UNCERTAINTY-AWARE SELECTION OF ONLINE ALGORITHMS WITH SIMULATOR ENSEMBLES

Anonymous authors
Paper under double-blind review

ABSTRACT

The performance of online reinforcement learning depends critically on design choices, especially those that affect exploration. These choices are often selected by fitting a simulator to offline data, evaluating candidate algorithms in that simulator, and deploying the best-performing one. The simplest *Plug-In selection rule* simply selects the best performing algorithm on the fitted simulator, making evaluations unreliable when the offline data used to fit the simulator are limited. We investigate *Uncertainty-Aware selection*, which forms an ensemble of simulators—for example, obtained by bootstrap resampling—and selects the online algorithm with the best average performance across the ensemble. While ensemble-based approaches have been used to mitigate distribution shift and facilitate sim-to-real transfer, we formally show that this approach can mitigate the effects of limited data when fitting the simulator and theoretically has significant regret gains compared to Plug-In selection in multi-armed bandits. We also empirically investigate the Uncertainty-Aware selection approach in deep RL experiments on robotic control tasks that involve selecting reward-shaping hyperparameters, and show that it leads to more reliable selection and improved online performance.

1 INTRODUCTION

Selecting an effective online reinforcement learning (RL) algorithm to deploy in practice is challenging because the deployment performance of online RL algorithms can vary substantially with design decisions (e.g., architecture, hyperparameters, and reward scaling or shaping) (Henderson et al., 2018; Wang et al., 2022; Fan & Glynn, 2025). Moreover, deployment of online RL in important domains, including robotics (Kober et al., 2013; Dulac-Arnold et al., 2019; Ibarz et al., 2021), healthcare (Trella et al., 2022; Figueroa et al., 2021), power grids (Zhang et al., 2018; Chen et al., 2022b), is often very expensive and thus, it can be infeasible to test out many candidate online RL algorithms in a real deployment.

In many domains, however, one may have access to a (potentially limited) offline dataset before deployment to warm start or inform online RL algorithm design decisions. A common approach is to use the offline dataset to form an environment simulator that can be used as a surrogate to assess candidate algorithms (Mandel et al., 2016). In robotics, system identification approaches use a limited offline data to calibrate a physical simulator for policy training prior to deployment (Ljung, 1999; Peng et al., 2018; Muratore et al., 2022b). In digital health interventions, digital twins built using offline data are used to simulate patients for assessing candidate online intervention algorithms before deployment on a study population (Björnsson et al., 2020; Trella et al., 2022; Xu et al., 2026).

The appeal of the simulator approach is its generality: a simulator can run any candidate algorithm end to end no matter how complex the online RL algorithm design grows. Simulators can be constructed nonparametrically, by replaying logged transitions through queues (Mandel et al., 2016), nearest-neighbor retrieval (Wang et al., 2022), rejection sampling (Tang et al., 2022) or parametrically, by fitting a model of the environment to the data (Trella et al., 2022; Ghosh et al., 2024; Moerland et al., 2023; Yang et al., 2023). The evaluated algorithm can be adaptive, history-dependent, or otherwise non-trivial, whereas offline policy evaluation (Precup et al., 2000; Jiang & Li, 2016; Uehara et al., 2026) scores a fixed policy and does not directly extend to online RL algorithms.

Many existing approaches to fitting a simulator from offline data involve fitting a single model of the environment and selecting the candidate algorithm that performs the best on that fitted simulator.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

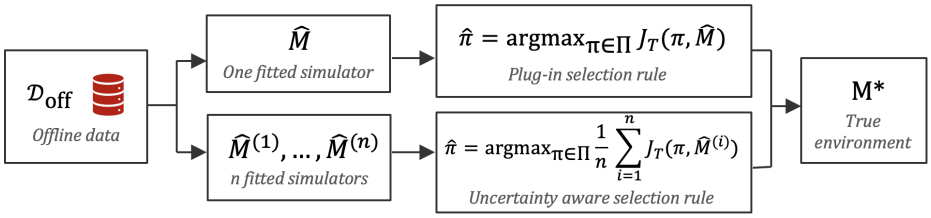


Figure 1: **Selecting an online RL algorithm using limited offline data.** Both rules fit simulators to $\mathcal{D}_{\text{off}}$ and then deploy the selected algorithm in the true environment M^*. The Plug-In rule commits to a single fitted simulator $\widehat{M}$; the Uncertainty-Aware rule averages the objective over $\widehat{M}^{(1)}, \ldots, \widehat{M}^{(n)}$, favoring algorithms that perform consistently across plausible realizations.

We call this selection approach the *Plug-In* selection rule. This approach has been used in practice in a variety of settings, including the Oralytics mobile health trial (Trella et al., 2022; 2024) and the educational game study (Mandel et al., 2016). In this paper, we identify a failure mode of the Plug-In simulator: it carries no information about how uncertain its estimate is, and we show that this omission can be costly when the offline data are scarce and the online horizon is long.

In this work, we advocate for an Uncertainty-Aware selection approach that constructs an *ensemble* of simulators (Figure 1). The ensemble consists of copies of the fitted simulator whose parameters are perturbed according to their estimation uncertainty, for example by refitting on bootstrap resamples of the offline data. Each candidate algorithm is scored by its average return across the simulators in the ensemble. Our contributions are as follows.

1. **Provable benefits of Uncertainty-Aware selection in bandits.** For UCB and Thompson Sampling in two-armed Gaussian bandits, we establish a provable and empirically validated mechanism by which accounting for simulator uncertainty improves algorithm selection. Specifically, Theorem 1 characterizes the asymmetric costs of exploration: underexploration incurs polynomial regret, whereas overexploration preserves logarithmic regret. Building on this result, we show that Uncertainty-Aware selection asymptotically chooses at least as much exploration as Plug-In (Proposition 2), resulting in no larger asymptotic regret growth and, with nonvanishing probability, a polynomial-to-logarithmic improvement (Theorem 3). Simulations support this mechanism, showing reductions in both the mean and variability of regret.
2. **Uncertainty-Aware selection as a curvature regularized version of Plug-In selection.** In Proposition 4, we formally show that Uncertainty-Aware selection criterion can be interpreted as a regularized version of the Plug-In selection criterion that depends on the curvature of the objective function with respect to the simulator parameters. This result holds for general decision-making settings (including bandits and Markov decision processes (MDPs)).
3. **Empirical benefit for specifying reward-shaping for deep RL.** We demonstrate the benefit of Uncertainty-Aware selection empirically in a deep RL reward-shaping task in which physics parameters of a MuJoCo simulator are identified from offline trajectories.

Taken together, our results show that ensembles of simulators are not only a tool for training robust policies: they provably improve the selection of online learning algorithms from limited offline data.

2 Problem Setup

Decision-making environment. States $S_{t,h}$ take values in state space $\mathcal{S}$ and actions $A_{t,h}$ in action space $\mathcal{A}$. M^* denotes the general decision-making environment, which specifies the distribution of the reward and next state given the most recent action, state, and history, i.e., $\mathbb{P}(R_{t,h}, S_{t,h+1} \mid A_{t,h}, S_{t,h}, \mathcal{H}_{t,h-1})$, where the history up to episode t and the within-episode timestep h:

$$\mathcal{H}_{t,h-1} := \left\{(S_{t,h'}, A_{t,h'}, R_{t,h'})\right\}_{h'=1}^{h-1} \cup \left\{(S_{t',h}, A_{t',h}, R_{t',h})\right\}_{t'=1,h=1}^{t'=t-1,h=H}.$$

In our theory and experiments M^* is either a finite-horizon MDP (Puterman, 2014) or a multi-armed bandit (MAB) environment (Lattimore & Szepesvári, 2020). In the MAB setting, $H = 1$ and $|\mathcal{S}| = 1$ so we use $\mathcal{H}_t = \{(A_{t'}, R_{t'})\}_{t'=1}^{t}$ to denote the history.

