Survival Instinct in Offline Reinforcement Learning and Implicit Human Bias in Data

Anqi Li [1] **Dipendra Misra** [2] **Andrey Kolobov** [2] **Ching-An Cheng** [2]

Abstract

We present a novel observation about the behavior of offline reinforcement learning (RL) algorithms: on many benchmark datasets, offline RL can produce well-performing and safe policies even when trained with "wrong" reward labels, such as those that are zero everywhere or are negatives of the true rewards. This phenomenon cannot be easily explained by offline RL's return maximization objective. Moreover, it gives offline RL a degree of robustness that is uncharacteristic of its online RL counterparts, which are known to be sensitive to reward design. We demonstrate that this surprising robustness property is attributable to an interplay between the notion of *pessimism* in offline RL algorithms and certain human biases implicit in common data collection practices. As we prove in this work, pessimism endows the agent with a *survival instinct*, i.e., an incentive to stay within the data support in the long term, while the limited and biased data coverage further constrains the set of survival policies. We argue that the survival instinct should be taken into account when interpreting results from existing offline RL benchmarks and when creating future ones. Our empirical and theoretical results suggest a new paradigm for RL, whereby an agent is "nudged" to learn a desirable behavior with imperfect reward but purposely biased data coverage.

1. Introduction

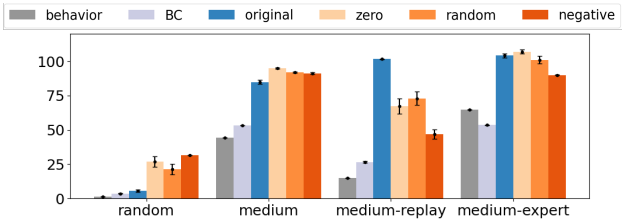
In offline reinforcement learning (RL), an agent optimizes its performance given an offline dataset. Despite being its main objective, we find that return maximization is not sufficient for explaining some of its empirical behaviors. In particular, *in many existing benchmark datasets, we observe that offline RL can produce surprisingly good policies even when trained on utterly wrong reward labels.*

In Figure 1, we present such results on the `hopper` task from D4RL (Fu et al., 2020), a popular offline RL benchmark, using a state-of-the-art offline RL algorithm ATAC (Cheng et al., 2022). The goal of an RL agent in the `hopper` task is to move forward as fast as possible while avoiding falling down. We trained ATAC agents on the original datasets and on three modified versions of each dataset, with "wrong" rewards: *1) zero*: assigning a zero reward to all transitions, *2) random*: labeling each transition with a reward sampled uniformly from $[0, 1]$, and *3) negative*: using the negation of the true reward. Although these wrong rewards contain no information about the underlying task or are even misleading, the policies learned from them in Figure 1 (left) often perform significantly better than the behavior (data collection) policy and the behavior cloning (BC) policy. They even outperform policies trained with the true reward (denoted as *original* in Figure 1) in some cases. This is puzzling, since RL is notorious for being sensitive to reward mis-specification (Leike et al., 2017; Hadfield-Menell et al., 2017; Ibarz et al., 2018; Pan et al., 2022): in general, maximizing the wrong rewards with RL leads to sub-optimal performance that can be worse than simply performing BC. In addition, these wrong-reward policies demonstrate a "safe" behavior, which keeps the `hopper` from falling down for a longer period than other comparators in Figure 1 (right). This is yet another peculiarity hard to link to return maximization, as none of the wrong rewards encourage the agent to stay alive. As we will show empirically in Section 4, these effects are not unique to ATAC or the `hopper` task. They occur with multiple offline RL algorithms, including ATAC (Cheng et al., 2022), PSPI (Xie et al., 2021), IQL (Kostrikov et al., 2021), CQL (Kumar et al., 2020) and the Decision Transformer (DT) (Chen et al., 2021), on dozens of datasets from D4RL (Fu et al., 2020) and Meta-World (Yu et al., 2020a) benchmarks.

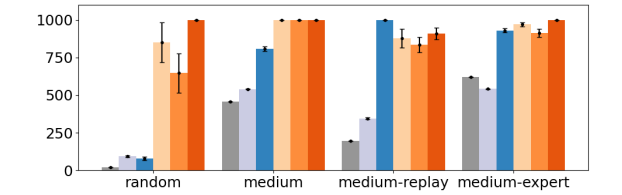
This robustness of offline RL is not only counter-intuitive but also cannot be explained fully by the literature. Offline RL theory (Cheng et al., 2022; Xie et al., 2021; Jin et al., 2021; Liu et al., 2020) provides performance guarantees only when the data reward is the true reward. Although offline imitation learning (IL) (Uehara and Sun, 2022; Chang

[1]University of Washington [2]Microsoft Research. Correspondence to: Anqi Li <anqil4@cs.washington.edu>.

Interactive Learning with Implicit Human Feedback Workshop at ICML 2023.



(a) Normalized scores



(b) Episode lengths

Figure 1: On the `hopper` task from D4RL (Fu et al., 2020), ATAC (Cheng et al., 2022), an offline RL algorithm, can produce high-performance and safe policies even when trained on wrong rewards.

et al., 2021; Ma et al., 2022b) makes no assumptions about reward, it only shows that the learned policy can achieve performance comparable to that of the behavior policy, not beyond. Robust offline RL (Zhang et al., 2022; Wu et al., 2023; Zhou et al., 2021; Ma et al., 2022a; Panaganti et al., 2022) shows that specialized algorithms can perform well when the size of data perturbation is small. In contrast, we demonstrate that standard offline RL algorithms can produce good policies even when we completely change the reward. Constrained offline RL algorithms (Le et al., 2019; Xu et al., 2022b; Lee et al., 2022a) can learn safe behaviors when constraint violations are both explicitly labeled and optimized. However, in the phenomena we observe, no safety signal is given to off-the-shelf, unconstrained offline RL algorithms. In Appendix A, we discuss the gap between the related work and our findings in more detail.

In this work, we provide an explanation for this seemingly surprising observation. We prove that this robustness property is attributed to the interplay between the use of pessimism in offline RL algorithms and the implicit bias in typical data collection processes. Offline RL algorithms often use pessimism to avoid taking actions that lead to unknown future events. We show that this risk-averse tendency bakes a *"survival instinct"* into the agent, an incentive to stay within the data coverage in the long term. On the other hand, the limited coverage of offline data further constrains the set of *survival policies* (policies that remain in the data support in the long term). When this set of survival policies correlates with policies that achieve high returns w.r.t. the true reward (as in the example in Figure 1), robust behavior emerges.

Our theoretical and empirical results have two important implications. First and foremost, offline RL has a survival instinct that leads to inherent robustness and safety properties that online RL does not have. Unlike online RL, offline RL is *doubly robust*: as long as the data reward is correct or the data has a positive implicit bias, a pessimistic offline RL agent can perform well. Moreover, offline RL is safe as long as the data only contains safe states; thus safe RL in the offline setup can be achieved without specialized algorithms. Second, because of the existence of the survival instinct, the data coverage has a profound impact on offline RL. While a large data coverage improves the best policy that can be learned by offline RL with the true reward, it can also make offline RL more sensitive to imperfect rewards. In other words, collecting a large set of diverse data might not be necessary or helpful (see Section 4). This goes against the common wisdom in the RL community that data should be as exploratory as possible (Munos, 2003; Chen and Jiang, 2019; Yarats et al., 2022).

We emphasize that survival instinct's implications should be taken into account when interpreting results on existing offline RL benchmarks and designing future ones. We should treat the evaluation of offline RL algorithms differently from online RL algorithms. We suggest evaluating the performance of an offline RL algorithm by training it with wrong rewards in addition to the true reward in order to ablate the performance due to return maximization from the effects of survival instinct and implicit data bias.

We believe that our findings shed new light on RL applicability and research. To practitioners, we demonstrate that offline RL does not always require the correct reward to succeed. This opens up the possibility of using offline RL in domains where obtain high-quality rewards is challenging. Research-wise, the existence of the survival instinct raises the question of how to design data collection or data filtering procedures that would help offline RL to leverage this instinct in order to improve RL's performance with incorrect or missing reward labels.

2. Background

The goal of offline RL is to solve an unknown Markov decision process (MDP) from offline data. Typically, an offline dataset is a collection of tuples, $\mathcal{D} := \{(s, a, r, s') | (s, a) \sim \mu(\cdot, \cdot), r = r(s, a), s' \sim P(\cdot|s, a)\}$, where r is the reward, P captures the MDP's transition dynamics, and μ denotes the state-action data distribution induced by some data collection process. Modern offline RL algorithms adopt pessimism to address the issue of policy learning when μ does not have the full state-action space as its support. The basic idea is to optimize for a performance lower bound that penalizes actions leading to out-of-support future states. Such a penalty can take the form of behavior regularization (Kostrikov et al., 2021; Fujimoto and Gu, 2021; Wu et al., 2019), negative bonuses to discourage visiting less

frequent state-action pairs (Jin et al., 2021; Rashidinejad et al., 2021; Yu et al., 2020b), pruning less frequent actions (Liu et al., 2020; Kidambi et al., 2020), adversarial training (Cheng et al., 2022; Xie et al., 2021; Uehara and Sun, 2022; Xie et al., 2022; Rigter et al., 2022) or value penalties in modified dynamic programming (Kumar et al., 2020; Yu et al., 2021).

We are interested in settings where the offline RL agent learns not from $\mathcal{D}$ itself but from its corrupted version $\tilde{\mathcal{D}}$ with a wrong reward $\tilde{r}$, i.e., $\tilde{\mathcal{D}} := \{(s,a,\tilde{r},s')|(s,a,s') \in \mathcal{D}, \tilde{r} = \tilde{r}(s,a) \in [-1,1]\}$. We assume $\tilde{r}$ is from a reward class $\tilde{\mathcal{R}}$ — which may not necessarily contain the true reward r — and we wish to explain why offline RL can learn good behaviors from the corrupted dataset $\tilde{\mathcal{D}}$ (e.g., as in Figure 1). To this end, we introduce notations and assumptions we will use in the paper.

Notation We focus on the setting of infinite-horizon discounted MDPs. We denote the task MDP that the agent aims to solve as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, P, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, and $\gamma \in [0,1)$ is the discount. Without loss of generality, we assume $r : \mathcal{S} \times \mathcal{A} \to [0,1]$. Let $\Delta(\mathcal{U})$ denote the set of probability distributions over a set $\mathcal{U}$. We denote a policy as $\pi : \mathcal{S} \to \Delta(\mathcal{A})$. For a reward function r, we define a policy π's state value function as $V_r^\pi(s) := \mathbb{E}_{\pi,P}[\sum_{t=0}^\infty \gamma^t r(s_t,a_t)|s_0 = s]$ and the state-action value function as $Q_r^\pi(s,a) := r(s,a) + \gamma \mathbb{E}_{s'\sim P(\cdot|s,a)}[V_r^\pi(s')]$. Solving MDP $\mathcal{M}$ requires learning a policy that maximizes the return at an initial state distribution $d_0 \in \Delta(\mathcal{S})$, that is, $\max_\pi \mathbb{E}_{s\sim d_0}[V_r^\pi(s)]$. We denote the optimal policy as π^* and the optimal value functions as V_r^* and Q_r^*. Given d_0, we define the average state-action visitation of a policy π as $d^\pi(s,a) := (1-\gamma)\mathbb{E}_{\pi,P}[\sum_{t=0}^\infty \gamma^t d_t^\pi(s,a)]$, where $d_t^\pi(s,a)$ denotes the probability of visiting (s,a) at time t when running π starting at an initial state sampled from d_0. Note that $(1-\gamma)\mathbb{E}_{s\sim d_0}[V_r^\pi(s)] = \mathbb{E}_{(s,a)\sim d^\pi}[r(s,a)]$. For a function $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, we use the shorthand $f(s,\pi) = \mathbb{E}_{a\sim\pi|s}[f(s,a)]$; similarly for $f : \mathcal{S} \to \mathbb{R}$, we write $f(p) = \mathbb{E}_{s\sim p}[f(s)]$, e.g., $V_r^\pi(d_0) = \mathbb{E}_{s\sim d_0}[V_r^\pi(s)]$. For a distribution p, we use $\text{supp}(p)$ to denote its support.

Assumption We make the typical assumption in offline RL that the data distribution μ assigns positive probabilities to these state-actions visited by running the optimal policy π^* starting from d_0.

Assumption 2.1 (Single-policy Concentrability)**.** We assume $\sup_{s\in\mathcal{S},a\in\mathcal{A}} \frac{d^{\pi^*}(s,a)}{\mu(s,a)} < \infty$.

This is a standard assumption in the offline RL literature, which in the worst case is a necessary condition to have no regret w.r.t. π^* (Zhan et al., 2022). There are generalized notions (Xie et al., 2021; 2022) which are weaker but requires other smoothness assumptions. We note that Assumption 2.1 does not assume that μ is the average state-action visitation frequency of a single behavior policy, nor does it assume μ has a full coverage of all states and actions or all policy distributions (Chen and Jiang, 2019; Foster et al., 2022).

3. Why Offline RL can Learn Right Behaviors from Wrong Rewards

We provide conditions under which offline RL's aforementioned robustness w.r.t. misspecified rewards emerges. Our main finding is summarized in the theorem below.

Theorem 3.1. *(Informal) Under Assumption 2.1 and certain regularity assumptions, if an offline RL algorithm Algo is set to be sufficiently pessimistic and the data distribution μ has a positive bias, for any data reward $\tilde{r} \in \tilde{\mathcal{R}}$, the policy $\hat{\pi}$ learned by Algo from the dataset $\tilde{\mathcal{D}}$ has performance guarantee*

$$V_r^{\pi^*}(d_0) - V_r^{\hat{\pi}}(d_0) \le O(\iota)$$

as well as safety guarantee

$$(1-\gamma)\sum_{t=0}^\infty \gamma^t \text{Prob}\left(\exists \tau \in [0,t], s_\tau \notin \textit{supp}(\mu)|\hat{\pi}\right) \le O(\iota)$$

for a small ι that decreases to zero as the degree of pessimism and dataset size increase.

Here is our key insight: because of pessimism, offline RL endows agents with a "survival instinct" — it implicitly solves a constrained MDP (CMDP) problem that enforces the policy to stay within the data support. When combined with a training data distribution that has a positive bias (e.g., all policies staying within data support are near-optimal) such a survival instinct results in robustness to reward misspecification. Overall, Theorem 3.1 has two important implications:

1. Offline RL is *doubly robust*: it can learn near optimal policies so long as either the reward label is correct or the data distribution is positively biased;
2. Offline RL is *intrinsically safe* when data are safe, regardless of reward labeling, without the need of explicitly modeling safety constraints.

In the following, we provide details and discussion of the statements above. The complete theoretical statements and proofs can be found in Appendix C.

3.1. Warm-up: Intuitions for Survival Instinct and Positive Data Bias

We first use a grid world example to build some intuitions. Figure 2 shows a goal-directed problem, where the true reward is $+1$ and -1 upon touching the key and the lava, respectively. The offline data is suboptimal and does not have full support. All data trajectories that touched the lava were stopped early, while others were allowed to continue until the end of the episode. The goal state (key) is an absorbing state, where the agent can stay beyond the episode length. We use the wrong rewards in Figure 1 to

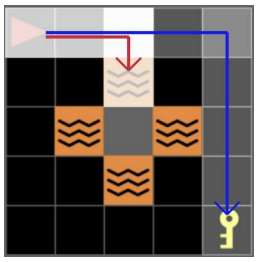


Figure 2: A grid world. BC (red); offline RL with wrong rewards (blue). The opacity indicates the frequency of a state in the data (more opaque means more frequent).

train PEVI (Jin et al., 2021), a finite-horizon, tabular offline RL method. PEVI performs dynamic programming similar to value iteration, but with a pessimistic value initialization and an instantaneous pessimism-inducing penalty of $O(-1/\sqrt{n(s,a)})$, where $n(s,a)$ is the empirical count in data. The penalties ensure that the learned value lower bounds the true one.

We see the offline RL agent learned with any wrong reward in Figure 1 is able to solve the problem despite data imperfection, while the BC agent that mimics the data directly fails. The main reasons are: *1)* There is a *data bias* whereby longer trajectories are closer to the goal, which we call a *length bias* (see Section 3.3). This positive data bias is due to bad trajectories (touching the lava or not reaching the goal) being cut short or timing out. *2)* Pessimism in PEVI gives the agent an *algorithmic bias* (survival instinct) that favors longer trajectories. Because of the pessimistic value initialization, PEVI treats trajectories shorter than the episode length as having the lowest return. As a result, by maximizing the pessimistically estimated values, PEVI learns good behaviors despite wrong rewards by leveraging the survival instinct and positive data bias *together*. We now make this claim more general.

3.2. Survival Instinct

The first key factor to the robustness of offline RL is a survival instinct originating from its pessimistic reasoning: offline RL algorithms tend to favor policies leading to in-support trajectories. Equivalently, offline RL implicitly solves a CMDP where the constraint enforces the agent's trajectories to stay within the data support, even when the algorithm does not explicitly model any constraints. We introduce some definitions to characterize this behavior more concretely.

Definition 3.2. Let $f, g : \mathcal{S} \times \mathcal{A} \to [-1, 1]$. A *CMDP* $\mathcal{C}(\mathcal{S}, \mathcal{A}, f, g, P, \gamma)$ is a constrained optimization problem: $\max_\pi V_f^\pi(d_0)$ s.t. $V_g^\pi(d_0) \leq 0$. Let $\pi^\dagger$ denote its optimal policy. For $\delta \geq 0$, we define the set of *δ-approximately optimal policies* $\Pi^\dagger(\delta) := \{\pi : V_f^{\pi^\dagger}(d_0) - V_f^\pi(d_0) \leq \delta, V_g^\pi(d_0) \leq \delta\}$.

We prove that offline RL trained on $\tilde{D}$, because of its pessimism, implicitly solves the CMDP:

$$\mathcal{C}_\mu(\tilde{r}) := \mathcal{C}(\mathcal{S}, \mathcal{A}, \tilde{r}, c_\mu, P, \gamma), \tag{1}$$

where $c_\mu(s,a) := \mathbb{1}[\mu(s,a) = 0]$ indicates whether s, a is out of the support of μ[1].

Proposition 3.3 (Survival Instinct)**.** *(Informal) Under certain regularity conditions on $\mathcal{C}_\mu(\tilde{r})$, the policy learned by an offline RL algorithm Algo with the offline dataset $\tilde{D}$ is ι-approximately optimal to $\mathcal{C}_\mu(\tilde{r})$, for some small ι which decreases as the algorithm becomes more pessimistic.*

Proposition 3.3 says if an offline RL algorithm is sufficiently pessimistic, then the learned policy is approximately optimal to $\mathcal{C}_\mu(\tilde{r})$. The learned policy has not only small regret with respect to the data reward $\tilde{r}$, but also small chances of escaping the support of the data distribution μ.

We highlight that the survival instinct in Proposition 3.3 is a *long-term* behavior. Such a survival behavior cannot be achieved by myopically staying within the action support of the data distribution in general (such as BC).[2] For instance, when some trajectories generating the data are truncated (e.g., due to early-stopping or intervention for safety reasons), taking in-support actions may still lead to out-of-support states in the future, as the BC agent in Section 3.1.

The key to prove Proposition 3.3 is to show that an offline RL algorithm by pessimism has small regret for not just $\tilde{r}$ but a set of reward functions consistent with $\tilde{r}$ on data but different outside of data, including the Lagrange reward of (1) (i.e., $\tilde{r} - \lambda c_\mu$, with $\lambda \geq 0$). We call this property admissibility and we show in Appendix E that many existing offline RL algorithms are admissible, including model-free algorithms ATAC (Cheng et al., 2022), VI-LCB (Rashidinejad et al., 2021) PPI/PQI (Liu et al., 2020), PSPI (Xie et al., 2021), as well as model-algorithms, ARMOR (Xie et al., 2022; Bhardwaj et al., 2023), MOPO (Yu et al., 2020b), MOReL (Kidambi et al., 2020), and CPPO (Uehara and Sun, 2022). By Proposition 3.3 these algorithms have survival instinct. Please see Appendix C for details.

3.3. Positive Data Bias

In addition to survival instinct, another key factor is an implicit positive human bias in common offline datasets. Typically these data manifest meaningful behaviors: For example, in collecting data for goal-oriented problems, data recording is normally stopped if the agent fails to reach the

[1]The use of an indicator function is not crucial here. We can extend the current analysis here to other costs that are zero in the support and strictly positive out of the support.

[2]BC requires a stronger condition to have survival instinct, such as when the data distribution μ is a (mixture of) average state-action visitation(s).

goal within certain time limit. Another example is problems where wrong decisions can have detrimental consequences, such as in robotics and healthcare scenarios (i.e. problems where safe RL focuses). In these domains, the data are collected by qualified policies only or under an intervention mechanism to prevent catastrophic failures.

Such a one-sided bias can creates an effect that staying within data support would lead to meaningful behaviors. Below we formally define the positive data bias.

Definition 3.4 (Positive Data Bias)**.** A distribution μ is $\frac{1}{\epsilon}$*-positively biased* w.r.t. a reward class $\tilde{\mathcal{R}}$ if

$$\max_{\tilde{r}\in\tilde{\mathcal{R}}} \max_{\pi\in\Pi^\dagger(\delta)} V_r^{\pi^*}(d_0) - V_r^{\pi}(d_0) \leq \epsilon + O(\delta) \qquad (2)$$

for all $\delta \geq 0$, where $\Pi^\dagger(\delta)$ denotes the set of δ-approximately optimal policy of $\mathcal{C}_\mu(\tilde{r})$.

We measure the degree of positive data bias based on how bad a policy can perform in terms of the true reward r when approximately solving the CMDP $\mathcal{C}_\mu(\tilde{r})$ in (1) defined with the wrong reward $\tilde{r} \in \tilde{\mathcal{R}}$. If the data distribution μ is ∞-positively biased, then any approximately optimal policy to $\mathcal{C}_\mu(\tilde{r})$ (which includes the policies learned by offline RL, as shown earlier) can achieve high return in the true reward r. We also can view the degree of positiveness as reflecting whether $\tilde{r}$ provides a similar ranking as r, *among policies within the support of* μ. When there is a positive bias, offline RL can learn with $\tilde{r}$ to perform well under r, even when $\tilde{r}$ is not aligned with r globally. This is contrast to online RL, which requires global alignment due to the exploratory nature of online RL.

Here we provide a few example data distributions that have a positive data bias. First, any policy is ∞-positively biased for any rewards resulting from potential-based reward shaping (Ng et al., 1999), since it provides the same ranking for all policies. Second, for a setup similar to IL, the data distribution μ is ∞-positively biased for any rewards $\tilde{r} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ if it is generated by the optimal policy, i.e., $\mu = d^{\pi^\star}$. Finally, a data distribution can have a positive bias when longer trajectories in the data have smaller optimality gap. This condition is typically satisfied when intervention is taken in the data collection process (despite the data collection policy being suboptimal), as in the motivating example in Figure 1. Later in Section 4, we will investigate deeper into this kind of length bias empirically. We formalize these conditions and provide more technical discussions in Appendix C.

Remark We highlight that the positive data bias assumption in Definition 3.4 is *different* from assuming that the data is collected by expert policies, which is typical in the IL literature. Positive data bias assumption can hold in cases when data is generated by highly suboptimal policies, which we observe in the `hopper` example from Section 1. On the other hand, there are also cases where IL can learn well, while positive data bias does not exist.

3.4. Summary: Offline RL is Doubly Robust and Intrinsically Safe

We have discussed the survival instinct from pessimism and the potential positive data bias in common datasets. Since survival instinct makes the policy of offline RL benefits from a favorable inductive bias of staying within the support of a positive data distribution, together they imply the robustness of offline RL (namely, Theorem 3.1).

We can view this phenomenon as a doubly robust property of offline RL. We borrow the name "doubly robust" from the offline policy *evaluation* literature (Dudík et al., 2011) to highlight the robustness of offline RL to reward misspecification as an offline policy *optimization* approach.

Corollary 3.5. *Under Assumption 2.1, offline RL can learn a near optimal policy, as long as the reward is correct,* or *the data has a positive implicit bias.*

Theorem 3.1 also implies that offline RL is an intrinsically safe learning algorithm, unlike its counterpart online RL where additional penalties or constraints need to be explicitly modelled (Achiam et al., 2017; Wachi and Sui, 2020; Wagener et al., 2021; Paternain et al., 2022; Nguyen and Cheng, 2023).

Corollary 3.6. *If μ only covers safe states and there exists a safe policy staying within the support of μ, then the policy of offline RL only visits safe states with high probability (see Theorem 3.1).*

We should note that covering safe states is a very mild assumption in the safe RL literature (Wagener et al., 2021; Hans et al., 2008), e.g., compared with all (data) states have a safe action. It does not require the data collection policies that generate μ are safe. This condition can be easily satisfied by filtering out unsafe states in post processing. The existence of a safe policy is also mild and common assumption.

4. Experiments

We empirically study the performance of offline RL algorithms under wrong rewards in a variety of tasks. We use the same set of wrong rewards as in Figure 1. *1)* zero: the zero reward, *2)* random: labeling each transition with a reward value randomly sampled from Unif$[0, 1]$, and *3)* negative: the negation of true reward. We consider five offline RL algorithms, ATAC (Cheng et al., 2022), PSPI (Xie et al., 2021), IQL (Kostrikov et al., 2021), CQL (Kumar et al., 2020) and decision transformer (DT)[3] (Chen et al., 2021). We deliber-

[3]We condition the decision transformer on the return of a trajectory sampled randomly from trajectories that achieve top 10% of returns in terms of data reward.

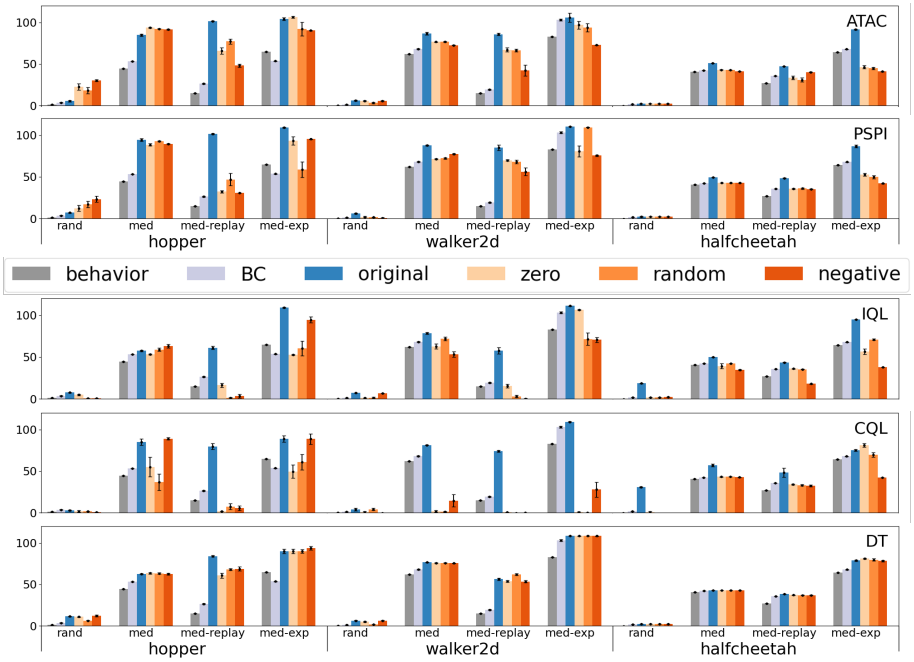


Figure 3: Normalized scores for locomotion tasks from D4RL. The mean and standard error for normalized scores are computed across 10 random seeds. For each random seed, we evaluate the final policy of each algorithm over 50 episodes.

ately choose offline RL algorithms to cover those that are provably pessimistic (Cheng et al., 2022; Xie et al., 2021) and those that are popular among practitioners (Kostrikov et al., 2021; Kumar et al., 2020), as well as an unconventional offline RL algorithm (Chen et al., 2021). We consider a variety of tasks from D4RL (Fu et al., 2020) and Meta-World (Yu et al., 2020a) ranging from safety-critical tasks (i.e., the agent dies when reaching bad states), goal-oriented tasks, and tasks that belong to neither. We train a total of around 16k offline RL agents (see Appendix B.8). Please see Appendix B for details.

Messages We would like to convey three main messages through our experiments. First, implicit data bias *can* exist naturally in a wide range of datasets, and offline RL algorithms that are sufficiently pessimistic can leverage such a bias to succeed when given wrong rewards. Second, offline RL algorithms, regardless of how pessimistic they are, become sensitive to reward when the data does not possess a positive bias. Third, offline RL algorithms without explicit pessimism, e.g., IQL (Kostrikov et al., 2021), can sometimes still be pessimistic enough to achieve good performance under wrong rewards.

Remark on negative results We consider "negative" results, i.e., when offline RL *fails* under wrong rewards, as important as the positive ones. Since they tell us *how* positive data bias can be broken or avoided. We hope our study can provide insights to researchers who are interested in actively incorporating positive bias in data collection, as well as who hope to design offline RL benchmarks specifically with or without positive data bias.

4.1. Locomotion Tasks from D4RL

We evaluate offline RL algorithms on three locomotion tasks, `hopper`, `walker2d`[4] and `halfcheetah`, from D4RL (Fu et al., 2020), a widely used offline RL benchmark. For each task, we consider four datasets with different qualites: random, medium, medium-replay, and medium-expert. We refer readers to (Fu et al., 2020) for the construction of these datasets. We measure policy performance in D4RL normalized scores (Fu et al., 2020). We provide three baselines 1) behavior: normalized score directly computed from the dataset, 2) BC: the behavior cloning policy, and 3) original: the policy produced by the offline RL algorithm using the original dataset (with true reward). The normalized scores for baselines and offline RL algorithms with wrong rewards are presented in Figure 3. The exact numbers can be found in tables in Appendix B.

Data bias exists in some D4RL datasets. We visualize the length bias of D4RL datasets in Figure 4. Here are our main observations. *1)* Most datasets for `hopper` and `walker2d`, with the exception of the medium-replay datasets, have a strong length bias, where longer trajectories have higher returns. This length bias is due to the safety-critical nature of `hopper` and `walker2d` tasks, as

[4]We remove terminal transitions from `hopper` and `walker2d` datasets when learning with wrong rewards. In Appendix B, we include a ablation study on the effect of removing terminals when using true reward.

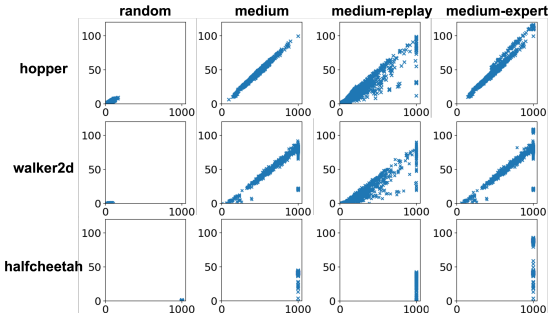


Figure 4: A visualization of length bias in datasets from D4RL (Fu et al., 2020). Each plot corresponds to a dataset for a task (row) with a dataset (column). Each trajectory in a dataset is represented as a data point with the x-coordinate being its episode length and y-coordinate being its normalized score.

the trajectories get terminated when reaching bad states. The length bias is especially salient in `hopper-medium`, where the normalized score is almost proportional to episode length. *2)* The medium-replay datasets for `hopper` and `walker2d` have more diverse behavior, so the bias is smaller. *3)* All `halfcheetah` datasets do not have length bias, as they all have the same length of 1000.

