Towards Reliable Offline Reinforcement Learning via Lyapunov Uncertainty Control

Ke Jiang[a], Yao Li[b], Xiaoyang Tan[a,*]
[a] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, China
[b] School of Computer and Information Technology, Shanxi University, China
{ke_jiang, x.tan}@nuaa.edu.cn, yao.l@sxu.edu.cn, * Corresponding Author

***Abstract*—Learning trustworthy and reliable offline policies presents significant challenges due to the inherent uncertainty in pre-collected datasets. In this paper, we propose a novel offline reinforcement learning method to tackle this issue. Inspired by the concepts of Lyapunov stability and control-invariant sets from control theory, the central idea is to introduce a restricted state space for the agent to operate within, which allows the learned models to exhibit reduced Bellman uncertainty and make reliable decisions. To achieve this, we regulate the expected Bellman uncertainty associated with the new policy, ensuring that its growth trend in subsequent states remains within acceptable limits. The resulting method, termed Lyapunov Uncertainty Control (LUC), is shown to guarantee that the agent remains within a low-uncertainty state enclosure throughout its entire trajectory. Furthermore, we perform extensive theoretical and experimental analysis to showcase the effectiveness and feasibility of the proposed LUC.**

***Index Terms*—Offline reinforcement learning, Lyapunov uncertainty control, Reliable control, Uncertainty**

I. Introduction

OFFLINE reinforcement learning (RL) allows policy learning from historical data without real-world interaction. However, ensuring reliable sequential decision-making from offline data poses a significant challenge in practical applications. For example, in healthcare [1], a reliable diagnostic agent requires avoiding unfamiliar approaches that may introduce errors in subsequent procedures. Similar requirements for reliability exist in fields such as autonomous driving [2], robotics control [3], and others, where all these fields are sensitive to those high-risk control sequences.

The reliability of offline RL is often undermined by the risk of deviating from its trustworthy regions, i.e., failing to perform stably reliable control [4] based on offline data. More precisely, a trustworthy and reliable offline-learned agent is suppose to avoid visiting states at which the learned models (e.g., Q-networks) fail to predict with enough confidence (i.e., be trustworthy), especially during testing stage. Otherwise, the compounding of model uncertainties would accelerate the deviation from trustworthy regions, making the agent generate unreliable control sequences, hence leading to severe consequences and failure of the tasks, as is shown in a robotic example in Figure 1 (left). Specifically, in practical applications, such safety requirements for reliable decision making are extremely strict [5], demanding that every decision made by the agent at each step be reliable. Meanwhile, current methods like the pessimistic and distributional robust RL (DDRL) methods fall short in handling this issue. For instance, Pessimistic methods such as MOPO [6], PBRL [7] and RORL [8] mainly concentrate on making the agent act in accordance with the demonstrations in the dataset. Although these methods are good at quantifying those out-of-distribution (OOD) data points, they may not entirely fulfill the aforementioned safety requirements. On the other hand, Robust RL methods [9], [10] aim to improve the agent's capacity to deal with distributional shift by establishing the uncertainty set of the transition function and optimizing the lower bound of the policy's long term returns under the worst case scenario within this set. Unfortunately, Robust RL also fails to take into account the safety issue described earlier. In other words, both kinds of methods are unable to prevent the learned agent from straying from the trustworthy regions and perform stably reliable control.

Inspired by control-invariant sets in control theory [11], [12], where a closed trustworthy region is delineated in the state space to offer a reliable working environment for the agent, as is shown in Figure 1 (middle). Based on this idea, a recent method named Lyapunov Density Model [4] defines a region based on data density distribution to ensure adequate data support for the agent during operation. However, setting such a region based on a common and pessimistic density criterion overlooks the issue of performance imbalances in complex environments, where data requirements for achieving a certain performance level may vary across regions. For example, in autonomous driving, data needs for learning on a smooth highway differ significantly from those on a rugged mountain road. Instead, we advocate using a metric linked to model performance (e.g., value functions, policies) - Bellman uncertainty - for region definition to meet reliability standards in such complex and data-imbalanced environments.

In particular, we aim to define a confined state space for the agent's stably reliable operation, where the learned models demonstrate reduced Bellman uncertainty and trustworthy predictions. To achieve this, we introduce a novel offline reinforcement learning method that regulates the expected Bellman uncertainty associated with the new policy. This regulation ensures that, at subsequent states, the growth trend of uncertainty remains within acceptable limits, allowing the agent to navigate low-uncertainty regions that serve as safe zones. Drawing inspiration from the control Lyapunov

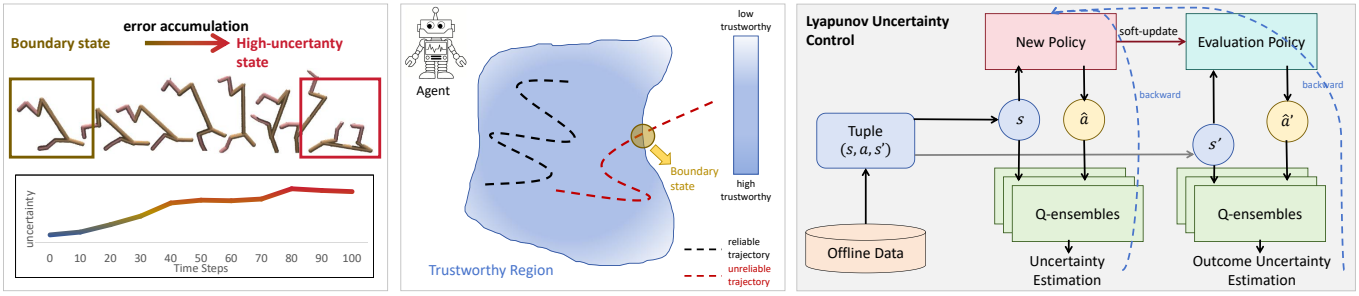

Fig. 1. (left) An illustration of the unreliable trajectory starting from the state with high risk of deviating from trustworthy region, i.e., boundary state, as demonstrated on a Halfcheetah robotic agent. (middle) The basic idea behind LUC - it scopes an enclosed trustworthy region for the agent to operate within, and makes the agent generate reliable trajectories marked as black line. (right) The framework of the proposed method that performs stably reliable control based on the reliability of the utilization at certain data points.

functions used in optimal control, we refer to our approach as Lyapunov Uncertainty Control (LUC). We implement our Lyapunov Uncertainty Control (LUC) method using a standard deviation-based uncertainty measure that relies on Q-ensembles, as described in [7]. The framework of LUC that makes decisions based on the reliability of given data points is illustrated in Figure 1 (right). Theoretically, we demonstrate that our approach can confine the learned agent to operate within a low-uncertainty state enclosure, resulting in secure and reliable trajectories. Furthermore, in certain scenarios, it can enhance the minimum performance bound of the new policy. Finally, we conduct extensive experiments to showcase the effectiveness and feasibility of LUC across various tailored benchmarks, including OOD benchmarks and those with poor demonstrations.

To sum up, the main contributions of our work are:

1) **Novel Lyapunov-Inspired Framework for Offline RL Reliability:** Introduced Lyapunov Uncertainty Control (LUC), a novel offline RL framework inspired by Lyapunov stability theory, which uniquely aims to ensure trajectory-level reliability by explicitly regulating the growth trend of expected Bellman uncertainty, rather than just penalizing current uncertainty.
2) **Theoretical Foundation for Reliable Policies:** Established a theoretical basis by defining *Lyapunov Policies* and proving their connection to reliable operation (confining trajectories within low-uncertainty regions), and linked the LUC objective to potentially improved performance lower bounds.
3) **Empirically Validated Robustness and Performance:** Demonstrated LUC's practical effectiveness through extensive experiments, showcasing competitive performance on standard D4RL benchmarks (particularly robust on non-expert data), superior handling of Out-of-Distribution (OOD) challenges, and efficacy in complex environments compared to relevant baselines.

II. Related Works

Offline reinforcement learning algorithms primarily grapple with the challenge of *distributional shift*, where the learned policy deviates from the behavior policy underlying the data, potentially leading to unreliable value estimates and poor performance. Approaches, as illustrated in Figure 2, to address this can be broadly categorized.

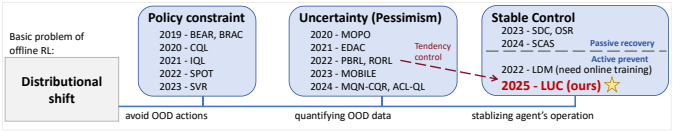

Fig. 2. Diagram illustrating the landscape of related offline RL methods and the positioning of LUC. The yellow star highlights methods specifically designed to achieve stable closed-loop control using offline data.

Policy Regularization and Constraint Methods: A major line of research aims to mitigate *distributional shift* by explicitly constraining the learned policy to remain close to the data distribution. Techniques include directly regularizing the policy divergence from the behavior policy (e.g., BEAR [13], BRAC [14], CQL [15]), implicitly learning value functions without OOD querying (e.g., IQL [16]), or restricting actions to the support set observed in the data (e.g., SPOT [17], SVR [18]). While effective in preventing extreme OOD actions, these methods can sometimes be overly conservative, limiting performance, especially when the optimal policy significantly differs from the behavior policy.

Uncertainty Quantification and Pessimism: Another prominent strategy employs pessimism, guided by quantifying the epistemic uncertainty inherent in learning from limited offline data. This uncertainty, often estimated using ensembles (e.g., PBRL [7], EDAC [19], RORL [8]), dropout, or by measuring Bellman error inconsistency (e.g., MOPO [6], MOBILE [20]), is incorporated into the learning objective to penalize actions or state transitions associated with high uncertainty. Other approaches leverage distributional RL concepts, like Monotonic Quantile Networks (MQN-CQR) [10], to achieve robustness or a form of pessimism by focusing on worst-case quantile estimates. Some methods, like ACL-QL [21], attempt to adapt the level of conservatism during learning, often based on uncertainty or related metrics, which encourages the agent to favor actions where the value estimates are deemed more reliable. A key distinction of our work is that while these pessimistic methods primarily penalize current uncertainty, they often overlook the potential for uncertainty to accumulate or grow over subsequent steps, which is critical for trajectory-level reliability.

Stable Control Methods: Drawing inspiration from control theory and addressing potential deviations during deployment, this category includes methods focusing on stability and recovery. State recovery approaches (e.g., SDC [22], OSR [5], SCAS [23]) aim to guide the agent back to familiar regions after a deviation occurs, often using learned dynamics models. These are primarily reactive mechanisms. Lyapunov-based ideas have also been explored; for instance, Lyapunov Density Models (LDM) [4] define safe regions based on data density, treating density as a candidate Lyapunov function. Our proposed Lyapunov Uncertainty Control (LUC) method also draws inspiration from Lyapunov stability but differs significantly. Instead of data density, LUC utilizes model uncertainty (approximated Bellman uncertainty) as the core metric. Crucially, LUC introduces a novel constraint mechanism focused on regulating the future growth trend of this uncertainty, aiming to proactively confine the agent's entire trajectory within a low-uncertainty enclosure, thereby ensuring stable and reliable control from the outset, rather than just reacting to deviations or relying solely on data density.

III. Notions

Some important notions are listed in this section.

Notion	Meaning
S / A	State / Action spaces
R	Reward function $R: S \times A \to \mathbb{R}$
γ	Discount factor
$\rho_0(s)$	Initial state distribution
$\mathcal{D}$	Static offline dataset
π / Π	New policy or any policy: $\pi: S \to A$ / Policy candidate set
f	Any function that $f: S \times A \to \mathbb{R}$
$d^\pi(s)$	State stationary distribution of policy π
π_β / π^*	Behavior policy / Optimal policy
$M(s,a)$ / $P(s'\|s,a)$	Deterministic / Stochastic transition function
$\mathcal{T}$ / $\hat{\mathcal{T}}$	(Online) Bellman / Empirical (Offline) Bellman operator
Q / $\hat{Q}$ / $Q^*(f^*)$	Q function / Empirical Q function / Optimal Q function
$\mathcal{G}_f(\pi)$	$f-\pi$ trustworthy region defined in Definition 1
$\zeta_f(s,\pi)$	Arbitrary uncertainty quantification based on f, s and π
C^*	Optimal coverage constant, s.t., $\sup_{s,a} \frac{\pi^*(a\|s)}{\pi_\beta(a\|s)} \le C^*$

IV. Preliminaries

In the standard formulation of reinforcement learning, a Markov Decision Process (MDP) is used to model the problem. It is represented by a tuple $(S, A, P, R, \gamma, \rho_0)$, where S denotes the state space, A represents the action space, M is the transition function (in a deterministic transitioned MDP, $M(s,a) = s'$, while in a stochastic transitioned MDP, $P(s'|s,a)$ is a distribution of states), R is the reward function with upper bound R_{max}, γ is the discount factor, and ρ_0 is the initial state distribution. A policy, denoted as $\pi : S \to A$, guides the decision-making process in interacting with the environment. To evaluate the expected cumulative rewards, a Q-value function $Q^\pi(s,a)$ is defined as $(1-\gamma)\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(a_t|s_t))|s,a]$. For convenience, the γ-discounted future state distribution (stationary state distribution) is defined as $d^\pi(s) = (1-\gamma)\sum t = 0^\infty \gamma^t Pr(s_t = s; \pi, \rho_0)$, with ρ_0 representing the initial state distribution and $(1-\gamma)$ is the normalization factor.

