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

Kaiyang Guo* **Yunfeng Shao** **Yanhui Geng**
Huawei Noah's Ark Lab

Abstract

Model-based offline reinforcement learning (RL) aims to find highly rewarding policy, by leveraging a previously collected static dataset and a dynamics model. While learned through reuse of static dataset, the dynamics model's 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 and the reward are intrinsically different factors in context of MDP, 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 benchmark tasks.

1 Introduction

In typical paradigm of RL, the agent actively interacts with environment and receives feedback to promote policy improvement. The essential trial-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, for a number 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

*Corresponding to: guokaiyang@huawei.com

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

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 meaning 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.[2] Then, such a state-action pair should not be treated specially.

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/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 a 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:

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

[2] 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 done 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 a 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 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 part 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 $\mathcal{T}^{sa}$, which later acts as the state of secondary player. Formally, $\mathcal{T}^{sa}$ 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 $\tau^{sa}(\cdot)$ re-denotes the plausible system transition $\tau(\cdot|s,a)$ for short, and $\mathbb{P}_T^{sa}$ is a given belief distribution over τ^{sa}. 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 distinguish with uncertainty set, we call it candidate set. 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.

The secondary player receives the candidate set $\mathcal{T}^{sa}$ as state. Thus, its state space can be denoted by $\bar{\mathcal{S}}=\Delta^N(\mathcal{S})$, i.e., the n-fold Cartesian product of probability simplex over $\mathcal{S}$. Note that the 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 continue the game. In the following, we 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), and omit the superscript sa in τ^{sa}, $\mathcal{T}^{sa}$ and $\mathbb{P}_T^{sa}$ when it is clear from the context.

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:

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

where the subscripts of τ and $\mathcal{T}$ denote time step, the expectation is over $s_0\sim\rho_0$, $s_{t>0}\sim\tau_{t-1}(\cdot|s_{t-1},a_{t-1}),a_t\sim\pi(\cdot|s_t)$ 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 introduced AMG and the robust MDP are twofold: 1) With a belief distribution over transitions, robust MDP will select only part of its supports into uncertainty set, and the set elements are treated indiscriminatingly. It indicates 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, due to 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 values of k or N, the impact to performance evaluation is ascertained, 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 $\mathcal{T}$. 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 $J(\pi)=\mathbb{E}_{\rho_0,\pi}\left[Q^{\pi}_{N,k}(s_0,a_0)\right]$.*

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 $\widetilde{T}(s'|s,a)=\mathbb{E}_{\widetilde{\mathbb{P}}^{sa}_T}\left[\tau^{sa}(s')\right]$ is defined with the reweighted belief distribution $\widetilde{\mathbb{P}}^{sa}_T$:*

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

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

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 (7), τ^{sa} itself is random following the belief distribution, thus $\mathbb{E}_{\tau^{sa},\pi}\left[Q^{\pi}_{N,k}(s',a')\right]$, as a functional of τ^{sa}, is also a random variable, whose cumulative density function is determined by the belief distribution $\mathbb{P}^{sa}_T$. Intuitively, we can treat $\mathbb{E}_{\tau^{sa},\pi}\left[Q^{\pi}_{N,k}(s',a')\right]$ 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^*: $F\Big(\mathbb{E}_{\tau^*,\pi}\left[Q^{\pi}_{N,k}(s',a')\right]\Big)=\frac{k-1}{N-1}$, i.e., the transition with $\frac{k-1}{N-1}$-quantile pessimism indicator. Besides, when $\mathbb{E}_{\tau^{sa},\pi}\left[Q^{\pi}_{N,k}(s',a')\right]$ 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 $\left\{\tau^{sa}:\mathbb{P}^{sa}_T(\tau^{sa})>0,\mathbb{E}_{\tau^{sa},\pi}\left[Q^{\pi}_{N,k}(s',a')\right]\geq F^{-1}(\delta)\right\}$. In this sense, the AMG is a successive interpolation between MBRL and robust MDP.*

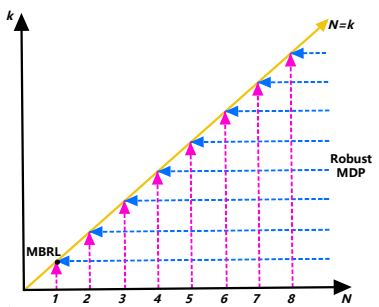

Figure 1: Monotonicity of Q-values. The arrows indicate the directions along which Q-values increase.

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. 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 monotonously improve 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 $\bar{\pi}^*(a|s)\propto\mu(a|s)\exp\left(\frac{1}{\alpha}\bar{Q}^*_{N,k}(s,a)\right)$.*

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 μ chosen as an improved policy from last sub-problem. By successively solving them, the impact from the initial reference policy is gradually eliminated.

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 $\{\pi_i\}$ monotonically improves regarding $J(\pi)$, i.e., $J(\pi_{i+1})\geq J(\pi_i)$. Especially, when $\pi_0(a|s)>0,\forall s,a$ and $\{\pi_i\}$ are obtained via regularized policy optimization in Theorem 4, we have $\frac{\pi_i(a|s)}{\pi_i(a'|s)}\to\infty$ for any s,a,a' such that $\lim_{i\to\infty}Q^{\pi_i}_{N,k}(s,a) > \lim_{i\to\infty}Q^{\pi_i}_{N,k}(s,a')$.*

Ideally, by combining Theorems 4 and 5, the policy for $J(\pi)$ can be continuously improved 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 convergence, then exactly solving each of them incurs 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. 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 in AMG and updating the function approximators. In detail, the latter procedure includes updating the Q-value with (11), updating the optimized policy with (14), and updating the target Q-value as well as the reference policy 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 G. The code is available online[3]

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. We use the v2 version of each dataset.

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. The later 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. The performance in the AMG closely tracks the true perfor-

[3]Code is released at `https://github.com/huawei-noah/HEBO/tree/master/PMDB` and `https://gitee.com/mindspore/models/tree/master/research/rl/pmdb`.

mance 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.

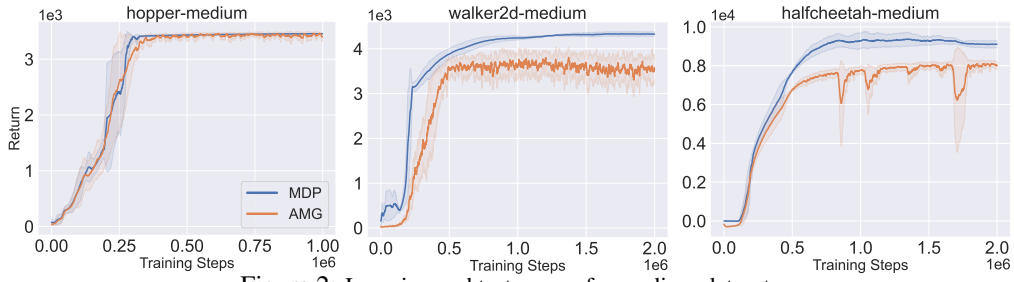

Figure 2: Learning and test curves for medium datasets.

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 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 trials, 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.

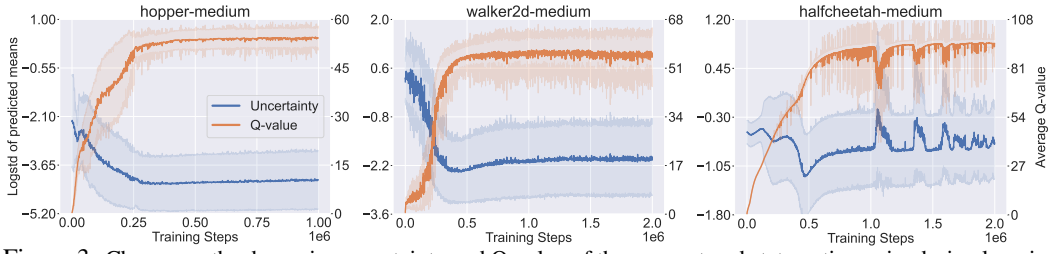

Figure 3: Change on the dynamics uncertainty and Q-value of the encountered state-action pairs during learning process. The dynamics uncertainty of state-action pair (s,a) is measured by $\log\left(\text{std}\left(\mathbb{E}_{\tau(\cdot|s,a)}[s'];\tau\in\mathcal{T}\right)\right)$.

In Figure 3, we also notice that the large dip of Q-value is accompanied with the sudden raise of dynamics uncertainty. We suspect this is due to the optimized policy being far from the dataset. We try to verify by checking the maximal return covered by the offline dataset. It shows that the maximal normalized returns provided by offline datasets are 99.6, 92.0 and 45.0 respectively for hopper, walker2d and halfcheetah, while the proposed approach achieves 106.8, 94.2 and 75.6. The policy optimization is more significant for halfcheetah (where we observe the large dip), indicating the policy should move further from the dataset.

The above finding also explains why the AMG performance in Figure 2 runs into large dip only for halfcheetah: with the larger dynamics uncertainty, the secondary player can choose the more pessimistic transition. However, we want to highlight that it is normal behavior of the proposed algorithm and does not mean instability, as we are handling the alternating Markov game, a specialization of zero-sum game. Besides, we can see that even when the AMG performance goes down the MDP performance is still stable.

5.3 Practical Impact of Hyperparameters in Sampling Procedure

Table 2 lists the impact of k. In each setting, we evaluate the learned policy in both the true MDP and the AMG. The performance in the AMGs improve when increasing k. This is consistent with the theoretical result, 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 H, 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 Related Works

Inadequate data coverage is the root of challenge in offline RL. Existing works differ in their methodology to reacting in face of limited system knowledge.

Model-free offline RL The prevalent idea is to find policy within the data manifold through model-free learning. Analogous to online RL, both policy-based and value-based approaches are devised to this end. Policy-based approaches directly constrain the optimized policy close to the behavior policy that collects data, via various measurements such as KL divergence [7], MMD [5] and action deviation [4, 8]. Value-based approaches instead reflect the policy regularization through value function. For example, CQL enforces small Q-value for OOD state-action pairs [9], AlgaeDICE penalizes return with the f-divergence between optimized and offline state-action distributions [11], and Fisher-BRC proposes a novel parameterization of the Q-value to encourage the generated policy close to the data [10]. Our proposed approach is more relevant to the value-based scope, and the key difference to existing works is that our Q-value is penalized through an adversarial choice of transition from plausible candidates.

Learning within the data manifold limits the degree to which the policy improves, and recent works attempt to relieve the restriction. Along the model-free line, EDAC [13] and PBRL [14] quantify uncertainty of Q-value via neural network ensemble, and assign penalty to Q-value depending on the uncertainty degree. In this way, the OOD state-action pairs are touchable if they pose low uncertainty on Q-value. 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].

Model-based offline RL Model-based approach is widely recognized due to its superior data efficiency. However, directly optimizing policy based on an offline learned model is vulnerable to model exploitation [22, 43]. A line of works improve the dynamics learning for seek of robustness [44] or adaptation [45] to distributional shift. In terms of policy learning, several works extend the idea from model-free approaches, and constrain the optimized policy close to the behavior policy when applying the dynamics model for planing [46] or policy optimization [47, 48]. There are also recent works incorporating uncertainty quantification of dynamics model to learn policy beyond data coverage. Especially, MOPO [21] and MOReL [6] perform policy improvement in states that may not directly occur in the static offline dataset, but can be predicted by leveraging the power of generalization. Compared to them, our approach does not explicitly characterize the dynamics uncertainty as reward penalty. There are also relevant works dealing with model ambiguity in light of Bayesian decision theory, which are discussed in Appendix A.

