When to Trust Your Simulator: Dynamics-Aware Hybrid Offline-and-Online Reinforcement Learning

Haoyi Niu[1*], **Shubham Sharma**[1,2*], **Yiwen Qiu**[1*], **Ming Li**[1,3*],
Guyue Zhou[1], **Jianming Hu**[1,4], **Xianyuan Zhan**[1,5†]
[1] Tsinghua University, Beijing, China
[2] Indian Institute of Technology, Bombay, India
[3] Shanghai Jiaotong University, Shanghai, China
[4] Beijing National Research Center for Information Science and Technology, Beijing, China
[5] Shanghai AI Laboratory, Shanghai, China
{t6.da.thu,shubh.am1107z,qywmei,liming18739796090}@gmail.com
hujm@mail.tsinghua.edu.cn
{zhouguyue,zhanxianyuan}@air.tsinghua.edu.cn

Abstract

Learning effective reinforcement learning (RL) policies to solve real-world complex tasks can be quite challenging without a high-fidelity simulation environment. In most cases, we are only given imperfect simulators with simplified dynamics, which inevitably lead to severe sim-to-real gaps in RL policy learning. The recently emerged field of offline RL provides another possibility to learn policies directly from pre-collected historical data. However, to achieve reasonable performance, existing offline RL algorithms need impractically large offline data with sufficient state-action space coverage for training. This brings up a new question: is it possible to combine learning from limited real data in offline RL and unrestricted exploration through imperfect simulators in online RL to address the drawbacks of both approaches? In this study, we propose the Dynamics-Aware **H**ybrid **O**ffline-and-**O**nline Reinforcement Learning (H2O) framework to provide an affirmative answer to this question. H2O introduces a dynamics-aware policy evaluation scheme, which adaptively penalizes the Q-function learning on simulated state-action pairs with large dynamics gaps, while also simultaneously allowing learning from a fixed real-world dataset. Through extensive simulation and real-world tasks, as well as theoretical analysis, we demonstrate the superior performance of H2O against other cross-domain online and offline RL algorithms. H2O provides a brand new hybrid offline-and-online RL paradigm, which can potentially shed light on future RL algorithm design for solving practical real-world tasks.

1 Introduction

Over recent years, criticism against reinforcement learning (RL) continues to pour regarding its poor real-world applicability. Although RL has demonstrated superhuman performance in solving complex tasks such as playing games [Mnih *et al.*, 2013; Silver *et al.*, 2017], its success is heavily dependent on the availability of an unbiased interactive environment, either the real system or a high-fidelity simulator, as well as millions of unrestricted trials and errors. However, constructing high-fidelity simulators can be extremely expensive or even impossible due to the complex system dynamics and unobservable information in the physical world. Less accurate simulators with simplifications, on the other hand, are easy to build, however, also lead to severe visual and dynamics sim-to-real

[*] Work done during internships at Institute for AI Industry Research (AIR), Tsinghua University.
[†] Corresponding author.

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

gaps when learning RL policies [Rao *et al.*, 2020; Peng *et al.*, 2018]. Among different sources of sim-to-real gaps, visual gaps are relatively well-addressed in a number of methods, e.g., through domain randomization [Tobin *et al.*, 2017], proper model setups that are less impacted by visual gaps [Lee *et al.*, 2021], or using a robust and transferable state-encoder [Bewley *et al.*, 2019; Rao *et al.*, 2020]. Dynamics gap on the other hand causes a systematic bias that is not easy to address through a modeling process. These dynamics gaps widely exist in almost every imperfect physical simulator with simplified dynamics, posing great challenges for RL when solving real-world tasks.

Recently, offline RL provides another possibility to directly learn policies from real-world data [Levine *et al.*, 2020; Fujimoto *et al.*, 2018; Kumar *et al.*, 2019], which has already been used in solving a number of practical problems, such as industrial control [Zhan *et al.*, 2022a], robotics [Lee *et al.*, 2021] and interactive recommendation [Xiao and Wang, 2021]. However, the policies learned through existing offline RL algorithms are often over-conservative due to the use of data-related policy constraints [Fujimoto *et al.*, 2019; Kumar *et al.*, 2019; Fujimoto and Gu, 2021], value regularizations [Kumar *et al.*, 2020; Kostrikov *et al.*, 2021a; Xu *et al.*, 2022b], or in-sample learning [Kostrikov *et al.*, 2021b; Xu *et al.*, 2022a] to combat the distributional shift [Kumar *et al.*, 2019]. This hurts the performance of offline RL policies and makes their performances strongly depend on the size and state-action space coverage of the offline dataset. In real-world scenarios, data collection can be expensive or restrictive. The state-action space coverage of actual offline datasets can be quite narrow, which directly limits the potential performance of offline RL policies.

Despite all the drawbacks, online RL with an imperfect simulator and offline RL with the real-world dataset bear indispensable potential when jointly used to learn high-performance policies and overcome the sim-to-real issues. Within a simulator, online RL agents can perform unrestricted exploration and have an access to a great quantity and diversity of state-action data, which offline datasets may have poor coverage. On the other hand, real-world datasets contain perfect dynamics information, which can be readily used to guide and correct erroneous information in online training with simulation data. This intuition has motivated researchers to explore how to incorporate offline real-world data with simulation-based online training, however, existing studies are only restricted to pure online [Eysenbach *et al.*, 2020] or offline [Liu *et al.*, 2022] learning settings. None of the prior works has successfully combined offline and online learning organically, while simultaneously handling the dynamics gaps between these two regimes.

In this paper, we present the first Hybrid Offline-and-Online Reinforcement Learning framework (H2O), that enables simultaneous policy learning with offline real-world datasets and simulation rollouts. The core of H2O is the introduction of dynamics-aware policy evaluation, which adaptively pushes down or pulls up Q-values on simulated data according to the dynamics gap evaluated against real data. We provide analyses and insights on why such dynamics-aware learning can potentially bridge the sim-to-real gap. Finally, we analyze the impact of each design component in H2O and demonstrate its superior performance against other cross-domain RL algorithms through extensive simulation and real-world experimental validations.

2 Related Work

2.1 Reinforcement Learning Using Simulators with Dynamics Gap

Dealing with the dynamics gaps in simulators has long been recognized as a challenging task. The most straightforward approach is to calibrate the simulated dynamics with the real-world counterpart using system identification methods [Ljung, 1998; Rajeswaran *et al.*, 2016; Chebotar *et al.*, 2019]. These methods typically need plenty of offline or even real-world interaction data to tune the parameters of the simulator [Yu *et al.*, 2017], which sometimes can be costly or offer little improvement if the simulator is over-simplified. Domain Randomization (DR) [Peng *et al.*, 2018; Andrychowicz *et al.*, 2020] provides another perspective that trains RL policies in a series of randomized simulated dynamics. DR has been shown to yield more adaptable policies in real-world, yet it often needs manually-specified randomized parameters and nuanced randomization distributions [Vuong *et al.*, 2019]. More recently, Dynamics Adaptation, e.g., DARC [Eysenbach *et al.*, 2020], augments the simulation reward $r(\mathbf{s}, \mathbf{a})$ with a dynamics gap-related penalization term $\Delta r(\mathbf{s}', \mathbf{s}, \mathbf{a})$ in online simulation-based RL training, derived from minimizing the divergence of distributions over real and simulation trajectories. DARC uses real data and trains two discriminators to evaluate $\Delta r(\mathbf{s}', \mathbf{s}, \mathbf{a})$, rather than directly leveraging the data for RL training. In a similar effort, DARA [Liu *et al.*, 2022] is the pure offline version of DARC, sacrificing the unlimited online exploration in the simulated domain. While existing RL studies have tackled the dynamics gap issue

from different perspectives, none of them is able to develop a structured framework by combining offline data with online learning, simultaneously addressing the dynamics gap.

2.2 Offline Reinforcement Learning

The recently emerged offline RL methods provide a new alternative to learning policies directly from a fixed, pre-collected dataset. A key challenge of offline RL is the *distributional shift* problem [Kumar *et al.*, 2019; Levine *et al.*, 2020], which is caused by counterfactual queries of function approximators (e.g., value function, police network) on out-of-distribution (OOD) samples. Conventional off-policy RL methods will suffer severe value overestimation due to distributional shift. Existing offline RL methods solve this problem by introducing various data-related regularizations to constrain policy learning from deviating too much from the offline dataset, such as using data distribution, support or dataset distance based policy constraints [Fujimoto *et al.*, 2019; Kumar *et al.*, 2019; Wu *et al.*, 2019; Fujimoto and Gu, 2021; Xu *et al.*, 2021; Li *et al.*, 2022a], regularizing the value function on OOD data [Kumar *et al.*, 2020; Kostrikov *et al.*, 2021a; Yu *et al.*, 2021; Li *et al.*, 2022b; Xu *et al.*, 2022b], in-sample learning [Kostrikov *et al.*, 2021b; Xu *et al.*, 2022a], and adding uncertainty-based penalties [Yu *et al.*, 2020; Kidambi *et al.*, 2020; Zhan *et al.*, 2022a,b]. Conservatism and pessimism are central principles [Buckman *et al.*, 2020] used in most existing offline RL algorithms, which cause the performances of these methods to heavily depend on the size, quality, and state-action space coverage of the offline dataset. In most practical settings, getting a dataset with large coverage is impractical. Meanwhile, if online interaction with a simulator is involved, although may not be accurate, can substantially supplement the state-action space coverage of offline datasets in policy learning. This motivates a new paradigm of combining both offline and online RL, which is explored in our proposed H2O framework.

3 Background

3.1 Reinforcement Learning

We consider the standard Markov Decision Process (MDP) setting, which is specified by a tuple $\mathcal{M} := (\mathcal{S}, \mathcal{A}, r, P_{\mathcal{M}}, \rho, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, $r(\mathbf{s}, \mathbf{a})$ is the reward function, $P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s}, \mathbf{a})$ refers to the transitional dynamics under $\mathcal{M}$, $\rho(\mathbf{s})$ depicts the initial state distribution, and $\gamma \in (0, 1)$ is the discount factor. The goal of RL is to learn a policy $\pi(\mathbf{a}|\mathbf{s})$ that maps a state $\mathbf{s}$ to an action $\mathbf{a}$ so as to maximize the expected cumulative discounted reward starting from ρ, i.e., $J(\mathcal{M}, \pi) = \mathbb{E}_{\mathbf{s}_0 \in \rho, \mathbf{a}_t \sim \pi(\cdot|\mathbf{s}_t), \mathbf{s}_{t+1} \sim P_{\mathcal{M}}(\cdot|\mathbf{s}, \mathbf{a})} \left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)\right]$. In conventional actor-critic formalism [Barto *et al.*, 1983; Sutton *et al.*, 1998], one learns an approximated Q-function $\hat{Q}$ by minimizing the squared Bellman error (referred as *policy evaluation*), and optimizes the policy π by maximizing the Q-function (referred as *policy improvement*) as follows:

$$\hat{Q} \leftarrow \arg\min_{Q} \mathbb{E}_{\mathbf{s}, \mathbf{a}, \mathbf{s}' \sim \mathcal{U}} \left[\left(Q(\mathbf{s}, \mathbf{a}) - \hat{\mathcal{B}}^{\pi} \hat{Q}(\mathbf{s}, \mathbf{a}) \right)^2 \right] \qquad \text{(Policy Evaluation)} \tag{1}$$

$$\hat{\pi} \leftarrow \arg\max_{\pi} \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \mathcal{U}} \left[\hat{Q}(\mathbf{s}, \mathbf{a}) \right] \qquad \text{(Policy Improvement)} \tag{2}$$

where $\mathcal{U}$ can either be the replay buffer B generated by a previous version of policy $\hat{\pi}$ through online environment interactions, or can be a fixed dataset $\mathcal{D} = \left\{ \left(\mathbf{s}_t^i, \mathbf{a}_t^i, \mathbf{s}_{t+1}^i, r_t^i \right) \right\}_{i=1}^{n}$ as in offline RL setting. $\hat{\mathcal{B}}^{\pi}$ is the Bellman operator, which is often used as the Bellman evaluation operator $\hat{\mathcal{B}}^{\pi} \hat{Q}(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{a}' \sim \hat{\pi}(\mathbf{a}'|\mathbf{s}')} \left[\hat{Q}(\mathbf{s}', \mathbf{a}') \right]$.

3.2 Offline RL via Value Regularization

In the offline setting, performing the standard Bellman update in Eq. (1) can result in serious overestimation of Q-values due to distributional shift. An effective approach to alleviating this problem is to regularize the value function on OOD actions, which is introduced in CQL [Kumar *et al.*, 2020]. CQL learns a policy $\hat{\pi}$ upon a lower-bounded Q-function that additionally pushes down the Q-values on actions induced by $\mu(\mathbf{a}|\mathbf{s})$ and pulls up Q-values on trustworthy offline data:

$$\min_{Q} \max_{\mu} \alpha \left(\mathbb{E}_{\mathbf{s} \sim \mathcal{D}, \mathbf{a} \sim \mu(\cdot|\mathbf{s})}[Q(\mathbf{s}, \mathbf{a})] - \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \mathcal{D}}[Q(\mathbf{s}, \mathbf{a})] \right) + \mathcal{E}\left(Q, \hat{\mathcal{B}}^{\pi} \hat{Q}\right) \tag{3}$$

where the dataset $\mathcal{D}$ is collected by some behavioral policy $\pi_{\mathcal{D}}(\cdot|\mathbf{s})$, and $\mu(\cdot|\mathbf{s})$ is some sampling distribution, which is often taken as $\hat{\pi}(\cdot|\mathbf{s})$ in prior works [Yu *et al.*, 2021; Li *et al.*, 2022b]. $\mathcal{E}\left(Q, \hat{\mathcal{B}}^{\pi} \hat{Q}\right)$

denotes the Bellman error in Eq. (1). It is worth noting that the above value regularization framework has the potential to handle data from two sources with some preference, i.e., unreliable pushed-down data and reliable pulled-up data. This motivates us to devise the H2O framework to perform simultaneous offline and online policy learning using a similar value regularization recipe.

