Supported Trust Region Optimization for Offline Reinforcement Learning

Yixiu Mao [1] Hongchang Zhang [1] Chen Chen [1] Yi Xu [2] Xiangyang Ji [1]

Abstract

Offline reinforcement learning suffers from the out-of-distribution issue and extrapolation error. Most policy constraint methods regularize the density of the trained policy towards the behavior policy, which is too restrictive in most cases. We propose Supported Trust Region optimization (STR) which performs trust region policy optimization with the policy constrained within the support of the behavior policy, enjoying the less restrictive *support constraint*. We show that, when assuming no approximation and sampling error, STR guarantees strict policy improvement until convergence to the optimal support-constrained policy in the dataset. Further with both errors incorporated, STR still guarantees safe policy improvement for each step. Empirical results validate the theory of STR and demonstrate its state-of-the-art performance on MuJoCo locomotion domains and much more challenging AntMaze domains.

1. Introduction

Offline Reinforcement Learning (RL) aims to learn a policy from a fixed dataset without further interactions. It can utilize existing large-scale datasets to learn safely and efficiently (Gulcehre et al., 2020; Fu et al., 2020). However, this also carries with it a major challenge: the evaluation of out-of-distribution (OOD) actions causes extrapolation error (Fujimoto et al., 2019) and overestimation.

Policy constraint methods try to address this by constraining the learned policy to be close to the behavior policy (Wu et al., 2019; Kumar et al., 2019; Wang et al., 2020; Fujimoto & Gu, 2021). Among them, Weighted Behavior Cloning (WBC) performs behavior cloning on the dataset, but assigns different weights to different data points to distill a better policy (Chen et al., 2020; 2021). A common practice is to set the weight as the exponentiated advantage function, leading to Exponentiated Advantage-Weighted Behavior Cloning (EAWBC) (Wang et al., 2018; Peng et al., 2019; Nair et al., 2020; Wang et al., 2020; Siegel et al., 2020). Mathematically, EAWBC is equivalent to a policy improvement step in RL with a KL constraint towards an implicit baseline policy from which the imitated actions are sampled (i.e. behavior policy). As a result, the policy of existing EAWBC methods is implicitly constrained by a *density constraint*, which is typically too restrictive to achieve good performance both theoretically and empirically (Kumar et al., 2019). As a simple example, when the dataset contains a small proportion of the optimal behavior, density constraint will lead to a sub-optimal policy.

On the other hand, from the perspective of optimization process, it is desirable that an RL algorithm can have the *safe policy improvement* guarantee, i.e., have a worst-case performance degradation bound for *each* policy update step. Benefit from this property, online trust region methods (Schulman et al., 2015; 2017) have shown supreme performance on both discrete and continuous tasks (Duan et al., 2016). However, in the offline setting with approximation and sampling error, few existing offline RL algorithms can ensure safe policy improvement for each step.

In this paper, we aim to address the above issues by proposing Supported Trust Region optimization (STR) based on EAWBC. STR performs trust region policy optimization with the policy constrained within the support of the behavior policy β. This less restrictive *support constraint* allows to seek the best behavior in the dataset and is usually sufficient to mitigate the extrapolation error. We start from an observation that the analytical policy update form of EAWBC is an equal-support update, which means the updated policy has the same support as the baseline policy. Based on it, STR utilizes importance sampling on the dataset to mimic sampling from the current policy, which makes the implicit baseline policy in EAWBC become the projection of the current policy on β. In this way, by initializing with an estimated behavior policy and adopting a relatively strong policy constraint, the policy of STR is able to deviate from β step by step with trust region updates, while still satisfying the support constraint to mitigate extrapolation error. In this process, STR gradually "sharpens" the action

[1]Department of Automation, Tsinghua University [2]School of Artificial Intelligence, Dalian University of Technology. Correspondence to: Yixiu Mao <myx21@mails.tsinghua.edu.cn>, Xiangyang Ji <xyji@tsinghua.edu.cn>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

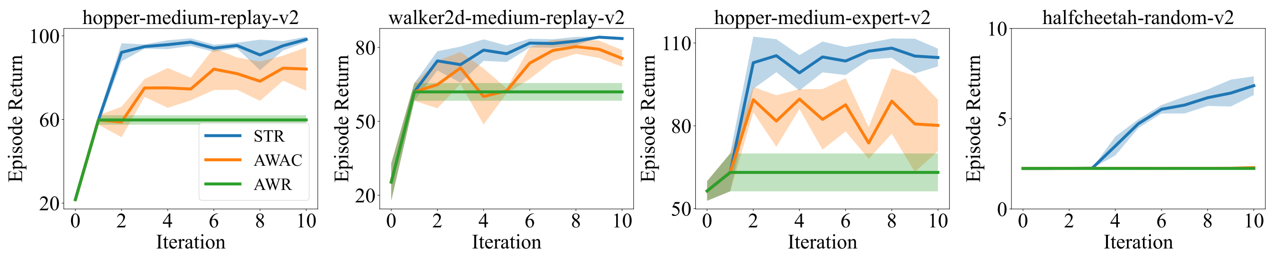


Figure 1. Experimental Verification of Theories. Both policy evaluation and policy improvement are trained to convergence at each iteration, and all algorithms adopt the same hyperparameter that controls the constraint strength. STR based on supported trust region update enjoys safe policy improvement for each iteration, and with a less restrictive *support constraint*, the final performance is better. By contrast, both one-step and multi-step EAWBC methods (AWR and AWAC) implicitly satisfying a *density constraint* to the behavior policy have sub-optimal performance. In halfcheetah-random, AWR and AWAC can hardly make a visible improvement over the behavior policy. Also, AWAC cannot guarantee safe policy improvement (most stark in walker2d-medium-replay and hopper-medium-expert).

distribution of β, giving higher weights to better actions, until it converges to the optimal action in the support of β, while prior EAWBC methods can only get a sub-optimal policy due to the implicit density constraint.

Theoretically, STR enjoys stronger guarantees. Under the same assumptions as prior EAWBC works (exact tabular Q) (Nair et al., 2020; Wang et al., 2020), STR guarantees strict policy improvement until convergence to the optimal support-constrained policy, exceeding the prior non-decreasing results that also have no performance guarantee at convergence. With Q-function approximation and sampling error, prior EAWBC works lose guarantees, while STR still ensures safe policy improvement for each step.

Empirically, we test STR on D4RL benchmark (Fu et al., 2020), including Gym-MuJoCo locomotion domains and much more challenging AntMaze domains. STR consistently outperforms state-of-the-art baselines and outperforms prior EAWBC methods by a large margin. We also conduct a validation experiment and confirm the theoretical superiority of STR, including the less restrictive support constraint and safe policy improvement (Figure 1). Besides, compared with prior EAWBC methods, STR maintains good performance over a wide range of the hyperparameter that controls the policy constraint strength.

2. Preliminaries

RL. In RL, the environment is typically assumed to be a Markov Decision Process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma, d_0)$, with state space $\mathcal{S}$, action space $\mathcal{A}$, transition dynamics $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, reward function $R : \mathcal{S} \times \mathcal{A} \to [0, R_{\max}]$, discount factor $\gamma \in [0, 1)$, and initial state distribution d_0 (Sutton & Barto, 2018). An agent interacts with the MDP according to a policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$. The goal of the agent is to find a policy that maximizes the expected discounted return: $\eta(\pi) = \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty} \gamma^t r_t]$, with $r_t = R(s_t, a_t)$. Here τ denotes a trajectory $(s_0, a_0, r_0, s_1, \dots)$ and $\tau \sim \pi$ is shorthand for indicating the distribution of τ depends on π: $s_0 \sim d_0, a_t \sim \pi(\cdot|s_t), s_t \sim \mathcal{P}(\cdot|s_t, a_t)$. For any policy π, we define the value function as $V^\pi(s) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s]$ and the state-action value function (Q-value function) as $Q^\pi(s, a) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a_0 = a]$. By the boundedness of rewards, we have $0 \le Q^\pi, V^\pi \le \frac{R_{\max}}{1-\gamma} =: V_{\max}$. The advantage function is $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$. For a policy π, the Bellman operator $\mathcal{T}^\pi$ is defined as $(\mathcal{T}^\pi f)(s, a) := R(s, a) + \gamma \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)} [f(s', \pi(s'))]$, where $f(s', \pi(s')) := \mathbb{E}_{a'\sim\pi(\cdot|s')} f(s', a')$. In addition, we use d_t^π to denote the state occupancy of the policy π at time step t: $d_t^\pi(s) := \mathbb{E}_\pi [\mathbb{I}[s_t = s]]$, and use d^π to denote the normalized and discounted state occupancy: $d^\pi(s) = (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t d_t^\pi(s)$. We also define the state-action occupancy with $\rho_t^\pi(s, a) = d_t^\pi(s)\pi(a|s)$ and $\rho^\pi(s, a) = d^\pi(s)\pi(a|s)$.

Offline RL. In offline RL, the agent is provided with a fixed dataset $\mathcal{D}$ collected by some behavior policy β. We define $\mathcal{M}_\mathcal{D}$ as the empirical MDP induced by the dataset $\mathcal{D}$ that uses the empirical transition model based on data counts. Ordinary approximate dynamic programming methods evaluate policy π by minimizing temporal difference error (Haarnoja et al., 2018), according to the following loss

$$L_Q(\theta) = \mathbb{E}_{(s,a,s')\sim\mathcal{D}}[(Q_\theta(s, a) - R(s, a) - \gamma \mathbb{E}_{a'\sim\pi_\phi(\cdot|s')} Q_{\theta'}(s', a'))^2], \quad (1)$$

where π_ϕ is a policy parameterized by ϕ, $Q_\theta(s, a)$ is a Q function parameterized by θ, and $Q_{\theta'}(s, a)$ is a target network whose parameters are updated via Polyak averaging.

Besides policy evaluation, a typical policy iteration also includes policy improvement. In continuous action space, a stochastic policy can be updated by reparameterization:

$$\phi \leftarrow \operatorname{argmax}_\phi \mathbb{E}_{s\sim\mathcal{D},\epsilon\sim\mathcal{N}(0,1)} [Q_\theta (s, f_\phi (\epsilon; s))] \quad (2)$$

In offline RL, OOD actions a' can produce erroneous values for $Q_{\theta'}(s', a')$ and lead to an inaccurate estimation of Q-values. Then in policy improvement, where the policy is optimized to maximize the estimated Q_θ, the policy will prefer OOD actions whose values have been overestimated, resulting in poor performance. To address this issue, WBC methods avoid explicitly maximizing Q, but imitate the actions with high Q-values from $\mathcal{D}$ to improve π.

3. A Unified Framework for EAWBC Works

In this section, we propose a unified framework for all prior EAWBC methods and point out their limitations. Later in Section 4, we will present our proposed algorithm based on this framework. All derivations of this section could be found in Appendix C.

The unified framework follows a policy iteration paradigm. At the i^{th} iteration, after evaluating some policy π_{pe} to get an advantage estimate $\hat{A}^{\pi_{\text{pe}}}$, it solves the following constrained optimization problem to update the policy, where π_{base} is some baseline policy:

$$\begin{aligned} \pi_{i+1} &= \underset{\pi}{\operatorname{argmax}} \underset{a\sim\pi}{\mathbb{E}}[\hat{A}^{\pi_{\text{pe}}}(s,a)] \\ s.t.\ \mathrm{D}_{\mathrm{KL}}(\pi\|\pi_{\text{base}})[s] &\le \epsilon, \quad \textstyle\sum_a \pi(a|s) = 1,\ \forall s \end{aligned} \tag{3}$$

The optimization problem above has a closed-form solution:

$$\begin{aligned} \pi_{i+1}(a|s) =& \pi_{\text{base}}(a|s) f(s,a;\pi_{\text{pe}}) \\ \text{where } f(s,a;\pi_{\text{pe}}) :=& \tfrac{1}{Z(s)} \exp\left(\tfrac{\hat{A}^{\pi_{\text{pe}}}(s,a)}{\lambda^*(s)}\right) \end{aligned} \tag{4}$$

Here $Z(s) = \sum_a \pi_{\text{base}}(a|s)\exp\left(\frac{\hat{A}^{\pi_{\text{pe}}}(s,a)}{\lambda^*(s)}\right)$ is the per-state normalizing factor, and $\lambda^*(s)$ is the solution of the following convex dual problem:

$$\min_{\lambda\ge 0}\ \epsilon\lambda + \lambda \log\left[\textstyle\sum_a \pi_{\text{base}}(a|s)\exp\left(\frac{\hat{A}^{\pi_{\text{pe}}}(s,a)}{\lambda}\right)\right] \tag{5}$$

In practice, the non-parametric solution in Eq.(4) can be projected onto the parametric policy class by minimizing the KL divergence:

$$\underset{\phi}{\operatorname{argmin}} \underset{s\sim\mathcal{D}}{\mathbb{E}} [\mathrm{D}_{\mathrm{KL}}(\pi_{i+1}(\cdot|s)\|\pi_\phi(\cdot|s))] \tag{6}$$

By the relationship between π_{i+1} and π_{base} in Eq.(4), Eq.(6) is equivalent to maximizing the following unified EAWBC objective:

$$J_U(\phi) = \underset{s\sim\mathcal{D},a\sim\pi_{\text{base}}}{\mathbb{E}} [f(s,a;\pi_{\text{pe}})\log(\pi_\phi(a|s))] \tag{7}$$

All prior EAWBC methods except ABM (Siegel et al., 2020) adopt the behavior policy β as π_{base} so that they can sample state-action pairs directly from $\mathcal{D}$ to optimize $J_U(\phi)$. Their main difference is what policy is selected as the evaluation policy π_{pe}. Typically, AWR (Peng et al., 2019) and MARWIL (Wang et al., 2018) use β as π_{pe}, while AWAC (Nair et al., 2020) and CRR (Wang et al., 2020) use the current policy π_i as π_{pe}. Therefore, AWR and MARWIL can be classified as one-step methods that simply perform one step of policy improvement using an on-policy estimate Q^β (Brandfonbrener et al., 2021), while AWAC and CRR belong to multi-step methods that repeatedly evaluate off-policy Q^π.