7 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] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, 2017.

[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] RT Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 2000.

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

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

[52] 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.

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

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

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

[56] Marc Rigter, Paul Duckworth, Bruno Lacerda, and Nick Hawes. Planning for risk-aversion and expected value in MDPs. In *ICAPS*, 2022.

[57] Esther Derman, Daniel Mankowitz, Timothy Mann, and Shie Mannor. A Bayesian approach to robust reinforcement learning. In *Uncertainty in Artificial Intelligence Conference*, pages 648–658, 2020.

[58] Marc Rigter, Bruno Lacerda, and Nick Hawes. Risk-averse Bayes-adaptive reinforcement learning. In *NeurIPS*, 2021.

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

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

[61] Marc Rigter, Bruno Lacerda, and Nick Hawes. RAMBO-RL: Robust adversarial model-based offline reinforcement learning. *arXiv preprint arXiv:2204.12581*, 2022.

Checklist

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 7 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 7.
   (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)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix G.
   (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 G.
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]

A Additional Related Works

Robust MDP and CVaR Criterion Bayesian decision theory provides a principled formalism for decision making under uncertainty. Robust MDP [23, 24], Conditional Value at Risk (CVaR) [49] and our proposed criterion can be deemed as the specializations of Bayesian decision theory, but derived from different principles and with different properties.

Robust MDP is proposed as a surrogate to optimize the percentile performance. Early works mainly focus on the algorithmic design and theoretical analysis in the tabular case [23, 24] or under linear function approximation [50]. Recently, it has also been extended to continuous action spaces and nonlinear cases, by integrating the advances of deep RL [51, 52]. Meanwhile, a variety of works generalize the uncertainty in regard to system disturbance [53] and action disturbance [54, 55]. Although robust MDP produces robust policy, it purely focuses on the percentile performance, and ignoring the other possibilities is reported to be over-conservative [25–27].

CVaR instead considers the average performance of the worst δ-fraction possibilities. Despite involving more information about the stochasticity, CVaR is still solely from the pessimistic view. Recent works propose to improve by maximizing the convex combination of mean performance and CVaR [25], or maximizing mean performance under CVaR constraint [56]. However, they are intractable regarding policy optimization, i.e., proved as an NP-hard problem or relying on heuristic. As comparison, the proposed AMG formulation presents an alternative way to tackle the entire spectrum of plausible transitions while also give more attention on the pessimistic parts. Besides, the policy optimization is with theoretical guarantee.

Apart from offline RL, Bayesian decision theory is also applied in other RL settings. Particularly, Bayesian RL considers that new observations are continually received and utilized to make adaptive decision. The goal of Bayesian RL is to fast **explore** and adapt, while that of offline RL is to sufficiently **exploit** the offline dataset to generate the best-effort policy supported or surrounded by the dataset. Recently, Bayesian robust RL [57] integrates the idea of robust MDP in the setting of Bayesian RL, where the uncertainty set is constructed to produce robust policy, and will be updated upon new observations to alleviate the degree of conservativeness. Besides, CVaR criterion is also considered in Bayesian RL [58].

B Theorem Proof

We first present and prove the fundamental inequalities applied to prove the main theorem, and then present the proofs for Sections 3 and 4 respectively. For conciseness, the subscripts N, k are omitted in Q-value and Bellman backup operator when clear from the context.

B.1 Preliminaries

Lemma 1. *Let $\lfloor\min\rfloor_i^k\, x_i$ denote the kth minimum in $\{x_i\}$, then*

$$\min_i (x_i - y_i) \le \lfloor\min\rfloor_i^k\, x_i - \lfloor\min\rfloor_i^k\, y_i \le \max_i (x_i - y_i), \quad \forall k = 1, 2, \cdots, N,$$

where N is the size of both $\{x_i\}$ and $\{y_i\}$.

Proof of Lemma 1. Denote $i^* = \arg\lfloor\min\rfloor_i^k\, x_i$ and $j^* = \arg\lfloor\min\rfloor_i^k\, y_i$. Next, we prove the first inequality. The proof is done by dividing into two cases.

Case 1: $y_{i^*} \ge y_{j^*}$

It is easy to check

$$\lfloor\min\rfloor_i^k\, x_i - \lfloor\min\rfloor_i^k\, y_i = x_{i^*} - y_{j^*} \ge x_{i^*} - y_{i^*} \ge \min_i (x_i - y_i).$$

Case 2: $y_{i^*} < y_{j^*}$

We prove by contradiction. Let $\mathcal{S}_x = \left\{ \arg\lfloor\min\rfloor_i^l\, x_i \;\middle|\; l = 1, 2, \cdots, k-1 \right\}$. Assume

$$y_s < y_{j^*}, \quad \forall s \in \mathcal{S}_x.$$

Since y_{j^*} is the kth minimum, the above assumption implies $\mathcal{S}_x \subseteq \mathcal{S}_y$. Meanwhile, according to the condition of case 2, $i^* \in \mathcal{S}_y$. Put these together, we have $\{i^*\} \cup \mathcal{S}_x \subseteq \mathcal{S}_y$. According to the definition of i^*, we know $i^* \notin \mathcal{S}_x$. This concludes that $\mathcal{S}_y$ has at least k elements, contradicting with its definition.

Thus,

$$\exists \bar{s} \in \mathcal{S}_x : y_{\bar{s}} \geq y_{j^*}.$$

By applying the above inequality and $x_{\bar{s}} \leq x_{i^*}$, we have

$$\lfloor \min \rfloor_i^k \, x_i - \lfloor \min \rfloor_i^k \, y_i = x_{i^*} - y_{j^*} \geq x_{\bar{s}} - y_{\bar{s}} \geq \min_i \left(x_i - y_i \right).$$

In summary, we have $\min_i \left(x_i - y_i \right) \leq \lfloor \min \rfloor_i^k \, x_i - \lfloor \min \rfloor_i^k \, y_i$ for both cases.

The second inequality can be proved by resorting to the first one. By respectively replacing x_i and y_i with $-x_i$ and $-y_i$ in first inequality, we obtain

$$\min_i \left(-x_i + y_i \right) \leq \lfloor \min \rfloor_i^k \, \left(-x_i \right) - \lfloor \min \rfloor_i^k \, \left(-y_i \right),$$

which can be rewritten as

$$\begin{aligned} \max_i \left(x_i - y_i \right) &\geq -\Big(\lfloor \min \rfloor_i^k \, \left(-x_i \right) - \lfloor \min \rfloor_i^k \, \left(-y_i \right) \Big) \\ &= \lfloor \min \rfloor_i^{N-k} \, x_i - \lfloor \min \rfloor_i^{N-k} \, y_i, \end{aligned}$$

where the last equation is due to $\lfloor \min \rfloor_i^k \, \left(-x_i \right) = -\lfloor \min \rfloor_i^{N-k} \, x_i$. As the above inequalities holds for any $k \in \{1, 2, \cdots, N\}$, we can replace $N - k$ by k, and this is right the second inequality in Lemma 1.

□

Corollary 1.

$$\left| \lfloor \min \rfloor_i^k \, x_i - \lfloor \min \rfloor_i^k \, y_i \right| \leq \max_i |x_i - y_i|, \quad \forall k = 1, 2, \cdots, N.$$

Proof of Corollary 1. The inequality can be attained through simple derivation based on Lemma 1, i.e.,

$$\lfloor \min \rfloor_i^k \, x_i - \lfloor \min \rfloor_i^k \, y_i \geq \min_i \left(x_i - y_i \right) \geq \min_i \left(- \left| x_i - y_i \right| \right) = - \max_i \left| x_i - y_i \right|$$

and

$$\lfloor \min \rfloor_i^k \, x_i - \lfloor \min \rfloor_i^k \, y_i \leq \max_i \left(x_i - y_i \right) \leq \max_i \left| x_i - y_i \right|.$$

Put them together, we obtain

$$\left| \lfloor \min \rfloor_i^k \, x_i - \lfloor \min \rfloor_i^k \, y_i \right| \leq \max_i \left| x_i - y_i \right|.$$

□

B.2 Proofs for Section 3

Proof of Theorem 1. Let Q_1 and Q_2 be two arbitrary Q function, then

$$
\begin{aligned}
&\|\mathcal{B}^\pi Q_1 - \mathcal{B}^\pi Q_2\|_\infty \\
&= \gamma \max_{s,a} \left| \mathbb{E}_{\mathbb{P}_T^N} \left[\lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_{\tau,\pi}\big[Q_1(s',a')\big] \right] - \mathbb{E}_{\mathbb{P}_T^N} \left[\lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_{\tau,\pi}\big[Q_2(s',a')\big] \right] \right| \\
&= \gamma \max_{s,a} \left| \mathbb{E}_{\mathbb{P}_T^N} \left[\lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_{\tau,\pi}\big[Q_1(s',a')\big] - \lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_{\tau,\pi}\big[Q_2(s',a')\big] \right] \right| \\
&\le \gamma \max_{s,a} \left(\mathbb{E}_{\mathbb{P}_T^N} \left| \lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_{\tau,\pi}\big[Q_1(s',a')\big] - \lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_{\tau,\pi}\big[Q_2(s',a')\big] \right| \right) \\
&\le \gamma \max_{s,a} \left(\mathbb{E}_{\mathbb{P}_T^N} \left[\max_{\tau\in\mathcal{T}} \left| \mathbb{E}_{\tau,\pi}\big[Q_1(s',a') - Q_2(s',a')\big] \right| \right] \right) \\
&\le \gamma \max_{s,a} \left(\mathbb{E}_{\mathbb{P}_T^N} \|Q_1 - Q_2\|_\infty \right) \\
&= \gamma \|Q_1 - Q_2\|_\infty,
\end{aligned}
$$

where the second inequality is due to Corollary 1. Thus, the pessimistic Bellman update operator $\mathcal{B}^\pi$ is a contraction mapping.

After convergence, it is easy to check $J(\pi) = \mathbb{E}_{\rho_0,\pi}\Big[Q^\pi(s_0,a_0)\Big]$ by recursively unfolding Q-function. □

Proof of Theorem 2. For conciseness, in this proof we drop the superscript sa in τ^{sa}, $\mathbb{P}_T^{sa}$ and $\widetilde{\mathbb{P}}_T^{sa}$.

The proof is based on the definition and the probability density function of order statistic [59]. For any random variables $X_1, X_2, \cdots, X_N$, their kth order statistic is defined as $\lfloor \min \rfloor_{n\in\{1,\cdots,N\}}^k X_n$, which is another random variable. Particularly, when $X_1, X_2, \cdots, X_N$ are independent and identically distributed following a probability density function $\mathbb{P}(x)$, the order statistic is with the probability density function

$$
\mathbb{P}_{N,k}(x) = \underbrace{\frac{N!}{(k-1)!(N-k)!}}_{C} \mathbb{P}(x)\big[F(x)\big]^{k-1}\big[1-F(x)\big]^{N-k},
$$

where $F(x)$ is the cumulative distribution corresponding to $\mathbb{P}(x)$.