4 Hybrid Offline-and-Online Reinforcement Learning

Naïvely combining offline and online data from different distributions may cause the mismatch between data distributions, causing issues similar to the distributional shift problem as in offline RL settings [Kumar *et al.*, 2019]. Caution needs to be taken when dealing with trustworthy offline data from the real world and the potentially problematic simulated online rollouts (described in Section 4.1). Secondly, the dynamics model underpinning the simulator is only roughly aligned with the real-world counterpart, and the dynamics sim-to-real gaps can be heterogeneous across different state-action pairs. Hence uniform value regularization upon all simulated samples may not be appropriate to fully leverage the potential of an imperfect simulator. A well-founded dynamics gap measurement for adaptive value regularization is needed (Section 4.2). Lastly, it is also worth noting that the biased simulated dynamics could induce erroneous next-state predictions, which may lead to problematic Bellman updates. Such errors also need to be properly considered in the algorithm design (Section 4.3). We approach all the aforementioned issues with what we call Dynamics-Aware Hybrid Offline-and-Online (H2O) RL, which is described in the following content.

4.1 Incorporating Offline Data in Online Learning

To address the potential mismatch of training data distributions from different data sources, existing online RL frameworks are not suitable, however, offline RL approaches such as value regularization can provide a viable foundation. We modify the previous value regularization scheme in offline RL and propose the dynamics-aware policy evaluation, which has the following general form:

$$\min_{Q} \max_{d^\phi} \beta \left[\mathbb{E}_{\mathbf{s},\mathbf{a}\sim d^\phi(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}[Q(\mathbf{s},\mathbf{a})] + \mathcal{R}(d^\phi)\right] + \widetilde{\mathcal{E}}\left(Q, \hat{\mathcal{B}}^\pi \hat{Q}\right) \tag{4}$$

where β is a positive scaling parameter. $d^\phi(\mathbf{s},\mathbf{a})$ is a particular state-action sampling distribution that associates with high dynamics-gap samples, and $\mathcal{R}(d^\phi)$ is a regularization term for d^ϕ to enforce this designed behavior. Ideally, we would want to penalize Q-values at the simulated samples with high dynamics gaps, rather than all of the samples. $\widetilde{\mathcal{E}}(Q, \hat{\mathcal{B}}^\pi \hat{Q})$ denotes the modified Bellman error of the mixed data from offline dataset $\mathcal{D}$ and the simulation rollout samples in online replay buffer B, which are generated by the real MDP $\mathcal{M}$ and the simulated MDP $\widehat{\mathcal{M}}$ respectively. Specifically, the Bellman error of the simulated data needs to be fixed. The terms marked in red and blue are the key design elements that account for the dynamics gaps in offline-and-online RL policy learning, which will be discussed in the following sub-sections.

4.2 Adaptive Value Regularization on High Dynamics-Gap Samples

To achieve adaptive value regularization, we can use the minimization term $\mathbb{E}_{\mathbf{s},\mathbf{a}\sim d^\phi(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})]$ to penalize the high-dynamics gap samples, and use the maximization term $\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}[Q(\mathbf{s},\mathbf{a})]$ to cancel the penalization on real offline data. The key question is how to properly design $d^\phi(\mathbf{s},\mathbf{a})$ to assign high probabilities to high-dynamics gap samples. In our study, we leverage the regularization term $\mathcal{R}(d^\phi)$ to control the behavior of $d^\phi(\mathbf{s},\mathbf{a})$. We choose $\mathcal{R}(d^\phi) = -D_{KL}(d^\phi(\mathbf{s},\mathbf{a})\|\omega(\mathbf{s},\mathbf{a}))$, where $D_{KL}(\cdot||\cdot)$ is the Kullback-Leibler (KL) divergence and $\omega(\mathbf{s},\mathbf{a})$ is a distribution that characterizes the dynamics gaps for samples in the state-action space. Hence maximizing over $\mathcal{R}(d^\phi)$ draws closer between d^ϕ and ω, and achieves our desired behavior. Note that the inner maximization problem over d^ϕ in Eq. (4) under this design corresponds to the following optimization problem:

$$\max_{d^\phi} \mathbb{E}_{\mathbf{s},\mathbf{a}\sim d^\phi(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})] - D_{KL}(d^\phi(\mathbf{s},\mathbf{a})\|\omega(\mathbf{s},\mathbf{a})) \quad \text{s.t.} \sum_{\mathbf{s},\mathbf{a}} d^\phi(\mathbf{s},\mathbf{a}) = 1, d^\phi(\mathbf{s},\mathbf{a}) \geq 0 \tag{5}$$

Above optimization problem admits a closed-form solution $d^\phi(\mathbf{s},\mathbf{a}) \propto \omega(\mathbf{s},\mathbf{a}) \exp\left(Q(\mathbf{s},\mathbf{a})\right)$ (see the derivation in Appendix A.1). Plugging this back into Eq. (4), the first term now corresponds to a weighted soft-maximum of Q-values at any state-action pair and the original problem transforms into the following form:

$$\min_{Q} \beta \left(\log \sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s},\mathbf{a}) \exp\left(Q(\mathbf{s},\mathbf{a})\right) - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}\left[Q(\mathbf{s},\mathbf{a})\right]\right) + \widetilde{\mathcal{E}}\left(Q, \hat{\mathcal{B}}^\pi \hat{Q}\right) \tag{6}$$

This result is intuitively reasonable, as we are penalizing more on Q-values with larger $\omega(\mathbf{s}, \mathbf{a})$, corresponding to those high dynamics-gap samples. Then the next question is, how do we practically evaluate $\omega(\mathbf{s}, \mathbf{a})$ given the offline dataset and the online simulation rollouts? Specifically, we can measure the dynamics gap between real and simulated dynamics on a state-action pair $(\mathbf{s}, \mathbf{a})$ as $u(\mathbf{s}, \mathbf{a}) := D_{KL}(P_{\widehat{\mathcal{M}}}(\mathbf{s}'|\mathbf{s}, \mathbf{a}) \| P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s}, \mathbf{a})) = \mathbb{E}_{s' \sim P_{\widehat{\mathcal{M}}}} \log(P_{\widehat{\mathcal{M}}}(\mathbf{s}'|\mathbf{s}, \mathbf{a}) / P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s}, \mathbf{a}))$. $\omega(s, a)$ can thus be represented as a normalized distribution of u, i.e., $\omega(\mathbf{s}, \mathbf{a}) = u(\mathbf{s}, \mathbf{a}) / \sum_{\tilde{\mathbf{s}}, \tilde{\mathbf{a}}} u(\tilde{\mathbf{s}}, \tilde{\mathbf{a}})$. Now, the challenge is how to evaluate the dynamics ratio $P_{\widehat{\mathcal{M}}} / P_{\mathcal{M}}$. Note that according to Bayes' rule:

$$\begin{aligned} \frac{P_{\widehat{\mathcal{M}}}(\mathbf{s}'|\mathbf{s}, \mathbf{a})}{P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s}, \mathbf{a})} &= \frac{p(\mathbf{s}'|\mathbf{s}, \mathbf{a}, \text{sim})}{p(\mathbf{s}'|\mathbf{s}, \mathbf{a}, \text{real})} = \frac{p(\text{sim}|\mathbf{s}, \mathbf{a}, \mathbf{s}')}{p(\text{sim}|\mathbf{s}, \mathbf{a})} \Big/ \frac{p(\text{real}|\mathbf{s}, \mathbf{a}, \mathbf{s}')}{p(\text{real}|\mathbf{s}, \mathbf{a})} \\ &= \frac{p(\text{sim}|\mathbf{s}, \mathbf{a}, \mathbf{s}')}{p(\text{real}|\mathbf{s}, \mathbf{a}, \mathbf{s}')} \Big/ \frac{p(\text{sim}|\mathbf{s}, \mathbf{a})}{p(\text{real}|\mathbf{s}, \mathbf{a})} = \frac{1 - p(\text{real}|\mathbf{s}, \mathbf{a}, \mathbf{s}')}{p(\text{real}|\mathbf{s}, \mathbf{a}, \mathbf{s}')} \Big/ \frac{1 - p(\text{real}|\mathbf{s}, \mathbf{a})}{p(\text{real}|\mathbf{s}, \mathbf{a})} \end{aligned} \tag{7}$$

Hence, we can approximate $p(\text{real}|\mathbf{s}, \mathbf{a}, \mathbf{s}')$ and $p(\text{real}|\mathbf{s}, \mathbf{a})$ with a pair of discriminators $D_{\Phi_{sas}}(\cdot|\mathbf{s}, \mathbf{a}, \mathbf{s}')$ and $D_{\Phi_{sa}}(\cdot|\mathbf{s}, \mathbf{a})$ respectively that are optimized with standard cross-entropy loss between real offline data and the simulated samples as in DARC [Eysenbach *et al.*, 2020].

4.3 Fixing Bellman Error due to Dynamics Gap

Due to the existence of dynamics gaps in the simulator, directly computing the Bellman error for simulated samples in the online replay buffer B (i.e., $\mathbb{E}_{\mathbf{s}, \mathbf{a}, \mathbf{s}' \sim B}[(Q - \hat{\mathcal{B}}^{\pi} \hat{Q})(\mathbf{s}, \mathbf{a})]^2$) is problematic. This is because the next state $\mathbf{s}'$ comes from the potentially biased simulated dynamics $P_{\widehat{M}}$, which can result in the miscalculation of target Q-values in Bellman updates. Ideally, we wish the next state $\mathbf{s}'$ comes from the real dynamics $P_{\mathcal{M}}$, however, such a $\mathbf{s}'$ given an arbitrary state-action pair $(\mathbf{s}, \mathbf{a}) \sim B$ is not obtainable during training. To fix this issue, we reuse the previously evaluated dynamics ratio in Eq. (7) as an importance sampling weight, and introduce the following modified Bellman error formulation for both offline dataset $\mathcal{D}$ and simulated data B:

$$\begin{aligned} \widetilde{\mathcal{E}}\left(Q, \hat{\mathcal{B}}^{\pi} \hat{Q}\right) &= \frac{1}{2} \mathbb{E}_{\mathbf{s}, \mathbf{a}, \mathbf{s}' \sim \mathcal{D}} \left[\left(Q - \hat{\mathcal{B}}^{\pi} \hat{Q}\right)(\mathbf{s}, \mathbf{a})\right]^2 + \frac{1}{2} \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim B} \mathbb{E}_{\mathbf{s}' \sim p_{\mathcal{M}}} \left[\left(Q - \hat{\mathcal{B}}^{\pi} \hat{Q}\right)(\mathbf{s}, \mathbf{a})\right]^2 \\ &= \frac{1}{2} \mathbb{E}_{\mathbf{s}, \mathbf{a}, \mathbf{s}' \sim \mathcal{D}} \left[\left(Q - \hat{\mathcal{B}}^{\pi} \hat{Q}\right)(\mathbf{s}, \mathbf{a})\right]^2 + \frac{1}{2} \mathbb{E}_{\mathbf{s}, \mathbf{a}, \mathbf{s}' \sim B} \left[\frac{P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s}, \mathbf{a})}{P_{\widehat{\mathcal{M}}}(\mathbf{s}'|\mathbf{s}, \mathbf{a})} \left(Q - \hat{\mathcal{B}}^{\pi} \hat{Q}\right)(\mathbf{s}, \mathbf{a})\right]^2 \end{aligned} \tag{8}$$

4.4 Practical Implementation

Combining Eq. (6) and (8), we obtain the final dynamics-aware policy evaluation procedure for H2O, which enables adaptive value regularization on high dynamics-gap samples as well as more reliable Bellman updates. H2O can be instantiated upon common actor-critic algorithms (e.g., Soft Actor-Critic (SAC) [Haarnoja *et al.*, 2018]). The pseudocode of H2O built upon SAC is presented in Algorithm 1, which uses the policy improvement objective from SAC and λ is a temperature parameter auto-tuned during training. Other policy improvement objectives, or simply maximizing the expected Q-values as in Eq. (2) are also compatible with the H2O framework.

For practical considerations, we introduce two relaxations in our implementation. First, in Eq. (6), we need to sample from the whole state-action space to compute the weighted average of exponentiated Q-values, which can be impractical and unnecessary since not all state-action pairs are of interest for policy learning. Instead, we approximate this value using the state-action samples in the mini-batch of the simulated replay buffer B. Second, the evaluation of the KL divergence in $u(\mathbf{s}, \mathbf{a}) = \mathbb{E}_{s' \sim P_{\widehat{\mathcal{M}}}} \log(P_{\widehat{\mathcal{M}}}(\mathbf{s}'|\mathbf{s}, \mathbf{a}) / P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s}, \mathbf{a}))$ involves sampling next states $\mathbf{s}'$ from $P_{\widehat{\mathcal{M}}}(\cdot|\mathbf{s}, \mathbf{a})$, which can be infeasible given a black-box simulator. We take a simplified approach that approximates the expected value by averaging over N random samples from a Gaussian distribution $\mathcal{N}(\mathbf{s}', \hat{\Sigma}_{\mathcal{D}})$, where $\hat{\Sigma}_{\mathcal{D}}$ is the covariance matrix of states computed from the real offline dataset. Although this treatment is simple, we find it highly effective and produces good performance in empirical experiments. Please turn to Appendix B.1 for more implementation details.

