Density-Scaled Regularization for Offline Reinforcement Learning

Anonymous authors
Paper under double-blind review

Abstract

Value-based offline RL methods are prone to overestimate the values of out-of-distribution (OOD) actions, and this is often addressed by regularizing the action-value function in the Bellman update. However, existing regularization methods can suffer from being too conservative, which can arise from over-penalizing the values for both in-distribution actions and out-of-support actions. We present a new regularization method for offline value-based methods, called Density-Scaled (DS) regularization, which penalizes the value function based on the relative action density of the behavior policy. We show a theoretical connection between our method and the existing Supported Value Regularization (SVR) method, demonstrating how the SVR solution for policy evaluation can be viewed as a limiting case of the solution from the DS regularized problem. Empirical results demonstrate that the DS penalty is competitive with the state-of-the-art techniques, more robust to misestimation of the behavior density compared to SVR, and allows greater flexibility in learning hyperparameters associated with the behavior policy.

1 Introduction

Offline reinforcement learning (offline RL) studies how an agent can learn an optimal policy for sequential decision-making from a dataset of a (typically suboptimal) policy's interactions with the environment (Levine et al., 2020). While standard online RL requires the agent to iteratively collects data through trial-and-error interaction and use this experience to improve its behavior over time, direct interaction with the environment can be expensive, unsafe, or impractical, for many real-world applications, such as healthcare, robotics, or recommendation systems. Offline RL addresses this limitation by learning policies solely from a fixed dataset of previously collected transitions, without further environment interaction. By leveraging pre-existing logs of experience, offline RL enables data reuse and avoids the risks associated with exploratory behavior in safety-critical domains.

Nevertheless, learning from static datasets introduces unique challenges, particularly due to the distributional mismatch between the learned policy and the behavior policy that generated the data (Fujimoto et al., 2018; Levine et al., 2020). Typical model-free RL algorithms estimate a value function using Bellman backups and subsequently derive a policy by maximizing these estimated values. However, in the offline setting, the learned policy may query the value function on actions that are poorly represented or entirely absent in the dataset. Function approximation can therefore extrapolate arbitrarily to these out-of-distribution (OOD) actions, and the maximization in the Bellman operator may amplify such errors, resulting in systematically over-optimistic value estimates that propagate through bootstrapping (Kumar et al., 2019; Fujimoto et al., 2019).

To improve generalization beyond experiences in the offline dataset, prior works have proposed several forms of regularization (Kostrikov et al., 2021a; Kumar et al., 2020; Wu et al., 2019; Xu et al., 2023), as discussed in more detail in Section 2. We focus on value-regularization approaches, which attempt to directly control extrapolation by enforcing conservative or pessimistic value estimates for OOD actions. A representative approach is Conservative Q-Learning (CQL) (Kumar et al., 2020), which adds an explicit regularization term that broadly penalizes the value of all actions proposed by the learned policy. This can lead to overly

pessimistic value estimates since actions within the data distribution are also penalized, and the specific form of the penalty can in theory lead to unbounded values. On the other hand, support-based methods, such as the state-of-the-art Supported Value Regularization (SVR) (Mao et al., 2023), aim to penalize the value of actions outside the support of the behavior policy without affecting the in-support actions. While this is empirically more effective than CQL and prior methods, SVR does not consider the relative uncertainty of value estimates from the action density in the dataset – for example, sparsely supported regions of the action space are not penalized any more than well-supported regions. These contrasting approaches highlight the trade-off between pessimism and generalization: pessimistic penalties that consider all actions can risk over-conservatism, whereas strict support constraints can forgo penalizing estimates in poorly-supported regions that may benefit from conservatism.

In this work, we provide a different approach for managing this tradeoff that combines advantageous aspects of both paradigms. By comparing the update solutions for CQL and SVR, we gain insight into their theoretical advantages and limitations, and devise a new regularization method that is more nuanced with respect to in-distribution and out-of-support actions. Our approach, termed Density-Scaled (DS) regularization, is a penalty that varies the level of conservatism with the density of actions in the dataset. While our penalty term may appear simple and heuristic at first, we analyze the solution and establish an interesting interpolating property that connects it to the existing support-constraint solution of SVR, and results in improved numerical stability compared to it. In summary, we make the following contributions:

- We introduce the Density-Scaled regularizer for offline policy evaluation, which penalizes the value function smoothly based on an estimated model for the behavior density. It can be implemented and applied in a straightforward manner for many model-free offline RL methods.
- We provide theoretical analysis for our method, proving that our penalty produces conservative estimates that lower bound the true Q-function. We also demonstrate that the solution to our regularized Bellman update resembles a generalized version of the SVR solution.
- We empirically demonstrate the effectiveness of our method across offline RL benchmarks and visual datasets with high-dimensional pixel observations. Furthermore, our ablation studies demonstrate that this penalty is more robust to inaccurate estimation of behavior policy compared to SVR.