However, due to the correspondence between the constrained optimization problem (Eq.(3)) and the WBC problem (Eq.(7)), when choosing β as π_{base}, the maximization of $J_U(\phi)$ implicitly regularizes π's density towards β. Density constraint is overly restrictive in many cases (Kumar et al., 2019). With a small or moderate ϵ, due to this implicit KL constraint, the learned policy may be highly sub-optimal. The following lemma confirms this statement.

Lemma 3.1. *If* $\mathrm{D}_{\mathrm{KL}}(\pi(\cdot|s)\|\beta(\cdot|s)) \le \epsilon, \forall s$ *is guaranteed, then the performance* η *has the following bound*

$$\eta(\pi) \le \eta(\beta) + \tfrac{V_{\max}}{\sqrt{2}(1-\gamma)}\sqrt{\epsilon} \tag{8}$$

To be less pessimistic, ϵ needs to be large and $\lambda^*(s)$ in Eq.(4) will be close to 0. Then when $\hat{A}^{\pi_{\text{pe}}}$ is estimated poorly, this will lead to a large and catastrophic policy update (Duan et al., 2016). We point out this trade-off between optimality and stability is the main reason that prior methods output a sub-optimal policy in practice. Our experiments in Section 5.3 support this claim.

To relax this implicit density constraint a bit, ABM (Siegel et al., 2020) first learns a policy π_{abm} at iteration i by maximizing $J_U(\phi)$ with $\pi_{\text{base}} = \beta, \pi_{\text{pe}} = \pi_i$. Then it obtains the next iterate π_{i+1} by maximizing $J_U(\phi)$ again with $\pi_{\text{base}} = \pi_{abm}, \pi_{\text{pe}} = \pi_i$. In this way, ABM intuitively relaxes one density constraint to two coupled ones. However, it is still a density constraint method in essence, and the second optimization needs to imitate the actions from π_{abm} rather than the dataset, which will bring extrapolation error as we find empirically in Section 5.

From a theoretical perspective, the guarantees of prior EAWBC works are not strong. It is proved in Wang et al. (2020) that when assuming exact tabular Q estimation, the multi-step methods (CRR and AWAC) lead to a non-decreasing Q. However even with this strong assumption, these works cannot prove strict monotonicity and have no performance guarantee at convergence. On the other hand, when considering sampling error and Q-function approximation, we always get an approximated and inaccurate Q, which is the main reason why offline RL algorithms suffer from extrapolation error and overestimation. However, no prior EAWBC works analyze this important setting. We find that with an approximate Q, only one-step EAWBC methods (AWR, MARWIL) can ensure safe policy improvement for that only one step, and those multi-step methods fail to guarantee similar results. We summarize in Table 1

Table 1. Performance improvement guarantees of EAWBC methods in offline RL.

Method	π_{pe}	π_{base}	Step	Performance Guarantees	
				w/o sampling and Q approx. error	w/ sampling and Q approx. error
AWR MARWIL	β	β	One-step	*Strictly* increasing for *one-step* w/o final performance guarantee	Safe improvement for *one-step*
AWAC CRR	π_i	β	Multi-step	*Non-decreasing* for *each step* w/o final performance guarantee	No guarantee
ABM	π_i	$\propto \beta \exp(\hat{A}^{\pi_i})$	Multi-step	No guarantee	No guarantee
STR (ours)	π_i	$\text{Proj}_{\text{supp}(\beta)}(\pi_i)$	Multi-step	*Strictly* increasing for *each step* to support-constrained optimal	Safe improvement for *each step*

the theories of EAWBC methods including our proposed STR to present a clear comparison. For a more detailed discussion of related works, please see Appendix A.

4. Supported Trust Region Optimization for Offline RL with Safe Policy Improvement

In this paper, we will relax the implicit density constraint of EAWBC methods to a support constraint, and propose a stable Supported Trust Region algorithm - STR. Benefit from the less restrictive support constraint, STR enjoys much stronger guarantees than prior EAWBC methods. In addition, STR imitates the samples totally from the dataset, thus avoiding the extrapolation error issue of ABM.

Definition 4.1 (Support-constrained policy)**.** The support-constrained policy class Π is defined as

$$\Pi = \{\pi \mid \pi(a|s) = 0 \text{ whenever } \beta(a|s) = 0\} \tag{9}$$

The support constraint set Π actually includes the KL density constraint set $\Pi_d = \{\pi \mid \mathrm{D}_{\mathrm{KL}}(\pi\|\beta)[s] \le \epsilon\}$ of prior EAWBC works, which means the support constraint is less restrictive. It allows to seek the best behavior in the dataset and is usually sufficient to mitigate the extrapolation error. Following prior works (Kumar et al., 2019), we also define the optimal support-constrained policy π^*_Π.

Definition 4.2 (Optimal support-constrained policy)**.** The optimal support-constrained policy π^*_Π is defined as:

$$\pi^*_\Pi(\cdot|s) := \operatorname*{argmax}_{\pi\in\Pi} Q^*_\Pi(s, \pi(s)) \tag{10}$$

where Q^*_Π satisfies the support-constrained Bellman optimality equation:

$$Q^*_\Pi(s,a) = R(s,a) + \gamma \mathop{\mathbb{E}}_{s'\sim P(\cdot|s,a)}\left[\max_{\pi\in\Pi} Q^*_\Pi(s', \pi(s'))\right]$$

Our key observation is that the closed-form policy update of EAWBC (Eq.(4)) is an equal-support update, which means $\text{supp}(\pi_{i+1}) = \text{supp}(\pi_{\text{base}})$ [1]. Considering this, if we initialize π_1 as β and choose the current policy π_i to be π_{base}, by a recursive argument, π_i is still within the support of β, but the density value of π_i can deviate much from β, even with a small constraint constant ϵ. It allows π_i to seek the best behavior in the dataset with small update steps. Therefore, we consider to optimize the following EAWBC objective at iteration i, with $\pi_{\text{base}} = \pi_i, \pi_{\text{pe}} = \pi_i$.

$$J(\phi) = \mathop{\mathbb{E}}_{s\sim\mathcal{D}, a\sim\pi_i}\left[f(s,a;\pi_i)\log(\pi_\phi(a|s))\right] \tag{11}$$

However, in practice, various errors may make π_i deviate from β's support. Even worse, this deviation will accumulate with iterations, eventually leading to large extrapolation error. To address this issue, rather than $\pi_{\text{base}} = \pi_i$, we expect the baseline policy π_{base} to be the projection of π_i on β. We find that Importance Sampling (IS) can satisfy this requirement exactly. Instead of sampling from π_i to optimize $J(\phi)$, STR adopts IS to sample from β and weight the objective by an IS ratio $\pi_i(a|s)/\beta(a|s)$.

$$J_{IS}(\phi) = \mathop{\mathbb{E}}_{s,a\sim\mathcal{D}}\left[\frac{\pi_i(a|s)}{\beta(a|s)} f(s,a;\pi_i)\log(\pi_\phi(a|s))\right] \tag{12}$$

This IS operation is crucial to reduce extrapolation error, because it implicitly only weighted-imitates the in-β-support actions of π_i. We give a more detailed explanation here. IS computes $\mathbb{E}_q[p(x)f(x)/q(x)]$ to estimate $\mathbb{E}_p f(x)$. When $\text{supp}(p) \subseteq \text{supp}(q)$ holds, IS is unbiased. However when the support condition does not hold, IS actually computes $\int_{\text{supp}(q)} p(x)f(x)dx$. For the EAWBC objective, π_i is p and β is q. Maximizing the IS objective $J_{IS}(\phi)$ in Eq.(12) is equivalent to maximizing

$$\tilde{J}(\phi) = \mathop{\mathbb{E}}_{s\sim\mathcal{D}, a\sim\tilde{\pi}_i}\left[f(s,a;\pi_i)\log(\pi_\phi(a|s))\right] \tag{13}$$

where $\tilde{\pi}_i := \text{Proj}_{\text{supp}(\beta)}(\pi_i)$ is obtained by projecting π_i onto β's support:

$$\tilde{\pi}_i(a|s) = \frac{\mathbb{I}[\beta(a|s) > 0]\pi_i(a|s)}{\sum_a \mathbb{I}[\beta(a|s) > 0]\pi_i(a|s)}$$

[1] It follows directly as $f(s,a;\pi_{\text{pe}}) > 0$. We use the exact definition of support ($= 0$).

Algorithm 1 STR (Tabular)

```
Input: Offline dataset $\mathcal{D}$, behavior policy $\beta$, constant $\epsilon$.
Initialize policy $\pi_1$ with $\beta$.
for $i = 1, 2, \dots, N$ do
  Policy evaluation:
      compute $\hat{Q}^{\pi_i}$ in empirical MDP $\mathcal{M}_\mathcal{D}$
  Compute $\hat{A}^{\pi_i}$:
      $\hat{A}^{\pi_i}(s,a) = \hat{Q}^{\pi_i}(s,a) - \mathbb{E}_{a\sim\pi_i}[\hat{Q}^{\pi_i}(s,a)]$.
  Policy improvement:
      $\pi_{i+1}(a|s) = \frac{1}{Z(s)}\pi_i(a|s)\exp\left(\frac{\hat{A}^{\pi_i}(s,a)}{\lambda^*(s)}\right)$.
end for
```

Therefore, if the support constraint is violated at some iteration i due to various errors: $\text{supp}(\pi_i) \not\subseteq \text{supp}(\beta)$, $J_{IS}(\phi)$ will only weighted-imitate the in-β-support actions of π_i and automatically pull π_{i+1} back into β's support, thus mitigating the extrapolation error in practice.

4.1. Theory of STR with a tabular Q

Algorithm 1 instantiates a version of STR in the tabular setting, which is very concise and can clearly show the theoretical advantages of STR.

The following proposition characterizes the equal-support property of STR formally and shows that the π-induced state (state-action) distribution also has the same property.

Proposition 4.3. *For π_i in Algorithm 1, $\text{supp}(\pi_i) = \text{supp}(\beta), \forall i$. It further implies $\text{supp}(d^{\pi_i}) = \text{supp}(d^\beta)$ and $\text{supp}(\rho^{\pi_i}) = \text{supp}(\rho^\beta)$.*[2]

All proofs of Section 4 could be found in Appendix B.

Remark 4.4. Algorithm 1 assumes direct access to β. Although we can only obtain an estimated $\hat{\beta} = n(s,a)/n(s)$ in practice where n is the number of data points in $\mathcal{D}$, it satisfies $\text{supp}(\hat{\beta}) \subseteq \text{supp}(\beta)$, and all theoretical results of STR will still hold, except that the optimal β-support-constrained policy becomes $\hat{\beta}$-support-constrained one. Further, when assuming no sampling error $|\mathcal{D}| = \infty$, $\hat{\beta}$ and β are the same.

Based on Proposition 4.3, we show that without approximation and sampling error (tabular Q and infinite $\mathcal{D}$), policy evaluation of STR under the empirical MDP $\mathcal{M}_\mathcal{D}$ gives the exact Q function under the true MDP.

Proposition 4.5. *In tabular MDP, if the offline dataset $\mathcal{D}$ is generated by a behavior policy β and $|\mathcal{D}| = \infty$, then we can have an exact evaluation of Q^{π_i} for all π_i in Algorithm 1.*

With an exact tabular Q, we show that Algorithm 1 guarantees strict policy improvement for each iteration until it converges to the optimal support-constrained policy π^*_Π. With the same assumption as CRR (Wang et al., 2020), it is much stronger than the prior non-decreasing results that also have no performance guarantee at convergence.

Theorem 4.6 (Strict policy improvement for each step)**.** *If we have the exact tabular estimation of Q, then π_i in Algorithm 1 guarantees monotonic improvement:*

$$Q^{\pi_{i+1}}(s,a) \geq Q^{\pi_i}(s,a) \quad \forall s, a. \tag{14}$$

and the improvement is strict in at least one (s,a) pair until the optimal support-constrained policy π^*_Π is found.

4.2. Theory of STR with an approximate Q

In this section, we relax the assumption in Section 4.1 that prior EAWBC works also make, by incorporating Q-function approximation and sampling error. Specifically, we model the Q function by a value function class $\mathcal{F} \subseteq (\mathcal{S}\times\mathcal{A} \to [0, V_{\max}])$ and remove the assumption on $|\mathcal{D}|$. With function approximation, we optimize one single objective by weighting each state with $d^{\pi_i}(s)$:

$$\max_\pi \alpha \mathop{\mathbb{E}}_{\substack{s\sim d^{\pi_i}\\ a\sim\pi}}[\hat{A}^{\pi_i}(s,a)] - V_{\max} \mathop{\mathbb{E}}_{s\sim d^{\pi_i}}[\mathrm{D}_{\mathrm{KL}}(\pi\|\pi_i)] \tag{15}$$

For ease of presentation, we use a penalty form rather than a constraint form here. Note that all prior EAWBC works use the penalty form in practice.

The optimization problem above also has a closed form solution, which is irrelevant to $d^{\pi_i}(s)$ and is the same as Eq.(4) except for a fixed Lagrange multiplier:

$$\begin{aligned} \pi_{i+1}(a|s) &= \tfrac{1}{Z(s)}\pi_i(a|s)\exp\left(\tfrac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}\right) \\ \text{where } Z(s) &= \textstyle\sum_a \pi_i(a|s)\exp\left(\tfrac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}\right) \end{aligned} \tag{16}$$

Eq.(16) is still an equal-support update and the property in Proposition 4.3 still holds[3]. Now we make two standard assumptions in the offline setting (Chen & Jiang, 2019).

Assumption 4.7 (Approximate Completeness)**.** For any π_i in STR, the following bound holds:

$$\max_{f\in\mathcal{F}}\min_{g\in\mathcal{F}} \|g - \mathcal{T}^{\pi_i}f\|^2_{2,\rho^\beta} \leq \epsilon_{\text{complete}} \tag{17}$$

Here $\|\cdot\|_{2,\rho^\beta} := \sqrt{\mathbb{E}_{\rho^\beta}[(\cdot)^2]}$ is the ρ^β-weighted 2-norm [4].