In the offline setting, Q-Learning [24] is used to learn a Q-value function $\hat{Q}(s,a)$ and a policy π from a dataset $\mathcal{D}$ collected by a behavior policy π_β. The dataset consists of quadruples $(s,a,r,s') \sim d^{\pi_\beta}(s)\pi_\beta(a|s)P(r|s,a)P(s'|s,a)$. The objective is to minimize the Bellman error over the offline dataset [24] and search for a good policy in the policy candidate set $\Pi \subset (S \to \triangle(A))$ under the supervision of a value-function class $\mathcal{F} \subset (S \times A \to [0, V_{max}])$ to model the Q-value function,

$$Q \longleftarrow \arg\min_{Q} \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\big[r + \gamma[\max_{\pi\in\Pi} \mathbb{E}_{a'\sim\pi(\cdot|s')} Q(s',a')] - Q(s,a)\big]^2 \quad (1)$$

More specifically, in this paper, we denote the optimal Bellman operator over Π as $\mathcal{T}^\Pi f(s,a) = r(s,a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}[\max_{\pi\in\Pi} \mathbb{E}_{a'\sim\pi(\cdot|s')} Q(s',a')]$, and the empirical Bellman operator as $\hat{\mathcal{T}}^\Pi f(s,a) = r(s,a) + \gamma\mathbb{E}_{s'\sim\hat{P}(\cdot|s,a)}[\max_{\pi\in\Pi} \mathbb{E}_{a'\sim\pi(\cdot|s')} Q(s',a')]$, where $\hat{P}$ is the empirical dynamics model based on the dataset. It is worth noting that the TD target in Eq.(1) is estimated by the empirical Bellman operator.

V. Method

This section provides a detailed introduction to our work. In Sec.V-A and V-B, we formally define our objective mathematically as obtaining a policy that consistently operates within a reliable region. Subsequently, in Sec.V-C, we present a specific algorithm to accomplish this objective. Finally, in Sec.V-D, we analyze the theoretical properties of the algorithm, demonstrating its capability to improve the performance lower bound of the learned policy under specific scenarios.

A. Objective: Operating within Trustworthy Regions

In this section, before we formally define our conceptual framework for Lyapunov Uncertainty Control, we need to introduce the formulation of the basic problem - how to make the agent operate within trustworthy regions. Specifically, analogous to control-invariant sets in control theory, we first define trustworthy regions in the state space, where the policy π can operate effectively. That is, we need the agent operate stably at the trustworthy data points over which the learned models (e.g., Q-networks) have *high-confidence* predictions, hence its introduced policy would never generate those high-risk behaviors (OOD actions or actions with OOD outcomes). Previous methods defined these regions using density models [4]; however, as mentioned earlier, a local region may not be trustworthy even with high density due to the complexity and nonlinearity of the underlying environment. Instead, we introduce a measurement based on epistemic uncertainty, denoted as $\zeta_f(s,a)$, where the precise computational method for $\zeta_f(s,a)$ will be presented later; however, it is a positive scalar that is closely related to the agent's current knowledge, reflecting the confidence of the learned value function f at the input data (s,a). If f has a high confidence and is trustworthy at (s,a), $\zeta_f(s,a)$ will be small; conversely, if f predicts inaccurately, $\zeta_f(s,a)$ will be large.

Using $\zeta_f(s,a)$, we can evaluate whether an induced policy π can make reliable decisions at a given state s by assessing

the uncertainty of the learned value function f. Furthermore, we derive the following definition of the $f-\pi$ trustworthy region in Definition 1:

Definition 1: **($f-\pi$ trustworthy region).** Given an arbitrary value function f, policy π and a threshold c, we define the $f-\pi$ trustworthy region over the state space,

$$\mathcal{G}_f(\pi) = \{s|\zeta_f(s,\pi) \leq c\} \tag{2}$$

where $\zeta_f(s,\pi) = \mathbb{E}_{a\sim\pi(\cdot|s)}\zeta_f(s,a)$.

As shown in Figure 1, where 'low-uncertainty region' illustrates the agent's $f-\pi$ trustworthy regions. In practice, if the agent operates beyond its trustworthy regions, it would accumulate decision errors, finally failing the task. On the other hand, if an agent with policy π always operate within its $f-\pi$ trustworthy regions, we call this policy a reliable policy, defined in Definition 2, because such policy will never behave improperly on its operating areas.

Definition 2: **(Reliable policy).** A policy π is reliable if it satisfies that $\forall s \in \mathcal{D}\cap\mathcal{G}_f(\pi)$, if $\forall t, s_t \in supp(P(s_t|s_0 = s,\pi))$, then $s_t \in \mathcal{G}_f(\pi)$.

The reliable policy in Definition 2 constrains the policy with an important property - the trajectory generated by it would lie within the $f-\pi$ trustworthy region, as in Definition 1, from which aspect, the policy can make reliable decisions because the learned models (e.g., Q-networks) have high confidence over every states along with the whole trajectory. However, it is computationally expensive to control the reliability of the whole trajectory, especially when the trajectory's length is infinite. This motivates us to develop a feasible control method, i.e., the Lyapunov reliable policy, to address the aforementioned issues, which will be introduced in the next subsection. Before that, we then need to show the existence of the reliable policy to guarantee that the problem is solvable.

Proposition 1: **(Existence of reliable policy.)** Suppose the dataset have a sufficient coverage over the optimal policy, i.e., $\sup_{s,a}\frac{\pi^*(a|s)}{\pi_\beta(a|s)} \leq C^*$. Then there exists a reliable policy.

Proof sketch. From the assumption that the dataset have a sufficient over the optimal policy, we have all the behaviors of the optimal policy would be supported by the dataset, i.e., $\pi^* \subseteq \mathcal{D}$. Then the optimal stationary state distribution is also supported by the dataset $d^{\pi^*}(s) \subseteq \mathcal{D}$. Due to the fact that all the states over the dataset are contained in the $f-\pi^*$ reliable region, and $d^{\pi^*}(s) \subseteq \mathcal{D}$, so the optimal policy would never operate beyond its $f-\pi^*$ reliable region, hence it is a reliable policy.

Proposition 1 shows that there would always exist a reliable policy in the MDP system with sufficient data, which motivates us to learn such a policy for reliable control.

B. Method: Learning a Lyapunov reliable policy

To learn a reliable policy in Definition 2, we need to control the outcomes of the learned policy not beyond the scoped trustworthy regions. Intuitively, inspired by the Lyapunov control [4], if we control model uncertainty for the current decision-making and its growth tendency along with whole generated trajectories at the same time, the agent will be more likely to stably operate within such trustworthy regions. We call such a policy as Lyapunov policy, if it satisfies the properties in Definition 3: not only manages uncertainty at the current step but also addresses the tendencies of these uncertainty one step ahead. In other words, a Lyapunov policy is capable of controlling current step uncertainty to encompass trustworthy regions over the state space, while also restricting one-step forward uncertainty to ensure trajectory reliability.

Definition 3: **(Lyapunov policy).** Given an arbitrary value function f and offline dataset $\mathcal{D}$. A policy π is a Lyapunov policy if it satisfies

$$1.\forall s \in \mathcal{D}, \zeta_f(s,\pi) \leq c; \tag{3}$$

$$2.\forall s \in \mathcal{D}, \max_{\hat{a}\in\pi}\zeta_f(M(s,\hat{a}),\pi) \leq \zeta_f(s,\pi). \tag{4}$$

where M is the deterministic transition.

The first condition of Lyapunov policy in Eq.(3) is aligned with the traditional pessimistic conditions like in [7], [8], [25]. Here, this spans the trustworthy regions of the learned policy to enhance the generalization ability of the agent. Then the second condition in Eq.(4) constrains the learned policy's outcomes to suppress the growth trend of uncertainty along with the trajectory. Then we have the following results, shown in Theorem 1, to demonstrate the effectiveness of the proposed Lyapunov policy - the aforementioned two conditions are sufficient to learn a reliable policy, which can control the agent stably operating within its trust worthy regions.

Theorem 1: In a deterministic transitioned MDP, a Lyapunov policy π is a reliable policy.

Proof sketch. We prove in a contradiction way. Denote the deterministic transition as $M(s,a) = s'$. First, we suppose the Lyapunov policy π is not a reliable policy, then we have: $\forall s_0 \in \mathcal{D}\cap\mathcal{G}_f(\pi)$, $\exists t, s_t \in supp(P(s_t|s_0,\pi))$, such that $s_t \notin \mathcal{G}_f(\pi)$. Then we have $\zeta_f(s_t,\pi) > c$. Then we aim to find the contradiction. From the definition of Lyapunov policy, $\forall s \in \mathcal{D}$, we have,

$$c < \zeta_f(s_t,\pi) \leq \max_{\hat{a}\in\pi}\zeta_f(P(s_{t-1},\hat{a}),\pi) \leq \zeta_f(s_{t-1},\pi) \tag{5}$$

Then we have $\zeta_f(s_{t-1},\pi) > c$. By recurrently applying the above derivation by t times, we would have $\zeta_f(s_0,\pi) > c$. Then we have $s_0 \notin \mathcal{G}_f(\pi)$, which is conflicted with $s_0 \in \mathcal{D}\cap\mathcal{G}_f(\pi)$. Hence completing the proof, and we can conclude that the Lyapunov policy is reliable.

Essentially, this theorem says that a Lyapunov policy is also a *Lyapunov reliable policy*. In other words, if a policy is a Lyapunov policy, then it will operate within its enclosed trustworthy region.

To summary, the main theoretical results in the above two subsections are used for problem formulation and functional requirements of the method - Definition 1 defines a trustworthy region for the agent to operate, and Definition 2 defines reliable policy based on it, which is able to verify the safety requirements, i.e., operate within the region defined by Definition 1; Definition 3 and Theorem 1 indicate what kind of policies need to be learned to ensure that the agent can operate stably within the trustworthy region without exceeding it. Proposition 1 demonstrates the existence of policies that meet these functional requirements, validating the feasibility of the method proposed in this paper.

C. Implementing Lyapunov Uncertainty Control by Value Estimation

In this section, we aim to learn the Lyapunov policy, as in Definition 3, from the offline data in a model-free manner, and provide a framework for practical implementation. To be specific, first we need to seek a way to realize the uncertainty quantifier ζ_f. Then we proposed the loss function to train the Q-networks and policy networks, i.e., the actor-critic framework, in an offline way. Besides, in Proposition 2, we also theoretically show the effectiveness of our practical algorithms in finding a Lyapunov policy.

We use the Bellman uncertainty quantifier as in [25] to implement the value-epistemic uncertainty in Definition 1,

$$\zeta_f(s,a) = \|\mathcal{T}f - \hat{\mathcal{T}}f\|(s,a) \tag{6}$$

where f is the learned value function. $\mathcal{T}$ is an arbitrary Bellman operator, while $\hat{\mathcal{T}}$ is its empirical version according to the dataset. Previous studies [19] suggest that Bellman uncertainty can rely on model predictions to evaluate state-action pairs. High variance in the model's prediction for a particular action implies inadequate data support, leading to low reliability. This property confirms that Bellman uncertainty aligns with the requirement in Definition 1.

Next, our objective is to acquire the Lyapunov reliable policy outlined in Definition 3 from the offline dataset using a model-free approach. Here, we present the Lyapunov value estimation, which straightforwardly penalizes not only the Q-value functions using the uncertainty quantifier from Eq.(6) at the current time step but also the increasing decision uncertainty tendency based on the next time step's situation, as,

$$\begin{aligned}\mathcal{L}_{LUC}(s,a,s',a',f) = \mathbb{E}_{a\sim\pi(\hat{a}|s)}\zeta_f(s,\hat{a}) \\ + \beta\cdot(\zeta_f(s',a') - \zeta_f(s,a))\end{aligned} \tag{7}$$

Subsequently, we apply regularization using the offline dataset $\mathcal{D}$ and the learned models (Q function f_k and new policy π), incorporating it as a penalty in the value estimation,

$$f_k(s,a) \leftarrow f_k(s,a) - \hat{\beta}\cdot\mathbb{E}_{a'\sim\pi(\cdot|s')}\mathcal{L}_{LUC}(s,a,s',a',f_k) \tag{8}$$

where f_k is the Q function learned at the k^{th} iteration. $\hat{\beta}$ is the balance coefficient. (s,a,s') is the tuple sampled from the offline dataset $\mathcal{D}$, and the π is the currently learned policy.

