Robust Policy Learning over Multiple Uncertainty Sets

Annie Xie [1] **Shagun Sodhani** [2] **Chelsea Finn** [1] **Joelle Pineau** [2] **Amy Zhang** [2]

Abstract

Reinforcement learning (RL) agents need to be robust to variations in safety-critical environments. While system identification methods provide a way to infer the variation from online experience, they can fail in settings where fast identification is not possible. Another dominant approach is robust RL which produces a policy that can handle worst-case scenarios, but these methods are generally designed to achieve robustness to a single uncertainty set that must be specified at train time. Towards a more general solution, we formulate the multi-set robustness problem to learn a policy robust to different perturbation sets. We then design an algorithm that enjoys the benefits of both system identification and robust RL: it reduces uncertainty where possible given a few interactions, but can still act robustly with respect to the remaining uncertainty. On a diverse set of control tasks, our approach demonstrates improved worst-case performance on new environments compared to prior methods based on system identification and on robust RL alone.

1. Introduction

Uncertainty is a prevalent challenge in most realistic reinforcement learning (RL) settings. Our work studies the uncertainty that arises when an agent is transferred to a new environment, after training on similar tasks related through a common set of underlying parameters, often referred to as the *context*. In safety-critical settings, we often care about the agent's worst-case performance on the distribution of plausible environments.

Robust RL is one of the primary approaches to this problem as it aims to learn a policy that performs well under worst-case perturbations to the context (Rajeswaran et al., 2016; Pinto et al., 2017; Mankowitz et al., 2020; Tessler et al.,

[1]Stanford University [2]Facebook AI Research. Correspondence to: Annie Xie <anniexie@stanford.edu>.
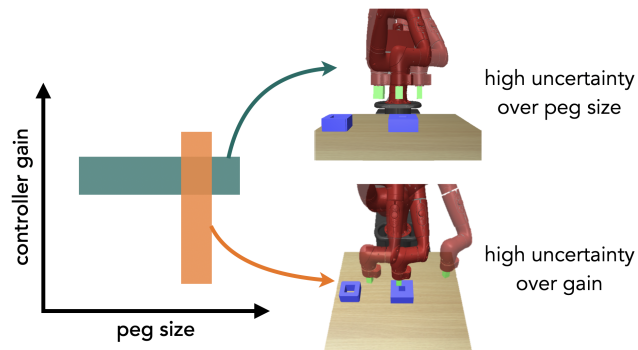
Figure 1. Two peg-insertion tasks. The first (green) presents high uncertainty in the peg size, while the second (orange) has high uncertainty in the controller gain.

2019; Vinitsky et al., 2020; Abraham et al., 2020). However, these solutions require a prior uncertainty set over the context, i.e., a set of its possible values, for the *test-time* environment to learn the robust policy for this set at train time. Building in this prior ahead of time can limit the flexibility of the resulting policy: a large uncertainty set produces an overly conservative policy that can potentially underperform in all environments, but a small uncertainty set can fail to represent the target environment (Mozian et al., 2020).

We, therefore, formulate and study the *multi-set robustness* problem (illustrated in Fig. 2) whose goal is to learn a policy with strong worst-case performance on new uncertainty sets. Since the optimal robust policy varies across different perturbation sets, we incorporate the uncertainty set as contextual information to the agent and learn a generalized set-conditioned policy.

However, naively contextualizing existing robust methods with the uncertainty set can still be sub-optimal as these methods do not reduce uncertainty over the context. In particular, the parameters that make up the context can sometimes be quickly identified, given a history of interactions. For example, consider a robot inserting a peg in one of the boxes in Fig. 1. The box closer to the robot only fits smaller pegs, while the box to the left can accommodate all sizes. Hence, the optimal policy should select the closer box for smaller pegs and the faraway box for larger ones. While the size of the peg cannot be estimated without additional trial and error, the strength of the robot's actions, on the other hand, can be identified after taking a handful of actions. Performing online system identification to reduce uncertainty
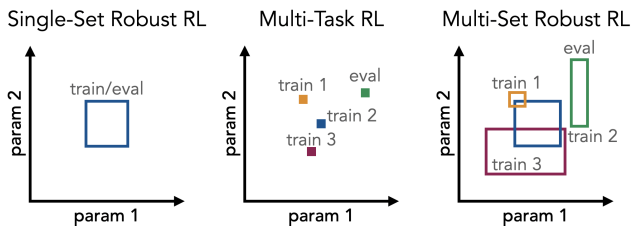
Figure 2. An illustration of the robust, multi-task, and multi-set robust RL setups. Robust RL learns a policy for a *single uncertainty set*, while multi-task RL optimizes a policy to solve a *collection of tasks*. Finally, multi-set robust RL aims to learn a policy that performs well with respect to a *collection of uncertainty sets*.

over this parameter can allow the agent to solve the task more effectively. Thus, we propose to enhance existing robust RL solutions by introducing uncertainty set-awareness and system identification capabilities.

To this end, we formulate the multi-set robustness problem to learn a policy that is robust to multiple uncertainty sets. We then propose a framework that consists of a probabilistic system identification model and our multi-set robust policy, which we condition on the uncertainty set inferred by the model. We call our approach System Identification and Risk-Sensitive Adaptation (SIRSA). We compare SIRSA to prior methods based on robust RL and on system identification on a suite of continuous control tasks, including the 7-DoF peg insertion task in Fig. 1, and find substantial improvements in worst-case performance on new environments. We also find that the policy learned with SIRSA can transfer to environments with misspecified priors and with non-stationary dynamics.

2. Related Work

Our work is at the intersection of robust control, Bayesian RL, and multi-task and meta-RL, which we review below.

Robust and risk-sensitive RL. The robust Markov decision process is a worst-case formulation of the RL problem with uncertainty in the transition probabilities, but can only be tractably solved in the tabular case (Morimoto & Doya, 2000; Nilim & El Ghaoui, 2005; Iyengar, 2005; Lim et al., 2013; Roy et al., 2017; Badrinath & Kalathil, 2021). Subsequent formulations treat the uncertainty as perturbations from a parameterized adversary, which can occur in the observations (Zhang et al., 2020a), transition dynamics (Pinto et al., 2017; Mankowitz et al., 2020; Tessler et al., 2019; Vinitsky et al., 2020), the reward function (Lin et al., 2020; Zahavy et al., 2020), or the underlying parameters of the environment (Rajeswaran et al., 2016; Abraham et al., 2020; Mehta et al., 2020). Our work formulates a new robust control problem: robustness to a distribution over uncertainty sets. These uncertainty sets characterize uncertainty over a set of unobserved environment parameters.

Worst-case solutions can be overly pessimistic, prompting

the adoption of a different risk metric, the conditional value-at-risk (CVaR), which allows control over the level of risk sensitivity through the hyperparameter α (Rockafellar et al., 2000). In RL, the CVaR objective can be optimized by sampling (Tamar et al., 2015), a distributional critic (Tang et al., 2019), or an ensemble of environment models (Mordatch et al., 2015; Rajeswaran et al., 2016; Derman et al., 2018; Mankowitz et al., 2020). We implement a sampling-based approximation to the CVaR objective, using a learned multi-task critic.

Bayesian RL and system identification. Another way to handle uncertainty in RL is with the Bayes-adaptive MDP (BAMDP) (Duff, 2002; Ross et al., 2007) (see Ghavamzadeh et al. (2016) for a review). As the agent accumulates experience, we can refine its uncertainty estimates about the environment, and adapt the policy to either the most likely MDP (Yu et al., 2017; 2018), a sample from the posterior over MDPs (Rakelly et al., 2019), or the full belief distribution (Brunskill, 2012; Guez et al., 2012; 2013; Lee et al., 2018; Zintgraf et al., 2020; Abraham et al., 2020; Mozian et al., 2020). Our work combines robust and Bayesian methods by deriving an uncertainty set from the belief and acting according to a risk-sensitive RL objective. Another different method at this intersection is RAMCP (Sharma et al., 2019), which robustly plans under misspecified prior beliefs in the Bayes-adaptive MDP. A key difference in our work is that we aim to *generalize* to new prior beliefs that describe novel environments. Furthermore, our experiments show that our framework can also handle misspecified priors.

While the BAMDP assumes the latent context is never observed, including at train time, we relax this assumption in our setting and access the context for each training environment to train a probabilistic system identification model via supervised learning. At test time, the model infers an uncertainty set over the true context from a partial trajectory. Unlike prior work in system identification for transfer (Yu et al., 2017; 2018; Kumar et al., 2021), we address the identifiability issues in systems where different contexts cannot be distinguished, and optimize a risk-sensitive objective to act robustly with respect to the non-identifiable parameters.

Multi-task and meta-RL. The multi-task RL setting aims to transfer knowledge between related tasks by learning the set of tasks together (Parisotto et al., 2016; Teh et al., 2017; Hausman et al., 2018; Yang et al., 2020; Yu et al., 2020; Sodhani et al., 2021). Meta-RL is a related setting whose goal is to rapidly adapt to new tasks (Finn et al., 2017; Rothfuss et al., 2019; Nagabandi et al., 2018; Song et al., 2020). Notably, context-based meta-RL algorithms extract information about new tasks from a few interactions (Duan et al., 2016; Wang et al., 2016; Perez et al., 2018; Rakelly et al., 2019; Zintgraf et al., 2019; Lee et al., 2019). Similarly, our method conditions the agent on the inferred uncertainty.

Bayesian meta-RL. Prior work in Bayesian meta-RL propose algorithms that train a policy conditioned on the posterior distribution (or belief) over the inferred context, allowing the agent to reason about task uncertainty (Humplik et al., 2019; Zintgraf et al., 2020; Zhang et al., 2021). However, our policy optimizes a risk-sensitive objective, rather than the expectation of the return over the belief. While Bayesian meta-RL agents balance exploration and exploitation in a new task based on uncertainty, our work focuses less on exploration in a new task. Instead, we design an agent that can robustly solve a new task under safety-critical conditions.