Assumption 4.8 (Concentrability)**.** For the policy π_i in STR, there exists a constant C such that,

$$\forall t, s, a : \frac{\rho_t^{\pi_i}(s,a)}{\rho^\beta(s,a)} \leq C \tag{18}$$

[2]Here $\text{supp}(\rho^{\pi_i}) = \text{supp}(\rho^\beta)$ is similar to the definition of batch-constrained policies in Fujimoto et al. (2019).

[3]The proof follows directly as that of Proposition 4.3.

[4]We will use the notation $\|f\|_{2,\mathcal{D}}$ for an empirical distribution of the dataset $\mathcal{D}$,where $\|f\|_{2,\mathcal{D}} = \sqrt{\frac{1}{|\mathcal{D}|}\sum_{(s,a,s')\in\mathcal{D}} f(s,a,s')^2}$.

By the equal-support property of STR: $\mathrm{supp}(\rho^{\pi_i}) = \mathrm{supp}(\rho^{\beta})$, this concentrability assumption is very likely to hold under any dataset distribution ρ^{β}.

Theorem 4.9 (FQE error bound)**.** *Under Assumption 4.7 and Assumption 4.8, with probability at least $1-\delta$, after K iterations of Fitted Q Evaluation* (FQE)*, which initializes $f_0 \in \mathcal{F}$ arbitrarily, and iterates K times:*

$$f_k \leftarrow \mathrm{argmin}_{f\in\mathcal{F}} \|f(s,a) - r - \gamma f_{k-1}(s', \pi(s'))\|_{2,\mathcal{D}}$$

the following bound holds:

$$\|Q^{\pi} - f_K\|_{1,\rho^{\pi}} \leq \frac{1-\gamma^K}{1-\gamma}\sqrt{C\epsilon_{gb}} + \gamma^K V_{\max} \tag{19}$$

$$\textit{where } \epsilon_{gb} := \frac{44{V_{\max}}^2 \log(|\mathcal{F}|K/\delta)}{|\mathcal{D}|} + 20\epsilon_{\textit{complete}}$$

The first term in Eq.(19) is the sampling and approximation error term, which goes to 0 with more data and a smaller inherent Bellman error $\epsilon_{\text{complete}}$. The second term is the optimization error term that goes to 0 with more iterations.

Then we formally present the "trust region" property of STR. Although we do not constrain $\mathrm{D_{KL}}(\pi\|\pi_i)$ explicitly, π_{i+1} and π_i in STR are close to each other.

Proposition 4.10 (Trust Region)**.** *For any π_{i+1}, π_i satisfying Eq.*(16)*, the following policy difference bound holds:*

$$\mathrm{D_{TV}}(\pi_{i+1}\|\pi_i)[s] \leq \alpha, \forall s$$
$$\mathrm{D_{KL}}(\pi_i\|\pi_{i+1})[s] \leq \alpha, \forall s$$
$$\mathrm{D_{KL}}(\pi_{i+1}\|\pi_i)[s] \leq \ \alpha(e^{\alpha} - e^{-\alpha})/2, \forall s$$

Finally, we show that with a moderate α, STR is guaranteed to be safe and avoid performance collapse.

Theorem 4.11 (Safe policy improvement for each step)**.** *Under Assumption 4.7 and Assumption 4.8, for π_{i+1}, π_i satisfying Eq.*(16)*, with $\epsilon^{\pi_{i+1}} := \max_s |\mathbb{E}_{a\sim\pi_{i+1}}[A^{\pi_i}(s,a)]|$, the following performance difference bound holds:*

$$\eta(\pi_{i+1}) - \eta(\pi_i) \geq \tfrac{V_{\max}}{(1-\gamma)\alpha}\mathbb{E}_{s\sim d^{\pi_i}}[\mathrm{D_{KL}}(\pi_{i+1}\|\pi_i)]$$
$$- \tfrac{2\alpha}{1-\gamma}\left(\tfrac{\gamma\epsilon^{\pi_{i+1}}}{1-\gamma} + \tfrac{1-\gamma^K}{1-\gamma}\sqrt{C\epsilon_{gb}} + \gamma^K V_{\max}\right)$$

4.3. The Practical Implementation of STR

We design the practical algorithm to be as simple as possible to avoid some complex modules confusing our algorithm's impact on the final performance.

Policy Improvement. Instead of training each iteration to convergence in both the evaluation and improvement stages, practical offline RL algorithms usually take one gradient step. Therefore, we maximize the following objective for policy improvement:

$$J_{\pi}(\phi) = \mathop{\mathbb{E}}_{(s,a)\sim\mathcal{D}}\left[\tfrac{\bar{\pi}_{\phi}(a|s)}{\beta(a|s)}\exp(\tfrac{A_{\theta}(s,a)}{\lambda})\log(\pi_{\phi}(a|s))\right] \tag{20}$$

Algorithm 2 STR (Practical)

```
Input: Offline dataset $\mathcal{D}$, constant $\lambda > 0$.
Initialize behavior policy $\beta_\omega$, policy network $\pi_\phi$, $Q$-network $Q_\theta$, and target $Q$-network $Q_{\theta'}$
// Behavior Policy Pre-training
for each gradient step do
    Update $\omega$ by maximizing $J_\beta(\omega)$ in Eq.(22)
end for
// Policy Training
Initialize policy $\pi_\phi$ with $\beta_\omega$
for each gradient step do
    Update $\theta$ by minimizing $L_Q(\theta)$ in Eq.(1)
    Update $\phi$ by maximizing $J_\pi(\phi)$ in Eq.(20)
    Update target network: $\theta' \leftarrow (1-\tau)\theta' + \tau\theta$
end for
```

where $\bar{\pi}_{\phi}$ means the detach of gradient, and

$$A_{\theta}(s,a) := Q_{\theta}(s,a) - \mathbb{E}_{\hat{a}\sim\pi_{\phi}}[Q_{\theta}(s,\hat{a})] \tag{21}$$

We represent the policy with a Gaussian distribution. In Eq.(21), we find that replacing the expectation with the policy's mean already obtains good performance. Also, it simplifies the training process without learning a V-function.

Besides, as all prior EAWBC works, we omit the partition function $Z(s)$ in Eq.(20), because it only affects the relative weight of different states in the training objective, not different actions. It is theoretically unimportant and empirically hard to estimate. We give a detailed and specific explanation for STR in Appendix D.1.

Density Estimator. We learn a Gaussian density estimator β_{ω} for the behavior policy by maximizing

$$J_{\beta}(\omega) = \mathbb{E}_{s,a\sim\mathcal{D}} \log \beta_{\omega}(a|s), \tag{22}$$

where ω is the parameter of the estimated behavior policy.

Policy Initialization. Theoretically, STR needs to initialize the actor with the behavior policy. In our implementation, we set the network structure of the actor and β_{ω} to be the same, and initialize the actor with β_{ω} directly.

Importance Sampling. To reduce the high variance of importance sampling (Precup et al., 2001), STR adopts Self-Normalized Importance Sampling (SNIS) in practice, which normalizes the IS ratio across the batch.

Overall Algorithm. Putting everything together, we present a practical version of STR in Algorithm 2.

5. Experiments

We test the effectiveness of STR (Algorithm 2) in terms of performance, safe policy improvement, and hyperparameter robustness using the D4RL benchmark (Fu et al., 2020). More experimental details are provided in Appendix D.

Table 2. Averaged normalized scores on MuJoCo locomotion and AntMaze tasks over five seeds.

Dataset (v2)	BC	OneStep	TD3+BC	CQL	IQL	AWAC	CRR	ABM	MPO	STR
halfcheetah-med	42.0	50.4	48.3	47.0	47.4	47.9	47.1	**50.9**	39.7	**51.8±0.3**
hopper-med	56.2	87.5	59.3	53.0	66.2	59.8	38.1	39.4	0.7	**101.3±0.4**
walker2d-med	71.0	**84.8**	83.7	73.3	78.3	83.1	59.7	17.2	-0.2	**85.9±1.1**
halfcheetah-med-replay	36.4	42.7	44.6	45.5	44.2	44.8	44.4	43.4	**51.6**	**47.5±0.2**
hopper-med-replay	21.8	98.5	60.9	88.7	94.7	69.8	25.5	74.7	48.2	**100.0±1.2**
walker2d-med-replay	24.9	61.7	81.8	81.8	73.8	78.1	27.0	**86.9**	3.8	**85.7±2.2**
halfcheetah-med-exp	59.6	75.1	90.7	75.6	86.7	64.9	85.2	70.2	13.2	**94.9±1.6**
hopper-med-exp	51.7	**108.6**	98.0	105.6	91.5	100.1	53.0	1.4	0.7	**111.9±0.6**
walker2d-med-exp	101.2	**111.3**	**110.1**	107.9	**109.6**	**110.0**	91.3	0.1	-0.3	**110.2±0.1**
halfcheetah-exp	92.9	88.2	**96.7**	**96.3**	**95.0**	81.7	93.5	17.6	-3.8	**95.2±0.3**
hopper-exp	110.9	106.9	107.8	96.5	109.4	109.5	108.7	2.4	0.7	**111.2±0.3**
walker2d-exp	107.7	**110.7**	**110.2**	108.5	**109.9**	**110.1**	108.9	64.9	-0.3	**110.1±0.1**
halfcheetah-rand	2.6	2.3	11.0	17.5	13.1	6.1	13.6	2.3	**27.4**	20.6±1.1
hopper-rand	4.1	5.6	8.5	7.9	7.9	9.2	16.1	15.2	**31.7**	**31.3±0.3**
walker2d-rand	1.2	6.9	1.6	5.1	5.4	0.2	4.9	2.6	1.6	4.7±3.8
locomotion total	784.2	1041.2	1013.2	1010.2	1033.1	975.6	817	489.1	214.6	**1162.2**
antmaze-umaze	66.8	54.0	73.0	82.6	89.6	80.0	43.8	87.0	0.0	**93.6±4.0**
antmaze-umaze-diverse	56.8	57.8	47.0	10.2	65.6	52.0	42.8	25.4	0.0	**77.4±7.2**
antmaze-med-play	0.0	0.0	0.0	59.0	76.4	0.0	0.4	0.0	0.0	**82.6±5.4**
antmaze-med-diverse	0.0	0.6	0.2	46.6	72.8	0.2	0.5	0.2	0.0	**87.0±4.2**
antmaze-large-play	0.0	0.0	0.0	16.4	**42.0**	0.0	0.0	0.0	0.0	**42.8±8.7**
antmaze-large-diverse	0.0	0.2	0.0	3.2	**46.0**	0.0	0.0	0.0	0.0	**46.8±7.6**
antmaze total	123.6	112.6	120.2	218	392.4	132.2	87.6	112.6	0.0	**430.2**

5.1. Comparisons on D4RL Benchmarks

We evaluate STR on D4RL in comparison to prior methods.

Tasks. We conduct experiments in Gym-MuJoCo locomotion domains and more challenging AntMaze domains. The latter consist of sparse-reward tasks and require "stitching" fragments of suboptimal trajectories traveling undirectedly to find a path from the start to the goal of the maze.

Baselines. Our offline RL baselines include Behavior Cloning (BC), OneStep RL (Brandfonbrener et al., 2021), TD3+BC (Fujimoto & Gu, 2021), CQL (Kumar et al., 2020), IQL (Kostrikov et al., 2022), AWAC (Nair et al., 2020), CRR (Wang et al., 2020), ABM (Siegel et al., 2020), and MPO (Abdolmaleki et al., 2018).

Comparison with Baselines. Results are shown in Table 2. For learning curves, please refer to Appendix D.6. We find that STR substantially outperforms state-of-the-art methods. It is worth noting that STR outperforms prior EAWBC methods by a large margin, which further demonstrates the advantages of the supported trust region update proposed by STR. Moreover, since ABM and MPO imitate the actions sampled from a learned policy rather than the dataset, they suffer from extrapolation error and fail in most tasks that have a narrow data coverage.

Runtime. We test the runtime of STR on halfcheetah-medium-replay on a GeForce RTX 3090. The results of STR and other baselines are shown in Figure 3 (Right). The runtime of STR is comparable to other baselines. Note that it only takes two minutes for the pre-training part.

5.2. Experimental Verification of the Theories

In this section, we demonstrate the safe policy improvement property of STR from experiments. For this, each policy iteration is trained to convergence in both evaluation and improvement parts. Also, we adopt a relatively large temperature λ in Eq.(20) that corresponds to a strong constraint.

The results are shown in Figure 1, which well support the theory. The performance of AWR and AWAC is limited by the implicit density constraint toward the sub-optimal behavior policy (Lemma 3.1). By contrast, with a less restrictive support constraint, STR is able to deviate more from the behavior policy to achieve better performance. In halfcheetah-random, due to the highly sub-optimal behavior policy, AWAC and AWR can hardly make a visible improvement, while STR obtains better performance. Furthermore, STR has an approximately monotonic performance improvement process in all domains, which demonstrates the safe policy improvement property of STR (Theorem 4.11). In contrast, AWAC has severe performance drops at some iterations (most stark in walker2d-medium-replay and hopper-medium-expert).

5.3. Empirical Study on the Constraint Strength

We investigate the effect of constraint strength on STR and AWAC (best performance among prior EAWBC methods in Table 2). For this, we vary the temperature λ in Eq.(20) which is positively correlated with the constraint strength. The results are presented in Figure 2. Note as the abscissa $1/\lambda$ increases, the constraint becomes looser.

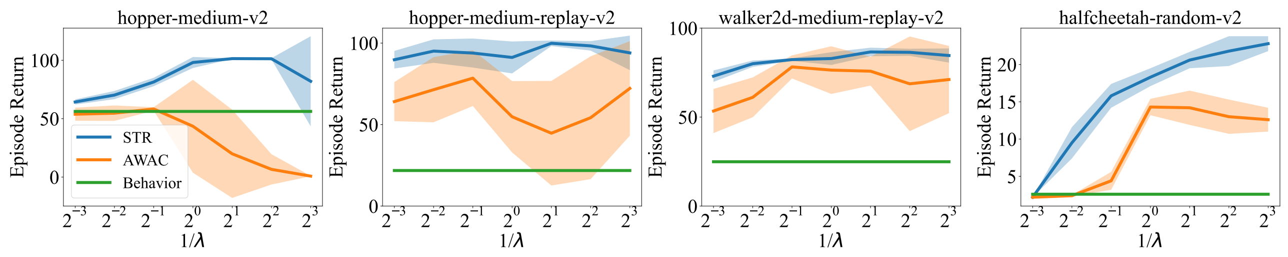


Figure 2. Performance of STR (support constraint) and AWAC (density constraint) with different constraint strength. As the abscissa $1/\lambda$ increases, the constraint becomes looser. STR is more robust to λ. The plots show the average and standard deviation over 5 seeds.