Offline RL can learn good policies with wrong rewards on datasets with strong length bias. On datasets with strong length bias (`hopper-random`, `hopper-medium` and `walker2d-medium`), we observe that ATAC and PSPI with wrong rewards generally produce well-performing policies, in a few cases even out-performing the policies learned from the true reward (original in Figure 4). DT is mostly insensitive to reward quality. IQL and CQL with wrong can sometimes achieve good performance; among the two, we find IQL to be more robust to wrong rewards and CQL with wrong rewards almost fails completely on all `walker2d` datasets.

Offline RL needs stronger reward signals on datasets with diverse behavior policy. The medium-replay datasets of `hopper` and `walker2d` are generated from multiple behavior policies. Due to their diverse nature, they are a multiple ways to stay within data support in a long term with various performance. As a result, the survival instinct of offline RL by itself is not sufficient to guarantee good performance. Generally, algorithms with wrong rewards under-perform the policies trained with true reward in the experiments. As datasets have a more diverse coverage, they can become less positively biased and offline RL requires a stronger reward signal to differentiate good survival policies and the bad ones. Practically speaking, when high-quality reward labeling is not available, a diverse dataset can even hurt offline RL performance. This is contrary to the common belief that larger data support is more preferable (Munos, 2003; Chen and Jiang, 2019; Yarats et al., 2022).

Offline RL requires good reward to perform well on datasets without length bias. In all four `halfcheetah` datasets, the trajectories all have the same length, as is demonstrated in Figure 4. This means that there is no data bias. We observe that all algorithms with wrong rewards at best perform similarly as the behavior and BC policies in most cases; this is an imitation-like effect due to the survival instinct. In the `halfcheetah-medium-expert` dataset, since there are a variety of surviving trajectories, with score from 0 to near 100, we observe that the performance of the resulting policy degrades as data reward becomes more different from the true reward.

Offline RL algorithms with provably pessimism produce safer policies. For `hopper` and `walker2d`, we observe that policies learned by ATAC and PSPI, regardless of the reward, can keep the agent from falling for longer. The episode lengths of IQL and CQL policies are often comparable to that of the behavior policies. We provide statistics of episode lengths in Appendix B.

Remark on benchmarking offline RL Our observations on the popular D4RL datasets raise an interesting question for benchmarking offline RL algorithms. A implicit positive data bias can give certain algorithms a hidden advantage, as they can already achieve good performance without using the reward. Without controlling data bias, it is hard to differentiate whether the performance of those algorithms are due to their ability to maximize returns or simply due the their survival instinct.

4.2. Goal-oriented Manipulation Tasks from Meta-World

We study goal-oriented manipulation tasks from the Meta-World benchmark (Yu et al., 2020a). We consider 15 goal-oriented tasks from Meta-World: the 10 tasks from the MT-10 set and the 5 testing tasks from the ML-45 set. For each task, we generate a dataset of 110 trajectories using the scripted policies provided by (Yu et al., 2020a); 10 are produced by the scripted policy, and 100 are generated by perturbing the scripted policy with a zero-mean Gaussian noise of standard deviation 0.5. In the datasets, unsuccessful trajectories receive a time-out signal after 500 steps (maximum episode length for Meta-World).

We measure the success rate of policies for 50 *new* goals unseen during training. Similar to D4RL experiments, we consider BC policies and policies trained with the true reward (original) as baselines. Since the training dataset is generated for a different set of goals, we do not compare the success rate of learned policies with the success rate in data.

Goal-oriented problems have data bias. Goal-oriented problems are fundamentally different from safety-critical tasks (such as `hopper` and `walker2d`). A good goal-oriented policy should reach the goal as fast as possible

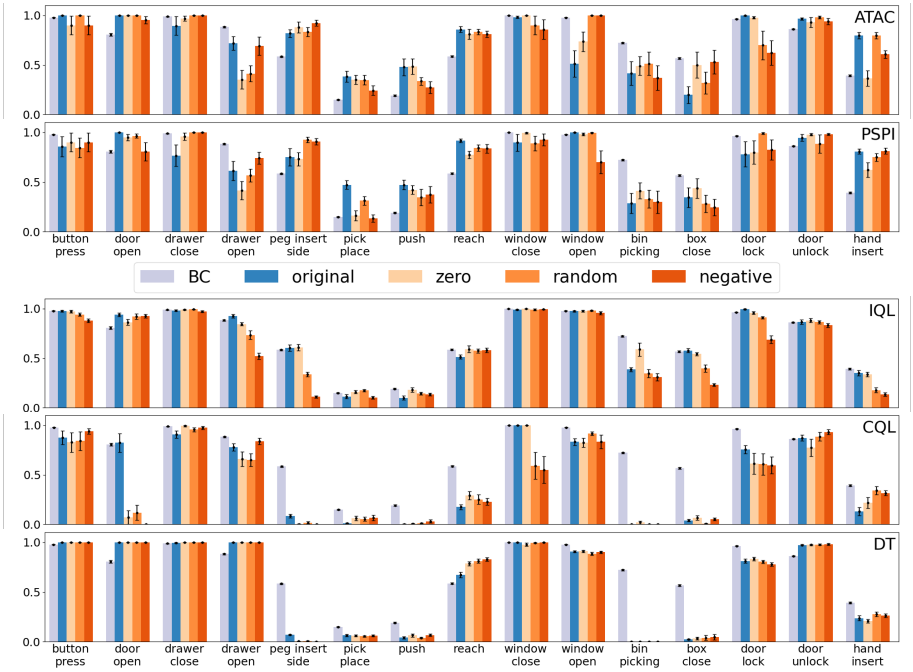


Figure 5: Success rate for goal-oriented tasks from Meta-World. The average success rate and confidence interval are computed across 10 random seeds. For each seed, we evaluate final policies for 50 episodes, each with a new goal unseen in the dataset.

rather than wandering around until the end of episode. Nevertheless, another form of length bias still exists here, because successful trajectories are labeled with a termination signal that indicates there is no timeouts and failed trajectories have a bounded length (see Section 3.1).

Offline RL can learn with wrong rewards on goal-oriented tasks. The success rate of learned policies with different rewards are shown in Figure 5. We observe that ATAC and PSPI with wrong rewards generally achieve comparable or better success rate than BC. The exceptions are often due to that ATAC or PSPI, even with the true reward, does not work as well as BC in those tasks, e.g. drawer-open, bin-picking and box-close. This could be caused by overfitting or optimization errors (due to challenges of dynamics programming over long horizon problems). In a number of tasks, such as peg-insert-side, push and reach, ATAC and PSPI with wrong rewards can out-perform BC by a margin. This shows that data collection for goal-oriented problems have a positive bias. This is remarkable as when learning with random and negative rewards, unsuccessful trajectories are long and generally have significantly higher return, as reward for each step is non-negative. The offline RL algorithms need to be sufficiently pessimistic and be able to plan over a long horizon to propagate pessimism to the unsuccessful data trajectories. For IQL, there is a gentle performance decrease as the reward becomes more different than the true reward, even though policies learned with wrong rewards are not much worse than BC in many cases. CQL shows robustness to reward in a few tasks such as button-press, drawer-close and door-unlock. Interestingly, DT performs almost uniformly across all rewards, potentially because DT does not explicitly maximize returns.

Remark on BC performance It is noticeable that BC policies in general achieve high performance, in many tasks often out-performing offline RL policies learned with true reward. This effect has also been observed in existing work on similar tasks (Zhou et al., 2022). We would like to clarify that the goal of our experiments is to study the behavior of offline RL algorithms when trained with wrong rewards, rather than showing that offline RL performs better than BC. We refer interested readers to existing studies (Zhou et al., 2022; Kumar et al., 2022) on when using offline RL algorithms is or is not preferable over BC.

5. Concluding Remarks

We present unexpected results on the robustness of offline RL to reward mis-specification. We show that this property originates from the interaction between the *survival instinct* of offline RL and hidden positive biases in common data collection processes. Our findings suggest extra considerations should be taken when interpreting offline RL results and designing future benchmarks. In addition, our findings open a new space for offline RL research: the possibility of designing algorithms that proactively leverage the survival instinct to learn policies for domains where rewards are nontrivial to specify or unavailable.

References

Achiam, J., Held, D., Tamar, A., and Abbeel, P. (2017). Constrained policy optimization. In *International conference on machine learning*, pages 22–31. PMLR.

Altman, E. (1999). *Constrained Markov decision processes*, volume 7. CRC press.

Bhardwaj, M., Xie, T., Boots, B., Jiang, N., and Cheng, C.-A. (2023). Adversarial model for offline reinforcement learning. *arXiv preprint arXiv:2302.11048*.

Chang, J., Uehara, M., Sreenivas, D., Kidambi, R., and Sun, W. (2021). Mitigating covariate shift in imitation learning via offline data with partial coverage. *Advances in Neural Information Processing Systems*, 34:965–979.

Chen, J. and Jiang, N. (2019). Information-theoretic considerations in batch reinforcement learning. In *International Conference on Machine Learning*, pages 1042–1051. PMLR.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. (2021). Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097.

Chen, Y., Zhang, X., Zhang, K., Wang, M., and Zhu, X. (2023). Byzantine-robust online and offline distributed reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 3230–3269. PMLR.

Cheng, C.-A., Xie, T., Jiang, N., and Agarwal, A. (2022). Adversarially trained actor critic for offline reinforcement learning. In *International Conference on Machine Learning*. PMLR.

Dai, B., Nachum, O., Chow, Y., Li, L., Szepesvári, C., and Schuurmans, D. (2020). Coindice: Off-policy confidence interval estimation. *Advances in neural information processing systems*, 33:9398–9411.

Dudík, M., Langford, J., and Li, L. (2011). Doubly robust policy evaluation and learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 1097–1104.

Foster, D. J., Krishnamurthy, A., Simchi-Levi, D., and Xu, Y. (2022). Offline reinforcement learning: Fundamental barriers for value function approximation. In *Conference on Learning Theory*, pages 3489–3489. PMLR.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. (2020). D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*.

Fujimoto, S. and Gu, S. S. (2021). A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145.

Hadfield-Menell, D., Milli, S., Abbeel, P., Russell, S. J., and Dragan, A. (2017). Inverse reward design. *Advances in neural information processing systems*, 30.

Hans, A., Schneegaß, D., Schäfer, A. M., and Udluft, S. (2008). Safe exploration for reinforcement learning. In *ESANN*, pages 143–148. Citeseer.

Hu, H., Yang, Y., Zhao, Q., and Zhang, C. (2023). The provable benefits of unsupervised data sharing for offline reinforcement learning. In *International Conference on Learning Representations*.

Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., and Amodei, D. (2018). Reward learning from human preferences and demonstrations in atari. *Advances in neural information processing systems*, 31.

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

Kidambi, R., Chang, J., and Sun, W. (2021). Mobile: Model-based imitation learning from observation alone. *Advances in Neural Information Processing Systems*, 34:28598–28611.

Kidambi, R., Rajeswaran, A., Netrapalli, P., and Joachims, T. (2020). Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33:21810–21823.

Kim, G.-H., Seo, S., Lee, J., Jeon, W., Hwang, H., Yang, H., and Kim, K.-E. (2021). Demodice: Offline imitation learning with supplementary imperfect demonstrations. In *International Conference on Learning Representations*.

Kostrikov, I., Nair, A., and Levine, S. (2021). Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*.

Kumar, A., Hong, J., Singh, A., and Levine, S. (2022). Should i run offline reinforcement learning or behavioral cloning? In *International Conference on Learning Representations*.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. (2020). Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191.

Le, H., Voloshin, C., and Yue, Y. (2019). Batch policy learning under constraints. In *International Conference on Machine Learning*, pages 3703–3712. PMLR.

Lee, J., Paduraru, C., Mankowitz, D. J., Heess, N., Precup, D., Kim, K.-E., and Guez, A. (2022a). Coptidice: Offline constrained reinforcement learning via stationary distribution correction estimation. In *International Conference on Learning Representations*.

Lee, S., Seo, Y., Lee, K., Abbeel, P., and Shin, J. (2022b). Offline-to-online reinforcement learning via balanced replay and pessimistic q-ensemble. In *Conference on Robot Learning*, pages 1702–1712. PMLR.

Leike, J., Martic, M., Krakovna, V., Ortega, P. A., Everitt, T., Lefrancq, A., Orseau, L., and Legg, S. (2017). Ai safety gridworlds. *arXiv preprint arXiv:1711.09883*.

Li, A., Boots, B., and Cheng, C.-A. (2023a). Mahalo: Unifying offline reinforcement learning and imitation learning from observations. *arXiv preprint arXiv:2303.17156*.

Li, J., Hu, X., Xu, H., Liu, J., Zhan, X., Jia, Q.-S., and Zhang, Y.-Q. (2023b). Mind the gap: Offline policy optimization for imperfect rewards. In *International Conference on Learning Representations*.

Liu, Y., Swaminathan, A., Agarwal, A., and Brunskill, E. (2020). Provably good batch off-policy reinforcement learning without great exploration. *Advances in neural information processing systems*, 33:1264–1274.

Liu, Z., Guo, Z., Yao, Y., Cen, Z., Yu, W., Zhang, T., and Zhao, D. (2023). Constrained decision transformer for offline safe reinforcement learning. *arXiv preprint arXiv:2302.07351*.

Ma, X., Liang, Z., Xia, L., Zhang, J., Blanchet, J., Liu, M., Zhao, Q., and Zhou, Z. (2022a). Distributionally robust offline reinforcement learning with linear function approximation. *arXiv preprint arXiv:2209.06620*.

Ma, Y., Shen, A., Jayaraman, D., and Bastani, O. (2022b). Versatile offline imitation from observations and examples via regularized state-occupancy matching. In *International Conference on Machine Learning*, pages 14639–14663. PMLR.

Munos, R. (2003). Error bounds for approximate policy iteration. In *International Conference on Machine Learning*, volume 3, pages 560–567. Citeseer.

Ng, A. Y., Harada, D., and Russell, S. (1999). Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning*, volume 99, pages 278–287. Citeseer.

Nguyen, H.-A. and Cheng, C.-A. (2023). Provable reset-free reinforcement learning by no-regret reduction. *arXiv preprint arXiv:2301.02389*.

Pan, A., Bhatia, K., and Steinhardt, J. (2022). The effects of reward misspecification: Mapping and mitigating misaligned models. In *International Conference on Learning Representations*.

Panaganti, K., Xu, Z., Kalathil, D., and Ghavamzadeh, M. (2022). Robust reinforcement learning using offline data. In *Advances in Neural Information Processing Systems*.

Paternain, S., Calvo-Fullana, M., Chamon, L. F., and Ribeiro, A. (2022). Safe policies for reinforcement learning via primal-dual methods. *IEEE Transactions on Automatic Control*.

Polosky, N., Da Silva, B. C., Fiterau, M., and Jagannath, J. (2022). Constrained offline policy optimization. In *International Conference on Machine Learning*, pages 17801–17810. PMLR.

Rashidinejad, P., Zhu, B., Ma, C., Jiao, J., and Russell, S. (2021). Bridging offline reinforcement learning and imitation learning: A tale of pessimism. *Advances in Neural Information Processing Systems*, 34:11702–11716.

Riedmiller, M. (2005). Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings 16*, pages 317–328. Springer.

Rigter, M., Lacerda, B., and Hawes, N. (2022). Rambo-RL: Robust adversarial model-based offline reinforcement learning. *Advances in neural information processing systems*.

Shi, L. and Chi, Y. (2022). Distributionally robust model-based offline reinforcement learning with near-optimal sample complexity. *arXiv preprint arXiv:2208.05767*.

Singh, A., Yu, A., Yang, J., Zhang, J., Kumar, A., and Levine, S. (2020). Cog: Connecting new skills to past experience with offline reinforcement learning. In *Conference on Robot Learning*. PMLR.

Smith, M., Maystre, L., Dai, Z., and Ciosek, K. (2023). A strong baseline for batch imitation learning. *arXiv preprint arXiv:2302.02788*.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Uehara, M. and Sun, W. (2022). Pessimistic model-based offline reinforcement learning under partial coverage. In *International Conference on Learning Representations*.

Van Hasselt, H., Doron, Y., Strub, F., Hessel, M., Sonnerat, N., and Modayil, J. (2018). Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*.

Wachi, A. and Sui, Y. (2020). Safe reinforcement learning in constrained markov decision processes. In *International Conference on Machine Learning*, pages 9797–9806. PMLR.

Wagener, N. C., Boots, B., and Cheng, C.-A. (2021). Safe reinforcement learning using advantage-based intervention. In *International Conference on Machine Learning*, pages 10630–10640. PMLR.

Wu, F., Li, L., Zhang, H., Kailkhura, B., Kenthapadi, K., Zhao, D., and Li, B. (2023). Copa: Certifying robust policies for offline reinforcement learning against poisoning attacks. In *International Conference on Learning Representations*.

Wu, Y., Tucker, G., and Nachum, O. (2019). Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*.

Xie, T., Bhardwaj, M., Jiang, N., and Cheng, C.-A. (2022). Armor: A model-based framework for improving arbitrary baseline policies with offline data. *arXiv preprint arXiv:2211.04538*.

Xie, T., Cheng, C.-A., Jiang, N., Mineiro, P., and Agarwal, A. (2021). Bellman-consistent pessimism for offline reinforcement learning. *Advances in neural information processing systems*, 34:6683–6694.

Xu, H., Zhan, X., Yin, H., and Qin, H. (2022a). Discriminator-weighted offline imitation learning from suboptimal demonstrations. In *International Conference on Machine Learning*, pages 24725–24742. PMLR.

Xu, H., Zhan, X., and Zhu, X. (2022b). Constraints penalized q-learning for safe offline reinforcement learning. In *AAAI*.

Xu, W., Ma, Y., Xu, K., Bastani, H., and Bastani, O. (2023). Uniformly conservative exploration in reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 10856–10870. PMLR.

Yang, R., Bai, C., Ma, X., Wang, Z., Zhang, C., and Han, L. (2022). Rorl: Robust offline reinforcement learning via conservative smoothing. In *Advances in Neural Information Processing Systems*.

Yarats, D., Brandfonbrener, D., Liu, H., Laskin, M., Abbeel, P., Lazaric, A., and Pinto, L. (2022). Don't change the algorithm, change the data: Exploratory data for offline reinforcement learning. *arXiv preprint arXiv:2201.13425*.

Yu, T., Kumar, A., Chebotar, Y., Hausman, K., Finn, C., and Levine, S. (2022). How to leverage unlabeled data in offline reinforcement learning. In *International Conference on Machine Learning*, pages 25611–25635. PMLR.

Yu, T., Kumar, A., Rafailov, R., Rajeswaran, A., Levine, S., and Finn, C. (2021). Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. (2020a). Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR.

Yu, T., Thomas, G., Yu, L., Ermon, S., Zou, J. Y., Levine, S., Finn, C., and Ma, T. (2020b). Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142.

Yu, W., Liu, C. K., and Turk, G. (2019). Policy transfer with strategy optimization. In *International Conference on Learning Representations*.

Yue, S., Wang, G., Shao, W., Zhang, Z., Lin, S., Ren, J., and Zhang, J. (2023). CLARE: Conservative model-based reward learning for offline inverse reinforcement learning. In *International Conference on Learning Representations*.

Zhan, W., Huang, B., Huang, A., Jiang, N., and Lee, J. (2022). Offline reinforcement learning with realizability and single-policy concentrability. In *Conference on Learning Theory*, pages 2730–2775. PMLR.

Zhang, X., Chen, Y., Zhu, X., and Sun, W. (2022). Corruption-robust offline reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 5757–5773. PMLR.

Zhou, G., Ke, L., Srinivasa, S., Gupta, A., Rajeswaran, A., and Kumar, V. (2022). Real world offline reinforcement learning with realistic data source. *arXiv preprint arXiv:2210.06479*.

Zhou, Z., Zhou, Z., Bai, Q., Qiu, L., Blanchet, J., and Glynn, P. (2021). Finite-sample regret bound for distributionally robust offline tabular reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 3331–3339. PMLR.

Zhu, Z., Lin, K., Dai, B., and Zhou, J. (2020). Off-policy imitation learning from observations. *Advances in Neural Information Processing Systems*, 33:12402–12413.

Ziebart, B. D., Maas, A. L., Bagnell, J. A., Dey, A. K., et al. (2008). Maximum entropy inverse reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 8, pages 1433–1438. Chicago, IL, USA.

Zolna, K., Novikov, A., Konyushkova, K., Gulcehre, C., Wang, Z., Aytar, Y., Denil, M., de Freitas, N., and Reed, S. (2020). Offline learning from demonstrations and unlabeled experience. *arXiv preprint arXiv:2011.13885*.

A. Related Work

Offline RL Offline RL studies the problem of return maximization given an offline dataset. Offline RL algorithms can be broadly classified into model-based approaches, e.g., (Yu et al., 2020b; Kidambi et al., 2020; Uehara and Sun, 2022; Rigter et al., 2022; Xie et al., 2022), and model-free approaches, e.g., (Jin et al., 2021; Fujimoto and Gu, 2021; Xie et al., 2021; Cheng et al., 2022; Kumar et al., 2020; Kostrikov et al., 2021). Since the agent needs to learn without online data collection, offline RL becomes more challenging when the offline dataset does not provide good coverage over the state-action distribution of all feasible policies. There have been two common strategies to handle insufficient coverage, behavior regularization approaches, e.g., (Fujimoto and Gu, 2021; Wu et al., 2019; Kostrikov et al., 2021), which restricts the learned policy to be close to the behavior policy, and value regularization approaches, e.g., (Jin et al., 2021; Liu et al., 2020; Xie et al., 2021; Cheng et al., 2022; Kumar et al., 2020; Xie et al., 2022) which provide a pessimistic value estimates for the learned policy. Both approaches can be viewed as optimizing for a performance lower bound of the agent's return.

Offline RL commonly assumes compatibility between offline dataset and MDP, i.e., the offline dataset should be generated from the task MDP of interest. Our work is distinctive in the offline RL literature as we study the behavior of existing offline RL algorithms when data reward *differs* from the true MDP reward. Existing tools for analyzing offline RL are not directly applicable to our setting: policies with good performance under data reward, as guaranteed by such tools, do not necessarily attain high return under the true reward. Our novel analysis is made possible by studying the properties of offline data distribution, an aspect mostly neglected by existing work in the literature.

Offline RL with imperfect reward In the offline RL literature, there is a line of work studying scenarios where the reward is missing in a subset of the dataset (Singh et al., 2020; Hu et al., 2023; Li et al., 2023a; Yu et al., 2022) or when the data reward is misspecified (Li et al., 2023b). These approaches propose to construct a new reward function and apply offline RL algorithms on the relabeled dataset. For example, (Singh et al., 2020; Yu et al., 2022) label the missing reward with a constant minimum reward. (Li et al., 2023a; Hu et al., 2023) learn a pessimistic reward function. (Li et al., 2023b) optimize for a reward function which minimizes the reward difference among expert data.

Different from this line of work, we are interested in running offline RL algorithms using the misspecified reward *as it is*. We show both theoretically and empirically that offline RL algorithms can produce well-performing policies with rewards different from the true reward as long as the underlying data distribution satisfies certain conditions. It can be an interesting future work to combine these techniques with our analysis to explore the robustness of offline RL algorithms under the class of learned reward functions.

Offline IL Offline imitation learning (IL) can be considered as a special case of offline RL when reward is missing from the entire dataset, or when the agent has the largest uncertainty about the reward (Cheng et al., 2022). In offline IL, the learner is instead given the information on whether a transition is generated by an expert. The goal of offline IL is to produce a policy that has comparable performance with the expert. (Zolna et al., 2020) learns a discriminator reward to classify expert and non-expert data, and apply offline RL algorithms with the learned reward. (Xu et al., 2022a) uses the discriminator as the weight for weighted behavior cloning. (Kim et al., 2021; Ma et al., 2022b; Zhu et al., 2020) proposes DICE (Dai et al., 2020)-style algorithms to minimize the divergence between the state-action or state-(next state) occupancy measure of the learner and the expert. (Chang et al., 2021; Kidambi et al., 2021) minimize state-action or state-(next state) distribution divergence under a pessimistic model. (Yue et al., 2023) extends maximum entropy inverse RL (Ziebart et al., 2008) to an offline setting. (Smith et al., 2023) proposes to run offline RL under an indicator reward function on whether the data is expert data. (Li et al., 2023a) learns a pessimistic reward function where the expert data labeled as the maximum reward.

Our experiments with zero reward function is similar to an offline IL setting with the behavior policy being the expert. However, we show that offline RL algorithms can sometimes achieve significant performance gain over behavior and behavior cloning policies, which is a phenomenon that cannot be explained by existing analysis in offline IL. We provide an explanation to this puzzle: in Appendix C.4, we show that the data distribution is ∞-positively biased (according our definition) for offline RL, when the data is generated by the optimal policy of the true MDP. This means that an admissible offline RL algorithm can achieve near-optimal performance with *any* reward function from its corresponding admissible reward class in this case. We empirically show that in Appendix B.6.

Robust offline RL Robust offline RL (Zhang et al., 2022; Chen et al., 2023; Wu et al., 2023; Panaganti et al., 2022; Yang et al., 2022; Zhou et al., 2021; Ma et al., 2022a; Shi and Chi, 2022) aims to develop new offline RL algorithms for cases when the compatibility assumption does not hold, i.e., the offline dataset may not be generated from the task MDP. Among

this line of work, (Panaganti et al., 2022; Zhou et al., 2021; Ma et al., 2022a; Shi and Chi, 2022) study the robust MDP setting, where the dataset can be generated from any MDP within a δ-ball of a nominal MDP. An adversary can choose any MDP from the δ-ball when evaluating the learned policy. (Zhang et al., 2022) considers when the adversary can modify ϵ fraction of transition tuples in the offline dataset, while (Wu et al., 2023) considers when up to K trajectories can be added in or removed from the original dataset. (Chen et al., 2023) uses the formulation of Byzantine-robust: when multiple offline datasets are presented to the learner, among them an α fraction are corrupted. (Yang et al., 2022) assumes the dataset is generated from the true MDP, but the state presented to the policy during test time can be a corrupted up to ϵ distance.

Our work is fundamentally different from this line of work in that we study the *inherent* and somewhat *unintended* robustness of offline RL algorithms. We show that, despite originally designed under the compatibility assumption, offline RL algorithms show strong robustness against perturbation of reward. Another salient difference is that robust offline RL typically requires an upper bound on the size of perturbation to the true or nominal MDP, and most of proposed algorithms assume knowledge of this upper bound. However, our work does not make assumptions on the size of perturbation and, moreover, the algorithms we study are not even aware of the existence of such perturbation. Indeed, we empirically show that offline RL algorithms, without any modifications, can succeed even when data reward is a constant or the negation of the true reward. Additionally, in this work, we consider the reward perturbation while most work in robust offline RL focuses more on dynamics perturbation.

Constrained and safe offline RL Our work is related to the literature of constrained and safe offline RL (Le et al., 2019; Xu et al., 2022b; Liu et al., 2023; Lee et al., 2022a; Polosky et al., 2022). In constrained offline RL, the agent solves for a CMDP given an offline dataset $\{(s, a, r, c, s')\}$. The goal of constrained offline RL is to produce a policy which maximizes the expected return while ensuring that constraint violation is below a given upper bound. (Le et al., 2019) uses an offline policy evaluation oracle to solve for the Lagrangian relaxation of the CMDP. (Xu et al., 2022b) constructs a pessimistic estimation of value and constraint violation. (Lee et al., 2022a; Polosky et al., 2022) proposes DICE (Dai et al., 2020)-style algorithms to solve for the optimal state-action occupancy measure. (Liu et al., 2023) proposes a variant of decision transformer (Chen et al., 2021) capable of producing policies for different constraint violation upper bounds during test time.

Our work, instead, considers a different scenario where the constraint is only given *implicitly* — through the support of the offline dataset. We assume that any transition in the dataset satisfies the constraint, which can be achieved through intervention during data collection. We show that offline RL can inherently produce approximately safe policies due to its survival instinct. Although our assumption seems more restrictive as in that we require a constraint violation-free dataset, we argue that our assumption is more practical in many safety critical scenarios: compared to executing unsafe actions, it is perhaps more reasonable to intervene before constraint violation happens. We believe that our findings open up new possibilities for applying offline RL as a safe RL oracle, which can be especially promising in an offline-online RL setup (Xu et al., 2023; Lee et al., 2022b).

B. Experiment Details

In the experiments, we train a total of around 16k offline RL agents (see Appendix B.8 for details). In this appendix, we provide details of experiment setup, implementation of algorithms, hyperparameter tuning, as well as additional experimental results and ablation studies. At the end of the appendix, we include Table 13 and Table 14 which contain the actual numbers used in generating Figure 3 and Figure 5.

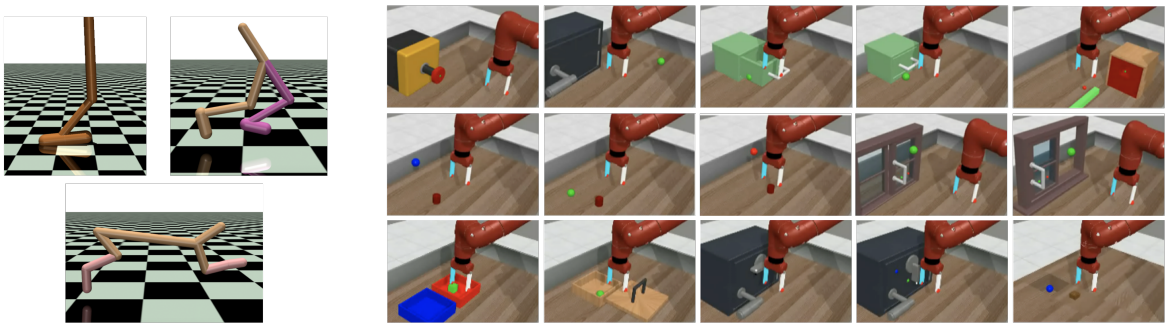
B.1. Benchmark Datasets

In the experiments, we consider three tasks (15 datasets[5]) from D4RL (Fu et al., 2020) and 15 datasets collected from the Meta-World (Yu et al., 2020a) benchmark. The tasks are visualized in Figure 6. Below we provide more details on these tasks.

D4RL We consider three locomotion tasks from D4RL (Fu et al., 2020), `hopper`, `walker2d` and `halfcheetah`[6] (see Figure 6). In the main paper, for each task, we use four datasets of different behavior policy quality: `random`, `medium`, `medium-replay`, and `medium expert`. We use the `expert` datasets in Appendix B.6 to study the robustness of

[5]We use 12 datasets (without `expert` datasets) in the main experiments.

[6]We use the v2 version of all three tasks.