Offline dataset. We assume we have access to an offline dataset consisting of T_{off} episodes from environment M^* each of horizon H collected by a behavior policy π_b:

$$\mathcal{D}_{\text{off}} := \big\{(S_{t,1}, A_{t,1}, R_{t,1}), (S_{t,2}, A_{t,2}, R_{t,2}), \dots (S_{t,H}, A_{t,H}, R_{t,H})\big\}_{t=1}^{T_{\text{off}}}.$$

The policy π_b takes $(\mathcal{H}_{t,h-1}, S_{t,h})$ as input and outputs a distribution over actions $\mathcal{A}$. Note, in the MAB special case where $H = 1$ and $|\mathcal{S}| = 1$, for simplicity, we just write $\mathcal{D}_{\text{off}} := \{(A_t, R_t)\}_{t=1}^{T_{\text{off}}}$.

Algorithm selection objective. Our goal is to use the offline dataset $\mathcal{D}_{\text{off}}$ to inform the selection of an online RL algorithm $\pi \in \Pi$, which specifies action selection probabilities $\pi(A_{t,h} \mid S_{t,h}, \mathcal{H}_{t,h-1}) := \mathbb{P}(A_{t,h} \mid S_{t,h}, \mathcal{H}_{t,h-1})$. The algorithm π will be deployed for T episodes in the same environment M^* where $\mathcal{D}_{\text{off}}$ was collected. Specifically, we want to select algorithm π to maximize an objective function $J_T(\pi, M^*)$. Unless stated otherwise, throughout this work, the objective could be the expected cumulative reward

$$J_T(\pi, M^*) = \mathbb{E}_{\pi, M^*}\left[\sum_{t=1}^{T}\sum_{h=1}^{H} R_{t,h}\right], \tag{1}$$

where the expectation is taken over the random trajectories induced by the online algorithm π. In other settings like robotic learning, one might also be interested in the expected reward on the final episode $J_T(\pi, M^*) = \mathbb{E}_{\pi, M^*}\big[\sum_{h=1}^{H} R_{T,h}\big]$.

Candidate algorithm class. The class of candidate history-dependent policies Π could include exploration strategies of algorithm (e.g., ϵ-greedy (Wunder et al., 2010), Thompson Sampling (Russo et al., 2018), bonus-based exploration (Bellemare et al., 2016)), model architectures, state specification, reward specification/reward shaping, and algorithm hyperparameters. We index candidate online algorithms by a tuning parameter $\theta \in \Theta$, so $\Pi = \{\pi_\theta : \theta \in \Theta\}$. For example, in UCB, θ controls the width of the exploration bonus; in ϵ-greedy Q-learning, it may control the exploration rate. In general, θ may index other design choices, such as model architecture, state representation, or reward shaping. We use $\theta_T^* = \text{argmax}_{\theta\in\Theta} J_T(\pi_\theta, M^*)$ to denote the optimal θ for a given T.

Plug-In simulation selection rule. We use the offline dataset $\mathcal{D}_{\text{off}}$ to fit an approximate environment model $\widehat{M}$ of M^*, which may be an explicit dynamics model or a learned generative world model. We then use $\widehat{M}$ as a proxy environment in which to evaluate candidate decision procedures.

A simple and commonly used simulator-based selection rule (Moerland et al., 2023; Wang et al., 2022; Yang et al., 2023) is the Plug-In selection rule which forms an estimated simulator $\widehat{M}$

$$\widehat{\pi}_{\text{plug}} \in \arg\max_{\pi\in\Pi} J_T(\pi, \widehat{M}). \tag{2}$$

The above solution can be solved empirically by repeatedly rolling out the algorithm π in environment $\widehat{M}$ for different choices of algorithm $\pi \in \Pi$. [1]

Uncertainty-Aware simulation selection rule. In this work we advocate for an Uncertainty-Aware (UA) selection approach. This approach involves using $\mathcal{D}_{\text{off}}$ to fit an ensemble of simulators and choosing the algorithm candidate that does well on average across an ensemble of fitted simulators $\widehat{M}^{(1)}, \dots \widehat{M}^{(n)}$. Then the Uncertainty-Aware selection rule selects the algorithm as follows:

$$\widehat{\pi}_{\text{UA}} = \arg\max_{\pi\in\Pi} \frac{1}{n}\sum_{i=1}^{n} J_T\big(\pi, \widehat{M}^{(i)}\big), \tag{3}$$

Rather than maximizing performance on the single fitted simulator $\widehat{M}$, (3) instead evaluates candidates on average across ensembles $\widehat{M}^{(n)}$, which favors algorithms with consistently strong performance across plausible realizations of the simulator $\widehat{M}$. There are a variety of ways to construct the ensemble of simulators—including bootstrap and adding noise to the fitted simulator $\widehat{M}$.

Throughout, we construct the ensemble of simulators $\widehat{M}^{(1)}, \dots \widehat{M}^{(n)}$ using bootstrap-based approaches. In particular, we use both empirical and parametric bootstrapping. Empirical bootstrap

[1] Picking the best RL algorithm using replay-style simulators (Mandel et al., 2016; Tang et al., 2022; Wang et al., 2022) is also Plug-In selection where the simulator is an empirical distribution fit non-parametrically.

resamples the episodes in $\mathcal{D}_{\text{off}} = \big\{(S_{t,1}, A_{t,1}, R_{t,1}), (S_{t,2}, A_{t,2}, R_{t,2}), \dots (S_{t,H}, A_{t,H}, R_{t,H})\big\}_{t=1}^{T_{\text{off}}}$ with replacement. In simpler models, e.g., Gaussian bandits, we use parametric bootstrap methods to construct the ensemble (van der Vaart, 1998, Chapter 23.1).

2.1 Related work

Our main finding is a more robust and plausible algorithm selection through ensembled simulators that are uncertainty-aware.

This approach is similar to domain randomization (DR) and its data-calibrated variants (Tobin et al., 2017; Peng et al., 2018; Ramos et al., 2019; Chebotar et al., 2019; Muratore et al., 2022a; Tiboni et al., 2023) that also train against a distribution of simulators. Existing literature on DR primarily concerns the mismatch between the simulator and the deployment environment, and its purpose is to make one fixed policy robust to that mismatch. In our setting there is no mismatch, since the offline data and the deployment come from the same environment; the spread of the ensemble reflects only the sampling variability of a finite dataset and vanishes as the data grow, and the gain arises because the candidates are online learners whose regret responds asymmetrically to the estimation error.

Ensembles are also standard inside model-based RL, where bootstrapped or probabilistic ensembles quantify epistemic uncertainty to drive exploration (Osband et al., 2016; Chua et al., 2018) or to penalize policies that visit states where the members disagree (Yu et al., 2020; Kidambi et al., 2020). In those methods the ensemble is part of the algorithm being run and changes how it acts; in ours the ensemble sits outside the candidates, which are evaluated as black boxes, and it changes only which candidate algorithm is chosen.