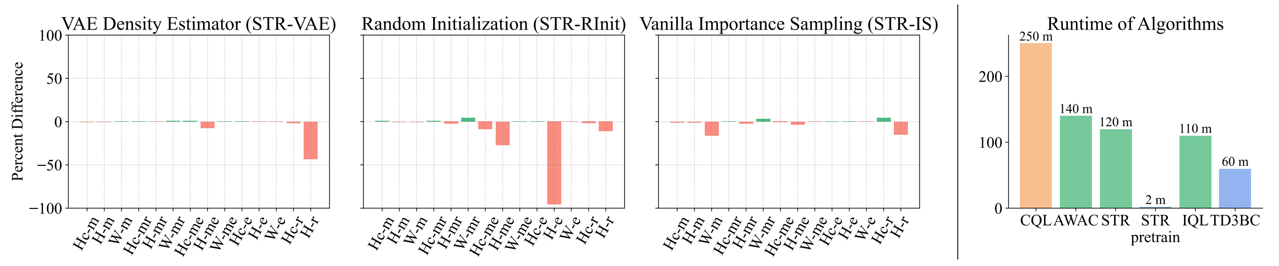


Figure 3. (Left) Percent difference of the performance of an ablation of STR, compared with the original algorithm. Hc = HalfCheetah, H = Hopper, W = Walker2d, m = medium, mr = medium-replay, me = medium-expert, e=expert, r=random. (Right) Runtime of various offline RL algorithms for halfcheetah-medium-replay on a GeForce RTX 3090.

As expected, when the constraint is relatively strong, due to the superiority of the support constraint, the performance of STR is better than that of AWAC, and both algorithms are relatively stable (with small standard deviations). As the constraint gets slightly weaker, the performance of both algorithms improves. However, when the constraint becomes too loose, in hopper-medium both STR and AWAC suffer from overestimation and are unstable (with large standard deviations), while in hopper-medium-replay and walker2d-medium-replay, STR is more stable than AWAC. In general, STR not only maintains good performance over a wide range of λ, but is also better than AWAC for every λ.

5.4. Ablation Study

We perform an ablation study over the components in our method, including the behavior density estimator, policy initialization, and importance sampling techniques. The results are shown in Figure 3.

Behavior Density Estimator. Following previous works (Fujimoto et al., 2019; Wu et al., 2022), we consider to replace the Gaussian density estimator β_ω with conditional variational auto-encoder (Sohn et al., 2015). We refer to this variant as STR-VAE. Overall, the performance of STR-VAE and STR is almost the same, except that STR-VAE obtains worse results on hopper-random.

Policy Initialization. Consistent with the theory of STR, we initialize the policy with the pre-trained β_ω. Here we evaluate an STR variant with random policy initialization, termed STR-RInit. STR-RInit achieves similar performance in most tasks except hopper-expert and hopper-medium-expert. The reason is that, in practice, the IS operation in STR will implicitly pull π_ϕ back into β's support, no matter how π_ϕ is initialized. Therefore, an inaccurate β_ω initialization or even random initialization could also achieve good results in most cases. However in hopper-expert, the data coverage is narrow and overestimation is more likely to happen. We find the random initialization will lead to severe overestimation of Q in hopper-expert.

Importance Sampling. STR adopts SNIS to reduce the variance of IS. Here we test an STR variant with vanilla importance sampling, termed STR-IS. As shown in Figure 3, the performance of STR-IS is slightly worse than STR due to higher variance, but the difference is not large.

6. Conclusion

We propose STR, an offline RL algorithm with supported trust region policy optimization based on EAWBC. STR relaxes the density constraint of prior EAWBC works to a support constraint and enjoys stronger theoretical guarantees, including strict policy improvement until convergence to the optimal support-constrained policy with an exact Q and safe policy improvement with an approximate Q. Empirical evaluations confirm the theoretical results and demonstrate STR's SoTA performance on offline RL benchmarks.

Acknowledgment

This work was supported by the National Key R&D Program of China under Grant 2018AAA0102801, National Natural Science Foundation of China under Grant 61827804.

References

Abdolmaleki, A., Springenberg, J. T., Tassa, Y., Munos, R., Heess, N., and Riedmiller, M. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*, 2018.

Achiam, J., Held, D., Tamar, A., and Abbeel, P. Constrained policy optimization. In *International conference on machine learning*, pp. 22–31. PMLR, 2017.

Agarwal, A., Jiang, N., Kakade, S. M., and Sun, W. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, pp. 10–4, 2019.

An, G., Moon, S., Kim, J.-H., and Song, H. O. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021.

Barth-Maron, G., Hoffman, M. W., Budden, D., Dabney, W., Horgan, D., TB, D., Muldal, A., Heess, N., and Lillicrap, T. Distributional policy gradients. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SyZipzbCb.

Brandfonbrener, D., Whitney, W., Ranganath, R., and Bruna, J. Offline rl without off-policy evaluation. *Advances in Neural Information Processing Systems*, 34:4933–4946, 2021.

Chatterjee, S. and Diaconis, P. The sample size required in importance sampling. *The Annals of Applied Probability*, 28(2):1099–1135, 2018.

Chen, J. and Jiang, N. Information-theoretic considerations in batch reinforcement learning. In *International Conference on Machine Learning*, pp. 1042–1051. PMLR, 2019.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Chen, X., Zhou, Z., Wang, Z., Wang, C., Wu, Y., and Ross, K. Bail: Best-action imitation learning for batch deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33:18353–18363, 2020.

Cheng, C.-A., Xie, T., Jiang, N., and Agarwal, A. Adversarially trained actor critic for offline reinforcement learning. *arXiv preprint arXiv:2202.02446*, 2022.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pp. 1329–1338. PMLR, 2016.

Emmons, S., Eysenbach, B., Kostrikov, I., and Levine, S. Rvs: What is essential for offline rl via supervised learning? *arXiv preprint arXiv:2112.10751*, 2021.

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

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Fujimoto, S., Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pp. 1587–1596. PMLR, 2018.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

Gulcehre, C., Wang, Z., Novikov, A., Paine, T., Gómez, S., Zolna, K., Agarwal, R., Merel, J. S., Mankowitz, D. J., Paduraru, C., et al. Rl unplugged: A suite of benchmarks for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:7248–7259, 2020.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Kakade, S. and Langford, J. Approximately optimal approximate reinforcement learning. In *In Proc. 19th International Conference on Machine Learning*. Citeseer, 2002.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Kumar, A., Fu, J., Soh, M., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32, 2019.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 1179–1191, 2020.

Liu, Y., Swaminathan, A., Agarwal, A., and Brunskill, E. Provably good batch off-policy reinforcement learning without great exploration. *Advances in neural information processing systems*, 33:1264–1274, 2020.

Lyu, J., Ma, X., Li, X., and Lu, Z. Mildly conservative q-learning for offline reinforcement learning. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=VYYf6S67pQc.

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

Peng, X. B., Kumar, A., Zhang, G., and Levine, S. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Precup, D., Sutton, R. S., and Dasgupta, S. Off-policy temporal-difference learning with function approximation. In *ICML*, pp. 417–424, 2001.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Siegel, N. Y., Springenberg, J. T., Berkenkamp, F., Abdolmaleki, A., Neunert, M., Lampe, T., Hafner, R., Heess, N., and Riedmiller, M. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. *arXiv preprint arXiv:2002.08396*, 2020.

Sohn, K., Lee, H., and Yan, X. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Wang, Q., Xiong, J., Han, L., Liu, H., Zhang, T., et al. Exponentially weighted imitation learning for batched historical data. *Advances in Neural Information Processing Systems*, 31, 2018.

Wang, Z., Novikov, A., Zolna, K., Merel, J. S., Springenberg, J. T., Reed, S. E., Shahriari, B., Siegel, N., Gulcehre, C., Heess, N., et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33: 7768–7778, 2020.

Wu, J., Wu, H., Qiu, Z., Wang, J., and Long, M. Supported policy optimization for offline reinforcement learning. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=KCXQ5HoM-fy.

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

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

A. Related Works

Offline RL. In offline RL, a fixed dataset is provided and no further interactions are allowed. As a result, ordinary off-policy RL algorithms suffer from extrapolation error due to OOD actions (Fujimoto et al., 2019) and have poor performance. Among various solutions, value penalty methods attempt to penalize the Q-values of OOD actions (Kumar et al., 2020; An et al., 2021; Lyu et al., 2022), while policy constraint methods force the trained policy to be close to the behavior policy by KL divergence (Wu et al., 2019), or by direct behavior cloning (Fujimoto & Gu, 2021). Recently, instead of density policy constraint which is too restrictive in many cases, some works consider the less restrictive support constraint to keep the learned policy within the support of behavior policy by Maximum Mean Discrepancy (MMD) (Kumar et al., 2019) or explicit density estimation (Wu et al., 2022), but their performance still leaves considerable room for improvement. Another branch of algorithms chooses to perform in-sample learning, by formulating the Bellman target without querying the values of actions not contained in the dataset. Among them, OneStep RL (Brandfonbrener et al., 2021) evaluates the behavior policy by SARSA and only performs one-step of constrained policy improvement without off-policy evaluation. IQL (Kostrikov et al., 2022) modifies the SARSA update by expectile regression to approximate an upper expectile of the value distribution, and enables multi-step dynamic programming.

Weighted Behavior Cloning. Among policy constraint methods, Weighted Behavior Cloning (WBC) reduces the RL problem to a supervised learning problem (Emmons et al., 2021). They modify an imitation learning algorithm either by filtering or weighting the actions in dataset to distill a better policy. For the filtering version, BAIL (Chen et al., 2020) proposes upper-envelope to select good state-action pairs for later imitation learning. 10%BC only uses the top 10% of transitions ordered by episode return to perform behavior cloning (Chen et al., 2021). However, they are intuitive methods and do not have theoretical guarantees of policy improvement. For the weighted version, prior works typically adopt Exponentiated Advantage-Weighted Behavior Cloning (EAWBC), with the weights being determined by the exponentiated advantage estimates. Examples of EAWBC methods include MARWIL (Wang et al., 2018), AWR (Peng et al., 2019), AWAC (Nair et al., 2020), CRR (Wang et al., 2020) and ABM (Siegel et al., 2020). We have summarized their essential similarities and differences in Section 3. At the implementation level, MARWIL and AWR are similar one-step methods and only differ in advantage estimation: with a learned value function V^{β}, MARWIL uses the single-path advantage estimate while AWR uses TD(λ) to approximate the episode return. CRR and AWAC are concurrent multi-step methods and the difference is that CRR uses distributional Q-function (Barth-Maron et al., 2018).

To some extent, our algorithm STR shares some similarities with an online algorithm MPO (Abdolmaleki et al., 2018), which starts from an inference perspective and essentially weighted-clones the actions from the current policy for policy improvement. In the offline setting, MPO does not satisfy the support constraint and suffers from large extrapolation error, as we find empirically in Section 5. Also, MPO has no theoretical analysis for the offline setting and its implementation is very complicated. On the other hand, STR initializes the policy with the behavior policy and utilizes importance sampling on the dataset to mimic actions from the current policy. It enforces the policy of STR within the support of the behavior policy, thus mitigating the extrapolation error and offering excellent empirical performance.

Trust Region and Safe Policy Improvement. In online RL, trust region methods, with two typical embodiments of Trust Region Policy Optimisation (TRPO) (Schulman et al., 2015) and Proximal Policy Optimisation (PPO) (Schulman et al., 2017), have shown impressive performance on both discrete and continuous tasks (Duan et al., 2016). They optimize the policy within a trusted neighborhood of the current policy, which empirically avoids taking aggressive updates towards risky directions, and theoretically guarantees safe policy improvement at each step (Schulman et al., 2015; Achiam et al., 2017). However in offline RL, there are few studies about trust region optimization. Besides, with approximation and sampling error, few offline RL algorithms can ensure safe policy improvement for each step. Recently, there are several offline RL works (Liu et al., 2020; Xie et al., 2021; Cheng et al., 2022) that guarantee safe policy improvement over the behavior policy. In contrast, we consider safe policy improvement with respect to the current policy, at each policy update step.

B. Proofs

We start out with the performance difference lemma (Kakade & Langford, 2002) that shows that the difference in policy performance $\eta(\pi') - \eta(\pi)$ can be decomposed as an expectation of advantages.

Lemma B.1. *Given two policies π', π,*

$$\eta(\pi') - \eta(\pi) = \frac{1}{1-\gamma} \sum_s d_{\pi'}(s) \sum_a \left[\pi'(a|s) A^{\pi}(s,a)\right] \tag{23}$$

Proof. Please see the proof of Lemma 6.1 in Kakade & Langford (2002). □

The complex dependency of $d_{\pi'}(s)$ on π' makes Equation (23) difficult to optimize directly. It is proved in Achiam et al. (2017) that the performance difference satisfies the following inequality.

Lemma B.2. *$\forall \pi', \pi$, with $\epsilon^{\pi'} := \max_s |\mathbb{E}_{a\sim\pi'}[A^\pi(s,a)]|$, the following bound holds:*

$$\eta(\pi') - \eta(\pi) \geq \frac{1}{1-\gamma} \mathop{\mathbb{E}}_{\substack{s\sim d^\pi \\ a\sim\pi'}} \left[A^\pi(s,a) - \frac{2\gamma\epsilon^{\pi'}}{1-\gamma} \mathrm{D_{TV}}(\pi' \| \pi)[s] \right]. \tag{24}$$

Proof. Please see the proof of Corollary 1 in Achiam et al. (2017). □

The bound Eq.(24) should be compared with Eq.(23). The term $\mathbb{E}_{s\sim d^\pi a\sim\pi'}[A^\pi(s,a)$ in Eq.(24) is an approximation to $\eta(\pi') - \eta(\pi)$, using the state distribution d^π instead of $d^{\pi'}$, which is known equal to $\eta(\pi') - \eta(\pi)$ to first order in the parameters of π' on a neighborhood around π (Kakade & Langford, 2002).

