locations (Figure 3b). As expected, the performance of TRPO with Neutral is competitive in the GWS configuration (Figure 3), but it falls sharply in the GWN configuration, where it is not significantly better than TRPO with Random. Instead, the performance of TRPO with MEMENTO is strong on both GWS and GWN. Despite the simplicity of the domain, solving an RL problem in GWN with an adverse goal location is far-fetched for both a random initialization and a naïve solution to the reward-free exploration over multiple environments.

5.4 Scaling to Larger Classes of Environments

In this section, we consider a class $\boldsymbol{\mathcal{M}}$ composed of ten different configurations of the continuous gridworlds presented in Section 5.1 (including the GWN as the worst-case configuration) which we call the *MultiGrid* domain. As before, we compare MEMENTO ($\alpha = 0.1$) and Neutral ($\alpha = 1$) on the exploration performance $\mathcal{E}^1_{\boldsymbol{\mathcal{M}}}$ achieved by the optimal strategy, in this case considering a uniformly distributed $p_{\boldsymbol{\mathcal{M}}}$. While the average performance of Neutral is slightly higher across



(a) MultiGrid: Learning to explore (left) and RL (right)

(b) Ant: Learning to explore (left) and RL (right)

(c) MiniGrid: EasyG (left) and AdvG (right)

(d) MiniGrid: RL on EasyG (left) and AdvG (right)

Figure 4: Exploration performance $\mathcal{E}^1_{\mathcal{M}}$ (95% c.i. over 4 runs) achieved by MEMENTO ($\alpha = 0.1$ **(a)**, $\alpha = 0.2$ **(b)**) and Neutral ($\alpha = 1$) in the in the *MultiGrid* **(a)** and *Ant* **(b)** domains. Average return $\mathcal{J}_{\mathcal{M}^R}$ (95% c.i. over 50 tasks **(a)**, 8 tasks **(b)**, 13 tasks **(d)**) obtained by TRPO with corresponding initialization (MEMENTO, Neutral, Random), in the *MultiGrid* **(a)**, *Ant* **(b)**, and *MiniGrid* **(d)** domains. *MiniGrid* domains are illustrated in **(c)**.

the class (Figure 4a left, left bar), MEMENTO still has a decisive advantage in the worst-case configuration (Figure 4a left, right bar). Just as in Section 5.3, this advantage transfer to RL, where we compare the average return $\mathcal{J}_{\mathcal{M}^R}$ achieved by TRPO with MEMENTO, Neutral, and Random initializations over 50 random goal locations in the GWN configuration (Figure 4a right).

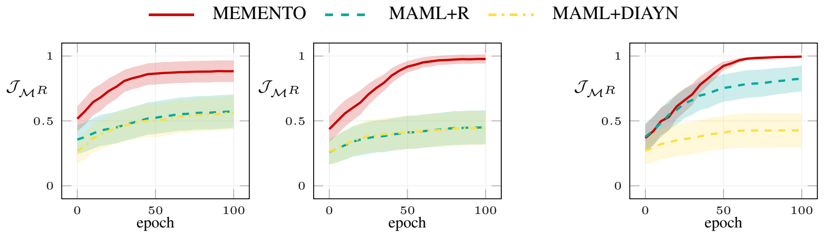
5.5 Scaling to Increasing Dimensions

In this section, we consider a class $\boldsymbol{\mathcal{M}}$ consisting of two Ant environments, with 29D states and 8D actions. In the first, sampled with probability $p_{\mathcal{M}_1} = 0.8$, the Ant faces a wide descending staircase (*Ant Stairs Down*). In the second, the Ant faces a narrow ascending staircase (*Ant Stairs Up*, sampled with probability $p_{\mathcal{M}_2} = 0.2$), which is significantly harder to explore than the former. In the mold of the gridworlds in Section 5.1, these two configurations are specifically designed to create an imbalance in the class. As in Section 5.1, we compare MEMENTO ($\alpha = 0.2$) against Neutral ($\alpha = 1$) on the exploration performance $\mathcal{E}^1_{\boldsymbol{\mathcal{M}}}$ achieved after 500 epochs. MEMENTO fares slightly better than Neutral both in the worst-case configuration (Figure 4b left, right bar) and, surprisingly, in the easier one (Figure 4b left, left bar).[2] Then, we design a set of incrementally challenging sparse-rewards RL tasks in the *Ant Stairs Up*, which give reward 1 upon reaching a certain step of the staircase. Also in this setting, TRPO with Memento initialization outperforms TRPO with Neutral and Random in terms of the average return $\mathcal{J}_{\mathcal{M}^R}$ (Figure 4b right). Note that these sparse-reward continuous control tasks are particularly arduous: TRPO with Neutral and Random barely learns anything, while even TRPO with MEMENTO does not handily reach the optimal average return (1) within 100 epochs.

5.6 Scaling to Visual Inputs

In this section, we consider a class $\boldsymbol{\mathcal{M}}$ of two partially-observable MiniGrid (Chevalier-Boisvert et al., 2018) environments, in which the observation is a 147D image of the agent's field of view. In Figure 4c, we provide a visualization of the domain: The easier configuration (EasyG, left) is sampled with probability $p_{\mathcal{M}_1} = 0.8$, the adverse configuration (AdvG, right) is sampled with probability $p_{\mathcal{M}_2} = 0.2$. Two factors make the AdvG more challenging to explore, which are the presence of a door at the top-left of the grid, and reversing the effect of agent's movements (e.g., the agent goes backward when it tries to go forward). Whereas in all the previous experiments we estimated the entropy on the raw input features, visual inputs require a wiser choice of a metric. As proposed in (Seo et al., 2021), we process the observations through a random encoder before computing the

[2] Note that this would not happen in general, as we expect MEMENTO to be better in the worst-case but worse on average. Apparently, the percentile objective positively biases the average performance in this setting.



(a) GridWorld with Slope: GWS (left) and GWN (right)

(b) MultiGrid

Figure 5: Average return $\mathcal{J}_{\mathcal{M}^R}$ achieved by TRPO initialized with MEMENTO ($\alpha = 0.35$ **(a)**, $\alpha = 0.1$ **(b)**), a MAML+R meta-policy, and a MAML+DIAYN meta-policy, when dealing with a set of RL tasks in the *GridWorld with Slope* **(a)** and the *MultiGrid* **(b)** domains. We provide 95% c.i. over 50 tasks. Note that the learning curves of MAML+R and MAML+DIAYN are almost overlapping in **(a)**.

entropy estimate in (3), while keeping everything else as in Algorithm 1. We run this slightly modified version of MEMENTO ($\alpha = 0.3$) and Neutral ($\alpha = 1$) for 300 epochs. Then, we compare TRPO with the learned initializations (as well as Random) on a series of sparse-reward tasks defined upon the class. As in previous settings, TRPO with MEMENTO results slightly worse than TRPO with Neutral in the easier configuration (Figure 4d, left), but significantly better in the worst-case (Figure 4d, right). Notably, TRPO from scratch struggles to learn the tasks, especially in the AdvG (Figure 4d, right).

5.7 Comparison with Meta-RL

In this section, we compare our approach against meta-training a policy with MAML (Finn et al., 2017) on the same *GridWorld with Slope* ($p_{\boldsymbol{\mathcal{M}}} = [0.8, 0.2]$) and *MultiGrid* (uniformly distributed $p_{\boldsymbol{\mathcal{M}}}$) domains that we have previously presented. Especially, we consider two relevant baselines. The first is MAML+R, to which we provide full access to the tasks (i.e., reward functions) during meta-training. Note that this gives MAML+R a great edge over MEMENTO, which operates reward-free training. The second is MAML+DIAYN (Gupta et al., 2018a), which operates unsupervised meta-training through an intrinsic reward function learned with DIAYN (Eysenbach et al., 2018). As in previous sections, we consider the average return $\mathcal{J}_{\mathcal{M}^R}$ achieved by TRPO initialized with the exploration strategy learned by MEMENTO or the meta-policy learned by MAML+R and MAML+DIAYN. TRPO with MEMENTO fares clearly better than TRPO with the meta-policies in all the configurations (Figures 5a, 5b). Even if it works fine in fast adaptation (see Appendix D.5), MAML struggles to encode the diversity of task distribution into a single meta-policy and to deal with the most adverse tasks in the long run. Moreover, DIAYN does not specifically handle multiple environments, and it fails to cope with the larger *MultiGrid* class (see Appendix D.5).

6 Preliminary Theoretical Analysis of the Problem

In this section, we aim to theoretically analyze the problem in (2), and especially, what makes a class of multiple CMPs hard to explore with a unique strategy. This has to be intended as a preliminary discussion on the problem, which could serve as a starting point for future works, rather than a thorough theoretical characterization. First, it is worth introducing some additional notation.

Lipschitz Continuity Let X, Y be two metric sets with metric functions d_X, d_Y. We say a function $f : X \to Y$ is L_f-Lipschitz continuous if it holds for some constant L_f $d_Y(f(x'), f(x)) \leq L_f d_X(x', x), \forall (x', x) \in X^2$, where the smallest L_f is the Lipschitz constant and the Lipschitz semi-norm is $\|f\|_L = \sup_{x', x \in X} \left\{ \frac{d_Y(f(x'), f(x))}{d_X(x', x)} : x' \neq x \right\}$. When dealing with probability distributions we need to introduce a proper metric. Let p, q be two probability measures, we will either consider the Wasserstein metric (Villani, 2008), defined as $d_{W_1}(p, q) = \sup_f \left\{ \left| \int_X f \, \mathrm{d}(p - q) \right| : \|f\|_L \leq 1 \right\}$, or the Total Variation (TV) metric, defined as $d_{TV}(p, q) = \frac{1}{2} \int_X \left| \mathrm{d}(p - q) \right|$.

Intuitively, learning to explore a class $\boldsymbol{\mathcal{M}}$ with a policy π is challenging when the state distributions induced by π in different $\mathcal{M} \in \boldsymbol{\mathcal{M}}$ are diverse. The more diverse they are, the more their entropy can vary, and the harder is to get a π with a large entropy across the class. To measure this diversity, we are interested in the supremum over the distances between the state distributions $(d_\pi^{\mathcal{M}_1}, \ldots, d_\pi^{\mathcal{M}_I})$ that a single policy $\pi \in \Pi$ realizes over the class $\boldsymbol{\mathcal{M}}$. We call this measure the *diameter* $\mathcal{D}_{\boldsymbol{\mathcal{M}}}$ of the class $\boldsymbol{\mathcal{M}}$. Since we have infinitely many policies in Π, computing $\mathcal{D}_{\boldsymbol{\mathcal{M}}}$ is particularly arduous. However, we are able to provide an upper bound to $\mathcal{D}_{\boldsymbol{\mathcal{M}}}$ defined through a Wasserstein metric.

Assumption 1. *Let $d_{\mathcal{S}}$ be a metric on $\mathcal{S}$. The class $\boldsymbol{\mathcal{M}}$ is L_{P^π}-Lipschitz continuous,*

$$d_{W_1}(P^\pi(\cdot|s'), P^\pi(\cdot|s)) \leq L_{P^\pi} d_{\mathcal{S}}(s', s), \quad \forall (s', s) \in \mathcal{S}^2,$$

where $P^\pi(s|\overline{s}) = \int_{\mathcal{A}} \pi(\overline{a}|\overline{s}) P(s|\overline{s}, \overline{a}) \,\mathrm{d}\overline{a}$ for $P \in \boldsymbol{\mathcal{M}}$, $\pi \in \Pi$, L_{P^π} is a constant $L_{P^\pi} < 1$.