Robust meta-RL. Prior work has also studied robust meta-RL but under different setups and objectives, including robustness against adversarial reward functions under a learned model (Lin et al., 2020) and robustness by learning diverse behaviors within a single MDP (Kumar et al., 2020; Zahavy et al., 2020). The objective of our work is to robustly adapt to new environments by training in a set of related MDPs. Most closely related is CARL (Zhang et al., 2020b), which prepares the agent for safety-critical few-shot adaptation through pre-training on related source environments. CARL captures uncertainty through a probabilistic dynamics model, fine-tunes the model with new data collected in adaptation episodes with the target environment, and generates risk-sensitive plans with respect to the fine-tuned model. In contrast to CARL, which relies on multiple rounds of trial-and-error for adaptation, the robust RL setting typically evaluates *zero-shot* performance on new environments. Without the opportunity to test different behaviors for adaptation, new challenges with system identifiability arise.

3. Problem Setup

We first introduce notation in the standard RL setting in Sec. 3.1 and the robust contextual MDP in Sec. 3.2. Then, we formalize the multi-set robustness objective in Sec. 3.3.

3.1. Preliminaries

A Markov decision process (MDP) or task is a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, p, r, \rho, \gamma \rangle$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, p is the state transition probability, r is the reward function, ρ is the initial state distribution, and $\gamma \in [0, 1)$ is the discount factor. The goal in standard RL is to learn a policy π that maximizes the expected sum of rewards $\mathcal{J}(\pi) := \mathbb{E}_{\pi, p}[G_\pi]$ where $G_\pi = \sum_{t=0}^\infty \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)$.

We also consider the contextual Markov decision process (CMDP) (Hallak et al., 2015) which, like the standard MDP, is equipped with a state space $\mathcal{S}$ and action space $\mathcal{A}$. It additionally has a context space $\mathcal{C}$ and function $\mathcal{M}$ that maps any context $c \in \mathcal{C}$ to an MDP $\mathcal{M}(c) = \langle \mathcal{S}, \mathcal{A}, p^c, r^c, \rho, \gamma \rangle$, where p^c and r^c are parameterized by c.[1]

---
[1]We refer to contexts and tasks interchangeably.

3.2. Robust Contextual Markov Decision Process

The exact context c describing the task is often unknown, especially when the agent is transferred to an entirely new task. Instead, there may be a prior belief $b(c)$ over the context c, from which we can derive an uncertainty set.

Formally, we define the **robust contextual Markov decision process (R-CMDP)**, which extends the CMDP with an initial uncertainty set $\Xi \subseteq \mathcal{C}$ over the contexts. We also assume a distribution $p(\Xi)$ from which we can draw samples. This uncertainty set is given at the beginning of an episode and can be interpreted as a prior over the true context. We focus on parameterized uncertainty sets, and Ξ refers to the parameters that define the set, e.g., the center and radius for ball sets. Then, one way to acquire a robust policy is to optimize the worst-case objective with respect to the uncertainty set $\mathcal{J}_{\min}(\pi) := \min_{c \in \Xi} \mathbb{E}_{\pi, p^c}[G_\pi^c]$ where $G_\pi^c = \sum_{t=0}^\infty \gamma^t r^c(\mathbf{s}_t, \mathbf{a}_t)$. In this work, we optimize a softer version of this worst-case objective: the conditional value-at-risk (CVaR) (Tamar et al., 2015; Rajeswaran et al., 2016; Tang et al., 2019), because we can reliably form a sampling-based approximation of the CVaR for optimization.

The CVaR objective is defined over the random variable G_π^Ξ of returns induced by the uniform distribution over contexts in the uncertainty set Ξ. First, the value-at-risk is given by the α-quantile of the return distribution

$$\text{VaR}_\alpha(G_\pi^\Xi) := \max\{y | P(G_\pi^\Xi \leq y) \leq \alpha\}.$$

Then, denoting $\mathcal{P}(c)$ as the uniform distribution over the set $\{c \in \Xi | G_\pi^c \leq \text{VaR}_\alpha(G_\pi^\Xi)\}$, the CVaR objective is

$$\mathcal{J}_\pi^{\text{CVaR}_\alpha}(\Xi) := \mathbb{E}_{\pi, c \sim \mathcal{P}(c)}[G_\pi^c],$$

the expected return over the lower α-percentile subset of the uncertainty set. When $\alpha = 1$, the objective is the average over the perturbation set, and when $\alpha \to 0$, the objective becomes the max-min objective.

3.3. Multi-Set Robustness

Optimizing a robust policy with respect to a new uncertainty set can be costly for each new policy. Hence, we aim to learn a single policy that can be robust to several different uncertainty sets. We do so by leveraging the multi-task RL setting to optimize a policy that can generalize to and provide good worst-case performance with respect to *new* uncertainty sets.

In particular, the learner has access to M training tasks $\{\mathcal{M}_i\}_{i=1}^M$ that are parameterized by M different observed contexts $\{c_i\}_{i=1}^M$. The goal of our setting is to learn a set-conditioned policy $\pi(\mathbf{a}|\mathbf{s}, \Xi)$ that maximizes the worst-case expected return with respect to all uncertainty sets from the distribution $p(\Xi)$. That is, we want to optimize $\mathbb{E}_{\Xi \sim p(\Xi)}\left[\mathcal{J}_\pi^{\text{CVaR}_\alpha}(\Xi)\right]$.

4. System Identification and its Challenges

Rather than behaving invariantly to different contexts as in robust RL, another approach is to condition the policy on the context or a distribution over the context. System identification methods based on this idea train a predictive model to produce either a point estimate of or a posterior distribution over the context, given a history $h_t = (\mathbf{s}_0, \mathbf{a}_0, r_0, \ldots, \mathbf{s}_{t-1}, \mathbf{a}_{t-1}, r_{t-1})$. This approach has demonstrated strong generalization performance, including transfer from simulation to the real world (Yu et al., 2017; 2018; Kumar et al., 2021).

However, as discussed by Dorfman & Tamar, many systems are often determined by parameters that are not easily identifiable from a limited amount of interaction. Recall our peg-insertion example from Fig. 1: the size of the peg cannot be determined within a single trial, but critically, the robot has to take this parameter into account to select a box to insert the peg into. In these low-data regimes, the system identification model can fail to accurately distinguish between multiple MDP contexts. Formally, the context c is non-identifiable from a dataset h if there is a set of other contexts $\mathcal{C}' \subseteq \mathcal{C}$ that can also explain the data.

Definition 4.1 (Context non-identifiability). Let $P^{c,\pi}(h_{:t})$ denote the probability distribution over histories at time-step t under the MDP $\mathcal{M}(c)$ and policy π. Then, the context c is non-identifiable from the dataset $h_{:t}$ collected by policy π if there exists a subset $\mathcal{C}' \neq \{c\}$ such that $P^{c,\pi}(h_{:t}) = P^{c',\pi}(h_{:t})$ for all $c \in \mathcal{C}'$.

As an aside, context non-identifiability can also be viewed as posterior collapse (Wang & Cunningham, 2020). In our setting, posterior collapse occurs when the prior and posterior belief distributions are equal, i.e., $b'(c|b, h) = b(c)$. Hence, one proxy measure of context identifiability is the entropy of the belief distribution, i.e., higher entropy indicates lower identifiability of the context. This problem is further exacerbated when knowledge of the context is critical to the task at hand, i.e., confusing it with a different context can lead to a large drop in performance.

Definition 4.2 (Critical contexts). Denote the optimal context-dependent policy by $\pi^*(c) \coloneqq \arg\max_\pi \mathbb{E}_{\pi,c}[G^c_\pi]$. Consider the set of contexts $\mathcal{C}'$ for which $P^{c,\pi^*(c)}(h) = P^{c',\pi^*(c)}(h)$ holds for all $c' \in \mathcal{C}'$. The worst-case gap for a context-dependent policy evaluated in the MDP with context c is $D(c) = \max_{c' \in \mathcal{C}'} G^c_{\pi^*(c)} - G^c_{\pi^*(c')}$. The context c is said to be critical if its worst-case gap is large, i.e., $D(c) > \epsilon$ for threshold ϵ.

It becomes clear when there is uncertainty around a critical context c, the gap can be significant. Hence, to be robust to the worst case of the non-identifiable set, our objective is to minimize the worst-case gap: $\min_\pi \max_{c \in \mathcal{C}'} G^c_{\pi^*(c)} - G^c_\pi$, or equivalently, $\max_\pi \min_{c \in \mathcal{C}'} G^c_\pi$. In the next section, we introduce our algorithm which re-estimates the uncertainty

set while taking actions that are robust at each time-step. In particular, we optimize

$$\max_{\pi_0} \min_{c'_0 \in \Xi_0} \left(\mathbb{E}[r^c(\mathbf{s}_0, \mathbf{a}_0)] + \max_{\pi_1} \min_{c'_1 \in \Xi_1} \left(\gamma \mathbb{E}[r^c(\mathbf{s}_1, \mathbf{a}_1)] + \cdots \right) \right)$$

with $\pi_t = \pi(\cdot|\mathbf{s}_t, \Xi_t)$, which lower bounds the desired objective $\max_\pi \min_{c \in \mathcal{C}'} G^c_\pi$.

