Robust Offline Reinforcement Learning with Gradient Penalty and Constraint Relaxation

Chengqian Gao¹, Ke Xu², Liu Liu², Deheng Ye², Peilin Zhao², Zhiqiang Xu¹,
¹ MBZUAI
² Tencent AI Lab
chengqian.gao@mbzuai.ac.ae
{kaylakxu, leonliuliu, dericye, masonzhao}@tencent.com
zhiqiangxu2001@gmail.com

Abstract

A promising paradigm for offline reinforcement learning (RL) is to constrain the learned policy to stay close to the dataset behaviors, known as policy constraint offline RL. However, existing works heavily rely on the purity of the data, exhibiting performance degradation or even catastrophic failure when learning from contaminated datasets containing impure trajectories of diverse levels, e.g., expert level, medium level, etc., while offline contaminated data logs exist commonly in the real world. To mitigate this, we first introduce gradient penalty over the learned value function to tackle the exploding Q-functions. We then relax the closeness constraints towards non-optimal actions with critic weighted constraint relaxation. Experimental results show that the proposed techniques effectively tame the non-optimal trajectories for policy constraint offline RL methods, evaluated on a set of contaminated D4RL Mujoco and Adroit datasets.

1 Introduction

Effective offline reinforcement learning (RL) should be able to extract policies with the maximum possible utility out of the static demonstrations without interacting with the environment [22][13][24]. One typical way of offline RL is to use policy constraint, enforcing the learned policy to stay close to the behavior policy that generated the dataset, involving various closeness metrics [13][3][40][19].

However, we find many policy constrained offline RL methods suffer performance degradation and even catastrophic failure (please see Figure 1) when trained on datasets containing different levels of policy trajectories. For example, methods in Figure 1 show better performance on the expert dataset while achieving lower scores on the expert-medium dataset. This is undesired as the medium-expert datasets contain more dynamics, i.e., both expert and medium-level data [9].

In fact, many real-world applications demand robust offline RL algorithms, such as robotic controlling tasks with datasets for multiple tasks or incomplete demonstrations [34][9], recommendation tasks with datasets containing non-user logs [15][17], and autonomous driving tasks with trajectories with various levels. In these cases, the dataset contains expert demonstrations and trajectories from non-experts who have not mastered the task [36]. Filtering out non-expert trajectories with human effort is either expensive or impossible, necessitating robust offline RL algorithms.

Figure 1: Performance degradation. We report the paper results of CQL [21], AWAC [29], UWAC [41], TD3+BC [11]. For BEAR-QL [3], we report the result from D4RL [9].

Preprint. Under review.
We integrate the proposed two techniques on the top of BEAR-QL \[3\] and TD3+BC \[11\], attaining why the observed performance degradation and catastrophic failure occur? To answer this question, we first introduce contaminated datasets, which contain trajectories from both expert and non-expert behavior policies, including expert-medium, expert-cloned, and expert-random datasets. By analyzing the learning behaviors on such datasets, we identify two key paths by which non-expert trajectories inhibit policy constraint based offline RL.

First, non-expert data can inhibit policy constraint based offline RL in a supervised manner (see Figure 2 left). The closeness constraint explicitly regles the policy to both expert and non-expert actions, leading to a compromised policy when the states visited by expert and non-expert policies show significant overlaps \[42\]. To tackle this issue, we propose critic weighted constraint relaxation (+ CR), which leverages a polished Q-function to relax the harmful closeness constraint towards non-expert actions.

A more important finding of this work is that non-expert trajectories may destroy the learned Q-function via out-of-distribution (OOD) actions (Figure 2 right). Policy improvements on the contaminated dataset make the learned policy closer to dataset expert actions while moving it away from non-expert decisions. This implicitly leads to the failed closeness constraint on non-expert states when expert and non-expert states follow different distributions. Such failed closeness constraints may result in OOD actions and in turn give rise to unstable Q-values (Theorem 3.1), sharp Q-function gradients, and finally catastrophic failures (Figure 3). We thus introduce the gradient penalty technique (+ GP) to suppress the observed sharp Q-function gradients induced from the failed closeness constraint. To justify the proposed GP technique, we show that there should be an upper bound for the norm of (optimal) Q-function gradients (Theorem 4.1).

We integrate the proposed two techniques on the top of BEAR-QL \[3\] and TD3+BC \[11\], attaining BEAR++ and TD3BC++. Evaluations on the contaminated datasets for D4RL mujoco and adroit contaminated dataset make the learned policy closer to dataset expert actions while moving it away from non-expert actions far aways from expert dataset states, policy improvement may lead to OOD actions. This implicitly leads to the failed closeness constraint on non-expert states far aways from expert dataset states, policy improvement may lead to OOD actions. For example, a Q-function is trained on \[\{ s_t, a_t \}_{t=0}^{N} \] and evaluated on \[\{ s_t, a_t \}_{t=0}^{N} \] with the trajectory distribution \[p_\tau(s_t, a_t) = d_0(s_0) \prod_{t=0}^{N-1} \pi(a_t | s_t) T(s_{t+1} | s_t, a_t).\]

2 Preliminaries

**Offline RL.** Offline RL algorithms aim to obtain policies from static set of interactions, \[D = \{ (s_t, a_t, s_{t+1}, r(s_t, a_t)) \}_{t=0}^{N} \]. One main challenge in offline RL is the distribution shift issues \[10\] or the extrapolation error \[13\] during training. For example, a Q-function is trained on dataset actions \[\mu(a_t | s_t)\] but is evaluated on policy actions \[\pi(a_t | s_t)\]:

\[
Q^{k+1}(s_t, a_t) = \mathbb{E}_{s_t, a_t, r, s_{t+1} \sim D} [ r + \gamma Q^k(s_{t+1}, \pi(a_{s_{t+1}}))] \tag{1}
\]

The learned policy may generate out-of-distribution (OOD) actions that differ from the dataset action since its optimization objective makes no other guarantee except generating high-value actions:

\[
\pi^{k+1} = \arg \max_{\pi} Q(s_t, \pi(s_t)) \tag{2}
\]

Policy improvements implicitly drive the policy to explore OOD actions \[16\], and policy evaluation exploits these OOD actions and in turn affects the policy improvement.
Policy constrained offline RL. One avenue towards offline RL is enforcing the learned policy to stay close to the behavior policy that generated the dataset:

$$\pi^{k+1} = \arg\max_{\pi} Q(s_t, \pi(s_t)), \quad \text{s.t. closeness constraint}$$  \hspace{1cm} (3)

Without loss of generality, we study and try to address the performance degradation and catastrophic failure issues in two policy constraint based offline RL algorithms, TD3+BC [11] and BEAR-QL [3].

TD3+BC adds a behavior cloning term on the top of TD3 [12], resulting in:

$$\pi = \arg\max_{\pi} \mathbb{E}_{(s_t, a_t) \sim D} \left[ \frac{\alpha}{\mathbb{E}[Q(s_t, a_t) - (\pi(s_t) - a_t)^2]} Q(s_t, \pi(s_t)) \right],$$  \hspace{1cm} (4)

where $\alpha$ is a hyperparameter controlling the strength of the regularizer.

BEAR-QL constrains the learned policy to have non-negligible support under the data distribution:

$$\pi = \arg\max_{\pi} \mathbb{E}_{s_t \sim D} \left[ Q(s_t, \pi(s_t)) \right], \quad \text{s.t. } \mathbb{E}_{s_t \sim D} [\text{MMD}(\beta(s_t), \pi(s_t))] \leq \epsilon,$$  \hspace{1cm} (5)

with $\beta$ for approximating the behavior policy and $\epsilon = 0.05$ for a threshold.

3 Catastrophic failure happens with exploding Q-grads

Policy constrained offline RL methods fail to learn meaningful policies on contaminated datasets that contain significantly multi-modal state distributions, e.g., expert-cloned, expert-random. We call this catastrophic failure as it happens with a destroyed Q-function with exploding gradients.