Theorem 6.1. *Let $\boldsymbol{\mathcal{M}}$ be a class of CMPs satisfying Ass. 1. Let $d_\pi^{\mathcal{M}}$ be the marginal state distribution over T steps induced by the policy π in $\mathcal{M} \in \boldsymbol{\mathcal{M}}$. We can upper bound the diameter $\mathcal{D}_{\boldsymbol{\mathcal{M}}}$ as*

$$\mathcal{D}_{\boldsymbol{\mathcal{M}}} := \sup_{\pi \in \Pi,\, \mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}} d_{W_1}(d_\pi^{\mathcal{M}'}, d_\pi^{\mathcal{M}}) \leq \sup_{P', P \in \boldsymbol{\mathcal{M}}} \frac{1 - L_{P^\pi}^T}{1 - L_{P^\pi}} \sup_{s \in \mathcal{S}, a \in \mathcal{A}} d_{W_1}(P'(\cdot|s,a), P(\cdot|s,a)).$$

Theorem 6.1 provides a measure to quantify the hardness of the exploration problem in a specific class of CMPs, and to possibly compare one class with another. However, the value of $\mathcal{D}_{\boldsymbol{\mathcal{M}}}$ might result, due to the supremum over Π, from a policy that is far away from the policies we actually deploy while learning, say $(\pi_0, \ldots, \pi_{\mathcal{E}}^*)$. To get a finer assessment of the hardness of $\boldsymbol{\mathcal{M}}$ we face in practice, it is worth considering a policy-specific measure to track during the optimization. We call this measure the π-*diameter* $\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi)$ of the class $\boldsymbol{\mathcal{M}}$. Theorem 6.2 provides an upper bound to $\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi)$ defined through a convenient TV metric.

Theorem 6.2. *Let $\boldsymbol{\mathcal{M}}$ be a class of CMPs, let $\pi \in \Pi$ be a policy, and let $d_\pi^{\mathcal{M}}$ be the marginal state distribution over T steps induced by π in $\mathcal{M} \in \boldsymbol{\mathcal{M}}$. We can upper bound the π-diameter $\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi)$ as*

$$\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi) := \sup_{\mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}} d_{TV}(d_\pi^{\mathcal{M}'}, d_\pi^{\mathcal{M}}) \leq \sup_{P', P \in \boldsymbol{\mathcal{M}}} T \mathop{\mathbb{E}}_{\substack{s \sim d_\pi^{\mathcal{M}} \\ a \sim \pi(\cdot|s)}} d_{TV}(P'(\cdot|s,a), P(\cdot|s,a)).$$

The last missing piece we aim to derive is a result to relate the π-diameter $\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi)$ of the class $\boldsymbol{\mathcal{M}}$ (Theorem 6.2) with the actual exploration objective, i.e., the entropy of the state visitations induced by the policy π over the environments in the class. In the following theorem, we provide an upper bound to the *entropy gap* induced by the policy π within the class $\boldsymbol{\mathcal{M}}$.

Theorem 6.3. *Let $\boldsymbol{\mathcal{M}}$ be a class of CMPs, let $\pi \in \Pi$ be a policy and $\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi)$ the corresponding π-diameter of $\boldsymbol{\mathcal{M}}$. Let $d_\pi^{\mathcal{M}}$ be the marginal state distribution over T steps induced by π in $\mathcal{M} \in \boldsymbol{\mathcal{M}}$, and let $\sigma_{\boldsymbol{\mathcal{M}}} \leq \sigma_{\mathcal{M}} := \inf_{s \in \mathcal{S}} d_\pi^{\mathcal{M}}(s), \forall \mathcal{M} \in \boldsymbol{\mathcal{M}}$. We can upper bound the entropy gap of the policy π within the model class $\boldsymbol{\mathcal{M}}$ as*

$$\sup_{\mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}} \left| H(d_\pi^{\mathcal{M}'}) - H(d_\pi^{\mathcal{M}}) \right| \leq \big(\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi)\big)^2 / \sigma_{\boldsymbol{\mathcal{M}}} + \mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi) \log(1/\sigma_{\boldsymbol{\mathcal{M}}})$$

7 Conclusions

In this paper, we addressed the problem of learning to explore a class of multiple reward-free environments with a unique general strategy. First, we formulated the problem within a tractable MSVE objective with percentile sensitivity. Then, we presented a policy gradient algorithm to optimize this objective. Finally, we provided an extensive experimental analysis to show its ability in learning to explore and the benefits it brings to subsequent RL problems. We believe that this paper motivates the importance of designing specific solutions to the relevant reward-free exploration problem over multiple environments.

As a final note, it is worth mentioning an alternative problem formulation in which MEMENTO can be employed with benefit. Especially, we could replace the class of environments of our setting with a single CMP specified under uncertainty (Satia & Lave Jr, 1973), and deal with the *robust reward-free exploration* problem with little or no modifications to MEMENTO.

REFERENCES

Jiří Ajgl and Miroslav Šimandl. Differential entropy estimation by particles. *IFAC Proceedings Volumes*, 2011.

Maxime Chevalier-Boisvert, Lucas Willems, and Suman Pal. Minimalistic gridworld environment for openai gym. *GitHub repository*, 2018.

Yinlam Chow and Mohammad Ghavamzadeh. Algorithms for cvar optimization in mdps. In *Advances in neural information processing systems*, 2014.

Imre Csiszár and Zsolt Talata. Context tree estimation for not necessarily finite memory processes, via bic and mdl. *IEEE Transactions on Information theory*, 2006.

M Deisenroth, G Neumann, and J Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2013.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *Proceedings of the International Conference on Machine Learning*, 2016.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2018.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the International Conference on Machine Learning*, 2017.

Zhaohan Daniel Guo, Mohammad Gheshlagi Azar, Alaa Saade, Shantanu Thakoor, Bilal Piot, Bernardo Avila Pires, Michal Valko, Thomas Mesnard, Tor Lattimore, and Rémi Munos. Geometric entropic exploration. *arXiv preprint arXiv:2101.02055*, 2021.

Abhishek Gupta, Benjamin Eysenbach, Chelsea Finn, and Sergey Levine. Unsupervised meta-learning for reinforcement learning. *arXiv preprint arXiv:1806.04640*, 2018a.

Abhishek Gupta, Russell Mendonca, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Meta-reinforcement learning of structured exploration strategies. In *Advances in Neural Information Processing Systems*, 2018b.

Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In *Proceedings of the International Conference on Machine Learning*, 2019.

Chi Jin, Akshay Krishnamurthy, Max Simchowitz, and Tiancheng Yu. Reward-free exploration for reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, 2020.

Ravi Kumar Kolla, LA Prashanth, Sanjay P Bhat, and Krishna Jagannathan. Concentration bounds for empirical conditional value-at-risk: The unbounded case. *Operations research letters*, 2019.

Prashanth L.A., Krishna Jagannathan, and Ravi Kolla. Concentration bounds for CVaR estimation: The cases of light-tailed and heavy-tailed distributions. In *Proceedings of the International Conference on Machine Learning*, 2020.

Lisa Lee, Benjamin Eysenbach, Emilio Parisotto, Eric Xing, Sergey Levine, and Ruslan Salakhutdinov. Efficient exploration via state marginal matching. *arXiv preprint arXiv:1906.05274*, 2019.

Hao Liu and Pieter Abbeel. Behavior from the void: Unsupervised active pre-training. *arXiv preprint arXiv:2103.04551*, 2021.

Alberto Maria Metelli, Mirco Mutti, and Marcello Restelli. Configurable markov decision processes. In *Proceedings of the International Conference on Machine Learning*, 2018a.

Alberto Maria Metelli, Matteo Papini, Francesco Faccio, and Marcello Restelli. Policy optimization via importance sampling. In *Advances in Neural Information Processing Systems*, 2018b.

Mirco Mutti and Marcello Restelli. An intrinsically-motivated approach for learning highly exploring and fast mixing policies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Mirco Mutti, Lorenzo Pratissoli, and Marcello Restelli. A policy gradient method for task-agnostic exploration. *arXiv preprint arXiv:2007.04640*, 2020.

Simone Parisi, Matteo Pirotta, and Marcello Restelli. Multi-objective reinforcement learning through continuous pareto manifold approximation. *Journal of Artificial Intelligence Research*, 2016.

Matteo Pirotta, Marcello Restelli, and Luca Bascetta. Policy gradient in lipschitz markov decision processes. *Machine Learning*, 2015.

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

Aravind Rajeswaran, Sarvjeet Ghotra, Balaraman Ravindran, and Sergey Levine. Epopt: Learning robust neural network policies using model ensembles. In *Proceedings of the International Conference on Learning Representations*, 2016.

R Tyrrell Rockafellar, Stanislav Uryasev, et al. Optimization of conditional value-at-risk. *Journal of risk*, 2000.

Jay K Satia and Roy E Lave Jr. Markovian decision processes with uncertain transition probabilities. *Operations Research*, 1973.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the International Conference on Machine Learning*, 2015.

Younggyo Seo, Lili Chen, Jinwoo Shin, Honglak Lee, Pieter Abbeel, and Kimin Lee. State entropy maximization with random encoders for efficient exploration. *arXiv preprint arXiv:2102.09430*, 2021.

Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 1948.

Harshinder Singh, Neeraj Misra, Vladimir Hnizdo, Adam Fedorowicz, and Eugene Demchuk. Nearest neighbor estimates of entropy. *American journal of mathematical and management sciences*, 2003.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Aviv Tamar, Yonatan Glassner, and Shie Mannor. Optimizing the cvar via sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2015.

Jean Tarbouriech and Alessandro Lazaric. Active exploration in markov decision processes. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2019.

Jean Tarbouriech, Matteo Pirotta, Michal Valko, DeepMind Paris, and Alessandro Lazaric. Improved sample complexity for incremental autonomous exploration in mdps. In *Advances in Neural Information Processing Systems*, 2020a.

Jean Tarbouriech, Shubhanshu Shekhar, Matteo Pirotta, Mohammad Ghavamzadeh, and Alessandro Lazaric. Active model estimation in markov decision processes. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2020b.

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

Tianbing Xu, Qiang Liu, Liang Zhao, and Jian Peng. Learning to explore via meta-policy gradient. In *Proceedings of the International Conference on Machine Learning*, 2018.

Chuheng Zhang, Yuanying Cai, Longbo Huang, and Jian Li. Exploration by maximizing rényi entropy for zero-shot meta rl. *arXiv preprint arXiv:2006.06193*, 2020a.

Xuezhou Zhang, Yuzhe Ma, and Adish Singla. Task-agnostic exploration in reinforcement learning. In *Advances in Neural Information Processing Systems*, 2020b.

Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. In *Proceedings of the International Conference on Learning Representations*, 2019.

A Related Work

Our work lies at the intersection of reward-free exploration, robust and risk-averse RL, and meta-RL.

The literature that relates the most with our work is the one pursuing a *MSVE objective* in reward-free settings. Some methods (Hazan et al., 2019; Lee et al., 2019) focus on learning a mixture of policies that is collectively MSVE optimal, while other (Tarbouriech & Lazaric, 2019; Mutti & Restelli, 2020) casts the MSVE as a dual (or surrogate) linear program in tabular settings. Successive works tackle MSVE at scale with non-parametric entropy estimation (Mutti et al., 2020; Liu & Abbeel, 2021; Seo et al., 2021), or introduce variations to the entropy objective, such as geometry-awareness (Guo et al., 2021) and Rényi generalization (Zhang et al., 2020a). To the best of our knowledge, all existing solutions are environment-specific and do not directly address multiple environments.

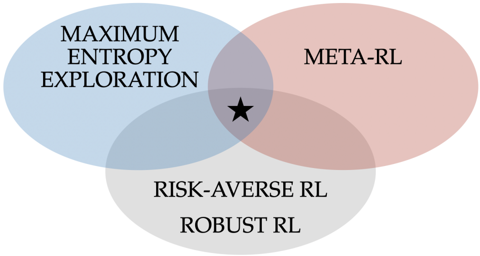


Figure 6: Where our work (star) stands in the literature.