2 Related Work

Our work falls under model-free offline RL. In this setting, value regularization and policy regularization are two commonly used approaches for alleviating the extrapolation error.

Value regularization methods aim to alleviate overestimation errors in critic learning. One class of approaches directly penalizes Q-values during training, typically by adding a conservative regularizer (Kumar et al., 2020; Kostrikov et al., 2021b; Mao et al., 2023; Shimizu et al., 2024). Ensemble-based methods reduce overestimation by taking the minimum over multiple Q-functions (Agarwal et al., 2020; Kumar et al., 2019), or by incorporating uncertainty quantification into the target (O'Donoghue et al., 2018; Wu et al., 2021; An et al., 2021). Such methods do not explicitly identify the OOD region but instead induce conservatism based on the policy density. In contrast, support-based regularization (Mao et al., 2023; Wu et al., 2022) aim to explicitly constrain the values of the value function on out-of-support regions. A different class of methods are referred to as in-sample or in-support learning methods, which avoid querying unseen actions in the Bellman backup altogether (Kostrikov et al., 2021a; Mao et al., 2024). This includes IQL (Kostrikov et al., 2021a), which employs expectile regression to approximate an upper expectile of the in-sample value distribution, and IVR (Xu et al., 2023), which considers in-sample optimization objectives derived from a behavior-regularized MDP problem. Our method contributes a novel explicit regularizer, which encourages smooth conservative estimates of the Q-function based on the behavior density over the entire action space.

Complementary to value regularization, *policy regularization* methods constrain the learned policy to remain close to the behavior policy, which prevents the Q-function from being evaluated on out-of-distribution actions, reducing errors in the Bellman update. Policy constraints are typically implemented via divergence penalties, such as KL or MMD regularization (Wu et al., 2019; Kumar et al., 2019; Jaques et al., 2019),

or behavior cloning terms, such as TD3-BC (Fujimoto & Gu, 2021) and ReBRAC (Tarasov et al., 2023). Another line of work known as one-step RL considers learning a policy with single regularized update step (Brandfonbrener et al., 2021; Park et al., 2025; Peng et al., 2019) Our method is compatible with existing policy regularization approaches and we evaluate an instance where both policy regularization and the DS penalty is used simultaneously.

3 Preliminaries

Markov Decision Processes (MDPs) provide a general mathematical framework for sequential decision-making. An MDP is defined by a tuple $M = (S, A, P, R, \gamma)$, where S is the state space, A the action space, $P(s' \mid s, a)$ the transition probability distribution for the dynamics of the system, $R(s, a)$ the reward function, and $\gamma \in [0, 1)$ the discount factor. The objective is to find the optimal policy π^* that maximizes the expected cumulative reward $J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 \sim \mu_0\right]$, where the expectation under π means $a_t \sim \pi(\cdot \mid s_t), s_{t+1} \sim P(\cdot \mid s_t, a_t)$, and μ_0 is the initial state distribution. Given a policy π, the state-action value function $Q^\pi(s, a)$ represents the expected cumulative reward when taking action a in state s and then following policy π: $Q^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a\right]$. The Q-function for a given policy can be computed by iterating the Bellman operator T^π, defined by $T^\pi Q(s, a) = r(s, a) + \gamma P^\pi Q(s, a)$, where P^π is the policy transition operator defined by $P^\pi Q(s, a) = \mathbb{E}_{s' \sim P(\cdot|s,a), a' \sim \pi(\cdot|s')}[Q(s', a')]$.