(a) D4RL (Fu et al., 2020) (b) Meta-World (Yu et al., 2020a)

Figure 6: We study offline RL with wrong rewards for a variety of tasks: (a) three locomotion tasks from D4RL (Fu et al., 2020); hopper (top left), walker2d (top right) and halfcheetah (bottom). the figures are from (Yu et al., 2019) (b) 15 goal-oriented tasks from Meta-World (Yu et al., 2020a). The 15 tasks are composed of the 10 tasks from MT-10 (top 2 rows) and the 5 testing tasks from ML-45 (bottom row).

offline RL on data generated by near-optimal policies.

hopper: The goal of `hopper` is to move forward as fast as possible without falling down. An episode terminates when either of the three happens: *1)* the height of the `hopper` is below a threshold, *2)* the body of the `hopper` is too tilted, or *3)* the velocity of any joint is too large. The original true reward function (which has been used to train online RL agents) is

$$r_{\text{hopper}} := \mathbb{1}[\texttt{hopper} \text{ is alive}] + \frac{1}{\Delta t}\Delta x - 0.001\|\tau\|^2,$$

where Δx is the displacement in the forward direction, and τ is the applied torque. The true reward function is designed such that the `hopper` can receive bonus when staying alive and moving forward. We show that, for a few `hopper` datasets, offline RL agent can succeed without such a carefully designed reward, and even with the negation of the true reward.

walker2d: The `walker2d` task is similar to `hopper`, except that the `walker2d` agent has a higher degree of freedom. We observe similarly that offline RL can learn good behaviors without a reward that is specifically designed to encourage staying alive and moving forward.

halfcheetah: The `halfcheetah` task differs from the two tasks above since episodes here always have the same length and never terminates early. The true reward of `halfcheetah` is

$$r_{\text{halfcheetah}} := \frac{1}{\Delta t}\Delta x - 0.1\|\tau\|^2.$$

Since the `halfcheetah` datasets (except for the `expert` dataset, see Appendix B.6) do not have a positive bias, we find that offline RL agent is sensitive to reward quality.

Meta-World We consider 15 goal-oriented manipulation tasks from (Yu et al., 2020a), see Figure 6. We use the 10 tasks from the MT-10 set and the 5 testing tasks from the ML-45 set. Each Meta-World task comes with a hand-crafted reward function that encourage progress toward task completion, e.g., when the end-effector grasps the target object, when the object is placed at the goal, etc. We refer readers to (Yu et al., 2020a) for the specific reward functions used for each tasks. We make a few modifications to the Meta-World environment: *1)* we shift and scale the true reward such that the agent receives a reward of range $[-1, 0]$ rather than $[0, 10]$ of the original environment, *2)* we terminate an episode and mark the last state with a terminal flag when the goal is achieved. We show that, data collected by goal-oriented tasks can have a positive bias, and an offline RL agent can make use of such bias to learn the right behavior even with wrong rewards. We would like to clarify that the data generated by the original Meta-World environment (with no terminal signals) does not have a positive bias in general, since all trajectories, success or failure, would have a same length of 500.

B.2. Implementation Details

We use the implementation of ATAC and PSPI from https://github.com/chinganc/lightATAC. We make a small modification to the ATAC and PSPI policy class. Instead of a $\tanh$-Gaussian policy, we use a scaled $\tanh$-Gaussian policy, i.e., $c \cdot \tanh(\mathcal{N}(\mu, \Sigma))$ with $c \geq 1$. We find scaled $\tanh$-Gaussian policies to be more numerically stable when the data has a lot of on-boundary actions, i.e., actions that takes the value of either -1 or 1 on any dimension. For PSPI, we let the critic f to minimize for $f(s, \pi)$ for all states from the dataset (rather than all initial states). We find it to provide better results when the effective horizon given by the discount factor is smaller than the actual episode length. We use the implementation of IQL and BC from https://github.com/gwthomas/IQL-PyTorch/, and the implementation of CQL from https://github.com/young-geng/CQL. For decision transformer, we use the implementation from https://github.com/kzl/decision-transformer. We feed the decision transformer with wrong rewards (consistent with data reward) during testing.

B.3. Hyperparameters

We use the default values (ones provided by the original paper or by the implementation we use) for most of the hyperparameters for all algorithms. For each algorithm, we tune one or two hyperparameters (with 4 combinations at most) which affect the degree of pessimism[7]. The values of hyperparamters of all algorithms are given in Table 1–4, where the choices of hyperparameters used in tuning are highlighted in blue.

We tune hyperparameter per each dataset and per reward label. The tuned values are provided in Table 9–12. During tuning, we evaluate each combination of hyperparameter(s) for 2 random seeds for D4RL and 3 random seeds for Meta-World. We choose the best-performing values and report the results of these hyperparameters over 10 *new* random seeds (not including the tuning seeds). We find the D4RL scores for the true reward to be close to what is reported in the original papers.

Hyperparameter	**Value**
Bellman consistency coefficient β	$\{0.1, 1, 10, 100\}$
Number of hidden layers	3
Number of hidden units	256
Nonlinearity	ReLU
Policy distribution	scaled $\tanh$-Gaussian
Action scale c	1.0 for D4RL 1.2 for Meta-World
Fast learning rate η_{fast}	5×10^{-4}
Slow learning rate η_{slow}	5×10^{-7}
Minibatch size $\lvert\mathcal{D}_{\text{mini}}\rvert$	256
Gradient steps	10^6
Warmstart (BC) steps	10^5
Temporal difference loss weight w	0.5
Target smoothing coefficient τ	0.005
Discount factor γ	0.99 for D4RL 0.998 for Meta-World
Minimum target value V_{min}	$\frac{2}{1-\gamma}\min(-1, \tilde{r}_{\text{min}})$
Maximum target value V_{max}	$\frac{2}{1-\gamma}\max(1, \tilde{r}_{\text{max}})$

Table 1: Hyperparameters for ATAC and PSPI

B.4. Episode Lengths for `hopper` and `walker2d` Tasks

We show the episode lengths for `hopper` and `walker2d` tasks in Figure 7. We observe that policies learned by ATAC and PSPI can consistently keep the agent from falling down for a significant longer period than the behavior policies. The only exception is when running PSPI on `hopper-medium-expert` dataset with random reward. This is potentially due to the fact that we use only 2 seeds for hyperparameter tuning. The average normalized scores and episode lengths during tuning are 101.1 and 925.0, respectively, higher than the final values of 58.8 and 546.0. We observe that DT policies also tend to keep the agent alive for a longer number of steps than behavior policies.

For IQL and CQL, it is relatively rare for them to achieve long episode lengths when trained with wrong rewards.

[7]For decision transformer, we do not tune hyperparameters since it has no notion of pessimism.

Hyperparameter	Value
Inverse temperature β	$\{0.3, 3\}$
Expectile τ	$\{0.7, 0.9\}$
Number of hidden layers	3
Number of hidden units	256
Nonlinearity	ReLU
Policy distribution	Gaussian
Learning rate η	3×10^{-4}
Minibatch size $\lvert\mathcal{D}_{\text{mini}}\rvert$	256
Gradient steps	10^6
Target smoothing coefficient α	0.005
Discount factor γ	0.99 for D4RL 0.998 for Meta-World

Table 2: Hyperparameters for IQL

Hyperparameter	Value
Critic pessimism coefficient α	$\{0.1, 1, 10, 100\}$
Number of hidden layers	2^*
Number of hidden units	256
Nonlinearity	ReLU
Policy distribution	tanh-Gaussian*
Learning rate η	3×10^{-4}
Minibatch size $\lvert\mathcal{D}_{\text{mini}}\rvert$	256
Gradient steps	10^6
Target smoothing coefficient τ	0.005
Discount factor γ	0.99 for D4RL 0.998 for Meta-World

Table 3: Hyperparameters for CQL. *We include an ablation study of CQL with 3 hidden layers and scaled tanh-Gaussian policy, i.e., the same architecture as ATAC and PSPI, in Appendix B.7.

IQL and CQL, when using negative reward, can keep the agent from falling for longer in `hopper-medium` and `hopper-medium-expert` datasets. We would like to note that our hyperparameters are tuned based on normalized scores rather than episode lengths. The statistics of episode lengths are included in Table 5.

B.5. Effect of Removing Terminal Transitions

For `hopper` and `walker2d` tasks, we remove the terminal transitions, i.e., the transitions where the `hopper` or walker falls, from the datasets with modified rewards. This is to satisfy the safety condition in Section 3.4. A terminal signal implicitly adds an absorbing[8], but in this case, unsafe, state into the data distribution μ. This means that staying within data support in a long term would not provide safety, as the agent can stay within support by falling. For D4RL datasets in particular, we need to remove terminal transitions (rather than just the terminal flag) since terminal transitions in the original datasets do not contain valid next states.[9]

This, however, means that there are two underlying variables: *1)* the data reward, and *2)* whether terminal transitions are removed, when comparing offline RL with wrong rewards and offline RL on the original dataset. To separate the effect of two variables, we study the performance of offline RL algorithms with the original true reward but the terminal transitions removed. The results[10] are listed in the No-terminal column in Table 13. We observe that ATAC and PSPI with the original true reward show similar results whether the terminal transitions are removed. IQL and CQL with true reward, interestingly, does significantly worse when the terminal transitions are removed. This is potentially due to the increased instability of

[8]This is also why we mark goal completion as terminal in Meta-World experiments. For goal-oriented tasks, terminal signal upon task completion implicitly creates an absorbing goal state in the data distribution.

[9]The authors of D4RL datasets set the next state to be a place-holder value when a transition is marked as terminal.

[10]We do not study decision transformer (DT) in this ablation since DT does not use terminal signals.

Hyperparameter	Value
Number of hidden layers	3
Number of hidden units	128
Nonlinearity	ReLU
Context length K	20
Dropout	0.1
Warmup steps	2×10^2
Learning rate η	5×10^{-4}
Minibatch size $\lvert\mathcal{D}_{\text{mini}}\rvert$	256
Gradient steps	7×10^4 for D4RL 5×10^3 for Meta-World
Gradient norm clip	0.25
Weight decay	10^{-4}
Return-to-goal conditioning	Return randomly sampled from the top 10% of trajectory returns in the data, w.r.t. the data reward

Table 4: Hyperparameters for the decision transformer (DT)

target network (Van Hasselt et al., 2018; Sutton and Barto, 2018) when there are no terminal signals.

B.6. Offline RL with Expert Datasets

In Section 3, we point out that data has a positive bias when it is generated by the expert policy, which is different from the length bias. We now experimentally[11] validate this claim. We consider the expert datasets from the D4RL locomotion tasks. As is visualized in Figure 8, `walker2d-expert` and `halfcheetah-expert` datasets do not have a length bias: almost all trajectories are of lengh 1000. We observe that all offline algorithms except decision transformer, when trained with wrong rewards, can achieve expert-level performance in these two datasets (see Figure 9). ATAC, PSPI and DT also perform well on the `hopper-expert` dataset while IQL and CQL receive lower scores when using wrong rewards. The actual normalized scores are listed in Table 6.

B.7. Ablation Study on CQL Architecture

Our results are generated by CQL with the architecture in Table 3, which is given by the original implementation. In this study, we examine the performance of CQL when using the same architecture as ATAC and PSPI, i.e., with 3 hidden layers and scaled $\tanh$-Gaussian. The normalized scores for D4RL datasets and success rate for Meta-World tasks are included in Table 7 and Table 8, respectively. We find the performance of CQL with 3 hidden layers and scaled $\tanh$-Gaussian policy, denoted as CQL_3, to be similar with the performance of CQL with the original architecture.

B.8. Compute Usage

For ATAC, PSPI, IQL, CQL and BC, each run takes around 5.5 hours on an NC4as_T4_v3 Azure virtual machine for both D4RL and Meta-World experiments. For D4RL experiments (including ablation studies), we use a total of 5 (algorithms) $\times$ 15 (D4RL datasets) $\times$ 5 (reward types) $\times$ 4 (hyperparameters) $\times$ 2 (seeds) $=$ 3000 runs for hyperparameter tuning and 6 (algorithms) $\times$ 15 (D4RL datasets) $\times$ 5 (reward types) $\times$ 10 (seeds) $=$ 4500 runs for generating the final results. For Meta-World experiments, we use 5 (algorithms) $\times$ 15 (Meta-World datasets) $\times$ 4 (reward types) $\times$ 4 (hyperparameters) $\times$ 3 (seeds) $=$ 3600 runs for hyperparameter tuning and 6 (algorithms) $\times$ 15 (Meta-World datasets) $\times$ 4 (reward types) $\times$ 10 (seeds) $=$ 3600 runs for generating the final results. This amount to 14700 runs or 66165 hours of training on NC4as_T4_v3 virtual machines.

For DT, we do not tune any parameters at large scale. Each run takes around 2.5 hours on an NC6s_v2 or ND6s Azure virtual machine. We train a total of 30 (D4RL and Meta-World datasets) $\times$ 4 (reward types) $\times$ 10 (seeds) $=$ 1200 DT agents.

[11] We will formally prove it in Appendix C.

Dataset	Behavior	BC	Algo	Original	Zero	Random	Negative	No-terminal
hopper random	22.1	93.2±5.0	ATAC	79.3±13.0	714.2±120.0	552.7±123.4	958.8±39.1	567.2±144.9
			PSPI	99.2±6.2	354.5±113.0	513.9±131.0	749.1±124.5	517.3±142.9
			IQL	108.7±4.0	137.1±18.0	12.8±0.1	12.1±2.7	14.5±1.0
			CQL	48.4±12.5	41.8±20.9	28.4±3.8	6.0±0.0	16.7±10.1
			DT	245.2±8.2	239.7±8.6	103.2±2.0	295.1±32.8	–
hopper medium	457.2	540.5±1.2	ATAC	808.4±15.3	999.7±0.3	998.9±1.0	997.6±1.9	840.2±19.8
			PSPI	906.8±16.9	897.7±16.2	993.0±6.4	907.8±6.0	833.9±29.7
			IQL	571.2±8.5	538.7±4.0	594.8±20.3	631.4±21.7	446.4±65.8
			CQL	851.1±35.9	594.7±123.9	447.2±97.4	962.2±17.0	503.7±123.0
			DT	630.4±6.7	641.1±8.2	638.8±6.1	633.4±9.1	–
hopper medium replay	197.0	345.4±7.0	ATAC	1000.0±0.0	904.5±44.3	909.4±35.3	909.6±29.5	1000.0±0.0
			PSPI	1000.0±0.0	1000.0±0.0	995.9±3.9	1000.0±0.0	1000.0±0.0
			IQL	598.5±19.8	222.5±33.2	22.3±4.2	88.1±65.5	100.9±53.6
			CQL	801.9±38.2	24.9±2.6	217.5±123.7	153.8±91.8	116.4±93.2
			DT	934.5±13.6	742.3±33.4	836.2±20.6	825.6±31.2	–
hopper medium expert	622.2	541.8±1.1	ATAC	930.1±13.1	965.3±10.5	836.5±71.7	998.3±1.6	885.1±36.3
			PSPI	974.5±6.1	848.0±44.6	546.0±84.4	988.9±4.2	969.8±5.3
			IQL	980.4±5.1	535.4±3.4	581.2±74.7	883.2±28.5	735.3±62.8
			CQL	800.8±31.4	516.6±72.7	633.1±87.9	901.9±55.7	489.1±103.0
			DT	817.4±22.5	813.8±22.0	817.8±17.0	852.1±17.4	–
walker2d random	20.4	81.2±2.1	ATAC	227.6±10.2	231.0±11.0	149.6±10.4	214.8±9.2	189.0±19.8
			PSPI	221.1±13.7	143.8±10.5	165.7±9.8	135.8±1.2	135.9±0.5
			IQL	240.5±3.8	83.7±4.9	79.9±21.4	182.3±1.6	62.8±23.0
			CQL	182.9±79.5	56.0±23.1	188.7±50.9	34.7±5.1	37.2±10.2
			DT	228.4±7.6	210.9±10.2	75.2±1.2	203.3±11.1	–
walker2d medium	839.6	895.3±2.4	ATAC	976.9±17.6	999.7±0.3	997.8±1.5	999.8±0.2	990.2±5.9
			PSPI	989.1±7.2	998.2±1.7	1000.0±0.0	973.0±4.9	991.7±4.7
			IQL	930.1±12.3	831.1±37.3	976.5±7.4	739.1±30.7	517.4±11.2
			CQL	991.0±3.0	105.3±34.7	86.9±32.7	321.5±107.8	130.4±35.0
			DT	968.7±3.3	955.6±7.5	951.5±4.7	954.9±6.9	–
walker2d medium replay	276.3	312.7±5.8	ATAC	923.1±16.4	967.8±21.5	968.4±19.8	817.4±83.6	985.8±10.0
			PSPI	914.4±30.4	997.1±2.7	995.3±3.2	931.8±53.2	997.3±1.2
			IQL	659.9±37.5	313.5±27.5	211.1±37.1	119.5±25.1	41.8±3.4
			CQL	945.1±14.9	71.3±27.5	35.1±6.9	72.8±23.3	87.5±26.7
			DT	776.1±21.3	755.5±10.7	841.9±14.8	740.8±17.8	–
walker2d medium expert	912.8	976.9±3.5	ATAC	929.8±44.4	979.4±15.9	999.4±0.6	1000.0±0.0	960.3±36.0
			PSPI	1000.0±0.0	918.9±21.3	1000.0±0.0	960.2±8.3	987.9±9.8
			IQL	999.6±0.4	989.0±2.9	808.8±60.9	856.7±21.7	334.1±12.5
			CQL	998.4±1.5	55.8±20.4	44.3±17.4	514.5±112.3	12.6±2.2
			DT	1000.0±0.0	999.1±0.9	1000.0±0.0	999.5±0.4	–

Table 5: Episode lengths for hopper and walker2d datasets. The mean and standard error of episode lengths are computed across 10 random seeds. For each random seed, we evaluate the final policy of each algorithm over 50 episodes.

Dataset	Behavior	BC	Algo	Original	Zero	Random	Negative	No-terminal
hopper expert	108.5	110.6±0.0	ATAC	111.1±0.1	110.9±0.1	110.7±0.2	110.9±0.1	111.4±0.0
			PSPI	110.9±0.2	111.1±0.1	111.1±0.1	111.0±0.2	110.9±0.3
			IQL	110.3±0.2	66.7±12.8	73.5±11.2	48.5±11.8	24.3±8.9
			CQL	101.2±3.0	0.7±0.0	0.7±0.0	66.0±12.5	0.6±0.0
			DT	107.2±0.7	107.7±0.6	108.5±0.3	109.0±0.3	–
walker2d expert	107.1	108.3±0.0	ATAC	110.2±0.1	108.0±0.1	107.1±0.5	107.8±0.0	110.3±0.0
			PSPI	109.3±0.0	107.9±0.1	102.1±2.8	107.8±0.2	109.5±0.0
			IQL	110.9±0.1	108.4±0.0	103.0±3.5	108.3±0.1	111.0±0.1
			CQL	108.3±0.2	108.2±0.2	108.0±0.1	108.1±0.2	108.3±0.3
			DT	109.0±0.1	108.9±0.1	109.0±0.1	109.0±0.1	–
halfcheetah expert	88.1	92.9±0.0	ATAC	94.0±0.4	92.4±0.4	92.4±1.2	93.1±0.1	–
			PSPI	94.6±0.1	92.9±0.3	88.0±1.3	93.5±0.3	–
			IQL	96.8±0.1	92.9±0.1	92.9±0.0	89.3±0.1	–
			CQL	92.3±0.9	93.9±0.4	93.7±0.6	94.0±0.2	–
			DT	92.6±0.1	92.8±0.1	92.6±0.1	92.8±0.1	–

Table 6: D4RL expert dataset normalized scores. The mean and standard error of normalized scores are computed across 10 random seeds. For each random seed, we evaluate the final policy of each algorithm over 50 episodes.

Figure 7: Episode lengths for `hopper` and `walker2d` datasets from D4RL. The mean and standard error of episode lengths are computed across 10 random seeds. For each random seed, we evaluate the final policy of each algorithm over 50 episodes. Note that the hyperparameters are tuned based on normalized scores rather than episode lengths.

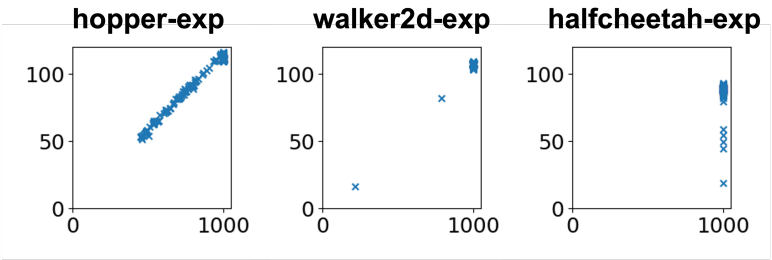


Figure 8: A visualization of hopper-expert, walker2d-expert and halfcheetah-expert datasets from D4RL. Each plot corresponds to a dataset for a task (row) with a dataset (column). Each trajectory in a dataset is represented as a data point with the x-coordinate being its episode length and y-coordinate being its normalized score.

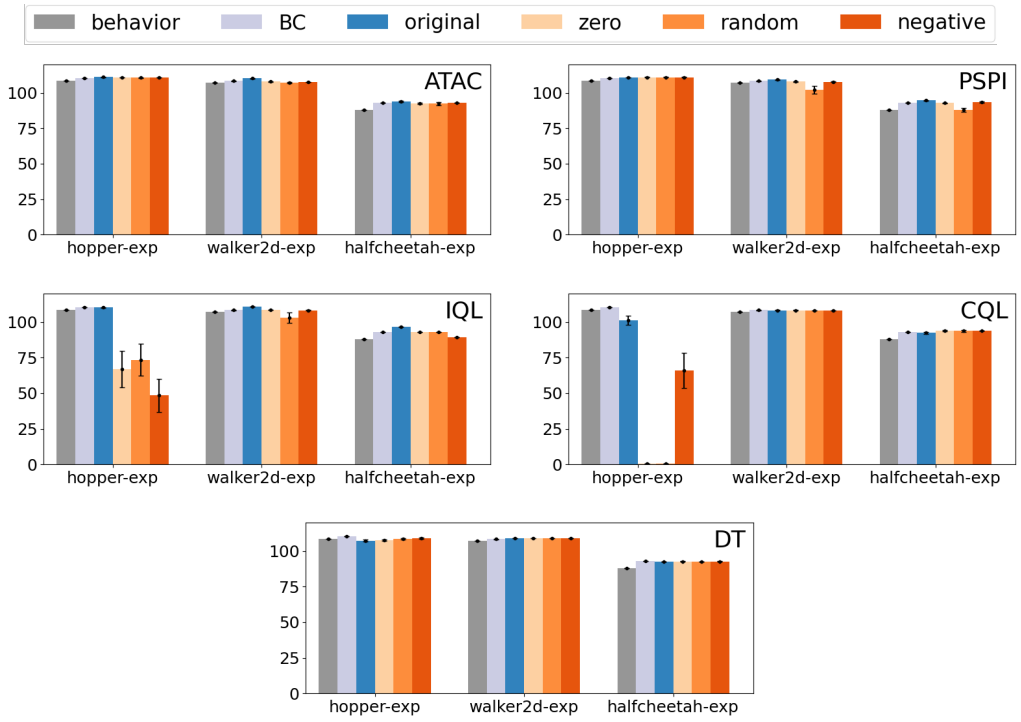


Figure 9: Normalized scores for expert datasets of hopper, walker2d and halfcheetah from D4RL (Fu et al., 2020). The mean and standard error of normalized scores are computed across 10 random seeds. For each random seed, we evaluate the final policy of each algorithm over 50 episodes.

Dataset	Algo	Original	Zero	Random	Negative	No-terminal
hopper-random	CQL	3.0±0.7	2.0±1.1	1.8±0.2	0.7±0.0	1.2±0.5
	CQL_3	11.2±2.2	2.0±0.4	7.7±3.5	3.9±2.9	10.1±3.2
hopper-medium	CQL	85.0±3.7	55.0±11.7	36.9±9.8	89.0±1.5	46.3±11.5
	CQL_3	48.0±12.2	61.7±12.2	17.3±9.8	75.8±6.9	32.8±10.3
hopper-medium-replay	CQL	79.7±3.7	1.9±0.1	7.6±3.7	5.8±2.7	4.4±2.8
	CQL_3	100.3±1.6	1.8±0.0	1.8±0.0	1.6±0.2	1.8±0.0
hopper-medium-expert	CQL	88.8±3.6	49.7±8.1	60.7±9.4	88.6±6.1	47.4±11.0
	CQL_3	104.7±1.4	40.6±10.7	17.3±9.0	48.2±8.7	30.0±9.5
walker2d-random	CQL	4.0±1.8	0.9±0.9	4.3±1.4	-0.4±0.0	0.2±0.4
	CQL_3	4.8±0.2	1.4±0.8	3.3±1.2	0.9±0.6	-0.2±0.1
walker2d-medium	CQL	81.4±0.3	2.0±0.9	1.2±0.7	14.5±7.4	2.1±0.8
	CQL_3	81.0±0.4	5.0±4.7	-0.2±0.0	13.3±7.7	0.1±0.3
walker2d-medium-replay	CQL	74.1±1.2	0.5±0.3	-0.2±0.1	0.2±0.3	0.4±0.4
	CQL_3	76.8±1.3	-0.2±0.1	0.1±0.1	0.1±0.3	0.3±0.3
walker2d-medium-expert	CQL	109.3±0.3	0.8±0.5	0.1±0.2	28.0±9.1	-0.2±0.0
	CQL_3	108.0±0.6	-0.2±0.1	-0.3±0.1	26.8±11.0	-0.2±0.0
halfcheetah-random	CQL	30.7±0.6	1.0±0.8	-1.4±0.3	-5.4±0.3	–
	CQL_3	31.3±0.6	0.5±0.4	1.9±0.1	-7.0±0.4	–
halfcheetah-medium	CQL	57.2±1.5	43.2±0.1	43.5±0.1	42.7±0.1	–
	CQL_3	65.1±0.6	43.1±0.1	43.3±0.1	42.2±0.1	–
halfcheetah-medium-replay	CQL	48.2±5.4	34.2±0.7	33.1±0.9	32.4±1.0	–
	CQL_3	47.8±5.2	38.3±1.0	37.7±1.1	32.6±1.2	–
halfcheetah-medium-expert	CQL	75.2±1.0	81.2±2.1	69.6±2.8	42.4±0.2	–
	CQL_3	88.0±1.7	75.5±1.7	80.0±2.5	42.6±0.1	–

Table 7: D4RL normalized scores for CQL with the original architecture in Table 3 (CQL) and with the same architecture as ATAC and PSPI (CQL_3), i.e., with 3 hidden layers and scaled $\tanh$-Gaussian. We find that CQL performs similarly under the two choices of architecture.

Dataset	Algo	Original	Zero	Random	Negative
button-press	CQL	0.88±0.07	0.83±0.10	0.84±0.09	0.94±0.03
	CQL_3	0.78±0.10	0.90±0.06	0.73±0.11	0.68±0.11
door-open	CQL	0.82±0.09	0.07±0.07	0.12±0.08	0.00±0.00
	CQL_3	0.47±0.15	0.29±0.14	0.00±0.00	0.38±0.15
drawer-close	CQL	0.91±0.04	0.99±0.01	0.96±0.02	0.97±0.02
	CQL_3	0.99±0.01	0.93±0.03	0.97±0.02	0.98±0.01
drawer-open	CQL	0.78±0.04	0.66±0.07	0.65±0.07	0.84±0.03
	CQL_3	0.86±0.03	0.79±0.09	0.78±0.06	0.80±0.05
peg-insert-side	CQL	0.08±0.02	0.00±0.00	0.02±0.01	0.00±0.00
	CQL_3	0.10±0.02	0.05±0.03	0.08±0.02	0.03±0.02
pick-place	CQL	0.01±0.00	0.06±0.02	0.05±0.02	0.07±0.03
	CQL_3	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
push	CQL	0.00±0.00	0.01±0.00	0.01±0.00	0.03±0.02
	CQL_3	0.03±0.01	0.00±0.00	0.00±0.00	0.01±0.01
reach	CQL	0.17±0.03	0.29±0.04	0.25±0.05	0.23±0.03
	CQL_3	0.35±0.05	0.35±0.03	0.37±0.05	0.22±0.06
window-close	CQL	1.00±0.00	1.00±0.00	0.59±0.13	0.55±0.13
	CQL_3	0.97±0.03	0.75±0.12	0.77±0.12	0.98±0.02
window-open	CQL	0.83±0.03	0.82±0.05	0.92±0.02	0.83±0.07
	CQL_3	0.84±0.07	0.94±0.02	0.93±0.02	0.77±0.09
bin-picking	CQL	0.00±0.00	0.02±0.02	0.00±0.00	0.00±0.00
	CQL_3	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
box-close	CQL	0.04±0.01	0.07±0.02	0.01±0.00	0.05±0.01
	CQL_3	0.03±0.02	0.00±0.00	0.01±0.00	0.01±0.00
door-lock	CQL	0.75±0.04	0.61±0.10	0.61±0.11	0.60±0.08
	CQL_3	0.63±0.06	0.64±0.09	0.57±0.10	0.62±0.09
door-unlock	CQL	0.87±0.03	0.77±0.09	0.89±0.04	0.93±0.03
	CQL_3	0.92±0.02	0.73±0.11	0.71±0.11	0.76±0.11
hand-insert	CQL	0.13±0.04	0.22±0.06	0.34±0.04	0.32±0.03
	CQL_3	0.04±0.03	0.08±0.03	0.18±0.03	0.08±0.02

Table 8: Meta-World success rate for CQL with the original architecture in Table 3 (CQL) and with the same architecture as ATAC and PSPI (CQL_3), i.e., with 3 hidden layers and scaled $\tanh$-Gaussian. We find that CQL performs similarly under the two choices of architecture.