Previous work considered *CVaR optimization* in RL, either to learn a policy that is averse to the risk induced by the volatility of returns (Tamar et al., 2015; Chow & Ghavamzadeh, 2014) or by changes in the environment dynamics (e.g., Rajeswaran et al., 2016). Here we account for a different source of risk, which is the one of running into a particularly unfavorable environment for the trained exploration strategy.

Finally, the two-stage learning setting we address has clear connections with the *meta-RL* problem setting (Finn et al., 2017), in which we would call meta-training the reward-free phase, and meta-testing the subsequent RL tasks. While some methods target exploration in meta-RL (e.g., Xu et al., 2018; Gupta et al., 2018b; Zintgraf et al., 2019), they usually assume access to rewards during meta-training, with the notable exception of (Gupta et al., 2018a). To the best of our knowledge, none of the existing works combine reward-free meta-training with a multiple-environments setting.

B Proofs

Proposition 4.1. *The policy gradient of the exploration objective $\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})$ w.r.t. $\boldsymbol{\theta}$ is given by*

$$\nabla_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) = \mathop{\mathbb{E}}_{\substack{\mathcal{M}\sim p_{\boldsymbol{\mathcal{M}}}\\ \tau\sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}}}}\left[\left(\sum_{t=0}^{T-1}\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a_{t,\tau}|s_{t,\tau})\right)\Big(H_\tau - \mathrm{VaR}_\alpha(H_\tau)\Big)\Big| H_\tau \le \mathrm{VaR}_\alpha(H_\tau)\right].$$

Proof. Let us start from expanding the exploration objective (2) to write

$$\begin{aligned}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi) &= \mathrm{CVaR}_\alpha(H_\tau)\\ &= \mathop{\mathbb{E}}_{\substack{\mathcal{M}\sim p_{\boldsymbol{\mathcal{M}}}\\ \tau\sim p_{\pi,\mathcal{M}}}}\big[H_\tau \mid H_\tau \le \mathrm{VaR}_\alpha(H_\tau)\big] = \frac{1}{\alpha}\int_{-\infty}^{\mathrm{VaR}_\alpha(H_\tau)} p_{\pi_{\boldsymbol{\theta}},\boldsymbol{\mathcal{M}}}(h)h\,\mathrm{d}h, \end{aligned} \tag{6}$$

where $p_{\pi_{\boldsymbol{\theta}},\boldsymbol{\mathcal{M}}}$ is the probability density function (pdf) of the random variable H_τ when the policy $\pi_{\boldsymbol{\theta}}$ is deployed on the class of environments $\boldsymbol{\mathcal{M}}$, and the last equality comes from the definition of CVaR (Rockafellar et al., 2000). Before computing the gradient of (6), we derive a preliminary result for later use, i.e.,

$$\begin{aligned}&\nabla_{\boldsymbol{\theta}}\int_{-\infty}^{\mathrm{VaR}_\alpha(H_\tau)} p_{\pi_{\boldsymbol{\theta}},\boldsymbol{\mathcal{M}}}(h)\,\mathrm{d}h\\ &\quad= \int_{-\infty}^{\mathrm{VaR}_\alpha(H_\tau)} \nabla_{\boldsymbol{\theta}}p_{\pi_{\boldsymbol{\theta}},\boldsymbol{\mathcal{M}}}(h)\,\mathrm{d}h + \nabla_{\boldsymbol{\theta}}\,\mathrm{VaR}_\alpha(H_\tau)p_{\pi_{\boldsymbol{\theta}},\boldsymbol{\mathcal{M}}}(\mathrm{VaR}_\alpha(H_\tau)) = 0,\end{aligned} \tag{7}$$

which follows directly from the Leibniz integral rule, noting that $\mathrm{VaR}_\alpha(H_\tau)$ depends on $\boldsymbol{\theta}$ through the pdf of H_τ. We now take the gradient of (6) to get

$$
\begin{aligned}
&\nabla_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi) \\
&= \nabla_{\boldsymbol{\theta}} \frac{1}{\alpha} \int_{-\infty}^{\mathrm{VaR}_\alpha(H_\tau)} p_{\pi_{\boldsymbol{\theta}}, \boldsymbol{\mathcal{M}}}(h) h \,\mathrm{d}h \\
&= \frac{1}{\alpha} \int_{-\infty}^{\mathrm{VaR}_\alpha(H_\tau)} \nabla_{\boldsymbol{\theta}} p_{\pi_{\boldsymbol{\theta}}, \boldsymbol{\mathcal{M}}}(h) h \,\mathrm{d}h + \frac{1}{\alpha} \nabla_{\boldsymbol{\theta}} \mathrm{VaR}_\alpha(H_\tau) \mathrm{VaR}_\alpha(H_\tau) p_{\pi_{\boldsymbol{\theta}}, \boldsymbol{\mathcal{M}}}(\mathrm{VaR}_\alpha(H_\tau)) \qquad (8) \\
&= \frac{1}{\alpha} \int_{-\infty}^{\mathrm{VaR}_\alpha(H_\tau)} \nabla_{\boldsymbol{\theta}} p_{\pi_{\boldsymbol{\theta}}, \boldsymbol{\mathcal{M}}}(h) \Big(h - \mathrm{VaR}_\alpha(H_\tau) \Big) \,\mathrm{d}h, \qquad (9)
\end{aligned}
$$

where (8) follows from the Leibniz integral rule, and (9) is obtained from (8) through (7), which we can rearrange to write $p_{\pi_{\boldsymbol{\theta}}, \boldsymbol{\mathcal{M}}}(\mathrm{VaR}_\alpha(H_\tau)) = \frac{1}{\nabla_{\boldsymbol{\theta}} \mathrm{VaR}_\alpha(H_\tau)} \int_{-\infty}^{\mathrm{VaR}_\alpha(H_\tau)} \nabla_{\boldsymbol{\theta}} p_{\pi_{\boldsymbol{\theta}}, \boldsymbol{\mathcal{M}}}(h) \,\mathrm{d}h$. All of the steps above are straightforward replications of the derivations by Tamar et al. (2015), Proposition 1. To conclude the proof we just have to compute the term $\nabla_{\boldsymbol{\theta}} p_{\pi_{\boldsymbol{\theta}}, \boldsymbol{\mathcal{M}}}(h)$, which is specific to our setting. Especially, we note that

$$
\begin{aligned}
&\nabla_{\boldsymbol{\theta}} p_{\pi_{\boldsymbol{\theta}}, \boldsymbol{\mathcal{M}}}(h) \\
&\quad = \int_{\boldsymbol{\mathcal{M}}} p_{\boldsymbol{\mathcal{M}}}(\mathcal{M}) \int_{\mathcal{T}} \nabla_{\boldsymbol{\theta}} p_{\pi_{\boldsymbol{\theta}}, \mathcal{M}}(\tau) \delta(h - H_\tau) \,\mathrm{d}\tau \,\mathrm{d}\mathcal{M} \qquad (10) \\
&\quad = \int_{\boldsymbol{\mathcal{M}}} p_{\boldsymbol{\mathcal{M}}}(\mathcal{M}) \int_{\mathcal{T}} p_{\pi_{\boldsymbol{\theta}}, \mathcal{M}}(\tau) \nabla_{\boldsymbol{\theta}} \log p_{\pi_{\boldsymbol{\theta}}, \mathcal{M}}(\tau) \delta(h - H_\tau) \,\mathrm{d}\tau \,\mathrm{d}\mathcal{M} \\
&\quad = \int_{\boldsymbol{\mathcal{M}}} p_{\boldsymbol{\mathcal{M}}}(\mathcal{M}) \int_{\mathcal{T}} p_{\pi_{\boldsymbol{\theta}}, \mathcal{M}}(\tau) \bigg(\sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_{t,\tau} | s_{t,\tau}) \bigg) \delta(h - H_\tau) \,\mathrm{d}\tau \,\mathrm{d}\mathcal{M}, \qquad (11)
\end{aligned}
$$

where (10) and (11) are straightforward from the definitions in Section 2, and $\mathcal{T}$ is the set of feasible trajectories of length T. Finally, the result follows by plugging (11) into (9), which gives

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi) = &\frac{1}{\alpha} \int_{\boldsymbol{\mathcal{M}}} p_{\boldsymbol{\mathcal{M}}}(\mathcal{M}) \int_{\mathcal{T}} p_{\pi_{\boldsymbol{\theta}}, \mathcal{M}}(\tau) \\
&\times \int_{-\infty}^{\mathrm{VaR}_\alpha(H_\tau)} \delta(h - H_\tau) \bigg(\sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_{t,\tau} | s_{t,\tau}) \bigg) \Big(h - \mathrm{VaR}_\alpha(H_\tau) \Big) \,\mathrm{d}h \,\mathrm{d}\tau \,\mathrm{d}\mathcal{M}.
\end{aligned}
$$

□

Theorem 6.1. *Let $\boldsymbol{\mathcal{M}}$ be a class of CMPs satisfying Ass. 1. Let $d_\pi^{\mathcal{M}}$ be the marginal state distribution over T steps induced by the policy π in $\mathcal{M} \in \boldsymbol{\mathcal{M}}$. We can upper bound the diameter $\mathcal{D}_{\boldsymbol{\mathcal{M}}}$ as*

$$
\mathcal{D}_{\boldsymbol{\mathcal{M}}} := \sup_{\pi \in \Pi,\ \mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}} d_{W_1}(d_\pi^{\mathcal{M}'}, d_\pi^{\mathcal{M}}) \leq \sup_{P', P \in \boldsymbol{\mathcal{M}}} \frac{1 - L_{P^\pi}^T}{1 - L_{P^\pi}} \sup_{s \in \mathcal{S}, a \in \mathcal{A}} d_{W_1}(P'(\cdot|s,a), P(\cdot|s,a)).
$$

Proof. The proof follows techniques from Pirotta et al. (2015). Let us report a preliminary result which states that the function $h_f(\overline{s}) = \int_{\mathcal{A}} \pi(\overline{a}|\overline{s}) \int_{\mathcal{S}} P(s|\overline{s}, \overline{a}) \,\mathrm{d}s \,\mathrm{d}\overline{a}$ has a Lipschitz constant equal to L_{P^π} (Pirotta et al., 2015, Lemma 3):

$$
\begin{aligned}
\big| h_f(\overline{s}') - h_f(\overline{s}) \big| &= \bigg| \int_{\mathcal{S}} f(s) \int_{\mathcal{A}} \pi(a|\overline{s}') P(s|\overline{s}', a) \,\mathrm{d}a \,\mathrm{d}s - \int_{\mathcal{S}} f(s) \int_{\mathcal{A}} \pi(a|\overline{s}) P(s|\overline{s}, a) \,\mathrm{d}a \,\mathrm{d}s \bigg| \\
&= \bigg| \int_{\mathcal{S}} f(s) \Big(P^\pi(s|\overline{s}') - P^\pi(s|\overline{s}) \Big) \,\mathrm{d}s \bigg| \leq L_{P^\pi} d_{\mathcal{S}}(\overline{s}', \overline{s}), \qquad (12)
\end{aligned}
$$

where $d_{\mathcal{S}}$ is a metric over $\mathcal{S}$ and $P^\pi(s|\overline{s}) = \int_{\mathcal{A}} \pi(\overline{a}|\overline{s}) P(s|\overline{s}, \overline{a}) \,\mathrm{d}\overline{a}$. Then, we note that the marginal state distribution over T steps $d_\pi^{\mathcal{M}}$ can be written as a sum of the contributions $d_{\pi,t}^{\mathcal{M}}$ related to any time step $t \in [T]$, which is

$$
d_\pi^{\mathcal{M}}(s) = \frac{1}{T} \sum_{t=0}^{T-1} d_{\pi,t}^{\mathcal{M}}(s). \qquad (13)
$$