3.1 Catastrophic failures on contaminated datasets

In order to mimic real-world logs that contain multi-level trajectories, we introduce contaminated datasets, which can be generated by contaminating an expert dataset with non-expert demonstrations. For instance, ER3, short for expert-random-30, refers to a dataset in which 70 percent are expert trajectories and 30 percent are from random behavior policies. Please refer to Appendix A for the detailed statistics and discussion about the contaminated datasets.

We run TD3+BC and BEAR-QL on a contaminated dataset, walker2d-expert-random-v0, as depicted in Figure 3. The catastrophic failure occurs after the percentage of random data exceeds 30%.

3.2 Analysis with distribution-constrained Q-iteration

Why do catastrophic failures occur, and always after the learned policy’s performance has improved? In order to give some insights, we use the analysis tools from [3], which involves a constrained Bellman backup operator, defined as:

$$\mathcal{T}^\Pi Q(s_t, a_t) := \mathbb{E}_{\pi \in \Pi} \left[ r + \gamma \max_{\pi} \mathbb{E}_{T(s_{t+1}|s_t, a_t)}[V_\pi(s_{t+1})] \right],$$  \hspace{1cm} (6)

with state value function $V_\pi(s_t) := \mathbb{E}_\pi [Q(s_t, \pi(a_t|s_t))]$ and a $\Pi$ to restrict the set of policies.

Theorem 3.1. The performance of distribution-constrained Q-iteration can be bounded as:

$$\lim_{k \to \infty} \mathbb{E}_{d_0} \left[ V^{\pi^k}(s_t) - V^{\Pi}(s_t) \right] \leq \frac{2\gamma}{(1 - \gamma)^2} C_{1, \mu} \mathbb{E}_\mu \left[ \max_{\pi \in \Pi} \mathbb{E}_\pi [Q(s_t, \pi(a_t))] \right]$$  \hspace{1cm} (7)

with the concentrability coefficient $C_{1, \mu}$ for quantifying how far the conditional distribution of the policy action $\pi(a_t|s_t) \sim \Pi$ is from the corresponding dataset action $\mu(a_t|s_t)$. $V^{\Pi}$ denotes the fixed point of $\mathcal{T}^{\Pi}$, $d_0$ denotes the initial state distribution.
To understand why the catastrophic failure happens, we simplify the concentrability coefficient $C_{11,\mu}^{[28]}$ as the distance between the decisions from $\pi$ and $\mu$.

Before the discussion, we first consider learning from a pure dataset generated by policies with similar decision-making capabilities. The policy improvement implicitly drives the learned policy out of the dataset distribution $[16]$, resulting in a large concentrability coefficient $[3]$ for all dataset states. Policy constraint based offline RL algorithms force the learned policy $\pi \sim \Pi$ close to the behavior policy $\mu$, yielding a low concentrability coefficient and thus making it possible to learn RL policy from static datasets.

For the contaminated dataset with two (or more) behavior policies with significantly different decision-making capacities, policy constrained offline RL faces a dilemma:

**Adhering to the non-expert dataset decisions (low $C_{11,\mu}$) leads to bad policies, but driving out of the non-expert trajectories (large $C_{11,\mu}$) faces OOD actions.**

In this case, behavior policies show different state visitations, as depicted by Figure $[2]$ and $[9]$. Policy improvements may implicitly drive the learned policy to be different from the non-expert dataset actions and thus leads to the failed closeness constraint on non-expert states, resulting in OOD actions and large $C_{11,\mu}$. We visualize this process in Figure $[4]$. The catastrophic failure is correlated with the failed constraint towards non-expert decisions, i.e., the increasing divergence between the policy and non-expert dataset actions. Please note that throughout the learned policy stays close to dataset expert actions.

The closeness constraint on non-expert state-action pairs is destroyed by the policy improvement, which is why it always occurs after achieving good performance (please see Figure $[3]$). The failed closeness constraint leads to OOD actions, which are recognized as the main challenge for offline RL that induces erroneous Q-values, overestimation problems, and bad policies. However, due to the dilemma mentioned above, finding a proper closeness metric for contaminated datasets is hard.

How to save the policy when OOD actions are inevitable? To the best of our knowledge, this work is the first to observe that OOD actions are correlated with extreme sharpness of the Q-values (with respect to actions) and not just overly large values. This inspires us to alleviate the impact of OOD actions from the perspective of gradient regularity.

4 **Recovering from catastrophic failure via gradient penalty**

In this part, we introduce a gradient penalty to minimize the impact of OOD actions induced by the policy improvements on non-expert state-action pairs. Then we give proof to support the proposed gradient penalty. Finally, we discuss the difference between our method and a previous work.

4.1 **Penalizing the unstable gradients**

Recall that the policy improvement step with neural network approximation is:

$$\pi_{\theta}^{k+1} = \arg \max_{\pi} Q(s_t, \pi(s_t))$$

(8)

In practice we run gradient ascent over the parameter space:

$$\theta = \theta + \alpha \cdot \nabla_{\theta} Q(s_t, a_t) \bigg|_{a_t = \pi_{\theta}(s_t)} \cdot \nabla_{\theta} \pi_{\theta}(s_t)$$

(9)

We then recall the reasons for catastrophic failure. The improved policy generates OOD actions on non-expert states (with large $C_{11,\mu}$), loosing the performance bound and resulting in unstable
We introduce a one-sided penalty to encourage the norm of the Q-function gradient w.r.t. non-expert action stays below 1 while avoiding over-punishment for expert alike actions. \( \lambda_{GP} \) controls the contribution of the gradient penalty term. In order to improve computational efficiency, we perform a gradient penalty in every \( N \) training steps (we empirically set \( N \) to 5 in our experiments).

Note that we do not specify the sampling distribution for action \( a \), as we find there is no significant performance difference between the following three sampling strategies: 1) the current policy action \( \sim \pi \), 2) the dataset action distribution \( \sim \mu \), and 3) random sampling over the action space \( a \sim \mathcal{A} \). We will discuss the different motivations behind these choices later.

### 4.2 Lipschitz property of the learned Q-function

To justify the proposed gradient penalty technique, we here prove that the Frobenius norm of the learned Q-function gradients w.r.t. input actions should be bounded.

**Theorem 4.1.** Suppose a policy \( \pi(a_t|s_t) \) on an MDP \( M = (\mathcal{S}, \mathcal{A}, r, \gamma, T) \) satisfies the inequality

\[
\left\| \frac{\partial \pi(a_{t+1}|s_{t+1})}{\partial a_t} \right\|_F \leq L_{\pi,T} < 1
\]
and the reward function \( r(s_t, a_t) \) satisfies \( \left\| \frac{\partial r(s_t, a_t)}{\partial a_t} \right\|_F \leq L_r \). If we denote the dimension of the action space as \( N \), then the magnitude of the gradient of the learned Q-function w.r.t. action can be upperbounded as:

\[
\left\| \nabla_a Q(s_t, a_t) \right\|_F \leq \frac{\sqrt{N}L_r}{1 - \gamma L_{\pi,T}}
\]

**Proof.** See Appendix B.

**Remark 4.2.** Theorem 4.1 holds for offline RL setting as the offline MDP is equal to an modified online MDP with a constrained Bellman backup operator [4]. It tells that the Q-prediction should not vary much for a perturbation in input action, suggesting that the observed exploding Q-function gradients is unreasonable and thus motivates the gradient penalty.

### 4.3 Difference with Fisher-BRC

A keen reader may note that proposed gradient penalty looks similar to the Fisher divergence term in Fisher-BRC [19]:

\[
\mathbb{E}_{s_t \sim \mathcal{D}} \left[ \text{Fisher} \left( \frac{\exp Q(s_t, \cdot)}{\sum_a \exp Q(s_t, a)} \mu(\cdot|s_t) \right) \right] = \mathbb{E}_{s_t \sim \mathcal{D}, a \sim \pi_{emb}(\cdot|s_t)} \left[ \left\| \nabla_a Q(s_t, a) - \nabla_a \log \mu(a|s_t) \right\|_F^2 \right].
\]

However, they are different as 1) Fisher-BRC utilizes gradients to measure the Fisher information distance between the learned policy and the behavior policy \( \mu \), while 2) our method serves to minimize
the negative impact of OOD actions. With different motivations, our method 1) does not require an entropy regularizer for recovering the Boltzmann policy $π_{emb}$, and 2) should be insensitive to the action sampling distribution (while Fisher-BRC needs $a \sim π_{emb}$). Results in Figure 5 verify the second conjecture, in which three types of sampling strategies show no performance difference for expert-random-10 (ER1), ER3, and ER5 settings. We perform gradient penalty w.r.t random actions in the experiments section.

