Model-Based Offline Reinforcement Learning with Pessimism-Modulated Dynamics Belief

Anonymous Author(s)
Affiliation
Address
email

Abstract

Model-based offline reinforcement learning (RL) aims to find highly rewarding policy, by leveraging a previously collected static dataset and a learned dynamics model. While the dynamics model is learned by reusing the static dataset, its generalization ability hopefully promotes policy learning if properly utilized. To that end, several works propose to quantify the uncertainty of predicted dynamics, and explicitly apply it to penalize reward. However, as the dynamics has different implication than the reward, characterizing the impact of dynamics uncertainty through reward penalty may incur unexpected tradeoff between model utilization and risk avoidance. In this work, we instead maintain a belief distribution over dynamics, and evaluate/optimize policy through biased sampling from the belief. The sampling procedure, biased towards pessimism, is derived based on an alternating Markov game formulation of offline RL. We formally show that the biased sampling naturally induces an updated dynamics belief with policy-dependent reweighting factor, termed *Pessimism-Modulated Dynamics Belief*. To improve policy, we devise an iterative regularized policy optimization algorithm for the game, with guarantee of monotonous improvement under certain condition. To make practical, we further devise an offline RL algorithm to approximately find the solution. Empirical results show that the proposed approach achieves state-of-the-art performance on a wide range of offline RL benchmark tasks.

1 Introduction

In typical paradigm of RL, the agent actively interacts with environment and receives feedback to promote policy improvement. The essential trail-and-error procedure can be costly, unsafe or even prohibitory in practice (e.g. robotics [1], autonomous driving [2], and healthcare [3]), thus constituting a major impediment to actual deployment of RL. Meanwhile, in vast of applications, historical data records are available to reflect the system feedback under a predefined policy. This raises the opportunity to learn policy in purely offline setting.

In offline setting, as no further interaction with environment is permitted, the dataset provides a limited coverage in state-action space. Then, the policy that induces out-of-distribution (OOD) state-action pairs can not be well evaluated in offline learning phase, and deploying it online potentially attains terrible performance. Recent studies have reported that applying vanilla RL algorithms to offline dataset exacerbates such a distributional shift [4–6], making them unsuitable for offline setting.

To tackle the distributional shift issue, a number of offline RL approaches have been developed. Specifically, one category of them propose to directly constrain the policy close to the one collecting data [4, 5, 7, 8], or penalize Q-value towards conservatism for OOD state-action pairs [9–11]. While they achieve remarkable performance gains, the policy regularizer and the Q-value penalty tightly restricts the produced policy within the data manifold. Instead, more recent works consider to

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

quantify the uncertainty of Q-value with neural network ensembles [12], where the consistent Q-value estimates indicate high confidence and can be plausibly used during learning process, even for OOD state-action pairs [13, 14]. However, the uncertainty quantification over OOD region highly relies on how neural network generalizes [15]. As the prior knowledge of Q-function is hard to acquire and insert into the neural network, the generalization is unlikely reliable to facilitate meaningful uncertainty quantification [16]. Notably, all these works are model-free.

Model-based offline RL optimizes policy based on a constructed dynamics model. Compared to the model-free approaches, one prominent advantage is that the prior knowledge of dynamics is easier to access. First, the generic prior like smoothness widely exists in various domains [17]. Second, the sufficiently learned dynamics models for relevant tasks can act as a data-driven prior for the concerned task [18–20]. With richer prior knowledge, the uncertainty quantification for dynamics is more trustworthy. Similar to the model-free approach, the dynamics uncertainty can be incorporated to find reliable policy beyond data coverage. However, an additional challenge is how to characterize the accumulative impact of dynamics uncertainty on the long-term reward, as the system dynamics is with entirely different implication compared to the reward or Q-value.

Although existing model-based offline RL literature theoretically bounds the impact of dynamics uncertainty on final performance, their practical variants characterize the impact through reward penalty [6, 21, 22]. Concretely, the reward function is penalized by the dynamics uncertainty for each state-action pair [21], or the agent is enforced to a low-reward absorbing state when the dynamics uncertainty exceeds a certain level [6]. While optimizing policy in these constructed MDPs stimulates anti-uncertainty behavior, the final policy tends to be over-conservative. For example, even the transition dynamics for a state-action pair is ambiguous among several possible candidates, these candidates may generate the states from which the system evolves similarly.[1] Then, such a state-action pair should not be treated specially. For a more detailed discussion of related works, see Appendix A.

Motivated by the above intuition, we propose pessimism-modulated dynamics belief for model-based offline RL. In contrast with the previous approaches, the dynamics uncertainty is not explicitly quantified. To characterize its impact, we maintain a belief distribution over system dynamics, and the policy is evaluated and optimized through biased sampling from it. The sampling procedure, biased towards pessimism, is derived based on an alternating Markov game (AMG) formulation of offline RL. We formally show that the biased sampling naturally induces an updated dynamics belief with policy-dependent reweighting factor, termed *Pessimism-Modulated Dynamics Belief*. Besides, the degree of pessimism is monotonously determined by the hyperparameters in sampling procedure.

The considered AMG formulation can be regarded as a generalization of robust MDP, which is proposed as an surrogate to optimize the percentile performance in face of dynamics uncertainty [23, 24]. However, robust MDP suffers from two significant shortcomings: 1) The percentile criterion is over-conservative since it fixates on a single pessimistic dynamics instance [25, 26]; 2) Robust MDP is constructed based on an uncertainty set, and the improper choice of uncertainty set would further aggravate the degree of conservatism [27, 28]. The AMG formulation is kept from these shortcomings. To solve the AMG, we devise an iterative regularized policy optimization algorithm, with guarantee of monotonous improvement under certain condition. To make practical, we further derive an offline RL algorithm to approximately find the solution, and empirically evaluate it on the offline RL benchmark D4RL. The results show that the proposed approach obviously outperforms previous state-of-the-art (SoTA) in 9 out of 18 environment-dataset configurations and performs competitively in the rest, without tuning hyperparameters for each task. The proof of theorems in this paper are presented in Appendix B.