In offline RL, the agent is given a static dataset $D = \{(s, a, r, s')_i\}_{i=1}^N$, where the transitions are collected from the trajectories of an unknown behavior policy $\beta(a \mid s)$ interacting in the MDP. We denote $d_\beta(s)$ the state visitation frequency of β; the state-action pairs (s, a) from the dataset D are sampled from $d_\beta(s) \cdot \beta(a \mid s)$. Standard (offline) actor-critic methods alternate between evaluating the action value function of the current policy, and improving the current policy using the action value function. Specifically, at iteration k, given the current policy $\pi^{(k)}$ and the action value function $Q^{(k)}$, we perform policy evaluation and policy improvement as follows:

$$Q^{(k+1)} = \arg\min_Q \ \mathbb{E}_{(s,a,s') \sim D} \Big[Q(s, a) - \big(r(s, a) + \gamma \mathbb{E}_{a' \sim \pi^{(k)}(\cdot|s')}[Q^{(k)}(s', a')] \big) \Big]^2 \quad \text{(policy evaluation)} \tag{1}$$

$$\pi^{(k+1)} = \arg\max_\pi \ \mathbb{E}_{s \sim D, a \sim \pi(\cdot|s)}[Q^{(k+1)}(s, a)] \quad \text{(policy improvement)} \tag{2}$$

Offline RL methods that perform vanilla policy evaluation and improvement based on these steps suffer from distribution shift during training, due to the evaluation of the Q-function on out-of-distribution (OOD) actions in the Bellman error. Specifically, the target Q-function $Q^{(k)}$ is evaluated on samples from the current policy $\pi^{(k)}$ to construct the Bellman target, i.e. $a' \sim \pi^{(k)}(\cdot \mid s')$, but the dataset contains only actions sampled from the behavior policy, i.e. $a' \sim \beta(\cdot \mid s')$, making its accuracy depend on Q-value estimates for actions outside the training distribution. This discrepancy between π and β can cause erroneous target Q-values, and overestimation occurs when the policy improvement step seeks to find a policy that maximizes $\mathbb{E}_{s \sim D, a \sim \pi(\cdot|s)}[Q(s, a)]$ (Kumar et al., 2019; Levine et al., 2020). This issues motivates the need for regularized objectives and conservative estimates of the value function.

4 Density-Scaled Regularization

We describe our Density-Scaled (DS) method for regularizing the policy evaluation step. We first motivate our work by discussing relevant existing forms of regularization and potential areas for improvement. Then, we construct our DS penalty and highlight its main theoretical results and appealing properties. Finally, we provide details for a practical implementation with Q-learning.

4.1 Motivation

We first describe how recent methods address the OOD and overestimation problem by regularizing the policy evaluation step. To learn the Q-function offline using samples from the behavior policy, we consider the squared Bellman error $B(Q) := \frac{1}{2}\mathbb{E}_{s \sim D, a \sim \beta(\cdot|s)}[(Q(s, a) - T^\pi \hat{Q}(s, a))^2]$, where $\hat{Q}$ is a target copy of Q, treated as a constant in optimization, and π is the current policy, which is the policy to be evaluated (we

leave the functional dependence on π implicit). We consider value regularization methods that augment the Bellman error with a regularization term R, leading to the optimization problem

$$\arg\min_{Q} B(Q) + \alpha \mathrm{R}(Q) \tag{3}$$

where α controls the strength of regularization. The regularization term is chosen in such a way so as to penalize the Q-function to avoid overestimation from OOD actions. Different choices of R lead to quantitative differences in the solutions to the regularized policy update (3). Here, we investigate the limitations of recent methods, highlighting areas where the behavior of their solutions may be sub-optimal. In particular, we consider the regularization term for two recent methods, CQL (Kumar et al., 2020) and SVR (Mao et al., 2023) and compare their corresponding solutions.

In CQL, the regularization term is given by

$$\mathrm{R}_{\mathrm{CQL}}(Q) = \mathbb{E}_{s\sim D, a\sim\pi(\cdot|s)}[Q(s,a)] - \mathbb{E}_{s\sim D, a\sim\beta(\cdot|s)}[Q(s,a)], \tag{4}$$

and an optimal solution to the regularized evaluation problem in Equation (3) is given by

$$Q_{\mathrm{CQL}}(s,a) = \begin{cases} T^\pi Q(s,a) - \alpha\left(\frac{\pi(a|s)}{\beta(a|s)} - 1\right) & \beta(a \mid s) > 0 \\ -\infty & \beta(a \mid s) = 0, \pi(a \mid s) > 0 \\ Q(s,a) & \beta(a \mid s) = \pi(a \mid s) = 0 \end{cases}$$