5 Interpretation of Dynamics-Aware Policy Evaluation

In this section, we analyze H2O and provide intuition on how it works when combining both offline and online learning. Interestingly, we find that the final form of dynamics-aware policy evaluation as given in Eq. (6) and (8) actually leads to a equivalent adaptive reward adjustment term on the Q-function, which depends on the dynamics gap distribution $\omega(\mathbf{s}, \mathbf{a})$ and the state-action distribution of the dataset $d_M^{\pi_{\mathcal{D}}}(\mathbf{s}, \mathbf{a})$. To show this, note that the weighted log-sum-exp term

Algorithm 1: Dynamics-Aware Hybrid Offline-and-Online Reinforcement Learning (H2O)

Data: an offline dataset $\mathcal{D}$ from the real world, an imperfect simulator with biased dynamics $\widehat{\mathcal{M}}$
Initialize: critic network Q_θ, target network $Q_{\bar{\theta}}$, actor network π_ϕ, replay buffer $B = \varnothing$ for simulated transitions, discriminators $D_{\Phi_{sas}}(\cdot|\mathbf{s}, \mathbf{a}, \mathbf{s}')$ and $D_{\Phi_{sa}}(\cdot|\mathbf{s}, \mathbf{a})$
for *step* $t = 1, \cdots, T$ **do**
  $B \leftarrow B \cup \textsc{Rollout}(\pi_\phi, \widehat{\mathcal{M}})$; ▷ Collect simulated data
  $\Phi_{sas} \leftarrow \Phi_{sas} - \eta \nabla_{\Phi_{sas}} \textsc{CrossEntropyLoss}(B, \mathcal{D}, \Phi_{sas})$; ▷ Update $D_{\Phi_{sas}}$
  $\Phi_{sa} \leftarrow \Phi_{sa} - \eta \nabla_{\Phi_{sa}} \textsc{CrossEntropyLoss}(B, \mathcal{D}, \Phi_{sa})$; ▷ Update $D_{\Phi_{sa}}$
  $\theta \leftarrow \theta - \eta_Q \nabla_\theta \left[\beta \left(\log \sum \omega(\mathbf{s}, \mathbf{a}) \exp\left(Q_\theta(\mathbf{s}, \mathbf{a})\right) - \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \mathcal{D}}\left[Q_\theta(\mathbf{s}, \mathbf{a})\right]\right) + \widetilde{\mathcal{E}}\left(Q_\theta, \hat{\mathcal{B}}^\pi Q_{\bar{\theta}}\right)\right]$;
  $\phi \leftarrow \phi + \eta_\pi \nabla_\phi \left[\mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \{\mathcal{D} \cup B\}}\left[Q_\theta(\mathbf{s}, \mathbf{a}) - \lambda \log \pi_\phi(\mathbf{a}|\mathbf{s})\right]\right]$;
  if *t % target_update_period = 0* **then**
    $\bar{\theta} \leftarrow (1 - \tau)\bar{\theta} + \tau\theta$; ▷ Soft update periodically
  end
end

$\log \sum_{\mathbf{s}, \mathbf{a}} \omega(\mathbf{s}, \mathbf{a}) \exp\left(Q(\mathbf{s}, \mathbf{a})\right) = \log \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \omega(\mathbf{s}, \mathbf{a})} \exp(Q(\mathbf{s}, \mathbf{a}))$ in Eq. (6) can be bounded as follows (see Appendix A.2 for detailed proof):

$$\mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \omega(\mathbf{s}, \mathbf{a})}[Q(\mathbf{s}, \mathbf{a})] \leq \log \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \omega(\mathbf{s}, \mathbf{a})} \exp(Q(\mathbf{s}, \mathbf{a})) \leq \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \omega(\mathbf{s}, \mathbf{a})}[Q(\mathbf{s}, \mathbf{a})] + \frac{\mathrm{Var}_\omega[\exp(Q(\mathbf{s}, \mathbf{a}))]}{2\exp(2Q_{min})} \tag{9}$$

where we denote the range of learned Q-values as $[Q_{min}, Q_{max}]$, and $\mathrm{Var}_\omega[\exp(Q(\mathbf{s}, \mathbf{a}))]$ denotes the variance of $\exp(Q(\mathbf{s}, \mathbf{a}))$ under samples drawn from distribution $\omega(\mathbf{s}, \mathbf{a})$. The LHS inequality is a result of Jensen's inequality, and the RHS inequality is a direct result from [Liao and Berg, 2018]. Above inequalities suggest that $\mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \omega(\mathbf{s}, \mathbf{a})}[Q(\mathbf{s}, \mathbf{a})]$ can be a reasonable approximation of our weighted log-sum-exp term if the Q-function takes large values and the gap between $\exp(Q_{min})$ and $\sqrt{Var_\omega[\exp(Q(\mathbf{s}, \mathbf{a}))]}$ is not too large. This can be practically satisfied if we let $\gamma \to 1$ and design the range of reward function as well as the episode done conditions properly.

With this approximation, we can consider the following approximated policy evaluation objective based on Eq. (6) and (8), which offers a much cleaner form for analysis:

$$\min_Q \beta \left(\mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \omega(\mathbf{s}, \mathbf{a})}[Q(\mathbf{s}, \mathbf{a})] - \mathbb{E}_{\mathbf{s}, \mathbf{a} \sim \mathcal{D}}\left[Q(\mathbf{s}, \mathbf{a})\right]\right) + \widetilde{\mathcal{E}}\left(Q, \hat{\mathcal{B}}^\pi \hat{Q}^k\right) \tag{10}$$

The ablation study in Appendix C.1 manifests that this approximated policy evaluation objective only yields mild performance drops against the original version in most cases, making it a reasonable substitution for theoretical analysis. Assuming that the Q-function is tabular, we can find the Q-value corresponding to the new objective by following an approximate dynamic programming approach and differentiating Eq. (10) with respect to Q^k in iteration k (see Appendix A.2 for details):

$$\hat{Q}^{k+1}(\mathbf{s}, \mathbf{a}) = (\hat{\mathcal{B}}^\pi \hat{Q}^k)(\mathbf{s}, \mathbf{a}) - \beta \left[\frac{\omega(\mathbf{s}, \mathbf{a}) - d_M^{\pi_\mathcal{D}}(\mathbf{s}, \mathbf{a})}{d_M^{\pi_\mathcal{D}}(\mathbf{s}, \mathbf{a}) + d_{\widehat{M}}^{\pi}(\mathbf{s}, \mathbf{a})}\right] \tag{11}$$

where $d_M^{\pi_\mathcal{D}}(\mathbf{s}, \mathbf{a})$ and $d_{\widehat{M}}^{\pi}(\mathbf{s}, \mathbf{a})$ are state-action marginal distributions under behavioral policy $\pi_\mathcal{D}$ and the learned policy π respectively. If we represent the second term in the RHS of above equation as $\nu(\mathbf{s}, \mathbf{a})$, we can see that $\nu(\mathbf{s}, \mathbf{a})$ can be perceived as an adaptive reward adjustment term, which penalizes or boosts the reward at a state-action pair $(\mathbf{s}, \mathbf{a})$ according to the relative difference between dynamics gap distribution $\omega(\mathbf{s}, \mathbf{a})$ and marginal state-action distribution of the offline dataset $d_M^{\pi_\mathcal{D}}(\mathbf{s}, \mathbf{a})$. Specifically, we can make the following interesting observations:

- If $\omega(\mathbf{s}, \mathbf{a}) > d_M^{\pi_\mathcal{D}}(\mathbf{s}, \mathbf{a})$, then $\nu(\mathbf{s}, \mathbf{a})$ serves as a reward penalty. This corresponds to the case that either the state-action pair $(\mathbf{s}, \mathbf{a})$ has large dynamics gap, or it belongs to OOD or relatively low data density areas ($d_M^{\pi_\mathcal{D}}(\mathbf{s}, \mathbf{a}) = 0$ or $< \omega(\mathbf{s}, \mathbf{a})$). Under such cases, we push down the Q-values and adopt conservatism similar to the typical treatment in offline RL settings [Kumar *et al.*, 2020]. The higher the dynamics gap is (larger ω), the more we penalize the Q-function.
- If $\omega(\mathbf{s}, \mathbf{a}) \leq d_M^{\pi_\mathcal{D}}(\mathbf{s}, \mathbf{a})$, then we are evaluating at some good data areas with low dynamics gap or have more information from offline data to control the potential dynamics gap in the simulated

online samples. Under such cases, $\nu(\mathbf{s}, \mathbf{a})$ serves as a reward boost term and adopts optimism. Although it overestimates the Q-values, we argue that this can be beneficial. As it can promote exploration in these "good" data regions and potentially help reduce variance during training.

With the adaptive reward adjustment term ν defined in Eq. (11), we can further show that the value function $V(\mathbf{s})$ is underestimated at high dynamics gap areas, which allows for safe policy learning in our offline-and-online setting. Detailed theoretical analysis can be found in Appendix A.3.

6 Experiments

In this section, we present empirical evaluations of H2O. We start by describing our experimental environment setups and the cross-domain RL baselines for comparison. We then evaluate H2O against the baseline methods in simulation environments and on a real wheel-legged robot. Ablation studies and empirical analyses of H2O are also reported. Additional experiment settings, ablations, and results can be found in Appendix B, C, and D respectively.

6.1 Experimental Environment Setups

Simulation-based experiments. We conduct simulation-based experiments in the MuJoCo physics simulator [Todorov *et al.*, 2012]. In particular, we construct three new simulation task environments (serve as the simulated environments) with intentionally introduced dynamics gaps upon the original MuJoCo-HalfCheetah task environments (serve as the real environments) by modifying the dynamics parameters: (1) **Gravity**: applying 2 times the gravitational acceleration in the simulation dynamics; (2) **Friction**: using 0.3 times the friction coefficient to make the agent harder to maintain balance; (3) **Joint Noise**: adding a random noise sampled from $\mathcal{N}(0, 1)$ on every dimension of the action space, mimicking a system with large control noise. As for the offline dataset from the real world (original simulation environment), we use the datasets of the corresponding task from standard offline RL benchmark D4RL [Fu *et al.*, 2020]. Specifically, we consider the Medium, Medium Replay and Medium Expert datasets, as in typical real-world scenarios, we do not use a random policy or do not have an expert policy for system control. The online training is performed on the modified simulation environment, and we evaluate the performance of the learned policy in the original unchanged MuJoCo environment in terms of the average return.

Real-world experiments. We also evaluate the performance of H2O on a real wheel-legged robot that moves on a pair of wheels to keep it balanced, as illustrated in Fig. 1(a). The state space of the robot is a quadratic-tuple $\mathcal{S} = (\theta, \dot{\theta}, x, \dot{x})$, where θ denotes the forward tilt angle of the body, x is the displacement of the robot, $\dot{\theta}$ and $\dot{x}$ are the angular and linear velocity respectively. The control action is the torque τ of the motors at the two wheels. Here we construct two tasks for real-world validation: (1) **Standing still**: We want to make the robot stand still and not move or fall down; (2) **Moving straight**: We want to control the robot to move forward at a target velocity v while keeping its balance. We record 100,000 human-controlled transitions with different reward functions for both tasks, which serve as the offline dataset used in RL training. Additional real-world experiment settings like reward functions and dataset analyses are elaborated in Appendix B.3.

The control task is relatively challenging, as the robot has to use two wheels to keep balance and can easily fall to the ground. The simulation environment is constructed based on Isaac Gym [Makoviychuk *et al.*, 2021], depicted in Fig. 1(b). The simulation environment suffers from intricate dynamics gaps. Notably, the electric motors in the real robot bear dead zones, which can lead to observational errors, whereas the simplified dynamics in simulation fail to capture this complicated situation. Also, the rolling friction applied on wheels is well-simulated, but the sliding friction cannot be modeled realistically in Isaac Gym. Furthermore, the friction coefficient between wheels and the ground as well as its wear-and-tear, are not possible to be configured exactly as in the real-world environment. All these dynamics gaps can cause serious sim-to-real transfer issues when deploying the RL policy learned in simulation to real-world scenarios.

6.2 Baselines

As H2O is the first RL framework under the hybrid offline-and-online setting, we can only compare it with some purely online or offline RL algorithms, as well as representative methods that partially incorporate offline data in online or offline policy learning in a less integrated manner, which are:

- **SAC (sim)** [Haarnoja *et al.*, 2018]: the SOTA off-policy online RL algorithm, which is trained within the imperfect simulator and evaluated in the original simulation or real-world environments.

Table 1: Average returns for MuJoCo-HalfCheetah tasks. Results are averaged over 5 random seeds.

Dataset	Unreal dynamics	SAC(sim)	CQL(real)	DARC	DARC+	H2O
Medium	Gravity	4513±513	6066±73	5011±456	5706±440	**7085±416**
	Friction	2684±2646	6066±73	6113±104	6047±112	**6848±445**
	Joint Noise	4137±805	6066±73	5484±171	5314±520	**7212±236**
Medium Replay	Gravity	4513±513	5774±214	5105±460	4958±540	**6813±289**
	Friction	2684±2646	5774±214	5503±263	5288±100	**5928±896**
	Joint Noise	4137±805	5774±214	5137±225	5230±209	**6747±427**
Medium Expert	Gravity	4513±513	3748±892	**4759±353**	72±109	**4707±779**
	Friction	2684±2646	3748±892	**9038±1480**	7989±3999	6745±562
	Joint Noise	4137±805	3748±892	**5288±104**	733±767	**5280±1329**

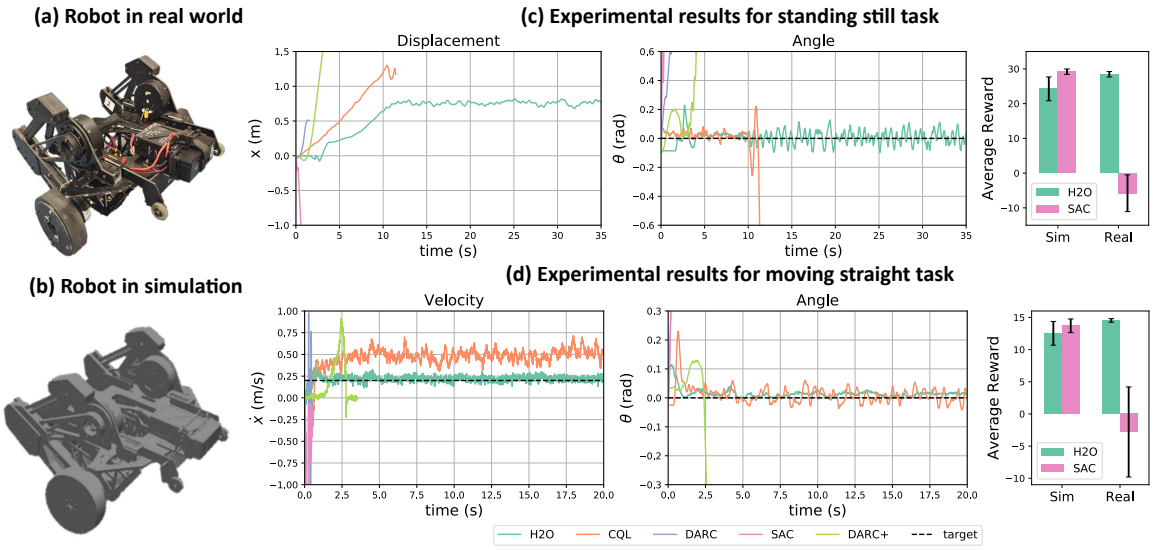

Figure 1: Real-world validation on a wheel-legged robot

- **CQL (real)** [Kumar *et al.*, 2020]: the representative offline RL algorithm using value regularization. We run CQL on the D4RL and real-world robotic control dataset, which is not impacted by the dynamics gaps, but suffers from limited state-action space coverage in the offline dataset.
- **DARC** [Eysenbach *et al.*, 2020]: DARC uses a reward correction term derived from a pair of binary discriminators to optimize the policy within an imperfect simulator. We train these discriminators with real and simulation samples and learn policy within the simulation.
- **DARC+**: a variant of DARC, in which the real offline data are not only utilized for discriminator training, but also used in policy evaluation and improvement. DARC+ showcases the model behavior when we naïvely combine offline and online policy learning under the DARC framework.