2 Preliminaries

Markov Decision Process (MDP) A MDP is depicted by the tuple $(\mathcal{S}, \mathcal{A}, T, r, \rho_0, \gamma)$, where $\mathcal{S}, \mathcal{A}$ are state and action spaces, $T(s'|s,a)$ is the transition probability, $r(s,a)$ is the reward function, $\rho_0(s)$ is the initial state distribution, and γ is the discount factor. The goal of RL is to find the policy $\pi : s \to \Delta(\mathcal{A})$ that maximizes the cumulative discounted reward:

$$J(\pi, T) = \mathbb{E}_{\rho_0, T, \pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right], \tag{1}$$

[1] Or from these states, the system evolves differently but generates similar rewards.

where $\Delta(\cdot)$ denotes the probability simplex. In typical RL paradigm, this is achieved via actively interacting with environment.

Offline RL In offline setting, the environment is unaccessible, and only a static dataset $\mathcal{D} = \{(s, a, r, s')\}$ is provided, containing the previously logged data samples under an unknown behavior policy. Offline RL aims to optimize the policy by solely leveraging the offline dataset.

To simplify the presentation, we assume the reward function r and initial state distribution ρ_0 are known. Then, the system dynamics is unknown only in terms of the transition probability T. Note that the considered formulation and the proposed approach can be easily extend to the general case without additional technical modification.

Robust MDP With the offline dataset, a straightforward strategy is first learning a dynamics model $\tau(s'|s, a)$ and then optimizing policy via simulation. However, due to the limitedness of available data, the learned model is inevitably imprecise. Robust MDP [23] is an surrogate to optimize policy with consideration of the ambiguity of dynamics. Concretely, robust MDP is constructed by introducing an uncertainty set $\mathcal{T} = \{\tau\}$ to contain plausible transition probabilities. If the uncertainty set includes the true transition with probability of $(1 - \delta)$, the performance of any policy π in true MDP can be lower bounded by $\min_{\tau\in\mathcal{T}} J(\pi, \tau)$ with probability of at least $(1 - \delta)$. Thus, the percentile performance for the true MDP can be optimized by finding a solution to

$$\max_{\pi} \min_{\tau\in\mathcal{T}} J(\pi, \tau). \tag{2}$$

Despite its popularity, Robust MDP suffers from two major shortcomings: First, the percentile criterion overly fixates on a single pessimistic transition instance, especially when there are multiple optimal policies for this transition but they lead to dramatically different performance for other transitions [25, 26]. This behavior results in unnecessarily conservative policy.

Second, the level of conservatism can be further aggravated when the uncertainty set is inappropriately constructed [27]. For a given policy π, the ideal situation is that $\mathcal{T}$ contains the $(1 - \delta)$ proportion of transitions with which the policy achieves higher performance than with the other δ proportion. Then, $\min_{\tau\in\mathcal{T}} J(\pi, \tau)$ is exactly the δ-quantile performance. This requires the uncertainty set to be policy-dependent, and during policy optimization the uncertainty set should change accordingly. Otherwise, if $\mathcal{T}$ is predetermined and keeps fixed, it is possible to have $\tau' \notin \mathcal{T}$ with non-zero probability and satisfying $J(\pi^*, \tau') > \min_{\tau\in\mathcal{T}} J(\pi^*, \tau)$, where π^* is the optimal policy for (2). Then, adding τ' into $\mathcal{T}$ does not affect the optimal solution of the problem (2). This indicates that we are essentially optimizing a δ'-quantile performance, where δ' can be much smaller than δ. In literature, the uncertainty sets are mostly predetermined before policy optimization [23, 29–31].

3 Pessimism-Modulated Dynamics Belief

In short, robust MDP is over-conservative due to the fixation on a single pessimistic transition instance and the predetermination of uncertainty set. In this work, we strive to take the entire spectrum of plausible transitions into account, and let the algorithm by itself determine which subset deserves more attention. To this end, we consider an alternating Markov game formulation of offline RL, based on which the proposed offline RL approach is derived.

3.1 Formulation

Alternating Markov game (AMG) The AMG is a specialization of two-player zero-sum game, depicted by $(\mathcal{S}, \bar{\mathcal{S}}, \mathcal{A}, \bar{\mathcal{A}}, G, r, \rho_0, \gamma)$. The game starts from a state sampled from ρ_0, then two players alternatively choose actions $a \in \mathcal{A}$ and $\bar{a} \in \bar{\mathcal{A}}$ under states $s \in \mathcal{S}$ and $\bar{s} \in \bar{\mathcal{S}}$, along with the game transition defined by $G(\bar{s}|s, a)$ and $G(s|\bar{s}, \bar{a})$. At each round, the primary player receives reward $r(s, a)$ and the secondary player receives its negative counterpart $-r(s, a)$.

Offline RL as AMG We formulate the offline RL problem as an AMG, where the primary player optimizes a reliable policy for our concerned MDP in face of stochastic disturbance from the secondary player. The AMG is constructed by augmenting the original MDP. As both have the transition probability, we use game transition and system transition to differentiate them.

For the primary player, its state space $\mathcal{S}$, action space $\mathcal{A}$ and reward function $r(s,a)$ are same with those in the original MDP. After the primary player acts, the game emits a N-size set of system transition candidates $\mathcal{T}^{sa}$. Formally, the set is generated according to

$$G\left(\bar{s}=\mathcal{T}^{sa}|s,a\right)=\prod_{\tau^{sa}\in\mathcal{T}^{sa}}\mathbb{P}_T^{sa}(\tau^{sa}), \tag{3}$$

where $\mathbb{P}_T^{sa}$ is a given belief distribution of system transition $\tau(\cdot|s,a)$ under state-action pair (s,a), and we use $\tau^{sa}(\cdot)$ to re-denote $\tau(\cdot|s,a)$ for short. According to (3), the elements in $\mathcal{T}^{sa}$ are independent and identically distributed samples following $\mathbb{P}_T^{sa}$. The major difference to uncertainty set in robust MDP is that the set introduced here is unfixed and stochastic for each step. To differentiate with uncertainty set we call it candidate set, but to avoid redundancy we reuse the notation of $\mathcal{T}$. The belief distribution $\mathbb{P}_T^{sa}$ can be chosen arbitrarily to incorporate knowledge of system transition. Particularly, when the prior distribution of system transition is accessible, $\mathbb{P}_T^{sa}$ can be obtained as the posterior by integrating the prior and the evidence $\mathcal{D}$ through Bayes' rule.