Hence, we can look at the Wasserstein distance of the state distributions for some $t \in [T]$ and $\mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}$. We obtain

$$
\begin{aligned}
& d_{W_1}(d_{\pi,t}^{\mathcal{M}'}, d_{\pi,t}^{\mathcal{M}}) \\
&\quad = \sup_f \left\{ \left| \int_{\mathcal{S}} \Big(d_{\pi,t}^{\mathcal{M}'}(s) - d_{\pi,t}^{\mathcal{M}}(s) \Big) f(s) \,\mathrm{d}s \right| : \|f\|_L \leq 1 \right\} && (14) \\
&\quad = \sup_f \left\{ \left| \int_{\mathcal{S}} \int_{\mathcal{A}} \int_{\mathcal{S}} \Big(d_{\pi,t-1}^{\mathcal{M}'}(\overline{s}) \pi(\overline{a}|\overline{s}) P'(s|\overline{s},\overline{a}) - d_{\pi,t-1}^{\mathcal{M}}(\overline{s}) \pi(\overline{a}|\overline{s}) P(s|\overline{s},\overline{a}) \Big) f(s) \,\mathrm{d}s \,\mathrm{d}\overline{a} \,\mathrm{d}\overline{s} \right| : \|f\|_L \leq 1 \right\} \\
&\quad = \sup_f \left\{ \left| \int_{\mathcal{S}} d_{\pi,t-1}^{\mathcal{M}'}(\overline{s}) \int_{\mathcal{A}} \int_{\mathcal{S}} \pi(\overline{a}|\overline{s}) \Big(P'(s|\overline{s},\overline{a}) - P(s|\overline{s},\overline{a}) \Big) f(s) \,\mathrm{d}s \,\mathrm{d}\overline{a} \,\mathrm{d}\overline{s} \right. \right. && (15) \\
&\qquad \left. \left. + \int_{\mathcal{S}} \Big(d_{\pi,t-1}^{\mathcal{M}'}(\overline{s}) - d_{\pi,t-1}^{\mathcal{M}}(\overline{s}) \Big) \int_{\mathcal{A}} \int_{\mathcal{S}} \pi(\overline{a}|\overline{s}) P(s|\overline{s},\overline{a}) f(s) \,\mathrm{d}s \,\mathrm{d}\overline{a} \,\mathrm{d}\overline{s} \right| : \|f\|_L \leq 1 \right\} && (16) \\
&\quad \leq \sup_f \left\{ \left| \int_{\mathcal{S}} d_{\pi,t-1}^{\mathcal{M}'}(\overline{s}) \int_{\mathcal{A}} \int_{\mathcal{S}} \pi(\overline{a}|\overline{s}) \Big(P'(s|\overline{s},\overline{a}) - P(s|\overline{s},\overline{a}) \Big) f(s) \,\mathrm{d}s \,\mathrm{d}\overline{a} \,\mathrm{d}\overline{s} \right| : \|f\|_L \leq 1 \right\} \\
&\qquad + \sup_f \left\{ \left| \int_{\mathcal{S}} \Big(d_{\pi,t-1}^{\mathcal{M}'}(\overline{s}) - d_{\pi,t-1}^{\mathcal{M}}(\overline{s}) \Big) \int_{\mathcal{A}} \int_{\mathcal{S}} \pi(\overline{a}|\overline{s}) P(s|\overline{s},\overline{a}) f(s) \,\mathrm{d}s \,\mathrm{d}\overline{a} \,\mathrm{d}\overline{s} \right| : \|f\|_L \leq 1 \right\} \\
&\quad \leq \sup_f \left\{ \int_{\mathcal{S}} d_{\pi,t-1}^{\mathcal{M}'}(\overline{s}) \int_{\mathcal{A}} \pi(\overline{a}|\overline{s}) \,\mathrm{d}\overline{a} \,\mathrm{d}\overline{s} \sup_{\overline{s} \in \mathcal{S}, \overline{a} \in \mathcal{A}} \left\{ \left| \int_{\mathcal{S}} \Big(P'(s|\overline{s},\overline{a}) - P(s|\overline{s},\overline{a}) \Big) f(s) \,\mathrm{d}s \right| \right\} : \|f\|_L \leq 1 \right\} \\
&\qquad + L_{P^\pi} \sup_f \left\{ \left| \int_{\mathcal{S}} \Big(d_{\pi,t-1}^{\mathcal{M}'}(\overline{s}) - d_{\pi,t-1}^{\mathcal{M}}(\overline{s}) \Big) \frac{h_f(\overline{s})}{L_{P^\pi}} \,\mathrm{d}\overline{s} \right| : \|f\|_L \leq 1 \right\} && (17) \\
&\quad = \sup_{s \in \mathcal{S}, a \in \mathcal{A}} d_{W_1}(P'(\cdot|s,a), P(\cdot|s,a)) + L_{P^\pi} d_{W_1}(d_{\pi,t-1}^{\mathcal{M}'}, d_{\pi,t-1}^{\mathcal{M}}), && (18)
\end{aligned}
$$

where we plugged the common temporal relation $d_{\pi,t}^{\mathcal{M}}(s') = \int_{\mathcal{S}} \int_{\mathcal{A}} d_{\pi,t-1}^{\mathcal{M}}(s) \pi(a|s) P(s'|s,a) \,\mathrm{d}s \,\mathrm{d}a$ into (14), we sum and subtract $\int_{\mathcal{S}} \int_{\mathcal{A}} \int_{\mathcal{S}} d_{\pi,t-1}^{\mathcal{M}'}(\overline{s}) \pi(\overline{a}|\overline{s}) P(s|\overline{s},\overline{a}) \,\mathrm{d}s \,\mathrm{d}\overline{a} \,\mathrm{d}\overline{s}$ to get (15), (16), and we apply the inequality in (12) to obtain (17) and then (18). To get rid of the dependence to the state distributions $d_{\pi,t-1}^{\mathcal{M}'}$ and $d_{\pi,t-1}^{\mathcal{M}}$, we repeatedly unroll (18) to get

$$
\begin{aligned}
d_{W_1}(d_{\pi,t}^{\mathcal{M}'}, d_{\pi,t}^{\mathcal{M}}) &\leq \left(\sum_{j=0}^{t} L_{P^\pi}^j \right) \sup_{s \in \mathcal{S}, a \in \mathcal{A}} d_{W_1}(P'(\cdot|s,a), P(\cdot|s,a)) + L_{P^\pi}^t d_{W_1}(D', D) && (19) \\
&= \left(\frac{1 - L_{P^\pi}^t}{1 - L_{P^\pi}} \right) \sup_{s \in \mathcal{S}, a \in \mathcal{A}} d_{W_1}(P'(\cdot|s,a), P(\cdot|s,a)) + L_{P^\pi}^t d_{W_1}(D', D), && (20)
\end{aligned}
$$

where we note that $d_{W_1}(d_{\pi,0}^{\mathcal{M}'}, d_{\pi,0}^{\mathcal{M}}) = d_{W_1}(D', D)$ to derive (19), and we assume $L_{P^\pi} < 1$ (Assumption 1) to get (20) from (19). As a side note, when the state and action spaces are discrete, a natural choice of a metric is $d_{\mathcal{S}}(s', s) = \mathbb{1}(s' \neq s)$ and $d_{\mathcal{A}} = \mathbb{1}(a' \neq a)$, which results in the Wasserstein distance being equivalent to the total variation, the constant $L_{P^\pi} = 1$, and $\sum_{j=0}^{t} L_{P^\pi}^j = t$. More details over the Lipschitz constant L_{P^π} can be found in Pirotta et al. (2015). Finally, we can exploit the result in (20) to write

$$
\begin{aligned}
d_{W_1}(d_{\pi}^{\mathcal{M}'}, d_{\pi}^{\mathcal{M}}) &= \sup_f \left\{ \left| \int_{\mathcal{S}} \left(\frac{1}{T} \sum_{t=0}^{T-1} d_{\pi,t}^{\mathcal{M}'}(s) - \frac{1}{T} \sum_{t=0}^{T-1} d_{\pi,t}^{\mathcal{M}}(s) \right) f(s) \,\mathrm{d}s \right| : \|f\|_L \leq 1 \right\} && (21) \\
&\leq \frac{1}{T} \sum_{t=0}^{T-1} \sup_f \left\{ \left| \int_{\mathcal{S}} \Big(d_{\pi,t}^{\mathcal{M}'}(s) - d_{\pi,t}^{\mathcal{M}}(s) \Big) f(s) \,\mathrm{d}s \right| : \|f\|_L \leq 1 \right\} \\
&\leq \frac{1}{T} \sum_{t=0}^{T-1} \frac{1 - L_{P^\pi}^t}{1 - L_{P^\pi}} \sup_{s \in \mathcal{S}, a \in \mathcal{A}} d_{W_1}(P'(\cdot|s,a), P(\cdot|s,a)) + L_{P^\pi}^t d_{W_1}(D', D) \\
&\leq \frac{1 - L_{P^\pi}^T}{1 - L_{P^\pi}} \sup_{s \in \mathcal{S}, a \in \mathcal{A}} d_{W_1}(P'(\cdot|s,a), P(\cdot|s,a)) + L_{P^\pi}^T d_{W_1}(D', D), && (22)
\end{aligned}
$$

in which we use (13) to get (21). The result follows from (22) by assuming the initial state distribution D to be shared across all the CMPs in $\boldsymbol{\mathcal{M}}$, and taking the supremum over $P', P \in \boldsymbol{\mathcal{M}}$. □

Theorem 6.2. *Let $\boldsymbol{\mathcal{M}}$ be a class of CMPs, let $\pi \in \Pi$ be a policy, and let $d_\pi^{\mathcal{M}}$ be the marginal state distribution over T steps induced by π in $\mathcal{M} \in \boldsymbol{\mathcal{M}}$. We can upper bound the π-diameter $\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi)$ as*

$$\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi) := \sup_{\mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}} d_{TV}(d_\pi^{\mathcal{M}'}, d_\pi^{\mathcal{M}}) \leq \sup_{P', P \in \boldsymbol{\mathcal{M}}} T \mathop{\mathbb{E}}_{\substack{s \sim d_\pi^{\mathcal{M}} \\ a \sim \pi(\cdot|s)}} d_{TV}(P'(\cdot|s,a), P(\cdot|s,a)).$$