Proposition 2: Suppose the action distribution of new policy $\pi(a|s)$ is positive correlated to the learnt Q function $f(s,a)$, i.e., $\pi(a|s) \propto f(s,a)$. Then the proposed Lyapunov value estimation in Eq.(7) induces a Lyapunov reliable policy as defined in Definition 3, which satisfies:

$$\forall s\in\mathcal{D}, \zeta_f(s,\pi)\le c; \forall s\in\mathcal{D}, \max_{\hat{a}\in\pi}\zeta_f(M(s,\hat{a}),\pi)\le\zeta_f(s,\pi).$$

Proof sketch. The first term in Eq. (8) aligns with the constraint $\forall s\in\mathcal{D}, \zeta_f(s,\pi)\le c$, which ensures that the new policy can make trustworthy decisions at every state from the offline dataset. On the other hand, in the deterministic transitioned MDP, if the learned policy utilizes all the offline data points, the second term in Eq. (8) implies that the new policy will not select actions with a high risk of uncertainty accumulation. This marginalizes those data points with a high risk of deviation and aligns with the constraint $\forall s\in\mathcal{D}, \max_{\hat{a}\in\pi}\zeta_f(M(s,\hat{a}),\pi)\le\zeta_f(s,\pi)$, thus completing the proof. *Proof is available in Appendix A.* Proposition 2 demonstrates that the policy induced by the value function trained with Eq.(8) may exhibit the traits of the Lyapunov reliable policy described in Definition 3, fulfilling the reliability criteria in our study.

Then like previous pessimistic methods [7], [8], [19], we approximate the uncertainty quantifier in Eq.(6) as the standard deviation as,

$$\zeta_f(s,a) \approx \beta\cdot Std(f^i(s,a)) = \beta\cdot\sqrt{\frac{1}{K}\sum_{i=1}^{K}\big(f^i(s,a)-\bar{f}(s,a)\big)} \tag{9}$$

where $\{f^i\}_{i=1}^K$ is the learned Q-ensembles and $\bar{f}$ is the average of the K Q-esembles, and β is the balance-coefficient. Then the objective in Eq.(7) is converted to,

$$\begin{aligned}\mathcal{L}_{LUC}(s,a,s',a',f) = (1-\beta)\cdot Std(f^i(s,a)) \\ + \beta\cdot\gamma\cdot Std(f^i(s',a'))\end{aligned} \tag{10}$$

where $U_f(s,a) = std(f^i(s,a))$ and $\{f^i\}_{i=1}^K$ is the learned K Q-ensembles. In practice, the balance coefficient β is usually selected in $(0,1)$. Then the regularization of LUC is,

$$\begin{aligned}\mathcal{L}_{LUC}(f^i,\pi) = \mathbb{E}_{(\hat{s},\hat{a},\hat{s'}\sim\hat{\mathcal{D}})}(f^i(\hat{s},\hat{a}) \\ - \hat{\beta}\cdot\mathbb{E}_{\hat{a'}\sim\pi(\cdot|\hat{s'})}\mathcal{L}_{LUC}(\hat{s},\hat{a},\hat{s'},\hat{a'},f))\end{aligned} \tag{11}$$

where $\hat{\mathcal{D}}$ is the constructed noisy dataset. To be specific, the noisy samples in $\hat{\mathcal{D}}$, as $\hat{x} = (\hat{s},\hat{a},\hat{s'})$, are the noised version of samples, $x = (s,a,s')$, in the original offline dataset $\mathcal{D}$, with $\hat{x} = x + \lambda\cdot\epsilon$, and ϵ is the attached perturbation. Previous studies [5], [7], [22], [26] have empirically demonstrated the effectiveness of noise injection in regulating the out-of-distribution (OOD) performance of the trained agent. In the majority of our experiments, ϵ is randomly drawn from a standard Gaussian distribution (also, in the OOD observation experiments in Appendix VI-E, ϵ is generated adversarially as in [8]).

Then the loss functions of the ensemble critic networks (L_c) and the actor network (L_a) are as,

$$\begin{aligned}\mathcal{L}_c = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\frac{1}{K}\sum_{i=1}^{K}\Big[r+\gamma\mathbb{E}_{a'\sim\pi(\cdot|s')}\big[\min_{j=1\dots K} f'_j(s',a')\big] \\ - f_i(s,a)\Big]^2 + \frac{1}{K}\sum_{i=1}^{K}\mathcal{L}_{LUC}(f^i,\pi)\end{aligned} \tag{12}$$

$$\mathcal{L}_a = \mathbb{E}_{s\sim\mathcal{D}}\mathbb{E}_{a\sim\pi(\cdot|s)}\big[\min_{i=1\dots K} f'_i(s',a')\big] \tag{13}$$

To sum up, we present our overall approach in Algorithm 1, as follows,

D. Theoretical Analysis

In this section, by Theorem 2, we aim to show that the value function obtained through k-step iterations of the empirical Bellman operator is determined by two factors concerning

Algorithm 1 The pseudocode of Lyapunov Uncertainty Control (LUC) algorithm

Input: Offline dataset $\mathcal{D}$. Max episode T.

Initialize the policy network, K Q-networks.
Perform the noise injection to generate the noisy dataset $\hat{\mathcal{D}}$.
while $t < T$ **do**
Sample mini-batch of transitions $(s, a, r, s') \sim \mathcal{D}$ and transitions $(\hat{s}, \hat{a}, \hat{s}') \sim \hat{\mathcal{D}}$
Update the K Q-networks by minimizing $\mathcal{L}_c$ according to Eq.(12)
Update the policy network minimizing $\mathcal{L}_\pi$ according to Eq.(13)
end while

Output: The learned policy network π.

the true optimal value function: 1) the single-step Bellman uncertainty generated by policies in the policy candidate set, and 2) the growth tendency of Bellman uncertainty along trajectories generated by policies in the policy candidate set. The former has been the focus of previous methods like PBRL [7]; however, Theorem 2 in this paper indicates that to learn a better value function, attention must be paid to both factors simultaneously.

Theorem 2: **(Performance lower bound.)** Given an MDP with max reward R_{max} and a dataset of size N. Given (s, a) pair, we denote its data density over the dataset is $d(s, a)$. Given an empirical Bellman operator $\hat{\mathcal{T}}^{\Pi}$ and an arbitrary policy candidate set Π, where $\hat{\mathcal{T}}^{\Pi} f(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim \hat{P}(s'|s,a)} \max_{\pi \in \Pi} f(s', \pi)$. Denote the learnt value function as f_k, with k iterations of $\hat{f}_k = \hat{\mathcal{T}}^{\Pi} \hat{f}_{k-1}$, and the true optimal value function as f^*. Then we have,

$$\|\hat{f}_k - f^*\|_d \leq \frac{C^*}{1-\gamma} \cdot \sup_{\pi \in \Pi} \sum_{s_0} d(s_0) \zeta_{\hat{f}_k}(s_0, \pi) + \mathcal{O}\Big(\sup_{\substack{\pi \in \Pi, T \geq 0 \\ s_0, a_0, d(s_0, a_0) > 0}} \mathbb{E}_{P(\tau_T | \pi, s_0, a_0)} \Big(\sum_{t=0}^{T-1} [\gamma^{t+1} \zeta_{\hat{f}_k, t+1} - \gamma^t \zeta_{\hat{f}_k, t}] \Big)^2 \Big) \tag{14}$$

where $\zeta_{\hat{f}_k, t}$ is the Bellman uncertainty at time step t, i.e., $\zeta_{\hat{f}_k, t} = \|\mathcal{T}^{\Pi} \hat{f}_k - \hat{\mathcal{T}}^{\Pi} \hat{f}_k\|(s_t, a_t)$. τ_T is the trajectory with length of T. And C^* is assumed by $\sup_{s,a} \frac{\pi^*(a|s)}{\pi_\beta(a|s)} \leq C^*$.

Proof of Theorem 2 is found in Appendix B. Theorem 2 primarily shows the following points:

1) The role of LUC term in enhancing the agent's performance - it helps optimize the lower bound of the agent's performance. Specifically, Theorem 2 illustrates that when iterated using the empirical Bellman operator, the difference between the learned value function $\hat{f}_k$ and the true optimal value function f^*, which is also known as the performance lower bound of the offline algorithm, can be controlled through the LUC method.

2) One intuitive way to understand Theorem 2 is that we control the right term in Eq.(14) by adjusting the policy candidate set, thereby enhancing the lower bound of the method's output policy performance. Specifically, we align the policy candidate set with the definition of Lyapunov reliable policy (Definition 3) through the loss function in Eq. (7).

3) Such operation can control the right term of Eq.(14). And then consequently improve the lower bound of the algorithm's performance. The proposed method would not conflict with the objective of approaching the optimal policy, under the assumption of optimal coverage.

Then to further simplify the calculation complexity, Proposition 3 indicates that one-step Lyapunov Uncertainty-penalization could bound the second term in Eq.14.

Proposition 3: If the first term of Eq.(14) is bounded, i.e., $\forall \pi \in \Pi$, we have $\mathbb{E}_{d(s_0)} \zeta_{\hat{f}_k}(s_0, \pi) \leq c$, then we can bound the second term with one-step Lyapunov Uncertainty-penalization, i.e., $\forall s \sim \mathcal{D}$,

$$\min_\pi [\gamma \mathbb{E}_{P(s'|s,\pi)} \zeta_{\hat{f}_k, t+1}(s', \pi) - \zeta_{\hat{f}_k, t+1}(s, \pi)]$$
$$\Rightarrow \min_\pi \mathbb{E}_{P(\tau_T | \pi, s_0 = s)} \Big(\sum_{t=0}^{T-1} [\gamma^{t+1} \zeta_{\hat{f}_k, t+1} - \gamma^t \zeta_{\hat{f}_k, t}]$$

Furthermore, assume the dataset fully covers dynamics modes, i.e., $\forall s, a \in \mathcal{D}, supp(P(s'|s, a)) \subseteq supp(\hat{P}(s'|s, a))$, then the left part is controlled by Lyapunov value estimation in Eq.(7).

Then the left term in Proposition 3 could be empirically estimated by the Lyapunov value estimation as in Definition 3. Theorem 2 and Proposition 3 demonstrate that to control the performance lower bound of the learned agent, despite controlling the current step's Bellman uncertainty, it is also important to control the one-step forward growth tendency of the Bellman uncertainty along with the whole trajectory, which is the main contribution of this paper. This helps the learned value function to be more likely to converge to the fixed point of the empirical Bellman operator, which is hence for controlling the performance lower bound of the learned agent.

VI. Experimental results

Our experiments primarily aim to address three key questions:

1) Can LUC enhance the state-of-the-art performance on standard MuJoCo benchmarks?
2) Is LUC capable of consistently learning reliable operation regions from noisy datasets with poor demonstrations?
3) Does LUC exhibit superior generalization ability in avoiding deviations from reliable regions under various types of OOD perturbations?

Our experimental section includes the following components: first, we validate the performance of the method proposed in this paper on standard D4RL benchmarks, particularly on non-expert datasets, demonstrating our method's superiority over others, addressing question 1. Next, to address question 2, we design noise data at different levels - where noise represents the discrepancy between the policy and the optimal policy, resulting in varying degrees of poor demonstrations. We then evaluate the performance of different algorithms on such noisy data. Subsequently, we introduce Out-of-distribution (OOD) MuJoCo benchmarks with various perturbations to

TABLE I
NORMALIZED SCORES ON STANDARD MUJOCO TASKS, AVERAGED OVER 8 RANDOM SEEDS. PART OF THE RESULTS ARE REPORTED IN THE RORL, MOBILE, OSR AND SCAS PAPERS. TOP SCORES FOR EACH TASK ARE HIGHLIGHTED. (·) INDICATES THE AVERAGE WITHOUT 'EXPERT' DATASETS.