For the secondary player, it receives the candidate set $\mathcal{T}^{sa}$ as state. Thus, we can denote its state space by $\bar{\mathcal{S}}=\Delta^N(\mathcal{S})$, i.e., an n-fold Cartesian product of probability simplex over $\mathcal{S}$. Note that this state $\mathcal{T}^{sa}$ also takes the role of action space, i.e., $\bar{\mathcal{A}}=\mathcal{T}^{sa}$, meaning that the action of secondary player is to choose a system transition from the candidate set. Given the chosen $\tau^{sa}\in\mathcal{T}^{sa}$, the game evolves by sampling τ^{sa}, i.e.,

$$G\left(s'|\bar{s}=\mathcal{T}^{sa},\bar{a}=\tau^{sa}\right)=\tau^{sa}(s'), \tag{4}$$

and the primary player receives s' to make the game continue. In the following, we omit the superscript sa in τ^{sa}, $\mathcal{T}^{sa}$ and $\mathbb{P}_T^{sa}$ when it is clear from the context. We also use $\mathbb{P}_T^N(\mathcal{T}^{sa})$ to compactly denote the game transition $G\left(\bar{s}=\mathcal{T}^{sa}|s,a\right)$ in (3).

For the above AMG, we consider a specific policy (explained below) for the secondary player, such that the cumulative discounted reward of the primary player with policy π can be written as:

$$J(\pi):=\mathop{\mathbb{E}}_{\rho_0,\pi,\mathbb{P}_T^N}\lfloor\min\rfloor_{\tau_0\in\mathcal{T}_0}^k\left[\mathop{\mathbb{E}}_{\tau_0,\pi,\mathbb{P}_T^N}\lfloor\min\rfloor_{\tau_1\in\mathcal{T}_1}^k\cdots\left[\mathop{\mathbb{E}}_{\tau_\infty,\pi}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\right]\right]\right], \tag{5}$$

where the subscripts of τ and $\mathcal{T}$ denote time step, the expectation is over $s_t\sim\rho_0$ or $\tau_{t-1}(\cdot|s_{t-1},a_{t-1}),a_t\sim\pi(\cdot|s_t)$ and $\mathcal{T}_t\sim\mathbb{P}_T^N$, and the operator $\lfloor\min\rfloor_{x\in\mathcal{X}}^k f(x)$ denotes finding kth minimum of $f(x)$ over $x\in\mathcal{X}$. The policy of secondary player is implicitly defined by the operator $\lfloor\min\rfloor_{x\in\mathcal{X}}^k f(x)$. When changing $k\in\{1,2,\cdots,N\}$, the secondary player exhibits various degree of adversarial or aggressive disturbance to the future reward. From the view of original MDP, this behavior raises flexible tendency ranging from pessimism to optimism when evaluating policy π.

The distinctions between the above AMG and robust MDP can be highlighted in the two aspects: 1) With a belief distribution over transitions, robust MDP will select only part of its supports with non-zero probability into uncertainty set, and the set elements are treated indiscriminatingly. It means that both the possibility of transitions out of the uncertainty set and the relative likelihood of transitions within the uncertainty set are discarded. However, in the AMG, the candidate set simply contains samples drawn from the belief distribution, implying no information drop in an average sense. Intuitively, by keeping richer knowledge of the system, the performance evaluation is more exact and away from excessive conservatism; 2) In robust MDP, the level of conservatism is expected to be controlled by its hyperparameter δ. However, as illustrated in Section 2, a smaller δ does not necessarily corresponds to a more conservative performance evaluation, because of the extra impact from uncertainty set construction. In contrast, for the AMG, the degree of conservatism is adjusted by the size of candidate size N and the order of minimum k. When changing k or N, the impact to performance evaluation is known for certain, as formalized in Theorem 3.

To evaluate $J(\pi)$, we define the following Bellman backup operator:

$$\mathcal{B}_{N,k}^{\pi}Q(s,a)=r(s,a)+\gamma\mathbb{E}_{\mathbb{P}_T^N}\Big[\lfloor\min\rfloor_{\tau\in\mathcal{T}}^k\mathbb{E}_{\tau,\pi}\left[Q(s',a')\right]\Big]. \tag{6}$$

As the operator depends on N, k and we emphasize pessimism in offline RL, we call it (N,k)-pessimistic Bellman backup operator. Compared to the standard Bellman backup operator in Q-learning, $\mathcal{B}_{N,k}^{\pi}$ additionally includes the expectation over $\mathcal{T}\sim\mathbb{P}_T^N$ and the k-minimum operator over $\tau\in\mathcal{T}$. Despite these differences, we prove that $\mathcal{B}_{N,k}^{\pi}$ is still a contraction mapping, based on which $J(\pi)$ can be easily evaluated.

Theorem 1 (Policy Evaluation)**.** *The (N,k)-pessimistic Bellman backup operator $\mathcal{B}^{\pi}_{N,k}$ is a contraction mapping. By starting from any function $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and repeatedly applying $\mathcal{B}^{\pi}_{N,k}$, the sequence converges to $Q^{\pi}_{N,k}$, with which we have $J(\pi) = \mathbb{E}_{\rho_0,\pi} Q^{\pi}_{N,k}(s_0, a_0)$.*

3.2 Pessimism-Modulated Dynamics Belief

With the converged Q-value, we are ready to establish a more direct connection between the AMG and the original MDP. The connection appears as the answer to a natural question: the calculation of (6) encompasses biased samples from the dynamics belief distribution, can we treat these samples as the unbiased ones sampling from another belief distribution? We give positive answer in the following theorem.

Theorem 2 (Equivalent MDP with Pessimism-Modulated Dynamics Belief)**.** *The alternating Markov game in* (5) *is equivalent to the MDP with tuple $(\mathcal{S}, \mathcal{A}, \widetilde{T}, r, \rho_0, \gamma)$, where the transition probability $\widetilde{T}(s'|s,a) = \mathbb{E}_{\widetilde{\mathbb{P}}^{sa}_T}[\tau^{sa}(s')]$ is defined with the reweighted belief distribution $\widetilde{\mathbb{P}}^{sa}_T$:*

$$\widetilde{\mathbb{P}}^{sa}_T(\tau^{sa}) \propto w\Big(\mathbb{E}_{\tau^{sa},\pi}\big[Q^{\pi}_{N,k}(s',a')\big]; k, N\Big)\mathbb{P}^{sa}_T(\tau^{sa}), \tag{7}$$

$$w(x;k,N) = \big[F(x)\big]^{k-1}\big[1-F(x)\big]^{N-k}, \tag{8}$$

and $F(\cdot)$ is cumulative density function. Furthermore, the value of $w(x;k,N)$ first increases and then decreases with x, and its maximum is obtained at the $\frac{k-1}{N-1}$ quantile, i.e., $x^* = F^{-1}\left(\frac{k-1}{N-1}\right)$.