Proof. The proof follows techniques from Metelli et al. (2018a), especially Proposition 3.1. Without loss of generality, we take $\mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}$. With some overloading of notation, we will alternatively identify a CMP with the tuple $\mathcal{M}$ or its transition model P. Let us start considering the TV between the marginal state distributions induced by π over $\mathcal{M}', \mathcal{M}$, we can write

$$\begin{aligned} & d_{TV}(d_\pi^{\mathcal{M}'}, d_\pi^{\mathcal{M}}) \\ &\quad = \frac{1}{2} \int_{\mathcal{S}} \left| d_\pi^{\mathcal{M}'}(s) - d_\pi^{\mathcal{M}}(s) \right| \mathrm{d}s = \frac{1}{2} \int_{\mathcal{S}} \left| \frac{1}{T} \sum_{t=0}^{T-1} d_{\pi,t}^{\mathcal{M}'}(s) - \frac{1}{T} \sum_{t=0}^{T-1} d_{\pi,t}^{\mathcal{M}}(s) \right| \mathrm{d}s && (23) \\ &\quad \leq \frac{1}{2T} \sum_{t=0}^{T-1} \int_{\mathcal{S}} \left| d_{\pi,t}^{\mathcal{M}'}(s) - d_{\pi,t}^{\mathcal{M}}(s) \right| \mathrm{d}s = \frac{1}{T} \sum_{t=0}^{T-1} d_{TV}(d_{\pi,t}^{\mathcal{M}'}, d_{\pi,t}^{\mathcal{M}}), && (24) \end{aligned}$$

where we use (13) to get (23). Then, we provide an upper bound to each term of the final sum in (24), i.e.,

$$\begin{aligned} & d_{TV}(d_{\pi,t}^{\mathcal{M}'}, d_{\pi,t}^{\mathcal{M}}) \\ &\quad = \frac{1}{2} \int_{\mathcal{S}} \left| d_{\pi,t}^{\mathcal{M}'}(s) - d_{\pi,t}^{\mathcal{M}}(s) \right| \mathrm{d}s \\ &\quad = \frac{1}{2} \int_{\mathcal{S}} \left| \int_{\mathcal{A}} \int_{\mathcal{S}} d_{\pi,t-1}^{\mathcal{M}'}(\overline{s}) \pi(\overline{a}|\overline{s}) P'(s|\overline{s}, \overline{a}) - d_{\pi,t-1}^{\mathcal{M}}(\overline{s}) \pi(\overline{a}|\overline{s}) P(s|\overline{s}, \overline{a}) \right| \mathrm{d}\overline{s} \, \mathrm{d}\overline{a} \, \mathrm{d}s && (25) \\ &\quad \leq \frac{1}{2} \int_{\mathcal{S}} \left| d_{\pi,t-1}^{\mathcal{M}'}(\overline{s}) - d_{\pi,t-1}^{\mathcal{M}}(\overline{s}) \right| \int_{\mathcal{A}} \int_{\mathcal{S}} \pi(\overline{a}|\overline{s}) P'(s|\overline{s}, \overline{a}) \, \mathrm{d}\overline{s} \, \mathrm{d}\overline{a} \, \mathrm{d}s && (26) \\ &\qquad + \frac{1}{2} \int_{\mathcal{S}} \int_{\mathcal{A}} d_{\pi,t-1}^{\mathcal{M}}(\overline{s}) \pi(\overline{a}|\overline{s}) \int_{\mathcal{S}} \left| P'(s|\overline{s}, \overline{a}) - P(s|\overline{s}, \overline{a}) \right| \mathrm{d}\overline{s} \, \mathrm{d}\overline{a} \, \mathrm{d}s && (27) \\ &\quad = d_{TV}(d_{\pi,t-1}^{\mathcal{M}'}, d_{\pi,t-1}^{\mathcal{M}}) + \mathop{\mathbb{E}}_{\substack{s \sim d_{\pi,t-1}^{\mathcal{M}} \\ a \sim \pi(\cdot|s)}} \Big[d_{TV}(P'(\cdot|s,a), P(\cdot|s,a)) \Big] && (28) \\ &\quad = \sum_{j=1}^{t-1} \mathop{\mathbb{E}}_{\substack{s \sim d_{\pi,j}^{\mathcal{M}} \\ a \sim \pi(\cdot|s)}} \Big[d_{TV}(P'(\cdot|s,a), P(\cdot|s,a)) \Big] + d_{TV}(D', D), && (29) \end{aligned}$$

where we use the temporal relation $d_{\pi,t}^{\mathcal{M}}(s') = \int_{\mathcal{S}} \int_{\mathcal{A}} d_{\pi,t-1}^{\mathcal{M}}(s) \pi(a|s) P(s'|s,a) \, \mathrm{d}s \, \mathrm{d}a$ to get (25), in which we sum and subtract $\int_{\mathcal{S}} \int_{\mathcal{A}} \int_{\mathcal{S}} d_{\pi,t-1}^{\mathcal{M}}(\overline{s}) \pi(\overline{a}|\overline{s}) P(s|\overline{s}, \overline{a}) \, \mathrm{d}s \, \mathrm{d}\overline{a} \, \mathrm{d}\overline{s}$ to obtain (26) and (27), and we repeatedly unroll (28) to write (29), noting that $d_{TV}(d_{\pi,0}^{\mathcal{M}'}, d_{\pi,0}^{\mathcal{M}}) = d_{TV}(D', D)$. Finally, we can plug (29) in (24) to get

$$
\begin{aligned}
& d_{TV}(d_\pi^{\mathcal{M}'}, d_\pi^{\mathcal{M}}) \\
& \qquad \leq \frac{1}{T} \sum_{t=0}^{T-1} d_{TV}(d_{\pi,t}^{\mathcal{M}'}, d_{\pi,t}^{\mathcal{M}}) \\
& \qquad \leq \frac{1}{T} \sum_{t=0}^{T-1} \sum_{j=1}^{t-1} \mathop{\mathbb{E}}_{\substack{s \sim d_{\pi,j}^{\mathcal{M}} \\ a \sim \pi(\cdot|s)}} \Big[d_{TV}(P'(\cdot|s,a), P(\cdot|s,a)) \Big] + d_{TV}(D', D) \\
& \qquad \leq \sum_{t=0}^{T-1} \int_{\mathcal{S}} \frac{1}{T} \sum_{j=0}^{T-1} d_{\pi,j}^{\mathcal{M}}(s) \mathop{\mathbb{E}}_{a \sim \pi(\cdot|s)} \Big[d_{TV}(P'(\cdot|s,a), P(\cdot|s,a)) \Big] \,\mathrm{d}s + d_{TV}(D', D) && (30) \\
& \qquad = \sum_{t=0}^{T-1} \mathop{\mathbb{E}}_{\substack{s \sim d_{\pi}^{\mathcal{M}} \\ a \sim \pi(\cdot|s)}} \Big[d_{TV}(P'(\cdot|s,a), P(\cdot|s,a)) \Big] + d_{TV}(D', D) && (31) \\
& \qquad = T \mathop{\mathbb{E}}_{\substack{s \sim d_{\pi}^{\mathcal{M}} \\ a \sim \pi(\cdot|s)}} \Big[d_{TV}(P'(\cdot|s,a), P(\cdot|s,a)) \Big] + d_{TV}(D', D), && (32)
\end{aligned}
$$

in which we have used (13) to obtain (31) from (30). The final result is straightforward from (31) by assuming the initial state distribution D to be shared across all the CMPs in $\boldsymbol{\mathcal{M}}$, and taking the supremum over $P', P \in \boldsymbol{\mathcal{M}}$. □

Theorem 6.3. *Let $\boldsymbol{\mathcal{M}}$ be a class of CMPs, let $\pi \in \Pi$ be a policy and $\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi)$ the corresponding π-diameter of $\boldsymbol{\mathcal{M}}$. Let $d_\pi^{\mathcal{M}}$ be the marginal state distribution over T steps induced by π in $\mathcal{M} \in \boldsymbol{\mathcal{M}}$, and let $\sigma_{\boldsymbol{\mathcal{M}}} \leq \sigma_{\mathcal{M}} := \inf_{s \in \mathcal{S}} d_\pi^{\mathcal{M}}(s), \forall \mathcal{M} \in \boldsymbol{\mathcal{M}}$. We can upper bound the entropy gap of the policy π within the model class $\boldsymbol{\mathcal{M}}$ as*

$$
\sup_{\mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}} \big| H(d_\pi^{\mathcal{M}'}) - H(d_\pi^{\mathcal{M}}) \big| \leq \big(\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi) \big)^2 / \sigma_{\boldsymbol{\mathcal{M}}} + \mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi) \log(1/\sigma_{\boldsymbol{\mathcal{M}}})
$$

Proof. Let us expand the entropy gap of the policy π as

$$
\begin{aligned}
& \sup_{\mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}} \big| H(d_\pi^{\mathcal{M}'}) - H(d_\pi^{\mathcal{M}}) \big| \\
& \qquad = \sup_{\mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}} \bigg\{ \bigg| - \int_{\mathcal{S}} d_\pi^{\mathcal{M}'}(s) \log d_\pi^{\mathcal{M}'}(s) \,\mathrm{d}s + \int_{\mathcal{S}} d_\pi^{\mathcal{M}}(s) \log d_\pi^{\mathcal{M}}(s) \,\mathrm{d}s \bigg| \bigg\} && (33) \\
& \qquad \leq \sup_{\mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}} \bigg\{ \bigg| \int_{\mathcal{S}} \Big(d_\pi^{\mathcal{M}}(s) - d_\pi^{\mathcal{M}'}(s) \Big) \log d_\pi^{\mathcal{M}}(s) \,\mathrm{d}s \bigg| + \bigg| \int_{\mathcal{S}} d_\pi^{\mathcal{M}'}(s) \Big(\log d_\pi^{\mathcal{M}'}(s) - \log d_\pi^{\mathcal{M}}(s) \Big) \,\mathrm{d}s \bigg| \bigg\} && (34) \\
& \qquad \leq \sup_{\mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}} \bigg\{ - \log \sigma_{\mathcal{M}} \int_{\mathcal{S}} \Big| d_\pi^{\mathcal{M}'}(s) - d_\pi^{\mathcal{M}'}(s) \Big| \,\mathrm{d}s + D_{KL}\big(d_\pi^{\mathcal{M}'} || d_\pi^{\mathcal{M}} \big) \bigg\} && (35) \\
& \qquad \leq \sup_{\mathcal{M}', \mathcal{M} \in \boldsymbol{\mathcal{M}}} \bigg\{ - \log \sigma_{\mathcal{M}} D_{TV}(d_\pi^{\mathcal{M}'}, d_\pi^{\mathcal{M}}) + \big(D_{TV}(d_\pi^{\mathcal{M}'}, d_\pi^{\mathcal{M}}) \big)^2 / \sigma_{\mathcal{M}} \bigg\} && (36) \\
& \qquad \leq \big(\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi) \big)^2 / \sigma_{\boldsymbol{\mathcal{M}}} - \mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi) \log \sigma_{\boldsymbol{\mathcal{M}}} && (37)
\end{aligned}
$$

in which we sum and subtract $\int_{\mathcal{S}} d_\pi^{\mathcal{M}'}(s) \log d_\pi^{\mathcal{M}}(s) \,\mathrm{d}s$ to obtain (34) from (33), $\log d_\pi^{\mathcal{M}}(s)$ is upper bounded with $\log \sigma_{\mathcal{M}}$ to get (35), and we use the reverse Pinsker's inequality $D_{KL}(p||q) \leq (D_{TV}(p,q))^2 / \inf_{x \in \mathcal{X}} q(x)$ (Csiszár & Talata, 2006, p. 1012 and Lemma 6.3) to obtain (10). Finally, we get the result by upper bounding $D_{TV}(d_\pi^{\mathcal{M}'}, d_\pi^{\mathcal{M}})$ with the π-diameter $\mathcal{D}_{\boldsymbol{\mathcal{M}}}(\pi)$ and $\sigma_{\mathcal{M}}$ with $\sigma_{\boldsymbol{\mathcal{M}}}$ in (36). □

C ALGORITHM

In this section, we provide additional details about the proposed method (MEMENTO). A full implementation of the algorithm can be found in the supplementary material.