	Task Name	CQL	PBRL	MOPO	RORL	MOBILE	SPOT	SDC	OSR	MQN-CQR	ACL-QL	SCAS	**LUC (Ours)**
	Year	2020	2022	2020	2022	**2023**	2022	2022	**2023**	**2024**	**2024**	**2024**	**2025**
halfcheetah	random	31.3±3.5	11.0±5.8	35.4±2.5	28.5±0.8	**39.3**±3.0	26.5±4.3	36.2±1.3	35.2±1.2	32.6±2.9	28.5±4.6	12.2±3.2	32.1 ± 0.6
	medium	46.9±0.4	57.9±1.5	42.3±1.6	66.8±0.7	**74.6**±1.2	58.4±1.0	47.1±0.2	48.8±0.2	45.1±1.5	69.8±2.5	46.6±0.2	68.3 ± 1.1
	medium-expert	95.0±1.4	92.3±1.1	63.3±38.0	107.8±1.1	108.2±2.5	86.9±4.3	101.3±3.6	94.7±0.9	71.1±4.9	87.4±7.5	91.7±2.7	**111.7**± 1.1
	medium-replay	45.3±0.3	45.1±8.0	53.1±2.0	61.9±1.5	**71.7**±1.2	52.2±1.2	47.3±1.0	46.8±0.4	45.3±7.9	55.9±6.7	44.0±0.3	65.9 ± 1.6
	expert	97.3±1.1	92.4±1.7	–	105.2±0.7	–	97.6±1.4	106.6±1.1	97.7±0.7	**109.1**±0.2	–	–	108.4± 0.5
hopper	random	5.3±0.6	26.8±9.3	11.7±0.4	31.4±0.1	**31.9**±0.6	28.7±3.1	10.6±0.3	13.5±3.4	13.2±0.6	**33.5**±1.2	31.4±0.1	31.8 ± 1.3
	medium	61.9±6.4	75.3±31.2	28.0±12.4	104.8±0.1	106.6±0.6	86.0±8.7	91.3±0.8	83.1±1.9	94.7±13.2	97.9±6.4	102.5±0.3	**106.9**± 0.4
	medium-expert	96.9±15.1	110.8±0.8	23.7±6.0	112.7±0.2	112.6±0.2	99.3±7.1	112.9±0.2	113.1±0.6	113.0±0.5	107.2±4.9	109.7±3.5	**114.2**± 1.3
	medium-replay	86.3±7.3	100.6±1.0	67.5±24.7	102.8±0.5	**103.9**±1.0	100.2±1.9	48.2±2.7	96.7±1.7	95.6±18.5	99.3±3.9	101.6±1.0	**104.1**± 0.8
	expert	106.5±9.1	110.5±0.4	–	112.8±0.2	–	112.3±0.7	112.6±0.1	113.1±0.5	111.3±0.1	–	–	**114.4**± 0.5
walker2d	random	5.4±1.7	8.1±4.4	13.6±2.6	21.4±0.2	17.9±3.0	5.0±2.1	14.3±4.5	14.1±3.9	22.6±6.1	22.3±1.2	1.4±1.1	**25.9** ± 1.1
	medium	79.5±3.2	89.6±0.7	17.8±19.3	102.4±1.4	87.7±1.1	86.4±2.7	81.1±0.4	85.1±0.7	80.0±0.5	79.3±8.2	82.3±3.0	**103.4**± 1.2
	medium-expert	109.1±0.2	110.1±0.3	44.6±12.9	121.2±1.5	115.2±0.7	112.0±0.5	105.3±4.0	112.9±0.8	112.1±8.9	113.4±6.3	108.4±3.7	**124.1**± 0.9
	medium-replay	76.8±10.0	77.7±14.5	39.0±9.6	90.4 ±0.5	89.9±1.5	91.6±2.8	30.3±4.7	87.8±1.0	52.3±16.7	**96.5**±4.2	78.1±4.5	92.3 ± 1.4
	expert	109.3±0.1	108.3±0.3	–	115.4±0.5	–	109.7±0.5	108.3±3.7	110.3±0.4	113.1±1.5	–	–	**116.3**± 0.4
Average score		70.2	74.4	(36.7)	85.7	(80.0)	76.9	70.2	76.7	74.1	(74.3)	(67.5)	**88.0 (81.7)**

increase the likelihood of entering high-uncertainty states, assessing the agent's OOD generalization capability, answering question 3. Finally, we conduct ablation experiments to verify the effectiveness of the LUC method.

A. Learning on standard MuJoCo benchmarks

We benchmark LUC on three D4RL [27] environments (Halfcheetah, Hopper, Walker2d), using the standard dataset configurations: 'random' (random policy), 'medium' (partially trained SAC [28]), 'medium-replay' (replay buffer from 'medium' training), 'expert' (fully trained SAC), and 'medium-expert' (50% medium, 50% expert data). We compare LUC with several offline RL algorithms, including CQL [15], PBRL [7], MOPO [6], RORL [8], MOBILE [20], SPOT [17], OSR [5], MQN-CQR [29], ACL-QL [21] and SCAS [23]. Among these, PBRL [7], MOPO [6], and MOBILE [20] are related to LUC as they are all based on uncertainty penalization techniques. On the other hand, the state recovery meyhods, SDC [22], OSR [5] and SCAS [23] are the most closely related to LUC at the aspect of objective - they both aim to make the learned agent behave reliably[1].

The results are presented in Table I. It is evident that our method, LUC, outperforms other methods in most tasks and achieves a higher overall score. Particularly, LUC performs significantly better on non-expert datasets, notably the 'medium-expert' dataset, showcasing its robustness against the reward shift due to the conservative regularization term on non-expert data. Additionally, LUC's performance in the 'hopper' and 'walker2d' environments surpasses the State-of-the-art (SOTA), possibly due to these environments being more susceptible to noise from non-expert data.

Examining Table I, we note that performance on 'medium-expert' datasets frequently surpasses that on 'expert' datasets. This phenomenon, while potentially counterintuitive, can be explained by: **Broader Data Coverage**: 'Expert' datasets, despite their high quality, might cover only a narrow region of the state-action space. In contrast, 'medium-expert' datasets provide more diverse trajectories by including 'medium' data, which can be crucial for better generalization and more robust value function learning in offline RL. **Implicit Regularization Effect**: The non-expert trajectories within 'medium-expert' data can implicitly regularize the learning process, potentially mitigating overfitting to the specific behaviors found solely within the 'expert' dataset. For these two reasons, agents would perform better on 'medium-expert' datasets, especially the proposed LUC, because of the larger coverage of (low-uncertainty) state space.

Separately, the notably poorer results on 'medium-replay' datasets compared to 'medium' datasets stem from the former's composition. 'Medium-replay' datasets encompass the entire replay buffer content generated throughout the 'medium' policy's training, including samples from the initial, often random, exploration stages. This heterogeneity results in a significantly lower 'signal-to-noise ratio,' hindering effective policy extraction.

B. Learning on tasks with different levels of non-expert noise

In this section, we modify the discrepancy between the behavioral policy and the optimal policy by blending datasets produced from expert and random policies at various proportions to generate noisy datasets at different levels. Evaluating these datasets not only validates the influence of non-expert problems on conventional conservative constraints but also confirms the robustness of the proposed LUC method against noise stemming from non-expert data. We compare several representative methods: CQL [15], PBRL [7], and SDC [22].

The results are depicted in Figure 3. It is evident that most constraint-based methods are influenced by non-expert issues, wherein an increase in the discrepancy between the behavioral policy and the optimal policy leads to a substantial performance decline. However, as the randomness level rises, the performance degradation of LUC is comparatively minor, suggesting that LUC demonstrates greater resilience to noise from non-expert data under suboptimal behavioral policy settings. Particularly at elevated randomness levels (e.g., 0.9), LUC can maintain effective performance on these benchmarks.

[1]Unfortunately, as the LDM method [4] is mainly used in model-based RL as a constraint on the model optimizer, it is unclear how this method could be properly used for the task of offline RL, so we did not make any comparison with this method at the current stage.

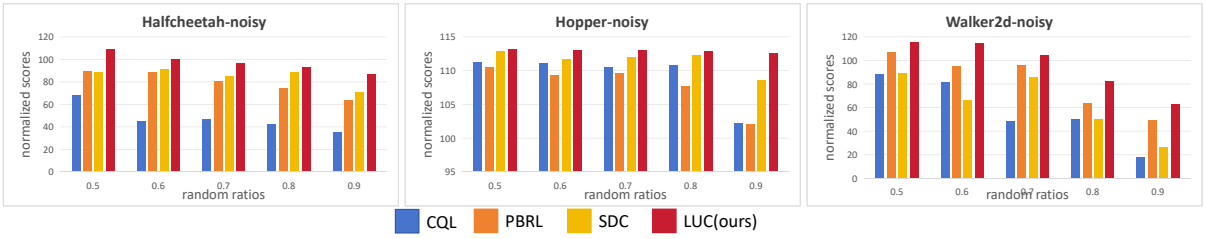

Fig. 3. Results of CQL, PBRL SDC and LUC on tasks with different levels of non-expert data.

C. Testing on Out-of-distribution MuJoCo benchmarks

To evaluate the agent's capacity to avoid straying from reliable regions, we introduce three types of OOD perturbations, applying varying intensities of perturbations at different intervals to the agent employed for a higher risk of deviation. This investigation encompasses three combinations of noise intensities and intervals . It is important to highlight that our approach in this study differs from the perturbation noise discussed in [8]; our method modifies the actual state in which the agent operates, rather than solely perturbing state observations. The research is centered on three MuJoCo environments: Halfcheetah, Hopper, and Walker2d; with the agent trained on 'medium-expert' datasets.

We have chosen four key algorithms, CQL, SDC, OSR and RORL, tailored for managing OOD states and observations, to contrast with the proposed LUC in these OOD benchmarks. The outcomes are detailed in Table II. Analysis reveals that LUC surpasses the other two methods across the majority of tasks, notably demonstrating substantial benefits in extensive OOD perturbation assignments like Halfcheetah and Walker2d. This implies that these settings might be more sensitive to OOD perturbations, necessitating advanced the agent to tackle OOD scenarios.

To delve deeper into the factors contributing to the superior performance of LUC in Halfcheetah-ood and Walker2d-ood tasks, we present the visualized results of LUC in Figure 4. The analysis reveals that each perturbation event leads the agent into high-uncertainty regions; however, LUC effectively prevents error accumulation and guides the agent back to lower-error (reliable) regions. This underscores the robustness of LUC in handling OOD scenarios by constraining the agent to operate within the reliable regions.

D. Complicated environments - AntMaze, Adroit and Finance

Compared to the MuJoCo environment, the AntMaze and Adroit environments require the agent to have the ability of multi-step dynamic planning, making them considered a more complex scenario. The results are shown in Table III and IV. In the AntMaze environment, based on the size and shape of the maze, it can be categorized into 'umaze (u),' 'medium (m),' and 'large (l)'; and based on different tasks, it can be classified as 'diverse (d)' and 'play (p)'. While the Adroit domain features three types of datasets: demonstration data from humans ("human"), expert data from a reinforcement learning policy ("expert"), and mixed data combining human demonstrations with an imitation policy ("cloned"). The tasks in Adroit are more complex than those in the Gym domain, and the inclusion of human demonstrations adds an additional layer of difficulty.

Here, we compare CQL [15], IQL [16], SPOT [17], ATAC [30], SDC [22], and OSR-10 [5] in AntMaze, while CQL [15] and PBRL [7] in Adroid.

From these benchmarks, we observe that our method outperforms the other methods in most benchmarks. In particular, our method performs much better in the "hammer," "door," and "large" maze benchmarks than other methods. This could be attributed to our method's ability to scope a safe region for the agent to stably operate within, thus addressing more complex environmental dynamics.

To evaluate performance in a setting derived from real-world data, we further compared LUC against CQL [15], SDC [22], OSR-10 [5], and SPOT [17] on the FinRL benchmark [31]. FinRL simulates trading across a pool of 30 stocks using 10 years of historical data. The agent's observation includes the number of shares held for each stock, five additional factor features per stock, and the current cash balance. The action space is 30-dimensional, corresponding to the desired transactions for each stock. We utilized the offline datasets for FinRL provided by NeoRL [32], which come in three quality levels: Low (L), Medium (M), and High (H), each comprising 100 trajectories. After offline training, all algorithms were evaluated through online interaction with the FinRL environment. These results are illustrated in Table V.

The FinRL results presented in Table V showcase LUC's strong performance in the simulated financial trading environment derived from real-world data, achieving the highest average score (521.4) among all compared methods. Notably, LUC significantly outperformed baselines on both the Low (L) and High (H) quality datasets, indicating its robustness in leveraging suboptimal data and effectively utilizing expert data without performance degradation. While highly competitive on the Medium (M) dataset, its superior overall performance, particularly at the extremes of data quality, suggests that LUC's focus on reliability through controlling uncertainty propagation translates effectively to complex, potentially noisy environments like financial markets.

E. Testing on environments with OOD observations

In this section, we conducted tests on benchmarks with OOD observations. This type of testing is primarily

TABLE II
THE RESULTS OF CQL, SDC, OSR, RORL AND LUC (OURS) ON OUT-OF-DISTRIBUTION MUJOCO BENCHMARKS. THE HIGHEST SCORES FOR EACH TASK ARE HIGHLIGHTED.