Let $g(\tau) = \mathbb{E}_{\tau,\pi}\big[Q^\pi(s',a')\big]$ for short. As τ is random following the belief distribution, g as the functional of τ is also a random variable. Its sample can be drawn by

$$
g = g(\tau), \quad \tau \sim \mathbb{P}_T(\tau).
$$

As the elements in $\mathcal{T}$ are independent and identically distributed samples from $\mathbb{P}_T(\tau)$, the elements in $\mathcal{G} = \{g(\tau) \mid \tau \in \mathcal{T}\}$ are also independent and identically distributed. Thus, $\lfloor \min \rfloor_{g\in\mathcal{G}}^k g$ is their kth order statistic, and we have

$$
\begin{aligned}
\mathbb{E}_{\mathbb{P}_T^N}\Big[\lfloor \min \rfloor_{\tau\in\mathcal{T}}^k g(\tau)\Big] = \mathbb{E}_{\mathbb{P}_T^N}\Big[\lfloor \min \rfloor_{g\in\mathcal{G}}^k g\Big] &= \int_{-\infty}^{\infty} \mathbb{P}_{N,k}(g) g dg \\
&= C\int_{-\infty}^{\infty} \mathbb{P}(g)\big[F(g)\big]^{k-1}\big[1-F(g)\big]^{N-k} g dg, \\
&= C\int_{-\infty}^{\infty} \left[\int_{\tau:g(\tau)=g} \mathbb{P}_T(\tau) d\nu(\tau) \right] \big[F(g)\big]^{k-1}\big[1-F(g)\big]^{N-k} g dg, \\
&= C\int_{-\infty}^{\infty} \int_{\tau:g(\tau)=g} \mathbb{P}_T(\tau)\big[F(g)\big]^{k-1}\big[1-F(g)\big]^{N-k} g d\nu(\tau) dg,
\end{aligned}
$$

$$
\begin{aligned}
&\overset{(*)}{=} C \int_{\tau} \int_{g=g(\tau)} \mathbb{P}_T(\tau)\big[F(g)\big]^{k-1}\big[1-F(g)\big]^{N-k} g dg d\nu(\tau), \\
&= \int_{\tau} \underbrace{C\,\mathbb{P}_T(\tau)\Big[F\big(g(\tau)\big)\Big]^{k-1}\Big[1-F\big(g(\tau)\big)\Big]^{N-k}}_{\widetilde{\mathbb{P}}_T(\tau)} g(\tau) d\nu(\tau) \\
&= \mathbb{E}_{\widetilde{\mathbb{P}}_T}[g(\tau)],
\end{aligned}
$$

where $\nu(\tau)$ is the reference measure based on which the belief distribution P_T^N is defined, and the equation $(*)$ is obtained by exchanging the orders of integration. The above equation can rewritten as

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

Taking this into consideration, the pessimistic Bellman backup operator in (6) is exactly the vanilla Bellman backup operator for the MDP with transition probability $\widetilde{T}(s'|s,a) = \mathbb{E}_{\widetilde{\mathbb{P}}_T}\left[\tau(s')\right]$. Then, evaluating/optimizing policy in the AMG is equivalent to evaluating/optimizing in this MDP.

To prove the property of w, we treat it as a composite function with form of $w\big(F(x)\big)$. Then, the derivative of w over F is

$$
\frac{\delta w}{\delta F} = F^{k-2}(1-F)^{N-k-1}\left[(k-1)-(N-1)F\right]. \tag{15}
$$

It is easy to check that $\frac{\delta w}{\delta F} \geq 0$ for $F \leq \frac{k-1}{N-1}$ and $\frac{\delta w}{\delta F} \leq 0$ for $F \geq \frac{k-1}{N-1}$. Thus, $w(F)$ reaches the maximum at $F = \frac{k-1}{N-1}$. Besides, as $F(\cdot)$ is the PDF of x, it monotonically increases with x. Put the monotonicity of w and F together, we know $w(F(x))$ first increases, then decreases with x and achieves the maximimum at $x^* = F^{-1}\left(\frac{k-1}{N-1}\right)$. □

Lemma 2 (Monotonicity of Pessimistic Bellman Backup Operator)**.** *Assume that $Q_1 \geq Q_2$ holds element-wisely, then $\mathcal{B}^\pi Q_1 \geq \mathcal{B}^\pi Q_2$ element-wisely.*

Proof of Lemma 2.

$$
\begin{aligned}
&\mathcal{B}^\pi Q_1(s,a) - \mathcal{B}^\pi Q_2(s,a) \\
&= \gamma \mathbb{E}_{\mathbb{P}_T^N}\Big[\lfloor\min\rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_{\tau,\pi}\big[Q_1(s',a')\big] - \lfloor\min\rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_{\tau,\pi}\big[Q_2(s',a')\big]\Big] \\
&\geq \gamma \mathbb{E}_{\mathbb{P}_T^N}\Big[\min_{\tau\in\mathcal{T}} \mathbb{E}_{\tau,\pi}\big[Q_1(s',a') - Q_2(s',a')\big]\Big] \\
&\geq 0, \qquad \forall s,a,
\end{aligned}
$$

where the first inequality is due to Lemma 1. □

Proof of Theorem 3. It is sufficient to prove $Q^\pi_{N,k+1} \geq Q^\pi_{N,k}, Q^\pi_{N+1,k} \leq Q^\pi_{N,k}$ and $Q^\pi_{N+1,N+1} \geq Q^\pi_{N,N}$ element-wisely. The idea is to first show $\mathcal{B}^\pi_{N,k+1}Q^\pi_{N,k} \geq Q^\pi_{N,k}, \mathcal{B}^\pi_{N+1,k}Q^\pi_{N,k} \leq Q^\pi_{N,k}$ and $\mathcal{B}^\pi_{N+1,N+1}Q^\pi_{N,N} \geq Q^\pi_{N,N}$. Then, the proof can be finished by recursively applying Lemma 2, for example:

$$
Q^\pi_{N,k+1} = \lim_{n\to\infty}\left(\mathcal{B}^\pi_{N,k+1}\right)^n Q^\pi_{N,k} \geq \cdots \geq \mathcal{B}^\pi_{N,k+1}Q^\pi_{N,k} \geq Q^\pi_{N,k}.
$$

Next, we prove the three inequalities in sequence.

$\boxed{\mathcal{B}^{\pi}_{N,k+1}Q^{\pi}_{N,k} \geq Q^{\pi}_{N,k}}$

$$
\begin{aligned}
&\mathcal{B}^{\pi}_{N,k+1}Q^{\pi}_{N,k}(s,a) \\
&= r(s,a) + \gamma \mathbb{E}_{\mathbb{P}_T^n}\left[\lfloor \min \rfloor^{k+1}_{\tau\in\mathcal{T}} \mathbb{E}_{\tau,\pi}\left[Q^{\pi}_{N,k}(s',a')\right]\right] \\
&\geq r(s,a) + \gamma \mathbb{E}_{\mathbb{P}_T^n}\left[\lfloor \min \rfloor^{k}_{\tau\in\mathcal{T}} \mathbb{E}_{\tau,\pi}\left[Q^{\pi}_{N,k}(s',a')\right]\right] \\
&= \mathcal{B}^{\pi}_{N,k}Q^{\pi}_{N,k}(s,a) \\
&= Q^{\pi}_{N,k}(s,a), \qquad \forall s,a,
\end{aligned}
$$

$\boxed{\mathcal{B}^{\pi}_{N+1,k}Q^{\pi}_{N,k} \leq Q^{\pi}_{N,k}}$

$$
\begin{aligned}
&\mathcal{B}^{\pi}_{N+1,k}Q^{\pi}_{N,k}(s,a) \\
&= r(s,a) + \gamma \mathbb{E}_{\mathcal{T}\sim\mathbb{P}_T^{N+1}}\left[\lfloor \min \rfloor^{k}_{\tau\in\mathcal{T}} \mathbb{E}_{\tau,\pi}\left[Q^{\pi}_{N,k}(s',a')\right]\right] \\
&= r(s,a) + \gamma \mathbb{E}_{\mathcal{T}'\sim\mathbb{P}_T^{N}}\left[\mathbb{E}_{\tau'\sim\mathbb{P}_T}\left[\lfloor \min \rfloor^{k}_{\tau\in\mathcal{T}'\cup\{\tau'\}} \mathbb{E}_{\tau,\pi}\left[Q^{\pi}_{N,k}(s',a')\right]\right]\right] \\
&\leq r(s,a) + \gamma \mathbb{E}_{\mathcal{T}'\sim\mathbb{P}_T^{N}}\left[\lfloor \min \rfloor^{k}_{\tau\in\mathcal{T}'} \mathbb{E}_{\tau,\pi}\left[Q^{\pi}_{N,k}(s',a')\right]\right] \\
&= \mathcal{B}^{\pi}_{N,k}Q^{\pi}_{N,k}(s,a) \\
&= Q^{\pi}_{N,k}(s,a), \qquad \forall s,a,
\end{aligned}
$$

where we divide the $(N+1)$-size set $\mathcal{T}$ into $\mathcal{T}'$ and $\{\tau'\}$, $\mathcal{T}'$ contains the first N elements and τ' is the last element. The second equality is due to the independence among the set elements.

$\boxed{\mathcal{B}^{\pi}_{N+1,N+1}Q^{\pi}_{N,N} \geq Q^{\pi}_{N,N}}$

$$
\begin{aligned}
&\mathcal{B}^{\pi}_{N+1,N+1}Q^{\pi}_{N,N}(s,a) \\
&= r(s,a) + \gamma \mathbb{E}_{\mathcal{T}\sim\mathbb{P}_T^{N+1}}\left[\max_{\tau\in\mathcal{T}} \mathbb{E}_{\tau,\pi}\left[Q^{\pi}_{N,k}(s',a')\right]\right] \\
&= r(s,a) + \gamma \mathbb{E}_{\mathcal{T}'\sim\mathbb{P}_T^{N}}\left[\mathbb{E}_{\tau'\sim\mathbb{P}_T}\left[\max_{\tau\in\mathcal{T}'\cup\{\tau'\}} \mathbb{E}_{\tau,\pi}\left[Q^{\pi}_{N,k}(s',a')\right]\right]\right] \\
&\geq r(s,a) + \gamma \mathbb{E}_{\mathcal{T}'\sim\mathbb{P}_T^{N}}\left[\max_{\tau\in\mathcal{T}'} \mathbb{E}_{\tau,\pi}\left[Q^{\pi}_{N,k}(s',a')\right]\right] \\
&= \mathcal{B}^{\pi}_{N,k}Q^{\pi}_{N,k}(s,a) \\
&= Q^{\pi}_{N,k}(s,a), \qquad \forall s,a,
\end{aligned}
$$

□

B.3 Proofs for Section 4

Analogous to the policy evaluation for non-regularized case, we define the KL-regularized Bellman update operator for a given policy π by

$$
\bar{\mathcal{B}}^{\pi}_{N,k}Q(s,a) = r(s,a) + \gamma \mathbb{E}_{\mathbb{P}_T^N}\left[\lfloor \min \rfloor^{k}_{\tau\in\mathcal{T}} \mathbb{E}_{\tau,\pi}\Big[Q(s',a') - \alpha D_{\mathrm{KL}}\big(\pi(\cdot|s) \;\big|\big|\; \mu(\cdot|s)\big)\Big]\right]. \tag{16}
$$

It is easy to check all proofs in last subsection adapt well for the KL-regularized case. We state the corresponding theorems and lemma as below, and apply them to prove the theorems in Section 4.