Lemma B.3 (Lemma 3.1). *If $\mathrm{D_{KL}}(\pi(\cdot|s)\|\beta(\cdot|s)) \leq \epsilon, \forall s$ is guaranteed, then the performance η has the following bound*

$$\eta(\pi) \leq \eta(\beta) + \frac{V_{\max}}{\sqrt{2}(1-\gamma)}\sqrt{\epsilon} \tag{25}$$

Proof. By Lemma B.1, we have

$$\begin{aligned}
|\eta(\beta) - \eta(\pi)| &= \frac{1}{1-\gamma} \left| \sum_s d_\beta(s) \sum_a [(\beta(a|s) - \pi(a|s)) Q^\pi(s,a)] \right| \\
&\leq \frac{1}{1-\gamma} \sum_s d_\beta(s) \sum_a [|\beta(a|s) - \pi(a|s)|\, |Q^\pi(s,a)]| \\
&\leq \frac{V_{\max}}{1-\gamma} \sum_s d_\beta(s) \sum_a [|\beta(a|s) - \pi(a|s)| \\
&= \frac{V_{\max}}{1-\gamma} \sum_s d_\beta(s) \mathrm{D_{TV}}(\pi\|\beta)[s] \\
&\leq \frac{V_{\max}}{\sqrt{2}(1-\gamma)} \sum_s d_\beta(s) \sqrt{\mathrm{D_{KL}}(\pi\|\beta)[s]} \qquad \text{(Pinsker's inequality)} \\
&\leq \frac{V_{\max}}{\sqrt{2}(1-\gamma)} \sqrt{\epsilon}
\end{aligned} \tag{26}$$

□

Proposition B.4 (Proposition 4.3). *For π_i in Algorithm 1, $\mathrm{supp}(\pi_i(\cdot|s)) = \mathrm{supp}(\beta(\cdot|s)), \forall i$. When assuming the MDP has a fixed initial state distribution d_0, it implies $\mathrm{supp}(d^{\pi_i}(\cdot)) = \mathrm{supp}(d^\beta(\cdot))$ and $\mathrm{supp}(\rho^{\pi_i}(\cdot,\cdot)) = \mathrm{supp}(\rho^\beta(\cdot,\cdot))$*

Proof. As $\frac{1}{Z(s)} \exp\left(\frac{\hat{A}^{\pi_i}(s,a)}{\lambda^*(s)}\right) > 0$, it holds that $\mathrm{supp}(\pi_{i+1}(\cdot|s)) = \mathrm{supp}(\pi_i(\cdot|s))$, $\forall s$. By a recursive argument, $\mathrm{supp}(\pi_i(\cdot|s)) = \mathrm{supp}(\beta(\cdot|s)), \forall i$. Here we use the exact definition of support ($= 0$), rather than $>$ some threshold. As the distribution of $d^\pi(s)$ is induced by the transition dynamics and the policy. With $\mathrm{supp}(\pi_i(\cdot|s)) = \mathrm{supp}(\beta(\cdot|s))$ and the same $\mathcal{P}$, it follows directly $\mathrm{supp}(d^{\pi_i}(\cdot)) = \mathrm{supp}(d^\beta(\cdot))$. As $\rho^\pi(s,a) = d^\pi(s)\pi(a|s)$, it also holds that $\mathrm{supp}(\rho^{\pi_i}(\cdot,\cdot)) = \mathrm{supp}(\rho^\beta(\cdot,\cdot))$. □

Proposition B.5 (Proposition 4.5). *In tabular MDP, if the offline dataset $\mathcal{D}$ is generated by a behavior policy β and $|\mathcal{D}| = \infty$, then we can have an exact evaluation of Q^{π_i} for all π_i in Algorithm 1.*

Proof. We construct the empirical MDP $\mathcal{M}_{\mathcal{D}}$ in tabular setting following Fujimoto et al. (2019). Specifically, $\mathcal{M}_{\mathcal{D}}$ is defined by the same action and state space as $\mathcal{M}$, with an additional terminal state s_{init}. $\mathcal{M}_{\mathcal{D}}$ has transition probabilities $\mathcal{P}_{\mathcal{D}}(s'|s,a) = \frac{N(s,a,s')}{\sum_{\tilde{s}} N(s,a,\tilde{s})}$, where $N(s,a,s')$ is the number of times the tuple (s,a,s') is observed in $\mathcal{D}$. If $\sum_{\tilde{s}} N(s,a,\tilde{s}) = 0$, then $\mathcal{P}_{\mathcal{D}}(s_{\text{init}}|s,a) = 1$, where $r(s_{\text{init}}, s, a)$ is to the initialized value of $Q(s,a)$.

Following Proposition B.4, any (s,a) such that $\rho^{\pi_i}(s,a) > 0$ must satisfy $\rho^{\beta}(s,a) > 0$. With the no sampling error assumption: $|\mathcal{D}| = \infty$, those (s,a) also satisfy $\mathcal{P}_{\mathcal{D}}(s'|s,a) = \mathcal{P}(s'|s,a)$ for all $s' \in \mathcal{S}$. Then following Lemma 1 in Fujimoto et al. (2019), we can conclude the proof. □

Theorem B.6 (Strict policy improvement for each step, Theorem 4.6)**.** *If we have the exact tabular estimation of Q, then π_i in Algorithm 1 guarantees monotonic improvement:*

$$Q^{\pi_{i+1}}(s,a) \geq Q^{\pi_i}(s,a) \quad \forall s, a. \tag{27}$$

and the improvement is strict in at least one (s,a) pair until the optimal support-constrained policy π^*_{Π} is found:

$$\pi_i = \pi^*_{\Pi} \tag{28}$$

Proof. When assuming exact evaluation of Q^{π_i} which holds if $|\mathcal{D}| = \infty$ by Proposition B.5, π_{i+1} in Algorithm 1 is the optimal solution of the following constrained optimization problem:

$$\begin{aligned} &\pi_{i+1} = \underset{\pi}{\operatorname{argmax}} \underset{a\sim\pi}{\mathbb{E}}[Q^{\pi_i}(s,a)] \\ &s.t. \mathrm{D}_{\mathrm{KL}}(\pi\|\pi_i) \leq \epsilon \\ &\sum_a \pi(a|s) = 1, \forall s \end{aligned} \tag{29}$$

we know that $\mathrm{D}_{\mathrm{KL}}(\pi_i\|\pi_i) = 0 < \epsilon \ \forall s$ is an strictly feasible solution to the optimization problem above. Therefore, $\mathbb{E}_{a\sim\pi_{i+1}}[Q^{\pi_i}(s,a)] \geq \mathbb{E}_{a\sim\pi_i}[Q^{\pi_i}(s,a)] \ \forall s$. It implies

$$\begin{aligned} &Q^{\pi_i}(s,a) \\ =&\mathbb{E}\left[r(s_t,a_t) + \gamma \sum_{s_{t+1},a_{t+1}} P(s_{t+1}|s_t,a_t)\pi_i(a_{t+1}|s_{t+1})Q^{\pi_i}(s_{t+1},a_{t+1})|s_t = s, a_t = a\right] \\ \leq&\mathbb{E}\left[r(s_t,a_t) + \gamma \sum_{s_{t+1},a_{t+1}} P(s_{t+1}|s_t,a_t)\pi_{i+1}(a_{t+1}|s_{t+1})Q^{\pi_i}(s_{t+1},a_{t+1})|s_t = s, a_t = a\right] \\ &\quad\quad \cdots \\ \leq&\mathbb{E}_{\pi_{i+1}}\left[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k},a_{t+k})|s_t = s, a_t = a\right] \\ =&Q^{\pi_{i+1}}(s,a) \end{aligned}$$

Therefore, $Q^{\pi_{i+1}}(s,a) \geq Q^{\pi_i}(s,a) \ \forall s, a$.

Then if in some iteration i, the optimization problem in Eq.(29) do not improve the objective function at any s, $i.e.$ $\mathbb{E}_{a\sim\pi_{i+1}}[Q^{\pi_i}(s,a)] = \mathbb{E}_{a\sim\pi_i}[Q^{\pi_i}(s,a)] \ \forall s$, it means that π_i is a minimizer of of the convex optimization problem in Eq.(29). Note that π_i is a interior point of the feasible set. Consider another convex optimization problem whose feasible region contains the original one in Eq.(29):

$$\begin{aligned} &\pi_{i+1} = \underset{\pi}{\operatorname{argmax}} \underset{a\sim\pi}{\mathbb{E}}[Q^{\pi_i}(s,a)] \\ &s.t. \mathrm{D}_{\mathrm{KL}}(\pi\|\pi_i) < \infty \\ &\sum_a \pi(a|s) = 1, \forall s \end{aligned} \tag{30}$$

Here $\mathrm{D}_{\mathrm{KL}}(\pi\|\pi_i) = \sum_a \pi(a|s)\log\frac{\pi(a|s)}{\pi_i(a|s)} < \infty$ is actually equivalent to $\mathrm{supp}(\pi(\cdot|s)) \subseteq \mathrm{supp}(\pi_i(\cdot|s)) = \mathrm{supp}(\beta(\cdot|s))$, and thus the feasible region of Eq.(30) is actually the support constrained policy set Π (Definition 4.1). Note that the problem in Eq.(29) and the problem in Eq.(30) share the same convex objective function and the convex feasible set of the former is contained by the one of the latter. Here we know π_i is a interior minimizer of the former convex optimization problem. It implies that π_i is also a local minimizer of the latter convex optimization problem, which further implies π_i is a global minimizer of the latter convex optimization problem by convexity. Therefore,

$$\pi_i = \operatorname*{argmax}_{\pi\in\Pi} \mathop{\mathbb{E}}_{a\sim\pi}[Q^{\pi_i}(s,a)] \tag{31}$$

By Bellman Equation of π_i, we have

$$\begin{aligned} Q^{\pi_i}(s,a) &= r(s,a) + \gamma \mathop{\mathbb{E}}_{s'\sim P(\cdot|s,a)} \mathop{\mathbb{E}}_{a'\sim\pi_i(\cdot|s')}[Q^{\pi_i}(s',a')] && (32)\\ &= r(s,a) + \gamma \mathop{\mathbb{E}}_{s'\sim P(\cdot|s,a)} \max_{\pi\in\Pi} \mathop{\mathbb{E}}_{a'\sim\pi(\cdot|s')}[Q^{\pi_i}(s',a')] && (33)\end{aligned}$$

It is actually the support-constrained Bellman optimality equation in Definition 4.2. So we have, $Q^{\pi_i}(s,a) = Q^*_\Pi(s,a),\ \forall(s,a)$. Further, by the equivalence of Eq.(31) and the definition of the optimal support-constrained policy π^*_Π in Eq.(10), we know $\pi_i = \pi^*_\Pi$. To conclude, if in some iteration i the optimization problem in Eq.(29) do not make improvement at any (s,a), it implies $Q^{\pi_{i+1}}(s,a) = Q^{\pi_i}(s,a)\ \forall s,a$. Then our analysis shows π_i equals the optimal support-constrained policy π^*_Π.

□

From now on, we relax the assumption and incorporate approximation error and sampling error. Specifically, instead of tabular Q, we approximate the Q functions by a value function class $\mathcal{F} \subseteq (\mathcal{S}\times\mathcal{A} \to [0, V_{\max}])$. At the same time, we do not make any assumption on $|\mathcal{D}|$.

To prove the FQE error bound of policy evaluation (Theorem 4.9), we first prove a lemma of generalization bound.

Lemma B.7 (generalization bound)**.** *Under the approximate completeness assumption (Assumption 4.7), with probability at least* $1-\delta$*, for all* $k = 1,\dots,K$ *and* $\forall\pi$*, we have:*

$$\|f_{k+1} - \mathcal{T}^\pi f_k\|^2_{2,\rho^\beta} \le \frac{22V^2_{\max}\log(|\mathcal{F}|K/\delta)}{|\mathcal{D}|} + 20\epsilon_{complete} \tag{34}$$

Proof. For any fixed f_{k-1}, FQE deals with the following regression problem on dataset

$$f_k \leftarrow \operatorname*{argmin}_{f\in\mathcal{F}} \sum_{i=1}^{|\mathcal{D}|} \left(f(s_i,a_i) - r_i - \gamma f_{k-1}(s'_i, \pi(s'_i))\right)^2 \tag{35}$$

In this regression problem, we have $|r_i + \gamma f_{k-1}(s'_i,\pi(s'_i))| \le 1 + \gamma V_{\max} \le 2V_{\max}$. And for our Bayes optimal solution, we have $|\mathcal{T}^\pi f_{k-1}(s,a)| = |r(s,a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a)} f_{k-1}(s',\pi(s'_i))| \le 1+\gamma V_{\max} \le 2V_{\max}$. Also note that Assumption 4.7 implies that $\min_{f\in\mathcal{F}}\|f - \mathcal{T}^\pi f_{k-1}\|^2_{2,\rho^\beta} \le \epsilon_{\text{complete}}$. Thus, we can apply least squares generalization bound here (Lemma A.11 in Agarwal et al. (2019)). With probability at least $1-\delta$, we have

$$\|f_k - \mathcal{T}^\pi f_{k-1}\|^2_{2,\rho^\beta} \le \frac{22{V_{\max}}^2\log(|\mathcal{F}|/\delta)}{|\mathcal{D}|} + 20\epsilon_{\text{complete}}$$

The above inequality holds for the fixed f_{k-1}. Since different π can induce different f_{k-1}, we apply a union bound over all possible $f_{k-1}\in\mathcal{F}$. Also, we apply a union bound over all $k = 1,\dots,K$. Therefore, with probability at least $1-\delta$, we have

$$\|f_k - \mathcal{T}^\pi f_{k-1}\|^2_{2,\rho^\beta} \le \frac{44{V_{\max}}^2\log(|\mathcal{F}|K/\delta)}{|\mathcal{D}|} + 20\epsilon_{\text{complete}}$$