5. Risk-Sensitive Adaptation via System Identification and Multi-Set Robustness

To address the challenges associated with non-identifiable systems, we propose a simple framework that consists of a probabilistic system identification model and a family of risk-sensitive policies $\pi(\mathbf{a}|\mathbf{s}, \Xi)$ conditioned on the uncertainty set inferred by the model. The resulting algorithm combines the benefits of system identification and risk-sensitive RL as it reduces the model uncertainty where possible while behaving cautiously with respect to the irreducible uncertainty. Our overall approach, which we call System Identification and Risk-Sensitive Adaptation (SIRSA), is illustrated in Fig. 3, with each component detailed below.

5.1. Probabilistic System Identification

To capture the epistemic uncertainty of an unknown environment at test time, we train an ensemble of models to predict the context c that parameterizes the environment's dynamics and reward function. Recall that at train time, the agent observes the context of each training task $\{c_i\}_{i=1}^M$, and for each task, collects a dataset of transitions $\{(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}'_t)\}_t$.

We learn an ensemble of B different models, where each model f_i maps an initial uncertainty set Ξ and a history h of H transitions to a context. In this work, the uncertainty set is an ℓ_1-ball with its own center μ and width σ. Then, each model f_i has parameters ψ_i that are trained with the mean squared error on the predicted context:

$$\mathcal{L}_{\psi_{1:B}} = \mathbb{E}_{(\mu,\sigma,h,c)\sim\mathcal{D}, j\sim\text{Unif}(B)} \left[\left(f_{\psi_j}(\mu, \sigma, h) - c \right)^2 \right],$$
(1)

where the initial uncertainty set $\Xi = (\mu, \sigma)$, given by the environment to the learner, offers an initial guess of the true context. We define the parameters of the posterior uncertainty set $\Xi' = (\mu', \sigma')$ as the mean and standard deviation of the ensemble:

$$\mu' = \mu \left(\left\{ f_{\psi_j}(\Xi, h) \right\}_{j=1}^B \right), \ \sigma' = \sigma \left(\left\{ f_{\psi_j}(\Xi, h) \right\}_{j=1}^B \right),$$
(2)

where $\mu(\cdot)$ and $\sigma(\cdot)$ compute the mean and standard deviation, respectively. At inference time, we recursively update the uncertainty set by using the set inferred from the previous time-step as the prior. That is, the perturbation set Ξ_t at time-step t has parameters $\mu_t = \mu(\{f_{\psi_j}(\mu_{t-1}, \sigma_{t-1}, h)\}_{j:1\ldots B})$ and $\sigma_t = $
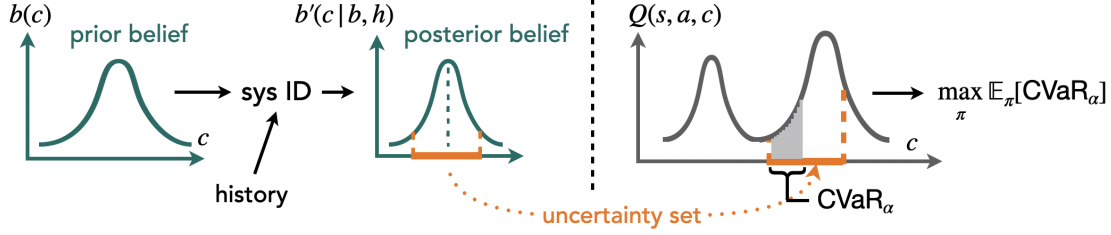
Figure 3. Our framework combines system identification with risk-sensitive RL to robustly adapt to new environments. First, the algorithm updates the uncertainty set over the context with the agent's recent history h. We then optimize a risk-sensitive policy over the returns within this uncertainty set.

$\sigma(\{f_{\psi_j}(\mu_{t-1}, \sigma_{t-1}, h)\}_{j:1\ldots B})$. Next, we describe how we optimize a risk-sensitive policy to act robustly with respect to the inferred uncertainty set.

5.2. Risk-Sensitive Policy Optimization

The CVaR objective is computed by finding, from the set of environments defined by the uncertainty set Ξ, the α-quantile that the policy performs worst in. While the distribution of G_π^Ξ is unknown, we can approximate its CVaR through a context-conditioned critic $Q_\theta(\mathbf{s}, \mathbf{a}, c)$. That is,

$$\mathbb{E}_{c \sim \mathcal{P}(c)}\left[G_\pi^c\right] \approx \mathbb{E}_{c \sim \mathcal{P}(c)}\left[\mathbb{E}_{\mathbf{s} \sim \mathcal{D}, \mathbf{a} \sim \pi(\cdot|\mathbf{s})}\left[Q_\theta(\mathbf{s}, \mathbf{a}, c)\right]\right],$$

where $\mathcal{P}(c)$ is the uniform distribution over the set $\{c \in \Xi | G_\pi^c \leq \mathrm{VaR}_\alpha(G_\pi^\Xi)\}$.

We can form a Monte-Carlo estimate of the CVaR as follows. Let $\tilde{c}_1, \ldots, \tilde{c}_N$ be N samples drawn i.i.d. from the uncertainty set Ξ, and $Q_1, \ldots, Q_N$ be their corresponding Q-values, i.e., $Q_i = Q_\theta(\mathbf{s}, \mathbf{a}, \tilde{c}_i)$. After sorting the contexts in ascending order based on their Q-values, $\tilde{c}_{[1]}, \ldots, \tilde{c}_{[N]}$, the empirical α-quantile is simply $Q_\theta(\mathbf{s}, \mathbf{a}, \tilde{c}_{[\lfloor \alpha N \rfloor]})$, and, the empirical CVaR approximation is

$$\frac{1}{\lfloor \alpha N \rfloor} \sum_{i=1}^{\lfloor \alpha N \rfloor} Q_\theta\left(\mathbf{s}, \mathbf{a}, \tilde{c}_{[i]}\right).$$

To update the policy π_ϕ, we can compute $\nabla_\phi J_\pi^{\mathrm{CVaR}_\alpha}$, the gradient of the approximated CVaR with respect to the policy parameters ϕ, with

$$\mathbb{E}_{\mathbf{s} \sim \mathcal{D}}\left[\sum_{i=1}^{\lfloor \alpha N \rfloor} \frac{\nabla_{\mathbf{a}} Q_\theta(\mathbf{s}, \mathbf{a}, \tilde{c}_{[i]})|_{\mathbf{a} \sim \pi_\phi}}{\lfloor \alpha N \rfloor} \nabla_\phi \pi_\phi(\mathbf{a}|\mathbf{s}, \Xi)\right]. \quad (3)$$

We construct our CVaR actor on top of Soft Actor-Critic (SAC) (Haarnoja et al., 2018). Our algorithm, which we call **S**ystem **I**dentification and **R**isk-**S**ensitive **A**daptation (SIRSA), is summarized in Alg. 1. We begin by training the actor and critic with the losses $\mathcal{J}_\pi$ and $\mathcal{J}_Q$ defined in SAC, and the ensemble of models with the loss defined in Eqn. 1. After $T_{\mathrm{threshold}}$ iterations, we update the actor π_ϕ based on the CVaR objective defined in Eqn. 3 instead of $\mathcal{J}_\pi$. Full implementation details can be found in Appendix A.

Algorithm 1 System Identification and Risk-Sensitive Adaptation (SIRSA)

Input: CVaR level α, threshold $T_{\mathrm{threshold}}$
Initialize replay buffers for each training task $\mathcal{D}[c]$
for $i = 1, 2, \ldots$ **do**
  Sample training environment from set of environments
  Initialize history $h_0 = \{\mathbf{s}_0\}$
  for each environment step **do**
    Take action $\mathbf{a}_t \sim \pi_\phi(\cdot|\mathbf{s}, \Xi_t)$
    Update history $h_t \leftarrow h_{t-1} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}_t')\}$
    Update uncertainty set Ξ_{t+1} according to Eqn. 2
  end for
  Update replay buffer $\mathcal{D}[c] \leftarrow \mathcal{D}[c] \cup h_T$
  for each gradient step **do**
    Sample batch from replay buffers $\bigcup_c \mathcal{D}[c]$
    Update critic parameters θ with $\nabla_\theta \mathcal{J}_Q$
    if $i < T_{\mathrm{threshold}}$ **then**
      Update actor parameters ϕ with $\nabla_\phi J_\pi$
    else
      Update actor parameters ϕ with $\nabla_\phi J_\pi^{\mathrm{CVaR}_\alpha}$
    end if
    Update ensemble parameters ψ_k with $\nabla_{\psi_k} \mathcal{L}_{\psi_k}$
  end for
end for

---

6. Experiments

We design several experiments to understand the effectiveness of our proposed approach compared to system identification and robust RL approaches in unseen environments. Specifically, we seek to answer the following questions:[2]

1. How does our method SIRSA compare to standard system identification and robust RL in terms of worst-case performance on new uncertainty sets?
2. Can SIRSA generalize to new test-time scenarios such as misspecified priors and non-stationary dynamics?
3. How does SIRSA respond to varying α-levels of risk sensitivity?

---

[2]Code and videos of our results are on our webpage: https://sites.google.com/view/sirsa-public/home.

6.1. Experimental Setup

Baselines. First, we consider multi-task RL baselines that train a context-conditioned policy $\pi(\mathbf{a}|\mathbf{s}, c)$.

- **Context-conditioned policy ensemble**. At *test* time, N^{ens} contexts $\{c^i\}_{i=1}^{N^{\text{ens}}}$ are sampled from the initial uncertainty set to create an ensemble of policies $\pi^{\text{ens}}(\cdot|\mathbf{s}) = \sum_{i=1}^{N^{\text{ens}}} \pi(\mathbf{a}|\mathbf{s}, c^i)/N^{\text{ens}}$. We use $N^{\text{ens}} = 5$.

- **Context-conditioned policy with true context (oracle)**. An oracle with access to the ground-truth context at test time, given as input to the context-conditioned policy.