Theorem 6 (Policy Evaluation for KL-Regularized AMG)**.** *The regularized (N,k)-pessimistic Bellman backup operator $\bar{\mathcal{B}}^{\pi}_{N,k}$ is a contraction mapping. By starting from any function $Q : \mathcal{S}\times\mathcal{A}\to\mathbb{R}$ and repeatedly applying $\bar{\mathcal{B}}^{\pi}_{N,k}$, the sequence converges to $\bar{Q}^{\pi}_{N,k}$, with which we have $\bar{J}(\pi;\mu) = \mathbb{E}_{\rho_0,\pi}\Big[\bar{Q}^{\pi}_{N,k}(s_0,a_0) - \alpha D_{\mathrm{KL}}\big(\pi(\cdot|s_0) \;\big|\big|\; \mu(\cdot|s_0)\big)\Big]$.*

Theorem 7 (Equivalent KL-Regularized MDP with Pessimism-Modulated Dynamics Belief)**.** *The KL-regularized alternating Markov game in* (9) *is equivalent to the KL-regularized MDP with tuple* $(\mathcal{S}, \mathcal{A}, \widetilde{T}, r, \rho_0, \gamma)$*, where the transition probability* $\widetilde{T}(s'|s,a) = \mathbb{E}_{\widetilde{\mathbb{P}}_T^{sa}}\left[\tau^{sa}(s')\right]$ *is defined with the reweighted belief distribution* $\widetilde{\mathbb{P}}_T^{sa}$*:*

$$\widetilde{\mathbb{P}}_T^{sa}(\tau^{sa}) \propto w\Big(\mathbb{E}_{\tau^{sa},\pi}\big[\bar{Q}_{N,k}^{\pi}(s',a')\big]; k, N\Big)\mathbb{P}_T^{sa}(\tau^{sa}), \tag{17}$$

$$w(x;k,N) = \big[F(x)\big]^{k-1}\big[1-F(x)\big]^{N-k}, \tag{18}$$

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.,* $x^* = F^{-1}\left(\frac{k-1}{N-1}\right)$*. Similar to the non-regularized case, the reweighting factor* w *reshapes the initial belief distribution towards being pessimistic in terms of* $\mathbb{E}_{\tau,\pi}\big[\bar{Q}_{N,k}^{\pi}(s',a')\big]$*.*

Lemma 3 (Monotonicity of Regularized Pessimistic Bellman Backup Operator)**.** *Assume that* $Q_1 \geq Q_2$ *holds element-wisely, then* $\bar{\mathcal{B}}_{N,k}^{\pi}Q_1 \geq \bar{\mathcal{B}}_{N,k}^{\pi}Q_2$ *element-wisely.*

Theorem 8 (Monotonicity in Regularized Alternating Markov Game)**.** *The converged Q-function* $\bar{Q}_{N,k}^{\pi}$ *are with the following properties:*

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

Proof of Theorem 4. The proof of contraction mapping basically follows the same steps in proof of Theorem 1 Let Q_1 and Q_2 be two arbitrary Q function.

$$\begin{aligned}
&\left\|\bar{\mathcal{B}}^* Q_1 - \bar{\mathcal{B}}^* Q_2\right\|_\infty \\
&= \gamma \max_{s,a}\Bigg|\mathbb{E}_{\mathbb{P}_T^N}\Bigg[\lfloor\min\rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_\tau\left[\alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_1(s',a')\right)\right] \\
&\qquad\qquad - \lfloor\min\rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_\tau\left[\alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_2(s',a')\right)\right]\Bigg]\Bigg| \\
&\leq \gamma \max_{s,a}\Bigg(\mathbb{E}_{\mathbb{P}_T^N}\Bigg|\lfloor\min\rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_\tau\left[\alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_1(s',a')\right)\right] \\
&\qquad\qquad - \lfloor\min\rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_\tau\left[\alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_2(s',a')\right)\right]\Bigg|\Bigg) \\
&\leq \gamma \max_{s,a}\left(\mathbb{E}_{\mathbb{P}_T^N}\left[\max_{\tau\in\mathcal{T}}\left|\mathbb{E}_\tau\left[\alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_1(s',a')\right)\right] - \mathbb{E}_\tau\left[\alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_2(s',a')\right)\right]\right|\right]\right) \\
&= \gamma \max_{s,a}\left(\mathbb{E}_{\mathbb{P}_T^N}\left[\max_{\tau\in\mathcal{T}}\left|\mathbb{E}_\tau\left[\alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_1(s',a')\right) - \alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_2(s',a')\right)\right]\right|\right]\right) \\
&\leq \gamma \max_{s,a}\left(\mathbb{E}_{\mathbb{P}_T^N}\left\|Q_1 - Q_2\right\|_\infty\right) \\
&= \gamma\left\|Q_1 - Q_2\right\|_\infty,
\end{aligned}$$

where the second inequality is obtained with Corollary 1, and the last inequality is due to $\left\|\alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_1(s,a)\right) - \alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_2(s,a)\right)\right\|_\infty \leq \left\|Q_1 - Q_2\right\|_\infty$. We present its proof by following [34]:

Suppose $\epsilon = \|Q_1 - Q_2\|_\infty$, then

$$\begin{aligned}
\alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_1(s,a)\right) &\leq \alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_2(s,a) + \frac{\epsilon}{\alpha}\right) \\
&= \alpha\log\mathbb{E}_\mu\exp\left(\frac{1}{\alpha}Q_2(s,a)\right) + \epsilon.
\end{aligned}$$

Similarly, $\alpha \log \mathbb{E}_\mu \exp\left(\frac{1}{\alpha} Q_1(s,a)\right) \geq \alpha \log \mathbb{E}_\mu \exp\left(\frac{1}{\alpha} Q_2(s,a)\right) - \epsilon$. The desired inequality is proved by putting them together.

Next, we prove $\bar{\pi}^*(a|s) \propto \mu(a|s) \exp\left(\frac{1}{\alpha}\bar{Q}^*(s,a)\right)$ is the optimal policy for $\bar{J}(\pi;\mu)$.

First, for any policy π',

$$
\begin{aligned}
&\bar{\mathcal{B}}^* \bar{Q}^{\pi'}(s,a) \\
&= r(s,a) + \gamma \mathbb{E}_{\mathbb{P}_T^N}\left[\lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_\tau \left[\alpha \log \mathbb{E}_\mu \exp\left(\frac{1}{\alpha}\bar{Q}^{\pi'}(s',a')\right)\right]\right] \\
&= r(s,a) + \gamma \mathbb{E}_{\mathbb{P}_T^N}\left[\lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_\tau \left[\alpha \log \mathbb{E}_\mu \exp\left(\frac{1}{\alpha}\bar{Q}^{\pi'}(s',a')\right)\right.\right. \\
&\qquad\qquad \left.\left. - \min_\pi \alpha D_{\mathrm{KL}}\left(\pi(\cdot|s') \,\middle|\middle|\, \frac{\mu(\cdot|s')\exp\frac{1}{\alpha}\bar{Q}^{\pi'}(s',\cdot)}{\mathbb{E}_\mu \exp\left(\frac{1}{\alpha}\bar{Q}^{\pi'}(s',a')\right)}\right)\right]\right] \\
&= r(s,a) + \gamma \mathbb{E}_{\mathbb{P}_T^N}\left[\lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_\tau \left[\max_\pi \left(\mathbb{E}_\pi\left[\bar{Q}^{\pi'}(s',a')\right] - \alpha D_{\mathrm{KL}}\left(\pi(\cdot|s') \,||\, \mu(\cdot|s')\right)\right)\right]\right] \\
&\geq r(s,a) + \gamma \mathbb{E}_{\mathbb{P}_T^N}\left[\lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_\tau \left[\mathbb{E}_{\pi'}\left[\bar{Q}^{\pi'}(s',a')\right] - \alpha D_{\mathrm{KL}}\left(\pi'(\cdot|s') \,||\, \mu(\cdot|s')\right)\right]\right] \\
&= \bar{\mathcal{B}}^{\pi'} \bar{Q}^{\pi'}(s,a) \\
&= \bar{Q}^{\pi'}(s,a), \qquad \forall s,a.
\end{aligned}
$$

By applying Lemma 3 recursively, we obtain

$$
\bar{Q}^*(s,a) = \lim_{n\to\infty} \left(\bar{\mathcal{B}}^*\right)^n \bar{Q}^{\pi'}(s,a) \geq \cdots \geq \bar{\mathcal{B}}^* \bar{Q}^{\pi'}(s,a) \geq \bar{Q}^{\pi'}(s,a), \quad \forall s,a. \tag{19}
$$

Besides,

$$
\begin{aligned}
&\bar{\mathcal{B}}^{\bar{\pi}^*} \bar{Q}^*(s,a) \\
&= r(s,a) + \gamma \mathbb{E}_{\mathbb{P}_T^N}\left[\lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_{\tau,\bar{\pi}^*}\left[\bar{Q}^*(s',a') - \alpha D_{\mathrm{KL}}\left(\bar{\pi}^*(\cdot|s) \,||\, \mu(\cdot|s)\right)\right]\right] \\
&= r(s,a) + \gamma \mathbb{E}_{\mathbb{P}_T^N}\left[\lfloor \min \rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_\tau \left[\alpha \log \mathbb{E}_\mu \exp\left(\frac{1}{\alpha}\bar{Q}^*(s',a')\right)\right]\right] \\
&= \bar{Q}^*(s,a), \qquad \forall s,a.
\end{aligned}
$$

By repeatedly applying $\bar{\mathcal{B}}^{\bar{\pi}^*}$ to the above equation, we obtain

$$
\bar{Q}^{\bar{\pi}^*}(s,a) = \lim_{n\to\infty} \left(\bar{\mathcal{B}}^{\bar{\pi}^*}\right)^n \bar{Q}^*(s,a) = \cdots = \bar{\mathcal{B}}^{\bar{\pi}^*} \bar{Q}^*(s,a) = \bar{Q}^*(s,a), \quad \forall s,a. \tag{20}
$$

By combining equations (19) and (20), we have

$$
\bar{Q}^{\bar{\pi}^*}(s,a) \geq \bar{Q}^{\pi'}(s,a), \quad \forall \pi', \forall s,a. \tag{21}
$$

Finally, by expanding $\bar{J}$ as stated in Theorem 6 and applying (21), the proof is completed

$$
\begin{aligned}
\bar{J}(\bar{\pi}^*;\mu) &= \mathbb{E}_{\rho_0,\bar{\pi}^*}\left[\bar{Q}^{\bar{\pi}^*}(s_0,a_0) - \alpha D_{\mathrm{KL}}\left(\bar{\pi}^*(\cdot|s_0) \,||\, \mu(\cdot|s_0)\right)\right] \\
&\geq \mathbb{E}_{\rho_0,\bar{\pi}^*}\left[\bar{Q}^{\pi'}(s_0,a_0) - \alpha D_{\mathrm{KL}}\left(\bar{\pi}^*(\cdot|s_0) \,||\, \mu(\cdot|s_0)\right)\right] \\
&\geq \mathbb{E}_{\rho_0,\pi'}\left[\bar{Q}^{\pi'}(s_0,a_0) - \alpha D_{\mathrm{KL}}\left(\pi'(\cdot|s_0) \,||\, \mu(\cdot|s_0)\right)\right] \\
&= \bar{J}(\pi';\mu), \quad \forall \pi'.
\end{aligned}
$$

□

Proof of Theorem 5. We first prove $J(\pi_{i+1}) > J(\pi_i)$. As the iteration requires $\bar{J}(\pi_{i+1};\pi_i) > \bar{J}(\pi_i;\pi_i) = J(\pi_i)$, it is sufficient to prove $J(\pi_{i+1}) \geq \bar{J}(\pi_{i+1};\pi_i)$. We do that by showing $Q^{\pi_{i+1}} \geq \bar{Q}^{\pi_{i+1}}$ element-wisely.