5 Constraint relaxation with the polished Q-function

The harmful closeness constraints toward non-expert dataset actions prevent the learned policy from optima, see also in [42, 32]. We further relax the harmful closeness constraint by critic weighted constraint relaxation (+ CR) in this part.

The key challenge to relaxing the harmful constraints is to indicate the optimality of the dataset actions. In offline RL, the learned Q-function might serve this purpose. As depicted in Figure 6, the polished Q-function could successfully discriminate expert decisions (heavy colors) and random actions (diluted), even when the policy performs not so well (left). On the other hand, without GP, the Q-function is not accurate even if the performance is good (right).

We use the Q-value to indicate the optimality, with a min-max normalization over a mini-batch:

$$W(s_t, a_t) = \frac{Q(s_t, a_t) - Q_{\min}}{Q_{\max} - Q_{\min}}$$

We then could rewrite the regularizer term in BEAR-QL as:

$$\pi = \arg\max_\pi \mathbb{E}_{s_t \sim D}\left[Q(s_t, \pi(s_t))\right] \text{ s.t. } \mathbb{E}_{s_t \sim D}[\text{MMD}(\beta(s_t), \pi(\cdot|s_t))] \cdot W(s_t, \beta(s_t)) \leq \epsilon, \quad (13)$$

and for TD3+BC we have:

$$\pi = \arg\max_\pi \mathbb{E}_{(s_t, a_t) \sim D}[\alpha \mathbb{E}[Q(s_t, a_t)]^{-1} Q(s_t, \pi(s_t)) - (\pi(s_t) - a_t)^2 \cdot W(s_t, a_t)] \quad (14)$$

Note that we do not propagate gradient through the relaxation weight, $W(s_t, a_t)$.

6 Experiments

We proposed two modifications for policy constraint based offline RL: 1) gradient penalty (+ GP) to alleviate the negative impacts of OOD actions induced from the failed closeness constraint and 2) critic weighted constraint relaxation (+ CR) for the harmful closeness constraint, attaining TD3BC++ and BEAR++, on the top of TD3+BC and BEAR-QL.

6.1 Setup

Datasets. We consider three types of contaminated datasets, expert-medium, expert-cloned and expert-random (see Appendix A). For performance on original tasks, please refer to Appendix C.3.
Table 1: Evaluation on the D4RL Mujoco Gym tasks. ER1 is short for Expert-random-10. We rerun all algorithms. With the proposed two techniques, BEAR++ and TD3BC++, could address the performance degradation and catastrophic failures issues. The highest performing scores are bolded.

| Task Setting | BC | %BC | CQL | BEAR-QL | TD3BC | BEAR++ | TD3BC++ |
|--------------|----|-----|-----|---------|-------|--------|---------|
| Expert       | 64.1±2.7 | 69.0±2.5 | 194.4±1.7 | 75.8±1.5 | 104.5±3.0 | 103.3±1.8 | 75.8±2.0 | 64.3±2.9 | 105.6±3.7 |
| Expert-random-10 | 31.3±3.0 | 73.4±1.7 | 104.5±1.4 | 104.3±2.0 | 104.3±1.3 | 104.3±1.3 | 104.3±1.3 | 104.3±1.3 | 104.3±1.3 |
| ER1          | 7.1±3.5 | 2.6±1.1 | 100.1±3.9 | 90.6±1.5 | 88.8±2.9 | 112.9±4.1 | 100.8±10.6 | 6.6±1.4 | 105.2±2.6 |
| ER2          | 8.0±3.1 | 6.0±2.1 | 97.8±1.1 | 9.2±2.2 | 82.5±1.7 | 7.5±1.1 | 97.8±1.1 | 8.9±1.9 | 102.3±0.9 |
| ER3          | 1.6±0.3 | 1.2±0.4 | 93.2±2.1 | 5.2±5.5 | 8.4±1.3 | 18.9±5.9 | 82.5±20 | 4.0±1.4 | 92.2±12.8 |
| Expert-medium | - | - | - | 18.9±10.1 | 58.4±4.0 | 39.2±1.7 | 58.4±17.2 | - | 67.8±29.1 |

Table 2: Evaluation on the D4RL Adroit domain, involving controlling a 24-DoF robotic hand to perform different tasks. EC1 is short for Expert-cloned-10, with cloned trajectories for non-expert behaviors. The highest performing scores are bolded.

| Task Setting | BC | %BC | CQL | BEAR-QL | TD3BC | BEAR++ | TD3BC++ |
|--------------|----|-----|-----|---------|-------|--------|---------|
| Expert       | 104.6±1.1 | 104.6±1.3 | 102.9±0.9 | 104.6±0.5 | 105.7±3.5 | 104.4±5.7 | 44.4±23.5 | 104.6±1.2 | 105.6±1.4 |
| EC3          | 102.3±1.7 | 100.9±0.9 | 109.1±5.1 | 104.6±1.0 | 95.1±0.1 | 95.1±0.1 | 95.1±0.1 | 104.6±1.4 | 105.6±2.5 |
| EC2          | 105.5±0.5 | 101.0±0.5 | - | 102.5±0.0 | 102.5±0.0 | 102.5±0.0 | 102.5±0.0 | 102.5±0.0 | 102.5±0.0 |
| EC7          | 52.5±3.9 | 43.1±0.1 | - | 62.7±0.4 | 62.7±0.4 | 62.7±0.4 | 62.7±0.4 | 62.7±0.4 | 62.7±0.4 |
| Expert-medium | - | - | - | 35.4±1.0 | 35.4±1.0 | 35.4±1.0 | 35.4±1.0 | 35.4±1.0 | 35.4±1.0 |

Evaluation. We train each algorithm for 1 million training time steps, evaluate them every 5000 time steps and finally report the mean and standard deviation of the normalized score [9] over the final 500 episodes (10 trajectories, 10 evaluations and 5 seeds). Please note that 5-seed evaluation is a common setting for offline RL evaluation [3, 10, 11, 24, 25, 33].

Baselines. We compare TD3BC++ and BEAR++ with BC [3] CQL [21], IQL [20], UWAQ [31], Fisher-BRC [19] and original BEAR-QL [3], TD3+BC [11]. We also examine percentile BC and percentile TD3+BC, i.e., run BC and TD3+BC on the top X% of transitions with higher immediate rewards. We set X% to the percentage of expert transitions in each dataset.

6.2 Results and Discussion

Performance degradation (Table 1). The proposed TD3BC++ and BEAR++ show resistance to performance degradation, i.e., agents trained on expert-medium datasets perform as well as expert datasets, for all 3 mujoco gym tasks.

Catastrophic failure (Table 1 and 2). BEAR-QL and TD3+BC suffer from catastrophic failure when learning on datasets contain low-level trajectories, e.g., expert-random and expert-cloned datasets. Fortunately, the proposed methods alleviate the catastrophic failure issues for all 7 tasks. And they could even help TD3+BC perform as well on ER7 as on the expert datasets, in 6 of 7 tasks.

Further penalization onOOD actions (Figure 7). Recall that the failed closeness constraint on non-expert decisions produces OOD policy actions that differ from the dataset actions. And the proposed gradient penalty successfully recovers the Q-function by penalizing the unstable sharp Q gradients. In light of this, we suspect that GP could also contribute to reducing the strength of the required closeness constraint for policy constraint based offline RL.

To investigate this, we run TD3+BC plus GP, changing the hyperparameter α to control the strength of BC term (Equation 14). Note that with α = 1, the agent prefers imitation while with α = 4 for RL.
Figure 7 demonstrates that GP alleviates the dependence on policy constraints for TD3+BC, and thus may save it from degrading to behavioral cloning.