6.3 Comparative Evaluation of H2O in Simulation and Real-World Experiments

In Table 1, we present the comparative results of H2O and other baselines on the simulation-based experiments. Notably, H2O achieves the best performance in almost all tasks compared with the baselines. Due to the relatively large dynamics gap, directly training SAC in simulation yields the poorest performance in most of the tasks. Offline RL method CQL achieves reasonable performance as compared with the online RL counterpart, which achieves second-highest scores in Medium and Medium Replay tasks. DARC performs surprisingly very well in the Medium Expert-Friction task, perhaps due to specific dataset and dynamics gap properties. DARC+ shows inferior performance compared with DARC in most tasks, indicating the insufficiency of naïvely combining offline and online policy learning. Nevertheless, H2O consistently provides superior performance under different offline datasets and dynamics gap settings, demonstrating the effectiveness of our proposed dynamics-aware policy evaluation scheme.

Table 2: Ablations on H2O. "-a" denotes no adaptive weights ω. "-dr" denotes no dynamics ratio to fix the Bellman error. "-reg" stands for no value regularization applied in the framework.

	H2O	H2O-a	H2O-dr	H2O-a-dr	H2O-reg	H2O-reg-dr	CQL	SAC
Adaptive ω	✓	✗	✓	✗	-	-	-	-
Uniform ω	✗	✓	✗	✓	-	-	-	-
Modified $\widetilde{\mathcal{E}}\left(Q, \hat{\mathcal{B}}^{\pi}\hat{Q}\right)$	✓	✓	✗	✗	✓	✗	-	-
Average Return	**6813±289**	6675±179	4721±196	5223±198	6501±147	5290±356	5774±214	4513±513

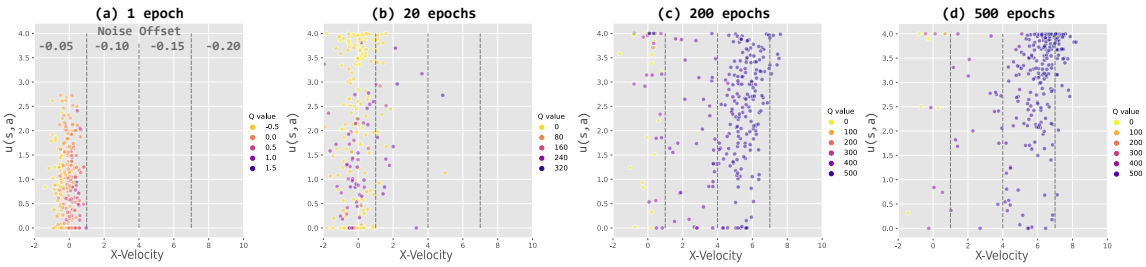

Figure 2: The dynamics gap measure $u(\mathbf{s}, \mathbf{a})$ evaluated during the training process

In the real-world experiments[3], H2O manifests overwhelmingly better transferability on both standing still and moving straight tasks. In the standing still task, as demonstrated in the comparative curves in Fig. 1(c), the robot with H2O policy remains steady after 11 seconds (s), while CQL bumps into the ground and goes out of control at 12s. DARC, DARC+, and SAC are not able to keep the robot balanced and quickly fail after initialization. In the moving straight task demonstrated in Fig. 1(d), H2O achieves great control performance to keep balanced and closely follow the target velocity $v = 0.2$m/s, while CQL exceeds v by a fairly large margin, nearly doubling the desired target velocity. Additionally, the angle θ illustrates that the robot controlled by H2O runs more smoothly than CQL. Once again, DARC, DARC+ and SAC fail at the beginning. Interestingly, we plot the evaluated average returns in both simulation and the real-world robot for SAC and H2O, revealing that achieving high performance in the simulation environment provides no guarantee for real-world transferability, if the simulator has a large sim-to-real gap. It is possible that policies with lower performance in the biased simulation could perform better in real-world scenarios. This suggests the potential need of reconsidering the widely adopted simulation-based evaluation or verification in mission-critical tasks, such as in autonomous driving and industrial control, that overly trust simulators with dynamics gaps can lead to serious consequences.

6.4 Ablation Study

In this section, we investigate the impact of each design component of H2O in the HalfCheetah-Medium Replay-Gravity task. In Table 2, we compare the performances of the variants of H2O by replacing adaptive value regularization weight ω into a uniform value, removing the importance weight in modified Bellman Error formulation $\widetilde{\mathcal{E}}$ or even removing the entire value regularization part from the framework. Without the adaptive value regularization, "H2O-a" demonstrates lower performance since the Q-values at high dynamics-gap samples are not properly regularized during training. Without the dynamics ratio as importance weight to fix the Bellman error, "H2O-dr", "H2O-a-dr" and "H2O-reg-dr" shows severe performance deterioration against their calibrated counterparts, which signifies the importance of correctly handling the Bellman updates when involving multiple sources of data. Removing the entire value regularization part from H2O leads to substantial performance degradation. However, it is noteworthy that "H2O-reg-dr" outperforms "H2O-dr", again suggests that fixing the Bellman error due to dynamics gap is very important in hybrid offline-and-online policy learning.

To further investigate the quality of the dynamics gap measure $u(\mathbf{s}, \mathbf{a})$ in H2O as well as its impact on the adaptive value regularization during the learning process, we construct a new test environment based on HalfCheetah. We apply specific noise offsets to actions based on X-velocity (see Fig. 2) in the simulation, such that larger X-velocity bears larger noise offsets. We plot the dynamics gap measure $u(\mathbf{s}, \mathbf{a})$ and the corresponding Q-values $Q(\mathbf{s}, \mathbf{a})$ for each simulated sample $(\mathbf{s}, \mathbf{a})$ from a

[3]Please refer to supplementary video for performances of H2O and baselines on the real wheel-legged robot.

batch in some stages of the learning process. Notably, in the HalfCheetah task, higher X-velocity corresponds to higher rewards and hence corresponds to higher Q-values, which at the same time may suffer from more severe dynamics gaps. We observe that as the course of learning proceeds, the dynamics gap measures u in H2O indeed capture the higher dynamics gap information at the later part of the training, and works as expected to penalize the Q-values on high dynamics-gap samples.

7 Conclusion and Perspectives

In this paper, we propose the dynamics-aware Hybrid Offline-and-Online RL (H2O) framework to combine offline and online RL policy learning, while simultaneously addressing the sim-to-real dynamics gaps in an imperfect simulator. H2O introduces a dynamics-aware policy evaluation scheme, which adaptively penalizes the Q-values as well as fixes the Bellman error on simulated samples with large dynamics gaps. This scheme can be shown as equivalent to a special kind of reward adjustment, which places reward penalties on high dynamics-gap samples, but boost reward in regions with low dynamics gaps and abundant offline data. Under H2O, both the offline dataset and the imperfect simulator are fully used and complement each other to extract the maximum information for policy learning. In particular, the limited size and state-action space coverage of the offline datasets are greatly supplemented by unrestricted simulated samples; the dynamics gaps in simulated samples can also be alleviated during policy learning given correct dynamics information from the real data. Through extensive simulation and real-world experiments, we demonstrate the superior performance of H2O against other cross-domain online and offline RL methods.

Online and offline RL have been two separately studied fields in the past, and both face a set of specific challenges. Few attempts have been made to organically combine these two different RL paradigms. H2O provides a brand new hybrid offline-and-online RL paradigm, which shows promising results when leveraging offline data and simulation-based online learning in an integrated framework. Still, there are several future works that can be done to further enhance the model performance, such as adopting a less conservative offline RL backbone algorithm as compared to CQL, and incorporating better dynamics gap quantification methods that do not need approximation. We hope the insights developed in this work can shed light on future hybrid RL algorithm design, which can potentially provide better RL solutions for practical deployment.

Acknowledgments and Disclosure of Funding

This work is supported by funding from Haomo.AI. Haoyi Niu is also funded by Tsinghua Undergraduate "Future Scholar" Scientific Research Grant, i.e. Tsinghua University Initiative Scientific Research Program (20217020007). The authors would also like to thank the anonymous reviewers for their feedback on the manuscripts.

References

Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020. 2

Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983. 3

Dimitris Bertsimas, David B Brown, and Constantine Caramanis. Theory and applications of robust optimization. *SIAM review*, 53(3):464–501, 2011. 16

Alex Bewley, Jessica Rigley, Yuxuan Liu, Jeffrey Hawke, Richard Shen, Vinh-Dieu Lam, and Alex Kendall. Learning to drive from simulation without real world labels. In *2019 International conference on robotics and automation (ICRA)*, pages 4818–4824. IEEE, 2019. 2

Jacob Buckman, Carles Gelada, and Marc G Bellemare. The importance of pessimism in fixed-dataset policy optimization. In *International Conference on Learning Representations*, 2020. 3

Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8973–8979. IEEE, 2019. 2

Benjamin Eysenbach, Shreyas Chaudhari, Swapnil Asawa, Sergey Levine, and Ruslan Salakhutdinov. Off-dynamics reinforcement learning: Training for transfer with domain classifiers. In *International Conference on Learning Representations*, 2020. 2, 5, 8, 19, 25

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. 7, 20

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021. 2, 3

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596, 2018. 2

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062. PMLR, 2019. 2, 3

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870, 2018. 5, 7

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2020. 3

Ilya Kostrikov, Rob Fergus, Jonathan Tompson, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, pages 5774–5783. PMLR, 2021. 2, 3

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *Deep RL Workshop NeurIPS 2021*, 2021. 2, 3

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pages 11761–11771, 2019. 2, 3, 4

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2020. 2, 3, 6, 8, 19, 20

Alex X Lee, Coline Manon Devin, Yuxiang Zhou, Thomas Lampe, Konstantinos Bousmalis, Jost Tobias Springenberg, Arunkumar Byravan, Abbas Abdolmaleki, Nimrod Gileadi, David Khosid, et al. Beyond pick-and-place: Tackling robotic stacking of diverse shapes. In *5th Annual Conference on Robot Learning*, 2021. 2

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020. 2, 3

Jianxiong Li, Xianyuan Zhan, Haoran Xu, Xiangyu Zhu, Jingjing Liu, and Ya-Qin Zhang. Distance-sensitive offline reinforcement learning. *arXiv preprint arXiv:2205.11027*, 2022. 3

Jinning Li, Chen Tang, Masayoshi Tomizuka, and Wei Zhan. Dealing with the unknown: Pessimistic offline reinforcement learning. In *Conference on Robot Learning*, pages 1455–1464. PMLR, 2022. 3, 19

JG Liao and Arthur Berg. Sharpening jensen's inequality. *The American Statistician*, 2018. 6, 16

Jinxin Liu, Zhang Hongyin, and Donglin Wang. Dara: Dynamics-aware reward augmentation in offline reinforcement learning. In *International Conference on Learning Representations*, 2022. 2

Lennart Ljung. System identification. In *Signal analysis and prediction*, pages 163–173. Springer, 1998. 2

Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021. 7

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. 1

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE, 2018. 2

Aravind Rajeswaran, Sarvjeet Ghotra, Balaraman Ravindran, and Sergey Levine. Epopt: Learning robust neural network policies using model ensembles. *arXiv preprint arXiv:1610.01283*, 2016. 2

Kanishka Rao, Chris Harris, Alex Irpan, Sergey Levine, Julian Ibarz, and Mohi Khansari. Rl-cyclegan: Reinforcement learning aware simulation-to-real. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11157–11166, 2020. 2

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017. 1

Richard S Sutton, Andrew G Barto, et al. Introduction to reinforcement learning. 1998. 3

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017. 2

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012. 7

Quan Vuong, Sharad Vikram, Hao Su, Sicun Gao, and Henrik I Christensen. How to pick the domain randomization parameters for sim-to-real transfer of reinforcement learning policies? *arXiv preprint arXiv:1903.11774*, 2019. 2

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019. 3

Teng Xiao and Donglin Wang. A general offline reinforcement learning framework for interactive recommendation. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI*, volume 2021, 2021. 2

Haoran Xu, Xianyuan Zhan, Jianxiong Li, and Honglei Yin. Offline reinforcement learning with soft behavior regularization. *arXiv preprint arXiv:2110.07395*, 2021. 3

Haoran Xu, Jiang Li, Jianxiong Li, and Xianyuan Zhan. A policy-guided imitation approach for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2022. 2, 3

Haoran Xu, Xianyuan Zhan, and Xiangyu Zhu. Constraints penalized q-learning for safe offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022. 2, 3

Wenhao Yu, Jie Tan, C Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453*, 2017. 2

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. In *Neural Information Processing Systems (NeurIPS)*, 2020. 3, 23

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in Neural Information Processing Systems*, 34, 2021. 3, 19, 23

Xianyuan Zhan, Haoran Xu, Yue Zhang, Xiangyu Zhu, Honglei Yin, and Yu Zheng. Deepthermal: Combustion optimization for thermal power generating units using offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022. 2, 3

Xianyuan Zhan, Xiangyu Zhu, and Haoran Xu. Model-based offline planning with trajectory pruning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022*, pages 3716–3722, 2022. 3

Checklist

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, Conclusions and Perspectives
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (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] See Section 4, 5 and Appendix A
   (b) Did you include complete proofs of all theoretical results? [Yes] See Section 5 and Appendix A
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 B.1
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 6
   (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 B.1
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]
   (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? [Yes]
   (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]

Appendix

A Theoretical Derivation and Analysis of H2O

In this section, we provide the derivation of the closed-form solution to Eq. (5) and the detailed theoretical analysis of H2O as discussed in Section 5. In particular, we first provide an approximated dynamics-aware policy evaluation objective of H2O based on Eq. (6) , after we claim the derivation of the original one, and (8) in the main text, which offers a much cleaner form for theoretical analysis. The approximation can be tight under some reasonable problem setups. Based on the approximated objective, we can show that the dynamics-aware policy evaluation is equivalent to adding an adaptive reward adjustment term to the original MDP. We can further show that this leads to an underestimated value function $V(\mathbf{s})$ in high dynamics-gap areas, which achieves desirable learning behavior under our offline-and-online policy learning setting involving imperfect simulators.