First,

$$\begin{aligned}
&\mathcal{B}^{\pi_{i+1}}\bar{Q}^{\pi_{i+1}}(s,a) - \bar{Q}^{\pi_{i+1}}(s,a) \\
&= \mathcal{B}^{\pi_{i+1}}\bar{Q}^{\pi_{i+1}}(s,a) - \bar{\mathcal{B}}^{\pi_{i+1}}\bar{Q}^{\pi_{i+1}}(s,a) \\
&= \gamma\mathbb{E}_{\mathbb{P}_T^N}\left[\lfloor\min\rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_{\tau,\pi_{i+1}}\left[\bar{Q}^{\pi_{i+1}}(s',a')\right]\right] \\
&\quad - \gamma\mathbb{E}_{\mathbb{P}_T^N}\left[\lfloor\min\rfloor_{\tau\in\mathcal{T}}^k \mathbb{E}_{\tau,\pi_{i+1}}\left[\bar{Q}^{\pi_{i+1}}(s',a') - \alpha D_{\mathrm{KL}}\big(\pi_{i+1}(\cdot|s') \;\big|\big|\; \pi_i(\cdot|s')\big)\right]\right] \\
&\geq \gamma\mathbb{E}_{\mathbb{P}_T^N}\left[\min_{\tau}\mathbb{E}_{\tau}\left[\alpha D_{\mathrm{KL}}\big(\pi_{i+1}(\cdot|s') \;\big|\big|\; \pi_i(\cdot|s')\big)\right]\right] \\
&\geq 0, \qquad \forall s,a,
\end{aligned} \tag{22}$$

where the first inequality is due to Lemma 1, the second inequality is due to the non-negativity of KL-divergence.

Then, by recursively applying Lemma 2 we obtain

$$Q^{\pi_{i+1}}(s,a) = \lim_{n\to\infty}\left(\mathcal{B}^{\pi_{i+1}}\right)^n \bar{Q}^{\pi_{i+1}}(s,a) \geq \cdots \geq \mathcal{B}^{\pi_{i+1}}\bar{Q}^{\pi_{i+1}}(s,a) \geq \bar{Q}^{\pi_{i+1}}(s,a), \quad \forall s,a. \tag{23}$$

By substituting into $J(\pi_{i+1})$ and $\bar{J}(\pi_{i+1};\pi_i)$, we have

$$\begin{aligned}
J(\pi_{i+1}) &= \mathbb{E}_{\rho_0,\pi_{i+1}} Q^{\pi_{i+1}}(s_0,a_0) \\
&\geq \mathbb{E}_{\rho_0,\pi_{i+1}} \bar{Q}^{\pi_{i+1}}(s_0,a_0) \\
&\geq \mathbb{E}_{\rho_0,\pi_{i+1}}\left[\bar{Q}^{\pi_{i+1}}(s_0,a_0) - \alpha D_{\mathrm{KL}}(\pi_{i+1}(\cdot|s_0) \;||\; \pi_i(\cdot|s_0))\right] \\
&= \bar{J}(\pi_{i+1};\pi_i).
\end{aligned} \tag{24}$$

To summarize, the proof is done via $J(\pi_{i+1}) \geq \bar{J}(\pi_{i+1};\pi_i) > \bar{J}(\pi_i;\pi_i) = J(\pi_i)$.

Next, we consider the special case where $\{\pi_i\}$ are obtained via regularized policy optimization in Theorem 4. For the $(i+1)$th step, π_{i+1} is the optimal solution for the sub-problem of maximizing $J(\pi;\pi_i)$. Thus, according to (21), $\bar{Q}^{\pi_{i+1}}(s,a) \geq \bar{Q}^{\pi'}(s,a), \forall\pi', \forall s,a$. For $\pi' = \pi_i$, the KL term in Q-value vanishes and we have $\bar{Q}^{\pi_{i+1}}(s,a) \geq Q^{\pi_i}(s,a)$. By combining it with (23), we obtain

$$Q^{\pi_{i+1}}(s,a) \geq \bar{Q}^{\pi_{i+1}}(s,a) \geq Q^{\pi_i}(s,a), \quad \forall s,a. \tag{25}$$

Then, the boundness of Q indicates the existence of $\lim_{i\to\infty} Q^{\pi_i}(s,a)$ and also $\lim_{i\to\infty} Q^{\pi_i}(s,a) = \lim_{i\to\infty} \bar{Q}^{\pi_i}(s,a), \forall s,a$.

For any s,a,a' satisfying $\lim_{i\to\infty} Q^{\pi_i}(s,a) > \lim_{i\to\infty} Q^{\pi_i}(s,a')$, it satisfies $\lim_{i\to\infty} \bar{Q}^{\pi_i}(s,a) > \lim_{i\to\infty} \bar{Q}^{\pi_i}(s,a')$. Thus,

$$\exists N, \epsilon > 0 \quad \forall j \geq N : \bar{Q}^{\pi_j}(s,a) - \bar{Q}^{\pi_j}(s,a') \geq \epsilon. \tag{26}$$

According to Theorem 4, the updated policy is with form of[4]

$$\pi_i(a|s) \propto \pi_{i-1}(a|s)\exp\left(\frac{1}{\alpha}\bar{Q}^{\pi_i}(s,a)\right).$$

Then, the policy ratio can be rewritten and bounded as

$$\frac{\pi_i(a|s)}{\pi_i(a'|s)} = \frac{\pi_N(a|s)}{\pi_N(a'|s)}\exp\left(\sum_{j=N}^{i}\frac{\bar{Q}^{\pi_j}(s,a) - \bar{Q}^{\pi_j}(s,a')}{\alpha}\right) \geq \frac{\pi_N(a|s)}{\pi_N(a'|s)}\exp\left(\frac{i-N}{\alpha}\epsilon\right), \;\; \forall i \geq N. \tag{27}$$

With the prerequisite of $\pi_0(a|s) > 0, \forall s,a$ and the form of policy update, we know $\pi_N(a|s) > 0, \forall s,a$, and further $\frac{\pi_N(a|s)}{\pi_N(a'|s)} > 0$. Then, as i approaches infinity in (27), we obtain $\frac{\pi_i(a|s)}{\pi_i(a'|s)} \to \infty$. □

[4]Strictly speaking, Theorem 4 shows $\bar{\pi}^*(a|s) \propto \mu(a|s)\exp\left(\frac{1}{\alpha}\bar{Q}^*_{N,k}(s,a)\right)$. Besides, we have shown $\bar{Q}^{\bar{\pi}^*}_{N,k}(s,a) = \bar{Q}^*_{N,k}(s,a)$ in (20). Thus, $\bar{\pi}^*(a|s) \propto \mu(a|s)\exp\left(\frac{1}{\alpha}\bar{Q}^{\bar{\pi}^*}_{N,k}(s,a)\right)$.

C Iterative Regularized Policy Optimization as Expectation–Maximization with Structured Variational Posterior

This section recasts the iterative regularized policy optimization as an Expectation-Maximization algorithm for policy optimization, where the Expectation step corresponds to a structured variational inference procedure for dynamics. To simplify the presentation, we consider the L-length horizon and let $\gamma = 1$ (thus omitted in the derivation). For infinite horizon $L \to \infty$, the discounted factor γ can be readily recovered by modifying the transition dynamics, such that any action produces a transition into an terminal state with probability $1 - \gamma$.

C.1 Review of RL as Probabilistic Inference

We first review the general framework of casting RL as probabilistic inference [60]. It starts by embedding the MDP into a probabilistic graphical model, as shown in Figure 4. Apart from the basic elements in MDP, an additional binary random variable $\mathcal{O}_t$ is introduced, where $\mathcal{O}_t = 1$ denotes that the action at time step t is optimal, and $\mathcal{O}_t = 0$ denotes the suboptimality. Its distribution is defined as[5]

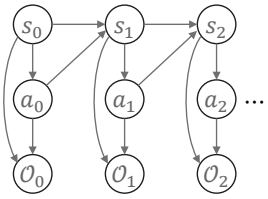

Figure 4: Probabilistic graphical model for RL as inference.

$$p(\mathcal{O}_t = 1 | s_t, a_t) = \exp\left(\frac{r(s_t, a_t)}{\alpha}\right), \tag{28}$$

where α is the hyperparameter. As we focus on the optimality, in the following we drop $= 1$ and use $\mathcal{O}_t$ to denote $\mathcal{O}_t = 1$ for conciseness. The remaining random variables in the probabilistic graphical model are s_t and a_t, whose distributions are defined by the system dynamics $\rho_0(s)$ and $T(s'|s, a)$ as well as a reference policy $\mu(a|s)$. Then, the joint distribution over all random variables for $t \in \{1, 2, \cdots, L\}$ can be written as

$$\mathbb{P}\left(s_{0:L}, a_{0:L}, \mathcal{O}_{0:L}\right) = \rho_0(s_0) \cdot \prod_{t=0}^{L-1} T(s_{t+1}|s_t, a_t)\mu(a_t|s_t) \cdot \mu(a_L|s_L) \exp\left(\sum_{t=0}^{L} \frac{r(s_t, a_t)}{\alpha}\right). \tag{29}$$

Regarding optimal control, a natural question to ask is what the trajectory should be like given the optimality over all time steps. This raises the posterior inference of $\mathbb{P}\left(s_{0:L}, a_{0:L}|\mathcal{O}_{0:L}\right)$. According to d-separation, the exact posterior follows the form of

$$\mathbb{P}\left(s_{0:L}, a_{0:L}|\mathcal{O}_{0:L}\right) = \mathbb{P}(s_0|\mathcal{O}_{0:L}) \cdot \prod_{t=0}^{L-1} \mathbb{P}(s_{t+1}|s_t, a_t, \mathcal{O}_{0:L})\mathbb{P}(a_t|s_t, \mathcal{O}_{0:L}) \cdot \mathbb{P}(a_L|s_L, \mathcal{O}_{0:L}). \tag{30}$$

Notice that the dynamics posterior $\mathbb{P}(s_0|\mathcal{O}_{0:L})$ and $\mathbb{P}(s_{t+1}|s_t, a_t, \mathcal{O}_{0:L})$ depends on $\mathcal{O}_{0:L}$, and in fact their concrete mathematical expressions are inconsistent with those of the system dynamics $\rho_0(s_0)$ and $T(s_{t+1}|s_t, a_t)$ [60]. This essentially poses the assumption that the dynamics itself can be controlled when referring to the optimality, unpractical in general.