	Halfcheetah-ood			Hopper-ood			Walker2d-ood		
	small	medium	large	small	medium	large	small	medium	large
CQL	56.2 ±0.8	52.9 ±7.4	42.1 ±19.7	110.4 ± 0.4	97.3 ±4.2	54.4 ±7.4	81.1 ±2.8	77.4 ±10.1	52.0 ±13.3
SDC	100.9 ±3.3	99.8 ±5.5	83.3 ±6.8	112.0 ±0.6	110.8 ±2.1	89.2 ±6.4	102.7 ±4.4	102.2 ±5.1	95.6 ±3.6
OSR	93.8 ±3.9	90.4 ±6.1	88.7 ±11.3	111.3 ±1.7	103.4 ±3.3	**85.7** ±9.7	112.7 ±1.1	110.5 ±3.1	105.8 ±3.6
RORL	102.7 ±2.7	94.3 ±5.4	82.4 ±13.4	**111.5** ±2.1	92.8 ±3.3	72.7 ±15.7	117.4 ±1.3	107.3 ±1.9	86.9 ±4.7
LUC(ours)	**104.8** ±2.1	**102.9** ±3.1	**99.0** ±5.5	110.9 ±3.1	**106.6** ±6.3	83.3 ±9.1	**119.4** ±0.7	**114.5** ±0.9	**111.9** ±1.8

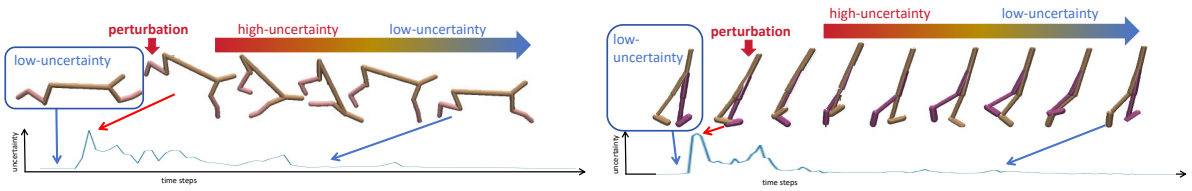

Fig. 4. Visualized results of LUC on OOD MuJoCo benchmarks - 'Halfcheetah' and 'Walker2d', with large scales of perturbations. The Bellman uncertainty (error) is estimated by the standard deviation uncertainty based on the learned Q-ensembles. The interval between every two frames is about five steps.

TABLE III
RESULTS OF CQL, IQL, SPOT, ATAC, SDC, OSR-1, SCAS AND **LUC(OURS)** ON OFFLINE **ANTMAZE** TASKS. THE RESULTS OF LUC ARE AVERAGED OVER 8 SEEDS. HIGHEST SCORES ARE BOLDED IN EACH TASK.

	CQL	IQL	SPOT	ATAC	SDC	OSR-10	SCAS	**LUC(Ours)**
u	82.6	87.5	93.5	70.6	89.0	89.9	90.4	**94.2±1.1**
u-d	10.2	62.2	40.7	54.3	57.3	**74.0**	63.8	66.3±5.7
m-p	59.0	71.2	74.7	72.3	71.9	66.0	76.6	**80.1±2.1**
m-d	46.6	70.0	79.1	68.7	78.7	80.0	**80.4**	78.5±3.7
l-p	16.4	39.6	35.3	38.5	37.2	37.9	**49.0**	45.1±7.4
l-d	3.2	47.5	36.3	43.1	33.2	37.9	**50.6**	48.6±7.6
avg.	36.3	63.0	59.9	57.9	61.2	64.3	68.4	**68.8**

aimed at evaluating the generalization ability of conservative/pessimistic methods in offline reinforcement learning when there may be some noise present in the environment to make the agent deviate from the in-distributional regions. To address this challenge, we compared the performance of the proposed LUC method with RORL [8] and OSR [5] methods. These two methods are also improvements upon traditional conservative methods, incorporating enhancements such as smoothness constraints (RORL) and recovery constraints (OSR) to enhance the generalization performance of models on unknown states. We utilize the models trained on the 'medium' datasets of the three benchmarks - 'Halfcheetah', 'Hopper' and 'Walker2d', and three kinds of OOD noises - 'random', 'action_diff' and 'min_Q' like in [8]. The basic difference between the constructed benchmarks in this section and those in Sec. VI-C lies in the (state/observation) spaces perturbed as shown in Figure 5.

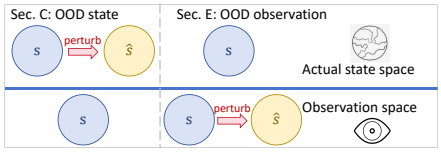

Fig. 5. Difference between OOD benchmarks in Sec.VI-C and in Sec.VI-E.

The results are shown in the Figure 6. We can observe that the proposed LUC method achieved good results on most benchmarks, particularly on the two types of adversarial attacks related to action differences. This may be because the learning approach in this paper has the weakest reliance on the behavioral policy, as the conservatism of LUC mainly stems from the evaluation of consequential reliability rather than a specific action distribution. In the context of min_Q, this tests the robustness of these methods in maintaining the optimality of the value function. In this type of attack, LUC also exhibited superior performance compared to other methods, indicating that LUC has better robustness in dealing with the OOD situations than other methods.

F. Evaluation on force-shifted halfcheetah benchmarks

To assess robustness against environmental variations during deployment, we constructed two challenging scenarios based on the Halfcheetah environment by modifying its physical parameters defined in the XML file. All models were trained offline on the standard 'medium-expert' dataset. **Scenario 1 (Mass)**: We modified the Z-component of the gravity vector from the standard (0, 0, -9.81) to simulate changes in effective weight or downward force. We tested Small (0, 0, -5.0), Medium (0, 0, -10.0), and Large (0, 0, -15.0) gravity settings during evaluation. **Scenario 2 (Friction)**: To introduce a different type of persistent perturbation, we added horizontal constant forces along the X and Y axes by setting the gravity vector to (1.0, 1.0, -9.81) (Small), (2.0, 2.0, -9.81) (Medium), and (3.0, 3.0, -9.81) (Large) during evaluation.

We compared LUC with RORL [8] and OSR [5] under these test conditions. Table VI shows the performance results, from which we observe that LUC's enhanced robustness compared to strong baselines like RORL and OSR. While extreme parameter shifts drastically impact all methods, LUC consistently performs best across most settings, particularly under moderate changes, highlighting the practical benefit of

TABLE IV
RESULTS OF CQL, PBRL AND **LUC(OURS)** ON OFFLINE ADROIT TASKS AVERAGED OVER 8 SEEDS. WE BOLD THE HIGHEST SCORES IN EACH TASK.

	pen-hu.	**pen-cl.**	**pen-ex.**	**hammer-hu.**	**hammer-cl.**	**hammer-ex.**	**door-hu.**	**door-cl.**	**door-ex.**	**avg.**
CQL	37.5	39.2	107.0	4.4	2.1	86.7	9.9	0.4	101.5	43.2
PBRL	35.4 ±3.3	**74.9** ±9.8	137.7 ±3.4	0.4 ±0.3	0.8 ±0.5	127.5 ±0.2	0.1 ±0.0	4.6 ±4.8	95.7 ±12.2	53.0
LUC(ours)	**41.7** ± 3.7	67.8 ±6.8	**143.2** ±2.3	**7.7** ±2.5	**4.7** ±2.6	**131.0** ±0.2	**7.9** ±3.2	**8.3** ±5.3	**103.7** ±9.4	**57.3**

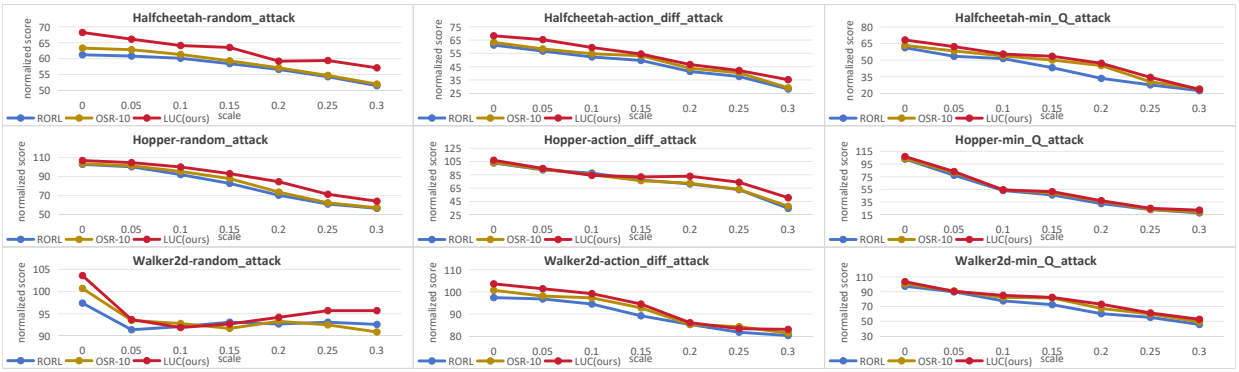

Fig. 6. Results of RORL, OSR and LUC on environments with OOD observations.

TABLE V
RESULTS OF **LUC(OURS)**, CQL, SDC, OSR-10 AND SCAS ON OFFLINE **FINRL** TASKS AVERAGED OVER 8 SEEDS. WE BOLD THE HIGHEST SCORES IN EACH TASK.

	CQL	SDC	OSR-10	SCAS	**LUC(Ours)**
L	411.3±15.0	389.7±43.2	421.4±31.4	424.3±26.5	**455.3**±22.8
M	563.8±118.0	577.1±223.8	544.7±214.5	**594.1**±73.4	589.7±127.1
H	450.8±114.8	461.4±119.2	433.4±142.3	487.1±81.7	**519.1**±106.3
avg.	475.3	476.1	466.5	501.8	**521.4**

its uncertainty-aware, Lyapunov-inspired control mechanism for reliable deployment in environments that may differ from training conditions.

TABLE VI
RESULTS OF **LUC (OURS)**, RORL AND OSR ON HALFCHEETAH-MASS AND HALFCHEETAH-FRICTION TASKS. WE BOLD THE HIGHEST SCORES IN EACH TASK.

	halfcheetah-mass			halfcheetah-friction		
	Small	**Medium**	**Large**	**Small**	**Medium**	**Large**
OSR-10	57.6±4.1	79.4±1.8	**34.3**±16.8	95.5±4.1	64.2±14.6	31.2±38.4
RORL	54.4±3.7	82.7±5.2	28.5±9.1	97.3±3.4	61.7±20.6	37.3±22.7
LUC	**60.6**±3.6	**96.8**±2.1	32.7±13.2	**103.7**±3.6	**67.3**±19.3	**48.9**±37.6

G. Ablation study / Sensitive analysis

In this section, we performed ablation experiments on the proposed LUC method to confirm its contribution to the overall framework. The findings, illustrated in Figure 7, demonstrate a substantial performance enhancement in the LUC module compared to LUC without reward shaping, particularly on certain non-expert datasets. This highlights the pivotal role of the LUC method in improving the performance of pessimistic offline RL approaches.

Furthermore, we perform the sensitive analysis for the two main hyperparameters in Eq.(10) and (11): β balances the impacts of immediate and long-term uncertainty; $\hat{\beta}$ controls the scale of the proposed LUC loss function. The results (on 'medium-expert' datasets) are shown in Figure 8, from which we could conclude the best selection of hyperparameter combinations for the three types of MuJoCo benchmarks, as is illustrated in Table VII.

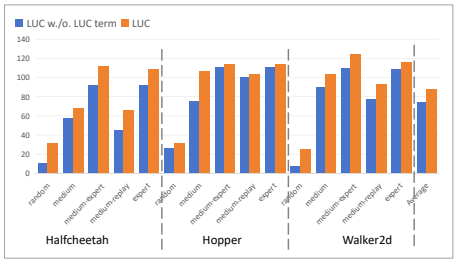

Fig. 7. Ablation study.

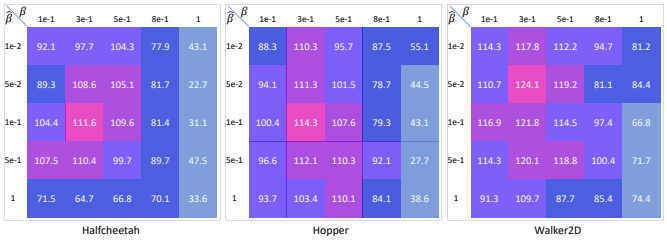

Fig. 8. Sensitive analysis for hyperparameters (β, $\hat{\beta}$) in Eq.(10) and (11).

More information on experiments, including network structures, device and more, can be found in Appendix C.

VII. DISCUSSION

A. Limitations

A core tenet of LUC is to improve the reliability and safety of online deployment by confining the agent's actions to

TABLE VII
HYPERPARAMETERS OF LUC IN STANDARD MUJOCO BENCHMARKS.

	Halfcheetah	**Hopper**	**Walker2d**
β	0.3	0.3	0.3
$\hat{\beta}$	0.1	0.1	0.05

low-uncertainty regions identified during offline training. The performance guarantees presented in this paper, particularly regarding optimality, are predicated on the assumption that the offline dataset sufficiently covers the state-action space visited by the optimal policy. This assumption is reflected theoretically in our performance bounds via the optimal coverage factor C^*, ($\sup_{s,a} \frac{\pi^*(a|s)}{\pi_\beta(a|s)} \leq C^*$), and empirically by LUC's superior results on high-quality datasets ('expert', 'medium-expert') versus low-quality ones ('random').