A similar distinction separates our rule from robust and distributionally robust RL (Iyengar, 2005; Nilim & El Ghaoui, 2005; Eysenbach & Levine, 2022), which optimize a worst case over an uncertainty set by solving an optimization problem: we show that a plain average over the ensemble is already a curvature regularization of the Plug-In criterion (Proposition 4), and that this average suffices for the regret gains of Section 3.

Finally, the object being selected differs from that of offline policy selection (Konyushova et al., 2021; Yang et al., 2022; Zitovsky et al., 2023), which ranks fixed policies through fitted value functions; we rank learners that keep learning after deployment, so what the simulator must get right is their ordering by online regret. Among the works that select online algorithms (Wang et al., 2022; Tang et al., 2022), no prior work asks how the estimation error of a data-fitted simulator propagates into the choice of an online learning algorithm, or shows that an ensemble corrects it. See Appendix A for a more complete review of the related literature.

3 Plug-In vs. UA selection in multi-armed bandits

In this section, we characterize the cost of poor hyperparameter selection and compare the Plug-In and Uncertainty-Aware selection rules in a simple **Gaussian multi-armed bandit environment** M^*, in which $H = 1$, $|\mathcal{S}| = 1$, $\mathcal{A} = \{1, 2\}$, and rewards $R_t | A_t = a \sim \mathcal{N}(\mu_a, \sigma^2)$. More specifically, we study two common MAB algorithms with a hyperparameter θ that impacts the amount of exploration. Our results center around two key findings, which we support empirically and theoretically:

(a) Underexploration can be substantially more costly than overexploration: an exploration parameter θ that is too small can lead to much larger regret than one that is too large.

(b) Uncertainty-Aware selection tends to favor greater exploration than Plug-In selection, thereby reducing the risk of costly underexploration and leading to improved regret.

Bandit algorithms. We focus on two MAB algorithms. For both algorithms, θ controls the amount of exploration: larger values of θ induce more exploration, while smaller values induce less exploration. The first, UCB (Auer et al., 2002; Lattimore & Szepesvári, 2020; Abbasi-Yadkori et al., 2011) at round t pulls the arm

$$A_t \in \arg\max_{a \in \mathcal{A}} \big\{\widehat{\mu}_a(t-1) + \sqrt{2\theta \log t / N_a(t-1)}\big\}, \tag{4}$$

where $N_a(t) = \sum_{\tau=1}^{t} \mathbb{1}(A_\tau = a)$ and $\widehat{\mu}_a(t) = \sum_{\tau=1}^{t} \mathbb{1}(A_\tau = a) R_\tau / N_a(t)$ are the running pull count and mean reward of arm a, respectively. Here the hyperparameter $\theta > 0$ is the *exploration radius*.

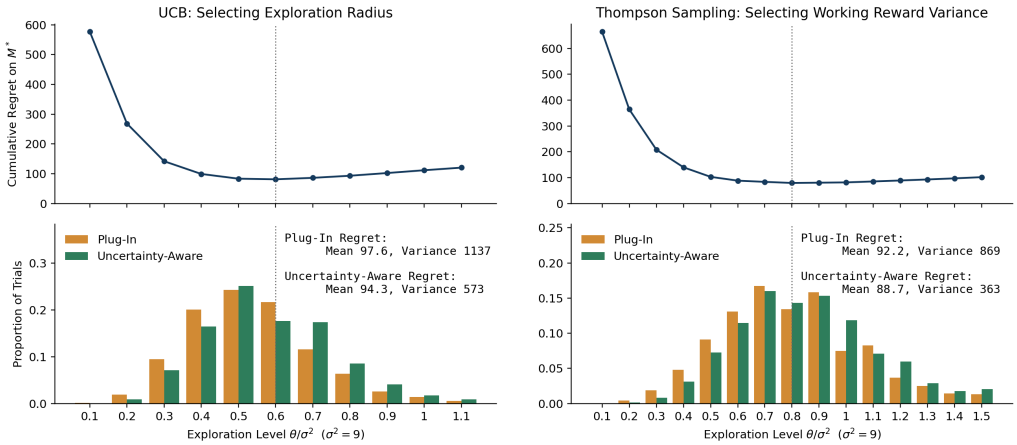


Figure 2: **Asymmetric costs of exploration and the benefit of Uncertainty-Aware selection.** Simulations are in a two-armed Gaussian bandit with $T = 5000$, $T_{\text{off}} = 25$, margin $\mu_1 - \mu_2 = 1$, and noise standard deviation $\sigma = 3$. *Top:* Regret of UCB (left) and Thompson Sampling (right) as a function of the exploration level θ (see (4) and (5)). Regret rises far more steeply below the optimal value θ^*_T (dotted vertical line) than above it, i.e., underexploration is more costly. *Bottom:* Using multiple samples of $\mathcal{D}_{\text{off}}$, Uncertainty-Aware selection (ensemble models constructed via bootstrap) more often selects higher values of θ compared to Plug-In, which leads to lower regret on average. Additionally, Uncertainty-Aware selection has lower variance in regret across trials.

The second algorithm, Thompson Sampling (TS) (Russo et al., 2018) forms a Bayesian model of the reward means where we have a flat prior over arm means μ_a, i.e., $P(\mu_a) \propto 1$, and rewards are conditionally Gaussian with variance θ, i.e., $R_t|A_t = a, \mu_a \sim \mathcal{N}(\mu_a, \theta)$. At each time t, the posterior distribution is then $\mu_a|\mathcal{H}_{t-1} \sim \mathcal{N}(\widehat{\mu}_a(t-1), \theta/N_a(t-1))$. TS selects actions via probability matching in that actions are selected according to the posterior probability that they are optimal:

$$\mathbb{P}(A_t = a \mid \mathcal{H}_{t-1}) = \mathbb{P}(a = \text{argmax}_{a' \in \mathcal{A}} \mu_{a'} \mid \mathcal{H}_{t-1}). \tag{5}$$

We refer to the hyperparameter $\theta > 0$ as the *working reward variance*.

3.1 Asymmetric costs of under- and overexploration

For both UCB and TS, given an exploration parameter θ, we measure the online performance of π_θ by its expected cumulative regret:

$$\text{Reg}_T(\pi_\theta, M^*) = T\mu^*(M^*) - J_T(\pi_\theta, M^*), \tag{6}$$

where $\mu^*(M^*) = \max_{a \in \{1,2\}} \mu_a$, and π_θ denotes either UCB or TS with hyperparameter θ. Figure 2 (top) illustrates a pronounced asymmetry: regret increases much more sharply when θ falls below the best-performing choice than when it increases above it. We observe the same pattern across a range of environment configurations in Appendix B.5.

The following result provides a theoretical characterization of this asymmetry for both UCB and TS.

Theorem 1 (Asymmetric regret of UCB and TS)**.** *Let M^* be a two-armed Gaussian bandit with $\mu_1 > \mu_2$ and $\sigma^2 > 0$. Let π_θ denote either the UCB algorithm (4) or the TS algorithm (5) with hyperparameter $\theta > 0$. Then:*

(i) For any $\theta \in (0, \sigma^2)$, $\lim_{T\to\infty} \frac{\log \text{Reg}_T(\pi_\theta, M^*)}{\log T} = 1 - \frac{\theta}{\sigma^2}$.