A.1 Derivation of the closed-form solution to $d^\phi(\mathbf{s}, \mathbf{a})$

The Lagrangian of the primal optimization problem in Eq. (5) is given by:

$$\mathcal{L}\left(d^\phi; \mu, \boldsymbol{\lambda}\right) = \mathbf{E}_{\mathbf{s},\mathbf{a}\sim d^\phi} Q(\mathbf{s}, \mathbf{a}) - D_{KL}\left(d^\phi(\mathbf{s}, \mathbf{a}) \| \omega(\mathbf{s}, \mathbf{a})\right) + \mu\left(\sum_{\mathbf{s},\mathbf{a}} d^\phi(\mathbf{s}, \mathbf{a}) - 1\right) + \boldsymbol{\lambda}(\mathbf{s}, \mathbf{a}) d^\phi(\mathbf{s}, \mathbf{a}) \tag{12}$$

where μ is the Lagrangian dual variable for the normalization constraint $\sum_{\mathbf{s},\mathbf{a}} d^\phi(\mathbf{s}, \mathbf{a}) = 1$, and $\boldsymbol{\lambda}(\mathbf{s}, \mathbf{a})$ is the Lagrangian dual variable for positivity constraints on d^ϕ. Setting the gradient of the Lagrangian w.r.t. d^ϕ to 0 yields:

$$d^{\phi *}(\mathbf{s}, \mathbf{a}) = \omega(\mathbf{s}, \mathbf{a}) \exp[Q(\mathbf{s}, \mathbf{a}) + \mu + \boldsymbol{\lambda}(\mathbf{s}, \mathbf{a}) - \mathbf{1}], \forall (\mathbf{s}, \mathbf{a}) \in \mathcal{S} \times \mathcal{A} \tag{13}$$

If we assume the dynamics gap distribution $\omega(s, a) > 0$ holds for all state-action pairs, hence $d^{\phi *}(\mathbf{s}, \mathbf{a}) > 0$ trivially holds, which implies $\boldsymbol{\lambda}(\mathbf{s}, \mathbf{a}) = 0$ for each state-action pair according to the complementary slackness condition. Utilizing the normalization constraint $\sum_{\mathbf{s},\mathbf{a}} d^\phi(\mathbf{s}, \mathbf{a}) = 1$, we have:

$$\sum_{\mathbf{s},\mathbf{a}} d^{\phi *}(\mathbf{s}, \mathbf{a}) = \sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s}, \mathbf{a}) \exp[Q(\mathbf{s}, \mathbf{a})] e^{\mu - \mathbf{1}} = 1 \tag{14}$$

Solving $e^{\mu - 1}$ using Eq. (14) and plugging it into Eq. (13), we then obtain the final closed-form solution for $d^{\phi *}$:

$$d^{\phi *}(\mathbf{s}, \mathbf{a}) = \frac{\omega(\mathbf{s}, \mathbf{a}) \exp[Q(\mathbf{s}, \mathbf{a})]}{\sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s}, \mathbf{a}) \exp[Q(\mathbf{s}, \mathbf{a})]} \propto \omega(\mathbf{s}, \mathbf{a}) \exp[Q(\mathbf{s}, \mathbf{a})] \tag{15}$$

Note that if we plug the exact solution $d^{\phi *}$ and the regularization term $\mathcal{R}(d^\phi) = -D_{KL}(d^\phi || \omega)$ in Eq. (4), we have:

$$\begin{aligned}
&\beta\left[\mathbb{E}_{\mathbf{s},\mathbf{a}\sim d^\phi(\mathbf{s},\mathbf{a})}[Q(\mathbf{s}, \mathbf{a}) - \log(d^\phi(\mathbf{s}, \mathbf{a}) / \omega(\mathbf{s}, \mathbf{a}))] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}[Q(\mathbf{s}, \mathbf{a})]\right] + \widetilde{\mathcal{E}}\left(Q, \hat{\mathcal{B}}^\pi \hat{Q}\right) \\
&= \beta\left[\mathbb{E}_{\mathbf{s},\mathbf{a}\sim d^\phi(\mathbf{s},\mathbf{a})}\left[Q(\mathbf{s}, \mathbf{a}) - \log\left(\frac{\exp[Q(\mathbf{s}, \mathbf{a})]}{\sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s}, \mathbf{a}) \exp[Q(\mathbf{s}, \mathbf{a})]}\right)\right] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}[Q(\mathbf{s}, \mathbf{a})]\right] + \widetilde{\mathcal{E}}\left(Q, \hat{\mathcal{B}}^\pi \hat{Q}\right) \\
&= \beta\left[\mathbb{E}_{\mathbf{s},\mathbf{a}\sim d^\phi(\mathbf{s},\mathbf{a})}\left[\log\left(\sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s}, \mathbf{a}) \exp[Q(\mathbf{s}, \mathbf{a})]\right)\right] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}[Q(\mathbf{s}, \mathbf{a})]\right] + \widetilde{\mathcal{E}}\left(Q, \hat{\mathcal{B}}^\pi \hat{Q}\right) \\
&= \beta\left[\log \sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s}, \mathbf{a}) \exp[Q(\mathbf{s}, \mathbf{a})] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}[Q(\mathbf{s}, \mathbf{a})]\right] + \widetilde{\mathcal{E}}\left(Q, \hat{\mathcal{B}}^\pi \hat{Q}\right)
\end{aligned} \tag{16}$$

In the last step, we can remove $\mathbb{E}_{\mathbf{s},\mathbf{a}\sim d^\phi(\mathbf{s},\mathbf{a})}$ as $\log \sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s}, \mathbf{a}) \exp[Q(\mathbf{s}, \mathbf{a})]$ is a value that does not depend on $(\mathbf{s}, \mathbf{a})$ any more. The objective in the last equation is exactly the objective we have used in Eq. (6). Note that the derivation is based on the exact solution of $d^{\phi *}$ rather than the proportional one.

A.2 Adaptive Reward Adjustment Under Dynamics-Aware Policy Evaluation

The weighted log-sum-exp term in Eq. (6) is quite cumbersome to work with. To draw more insights from H2O as well as to make the analysis simpler, we will derive a reasonably approximate version

of the original dynamics-aware policy evaluation objective. Before presenting the final form, we first introduce the following lemma from [Liao and Berg, 2018]:

Lemma 1. *(A sharpened version of Jensen's inequality [Liao and Berg, 2018]). Let X be a one-dimensional random variable with $P(X \in (a,b)) = 1$, where $-\infty \leq a < b \leq \infty$. Let $\varphi(x)$ be a twice differentiable function on (a,b), we have:*

$$\inf_{x\in(a,b)} \frac{\varphi''(x)}{2} var(X) \leq E[\varphi(X)] - \varphi(E(X)) \leq \sup_{x\in(a,b)} \frac{\varphi''(x)}{2} var(X) \tag{17}$$

Define $[Q_{min}, Q_{max}]$ as the range of learned Q-values (we assume $r > 0$, hence $Q_{min} > 0$), and $\text{Var}_\omega[\exp(Q(\mathbf{s},\mathbf{a}))]$ is the variance of $\exp(Q(\mathbf{s},\mathbf{a}))$ under $(\mathbf{s},\mathbf{a})$ samples drawn from the distribution $\omega(\mathbf{s},\mathbf{a})$. With Lemma 1, we can have the following result on the weighted log-sum-exp term:

Corollary 1. *The weighted log-sum-exp term $\log \sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s},\mathbf{a}) \exp(Q(\mathbf{s},\mathbf{a}))$ can be reasonably approximated by $\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})]$ if $\sqrt{Var_\omega[\exp(Q(\mathbf{s},\mathbf{a}))]}/\exp(Q_{min})$ is small. In particular, following bounds on the weighted log-sum-exp term holds:*

$$\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})] \leq \log \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(\mathbf{s},\mathbf{a})} \exp(Q(\mathbf{s},\mathbf{a})) \leq \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})] + \frac{Var_\omega[\exp(Q(\mathbf{s},\mathbf{a}))]}{2\exp(2Q_{min})} \tag{18}$$

Proof. The LHS inequality is a straightforward result of Jensen's inequality:

$$\log \sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s},\mathbf{a}) \exp(Q(\mathbf{s},\mathbf{a})) = \log \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(\mathbf{s},\mathbf{a})} \exp(Q(\mathbf{s},\mathbf{a})) \geq \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})]$$

The RHS inequality directly follows from Lemma 1 by setting $\varphi(\cdot) = \log(\cdot)$, $x = \exp(Q(\mathbf{s},\mathbf{a}))$ with $(\mathbf{s},\mathbf{a})$ sampled from the dynamics gap distribution $\omega(\mathbf{s},\mathbf{a})$, $a = Q_{min}$ and $b = Q_{max}$:

$$\log \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(\mathbf{s},\mathbf{a})} \exp(Q(\mathbf{s},\mathbf{a})) \leq \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})] + \frac{\text{Var}_\omega[\exp(Q(\mathbf{s},\mathbf{a}))]}{2\exp(2Q_{min})}$$

where we use the relationship $\inf_{x\in(a,b)} \log(x)''/2 = \inf_{x\in(a,b)} -1/2x^2 = \inf_{\exp(Q)\in(\exp(Q_{min}),\exp(Q_{max}))} -1/2\exp(2Q) = -1/2\exp(2Q_{min})$. □

Corollary 1 suggests that $\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})]$ can be a reasonable approximation of the weighted log-sum-exp term in Eq. (6) if $\sqrt{Var_\omega[\exp(Q(\mathbf{s},\mathbf{a}))]}/\exp(Q_{min})$ is relatively small compared to $\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})]$. This can be practically satisfied if we properly design the value range of reward function $r \in [R_{min}, R_{max}]$ and the episode done condition to control the gap between Q_{max} and Q_{min}, as well as let $\gamma \to 1$ to encourage the learned Q function in taking large values.

With above approximation, we instead consider the following approximated policy evaluation objective of H2O based on Eq. (6) and (8) in the main text, which is much easier for our analysis:

$$\begin{aligned}\min_Q \beta \left(\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\omega(\mathbf{s},\mathbf{a})}[Q(\mathbf{s},\mathbf{a})] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}[Q(\mathbf{s},\mathbf{a})]\right) + \frac{1}{2}\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[\left(Q - \hat{\mathcal{B}}^\pi \hat{Q}\right)(\mathbf{s},\mathbf{a})\right]^2 \\ + \frac{1}{2}\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim B}\left[\frac{P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s},\mathbf{a})}{P_{\widehat{\mathcal{M}}}(\mathbf{s}'|\mathbf{s},\mathbf{a})}\left(Q - \hat{\mathcal{B}}^\pi \hat{Q}\right)(\mathbf{s},\mathbf{a})\right]^2\end{aligned} \tag{19}$$

Note that in the original objective of H2O (Eq. (6)), we perform minimization on the weighted soft-maximum of Q-values, whereas in the approximated objective, we are minimizing on the weighted mean of Q-values. Both objectives penalize Q-values with high dynamics gap density ω, but the original objective can be seen as minimizing the worst-case objective as in robust optimization [Bertsimas *et al.*, 2011], which often leads to better results under uncertainty. Additional empirical comparisons between the original H2O policy evaluation objective in Eq. (6) and the approximated version in Eq. (19) are provided in Appendix C.1. Despite the differences, the approximated objective provides a much cleaner form of our analysis, from which we can gain some insights about how H2O works by combining both offline and online learning.

Consider the tabular and approximate dynamic programming setting, by setting the derivative of the approximated objective Eq. (19) with respect to Q^k to zero in iteration k, we have

$$\begin{aligned}
&\beta\left(\omega(\mathbf{s},\mathbf{a}) - d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a})\right) + \mathbb{E}_{\mathbf{s}'} d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a}) P_\mathcal{M}(\mathbf{s}'|\mathbf{s},\mathbf{a}) \left(Q - \hat{\mathcal{B}}^\pi \hat{Q}\right)(\mathbf{s},\mathbf{a}) \\
&\qquad + \mathbb{E}_{\mathbf{s}'} d_{\widehat{M}}^{\pi}(\mathbf{s},\mathbf{a}) P_{\widehat{\mathcal{M}}}(\mathbf{s}'|\mathbf{s},\mathbf{a}) \cdot \frac{P_\mathcal{M}(\mathbf{s}'|\mathbf{s},\mathbf{a})}{P_{\widehat{\mathcal{M}}}(\mathbf{s}'|\mathbf{s},\mathbf{a})} \cdot \left(Q - \hat{\mathcal{B}}^\pi \hat{Q}\right)(\mathbf{s},\mathbf{a}) = 0 \\
\Rightarrow\ &\beta\left(\omega(\mathbf{s},\mathbf{a}) - d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a})\right) + \mathbb{E}_{\mathbf{s}'} P_\mathcal{M}(\mathbf{s}'|\mathbf{s},\mathbf{a}) \left(d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a}) + d_{\widehat{M}}^{\pi}(\mathbf{s},\mathbf{a})\right) \left(Q - \hat{\mathcal{B}}^\pi \hat{Q}\right)(\mathbf{s},\mathbf{a}) = 0 \\
\Rightarrow\ &\hat{Q}^{k+1}(\mathbf{s},\mathbf{a}) = (\hat{\mathcal{B}}^\pi \hat{Q}^k)(\mathbf{s},\mathbf{a}) - \beta \left[\frac{\omega(\mathbf{s},\mathbf{a}) - d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a})}{d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a}) + d_{\widehat{M}}^{\pi}(\mathbf{s},\mathbf{a})}\right] = (\hat{\mathcal{B}}^\pi \hat{Q}^k)(\mathbf{s},\mathbf{a}) - \beta\nu(\mathbf{s},\mathbf{a}) \quad (20)
\end{aligned}$$

where $d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a})$ and $d_{\widehat{M}}^{\pi}(\mathbf{s},\mathbf{a})$ are state-action marginal distributions under behavioral policy $\pi_\mathcal{D}$ and the learned policy π respectively. Note that due to the introduction of the dynamics ratio $P_\mathcal{M}/P_{\widehat{\mathcal{M}}}$ as importance sampling weight, both the Bellman operators for Bellman errors of the offline dataset $\mathcal{D}$ and simulated data B are now defined on the true dynamics $\mathcal{M}$ (i.e., $\hat{\mathcal{B}}^\pi \hat{Q} = \hat{\mathcal{B}}^\pi_\mathcal{M} \hat{Q}$), hence can be combined. This is not possible without $P_\mathcal{M}/P_{\widehat{\mathcal{M}}}$, as we face $\hat{\mathcal{B}}^\pi_\mathcal{M} \hat{Q}$ and $\hat{\mathcal{B}}^\pi_{\widehat{\mathcal{M}}} \hat{Q}$ for the Bellman error of offline data $\mathcal{D}$ and simulated data B respectively.