Dataset	Original	Zero	Random	Negative	No-terminal
hopper-random	$\beta = 100.0$	$\beta = 100.0$	$\beta = 10.0$	$\beta = 1.0$	$\beta = 10.0$
hopper-medium	$\beta = 10.0$	$\beta = 100.0$	$\beta = 100.0$	$\beta = 1.0$	$\beta = 100.0$
hopper-medium-replay	$\beta = 100.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 0.1$	$\beta = 100.0$
hopper-medium-expert	$\beta = 1.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 1.0$	$\beta = 1.0$
hopper-expert	$\beta = 0.1$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 0.1$	$\beta = 0.1$
walker2d-random	$\beta = 10.0$	$\beta = 0.1$	$\beta = 1.0$	$\beta = 0.1$	$\beta = 100.0$
walker2d-medium	$\beta = 10.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 0.1$	$\beta = 100.0$
walker2d-medium-replay	$\beta = 100.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 100.0$
walker2d-medium-expert	$\beta = 10.0$	$\beta = 10.0$	$\beta = 1.0$	$\beta = 0.1$	$\beta = 10.0$
walker2d-expert	$\beta = 1.0$	$\beta = 100.0$	$\beta = 0.1$	$\beta = 0.1$	$\beta = 1.0$
halfcheetah-random	$\beta = 0.1$	$\beta = 0.1$	$\beta = 0.1$	$\beta = 0.1$	–
halfcheetah-medium	$\beta = 1.0$	$\beta = 100.0$	$\beta = 1.0$	$\beta = 0.1$	–
halfcheetah-medium-replay	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 0.1$	–
halfcheetah-medium-expert	$\beta = 0.1$	$\beta = 0.1$	$\beta = 0.1$	$\beta = 0.1$	–
halfcheetah-expert	$\beta = 0.1$	$\beta = 100.0$	$\beta = 0.1$	$\beta = 0.1$	–
button-press	$\beta = 10.0$	$\beta = 10.0$	$\beta = 1.0$	$\beta = 100.0$	–
door-open	$\beta = 100.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 1.0$	–
drawer-close	$\beta = 1.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 100.0$	–
drawer-open	$\beta = 10.0$	$\beta = 1.0$	$\beta = 10.0$	$\beta = 10.0$	–
peg-insert-side	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	–
pick-place	$\beta = 1.0$	$\beta = 10.0$	$\beta = 1.0$	$\beta = 1.0$	–
push	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	–
reach	$\beta = 10.0$	$\beta = 100.0$	$\beta = 10.0$	$\beta = 1.0$	–
window-close	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	–
window-open	$\beta = 0.1$	$\beta = 1.0$	$\beta = 10.0$	$\beta = 100.0$	–
bin-picking	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	–
box-close	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	–
door-lock	$\beta = 10.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 0.1$	–
door-unlock	$\beta = 100.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 10.0$	–
hand-insert	$\beta = 10.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	–

Table 9: Tuned hyperparameter values for ATAC

Dataset	Original	Zero	Random	Negative	No-terminal
hopper-random	$\beta = 10.0$	$\beta = 10.0$	$\beta = 100.0$	$\beta = 100.0$	$\beta = 100.0$
hopper-medium	$\beta = 100.0$	$\beta = 10.0$	$\beta = 100.0$	$\beta = 1.0$	$\beta = 100.0$
hopper-medium-replay	$\beta = 10.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 1.0$	$\beta = 1.0$
hopper-medium-expert	$\beta = 1.0$	$\beta = 10.0$	$\beta = 100.0$	$\beta = 1.0$	$\beta = 1.0$
hopper-expert	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 0.1$	$\beta = 1.0$
walker2d-random	$\beta = 10.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 1.0$	$\beta = 1.0$
walker2d-medium	$\beta = 10.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 0.1$	$\beta = 10.0$
walker2d-medium-replay	$\beta = 10.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 1.0$	$\beta = 100.0$
walker2d-medium-expert	$\beta = 1.0$	$\beta = 1.0$	$\beta = 10.0$	$\beta = 0.1$	$\beta = 10.0$
walker2d-expert	$\beta = 1.0$	$\beta = 1.0$	$\beta = 10.0$	$\beta = 0.1$	$\beta = 1.0$
halfcheetah-random	$\beta = 100.0$	$\beta = 100.0$	$\beta = 0.1$	$\beta = 1.0$	–
halfcheetah-medium	$\beta = 1.0$	$\beta = 10.0$	$\beta = 1.0$	$\beta = 0.1$	–
halfcheetah-medium-replay	$\beta = 10.0$	$\beta = 1.0$	$\beta = 10.0$	$\beta = 0.1$	–
halfcheetah-medium-expert	$\beta = 0.1$	$\beta = 1.0$	$\beta = 0.1$	$\beta = 0.1$	–
halfcheetah-expert	$\beta = 0.1$	$\beta = 1.0$	$\beta = 10.0$	$\beta = 0.1$	–
button-press	$\beta = 0.1$	$\beta = 10.0$	$\beta = 1.0$	$\beta = 100.0$	–
door-open	$\beta = 10.0$	$\beta = 1.0$	$\beta = 10.0$	$\beta = 0.1$	–
drawer-close	$\beta = 1.0$	$\beta = 1.0$	$\beta = 10.0$	$\beta = 10.0$	–
drawer-open	$\beta = 100.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 10.0$	–
peg-insert-side	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 10.0$	–
pick-place	$\beta = 10.0$	$\beta = 1.0$	$\beta = 10.0$	$\beta = 1.0$	–
push	$\beta = 100.0$	$\beta = 10.0$	$\beta = 1.0$	$\beta = 1.0$	–
reach	$\beta = 100.0$	$\beta = 100.0$	$\beta = 10.0$	$\beta = 10.0$	–
window-close	$\beta = 0.1$	$\beta = 0.1$	$\beta = 1.0$	$\beta = 0.1$	–
window-open	$\beta = 10.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 0.1$	–
bin-picking	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	–
box-close	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	$\beta = 1.0$	–
door-lock	$\beta = 1.0$	$\beta = 1.0$	$\beta = 10.0$	$\beta = 10.0$	–
door-unlock	$\beta = 100.0$	$\beta = 10.0$	$\beta = 10.0$	$\beta = 10.0$	–
hand-insert	$\beta = 10.0$	$\beta = 100.0$	$\beta = 1.0$	$\beta = 1.0$	–

Table 10: Tuned hyperparameter values for PSPI

Dataset	Original	Zero	Random	Negative	No-terminal
hopper-random	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.9$	$\beta = 3.0, \tau = 0.7$
hopper-medium	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$
hopper-medium-replay	$\beta = 3.0, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.9$	$\beta = 3.0, \tau = 0.9$
hopper-medium-expert	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$
hopper-expert	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$
walker2d-random	$\beta = 3.0, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$
walker2d-medium	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$
walker2d-medium-replay	$\beta = 3.0, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$
walker2d-medium-expert	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$
walker2d-expert	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$
halfcheetah-random	$\beta = 3.0, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	–
halfcheetah-medium	$\beta = 3.0, \tau = 0.9$	$\beta = 0.3, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–
halfcheetah-medium-replay	$\beta = 3.0, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.9$	–
halfcheetah-medium-expert	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	–
halfcheetah-expert	$\beta = 3.0, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–
button-press	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–
door-open	$\beta = 3.0, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–
drawer-close	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–
drawer-open	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	–
peg-insert-side	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–
pick-place	$\beta = 0.3, \tau = 0.9$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–
push	$\beta = 0.3, \tau = 0.9$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.9$	$\beta = 0.3, \tau = 0.9$	–
reach	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–
window-close	$\beta = 0.3, \tau = 0.9$	$\beta = 0.3, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–
window-open	$\beta = 0.3, \tau = 0.9$	$\beta = 3.0, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.9$	–
bin-picking	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–
box-close	$\beta = 0.3, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.9$	–
door-lock	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–
door-unlock	$\beta = 0.3, \tau = 0.7$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.9$	$\beta = 0.3, \tau = 0.7$	–
hand-insert	$\beta = 0.3, \tau = 0.9$	$\beta = 3.0, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	$\beta = 0.3, \tau = 0.7$	–

Table 11: Tuned hyperparameter values for IQL

Dataset	Original	Zero	Random	Negative	No-terminal
hopper-random	$\alpha = 0.1$	$\alpha = 0.1$	$\alpha = 100.0$	$\alpha = 0.1$	$\alpha = 1.0$
hopper-medium	$\alpha = 1.0$	$\alpha = 10.0$	$\alpha = 10.0$	$\alpha = 10.0$	$\alpha = 10.0$
hopper-medium-replay	$\alpha = 10.0$	$\alpha = 10.0$	$\alpha = 100.0$	$\alpha = 10.0$	$\alpha = 10.0$
hopper-medium-expert	$\alpha = 10.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 10.0$	$\alpha = 10.0$
hopper-expert	$\alpha = 10.0$	$\alpha = 1.0$	$\alpha = 0.1$	$\alpha = 10.0$	$\alpha = 100.0$
walker2d-random	$\alpha = 0.1$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 10.0$
walker2d-medium	$\alpha = 10.0$	$\alpha = 10.0$	$\alpha = 1.0$	$\alpha = 10.0$	$\alpha = 10.0$
walker2d-medium-replay	$\alpha = 10.0$	$\alpha = 10.0$	$\alpha = 1.0$	$\alpha = 100.0$	$\alpha = 10.0$
walker2d-medium-expert	$\alpha = 10.0$	$\alpha = 0.1$	$\alpha = 1.0$	$\alpha = 10.0$	$\alpha = 100.0$
walker2d-expert	$\alpha = 10.0$	$\alpha = 10.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 10.0$
halfcheetah-random	$\alpha = 0.1$	$\alpha = 0.1$	$\alpha = 1.0$	$\alpha = 100.0$	–
halfcheetah-medium	$\alpha = 0.1$	$\alpha = 1.0$	$\alpha = 1.0$	$\alpha = 100.0$	–
halfcheetah-medium-replay	$\alpha = 0.1$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	–
halfcheetah-medium-expert	$\alpha = 100.0$	$\alpha = 1.0$	$\alpha = 1.0$	$\alpha = 100.0$	–
halfcheetah-expert	$\alpha = 100.0$	$\alpha = 10.0$	$\alpha = 100.0$	$\alpha = 100.0$	–
button-press	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 10.0$	$\alpha = 100.0$	–
door-open	$\alpha = 1.0$	$\alpha = 10.0$	$\alpha = 100.0$	$\alpha = 10.0$	–
drawer-close	$\alpha = 100.0$	$\alpha = 10.0$	$\alpha = 10.0$	$\alpha = 10.0$	–
drawer-open	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 10.0$	–
peg-insert-side	$\alpha = 1.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 10.0$	–
pick-place	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	–
push	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	–
reach	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 10.0$	–
window-close	$\alpha = 1.0$	$\alpha = 0.1$	$\alpha = 10.0$	$\alpha = 10.0$	–
window-open	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	–
bin-picking	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	–
box-close	$\alpha = 1.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	–
door-lock	$\alpha = 1.0$	$\alpha = 10.0$	$\alpha = 10.0$	$\alpha = 100.0$	–
door-unlock	$\alpha = 100.0$	$\alpha = 10.0$	$\alpha = 10.0$	$\alpha = 10.0$	–
hand-insert	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	$\alpha = 100.0$	–

Table 12: Tuned hyperparameter values for CQL

Dataset	Behavior	BC	Algo	Original	Zero	Random	Negative	No-terminal
hopper random	1.2	3.6±0.2	ATAC	5.6±0.8	22.6±3.8	18.4±3.8	30.5±1.1	18.8±4.5
			PSPI	7.2±0.3	12.3±3.5	17.3±3.9	23.5±3.7	17.1±4.4
			IQL	7.7±0.2	5.0±0.7	1.0±0.0	0.9±0.1	1.0±0.0
			CQL	3.0±0.7	2.0±1.1	1.8±0.2	0.7±0.0	1.2±0.5
			DT	11.5±0.3	11.0±0.4	6.5±0.1	12.3±0.8	–
hopper medium	44.3	53.3±0.1	ATAC	84.8±1.5	94.0±0.6	92.3±0.3	91.5±0.5	88.1±2.0
			PSPI	94.5±1.7	88.4±1.3	92.7±0.5	89.4±0.6	87.2±3.0
			IQL	57.4±0.9	53.1±0.4	59.0±2.1	63.0±2.1	43.0±7.3
			CQL	85.0±3.7	55.0±11.7	36.9±9.8	89.0±1.5	46.3±11.5
			DT	62.4±0.6	63.5±0.8	63.3±0.6	62.6±0.8	–
hopper medium replay	15.0	26.7±0.6	ATAC	101.5±0.2	65.7±3.9	77.2±2.9	48.1±2.1	99.4±0.6
			PSPI	101.4±0.3	32.4±1.3	46.9±7.2	31.0±0.0	99.1±0.4
			IQL	61.0±1.9	16.5±2.5	1.5±0.3	3.5±2.2	6.1±3.1
			CQL	79.7±3.7	1.9±0.1	7.6±3.7	5.8±2.7	4.4±2.8
			DT	84.0±0.9	60.7±2.8	67.8±1.1	68.8±2.6	–
hopper medium expert	64.8	53.6±0.1	ATAC	104.2±1.5	106.4±1.1	92.2±8.0	90.4±0.4	98.9±4.1
			PSPI	108.9±0.6	93.3±5.0	58.8±9.4	95.3±0.3	108.3±0.6
			IQL	109.2±0.5	52.8±0.3	60.5±8.7	94.5±3.7	77.9±7.6
			CQL	88.8±3.6	49.7±8.1	60.7±9.4	88.6±6.1	47.4±11.0
			DT	89.9±2.7	89.8±2.6	89.7±2.2	93.8±2.3	–
walker2d random	0.0	1.4±0.0	ATAC	6.4±0.7	5.8±0.5	3.2±0.9	5.6±0.6	4.9±1.1
			PSPI	6.3±0.4	2.1±0.8	1.6±0.5	1.0±0.5	5.5±0.0
			IQL	7.2±0.1	1.3±0.1	1.4±0.7	6.7±0.0	1.8±1.0
			CQL	4.0±1.8	0.9±0.9	4.3±1.4	-0.4±0.0	0.2±0.4
			DT	6.5±0.2	5.3±0.5	1.7±0.0	6.4±0.2	–
walker2d medium	62.0	68.1±0.5	ATAC	86.8±1.6	76.6±0.3	76.8±0.7	72.7±0.2	86.4±0.6
			PSPI	87.7±0.7	71.2±0.5	72.2±0.6	77.5±0.5	85.2±0.5
			IQL	78.4±1.1	62.8±3.0	72.0±2.2	53.3±3.3	39.3±0.9
			CQL	81.4±0.3	2.0±0.9	1.2±0.7	14.5±7.4	2.1±0.8
			DT	76.6±0.3	75.7±0.5	75.8±0.2	75.5±0.3	–
walker2d medium replay	14.8	19.3±0.4	ATAC	85.7±1.5	67.2±2.6	66.5±1.6	42.3±6.7	80.2±1.0
			PSPI	84.9±3.2	69.9±0.7	68.1±2.2	56.2±4.5	81.3±0.8
			IQL	57.7±3.6	15.5±2.2	3.2±1.2	0.2±0.7	0.7±0.1
			CQL	74.1±1.2	0.5±0.3	-0.2±0.1	0.2±0.3	0.4±0.4
			DT	56.3±1.3	53.9±1.0	62.0±1.1	53.5±1.5	–
walker2d medium expert	82.7	103.0±1.0	ATAC	105.9±5.4	96.8±4.4	93.6±4.8	72.7±0.3	108.9±4.5
			PSPI	110.0±0.1	80.6±6.3	109.2±0.1	75.5±0.7	111.7±1.3
			IQL	111.0±0.2	106.2±0.4	71.5±7.4	70.7±2.9	28.0±0.7
			CQL	109.3±0.3	0.8±0.5	0.1±0.2	28.0±9.1	-0.2±0.0
			DT	108.5±0.1	108.4±0.1	108.5±0.2	108.4±0.2	–
halfcheetah random	-0.1	1.9±0.0	ATAC	2.2±0.0	2.2±0.0	2.2±0.0	2.2±0.0	–
			PSPI	2.6±0.4	2.3±0.0	2.2±0.0	2.3±0.0	–
			IQL	18.6±0.3	1.8±0.1	2.0±0.1	2.2±0.0	–
			CQL	30.7±0.6	1.0±0.8	-1.4±0.3	-5.4±0.3	–
			DT	2.3±0.0	2.3±0.0	2.3±0.0	2.3±0.0	–
halfcheetah medium	40.7	42.2±0.0	ATAC	51.2±0.1	43.1±0.1	43.0±0.1	41.4±0.1	–
			PSPI	49.4±0.1	43.0±0.1	42.9±0.1	42.8±0.1	–
			IQL	50.0±0.1	39.3±2.8	42.3±0.1	34.7±0.3	–
			CQL	57.2±1.5	43.2±0.1	43.5±0.1	42.7±0.1	–
			DT	43.0±0.0	43.0±0.0	43.0±0.0	43.0±0.0	–
halfcheetah medium replay	27.2	35.9±0.1	ATAC	47.4±0.1	33.7±2.2	31.1±2.5	40.3±0.2	–
			PSPI	48.5±0.2	35.6±0.6	36.1±0.7	35.5±0.5	–
			IQL	43.7±0.2	36.1±0.5	35.3±0.3	18.1±0.4	–
			CQL	48.2±5.4	34.2±0.7	33.1±0.9	32.4±1.0	–
			DT	38.6±0.1	37.2±0.2	37.1±0.2	37.0±0.2	–
halfcheetah medium expert	64.4	67.9±0.4	ATAC	91.5±0.6	46.4±2.0	44.9±1.2	41.0±0.1	–
			PSPI	86.7±1.9	52.6±1.7	49.8±1.9	42.2±0.1	–
			IQL	94.8±0.1	56.5±3.2	70.8±0.8	37.9±0.2	–
			CQL	75.2±1.0	81.2±2.1	69.6±2.8	42.4±0.2	–
			DT	78.8±0.8	81.1±0.7	79.6±1.4	78.3±0.9	–

Table 13: D4RL normalized scores. The mean and standard error of normalized scores are computed across 10 random seeds. For each random seed, we evaluate the final policy over 50 episodes.

Dataset	BC	Algo	Original	Zero	Random	Negative
button-press	0.98±0.00	ATAC	1.00±0.00	0.90±0.09	1.00±0.00	0.90±0.09
		PSPI	0.86±0.10	0.90±0.09	0.84±0.10	0.90±0.09
		IQL	0.98±0.01	0.97±0.01	0.94±0.02	0.88±0.02
		CQL	0.88±0.07	0.83±0.10	0.84±0.09	0.94±0.03
		DT	1.00±0.00	1.00±0.00	1.00±0.00	1.00±0.00
door-open	0.81±0.02	ATAC	1.00±0.00	1.00±0.00	1.00±0.00	0.95±0.04
		PSPI	1.00±0.00	0.95±0.03	0.96±0.02	0.81±0.09
		IQL	0.94±0.02	0.86±0.03	0.92±0.03	0.92±0.02
		CQL	0.82±0.09	0.07±0.07	0.12±0.08	0.00±0.00
		DT	1.00±0.00	1.00±0.00	1.00±0.00	1.00±0.00
drawer-close	0.99±0.00	ATAC	0.89±0.09	0.97±0.03	1.00±0.00	1.00±0.00
		PSPI	0.77±0.11	0.96±0.04	1.00±0.00	1.00±0.00
		IQL	0.98±0.01	0.99±0.01	0.99±0.00	0.97±0.01
		CQL	0.91±0.04	0.99±0.01	0.96±0.02	0.97±0.02
		DT	0.99±0.01	1.00±0.00	1.00±0.00	1.00±0.00
drawer-open	0.88±0.01	ATAC	0.72±0.07	0.35±0.09	0.41±0.08	0.69±0.09
		PSPI	0.61±0.10	0.42±0.09	0.57±0.07	0.74±0.06
		IQL	0.93±0.02	0.84±0.02	0.73±0.04	0.52±0.03
		CQL	0.78±0.04	0.66±0.07	0.65±0.07	0.84±0.03
		DT	1.00±0.00	1.00±0.00	1.00±0.00	1.00±0.00
peg-insert-side	0.58±0.01	ATAC	0.82±0.04	0.88±0.06	0.83±0.05	0.92±0.03
		PSPI	0.75±0.09	0.73±0.07	0.92±0.03	0.91±0.03
		IQL	0.60±0.03	0.61±0.03	0.34±0.02	0.11±0.01
		CQL	0.08±0.02	0.00±0.00	0.02±0.01	0.00±0.00
		DT	0.07±0.01	0.00±0.00	0.01±0.00	0.00±0.00
pick-place	0.15±0.01	ATAC	0.38±0.06	0.35±0.05	0.35±0.05	0.24±0.05
		PSPI	0.47±0.04	0.16±0.05	0.31±0.04	0.14±0.04
		IQL	0.11±0.02	0.16±0.02	0.17±0.01	0.10±0.01
		CQL	0.01±0.00	0.06±0.02	0.05±0.02	0.07±0.03
		DT	0.06±0.01	0.06±0.01	0.06±0.01	0.06±0.01
push	0.19±0.01	ATAC	0.48±0.09	0.48±0.08	0.34±0.04	0.27±0.06
		PSPI	0.47±0.05	0.42±0.04	0.35±0.08	0.37±0.08
		IQL	0.10±0.02	0.18±0.02	0.14±0.01	0.13±0.01
		CQL	0.00±0.00	0.01±0.00	0.01±0.00	0.03±0.02
		DT	0.04±0.01	0.06±0.02	0.04±0.01	0.07±0.01
reach	0.59±0.01	ATAC	0.86±0.03	0.81±0.05	0.83±0.03	0.81±0.03
		PSPI	0.92±0.02	0.77±0.04	0.84±0.03	0.84±0.04
		IQL	0.51±0.02	0.59±0.04	0.57±0.02	0.58±0.02
		CQL	0.17±0.03	0.29±0.04	0.25±0.05	0.23±0.03
		DT	0.67±0.03	0.79±0.02	0.81±0.02	0.83±0.02
window-close	1.00±0.00	ATAC	0.98±0.01	1.00±0.00	0.90±0.09	0.86±0.10
		PSPI	0.90±0.08	0.99±0.01	0.89±0.07	0.92±0.06
		IQL	0.99±0.01	1.00±0.00	0.99±0.01	0.99±0.00
		CQL	1.00±0.00	1.00±0.00	0.59±0.13	0.55±0.13
		DT	1.00±0.00	0.98±0.01	0.99±0.00	1.00±0.00
window-open	0.98±0.00	ATAC	0.51±0.13	0.74±0.10	1.00±0.00	1.00±0.00
		PSPI	1.00±0.00	0.98±0.01	1.00±0.00	0.70±0.12
		IQL	0.97±0.01	0.98±0.01	0.98±0.00	0.96±0.02
		CQL	0.83±0.03	0.82±0.05	0.92±0.02	0.83±0.07
		DT	0.91±0.01	0.91±0.01	0.89±0.02	0.90±0.01
bin-picking	0.72±0.01	ATAC	0.41±0.12	0.49±0.09	0.51±0.12	0.37±0.12
		PSPI	0.29±0.10	0.41±0.08	0.33±0.09	0.30±0.11
		IQL	0.39±0.02	0.59±0.07	0.35±0.04	0.31±0.04
		CQL	0.00±0.00	0.02±0.02	0.00±0.00	0.00±0.00
		DT	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
box-close	0.57±0.01	ATAC	0.20±0.09	0.50±0.13	0.32±0.11	0.53±0.12
		PSPI	0.34±0.10	0.44±0.10	0.28±0.09	0.24±0.08
		IQL	0.58±0.02	0.54±0.02	0.40±0.04	0.23±0.01
		CQL	0.04±0.01	0.07±0.02	0.01±0.00	0.05±0.01
		DT	0.02±0.00	0.03±0.01	0.04±0.02	0.05±0.03
door-lock	0.96±0.00	ATAC	1.00±0.00	0.98±0.01	0.70±0.14	0.62±0.12
		PSPI	0.78±0.12	0.80±0.12	0.99±0.01	0.82±0.10
		IQL	0.99±0.00	0.96±0.01	0.91±0.01	0.69±0.04
		CQL	0.75±0.04	0.61±0.10	0.61±0.11	0.60±0.08
		DT	0.81±0.02	0.83±0.02	0.80±0.02	0.78±0.02
door-unlock	0.86±0.00	ATAC	0.97±0.01	0.93±0.05	0.98±0.01	0.94±0.03
		PSPI	0.95±0.04	0.98±0.01	0.88±0.09	0.98±0.01
		IQL	0.87±0.02	0.88±0.02	0.86±0.02	0.83±0.02
		CQL	0.87±0.03	0.77±0.09	0.89±0.04	0.93±0.03
		DT	0.97±0.01	0.97±0.01	0.98±0.01	0.98±0.01
hand-insert	0.39±0.01	ATAC	0.80±0.03	0.36±0.08	0.80±0.03	0.61±0.04
		PSPI	0.81±0.02	0.62±0.08	0.75±0.04	0.81±0.03
		IQL	0.35±0.03	0.34±0.02	0.18±0.03	0.14±0.02
		CQL	0.13±0.04	0.22±0.06	0.34±0.04	0.32±0.03
		DT	0.24±0.03	0.21±0.02	0.28±0.02	0.26±0.02

Table 14: Meta-World success rate. The mean and standard error of success rate are computed across 10 random seeds. For each random seed, we evaluate the final policy over 50 episodes.

C. Technical Details of Section 3

Here we provide the details of our theory on the robustness of offline RL in Section 3.

Theorem 3.1. *(Informal) Under Assumption 2.1 and certain regularity assumptions, if an offline RL algorithm Algo is set to be sufficiently pessimistic and the data distribution μ has a positive bias, for any data reward $\tilde{r} \in \tilde{\mathcal{R}}$, the policy $\hat{\pi}$ learned by Algo from the dataset $\tilde{\mathcal{D}}$ has performance guarantee*

$$V_r^{\pi^*}(d_0) - V_r^{\hat{\pi}}(d_0) \leq O(\iota)$$

as well as safety guarantee

$$(1-\gamma)\sum_{t=0}^{\infty}\gamma^t \mathrm{Prob}\left(\exists \tau \in [0,t], s_\tau \notin \mathit{supp}(\mu)|\hat{\pi}\right) \leq O(\iota)$$

for a small ι that decreases to zero as the degree of pessimism and dataset size increase.

Below we first provide in Appendix C.1 some background for stating the formal version Theorem 3.1 in Appendix C.2. Then we present the formal claims on survival instinct and conditions on positive data bias in Appendix C.3 and Appendix C.4, respectively. Finally, we prove Theorem 3.1 in Appendix C.5. Our discussion here will be focusing the discounted infinite-horizon step. We discuss the finite horizon variant in Appendix D.

C.1. Definitions

First, we introduce some definitions so that we can state the formal version of Theorem 3.1. We define constrained MDPs.

Definition C.1 (Constrained MDP)**.** Let $f, g : \mathcal{S} \times \mathcal{A} \to [-1, 1]$. A CMDP problem $\mathcal{C}(\mathcal{S}, \mathcal{A}, f, g, P, \gamma)$ is defined as

$$\max_{\pi} \quad V_f^{\pi}(d_0), \quad \text{s.t.} \quad V_g^{\pi}(d_0) \leq 0.$$

Let $\pi^\dagger$ denote its optimal policy. For $\delta \geq 0$, we define the set of δ-*approximately optimal policies*,

$$\Pi_{f,g}^{\dagger}(\delta) = \{\pi : V_f^{\pi^\dagger}(d_0) - V_f^{\pi}(d_0) \leq \delta, V_g^{\pi}(d_0) \leq \delta\}$$

We will drop the subscript f, g from $\Pi_{f,g}^{\dagger}$ when it is clear from the context. In addition, we define a *sensitivity function* to approximately optimal policies,

$$\kappa(\delta) \coloneqq \max_{\pi \in \Pi^\dagger(\delta)} V_f^{\pi}(d_0) - V_f^{\pi^\dagger}(d_0)$$

The sensitivity function measures how much the approximately optimal policies (which can slightly violate the constraint of the CMDP) would perform better than the optimal policy of the CMDP (which satisfies the constraint). By definition, $\kappa(\delta) \in [0, \frac{2}{1-\gamma}]$ and its value can be smaller than $\frac{2}{1-\gamma}$.

By duality, a CMDP problem is related to a saddle-point problem induced by its constraint. We review the definition of saddle points and this fact below.

Definition C.2. Given a bifunction $\mathcal{L}(x, y) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, we say $(x^\dagger, y^\dagger) \in \mathcal{X} \times \mathcal{Y}$ is a *saddle point* of $\mathcal{L}$, if for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ it holds that $\mathcal{L}(x, y^\dagger) \leq \mathcal{L}(x^\dagger, y^\dagger) \leq \mathcal{L}(x^\dagger, y)$.

Lemma C.3 (Duality)**.** *(Altman, 1999) Let Π denote the space of policies. Consider a CMDP $\mathcal{C} \coloneqq \mathcal{C}(\mathcal{S}, \mathcal{A}, f, g, P, \gamma)$. Define the Lagrangian $\mathcal{L}(\pi, \lambda) \coloneqq V_f^{\pi}(d_0) - \lambda V_g^{\pi}(d_0)$ for $\pi \in \Pi$ and $\lambda \geq 0$. Let $\pi^\dagger$ and $\lambda^\dagger$ denote the optimal policy and Lagrange multiplier, i.e.,*

$$\pi^\dagger \in \arg\max_{\pi} \min_{\lambda \geq 0} \mathcal{L}(\pi, \lambda) \quad \textit{and} \quad \lambda^\dagger \in \arg\min_{\lambda \geq 0} \max_{\pi} \mathcal{L}(\pi, \lambda)$$

Then together they form a saddle point $(\pi^\dagger, \lambda^\dagger)$ to the Lagrangian $\mathcal{L}$.

We also introduce a definition to characterize the degree of pessimism of offline RL algorithms. We call it an admissibility condition defined below, which measures the size of the data-consistent reward functions that the offline RL algorithm can have a small regret.

Definition C.4. [Admissibility] For $R \geq 0$, we say an offline RL algorithm $Algo$ is *R-admissible* with respect to $\tilde{\mathcal{R}}$, if for any $\tilde{r} \in \tilde{\mathcal{R}}$, given $\tilde{\mathcal{D}}$, $Algo$ learns a policy $\hat{\pi}$ satisfying the following with high probability: for any policy $\pi \in \Pi$,

$$\max_{\bar{r} \in \mathcal{R}_R(\tilde{r})} V_{\bar{r}}^{\pi}(d_0) - V_{\bar{r}}^{\hat{\pi}}(d_0) \leq \mathcal{E}(\pi, \mu),$$

where we define a data-consistent reward class

$$\mathcal{R}_R(\tilde{r}) := \{\bar{r} : \bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in \text{supp}(\mu) \text{ and } \bar{r}(s,a) \in [-R, 1], \forall (s,a) \in \mathcal{S} \times \mathcal{A}\},$$

$\mathcal{E}(\pi, \mu)$ is some regret upper bound such that $\mathcal{E}(\pi, \mu) = o(1)$ if $\sup_{s \in \mathcal{S}, a \in \mathcal{A}} \frac{d^{\pi}(s,a)}{\mu(s,a)} < \infty$, and $o(1)$ denotes a term that vanishes as the dataset size becomes infinite.

In Appendix E, we provide proofs of the degrees of admissibility for various existing algorithms, including model-free algorithms ATAC (Cheng et al., 2022), VI-LCB (Rashidinejad et al., 2021) PPI/PQI (Liu et al., 2020), PSPI (Xie et al., 2021), as well as model-algorithms, ARMOR (Xie et al., 2022; Bhardwaj et al., 2023), MOPO (Yu et al., 2020b), MOReL (Kidambi et al., 2020), and CPPO (Uehara and Sun, 2022). We show that their degree of admissibility increases as the algorithm becomes more pessimistic (by setting the hyperparameters).

Finally, we recall the definition of positive data bias in Section 3 for completeness, where the implicit CMDP $\mathcal{C}_{\mu}(\tilde{r})$ is defined in (1).

Definition C.5 (Positive Data Bias)**.** A distribution μ is $\frac{1}{\epsilon}$*-positively biased* w.r.t. a reward class $\tilde{\mathcal{R}}$ if

$$\max_{\tilde{r} \in \tilde{\mathcal{R}}} \max_{\pi \in \Pi^{\dagger}(\delta)} V_r^{\pi^*}(d_0) - V_r^{\pi}(d_0) \leq \epsilon + O(\delta) \tag{2}$$

for all $\delta \geq 0$, where $\Pi^{\dagger}(\delta)$ denotes the set of δ-approximately optimal policy of $\mathcal{C}_{\mu}(\tilde{r})$.

Note that the CMDP $\mathcal{C}_{\mu}(\tilde{r})$ is feasible because of the assumption on concentrability (Assumption 2.1). That is, we know π^* is feasible policy, as $(1-\gamma)V_{c_{\mu}}^{\pi^*}(d_0) = \mathbb{E}_{d^{\pi^*}}[\mathbb{1}[(s,a) \notin \text{supp}(\mu)]] = 0$ is implied by $\sup_{s,a} \frac{d^{\pi^*}(s,a)}{\mu(s,a)} < \infty$.

C.2. Formal Statement of the Main Theorem

Now we state the formal version of Theorem 3.1.