□

Theorem B.8 (FQE error bound, Theorem 4.9)**.** *Under Assumption 4.7 and Assumption 4.8, with probability at least* $1-\delta$*, after* K *iterations of* FQE*, which initializes* $f_0 \in \mathcal{F}$ *arbitrarily, and iterates* K *times:*

$$f_k \leftarrow \operatorname*{argmin}_{f \in \mathcal{F}} \|f(s,a) - r - \gamma f_{k-1}(s', \pi(s'))\|_{2,\mathcal{D}}$$

the following bound holds:

$$\|Q^\pi - f_K\|_{1,\rho^\pi} \le \frac{1-\gamma^K}{1-\gamma}\sqrt{C\epsilon_{gb}} + \gamma^K V_{\max} \tag{36}$$

$$\textit{where } \epsilon_{gb} := \frac{44{V_{\max}}^2 \log(|\mathcal{F}|K/\delta)}{|\mathcal{D}|} + 20\epsilon_{\textit{complete}} \tag{37}$$

Proof. we first bound $\|f_K - Q^\pi\|_{2,d_t^\pi \times \pi}$ for all time step t.

$$\begin{aligned}\|f_K - Q^\pi\|_{2,d_t^\pi \times \pi} &= \|f_K - \mathcal{T}^\pi f_{K-1} + \mathcal{T}^\pi f_{K-1} - Q^\pi\|_{2,d_t^\pi \times \pi} \\ &\le \underbrace{\|f_K - \mathcal{T}^\pi f_{K-1}\|_{2,d_t^\pi \times \pi}}_{(1)} + \underbrace{\|\mathcal{T}^\pi f_{K-1} - \mathcal{T}^\pi Q^\pi\|_{2,d_t^\pi \times \pi}}_{(2)} \end{aligned} \tag{38}$$

For term (1): following Assumption 4.8 and Lemma B.7, with probability at least $1-\delta$, for all $k = 1, \dots, K$ and $\forall \pi$ that satisfies Assumption 4.8, we have

$$\begin{aligned}\|f_k - \mathcal{T}^\pi f_{k-1}\|_{2,d_t^\pi \times \pi} &\le \sqrt{C}\,\|f_k - \mathcal{T}^\pi f_{k-1}\|_{2,\rho^\beta} && \text{(Assumption 4.8)} \\ &\le \sqrt{C}\sqrt{\frac{44{V_{\max}}^2 \log(|\mathcal{F}|K/\delta)}{|\mathcal{D}|} + 20\epsilon_{\text{complete}}} && \text{(Lemma B.7)}\end{aligned} \tag{39}$$

For term (2):

$$\begin{aligned}\|\mathcal{T}^\pi f_{K-1} - \mathcal{T}^\pi Q^\pi\|_{2,d_t^\pi \times \pi} &= \sqrt{\mathbb{E}_{(s,a)\sim d_t^\pi \times \pi}\left[\left((\mathcal{T}^\pi f_{K-1})(s,a) - (\mathcal{T}^\pi Q^\pi)(s,a)\right)^2\right]} \\ &= \sqrt{\mathbb{E}_{(s,a)\sim d_t^\pi \times \pi}\left[\left(\gamma \mathbb{E}_{s'\sim P(\cdot|s,a)}\mathbb{E}_{a'\sim\pi(\cdot|s')}\left[f_{K-1}(s',a') - Q^\pi(s',a')\right]\right)^2\right]} \\ &\le \gamma\sqrt{\mathbb{E}_{(s,a)\sim d_t^\pi \times \pi, s'\sim P(\cdot|s,a), a'\sim\pi(\cdot|s')}\left[\left(f_{K-1}(s',a') - Q^\pi(s',a')\right)^2\right]} \quad \text{(Jensen's inequality)} \\ &= \gamma\sqrt{\mathbb{E}_{(s',a')\sim d_{t+1}^\pi \times \pi}\left[\left(f_{K-1}(s',a') - Q^\pi(s',a')\right)^2\right]} \\ &= \gamma\,\|f_{K-1} - Q^\pi\|_{2,d_{t+1}^\pi \times \pi}\end{aligned} \tag{40}$$

Combine term (1) and term (2):

$$\|f_K - Q^\pi\|_{2,d_t^\pi \times \pi} \le \sqrt{C\epsilon_{gb}} + \gamma\,\|f_{K-1} - Q^\pi\|_{2,d_{t+1}^\pi \times \pi} \tag{41}$$

where

$$\epsilon_{gb} := \frac{44{V_{\max}}^2 \log(|\mathcal{F}|K/\delta)}{|\mathcal{D}|} + 20\epsilon_{\text{complete}} \tag{42}$$

Note that we can apply the same analysis on $\|f_{K-1} - Q^\pi\|_{2,d_{t+1}^\pi \times \pi}$. We recursively repeat the same process K times:

$$\begin{aligned}\|f_K - Q^\pi\|_{2,d_t^\pi \times \pi} &\le \sum_{k=0}^{K-1} \gamma^t \sqrt{C\epsilon_{gb}} + \gamma^K\,\|f_0 - Q^\pi\|_{2,d_{t+K}^\pi \times \pi} \\ &\le \frac{1-\gamma^K}{1-\gamma}\sqrt{C\epsilon_{gb}} + \gamma^K V_{\max}\end{aligned} \tag{43}$$

Then we derive the bound $\|f_K - Q^\pi\|_{2,\rho^\pi}$ with distribution $\rho^\pi = d^\pi \times \pi$:

$$\begin{aligned}
\|f_K - Q^\pi\|_{2,\rho^\pi} &= \|f_K - Q^\pi\|_{2,d^\pi\times\pi} \\
&= \sqrt{\sum_s d^\pi(s)\sum_a \pi(a|s)(f_K(s,a) - Q^\pi(s,a))^2} \\
&= \sqrt{\sum_s (1-\gamma)\sum_{t=0}^{\infty}\gamma^t d_t^\pi(s)\sum_a \pi(a|s)(f_K(s,a) - Q^\pi(s,a))^2} \\
&= \sqrt{(1-\gamma)\sum_{t=0}^{\infty}\gamma^t \sum_s d_t^\pi(s)\sum_a \pi(a|s)(f_K(s,a) - Q^\pi(s,a))^2} \\
&= \sqrt{(1-\gamma)\sum_{t=0}^{\infty}\gamma^t \|f_K - Q^\pi\|^2_{2,d_t^\pi\times\pi}} \\
&\le \sqrt{(1-\gamma)\sum_{t=0}^{\infty}\gamma^t \left(\frac{1-\gamma^K}{1-\gamma}\sqrt{C\epsilon_{gb}} + \gamma^K V_{\max}\right)^2} \\
&= \frac{1-\gamma^K}{1-\gamma}\sqrt{C\epsilon_{gb}} + \gamma^K V_{\max}
\end{aligned}$$

Finally we apply the inequality between weighted $L1$-norm and weighted $L2$-norm, and conclude the proof:

$$\begin{aligned}
\|f_K - Q^\pi\|_{1,\rho^\pi} &\le \|f_K - Q^\pi\|_{2,\rho^\pi} && \text{(Jensen's inequality)} \\
&\le \frac{1-\gamma^K}{1-\gamma}\sqrt{C\epsilon_{gb}} + \gamma^K V_{\max}
\end{aligned}$$

□

With some derivations, we can translate the Q error bound to the A error bound, which is closer to the performance difference (Lemma B.1,Lemma B.2). For simplicity of the final results, we assume $\alpha \le 0.48$. A small α leads to a trust region update. Please note $\alpha \le 0.48$ is not a necessary requirement here, just for simplicity of the results.

Lemma B.9 (Advantage error bound)**.** *Under Assumption 4.7, Assumption 4.8 and $\alpha \le 0.48$, with probability at least $1-\delta$, for any π_{i+1}, π_i satisfying Eq.(16), the following bound holds:*

$$\left|\mathbb{E}_{s\sim d_{\pi_i}, a\sim\pi_{i+1}}\left[A^{\pi_i}(s,a) - \hat{A}^{\pi_i}(s,a)\right]\right| \le 2\alpha\left(\frac{1-\gamma^K}{1-\gamma}\sqrt{C\epsilon_{gb}} + \gamma^K V_{\max}\right) \tag{44}$$

Proof.

$$\begin{aligned}
&\left|\mathbb{E}_{s\sim d_{\pi_i}, a\sim\pi_{i+1}}\left[A^{\pi_i}(s,a) - \hat{A}^{\pi_i}(s,a)\right]\right| \\
=&\left|\mathbb{E}_{s\sim d_{\pi_i}, a\sim\pi_{i+1}}\left[Q^{\pi_i}(s,a) - \hat{Q}^{\pi_i}(s,a)\right] + \mathbb{E}_{s\sim d_{\pi_i}, a\sim\pi_i}\left[\hat{Q}^{\pi_i}(s,a) - Q^{\pi_i}(s,a)\right]\right| \\
=&\left|\mathbb{E}_{s\sim d_{\pi_i}, a\sim\pi_i}\left[\left(\frac{\pi_{i+1}}{\pi_i} - 1\right)\left(Q^{\pi_i}(s,a) - \hat{Q}^{\pi_i}(s,a)\right)\right]\right| \\
=&\left|\mathbb{E}_{s\sim d_{\pi_i}, a\sim\pi_i}\left[\left(\frac{1}{Z(s)}\exp\left(\frac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}\right) - 1\right)\left(Q^{\pi_i}(s,a) - \hat{Q}^{\pi_i}(s,a)\right)\right]\right| \\
\le&\, \mathbb{E}_{s\sim d_{\pi_i}, a\sim\pi_i}\left[\left|\frac{1}{Z(s)}\exp\left(\frac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}\right) - 1\right|\left|Q^{\pi_i}(s,a) - \hat{Q}^{\pi_i}(s,a)\right|\right]
\end{aligned} \tag{45}$$

Now we bound the term $\left|\frac{1}{Z(s)}\exp\left(\frac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}\right) - 1\right|$.

Because

$$\left|\hat{A}^{\pi_i}(s,a)\right| = \left|\hat{Q}^{\pi_i}(s,a) - \mathbb{E}_{a\sim\pi_i}\left[\hat{Q}^{\pi_i}(s,a)\right]\right| \leq |V_{\max} - 0| = V_{\max}$$

$$Z(s) = \sum_a \pi_i(a|s)\exp(\frac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}) \leq \sum_a \pi_i(a|s)\exp(\frac{\alpha V_{\max}}{V_{\max}}) = \exp(\alpha) \tag{46}$$

On the other hand, by Jensen's inequality,

$$Z(s) = \sum_a \pi_i(a|s)\exp(\frac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}) \geq \exp(\frac{\alpha\sum_a \pi_i(a|s)\hat{A}^{\pi_i}(s,a)}{V_{\max}}) = 1 \tag{47}$$

Note that here $\sum_a \pi_i(a|s)\hat{A}^{\pi_i}(s,a) = 0$. It is because we calculate $\hat{A}^{\pi_i}$ from $\hat{Q}^{\pi_i}$ directly: $\hat{A}^{\pi_i}(s,a) := \hat{Q}^{\pi_i}(s,a) - \mathbb{E}_{a\sim\pi_i}\hat{Q}^{\pi_i}(s,a)$.

For the numerator, $\exp(-\alpha) \leq \exp\left(\frac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}\right) \leq \exp(\alpha)$

Therefore,

$$\exp(-2\alpha) \leq \frac{1}{Z(s)}\exp\left(\frac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}\right) \leq \exp(\alpha)$$

$$\left|\frac{1}{Z(s)}\exp\left(\frac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}\right) - 1\right| \leq \max\left\{1-\exp(-2\alpha), \exp(\alpha)-1\right\}$$

We assume $\alpha \in [0, 0.48)$ for small policy update. Then it holds that for $\alpha \in [0, 0.48)$, $1 - \exp(-2\alpha) \geq \exp(\alpha) - 1$.

Therefore,

$$\left|\frac{1}{Z(s)}\exp\left(\frac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}\right) - 1\right| \leq 1 - \exp(-2\alpha) \leq 2\alpha \tag{48}$$

So we bound the term $\left|\frac{1}{Z(s)}\exp\left(\frac{\hat{A}^{\pi_i}(s,a)}{\alpha}\right) - 1\right|$ in Eq.(45) with the constant 2α. Now we can conclude the proof by applying Theorem B.8 directly. □

Proposition B.10 (Trust Region, Proposition 4.10)**.** *For any π_{i+1}, π_i satisfying Eq.(16), the following policy difference bound holds* [5]*:*

$$\mathrm{D}_{\mathrm{TV}}(\pi_{i+1}\|\pi_i)[s] \leq \alpha, \forall s \tag{49}$$

$$\mathrm{D}_{\mathrm{KL}}(\pi_i\|\pi_{i+1})[s] \leq \alpha, \forall s \tag{50}$$

$$\mathrm{D}_{\mathrm{KL}}(\pi_{i+1}\|\pi_i)[s] \leq \frac{\alpha(e^{\alpha} - e^{-\alpha})}{2}, \forall s \tag{51}$$

Proof. For $\mathrm{D}_{\mathrm{TV}}(\pi_{i+1}\|\pi_i)$:

$$\begin{aligned}
\mathrm{D}_{\mathrm{TV}}(\pi_{i+1}\|\pi_i)[s] &= \frac{1}{2}\sum_a |\pi_{i+1}(a|s) - \pi_i(a|s)| \\
&= \frac{1}{2}\sum_a \left|\frac{\pi_{i+1}(a|s)}{\pi_i(a|s)} - 1\right|\pi_i(a|s) \\
&= \frac{1}{2}\sum_a \left|\frac{1}{Z(s)}\exp\left(\frac{\alpha\hat{A}^{\pi_i}(s,a)}{V_{\max}}\right) - 1\right|\pi_i(a|s) \qquad \text{by Eq.(16)} \\
&\leq \frac{1}{2}\sum_a 2\alpha\pi_i(a|s) \qquad \text{by Eq.(48)} \\
&= \alpha
\end{aligned}$$

[5]we assume $\alpha \leq 0.48$, which is not a necessary requirement, just for simplicity of the final results. A small α leads to a trust region update: the closer α is to 0, the closer π_{i+1} and π_i are to each other.