We can see that $\nu(\mathbf{s},\mathbf{a}) = \frac{\omega(\mathbf{s},\mathbf{a}) - d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a})}{d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a}) + d_{\widehat{M}}^{\pi}(\mathbf{s},\mathbf{a})}$ in Eq.(20) corresponds to an adaptive reward adjustment term, which penalizes or boosts the reward at a state-action pair $(\mathbf{s},\mathbf{a})$ depending on the relative difference between $\omega(\mathbf{s},\mathbf{a})$ and $d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a})$. If $\omega(\mathbf{s},\mathbf{a}) > d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a})$ (high dynamics gap or OOD areas), ν acts as a reward penalty on state-action pair $(\mathbf{s},\mathbf{a})$; otherwise, it serves as a reward boost term to encourage exploration in low dynamics-gap areas. For more discussion on ν, please refer to Section 5 in the main text.

A.3 Lower Bounded Value Estimates on High Dynamics-Gap Samples

In this section, we show that the approximated dynamics-aware policy evaluation of H2O in Eq. (19) learns an underestimated value function on high dynamics-gap areas. We first discuss in Theorem 1, the case under the absence of sampling error, and further incorporate sampling error in Theorem 2 under some mild assumptions. All theoretical analyses are given under tabular settings. In continuous control problems, the continuous state-action space can be approximately discretized into a tabular form, but the tabular form may be large.

Theorem 1. *Assuming no sampling error in the empirical Bellman updates ($\hat{\mathcal{B}}^\pi = \mathcal{B}^\pi$), the value function learned via Eq. (19) lower bounds the actual value function (i.e., $\hat{V}^\pi(\mathbf{s}) \le V^\pi(\mathbf{s})$) in high dynamics-gap data regions, which satisfy $\sum_\mathbf{a} \omega(\mathbf{s},\mathbf{a}) > \sum_\mathbf{a} d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a})\zeta^\pi(\mathbf{s},\mathbf{a})$, with $\zeta^\pi(\mathbf{s},\mathbf{a})$ given as:*

$$\zeta^\pi(\mathbf{s},\mathbf{a}) = \frac{d_M^{\pi_\mathcal{D}}(\mathbf{s}) \max_\mathbf{a}\{\frac{\pi_\mathcal{D}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})}\} + d_{\widehat{M}}^{\pi}(\mathbf{s})}{d_M^{\pi_\mathcal{D}}(\mathbf{s}) \frac{\pi_\mathcal{D}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})} + d_{\widehat{M}}^{\pi}(\mathbf{s})} \ge 1, \quad \forall \mathbf{s},\mathbf{a} \tag{21}$$

Proof. Note that in Eq. (20), for state-action pairs with $\omega(\mathbf{s},\mathbf{a}) > d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a})$, we have potential underestimation on the Q-function. But this condition can be over-restrictive. We derive a more relaxed lower bounded condition for the state-value function $\hat{V}^\pi(\mathbf{s})$.

Taking expectation of Eq. (20) over the distribution $\pi(\mathbf{a}|\mathbf{s})$, we have

$$\begin{aligned}
\hat{V}^{k+1}(\mathbf{s}) &= (\hat{\mathcal{B}}^\pi \hat{V}^k)(\mathbf{s}) - \mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\left[\beta\left(\frac{\omega(\mathbf{s},\mathbf{a}) - d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a})}{d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a}) + d_{\widehat{M}}^{\pi}(\mathbf{s},\mathbf{a})}\right)\right] \\
&= (\hat{\mathcal{B}}^\pi \hat{V}^k)(\mathbf{s}) - \beta \cdot \underbrace{\sum_\mathbf{a} \frac{\omega(\mathbf{s},\mathbf{a}) - d_M^{\pi_\mathcal{D}}(\mathbf{s},\mathbf{a})}{d_M^{\pi_\mathcal{D}}(\mathbf{s}) \frac{\pi_\mathcal{D}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})} + d_{\widehat{M}}^{\pi}(\mathbf{s})}}_{\Delta(\mathbf{s})}
\end{aligned} \tag{22}$$

where $d_M^{\pi_\mathcal{D}}(\mathbf{s})$ and $d_{\widehat{M}}^{\pi}(\mathbf{s})$ are state marginal distributions of behavior policy $\pi_\mathcal{D}$ and the learned policy π. Above condition implies that the value iterates on states with $\Delta(\mathbf{s}) > 0$ will incur some

underestimation, i.e., $\hat{V}^{k+1}(\mathbf{s}) < (\hat{\mathcal{B}}^{\pi}\hat{V}^{k})(\mathbf{s})$. We are interested to find how does this underestimation corresponds to the extent of dynamics gaps on these states. Note that

$$\sum_{\mathbf{a}} \frac{\omega(\mathbf{s},\mathbf{a})}{d_M^{\pi_{\mathcal{D}}}(\mathbf{s})\frac{\pi_{\mathcal{D}}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})} + d_{\widehat{M}}^{\pi}(\mathbf{s})} \geq \frac{\sum_{\mathbf{a}}\omega(\mathbf{s},\mathbf{a})}{d_M^{\pi_{\mathcal{D}}}(\mathbf{s})\max_{\mathbf{a}}\{\frac{\pi_{\mathcal{D}}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})}\} + d_{\widehat{M}}^{\pi}(\mathbf{s})} \tag{23}$$

We can consider a relaxed condition to make $\Delta(\mathbf{s}) > 0$ by enforcing the following relationship:

$$\frac{\sum_{\mathbf{a}}\omega(\mathbf{s},\mathbf{a})}{d_M^{\pi_{\mathcal{D}}}(\mathbf{s})\max_{\mathbf{a}}\{\frac{\pi_{\mathcal{D}}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})}\} + d_{\widehat{M}}^{\pi}(\mathbf{s})} > \sum_{\mathbf{a}} \frac{d_M^{\pi_{\mathcal{D}}}(\mathbf{s},\mathbf{a})}{d_M^{\pi_{\mathcal{D}}}(\mathbf{s})\frac{\pi_{\mathcal{D}}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})} + d_{\widehat{M}}^{\pi}(\mathbf{s})} \tag{24}$$

Above inequality leads to the following condition,

$$\sum_{\mathbf{a}} \omega(\mathbf{s},\mathbf{a}) > \sum_{\mathbf{a}} \left[d_M^{\pi_{\mathcal{D}}}(\mathbf{s},\mathbf{a}) \cdot \frac{d_M^{\pi_{\mathcal{D}}}(\mathbf{s})\max_{\mathbf{a}}\{\frac{\pi_{\mathcal{D}}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})}\} + d_{\widehat{M}}^{\pi}(\mathbf{s})}{d_M^{\pi_{\mathcal{D}}}(\mathbf{s})\frac{\pi_{\mathcal{D}}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})} + d_{\widehat{M}}^{\pi}(\mathbf{s})} \right] = \sum_{\mathbf{a}} d_M^{\pi_{\mathcal{D}}}(\mathbf{s},\mathbf{a})\zeta^{\pi}(\mathbf{s},\mathbf{a}) \tag{25}$$

where $\zeta^{\pi}(\mathbf{s},\mathbf{a})$ is given by Eq. (21). It can be easily observed that $\zeta^{\pi}(\mathbf{s},\mathbf{a}) \geq 1$, for $\forall(\mathbf{s},\mathbf{a})$ and only depends on the offline dataset properties ($\pi_{\mathcal{D}}$, $d_M^{\pi_{\mathcal{D}}}$) as well as the policy properties (π, $d_{\widehat{M}}^{\pi}$). This establishes a condition between the dynamics gap of a state $\sum_{\mathbf{a}}\omega(\mathbf{s},\mathbf{a})$ as well as a threshold characterized only by the offline dataset and the current policy π, $\sum_{\mathbf{a}} d_M^{\pi_{\mathcal{D}}}(\mathbf{s},\mathbf{a})\zeta^{\pi}(\mathbf{s},\mathbf{a})$.

Now, since the exact Bellman operator $\mathcal{B}^{\pi}$ is a contraction mapping, we have:

$$||\mathcal{B}^{\pi}\hat{V}^{k+1} - B^{\pi}\hat{V}^{k}|| = ||(B^{\pi}\hat{V}^{k+1} - \beta\Delta) - (B^{\pi}\hat{V}^{k} - \beta\Delta)|| \leq \gamma||\hat{V}^{k+1} - \hat{V}^{k}|| \tag{26}$$

which suggests that the state value function updates $\hat{V}^{k+1} = B^{\pi}\hat{V}^{k} - \beta\Delta$ in Eq. (22) are also contraction mappings. Based on the contraction mapping theorem, a fixed point $\hat{V}^{\pi}$ exists when we recursively update $\hat{V}^{k}$ using Eq. (22). We can compute the fixed point of the recursion in Eq. (22), and obtain the following estimated policy value:

$$\hat{V}^{\pi}(\mathbf{s}) = V^{\pi}(\mathbf{s}) - \beta\left[(I - \gamma P^{\pi})^{-1}\Delta\right](\mathbf{s}) \tag{27}$$

in which the operator $(I - \gamma P^{\pi})^{-1}$ is positive semi-definite. For high dynamics-gap states $\mathbf{s}$ that satisfy the condition in Eq. (25), we will have $\Delta(\mathbf{s}) > 0$, thus resulting in $\hat{V}^{\pi}(\mathbf{s}) < V^{\pi}(\mathbf{s})$. □

By inspecting the form of $\zeta^{\pi}(\mathbf{s},\mathbf{a})$ in Theorem 1, we can draw some interesting insights. Note that as $\zeta^{\pi}(\mathbf{s},\mathbf{a}) \geq 1$, compared with element-wise condition $\omega(\mathbf{s},\mathbf{a}) > d_M^{\pi_{\mathcal{D}}}(\mathbf{s},\mathbf{a})$, the new condition $\sum_{\mathbf{a}}\omega(\mathbf{s},\mathbf{a}) > \sum_{\mathbf{a}} d_M^{\pi_{\mathcal{D}}}(\mathbf{s},\mathbf{a})\zeta^{\pi}(\mathbf{s},\mathbf{a})$ is more tolerant on simulated samples in terms of their dynamics gaps. Only samples with sufficiently large dynamics gap will lead to underestimated state values. In particular, for state-action pairs with $d_M^{\pi_{\mathcal{D}}}(\mathbf{s},\mathbf{a}) > 0$ and $\pi(\mathbf{a}|\mathbf{s}) \to 0$, even if the dynamics gap $\omega(\mathbf{s},\mathbf{a})$ is large, it will not necessarily lead to underestimated values. This is reasonable as the learned policy $\pi(\mathbf{a}|\mathbf{s})$ is not likely to visit these state-action pairs, the state values need not to be over pessimistically estimated. In OOD regions ($d_M^{\pi_{\mathcal{D}}}(\mathbf{s},\mathbf{a}) = 0$), we will generally obtain underestimated state value functions, and the level of underestimation is proportional to $\sum_{\mathbf{a}}\left[\omega(\mathbf{s},\mathbf{a})/d_{\widehat{M}}^{\pi}(\mathbf{s},\mathbf{a})\right]$. Again, high dynamics-gap OOD samples less visited by the policy will get heavier penalization, while frequently visited low dynamics-gap samples are less impacted. This treatment of H2O is different from most offline RL algorithms. H2O is less conservative and more adaptive with respect to the dynamics gap measures in simulated samples.

To extend the analysis to the setting with sampling error, we first make the following three assumptions. Let $\mathcal{M}$ and $\widehat{\mathcal{M}}$ be the real and simulated MDP, and $\overline{\mathcal{M}}$ be the empirical MDP under the true dynamics, we assume:

Assumption 1. *The dynamics ratio $P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s},\mathbf{a})/P_{\widehat{M}}(\mathbf{s}'|\mathbf{s},\mathbf{a})$ in H2O can be accurately estimated.*

Assumption 2. *The reward function $r(\mathbf{s},\mathbf{a}) \in [0, R_{max}]$ is explicitly defined and only depends on state-action pairs $(\mathbf{s},\mathbf{a})$.*

Assumption 3. *For $\forall \mathbf{s},\mathbf{a} \in \mathcal{M}$, the following relationship of the transition dynamics holds with probability greater than $1-\delta$, $\delta \in (0,1)$:*

$$||P_{\overline{\mathcal{M}}}(\mathbf{s}'|\mathbf{s},\mathbf{a}) - P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s},\mathbf{a})||_1 \leq \frac{C_{P,\delta}}{\sqrt{|\mathcal{D}(\mathbf{s},\mathbf{a})|}} \tag{28}$$

Assumption 2 indicates that the reward does not depend on transition dynamics, which is a mild assumption for many real-world problems. In many cases, we use human designed reward functions based on the currently observed state and action information from the system, rather than using the raw reward signal from a black-box environment. Assumption 3 is a commonly adopted assumption in theoretical analysis of prior works [Kumar *et al.*, 2020; Yu *et al.*, 2021; Li *et al.*, 2022b]. Moreover, as discussed in Section A.2, if the dynamics ratio can be accurately evaluated (Assumption 1), using it as an importance weight will correct the Bellman error, which makes the Bellman operators in Eq. (6) and the approximated version Eq. (19) all defined on the real dynamics $\mathcal{M}$, regardless of whether the training data is from the offline dataset $\mathcal{D}$ or simulated data B (i.e., $\hat{\mathcal{B}}^{\pi}\hat{Q} = \hat{\mathcal{B}}^{\pi}_{\mathcal{M}}\hat{Q}$).