Assumption C.6 (Data Assumption)**.**

1. For all $\tilde{r} \in \tilde{\mathcal{R}}$, $\tilde{r}(s,a) \in [-1, 1]$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$.
2. The data distribution μ is $\frac{1}{\epsilon}$-positively biased with respect to $\tilde{\mathcal{R}}$.
3. There is a policy π satisfying $V_{c_{\mu}} \leq 0$.
4. For any $\tilde{r} \in \tilde{\mathcal{R}}$, the CMDP $\mathcal{C}_{\mu}(\tilde{r}) := \mathcal{C}(\mathcal{S}, \mathcal{A}, \tilde{r}, c_{\mu}, P, \gamma)$ in (1) has a sensitivity function $\kappa(\delta) \leq K$ for some $K < \infty$ and $\lim_{\delta \to 0^+} \kappa(\delta) = 0$. In addition, its solution $\pi^{\dagger}$ satisfies $\sup_{s,a} \frac{d^{\pi^{\dagger}}(s,a)}{\mu(s,a)} < \infty$.

Theorem C.7 (Main Result)**.** *Under the data assumption in Assumption C.6, consider an offline RL algorithm Algo that is sufficiently pessimistic to be* $(\frac{K}{\delta} + 2)$*-admissible with respect to* $\tilde{\mathcal{R}}$*. Then for* any $\tilde{r} \in \tilde{\mathcal{R}}$*, with high probability, the policy* $\hat{\pi}$ *learned by Algo from the dataset* $\tilde{\mathcal{D}} := \{(s,a,\tilde{r},s') | (s,a) \sim \mu, \tilde{r} = \tilde{r}(s,a), s' \sim P(\cdot|s,a)\}$ *has both performance guarantee*

$$V_r^{\pi^*}(d_0) - V_r^{\hat{\pi}}(d_0) \leq \epsilon + O(\iota)$$

and safety guarantee

$$(1-\gamma) \sum_{t=0}^{\infty} \gamma^t \text{Prob}\left(\exists \tau \in [0,t] s_{\tau} \notin \textit{supp}(\mu) | \hat{\pi}\right) \leq \iota$$

where $\iota := \delta + \kappa(\delta) + o(1)$ *and* $o(1)$ *denotes a term that vanishes as the dataset size becomes infinite.*

In Assumption C.6, the first assumption supposes the data reward (which may be the wrong reward) is bounded; the second assumption is that the data distribution is positively biased in view of the reward class $\tilde{\mathcal{R}}$. Finally, we make a regularity assumption on the convergence of $\kappa(\delta)$ to zero and $\sup_{s,a} \frac{d^{\pi^\dagger}(s,a)}{\mu(s,a)} < \infty$ to rule out some unrealistic $\tilde{\mathcal{R}}$ and P combinations. We note that the condition on the bounded density ratio is always true for tabular problem, and a degenerate case can happen when the state-action space is, e.g., continuous. On the other hand, a degenerate case that κ is discontinuous at $\delta = 0$ can happen due to the infinite problem horizon in the worst case; roughly speaking, this happens when it is impossible to determine the constraint violation using a finite-horizon estimator. We provide the following lemma to quantify the rate of $\kappa(\delta)$, which is proved in Appendix C.6. In Appendix D, we show this regularity condition is always true in the finite horizon setting.

Lemma C.8. *Suppose there is $\lambda^\dagger \leq L$ with some $L < \infty$ such that $\lambda^\dagger \in \arg\min_{\lambda \geq 0} \max_\pi V^\pi_{\tilde{r}}(d_0) - \lambda V^\pi_{c_\mu}(d_0)$. Then $\kappa(\delta) \leq L\delta$.*

Under the data assumption Assumption C.6, Theorem C.7 shows that if the offline RL algorithm $Algo$ is set to be sufficiently pessimistic (relative to the upper bound of the sensitivity function κ, which we know is at most $\frac{1}{1-\gamma}$ by the first assumption in Assumption C.6), then $Algo$ have both a performance guarantee to the true reward r and the a safety guarantee that the learned policy would have a small probability of leaving the data support.

We recall from Section 3 that Theorem C.7 is the effects of the survival instinct of offline RL and the positive bias in the data distribution. As we will show, the admissibility condition in Theorem C.7 would ensure that the survival instinct, which is what we stated informally as Proposition 3.3. In addition, we will provide concrete sufficient conditions for the data distribution being positively bias, as we alluded in Section 3.3. We discuss these details below and then we prove Theorem C.7 in Appendix C.5.

C.3. Survival Instinct

We show the formal version of Proposition 3.3 which says that if the offline algorithm is admissible then it has a survival instinct. That is, it implicitly solves the CMDP in (1) even though the constraint is never explicitly modeled in the algorithm. In Appendix E, we prove the admissibility for various existing algorithms, including model-free algorithms ATAC (Cheng et al., 2022), VI-LCB (Rashidinejad et al., 2021) PPI/PQI (Liu et al., 2020), PSPI (Xie et al., 2021), as well as model-algorithms, ARMOR (Xie et al., 2022; Bhardwaj et al., 2023), MOPO (Yu et al., 2020b), MOReL (Kidambi et al., 2020), and CPPO (Uehara and Sun, 2022).

Proposition C.9. *For $\tilde{r} \in \tilde{\mathcal{R}}$, assume the CMDP $\mathcal{C}_\mu(\tilde{r}) := \mathcal{C}(\mathcal{S}, \mathcal{A}, \tilde{r}, c_\mu, P, \gamma)$ has a sensitivity function such that $\lim_{\delta \to 0^+} \kappa(\delta) = 0$ and $\kappa(\delta) \leq K$ for some $K \leq \frac{1}{1-\gamma}$. Consider an offline RL algorithm $Algo$ that is $(\frac{K}{\delta} + 2)$-admissible with respect to $\tilde{\mathcal{R}}$. Then, with high probability, the policy $\hat{\pi}$ learned by $Algo$ with the dataset $\tilde{\mathcal{D}} := \{(s, a, \tilde{r}, s') | (s, a) \sim \mu, \tilde{r} = \tilde{r}(s, a), s' \sim P(\cdot|s, a)\}$ satisfies*

$$V^{\pi^\dagger}_{\tilde{r}}(d_0) - V^{\hat{\pi}}_{\tilde{r}}(d_0) \leq o(1)$$
$$V^{\hat{\pi}}_{c_\mu}(d_0) \leq \delta + \kappa(\delta) + o(1)$$

where $o(1)$ denotes a term that vanishes as the dataset size becomes infinite.

Proof of Proposition C.9. To begin, we introduce some extra notations. Given $\mathcal{C}_\mu(\tilde{r})$, we define a relaxed CMDP problem: $\max_\pi V^\pi_{\tilde{r}}(d_0), \text{s.t.} V^\pi_{c_\mu}(d_0) \leq \delta$ and its Lagrangian $\mathcal{L}_\delta(\pi, \lambda) := V^\pi_{\tilde{r}}(d_0) - \lambda(V^\pi_{c_\mu}(d_0) - \delta)$, where $\lambda \geq 0$ denotes the Lagrange multiplier. Let $\pi^\dagger_\delta$ and $\lambda^\dagger_\delta$ denote the optimal policy and Lagrange multiplier of this relaxed CMDP, and let $\pi^\dagger$ denote the optimal policy to the CMDP $\mathcal{C}_\mu(\tilde{r})$. Finally we define a shorthand $r^\dagger(s, a) := \tilde{r}(s, a) - (\lambda^\dagger_\delta + 1)c_\mu(s, a)$.

Now we bound the regret and the constraint violation of $\hat{\pi}$ in the CMDP $\mathcal{C}_\mu(\tilde{r})$. The following holds for any $\delta \geq 0$. For the regret, we can derive

$$\begin{aligned}
V^{\pi^\dagger}_{\tilde{r}}(d_0) - V^{\hat{\pi}}_{\tilde{r}}(d_0) &= V^{\pi^\dagger}_{\tilde{r}}(d_0) - (\lambda^\dagger_\delta + 1)V^{\pi^\dagger}_{c_\mu}(d_0) - V^{\hat{\pi}}_{\tilde{r}}(d_0) && (V^{\pi^\dagger}_{c_\mu}(d_0) = 0) \\
&\leq V^{\pi^\dagger}_{\tilde{r}}(d_0) - (\lambda^\dagger_\delta + 1)V^{\pi^\dagger}_{c_\mu}(d_0) - V^{\hat{\pi}}_{\tilde{r}}(d_0) + (\lambda^\dagger_\delta + 1)V^{\hat{\pi}}_{c_\mu}(d_0) && (V^{\hat{\pi}}_{c_\mu}(d_0) \geq 0) \\
&= V^{\pi^\dagger}_{r^\dagger}(d_0) - V^{\hat{\pi}}_{r^\dagger}(d_0).
\end{aligned}$$

Here we use the additivity of value function over reward, namely, for any reward functions r_1, r_2 and policy π, we have $V^{\pi}_{r_1} + V^{\pi}_{r_2} = V^{\pi}_{r_1+r_2}$. Similarly for the constraint violation, we can derive

$$\begin{aligned}
V^{\hat{\pi}}_{c_\mu}(d_0) &= (V^{\hat{\pi}}_{c_\mu}(d_0) - \delta) + \delta \\
&= (\lambda^\dagger_\delta + 1)(V^{\hat{\pi}}_{c_\mu}(d_0) - \delta) - \lambda^\dagger_\delta(V^{\hat{\pi}}_{c_\mu}(d_0) - \delta) + \delta \\
&= -V^{\hat{\pi}}_{\tilde{r}}(d_0) + (\lambda^\dagger_\delta + 1)(V^{\hat{\pi}}_{c_\mu}(d_0) - \delta) + V^{\hat{\pi}}_{\tilde{r}}(d_0) - \lambda^\dagger_\delta(V^{\hat{\pi}}_{c_\mu}(d_0) - \delta) + \delta \\
&= \mathcal{L}_\delta(\hat{\pi}, \lambda^\dagger_\delta) - \mathcal{L}_\delta(\hat{\pi}, \lambda^\dagger_\delta + 1) + \delta \\
&\le \mathcal{L}_\delta(\pi^\dagger_\delta, \lambda^\dagger_\delta + 1) - \mathcal{L}_\delta(\hat{\pi}, \lambda^\dagger_\delta + 1) + \delta \\
&= V^{\pi^\dagger_\delta}_{r^\dagger}(d_0) - V^{\hat{\pi}}_{r^\dagger}(d_0) + \delta \\
&\le V^{\pi^\dagger}_{r^\dagger}(d_0) - V^{\hat{\pi}}_{r^\dagger}(d_0) + \delta + \kappa(\delta)
\end{aligned}$$

where the first inequality is due to that $(\pi^\dagger_\delta, \lambda^\dagger_\delta)$ is a saddle point to $\mathcal{L}_\delta$ (see Definition C.2), and the second inequality follows from

$$\begin{aligned}
V^{\pi^\dagger_\delta}_{r^\dagger}(d_0) &= V^{\pi^\dagger_\delta}_{\tilde{r}}(d_0) - (\lambda^\dagger_\delta + 1)V^{\pi^\dagger_\delta}_{c_\mu}(d_0), \\
&\le V^{\pi^\dagger_\delta}_{\tilde{r}}(d_0) - (\lambda^\dagger_\delta + 1)V^{\pi^\dagger}_{c_\mu}(d_0) && \text{(using } V^{\pi^\dagger}_{c_\mu} \le V^{\pi^\dagger_\delta}_{c_\mu}\text{)} \\
&\le V^{\pi^\dagger}_{\tilde{r}}(d_0) + \kappa(\delta) - (\lambda^\dagger_\delta + 1)V^{\pi^\dagger}_{c_\mu}(d_0) && \text{(Definition of } \kappa.\text{)} \\
&\le V^{\pi^\dagger}_{r^\dagger}(d_0) + \kappa(\delta)
\end{aligned}$$

Next we bound on $V^{\pi^\dagger}_{r^\dagger}(d_0) - V^{\hat{\pi}}_{r^\dagger}(d_0)$. By Lemma C.13 and Assumption C.6, we know $\lambda^\dagger_\delta \le \frac{K}{\delta}$, so $r^\dagger(s,a) \in [-\frac{K}{\delta} - 2, 1]$. This and the definition of c_μ together imply that $r^\dagger \in \mathcal{R}_\mu(\tilde{r}, \frac{K}{\delta})$. Since the offline RL algorithm $Algo$ is $\frac{K}{\delta}$-admissible, we have $V^{\pi^\dagger}_{r^\dagger}(d_0) - V^{\hat{\pi}}_{r^\dagger}(d_0) \le \mathcal{E}(\pi^\dagger, \mu)$. Finally, we use the fact that $\pi^\dagger$ by definition satisfies $\sup_{s,a} \frac{d^{\pi^\dagger}(s,a)}{\mu(s,a)} < \infty$, so $\mathcal{E}(\pi^\dagger, \mu) = o(1)$. Thus, we have proved

$$\begin{aligned}
V^{\pi^\dagger}_{\tilde{r}}(d_0) - V^{\hat{\pi}}_{\tilde{r}}(d_0) &\le o(1) \\
V^{\hat{\pi}}_{c_\mu}(d_0) &\le o(1) + \delta + \kappa(\delta)
\end{aligned}$$

That is, $\hat{\pi} \in \Pi_\iota(\mathcal{C}_\mu(\tilde{r}))$ with $\iota = \delta + \kappa(\delta) + o(1)$.

□

C.4. Implicit Data Bias

Below we provide example conditions for data distributions to have a positive implicit bias.

Proposition C.10. *Under Assumption 2.1, the followings are true.*

1. *(RL setup) Suppose for any $\tilde{r}$ in $\tilde{\mathcal{R}}$, there is $h : \mathcal{S} \to \mathbb{R}$ such that*

$$|r(s,a) + \gamma \mathbb{E}_{s' \sim \mathcal{P}|s,a}[h(s)] - h(s) - \tilde{r}(s,a)| \le \begin{cases} \epsilon_1, & s, a \in supp(\mu) \\ \epsilon_2, & otherwise \end{cases}$$

   for some $\epsilon_1, \epsilon_2 < \infty$. Then μ is $\frac{1-\gamma}{2\epsilon_1}$-positively biased to $\tilde{\mathcal{R}}$.

2. *(IL setup) If $\mu = d^{\pi^*}$, then μ is ∞-positively biased with respect to $\tilde{\mathcal{R}} = \{\tilde{r} : \tilde{r} : \mathcal{S} \times \mathcal{A} \to [-1, 1]\}$. If $\mu = d^{\pi^e}$ such that $V^{\pi^*}_r(d_0) - V^{\pi^e}_r(d_0) \le \frac{\epsilon'}{1-\gamma}$ and π^e is deterministic, then μ is $\frac{1-\gamma}{\epsilon'}$-positively biased to the previous $\tilde{\mathcal{R}}$.*

3. *(Length bias) Suppose $V^{\pi}_{c_\mu}(d_0) \le \delta$ implies $V^*_r(d_0) - V^{\pi}_r(d_0) \le \frac{\epsilon' + O(\delta)}{1-\gamma}$. Then μ is $O\left(\frac{1-\gamma}{\epsilon'}\right)$-positively biased with respect to $\tilde{\mathcal{R}} = \{\tilde{r} : \tilde{r} : \mathcal{S} \times \mathcal{A} \to [-1, 1]\}$. Below we list some sufficient intervention conditions on data collection that leads to the length bias:*

(a) For all $(s,a) \in supp(\mu)$, $R_{\max} - r(s,a) \le \epsilon'$, where $R_{\max} = \sup_{s\in\mathcal{S},a\in\mathcal{A}} r(s,a) \le 1$.
(b) For all $(s,a) \in supp(\mu)$, $Q^*_r(s,a) - V^*_r(s) \le \epsilon'$.

The first example is the typical RL assumption, which says μ always has a positive bias to $\tilde{\mathcal{R}}$ if all rewards in $\tilde{\mathcal{R}}$ provide the same ranking of policies in the support (Ng et al., 1999). The second assumes the data is collectd by a single behavior policy, which includes the typical IL assumption that the data is collected by an optimal policy. Finally, the third example describes a length bias. Note that $V^{\pi}_{c_\mu}(s,a)$ measures the probability of going out of data support after taking a at s, which is disproportional to expected length the agent can survive. This condition assumes longer trajectories in the data to have smaller optimality gap.

This length bias condition is typically satisfied when intervention is taken in the data collection process (despite the data collection policy being suboptimal), as we saw in the motivating example in Figure 1 and Section 3.1, and empirically in Section 4. We give two sufficient conditions on such interventions. The first one is that the data collection is stopped when the instantaneous rewards is far from the maximum rewards, and the second is that the data collection agent does not take an action that no policy (even the optimal one) can perform well in the future. The consequence of these types of interventions is that longer trajectories perform better generally. These conditions are satisfied in `hopper` and `walker2d` dataset (except for -medium-replay) in Section 4, as well as in the grid world example in Section 3.1.

The proof of Proposition C.10 is based on the performance difference lemma, which can be proved by a telescoping sum decomposition.

Lemma C.11 (Performance Difference Lemma)**.** *Consider a discounted infinite-horizon MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, P, \gamma)$. For any policy π and any $h : \mathcal{S} \to \mathbb{R}$, it holds that*

$$V^{\pi}_r(d_0) - h(d_0) = \frac{1}{1-\gamma}\mathbb{E}_{s,a\sim d^{\pi}}[r(s,a) + \gamma\mathbb{E}_{s'\sim P|s,a}[h(s')] - h(s)].$$

In particular, this implies for any policies π, π',

$$V^{\pi}_r(d_0) - V^{\pi'}_r(d_0) = \frac{1}{1-\gamma}\mathbb{E}_{s,a\sim d^{\pi}}[Q^{\pi'}_r(s,a) - V^{\pi'}_r(s)].$$

C.4.1. Proof of RL setup in Proposition C.10

Proof. Given $\tilde{r} \in \tilde{\mathcal{R}}$, consider some $h : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ satisfying the assumption. Define

$$\hat{r}(s,a) = r(s,a) + \gamma\mathbb{E}_{s'\sim\mathcal{P}|s,a}[h(s)] - h(s).$$

Notice

$$\begin{aligned}
&(1-\gamma)(V^{\pi^*}_r(d_0) - V^{\pi}_r(d_0)) \\
&= (1-\gamma)(V^{\pi^*}_r(d_0) - h(d_0) - V^{\pi}_r(d_0) + h(d_0)) \\
&= \mathbb{E}_{s,a\sim d^{\pi^*}}[r(s,a) + \gamma\mathbb{E}_{s'\sim P|s,a}[h(s')] - h(s)] - \mathbb{E}_{s,a\sim d^{\pi}}[r(s,a) + \gamma\mathbb{E}_{s'\sim P|s,a}[h(s')] - h(s)] \\
&= \mathbb{E}_{s,a\sim d^{\pi^*}}[\hat{r}(s,a)] - \mathbb{E}_{s,a\sim d^{\pi}}[\hat{r}(s,a)] \\
&= (1-\gamma)(V^{\pi^*}_{\hat{r}}(d_0) - V^{\pi}_{\hat{r}}(d_0))
\end{aligned}$$

By Lemma C.11, for any $\pi \in \Pi^{\dagger}(\delta)$, it holds

$$\begin{aligned}
&(1-\gamma)(V^{\pi^*}_r(d_0) - V^{\pi}_r(d_0)) \\
&= (1-\gamma)(V^{\pi^*}_{\hat{r}}(d_0) - V^{\pi}_{\hat{r}}(d_0)) \\
&= \mathbb{E}_{s,a\sim d^{\pi^*}}[\hat{r}(s,a)] - \mathbb{E}_{s,a\sim d^{\pi}}[\hat{r}(s,a)] \\
&= \mathbb{E}_{s,a\sim d^{\pi^*}}[\tilde{r}(s,a)] - \mathbb{E}_{s,a\sim d^{\pi}}[\tilde{r}(s,a)] + \mathbb{E}_{s,a\sim d^{\pi^*}}[\hat{r}(s,a) - \tilde{r}(s,a)] - \mathbb{E}_{s,a\sim d^{\pi}}[\hat{r}(s,a) - \tilde{r}(s,a)] \\
&= (1-\gamma)(V^{\pi^*}_{\tilde{r}}(d_0) - V^{\pi}_{\tilde{r}}(d_0)) + \mathbb{E}_{s,a\sim d^{\pi^*}}[\hat{r}(s,a) - \tilde{r}(s,a)] - \mathbb{E}_{s,a\sim d^{\pi}}[\hat{r}(s,a) - \tilde{r}(s,a)]
\end{aligned}$$

Since π^* is assumed to be in the support of μ in Assumption 2.1, we have

$$\begin{aligned}
&\mathbb{E}_{s,a\sim d^{\pi^*}}[\hat{r}(s,a)-\tilde{r}(s,a)]-\mathbb{E}_{s,a\sim d^{\pi}}[\hat{r}(s,a)-\tilde{r}(s,a)]\\
&=\mathbb{E}_{s,a\sim d^{\pi^*}}[\hat{r}(s,a)-\tilde{r}(s,a)|(s,a)\in \text{supp}(\mu)]-\mathbb{E}_{s,a\sim d^{\pi}}[\hat{r}(s,a)-\tilde{r}(s,a)|(s,a)\in \text{supp}(\mu)]\\
&\quad-\mathbb{E}_{s,a\sim d^{\pi}}[\hat{r}(s,a)-\tilde{r}(s,a)|(s,a)\notin \text{supp}(\mu)]\\
&\le 2\epsilon_1-\sup_{s,a}|\hat{r}(s,a)-\tilde{r}(s,a)|\times\mathbb{E}_{s,a\sim d^{\pi}}[\mathbb{1}[s,a\notin \text{supp}(\mu)]]\\
&\le 2\epsilon_1+(1-\gamma)\epsilon_2 V^{\pi}_{c_\mu}(d_0)
\end{aligned}$$

Putting these two inequalities together, we have shown

$$(1-\gamma)(V_r^{\pi^*}(d_0)-V_r^{\pi}(d_0))\le(1-\gamma)(V_{\tilde{r}}^{\pi^*}(d_0)-V_{\tilde{r}}^{\pi}(d_0))+2\epsilon+\epsilon_2(1-\gamma)V^{\pi}_{c_\mu}(d_0)$$

Since $\pi\in\Pi^\dagger(\delta)$, it implies

$$(1-\gamma)(V_r^{\pi^*}(d_0)-V_r^{\pi}(d_0))\le(1-\gamma)\delta+2\epsilon_1+(1-\gamma)\epsilon_2\delta=2\epsilon_1+O(\delta)$$

Thus, the distribution is $\frac{1-\gamma}{2\epsilon_1}$-positively biased. □

C.4.2. Proof of IL setup in Proposition C.10

Proof. Let $\mu=d^{\pi^*}$. We can write

$$\begin{aligned}
&(1-\gamma)(V_r^{\pi^*}(d_0)-V_r^{\pi}(d_0))\\
&=\mathbb{E}_{d^\pi}[V_r^{\pi^*}(s)-Q_r^{\pi^*}(s,a)]\\
&=\mathbb{E}_{d^\pi}[V_r^{\pi^*}(s)-Q_r^{\pi^*}(s,a)|(s,a)\in \text{supp}(\mu)]+\mathbb{E}_{d^\pi}[V_r^{\pi^*}(s)-Q_r^{\pi^*}(s,a)|(s,a)\notin \text{supp}(\mu)]\\
&\le\delta
\end{aligned}$$

So the distribution is ∞-positively biased. For the case with the suboptimal expert π^e, we can derive similarly

$$\begin{aligned}
&(1-\gamma)(V_r^{\pi^*}(d_0)-V_r^{\pi}(d_0))\\
&=(1-\gamma)(V_r^{\pi^*}(d_0)-V_r^{\pi^e}(d_0))+(1-\gamma)(V_r^{\pi^e}(d_0)-V_r^{\pi}(d_0))\\
&\le\epsilon'+\mathbb{E}_{d^\pi}[V_r^{\pi^e}(s)-Q_r^{\pi^e}(s,a)]\\
&=\epsilon'+\mathbb{E}_{d^\pi}[V_r^{\pi^e}(s)-Q_r^{\pi^e}(s,a)|(s,a)\in \text{supp}(\mu)]+\mathbb{E}_{d^\pi}[V_r^{\pi^e}(s)-Q_r^{\pi^e}(s,a)|(s,a)\notin \text{supp}(\mu)]\\
&\le\epsilon'+\delta
\end{aligned}$$

Therefore, μ is $\frac{1-\gamma}{\epsilon'}$-positively biased. □

C.4.3. Proof of Length Bias in Proposition C.10

Item 3a

Proof. Suppose $R_{\max}-r(s,a)\le\epsilon'$ for $(s,a)\in\text{supp}(\mu)$.

$$\begin{aligned}
&(1-\gamma)(V_r^{\pi^*}(d_0)-V_r^{\pi}(d_0))\\
&=\mathbb{E}_{d^{\pi^*}}[r(s,a)]-\mathbb{E}_{d^\pi}[r(s,a)]\\
&=\mathbb{E}_{d^{\pi^*}}[r(s,a)]-\mathbb{E}_{d^\pi}[r(s,a)|(s,a)\in \text{supp}(\mu)]-\mathbb{E}_{d^\pi}[r(s,a)|(s,a)\notin \text{supp}(\mu)]\\
&\le R_{\max}-\mathbb{E}_{d^\pi}[(R_{\max}-\epsilon')|(s,a)\in \text{supp}(\mu)]-0\\
&=\mathbb{E}_{d^\pi}[\epsilon'|(s,a)\in \text{supp}(\mu)]+[R_{\max}|(s,a)\notin \text{supp}(\mu)]\\
&\le\epsilon'+(1-\gamma)R_{\max}\delta
\end{aligned}$$

□

Item 3b

Proof. Suppose $s, a \in \text{supp}(\mu)$ only if $V_r^*(s) - Q_r^*(s,a) \leq \epsilon'$. Then

$$\begin{aligned}
&(1-\gamma)(V_r^{\pi^*}(d_0) - V_r^{\pi}(d_0)) \\
&= \mathbb{E}_{d^\pi}[V_r^{\pi^*}(s) - Q_r^{\pi^*}(s,a)] \\
&= \mathbb{E}_{d^\pi}[V_r^{\pi^*}(s) - Q_r^{\pi^*}(s,a) | (s,a) \in \text{supp}(\mu)] + \mathbb{E}_{d^\pi}[V_r^{\pi^*}(s) - Q_r^{\pi^*}(s,a) | (s,a) \notin \text{supp}(\mu)] \\
&\leq \epsilon' + \delta
\end{aligned}$$

□

Note that more broadly, we can relax the above condition to that $Q_{c_\mu}^*(s,a) - V_{c_\mu}^*(s) < \alpha$ implies $V_r^*(s) - Q_r^*(s,a) \leq \beta + O(\alpha)$, for a potentially smaller β, where we recall that $V_{c_\mu}^*, Q_{c_\mu}^*$ are the optimal value functions to (minimizing) the cost c_μ and V_r^*, Q_r^* are the optimal value functions to (maximizing) the reawrd r. We can use the lemma below to prove the degree of positive bias is $\frac{1-\gamma}{\beta}$ in this more general case. The above is a special case for $\alpha = 1$, since $c_\mu(s,a) = 1$ when s, a is not in the support of μ.

Lemma C.12. *Define $\mathcal{A}_s^\alpha = \{a \in \mathcal{A} : Q_{c_\mu}^*(s,a) - V_{c_\mu}^*(s) < \alpha\}$. Suppose for all $s \in \mathcal{S}$, $a \in \mathcal{A}_s^\alpha$ satisfies $V_r^{\pi'}(s) - Q_r^{\pi'}(s,a) \leq \beta$. For π such that $V_{c_\mu}^\pi(d_0) \leq \delta$, it holds for any π'*

$$V_r^{\pi'}(d_0) - V_r^\pi(d_0) \leq \frac{\beta + \delta/\alpha}{1-\gamma}$$

Proof.

$$\begin{aligned}
(1-\gamma)(V_r^{\pi'}(d_0) - V_r^\pi(d_0)) &= \mathbb{E}_{d^\pi}[V_r^{\pi'}(s) - Q_r^{\pi'}(s,a)] \\
&= \mathbb{E}_{d^\pi}[V_r^{\pi'}(s) - Q_r^{\pi'}(s,a) | a \in \mathcal{A}_s^\alpha] + \mathbb{E}_{d^\pi}[V_r^{\pi'}(s) - Q_r^{\pi'}(s,a) | a \notin \mathcal{A}_s^\alpha] \\
&\leq \beta + \frac{1}{1-\gamma}\mathbb{E}_{d^\pi}[\mathbb{1}[a \notin \mathcal{A}_s^\alpha]]
\end{aligned}$$

On the other hand, we have

$$\begin{aligned}
(1-\gamma)\delta \geq (1-\gamma)V_{c_\mu}^\pi(d_0) &= (1-\gamma)(V_{c_\mu}^\pi(d_0) - V_{c_\mu}^*(d_0)) \\
&= \mathbb{E}_{d^\pi}[Q_{c_\mu}^*(s,a) - V_{c_\mu}^*(s) | a \in \mathcal{A}_s] + \mathbb{E}_{d^\pi}[Q_{c_\mu}^*(s,a) - V_{c_\mu}^*(s) | a \notin \mathcal{A}_s] \\
&\geq \mathbb{E}_{d^\pi}[Q_{c_\mu}^*(s,a) - V_{c_\mu}^*(s) | a \notin \mathcal{A}_s] \\
&\geq \alpha \mathbb{E}_{d^\pi}[\mathbb{1}[a \notin \mathcal{A}_s]]
\end{aligned}$$

Therefore,

$$(1-\gamma)(V_r^{\pi'}(d_0) - V_r^\pi(d_0)) \leq \beta + \frac{\delta}{\alpha}$$

□

C.5. Proof of Theorem C.7

Proof. By Proposition C.9, we have proved

$$\begin{aligned}
V_{\tilde{r}}^{\pi^\dagger}(d_0) - V_{\tilde{r}}^{\hat{\pi}}(d_0) &\leq o(1) \\
V_{c_\mu}^{\hat{\pi}}(d_0) &\leq o(1) + \delta + \kappa(\delta)
\end{aligned}$$

That is, $\hat{\pi} \in \Pi_\iota(\mathcal{C}_\mu(\tilde{r}))$ with $\iota = \delta + \kappa(\delta) + o(1)$.