Variational inference can be applied to correct this issue. Concretely, define the variational approximation to the exact posterior by

$$\widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}\right) = \rho_0(s_0) \cdot \prod_{t=0}^{L-1} T(s_{t+1}|s_t, a_t)\pi(a_t|s_t) \cdot \pi(a_L|s_L). \tag{31}$$

Its difference to (30) is enforcing the dynamics posterior to match the practical one. Under this structure, the variational posterior can be adjusted by optimizing π to best approximate the exact

[5] Assume the reward function is non-positive such that the probability is not larger than one. If the assumption is unsatisfied, we can subtract the reward function by its maximum, without changing the optimal policy.

posterior. The optimization is executed under measure of KL divergence, i.e.,

$$
\begin{aligned}
&D_{\mathrm{KL}}\Big(\widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}\right) \Big|\Big| \mathbb{P}\left(s_{0:L}, a_{0:L} | \mathcal{O}_{0:L}\right)\Big) = \int \widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}\right) \log \frac{\widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}\right)}{\mathbb{P}\left(s_{0:L}, a_{0:L} | \mathcal{O}_{0:L}\right)} ds_{0:L} da_{0:L} \\
&= \int \widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}\right) \log \frac{\widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}\right)}{\mathbb{P}\left(s_{0:L}, a_{0:L}, \mathcal{O}_{0:L}\right)} ds_{0:L} da_{0:L} + \log \mathbb{P}(\mathcal{O}_{0:L}) \\
&= \mathbb{E}_{\rho_0, T, \pi}\left[\sum_{t=0}^{L}\left(-\frac{r(s_t, a_t)}{\alpha} + \log \frac{\pi(a_t|s_t)}{\mu(a_t|s_t)}\right)\right] + \log \mathbb{P}(\mathcal{O}_{0:L}) \\
&= \frac{1}{\alpha}\mathbb{E}_{\rho_0, T, \pi}\left[\sum_{t=0}^{L}\Big(- r(s_t, a_t) + \alpha D_{\mathrm{KL}}\Big(\pi(\cdot|s_t) \Big|\Big| \mu(\cdot|s_t)\Big)\Big)\right] + \log \mathbb{P}(\mathcal{O}_{0:L}),
\end{aligned} \tag{32}
$$

where the third equation is obtained by substituting (29) and (31). As the second term in (32) is constant, minimizing the above KL divergence is equivalent to maximize the cumulative reward with policy regularizer. Several fascinating online RL methods can be treated as algorithmic instances based on this framework [34, 35].

To summarize, the structured variational posterior with form (31) is vital to ensure the inferred policy meaningful in the actual environment.

C.2 Pessimism-Modulated Dynamics Belief as Structured Variational Posterior

The probabilistic graphical model is previously devised for online RL. In offline setting, the environment can not be interacted to minimize (32). A straightforward modification to reflect this is to add the transition dynamics as a random variable in the graph, as shown in Figure 5. We assume the transition follows a predefined belief distribution, i.e., $\mathbb{P}_T^{sa}(\tau^{sa})$ introduced in Subsection 3.1. To make its dependence on (s, a) explicit, let $\mathbb{P}_T(\tau^{sa}|s, a)$ redenote $\mathbb{P}_T^{sa}(\tau^{sa})$. For conciseness, we drop the superscript sa in τ^{sa} in the remainder.

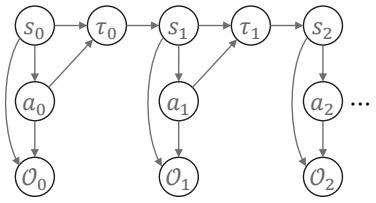

Figure 5: Probabilistic graphical model for offline RL as inference.

The joint distribution over all random variables in Figure 5 for $t \in \{1, 2, \cdots, L\}$ can be written as

$$
\begin{aligned}
\mathbb{P}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1}, \mathcal{O}_{0:L}\right) = \rho_0(s_0) \cdot \prod_{t=0}^{L-1} \mathbb{P}_T(\tau_t|s_t, a_t)\tau_t(s_{t+1})\mu(a_t|s_t) \\
\cdot \mu(a_L|s_L) \exp\left(\sum_{t=0}^{L} \frac{r(s_t, a_t)}{\alpha}\right).
\end{aligned} \tag{33}
$$

Similar to online setting, we wonder what the trajectory should be like given the optimality over all time steps. By examining the conditional independence in the probabilistic graphical model, the exact posterior follows the form of

$$
\begin{aligned}
\mathbb{P}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1} | \mathcal{O}_{0:L}\right) = \mathbb{P}(s_0|\mathcal{O}_{0:L}) \cdot \prod_{t=0}^{L-1} \mathbb{P}(\tau_t|s_t, a_t, \mathcal{O}_{0:L})\mathbb{P}(s_{t+1}|\tau_t, \mathcal{O}_{0:L})\mathbb{P}(a_t|s_t, \mathcal{O}_{0:L}) \\
\cdot \mathbb{P}(a_L|s_L, \mathcal{O}_{0:L}).
\end{aligned} \tag{34}
$$

Unsurprisingly, $s_{0:T}$ and $\tau_{0:T}$ again depend on $\mathcal{O}_{0:L}$, indicating that the system transition and its belief can be controlled when referring to optimality. In other words, it leads to over-optimistic inference.

To emphasize pessimism, we define a novel structured variational posterior:

$$
\widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1}\right) = \rho_0(s_0) \cdot \prod_{t=0}^{L-1} \widetilde{\mathbb{P}}_T(\tau_t|s_t, a_t)\tau_t(s_{t+1})\pi(a_t|s_t) \cdot \pi(a_L|s_L), \tag{35}
$$

with $\widetilde{\mathbb{P}}_T$ being the *Pessimism-Modulated Dynamics Belief (PMDB)* constructed via the KL-regularized AMG (see Theorem 7):

$$\widetilde{\mathbb{P}}_T(\tau|s,a) \propto w\Big(\mathbb{E}_{\tau,\pi}\big[\bar{Q}^\pi_{N,k}(s',a')\big]; k, N\Big)\mathbb{P}_T(\tau|s,a), \tag{36}$$

$$w(x;k,N) = \big[F(x)\big]^{k-1}\big[1-F(x)\big]^{N-k}, \tag{37}$$

$F(\cdot)$ is cumulative density function and $\bar{Q}^\pi_{N,k}$ is the Q-value for the KL-regularized AMG. As discussed, w reshapes the initial belief distribution towards being pessimistic in terms of $\mathbb{E}_{\tau,\pi}\big[\bar{Q}^\pi_{N,k}(s',a')\big]$.

It seems that we need to solve the AMG to obtain $\bar{Q}^\pi_{N,k}$ and further define $\widetilde{\mathbb{P}}_T$. In fact, $\bar{Q}^\pi_{N,k}$ is also the Q-value for the MDP considered in (35). This can be verified by checking Theorem 7: the KL-regularized AMG is equivalent to the MDP with transition $\widetilde{T}(s'|s,a) = \mathbb{E}_{\widetilde{\mathbb{P}}_T}[\tau(s')]$, which can be implemented by sampling first $\tau \sim \widetilde{\mathbb{P}}_T$ and then $s' \sim \tau$, right the procedure in (35).

To best approximate the exact posterior, we optimize the variational posterior by minimizing

$$\begin{aligned}
&D_{\mathrm{KL}}\Big(\widehat{\mathbb{P}}\left(s_{0:L},a_{0:L},\tau_{0:L-1}\right)\Big|\Big|\,\mathbb{P}\left(s_{0:L},a_{0:L},\tau_{0:L-1}|\mathcal{O}_{0:L}\right)\Big)\\
&=\int \widehat{\mathbb{P}}\left(s_{0:L},a_{0:L},\tau_{0:L-1}\right)\log\frac{\widehat{\mathbb{P}}\left(s_{0:L},a_{0:L},\tau_{0:L-1}\right)}{\mathbb{P}\left(s_{0:L},a_{0:L},\tau_{0:L-1}|\mathcal{O}_{0:L}\right)}ds_{0:L}da_{0:L}\\
&=\int \widehat{\mathbb{P}}\left(s_{0:L},a_{0:L},\tau_{0:L-1}\right)\log\frac{\widehat{\mathbb{P}}\left(s_{0:L},a_{0:L},\tau_{0:L-1}\right)}{\mathbb{P}\left(s_{0:L},a_{0:L},\tau_{0:L-1},\mathcal{O}_{0:L}\right)}ds_{0:L}da_{0:L}+\log\mathbb{P}(\mathcal{O}_{0:L})\\
&=\underbrace{\mathbb{E}_{\rho_0,\widetilde{\mathbb{P}}_T,\tau_{0:L-1},\pi}\left[\sum_{t=0}^{L}\left(-\frac{r(s_t,a_t)}{\alpha}+D_{\mathrm{KL}}\Big(\pi(\cdot|s_t)\Big|\Big|\,\mu(\cdot|s_t)\Big)\right)\right]}_{M(\pi;\mu)}\\
&\quad+\mathbb{E}_{\rho_0,\widetilde{\mathbb{P}}_T,\tau_{0:L-1},\pi}\left[\sum_{t=0}^{L-1}\log w\Big(\mathbb{E}_{\tau_t,\pi}\big[\bar{Q}^\pi_{N,k}(s_{t+1},a_{t+1})\big];k,N\Big)\right]+(L-1)\cdot\log C\\
&\overset{(*)}{=} M(\pi;\mu)+(L-1)\cdot\log C\\
&\quad+\sum_{t=0}^{L-1}\mathbb{E}_{\rho_0,\pi,\widehat{\mathbb{P}}_T}\left[\mathbb{E}_{\tau_0,\pi,\widehat{\mathbb{P}}_T}\cdots\left[\mathbb{E}_{\tau_{t-1},\pi,\widetilde{\mathbb{P}}_T}\left[\log w\Big(\mathbb{E}_{\tau_t,\pi}\big[\bar{Q}^\pi_{N,k}(s_{t+1},a_{t+1})\big];k,N\Big)\right]\right]\right]\\
&\approx M(\pi;\mu)+(L-1)\cdot(\log C'+\log C),
\end{aligned}\tag{38}$$

where the equation $(*)$ is by unfolding the expectation sequentially over each step, $C = \frac{N!}{(k-1)!(N-k)!}$ is the normalization constant in (36), and $C' = \frac{(k-1)^{k-1}(N-k)^{N-k}}{(N-1)^{N-1}}$ is used to approximate w. To clarify the approximation, recall Theorem 7 stating that a sample $\tau_t \sim \widetilde{\mathbb{P}}_T$ can be equivalently drawn by finding $\tau_t = \arg\lfloor\min\rfloor^k_{\tau\in\mathcal{T}_t}\mathbb{E}_{\tau,\pi}\left[\bar{Q}^\pi_{N,k}(s_{t+1},a_{t+1})\right]$ based on another sampling procedure $\mathcal{T}_t = \{\tau\}^N \sim \mathbb{P}_T^N$. Then, given $\mathcal{T}_t$, we observe that $\mathbb{E}_{\tau_t,\pi}\big[\bar{Q}^\pi_{N,k}(s_{t+1},a_{t+1})\big]$ is the empirical $\frac{k-1}{N-1}$ quantile of the random variable $\mathbb{E}_{\tau,\pi}\big[\bar{Q}^\pi_{N,k}(s_{t+1},a_{t+1})\big]$, i.e., $F\left(\mathbb{E}_{\tau_t,\pi}\big[\bar{Q}^\pi_{N,k}(s_{t+1},a_{t+1})\big]\right) \approx \frac{k-1}{N-1}$. By substituting into w, we obtain $w \approx C'$.

Note that $-\alpha M(\pi;\mu)$ is exactly the return of KL-regularized MDP in Theorem 7. By the equivalence of this KL-regularized MDP and the KL-regularized AMG in (9), we have $M(\pi;\mu) = -\frac{\bar{J}(\pi;\mu)}{\alpha}$. Thus, minimization of (38) is equivalent to maximization of $\bar{J}(\pi;\mu)$.