for all $a \in A$ and $s \in D$. In SVR, the regularization term is given by

$$\mathrm{R}_{\mathrm{SVR}}(Q) = \mathbb{E}_{s\sim D}\left[\sum_{a\notin \mathrm{supp}(\beta)} \mu(a \mid s)(Q(s,a) - Q_{\min})^2\right] \tag{5}$$

$$= \mathbb{E}_{s\sim D, a\sim\mu(\cdot|s)}[(Q(s,a) - Q_{\min})^2] - \mathbb{E}_{s\sim D, a\sim\beta(\cdot|s)}\left[\frac{\mu(a \mid s)}{\beta(a \mid s)}(Q(s,a) - Q_{\min})^2\right] \tag{6}$$

with $Q_{\min}$ being the minimum Q-value (for any policy) in the dataset and μ is a policy with a wider support than β, set to be the current policy with a higher variance. The optimal solution to the regularized evaluation problem in Equation (3) is given by (Mao et al., 2023, Eq. (7))

$$Q_{\mathrm{SVR}}(s,a) = \begin{cases} T^\pi Q(s,a) & \beta(a \mid s) > 0 \\ Q_{\min} & \beta(a \mid s) = 0 \end{cases}$$

for all $a \in A$ and $s \in D$. We provide formal statements and expanded proofs of the above solutions in Appendix A.2.

While CQL smoothly reduces the Q-function by a term that depends on the ratio $\frac{\pi(a|s)}{\beta(a|s)}$, it is not lower-bounded, and can overly penalize the Q-function even for in-distribution actions. On the other hand, the solution provided by SVR is discontinuous, entirely avoiding penalization of in-distribution values. Although the SVR solution ensures that in-distribution Q-values are not penalized, it does not consider the possibility that in-distribution regions can still lead to poor estimates of the value function, especially in cases where the action density of the behavior policy is low. We are thus motivated to reconcile the positive aspects of both methods, seeking a regularization term that can smoothly vary the level of penalization based on the action density, while avoiding over-conservatism. To this end, we introduce our Density-Scaled (DS) penalty.

4.2 Density-Scaled Regularization

Based on the previous considerations, we construct a regularization term that achieves two properties: (1) underestimates the Q-function proportionately to the behavior density, which is a natural measure for how 'in-distribution' an action is; and (2) lower bounds the Q-function on out-of-support actions. Our DS method uses the following penalty:

$$\mathrm{R}_{\mathrm{DS}}(Q) = \mathbb{E}_{s\sim D, a\sim\pi(\cdot|s)}\left[(\max_a \beta(a \mid s) - \beta(a \mid s))(Q(s,a) - Q_{\min})^2\right]. \tag{7}$$

The regularization term involves an expectation over the policy π, which may be the current policy or an arbitrary policy chosen to weight the penalty term for actions deemed important. In general, to penalize OOD actions effectively, π should be chosen to have greater support than β or a wider distribution.

In practice, computing the penalty requires an explicit estimate $\hat{\beta}$ of the behavior density. A smooth estimator enables us to interpolate the action density from the dataset D and query it on arbitrary (including OOD) actions outside the dataset. Here, we learn a smooth behavior policy $\hat{\beta}(a|s;\psi)$ parameterized by ψ using neural networks, which are powerful interpolators.

We consider policy evaluation with this penalty and characterize its solution in Theorem 1.

Theorem 1 (DS update)**.** *Given a policy $\pi(a \mid s)$, behavior policy $\beta(a \mid s)$, and regularization parameter α, let $C_s = \max_a \beta(a \mid s)$ for each s, and $k_{s,a} = \left(\frac{\pi(a|s)}{\beta(a|s)} - \frac{\pi(a|s)}{C_s}\right)$. For $a \in A$ and $s \in D$, the solution to*

$$\arg\min_Q B(Q) + \frac{\alpha}{2}\mathbb{E}_{s\sim D, a\sim\pi(\cdot|s)}\left[\left(1 - \frac{\beta(a \mid s)}{C_s}\right)(Q(s,a) - Q_{\min})^2\right] \tag{8}$$

is given by

$$Q_{\mathrm{DS}}(s,a) = \begin{cases} T^\pi Q(s,a) & \beta(a \mid s) = C_s \\ \frac{1}{1+\alpha k_{s,a}} T^\pi Q(s,a) + \frac{\alpha k_{s,a}}{1+\alpha k_{s,a}} Q_{\min} & 0 < \beta(a \mid s) < C_s \\ Q_{\min} & \beta(a \mid s) = 0 \end{cases} \tag{9}$$

All proofs are in Appendix A.2. Inspecting the solution, we see that the DS objective results in the standard policy target $T^\pi Q(s,a)$ when the action a is at the mode of the behavior distribution. When $0 < \beta(a \mid s) < C_s$, the solution is given by a convex combination between $T^\pi Q(s,a)$ and $Q_{\min}$. On OOD actions, the Q-function is maximally penalized to $Q_{\min}$. An interesting property is that it can be viewed as a generalization, or smooth interpolation, of SVR.