(ii) For any $\theta \geq \sigma^2$, $\lim_{T\to\infty} \frac{\text{Reg}_T(\pi_\theta, M^*)}{\log T} = \frac{2\theta}{\mu_1 - \mu_2}$.

Theorem 1 identifies σ^2 as a critical exploration level for both algorithms. Below this threshold, regret grows polynomially as $T^{1-\theta/\sigma^2+o(1)}$; at or above the threshold, it grows only logarithmically, with leading constant $2\theta/(\mu_1 - \mu_2)$. Thus, insufficient exploration can change the order of regret from logarithmic to polynomial, whereas additional exploration above the threshold affects only the leading constant of the logarithmic regret. This asymmetry is also reflected in the finite-horizon results in Figure 2 (top).

Theorem 1 is closely related to the results in Fan & Glynn (2025) and Fan & Glynn (2022). In Appendices D and E, we establish stronger results for both algorithms: the regret exponent converges *uniformly* in θ over compact subsets of $(0, \infty)$, while the leading-order logarithmic asymptotics hold uniformly over compact subsets of (σ^2, ∞). We also provide associated finite-horizon regret bounds that are uniform in both θ and the suboptimality gap (Lemma 8 and 14). These uniformity results are important for analyzing data- and horizon-dependent hyperparameter selection. Our uniform polynomial lower bounds for both algorithms exploit low initial rewards that delay further sampling of the optimal arm, following the intuition of Fan & Glynn (2025). For the upper bounds, we adapt a pull-count decomposition argument to obtain the required uniform control; the TS analysis further builds on the posterior-odds comparison of Agrawal & Goyal (2013), for which we derive uniform bounds on truncated Gaussian odds to handle underexploration.

3.2 Uncertainty-Aware selection favors exploration and improves regret

Uncertainty-Aware selection lowers both the mean and the variance of regret (Figure 2). We empirically compare the Plug-In and Uncertainty-Aware selection rules for UCB and Thompson Sampling. We use a Gaussian reward two-armed bandit simulation environment $M^* = \big(\mathcal{N}(\mu_1, \sigma_1^2), \mathcal{N}(\mu_2, \sigma_2^2)\big)$. The Plug-In selection rule fits $\widehat{M} = \big(\mathcal{N}(\widehat{\mu}_1, \widehat{\sigma}^2), \mathcal{N}(\widehat{\mu}_2, \widehat{\sigma}^2)\big)$, where $\widehat{\mu}_a$ is the sample mean of arm $a \in \{1, 2\}$ and $\widehat{\sigma}^2$ is the reward variance pooled across the two arms, both estimated using $\mathcal{D}_{\text{off}}$. The Uncertainty-Aware selection rule forms an ensemble of simulators $\widehat{M}^{(1)}, \ldots, \widehat{M}^{(n)}$ by a parametric bootstrap: member i is $\widehat{M}^{(i)} = \big(\mathcal{N}\big(\widehat{\mu}_1^{(i)}, (\widehat{\sigma}^{(i)})^2\big), \mathcal{N}\big(\widehat{\mu}_2^{(i)}, (\widehat{\sigma}^{(i)})^2\big)\big)$, where $\widehat{\mu}_a^{(i)}$ and $(\widehat{\sigma}^{(i)})^2$ are the same estimates computed on a fresh dataset of T_{off} pairs drawn from $\widehat{M}$ under the behavior policy (Appendix B.3).

Figure 2 (bottom) shows that Uncertainty-Aware selection has lower regret compared to Plug-In. The lower regret under UA selection is because it selects higher values of exploration compared to Plug-In. Specifically, for Plug-In selection 61.0% of the regret is due to selecting a θ below θ_T^*, while for Uncertainty-Aware selection only 50.6% of regret is due to selecting a θ below θ_T^*. The Plug-In rule therefore underperforms not because its estimate of θ is noisy, but because the errors that fall on the side of too little exploration are the expensive ones, and their cost compounds with the horizon. Furthermore, the variance of the cumulative regret across trials is dramatically lower under Uncertainty-Aware selection compared to Plug-In (573 vs 1137 for UCB; 363 vs 869 for TS). We discuss how Uncertainty-Aware selection can be interpreted as a curvature regularized version of the Plug-In objective in Section 4.

Uncertainty-Aware selection leads to greater exploration (Figure 3). Figure 3 illustrates the mechanism behind improvement in regret under UA selection. Across much of the region of $(\widehat{\Delta}, \widehat{\sigma})$ values supported by the offline data, the Uncertainty-Aware rule selects a normalized exploration parameter θ/σ^2 that is at least as large as that selected by the Plug-In rule. Thus, accounting for uncertainty in the fitted simulator systematically guards against selections that are overly exploitative under a favorable but inaccurate point estimate. This effect is not simply a uniform increase in exploration: it is concentrated in regions where uncertainty about the reward gap and noise level can change the relative performance of candidate settings. By avoiding these costly underexploratory selections while making similar choices when the fitted model is sufficiently informative, the Uncertainty-Aware rule reduces both the mean and variability of regret observed above.

We next provide theoretical support for these empirical findings. For simplicity, we assume the offline dataset $\mathcal{D}_{\text{off}}$ was collected by a behavior policy that samples each arm $T_{1,\text{off}} := \lfloor T_{\text{off}}/2 \rfloor$ times, and let $\widehat{M} = \big(\mathcal{N}(\widehat{\mu}_1, \widehat{\sigma}_1^2), \mathcal{N}(\widehat{\mu}_2, \widehat{\sigma}_2^2)\big)$ denote the fitted environment. For either UCB or TS, let π_θ denote the algorithm with hyperparameter θ. For the selection analysis, we restrict the candidate parameters to a compact interval $\Theta = [\underline{\theta}, \overline{\theta}]$ satisfying $0 < \underline{\theta} < \sigma^2 < \overline{\theta} < \infty$.

The Plug-In rule selects $\widehat{\theta}_{\text{plug},T} \in \arg\max_{\theta \in \Theta} J_T(\pi_\theta, \widehat{M})$. For the theoretical analysis of the Uncertainty-Aware rule, we consider a simplified parametric-bootstrap ensemble (van der Vaart, 1998) that perturbs the fitted arm means according to their estimated sampling uncertainty. Conditional on $\mathcal{D}_{\text{off}}$, each ensemble member is

$$\widehat{M}^{(i)} = \Big(\mathcal{N}\big(\widehat{\mu}_1 + \widehat{\sigma}_1/\sqrt{T_{1,\text{off}}} \cdot Z_1^{(i)},\, \widehat{\sigma}_1^2\big),\, \mathcal{N}\big(\widehat{\mu}_2 + \widehat{\sigma}_2/\sqrt{T_{1,\text{off}}} \cdot Z_2^{(i)},\, \widehat{\sigma}_2^2\big)\Big), \tag{7}$$