It is not surprising that GP can reduce the dependence on the strength of constraints for policy constraint based offline RL algorithms. Because the failed closeness constraint can be caused not only by the policy improvement on contaminated datasets but also by poor closeness metrics.

**Ablation study (Figure 8).** We ablate the effects of the two proposed techniques when applied individually. For datasets contain many low-level demonstrations (ER3 and ER5 settings), the gradient penalty stabilizes Q-values and prevents complete catastrophic failure, and the constraint relaxation with polished Q-weights brings performance back up to the expert level. For datasets that contain a medium level or a low proportion of random demonstrations (EM and ER1 settings), catastrophic failures do not occur. In this case, constraint relaxation alone is effective, and it performs better in conjunction with a polished Q-function.

**Comparison with the naïve solution (Table 1 and 2).** %BC and %TD3+BC show slight resistance to performance degradation and catastrophic failures. We also set X to X ± 10 and find it performs worse.

**Gradient penalty w.r.t. input states.** We also investigate the effect of gradient penalty w.r.t. input states. However, we discover experimentally that it performs much worse.

**Computational cost comparison.** We train TD3+BC and TD3BC++ agents for 1 million time steps. The wall clock time of TD3+BC is 160m, and 173m for TD3BC++, indicating that the two techniques proposed in this paper are light and efficient plugins for policy constrained offline RL.

7 Related work

**Policy constrained offline RL.** One main approach for offline RL is to enforce the learned policy stay close to the behavior policy, involved with various closeness measurements such as KL-divergence [18], maximum mean discrepancy (MMD) [3], Wasserstein distance [40], Fisher divergence [19] and even Euclidean distance [11]. Closeness constraints could help avoid OOD actions. However, when training on contaminated datasets with non-expert demonstrations, a common setting in real-world applications, these methods show performance degradation and even catastrophic failure in our observation. The proposed two techniques serve to mitigate such issues.

**Value regularization offline RL.** Another offline RL approach is modifying the Q-values to prevent overestimation on OOD actions. This can be achieved by directly penalizing the Q-values of OOD actions in the regression target, e.g., CQL [21], IQL [20], SAC-N [2] or discounting them with uncertainty measurements e.g., UWAC [31], EDAC [2], PBRL [4], RORL [44]. The proposed GP
aims not to prevent the OOD actions but rather to minimize the negative impact (unstable Q-function gradients) of OOD actions caused by the failed closeness constraint on non-expert states.

**Lipschitzness in reinforcement learning.** Our method penalizes the sharp gradient derived from the critic, which is similar to enforcing the learned Q-function to be locally Lipschitz-continuous. Lipschitzness is often used for stabilizing generative adversarial network (GAN) training. It can be achieved by gradient penalty [37], spectral normalization [27], gradient normalization [43], etc. In online RL, [14] use spectral normalization to better the optimization dynamics of the Bellman backups. [23] utilize Lipschitz continuity between MDPs to transfer knowledge for lifelong RL tasks. [26] promotes a local Lipschitz discriminator for robust generative adversarial imitation learning (GAIL) algorithms. Our method aims to minimize the effects of the non-expert trajectories, which carries a different motivation.

**Learning from non-expert trajectories.** This work focus on the influence of non-expert trajectories in the offline RL setting. Similarly, [45] proposes an algorithm to address this issue, assuming clustering methods can recognize transitions from different behavior policies. Besides, [47] consider the task of training policy from datasets with adversarial corruptions. Our method does not rely on such assumptions. In addition, [29] proposes an advantage-based weighting to distinguish expert and non-expert data, which is superior to our CR technique. We leave this issue for further work.

Learning from non-expert data is also a key challenge in imitation learning. Methods in this topic can be mainly divided into two types. Ranking-based methods learn a policy from demonstrations annotated with rankings [11, 5, 6, 8]. Confidence-based methods construct or learn a confidence value function describing the quality of demonstrations and then reweight training samples for imitation [38, 42, 48, 35, 32, 7, 46, 39]. Our method utilizes the learned Q-function to indicate the optimality of the transition, which is close to the confidence-based methods.

### 8 Conclusion

By analysing the learning behaviors on dataset generated by multiple distinct behavior policies, we identify two approaches in which non-expert trajectories inhibit policy constraint offline RL: 1) the harmful closeness constraint towards non-expert actions on overlap states and 2) the failed closeness constraint on non-expert states that causing OOD actions. The proposed CR and GP techniques correspond to these two pathways, respectively, and their effectiveness is empirically evaluated on expert-medium, expert-cloned, and expert-random settings.

The proposed two plugins together extend the applicability of the policy constraint based offline RL to contaminated datasets. Beyond this, the proposed gradient penalty, can help mitigate the negative impacts of OOD actions when the policy constraint fails (on contaminated datasets) or when the constraint has to be weak (to improve over the behavior policies). We hope this finding would encourage researchers to tackle offline reinforcement learning differently than regularizing the Q-values or constraining the learned policy.
References

[1] R. Akrour, M. Schoenauer, and M. Sebag. Preference-based policy learning. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pages 12–27. Springer, 2011.

[2] G. An, S. Moon, J.-H. Kim, and H. O. Song. Uncertainty-based offline reinforcement learning with diversified Q-ensemble. arXiv preprint arXiv:2110.01548, 2021.

[3] K. Aviral, F. Justin, S. Matthew, T. George, and L. Sergey. Stabilizing off-policy Q-learning via bootstrapping error reduction. In Advances in Neural Information Processing Systems, pages 11761–11771, 2019.

[4] C. Bai, L. Wang, Z. Yang, Z. Deng, A. Garg, P. Liu, and Z. Wang. Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning. arXiv preprint arXiv:2202.11566, 2022.

[5] D. Brown, W. Goo, P. Nagarajan, and S. Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In International Conference on Machine Learning, pages 783–792. PMLR, 2019.

[6] D. S. Brown, W. Goo, and S. Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In Conference on robot learning, pages 330–359. PMLR, 2020.

[7] Z. Cao and D. Sadigh. Learning from imperfect demonstrations from agents with varying dynamics. IEEE Robotics and Automation Letters, 6(3):5231–5238, 2021.

[8] L. Chen, R. Paleja, and M. Gombolay. Learning from suboptimal demonstration via self-supervised reward regression. In Conference on Robot Learning, pages 1262–1277. PMLR, 2020.

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

[10] J. Fu, A. Kumar, M. Soh, and S. Levine. Diagnosing bottlenecks in deep Q-learning algorithms. In International Conference on Machine Learning, pages 2021–2030. PMLR, 2019.

[11] S. Fujimoto and S. S. Gu. A minimalist approach to offline reinforcement learning. Advances in Neural Information Processing Systems, 34, 2021.

[12] S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In International Conference on Machine Learning, pages 1587–1596. PMLR, 2018.

[13] S. Fujimoto, D. Meger, and D. Precup. Off-policy deep reinforcement learning without exploration. In International Conference on Machine Learning, pages 2052–2062. PMLR, 2019.

[14] F. Gogianu, T. Berariu, M. Rosca, C. Clopath, L. Busoniu, and R. Pascanu. Spectral normalisation for deep reinforcement learning: an optimisation perspective. In International Conference on Machine Learning, pages 3734–3744. PMLR, 2021.

[15] I. Gunes, C. Kaleli, A. Bilge, and H. Polat. Shilling attacks against recommender systems: a comprehensive survey. Artificial Intelligence Review, 42(4):767–799, 2014.

[16] Y. Hu, Z. Ji, and M. Telgarsky. Actor-critic is implicitly biased towards high entropy optimal policies. arXiv preprint arXiv:2110.11280, 2021.

[17] H. Huang, J. Mu, N. Z. Gong, Q. Li, B. Liu, and M. Xu. Data poisoning attacks to deep learning based recommender systems. arXiv preprint arXiv:2101.02644, 2021.

[18] N. Jaques, A. Ghandeharioun, J. H. Shen, C. Ferguson, A. Lapedriza, N. Jones, S. Gu, and R. Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. arXiv preprint arXiv:1907.00456, 2019.