We also compare to the system ID ablation of SIRSA, which optimizes the expected return rather than the CVaR:

- **Set-conditioned policy with system identification** (Yu et al., 2017). Along with a set-conditioned policy $\pi(\mathbf{a}|\mathbf{s}, \Xi)$, this baseline trains a system identification model that maps the history of last H states and actions to a belief over the context. The belief inferred by the model is given to the policy.

Finally, we compare to existing robust/risk-sensitive RL methods. Like our approach, each of these methods controls the risk level through the CVaR hyperparameter α. We run each algorithm with $\alpha \in \{0.25, 0.5, 0.75, 1.0\}$, and report the results for the most performant policy in this section. In Appendix C.2, we report the full results for each value of α.

- **EPOpt** (Rajeswaran et al., 2016). A domain randomization method that optimizes the CVaR objective by training on the α-worst percentile of all training environments.

- **Multi-Set EPOpt**. We design a stronger variant of EPOpt by training a multi-set robust policy $\pi(\mathbf{a}|\mathbf{s}, \Xi)$ on the α-worst percentile of environments in each set $\Xi \subseteq \mathcal{C}$.

- **Worst Cases Policy Gradients (WCPG)** (Tang et al., 2019). This comparison trains a family of α conditional policies $\pi(\mathbf{a}|\mathbf{s}, \alpha)$ with varying levels of risk sensitivity. In order to approximate the CVaR across different α-levels, the future return generated by policy π is modeled as a Gaussian distribution and approximated by a distributional critic, allowing the CVaR to be computed in closed form. During training, we sample α uniformly from $[0, 1]$. At inference time, we evaluate the policy at the α-levels $\{0.25, 0.5, 0.75, 1.0\}$. Like EPOpt, this comparison trains on the entire range of contexts as its uncertainty set.

- **Multi-Set WCPG** (Tang et al., 2019). We design a stronger variant of WCPG by training a multi-set robust policy $\pi(\mathbf{a}|\mathbf{s}, \alpha, \Xi)$ with WCPG.

Environments. We design several environments to evaluate our approach, and in each, vary one or more parameters that affect the dynamics and/or reward function. All methods can access the true context of each *training* environment. However, at *inference* time, only an initial uncertainty set

Environment	Uncertain Params.	Range
Point mass	Obstacle size	$[0.025, 0.075]$
	Velocity	$[0.06, 0.1]$
Minitaur	Torso mass	$[-0.2, 0.2]$
	Leg mass	$[-0.2, 0.2]$
	Leg failure (x4)	$[0.0, 1.0]$
Half-cheetah	Torso mass	$[-0.5, 0.5]$
	Joint friction	$[0.1, 0.9]$
	Joint failure (x6)	$[0.0, 1.0]$
Peg insertion	Step size	$[0.5, 1.5]$
	Peg size	$[0.0125, 0.0225]$

Table 1. Range of parameter values $\mathcal{C}$ in our environments. Let L denote the lower limit and H denote the upper limit of the range. Then, each uncertainty set has a sampled center $\mu \sim$ `Unif`$(L + 0.1 \cdot (H - L), H - 0.1 \cdot (H - L))$ and a sampled width $\sigma \sim$ `Unif`$(0.1 \cdot (H - L), \min(\mu - L, H - \mu))$.

is provided, and none of the methods (with the exception of the context-conditioned oracle) have access to the true parameter values. In Table 1, we tabulate the ranges for the different parameters, and describe the environments below:

- **Point mass navigation**. A point mass has to navigate around a roundabout with uncertainty in the size of the roundabout and the precise velocity. We additionally design two variants: **Point mass (obstacle)** where only the obstacle size is uncertain and **Point mass (velocity)** where only the velocity of the agent is uncertain.

- **Minitaur** (Tan et al., 2018). A simulated 8-DoF minitaur robot with uncertainty in the mass and leg failure rate.

- **Half-cheetah** (Brockman et al., 2016; Vinitsky et al., 2020). Modified OpenAI Gym environment with uncertainty in the mass, joint friction, and joint failure rate.

- **Peg insertion** (Zhao et al., 2020; Schoettler et al., 2020). A simulated 7-DoF Sawyer robot arm is to insert a peg into one of the boxes (see Fig. 1). The uncertainty is in the position controller's step size and the size of the peg.

Full descriptions of each environment are in Appendix B.

Evaluation metrics. To evaluate each method, we construct test-time uncertainty sets centered around new contexts not seen during training. We then evaluate a policy's performance by, first, uniformly sampling K context vectors from each set, then, rolling out the policy in each of the K sampled environments. We are in particular interested in the **worst-case performance**, approximated by the minimum return of the K rollouts, and additionally report the **average-case performance**, approximated by the mean return of the K rollouts. In all experiments, we use $K = 50$.

6.2. Robustness to New Uncertainty Sets

Point mass. We first seek to better understand the strengths and weaknesses of prior methods in the Point mass (obsta-

Uncertain Param.	ID Error	Method	
		System ID	Set-EPOpt
Obstacle size	0.071 ± 0.002	37.7 ± 0.4	$\mathbf{39.4 \pm 0.5}$
Velocity	0.035 ± 0.000	$\mathbf{37.9 \pm 0.0}$	37.3 ± 0.1

Table 2. Worst-case performance of system ID and Set-EPOpt when (1) the obstacle size and when (2) the velocity is uncertain.

cle) and Point mass (velocity) environments. The former represents a parameter that is difficult to precisely identify since it would require making contact with the obstacle to infer its size. In contrast, the latter parameter can be exactly estimated given a single time-step. We compare System ID to Multi-Set EPOpt, which acts as the representative of robust RL methods. In Table 2, we compare the worst-case performance of the two methods across 20 test uncertainty sets, and find that firstly the identification error of the obstacle size is indeed higher than that of the velocity parameter, confirming our intuition. In the Point mass (obstacle) domain, Set-EPOpt outperforms System ID as the uncertain parameter cannot be exactly identified without incurring a penalty. In the Point mass (velocity) domain, we see the reverse result: the System ID method correctly adapts to the predicted context, whereas Set-EPOpt acts conservatively without precise identification of the parameter. The trajectories taken by these policies are visualized in Appendix C.1.

High-dimensional domains. In Table 3, we present the results in the remaining domains. In terms of the worst-case performance (see the "Sample Min" column), many of the studied baselines perform competitively against each other. System ID, which optimizes for the expectation of the return over its inferred belief, attains strong *average-case* returns as a result (see the "Sample Mean" column). However, there is no single baseline that outperforms the rest in all settings in terms of worst-case returns, the primary metric we are interested in. On the other hand, SIRSA, which inherits from both system identification and robust RL algorithms, consistently achieves high worst-case returns across the different environments. Interestingly, all methods demonstrate similarly strong performance in the Minitaur domain, which suggests that context awareness is not as critical in this domain.

In general, the multi-set robust RL baselines demonstrate better worst-case as well as average-case performance than their single-set counterparts. Without access to a more informative prior, the latter group of policies is trained to act robustly to the maximal uncertainty set: the union over all of the tasks seen during training. As a result, their behavior can be overly conservative.

Maximum initial uncertainty. We now evaluate the policies learned by each method on the maximum uncertainty set in the Half-Cheetah domain, defined as the range of pa-

Task	Method	Sample Min	Sample Mean
Point mass	Ensemble	36.8 ± 0.3	$\mathbf{41.7 \pm 0.2}$
	System ID	$\mathbf{37.6 \pm 0.3}$	41.5 ± 0.2
	EPOpt	36.3 ± 0.6	40.5 ± 0.4
	Set-EPOpt	37.1 ± 0.5	40.7 ± 0.4
	WCPG	34.8 ± 0.6	39.3 ± 0.5
	Set-WCPG	34.7 ± 0.7	39.0 ± 0.5
	SIRSA (Ours)	$\mathbf{37.9 \pm 0.2}$	$\mathbf{41.7 \pm 0.1}$
	Oracle	38.6 ± 0.3	42.6 ± 0.1
Minitaur	Ensemble	178.0 ± 7.1	212.5 ± 3.6
	System ID	174.0 ± 10.8	211.8 ± 2.9
	EPOpt	172.2 ± 7.3	199.9 ± 5.5
	Set-EPOpt	183.1 ± 7.5	216.7 ± 3.7
	WCPG	165.5 ± 17.7	193.7 ± 19.6
	Set-WCPG	174.5 ± 10.0	206.9 ± 4.7
	SIRSA (Ours)	$\mathbf{187.8 \pm 7.6}$	$\mathbf{214.3 \pm 2.5}$
	Oracle	172.2 ± 4.1	208.0 ± 3.0
Half-cheetah	Ensemble	$\mathbf{3988 \pm 75}$	4714 ± 38
	System ID	3774 ± 318	4393 ± 260
	EPOpt	2272 ± 218	2717 ± 324
	Set-EPOpt	3806 ± 224	4477 ± 231
	WCPG	3747 ± 229	4305 ± 314
	Set-WCPG	3871 ± 207	4433 ± 253
	SIRSA (Ours)	$\mathbf{4146 \pm 112}$	$\mathbf{4872 \pm 74}$
	Oracle	4246 ± 59	4851 ± 38
Peg insertion	Ensemble	45.2 ± 3.0	92.9 ± 1.8
	System ID	73.7 ± 4.3	101.3 ± 4.0
	EPOpt	43.2 ± 5.4	75.1 ± 8.2
	Set-EPOpt	70.6 ± 3.6	96.9 ± 2.7
	WCPG	33.8 ± 7.2	63.2 ± 6.9
	Set-WCPG	68.3 ± 4.6	92.0 ± 4.4
	SIRSA (Ours)	$\mathbf{83.4 \pm 4.5}$	$\mathbf{109.5 \pm 2.6}$
	Oracle	78.2 ± 3.6	122.0 ± 1.5

Table 3. We evaluate each policy on 20 uncertainty sets at test time by drawing samples from the set and evaluating the policy's performance on each sampled environment. We report the minimum and mean performance over the samples as an approximation to the average-case and worst-case performance on the perturbation set. The means and standard errors are computed over 10 policies for each method trained with random seeds. We bold the highest results that are more than a standard error higher than others.

rameters $\mathcal{C}$ in Table 1. This also represents the evaluation setup of standard single-set robust RL. We report the results in Table 5. Notably, the single-set and multi-set robust RL methods perform comparably, where Set-EPOpt even outperforms EPOpt, indicating that multi-set robust policies can generalize even to the maximum uncertainty set $\mathcal{C}$.