Based on these assumptions, we can show that with high probability $\geq 1-\delta$, the difference between the empirical Bellman operator $\mathcal{B}^{\pi}_{\overline{\mathcal{M}}}$ and the actual Bellman operator $\mathcal{B}^{\pi}_{\widehat{\mathcal{M}}}$ can be bounded as:

$$\begin{aligned} |\mathcal{B}^{\pi}_{\overline{\mathcal{M}}}\hat{V}(\mathbf{s}) - \mathcal{B}^{\pi}_{\mathcal{M}}\hat{V}(\mathbf{s})| &= |(r(\mathbf{s},\mathbf{a}) - r(\mathbf{s},\mathbf{a})) + \gamma \sum_{\mathbf{s}'} (P_{\overline{\mathcal{M}}}(\mathbf{s}'|\mathbf{s},\mathbf{a}) - P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s},\mathbf{a}))\hat{V}(\mathbf{s}')| \\ &= \gamma |\sum_{\mathbf{s}'} (P_{\overline{\mathcal{M}}}(\mathbf{s}'|\mathbf{s},\mathbf{a}) - P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s},\mathbf{a}))\hat{V}(\mathbf{s}')| \leq \frac{\gamma C_{P,\delta} R_{max}}{(1-\gamma)\sqrt{|\mathcal{D}(\mathbf{s},\mathbf{a})|}} \end{aligned} \tag{29}$$

With the above bound, we can introduce the following theorem that incorporates the sampling error:

Theorem 2. *When considering the sampling error, the learned value function via Eq. (19) lower bounds the actual value function at high dynamics-gap states that satisfy the following condition:*

$$\sum_{\mathbf{a}} \omega(\mathbf{s},\mathbf{a}) > \sum_{\mathbf{a}} d^{\pi_{\mathcal{D}}}_{M}(\mathbf{s},\mathbf{a})\zeta^{\pi}(\mathbf{s},\mathbf{a}) + \frac{\gamma C_{P,\delta} R_{max} \cdot \left[d^{\pi_{\mathcal{D}}}_{M}(\mathbf{s}) \max_{\mathbf{a}}\{\frac{\pi_{\mathcal{D}}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})}\} + d^{\pi}_{\widehat{M}}(\mathbf{s}) \right]}{\beta(1-\gamma)\sqrt{|\mathcal{D}(\mathbf{s},\mathbf{a})|}} \tag{30}$$

Proof. Similar to the proof of Theorem 1, when incorporating the sampling error, the fixed point of the recursion in Eq. (22) gives the following result:

$$\hat{V}^{\pi}(\mathbf{s}) \leq V^{\pi}(\mathbf{s}) - \beta\left[(I-\gamma P^{\pi})^{-1}\Delta\right](\mathbf{s}) + \left[(I-\gamma P^{\pi})^{-1}\frac{\gamma C_{P,\delta} R_{max}}{(1-\gamma)\sqrt{|\mathcal{D}(\mathbf{s},\mathbf{a})|}}\right](\mathbf{s}) \tag{31}$$

Following the derivation in Eq.(22)-(25), in order to lower bound the true value function at high dynamics gap states, we need to have

$$\beta \frac{\sum_{\mathbf{a}} \omega(\mathbf{s},\mathbf{a})}{d^{\pi_{\mathcal{D}}}_{M}(\mathbf{s}) \max_{\mathbf{a}}\{\frac{\pi_{\mathcal{D}}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})}\} + d^{\pi}_{\widehat{M}}(\mathbf{s})} > \beta \sum_{\mathbf{a}} \frac{d^{\pi_{\mathcal{D}}}_{M}(\mathbf{s},\mathbf{a})}{d^{\pi_{\mathcal{D}}}_{M}(\mathbf{s})\frac{\pi_{\mathcal{D}}(\mathbf{a}|\mathbf{s})}{\pi(\mathbf{a}|\mathbf{s})} + d^{\pi}_{\widehat{M}}(\mathbf{s})} + \frac{\gamma C_{P,\delta} R_{max}}{(1-\gamma)\sqrt{|\mathcal{D}(\mathbf{s},\mathbf{a})|}} \tag{32}$$

Re-arranging terms in the above inequality, we can easily obtain the final form in Eq. (30). □

Note that under the case with sampling error, value underestimation will occur on simulated samples with even larger dynamics gap as compared to the case without sampling error. To guarantee reliable policy update on these risky data areas, β should be reasonably large to scale down the impact due to the involvement of sampling error.

B Implementation Details and Experiment Setup

B.1 Implementation Details

The implementation details for H2O[4] and other baselines in our experiments are specified as follows:

- **Discriminators**. In H2O, DARC and DARC+, we train two discriminator networks $D_{sas}(\mathbf{s},\mathbf{a},\mathbf{s}')$ and $D_{sa}(\mathbf{s},\mathbf{a})$ to approximate $p(\text{real}|\mathbf{s},\mathbf{a},\mathbf{s}')$ and $p(\text{real}|\mathbf{s},\mathbf{a})$ respectively. We use the activation function of "2×`Tanh`" (soft clip the output values to $[-2,2]$) before the final `Softmax` layer that maps the network outputs into real/simulation domain prediction probabilities. Moreover, we follow the treatment in DARC [Eysenbach *et al.*, 2020] that add the results before the `Softmax` layer of $D_{sa}(\mathbf{s},\mathbf{a})$ to the soft-clipped results in $D_{sas}(\mathbf{s},\mathbf{a})$ to compute the final `Softmax` outputs. This enables $D_{sas}(\mathbf{s},\mathbf{a})$ to propagate gradients back through the $D_{sa}(\mathbf{s},\mathbf{a})$ network, guaranteeing

[4] Our code is available at https://github.com/t6-thu/H2O

the coupling of two discriminators. The training update frequency of the discriminators is aligned with the policy update iterations. Using the discriminator-based dynamics ratio estimation regime in Eq. (7), the KL divergence $u(\mathbf{s}, \mathbf{a})$ between the real and simulated dynamics is approximated in a sample-based manner ($N = 10$ in all the tasks):

$$
\begin{aligned}
u(\mathbf{s}, \mathbf{a}) := D_{KL}\left(P_{\widehat{\mathcal{M}}} \| P_{\mathcal{M}}\right) &\approx \sum_{\mathbf{s}_i' \sim P_{\widehat{\mathcal{M}}}\left(\mathbf{s}_i' | \mathbf{s}, \mathbf{a}\right)}^{N} \log \frac{P_{\widehat{\mathcal{M}}}\left(\mathbf{s}_i' | \mathbf{s}, \mathbf{a}\right)}{P_{\mathcal{M}}\left(\mathbf{s}_i' | \mathbf{s}, \mathbf{a}\right)} \\
&= \sum_{\mathbf{s}_i' \sim P_{\widehat{\mathcal{M}}}\left(\mathbf{s}_i' | \mathbf{s}, \mathbf{a}\right)}^{N} \log \left[\frac{1 - p\left(\text{real} | \mathbf{s}, \mathbf{a}, \mathbf{s}'\right)}{p\left(\text{real} | \mathbf{s}, \mathbf{a}, \mathbf{s}'\right)} / \frac{1 - p\left(\text{real} | \mathbf{s}, \mathbf{a}\right)}{p\left(\text{real} | \mathbf{s}, \mathbf{a}\right)}\right] \\
&= \sum_{\mathbf{s}_i' \sim P_{\widehat{\mathcal{M}}}\left(\mathbf{s}_i' | \mathbf{s}, \mathbf{a}\right)}^{N} \log \left[\frac{1 - D_{\Phi_{sas}}\left(\cdot | \mathbf{s}, \mathbf{a}, \mathbf{s}'\right)}{D_{\Phi_{sas}}\left(\cdot | \mathbf{s}, \mathbf{a}, \mathbf{s}'\right)} / \frac{1 - D_{\Phi_{sa}}\left(\cdot | \mathbf{s}, \mathbf{a}\right)}{D_{\Phi_{sa}}\left(\cdot | \mathbf{s}, \mathbf{a}\right)}\right]
\end{aligned}
\tag{33}
$$

where $P_{\widehat{\mathcal{M}}}(\cdot|\mathbf{s}, \mathbf{a})$ is approximated by $\mathcal{N}(\mathbf{s}', \hat{\Sigma}_{\mathcal{D}})$, and $\hat{\Sigma}_{\mathcal{D}}$ is the covariance matrix of states estimated from the real dataset.

- **Replay buffer size**. For practical considerations, we approximate the log-sum-exp term in Eq. (6) $\log \sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s}, \mathbf{a}) \exp\left(Q(\mathbf{s}, \mathbf{a})\right)$ as well as the dynamics gap distribution $\omega(\mathbf{s}, \mathbf{a}) = u(\mathbf{s}, \mathbf{a}) / \sum_{\tilde{\mathbf{s}},\tilde{\mathbf{a}}} u(\tilde{\mathbf{s}}, \tilde{\mathbf{a}})$ using mini-batch of simulated samples rather than evaluating over the whole state-action space. To achieve a reasonable approximation, the replay buffer should be set relatively large to enable the mini-batch data sampled from the replay buffer close to samples from a uniform distribution defined on state-action space. In both our simulation and real-world experiments, we make replay buffer accommodate **10x** transitions against the offline dataset $\mathcal{D}$.

- **Min Q weight**. H2O uses a fixed value for β in Eq. (6) rather than auto-tuning the min Q weight parameter α (see Eq. (3)) as in the original CQL paper [Kumar *et al.*, 2020] with an additional Lagrange threshold parameter. H2O and H2O(v) only have a single hyperparameter β, and we use only 3 values for β in different experiments (0.01 for all simulation experiments, 0.1 for **Standing Still** and 1.0 for **Moving Straight** in real-world validation). To build a cleaner comparison, we disable some icing-on-the-cake tricks (e.g., auto-tuning min Q weight) in H2O that are inherited from CQL, and so does the CQL baseline. For the CQL baseline, we follow the suggested best configurations in a public CQL implementation[5], and choose α =2.0 for Mujoco experiments and 10.0 by default in the repository to run the real-world tasks. The min Q weights of H2O and CQL are chosen to be higher values in real-world experiments as we find the value regularization terms take small values and do not offer sufficient regularization, probably due to the reasonably good quality of the real dataset.

Other network structure and model training parameters are listed in Table 3. We keep the identical setting in all compatible methods, including the activation function, double-Q function, temperature parameter in SAC and its automatic tuning scheme, etc. Only a few adjustments (i.e. network architecture, batch size) are made different in real-world experiments to accommodate the change in state and action space dimensions in the wheel-legged robot tasks. Generally speaking, H2O needs little hyperparameter tuning when solving different tasks.

B.2 Simulation Experiment Setting

In this section, we explain why we choose the original MuJoCo environment as the target domain and the MuJoCo environment with modified dynamics as the source domain. The primary reason is to use the standard benchmark datasets in D4RL [Fu *et al.*, 2020] for model evaluation. Using the D4RL datasets as real offline data, the evaluated model performance is directly comparable to the vast literature in the area of offline RL. However, if we handcraft a "broken agent" as target task, then it will make the readers hard to compare our model performance with pure offline RL approaches. Second, in most real-world application scenarios, real systems are far more complex than the abstracted simulation environments. Due to complex internal dynamics or unobserved

[5] https://github.com/young-geng/CQL

Table 3: Hyperparameters. "-" denotes the same choice in simulation and real-world tasks

Hyper-parameter	Value (Sim)	Value (Real)
Shared		
Number of hidden layers (Actor and Critic)	2	-
Number of hidden layers (Discriminators)	1	-
Number of hidden units per layer	256	32
Learning rates (all)	3×10^{-4}	-
Discount factor	0.99	-
Nonlinearity	ReLU	-
Nonlinearity (discriminator output layer)	2×Tanh	-
Target smoothing coefficient	5×10^{-3}	-
Batch size	256 / 512	32
Optimizer	Adam	-
H2O & H2O(v)		
KL Divergence clipping range	$[1 \times 10^{-45}, 10]$	-
Dynamics ratio on TD error clipping range	$[1 \times 10^{-5}, 1]$	-
Min Q weight β	0.01	0.1 / 1.0
Replay buffer size	$10 \times \lvert\mathcal{D}\rvert$	-
DARC & DARC+		
Δr clipping range	$[-10, 10]$	-
Replay buffer size	10^6	-
CQL		
Min Q weight α	2.0	10.0
SAC		
Replay buffer size	10^6	-

information, building a high-fidelity simulator can be extremely costly, and even existing the best-effort simulator may still suffer from severe sim-to-real transfer issue. In other words, simulation environments empirically bear flawed dynamics upon the original (real-world) settings.

B.3 Real-World Experiment Setting

We use a real wheel-legged robot for real-world validation of H2O. The control action is the sum of the torque τ of the motors at the two wheels. Each of the motor output the torque of $\frac{\tau}{2}$. The control frequency of the robot is 200Hz. In the follows, we describe our two experiments in detail:

(1) **Standing still**: The state space of the robot is represented by $\mathbf{s} = (\theta, \dot{\theta}, x, \dot{x})$, where θ denotes the forward tilt angle of the body, x is the displacement of the robot, $\dot{\theta}$ and $\dot{x}$ are the angular and linear velocity respectively. We collect a dataset containing 100,000 human controlled transitions of $(\mathbf{s}, a, \mathbf{s}', r, d)$, where $\mathbf{s}$ is the current state, a is the torque of the motor, $\mathbf{s}'$ is the next state, r is the reward and d is the flag of terminal. The dataset is collected near the balanced state. Since we want to keep the robot standing still, the reward r is calculated by the following formulation:

$$r = 30.0 - \theta^2 - \dot{\theta}^2 - x^2 - \dot{x}^2 - \tau^2 \tag{34}$$

When the robot stand still at the position of zero, $-\theta^2 - \dot{\theta}^2 - x^2 - \dot{x}^2$ will reach the maximum value. To prolong the motor life, we add a penalty on torque values τ^2 in the formulation. The constant 30.0 is to keep the reward to be positive. During performance evaluation, we run all algorithms for 50 epochs and report the final results in the main text.