[19] I. Kostrikov, R. Fergus, J. Tompson, and O. Nachum. Offline reinforcement learning with Fisher divergence critic regularization. In International Conference on Machine Learning, pages 5774–5783. PMLR, 2021.

[20] I. Kostrikov, A. Nair, and S. Levine. Offline reinforcement learning with implicit Q-learning. arXiv preprint arXiv:2110.06169, 2021.

[21] A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative Q-learning for offline reinforcement learning. In Advances in Neural Information Processing Systems, 2020.
[22] S. Lange, T. Gabel, and M. Riedmiller. Batch reinforcement learning. In *Reinforcement learning*, pages 45–73. Springer, 2012.

[23] E. Lecarpentier, D. Abel, K. Asadi, Y. Jinnai, E. Rachelson, and M. L. Littman. Lipschitz lifelong reinforcement learning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, pages 8270–8278. AAAI Press, 2020.

[24] S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[25] Y. Ma, D. Jayaraman, and O. Bastani. Conservative offline distributional reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[26] F. Memarian, A. Hashemi, S. Niekum, and U. Topcu. Robust generative adversarial imitation learning via local Lipschitzness. *arXiv preprint arXiv:2107.00116*, 2021.

[27] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[28] R. Munos and C. Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(5), 2008.

[29] A. Nair, M. Dalal, A. Gupta, and S. Levine. Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[30] D. A. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

[31] E. Rachelson and M. G. Lagoudakis. On the locality of action domination in sequential decision making. In *International Symposium on Artificial Intelligence and Mathematics*, 2010.

[32] F. Sasaki and R. Yamashina. Behavioral cloning from noisy demonstrations. In *International Conference on Learning Representations*, 2020.

[33] S. Sinha, A. Mandlekar, and A. Garg. S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics. In *Conference on Robot Learning*, pages 907–917. PMLR, 2022.

[34] M. Sun and X. Ma. Adversarial imitation learning from incomplete demonstrations. *arXiv preprint arXiv:1905.12310*, 2019.

[35] V. Tangkaratt, B. Han, M. E. Khan, and M. Sugiyama. Variational imitation learning with diverse-quality demonstrations. In *International Conference on Machine Learning*, pages 9407–9417. PMLR, 2020.

[36] V. Tangkaratt and M. Sugiyama. Improving generative adversarial imitation learning with non-expert demonstrations. *OpenReview*, 2018.

[37] H. Thanh-Tung, T. Tran, and S. Venkatesh. Improving generalization and stability of generative adversarial networks. In *International Conference on Learning Representations*, 2019.

[38] Q. Wang, J. Xiong, L. Han, P. Sun, H. Liu, and T. Zhang. Exponentially weighted imitation learning for batched historical data. In *Advances in Neural Information Processing Systems*, pages 6291–6300, 2018.

[39] Y. Wang, C. Xu, B. Du, and H. Lee. Learning to weight imperfect demonstrations. In *International Conference on Machine Learning*, pages 10961–10970. PMLR, 2021.

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

[41] Y. Wu, S. Zhai, N. Srivastava, J. Susskind, J. Zhang, R. Salakhutdinov, and H. Goh. Uncertainty weighted actor-critic for offline reinforcement learning. In *International Conference on Machine Learning*, pages 11319–11328. PMLR, 2021.

[42] Y.-H. Wu, N. Charoenphakdee, H. Bao, V. Tangkaratt, and M. Sugiyama. Imitation learning from imperfect demonstration. In *International Conference on Machine Learning*, pages 6818–6827. PMLR, 2019.

[43] Y.-L. Wu, H.-H. Shuai, Z.-R. Tam, and H.-Y. Chiu. Gradient normalization for generative adversarial networks. In *International Conference on Computer Vision*, pages 6373–6382, 2021.

[44] R. Yang, C. Bai, X. Ma, Z. Wang, C. Zhang, and L. Han. Rorl: Robust offline reinforcement learning via conservative smoothing. *arXiv preprint arXiv:2206.02829*, 2022.
[45] H. Zhang, J. Shao, Y. Jiang, S. He, and X. Ji. Reducing conservativeness oriented offline reinforcement learning. *arXiv preprint arXiv:2103.00098*, 2021.

[46] S. Zhang, Z. Cao, D. Sadigh, and Y. Sui. Confidence-aware imitation learning from demonstrations with varying optimality. *arXiv preprint arXiv:2110.14754*, 2021.

[47] X. Zhang, Y. Chen, J. Zhu, and W. Sun. Corruption-robust offline reinforcement learning. *arXiv preprint arXiv:2106.06630*, 2021.

[48] Z. Zhu, K. Lin, B. Dai, and J. Zhou. Learning sparse rewarded tasks from sub-optimal demonstrations. *arXiv preprint arXiv:2004.00530*, 2020.
Appendices

A The contaminated D4RL datasets

We first provide details about the contaminated D4RL datasets to accommodate reproducibility. Then we give evidence to support the description of the different state overlaps in Figure 2. And finally, we provide some perspectives on the proposed contaminated D4RL datasets.

A.1 Dataset statistics

The contaminated D4RL mujoco gym datasets. Each contaminated dataset contains trajectories from two different levels of policies. We use the D4RL medium-expert datasets for expert-medium settings, which are combinations of expert and medium-level trajectories and are about twice the size of the corresponding expert or medium datasets.

We also contaminate the expert demonstrations with random-level trajectories. For example, ER-1 (short for Expert-random-10) represents a dataset constructed by first loading an expert dataset and then replacing the final 10 percent transitions with tuples from random trajectories (the first 10 percent in the corresponding random dataset). We provide statistics:

| Task | Setting                | Total transition | Expert transition | Random transition | Averaged reward |
|------|------------------------|------------------|-------------------|-------------------|----------------|
| Hopper | Expert-random-10       | 999,034          | 899,131           | 99,903            | 3.53           |
|       | Expert-random-30       | 999,034          | 699,324           | 299,710           | 3.33           |
|       | Expert-random-50       | 999,034          | 499,517           | 499,517           | 3.13           |
|       | Expert-random-70       | 999,034          | 299,711           | 699,323           | 2.93           |
| Walker | Expert-random-10       | 999,304          | 899,374           | 99,930            | 4.25           |
|       | Expert-random-30       | 999,304          | 699,513           | 299,791           | 3.33           |
|       | Expert-random-50       | 999,304          | 499,652           | 499,652           | 2.39           |
|       | Expert-random-70       | 999,304          | 299,792           | 699,512           | 1.45           |
| Halfcheetah | Expert-random-10   | 998,999          | 899,100           | 99,899            | 10.94          |
|       | Expert-random-30       | 998,999          | 699,500           | 299,499           | 8.44          |
|       | Expert-random-50       | 998,999          | 499,500           | 499,499           | 5.95          |
|       | Expert-random-70       | 998,999          | 299,700           | 699,299           | 3.46          |

The contaminated D4RL adroit datasets. The contaminated D4RL Adroit datasets can be constructed in a similar way, except that the non-expert trajectories are from cloned agents, i.e., imitation policies trained from the human-level demonstrations. Statistics of the contaminated D4RL Adroit datasets used in our evaluations are:

| Task | Setting                | Total transition | Expert transition | Cloned transition | Averaged reward |
|------|------------------------|------------------|-------------------|------------------|----------------|
| Door | Expert-cloned-10       | 995,000          | 895,500           | 99,500           | 13.08          |
|      | Expert-cloned-30       | 995,000          | 696,500           | 298,500          | 10.13          |
|      | Expert-cloned-50       | 995,000          | 497,500           | 497,500          | 7.16           |
|      | Expert-cloned-70       | 995,000          | 298,500           | 696,500          | 4.83           |
| Hammer | Expert-cloned-10      | 995,000          | 895,500           | 99,500           | 55.31          |
|       | Expert-cloned-30       | 995,000          | 696,500           | 298,500          | 42.79          |
|       | Expert-cloned-50       | 995,000          | 497,500           | 497,500          | 30.06          |
|       | Expert-cloned-70       | 995,000          | 298,500           | 696,500          | 19.32          |
| Pen  | Expert-cloned-10       | 495,000          | 445,500           | 49,500           | 30.73          |
|      | Expert-cloned-30       | 495,000          | 346,500           | 148,500          | 25.96          |
|      | Expert-cloned-50       | 495,000          | 247,500           | 247,500          | 21.05          |
|      | Expert-cloned-70       | 495,000          | 148,500           | 346,500          | 20.61          |
| Relocate | Expert-cloned-10    | 995,000          | 895,500           | 99,500           | 19.44          |
|      | Expert-cloned-30       | 995,000          | 696,500           | 298,500          | 15.10          |
|      | Expert-cloned-50       | 995,000          | 497,500           | 497,500          | 10.80          |
|      | Expert-cloned-70       | 995,000          | 298,500           | 696,500          | 8.27           |
A.2 Different state overlaps

In Figure 2, we highlight two distinct situations involving different expert and non-expert state overlaps. When states visited by experts show great overlaps with non-expert states, the harmful closeness constraint toward non-expert decisions inhibits. For situation that expert states and non-expert states are well-distinguished, the failed closeness constraint happens as the learned policy is improved, showing different policy actions for dataset non-expert states.