Initial uncertainty ablation. To further understand the improvements from the initial uncertainty set, we evaluate an ablation of our approach that does not access the initial uncertainty set during training or at test time. In Table 4, we report the sample minimum and mean, approximating the worst- and average-case performance. Without the initial

Task	Prior?	Sample Min	Sample Mean
Point mass	No	34.8 ± 0.5	40.3 ± 0.4
	Yes	37.9 ± 0.2	41.8 ± 0.1
Minitaur	No	147.0 ± 8.7	195.9 ± 4.8
	Yes	187.8 ± 7.6	214.3 ± 2.5
Half-cheetah	No	3614 ± 186	4508 ± 211
	Yes	4146 ± 112	4872 ± 74
Peg insertion	No	40.1 ± 5.9	71.5 ± 8.0
	Yes	83.4 ± 4.5	109.5 ± 2.6

Table 4. Worst-case and average-case performance with and without the initial uncertainty set. Means and standard errors are computed over 10 policies trained with different random seeds.

Method	Max Uncertainty Set Return	Non-stationary Env Return
Ensemble	3619 ± 95	4698 ± 31
System ID	3608 ± 468	4277 ± 341
EPOpt	1516 ± 253	—
Set-EPOpt	3360 ± 498	4336 ± 247
WCPG	3598 ± 339	—
Set-WCPG	3055 ± 456	3950 ± 486
SIRSA (Ours)	$\mathbf{4281 \pm 73}$	$\mathbf{4913 \pm 65}$
Oracle	4203 ± 91	4580 ± 63

Table 5. **Middle:** Mean and standard error of the return over 10 random seeds on the maximum uncertainty set in the Half-Cheetah domain. **Right:** Mean and standard error of the return over 10 random seeds on a non-stationary Half-Cheetah environment, wherein the unobserved context changes every 50 timesteps.
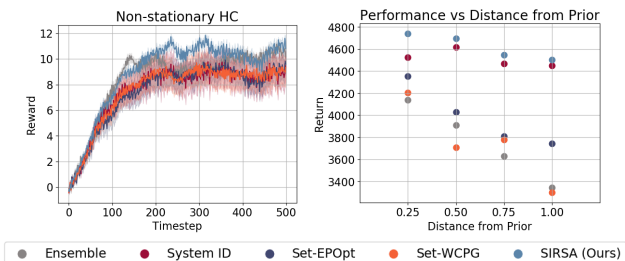
prior, SIRSA performs significantly worse in all four tasks, indicating the importance of providing this information.

6.3. Generalization under Misspecification

Non-stationarity. Most real-world environments are dynamic. For example, a robot's mass can vary over time as it carries different payloads. Does training for multi-set robustness facilitate generalization to non-stationary environments at test time? Intuitively, if the non-stationary parameters remain contained within a finite range, we can capture the changing environment with a single uncertainty set.

In this experiment, we design a non-stationary variant of the Half-Cheetah environment. Specifically, we sample an uncertainty set at the beginning of the episode, and at every 50 timesteps, we set the parameters of the environment to a new context sampled from the initial uncertainty set. The episode is terminated after 500 timesteps. We report the results aggregated over 10 different rollouts, each corresponding to a different initial uncertainty set, in Table 5. The best-performing policy is that learned by SIRSA, indicating that adaptive risk sensitivity to a given uncertainty set is a reasonable solution to overcome non-stationarity. In Fig. 4 (left), we plot the reward attained by the agent over the 500 timesteps of one of the rollouts. Here, the policy learned with SIRSA attains higher rewards at each timestep.

Initial uncertainty set. So far, we provided an initial uncertainty set $\Xi_0 = (\mu_0, \sigma_0)$ that correctly informs the agent about the environment. How robust is the agent when this prior is misspecified, i.e., when the initial perturbation set does not contain the true environment? In this experiment, we provide intentionally misspecified sets to the agent at test time. For each set Ξ_0, we sample contexts of the form $\mu_0 + w\sigma_0$, where $w \in \{-(1+r), 1+r\}^d$, d is the number of context variables, and r varies between $0.25, 0.50, 0.75$, and 1.00. We evaluate on the corresponding environments, and plot the *average* over these samples as a function of r in Fig. 4 (right). The multi-set robust RL methods, Set-EPOpt and Set-WCPG, as well as the Ensemble baseline tend to drop in performance faster as the environment deviates more

Figure 4. **Left:** Reward versus time in the Half-Cheetah environment with non-stationary parameters. The solid lines represent mean and the shaded regions represent the standard error over 10 policies trained with different random seeds. **Right:** Performance on increasingly misspecified priors. Each data point is averaged over 10 policies trained with different random seeds.

from the prior uncertainty set. In contrast, SIRSA and System ID are capable of identifying contexts outside of the initial uncertainty set and degrade more gracefully.

6.4. Sensitivity Analysis

Our algorithm introduces several hyperparameters that determine the computation cost and the robustness of the policy. We study the CVaR hyperparameter α below, the effect of the number of CVaR samples N, and the effect of the system identification ensemble size B.

CVaR level α. Lower levels of α in principle lead to a more robust policy. However, too low levels of α can harm performance as the actor becomes too conservative. In terms of the computational cost, however, the algorithm computes $\lfloor \alpha N \rfloor$ gradients to approximate the
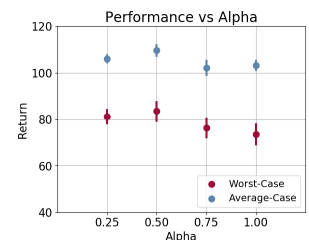

Figure 5. SIRSA with different values of α. Means and standard errors are over 10 seeds.
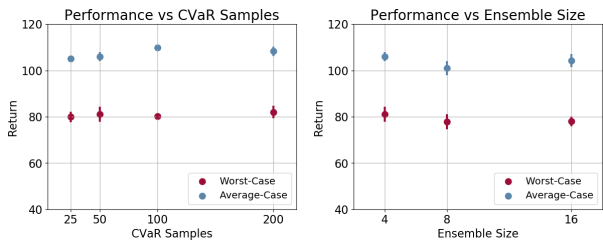
Figure 6. **Left:** Performance of SIRSA with different amounts of CVaR samples. **Right:** Performance of SIRSA with different ensemble sizes. Both plots depict the means and standard errors over 10 random seeds.

CVaR gradient, is therefore more efficient with smaller α's. In Fig. 5, we plot the average- and worst-case performance of SIRSA with $\alpha \in \{0.25, 0.5, 0.75, 1.0\}$ in Peg Insertion, and find the best performing value of α is 0.5, which strikes a good balance in risk sensitivity and computation cost.

Number of CVaR samples N**.** Increasing the number of Monte-Carlo samples we use to approximate the CVaR objective can improve the estimate of the function. However, it is also more costly since it requires computing $\lfloor \alpha N \rfloor$ gradients. In Fig. 6 (left), we plot the average and worst-case performance of SIRSA for $N \in \{25, 50, 100, 200\}$ in the Peg Insertion domain. From these results, we conclude that there is no significant benefit to increasing the CVaR samples beyond 50.

Number of ensemble models B**.** Increasing the ensemble size can potentially lead to improved estimates of the posterior belief distribution, but requires training more model parameters. In Fig. 6 (right), we plot the average and worst-case performance for $B \in \{4, 8, 16\}$ in the Peg Insertion domain. Overall, the performance of SIRSA is agnostic to the ensemble size in this domain.

7. Discussion

Many robust RL solutions require a prior uncertainty set over the unobserved parameters of the *test-time* environment to learn a robust policy for this set at training time. To alleviate the need to build in this prior, we introduced and studied the multi-set robustness problem to facilitate generalization to new uncertainty sets. We further recognized the potential sub-optimality of memoryless robust RL methods in systems with parameters that can be identified from a history of interactions, and designed a framework that combines probabilistic system identification with the multi-set robust RL objective. Our method improves upon existing methods on a range of challenging control domains in terms of worst-case performance on new uncertainty sets.

While we believe the multi-set robustness problem represents a more general and useful framing of robustness to

variations, there is also a number of interesting future directions. For example, SIRSA currently assumes the contexts that underlie the training tasks are observed to train an ensemble of predictive models via supervised learning. To remove this assumption, one can leverage tools from unsupervised representation learning to learn a representation of the true context. Another question is whether there are robustness benefits when the agent explicitly seeks exploratory actions that minimize its uncertainty over the parameters. In our experiments, we designed two types of parameters: non-identifiable parameters whose uncertainty cannot be reduced at all and identifiable parameters whose uncertainty can be reduced within a single timestep. In settings where identifiable parameters require coordinated sequences of actions to reduce uncertainty over, the agent needs to be able to balance exploration, exploitation, and robustness.

Acknowledgements

References

Abraham, I., Handa, A., Ratliff, N., Lowrey, K., Murphey, T. D., and Fox, D. Model-based generalization under parameter uncertainty using path integral control. *IEEE Robotics and Automation Letters*, 5(2):2864–2871, 2020.