C.3 Full Expectation-Maximization Algorithm

In previous subsection, the reference policy μ is assumed as a prior, and the optimized policy would be constrained close to it through KL divergence. In practice, the prior of optimal policy can not

easily obtained, and a popular methodology to handle this is to learn the prior itself in the data-driven way, i.e., the principle of empirical Bayes.

The prior learning is done by maximizing the log-marginal likelihood:

$$L(\mu) = \log \mathbb{P}\left(\mathcal{O}_{0:L}\right) = \log \int \mathbb{P}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1}, \mathcal{O}_{0:L}\right) ds_{0:L} da_{0:L} d\tau_{0:L-1}, \tag{39}$$

where $\mathbb{P}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1}, \mathcal{O}_{0:L}\right)$ is given in (33). As the log function includes a high-dimensional integration, evaluating $L(\mu)$ incurs intensive computation. Expectation-Maximization algorithm instead considers a lower bound of $L(\mu)$ to make the evaluation/optimization tractable:

$$\begin{aligned} L(\mu) \geq & \log \mathbb{P}\left(\mathcal{O}_{0:L}\right) - D_{\mathrm{KL}}\Big(\widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1}\right) \Big|\Big| \mathbb{P}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1} | \mathcal{O}_{0:L}\right)\Big) \\ = & \int \widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1}\right) \log \mathbb{P}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1}, \mathcal{O}_{0:L}\right) ds_{0:L} da_{0:L} d\tau_{0:L-1} \\ & - \mathcal{H}\left[\widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1}\right)\right], \end{aligned} \tag{40}$$

where the inequality is due to the non-negativity of KL divergence, and $\widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1}\right)$ is an approximation to the exact posterior $\mathbb{P}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1} | \mathcal{O}_{0:L}\right)$. The lower bound is tighter with the more exact approximation for the posterior. In previous subsection, we introduce the structured variational approximation with form of (35) to emphasize pessimism on the transition dynamics. Although this variational posterior would lead to non-zero KL term, it promotes learning robust policy as we discussed in previous subsection. Since that the variational posterior is with an adjustable policy π, we denote the lower bound by $\bar{L}(\mu; \pi)$.

By substituting (35) into (40), it follows

$$\begin{aligned} \bar{L}(\mu; \pi) = & \mathbb{E}_{\rho_0, \widetilde{\mathbb{P}}_T, \tau_{0:L-1}, \pi}\left[\sum_{t=0}^{L} \log \mu(a_t|s_t)\right] + C'' \\ = & \mathbb{E}_{\rho_0, \widetilde{\mathbb{P}}_T, \tau_{0:L-1}, \pi}\left[\sum_{t=0}^{L} \log \mu(a_t|s_t) - \log \pi(a_t|s_t) + \log \pi(a_t|s_t)\right] + C'' \\ = & \mathbb{E}_{\rho_0, \widetilde{\mathbb{P}}_T, \tau_{0:L-1}, \pi}\left[\sum_{t=0}^{L} -D_{\mathrm{KL}}(\pi(\cdot|s_t) \,||\, \mu(\cdot|s_t))\right] + C''', \end{aligned} \tag{41}$$

where C'' and C''' includes the constant terms irrelevant to μ. According to the form of (41), given fixed π, the optimal prior policy to maximize $\bar{L}(\mu; \pi)$ is obtained as $\mu = \pi$. Maximizing the lower bound is known as Maximization step.

Recall π in the variational posterior is adjustable, we can optimize it by minimizing $D_{\mathrm{KL}}\Big(\widehat{\mathbb{P}}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1}\right) \Big|\Big| \mathbb{P}\left(s_{0:L}, a_{0:L}, \tau_{0:L-1} | \mathcal{O}_{0:L}\right)\Big)$ to tighten the bound. The minimization procedure is known as Expectation step. In our case, the minimization problem is exactly the one discussed in previous subsection.

When repeatedly and alternately applying the Expectation and Maximization steps, the iterative regularized policy optimization algorithm is recovered. According to Theorem 5, both π and μ continuously improve regarding the objective function.

D Algorithm and Implementation Details for Model-Based Offline RL with PMDB

The pseudocode for model-based offline RL with PMDB is presented in Algorithm 1. As ρ_0 is unknown in practice, we uniformly sample states from $\mathcal{D}$ as the initial $\{s_0\}$. In Step 4, the primary players act according to the non-parametric policy π, rather than its approximated policy π_ϕ. This is because during learning process π_ϕ is not always trained adequately to approximate π, then following π_ϕ will visit unexpected states. In Step 11, the reference policy $\pi_{\phi'}$ is returned as the final policy, considering that it is more stable than π_ϕ.

Algorithm 1 Model-Based Offline RL with PMDB

Require: $\mathcal{D}, \mathbb{P}_T, N, k, M$.

1: **Approximator initialization:** Randomly initialize Q-function $Q_\theta(s,a)$ and policy $\pi_\phi(a|s)$; Initialize target Q-function $Q_{\theta'}(s,a)$ and reference policy $\pi_{\phi'}(a|s)$ with $\theta' \leftarrow \theta, \phi' \leftarrow \phi$.

2: **Game initialization:** Randomly sample C states from $\mathcal{D}$, as the initial states for C paralleled games $\{s\}$.

3: **for** step $t = 1, 2, \cdots, M$ **do**

4: **Primary players:** Sample actions according to

$$\pi(a|s) \propto \pi_{\phi'}(a|s) \exp\left(\frac{1}{\alpha} Q_\theta(s,a)\right).$$

5: **Game transitions:** Sample candidate sets $\{\mathcal{T}\}$ according to (3).

6: **Update:** Sample a batch of transitions from $\mathcal{D}$, together with the C-size game transitions $\{(s,a,\mathcal{T})\}$, to update θ and ϕ via one-step gradient descent regarding (11) and (14).

7: **Secondary players:** Determine whether to exploit or explore: with probability of $(1-\epsilon)$,

$$\bar{\tau} = \arg\lfloor\min\rfloor_{\tau\in\mathcal{T}}^{k} \mathbb{E}_\tau \left[\alpha \log \mathbb{E}_{\pi_{\phi'}} \exp\left(\frac{1}{\alpha} Q_\theta(s',a')\right)\right],$$

otherwise randomly choose $\bar{\tau}$ from $\mathcal{T}$.

8: **Game transitions:** Sample states following $\{\bar{\tau}\}$ to update $\{s\}$. For terminal states in $\{s\}$, use random samples from $\mathcal{D}$ to replace them.

9: **Moving-average update:** Update reference policy and target Q-function with

$$\phi' \leftarrow \omega_1 \phi + (1-\omega_1)\phi',$$
$$\theta' \leftarrow \omega_2 \theta + (1-\omega_2)\theta'.$$

10: **end for**

11: **return** $\pi_{\phi'}$.

Computing expectation Algorithm 1 involves the computation of expectation. In discrete domains, the expectation can be computed exactly. In continuous domains, we use Monte Carlo methods to approximate it. Concretely, for the expectation over states we apply vanilla Monte Carlo sampling, while for the expectation over actions we apply importance sampling. To elaborate, the expectation over actions can be written as

$$\begin{aligned}\mathbb{E}_\mu \exp\left(\frac{1}{\alpha} Q(s,a)\right) &= \frac{1}{2}\left[\mathbb{E}_\mu \exp\left(\frac{1}{\alpha} Q(s,a)\right) + \mathbb{E}_q \frac{\mu(a|s)\exp\left(\frac{1}{\alpha}Q(s,a)\right)}{q(a|s)}\right] \\ &\approx \frac{1}{2n}\left[\sum_{a_i\sim\mu(\cdot|s)}^{n} \exp\left(\frac{1}{\alpha} Q(s,a_i)\right) + \sum_{a_i\sim q(\cdot|s)}^{n} \frac{\mu(a_i|s)\exp\left(\frac{1}{\alpha}Q(s,a_i)\right)}{q(a_i|s)}\right],\end{aligned}$$

where q is the proposal distribution.

In Algorithm 1, the above expectation is computed for both $s \in \mathcal{D}$ and $s \in \mathcal{D}'$. For $s \in \mathcal{D}$, we choose

$$q(\cdot|s) = \mathcal{N}(\,\cdot\,; a, \sigma^2 I), \quad \text{where } a|s \sim \mathcal{D},$$

i.e., the samples are drawn close to the data points, and σ^2 determines how much they keep close. For example, in Step 6 a batch of $\{(s,a,s')\}$ are sampled from $\mathcal{D}$ to calculate (14), then we construct the proposal distribution as above for each (s,a) in the batch. The motivation of drawing actions near the data samples is to enhance learning in the multi-modal scenario, where the offline dataset $\mathcal{D}$ is collected by mixture of multiple policies. If μ is single-modal (say the widely adopted Gaussian policy) and we solely draw samples from it to approximate the expectation, these samples will be locally clustered. Then, applying them to update π_θ in (14) can be easily get stuck at local optimum.

For $s \in \mathcal{D}'$, we choose

$$q(\cdot|s) = \pi_\theta(\cdot|s).$$

The reason is that π_θ is an approximator to the improved policy with higher Q-value, and sampling from it hopefully reduces variance of the Monte Carlo estimator.

Although applying Monte Carlo methods to approximate the expectation incurs extra computation, all the operators can be executed in parallel. In the experiments, we use 10 and 20 samples respectively for the expectations over state and action, and the algorithm is run on a single machine with one Quadro RTX 6000 GPU. The results show that in average it takes 73.4 s to finish 1k training steps, and the GPU memory cost is 2.5 GB.

Several future directions regarding the Monte Carlo method are worthy to explore. For example, by reducing the sample size for the expectation over state, the optimized policy additionally tends to avoid the risk due to aleatoric uncertainty (while in this work we focus on epistemic uncertainty). Besides, the computational cost can be reduced by more aggressive Monte Carlo approximation, for example only using mean action to compute the expectation in terms of policy. We leave these as future work.

E Choice of Initial Dynamics Belief

In offline setting, extra knowledge is strongly desired to aggressively optimize policy. The initial dynamics belief provides an interface to absorb the aforehand knowledge of system transition. In what follows, we illustrate several potential usecases:

- Consider the physical system where the dynamics can be described as mathematical expression but with uncertain parameter. If we have a narrow distribution over the parameter (according to expert knowledge or inferred from data), the system is almost known for certain. Here, both the mathematical expression and narrow distribution provide more information.
- Consider the case where we know the dynamics is smooth with probability of 0.7 and periodic with probability of 0.3. Gaussian processes (GPs) with RBF kernel and periodic kernel can well encode these prior knowledge. Then, the 0.7-0.3 mixture of the two GPs trained with offline data can act as the dynamics belief to provide more information.
- In the case where multi-task datasets are available, we can train dynamics models using each of the datasets and assign likelihood ratios to these models. If the likelihood ratio well reflects the similarity between the concerned task and the offline tasks, the multi-task datasets promote knowledge.

The performance gain is expected to monotonously increase with the amount of correct knowledge. As an impractical but intuitive example, with the exact knowledge of system transition (the initial belief is a delta function), the proposed approach is actually optimizing policy as in real system.

In practice, the expert knowledge is not available everywhere. When unavailable, the best we can hope for is that the final policy stays close to the dataset, but unnecessary to be fully covered (as we want to utilize the generalization ability of dynamics model at least around the data). To that end, the dynamics belief is desired to be certain at the region in distribution of dataset, and turns more and more uncertain when departing. It has been reported that the simple model ensemble leads to such a behavior [12]. In this sense, the uniform distribution over learned dynamics ensemble can act as a quite common belief. In the experiments, we apply it for fair comparison with baseline methods.