Then by $\frac{1}{\epsilon}$-positively assumption on μ, we have the performance bound

$$V_r^{\pi^\dagger}(d_0) - V_r^{\hat{\pi}}(d_0) \leq O(\iota)$$

To bound of out of support probability, we derive

$$\begin{aligned}
V_{c_\mu}^{\hat{\pi}}(d_0) &= \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{d_t^{\hat{\pi}}}[\mathbb{1}[\mu(s,a)=0]] \\
&= (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \left(\sum_{\tau=0}^{t} \mathbb{E}_{d_\tau^{\hat{\pi}}}[\mathbb{1}[\mu(s,a)=0]] \right) \\
&\geq (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \mathrm{Prob}\left(\exists \tau \in [0,t] s_\tau \notin \mathrm{supp}(\mu) | \hat{\pi}\right)
\end{aligned}$$

where second equality follows (Wagener et al., 2021)[Lemma 2]. □

C.6. Technical Details of Constrained MDP

Lemma C.13. *For $g : \mathcal{S} \times \mathcal{A} \to [0, \infty)$, let $\mathcal{C}(\mathcal{S}, \mathcal{A}, f, g, P, \gamma)$ be a CMDP with sensitivity function $\kappa(\delta)$. Assume $\mathcal{C}(\mathcal{S}, \mathcal{A}, f, g, P, \gamma)$ is feasible. Consider a relaxed CMDP, $\max_\pi V_f^\pi(d_0)$, s.t. $V_g^\pi(d_0) \leq \delta$ and let $\lambda_\delta^\dagger \geq 0$ denote its optimal Lagrange multiplier. Then $\lambda_\delta^\dagger \leq \frac{\kappa(\delta)}{\delta}$.*

Proof. Denote the optimal policy of the relaxed CMDP as $\pi_\delta^\dagger$ and the optimal policy of the original CMDP as $\pi^\dagger$. By construction, $\pi_\delta^\dagger \in \Pi^\dagger(\delta)$. Therefore, we have $V_f^{\pi_\delta^\dagger}(d_0) - V_f^{\pi^\dagger}(d_0) = \max_{\pi \in \Pi^\dagger(\delta)} V_f^\pi(d_0) - V_f^{\pi^\dagger}(d_0) = \kappa(\delta)$.

To bound $\lambda_\delta^\dagger$, we use that the fact that $(\pi_\delta^\dagger, \lambda_\delta^\dagger)$ is a saddle point to the Lagrangian of the relaxed problem $\mathcal{L}_\delta(\pi, \lambda) := V_f^\pi(d_0) - \lambda(V_g^\pi(d_0) - \delta)$. As a result, we can derive

$$V_f^{\pi^\dagger}(d_0) - \lambda_\delta^\dagger(V_g^{\pi^\dagger}(d_0) - \delta) = \mathcal{L}_\delta(\pi^\dagger, \lambda_\delta^\dagger) \leq \mathcal{L}_\delta(\pi_\delta^\dagger, \lambda_\delta^\dagger) \leq \mathcal{L}_\delta(\pi_\delta^\dagger, 0) = V_f^{\pi_\delta^\dagger}(d_0).$$

Since $V_g^{\pi^\dagger}(d_0) = 0$, the above inequality implies

$$\lambda_\delta^\dagger \leq \frac{V_f^{\pi_\delta^\dagger}(d_0) - V_f^{\pi^\dagger}(d_0)}{\delta} = \frac{\kappa(\delta)}{\delta}$$

□

Lemma C.14. *Suppose there is $\lambda^\dagger \leq L$ with some $L < \infty$ such that $\lambda^\dagger \in \arg\min_{\lambda \geq 0} \max_\pi V_{\tilde{r}}^\pi(d_0) - \lambda V_{c_\mu}^\pi(d_0)$. Then $\kappa(\delta) \leq L\delta$.*

Proof. First we note by[12] (Nguyen and Cheng, 2023, Theorem 4.1) we know that $(\pi^\dagger, \lambda^\dagger)$ is a saddle-point to the CMDP $\mathcal{C}_\mu(\tilde{r}) = \mathcal{C}(\mathcal{S}, \mathcal{A}, \tilde{r}, c_\mu, P, \gamma)$ in (1). Let $f = \tilde{r}$ and $g = c_\mu$. Define the Lagrangian $\mathcal{L}_\delta(\pi, \delta) = V_f^\pi(d_0) - \lambda(V_g^\pi(d_0) - \delta)$ for the relaxed CMDP problem $\max_{\pi : V_g^\pi(d_0) \leq \delta} V_f^\pi(d_0)$. Consider a saddle-point $(\pi_\delta^\dagger, \lambda_\delta^\dagger)$ to this relaxed problem .

By duality, we can first derive

$$\max_{\pi \in \Pi^\dagger(\delta)} V_f^\pi(d_0) - V_f^{\pi^\dagger}(d_0) = \max_{\pi : V_g^\pi(d_0) \leq \delta} V_f^\pi(d_0) - V_f^{\pi^\dagger}(d_0) = \mathcal{L}_\delta(\pi_\delta^\dagger, \lambda_\delta^\dagger) - V_f^{\pi^\dagger}(d_0)$$

[12] We use this theorem because the CMDP here does not satisfy the Slater's condition.

Then we upper bound $\mathcal{L}_\delta(\pi^\dagger_\delta, \lambda^\dagger_\delta)$:

$$\begin{aligned}
\mathcal{L}_\delta(\pi^\dagger_\delta, \lambda^\dagger_\delta) &\leq \mathcal{L}_\delta(\pi^\dagger_\delta, \lambda^\dagger) \\
&= V_f^{\pi^\dagger_\delta}(d_0) - \lambda^\dagger(V_f^{\pi^\dagger_\delta}(d_0) - \delta) \\
&= V_f^{\pi^\dagger_\delta}(d_0) - \lambda^\dagger V_f^{\pi^\dagger_\delta}(d_0) + \delta\lambda^\dagger \\
&= \mathcal{L}(\pi^\dagger_\delta, \lambda^\dagger) + \delta\lambda^\dagger \\
&\leq \mathcal{L}(\pi^\dagger, \lambda^\dagger) + \delta\lambda^\dagger \\
&\leq \mathcal{L}(\pi^\dagger, 0) + \delta\lambda^\dagger.
\end{aligned}$$

Thus we have

$$\max_{\pi \in \Pi^\dagger(\delta)} V_f^\pi(d_0) - V_f^{\pi^\dagger}(d_0) \leq \mathcal{L}(\pi^\dagger, 0) + \delta\lambda^\dagger - V_f^{\pi^\dagger}(d_0) = V_f^{\pi^\dagger}(d_0) + \delta\lambda^\dagger - V_f^{\pi^\dagger}(d_0) = \delta\lambda^\dagger$$

□

D. Finite-Horizon Version

We discuss how to interpret Theorem C.7 in the finite-horizon setup.

Assumption D.1 (Data Assumption)**.**

1. For all $\tilde{r} \in \tilde{\mathcal{R}}$, $\tilde{r}(s,a) \in [-1, 1]$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$.
2. The data distribution μ is $\frac{1}{\epsilon}$-positively biased with respect to $\tilde{\mathcal{R}}$.
3. There is a policy π satisfying $V_{c_\mu} \leq 0$.
4. For any $\tilde{r} \in \tilde{\mathcal{R}}$, the CMDP $\mathcal{C}_\mu(\tilde{r}) := \mathcal{C}(\mathcal{S}, \mathcal{A}, \tilde{r}, c_\mu, P, \gamma)$ in (1) has a sensitivity function $\kappa(\delta) \leq K$ for some $K < \infty$ and $\lim_{\delta \to 0^+} \kappa(\delta) = 0$. In addition, its solution $\pi^\dagger$ satisfies $\sup_{s,a} \frac{d^{\pi^\dagger}(s,a)}{\mu(s,a)} < \infty$.

Theorem C.7 (Main Result)**.** *Under the data assumption in Assumption C.6, consider an offline RL algorithm Algo that is sufficiently pessimistic to be $(\frac{K}{\delta} + 2)$-admissible with respect to $\tilde{\mathcal{R}}$. Then for* any *$\tilde{r} \in \tilde{\mathcal{R}}$, with high probability, the policy $\hat{\pi}$ learned by Algo from the dataset $\tilde{\mathcal{D}} := \{(s, a, \tilde{r}, s') | (s,a) \sim \mu, \tilde{r} = \tilde{r}(s,a), s' \sim P(\cdot|s,a)\}$ has both performance guarantee*

$$V_r^{\pi^*}(d_0) - V_r^{\hat{\pi}}(d_0) \leq \epsilon + O(\iota)$$

and safety guarantee

$$(1-\gamma)\sum_{t=0}^{\infty} \gamma^t \text{Prob}\left(\exists \tau \in [0,t] s_\tau \notin \textit{supp}(\mu) | \hat{\pi}\right) \leq \iota$$

where $\iota := \delta + \kappa(\delta) + o(1)$ and $o(1)$ denotes a term that vanishes as the dataset size becomes infinite.

Notation First we the finite-horizon setup. To translate the previous notation to the finite-horizon setting, we suppose the state s contains time information and the state space is layered. That is, $\mathcal{S} = \bigcup_{0=1}^{H-1} \mathcal{S}_t$, where H is the problem horizon, and $\mathcal{S}_t$ denotes the set of states at time t. For example, for a trajectory $s_0, s_2, \ldots, s_{H-1}$ starting from $t = 0$, we have $s_t \in \mathcal{S}_t$, for $t \in [0, H-1]$. Therefore, we can use the previous notation to model time-varying functions naturally (needed in the finite horizon), without explicitly listing the time dependency. In this section, with abuse of notation, we define the value V_r^π of a policy π to reward r at $s \in \mathcal{S}_\tau$ as

$$V_r^\pi(s) := \mathbb{E}_{\pi,P}\left[\sum_{t=\tau}^{H-1} \tilde{r}(s_t, a_t) \mid s_\tau = s\right]$$

MDP and CMDP Problems The task MDP becomes $\mathcal{M}_H = (\mathcal{S}, \mathcal{A}, r, P, H)$ and solving it means

$$\max_{\pi} \mathbb{E}_{\pi,P}\left[\sum_{t=0}^{H-1} r(s_t, a_t) \mid s_0 \sim d_0\right]$$

and the CMDP problem in (1) becomes

$$\max_{\pi} \mathbb{E}_{\pi,P}\left[\sum_{t=0}^{H-1} \tilde{r}(s_t, a_t) \mid s_0 \sim d_0\right] \quad \text{s.t.} \quad \mathbb{E}_{\pi,P}\left[\sum_{t=0}^{H-1} c_\mu(s_t, a_t) \mid s_0 \sim d_0\right] \leq 0$$

D.1. Main Theorem for Finite-horizon Problems

Assumption D.2 (Data Assumption (Finite Horizon))**.**

1. For all $\tilde{r} \in \tilde{\mathcal{R}}$, $\tilde{r}(s,a) \in [-1, 1]$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$.
2. The data distribution μ is $\frac{1}{\epsilon}$-positively biased with respect to $\tilde{\mathcal{R}}$.
3. The solution to the CDMP $\pi^\dagger$ satisfies $\sup_{s,a} \frac{d^{\pi^\dagger}(s,a)}{\mu(s,a)} < \infty$, where we note that $d^{\pi^\dagger}$ is defined without discounts but as the average over the episode length.

Theorem D.3 (Main Result (Finite Horizon))**.** *Under the data assumption in Assumption C.6, consider an offline RL algorithm Algo that is sufficiently pessimistic to be $(\frac{K}{\delta} + 2)$-admissible with respect to $\tilde{\mathcal{R}}$. Then for* any *$\tilde{r} \in \tilde{\mathcal{R}}$, with high probability, the policy $\hat{\pi}$ learned by Algo from the dataset $\tilde{\mathcal{D}} := \{(s, a, \tilde{r}, s') | (s,a) \sim \mu, \tilde{r} = \tilde{r}(s,a), s' \sim P(\cdot|s,a)\}$ has both performance guarantee*

$$V_r^{\pi^*}(d_0) - V_r^{\hat{\pi}}(d_0) \leq \epsilon + O(\iota)$$

and safety guarantee

$$\text{Prob}\left(\exists \tau \in [0, H-1], s_\tau \notin \textit{supp}(\mu) | \hat{\pi}\right) \leq \iota$$

where $\iota := \delta + \kappa(\delta) + o(1)$ and $o(1)$ denotes a term that vanishes as the dataset size becomes infinite.

The main difference between the infinite-horizon setup and the finite-horizon setup is that the finite-horizon setup always satisfies the regularity assumption (the third point) in Assumption C.6. In addition, the safety guarantee is more explicit compared the discounted infinite-horizon counterpart. This is because $\text{Prob}\left(\exists \tau \in [0, H-1], s_\tau \notin \text{supp}(\mu) | \hat{\pi}\right) \leq V_{c_\mu}^{\hat{\pi}}(d_0)$ for the finite horizon, whereas we need an additional conversion in Appendix C.5.

We prove $\lim_{\delta \to 0^+} \kappa(\delta) = 0$ is always true for the finite horizon version.

Proposition D.4. *For the H-horizon constrained MDP of (1), we have an optimal dual variable $\lambda^\dagger = 2H + 1$ and $\kappa(\delta) \leq \min\{2H, (2H+1)\delta\}$.*

Proof. Define the Lagrange reward $r^\dagger(s,a) = \tilde{r}(s,a) - \lambda^\dagger c_\mu(s,a)$. Let $\hat{\pi}^\dagger$ denote the optimal policy to the Lagrange reward. Notice that $V_{c_\mu}^\pi(d_0) \geq 1$ for any π such that $V_{c_\mu}^\pi(d_0) > 0$. Therefore, with $\lambda^\dagger > 2H$,

$$\begin{aligned}
\max_{\pi: V_{c_\mu}^\pi(d_0) > 0} V_{\tilde{r}}^\pi(d_0) - \lambda^\dagger V_{c_\mu}^\pi(d_0) &\leq \max_{\pi: V_{c_\mu}^\pi(d_0) > 0} V_{\tilde{r}}^\pi(d_0) - \lambda^\dagger \\
&< -H \\
&\leq \max_{\pi: V_{c_\mu}^\pi(d_0) = 0} V_{\tilde{r}}^\pi(d_0) - \lambda^\dagger V_{c_\mu}^\pi(d_0)
\end{aligned}$$

Therefore, $\pi^\dagger$ satisfies $V_{c_\mu}^{\pi^\dagger}(d_0) = 0$. This implies

$$\max_{\pi} V_{\tilde{r}}^\pi(d_0) - \lambda^\dagger V_{c_\mu}^\pi(d_0) \geq \min_{\lambda \geq 0} \max_{\pi} V_{\tilde{r}}^\pi(d_0) - \lambda V_{c_\mu}^\pi(d_0) = V_{\tilde{r}}^{\pi^\dagger}(d_0)$$

On the other hand, it is always true that

$$\max_{\pi} V_{\tilde{r}}^{\pi}(d_0) - \lambda^{\dagger} V_{c_\mu}^{\pi}(d_0) \geq \min_{\lambda \geq 0} \max_{\pi} V_{\tilde{r}}^{\pi}(d_0) - \lambda V_{c_\mu}^{\pi}(d_0)$$

Therefore,

$$\lambda^{\dagger} \in \min_{\lambda \geq 0} \max_{\pi} V_{\tilde{r}}^{\pi}(d_0) - \lambda V_{c_\mu}^{\pi}(d_0)$$

Consider $\lambda^{\dagger} = 2H + 1$. By Lemma C.8, we have $\kappa(\delta) \leq (2H+1)\delta$. In addition $\kappa(\delta) \leq 2H$ by definition. □

D.2. Length Bias

For finite-horizon problems, we provide a simple sufficient condition for length bias, which assumes long trajectories obtained in data collections are near optimal.

Proposition D.5. *Suppose the data are generated by rolling out policies starting from $t = 0$ and the initials state distribution d_0. Let $\xi = (s_0, a_0, s_1, \ldots, s_{T_\xi - 1}, a_{T_\xi - 1})$ denote a trajectory of length T_ξ. If with probability one (over randomness of ξ) that $T_\xi = H$ implies*

$$\sum_{t=0}^{H-1} r(s_t, a_t) \geq V^{\pi^*}(d_0) - H\epsilon'$$

then the data distribution is $\frac{1}{H\epsilon'}$-positively biased.

Proof. Consider some $\pi \in \Pi^{\dagger}(\delta)$. We can derive

$$\begin{aligned}
V_r^{\pi*}(d_0) - V_r^{\pi}(d_0) &= V_r^{\pi*}(d_0) - \mathbb{E}_{\pi,P}\left[\sum_{t=0}^{H-1} r(s_t, a_t)\right] \\
&= \mathbb{E}_{\pi,P}\left[V_r^{\pi*}(d_0) - \sum_{t=0}^{H-1} r(s_t, a_t) \mid \exists \tau \in [0, H-1], s_\tau \notin \text{supp}(\mu)\right] \\
&\quad + \mathbb{E}_{\pi,P}\left[V_r^{\pi*}(d_0) - \sum_{t=0}^{H-1} r(s_t, a_t) \mid \forall \tau \in [0, H-1], s_\tau \in \text{supp}(\mu)\right] \\
&\leq H \times \text{Prob}\left(\exists \tau \in [0, H-1], s_\tau \notin \text{supp}(\mu) | \pi\right) + H\epsilon' \\
&\leq H \times V_{c_\mu}^{\pi}(d_0) + H\epsilon' \\
&\leq H\delta + H\epsilon'
\end{aligned}$$

□

D.3. Goal-oriented Problems

Finally we make a remark on positive data bias in the goal-oriented finite-horizon setting. In the infinite-horizon setting that a goal is marked as an absorbing state, which means that the agent once entering will stay there forever. The exact instantaneous (wrong) reward obtained at this absorbing state is not relevant (it can be anything in $[-1, 1]$) since the admissibility condition has already accounted for the range of the associated Lagrange reward. Namely, the agent is set pessimistic such that it views partial transitions/trajectories obtaining a worse return than $-\frac{1}{1-\gamma}$, which is a lower bound of the return the absorbing goal state.

To make sure the finite-horizon setting has the same kind of positive data bias, one way is to virtually extend the problem's original horizon (say H) by (as least) one and let the goal state to be the only state where the agent can stay until the last time step[13] H (i.e., all the other trajectories continue maximally up to time step $H - 1$ and therefore have a length at most

[13] We use zero-based numbering here where 0 is the initial step and H is the last step for a problem with horizon $H + 1$.

H). Then we apply the offline RL algorithm to this extended problem (e.g., of horizon $H+1$). We would need to set the horizon in the previous theoretical results accordingly for this longer horizon.

The reason for this extension is to ensure that the effect of absorbing state in the infinite-horizon setting (which ensures trajectories going into the absorbing state is by definition the longest) can carry over to the finite horizon case. If we apply the agent directly to solve the H step problem without such an extension, there may be other trajectories which can be as long as a goal-reaching one. As a result, there is no positive data bias.

An alternate way to create the positive data bias in the finite horizon setting is to truncate trajectories that do not reach goal at time step $H-1$ to be no longer than $H-1$. That is, they now time out at time step $H-2$, whereas only goal-reaching trajectories can continue up to time step $H-1$.

E. Algorithm Specific Results

In this appendix, we provide a few examples of admissible offline RL algorithms. We choose algorithms to cover both model-free (Cheng et al., 2022; Xie et al., 2021; Rashidinejad et al., 2021; Liu et al., 2020) and model-based (Xie et al., 2022; Bhardwaj et al., 2023; Uehara and Sun, 2022; Yu et al., 2020b; Kidambi et al., 2020) approaches. The pessimism in these algorithms are constructed through different ways, including adversarial training (Cheng et al., 2022; Xie et al., 2021; 2022; Bhardwaj et al., 2023; Uehara and Sun, 2022), value bonus (Rashidinejad et al., 2021; Yu et al., 2020b) and action truncation (Liu et al., 2020; Kidambi et al., 2020). These algorithms are listed in Table 15.

	Adversarial Training	Value Bonus	Action Truncation
Model-free	ATAC (Cheng et al., 2022) (Appendix E.1) PSPI (Xie et al., 2021) (Appendix E.2)	VI-LCB (Rashidinejad et al., 2021) (Appendix E.5)	PPI/PQI (Liu et al., 2020) (Appendix E.6)
Model-based	ARMOR (Xie et al., 2022; Bhardwaj et al., 2023) (Appendix E.3) CPPO (Uehara and Sun, 2022) (Appendix E.4)	MOPO (Yu et al., 2020b) (Appendix E.7)	MOReL (Kidambi et al., 2020) (Appendix E.7)

Table 15: We show that all of the offline RL algorithms above are admissible. Note that this is not a complete list of all admissible offline RL algorithms.

We recall the definition of admissibility below.

Definition E.1. [Admissibility] For $R \geq 0$, we say an offline RL algorithm $Algo$ is *R-admissible* with respect to $\tilde{\mathcal{R}}$, if for any $\tilde{r} \in \tilde{\mathcal{R}}$, given $\tilde{\mathcal{D}}$, $Algo$ learns a policy $\hat{\pi}$ satisfying the following with high probability: for any policy $\pi \in \Pi$,

$$\max_{\bar{r} \in \mathcal{R}_R(\tilde{r})} V_{\bar{r}}^{\pi}(d_0) - V_{\bar{r}}^{\hat{\pi}}(d_0) \leq \mathcal{E}(\pi, \mu),$$

where we define a data-consistent reward class

$$\mathcal{R}_R(\tilde{r}) := \{\bar{r} : \bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in \text{supp}(\mu) \text{ and } \bar{r}(s,a) \in [-R, 1], \forall (s,a) \in \mathcal{S} \times \mathcal{A}\},$$

$\mathcal{E}(\pi, \mu)$ is some regret upper bound such that $\mathcal{E}(\pi, \mu) = o(1)$ if $\sup_{s \in \mathcal{S}, a \in \mathcal{A}} \frac{d^{\pi}(s,a)}{\mu(s,a)} < \infty$, and $o(1)$ denotes a term that vanishes as the dataset size becomes infinite.

For the sake of clarity, we consider the tabular setting in our analysis. That is, we assume the state space $\mathcal{S}$ and the action space $\mathcal{A}$ are countable and finite. (We use $|\mathcal{S}|$ and $|\mathcal{A}|$ to denote their cardinalities). To establish admissibility, we sometimes make minor changes over the function classes in these algorithms, because some algorithms assume that the reward is known or has value bounded by $[0, 1]$. We highlight these changes in blue. We would like to clarify that the goal of our analysis is to prove that these algorithms are admissible rather than provide a tight bound for their performance. We recommend readers who are interested in performance bound to read the original papers, as their bound can be tighter than ours.

One remarkable technical details in our analysis is that there is no need of using an union bound over all rewards in $\mathcal{R}_R(\tilde{r})$ in the admissibility definition when proving these high probability statements. Instead we found that we can reuse the bound proved for a single reward directly. The main reason is that all rewards in the admissibility definition agree with each other on the support of the data distribution, and the statistical analysis of concentration only happens within this support. Outside of the support, a uniform bound based on the reward range can be used to bound the error. This will be made more clearly later in the derivations.

E.1. ATAC

We first show that ATAC (Cheng et al., 2022) is an admissible offline RL algorithm. We first introduce notations that we will use for analyzing ATAC, which will also be useful for studying PSPI in Appendix E.2.

E.1.1. NOTATIONS

For any $\mu, \nu \in \Delta(\mathcal{S} \times \mathcal{A})$, we denote $(\mu \setminus \nu)(s,a) \coloneqq \max(\mu(s,a) - \nu(s,a), 0)$. For any $\mu \in \Delta(\mathcal{S} \times \mathcal{A})$ and any $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, we define $\langle \mu, f \rangle \coloneqq \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mu(s,a) f(s,a)$. For an MDP with transition probability $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, we define $(\mathcal{P}^\pi f)(s,a) \coloneqq \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} f(s', \pi)$ for any $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $\pi : \mathcal{S} \times \Delta(\mathcal{A})$.

Given function class $\mathcal{F} \subseteq (\mathcal{S} \times \mathcal{A} \to \mathbb{R})$, policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ and reward $\bar{r} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, we introduce Bellman error transfer coefficient (Xie et al., 2021; Cheng et al., 2022) to measure the distribution shift between two probability measure μ and ν.

Definition E.2 (Bellman error transfer coefficient)**.** The Bellman error transfer coefficient between ν and μ under function class $\mathcal{F}$, policy π and reward $\bar{r}$ is defined as,

$$\mathscr{C}(\nu; \mu, \mathcal{F}, \pi, \bar{r}) \coloneqq \max_{f \in \mathcal{F}} \frac{\|f - \bar{r} - \mathcal{P}^\pi f\|_{2,\nu}^2}{\|f - \bar{r} - \mathcal{P}^\pi f\|_{2,\mu}^2}. \tag{3}$$

We note that the Bellman error transfer coefficient is a weaker notion than the density ratio, which is established in the lemma below.

Lemma E.3. *For any distributions $\nu, \mu \in \Delta(\mathcal{S} \times \mathcal{A})$, function class $\mathcal{F} \subseteq (\mathcal{S} \times \mathcal{A} \to \mathbb{R})$, policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ and reward $\bar{r} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, we have*

$$\mathscr{C}(\nu; \mu, \mathcal{F}, \pi, \bar{r}) \leq \left(\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{\nu(s,a)}{\mu(s,a)} \right). \tag{4}$$

Proof.

$$\begin{aligned}
\|f - \bar{r} - \mathcal{P}^\pi f\|_{2,\nu}^2 &= \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} \nu(s,a) \big(f(s,a) - \bar{r}(s,a) - \mathcal{P}^\pi f(s,a)\big)^2 \\
&= \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mu(s,a) \frac{\nu(s,a)}{\mu(s,a)} \big(f(s,a) - \bar{r}(s,a) - \mathcal{P}^\pi f(s,a)\big)^2 \\
&\leq \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mu(s,a) \left(\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{\nu(s,a)}{\mu(s,a)} \right) \big(f(s,a) - \bar{r}(s,a) - \mathcal{P}^\pi f(s,a)\big)^2 \\
&= \left(\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{\nu(s,a)}{\mu(s,a)} \right) \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mu(s,a) \big(f(s,a) - \bar{r}(s,a) - \mathcal{P}^\pi f(s,a)\big)^2 \\
&= \left(\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{\nu(s,a)}{\mu(s,a)} \right) \|f - \bar{r} - \mathcal{P}^\pi f\|_{2,\mu}^2
\end{aligned}$$

Therefore,

$$\begin{aligned}
\mathscr{C}(\nu; \mu, \mathcal{F}, \pi, \bar{r}) &= \max_{f \in \mathcal{F}} \frac{\|f - \bar{r} - \mathcal{P}^\pi f\|_{2,\nu}^2}{\|f - \bar{r} - \mathcal{P}^\pi f\|_{2,\mu}^2} \\
&\leq \left(\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{\nu(s,a)}{\mu(s,a)} \right) \max_{f \in \mathcal{F}} \frac{\|f - \bar{r} - \mathcal{P}^\pi f\|_{2,\mu}^2}{\|f - \bar{r} - \mathcal{P}^\pi f\|_{2,\mu}^2} \\
&= \sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{\nu(s,a)}{\mu(s,a)}
\end{aligned}$$

□

E.1.2. ANALYSIS

ATAC considers a critic function class $\mathcal{F}$ and a policy class Π. Since we consider the tabular setting, we choose $\mathcal{F} : (\mathcal{S} \times \mathcal{A} \to [-V_{\max}, V_{\max}])$[14] with $\frac{1}{1-\gamma} \leq V_{\max} < \infty$ and $\Pi : (\mathcal{S} \to \Delta(\mathcal{A}))$. ATAC formulates offline RL as the following Stackelberge game,

$$\begin{aligned} \hat{\pi} &\in \arg\max_{\pi \in \Pi} \mathcal{L}_{\tilde{\mathcal{D}}}(\pi, f^{\pi}) \\ \text{s.t.} \quad f^{\pi} &\in \arg\min_{f \in \mathcal{F}} \mathcal{L}_{\tilde{\mathcal{D}}}(\pi, f) + \beta \mathcal{E}_{\tilde{\mathcal{D}}}(\pi, f), \end{aligned} \tag{5}$$

with $\beta \geq 0$ being hyperparameter, and

$$\begin{aligned} \mathcal{L}_{\tilde{\mathcal{D}}}(\pi, f) &:= \mathbb{E}_{\tilde{\mathcal{D}}}\big[f(s,\pi) - f(s,a)\big], \\ \mathcal{E}_{\tilde{\mathcal{D}}}(\pi) &:= \mathbb{E}_{\tilde{\mathcal{D}}}\big[\big(f(s,a) - \tilde{r} - \gamma f(s',\pi)\big)^2\big] - \min_{f' \in \mathcal{F}} \mathbb{E}_{\tilde{\mathcal{D}}}\big[\big(f'(s,a) - \tilde{r} - \gamma f(s',\pi)\big)^2\big]. \end{aligned} \tag{6}$$

For any policy π, the critic f^{π} provides a relative pessimistic (with respect to the behavior policy μ) value estimate of the policy π. The hyperparameter β balances pessimism, given by $\mathcal{L}_{\tilde{\mathcal{D}}}(\pi, f)$, and Bellman consistency, given by $\mathcal{E}_{\tilde{\mathcal{D}}}(\pi)$. The learned policy $\hat{\pi}$ maximizes the relative pessimistic value estimate given by the critic.

We state the performance guarantee of ATAC with respect to any comparator policy π under the set of data-consistent reward functions $\{\bar{r} : \bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in \text{supp}(\mu) \text{ and } |\bar{r}(s,a)| \leq (1-\gamma)V_{\max}, \forall (s,a) \in \mathcal{S} \times \mathcal{A}\}$ in the following proposition. We make an additional assumption on the data distribution for ATAC, which is needed in its original proof.

Assumption E.4. For ATAC, we assume μ is the (mixture) average of state-action visitation distribution for d_0.

Proposition E.5. *Fix some $\tilde{r} : \mathcal{S} \times \mathcal{A} \to [-1, 1]$. Consider $\mathcal{F} : (\mathcal{S} \times \mathcal{A}) \to [-V_{max}, V_{max}]$ with $\frac{1}{1-\gamma} \leq V_{max} < \infty$ and $\Pi : (\mathcal{S} \to \Delta(\mathcal{A}))$. Let $\hat{\pi}$ be the solution to (5) and let $\pi \in \Pi$ be any comparator policy. Let $\nu \in \Delta(s,a)$ be an arbitrary distribution. Under Assumption E.4, for any $\delta \in (0,1]$, choosing $\beta = \tilde{\Theta}\left(\frac{1}{V_{max}} \sqrt[3]{\frac{C|\tilde{\mathcal{D}}|^2}{\left(|\mathcal{S}||\mathcal{A}| \log(1/\delta)\right)^2}}\right)$, with probability $1-\delta$, it holds that*

$$V_{\bar{r}}^{\pi}(d_0) - V_{\bar{r}}^{\hat{\pi}}(d_0) \leq \tilde{O}\left(\frac{V_{max}\big(\mathscr{C}(\nu; \mu, \mathcal{F}, \pi, \bar{r})|\mathcal{S}||\mathcal{A}| \log(1/\delta)\big)^{1/3}}{(1-\gamma)|\tilde{\mathcal{D}}|^{1/3}}\right) + \frac{\langle d^{\pi} \setminus \nu, \bar{r} + \mathcal{P}^{\pi} f^{\pi} - f^{\pi}\rangle}{1-\gamma} \tag{7}$$

for all $\bar{r}$ such that $\bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in supp(\mu)$ and $|\bar{r}(s,a)| \leq (1-\gamma)V_{max}, \forall (s,a) \in \mathcal{S} \times \mathcal{A}$.

Proof. We first show that $\mathcal{F} : (\mathcal{S} \times \mathcal{A} \to [-V_{\max}, V_{\max}])$ is both realizable and Bellman complete with repect to $\bar{r}$. For any $\bar{r}$ such that $|\bar{r}(s,a)| \leq (1-\gamma)V_{\max}$, we have $Q_{\bar{r}}^{\pi}(s,a) \in [-V_{\max}, V_{\max}]$. This means that $Q_{\bar{r}}^{\pi} \in \mathcal{F}$ for any $\pi \in \Pi$, $s \in \mathcal{S}$, $a \in \mathcal{A}$. Moreover, for any $f \in \mathcal{F}$, $|\bar{r}(s,a) + \mathcal{P}^{\pi} f(s,a)| \leq |\bar{r}(s,a)| + |\mathcal{P}^{\pi} f(s,a)| \leq (1-\gamma)V_{\max} + \gamma V_{\max} = V_{\max}$. In other words, $(\bar{r}(s,a) + \mathcal{P}^{\pi} f(s,a)) \in \mathcal{F}$ for any $f \in \mathcal{F}$.