Badrinath, K. P. and Kalathil, D. Robust reinforcement learning using least squares policy iteration with provable performance guarantees. In *International Conference on Machine Learning*, pp. 511–520. PMLR, 2021.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Brunskill, E. Bayes-optimal reinforcement learning for discrete uncertainty domains. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pp. 1385–1386, 2012.

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

Derman, E., Mankowitz, D. J., Mann, T. A., and Mannor, S. Soft-robust actor-critic policy-gradient. *arXiv preprint arXiv:1803.04848*, 2018.

Dorfman, R. and Tamar, A. Offline meta reinforcement learning. *arXiv e-prints*, pp. arXiv–2008, 2020.

Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. RL2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Duff, M. O. *Optimal Learning: Computational procedures for Bayes-adaptive Markov decision processes*. University of Massachusetts Amherst, 2002.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. *International Conference on Machine Learning (ICML)*, 2017.

Ghavamzadeh, M., Mannor, S., Pineau, J., and Tamar, A. Bayesian reinforcement learning: A survey. *arXiv preprint arXiv:1609.04436*, 2016.

Guez, A., Silver, D., and Dayan, P. Efficient bayes-adaptive reinforcement learning using sample-based search. *arXiv preprint arXiv:1205.3109*, 2012.

Guez, A., Silver, D., and Dayan, P. Scalable and efficient bayes-adaptive reinforcement learning based on monte-carlo tree search. *Journal of Artificial Intelligence Research*, 48:841–883, 2013.

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

Hallak, A., Di Castro, D., and Mannor, S. Contextual markov decision processes. *arXiv preprint arXiv:1502.02259*, 2015.

Hausman, K., Springenberg, J. T., Wang, Z., Heess, N., and Riedmiller, M. Learning an embedding space for transferable robot skills. *International Conference on Learning Representations (ICLR)*, 2018.

Humplik, J., Galashov, A., Hasenclever, L., Ortega, P. A., Teh, Y. W., and Heess, N. Meta reinforcement learning as task inference. *arXiv preprint arXiv:1905.06424*, 2019.

Iyengar, G. N. Robust dynamic programming. *Mathematics of Operations Research*, 30(2):257–280, 2005.

Kumar, A., Fu, Z., Pathak, D., and Malik, J. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

Kumar, S., Kumar, A., Levine, S., and Finn, C. One solution is not all you need: Few-shot extrapolation via structured maxent rl. *Neural Information Processing Systems (NeurIPS)*, 33, 2020.

Lee, A. X., Nagabandi, A., Abbeel, P., and Levine, S. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *arXiv preprint arXiv:1907.00953*, 2019.

Lee, G., Hou, B., Mandalika, A., Lee, J., Choudhury, S., and Srinivasa, S. S. Bayesian policy optimization for model uncertainty. *arXiv preprint arXiv:1810.01014*, 2018.

Lim, S. H., Xu, H., and Mannor, S. Reinforcement learning in robust markov decision processes. *Neural Information Processing Systems (NeurIPS)*, 26:701–709, 2013.

Lin, Z., Thomas, G., Yang, G., and Ma, T. Model-based adversarial meta-reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Mankowitz, D. J., Levine, N., Jeong, R., Abdolmaleki, A., Springenberg, J. T., Shi, Y., Kay, J., Hester, T., Mann, T., and Riedmiller, M. Robust reinforcement learning for continuous control with model misspecification. In *International Conference on Learning Representations (ICLR)*, 2020.

Mehta, B., Diaz, M., Golemo, F., Pal, C. J., and Paull, L. Active domain randomization. In *Conference on Robot Learning (CoRL)*, pp. 1162–1176. PMLR, 2020.

Mordatch, I., Lowrey, K., and Todorov, E. Ensemble-cio: Full-body dynamic motion planning that transfers to physical humanoids. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5307–5314. IEEE, 2015.

Morimoto, J. and Doya, K. Robust reinforcement learning. pp. 1061–1067, 2000.

Mozian, M., Higuera, J. C. G., Meger, D., and Dudek, G. Learning domain randomization distributions for training robust locomotion policies. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6112–6117. IEEE, 2020.

Nagabandi, A., Clavera, I., Liu, S., Fearing, R. S., Abbeel, P., Levine, S., and Finn, C. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

Nilim, A. and El Ghaoui, L. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

Parisotto, E., Ba, L. J., and Salakhutdinov, R. Actor-mimic: Deep multitask and transfer reinforcement learning. *International Conference on Learning Representations (ICLR)*, 2016.

Perez, C. F., Such, F. P., and Karaletsos, T. Efficient transfer learning and online adaptation with latent variable models for continuous control. *arXiv preprint arXiv:1812.03399*, 2018.

Pinto, L., Davidson, J., Sukthankar, R., and Gupta, A. Robust adversarial reinforcement learning. In *International Conference on Machine Learning (ICML)*, pp. 2817–2826. PMLR, 2017.

Rajeswaran, A., Ghotra, S., Ravindran, B., and Levine, S. Epopt: Learning robust neural network policies using model ensembles. In *International Conference on Learning Representations (ICLR)*, 2016.

Rakelly, K., Zhou, A., Quillen, D., Finn, C., and Levine, S. Efficient off-policy meta-reinforcement learning via probabilistic context variables. *International Conference on Machine Learning (ICML)*, 2019.

Rockafellar, R. T., Uryasev, S., et al. Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42, 2000.

Ross, S., Chaib-draa, B., and Pineau, J. Bayes-adaptive pomdps. In *NIPS*, pp. 1225–1232, 2007.

Rothfuss, J., Lee, D., Clavera, I., Asfour, T., and Abbeel, P. Promp: Proximal meta-policy search. *International Conference on Learning Representations (ICLR)*, 2019.

Roy, A., Xu, H., and Pokutta, S. Reinforcement learning under model mismatch. *Advances in neural information processing systems*, 30, 2017.

Schoettler, G., Nair, A., Ojea, J. A., Levine, S., and Solowjow, E. Meta-reinforcement learning for robotic industrial insertion tasks. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9728–9735. IEEE, 2020.

Sharma, A., Harrison, J., Tsao, M., and Pavone, M. Robust and adaptive planning under model uncertainty. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29, pp. 410–418, 2019.

Sodhani, S., Zhang, A., and Pineau, J. Multi-task reinforcement learning with context-based representations. *arXiv preprint arXiv:2102.06177*, 2021.

Song, X., Yang, Y., Choromanski, K., Caluwaerts, K., Gao, W., Finn, C., and Tan, J. Rapidly adaptable legged robots via evolutionary meta-learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3769–3776. IEEE, 2020.

Tamar, A., Glassner, Y., and Mannor, S. Optimizing the cvar via sampling. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Tan, J., Zhang, T., Coumans, E., Iscen, A., Bai, Y., Hafner, D., Bohez, S., and Vanhoucke, V. Sim-to-real: Learning agile locomotion for quadruped robots. *arXiv preprint arXiv:1804.10332*, 2018.

Tang, Y. C., Zhang, J., and Salakhutdinov, R. Worst cases policy gradients. *arXiv preprint arXiv:1911.03618*, 2019.

Teh, Y. W., Bapst, V., Czarnecki, W. M., Quan, J., Kirkpatrick, J., Hadsell, R., Heess, N., and Pascanu, R. Distral: Robust multitask reinforcement learning. *Neural Information Processing Systems (NeurIPS)*, 2017.

Tessler, C., Efroni, Y., and Mannor, S. Action robust reinforcement learning and applications in continuous control. In *International Conference on Machine Learning (ICML)*, pp. 6215–6224. PMLR, 2019.

Vinitsky, E., Du, Y., Parvate, K., Jang, K., Abbeel, P., and Bayen, A. Robust reinforcement learning using adversarial populations. *arXiv preprint arXiv:2008.01825*, 2020.

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., Blundell, C., Kumaran, D., and Botvinick, M. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

Wang, Y. and Cunningham, J. P. Posterior collapse and latent variable non-identifiability. In *Third Symposium on Advances in Approximate Bayesian Inference*, 2020.

Yang, R., Xu, H., Wu, Y., and Wang, X. Multi-task reinforcement learning with soft modularization. *Neural Information Processing Systems (NeurIPS)*, 2020.

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., and Finn, C. Gradient surgery for multi-task learning. *Neural Information Processing Systems (NeurIPS)*, 2020.

Yu, W., Tan, J., Liu, C. K., and Turk, G. Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453*, 2017.

Yu, W., Liu, C. K., and Turk, G. Policy transfer with strategy optimization. *arXiv preprint arXiv:1810.05751*, 2018.

Zahavy, T., Barreto, A., Mankowitz, D. J., Hou, S., O'Donoghue, B., Kemaev, I., and Singh, S. Discovering a set of policies for the worst case reward. In *International Conference on Learning Representations (ICLR)*, 2020.

Zhang, H., Chen, H., Xiao, C., Li, B., Liu, M., Boning, D., and Hsieh, C.-J. Robust deep reinforcement learning against adversarial perturbations on state observations. *arXiv preprint arXiv:2003.08938*, 2020a.

Zhang, J., Cheung, B., Finn, C., Levine, S., and Jayaraman, D. Cautious adaptation for reinforcement learning in safety-critical settings. In *International Conference on Machine Learning*, pp. 11055–11065. PMLR, 2020b.

Zhang, J., Wang, J., Hu, H., Chen, T., Chen, Y., Fan, C., and Zhang, C. Metacure: Meta reinforcement learning with empowerment-driven exploration. In *International Conference on Machine Learning*, pp. 12600–12610. PMLR, 2021.