While this coverage requirement is less stringent than the standard Concentrability assumption [14], achieving it fully in practical offline settings is not always feasible. A key limitation arises when the optimal trajectory intersects significantly with high-uncertainty regions (i.e., areas poorly represented in the dataset). In these cases, LUC's inherent conservatism, designed to prevent unsafe OOD actions, will necessarily hinder the agent from exploring these potentially high-reward states. This embodies a critical trade-off often faced in offline RL: maximizing reliability based on known data versus pursuing potentially higher but riskier performance via exploration into uncertain regions. Therefore, when optimal coverage is lacking, LUC is expected to converge to a locally-optimal policy reliable within the data-supported, low-uncertainty domain. This policy ensures stable control but its performance ceiling is determined by the data's limitations rather than the true global optimum.

Besides, the results from these OOD experiments demonstrate that LUC, by leveraging its uncertainty estimation to identify and avoid unreliable regions, generally maintains better performance and stability compared to baselines when encountering these distributional shifts. This suggests that the low-uncertainty constraint helps maintain reliability during generalization to unseen (but related) high-dimensional states by guiding the agent away from areas where its learned model is likely to make poor predictions. However, we also acknowledge in this section that generalization to drastically different high-dimensional distributions or extremely high-dimensional inputs (e.g., raw pixels) remains a challenge and is an area for future investigation, potentially requiring more advanced uncertainty quantification techniques.

B. Future work

Several promising directions for future research include: 1) **Adaptive Constraint**: Developing methods to adaptively adjust the strength (or "effect-level") of the LUC constraint based on local environment characteristics, data density, or evolving task demands, potentially allowing for a better balance between safety and exploration where appropriate. 2) **Advanced Uncertainty Quantification**: Exploring the integration of LUC with more sophisticated uncertainty quantification (UQ) techniques beyond the current ensemble standard deviation. Methods like those based on estimating Bellman inconsistency [20], Bayesian approaches, or distributional RL metrics could potentially offer more accurate or scalable uncertainty estimates, especially in high-dimensional state spaces. 3) **Theoretical Analysis under Looser Assumptions**: Extending the theoretical analysis to establish performance guarantees or convergence properties for LUC under weaker assumptions, particularly relaxing the requirement for optimal dataset coverage, thereby broadening the scope of its formally guaranteed applicability.

VIII. CONCLUSION

This paper aims to identify a reliable operational region for the agent based on offline data. To achieve this, we introduce the Lyapunov Uncertainty Control (LUC) algorithm in an offline, model-free manner. Theoretically, in deterministic MDPs or when the dataset fully covers all dynamic modes, LUC can confine the agent's operations within low-uncertainty areas, thereby enhancing decision-making reliability. Empirically, LUC-trained agents demonstrate superior robustness and reliability in high-risk scenarios compared to various other methods. In future works, LUC can serve as a versatile tool in diverse offline reinforcement learning frameworks, including model-based approaches, potentially paving the way for new research opportunities.

ACKNOWLEDGMENTS

This work is partially supported by National Natural Science Foundation of China (62476128).

APPENDIX

A. Proofs of main theorems in Sec.V-C

Proposition 2. Suppose the action distribution of new policy $\pi(a|s)$ is positive correlated to the learnt Q function $f(s,a)$, i.e., $\pi(a|s) \propto f(s,a)$. Then the proposed Lyapunov value estimation induces a Lyapunov policy as in Definition 3.

Proof of Proposition 2. Denote the empirical behavior of the dataset $\mathcal{D}$ as π_β, whose actions are always supported by the dataset. The Lyapunov value estimation implicitly penalize the new policy π at an given state s with two aspects:

1) $\min_\pi \mathbb{E}_{a\sim\pi(a|s)}\zeta_f(s,a)$; This constrains the new policy would not generated OOD actions beyond the demonstration of the offline data, i.e., $supp(\pi(a|s)) \subset supp(\pi_\beta(a|s))$. In previous works [17], [18], such supported constraint is often achieved in a data density based way as $\min_\pi \sum_{a\notin\pi_\beta} \pi(a|s)$. Then we will show that the current step's error controlling minimizes the upper bound of the above supported constraint,

$$\min_\pi \sum_{a\notin\pi_\beta} \pi(a|s) \Leftrightarrow \max_\pi \mathbb{E}_{a\in\pi(a|s)} d(s,a) \tag{15}$$

$$\leq^{(a)} \max_\pi C \cdot \mathbb{E}_{a\in\pi(a|s)} \frac{1}{\zeta_f^2(s,a)} \tag{16}$$

$$\Leftrightarrow \min_\pi \mathbb{E}_{a\in\pi(a|s)} \zeta_f^2(s,a) \tag{17}$$

$$\Leftrightarrow^{(b)} \min_\pi \mathbb{E}_{a\in\pi(a|s)} \zeta_f(s,a) \tag{18}$$

The inequality (a) holds because of the Lemma 1 in Appendix B. The equivalence (b) holds because the Bellman uncertainty is always non-negative.

Then with this constraint, we can have the new policy would reject the actions that is not supported by the behavior policy. This means, the new policy would only select the data-supported actions.

2) $\forall s, a \in \mathcal{D}$, we have $\zeta_f(M(s,a),\pi) \leq \zeta_f(s,a)$. This could also be consider as,

$$\min_{a\in\pi_\beta} \zeta_f(s,a) - \zeta_f(M(s,a),\pi) \tag{19}$$

$$\leq^{(a)} \min_{a\in\pi} \zeta_f(s,a) - \zeta_f(M(s,a),\pi) \tag{20}$$

$$\leq \zeta_f(s,a_m) - \zeta_f(M(s,a_m),\pi) \tag{21}$$

where $a_m = \arg\max_{a_m\in\pi} \zeta_f(M(s,a_m),\pi)$. The (a) holds because of the assumption that the condition 1) is perfectly confirmed, i.e., $supp(\pi(a|s)) \subset supp(\pi_\beta(a|s))$.

Then we have $\forall s \in \mathcal{D}, \max_{\hat{a}\in\pi} \zeta_f(M(s,\hat{a}),\pi) \leq \zeta_f(s,\pi)$. And we can conclude that the Lyapunov value estimation would help to induce a Lyapunov policy.

B. Proofs of main theorems in Sec.V-D

First we define the policy candidate set Π based on a version space of all the functions $f \in \mathcal{F}$, where we have $\forall\pi \in \Pi, \exists f \in \mathcal{F}, \pi(a|s) \propto f(s,a)$, and $\forall f \in \mathcal{F}, \exists\pi \in \Pi, \pi$ is the greedy policy according to f. Then we define a corresponding Bellman operator $\mathcal{T}^\Pi f(s,a) = r + \gamma\mathbb{E}_{s'\sim P}\max_{\pi\in\Pi} f(s',\pi)$ and its empirical version is $\hat{\mathcal{T}}^\Pi f(s,a) = r + \gamma\mathbb{E}_{s'\sim\hat{P}}\max_{\pi\in\Pi} f(s',\pi)$. Then before the introduction of theoretical results, a basic assumption should be made.

Assumption 1:

(Optimal coverage.) [33] We assume the dataset have sufficient coverage over the optimal policy's visitation, i.e., $\sup_{s,a}\frac{\pi^*(a|s)}{\pi_\beta(a|s)} \leq C^*$, and $\pi^* \in \Pi$.

Similar assumptions has been utilized in theoretical analysis for offline RL [33]. Compared with the more common assumption - Concentrability assumption [13], [34] that the dataset should fully cover the whole state space, Assumption 1 is much looser and more feasible in practice.

Definition 4: **(Recoverability)** Define the recoverability of a given policy π from the given (s_0,a_0) pair,

$$R(\pi)\big|_{s_0,a_0} = \inf_{T\geq 0} \mathbb{E}_{s_T,a_T\sim P(s_T,a_T|\pi,s_0,a_0)} \frac{d(s_T,a_T)}{d(s_0,a_0)} \tag{22}$$

where $d(s,a)$ is the data density at (s,a).

Definition 5: **(Recoverability risk)** We define the most risk of a policy π that the agent is able to recover to the regions with low Bellman uncertainty, i.e., its familiar regions, from the given (s_0,a_0) pair,

$$Risk(\pi)\big|_{s_0,a_0} = \sup_{T\geq 0} \mathbb{E}_{P(s_T,a_T|\pi,s_0,a_0)} \frac{\|\mathcal{T}^\pi f - \hat{\mathcal{T}}^\pi f\|(s_T,a_T)}{\|\mathcal{T}^\pi f - \hat{\mathcal{T}}^\pi f\|(s_0,a_0)}$$
$$= \sup_{T\geq 0} \mathbb{E}_{s_T,a_T} \frac{\zeta_T}{\zeta_0} \tag{23}$$

where $\mathcal{T}^\pi$ is the true Bellman operator and $\hat{\mathcal{T}}^\pi$ is the empirical Bellman.

Lemma 1: Given an MDP with max reward R_{max} and a dataset of size N. The dimension of state space is $|S|$ and that of action space is $|A|$. Given (s,a) pair, we denote its data density over the dataset is $d(s,a)$. Then with probability $1-\delta$, we have,

$$d(s,a) \leq \gamma^2 \cdot R_{max}^2 \frac{2}{N\cdot\|\mathcal{T}^\pi f - \hat{\mathcal{T}}^\pi f\|^2(s,a)} \log(\frac{|S||A|\cdot 2^{|S|}}{\delta})$$

where $\|\mathcal{T}^\pi f - \hat{\mathcal{T}}^\pi f\|(s,a)$ is Bellman uncertainty.

Proof of Lemma 1.

$$\|\mathcal{T}^\pi f - \hat{\mathcal{T}}^\pi f\|(s,a)$$
$$= \gamma\|\sum_{s'}(\hat{P}(s'|s,a) - P(s'|s,a))\cdot f(s',\pi)\| \tag{24}$$
$$\leq \gamma\cdot R_{max}\cdot\|\hat{P}(s'|s,a) - P(s'|s,a)\|_1 \tag{25}$$
$$\leq^{(a)} \gamma\cdot R_{max}\cdot\sqrt{\frac{2}{N(s,a)}\log(\frac{|S||A|\cdot 2^{|S|}}{\delta})} \tag{26}$$
$$\Rightarrow d(s,a)$$
$$\leq \gamma^2\cdot R_{max}^2\frac{2}{N\cdot\|\mathcal{T}^\pi f - \hat{\mathcal{T}}^\pi f\|^2(s,a)}\log(\frac{|S||A|\cdot 2^{|S|}}{\delta})$$

The inequality (a) holds because of the ***Proposition 9*** *in [35]. And $N(s,a)$ is the number of (s,a) samples in the dataset, so the density $d(s,a) = \frac{N(s,a)}{N}$. Completing the proof.*

Lemma 2: For any policy π and (s_0,a_0) pair, we have,

$$Risk(\pi)\big|_{s_0,a_0} = \sup_{T\geq 0}\mathbb{E}_\pi\left[\frac{\sum_{t=0}^{T-1}[\gamma^{t+1}\zeta_{t+1} - \gamma^t\zeta_t] + \zeta_0}{\zeta_0\cdot\gamma^T}\big|s_0,a_0\right]$$

Proof of Lemma 2.

$$Risk(\pi)\big|_{s_0,a_0} = \sup_{T\geq 0}\mathbb{E}_\pi\left[\frac{\zeta_T}{\zeta_0}\big|s_0,a_0\right] \tag{27}$$
$$= \sup_{T\geq 0}\mathbb{E}_\pi\left[\frac{\gamma^T\zeta_T - \gamma^{T-1}\zeta_{T-1} + ... + \gamma\zeta_1 - \zeta_0 + \zeta_0}{\zeta_0\cdot\gamma^T}\big|s_0,a_0\right]$$
$$= \sup_{T\geq 0}\mathbb{E}_\pi\left[\frac{\sum_{t=0}^{T-1}[\gamma^{t+1}\zeta_{t+1} - \gamma^t\zeta_t] + \zeta_0}{\zeta_0\cdot\gamma^T}\big|s_0,a_0\right] \tag{28}$$

Completing the proof.

Lemma 3: Given an arbitrary Bellman operator (maybe empirical Bellman operator) $\mathcal{T}$ and an arbitrary policy candidate set Π. We have $\mathcal{T}f(s,a) = r + \gamma\mathbb{E}_{s'}\max_{\pi\in\Pi} f(s',\pi)$. Then for any value function $f_1, f_2 \in \mathcal{F}$ and $t \geq 0$, we have,

$$\|\mathcal{T}^{(t)}f_1 - \mathcal{T}^{(t)}f_2\|_d \leq \sup_{s_0,a_0,d(s_0,a_0)>0}\frac{\gamma^t}{R(\hat{\pi})\big|_{s_0,a_0}}\|f_1 - f_2\|_d$$

We denote the greedy policy induced by f_1 as π_1 and the greedy policy induced by f_2 as π_2. Then $\hat{\pi}$ is the pessimistic policy of f_1 and f_2, i.e., $\hat{\pi}(a|s) = \pi_1(a|s)$ if $f_1(s,\pi_1) \leq f_2(s,\pi_2)$ and $\hat{\pi}(a|s) = \pi_2(a|s)$ if $f_2(s,\pi_2) \leq f_1(s,\pi_1)$. And $\|x\|_d = \sum_x d(x)|x|$ is a distributional weighted norm, where d here is the density of the dataset.