By construction of $\bar{r}$, we have $\bar{\mathcal{D}} := \{(s,a,\bar{r},s') | (s,a,s') \in \mathcal{D}, \bar{r} = \bar{r}(s,a)\} \equiv \tilde{\mathcal{D}}$. That is, solving the Stackelberg game (5) given by $\tilde{\mathcal{D}}$ is equivalent to solving the game given by $\bar{\mathcal{D}}$. The rest of the proof follows from Theorem C.12 in (Li et al., 2023a) by choosing the reward class to only contain the reward $\bar{r}$, i.e., $\mathcal{G} = \{\bar{r}\}$ (which implies $d_{\mathcal{G}} = 0$) and using $d_{\mathcal{F},\Pi} = \tilde{O}(|\mathcal{S}||\mathcal{A}| \log(1/\delta))$.

Note that the above derivation does not need an additional union bound to cover all rewards in $\{\bar{r} : \bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in \text{supp}(\mu) \text{ and } |\bar{r}(s,a)| \leq (1-\gamma)V_{\max}, \forall (s,a) \in \mathcal{S} \times \mathcal{A}\}$. This is because the concentration analysis is only taken on the support of the data distribution, where all rewards in this reward class agree, and a uniform bound based on $V_{\max}$ is used for out of support places, which again applies to all the rewards in this reward class.

□

We then show that ATAC (Cheng et al., 2022) is admissible based on Proposition E.5.

[14] The original theoretical statement of ATAC assumes $\mathcal{F}$ to contain only non-negative functions. However, the analysis can be extended to any $\mathcal{F}$ with bounded value, such as $[-V_{\max}, V_{\max}]$. We note that the practical implementation of ATAC using function approximators does not make assumption on $\mathcal{F}$ being non-negative.

Corollary E.6 (ATAC is admissible)**.** *For any $V_{max} \geq \frac{R}{1-\gamma}$, ATAC is R-admissible with respect to $\tilde{\mathcal{R}}$ for any $\tilde{\mathcal{R}} \subseteq (\mathcal{S} \times \mathcal{A} \to [-1, 1])$.*

Proof. Given any $\tilde{r} \in \tilde{\mathcal{R}}$, we want to show that, with high probability, $V^{\pi}_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0) = o(1)$ for all $\bar{r} \in \mathcal{R}_R(\tilde{r})$ and all policy π such that $\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{d^{\pi}(s,a)}{\mu(s,a)} < \infty$. For any such π, define $C_\infty := \sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{d^{\pi}(s,a)}{\mu(s,a)}$, we have $\mathscr{C}(\nu; \mu, \mathcal{F}, \pi, \bar{r}) \leq C_\infty$ for any $\bar{r}$. By taking $\nu = d^{\pi}$ in Proposition E.5, we have, with probability $1-\delta$,

$$
\begin{aligned}
V^{\pi}_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0) &\leq \tilde{O}\left(\frac{V_{\max}\big(C_\infty|\mathcal{S}||\mathcal{A}|\log(1/\delta)\big)^{1/3}}{(1-\gamma)|\tilde{\mathcal{D}}|^{1/3}}\right) + \frac{\langle d^{\pi} \setminus d^{\pi}, \bar{r} + \mathcal{P}^{\pi} f^{\pi} - f^{\pi}\rangle}{1-\gamma} \\
&= \tilde{O}\left(\frac{V_{\max}\big(C_\infty|\mathcal{S}||\mathcal{A}|\log(1/\delta)\big)^{1/3}}{(1-\gamma)|\tilde{\mathcal{D}}|^{1/3}}\right) = o(1)
\end{aligned}
$$

for all $\bar{r}$ such that $\bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in \text{supp}(\mu)$ and $|\bar{r}(s,a)| \leq (1-\gamma)V_{\max}, \forall (s,a) \in \mathcal{S}\times\mathcal{A}$. Since $V_{\max} \geq \frac{R}{1-\gamma}$, we have that $V^{\pi}_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0) = o(1)$ for all $\bar{r} \in \mathcal{R}_R(\tilde{r})$.

□

E.2. PSPI

We show that PSPI (Xie et al., 2021) is also admissible. PSPI (Xie et al., 2021) is similar to ATAC. Given critic function class $\mathcal{F} : (\mathcal{S}\times\mathcal{A} \to [-V_{\max}, V_{\max}])$[15] with $\frac{1}{1-\gamma} \leq V_{\max} < \infty$ and policy class $\Pi : (\mathcal{S} \to \Delta(\mathcal{A}))$, PSPI solves the following Stackelberg game,

$$
\begin{aligned}
&\hat{\pi} \in \arg\max_{\pi\in\Pi} (1-\gamma) f^{\pi}(d_0, \pi) \\
\text{s.t.} \quad & f^{\pi} \in \arg\min_{f\in\mathcal{F}} (1-\gamma) f(d_0, \pi) + \beta \mathcal{E}_{\tilde{\mathcal{D}}}(\pi, f),
\end{aligned} \tag{8}
$$

with $\beta \geq 0$ being hyperparameter, and

$$
\mathcal{E}_{\tilde{\mathcal{D}}}(\pi) := \mathbb{E}_{\tilde{\mathcal{D}}}\big[\big(f(s,a) - \tilde{r} - \gamma f(s', \pi)\big)^2\big] - \min_{f'\in\mathcal{F}} \mathbb{E}_{\tilde{\mathcal{D}}}\big[\big(f'(s,a) - \tilde{r} - \gamma f(s', \pi)\big)^2\big].
$$

In PSPI, the critic f^{π} provides an absolute pessimistic value estimate of policy π. The hyperparameter β trades off pessimism and Bellman-consistency. The learned policy $\hat{\pi}$ maximizes such a pessimistic value. We state the performance guarantee of the learned policy $\hat{\pi}$ with respect to any comparator policy π under the set of data-consistent reward functions $\{\bar{r} : \bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in \text{supp}(\mu) \text{ and } |\bar{r}(s,a)| \leq (1-\gamma)V_{\max}, \forall (s,a) \in \mathcal{S}\times\mathcal{A}\}$ in the following proposition.

Proposition E.7 ((Xie et al., 2021; Li et al., 2023a))**.** *Let $\hat{\pi}$ be the solution to (5) and let $\pi \in \Pi$ be any comparator policy. Let $\nu \in \Delta(s,a)$ be an arbitrary distribution. For any $\delta \in (0,1]$, choosing $\beta = \tilde{\Theta}\left(\frac{1}{V_{max}}\sqrt[3]{\frac{C|\tilde{\mathcal{D}}|^2}{\big(|\mathcal{S}||\mathcal{A}|\log(1/\delta)\big)^2}}\right)$, with probability $1-\delta$,*

$$
V^{\pi}_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0) \leq \tilde{O}\left(\frac{V_{max}\big(\mathscr{C}(\nu; \mu, \mathcal{F}, \pi, \bar{r})|\mathcal{S}||\mathcal{A}|\log(1/\delta)\big)^{1/3}}{(1-\gamma)|\tilde{\mathcal{D}}|^{1/3}}\right) + \frac{\langle d^{\pi} \setminus \nu, \bar{r} + \mathcal{P}^{\pi} f^{\pi} - f^{\pi}\rangle}{1-\gamma} \tag{9}
$$

for all $\bar{r}$ such that $\bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in supp(\mu)$ and $|\bar{r}(s,a)| \leq (1-\gamma)V_{max}, \forall (s,a) \in \mathcal{S}\times\mathcal{A}$.

Proof. The proof is similar to that of Proposition E.5. First, observe that $\mathcal{F} : (\mathcal{S}\times\mathcal{A} \to [-V_{\max}, V_{\max}])$ is both realizable and Bellman complete with respect to $\bar{r}$. The rest of the proof follows from Theorem D.1 in (Li et al., 2023a) (taking $\mathcal{G} = \{\bar{r}\}$) and Lemma E.3. □

[15] Similar to ATAC, we extend the critic function class to contain functions that can take negative values. The theoretical analysis can be generalized to this case. The practical implementation of PSPI can directly handle critics which take negative values.

The proposition above implies that PSPI (Xie et al., 2021) is admissible. We omit the proof of the corollary below as it is the same as the proof of Corollary E.6.

Corollary E.8 (PSPI is admissible)**.** *For any $V_{max} \geq \frac{R}{1-\gamma}$, PSPI is R-admissible with respect to $\tilde{\mathcal{R}}$ for any $\tilde{\mathcal{R}} \subseteq (\mathcal{S} \times \mathcal{A} \to [-1, 1])$.*

E.3. ARMOR

We show that ARMOR (Xie et al., 2022; Bhardwaj et al., 2023) is admissible. ARMOR is a model-based offline RL algorithm. We denote a model as $M = (\mathcal{S}, \mathcal{A}, P_M, r_M, \gamma)$, where $P_M : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the model dynamics, and $r_M : \mathcal{S} \times \mathcal{A} \to [-R_{\max}, R_{\max}]$[16] is the reward function with $1 \leq R_{\max} < \infty$. Since we consider the tabular setting, we use a model class that contains all possible dynamics and all reward functions bounded within $[-R_{\max}, R_{\max}]$, i.e., $\mathcal{M}_{model} = \{M : P_M \in (\mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})), r_M \in (\mathcal{S} \times \mathcal{A} \to [-R_{\max}, R_{\max}])\}$. For any reference policy $\pi_{\text{ref}} : \mathcal{S} \to \Delta(\mathcal{A})$, ARMOR solves the following two-player game,

$$\hat{\pi} = \arg\max_{\pi \in \Pi} \min_{M \in \mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}}} V^{\pi}_M(d_0) - V^{\pi_{\text{ref}}}_M(d_0) \tag{10}$$

where $V^{\pi}_M(d_0) := \mathbb{E}_{\pi, P_M}[\sum_{t=0}^{\infty} \gamma^t r_M(s_t, a_t) | s_0 \sim d_0]$, and

$$\mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}} := \{M \in \mathcal{M}_{model} : \mathcal{E}_{\tilde{\mathcal{D}}}(M) - \min_{M' \in \mathcal{M}_{model}} \mathcal{E}_{\tilde{\mathcal{D}}}(M') \leq \alpha\}, \tag{11}$$

with

$$\mathcal{E}_{\tilde{\mathcal{D}}}(M) := \sum_{(s,a,\tilde{r},s') \in \tilde{\mathcal{D}}} -\log P_M(s'|s,a) + (r_M(s,a) - \tilde{r})^2. \tag{12}$$

Intuitively, ARMOR constructs a relative pessimistic model (with respect to π_{ref}) M which is also approximately consistent with data. The learned policy $\hat{\pi}$ maximizes the value estimate given by the pessimistic model. We define the generalized single policy concentrability to characterize the distribution shift from μ to the state-action visitation d^{π} of any policy π.

Definition E.9 (Generalized Single-policy Concentrability (Bhardwaj et al., 2023))**.** We define the generalized single-policy concentration for policy π, model class $\mathcal{M}_{model}$, reward $\bar{r}$ and data distribution μ as

$$\mathscr{C}_{model}(\pi; \mathcal{M}_{model}, \bar{r}) := \sup_{M \in \mathcal{M}_{model}} \frac{\mathbb{E}_{d^{\pi}}[\mathcal{E}(M; \bar{r})]}{\mathbb{E}_{\mu}[\mathcal{E}(M; \bar{r})]}$$

with $\mathcal{E}(M; \bar{r}) = D_{TV}(P_M(\cdot|s,a), P(\cdot|s,a))^2 + (r_M(s,a) - \bar{r}(s,a))^2$.

The single-policy concentrability $\mathscr{C}_{model}(\pi)$ can be considered as a model-based version of the Bellman error transfer coefficient in Definition E.2. Following the same steps as the proof of Lemma E.3, it can be shown that $\mathscr{C}_{model}(\pi; \mathcal{M}_{model}, \bar{r}) \leq \sup_{(s,a) \in \mathcal{S} \times \mathcal{A}} \frac{d^{\pi}(s,a)}{\mu(s,a)}$. We provide the high-probability performance guarantee for the learned policy $\hat{\pi}$. This implies that ARMOR is an admissible algorithm with a sufficiently large $R_{\max}$.

Proposition E.10. *For any $\delta \in (0, 1]$, there exists an absolute constant c such that when choosing $\alpha = c|\mathcal{S}|^2|\mathcal{A}| \log(1/\delta)$, for any comparator policy $\pi \in \Pi$, with probability $1 - \delta$, the policy $\hat{\pi}$ learned by ARMOR (10) satisfies*

$$V^{\pi}_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0) \leq O\left(\sqrt{\mathscr{C}_{model}(\pi; \mathcal{M}_{model}, \bar{r})} \frac{R_{max}}{(1-\gamma)^2} \sqrt{\frac{|\mathcal{S}|^2|\mathcal{A}| \log(1/\delta)}{|\tilde{\mathcal{D}}|}}\right), \tag{13}$$

for all $\bar{r}$ such that $\bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in supp(\mu)$ and $|\bar{r}(s,a)| \leq R_{max}, \forall (s,a) \in \mathcal{S} \times \mathcal{A}$.

Proof. By construction, we have $\bar{M} = (\mathcal{S}, \mathcal{A}, P, \bar{r}, \gamma) \in \mathcal{M}_{model}$. Since $\bar{r}(s,a) = \tilde{r}(s,a)$ for any $(s,a) \in \text{supp}(\mu)$, we have $\mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}} \equiv \mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}}$. The rest of the proof follows mostly from Theorem 2 in (Bhardwaj et al., 2023) by taking $\pi_{\text{ref}} = \mu$. By a similar argument as in the proof of Proposition E.5, we do not need an additional union bound to cover all rewards in $\{\bar{r} : \bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in \text{supp}(\mu) \text{ and } |\bar{r}(s,a)| \leq R_{\max}, \forall (s,a) \in \mathcal{S} \times \mathcal{A}\}$. □

[16] The theoretical statement of ARMOR assumes the value of r_M is bounded by $[0, 1]$. The original analysis can be extended as long as the value of r_M is bounded by a finite value $R_{\max} < \infty$. The practical implementation of ARMOR does not assume the reward only takes value in $[0, 1]$.

Corollary E.11 (ARMOR is admissible)**.** *For any $R_{max} \geq R$, ARMOR is R-admissible with respect to $\tilde{\mathcal{R}}$ for any $\tilde{\mathcal{R}} \subseteq (\mathcal{S} \times \mathcal{A} \to [-1, 1])$.*

Proof. Given any $\tilde{r} \in \tilde{\mathcal{R}}$, we want to show that, with high probability, $V^{\pi}_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0) = o(1)$ for all $\bar{r} \in \mathcal{R}_R(\tilde{r})$ and all policy π such that $\sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{d^{\pi}(s,a)}{\mu(s,a)} < \infty$. For any such π, $\mathscr{C}_{model}(\pi; \mathcal{M}_{model}, \bar{r}) \leq \sup_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{d^{\pi}(s,a)}{\mu(s,a)} := C_{\infty} < \infty$. By Proposition E.10, with probability $1-\delta$, we have

$$V^{\pi}_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0) \leq O\left(\sqrt{C_{\infty}} \frac{R_{\max}}{(1-\gamma)^2} \sqrt{\frac{|\mathcal{S}|^2|\mathcal{A}| \log(1/\delta)}{|\tilde{\mathcal{D}}|}}\right) = o(1). \tag{14}$$

for all $\bar{r}$ such that $\bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in \text{supp}(\mu)$ and $|\bar{r}(s,a)| \leq R, \forall (s,a) \in \mathcal{S} \times \mathcal{A}$. Therefore, $V^{\pi}_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0) = o(1)$ for all $\bar{r} \in \mathcal{R}_R(\tilde{r})$ with high probability. □

E.4. CPPO

CPPO(Uehara and Sun, 2022) is another admissible model-based offline algorithm. Again, we denote a model as $M = (\mathcal{S}, \mathcal{A}, P_M, r_M, \gamma)$, where $P_M : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the model dynamics, and $r_M : \mathcal{S} \times \mathcal{A} \to [-R_{\max}, R_{\max}]$[17] is the reward function with $1 \leq R_{\max} < \infty$. CPPO solves the two-player game,

$$\hat{\pi} = \arg\max_{\pi \in \Pi} \min_{M \in \mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}}} V^{\pi}_M(d_0) \tag{15}$$

with $V^{\pi}_M(d_0) := \mathbb{E}_{\pi, P_M}[\sum_{t=0}^{\infty} \gamma^t r_M(s_t, a_t) | s_0 \sim d_0]$ and $\mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}}$ defined in (11). CPPO constructs a pessimistic model M which is also approximately consistent with data. The learned policy $\hat{\pi}$ maximizes the value estimate given by the pessimistic model. We provide the high-probability performance guarantee for the learned policy $\hat{\pi}$. This implies that CPPO is an admissible algorithm with sufficiently large $R_{\max}$.

Proposition E.12. *For any $\delta \in (0, 1]$, there exists an absolute constant c such that when choosing $\alpha = c|\mathcal{S}|^2|\mathcal{A}| \log(1/\delta)$, for any comparator policy $\pi \in \Pi$, with probability $1-\delta$, the policy $\hat{\pi}$ learned by CPPO (15) satisfies*

$$V^{\pi}_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0) \leq O\left(\sqrt{\mathscr{C}_{model}(\pi; \mathcal{M}_{model}, \bar{r})} \frac{R_{max}}{(1-\gamma)^2} \sqrt{\frac{|\mathcal{S}|^2|\mathcal{A}| \log(1/\delta)}{|\tilde{\mathcal{D}}|}}\right), \tag{16}$$

for all $\bar{r}$ such that $\bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in supp(\mu)$ and $|\bar{r}(s,a)| \leq R_{max}, \forall (s,a) \in \mathcal{S} \times \mathcal{A}$.

Proof. By construction, we have $\bar{M} = (\mathcal{S}, \mathcal{A}, P, \bar{r}, \gamma) \in \mathcal{M}_{model}$. Since $\bar{r}(s,a) = \tilde{r}(s,a)$ for any $(s,a) \in \text{supp}(\mu)$, we have $\mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}} \equiv \mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}}$. The rest of the proof follows similarly from the proof of Theorem 2 in (Bhardwaj et al., 2023). By Lemma 5 from (Bhardwaj et al., 2023), with high probability, $\bar{M} \in \mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}}$. Therefore,

$$\begin{aligned}
V^{\pi}_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0) = V^{\pi}_{\bar{M}}(d_0) - V^{\hat{\pi}}_{\bar{M}}(d_0) &\leq V^{\pi}_{\bar{M}}(d_0) - \min_{M \in \mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}}} \left(V^{\hat{\pi}}_M(d_0)\right) \\
&\leq V^{\pi}_{\bar{M}}(d_0) - \min_{M \in \mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}}} (V^{\pi}_M(d_0)) \\
&\leq \max_{M \in \mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}}} |V^{\pi}_{\bar{M}}(d_0) - V^{\pi}_M(d_0)|
\end{aligned}$$

where the second inequality follows from the optimality of $\hat{\pi}$. By (20) from (Bhardwaj et al., 2023), for any $M \in \mathcal{M}^{\alpha}_{\tilde{\mathcal{D}}}$, $|V^{\pi}_{\bar{M}}(d_0) - V^{\pi}_M(d_0)| \leq O\left(\sqrt{\mathscr{C}_{model}(\pi; \mathcal{M}_{model}, \bar{r})} \frac{R_{\max}}{(1-\gamma)^2} \sqrt{\frac{|\mathcal{S}|^2|\mathcal{A}| \log(1/\delta)}{|\tilde{\mathcal{D}}|}}\right)$. Following a similar argument to the proof of Proposition E.5, there is no need to take the union bound with respect to $\bar{r}$, which concludes the proof. □

We omit the proof for the following corollary as it is the same of the proof of Corollary E.11.

Corollary E.13 (CPPO is admissible)**.** *For any $R_{max} \geq R$, CPPO is R-admissible with respect to $\tilde{\mathcal{R}}$ for any $\tilde{\mathcal{R}} \subseteq (\mathcal{S} \times \mathcal{A} \to [-1, 1])$.*

Algorithm 1 VI-LCB (Rashidinejad et al., 2021)

Input: Batch dataset $\tilde{\mathcal{D}}$, discount factor γ, confidence level δ, and value range $V_{\max}$.
Set $J := \frac{\log N}{1-\gamma}$.
Randomly split $\tilde{\mathcal{D}}$ into $J+1$ sets $\tilde{\mathcal{D}}_t = \{(s_i, a_i, r_i, s_i')\}_{i=1}^m$ for $j \in \{0, 1, \cdots, J\}$ with $m = \frac{N}{J+1}$.
Set $m_0(s,a) := \sum_{i=1}^m \mathbb{1}\{(s_i, a_i) = (s,a)\}$ based on dataset $\tilde{\mathcal{D}}_0$.
For all $a \in \mathcal{A}$ and $s \in \mathcal{S}$, initialize $Q_0(s,a) = -V_{\max}$, $V_0(s) = -V_{\max}$ and set $\pi_0(s) = \arg\max_a m_0(s,a)$.
for $j = 1, \cdots, J$ **do**
  Initialize $r_t(s,a) = 0$ and set $P_{s,a}^j$ to be a random probability vector.
  Set $m_t(s,a) := \sum_{i=1}^m \mathbb{1}\{(s_i, a_i) = (s,a)\}$ based on dataset $\tilde{\mathcal{D}}_t$.
  Compute penalty $b_t(s,a)$ for $L = 2000 \log(2(J+1)|\mathcal{S}||\mathcal{A}|/\delta)$

$$b_t(s,a) := 2V_{\max}\sqrt{\frac{L}{\max(m_t(s,a), 1)}}.$$

  for $(s,a) \in (\mathcal{S}, \mathcal{A})$ **do**
    if $m_t(s,a) \geq 1$ **then**
      Set $P_{s,a}^j$ to be empirical transitions and $r_t(s,a)$ be empirical average of rewards.
    end if
    Set $Q_j(s,a) \leftarrow r_t(s,a) - b_t(s,a) + \gamma P_{s,a}^j V_{j-1}$.
  end for
  Compute $V_t^{mid}(s) \leftarrow \max_a Q_t(s,a)$ and $\pi_t^{mid}(s) \in \arg\max_a Q_t(s,a)$.
  for $s \in \mathcal{S}$ **do**
    if $V_t^{mid}(s) \leq V_{j-1}(s)$ **then**
      $V_t(s) \leftarrow V_{j-1}(s)$ and $\pi_t(s) \leftarrow \pi_{j-1}(s)$.
    else
      $V_t(s) \leftarrow V_t^{mid}(s)$ and $\pi_t(s) \leftarrow \pi_t^{mid}(s)$.
    end if
  end for
end for
Return $\hat{\pi} = \pi_J$

E.5. VI-LCB

We show VI-LCB (Rashidinejad et al., 2021) is admissible. VI-LCB is a pessimistic version of value iteration. It adds negative bonuses to the Bellman backup step in order to underestimate the value when there are missing data. By being pessimistic in value estimation, it can overcome the issue of μ being non-exploratory. We recap the VI-LCB algorithm in Algorithm 1. We highlight the changes we make in blue. These changes are due to that originally the authors in (Rashidinejad et al., 2021) assume the rewards are non-negative, but the rewards in the admissibility definition Definition C.4 can take negative values. Therefore, we make these changes accordingly. We state and prove the guarantee below.[18]

Proposition E.14. *For $V_{\max} \geq \frac{R}{1-\gamma}$, VI-LCB is R-admissible with respect to any $\tilde{\mathcal{R}} \subseteq (\mathcal{S} \times \mathcal{A} \to [-1, 1])$.*

Proof. First, we introduce a factor of 2 in their proof of Lemma 1 (Rashidinejad et al., 2021) to accomodate that the rewards here can be negative. This is reflected in the updated bonus definition in Algorithm 1. This updated Lemma 1 of (Rashidinejad et al., 2021) now captures the good event that the bonus can upper bound the error of the empirical estimate of Bellman backup. It shows this good event is true with high probability.

We remark that while originally Lemma 1 of (Rashidinejad et al., 2021) is proved for a single reward function, it actually holds for simultaneously for all the reward functions in $\mathcal{R}_R(\tilde{r})$, without the need of introducing additionally a union bound. The reason is that in the proof of Lemma 1 in (Rashidinejad et al., 2021), the concentration is used only for the estimating

[17]The original algorithm assumes that the reward function is known. Here we consider the generalization of the algorithm with unknown reward.

[18]We referenced to lemmas and equations based on the version arXiv:2103.12021.

Algorithm 2 Pessimistic Policy Iteration (PPI) (Liu et al., 2020)

1: **Input:** Batch dataset $\tilde{\mathcal{D}}$, function class $\mathcal{F}$, probability estimator $\hat{\mu}$, hyperparameter b.
2: **for** $i \in \{0, \dots, I-1\}$ **do**
3: **for** $j \in \{0, \dots, J\}$ **do**
4: $f_{i,j+1} \leftarrow \arg\min_{f \in \mathcal{F}} \mathcal{L}_{\tilde{D}}(f, f_{i,j}, \hat{\pi}_i)$
5: **end for**
6: $\hat{\pi}_{i+1} \leftarrow \arg\max_{\pi} \mathbb{E}_{\tilde{\mathcal{D}}}[\mathbb{E}_{\pi}[\zeta \circ f_{i,J+1}]$
7: **end for**
8: **Return** $\hat{\pi} = \hat{\pi}_I$

Algorithm 3 Pessimistic Q Iteration (PQI) (Liu et al., 2020)

1: **Input:** Batch dataset $\tilde{\mathcal{D}}$, function class $\mathcal{F}$, probability estimator $\hat{\mu}$, hyperparameter b.
2: **for** $i \in \{0, \dots, I-1\}$ **do**
3: $f_{i+1} \leftarrow \arg\min_{f \in \mathcal{F}} \mathcal{L}_{\tilde{D}}(f, f_i)$
4: $\hat{\pi}_{i+1}(s) \leftarrow \arg\max_a \zeta \circ f_{i+1}(s, a)$
5: **end for**
6: **Return** $\hat{\pi} = \hat{\pi}_I$

the Bellman on the data support. This bound would apply to all rewards in $\mathcal{R}_R(\tilde{r})$ since they agree exactly on the data. For state-action pairs out of the support, the proof takes a uniform bound based on the size of $V_{\max}$, which also applies to all rewards in $\mathcal{R}_R(\tilde{r})$.

Therefore, we can apply the updated Lemma 1 of (Rashidinejad et al., 2021) to prove the desired high probability bound needed in the admissibility condition. Under this good event, we can use the upper bound in (54b) in (Rashidinejad et al., 2021) to bound the regret. Consider some π such that $C := \sup_{s,a} \frac{d^{\pi(s,a)}}{\mu(s,a)} < \infty$. Suppose $\tilde{D}$ has N transitions. Take some $V_{\max} \geq \frac{R}{1-\gamma}$. Then running VI-LCB with J iterations ensures[19] for any $\bar{r} \in \mathcal{R}_R(\tilde{r})$,

$$V_{\bar{r}}^{\pi}(d_0) - V_{\bar{r}}^{\hat{\pi}}(d_0) \leq \gamma^J 2V_{\max} + \frac{64V_{\max}}{1-\gamma}\sqrt{\frac{L|\mathcal{S}||\mathcal{A}|C(J+1)}{N}}$$

where we mark the changes due to using rewards which can takes negative values in blue. Setting $J = \frac{\log N}{1-\gamma}$ as in (Rashidinejad et al., 2021), with probability $1-\delta$, it holds that

$$\begin{aligned} V_{\bar{r}}^{\pi}(d_0) - V_{\bar{r}}^{\hat{\pi}}(d_0) &\leq \tilde{O}\left(\gamma^J V_{\max} + \frac{V_{\max}}{1-\gamma}\sqrt{\frac{L|\mathcal{S}||\mathcal{A}|CJ}{N}}\right) \\ &= \tilde{O}\left(V_{\max}\sqrt{\frac{|\mathcal{S}||\mathcal{A}|C\log\frac{1}{\delta}}{N(1-\gamma)^3}}\right) \end{aligned}$$

Thus, VI-LCB is R-admissible. □

E.6. PPI and PQI

We show PPI and PQI proposed in (Liu et al., 2020) are admissible. We present their algorithms in Algorithm 2 and Algorithm 3, with $(\zeta \circ f)(s,a) := \zeta(s,a)f(s,a)$ for any $\zeta, f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. These algorithms use a probability estimator of the data distribution (denoted as $\hat{\mu}$), which in the tabular case is the empirical distribution. Based on $\hat{\mu}$, they define a filter function

$$\zeta(s, a; \hat{\mu}, b) = \mathbb{1}[\hat{\mu}(s,a) \geq b]$$

This filter function classifies whether a state-action pair is in the support, and it is used to modify the Bellman operators in dynamics programming as shown in line 6 of Algorithm 2 and line 4 of Algorithm 3. As a result, the Bellman backup and

[19] We add back an extra dependency on $|\mathcal{A}|$ as their original proof assumes π is deterministic, which is not necessarily the case here.

the policy optimization only consider in-support actions, which mitigates the issue of learning with non-exploratory μ. The loss functions (i.e., $\mathcal{L}_{\tilde{\mathcal{D}}}$) in Algorithm 2 and Algorithm 3 denote the squared Bellman error (with a target network) as used in fitted Q iteration (Riedmiller, 2005). We omit the details here.

In summary, PPI and PQI follow the typical policy iteration and value iteration schemes, except that the backup and the argmax are only taken within the observed actions (or actions with sufficiently evidence to be in the support when using function approximators). This idea is similar to the spirit of the later IQL algorithm (Kostrikov et al., 2021), except IQL doesn not construct the filter function explicitly but relies instead on expectile maximization.