Zhao, T. Z., Nagabandi, A., Rakelly, K., Finn, C., and Levine, S. Meld: Meta-reinforcement learning from images via latent state models. *arXiv preprint arXiv:2010.13957*, 2020.

Zintgraf, L., Shiarlis, K., Igl, M., Schulze, S., Gal, Y., Hofmann, K., and Whiteson, S. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. *International Conference on Learning Representations (ICLR)*, 2020.

Zintgraf, L. M., Shiarlis, K., Kurin, V., Hofmann, K., and Whiteson, S. Fast context adaptation via meta-learning. *International Conference on Machine Learning (ICML)*, 2019.

Appendix

A. Implementation Details

Below, we provide details of the implementation of our algorithm SIRSA and the baselines.

A.1. SIRSA (OURS)

The System ID baseline shares the same implementation details and hyperparameters as SIRSA, except it does not implement the CVaR objective.

System identification model. We train an ensemble of $B = 4$ models, which are MLPs with 2 fully-connected layers of size 64 in the Point Mass domain; 2 fully-connected laters of size 256 in all other domains. Each model takes a $(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ tuple, outputs a prediction for the context, and is trained with the MSE of the predicted and true context.

Policy and critic networks. The policy and critic networks are MLPs with 2 fully-connected layers of size 64 in the Point Mass domain; 2 fully-connected layers of size 256 in all other domains.

In the Minitaur domain, training the critic was somewhat unstable. We therefore implement REDQ (Chen et al., 2021), which has been found to stabilize and accelerate learning. It trains an ensemble of M critic networks. To compute the Q-values, REDQ randomly subsamples 2 of the critic networks and take their minimum. In our Minitaur experiments, we use $M = 8$ **for our method as well as all comparisons**.

CVaR approximation. In our experiments, we use $N = 50$ CVaR samples to approximate the gradient of the CVaR.

Training phases. Before updating the policy with the CVaR objective, we first train the actor and critic networks with the SAC objectives:

$$\mathcal{J}_Q = \mathbb{E}_{(\mathbf{s},\mathbf{a}) \sim \mathcal{D}} \left[\frac{1}{2} \left(Q_\theta(\mathbf{s},\mathbf{a}) - \hat{Q}(\mathbf{s},\mathbf{a}) \right)^2 \right]$$

with

$$\hat{Q}(\mathbf{s},\mathbf{a}) = r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{s' \sim p} \left[V_\psi(s') \right]$$

where V_ψ is a target network, whose weights are an exponentially moving average of the value function weights. Then, after $T_{\text{threshold}}$ iterations, we optimize policy with the CVaR objective defined in Eqn. 3 instead.

In Point Mass, we optimize the SAC objectives for 25K iterations then optimize the CVaR for another 25K iterations, for a total of 50K training iterations. In the Minitaur and Peg Insertion domains, we pre-train for 150K iterations then optimize CVaR for 150K iterations for a total of 300K. In Half-Cheetah, the pre-training is 2.5M, and the the CVaR optimization is 0.5M long, for a total of 3M steps.

In the longer-horizon tasks, i.e., Minitaur and Half-Cheetah, we found that the error in the inferred uncertainty set can accumulate over longer roll-outs. In these cases, we found that resetting the prior back to the initially given uncertainty set produces better results.

A.2. EPOPT (RAJESWARAN ET AL., 2016)

Policy and critic networks. The policy and critic networks are MLPs with 2 fully-connected layers of size 64 in the Point Mass domain, and with 2 fully-connected layers of size 256 in all other domains. These networks are trained with the SAC objectives. To sample a batch of size D that they train on, we first sample D/α s-a-s' tuples from the replay buffer. Then, we sort the samples by the return of the trajectory they came from, and return the lowest D tuples.

A.3. WCPG (TANG ET AL., 2019)

Policy and critic networks. The policy and critic networks are MLPs with 2 fully-connected layers of size 64 in the Point Mass domain; 2 fully-connected layers of size 256 in all other domains. These networks are trained with SAC.

Q-variance network. WCPG requires an estimate of the variance of $Q(\mathbf{s},\mathbf{a},\Xi)$. We train an MLP with 2 fully-connected layers of size 256 via the MSE to predict the variance of $Q(\mathbf{s},\mathbf{a},\Xi)$. To generate the target variance this network regresses to, we generate a Monte-Carlo approximation: we sample 50 contexts $\{c_i\}_{i:1...50}$ from Ξ, evaluate them with our context-conditioned critic $Q_\theta(\mathbf{s},\mathbf{a},c_i)$, and compute their sample variance.

CVaR approximation. WCPG assumes that the Q-values follow a Gaussian distribution, and can therefore compute the CVaR of the uncertainty set $\Xi = (\mu, \sigma)$ in closed form as follows:

$$Q(\mathbf{s},\mathbf{a},\mu) - (\phi(\alpha)/\Phi(\alpha))\sqrt{\text{Q-Var}(\mathbf{s},\mathbf{a},\Xi)},$$

where $\text{Q-Var}(\mathbf{s},\mathbf{a},\Xi)$ represents the output of the Q-variance network, $\phi(\cdot)$ is the standard normal distribution, and $\Phi(\cdot)$ is its CDF:

$$\Psi(x) = \frac{1}{2}(1 + \text{erf}(x/\sqrt{2})).$$

B. Environment Details

In this section, we provide details of each of the four experimental domains. The domains are visualized in Fig. 7.

B.1. POINT MASS

In this environment, the agent is a point mass particle and must go around the roundabout to the goal on its other side. The x-velocity of the agent is fixed within a task but its precise value is unknown. The size of the roundabout r is also unknown. Each episode is 50 timesteps long, and the state consists of the xy-position of the agent and whether
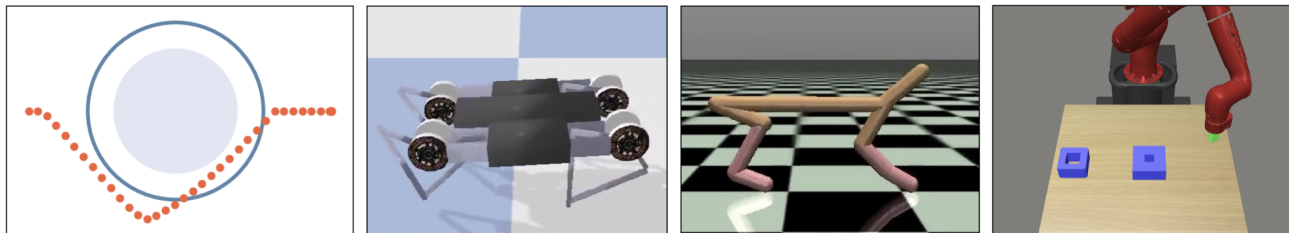
Figure 7. The domains in our evaluation. **Point Mass:** In this domain, there is uncertainty in the size of the obstacle (in lavender). The blue concentric circle represents the worst-case obstacle size within the uncertainty set. The agent must navigate around the obstacle. Its path is highlighted in pink-fuschia. **Minitaur:** The uncertainty lies in the mass of the robot and the failure rate of one of its legs. **Half-Cheetah:** The uncertainty lies in the mass of the agent, the friction of its joints, and failure rate of one of its joints. **Peg Insertion:** This domain has uncertainty over the size of the peg and the step size of the robot's actions.

the agent is on top of the roundabout. There is one action input which controls the change in y-position. The reward function is defined as

$$r_t = 1 - \mathbb{1}(x_t^2 + y_t^2 < r^2) - 8|y_t|.$$

Hence, the agent is encouraged to take a tight turn around the roundabout without colliding with it. We train on 20 different uncertainty sets, with 3 sampled contexts from each set, for a total of 60 different contexts.

B.2. MINITAUR

This environment simulates an 8-DoF minitaur robot whose objective is to walk forward as quickly as possible. The mass of the robot's base and mass of the legs vary across tasks but are unknown. In each task, there is also a probability of failure p_{fail} for one of the four legs, which is also unknown. Each episode terminates when the robot falls or after 500 timesteps. At each timestep, the action input to one leg is dropped with probability p_{fail}. The agent's state consists of the robot's roll, roll rate, pitch, pitch rate, and the angles of each of the eight motors. We also append the history of last five actions, which the reward function depends on. The reward function is defined as

$$r_t = v_t - 0.01\|\mathbf{a}_t - 2\mathbf{a}_{t-1} + \mathbf{a}_{t-2}\|$$

where v_t is the observed velocity of the robot. We train on 80 different uncertainty sets, with 10 sampled contexts from each set, for a total of 800 different contexts.

B.3. HALF-CHEETAH

This environment modifies the Half-Cheetah task from OpenAI Gym (Brockman et al., 2016). The agent's objective is to run forward as quickly as possible, starting from rest. The mass of the agent and the friction of the joints vary across tasks and are unknown. Like in the Minitaur environment, there is a probability of failure p_{fail} for one of the six joints, which varies across tasks. Each episode lasts 500 timesteps.

At each timestep, the action input to one of the joints is dropped with probability p_{fail}. The agent's state consists of the velocity of the agent's center of mass and angular velocity of each of its six joints, and actions correspond to torques applied to each joint. The reward function is

$$r_t = v_t - 0.05\|\mathbf{a}_t\|$$

where v_t is the observed velocity of the agent. We train on 80 different uncertainty sets, with 10 sampled contexts from each set, for a total of 800 different contexts.

B.4. SAWYER PEG INSERTION

In this modified peg-insertion task (Zhao et al., 2020), a 7-DoF Sawyer robot arm needs to insert the peg attached to its end-effector into one of the two boxes in as few timesteps as possible. Across tasks, the scaling of the joint position controller and size of the peg vary and are unknown. The first box is closer to the initial position of the robot, but it also has a smaller hole. The second box on the other hand is farther from the initial position, and has a larger hole that will allow the agent to always successfully insert the peg into. Each episode lasts 50 timesteps, which is only enough time to try one of the boxes. The agent's state consists of the robot's joint angles, joint velocities, and end-effector pose. The reward at each time-step is 1 if the peg is successfully inserted into one of the boxes and 0 otherwise.