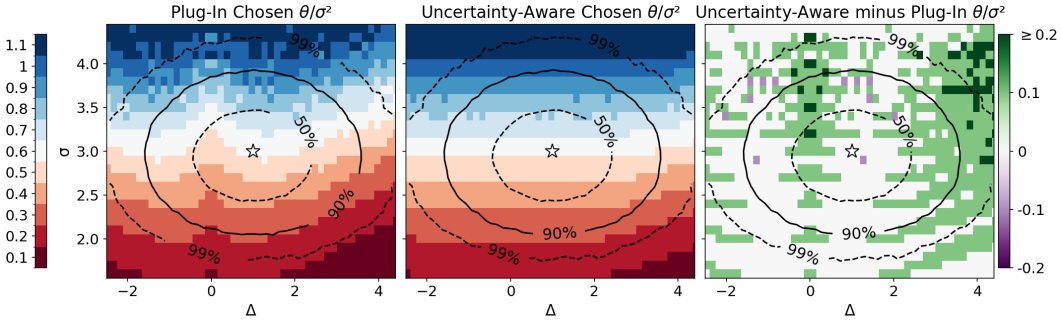


Figure 3: **Uncertainty-Aware (UA) selection leads to greater exploration than Plug-In under UCB.** We independently sample 10^6 $\mathcal{D}_{\text{off}}$ datasets and compute an estimate of the margin $\Delta = \mu_1 - \mu_2$ and the reward standard deviation σ. The star depicts the true value of $\Delta = 1$ and $\sigma = 3$ in the environment. The contour lines at 50% depicts where 50% of the 10^6 estimated $\widehat{\Delta}, \widehat{\sigma}$ are; similarly for the 90%, and 99% contour lines. We depict the selected value of θ/σ^2 for each combination of (Δ, σ) under the Plug-In selection rule (left) and that for the UA selection rule (center). In the right most plot, we depict the difference in chosen θ/σ^2 for the Plug-In and UA rules. The plot shows that the UA selection rule leads to similar or greater exploration than Plug-In for most values of (Δ, σ). Combined with the asymmetric cost of underexploration (Section 3.1), this helps explain how UA decreases regret compared to Plug-In.

where $Z_1^{(i)}, Z_2^{(i)} \overset{\text{iid}}{\sim} \mathcal{N}(0,1)$. We analyze the population counterpart of the finite-ensemble UA objective in (3), obtained in the large-ensemble limit:

$$\widehat{\theta}_{\text{UA},T} := \arg\max_{\theta \in \Theta} \mathbb{E}\Big[J_T(\pi_\theta, \widehat{M}^{(i)}) \,\Big|\, \mathcal{D}_{\text{off}} \Big]. \tag{8}$$

The maximizers $\widehat{\theta}_{\text{plug},T}$ and $\widehat{\theta}_{\text{UA},T}$ exist almost surely over $\mathcal{D}_{\text{off}}$ (see Appendices D, E).

Our first result is Proposition 2 which shows that, by accounting for uncertainty in the fitted environment, the UA selection rule asymptotically never selects less exploration than the Plug-In rule.

Proposition 2 (Uncertainty-Aware selection leads to higher exploration)**.** *Consider a two-armed Gaussian bandit M^* with $\mu_1 > \mu_2$ and $\sigma^2 > 0$. For either UCB or TS, almost surely over $\mathcal{D}_{\text{off}}$,* $\liminf_{T\to\infty} \widehat{\theta}_{\text{UA},T} \geq \lim_{T\to\infty} \widehat{\theta}_{\text{plug},T}$.

The greater exploration induced by UA selection translates into a more favorable asymptotic regret growth rate. To state this comparison, define the asymptotic log ratio between the cumulative regret growth rates of Plug-In and UA selection:

$$G(\mathcal{D}_{\text{off}}) := \liminf_{T\to\infty} \frac{\log \text{Reg}_T(\pi_{\widehat{\theta}_{\text{plug},T}}, M^*)}{\log \text{Reg}_T(\pi_{\widehat{\theta}_{\text{UA},T}}, M^*)}. \tag{9}$$

Theorem 3. *In a two-armed Gaussian bandit M^* with $\mu_1 > \mu_2$ and $\sigma^2 > 0$, for either UCB or TS:*

(i) For any $T_{\text{off}} \geq 4$, $G(\mathcal{D}_{\text{off}}) \geq 1$ almost surely.

(ii) $\liminf_{T_{\text{off}}\to\infty} \mathbb{P}(G(\mathcal{D}_{\text{off}}) = \infty) \geq \frac{1}{4}$.

Proposition 2 provides the intuition for Theorem 3(i): UA asymptotically selects at least as much exploration as Plug-In, while the regret exponent decreases with the exploration parameter. A formal argument requires uniform regret bounds for the data-dependent sequences $\widehat{\theta}_{\text{plug},T}$ and $\widehat{\theta}_{\text{UA},T}$. Part (ii) further shows that UA can strictly improve the regret growth rate. The proof identifies an event whose probability converges to $1/4$ as $T_{\text{off}} \to \infty$, on which Plug-In selects an exploration parameter below the critical level σ^2, whereas UA selects above it. By the asymmetric regret analysis in Section 3.1, the resulting deployment regrets are polynomial and logarithmic in T, respectively, and hence $G(\mathcal{D}_{\text{off}}) = \infty$. Proofs for UCB and TS are given in Appendices D and E, respectively.

4 BANDITS AND BEYOND: UNCERTAINTY-AWARE SIMULATION SELECTION AS A CURVATURE REGULARIZATION

Motivated by the bandit results, we next give a local characterization of how UA selection differs from Plug-In selection. This characterization does not rely on the bandit setting: it applies whenever an offline dataset is used to fit a simulator and candidate decision procedures are evaluated within that simulator. The simulator may be an explicit dynamics-and-reward model or a learned latent world model, and the evaluation criterion J_T may be application-specific, e.g., could incorporate risk-sensitive utility, safety, or another deployment criterion.

Plug-In selection ranks candidates using a single fitted simulator, whereas UA selection ranks them by average performance across plausible perturbations of that simulator. We show that, locally, this averaging induces a curvature-dependent adjustment to the Plug-In objective. This provides one interpretation of UA selection as regularization at the selection stage: it can downweight candidates whose favorable Plug-In evaluation is sensitive to uncertainty in the fitted simulator, without modifying the candidate algorithm itself.

Curvature correction for simulator uncertainty. To make this interpretation precise, consider a parametric simulator family $\{M_\lambda : \lambda \in \mathbb{R}^d\}$ and suppose that the fitted simulator is $\widehat{M} = M_{\widehat{\lambda}}$. In particular, λ may parameterize transition dynamics, rewards, or components of a latent world model.

Let an ensemble simulator take the form $M^{(i)} = M_{\widehat{\lambda}+\epsilon^{(i)}}$ where, conditional on $\mathcal{D}_{\text{off}}$, $\mathbb{E}[\epsilon^{(i)} \mid \mathcal{D}_{\text{off}}] = 0$, $\mathbb{E}[\epsilon^{(i)}(\epsilon^{(i)})^\top \mid \mathcal{D}_{\text{off}}] = \Sigma$, and $\mathbb{E}[\|\epsilon^{(i)}\|_2^3 \mid \mathcal{D}_{\text{off}}] < \infty$. Here, the perturbations $\epsilon^{(i)}$ may arise from an empirical bootstrap, a parametric bootstrap, posterior samples, or another ensemble construction intended to represent uncertainty in the fitted simulator.