C.1 The Benefits of the Baseline

In this section, we provide theoretical and empirical motivations to corroborate the use of the baseline $b = -\operatorname{VaR}_\alpha(H_\tau)$ into the Monte Carlo policy gradient estimator (Section 4, Equation 5). Thus, we compare the properties of two alternatives policy gradient estimator, with and without a baseline, i.e.,

$$\widehat{\nabla}_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) = \frac{1}{\alpha N}\sum_{i=1}^{N} f_{\tau_i}\, \big(\widehat{H}_{\tau_i} - \widehat{\operatorname{VaR}}_\alpha(H_{\tau_i})\big)\, \mathbb{1}(\widehat{H}_{\tau_i} \le \widehat{\operatorname{VaR}}_\alpha(H_\tau)),$$

$$\widehat{\nabla}^{b}_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) = \frac{1}{\alpha N}\sum_{i=1}^{N} f_{\tau_i}\, \big(\widehat{H}_{\tau_i} - \operatorname{VaR}_\alpha(H_{\tau_i}) - b\big)\, \mathbb{1}(\widehat{H}_{\tau_i} \le \widehat{\operatorname{VaR}}_\alpha(H_\tau)).$$

where $f_{\tau_i} = \sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_{t,\tau_i} | s_{t,\tau_i})$. The former ($\widehat{\nabla}_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}$) is known to be asymptotically unbiased (Tamar et al., 2015), but it is hampered by the estimation error of the VaR term to be subtracted to each $\widehat{H}_{\tau_i}$ in finite sample regimes (Kolla et al., 2019). The latter ($\widehat{\nabla}^{b}_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}$) introduces some bias in the estimate, but it crucially avoids the estimation error of the VaR term to be subtracted, as it cancels out with the baseline b. The following proposition, along with related lemmas, assesses the critical number of samples (n^*) for which an upper bound to the bias of $\widehat{\nabla}^{b}_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}$ is lower to the estimation error of $\widehat{\nabla}_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}$.

Lemma C.1. *The expected bias of the policy gradient estimate $\widehat{\nabla}^{b}_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})$ can be upper bounded as*

$$\mathop{\mathbb{E}}_{\substack{\mathcal{M}\sim\boldsymbol{\mathcal{M}}\\ \tau_i\sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}}}} \big[\text{bias}\big] = \mathop{\mathbb{E}}_{\substack{\mathcal{M}_i\sim\boldsymbol{\mathcal{M}}\\ \tau_i\sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}_i}}} \big[\nabla_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) - \widehat{\nabla}^{b}_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})\big] \le \mathcal{U}\alpha b,$$

where $\mathcal{U}$ is a constant such that $f_{\tau_i} \le \mathcal{U}$ for all τ_i.

Proof. This Lemma can be easily derived by means of

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\substack{\mathcal{M}_i\sim\boldsymbol{\mathcal{M}}\\ \tau_i\sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}_i}}} \big[\text{bias}\big]\\
&= \mathop{\mathbb{E}}_{\substack{\mathcal{M}_i\sim\boldsymbol{\mathcal{M}}\\ \tau_i\sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}_i}}} \Big[\nabla_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) - \widehat{\nabla}^{b}_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}})\Big]\\
&= \nabla_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) - \mathop{\mathbb{E}}_{\substack{\mathcal{M}_i\sim\boldsymbol{\mathcal{M}}\\ \tau_i\sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}_i}}} \Big[\frac{1}{\alpha N}\sum_{i=1}^{N} f_{\tau_i}\, \big(\widehat{H}_{\tau_i} - \operatorname{VaR}_\alpha(H_{\tau_i}) - b\big)\, \mathbb{1}(\widehat{H}_{\tau_i} \le \widehat{\operatorname{VaR}}_\alpha(H_\tau))\Big]\\
&= \nabla_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) - \mathop{\mathbb{E}}_{\substack{\mathcal{M}\sim\boldsymbol{\mathcal{M}}\\ \tau\sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}}}} \Big[f_{\tau}\, \big(\widehat{H}_{\tau} - \operatorname{VaR}_\alpha(H_{\tau}) - b\big)\, \mathbb{1}(\widehat{H}_{\tau} \le \widehat{\operatorname{VaR}}_\alpha(H_\tau))\Big] && (38)\\
&= \nabla_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) - \nabla_{\boldsymbol{\theta}}\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) + \mathop{\mathbb{E}}_{\substack{\mathcal{M}\sim\boldsymbol{\mathcal{M}}\\ \tau\sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}}}} \Big[f_{\tau}\, b\, \mathbb{1}(\widehat{H}_{\tau} \le \widehat{\operatorname{VaR}}_\alpha(H_\tau))\Big] && (39)\\
&= \mathop{\mathbb{E}}_{\substack{\mathcal{M}\sim\boldsymbol{\mathcal{M}}\\ \tau\sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}}}} \Big[f_{\tau}\, b\, \mathbb{1}(\widehat{H}_{\tau} \le \widehat{\operatorname{VaR}}_\alpha(H_\tau))\Big] \le \mathcal{U}\alpha b, && (40)
\end{aligned}$$

where (39) follows from (38) by noting that the estimator without the baseline term is unbiased (Tamar et al., 2015), and (40) is obtained by upper bounding f_τ with $\mathcal{U}$ and noting that $\mathbb{E}_{\substack{\mathcal{M}\sim\boldsymbol{\mathcal{M}}\\ \tau\sim p_{\pi_{\boldsymbol{\theta}},\mathcal{M}}}} \big[\mathbb{1}(\widehat{H}_\tau \le \widehat{\operatorname{VaR}}_\alpha(H_\tau))\big] = \alpha$. □

Lemma C.2 (VaR concentration bound from L.A. et al. (2020))**.** *Let X be a continuous random variable with a pdf f_X for which there exist $\eta, \Delta > 0$ such that $f_X(x) > \eta$ for all $x \in \big[\operatorname{VaR}_\alpha(X) - \frac{\Delta}{2}, \operatorname{VaR}_\alpha(X) + \frac{\Delta}{2}\big]$. Then, for any $\epsilon > 0$ we have*

$$Pr\big[|\,\widehat{\operatorname{VaR}}_\alpha(X)_\alpha - \operatorname{VaR}_\alpha(X)| \ge \epsilon\big] \le 2\exp\big(-2n\eta^2 \min(\epsilon^2, \Delta^2)\big),$$

where $n \in \mathbb{N}$ is the number of samples employed to estimate $\widehat{\operatorname{VaR}}_\alpha(X)$.

Proposition C.3. *Let $\widehat{\nabla}_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\mathcal{M}}(\pi_{\boldsymbol{\theta}})$ and $\widehat{\nabla}^{b}_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\mathcal{M}}(\pi_{\boldsymbol{\theta}})$ be policy gradient estimates with and without a baseline. Let f_H be the pdf of H_τ, for which there exist $\eta, \Delta > 0$ such that $f_H(H_\tau) > \eta$ for all $H_\tau \in \big[\mathrm{VaR}_\alpha(H_\tau) - \frac{\Delta}{2}, \mathrm{VaR}_\alpha(H_\tau) + \frac{\Delta}{2} \big]$. The number of samples n^* for which the estimation error ϵ of $\widehat{\nabla}_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\mathcal{M}}(\pi_{\boldsymbol{\theta}})$ is lower than the bias of $\widehat{\nabla}^{b}_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\mathcal{M}}(\pi_{\boldsymbol{\theta}})$ with at least probability $\delta \in (0,1)$ is given by*

$$n^* = \frac{\log 2/\delta}{2\eta^2 \min(\mathcal{U}^2 \alpha^2 b^2, \Delta^2)}.$$

Proof. The proof is straightforward by considering the estimation error ϵ of $\widehat{\nabla}_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\mathcal{M}}(\pi_{\boldsymbol{\theta}})$ equal to the upper bound of the bias of $\widehat{\nabla}^{b}_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\mathcal{M}}(\pi_{\boldsymbol{\theta}})$ from Lemma C.1, i.e., $\epsilon = \mathcal{U} \alpha b$. Then, we set $\delta = 2 \exp\big(-2n^* \eta^2 \min(\mathcal{U}^2 \alpha^2 b^2, \Delta^2) \big)$ from Lemma C.2, which gives the result through simple calculations. □

The Proposition C.3 proves that there is little incentive to choose the policy gradient estimator $\widehat{\nabla}_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\mathcal{M}}$ when the number of trajectories is lower than n^*, as its estimation error would exceed the bias introduced by the alternative estimator $\widehat{\nabla}^{b}_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\mathcal{M}}$. Unfortunately, it is not easy to compute n^* in our setting, as we do not assume to know the distribution of H_τ, but the requirement is arguably seldom matched in practice.

Moreover, we can empirically show that the baseline $b = -\mathrm{VaR}_\alpha(H_\tau)$ might benefit the variance of the policy gradient estimation, at the expense of the additional bias which is anyway lower than the estimation error of $\widehat{\nabla}_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\mathcal{M}}$. In Figure 7 (left), we can see that the exploration performance $\mathcal{E}^{\alpha}_{\mathcal{M}}$ obtained by MEMENTO with and without the baseline is essentially the same in the illustrative *GridWorld with Slope* domain. Whereas Figure 7 (right) suggests a slightly inferior variance for the policy gradient estimate employed by MEMENTO with the baseline.

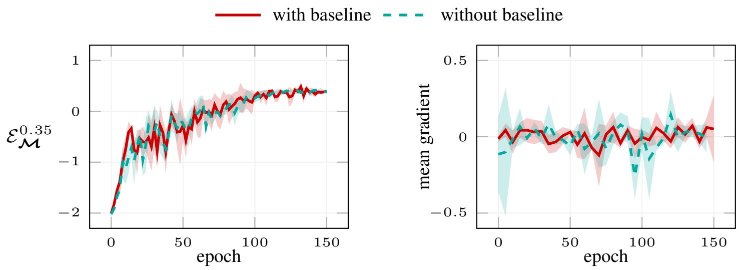


Figure 7: Comparison of the exploration performance $\mathcal{E}^{0.35}_{\mathcal{M}}$ (left) and sampled gradients of the policy mean (right) achieved by MEMENTO ($\alpha = 0.35$) with and without the baseline $b = -\mathrm{VaR}_\alpha(H_\tau)$ in the policy gradient estimation (5). We provide 95% c.i. over 4 runs.

C.2 Importance Weighted Entropy Estimation

As done in (Mutti et al., 2020), we build on the estimator in (3) to consider the case in which the target policy $\pi_{\boldsymbol{\theta}'}$ differs from the sampling policy $\pi_{\boldsymbol{\theta}}$. The idea is to combine two successful policy-search methods. The first one is POIS (Metelli et al., 2018b), to perform the optimization offline via importance sampling, allowing for an efficient exploitation of the samples collected with previous policies. We thus adopt an Importance-Weighted (IW) entropy estimator (Ajgl & Šimandl, 2011) of the form

$$\widehat{H}^{\mathrm{IW}}_{\tau_i} = -\sum_{t=0}^{T-1} \frac{\sum_{j \in \mathcal{N}^k_t} w_j}{k} \ln \frac{\Gamma(\frac{p}{2}+1) \sum_{j \in \mathcal{N}^k_t} w_j}{\big\| s_{t,\tau_i} - s^{k\text{-NN}}_{t,\tau_i} \big\|^p \pi^{\frac{p}{2}}} + \ln k - \Psi(k), \tag{41}$$

where $\ln k - \Psi(k)$ is a bias correction term in which Ψ is the Digamma function, $\mathcal{N}^k_i$ is the set of indices of the k-NN of s_{t,τ_i}, and w_j are the normalized importance weights of samples s_{j,τ_i}. To compute these importance weights we consider a dataset $\mathcal{D} = \{s_{t,\tau_i}\}_{t=0}^{T-1}$ by looking each state

Algorithm 2 MEMENTO

Input: initial policy $\pi_{\boldsymbol{\theta}_0}$, exploration horizon T, number of trajectories N, batch-size B, percentile α, learning rate β, trust-region threshold δ, sampling distribution $p_{\boldsymbol{\mathcal{M}}}$