Proposition E.15. *Suppose $\mathcal{F} = (\mathcal{S} \times \mathcal{A} \to [-V_{\max}, V_{\max}])$[20]. For $V_{\max} \geq \frac{R}{1-\gamma}$, PPI/PQI is R-admissible with respect any $\tilde{\mathcal{R}} \subseteq (\mathcal{S} \times \mathcal{A} \to [-1, 1])$.*

Proof. We prove for PPI; the proof for PQI follows similarly. Consider some π such that $C := \sup_{s,a} \frac{d^{\pi(s,a)}}{\mu(s,a)} < \infty$. Suppose $\tilde{D}$ has N transitions. Take some $V_{\max} \geq \frac{R}{1-\gamma}$ and some $\bar{r}$ from $\mathcal{R}_R(\tilde{r})$. We use the Corollary 1 in (Liu et al., 2020): With I large enough, it holds with probability $1-\delta$,

$$V_{\bar{r}}^{\pi}(d_0) - V_{\bar{r}}^{\hat{\pi}}(d_0) \leq \tilde{O}\left(\frac{V_{\max}}{(1-\gamma)^3}\left(\sqrt{\frac{|\mathcal{S}||\mathcal{A}|\ln(1/\delta)}{N}} + \frac{\sup_{s,a,t} d_t^{\pi}(s,a)}{b}\epsilon_\mu + (1-\gamma)\mathbb{E}_{\pi^*,P}[\mathbb{1}[\mu(s,a) \leq 2b]]\right)\right)$$

where $\epsilon_\mu = D_{TV}(\hat{\mu}, \mu) = O(\frac{1}{\sqrt{N}})$. Further,

$$\begin{aligned}
(1-\gamma)\mathbb{E}_{\pi^*,P}[\mathbb{1}[\mu(s,a) \leq 2b]] &= \sum_{s,a} d^{\pi^*}(s,a)\mathbb{1}[\mu(s,a) \leq 2b] \\
&= \sum_{s,a} \mu(s,a)\frac{d^{\pi^*}(s,a)}{\mu(s,a)}\mathbb{1}[\mu(s,a) \leq 2b] \\
&\leq C\mathbb{E}_\mu[\mathbb{1}[\mu(s,a) \leq 2b]].
\end{aligned}$$

We can then upper bound the performance difference above as

$$V_{\bar{r}}^{\pi}(d_0) - V_{\bar{r}}^{\hat{\pi}}(d_0) \leq \tilde{O}\left(\frac{V_{\max}}{(1-\gamma)^3}\left(\sqrt{\frac{|\mathcal{S}||\mathcal{A}|\ln(1/\delta)}{N}} + \frac{\sup_{s,a,t} d_t^{\pi}(s,a)}{b}\frac{1}{\sqrt{N}} + C\mathbb{E}_\mu[\mathbb{1}[\mu(s,a) \leq 2b]]\right)\right)$$

We bound the latter two terms by tuning b. Notice $\mathbb{E}_\mu[\mathbb{1}[\mu(s,a) \leq 2b]] \leq |\mathcal{S}||\mathcal{A}|2b$ at most. This implies

$$\begin{aligned}
&\inf_b \frac{\sup_{s,a,t} d_t^{\pi}(s,a)}{b}\frac{1}{\sqrt{N}} + C\mathbb{E}_\mu[\mathbb{1}[\mu(s,a) \leq 2b]] \\
&\leq \inf_b \frac{\sup_{s,a,t} d_t^{\pi}(s,a)}{b}\frac{1}{\sqrt{N}} + C|\mathcal{S}||\mathcal{A}|2b \\
&= O\left(\frac{\sqrt{C|\mathcal{S}||\mathcal{A}|}}{N^{1/4}}\right)
\end{aligned}$$

Thus,

$$V_{\bar{r}}^{\pi}(d_0) - V_{\bar{r}}^{\hat{\pi}}(d_0) \leq \tilde{O}\left(\frac{V_{\max}}{(1-\gamma)^3}\left(\sqrt{\frac{|\mathcal{S}||\mathcal{A}|\ln(1/\delta)}{N}} + \frac{\sqrt{C|\mathcal{S}||\mathcal{A}|}}{N^{1/4}}\right)\right)$$

We can apply this similar argument used in the previous proofs to show that this bound simultaneously applies to all $\bar{r} \in \mathcal{R}_R(\tilde{r})$ since the high-probability concentration analysis is only taken on the data distribution where all rewards in $\mathcal{R}_R(\tilde{r})$ are equal. Thus, when $V_{\max} \geq \frac{R}{1-\gamma}$, PPQ (and similarly PQI) is R-admissibile. □

[20] The original algorithms of PPI and PQI assumes $\mathcal{F} = (\mathcal{S} \times \mathcal{A} \to [0, V_{\max}])$. We note that our extension to $[-V_{\max}, V_{\max}]$ do not change the performance bound as it absorbed by the $\tilde{O}$ notation.

E.7. MOReL and MOPO

MOReL (Kidambi et al., 2020) and MOPO (Yu et al., 2020b) are two popular model-based offline RL algorithms. The two algorithms operate in a similar manner, except that MOReL (Kidambi et al., 2020) uses the truncation approach (similar to PPI and PQI (Liu et al., 2020)) which classifies state-actions into known and unknown sets, whereas MOPO (Yu et al., 2020b) uses negative bonuses (similar to VI-LCB (Rashidinejad et al., 2021)). In this section, we show both algorithms are admissible.

We denote a model as $M = (\mathcal{S}, \mathcal{A}, P_M, r_M, \gamma)$, where $P_M : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the model dynamics, and $r_M : \mathcal{S} \times \mathcal{A} \to [-1, 1]$[21] with $1 \le R_{\max} < \infty$ being the reward function. The model class is denoted as $\mathcal{M}_{model}$.

E.7.1. MOREL

We analyze a variant of MOReL which also learns the reward function, whereas MOReL in the originally paper (Kidambi et al., 2020) assumes the reward function is given. We suppose model reward r_M and model dynamics P_M are learned by

$$r_M^\star, P_M^\star \in \operatorname*{arg\,min}_{M \in \mathcal{M}_{model}} \mathcal{E}_{\tilde{\mathcal{D}}}(M)$$

where

$$\mathcal{E}_{\tilde{\mathcal{D}}}(M) := \sum_{(s,a,\tilde{r},s') \in \tilde{\mathcal{D}}} -\log P_M(s'|s,a) + (r_M(s,a) - \tilde{r})^2.$$

We define the set of known state-actions $\mathcal{K}_\xi$ as

$$\mathcal{K}_\xi = \Big\{(s,a) : \sup_{M_1, M_2 \in \mathcal{M}_{\tilde{\mathcal{D}}}^\alpha} \big\{|r_{M_1}(s,a) - r_{M_2}(s,a)| + \gamma V_{\max} D_{TV}(P_{M_1}(\cdot|s,a), P_{M_2}(\cdot|s,a))\big\} \le \xi\Big\}, \tag{17}$$

with

$$\mathcal{M}_{\tilde{\mathcal{D}}}^\alpha := \{M \in \mathcal{M}_{model} : \mathcal{E}_{\tilde{\mathcal{D}}}(M) - \min_{M' \in \mathcal{M}_{model}} \mathcal{E}_{\tilde{\mathcal{D}}}(M') \le \alpha\},$$

We note that α can be chosen as $\Theta(|\mathcal{S}|^2|\mathcal{A}| \log 1/\delta)$ so that $\mathcal{M}_\alpha$ contains model $(\mathcal{S}, \mathcal{A}, P, \tilde{r}, \gamma)$ with probability $1 - \delta$ for any $\delta \in (0, 1]$ (see (Bhardwaj et al., 2023) for details).

Remark Note that we change the definition for the set of known state-actions. In (Kidambi et al., 2020), the set is given by

$$\mathcal{K}_\xi^0 = \{(s,a) : |r_M(s,a) - \tilde{r}(s,a)| + \gamma V_{\max} D_{TV}(P_M(\cdot|s,a), P(\cdot|s,a)) \le \xi\}. \tag{18}$$

However, it is not impossible to use such a set in practice since the reward function $\tilde{r}$ and true model $P(\cdot|s,a)$ are not given to the learner. Our known state-action set $\mathcal{K}_\xi$, in comparison, is easier to construct. It simply measures the maximum disagreement between two data-consistent models. With a considerate choice of α (see (Bhardwaj et al., 2023)), $\mathcal{K}_\xi$ is always a subset of $\mathcal{K}_\xi^0$. Our definition also matches better with the practical implementation in (Kidambi et al., 2020) which uses the disagreement between any two models in an ensemble.

Then MOReL learns policy $\hat{\pi}$ by solving the MDP $(\mathcal{S}, \mathcal{A}, \hat{P}, \hat{r}, \gamma)$, where

$$\hat{r}(s,a) = \begin{cases} r_M^\star(s,a), & s,a \in \mathcal{K}_\xi \\ -X, & \text{otherwise} \end{cases}$$

$$\hat{P}(s'|s,a) = \begin{cases} P_M^\star(s'|s,a), & s,a \in \mathcal{K}_\xi \\ \mathbb{1}(s' = s^\dagger), & \text{otherwise} \end{cases}$$

Note MOReL introduces an absorbing state $s^\dagger$.

Below we prove a performance guarantee of MOReL. The proof here is different from that in the original paper, because here we need to consider reward learning and the original proof does not provide an exact rate that converges to zero as the data size increases. (In the original paper there is a non-zero bias term that depends on the comparator policy.)

[21] Both MOPO and MOReL assume that the reward function is known. Here we study the variants of the two algorithms which also learn the reward function.

Proposition E.16 (MOReL Performance Guarantee)**.** *Given data $\tilde{\mathcal{D}} = \{(s, a, \tilde{r}, s')\}$ of size N of an unknown MDP $(\mathcal{S}, \mathcal{A}, \tilde{r}, P, \gamma)$. Suppose $\tilde{r}, r^\star_M \in [-1, 1]$ and $(\tilde{r}, P) \in \mathcal{M}^\alpha_{\tilde{\mathcal{D}}}$. Choose $\xi = \min(1, O(V_{max}(|\mathcal{S}|^2|\mathcal{A}|\log(1/\delta)/N)^{1/4}))$. Let $X \geq R$. For π such that $\sup_{s,a} \frac{d^\pi(s,a)}{\mu(s,a)} = C < \infty$, with probability $1-\delta$, it holds that*

$$(1-\gamma)\left(V^\pi_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0)\right) \leq O\left(V_{\max}\left(\frac{|\mathcal{S}|^2|\mathcal{A}|\log(\frac{1}{\delta})}{N}\right)^{1/4}\right), \quad \textit{with } V_{\max} \coloneqq \frac{R}{1-\gamma}$$

for all $\bar{r} \in \mathcal{R}_R(\tilde{r}) \coloneqq \{\bar{r} : \bar{r}(s,a) = \tilde{r}(s,a), \forall (s,a) \in supp(\mu)$ and $\bar{r}(s,a) \in [-R, 1], \forall (s,a) \in \mathcal{S} \times \mathcal{A}\}$.

Proof. We extend $\tilde{r}$ as $\tilde{r}(s^\dagger, a) = -X$ for all $a \in \mathcal{A}$. Let $\hat{M} = (\hat{r}, \hat{P})$.

First, by the optimality of $\hat{\pi}$, we write

$$\begin{aligned}
&V^\pi_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0)\\
=&V^\pi_{\bar{r}}(d_0) - V^\pi_{\hat{M}}(d_0) + V^\pi_{\hat{M}}(d_0) - V^{\hat{\pi}}_{\hat{M}}(d_0) + V^{\hat{\pi}}_{\hat{M}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0)\\
\leq&V^\pi_{\bar{r}}(d_0) - V^\pi_{\hat{M}}(d_0) + V^{\hat{\pi}}_{\hat{M}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0)
\end{aligned}$$

where $V^\pi_{\hat{M}}$ denotes the value of policy π with respect to the reward $\hat{r}$ and dynamics $\hat{P}$. The last inequality follows from the optimality of $\hat{\pi}$ on $\hat{M}$.

Then we notice the fact that $\mathcal{K}_\xi \subseteq \text{supp}(\mu)$ with a small ξ. The proof of the lemma below is given in Appendix E.7.3.

Lemma E.17. *If $\xi \leq 1$, $\mathcal{K}_\xi \subseteq supp(\mu)$.*

Let $\hat{d}^\pi$ denote the average state-action visitation of π with respect to $\hat{P}$.

$$\begin{aligned}
&(1-\gamma)\left(V^{\hat{\pi}}_{\hat{M}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0)\right)\\
=&\mathbb{E}_{s,a\sim\hat{d}^{\hat{\pi}}}[\hat{r}(s,a) + \gamma\mathbb{E}_{s'\sim\hat{P}|s,a}[V^{\hat{\pi}}_{\bar{r}}(s')] - \bar{r}(s,a) - \gamma\mathbb{E}_{s'\sim P|s,a}[V^{\hat{\pi}}_{\bar{r}}(s')]]\\
\leq&\mathbb{E}_{s,a\sim\hat{d}^{\hat{\pi}}}[\hat{r}(s,a) + \gamma\mathbb{E}_{s'\sim\hat{P}|s,a}[V^{\hat{\pi}}_{\bar{r}}(s')] - \bar{r}(s,a) - \gamma\mathbb{E}_{s'\sim P|s,a}[V^{\hat{\pi}}_{\bar{r}}(s')]|(s,a) \notin \mathcal{K}_\xi]\\
&\quad + \mathbb{E}_{s,a\sim\hat{d}^{\hat{\pi}}}[|\hat{r}(s,a) - \bar{r}(s,a)| + \gamma V_{\max} D_{TV}(\hat{P}(\cdot|s,a), P(\cdot|s,a))|(s,a) \in \mathcal{K}_\xi]\\
\leq&\xi + \mathbb{E}_{s,a\sim\hat{d}^{\hat{\pi}}}[\hat{r}(s,a) + \gamma\mathbb{E}_{s'\sim\hat{P}|s,a}[V^{\hat{\pi}}_{\bar{r}}(s')] - \bar{r}(s,a) - \gamma\mathbb{E}_{s'\sim P|s,a}[V^{\hat{\pi}}_{\bar{r}}(s')]|(s,a) \notin \mathcal{K}_\xi]\\
\leq&\xi + \mathbb{E}_{s,a\sim\hat{d}^{\hat{\pi}}}[-V_{\max} - \bar{r}(s,a) + \gamma V_{\max}|(s,a) \notin \mathcal{K}_\xi]\\
\leq&\xi
\end{aligned}$$

where in the second inequality we use $\mathcal{K}_\xi \subseteq \text{supp}(\mu)$ (which implies $\tilde{r} = \bar{r}$ in $\mathcal{K}_\xi$).

Similarly we can show

$$\begin{aligned}
&(1-\gamma)\left(V^\pi_{\hat{M}}(d_0) - V^\pi_{\bar{r}}(d_0)\right)\\
\geq&-\xi + \mathbb{E}_{s,a\sim d^\pi}[\hat{r}(s,a) + \gamma\mathbb{E}_{s'\sim\hat{P}|s,a}[V^\pi_{\hat{M}}(s')] - \bar{r}(s,a) - \gamma\mathbb{E}_{s'\sim P|s,a}[V^\pi_{\hat{M}}(s')]|(s,a) \notin \mathcal{K}_\xi]\\
\geq&-\xi - 2V_{\max}\mathbb{E}_{s,a\sim d^\pi}[\mathbb{1}[(s,a) \notin \mathcal{K}_\xi]]
\end{aligned}$$

Combining the three inequalities above gives

$$(1-\gamma)\left(V^\pi_{\bar{r}}(d_0) - V^{\hat{\pi}}_{\bar{r}}(d_0)\right) \leq 2V_{\max}\mathbb{E}_{s,a\sim d^\pi}[\mathbb{1}[(s,a) \notin \mathcal{K}_\xi]] + 2\xi$$

By the concentratability assumption, we can further upper bound

$$\mathbb{E}_{s,a\sim d^\pi}[\mathbb{1}[(s,a) \notin \mathcal{K}_\xi]] \leq C\mathbb{E}_{s,a\sim\mu}[\mathbb{1}[(s,a) \notin \mathcal{K}_\xi]]$$

Therefore we have

$$\begin{aligned}(1-\gamma)\left(V_{\tilde{r}}^{\pi}(d_0)-V_{\tilde{r}}^{\hat{\pi}}(d_0)\right) &\leq 2V_{\max}\mathbb{E}_{s,a\sim\mu}[\mathbb{1}[(s,a)\notin\mathcal{K}_{\xi}]]+2\xi\\ &\leq 2V_{\max}\frac{\mathbb{E}_{s,a\sim\mu}[E_{\text{sup}}(s,a)]}{\xi}+2\xi\\ &\leq 4V_{\max}\frac{\sup_{r_M,P_M}\mathbb{E}_{s,a\sim\mu}[E(s,a;r_M,P_M)]}{\xi}+2\xi\end{aligned}$$

where the last step is Markov inequality,

$$E_{\text{sup}}(s,a)=\sup_{M_1,M_2\in\mathcal{M}_{\mathcal{D}}^{\alpha}}\left\{|r_{M_1}(s,a)-r_{M_2}(s,a)|+\gamma V_{\max}D_{TV}(P_{M_1}(\cdot|s,a),P_{M_2}(\cdot|s,a))\right\},$$

and

$$E(s,a;r_M,P_M)=|r_M(s,a)-\tilde{r}(s,a)|+\gamma V_{\max}D_{TV}(P_M(\cdot|s,a),P(\cdot|s,a)).$$

Now we upper bound the expectation of E over μ. With probability greater than $1-\delta$,

$$\begin{aligned}\mathbb{E}_{\mu}[E(s,a;r_M,P_M)] &\leq \sqrt{\mathbb{E}_{\mu}[(|r_M(s,a)-\tilde{r}(s,a)|+\gamma V_{\max}D_{TV}(P_M(\cdot|s,a),P(\cdot|s,a)))^2]}\\ &\leq \sqrt{2}\sqrt{\mathbb{E}_{\mu}[(r_M(s,a)-\tilde{r}(s,a))^2+(\gamma V_{\max})^2D_{TV}(P_M(\cdot|s,a),P(\cdot|s,a))^2]}\\ &\leq (1+\gamma V_{\max})\sqrt{2}\sqrt{\mathbb{E}_{\mu}[(r_M(s,a)-\tilde{r}(s,a))^2+D_{TV}(P_M(\cdot|s,a),P(\cdot|s,a))^2]}\\ &\leq V_{\max}\sqrt{2}\sqrt{\mathbb{E}_{\mu}[(r_M(s,a)-\tilde{r}(s,a))^2+D_{TV}(P_M(\cdot|s,a),P(\cdot|s,a))^2]}\\ &\leq V_{\max}\sqrt{2}\sqrt{\frac{|\mathcal{S}|^2|\mathcal{A}|\log(\frac{1}{\delta})}{N}}\end{aligned}$$

where the last step is based on Lemma E.18.

Lemma E.18. *(Xie et al., 2022) With probability $1-\delta$, for any MDP model M,*

$$\mathbb{E}_{\mu}[|r_M(s,a)-r(s,a)|^2+D_{TV}(P_M(\cdot|s,a),P(\cdot|s,a))^2]\leq O\left(\frac{\mathcal{E}_{\mathcal{D}}(M)-\min_{M'}\mathcal{E}_{\mathcal{D}}(M')+|\mathcal{S}|^2|\mathcal{A}|\log(\frac{1}{\delta})}{N}\right).$$

Thus, we have

$$\begin{aligned}(1-\gamma)\left(V_{\tilde{r}}^{\pi}(d_0)-V_{\tilde{r}}^{\hat{\pi}}(d_0)\right) &\leq O\left(\frac{V_{\max}^2\sqrt{|\mathcal{S}|^2|\mathcal{A}|\log(\frac{1}{\delta})}}{\xi\sqrt{N}}\right)+2\xi\\ &\leq O\left(V_{\max}\left(\frac{|\mathcal{S}|^2|\mathcal{A}|\log(\frac{1}{\delta})}{N}\right)^{1/4}\right).\end{aligned}$$

□

By Proposition E.16, we can apply the previous analysis technique to MOReL to show it is admissible without addition union bounds, which leads to the corollary below.

Corollary E.19. *For $X\geq R$, MOReL is R-admissible with respect any $\tilde{\mathcal{R}}\subseteq(\mathcal{S}\times\mathcal{A}\to[-1,1])$.*

E.7.2. MOPO

MOPO (Yu et al., 2020b) is very similar to MOReL except that it uses negative bonuses (like VI-LCB) instead of truncation. In the original paper of MOPO, the authors assume the reward is given. Here we consider a variant that also learns the reward. Specifically, given $P_M^\star$ and $r_M^\star$ above it solves the MDP $(\mathcal{S}, \mathcal{A}, P_M^\star, \hat{r}, \gamma)$, where

$$\hat{r}(s,a) = r_M^\star(s,a) - b(s,a)$$

While the original MOPO paper does not give a specific design of $b(s,a)$ that is provably correct, in principle making MOPO provably correct is possible. Essentially, we need to choose a large enough bonus to cover the error of the model-based Bellman operator (defined by the estimated reward and dynamics). This would lead to a bonus of order $V_{\max}$ (see the analysis of VI-LCB in Appendix E.5). Then we can proceed with an analysis that combines that of MOReL and VI-LCB to prove MOPO's performance guarantee. This can then be used to show (like the previous proofs) that MOPO is admissible. We omit the proof here.

Corollary E.20. *Suppose $b(s,a) = \Theta(V_{\max})$ for $(s,a) \notin supp(\mu)$. For $V_{\max} \geq \frac{R}{1-\gamma}$, MOPO is R-admissible with respect any $\tilde{\mathcal{R}} \subseteq (\mathcal{S} \times \mathcal{A} \to [-1,1])$.*

E.7.3. Proof of Technical Lemma

Proof of Lemma E.17. Consider any $(\bar{s}, \bar{a}) \notin \text{supp}(\mu)$. Given any $M = (\mathcal{S}, \mathcal{A}, P_M, r_M, \gamma) \in \mathcal{M}_{\tilde{\mathcal{D}}}^{\alpha}$, we construct $M' = (\mathcal{S}, \mathcal{A}, P_M, r_{M'}, \gamma)$ with $r_{M'} = r_M$ for all $(s,a) \in (\mathcal{S} \times \mathcal{A}) \setminus \{(\bar{s}, \bar{a})\}$, and $r_{M'}(\bar{s}, \bar{a}) = -\text{sign}(r_M(\bar{s}, \bar{a}))$. By construction, since M and M' agrees on all state-actions except $(\bar{s}, \bar{a})$, which is not in data support, we have $M' \in \mathcal{M}_{\tilde{\mathcal{D}}}^{\alpha}$. However, we have that

$$\begin{aligned}
&\sup_{M_1, M_2 \in \mathcal{M}_{\tilde{\mathcal{D}}}^{\alpha}} \left\{ |r_{M_1}(\bar{s}, \bar{a}) - r_{M_2}(\bar{s}, \bar{a})| + \gamma V_{\max} D_{TV}(P_{M_1}(\cdot|\bar{s}, \bar{a}), P_{M_2}(\cdot|\bar{s}, \bar{a})) \right\} \\
\geq & |r_M(\bar{s}, \bar{a}) - r_{M'}(\bar{s}, \bar{a})| + \gamma V_{\max} D_{TV}(P_M(\cdot|\bar{s}, \bar{a}), P_M(\cdot|\bar{s}, \bar{a})) \\
= & |r_M(\bar{s}, \bar{a}) - r_{M'}(\bar{s}, \bar{a})| \geq 1 \geq \xi.
\end{aligned}$$

This means that $(\bar{s}, \bar{a}) \notin \mathcal{K}_\xi$. Therefore $\mathcal{K}_\xi \subseteq \text{supp}(\mu)$. □

F. Grid World

In this section, we describe the full details of the grid world study that was discussed in Section 3.1.

Environment. We consider goal-directed navigation in the environment shown in Figure 10. We use a fixed grid world layout shown in the figure. The agent's state is given by a tuple (x, y, d) where $(x, y) \in [5]^2$ represents the 2-d coordinate

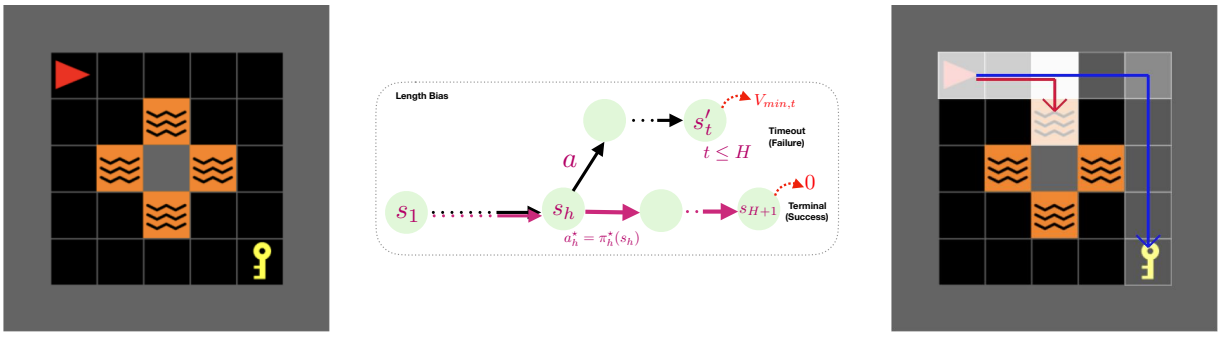


Figure 10: **Left**: A goal-directed gridworld navigation task where the agent (red triangle) has to reach the goal (key) and avoid lava (orange waves). **Center**: Shows the length bias that helps offline RL succeed even when rewards are entirely replaced with random or adversarial values. **Right**: Results on the grid world task. Behavior cloning fails to solve the task (red path) while using an offline RL method leads to success (blue path) *even with wrong rewards.* The opacity of the grid square overlay indicates how frequent that state is in the dataset (more opaque means more frequent).

and $d \in \{N, W, E, S\}$ encodes the North, West, East, and South direction respectively that the agent is facing. The agent's action space is $\mathcal{A} = \{f, l, r\}$ where f denotes the action of moving to grid square in front of the agent, l denotes a left turn of 90 degrees, and r denotes a right turn of 90 degrees. The agent can go through the lava (orange wavy square) but cannot go through the wall (grey square). The goal (key) is an absorbing state. The agent gets a reward of +1 when first visiting the goal followed by a reward of 0 for then staying in the goal. The agent gets a reward of -1 for reaching lava and a reward of -0.01 for all other actions to incentivize the agent to reach the goal along the shortest safe path. We use a horizon of $H = 20$. The agent deterministically starts in the top-right corner shown in Figure 10.

Dataset. We collect a dataset $\mathcal{D}$ of 500 episodes by taking 100 identical optimal episodes along with 400 identical episodes that all touch the topmost lava field. We introduce a length bias by terminating an episode if the agent touches the lava or fails to reach the goal in $H - 1$ steps (i.e., the agent doesn't survive). In addition to the original setting with the observed reward, we consider three additional settings where the rewards in the dataset are replaced by: (i) a constant zero reward, (ii) their negative, and (iii) a random number sampled uniformly in $[0, 1]$.

Algorithm 4 PEVI($\mathcal{S}, \mathcal{A}, H, \mathcal{D}, \beta, V_{\min}, V_{\max}$) (Jin et al., 2021). We are given access to a set $\mathcal{D}$ of episodes (τ). Additionally, we are given a hyperparameter $\beta \in \mathcal{R}_{\geq 0}$ that controls pessimism. The hyperparameters $V_{\min}, V_{\max} \in \mathcal{R}^H$ indicate bounds on policy value with $V_{\min,h}$ and $V_{\max,h}$ denoting the minimum and maximum policy values respectively for time step $h \in [H]$. Unlike the original PEVI algorithm of (Jin et al., 2021), we treat $V_{\min}, V_{\max}$ as hyperparameters.

Set $V_{H+1} = 0$
for $h = H, \cdots, 1$ **do**
  for $s \in \mathcal{S}$ **do**
    for $a \in \mathcal{A}$ **do**
      $n_h(s,a) = \sum_{\tau \in \mathcal{D}} \mathbb{1}\{s_h^\tau = s \wedge a_h^\tau = a\}$
      If $n_h(s,a) > 0$
        $\tilde{Q}_h(s,a) = \frac{1}{n_h(s,a)} \sum_{\tau \in \mathcal{D}} \mathbb{1}\{s_h^\tau = s \wedge a_h^\tau = a\}\{r_h^\tau + V_{h+1}(s_{h+1}^\tau)\}$
        $\bar{Q}_h(s,a) = \tilde{Q}_h(s,a) - \frac{\beta}{\sqrt{n_h(s,a)}}$
        $\widehat{Q}_h(s,a) = \max\{V_{\min,h}, \min\{\bar{Q}_h(s,a), V_{\max,h}\}\}$
      Else
        $\widehat{Q}_h(s,a) = V_{\min,h}$
    end for
    $\pi_h(s) = \arg\max_{a \in \mathcal{A}} \widehat{Q}_h(s,a)$
    $V_h(s) = \widehat{Q}_h(s, \pi_h(s))$
  end for
end for
Return $\pi = (\pi_{1:H})$

Methods. We evaluate two methods: behavior cloning which simply takes the action with the highest empirical count in a given state and a random action if the state was unseen, and PEVI (Jin et al., 2021) an offline RL method with guarantees for tabular MDP. We provide a pseudocode of PEVI in Algorithm 4. Unlike the original PEVI in (Jin et al., 2021), we provide minimum and maximum bounds for the value function for each time step as an additional hyperparameter. These bounds should be dictated by our CMDP framework and should penalize out-of-distribution actions. In principle, we need to set these bounds such that the algorithm is admissible. Please see Appendix C and Appendix D for details.

Intuitively, PEVI performs dynamic programming similar to a standard value iteration with two key changes. Firstly, it uses pessimistic value initialization where the learned value $V_h(s)$ of a state s at time step h is set to the lowest possible return $V_{min,h}$ if the state is unseen at time step h. Secondly, when performing value iteration it adds a pessimism penalty of $-\beta/\sqrt{n(s,a)}$ to the dataset reward where $n(s,a)$ is the empirical state-action count in $\mathcal{D}$. This pessimism penalty ensures that the learned value function lower bounds the true value function (hence a pessimistic estimate). In our experiments, we vary β and set $V_{\min,h} = \tilde{V}_{\min,h} - \beta(H - h + 1) - 1$ where $\tilde{V}_{\min,h}$ is the minimum possible return for the given reward type that we are working with. The value of $\tilde{V}_{\min,h}$ is $-(H - h + 1)$ for the original reward, -1 for the negative reward, and 0 for the zero reward and random reward cases). We define $V_{\max,h} = \tilde{V}_{\max,h}$ where $\tilde{V}_{\max,h}$ is the maximum possible return for the given reward type. The value of $\tilde{V}_{\max,h}$ is 1 of the original reward, $(H - h + 1)$ for the random reward and the negative

Reward Type	Behavior Cloning	PEVI
Original Reward	-1.19 ± 0.00	0.92 ± 0.00
Zero Reward	-1.19 ± 0.00	0.92 ± 0.00
Random Reward	-1.19 ± 0.00	0.92 ± 0.00
Negative Reward	-1.19 ± 0.00	0.92 ± 0.00

Table 16: **Gridworld results:** Mean return on 1000 test episodes. PEVI achieves the optimal return of $V^\star = 0.92$.

reward, and 0 for the zero reward case.

Results. We show numerical results in Table 16. We visualize results in Figure 10. The behavior cloning simply imitates the most common trajectory in the dataset which results in going to the lava (shown in red in the figure). In contrast, the PEVI is able to reach the goal in every reward situation.

Explanation. The reason why PEVI works can be understood simply by pessimistic initialization and length bias. The length bias in this setting is visualized in the center of Figure 10. PEVI assigns the final state s'_t of a failed trajectory will get a pessimistic value of $V_t(s'_t) = V_{\min,_t}$. This value isn't updated as we never take any action in s'_t due to timeout. In our case, s'_t will be the state where the agent first visits the lava. In contrast, the final state of a successful trajectory s_{H+1} gets assigned a value of 0. If $V_{\min}$ is sufficiently negative, then PEVI will only consider policies that stay all H steps within the data support. Due to length bias, this is not true for the non-surviving trajectories that reach the lava. Further, the only trajectories in our case that complete end up reaching and staying in the goal. Therefore, under the assumption that $V_{\min,h}$ are all sufficiently small, PEVI will learn a policy reaches the goal irrespective of the correctness of the reward in the data. The sufficiency of the negativity of $V_{\min,h}$ is implied by our CMDP framework. Lastly, note that as the world is deterministic, we didn't need to use the pessimism arising from the $\frac{-\beta}{\sqrt{n(s,a)}}$ penalty term.