The Plug-In selection rule selects $\widehat{\theta}_{\text{plug},T} \in \arg\max_{\theta\in\Theta} J_T(\pi_\theta, M_{\widehat{\lambda}})$. The UA objective here maximizes the population counterpart of the finite-ensemble UA objective from (3):

$$\widehat{\theta}_{\text{UA},T} \in \arg\max_{\theta\in\Theta} \mathbb{E}\left[J_T\big(\pi_\theta, M_{\widehat{\lambda}+\epsilon^{(i)}}\big) \,\middle|\, \mathcal{D}_{\text{off}}\right]. \tag{10}$$

Proposition 4 (Ensembles induce a curvature correction)**.** *Suppose $\lambda \mapsto J_T(\pi_\theta, M_\lambda)$ is twice continuously differentiable and its Hessian is L-Lipschitz, i.e., $\|\nabla^2_\lambda J_T(\pi_\theta, M_\lambda) - \nabla^2_\lambda J_T(\pi_\theta, M_{\lambda'})\|_{\text{op}} \le L\|\lambda - \lambda'\|_2$ for all λ, λ', where $\|\cdot\|_{\text{op}}$ is the operator norm. Then*

$$\left|\mathbb{E}\big[J_T(\pi_\theta, M_{\widehat{\lambda}+\epsilon^{(i)}}) \mid \mathcal{D}_{\text{off}}\big] - J_T(\pi_\theta, M_{\widehat{\lambda}}) - \tfrac{1}{2}\operatorname{Tr}\left(\nabla^2_\lambda J_T(\pi_\theta, M_{\widehat{\lambda}})\Sigma\right)\right| \le \frac{L}{6}\,\mathbb{E}\big[\|\epsilon^{(i)}\|_2^3 \mid \mathcal{D}_{\text{off}}\big]. \tag{11}$$

Proposition 4 shows that UA selection is locally a curvature-adjusted version of Plug-In selection. In directions of simulator uncertainty along which a candidate's value is locally concave, the correction in (11) is negative: performance lost under unfavorable perturbations exceeds performance gained under equally sized favorable perturbations. UA selection therefore lowers the ranking of candidates whose high Plug-In score is brittle to uncertainty in the fitted simulator. UA selection favors candidates according to their value curvature in the uncertain directions of the simulator. This is why we view UA selection as a form of regularization rather than as a rule that uniformly penalizes exploration, optimism, or any particular hyperparameter value. The proof is in Appendix F.

5 DEEP RL EXPERIMENTS

We apply UA selection to continuous control in MuJoCo (Todorov et al., 2012), tuning the reward-shaping weights of PPO (Schulman et al., 2017; Raffin et al., 2021) in the setup of Xie et al. (2025), on simulators whose physics parameters are identified from offline data.

Tasks and online learners. The tasks are `Hopper` and `Walker2d` from Gymnasium (Towers et al., 2025), in which a planar robot with three (`Hopper`) or six (`Walker2d`) actuated joints must move forward without falling, with horizon $H = 250$. Reward shaping trains the learner on a hand-designed reward with several terms, and the weights on those terms strongly affect the trained policy (Xie et al., 2025). Here the tuning parameter $\theta = (\theta_{\text{surv}}, \theta_{\text{fwd}}, \theta_{\text{ctrl}})$ collects the weights of three handcrafted heuristic reward terms, a survival bonus, a forward-progress term and a control-cost penalty, with $\theta_{\text{surv}} = 1$ fixed (Appendix C.1). A larger ratio $\theta_{\text{fwd}}/\theta_{\text{ctrl}}$ pays more for moving

Table 1: Success score of the weight vector selected by the Plug-In and UA rules on the true environment, mean $\pm$ standard error and 10th percentile over 300 paired replications; higher is better. The better of the two rules is in bold, and * marks a significant difference.

	`Hopper`, $J_T(\pi_{\theta^*_T}, M^*) = 0.991$				`Walker2d`, $J_T(\pi_{\theta^*_T}, M^*) = 0.689$			
T_{off}	Plug-In	UA	q_{10} Plug-In	q_{10} UA	Plug-In	UA	q_{10} Plug-In	q_{10} UA
20	0.933 ± 0.007	$\mathbf{0.956 \pm 0.005}^*$	0.764	**0.930**	0.585 ± 0.006	$\mathbf{0.594 \pm 0.006}$	0.456	0.456
80	0.965 ± 0.003	$\mathbf{0.971 \pm 0.003}$	0.930	0.930	0.594 ± 0.005	$\mathbf{0.616 \pm 0.005}^*$	0.456	**0.517**
320	0.976 ± 0.002	$\mathbf{0.981 \pm 0.002}$	0.930	**0.936**	0.590 ± 0.005	$\mathbf{0.610 \pm 0.005}^*$	0.456	**0.527**

forward and charges less for actuation, so the trained policy runs harder; we call such a θ more aggressive. The candidate algorithm π_θ is PPO trained on the reward with weights θ, with every other setting held fixed (Appendix C.3), and the candidate set Θ is a grid of 25 weight vectors on `Hopper` and 30 on `Walker2d` (Appendix C.2).

Following Xie et al. (2025), the performance is measured by a success score that is zero if the average squared actuation over the episode exceeds a bound $c_{\max}$, and otherwise rises linearly from 0 to 1 as the distance traveled increases from a threshold x_{lo} to a threshold x_{hi} (Appendix C.1). We evaluate the performance of the trained policy at $T = 3 \times 10^5$ environment steps. Every reported value is a Monte Carlo estimate over 50 evaluation episodes, which we call the success score.

Simulator and offline data. The simulator family is $\{M_\lambda\}$, where M_λ is the MuJoCo model of the robot with $\lambda = (\lambda_{\text{mass}}, \lambda_{\text{fric}}, \lambda_{\text{damp}}) \in \mathbb{R}^3$ multiplying the body masses, the friction coefficients and the joint damping of a baseline model. Estimating λ from logged data is standard practice in robotics, known as system identification (Ljung, 1999; Tiboni et al., 2023). The true environment is $M^* = M_{\lambda^*}$ with $\lambda^* = (1.5, 0.7, 1.4)$, a robot that is heavier, slipperier and more damped than the baseline model; the practitioner typically knows the model but not λ^*. We add Gaussian noise to the observations, in the offline data, in every simulator and on the true environment, so that identifying λ from $\mathcal{D}_{\text{off}}$ is a nontrivial task (Appendix C.4).

The offline dataset $\mathcal{D}_{\text{off}}$ consists of $T_{\text{off}} \in \{20, 80, 320\}$ episodes collected on M^* by a behavior policy π_b that draws actions uniformly at random. For each offline episode $t = 1, \ldots, T_{\text{off}}$ we fit $\widehat{\lambda}_t$ by nonlinear least squares over its transitions (Tiboni et al., 2023) combined with a bias-correction method called SIMEX (Cook & Stefanski, 1994; Carroll et al., 2006). The Plug-In simulator is $\widehat{M} = M_{\widehat{\lambda}}$ with $\widehat{\lambda} = T_{\text{off}}^{-1} \sum_{t=1}^{T_{\text{off}}} \widehat{\lambda}_t$, the average of the per-episode estimates. The UA ensemble is the empirical bootstrap over episodes. Member $i = 1, \ldots, n = 12$ is $\widehat{M}^{(i)} = M_{\widehat{\lambda}^{(i)}}$ with $\widehat{\lambda}^{(i)} = T_{\text{off}}^{-1} \sum_{t \in \mathcal{I}^{(i)}} \widehat{\lambda}_t$, where $\mathcal{I}^{(i)}$ is a multiset of T_{off} episodes. We give both rules the same budget of $K = 12$ training runs per θ. We report the success score of the selected $\widehat{\theta}$ on the true environment, averaged over 300 replications, each of which draws a fresh $\mathcal{D}_{\text{off}}$ that both rules then see (Appendix C.7).