Since this is a sparse-reward task, we first pre-train an agent on the dense rewards for 300K environment steps with Soft Actor-Critic and save its replay buffer. We then initialize training of our method and of all comparisons with the restored replay buffer. We train on 80 different uncertainty sets, with 10 sampled contexts from each set, for a total of 800 different contexts.
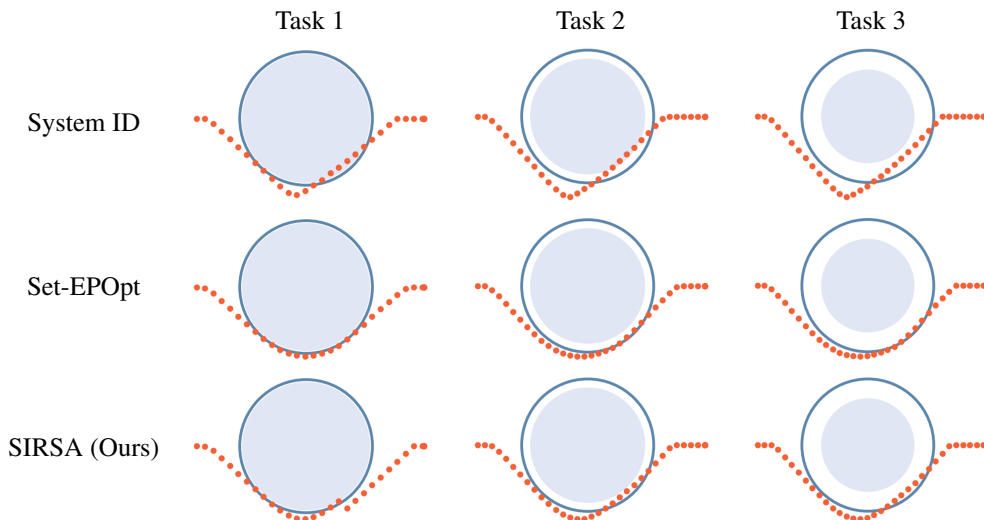
Table 6. Visualizations of the trajectories taken by policies learned with System ID, Set-EPOpt, and SIRSA in the Point Mass domain. The maximum obstacle size of this particular uncertainty set is demarcated by the unfilled blue circle, while the true obstacle in the environment is shaded in purple. The trajectory taken by the agent (from left to right) is in orange.

C. Experimental Results

C.1. POINT MASS VISUALIZATIONS

In Table 6, we visualize the trajectories of policies learned by System ID, Set-EPOpt, and SIRSA ($\alpha = 0.25$) in the Point Mass domain. The unfilled blue circle marks the maximum obstacle size of the particular uncertainty set, while the true obstacle is shaded in lavender. The agent always starts to the left of the obstacle and must reach the right side. We mark the trajectory in orange. The trajectories taken by System ID (top row) tend to be overly optimistic and make contact with the obstacle, incurring penalty. Meanwhile, the trajectories taken by Set-EPOpt (middle row) are always along the maximum obstacle of the uncertainty set, which can be overly conservative, e.g., in the third task (third column), when the true obstacle is smaller. Finally, SIRSA (bottom row) strikes a balance between the two. Specifically, in the first task (first column), the agent initially makes contact with the obstacle but corrects its trajectory thereafter. In the subsequent two tasks, the agent is not too conservative but avoids the obstacle most of the time.

C.2. PERFORMANCE FOR DIFFERENT CVAR α'S

In Tables 7 and 8, we report the full results for the α-dependent methods, EPOpt, Set-EPOpt, WCPG, Set-WCPG, and SIRSA (Ours), for α values in $\{0.25, 0.50, 0.75, 1.00\}$.

Task	Method	α	Min	Mean
Point mass	EPOpt	0.25	34.3 ± 0.6	38.7 ± 0.6
		0.50	35.2 ± 0.6	39.3 ± 0.4
		0.75	36.1 ± 0.6	39.7 ± 0.4
		1.00	36.3 ± 0.6	40.5 ± 0.4
	Set-EPOpt	0.25	34.4 ± 0.5	38.5 ± 0.4
		0.50	35.3 ± 0.6	39.2 ± 0.5
		0.75	36.1 ± 0.5	40.1 ± 0.3
		1.00	37.1 ± 0.5	40.7 ± 0.4
	WCPG	0.25	34.8 ± 0.6	39.3 ± 0.5
		0.50	34.7 ± 0.6	39.3 ± 0.5
		0.75	34.6 ± 0.6	39.3 ± 0.5
		1.00	34.4 ± 0.6	39.3 ± 0.5
	Set-WCPG	0.25	34.7 ± 0.7	39.0 ± 0.5
		0.50	34.6 ± 0.7	39.1 ± 0.5
		0.75	34.6 ± 0.7	39.2 ± 0.5
		1.00	34.6 ± 0.7	39.2 ± 0.6
	SIRSA (Ours)	0.25	37.9 ± 0.2	41.8 ± 0.1
		0.50	37.9 ± 0.2	41.7 ± 0.1
		0.75	37.5 ± 0.2	41.9 ± 0.1
		1.00	37.4 ± 0.3	41.3 ± 0.3
Minitaur	EPOpt	0.25	131.8 ± 10.7	168.7 ± 7.0
		0.50	160.1 ± 8.4	192.5 ± 5.2
		0.75	171.3 ± 7.4	196.6 ± 4.4
		1.00	172.2 ± 7.3	199.9 ± 5.5
	Set-EPOpt	0.25	99.3 ± 8.0	141.2 ± 6.2
		0.50	180.3 ± 8.7	211.5 ± 3.6
		0.75	181.7 ± 6.7	213.9 ± 2.4
		1.00	183.1 ± 7.5	216.7 ± 3.7
	WCPG	0.25	163.0 ± 17.4	191.5 ± 19.1
		0.50	163.5 ± 17.6	193.1 ± 19.3
		0.75	163.3 ± 18.1	193.6 ± 19.5
		1.00	165.5 ± 17.7	193.7 ± 19.6
	Set-WCPG	0.25	173.1 ± 9.2	204.7 ± 5.5
		0.50	173.4 ± 9.1	205.7 ± 5.3
		0.75	173.1 ± 9.3	206.7 ± 4.9
		1.00	174.5 ± 10.0	206.9 ± 4.7
	SIRSA (Ours)	0.25	169.1 ± 9.0	205.4 ± 3.8
		0.50	167.1 ± 9.3	199.6 ± 3.5
		0.75	149.6 ± 11.5	192.7 ± 5.0
		1.00	187.8 ± 7.6	214.3 ± 2.5

Table 7. Evaluation on 20 new uncertainty sets. Means and standard errors are computed over 10 random seeds for each method.

Task	Method	α	Min	Mean
Half-Cheetah	EPOpt	0.25	1292 ± 95	1478 ± 125
		0.50	1434 ± 165	1809 ± 214
		0.75	1863 ± 172	2192 ± 252
		1.00	2272 ± 218	2718 ± 325
	Set-EPOpt	0.25	2148 ± 42	2454 ± 47
		0.50	2845 ± 29	3274 ± 93
		0.75	3246 ± 206	3703 ± 171
		1.00	3811 ± 224	4474 ± 232
	WCPG	0.25	3703 ± 256	4264 ± 334
		0.50	3724 ± 263	4279 ± 328
		0.75	3688 ± 231	4291 ± 321
		1.00	3747 ± 229	4304 ± 316
	Set-WCPG	0.25	3418 ± 435	3973 ± 480
		0.50	3476 ± 445	3985 ± 476
		0.75	3823 ± 200	4420 ± 256
		1.00	3873 ± 207	4432 ± 253
	SIRSA (Ours)	0.25	3742 ± 209	4288 ± 228
		0.50	4126 ± 72	4806 ± 75
		0.75	3583 ± 298	4103 ± 280
		1.00	4146 ± 112	4872 ± 73
Peg Insertion	EPOpt	0.25	0.0 ± 0.0	0.0 ± 0.0
		0.50	11.4 ± 7.6	17.7 ± 11.2
		0.75	43.2 ± 5.4	75.1 ± 8.2
		1.00	41.3 ± 5.3	93.7 ± 4.9
	Set-EPOpt	0.25	22.1 ± 8.4	33.3 ± 11.2
		0.50	57.8 ± 7.1	78.0 ± 9.5
		0.75	70.6 ± 3.6	96.9 ± 2.7
		1.00	67.4 ± 4.7	77.8 ± 8.3
	WCPG	0.25	20.2 ± 5.4	41.7 ± 8.7
		0.50	28.8 ± 6.3	57.3 ± 8.0
		0.75	33.8 ± 7.2	63.2 ± 6.9
		1.00	31.8 ± 5.9	64.1 ± 7.3
	Set-WCPG	0.25	56.6 ± 4.5	80.4 ± 4.7
		0.50	64.0 ± 4.2	86.5 ± 4.2
		0.75	66.4 ± 4.0	89.4 ± 3.9
		1.00	68.3 ± 4.6	92.0 ± 4.4
	SIRSA (Ours)	0.25	81.1 ± 3.3	106.0 ± 1.9
		0.50	83.4 ± 4.5	109.5 ± 2.6
		0.75	76.2 ± 4.4	102.1 ± 3.4
		1.00	73.5 ± 4.7	103.1 ± 2.3

Table 8. Evaluation on 20 new uncertainty sets. Means and standard errors are computed over 10 random seeds for each method.