For $\mathrm{D_{KL}}(\pi_i \| \pi_{i+1})$:

$$\begin{aligned}
\mathrm{D_{KL}}(\pi_i \| \pi_{i+1})[s] &= \sum_a \pi_i(a|s) \log \frac{\pi_i(a|s)}{\pi_{i+1}(a|s)} \\
&= \sum_a \pi_i(a|s) \log \left[Z(s) \exp \left(\frac{-\alpha \hat{A}^{\pi_i}(s,a)}{V_{\max}} \right) \right] && \text{by Eq.(16)} \\
&= \log Z(s) - \sum_a \pi_i(a|s) \frac{\alpha \hat{A}^{\pi_i}(s,a)}{V_{\max}} \\
&= \log Z(s) \leq \alpha && \text{by Eq.(46)}
\end{aligned}$$

For $\mathrm{D_{KL}}(\pi_{i+1} \| \pi_i)$:

$$\begin{aligned}
\mathrm{D_{KL}}(\pi_{i+1} \| \pi_i)[s] &= \sum_a \pi_{i+1}(a|s) \log \frac{\pi_{i+1}(a|s)}{\pi_i(a|s)} \\
&= \sum_a \pi_{i+1}(a|s) \log \left[\frac{1}{Z(s)} \exp \left(\frac{\alpha \hat{A}^{\pi_i}(s,a)}{V_{\max}} \right) \right] && \text{by Eq.(16)} \\
&= \sum_a \pi_{i+1}(a|s) \frac{\alpha \hat{A}^{\pi_i}(s,a)}{V_{\max}} - \log Z(s) \\
&\leq \sum_a \frac{1}{Z(s)} \pi_i(a|s) \exp(\frac{\alpha \hat{A}^{\pi_i}(s,a)}{V_{\max}}) \frac{\alpha \hat{A}^{\pi_i}(s,a)}{V_{\max}} - 0 && \text{by Eq.(16) and Eq.(47)} \\
&\leq \sum_a \pi_i(a|s) \exp(\frac{\alpha \hat{A}^{\pi_i}(s,a)}{V_{\max}}) \frac{\alpha \hat{A}^{\pi_i}(s,a)}{V_{\max}} && \text{by Eq.(47)} \qquad (52)
\end{aligned}$$

An obvious upper bound of Eq.(52) is $\alpha \exp(\alpha)$ obtained by substituting $\hat{A}^{\pi_i}(s,a)$ with $V_{\max}$.

To derive a tighter upper bound of Eq.(52), consider the following problem, where $N = |\mathcal{A}|$:

$$\begin{aligned}
&\max \sum_{i=1}^{N} y_i x_i \exp(x_i) \\
&s.t. \sum_{i=1}^{N} y_i x_i = 0, \ \sum_{i=1}^{N} y_i = 1, \ -\alpha \leq x_i \leq \alpha
\end{aligned}$$

For $\alpha \in [0, 0.48)$ and for any fixed y, the optimization problem is convex with respect to x. Therefore, in the optimal solution, x_i can be found at the boundary. So we assume $x_1 \sim x_k = -\alpha$, $x_{k+1} \sim x_N = \alpha$. The optimization problem becomes:

$$\begin{aligned}
&\max \sum_{i=1}^{k} y_i(-\alpha)\exp(-\alpha) + \sum_{i=k+1}^{N} y_i \alpha \exp(\alpha) \\
&s.t. \sum_{i=1}^{k} y_i(-\alpha) + \sum_{i=k+1}^{N} y_i \alpha = 0, \ \sum_{i=1}^{N} y_i = 1
\end{aligned}$$

The constraint implies $\sum_{i=1}^{k} y_i = \sum_{i=k+1}^{N} y_i = 1/2$. Then the objective is equal to $\frac{\alpha(e^{\alpha} - e^{-\alpha})}{2}$, which concludes the proof. □

Theorem B.11 (Safe policy improvement for each step, Theorem 4.11)**.** *Under Assumption 4.7 and Assumption 4.8, for π_{i+1}, π_i satisfying Eq.(16), with $\epsilon^{\pi_{i+1}} := \max_s |\mathbb{E}_{a \sim \pi_{i+1}}[A^{\pi_i}(s,a)]|$, the following performance difference bound holds:*

$$\eta(\pi_{i+1}) - \eta(\pi_i) \geq \frac{V_{\max}}{(1-\gamma)\alpha} \mathbb{E}_{s \sim d^{\pi_i}}[\mathrm{D_{KL}}(\pi_{i+1} \| \pi_i)] - \frac{2\gamma \epsilon^{\pi_{i+1}}}{(1-\gamma)^2} \alpha - \frac{2\alpha}{1-\gamma} \left(\frac{1-\gamma^K}{1-\gamma} \sqrt{C\epsilon_{gb}} + \gamma^K V_{\max} \right)$$

Proof. We start with Lemma B.2:

$$\begin{aligned}
\eta\left(\pi_{i+1}\right)-\eta(\pi_i) &\geq \frac{1}{1-\gamma} \underset{\substack{s\sim d^{\pi_i}\\ a\sim\pi_{i+1}}}{\mathbb{E}}\left[A^{\pi_i}(s,a)-\frac{2\gamma\epsilon^{\pi_{i+1}}}{1-\gamma}\mathrm{D_{TV}}(\pi_{i+1}\|\pi_i)[s]\right] \\
&\geq \frac{1}{1-\gamma} \underset{\substack{s\sim d^{\pi_i}\\ a\sim\pi_{i+1}}}{\mathbb{E}} A^{\pi_i}(s,a)-\frac{2\gamma\epsilon^{\pi_{i+1}}}{(1-\gamma)^2}\alpha \qquad \text{by the TV bound in Proposition B.10} \\
&\geq \frac{1}{1-\gamma} \underset{\substack{s\sim d^{\pi_i}\\ a\sim\pi_{i+1}}}{\mathbb{E}} \hat{A}^{\pi_i}(s,a)-\frac{2\alpha}{1-\gamma}\left(\frac{1-\gamma^K}{1-\gamma}\sqrt{C\epsilon_{gb}}+\gamma^K V_{\max}\right)-\frac{2\gamma\epsilon^{\pi_{i+1}}}{(1-\gamma)^2}\alpha \qquad \text{by Lemma B.9}
\end{aligned}$$

Please note that π_{i+1} is the optimal solution of the optimization problem in Eq.(15):

$$\pi_{i+1}=\underset{\pi}{\operatorname{argmax}}\ \alpha \underset{\substack{s\sim d^{\pi_i}\\ a\sim\pi}}{\mathbb{E}}[\hat{A}^{\pi_i}(s,a)]-V_{\max}\underset{s\sim d^{\pi_i}}{\mathbb{E}}[\mathrm{D_{KL}}(\pi\|\pi_i)]$$

and π_i is a feasible solution that makes the objective 0. Therefore,

$$\alpha \underset{\substack{s\sim d^{\pi_i}\\ a\sim\pi_{i+1}}}{\mathbb{E}}[\hat{A}^{\pi_i}(s,a)]-V_{\max}\underset{s\sim d^{\pi_i}}{\mathbb{E}}[\mathrm{D_{KL}}(\pi_{i+1}\|\pi_i)]\geq 0$$

Now we can conclude the proof:

$$\eta(\pi_{i+1})-\eta(\pi_i)\geq\frac{V_{\max}}{(1-\gamma)\alpha}\mathbb{E}_{s\sim d^{\pi_i}}[\mathrm{D_{KL}}(\pi_{i+1}\|\pi_i)[s]]-\frac{2\gamma\epsilon^{\pi_{i+1}}}{(1-\gamma)^2}\alpha-\frac{2\alpha}{1-\gamma}\left(\frac{1-\gamma^K}{1-\gamma}\sqrt{C\epsilon_{gb}}+\gamma^K V_{\max}\right)$$

□

C. Derivations of the EAWBC Framework

At the i^{th} iteration, the unified algorithm solves the following constrained optimization problem to update the policy

$$\begin{aligned}
\pi_{i+1}=\underset{\pi}{\operatorname{argmax}}\ &\underset{a\sim\pi}{\mathbb{E}}[\hat{A}^{\pi_{\text{pe}}}(s,a)] \\
s.t.\ \mathrm{D_{KL}}(\pi\|\pi_{\text{base}})[s]\leq\epsilon,&\quad\sum_a\pi(a|s)=1,\ \forall s
\end{aligned} \tag{53}$$

The constrained optimization problem in Eq.(53) is convex, and the Lagrangian is:

$$\mathcal{L}(\pi,\lambda,\nu)=\underset{a\sim\pi}{\mathbb{E}}\left[\hat{A}^{\pi_{\text{pe}}}(s,a)\right]+\lambda[\epsilon-\mathrm{D_{KL}}(\pi(\cdot|s))\|\pi_{\text{base}}(\cdot|s))]+\nu\left(\sum_a\pi(a|s)-1\right) \tag{54}$$

The KKT condition gives:

$$\frac{\partial\mathcal{L}}{\partial\pi}=\hat{A}^{\pi_{\text{pe}}}(s,a)+\lambda\log\pi_{\text{base}}(a|s)-\lambda\log\pi(a|s)-\lambda+\nu=0 \tag{55}$$

Solving for π gives the closed form solution π^*:

$$\pi^*(a|s)=\pi_{\text{base}}(a|s)\exp(\frac{\hat{A}^{\pi_{\text{pe}}}(s,a)+\nu-\lambda}{\lambda}) \tag{56}$$

By the condition $\sum_a\pi^*(a|s)=1$, we have

$$\sum_a\pi_{\text{base}}(a|s)\exp(\frac{\hat{A}^{\pi_{\text{pe}}}(s,a)+\nu-\lambda}{\lambda})=1 \tag{57}$$

$$\Rightarrow\ \exp(\frac{\lambda-\nu}{\lambda})=\sum_a\pi_{\text{base}}(a|s)\exp(\frac{\hat{A}^{\pi_{\text{pe}}}(s,a)}{\lambda}) \tag{58}$$

$$\Rightarrow\ \nu=\lambda-\lambda\log\left[\sum_a\pi_{\text{base}}(a|s)\exp(\frac{\hat{A}^{\pi_{\text{pe}}}(s,a)}{\lambda})\right] \tag{59}$$

Now consider the dual problem to solve for the Lagrangian multiplier λ^*. By Substitute Eq.(55) into Eq.(54), we have

$$\max_{\pi} \mathcal{L} = \epsilon\lambda + \lambda - \nu \tag{60}$$

Then Substituting ν with Eq.(59), we obtain the dual function:

$$g(\lambda) = \max_{\pi} \mathcal{L} = \epsilon\lambda + \lambda \log \left[\sum_a \pi_{\text{base}}(a|s) \exp(\frac{\hat{A}^{\pi_{\text{pe}}}(s,a)}{\lambda}) \right] \tag{61}$$

Therefore, we can obtain $\lambda^*(s)$ by solving the following convex dual problem:

$$\lambda^*(s) = \underset{\lambda \geq 0}{\operatorname{argmin}} \ \epsilon\lambda + \lambda \log \left[\sum_a \pi_{\text{base}}(a|s) \exp \left(\frac{\hat{A}^{\pi_{\text{pe}}}(s,a)}{\lambda} \right) \right] \tag{62}$$

Now we replace the term $\exp(\frac{\nu-\lambda}{\lambda})$ in Eq.(56) with a per-state normalizing factor $Z(s)$ and finally present the analytical solution of the constrained optimization problem in Eq.(53):

$$\begin{aligned} \pi_{i+1}(a|s) =& \pi_{\text{base}}(a|s) f(s,a;\pi_{\text{pe}}) \\ \text{where } f(s,a;\pi_{\text{pe}}) :=& \frac{1}{Z(s)} \exp \left(\frac{\hat{A}^{\pi_{\text{pe}}}(s,a)}{\lambda^*(s)} \right) \\ Z(s) :=& \sum_a \pi_{\text{base}}(a|s) \exp \left(\frac{\hat{A}^{\pi_{\text{pe}}}(s,a)}{\lambda^*(s)} \right) \end{aligned} \tag{63}$$

In practice, the non-parametric solution in Eq.(63) can be projected onto the parametric policy class by minimizing the KL divergence:

$$\begin{aligned} &\underset{\phi}{\operatorname{argmin}} \ \mathop{\mathbb{E}}_{s\sim\mathcal{D}} \left[\mathrm{D}_{\mathrm{KL}}(\pi_{i+1}(\cdot|s) \| \pi_\phi(\cdot|s)) \right] && (64) \\ =& \underset{\phi}{\operatorname{argmin}} \mathop{\mathbb{E}}_{s\sim\mathcal{D}, a\sim\pi_{i+1}(\cdot|s)} \left[\log \left(\frac{\pi_{i+1}(a|s)}{\pi_\phi(a|s)} \right) \right] && (65) \\ =& \underset{\phi}{\operatorname{argmin}} \mathop{\mathbb{E}}_{s\sim\mathcal{D}, a\sim\pi_{base}(\cdot|s)} \left[f(s,a;\pi_{\text{pe}}) \log \left(\frac{\pi_{i+1}(a|s)}{\pi_\phi(a|s)} \right) \right] \quad \text{by Eq.(63)} && (66) \\ =& \underset{\phi}{\operatorname{argmax}} \mathop{\mathbb{E}}_{s\sim\mathcal{D}, a\sim\pi_{base}(\cdot|s)} \left[f(s,a;\pi_{\text{pe}}) \log(\pi_\phi(a|s)) \right] && (67) \end{aligned}$$

D. Experimental Details and Extended Results

D.1. Reasonableness for STR to omit $Z(s)$ in practice

In Eq.(20), as all prior EAWBC works, we omit the normalization factor $Z(s)$, because it only affects the relative weight of different states in the training objective, not different actions. The EAWBC objective in Eq.(12) is derived from the minimization of $\mathrm{D}_{\mathrm{KL}}(\pi_{i+1}(\cdot|s)\|\pi_\phi(\cdot|s))$. Since the weight at each state do not have specific meaning, we can minimize this KL divergence under any distribution whose support is equal to $d^{\pi_{i+1}}(s)$. And because of the equal-support property of STR $\mathrm{supp}(d^{\pi_{i+1}}(\cdot)) = \mathrm{supp}(d^\beta(\cdot))$ by Proposition 4.3, $d^\beta(s)$ is a qualified distribution, which allows us to sample directly from $\mathcal{D}$ to optimize. Since the density of d^β already differs from $d^{\pi_{i+1}}$, it has no meaning to restore the correct $d^\beta(s)$ density by the normalization factor $Z(s)$, which is empirically hard to estimate and will introduce more instability.