We here provided some visualizations of the distribution of expert and non-expert states in the expert-medium, expert-random, and expert-cloned settings.

![Distribution of states in different D4RL tasks](image)

**Figure 9:** We use UMAP to reduce the dimensionality of states in different D4RL tasks. Expert states are visited by expert behavior policies, and non-expert states are from the medium, random or the cloned policies. We enlarge the dot size of expert states for clarity.

**Great state overlaps.** In this situation, e.g., expert-medium datasets, states visited by expert-level behavior policies show great overlap with that of medium agents. Therefore, the closeness constraint towards non-expert actions may prevent the learned policy from moving closer to the expert decisions. Although offline RL with support-based policy constraints, e.g., BEAR, holds the promise to solve such issues, their exquisite metrics are often difficult to achieve. We alleviate the observed performance degradation by introducing a Q-weight for the policy constraint based method (+CR).

**Less state overlaps.** For datasets contaminated by low-level demonstrations, e.g., random and cloned level data, the expert and non-expert states show greatly different distributions. In this case, policy improvement inevitably changes the policy actions on non-expert states, increasing the probability of generating OOD decisions. This can be dangerous as OOD actions have been widely recognized as the source of exploding value function and the failed learning process. We suppress the OOD actions with the proposed GP technique.

**The success of BC on adroit tasks.** For the simple mujoco tasks (controlling 3 or 6-DoF robotics), states visited by expert policies show great overlap with those visited by non-expert policies. With overlapped states, constraints toward non-expert actions affect the decision quality on expert states. In contrast, such impacts are eliminated with fewer overlaps under the complex Adroit tasks (24-DoF robotics). The records of non-expert state-action pairs less influence the decisions for expert states, thus leading to the success of BC agents on complex Adroit tasks.
A.3 The harmful and the failed closeness constraint.

Figure 10: Visualization of the harmful closeness constraint (left) and the failed closeness constraint (middle and right). Divergence: the 75th percentile of the squared error between the decisions from the learned policy and the corresponding dataset actions.

**The harmful closeness constraint.** When expert and non-expert behavior policies share great state overlaps, two similar dataset states may correspond to two (or more) completely different actions. The closeness constraint towards non-expert one would inhibit the policy improvement in a supervised fashion.

We visualize it in Figure 10 left. The divergence between decisions from the learned policy and the expert behaviors becomes smaller as the policy improves. However, the distance to the non-expert dataset actions also becomes smaller. This contributes to the observed performance degradation. Although ideal support-based policy constraint methods hold the promise to handle this situation, empirically, their performance deteriorates.

**The failed closeness constraint.** The main contribution of this paper is the finding that the policy improvement induces the failed closeness constraint on non-expert dataset states. That is, the policy improvement implicitly drives the learned policy to be different from the decisions recorded for non-expert states, inducing dangerous OOD actions.

In the middle and the right-hand side of Figure 10, we visualize the failed closeness constraint on non-expert dataset states, which happens after the policy achieves a good performance.

A.4 Remarks

**Are expert-random datasets too extreme for mimicking real-life scenarios?** The proposed contaminated dataset can be used to simulate the training behavior on a dataset containing two distinct behavior policies. In this context, what matters is not the non-expert behavior policies’ quality but the states’ overlap between the experts and non-experts.

Such datasets do not necessarily have to be constructed by expert and random policies. For example, the catastrophic failures on expert-cloned datasets, see Table 2, indicate that the learned Q-functions are destroyed by the sharp Q-function gradients, though the cloned behavior policies are far away from randoms.

**Difference with the D4RL replay datasets.** This work focuses on training from contaminated datasets, including three different instances, expert-medium, expert-cloned, and expert-random. Another similar setting is the medium-replay or the full-replay dataset, which records all the interactions during the training. However, we think there is a significant difference between the two settings.

Firstly, the contaminated dataset better fits offline reinforcement learning scenarios. Recall that the primary motivation of offline RL is to avoid the risky interactions for training policy from random initializations. Thus, it is unfeasible to collect logs like medium-replay or full-replay datasets in most situations. On the other hand, the contaminated dataset is used to simulate the training behavior on a dataset where two different behavioral policies exist. We believe a dataset with multiple behavior policies is a really common setting for real-life applications.

Another difference is that the medium-replay and full-replay dataset have a wider distribution of state-action pairs and thus a lower probability of inducing OOD actions than the contaminated dataset considered in this paper. The proportions of expert (medium) trajectories in the replay datasets may also be smaller.
B Lipschitz property of the learned Q-function over action domain

In this section, we provide proof for Theorem 4.1. In order to prove the desired Lipschitz continuity property of the learned Q-function, we need to give an upper bound of the magnitude of the Q-function gradient with respect to the input action, i.e., \( \| \frac{\partial Q(s_t, a_t)}{\partial a_t} \|_F \) is bounded.

For notational clarity, we use \( \mathbb{E}_{s_{t+k}|s_t} \) to denote the expectation of the argument with respect to the conditional distribution of future state \( s_{t+k} \) given that the agent starts from current \( s_t \) and follows the policy \( \pi_t(a_t|s_t) \), i.e., \( \mathbb{E}_{s_t} \left( \prod_{k=0}^{\infty} \pi_t(a_{t+j}|s_{t+j})T(s_{t+j}|s_{t+j}, a_{t+j}) \right) \). Then we can rewrite the learned Q-function \( Q^\pi(s_t, a_t) = \sum_{k=0}^{\infty} \gamma^k \mathbb{E}_{s_{t+k}|s_t} [r(s_{t+k}, a_{t+k})] \). Our proof starts from the upper bound of the Jacobian of the Q-function w.r.t one dimension of the action space. In such a case, we denote the \( i \)-th dimension of the action space as \( a_i^t \). We then drive to the case of multi-dimensional action space and complete our proof.

**Proposition B.1.** Suppose a policy \( \pi \) on an MDP \( M = (S, A, T, d_0, r, \gamma) \) satisfies the following inequality for any given non-negative integer \( t \):

\[
\left\| \frac{\partial \pi(a_{t+1}|s_{t+1})}{\partial a_t} \right\|_F \leq L_{\pi,T},
\]

then it holds for any given non-negative integer \( k \), and \( t \):

\[
\left\| \nabla_{a_i^t} \mathbb{E}_{s_{t+k}|s_t}[r(s_{t+k}, a_{t+k})] \right\| \leq L_{\pi,T} \cdot \mathbb{E}_{s_{t+1}|s_t} \left\| \nabla_{a_i^t} \mathbb{E}_{s_{t+k}|s_{t+1}}[r(s_{t+k}, a_{t+k})] \right\|.
\]

**Proof.**

\[
\left\| \nabla_{a_i^t} \mathbb{E}_{s_{t+k}|s_t}[r(s_{t+k}, a_{t+k})] \right\| = \left\| \nabla_{a_i^t} \mathbb{E}_{s_{t+1}|s_t} \mathbb{E}_{s_{t+k}|s_{t+1}}[r(s_{t+k}, a_{t+k})] \right\| \cdot \frac{\partial a_i^{t+1}}{\partial a_i^t}
\]

\[
\leq \left\| \nabla_{a_i^t} \mathbb{E}_{s_{t+1}|s_t} \mathbb{E}_{s_{t+k}|s_{t+1}}[r(s_{t+k}, a_{t+k})] \right\| \cdot \frac{\partial a_i^{t+1}}{\partial a_i^t}
\]

\[
= \frac{\partial a_i^{t+1}}{\partial a_i^t} \cdot \mathbb{E}_{s_{t+1}|s_t} \left\| \nabla_{a_i^t} \mathbb{E}_{s_{t+k}|s_{t+1}}[r(s_{t+k}, a_{t+k})] \right\|
\]

\[
\leq L_{\pi,T} \cdot \mathbb{E}_{s_{t+1}|s_t} \left\| \nabla_{a_i^t} \mathbb{E}_{s_{t+k}|s_{t+1}}[r(s_{t+k}, a_{t+k})] \right\|
\]

\( \square \)

The above proposition gives a derivation from a mild assumption, which is helpful for our next step proof.