Proof of Lemma 3. First we denote $\pi_1(a|s) = \arg\max_{\pi\in\Pi} f_1(s,\pi)$ and $\pi_2(a|s) = \arg\max_{\pi\in\Pi} f_2(s,\pi)$. Then,

$$(\mathcal{T}f_1 - \mathcal{T}f_2)(s,a) = \gamma\mathbb{E}_{P(s'|s,a)}[f_1(s',\pi_1) - f_2(s',\pi_2)]$$
$$\leq \gamma\mathbb{E}_{P(s'|s,a)}[f_1(s',\pi_1) - f_2(s',\pi_1)]$$

On the other hand,

$$(\mathcal{T}f_1 - \mathcal{T}f_2)(s,a) \geq \gamma \mathbb{E}_{P(s'|s,a)}[f_1(s',\pi_2) - f_2(s',\pi_2)]$$

Then we construct $\hat{\pi}(a|s) = \pi_1(a|s)$ if $f_1(s,\pi_1) \leq f_2(s,\pi_2)$ and $\hat{\pi}(a|s) = \pi_2(a|s)$ if $f_2(s,\pi_2) \leq f_1(s,\pi_1)$. So we have,

$$|\mathcal{T}f_1 - \mathcal{T}f_2|(s,a) \leq \gamma |\mathbb{E}_{P(s'|s,a)}[f_1(s',\hat{\pi}) - f_2(s',\hat{\pi})]|$$

Then if we recursively apply the $\hat{\pi}$, we would have,

$$\begin{aligned} &|\mathcal{T}^{(t)}f_1 - \mathcal{T}^{(t)}f_2|(s,a) \\ \leq &\gamma^t \cdot |\mathbb{E}_{P(s_t,a_t|\hat{\pi},s_0=s,a_0=a)}[f_1(s_t,a_t) - f_2(s_t,a_t)]| \end{aligned} \tag{29}$$

Then we aim to bound the $\|\mathcal{T}^{(t)}f_1 - \mathcal{T}^{(t)}f_2\|_d$,

$$\|\mathcal{T}^{(t)}f_1 - \mathcal{T}^{(t)}f_2\|_d \tag{30}$$

$$= \sum_{s_0,a_0} |\mathcal{T}^{(t)}f_1 - \mathcal{T}^{(t)}f_2|(s_0,a_0)d(s_0,a_0) \tag{31}$$

$$\leq \gamma^t \cdot \sum_{s_0,a_0}\sum_{s_t,a_t} d(s_0,a_0)P(s_t,a_t|\hat{\pi},s_0,a_0) \|f_1(s_t,a_t) - f_2(s_t,a_t)\| \tag{32}$$

$$\leq \sup_{s_0,a_0,d(s_0,a_0)>0} \frac{\gamma^t}{R(\hat{\pi})\big|_{s_0,a_0}} \cdot \sum_{s_t,a_t} d(s_t,a_t) \|f_1(s_t,a_t) - f_2(s_t,a_t)\| \tag{33}$$

$$= \sup_{s_0,a_0,d(s_0,a_0)>0} \frac{\gamma^t}{R(\hat{\pi})\big|_{s_0,a_0}} \cdot \|[f_1(s_t,a_t) - f_2(s_t,a_t)]\|_d$$

Completing the proof.

Lemma 4: Denote the learnt value function as f_k, with k iterations of $f_k = \mathcal{T}f_{k-1}$, and the fixed point of $\mathcal{T}$ is f^*. Then,

$$\|f_k - f^*\|_d \leq R(\pi_0)\cdot\gamma^k \cdot \|\triangle_{(0)}\|_d + \epsilon_{max}\cdot\sum_{t=1}^{k}[R(\hat{\pi}_{t-1})\cdot\gamma^t] + \epsilon_{max} \tag{34}$$

where $R(\pi_k) = \sum_{s_0,a_0} \frac{1}{R(\pi_k)\big|_{s_0,a_0}}$, $\epsilon_{max} = \max_{t\leq k-1}\|f_{t+1} - \mathcal{T}f_t\|_d$ and $\triangle_{(0)} = \|f_0 - f^*\|_\infty$.

Proof of Lemma 4.

$$\|f_k - f^*\|_d \leq \|\mathcal{T}f_{k-1} - f^*\|_d + \|f_k - \mathcal{T}f_{k-1}\|_d \tag{35}$$

$$\leq \|\mathcal{T}^{(2)}f_{k-2} - f^*\|_d + R(\pi_{k-1})\cdot\gamma\cdot\epsilon_{max} + \epsilon_{max}$$

$$\cdots\cdots\cdots\cdots \tag{36}$$

$$\leq \|\mathcal{T}^{(k)}f_0 - f^*\|_d + \epsilon_{max}\cdot\sum_{t=1}^{k}[R(\hat{\pi}_{t-1})\cdot\gamma^t] + \epsilon_{max}$$

$$\leq^{(a)} R(\pi_0)\cdot\gamma^k\cdot\|\triangle_{(0)}\|_d + \epsilon_{max}\cdot\sum_{t=1}^{k}[R(\hat{\pi}_{t-1})\cdot\gamma^t]\epsilon_{max}$$

The inequality (a) holds because of Lemma 3. Completing the proof.

Lemma 5: Given an MDP with max reward R_{max} and a dataset of size N. The dimension of state space is $|S|$ and that of action space is $|A|$. Given (s,a) pair, we denote its data density over the dataset is $d(s,a)$. Given an empirical Bellman operator $\hat{\mathcal{T}}$ and an arbitrary policy candidate set Π, where $\hat{\mathcal{T}}f(s,a) = r(s,a) + \gamma\mathbb{E}_{s'\sim\hat{P}(s'|s,a)}\max_{\pi\in\Pi} f(s',\pi)$. Denote the learnt value function as f_k, with k iterations of $\hat{f}_k = \hat{\mathcal{T}}\hat{f}_{k-1}$, and the fixed point of $\hat{\mathcal{T}}$ is $\hat{f}^*$. Then we have,

$$\|\hat{f}_k - \hat{f}^*\|_d \leq \mathcal{O}\Big(\sup_{s_0,a_0,d(s_0,a_0)>0}\sup_{\pi\in\Pi,T\geq 0} \mathbb{E}_{s_T,a_T\sim P(s_T,a_T|\pi,s_0,a_0)}(\sum_{t=0}^{T-1}\gamma^{t+1}\zeta_{t+1} - \gamma^t\zeta_t)^2\Big)$$

where ζ_t is the Bellman uncertainty at time step t, i.e., $\zeta_t = \|\mathcal{T}^\pi f - \hat{\mathcal{T}}^\pi f\|(s_t,a_t)$.

Proof of Lemma 5. From Lemma 4 we have known that if we want to bound $\|\hat{f}_k - \hat{f}^*\|_d$, we should bound $\frac{1}{R(\pi)\big|_{s_0,a_0}}$ at each time steps.

$$\frac{1}{R(\pi)\big|_{s_0,a_0}} = \sup_{T\geq 0}\mathbb{E}_{s_T,a_T\sim P(s_T,a_T|\pi,s_0,a_0)}\frac{d(s_0,a_0)}{d(s_T,a_T)} \tag{37}$$

With Lemma 1, we have,

$$\frac{1}{R(\pi)\big|_{s_0,a_0}} = \sup_{T\geq 0}\mathbb{E}_{s_T,a_T\sim P(s_T,a_T|\pi,s_0,a_0)}\frac{d(s_0,a_0)}{d(s_T,a_T)} \tag{38}$$

$$\leq^{(a)} \sup_{T\geq 0}\Big(\mathbb{E}_{P(s_T,a_T|\pi,s_0,a_0)}\frac{\|\mathcal{T}^\pi f - \hat{\mathcal{T}}^\pi f\|(s_T,a_T)}{\|\mathcal{T}^\pi f - \hat{\mathcal{T}}^\pi f\|(s_0,a_0)}\Big)^2$$

The inequality (a) holds because of $d(s_0,a_0)\cdot\zeta_0 \leq \sup_T \mathbb{E}_{s_T,a_T}d(s_T,a_T)\cdot\zeta_T$. Then following Lemma 2, we have,

$$\sup_{T\geq 0}\mathbb{E}_{s_T,a_T\sim P(s_T,a_T|\pi,s_0,a_0)}\frac{\|\mathcal{T}^\pi f - \hat{\mathcal{T}}^\pi f\|^2(s_T,a_T)}{\|\mathcal{T}^\pi f - \hat{\mathcal{T}}^\pi f\|^2(s_0,a_0)}$$

$$= \sup_{T\geq 0}\mathbb{E}_{s_T,a_T\sim P(s_T,a_T|\pi,s_0,a_0)}\Big(\frac{\sum_{t=0}^{T-1}[\gamma^{t+1}\zeta_{t+1} - \gamma^t\zeta_t]}{\zeta_0\cdot\gamma^T} + \frac{1}{\gamma^T}\Big)^2$$

$$\leq \mathcal{O}\left(\sup_{\pi\in\Pi,T\geq 0}\mathbb{E}_{s_T,a_T\sim P(s_T,a_T|\pi,s_0,a_0)}(\sum_{t=0}^{T-1}\gamma^{t+1}\zeta_{t+1} - \gamma^t\zeta_t)^2\right)$$

Completing the proof.

Please note that Lemma 5 holds for any estimation value function f. And we can utilize the learned $\hat{f}_k$ at the k^{th} iteration.

Then we give the proof for Theorem 2 in the main text.

Proof of Theorem 2.

$$\|\hat{f}_k - f^*\|_d \leq \|\hat{f}_k - \hat{f}^*\|_d + \|\hat{f}^* - f^*\|_d \tag{39}$$

where $\hat{f}^*$ is the fixed point of $\hat{\mathcal{T}}^\Pi$. Then Lemma 5 bounds $\|\hat{f}_k - \hat{f}^*\|_d$, i.e.,

$$\|\hat{f}_k - \hat{f}^*\|_d \leq \mathcal{O}\Big(\sup_{\substack{\pi\in\Pi,T\geq 0\\ s_0,a_0,d(s_0,a_0)>0}} \mathbb{E}_{s_T,a_T\sim P(s_T,a_T|\pi,s_0,a_0)}(\sum_{t=0}^{T-1}\gamma^{t+1}\zeta_{t+1} - \gamma^t\zeta_t)^2\Big)$$

On the other hand,

$$\|\hat{f}^* - f^*\|_d \leq \|\hat{\mathcal{T}}^\Pi\hat{f}^* - \hat{\mathcal{T}}^\Pi f^*\|_d + \|\hat{\mathcal{T}}^\Pi f^* - \mathcal{T}^\Pi f^*\|_d \tag{40}$$

$$\leq \|\hat{\mathcal{T}}^\Pi f^* - \mathcal{T}^\Pi f^*\|_d + \gamma\|\hat{f}^* - f^*\|_d \tag{41}$$

$$\Rightarrow \|\hat{f}^* - f^*\|_d \leq \frac{\|\hat{\mathcal{T}}^\Pi f^* - \mathcal{T}^\Pi f^*\|_d}{1-\gamma} \tag{42}$$

Then due to the optimal coverage assumption that $\sup_{s,a} \frac{\pi^*(a|s)}{\pi_\beta(a|s)} \leq C^*$, and $\pi^* \in \Pi$, we have,

$$\|\hat{\mathcal{T}}^{\Pi} f^* - \mathcal{T}^{\Pi} f^*\|_d \tag{43}$$
$$= \|\sum_{s'}(P(s'|s,a) - \hat{P}(s'|s,a)) \max_{\pi \in \Pi} f^*(s',\pi)\|_d \tag{44}$$
$$= \|\sum_{s'}(P(s'|s,a) - \hat{P}(s'|s,a)) f^*(s',\pi^*)\|_d \tag{45}$$
$$\leq C^* \sup_{\pi \in \Pi} \sum_{s_0,a_0} d(s_0,a_0) \mathbb{E}_{a\sim\pi(a|s_0)} \|\hat{\mathcal{T}}^{\Pi} f^* - \mathcal{T} f^*\|(s_0,a)$$
$$\leq C^* \sup_{\pi \in \Pi} \sum_{s_0,a_0} d(s_0,a_0) \mathbb{E}_{a\sim\pi(a|s_0)} \|\hat{\mathcal{T}}^{\Pi} \hat{f}_k - \mathcal{T} \hat{f}_k\|(s_0,a)$$

The last inequality holds because the assumption that the optimal policy is reliable, so its value function would have an ideally low uncertainty.

Therefore, we complete the proof.

Here we give the proof of Proposition 3 in the main text.