(2) **Moving straight**: The state space of the robot is represented by $\mathbf{s} = (\theta, \dot{\theta}, \dot{x})$, which does not include x of the robot since we only want to keep the velocity of the robot stable. We collect a dataset containing 100,000 human controlled transitions of $(\mathbf{s}, a, \mathbf{s}', r, d)$. The dataset is collected when the robot moves forward. We want to keep the robot keep the target speed of 0.2m/s and the r is calculated by the following formulation:

$$r = 15.0 - (\dot{x} - 0.2)^2 - \tau^2 \tag{35}$$

The speed $\dot{x}$ is penalized from deviating 0.2m/s using $(\dot{x} - 0.2)^2$, and the penalization on torque values τ^2 also remains. We add 15.0 since we want to keep the reward to be positive. To illustrate the

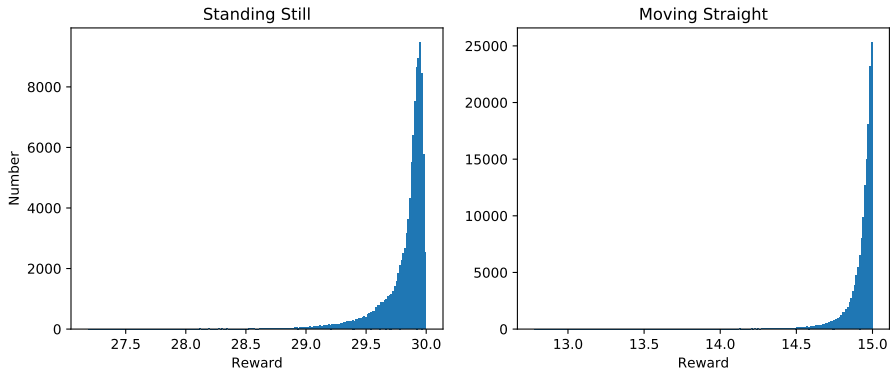

Figure 3: Single-step reward distribution in human-collected datasets of Standing Still and Moving Straight tasks

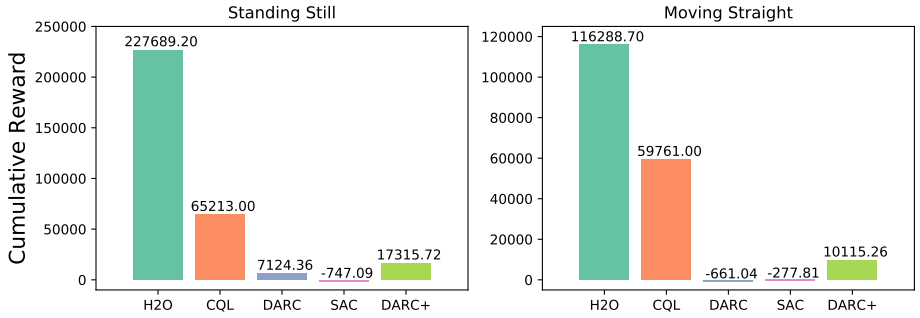

Figure 4: Cumulative rewards of different baselines recorded in real-world validation

human performance of the collected datasets, we visualize the single-reward distribution of both tasks in Figure 3. During performance evaluation, we run all algorithms for 100 epochs and report the final results in the main text. In the real-world validation, we plot the cumulative reward in one recorded episode of each comparative method into Figure 4. In standing still task, an episode is terminated by failing down or standing still for 30 seconds, while failing into the ground or moving forward steadily for 20 seconds in the moving straight task.

C Additional Ablations

C.1 Additional Comparative Results for H2O(v)

In Appendix A.2, we obtain an approximated version for our dynamics-aware policy evaluation in Eq. (19), getting rid of the cumbersome log-sum-exp term. To examine the behavior and performance of this variant (referred to as H2O(v)), we compare it with original H2O in Table 4. All the scores for H2O(v) are averaged over 3 seeds. Based on the empirical results, we find that H2O(v) exhibits similar level of performance as compared with H2O. The original H2O generally outperforms H2O(v), while in a few cases, H2O(v) performs better. As expected, the results also show that H2O(v) is less robust compared with the original H2O in some environments (e.g., HalfCheetah with modified friction coefficient) and produces unstable scores under different datasets. This probably is due to the absence of worst-case optimization as in original H2O, in which the value regularization minimizes the weighted soft maximum of Q-values under highly dynamics-gap samples. These indicate that the approximation scheme used in Eq. (19) can be a reasonable simplification of the original H2O, which trades off some robustness with less computation complexity. To guarantee the best performance, the original H2O should be used in practical deployment.

Table 4: Comparison with H2O(v) on average returns.

Dataset	Unreal dynamics	H2O(v)	H2O	H2O - H2O(v)
Medium	Gravity	**7040±517**	**7085±416**	45
	Friction	5132±2041	**6848±445**	1716
	Joint Noise	**7116±24**	**7212±236**	96
Medium Replay	Gravity	6589±281	**6813±289**	224
	Friction	**6637±456**	5928±896	-709
	Joint Noise	**6822±45**	**6747±427**	-75
Medium Expert	Gravity	**4798±681**	**4707±779**	-91
	Friction	4726±2878	**6745±562**	2019
	Joint Noise	4623±995	**5280±1329**	657

C.2 Additional Experiments for Reverse KL

We tested another variant of H2O which uses a learned dynamics model $P_{\widetilde{\mathcal{M}}}(\mathbf{s}'_i \mid \mathbf{s}, \mathbf{a})$ from offline data as in model-based offline RL methods , and then use the reverse-KL in Eq.(36) to estimate the dynamics gap. In this implementation, we use the deep neural network to learn a probabilistic model $P_{\widetilde{\mathcal{M}}}$ that approximates $P_{\mathcal{M}}$ similar to MOPO [Yu *et al.*, 2020] and COMBO [Yu *et al.*, 2021]. The next state s' is directly sampled from $P_{\widetilde{\mathcal{M}}}$, and we again approximate the dynamics ratio using the same two discriminators. The final performance of this variant does not show noticeable performance improvement, at the extra cost of learning an additional dynamics model. Results are presented in Table 5.

$$u(\mathbf{s}, \mathbf{a}) := D_{KL}\left(P_{\mathcal{M}} \| P_{\widehat{\mathcal{M}}}\right) \approx \sum_{\mathbf{s}'_i \sim P_{\widetilde{\mathcal{M}}}(\mathbf{s}'_i \mid \mathbf{s}, \mathbf{a})}^{N} \log \frac{P_{\widetilde{\mathcal{M}}}(\mathbf{s}'_i \mid \mathbf{s}, \mathbf{a})}{P_{\widehat{\mathcal{M}}}(\mathbf{s}'_i \mid \mathbf{s}, \mathbf{a})} \tag{36}$$

Table 5: Comparison of H2O-KL (original version) and H2O-Reverse-KL.

HalfCheeetah_Gravity	**Medium**	**Medium Replay**	**Medium Expert**
H2O-KL (original version)	7085±416	6813±289	4707±779
H2O-Reverse-KL	7065±170	6476±129	4709±274

C.3 Ablation on Real-World Data Consumption

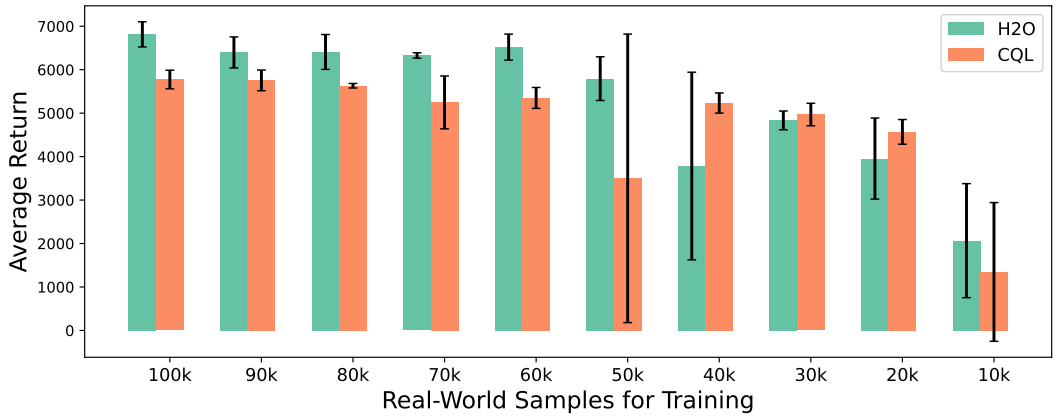

Figure 5: Average return of H2O and CQL with real-world data reduction on HalfCheetah Medium Replay task with modified gravity acceleration. Averaged over 3 seeds.

We ablate the amount of real-world data for comparison between H2O and CQL. We conduct the experiments in HalfCheetah environment with Medium Replay dataset and 'simulation' environment

with two times gravity acceleration for H2O setup. The simulation buffer in H2O keeps the capacity of 1M transitions as the real data consumption declines. The average return is acquired after 1M training steps for all the experiments and implementation details is the same as the aforementioned. As illustrated in Fig. 5, H2O with half of the real data could still exceed the average return that CQL with original dataset yields. Nonetheless, too much reduction on real-world data would deteriorate the dynamics gap quantification and finally have an impact on the performance. Interestingly, H2O with only 10k real-world samples goes against the above analysis probably because the huge quantity of simulation data really compensate the lack of real-world data despite of bearing dynamics gap. To summarize, H2O maintains its competence with moderate reduction on real-world samples involved in training, whereas heavy reduction on real data consumption would lead to substantial performance drop in principle.

D Additional Experiment Results

D.1 Additional Experiments on Walker2d

We further conducted a set of additional experiments on Walker2d with Medium Replay dataset with various types of modified dynamics. Results are presented in Table 6 (averaged over 3 random seeds), with exactly the same hyperparameter setting as that of HalfCheetah tasks in our paper. It is found the H2O achieves the best performance among all other baselines.

Table 6: Average returns for MuJoCo-Walker2d Medium Replay tasks. Averaged over 3 seeds.

Dataset	Unreal Dynamics	SAC	CQL	DARC	DARC+	H2O
Walker2d Medium Replay	**Gravity**	1233±841	1445±1077	1987±965	1618±1446	**2187±1103**
	Friction	2879±569	1445±1077	2518±1244	2375±579	**3656±582**
	Joint Noise	852±386	1445±1077	64±115	630±561	**2998±854**

D.2 Additional Experiments on Random Datasets

We have empirically observed in Table 1 that DARC-style methods struggle in tasks on Medium and Medium Replay datasets. It is somewhat surprising that DARC struggles with low-quality real-world data and even could not outperform CQL, but has a competitive performance on Medium Expert dataset. A possible explanation of this might associate with the limitation in DARC's theoretical derivation, that it derives the dynamics gap-related reward penalty Δr by minimizing the gap between the policy trajectory and the real-world idealized optimal policy π^*. Thus, DARC might unleash more potential over real-world datasets with high quality expert data theoretically. By contrast, H2O is developed under a completely different value regularization framework, without suffering from this problem. To validate the above analyses, we evaluate H2O and baselines on HalfCheetah Random dataset with various types of modified dynamics in Table 7.

It can be observed that DARC-style algorithms indeed perform badly when given low-quality data, due to their theoretical foundation of trajectory distribution divergence minimization. Again, we find H2O performs very well even given the random dataset, which greatly surpasses the performance of pure online or offline baselines. Note that the average Q-values induced by a near-optimal policy on simulation samples could be extremely higher than those on real dataset for its low-quality, the value regularization term would be substantially larger as the learned policy is improved. Consequently, we have to halve the Min Q weight β in these tasks, to scale the value regularization to a reasonable quantity for more stable training.

Table 7: Average returns for MuJoCo-HalfCheetah Random tasks. Averaged over 3 seeds.

Dataset	Unreal Dynamics	SAC	CQL	DARC	DARC+	H2O
HalfCheetah Random	**Gravity**	4513±513	2465±180	357±617	-97±121	**4602±223**
	Friction	2684±2646	2465±180	537±250	425±99	**4862±1608**

D.3 Learning Curves

With all the comparative results in Table 1 and Table 4, we visualize the cumulative returns in the course of training in Fig. 6. It is interesting to note that DARC+ (use both online and offline data for policy evaluation) performs worse in most cases as compared with DARC, and in some cases even fails completely (Gravity and Joint Noise environments under Medium Expert dataset), suggesting the necessity for carefully combining the offline and online learning. It also needs to be emphasized that we accommodate DARC into our offline-and-online setting so it slightly differs from the original pure online setting as in [Eysenbach *et al.*, 2020], in which we do not allow the periodical data collection from real world. Nevertheless, we note from the results that the proposed H2O, and even its simplified version H2O(v) outperform the baseline methods in most of the tasks, which demonstrates the effectiveness of H2O.

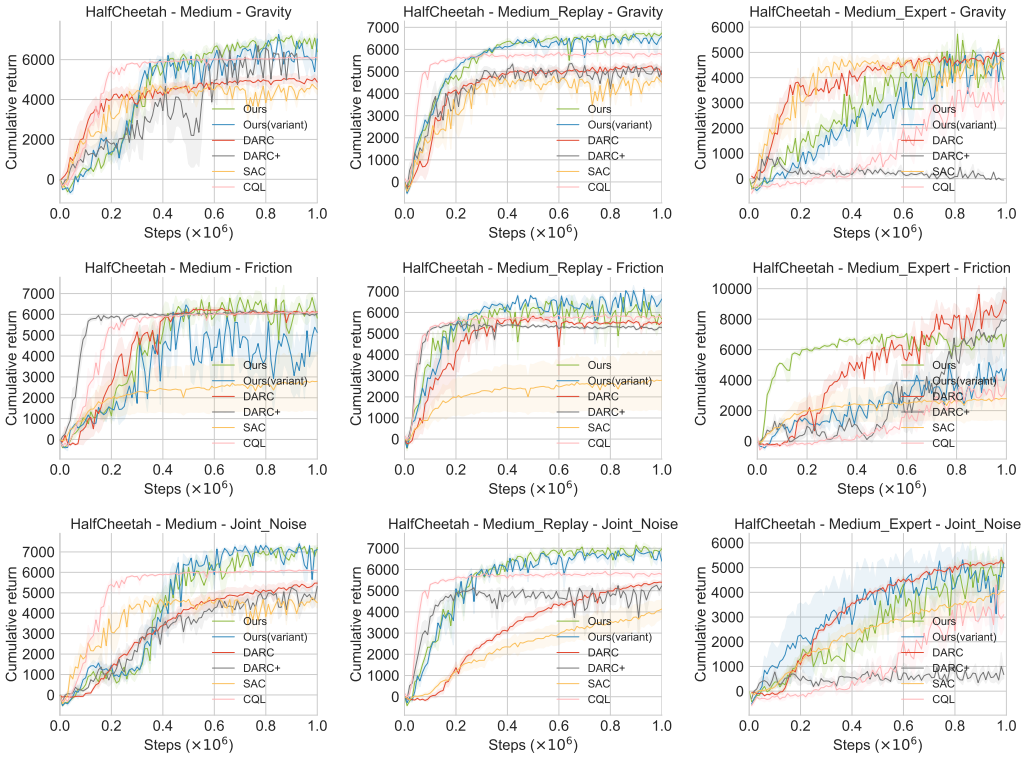

Figure 6: Corresponding learning curves for Table 1