Results. Table 1 reports the six cells, two tasks by three offline dataset sizes. UA selection has the higher mean success score in every cell, by 0.005 to 0.023, significantly in three. Its 10th percentile (last two columns) is higher in four cells and equal in the other two. Figure 7 in Appendix C.8 validates the regularization effect of UA (Section 4).

6 Discussion

Our setting isolates simulator-estimation uncertainty by considering offline data and online deployment from the same environment. This is deliberately complementary to work on distribution shift, where techniques such as domain randomization can improve robustness by accounting for variation between training and deployment conditions. Our results show that simulator ensembles can improve algorithm selection even in the absence of such shift: averaging across plausible simulators mitigates uncertainty arising solely from limited offline data. An important direction for future work is to study how Uncertainty-Aware selection and distribution-shift methods can be combined when the deployment environment may differ systematically from the environment that generated the offline data. Other directions include extending the theoretical analysis beyond bandits to MDPs and a broader range of online algorithms, and studying richer algorithm-design choices such as architectures, state representations, and reward specifications. Finally, practical progress will require ensemble-construction methods that quantify uncertainty reliably in complex simulators.

AI USE STATEMENT

In this work, we used generative AI tools to assist with implementing experimental code, editing draft text for clarity and readability, and checking and refining mathematical derivations and proofs. We did not use generative AI tools to generate synthetic datasets; propose or refine hypotheses; formulate theoretical models and mathematical claims; design the experimental methodology; clean or analyze data; or interpret experimental results. All AI-assisted code was reviewed by multiple authors and verified through the experimental evaluation pipeline. All AI-assisted mathematical content was independently checked and verified by the authors, and all AI-assisted text was reviewed and revised by the authors. We take responsibility for the final content of this work, including text, claims, proofs, code, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Proofs for all theoretical results and additional experimental details are in our Appendices. The code to reproduce the main results in our paper is included in our submission and will be released publicly on GitHub.

REFERENCES

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems*, 24, 2011.

Shipra Agrawal and Navin Goyal. Further optimal regret bounds for Thompson sampling. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 31 of *Proceedings of Machine Learning Research*, pp. 99–107, 2013.

Yarden As, Chengrui Qu, Benjamin Unger, Dongho Kang, Max van der Hart, Laixi Shi, Stelian Coros, Adam Wierman, and Andreas Krause. SPiDR: A simple approach for zero-shot safety in sim-to-real transfer. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*, pp. 100933–100975, 2025.

Karl Johan Åström and Peter Eykhoff. System identification—a survey. *Automatica*, 7(2):123–162, 1971.

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

Juan Cruz Barsce, Jorge A. Palombarini, and Ernesto Martínez. A hierarchical two-tier approach to hyper-parameter optimization in reinforcement learning. In *Anales del Simposio Argentino de Inteligencia Artificial (ASAI), 48 Jornadas Argentinas de Informática (JAIIO)*, pp. 32–38, 2019.

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Rémi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Bergthor Björnsson, Carl Borrebaeck, Nils Elander, Thomas Gasslander, Danuta R. Gawel, Mika Gustafsson, Rebecka Jörnsten, Eun Jung Lee, Xinxiu Li, Sandra Lilja, David Martínez-Enguita, Andreas Matussek, Per Sandström, Samuel Schäfer, Margaretha Stenmarker, Xiao-Feng Sun, Oleg Sysoev, Huan Zhang, and Mikael Benson. Digital twins to personalize medicine. *Genome Medicine*, 12(1):4, 2020. doi: 10.1186/s13073-019-0701-3.

Raymond J. Carroll, David Ruppert, Leonard A. Stefanski, and Ciprian M. Crainiceanu. *Measurement Error in Nonlinear Models: A Modern Perspective*. Chapman and Hall/CRC, 2nd edition, 2006.

Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8973–8979. IEEE, 2019.

Xiaoyu Chen, Jiachen Hu, Chi Jin, Lihong Li, and Liwei Wang. Understanding domain randomization for sim-to-real transfer. In *International Conference on Learning Representations*, 2022a.

Xin Chen, Guannan Qu, Yujie Tang, Steven Low, and Na Li. Reinforcement learning for selective key applications in power systems: Recent advances and future challenges. *IEEE Transactions on Smart Grid*, 13(4):2935–2958, 2022b.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

John R. Cook and Leonard A. Stefanski. Simulation-extrapolation estimation in parametric measurement error models. *Journal of the American Statistical Association*, 89(428):1314–1328, 1994.

John C. Duchi and Hongseok Namkoong. Learning models with uniform performance via distributionally robust optimization. *The Annals of Statistics*, 49(3):1378–1406, 2021.

Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*, 2019.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*, volume 57 of *Monographs on Statistics and Applied Probability*. Chapman and Hall/CRC, Boca Raton, FL, 1994.

Benjamin Eysenbach and Sergey Levine. Maximum entropy RL (provably) solves some robust RL problems. In *International Conference on Learning Representations (ICLR)*, 2022.

Lin Fan and Peter W. Glynn. The typical behavior of bandit algorithms. *arXiv preprint arXiv:2210.05660*, 2022.

Lin Fan and Peter W Glynn. The fragility of optimized bandit algorithms. *Operations Research*, 73 (6):3173–3198, 2025.

Caroline A. Figueroa, Adrian Aguilera, Bibhas Chakraborty, Arghavan Modiri, Jai Aggarwal, Nina Deliu, Urmimala Sarkar, Joseph Jay Williams, and Courtney R. Lyles. Adaptive learning algorithms to optimize mobile applications for behavioral health: guidelines for design decisions. *Journal of the American Medical Informatics Association*, 28(6):1225–1234, 2021.

Tesshu Fujinami, Bruce D. Lee, Nikolai Matni, and George J. Pappas. Domain randomization is sample efficient for linear quadratic control. In *Proceedings of the 7th Annual Learning for Dynamics and Control Conference (L4DC)*, volume 283 of *Proceedings of Machine Learning Research*, pp. 907–919, 2025.

Asim H. Gazi, Daiqi Gao, Susobhan Ghosh, Ziping Xu, Anna L. Trella, Predrag Klasnja, and Susan A. Murphy. Digital twins for just-in-time adaptive interventions (JITAIs): Framework for optimizing and continually improving JITAIs. *Journal of Medical Internet Research*, 28:e72830, 2026. doi: 10.2196/72830.

Susobhan Ghosh, Yongyi Guo, Pei-Yao Hung, Lara Coughlin, Erin Bonar, Inbal Nahum-Shani, Maureen Walton, and Susan Murphy. reBandit: Random effects based online RL algorithm for reducing cannabis use. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*, pp. 7278–7286, 2024. doi: 10.24963/ijcai.2024/805.

Robert D. Gordon. Values of mills' ratio of area to bounding ordinate and of the normal probability integral for large values of the argument. *The Annals of Mathematical Statistics*, 12(3):364–366, 1941.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Julian Ibarz, Jie Tan, Chelsea Finn, Mrinal Kalakrishnan, Peter Pastor, and Sergey Levine. How to train your robot with deep reinforcement learning: lessons we have learned. *The International Journal of Robotics Research*, 40(4–5):698–721, 2021.

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

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, pp. 652–661. PMLR, 2016.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33: 21810–21823, 2020.

Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Ksenia Konyushova, Yutian Chen, Thomas Paine, Caglar Gulcehre, Cosmin Paduraru, Daniel J Mankowitz, Misha Denil, and Nando de Freitas. Active offline policy selection. *Advances in Neural Information Processing Systems*, 34:24631–24644, 2021.

Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020.

Chenhao Li, Andreas Krause, and Marco Hutter. Uncertainty-aware robotic world model makes offline model-based reinforcement learning work on real robots. *arXiv preprint arXiv:2504.16680*, 2025.

Lennart Ljung. *System Identification: Theory for the User*. Prentice Hall, 2nd edition, 1999.

Travis Mandel, Yun-En Liu, Emma Brunskill, and Zoran Popović. Offline evaluation of online reinforcement learning algorithms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J. Pal, and Liam Paull. Active domain randomization. In *Conference on Robot Learning*, pp. 1162–1176, 2020.

Thomas M. Moerland, Joost Broekens, Aske Plaat, and Catholijn M. Jonker. Model-based reinforcement learning: A survey. *Foundations and Trends in Machine Learning*, 16(1):1–118, 2023.

Fabio Muratore, Christian Eilers, Michael Gienger, and Jan Peters. Data-efficient domain randomization with Bayesian optimization. *IEEE Robotics and Automation Letters*, 6(2):911–918, 2021.

Fabio Muratore, Theo Gruner, Florian Wiese, Boris Belousov, Michael Gienger, and Jan Peters. Neural posterior domain randomization. In *Conference on Robot Learning (CoRL)*, volume 164 of *Proceedings of Machine Learning Research*, pp. 1532–1542, 2022a.

Fabio Muratore, Fabio Ramos, Greg Turk, Wenhao Yu, Michael Gienger, and Jan Peters. Robot learning from randomized simulations: A review. *Frontiers in Robotics and AI*, 9:799893, 2022b.

Yurii Nesterov and Boris T Polyak. Cubic regularization of newton method and its global performance. *Mathematical programming*, 108(1):177–205, 2006.

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

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. *Advances in neural information processing systems*, 29, 2016.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3803–3810. IEEE, 2018.

Doina Precup, Richard S. Sutton, and Satinder Singh. Eligibility traces for off-policy policy evaluation. In *International Conference on Machine Learning (ICML)*, pp. 759–766, 2000.

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

James Queeney, Erhan Can Ozcan, Ioannis Ch. Paschalidis, and Christos G. Cassandras. Optimal transport perturbations for safe reinforcement learning with robustness guarantees. *Transactions on Machine Learning Research*, 2024.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

Aravind Rajeswaran, Sarvjeet Ghotra, Balaraman Ravindran, and Sergey Levine. EPOpt: Learning robust neural network policies using model ensembles. In *International Conference on Learning Representations (ICLR)*, 2017.

Fabio Ramos, Rafael Carvalhaes Possas, and Dieter Fox. BayesSim: Adaptive domain randomization via probabilistic inference for robotics simulators. In *Robotics: Science and Systems (RSS)*, 2019.

Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. A tutorial on Thompson sampling. *Foundations and Trends in Machine Learning*, 11(1):1–96, 2018.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Elena Smirnova, Elvis Dohmatob, and Jérémie Mary. Distributionally robust reinforcement learning. *arXiv preprint arXiv:1902.08708*, 2019.

Shengpu Tang, Felipe Vieira Frujeri, Dipendra Misra, Alex Lamb, John Langford, Paul Mineiro, and Sebastian Kochman. Towards data-driven offline simulations for online reinforcement learning. *arXiv preprint arXiv:2211.07614*, 2022.

Gabriele Tiboni, Karol Arndt, and Ville Kyrki. DROPO: Sim-to-real transfer with offline domain randomization. *Robotics and Autonomous Systems*, 166:104432, 2023.

Gabriele Tiboni, Pascal Klink, Jan Peters, Tatiana Tommasi, Carlo D'Eramo, and Georgia Chalvatzaki. Domain randomization via entropy maximization. In *International Conference on Learning Representations*, 2024.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 23–30. IEEE, 2017.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5026–5033. IEEE, 2012.

Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U. Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Hannah Tan, and Omar G. Younis. Gymnasium: A standard interface for reinforcement learning environments. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025), Datasets and Benchmarks Track*, pp. 163114–163129, 2025.

Anna L Trella, Kelly W Zhang, Inbal Nahum-Shani, Vivek Shetty, Finale Doshi-Velez, and Susan A Murphy. Designing reinforcement learning algorithms for digital interventions: pre-implementation guidelines. *Algorithms*, 15(8):255, 2022.

Anna L. Trella, Kelly W. Zhang, Stephanie M. Carpenter, David Elashoff, Zara M. Greer, Inbal Nahum-Shani, Dennis Ruenger, Vivek Shetty, and Susan A. Murphy. Oralytics reinforcement learning algorithm. *arXiv preprint arXiv:2406.13127*, 2024.

Masatoshi Uehara, Chengchun Shi, and Nathan Kallus. A review of off-policy evaluation in reinforcement learning. *Statistical Science*, 41(3):561–581, 2026. doi: 10.1214/25-STS985.

Aad W. van der Vaart. *Asymptotic Statistics*, volume 3 of *Cambridge Series in Statistical and Probabilistic Mathematics*. Cambridge University Press, 1998.

Han Wang, Archit Sakhadeo, Adam White, James Bell, Vincent Liu, Xutong Zhao, Puer Liu, Tadashi Kozuno, Alona Fyshe, and Martha White. No more pesky hyperparameters: Offline hyperparameter tuning for RL. *Transactions on Machine Learning Research*, 2022.

Michael Wunder, Michael L. Littman, and Monica Babes. Classes of multiagent Q-learning dynamics with ϵ-greedy exploration. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pp. 1167–1174, 2010.

Yuqing Xie, Jiayu Chen, Wenhao Tang, Ya Zhang, Chao Yu, and Yu Wang. Automatic reward shaping from multi-objective human heuristics. *arXiv preprint arXiv:2512.15120*, 2025.

Ziping Xu, Yuyi Chang, Chenshun Ni, Nithin Sugavanam, Asim H. Gazi, Pedja Klasnja, Emre Ertin, and Susan A. Murphy. A diffusion-model subpopulation digital twin for mobile health deployment: A case study on the HeartSteps intervention. *arXiv preprint arXiv:2607.21403*, 2026. URL `https://arxiv.org/abs/2607.21403`.

Mengjiao Yang, Bo Dai, Ofir Nachum, George Tucker, and Dale Schuurmans. Offline policy selection under uncertainty. In *International Conference on Artificial Intelligence and Statistics*, pp. 4376–4396. PMLR, 2022.

Sherry Yang, Ofir Nachum, Yilun Du, Jason Wei, Pieter Abbeel, and Dale Schuurmans. Foundation models for decision making: Problems, methods, and opportunities. *arXiv preprint arXiv:2303.04129*, 2023.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y. Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*, volume 33, pp. 14129–14142, 2020.

Dongxia Zhang, Xiaoqing Han, and Chunyu Deng. Review on the research and practice of deep learning and reinforcement learning in smart grids. *CSEE Journal of Power and Energy Systems*, 4(3):362–370, 2018.

Joshua P Zitovsky, Daniel De Marchi, Rishabh Agarwal, and Michael Rene Kosorok. Revisiting Bellman errors for offline model selection. In *International conference on machine learning*, pp. 43369–43406. PMLR, 2023.