In right-hand side of (18), τ^{sa} itself is random following the belief distribution, thus $\mathbb{E}_{\tau^{sa},\pi}\big[Q^{\pi}_{N,k}(s',a')\big]$, as a functional of τ^{sa}, is also a random variable, and its cumulative density function is determined by the belief distribution $\mathbb{P}^{sa}_T$. Intuitively, we can treat $\mathbb{E}_{\tau^{sa},\pi}\big[Q^{\pi}_{N,k}(s',a')\big]$ as a pessimism indicator for transition τ^{sa}, with larger value indicating less pessimism.

From Theorem 2, the maximum of w is obtained at τ^*: $F\Big(\mathbb{E}_{\tau^*,\pi}\big[Q^{\pi}_{N,k}(s',a')\big]\Big) = \frac{k-1}{N-1}$, i.e., the transition with $\frac{k-1}{N-1}$-quantile pessimism indicator. Besides, when $\mathbb{E}_{\tau^{sa},\pi}\big[Q^{\pi}_{N,k}(s',a')\big]$ departs the $\frac{k-1}{N-1}$ quantile, the reweighting coefficient for its τ^{sa} decreases. Considering the effect of w to $\widetilde{\mathbb{P}}^{sa}_T$ and the equivalence between the AMG and the refined MDP, we can say that $J(\pi)$ is a soft percentile performance. Compared to the standard percentile criteria, $J(\pi)$ is derived by reshaping belief distribution towards concentrating around a certain percentile, rather than fixating on a single percentile point. Due to this feature, we term $\widetilde{\mathbb{P}}^{sa}_T$ *Pessimism-Modulated Dynamics Belief (PMDB)*.

Lastly, recall that all the above derivations are with hyperparameters k and N, we present the monotonicity of $Q^{\pi}_{N,k}$ over them in Theorem 3. Furthermore, by combining Theorem 1 with Theorem 3, we conclude that $J(\pi)$ decreases with N and increases with k.

Theorem 3 (Monotonicity)**.** *The converged Q-function $Q^{\pi}_{N,k}$ are with the following properties:*

- *Given any k, the Q-function $Q^{\pi}_{N,k}$ element-wisely decreases with $N \in \{k, k+1, \cdots\}$.*
- *Given any N, the Q-function $Q^{\pi}_{N,k}$ element-wisely increases with $k \in \{1, 2, \cdots, N\}$.*
- *The Q-function $Q^{\pi}_{N,N}$ element-wisely increases with N.*

Remark 1 (Special Cases). *For $N = k = 1$, we have $\widetilde{\mathbb{P}}^{sa}_T = \mathbb{P}^{sa}_T$. Then, the performance is evaluated through sampling the initial belief distribution. This resembles the common methodology in model-based RL (MBRL), with dynamics belief defined by the uniform distribution over dynamics model ensembles. For $k = \delta(N-1)+1$ and $N \to \infty$, $\widetilde{\mathbb{P}}^{sa}_T$ asymptotically collapses to be a delta function. Then, $J(\pi)$ degrades to fixate on a single transition instance. It is equivalent to the robust MDP with the uncertainty set constructed as $\Big\{\tau^{sa} : \mathbb{P}^{sa}_T(\tau^{sa}) > 0, \mathbb{E}_{\tau^{sa},\pi}\big[Q^{\pi}_{N,k}(s',a')\big] \geq F^{-1}(\delta)\Big\}$. In this sense, the AMG is a successive interpolation between model-based RL and robust MDP.*

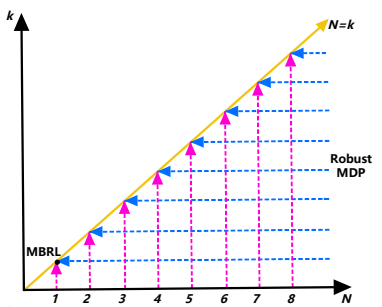

Figure 1: Monotonicity of Q-values. The arrows indicate the directions along which Q-values increase.

4 Policy Optimization with Pessimism-Modulated Dynamics Belief

In this section, we optimize policy by maximizing $J(\pi)$. The major consideration is that the methodology should adapt well for both discrete and continuous action spaces. In continuous setting of MDP, the policy can be updated by following stochastic/deterministic policy gradient [32, 33]. However, for the AMG, evaluating $J(\pi)$ itself involves an inner dynamic programming procedure as in Theorem 1. As each evaluation of $J(\pi)$ can only produce one exact gradient, it is inefficient to maximize $J(\pi)$ via gradient-based method. Instead, in this section we consider a series of sub-problems with Kullback–Leibler (KL) regularization. Solving each sub-problem makes prominent update to the policy, and the sequence of solutions for sub-problems converges to the optimal policy regarding $J(\pi)$. Based on this idea, we further derive offline RL algorithm to approximately find the solution.

4.1 Iterative Regularized Policy Optimization

Define the KL-regularized return for the AMG by

$$\bar{J}(\pi;\mu) := \mathop{\mathbb{E}}_{\rho_0,\pi,\mathbb{P}_T^N} \lfloor \min \rfloor_{\tau_0\in\mathcal{T}_0}^k \left[\mathop{\mathbb{E}}_{\tau_0,\pi,\mathbb{P}_T^N} \lfloor \min \rfloor_{\tau_1\in\mathcal{T}_1}^k \cdots \left[\mathop{\mathbb{E}}_{\tau_\infty,\pi} \left[\sum_{t=0}^{\infty} \gamma^t \Big(r(s_t,a_t) - \alpha D_{\mathrm{KL}}\big(\pi(\cdot|s_t) \,\big|\big|\, \mu(\cdot|s_t)\big)\Big)\right]\right]\right], \tag{9}$$

where $\alpha \geq 0$ is the strength of regularization, and μ is a reference policy to keep close with.

KL-regularized MDP is considered in previous works to enhance exploration, improve robustness to noise or insert expert knowledge [34–38]. Here, the idea is to constrain the optimized policy in neighbour of a reference policy so that the inner problem is adequately evaluated for such a small policy region.