D.2. Experimental Details

For the MuJoCo locomotion tasks, we average returns over 10 evaluation trajectories and 5 random seeds, while for the Ant Maze tasks, we average over 100 evaluation trajectories and 5 random seeds. Following the suggestions of the authors of the dataset, we subtract 1 from the rewards for the Ant Maze datasets. We choose TD3 (Fujimoto et al., 2018) as our base algorithm and optimize a deterministic policy. To compute the importance sampling ratio, we need the density of

Table 3. Hyperparameters of policy training in STR.

	Hyperparameter	Value
STR	Optimizer	Adam (Kingma & Ba, 2014)
	Critic learning rate	3×10^{-4}
	Actor learning rate	3×10^{-4} with cosine schedule
	Batch size	256
	Discount factor	0.99
	Number of iterations	10^6
	Target update rate τ	0.005
	Policy update frequency	2
	Number of Critics	4
	Temperature λ	$\{0.5, 2\}$ for Gym-MuJoCo $\{0.1\}$ for AntMaze
	Variance of Gaussian Policy	0.1
Architecture	Actor	input-256-256-output
	Critic	input-256-256-1

any action under the deterministic policy. For this, we assume all policies are Gaussian with a fixed variance 0.1. Note that the only hyperparameter we tuned is the temperature λ. We use $\lambda = 0.1$ for Ant Maze tasks and $\lambda = \{0.5, 2\}$ for MuJoCo locomotion tasks ($\lambda = 2$ for expert and medium-expert datasets, $\lambda = 0.5$ for medium, medium-replay, random datasets). And following previous work (Brandfonbrener et al., 2021), we clip exponentiated advantages to $(-\infty, 100]$. All hyperparameters are included in Table 3.

D.3. From Theoretical to Practical

The practical STR algorithm takes larger update steps (smaller λ) than what theory recommends, which is common among trust region methods. In addition, the behavior density is estimated using a specific model, which will inevitably have errors. However, compared with other algorithms that require the behavior density, STR is less susceptible to such errors. This is because STR only needs to query the behavior density of the in-dataset (s, a) pairs, i.e., $\hat{\beta}(a|s)$ where $(s, a) \sim \mathcal{D}$, and therefore does not require much generalization ability of the model, making it relatively easier to estimate accurately.

D.4. KL Divergence between Trained Policy and Behavior Policy.

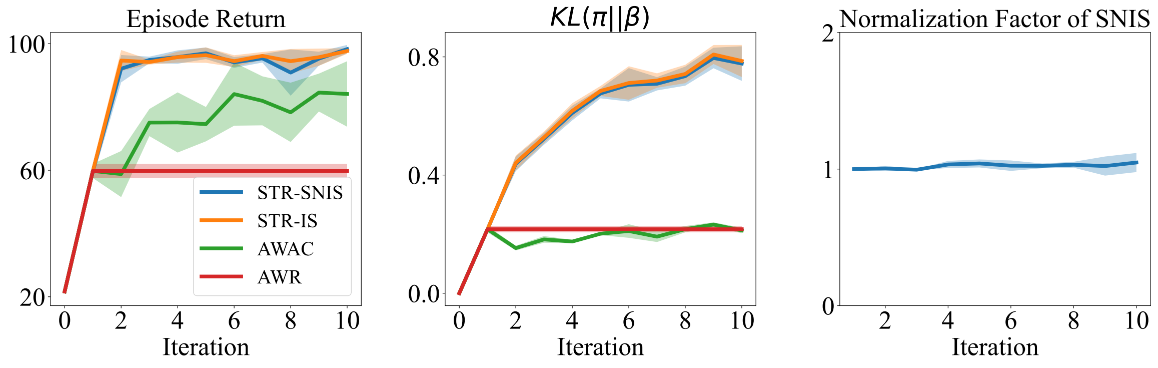
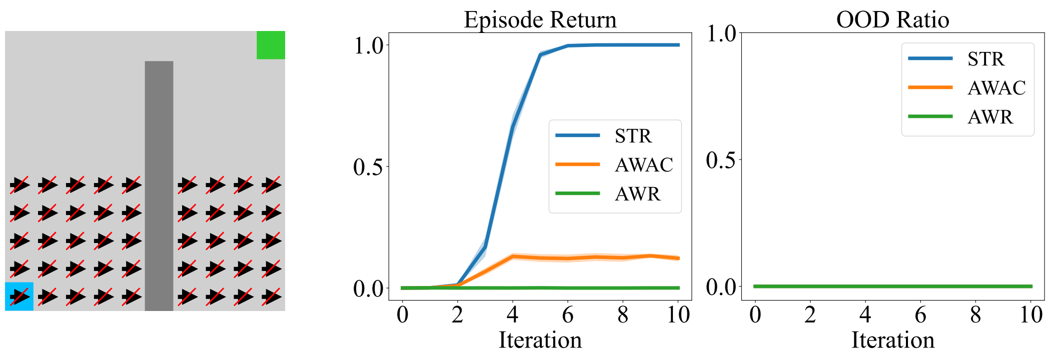


Figure 4. Extended Results on hopper-medium-replay-v2. The experimental setup is the same as that in Section 5.2.

In this section, we investigate the KL divergence between the trained policy and the behavior policy under various EAWBC algorithms. We conduct experiments in the same setup as Section 5.2 (Figure 1) for hopper-medium-replay-v2. Both policy evaluation and policy improvement are trained to convergence at each iteration, and all algorithms adopt the same hyperparameter that controls the constraint strength. The results are shown in Figure 4. Compared to STR-SNIS and STR-IS, AWAC and AWR have substantially smaller $\mathrm{KL}(\pi||\beta)$ and inferior performance due to their implicit density constraint. On the other hand, $\mathrm{KL}(\pi||\beta)$ of STR-IS and STR-SNIS increases significantly with iteration. With a less restrictive support constraint, STR is able to deviate more from the behavior policy (larger $\mathrm{KL}(\pi||\beta)$) to achieve better performance. Note that STR-IS and STR-SNIS yield similar results. This is because the normalization factor of SNIS is actually very close to 1. Therefore, IS plays a critical role in STR by relaxing the density constraint to support constraint while SNIS is a non-critical component. In addition, compared with IS, the variance reduction effect of SNIS does not implicitly decreases $\mathrm{KL}(\pi||\beta)$ (Chatterjee & Diaconis, 2018).

D.5. Experimental Verification of Theories in the Tabular Setting



(a) Toy Maze (b) Learning Curves on the Toy Maze

Figure 5. (a) The maze environment with OOD actions. (b) The Episode Return of STR, AWAC and AWR (averaged over 1000 trajectories). (c) The OOD Ratio of STR, AWAC and AWR. The curves are averaged over 5 seeds, with the shaded area representing the standard deviation across seeds. SVR guarantees strict policy improvement until convergence to the optimal support-constrained policy.

To verify the theories in Section 4.1, we conduct a 10×10 maze experiment. As depicted in Figure 5a, the task is to navigate from bottom-left to top-right, with a wall in the middle. The agent receives a reward of 1 upon reaching the goal. Episodes are terminated after 25 steps and γ is set to 0.9. We first collect $10,000$ transitions using a random policy. Then we remove all the transitions containing rightward actions in the lower half of the maze, so that the rightward action in that region is Out-of-Distribution (OOD). For some policy π, the OOD Ratio indicates the proportion of (s, a) not in the dataset among all (s, a) pairs satisfying $\pi(a|s) > 0$. The results are shown in Figure 5b. The performance of STR increases monotonically with iteration until it converges to the optimal policy, verifying Theorem 4.6. Furthermore, the policy obtained by STR is completely within the behavior support (0 OOD Ratio). For AWAC and AWR, due to being implicitly subject to the more restrictive $\mathrm{KL}(\pi||\beta)$ density constraint, their policies are also within the support. However, they do not have strictly increasing performance and have no guarantees at convergence. This is consistent with our theoretical analysis (Table 1).

D.6. Learning Curves of STR on MuJoCo and Antmaze Tasks

Learning curves of STR on MuJoCo locomotion tasks and Antmaze tasks are presented in Figure 6 and Figure 7 respectively. The curves are averaged over 5 seeds, with the shaded area representing the standard deviation across seeds.

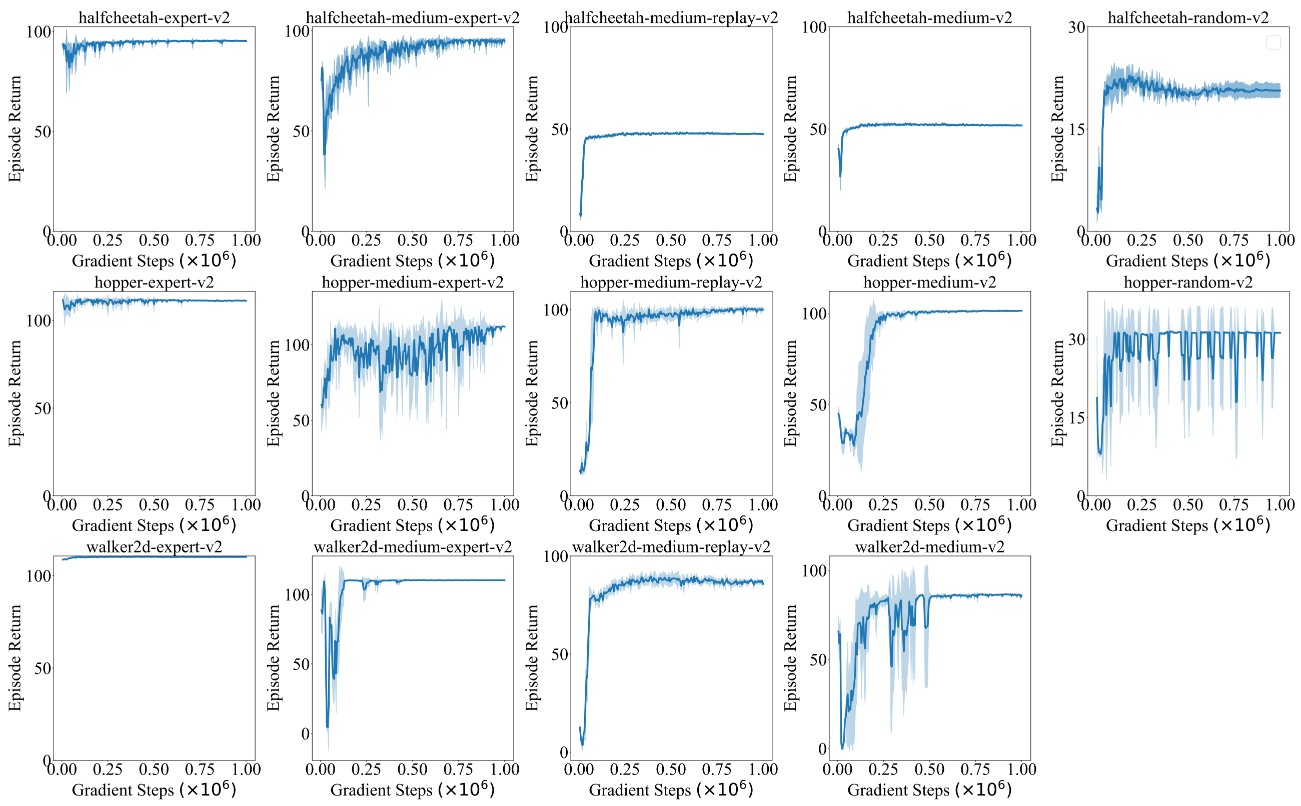


Figure 6. Learning Curves of STR on MuJoCo Locomotion Tasks.

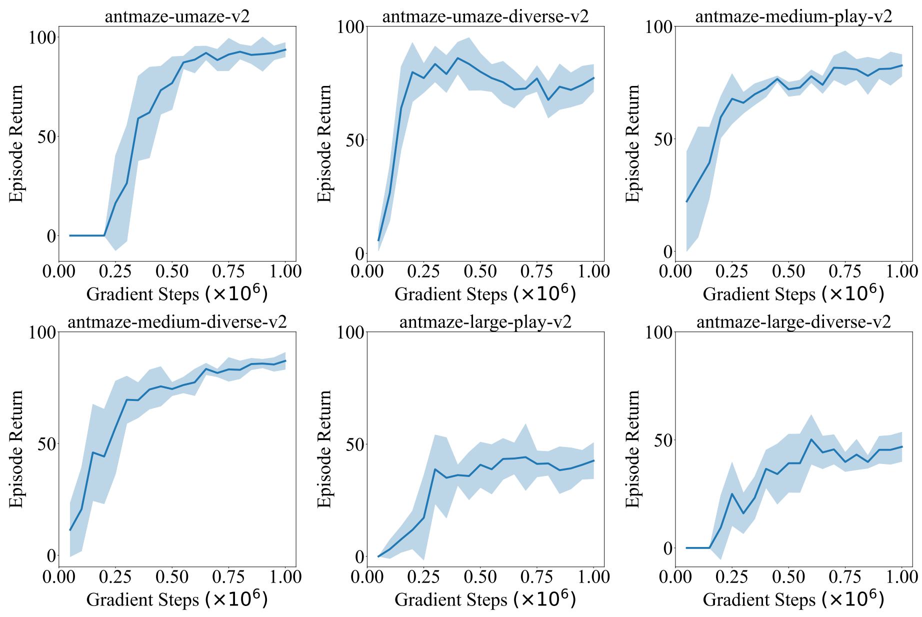


Figure 7. Learning Curves of STR on AntMaze Tasks.