Proof of Proposition 3. $\min_\pi \mathbb{E}_{a\sim\pi(a|s)} \zeta_f(s,a)$; This constrains the new policy would not generated OOD actions beyond the demonstration of the offline data, i.e., $supp(\pi(a|s)) \subset supp(\pi_\beta(a|s))$. In previous works [17], [18], such supported constraint is often achieved in a data density based way as $\min_\pi \sum_{a \notin \pi_\beta} \pi(a|s)$. Then we will show that the current step's error controlling minimizes the upper bound of the above supported constraint,

$$\min_\pi \sum_{a \notin \pi_\beta} \pi(a|s) \Leftrightarrow \max_\pi \mathbb{E}_{a\in\pi(a|s)} d(s,a) \tag{46}$$
$$\leq^{(a)} \max_\pi C \cdot \mathbb{E}_{a\in\pi(a|s)} \frac{1}{\zeta_f^2(s,a)} \tag{47}$$
$$\Leftrightarrow \min_\pi \mathbb{E}_{a\in\pi(a|s)} \zeta_f^2(s,a) \tag{48}$$
$$\Leftrightarrow^{(b)} \min_\pi \mathbb{E}_{a\in\pi(a|s)} \zeta_f(s,a) \tag{49}$$

The inequality (a) holds because of the Lemma 1 in Appendix B. The equivalence (b) holds because the Bellman uncertainty is always non-negative.

Then with this constraint, we can have the new policy would reject the actions that is not supported by the behavior policy. This means, the new policy would only select the data-supported actions. Then,

$$\min_\pi \mathbb{E}_{P(\tau_T|\pi,s_0=s,a_0=a)} (\sum_{t=0}^{T-1} [\gamma^{t+1} \zeta_{\hat{f}_k,t+1} - \gamma^t \zeta_{\hat{f}_k,t}]$$
$$\leq^{(a)} \min_\pi \frac{1-\gamma^T}{1-\gamma} \max_{s_t \in P(s_t|\pi,s_0=s)} (\gamma \mathbb{E}_{P(s_{t+1}|s_t,\pi)} \zeta_{\hat{f}_k}(s_{t+1},\pi) - \zeta_{\hat{f}_k}(s_t,\pi)) \tag{50}$$
$$\leq^{(b)} \min_\pi \frac{1-\gamma^T}{1-\gamma} \max_{s_t \in P(s_t|\pi_\beta,s_0=s)} (\gamma \mathbb{E}_{P(s_{t+1}|s_t,\pi)} \zeta_{\hat{f}_k}(s_{t+1},\pi) - \zeta_{\hat{f}_k}(s_t,\pi)) \tag{51}$$
$$\leq^{(c)} \min_\pi \frac{1-\gamma^T}{1-\gamma} \max_{s\in\mathcal{D}} (\gamma \mathbb{E}_{P(s'|s,\pi)} \zeta_{\hat{f}_k}(s',\pi) - \zeta_{\hat{f}_k}(s,\pi))$$

The inequality (a) is obtained using the formula for the sum of a geometric series. The inequality (b) is due to $\pi \subseteq \pi_\beta$, where π_β is the behavior policy. Finally, inequality (c) holds because $P(s_t|\pi_\beta, s_0 = s) \subseteq \mathcal{D}$. Then we have, $\forall s \in \mathcal{D}$,

$$\min_\pi (\gamma \mathbb{E}_{P(s'|s,\pi)} \zeta_{\hat{f}_k}(s',\pi) - \zeta_{\hat{f}_k}(s,\pi)) \tag{52}$$
$$\Rightarrow \min_\pi \mathbb{E}_{P(\tau_T|\pi,s_0=s)} (\sum_{t=0}^{T-1} [\gamma^{t+1} \zeta_{\hat{f}_k,t+1} - \gamma^t \zeta_{\hat{f}_k,t}] \tag{53}$$

Then we would bound the left part through the Lyapunov value estimation. Due to $\pi(a|s) \propto \hat{f}_k(s,a)$, then the penalization to the s,a pairs is equivalent to minimize the preference of π to the action a at state s. In this way, the Lyapunov Uncertainty-penalization could be converted to,

$$\min_\pi (\gamma \max_{s' \in \hat{P}(s'|s,\pi)} \zeta_{\hat{f}_k}(s',\pi) - \zeta_{\hat{f}_k}(s,\pi)) \tag{54}$$
$$\geq \min_\pi (\gamma \max_{s' \in P(s'|s,\pi)} \zeta_{\hat{f}_k}(s',\pi) - \zeta_{\hat{f}_k}(s,\pi)) \tag{55}$$
$$\geq \min_\pi (\gamma \mathbb{E}_{P(s'|s,\pi)} \zeta_{\hat{f}_k}(s',\pi) - \zeta_{\hat{f}_k}(s,\pi)) \tag{56}$$

The first inequality holds because of $P(s'|s,a) \subseteq \hat{P}(s'|s,a)$. Completing the proof.

C. Additional experimental details

1) Neural network structures of LUC: In this section, we introduce the structure[2] of the networks we use in this paper: policy network and Q network. The structure of the policy network and Q networks is as shown in Table VIII, where 's_dim' is the dimension of states and 'a_dim' is the dimension of actions. 'h_dim' is the dimension of the hidden layers, which is usually 256 in our experiments. The policy network is a Guassian policy and the Q networks includes ten Q function networks and ten target Q function networks.

TABLE VIII
THE STRUCTURE OF THE POLICY NET AND THE Q NETWORKS.

policy net	Q net
Linear(s_dim, 256)	Linear(s_dim, h_dim)
Relu()	Relu()
Linear(h_dim, h_dim)	Linear(h_dim, h_dim)
Relu()	Relu()
Linear(h_dim, a_dim)	Linear(h_dim, 1)

2) Compute resources: We conducted all our experiments using a server equipped with one Intel Xeon Gold 5218 CPU, with 32 cores and 64 threads, and 256GB of DDR4 memory. We used a NVIDIA RTX3090 GPU with 24GB of memory for our deep learning experiments. All computations were performed using Python 3.8 and the PyTorch deep learning framework.

REFERENCES

[1] S. Tang and J. Wiens, "Model selection for offline reinforcement learning: Practical considerations for healthcare settings," in *Machine Learning for Healthcare Conference*. PMLR, 2021, pp. 2–35.

[2]We constructed LUC based on the RORL project: https://github.com/YangRui2015/RORL

[2] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. A. Sallab, S. K. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 4909–4926, 2022. [Online]. Available: https://doi.org/10.1109/TITS.2021.3054625

[3] A. Lobbezoo, Y. Qian, and H. Kwon, "Reinforcement learning for pick and place operations in robotics: A survey," *Robotics*, vol. 10, no. 3, p. 105, 2021. [Online]. Available: https://doi.org/10.3390/robotics10030105

[4] K. Kang, P. Gradu, J. J. Choi, M. Janner, C. Tomlin, and S. Levine, "Lyapunov density models: Constraining distribution shift in learning-based control," in *International Conference on Machine Learning*. PMLR, 2022, pp. 10 708–10 733.

[5] K. Jiang, J.-Y. Yao, and X. Tan, "Recovering from out-of-sample states via inverse dynamics in offline reinforcement learning," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[6] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Y. Zou, S. Levine, C. Finn, and T. Ma, "MOPO: model-based offline policy optimization," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/a322852ce0df73e204b7e67cbbef0d0a-Abstract.html

[7] C. Bai, L. Wang, Z. Yang, Z. Deng, A. Garg, P. Liu, and Z. Wang, "Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning," in *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. [Online]. Available: https://openreview.net/forum?id=Y4cs1Z3HnqL

[8] R. Yang, C. Bai, X. Ma, Z. Wang, C. Zhang, and L. Han, "RORL: robust offline reinforcement learning via conservative smoothing," *CoRR*, vol. abs/2206.02829, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2206.02829

[9] K. Panaganti, Z. Xu, D. Kalathil, and M. Ghavamzadeh, "Robust reinforcement learning using offline data," *Advances in neural information processing systems*, vol. 35, pp. 32 211–32 224, 2022.

[10] L. Shi and Y. Chi, "Distributionally robust model-based offline reinforcement learning with near-optimal sample complexity," *Journal of Machine Learning Research*, vol. 25, no. 200, pp. 1–91, 2024.

[11] E. C. Kerrigan, *Robust constraint satisfaction: Invariant sets and predictive control*. University of London, 2000.

[12] C. Richter and N. Roy, "Safe visual navigation via deep learning and novelty detection," *Robotics: Science and Systems Foundation*, 2017.

[13] A. Kumar, J. Fu, M. Soh, G. Tucker, and S. Levine, "Stabilizing off-policy q-learning via bootstrapping error reduction," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 11 761–11 771. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/c2073ffa77b5357a498057413bb09d3a-Abstract.html

[14] Y. Wu, G. Tucker, and O. Nachum, "Behavior regularized offline reinforcement learning," *CoRR*, vol. abs/1911.11361, 2019. [Online]. Available: http://arxiv.org/abs/1911.11361

[15] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/0d2b2061826a5df3221116a5085a6052-Abstract.html

[16] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," in *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. [Online]. Available: https://openreview.net/forum?id=68n2s9ZJWF8

[17] J. Wu, H. Wu, Z. Qiu, J. Wang, and M. Long, "Supported policy optimization for offline reinforcement learning," in *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., 2022. [Online]. Available: http://papers.nips.cc/paper_files/paper/2022/hash/caa934a507a952698d54efb24845fc4b-Abstract-Conference.html

[18] Y. Mao, H. Zhang, C. Chen, Y. Xu, and X. Ji, "Supported value regularization for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[19] G. An, S. Moon, J. Kim, and H. O. Song, "Uncertainty-based offline reinforcement learning with diversified q-ensemble," in *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021, pp. 7436–7447. [Online]. Available: https://proceedings.neurips.cc/paper/2021/hash/3d3d286a8d153a4a58156d0e02d8570c-Abstract.html

[20] Y. Sun, J. Zhang, C. Jia, H. Lin, J. Ye, and Y. Yu, "Model-bellman inconsistency for model-based offline reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2023, pp. 33 177–33 194.

[21] K. Wu, Y. Zhao, Z. Xu, Z. Che, C. Yin, C. H. Liu, Q. Qiu, F. Feng, and J. Tang, "Acl-ql: Adaptive conservative level in q-learning for offline reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[22] H. Zhang, J. Shao, Y. Jiang, S. He, G. Zhang, and X. Ji, "State deviation correction for offline reinforcement learning," in *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022*. AAAI Press, 2022, pp. 9022–9030. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/20886

[23] Y. Mao, Q. Wang, C. Chen, Y. Qu, and X. Ji, "Offline reinforcement learning with ood state correction and ood action suppression," *arXiv preprint arXiv:2410.19400*, 2024.

[24] C. J. C. H. Watkins and P. Dayan, "Technical note q-learning," *Mach. Learn.*, vol. 8, pp. 279–292, 1992. [Online]. Available: https://doi.org/10.1007/BF00992698

[25] Y. Jin, Z. Yang, and Z. Wang, "Is pessimism provably efficient for offline rl?" in *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 2021, pp. 5084–5096. [Online]. Available: http://proceedings.mlr.press/v139/jin21e.html

[26] M. Laskey, J. Lee, R. Fox, A. D. Dragan, and K. Goldberg, "DART: noise injection for robust imitation learning," in *1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings*, ser. Proceedings of Machine Learning Research, vol. 78. PMLR, 2017, pp. 143–156. [Online]. Available: http://proceedings.mlr.press/v78/laskey17a.html

[27] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4RL: datasets for deep data-driven reinforcement learning," *CoRR*, vol. abs/2004.07219, 2020. [Online]. Available: https://arxiv.org/abs/2004.07219

[28] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[29] C. Bai, T. Xiao, Z. Zhu, L. Wang, F. Zhou, A. Garg, B. He, P. Liu, and Z. Wang, "Monotonic quantile network for worst-case offline reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[30] C.-A. Cheng, T. Xie, N. Jiang, and A. Agarwal, "Adversarially trained actor critic for offline reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 3852–3878.

[31] X.-Y. Liu, H. Yang, Q. Chen, R. Zhang, L. Yang, B. Xiao, and C. D. Wang, "Finrl: A deep reinforcement learning library for automated stock trading in quantitative finance," *arXiv preprint arXiv:2011.09607*, 2020.

[32] R.-J. Qin, X. Zhang, S. Gao, X.-H. Chen, Z. Li, W. Zhang, and Y. Yu, "Neorl: A near real-world benchmark for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 753–24 765, 2022.

[33] T. Xie, C.-A. Cheng, N. Jiang, P. Mineiro, and A. Agarwal, "Bellman-consistent pessimism for offline reinforcement learning," *Advances in neural information processing systems*, vol. 34, pp. 6683–6694, 2021.

[34] R. Munos, "Error bounds for approximate value iteration," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 20, no. 2. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005, p. 1006.

[35] M. Ghavamzadeh, M. Petrik, and Y. Chow, "Safe policy improvement by minimizing robust baseline regret," *Advances in Neural Information Processing Systems*, vol. 29, 2016.