To optimize $\bar{J}(\pi;\mu)$, we introduce the soft (N,k)-pessimistic Bellman backup operator:

$$\bar{\mathcal{B}}^*_{N,k} Q(s,a) = r(s,a) + \gamma \mathbb{E}_{\mathbb{P}_T^N}\left[\lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_\tau \left[\alpha \log \mathbb{E}_\mu \exp\left(\frac{1}{\alpha} Q(s',a')\right)\right]\right]. \tag{10}$$

Theorem 4 (Regularized Policy Optimization)**.** *The soft (N,k)-pessimistic Bellman backup operator $\bar{\mathcal{B}}^*_{N,k}$ is a contraction mapping. By starting from any function $Q : \mathcal{S}\times\mathcal{A}\to\mathbb{R}$ and repeatedly applying $\bar{\mathcal{B}}^*_{N,k}$, the sequence converges to $\bar{Q}^*_{N,k}$, with which the optimal policy for $\bar{J}(\pi;\mu)$ is obtained as $\bar{\pi}^*(a|s) \propto \mu(a|s)\exp\Big(\frac{1}{\alpha}\bar{Q}^*_{N,k}(s,a)\Big)$.*

Apparently, the solved policy $\bar{\pi}^*$ depends on the reference policy μ, and setting μ arbitrarily aforehand can result in suboptimal policy for $J(\pi)$. In fact, we can construct a sequence of sub-problems with the objective $\bar{J}(\pi;\mu)$, where μ is chosen as an improved policy from last sub-problem. By successively solving them, the optimal policy that maximizes $J(\pi)$ is attained.

Theorem 5 (Iterative Regularized Policy Optimization)**.** *By starting from any stochastic policy $\pi_0 : s\to\Delta(\mathcal{A})$ and repeatedly finding $\pi_{i+1} : \bar{J}(\pi_{i+1};\pi_i) > \bar{J}(\pi_i;\pi_i)$, the sequence of $\{\pi_i\}$ converges to the optimal policy π^* for $J(\pi)$. Besides, policies in the sequence are monotonically improved regarding $J(\pi)$, i.e., $J(\pi_{i+1}) \geq J(\pi_i)$.*

Ideally, by combining Theorems 4 and 5, the optimal solution for $J(\pi)$ can be obtained by infinitely applying soft pessimistic Bellman backup operator for each of the sequential sub-problems.

Remark 2 (Iterative Regularized Policy Optimization as Expectation–Maximization with Structured Variational Posterior). *According to Theorem 2, PMDB can be recovered with the converged Q-function $Q^{\pi^*}_{N,k}$. From an end-to-end view, we have an initial dynamics belief $\mathbb{P}^{sa}_T$, then via the calculation based on the belief samples and the reward function, we obtain the updated dynamics belief $\widetilde{\mathbb{P}}^{sa}_T$. It is likely that we are doing some form of posterior inference, where the evidence comes from the reward function. In fact, the iterative regularized policy optimization can be formally recast as an Expectation-Maximization algorithm for offline policy optimization, where the Expectation step correponds to a structured variational inference procedure for dynamics. We elaborate it in Appendix C.*

4.2 Offline Reinforcement Learning with Pessimism-Modulated Dynamics Belief

While solving each sub-problem makes prominent update to policy compared with the policy gradient method, we may need to construct several sub-problems before optimally maximizing $J(\pi)$, then exactly solving each of the sub-problem can incur unnecessary computation. For practical consideration, next we introduce a smooth-evolving reference policy, with which the explicit boundary between sub-problems is blurred. Based on this reference policy, and by further adopting function approximator, we devise an offline RL algorithm to approximately maximize $J(\pi)$.

The idea of smooth-evolving reference policy is inspired by the soft-updated target network in deep RL literature [39, 40]. That is setting the reference policy as a slowly tracked copy of the policy being optimized. Formally, consider a parameterized policy π_ϕ with the parameter ϕ. In the process of optimizing π_ϕ, the reference policy is set as $\mu = \pi_{\phi'}$, where ϕ' is the moving average of $\phi : \phi' \leftarrow \omega_1\phi + (1-\omega_1)\phi'$. With small enough ω_1, the Q-value of the state-action pairs induced by $\pi_{\phi'}$ (or its slight variant) can be sufficiently evaluated, before being used to update the policy. Next, we detail the loss functions to learn Q-value and policy with neural network approximators.

Denote the parameterized Q-function by Q_θ with the parameter θ. It is trained by minimizing the Bellman residual of both the AMG and the empirical MDP:

$$L_Q(\theta) = \mathbb{E}_{(s,a,\mathcal{T})\sim\mathcal{D}'}\left[\left(Q_\theta(s,a) - \widehat{Q}_{\text{AMG}}(s,a)\right)^2\right] + \mathbb{E}_{(s,a,s')\sim\mathcal{D}}\left[\left(Q_\theta(s,a) - \widehat{Q}_{\text{MDP}}(s,a)\right)^2\right], \quad (11)$$

with

$$\widehat{Q}_{\text{AMG}}(s,a) = r(s,a) + \gamma \lfloor \min \rfloor_{\tau\in\mathcal{T}}^{k} \mathbb{E}_\tau \left[\alpha \log \mathbb{E}_{\pi_{\phi'}} \exp\left(\frac{1}{\alpha} Q_{\theta'}(s',a')\right)\right], \quad (12)$$

$$\widehat{Q}_{\text{MDP}}(s,a) = r(s,a) + \gamma \cdot \alpha \log \mathbb{E}_{\pi_{\phi'}} \exp\left(\frac{1}{\alpha} Q_{\theta'}(s',a')\right), \quad (13)$$

where $Q_{\theta'}$ represent the target Q-value softly updated for stability [40], i.e., $\theta' \leftarrow \omega_2\theta + (1-\omega_2)\theta'$, and $\mathcal{D}'$ is the on-policy data buffer for the AMG. Since the game transition is known, the game can be executed with multiple counterparts in parallel, and the buffer only collects the latest sample for each of them. To promote direct learning from $\mathcal{D}$, we also include the Bellman residual of the empirical MDP in (11).