Output: exploration policy $\pi_{\boldsymbol{\theta}_h}$

```
for epoch $= 0, 1, \dots$, until convergence do
    for $i = 1, 2, \dots, N$ do
        sample an environment $\mathcal{M}_i \sim p_{\boldsymbol{\mathcal{M}}}$
        for $j = 1, 2, \dots, B$ do
            sample a trajectory $\tau_j \sim p_{\pi_{\boldsymbol{\theta}}, \mathcal{M}_i}$ of length $T$
        end for
    end for
    initialize dataset $\mathcal{D} = \emptyset$, off-policy step $h = 0$ and $\boldsymbol{\theta}_h = \boldsymbol{\theta}$
    while $\widehat{D}_{KL}(\pi_{\boldsymbol{\theta}_0} || \pi_{\boldsymbol{\theta}_h}) \leq \delta$ do
        for $j = 1, 2, \dots, B$ do
            estimate $H_{\tau_j}$ with (41)
            append $\widehat{H}_{\tau_j}$ to $\mathcal{D}$
        end for
        sort $\mathcal{D}$ and split it in $\mathcal{D}_\alpha$ and $\mathcal{D}_{1-\alpha}$
        compute a gradient step $\boldsymbol{\theta}_{h+1} = \boldsymbol{\theta}_h + \beta \widehat{\nabla}_{\boldsymbol{\theta}_h} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}_h})$
        $h \leftarrow h + 1$
    end while
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_h$
end for
```

encountered in a trajectory as an unweighted particle. Then, we expand it as $\mathcal{D}_{\tau_i} = \{(\tau_{i,t}, s_t)\}_{t=0}^{T-1}$, where $\tau_{i,t} = (s_{0,\tau_i}, \dots, s_{t,\tau_i})$ is the portion of the trajectory that leads to state s_{t,τ_i}. This allows to associate each particle s_{t,τ_i} to its importance weight $\widehat{w}_t$ and normalized importance weight w_t for any pair of target ($\pi_{\boldsymbol{\theta}'}$) and sampling ($\pi_{\boldsymbol{\theta}}$) policies:

$$\widehat{w}_t = \frac{p(\tau_{i,t} | \pi_{\boldsymbol{\theta}'})}{p(\tau_{i,t} | \pi_{\boldsymbol{\theta}})} = \prod_{z=0}^{t} \frac{\pi_{\boldsymbol{\theta}'}(a_{z,\tau_i} | s_{z,\tau_i})}{\pi_{\boldsymbol{\theta}}(a_{z,\tau_i} | s_{z,\tau_i})}, \qquad\qquad w_t = \frac{\widehat{w}_t}{\sum_{n=0}^{T-1} \widehat{w}_n}.$$

The estimator in (41) is then optimized via gradient ascent. The second policy-search method used during the optimization is TRPO (Schulman et al., 2015), to perform subsequent optimizations within a trust-region around the current policy. The trust-region constraint is obtained by imposing

$$\widehat{D}_{KL}(\pi_{\boldsymbol{\theta}'} || \pi_{\boldsymbol{\theta}}) = \frac{1}{T} \sum_{t=0}^{T-1} \ln \frac{k/T}{\sum_{j \in \mathcal{N}_t^k} w_j} \leq \delta,$$

where $\widehat{D}_{KL}(\pi_{\boldsymbol{\theta}'} || \pi_{\boldsymbol{\theta}})$ is a non-parametric IW k-NN estimate of the Kullback-Leibler (KL) divergence (Ajgl & Šimandl, 2011). Its value is computed as in (Mutti et al., 2020), by considering the entire batch of trajectories collected to execute the off-policy optimization steps as a single trajectory.

C.3 Algorithmic Details of MEMENTO

In this section, we provide an extended pseudocode (Algorithm 2) of MEMENTO, along with some additional comments.

Given a probability distribution $p_{\boldsymbol{\mathcal{M}}}$, the algorithm operates by iteratively sampling an environment $\mathcal{M}_i \in \boldsymbol{\mathcal{M}}$ drawn according to $p_{\boldsymbol{\mathcal{M}}}$ and then sampling B trajectories of length T from it using $\pi_{\boldsymbol{\theta}}$, where B is the dimension of each mini-batch. Then, the estimate of the entropy of each mini-batch $\widehat{H}_{\tau_j}$ is computed by means of the estimator in (41) and appended to the dataset $\mathcal{D}$. Once obtained the final dataset $\mathcal{D}$, we can straightforwardly derive a risk-sensitive policy update by just subsampling from it, so that to keep only the realizations below the α-percentile. This can be easily done by sorting $\mathcal{D}$ in ascending order and considering only the αN first mini-batches. Then, we can compute the gradient as follows:

$$\widehat{\nabla}_{\boldsymbol{\theta}} \mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}(\pi_{\boldsymbol{\theta}}) = \frac{1}{\alpha N} \sum_{i=1}^{N} f_{\tau_i} \ \widehat{H}_{\tau_i} \ \mathbb{1}(\widehat{H}_{\tau_i} \leq \widehat{\mathrm{VaR}}_{\alpha}(H_{\tau})).$$

The operations carried out once all the trajectories have been sampled are executed in a fully off-policy manner, in which we repeat the same steps until the trust-region boundary is reached or until the number of off-policy iterations exceeds a specified limit. The reason why we introduce an additional parameter B, instead of considering one trajectory at a time, is due to the fact that a significant amount of samples (see the parameters in Table 1) is needed to obtain a reliable estimate of the entropy, noting that the entropy estimator is only asymptotically unbiased.

D Experiments

In this section, we report an extensive description of the conducted experiments, with the corresponding hyperparameter values and some additional plots and experiments.

D.1 Environments

We use three different environments in our experiments. The first one is a custom implementation of a gridworld, coded from scratch. The second one is an adapted version of the rllab Ant-Maze environment (Duan et al., 2016).

D.1.1 GridWorld with Slope

In *GridWorld with Slope* (2D states, 2D actions), the agent can move inside a map composed of four rooms connected by four narrow hallways, by choosing at each step how much to move on the x and y axes. The side of the environment measures 2 units and the maximum viable space of the agent at each step is 0.2. Thus, the agent needs around 10 steps to go from one side to the other on a straight line. When the agent collides with the external borders or with the internal walls, it is re-positioned according to a custom function. This is done not only to make the interaction more realistic, but also to limit the possibility to have a negative infinite entropy resulting from the k-NN computation, which can occur when the samples are too close and the value of the parameter k is not high enough. This precaution is particularly useful in our scenario, due to the presence of a slope, and especially in the *adversarial* configuration GWN, because of the initial position of the agent, which is sampled in a small square in the top-right corner. It is easy to see that in the first epochs in the GWN environment, the agent would repeatedly collide with the top-border, leading in general to a much more lower entropy w.r.t. to GWS.

The slope is applied only in the upper half of the environment, since we found this to be a good trade-off between the intention of maintaining a difference in terms of risk among the two configurations and the overall complexity of the exploration. Indeed, we noted that by applying the slope to the whole GridWorld, the advantage in terms of exploration entailed by the risk-averse approach is even higher, but it struggles to explore the bottom states of the environment with a reasonable number of samples. The slope is computed as $s \sim \mathcal{N}(\frac{\Delta_{max}}{2}, \frac{\Delta_{max}}{20})$, where $\Delta_{max} = 0.2$ is the maximum step that the agent can perform.

D.1.2 MultiGrid

In *MultiGrid*, everything works as in *GridWorld with Slope*, but we indeed have 10 configurations. These environments differ for both the shape and the type of slope to which they are subject to. The *adversarial* configuration is still GWN, but the slope is computed as $s \sim \mathcal{N}(\frac{\Delta_{max}}{2.6}, \frac{\Delta_{max}}{20})$, where $\Delta_{max} = 0.2$. The other 9 gridworlds have instead a different arrangement of the walls (see the heatmaps in Figure 10) and the slope, computed as $s \sim \mathcal{N}(\frac{\Delta_{max}}{3.2}, \frac{\Delta_{max}}{20})$ with $\Delta_{max} = 0.2$, is applied over the entire environment. Two configurations are subject to south-facing slope, three to east-facing slope, one to south-east-facing slope and three to no slope at all.

D.1.3 Ant Stairs

We adopt the Ant-Maze environment (29D states, 8D actions) of rllab (Duan et al., 2016) and we exploit its malleability to build two custom configurations which could fit our purposes. The adverse configuration consists of a narrow ascending staircase (*Ant Stairs Up*) made up of an initial square (the initial position of the Ant), followed by three blocks of increasing height. The simpler configuration consists of a wide descending staircase (*Ant Stairs Down*), made up of 3×3 blocks of decreasing height and a final 1×3 flat area. Each block has a side length slightly greater than the Ant size. A visual representation of such settings is provided in Figure 11. During the *learning to explore* phase, $\mathcal{E}^{\alpha}_{\boldsymbol{\mathcal{M}}}$ is maximized over the x,y spatial coordinates of the ant's torso.

D.1.4 MiniGrid

We use the MiniGrid suite (Chevalier-Boisvert et al., 2018), which consists of a set of fast and light-weighted gridworld environments. The environments are partially observable, with the dimension of the agent's field of view having size $7 \times 7 \times 3$. Both the observation space $\mathcal{S}$ and the action space $\mathcal{A}$ are discrete, and in each tile of the environment there can be only one object at the same time. The set of objects is $O = \{wall, floor, lava, door, key, ball, box, goal\}$. The agent can move inside the grid and interact with these objects according to their properties. In particular, the actions comprise turning left, turning right, moving forward, picking up an object, dropping an object and toggling, i.e., interacting with the objects (e.g., to open a door). We exploit the suite's malleability to build two custom environments. The simpler one has a size of 18×18, and it simply contains some sparse walls. The adverse configuration is smaller, 10×10, and is characterized by the presence of a door at the top of a narrow hallway. The door is closed but not locked, meaning that the agent can open it without using a key. Moreover, we modify the movement of the agent so that the direction is given by the bottom of the triangle instead of the top. The intuition is that by doing this we are essentially changing the shape of the agent, hence causing the policy to struggle in the exploration.

As regards the training procedure, everything remains the same, except for two differences. The first difference is that the k-NN computation is performed on the representation space generated by a fixed random encoder. Note that this random encoder is not part of the policy. It is randomly initialized and not updated during the training in order to produce a more stable entropy estimate. In addition, before computing the distances, we apply to its output a random Gaussian noise $\epsilon \sim \mathcal{N}(0.001, 0.001)$ truncated in $[0, 0.001]$. We do this to avoid the aliasing problem, which occurs when we have many samples (more than k) in the same position, thus having zero distance and producing a negative infinite entropy estimate. The homogeneity of the MiniGrid environments in terms of features make this problem more frequent. The second difference is the addition of a bootstrapping procedure for the easy configuration, meaning that we use only a subset of the mini-batches of the easy configuration to update the policy. Especially, we randomly sample a number of mini-batches that is equal to the dimension of the $\mathcal{D}_\alpha$ dataset so that Neutral uses the same number of samples of MEMENTO. The reason why we avail this method is to avoid a clear advantage for Neutral in learning effective representations, since it usually access more samples than MEMENTO. Note that it is not a stretch, since we are essentially balancing the information available to the two algorithms.

D.2 Class of Policies

In all the experiments but one the policy is a Gaussian distribution with diagonal covariance matrix. It takes as input the environment state features and outputs an action vector $a \sim \mathcal{N}(\mu, \sigma^2)$. The mean μ is state-dependent and is the downstream output of a densely connected neural network. The standard deviation is state-independent and it is represented by a separated trainable vector. The dimension of μ, σ, and a vectors is equal to the action-space dimension of the environment. The only experiment with a different policy is the MiniGrid one, for which we adopt the architecture recently proposed by (Seo et al., 2021). Thus, we use a random encoder made up of 3 convolutional layers with kernel 2, stride 1, and padding 0, each activated by a ReLU function, and with 16, 32 and 64 filters respectively. The first ReLU is followed by a 2D max pooling layer with kernel 2. The output of the encoder is a 64 dimensional tensor, which is then fed to a feed-forward neural network with two fully-connected layers with hidden dimension 64 and a Tanh activation function.