F Automatically Adjusting KL Coefficient

In Section 4, the KL regularizer is introduced to restrict π_ϕ in a small region near $\pi_{\phi'}$, such that the Q-value can be evaluated sufficiently before policy improvement. Apart from fixing the KL coefficient α throughout, we provide a strategy to automatically adjust it.

Note that the optimal policy to minimize L_P in (14) is $\frac{\pi_{\phi'}(\cdot|s)\exp\left(\frac{1}{\alpha}Q_\theta(s,\cdot)\right)}{\mathbb{E}_{\pi_{\phi'}}\left[\exp\left(\frac{1}{\alpha}Q_\theta(s,a)\right)\right]}$. The criterion of choosing α is to constrain the KL divergence between this policy and $\pi_{\phi'}$ smaller than a specified constant, i.e.,

$$D_{\mathrm{KL}}\left(\frac{\pi_{\phi'}(\cdot|s)\exp\left(\frac{1}{\alpha}Q_\theta(s,\cdot)\right)}{\mathbb{E}_{\pi_{\phi'}}\left[\exp\left(\frac{1}{\alpha}Q_\theta(s,a)\right)\right]}\,\middle\|\,\pi_{\phi'}(\cdot|s)\right)\leq d. \tag{42}$$

Finding α to satisfy the above inequation is intractable, instead we consider a surrogate of the KL divergence:

$$
\begin{aligned}
&D_{\mathrm{KL}}\left(\frac{\pi_{\phi'}(\cdot|s)\exp\left(\frac{1}{\alpha}Q_\theta(s,\cdot)\right)}{\mathbb{E}_{\pi_{\phi'}}\left[\exp\left(\frac{1}{\alpha}Q_\theta(s,a)\right)\right]}\;\middle\|\;\pi_{\phi'}(\cdot|s)\right)\\
&=\mathbb{E}_{\pi_{\phi'}}\left[\frac{\exp\left(\frac{1}{\alpha}Q_\theta(s,a)\right)}{\mathbb{E}_{\pi_{\phi'}}\left[\exp\left(\frac{1}{\alpha}Q_\theta(s,a)\right)\right]}\cdot\frac{1}{\alpha}Q_\theta(s,a)\right]-\log\mathbb{E}_{\pi_{\phi'}}\left[\exp\left(\frac{1}{\alpha}Q_\theta(s,a)\right)\right]\\
&\leq\frac{1}{\alpha}\left(\mathbb{E}_{\pi_{\phi'}}\left[\frac{\exp\left(\frac{1}{\alpha_0}Q_\theta(s,a)\right)}{\mathbb{E}_{\pi_{\phi'}}\left[\exp\left(\frac{1}{\alpha_0}Q_\theta(s,a)\right)\right]}\cdot Q_\theta(s,a)\right]-\mathbb{E}_{\pi_{\phi'}}\left[Q_\theta(s,a)\right]\right),
\end{aligned}
$$

where α_0 is a predefined lower bound of α.

Then, (42) can be satisfied by setting

$$
\alpha\geq\frac{1}{d}\left(\mathbb{E}_{\pi_{\phi'}}\left[\frac{\exp\left(\frac{1}{\alpha_0}Q_\theta(s,a)\right)}{\mathbb{E}_{\pi_{\phi'}}\left[\exp\left(\frac{1}{\alpha_0}Q_\theta(s,a)\right)\right]}\cdot Q_\theta(s,a)\right]-\mathbb{E}_{\pi_{\phi'}}\left[Q_\theta(s,a)\right]\right).
$$

Combining with the predefined lower bound, we choose α as

$$
\alpha=\max\left(\frac{1}{d}\left(\mathbb{E}_{\pi_{\phi'}}\left[\frac{\exp\left(\frac{1}{\alpha_0}Q_\theta(s,a)\right)}{\mathbb{E}_{\pi_{\phi'}}\left[\exp\left(\frac{1}{\alpha_0}Q_\theta(s,a)\right)\right]}\cdot Q_\theta(s,a)\right]-\mathbb{E}_{\pi_{\phi'}}\left[Q_\theta(s,a)\right]\right),\alpha_0\right).
$$

In practice, the expectation can be estimated over Monte Carlo samples. Note that the coefficient can be computed individually for each state, picking d is hopefully easier than picking α suitable for all states.

G Additional Experimental Setup

Task Domains We evaluate the proposed methods and the baselines on eighteen domains involving three environments (hopper, walker2d, halfcheetah), each with six dataset types. The dataset types are collected by different policies, denoted by *random*: a randomly initialized policy, *expert*: a policy trained to completion with SAC, *medium*: a policy trained to approximately 1/3 the performance of the expert, *medium-expert*: 50-50 mixture of medium and expert data, *medium-replay*: the replay buffer of a policy trained up to the performance of the medium agent, *full-replay*: the replay buffer of a policy trained up to the performance of the expert agent.

Dynamics Belief We adopt an uniform distribution over dynamics model ensemble as the initial belief. The ensemble contains 100 neural networks, each is with 4 hidden layers and 256 hidden units per layer. All the neural networks are trained independently with the sample dataset $\mathcal{D}$ and in parallel. The training process stops after the average training loss does not change obviously. Specifically, the number of epochs for hopper-random and walker2d-medium are 2000, and those for other tasks are 1000. Note that the level of pessimism depends on the candidate size N ($= 10$ by default), rather than the ensemble size.

Policy Network and Q Network The policy network is with 3 hidden layers and 256 hidden units per layer. It outputs the mean and the diagonal variance for a Gaussian distribution, which is then transformed via tanh function to generate the policy. When evaluating our approach, we apply the deterministic policy, where the action is the tanh transformation of the Gaussian mean. The Q network is with the same architecture as the policy network except the output layer. Similar to existing RL approaches [35], we make use of two Q networks and apply the minimum of them for calculation in Algorithm 1, in order to mitigate over-estimation when learning in the AMG. The policy learning stops after the performance in AMG does not change obviously. Specifically, the gradient steps for walker2d-random, halfcheetah-random and hopper with all dataset types are 1 million, and those for other tasks are 2 millon.

Hyperparameters We list the detailed hyperparameters in Table 3.

Parameter	Value
dynamics learning rate	10^{-4}
policy learning rate	$3 \cdot 10^{-5}$
Q-value learning rate	$3 \cdot 10^{-4}$
discounted factor (γ)	0.99
smoothing coefficient for policy (ω_1)	10^{-5}
smoothing coefficient for Q-value (ω_2)	$5 \cdot 10^{-3}$
Exploration ratio for secondary player (ϵ)	0.1
KL coefficient (α)	0.1
variance for important sampling (σ^2)	0.01
Batch size for dynamics learning	256
Batch size for AMG and MDP	128
Maximal horizon of AMG	1000

Table 3: Hyperparameters

H Practical Impact of N

Table 4 lists the impact of N. The performance in the AMGs improve when decreasing k. Regarding the performance in true MDPs, we notice that $N = 15$ corresponds to the best performance for hopper, but for the others $N = 5$ is better.

	hopper-medium		walker2d-medium		halfcheetah-medium	
N	MDP	AMG	MDP	AMG	MDP	AMG
5	90.2±25.4	108.6±2.2	112.7±0.9	101.7±5.7	79.8±0.4	69.5±1.6
10	106.8±0.2	105.2±1.6	94.2±1.1	77.2±3.7	75.6±1.3	67.3±1.1
15	107.3±0.2	103.1±1.8	92.1±0.3	68.3±6.7	75.4±0.4	63.2±2.3

Table 4: Impact of N, with $k = 2$.

I Ablation of Randomness of $\mathcal{T}$

Compared to the standard Bellman backup operator in Q-learning, the proposed one additionally includes the expectation over $\mathcal{T} \sim \mathcal{P}_T^N$ and the k-minimum operator over $\tau \in \mathcal{T}$. We report the impact of choosing different k in Table 2, and present the impact of the randomness of $\mathcal{T}$ as below. Fixed $\mathcal{T}$ denotes that after sampling once $\mathcal{T}$ from the belief distribution we keep it fixed during policy optimization.

Task Name	Stochastic $\mathcal{T}$	Fixed $\mathcal{T}$
hopper-medium	106.8 ± 0.2	106.2 ± 0.3
walker2d-medium	94.2 ± 1.1	90.1 ± 4.3
halfcheetah-medium	75.6 ± 1.3	73.1 ± 2.8

Table 5: Impact of randomness of $\mathcal{T}$

We observe that the randomness of $\mathcal{T}$ has a mild effect on the performance in average. The reason can be that we apply the uniform distribution over dynamics ensemble as initial belief (without additional knowledge to insert). The model ensemble is reported to produce low uncertainty estimation in distribution of data coverage and high estimation when departing the dataset [12]. This property makes the optimized policy keep close to the dataset, and it does not rely on the randomness of ensemble elements. However, involving the randomness can lead to more smooth variation of the estimated uncertainty, which benefits the training process and results in better performance. Apart from these empirical results, we highlight that in cases with more informative dynamics belief, only picking several fixed samples from the belief distribution as $\mathcal{T}$ will result in the loss of knowledge.

J Weighting AMG Loss and MDP Loss in (11)

In (11), the Q-function is trained to minimize the Bellman residuals of both the AMG and the empirical MDP, equipped with the same weight (both are 1). In the following table, we show experiment results to check the impact of different weights.

Task Name	**0.5:1.5**	**1.0:1.0**	**1.5:0.5**
hopper-medium	106.6 $\pm$ 0.3	106.8 $\pm$ 0.2	106.5 $\pm$ 0.3
walker2d-medium	93.8 $\pm$ 1.5	94.2 $\pm$ 1.1	93.1 $\pm$ 1.3
halfcheetah-medium	75.2 $\pm$ 0.8	75.6 $\pm$ 1.3	76.1 $\pm$ 1.0

Table 6: Impact of weights in (11)

The results suggests that the performance does not obviously depend on the weights. But in cases with available expert knowledge about dynamics, the weights can be adjusted to match our confidence on the knowledge, i.e., the less confidence, the smaller weight for AMG.

K Comparison with RAMBO

We additionally compared the proposed approach with RAMBO [61], a concurrent work that also formulates offline RL as a two-player zero-sum game. The results of RAMBO for random, medium, medium-expert and medium-replay are taken from [61]. For the other two dataset types, we run the official code and follow the hyperparameter search procedure reported in its paper.

Task Name	**BC**	**BEAR**	**BRAC**	**CQL**	**MOReL**	**EDAC**	**RAMBO**	**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	25.4±7.5	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	87.0±15.4	**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**	50.0±8.1	**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**	88.2±20.5	**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	99.5±4.8	**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	105.2 ±2.1	**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	0.0±0.3	**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	84.9 ±2.6	**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**	1.6±2.3	**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**	56.7±39.0	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	**89.2±6.7**	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**	88.3±4.9	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	**39.5±3.5**	**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	**77.9 ±4.0**	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**	79.3±15.1	**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	95.4 ±5.4	**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	68.7± 5.3	**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	87.0±3.2	**90.0±0.8**
Average	49.9	52.4	54.0	73.7	65.1	85.2	68.0	**88.2**

Table 7: Extended Results for D4RL datasets.

The results show our approach outperforms RAMBO on most of considered tasks. One reason can be that the problem formulation of RAMBO is based on robust MDP, whose defects are discussed in Section 2 and Appendix A.