As with policy update, Theorem 4 states that the optimal policy for $\bar{J}(\pi;\mu)$ is propotional to $\mu(a|s)\exp\left(\frac{1}{\alpha}\bar{Q}^*_{N,k}(s,a)\right)$. Then, we update π_ϕ by supervisedly learning this policy, with μ and $\bar{Q}^*_{N,k}$ replaced by the smooth-evolving reference policy and the learned Q-value:

$$\begin{aligned} L_P(\phi) &= \mathbb{E}_{s\sim\mathcal{D}\cup\mathcal{D}'}\left[D_{\text{KL}}\left(\frac{\pi_{\phi'}(\cdot|s)\exp\left(\frac{1}{\alpha}Q_\theta(s,\cdot)\right)}{\mathbb{E}_{\pi_{\phi'}}\left[\exp\left(\frac{1}{\alpha}Q_\theta(s,a)\right)\right]} \,\middle\|\, \pi_\phi(\cdot|s)\right)\right] \\ &= A\cdot\mathbb{E}_{\substack{s\sim\mathcal{D}\cup\mathcal{D}' \\ a\sim\pi_{\phi'}}}\left[\exp\left(\frac{1}{\alpha}Q_\theta(s,a)\right)\log\pi_\phi(a|s)\right] + B, \end{aligned} \quad (14)$$

where A and B are constant terms. In general, (14) can be replaced by any tractable function that measures the similarity of distributions. For example, when π_ϕ is Gaussian, we can apply the recent proposed β-NLL [41], in which each data point's contribution to the negative log-likelihood loss is weighted by the β-exponentiated variance to improve learning heteroscedastic behavior.

To summarize, the algorithm alternates between collecting on-policy data samples and updating the function approximators. In detail, the latter procedure includes updating the policy with (11), updating the Q-value with (14), and updating reference policy as well as target Q-value with the moving-average rule. The complete algorithm is listed in Appendix D.

5 Experiments

Through the experiments, we aim to answer the following questions: 1) How does the proposed approach compared to the previous SoTA offline RL algorithms on standard benchmark? 2) How does the learning process in the AMG connect with the performance change in the original MDP? 3) Section 3 presents the monotonicity of $J(\pi)$ over N and k for any specified π, and it is easy to verify that this statement also holds when considering optimal policy for each setting of (N,k). However, with neural network approximator, our proposed offline RL algorithm approximately solves the AMG. Then, how is the monotonicity satisfied in this case?

We consider the Gym domains in the D4RL benchmark [42] to answer these questions. As PMDB relies on the initial dynamics belief, inserting additional knowledge into the initial dynamics belief will result in unfair comparison. To avoid that, we consider an uniform distribution over dynamics model ensembles as the initial belief. The dynamics model ensembles are trained in supervised manner with the offline dataset. This is similar to previous model-based works [6, 21], where the dynamics model ensembles are considered for dynamics uncertainty quantification. Since hyperparameter tuning for offline RL algorithms requires extra online test for each task, we purposely keep the same hyperparameters for all tasks, except when answering the last question. Especially, the hyperparameters in sampling procedure are $N = 10$ and $k = 2$. The more detailed setup for experiments and hyperparameters can be found in Appendix E.

5.1 Performance Comparison

We compare the proposed offline RL algorithm with the baselines including: BEAR [5] and BRAC [7], the model-free approaches based on policy constraint, CQL [9], the model-free approach by penalizing Q-value, EDAC [13], the previous SoTA on the D4RL benchmark, MOReL [6], the model-based approach which terminates the trajectory if the dynamics uncertainty exceeds a certain degree, and BC, the behavior cloning method. These approaches are evaluated on a total of eighteen domains involving three environments (hopper, walker2d, halfcheetah) and six dataset types (random, medium, expert, medium-expert, medium-replay, full-replay) per environment.

The results are summarized in Table 1. Our approach PMDB obviously improves over the previous SoTA on 9 tasks and performs competitively in the rest. Although EDAC achieves better performance in walker2d with several dataset types, its hyperparameters are tuned individually for each task. As presented later, the experiments on the impact of hyperparameters indicate that larger k or smaller N could generate better results for walker2d and halfcheetah. We also find that PMDB significantly outperforms MOReL, another model-based approach. It is encouraging that our model-based approach achieves competitive or better performance compared with the SoTA model-free approach, as model-based approach naturally has better support for multi-task learning and transfer learning, where the offline data from relevant tasks can be further leveraged.

Task Name	BC	BEAR	BRAC	CQL	MOReL	EDAC	PMDB
hopper-random	3.7±0.6	3.6±3.6	8.1±0.6	5.3±0.6	**38.1±10.1**	25.3±10.4	32.7±0.1
hopper-medium	54.1±3.8	55.3±3.2	77.8±6.1	61.9±6.4	84.0±17.0	101.6±0.6	**106.8±0.2**
hopper-expert	107.7±9.7	39.4±20.5	78.1±52.6	106.5±9.1	80.4±34.9	**110.1±0.1**	**111.7±0.3**
hopper-medium-expert	53.9±4.7	66.2±8.5	81.3±8.0	96.9±15.1	105.6±8.2	**110.7±0.1**	**111.8±0.6**
hopper-medium-replay	16.6±4.8	57.7±16.5	62.7±30.4	86.3±7.3	81.8±17.0	101.0±0.5	**106.2±0.6**
hopper-full-replay	19.9±12.9	54.0±24.0	107.4±0.5	101.9±0.6	94.4±20.5	105.4±0.7	**109.1±0.2**
walker2d-random	1.3±0.1	4.3±1.2	1.3±1.4	5.4±1.7	16.0±7.7	16.6±7.0	**21.8±0.1**
walker2d-medium	70.9±11.0	59.8±40.0	59.7±39.9	79.5±3.2	72.8±11.9	92.5±0.8	**94.2±1.1**
walker2d-expert	108.7±0.2	110.1±0.6	55.2±62.2	109.3±0.1	62.6±29.9	**115.1±1.9**	**115.9±1.9**
walker2d-medium-expert	90.1±13.2	107.0±2.9	9.3±18.9	109.1±0.2	107.5±5.6	**114.7±0.9**	111.9±0.2
walker2d-medium-replay	20.3±9.8	12.2±4.7	40.1±47.9	76.8±10.0	40.8±20.4	**87.1±2.3**	79.9±0.2
walker2d-full-replay	68.8±17.7	79.6±15.6	96.9±2.2	94.2±1.9	84.8±13.1	**99.8±0.7**	95.4±0.7
halfcheetah-random	2.2±0.0	12.6±1.0	24.3±0.7	31.3±3.5	**38.9±1.8**	28.4±1.0	**37.8 ± 0.2**
halfcheetah-medium	43.2±0.6	42.8±0.1	51.9±0.3	46.9±0.4	60.7±4.4	65.9±0.6	**75.6± 1.3**
halfcheetah-expert	91.8±1.5	92.6±0.6	39.0±13.8	97.3±1.1	8.4±11.8	**106.8±3.4**	**105.7± 1.0**
halfcheetah-medium-expert	44.0±1.6	45.7±4.2	52.3±0.1	95.0±1.4	80.4±11.7	106.3±1.9	**108.5±0.5**
halfcheetah-medium-replay	37.6±2.1	39.4±0.8	48.6±0.4	45.3±0.3	44.5±5.6	61.3±1.9	**71.7±1.1**
halfcheetah-full-replay	62.9±0.8	60.1±3.2	78.0±0.7	76.9±0.9	70.1±5.1	84.6±0.9	**90.0±0.8**
Average	49.9	52.4	54.0	73.7	65.1	85.2	**88.2**