D.3 Hyperparameter Values

D.3.1 Learning to Explore

In Table 1, we report the parameters of MEMENTO and Neutral that are used in the experiments described in Section 5.1, Section 5.2, Section 5.4 and Section 5.5.

Table 1: MEMENTO and Neutral Parameters

	GridWorld with Slope	MultiGrid	Ant	MiniGrid
Number of epochs	150	50	400	300
Horizon (T)	400	400	400	150
Number of traj. (N)	200	500	150	100
Mini-batch dimension (B)	5	5	5	5
α-percentile	0.35	0.1	0.2	0.3
Sampling dist. ($p_{\mathcal{M}}$)	$[0.8, 0.2]$	$[0.1, \ldots, 0.1]$	$[0.8, 0.2]$	$[0.8, 0.2]$
KL threshold (δ)	15	15	15	15
Learning rate (β)	10^{-5}	10^{-5}	10^{-5}	10^{-5}
Number of neighbors (k)	30	30	500	50
Policy hidden layer sizes	(300,300)	(300,300)	(400,300)	*
Policy hidden layer act. funct.	ReLU	ReLU	ReLU	*
Number of seeds	4	4	4	4

* See Section D.2 for full details on the architecture.

D.3.2 Reinforcement Learning

In Table 2, we report the TRPO parameters that are used in the experiments described in Section 5.3, Section 5.4, Section 5.5 and Section 5.7.

Table 2: TRPO Parameters for Goal-Based RL

	GridWorld with Slope	MultiGrid	Ant	MiniGrid
Number of Iter.	100	100	100	200
Horizon	400	400	400	150
Sim. steps per Iter.	1.2×10^4	1.2×10^4	4×10^5	7.5×10^3
δ_{KL}	10^{-4}	10^{-4}	10^{-2}	10^{-4}
Discount (γ)	0.99	0.99	0.99	0.99
Number of seeds	50	50	8	13
Number of goals	50	50	8	13

D.3.3 Meta-RL

In Table 3 and Table 4, we report the MAML and DIAYN parameters that are used in the experiments described in Section 5.7, in order to meta-train a policy on the *GridWorld with Slope* and *MultiGrid* domains. For MAML, we adopted the codebase at https://github.com/tristandeleu/pytorch-maml-rl, while for DIAYN, we used the original implementation.

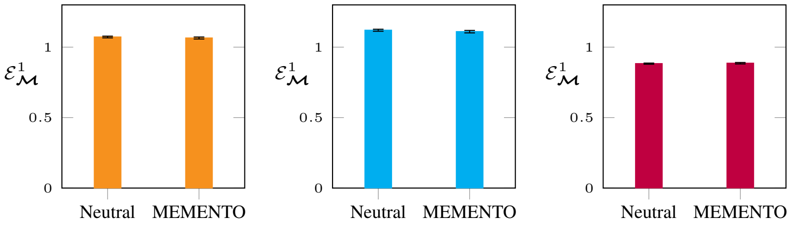
Table 3: MAML Parameters

	GridWorld with Slope	MultiGrid
Number of batches	200	200
Meta batch size	20	20
Fast batch size	30	30
Num. of Grad. Step	1	1
Horizon	400	400
Fast learning rate	0.1	0.1
Policy hidden layer sizes	(300,300)	(300,300)
Policy hidden layer act. function	ReLU	ReLU
Number of seeds	4	4

Table 4: DIAYN Parameters

	GridWorld with Slope	MultiGrid
Number of epochs	1000	1000
Horizon	400	400
Number of skills	20	20
Learning rate	3×10^{-4}	3×10^{-4}
Discount (γ)	0.99	0.99
Policy hidden layer sizes	(300,300)	(300,300)
Policy hidden layer act. function	ReLU	ReLU
Number of seeds	4	4

D.4 Counterexample: When Percentile Sensitivity Does Not Matter



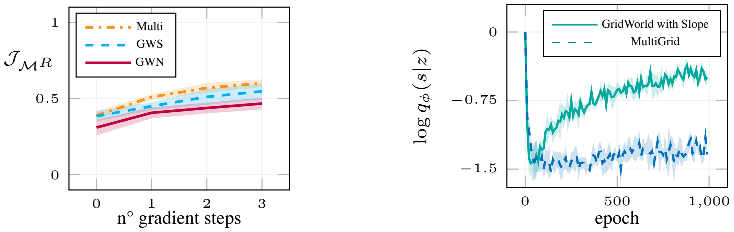
(a) $p_{\mathcal{M}} = [0.8, 0.2]$ (b) $p_{\mathcal{M}} = [1, 0]$ (c) $p_{\mathcal{M}} = [0, 1]$

Figure 8: Comparison of the exploration performance $\mathcal{E}^1_{\mathcal{M}}$ obtained by MEMENTO ($\alpha = 0.35$) and Neutral ($\alpha = 1$) in the *GridWorld Counterexample* domain. The polices are trained (50 epochs, 8×10^4 samples per epoch) on the configuration **(a)** and tested on **(a, b, c)**. We provide 95% c.i. over 4 runs.

In this section, we provide a convenient example to confirm the fact that there are classes of environments in which we would not need any particularly smart solution for the multiple environments problem, beyond a naïve, risk-neutral approach. We consider two GridWorld environments that differ for the shape of the traversable area, sampled according to $p_{\mathcal{M}} = [0.8, 0.2]$, and we run MEMENTO with $\alpha = 0.35$ and Neutral ($\alpha = 1$), obtaining the two corresponding exploration policies. In Figure 8 we show the performance (measured by $\mathcal{E}^1_{\mathcal{M}}$) obtained by executing those policies on each setting. Clearly, regardless of what configuration we consider, there is no advantage deriving from the use of a risk-averse approach as MEMENTO, meaning that the class of environments $\mathcal{M}$ is balanced in terms of hardness of exploration.

D.5 Further Details on Meta-RL Experiments

In this section, we provide additional details on the experiments of Section 5.7. Especially, we show that MAML does perform well on its own objective, which is to learn a fast-adapting policy



(a) MAML: Fast Adaptation

(b) DIAYN: Discriminability

Figure 9: We illustrate the fast-adapting behavior of MAML in the *GridWorld with Slope* **(a)**, and the skills discriminability of DIAYN as a function of learning epochs **(b)**. We provide 95% c.i. over 4 runs.

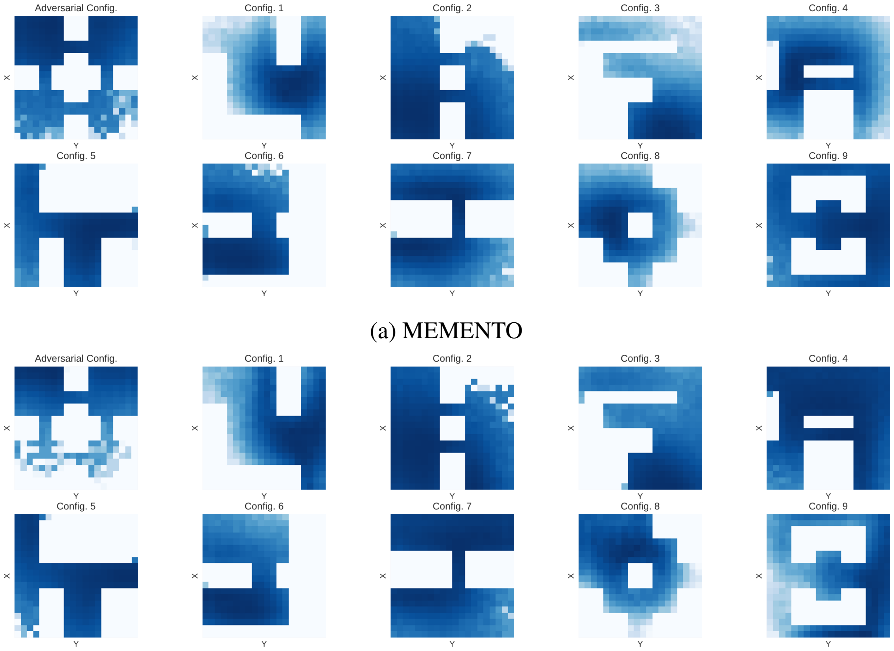
during meta-training (Figure 9a). Instead, in Figure 9b we highlight the performance measure of DIAYN Eysenbach et al. (2018). In particular, the more $\log q_\phi(s|z)$ grows with the learning epochs, the better is the intrinsic reward we feed to MAML+DIAYN. Clearly, DIAYN struggles to deal with the larger *MultiGrid* class of environments, which explains the inferior performance of MAML+DIAYN in this domain.

D.6 Additional Visualizations

In this section, we provide some additional visualizations, which are useful to better understand some of the domains used in the experiments of Section 5. In Figure 10 we report the state-visitation frequencies achieved by MEMENTO (Figure 10a) and Neutral (Figure 10b) in each configuration of the *MultiGrid* domain. Clearly, MEMENTO manages to obtain a better exploration in the adversarial configuration w.r.t. Neutral, especially in the bottom part of the environment, which is indeed the most difficult part to visit. On the other environments, the performance is overall comparable. In Figure 11 we show a render of the *Ant Stairs* domain, illustrating both the environments used in the experiments of Section 5.5. Note that the front walls are hidden to allow for a better visualization.

E Future Directions

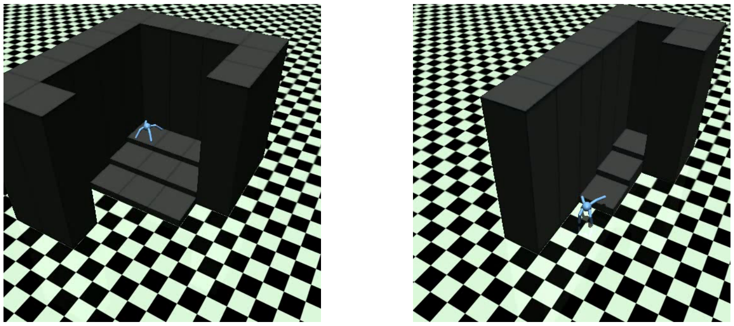
First, it is worth mentioning an alternative setting in which MEMENTO can be employed with benefit (with little or no modifications). This is the the *robust reward-free exploration* problem, in which we just have to replace the class of environments with a single CMP specified under uncertainty (Satia & Lave Jr, 1973). Secondly, in this work we focused on a specific solution for an essentially multi-objective problem, by establishing a preference over the environments through the CVaR objective. Instead, a future direction could pursue learning a direct approximation of the Pareto frontier (Parisi et al., 2016) of the exploration strategies over multiple environments. Another promising direction is to assume some control over the class distribution during the learning to explore process, either by an external supervisor or by the agent itself (Metelli et al., 2018a). Lastly, future work may establish regret guarantees for the reward-free exploration problem over multiple environments, in a similar flavor to the reward-free RL problem in a single environment (Jin et al., 2020).



(a) MEMENTO

(b) Neutral

Figure 10: Heatmaps of the state visitations (200 trajectories) induced by the exploration policies trained with MEMENTO ($\alpha = 0.1$) **(a)** and Neutral ($\alpha = 1$) **(b)** in the *MultiGrid* domain.



(a) Ant Stairs Down

(b) Ant Stairs Up

Figure 11: Illustration of the *Ant Stairs* domain. We show a render of the *Ant Stairs Down* environment **(a)** and of the adverse *Ant Stairs Up* environment **(b)**.