**Proposition B.2.** Suppose a policy \( \pi \) on an MDP \( M = (S, A, T, d_0, r, \gamma) \) satisfies the following inequality for any given non-negative integer \( t \):

\[
\left\| \frac{\partial \pi(a_{t+1}|s_{t+1})}{\partial a_t} \right\|_F \leq L_{\pi,T}
\]

\[
\left\| \frac{\partial r(s_t, a_t)}{\partial a_t} \right\|_F \leq L_r,
\]

then it holds for any given non-negative integer \( t \):

\[
\left\| \nabla_{a_i^t} \mathbb{E}_{s_{t+k}|s_t}[r(s_{t+k}, a_{t+k})] \right\| \leq L_{\pi,T}^k \cdot L_r.
\]
we consider the meaning of \( Q \)-function w.r.t. action can be upperbounded as:

Then we consider the case of multi-dimensional action space. An upper bound formulation of the learned Q-function gradient w.r.t. action can be derived by using Proposition B.1 and Proposition B.2.

**Theorem 4.1** Suppose a policy \( \pi(a_t | s_t) \) on an MDP \( M = \langle S, A, T, d_0, r, \gamma \rangle \) satisfies the inequality 

\[
\left\| \frac{\partial \pi(a_t | s_t)}{\partial a_t} \right\|_F < L_{\pi, T} < 1 \text{ and the reward function } r(s_t, a_t) \text{ satisfies } \left\| \frac{\partial r(s, a_t)}{\partial a_t} \right\|_F \leq L_r. \text{ If we denote the dimension of the action space as } N, \text{ then the magnitude of the gradient of the learned Q-function w.r.t. action can be upperbounded as:}
\]

\[
\left\| \nabla_{a_t} Q^\pi(s_t, a_t) \right\|_F \leq \frac{\sqrt{NL_r}}{1 - \gamma L_{\pi, T}}.
\]

**Proof.**

\[
\left\| \nabla_{a_t} Q^\pi(s_t, a_t) \right\|_F^2 = \sum_{i=0}^{N} \left( \nabla_{a_t} Q^\pi(s_t, a_t) \right)^2
\]

\[
= \sum_{i=0}^{N} \left( \sum_{k=0}^{\infty} \gamma^k \nabla_{a_t} E_{s_{t+k} | s_t} \left[ r(s_{t+k}, a_{t+k}) \right] \right)^2
\]

\[
\leq \sum_{i=0}^{N} \left( \sum_{k=0}^{\infty} \gamma^k \left| \nabla_{a_t} E_{s_{t+k} | s_t} \left[ r(s_{t+k}, a_{t+k}) \right] \right| \right)^2
\]

\[
= \sum_{i=0}^{N} \left( \sum_{k=0}^{\infty} \gamma^k \cdot L_{\pi, T}^k \cdot L_r \right)^2
\]

\[
= N \left( L_r \sum_{k=0}^{\infty} (\gamma L_{\pi, T})^k \right)^2,
\]

finally, we have:

\[
\left\| \nabla_{a_t} Q^\pi(s_t, a_t) \right\|_F \leq \sqrt{NL_r} \sum_{k=0}^{\infty} (\gamma L_{\pi, T})^k
\]

\[
= \frac{\sqrt{NL_r}}{1 - \gamma L_{\pi, T}}.
\]

To better understand the proposed bound (20), we give some perspective on the constants in this formulation. Clearly, \( L_r \) is the Lipschitz constant of the reward function w.r.t. the input action. Then we consider the meaning of \( L_{\pi, T} \cdot \left\| \frac{\partial \pi(a_{t+1} | s_{t+1})}{\partial a_t} \right\|_F \) measures the change in the policy action \( a_{t+1} \) at
next state $s_{t+1}$ if we give an infinitesimal perturbation in the current policy action $a_t$. We denote its upper bound as $L_{\pi,T}$ as the Jacobian is related with the policy \( \pi \) and the environment dynamics \( T \):

$$
\frac{\partial \pi(a_{t+1}|s_{t+1})}{\partial a_t} = \frac{\partial}{\partial a_t} \pi\left(a_{t+1}|T(s_{t+1}|s_t, a_t)\right)
= \frac{\partial T(s_{t+1}|s_t, a_t)}{\partial a_t} \cdot \frac{\partial \pi(a_{t+1}|s')}{\partial s'}_{s'=T(s_{t+1}|s_t, a_t)}
$$

Then we can derive the upper bound of the Jacobian as:

$$
\left\| \frac{\partial \pi(a_{t+1}|s_{t+1})}{\partial a_t} \right\|_F \leq \left\| \frac{\partial T(s_{t+1}|s_t, a_t)}{\partial a_t} \right\|_F \cdot \left\| \frac{\partial \pi(a_{t+1}|s_{t+1})}{\partial s_{t+1}} \right\|_F
$$

The proposed constant $L_{\pi,T}$ is related with two Lipschitz constants, the first one for the policy \( \pi \) w.r.t. the state space and another one for the environment dynamics \( T \) w.r.t. the action space.

We refer the interested readers to \[26\] for the proof of the upper bound for optimal Q-function gradients w.r.t. state space. For the Lipschitz continuity of the value function, see \[31\].

\section{C Experiment details}

We run our experiments on a single machine with 8 RTX3090 GPUs. All D4RL datasets use the v0 version.

\subsection*{C.1 Baselines}

| Hyperparameter | Walker2d | Hopper | Halfcheetah | Door | Hammer | Pen | Relocate |
|----------------|---------|--------|------------|------|--------|-----|---------|
| min_q_weight   | 10      | 20     | 20         | 20   | 50     | -   | -       |

Table 5: Hyperparameter for CQL. We sweep it within the range of \{5, 10, 20, 50, 100\}.

| Hyperparameter | Walker2d | Hopper | Halfcheetah | Door | Hammer | Pen | Relocate |
|----------------|---------|--------|------------|------|--------|-----|---------|
| f_reg          | 1       | 1      | 1          | 5    | 10     | 0.01| 0.1     |

Table 6: Hyperparameter for Fisher-BRC.

- CQL. We use a modular PyTorch implementation of CQL\[1\]. We are very sorry that we cannot reproduce it on Adroit hammer and relocate tasks. To be more specific, for these omitted, the final D4RL normalized scores we got, across all swept paremeters, are about zero (random). We thus have to omit these irrational scores to prevent distress or offense to other readers and authors. Table 5 shows the hyperparameters used in our experiments.

- BEAR-QL. We use the recommended Github implementation\[2\]. We follow the recommended settings for mujoco tasks, and for four Adroit tasks, we use the Gaussian kernel.

- UWAC. We use the official implementation\[3\] with default hyperparameters.

- IQL. We use the authors’ implementation in JAX\[4\] which is really really fast.

- Fisher-BRC. We use the author’s implementation\[5\] we sweep the best hyperparameters for D4RL Adroit expert tasks and follow the suggested settings for D4RL mujoco tasks.