Table 1: Results for D4RL datasets. Each result is the normalized score computed as (score − random policy score) / (expert policy score − random policy score), ± standard deviation. The score of our proposed approach is averaged over 4 random seeds, and the results of the baselines are taken from [13].

5.2 Learning in Alternating Markov Game

Figure 2 presents the learning curves in the AMG, as well as the received return when deploying the policy being learned in true MDP. We observe that the performance in the AMG closely tracks the true performance from the lower side, implying that it can act as a reasonable surrogate to evaluate/optimize performance for the true MDP. Besides, the performance in the AMG improves nearly monotonously, verifying the effectiveness of the proposed algorithm to approximately solve the game.

Recall that PMDB does not explicitly quantify dynamics uncertainty to penalize return, Figure 3 checks how the dynamics uncertainty and the Q-value of visited state-action pairs change during the

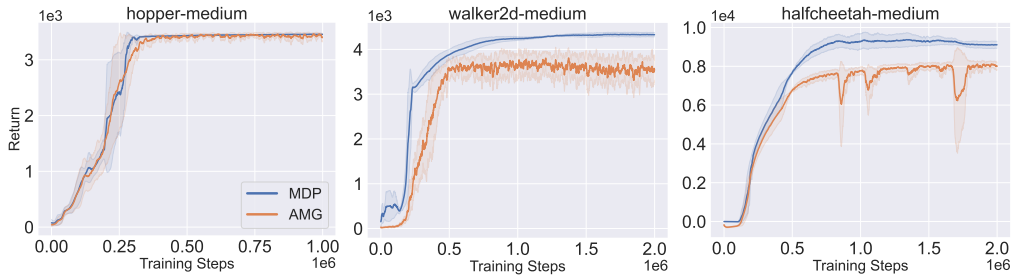

Figure 2: Learning and test curves for medium datasets.

learning process. The uncertainty is measured by the logarithm of standard deviation of the predicted means from the N dynamics samples, i.e., $\log\left(\text{std}\left(\mathbb{E}_\tau[s'];\tau\in\mathcal{T}\right)\right)$. The policy being learned is periodically tested in the AMG for ten trails, and we collect the whole ten trajectories of state-action pairs. The solid curves in Figure 3 denote the mean uncertainty and Q-value over the collected pairs, and shaded regions denote the standard deviation. From the results, the dynamics uncertainty first sharply decreases and then keeps a slowly increasing trend. Besides, in the long-term view, the Q-value is correlated with the degree of uncertainty negatively in the first phase and positively in the second phase. This indicates that the policy first moves to the in-distribution region and then tries to get away by resorting to the generalization of dynamics model.

Figure 3: Change on the dynamics uncertainty and Q-value of the encountered state-action pairs during learning process. The dynamics uncertainty of state-action pair (s,a) is measured by $\log\left(\text{std}\left(\mathbb{E}_{\tau(\cdot|s,a)}[s'];\tau\in\mathcal{T}\right)\right)$.

5.3 Practical Impact of Hyperparameters in Sampling Procedure

Table 4 lists the impact of k. In each setting, we evaluate the learned policy in both the true MDP and the AMG where the policy is learned. The performance in the AMGs improve when increasing k. This is consistent with the theoretical part, even that we approximately solve the game. Regarding the performance in true MDPs, we notice that $k=2$ corresponds to the best performance for hopper, but for the others $k=3$ is better. This indicates that tuning hyperparameter online can further improve the performance. The impact of N is presented in Appendix F, suggesting the opposite monotonicity.

	hopper-medium		walker2d-medium		halfcheetah-medium	
k	MDP	AMG	MDP	AMG	MDP	AMG
1	106.2±0.2	91.6±2.2	82.6±0.5	33.3±2.6	70.7±0.8	63.1±0.2
2	106.8±0.2	105.2±1.6	94.2±1.1	77.2±3.7	75.6±1.3	67.3±1.1
3	90.8±17.5	106.6±2.1	105.1±0.2	82.5±0.5	77.3±0.5	70.1±0.2

Table 2: Impact of k, with $N=10$.

6 Discussion

We proposed model-based offline RL with Pessimism-Modulated Dynamics Belief (PMDB), a framework to reliably learn policy from offline dataset, with the ability of leveraging dynamics prior knowledge. Empirically, the proposed approach outperforms the previous SoTA in a wide range of D4RL tasks. Compared to the previous model-based approaches, we characterize the impact of dynamics uncertainty through biased sampling from the dynamics belief, which implicitly induces PMDB. As PMDB is with the form of reweighting an initial dynamics belief, it provides a principled way to insert prior knowledge via the belief to boost policy learning. However, posing a valuable dynamics belief for arbitrary task is challenging, as the expert knowledge is not always available. Besides, an over-aggressive belief may still incur high-risk behavior in reality. Encouragingly, recent works have done active research to learn data-driven prior from relevant tasks. We believe that integrating them as well as developing safe criterion to design/learn dynamics belief would further promote practical deployment of offline RL.

References

[1] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *IEEE ICRA*, 2017.

[2] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A. Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE TITS*, 2021.

[3] Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys*, 2021.

[4] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *ICML*, 2019.

[5] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy Q-learning via bootstrapping error reduction. In *NeurIPS*, 2019.

[6] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOReL: Model-based offline reinforcement learning. In *NeurIPS*, 2020.