---

1 Code and license: https://github.com/young-geng/cql
2 Code and license: https://github.com/rail-berkeley/d4rl_evaluations
3 Code and license: https://github.com/apple/ml-uwac
4 Code and license: https://github.com/ikostrikov/implicit_q_learning
5 Code and license: https://github.com/google-research/google-research/tree/master/fisher_brc
C.2 The proposed method

Implementation. We recommend interested readers to reproduce results of TD3BC++ on the top of TD3+BC, which is really a minimalist approach to offline RL. The proposed plugin involves two algorithmic modifications:

```python
# Compute critic loss
critic_loss = F.mse_loss(current_Q1, target_Q) + F.mse_loss(current_Q2, target_Q)
+ if self.total_it % N == 0:  # We empirically set N to 5.
+ _state_rep = state.clone().detach().repeat(16, 1).requires_grad_(True)
+ _random_action = torch.rand(size=self.actor(_state_rep).size(),
+ requires_grad=True
+ ) * 2 - 1.0
+ _random_action = _random_action.to(device)
+ _current_Q1, _current_Q2 = self.critic(_state_rep, _random_action)
+ grad_q1_wrt_random_action = torch.autograd.grad(
+ outputs=_current_Q1.sum(),
+ inputs=_random_action,
+ create_graph=True
+ )[0].norm(p=2, dim=-1)
+ grad_q2_wrt_random_action = torch.autograd.grad(
+ outputs=_current_Q2.sum(),
+ inputs=_random_action,
+ create_graph=True
+ )[0].norm(p=2, dim=-1)
+ grad_q_wrt_random_action = F.relu(grad_q1_wrt_random_action - self.k) ** 2 +
+ F.relu(grad_q2_wrt_random_action - self.k) ** 2
+ critic_loss = critic_loss + grad_q_wrt_random_action.mean() * self.lambda_GP
...

# Compute actor loss
- # actor_loss = -lmbda * Q. mean() + F. mse_loss(pi, action)
+ current_Q = ((current_Q1 + current_Q2) * 0.5).squeeze().detach()
+ actor_loss = -lmbda * Q.mean() +
+ (F.mse_loss(pi, action, reduction='none').mean(axis=-1) * current_Q).mean()
```

Listing 1: The proposed two small changes on the top of TD3+BC.

Hyperparameters used for experiments. Our modification involves a weight factor $\lambda_{GP}$ for gradient penalty loss $L_{GP}$. As for the backbone algorithm, TD3+BC, we find $\alpha$, a factor to control the strength of BC term in Equation 4, affects performance the most.\(^8\) Use $\alpha = 2.5$ for their experiments on D4RL mujoco gym tasks. However, we find it does not work for Adroit tasks. We sweep it within the range of $\{0.05, 0.1, 0.2, 0.5, 1, 2, 2.5, 3, 4\}$ and select the maximum possible value that works. Note that, TD3+BC with a low value of $\alpha$ may degenerate to imitation (BC term will dominate the learning) rather than RL. We report the settings used for our experiments:

|                        | Walker2d | Hopper | Halfcheetah | Door | Hammer | Pen | Relocate |
|------------------------|---------|--------|-------------|------|--------|-----|----------|
| TD3+BC                 | $\alpha$ | 2.5    | 2.5         | 2.5  | 0.5    | 0.2 | 0.5      | 0.02     |
| TD3BC++                | $\alpha$ | 2.5    | 2.5         | 2.5  | 0.5    | 0.2 | 0.5      | 0.02     |
| BEAR++                 | $\lambda_{GP}$ | 1     | 1           | 1    | 1      | 1   | 1        | 0.01     |

Table 7: Hyperparameters for TD3+BC, TD3BC++, and BEAR++.

Hyperparameter study. We fix the BC term $\alpha = 2.5$ and vary the gradient penalty term $\lambda_{GP}$ in TD3BC++, sweeping on four different settings. Results are shown in figure 11. For the EM setting, a small GP term ($0.01, 0.02$, or $0.05$) can have a stabilizing effect on training while an overlarge one would inhibit the learned Q-function. As for the difficult ER3, ER5, and ER7 settings, we recommend practitioners choose a medium value ($1, 2, 5$) to stabilize learning while avoiding making the Q-function too flat.

---

6 Code and license: https://github.com/sfujim/TD3_BC
We note that TD3BC++ performs much lower than TD3+BC on the Hopper medium task and returns work will inevitably inherit the social impact of application contexts. As it enjoys many advantages, such as easy implementation, small training costs, and no need for extensive domain knowledge, one can apply it to various scenarios. Therefore, we believe that this work will inevitably inherit the social impact of application contexts.

It would be surprising to see that the proposed plugins alleviate the observed performance degradation and catastrophic failure issues for policy constrained offline RL. With them, one can make greater use of static demonstrations to obtain stronger agents. To this degree, we believe our social impact lies in expanding the applicability of policy constraint based offline reinforcement learning methods.

C.3 Evaluation on D4RL mujoco gym tasks.

This work focuses on addressing the performance degradation and the catastrophic failure issues for policy constraint based offline RL algorithms. Therefore, we are more concerned with the performance on contaminated datasets with non-expert trajectories. In order to show the potential influence of the proposed methods, we report the results of BEAR++ and TD3BC++ on classic D4RL mujoco gym tasks, with hyperparameters in Table 7.

Table 8: Evaluation on the original D4RL mujoco gym tasks.

| Task       | Setting     | BC  | CQL | Fisher-BRC | AWAC | BEAR | TD3+BC | BEAR ++ | TD3BC ++ |
|------------|-------------|-----|-----|------------|------|------|--------|---------|----------|
| Halfcheet  | Expert      | 105.20 | 82.40 | 108.40 | 78.50 | 103.77 | 105.70 | 104.53 | 105.87 |
|           | Medium-expert | 67.60 | 27.10 | 93.30 | 36.80 | 49.25 | 97.90 | 91.01 | 105.26 |
|           | Medium      | 36.60 | 37.20 | 41.30 | 37.40 | 37.09 | 42.80 | 36.85 | 40.78 |
|           | Random      | 2.00  | 21.70 | 33.30 | 2.20  | 2.26  | 10.20 | 2.25  | 6.98   |
| Hopper     | Expert      | 111.30 | 111.20 | 112.30 | 85.20 | 61.50 | 112.20 | 111.36 | 112.23 |
|           | Medium-expert | 89.00 | 111.40 | 112.40 | 89.90 | 85.12 | 112.20 | 110.28 | 111.57 |
|           | Medium      | 30.00 | 44.20 | 99.40 | 72.00 | 37.89 | 99.50 | 39.84 | 30.78 |
|           | Random      | 9.50  | 10.70 | 11.30 | 9.60  | 10.22 | 11.00 | 10.04 | 10.58 |
| Walker2d   | Expert      | 56.00  | 103.80 | 103.00 | 57.00 | 75.13 | 105.70 | 97.20 | 104.68 |
|           | Medium-expert | 12.00 | 68.10 | 105.20 | 42.70 | 56.08 | 101.10 | 74.13 | 104.46 |
|           | Medium      | 11.40 | 57.50 | 78.80 | 30.10 | 57.87 | 79.70 | 62.46 | 75.79 |
|           | Random      | 1.20  | 2.70  | 1.50  | 5.10  | 3.27  | 1.40  | 19.90 | 5.26   |

We note that TD3BC++ performs much lower than TD3+BC on the Hopper medium task and returns to the original performance after reducing the strength of the GP term ($\lambda_{GP} = 0.02$, Score=100.13, 5 seeds). This indicates that one needs to select a suitable $\lambda_{GP}$ value, based on the quality of the dataset and the difficulty of the task.

D Broader impact

Policy constraint based offline RL is a crucial approach to data-driven decision-making machines. As it enjoys many advantages, such as easy implementation, small training costs, and no need for extensive domain knowledge, one can apply it to various scenarios. Therefore, we believe that this work will inevitably inherit the social impact of application contexts.

It would be surprising to see that the proposed plugins alleviate the observed performance degradation and catastrophic failure issues for policy constrained offline RL. With them, one can make greater use of static demonstrations to obtain stronger agents. To this degree, we believe our social impact lies in expanding the applicability of policy constraint based offline reinforcement learning methods.