[7] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[8] Seyed Kamyar Seyed Ghasemipour, Dale Schuurmans, and Shixiang Shane Gu. EMaQ: Expected-max Q-learning operator for simple yet effective offline and online RL. In *ICML*, 2021.

[9] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In *NeurIPS*, 2020.

[10] Ilya Kostrikov, Rob Fergus, Jonathan Tompson, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization. In *ICML*, 2021.

[11] Ofir Nachum, Bo Dai, Ilya Kostrikov, Yinlam Chow, Lihong Li, and Dale Schuurmans. AlgaeDICE: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074*, 2019.

[12] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for Bayesian uncertainty in deep learning. In *NeurIPS*, 2019.

[13] Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified Q-ensemble. In *NeurIPS*, 2021.

[14] Chenjia Bai, Lingxiao Wang, Zhuoran Yang, Zhihong Deng, Animesh Garg, Peng Liu, and Zhaoran Wang. Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning. *arXiv preprint arXiv:2202.11566*, 2022.

[15] Andrew G Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. In *NeurIPS*, 2020.

[16] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. In *NeurIPS*, 2019.

[17] Andy Zeng, Shuran Song, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *IEEE T-RO*, 2020.

[18] Amy Zhang, Clare Lyle, Shagun Sodhani, Angelos Filos, Marta Kwiatkowska, Joelle Pineau, Yarin Gal, and Doina Precup. Invariant causal prediction for block MDPs. In *ICML*, 2020.

[19] Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *ICLR*, 2021.

[20] Philip J Ball, Cong Lu, Jack Parker-Holder, and Stephen Roberts. Augmented world models facilitate zero-shot dynamics generalization from a single offline environment. In *ICML*, 2021.

[21] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *NeurIPS*, 2020.

[22] Cong Lu, Philip J. Ball, Jack Parker-Holder, Michael A. Osborne, and Stephen J. Roberts. Revisiting design choices in model-based offline reinforcement learning. In *ICLR*, 2022.

[23] Arnab Nilim and Laurent El Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 2005.

[24] Wolfram Wiesemann, Daniel Kuhn, and Berç Rustem. Robust Markov decision processes. *Mathematics of Operations Research*, 2013.

[25] Elita A. Lobo, Mohammad Ghavamzadeh, and Marek Petrik. Soft-robust algorithms for batch reinforcement learning. *arXiv preprint arXiv:2011.14495*, 2020.

[26] Esther Derman, Daniel J. Mankowitz, Timothy A. Mann, and Shie Mannor. Soft-robust actor-critic policy-gradient. In *UAI*, 2018.

[27] Marek Petrik and Reazul Hasan Russel. Beyond confidence regions: Tight bayesian ambiguity sets for robust MDPs. In *NeurIPS*, 2019.

[28] Bahram Behzadian, Reazul Hasan Russel, Marek Petrik, and Chin Pang Ho. Optimizing percentile criterion using robust MDPs. In *AISTATS*, 2021.

[29] Romain Laroche, Paul Trichelair, and Remi Tachet Des Combes. Safe policy improvement with baseline bootstrapping. In *ICML*, 2019.

[30] Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High-confidence off-policy evaluation. In *AAAI*, 2015.

[31] Chin Pang Ho, Marek Petrik, and Wolfram Wiesemann. Fast Bellman updates for robust MDPs. In *ICML*, 2018.

[32] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NeurIPS*, 1999.

[33] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *ICML*, 2014.

[34] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *ICML*, 2017.

[35] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *ICML*, 2018.

[36] Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized Markov decision processes. In *ICML*, 2019.

[37] Alexandre Galashov, Siddhant M. Jayakumar, Leonard Hasenclever, Dhruva Tirumala, Jonathan Schwarz, Guillaume Desjardins, Wojciech M. Czarnecki, Yee Whye Teh, Razvan Pascanu, and Nicolas Heess. Information asymmetry in KL-regularized RL. In *ICLR*, 2019.

[38] Noah Y. Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin A. Riedmiller. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. In *ICLR*, 2020.

[39] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 2015.

[40] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *ICLR*, 2016.

[41] Maximilian Seitzer, Arash Tavakoli, Dimitrije Antic, and Georg Martius. On the pitfalls of heteroscedastic uncertainty estimation with probabilistic neural networks. *arXiv preprint arXiv:2203.09168*, 2022.

[42] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[43] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *NeurIPS*, 2019.

[44] Byung-Jun Lee, Jongmin Lee, and Kim Kee-Eung. Representation balancing offline model-based reinforcement learning. In *ICLR*, 2020.

[45] Toru Hishinuma and Kei Senda. Weighted model estimation for offline model-based reinforcement learning. In *NeurIPS*, 2021.

[46] Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. In *ICLR*, 2021.

[47] Tatsuya Matsushima, Hiroki Furuta, Yutaka Matsuo, Ofir Nachum, and Shixiang Gu. Deployment-efficient reinforcement learning via model-based offline optimization. In *ICLR*, 2021.

[48] Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. COMBO: Conservative offline model-based policy optimization. In *NeurIPS*, 2021.

[49] Aviv Tamar, Huan Xu, and Shie Mannor. Scaling up robust MDPs by reinforcement learning. *arXiv preprint arXiv:1306.6189*, 2013.

[50] Romain Laroche, Paul Trichelair, and Remi Tachet Des Combes. Safe policy improvement with baseline bootstrapping. In *ICML*, 2019.

[51] Daniel J. Mankowitz, Nir Levine, Rae Jeong, Abbas Abdolmaleki, Jost Tobias Springenberg, Timothy A. Mann, Todd Hester, and Martin A. Riedmiller. Robust reinforcement learning for continuous control with model misspecification. In *ICLR*, 2020.

[52] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *ICML*, 2017.

[53] Chen Tessler, Yonathan Efroni, and Shie Mannor. Action robust reinforcement learning and applications in continuous control. In *ICML*, 2019.

[54] Elena Smirnova, Elvis Dohmatob, and Jérémie Mary. Distributionally robust reinforcement learning. In *ICML Workshop*, 2019.

[55] H.A. David and H.N. Nagaraja. *Order Statistics*. Wiley, 2004.

[56] Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 6 and Appendix D, we also point out the possible future directions to improve.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 6.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix B.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The code is proprietary for now, but we are processing to make publicly accessible.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix E.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix E.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 5.
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]