Observation 1. *Q_{DS} tends to Q_{SVR} as $\alpha \to 0$.*

This provides another way to compute an approximate support-constraint solution. Compared to the SVR's regularization term (Eq. (6)), our DS penalty avoids computation of the importance ratio $\frac{\mu(a|s)}{\beta(a|s)}$, which can potentially have high variance. Later, we investigate this difference empirically by designing an experiment to compare our method when the variance of behavior density estimator is varied. In addition, compared to CQL, DS is lower bounded (by $Q_{\min}$) on out-of-support actions, and thus avoid excessive pessimism.

We now consider the properties of the operator defined by the DS solution of Theorem 1. Specifically, given a policy π, we define the operator T^π_{DS} by $T^\pi_{\mathrm{DS}} Q(s,a) := Q_{\mathrm{DS}}(s,a)$. In Proposition 1, we show that the operator preserves the contraction property, similar to the standard Bellman operator. We also demonstrate in Proposition 2 that repeated evaluations lead to a fixed point that satisfies a lower bound of the true value function.

Proposition 1 (Contraction)**.** *The operator $T^\pi_{\mathrm{DS}} Q(s,a)$ is a γ-contraction operator in the L_∞ norm.*

Proposition 2 (Fixed point)**.** *For any π, the fixed point of T^π_{DS}, denoted by f, exists and satisfies*

$$\begin{cases} Q_{\min} \le f(s,a) \le Q^\pi(s,a) & \beta(a \mid s) > 0 \\ f(s,a) = Q_{\min} & \beta(a \mid s) = 0 \end{cases}$$

These results ensure that policy evaluation with the DS penalty results in conservative estimates of the Q-function, while being lower bounded by the minimum Q value on out-of-support regions.

4.3 Practical Implementation

We now describe a practical implementation of our method, based on a conventional deep Q-learning algorithm (e.g. Mnih et al. (2013); Haarnoja et al. (2018)). It proceeds by first estimating the behavior policy from the dataset. Next, we perform approximate policy evaluation and policy improvement in alternating

Algorithm 1 Density-Scaled Q-learning

Require: Offline dataset $D = \{(s, a, r, s')\}$, DS regularization coefficient α, target averaging rate τ
  Initialize Q-network $Q(s, a; \theta)$, target Q-network $Q'(s, a; \theta')$, behavior policy $\beta(a \mid s; \psi)$, policy network $\pi(a \mid s; \phi)$
  ▷ Estimate behavior policy via behavior cloning
  for gradient_step $= 1$ to M **do**
    Sample random minibatch $(s, a) \sim D$
    Update behavior policy parameters ψ by minimizing Eq. (10)
  end for
  ▷ Policy and critic training
  for gradient_step $= 1$ to N **do**
    Sample random minibatch of $(s, a, r, s') \sim D$
    Update critic parameters θ by minimizing Eq. (11)
    Update policy parameters ϕ by minimizing Eq. (12)
    Update target network parameters: $\theta' \leftarrow (1 - \tau)\theta' + \tau\theta$
  end for

steps, where the objective for the Q-function is augmented with the DS penalty. The full algorithm is provided in Algorithm 1.

Behavior policy. Given the dataset $D = \{(s, a, r, s')\}$, we estimate the behavior policy β by learning a parameterized model $\beta(a \mid s; \psi)$ by maximum likelihood estimation, using the following loss:

$$\arg\min_{\psi} L_\beta(\psi) = -\mathbb{E}_{(s,a)\sim D} \log[\beta(a \mid s; \psi)]. \tag{10}$$

We use a multivariate Gaussian model for β, i.e. $a \mid s \sim N(\mu(s; \psi), \sigma^2(s; \psi))$.

Policy evaluation. Given the current policy $\pi(a \mid s; \phi_k)$, the estimated behavior policy $\beta(a \mid s; \psi)$, the current Q-network $Q(s, a; \theta)$, target Q-network $Q'(s, a; \theta')$, and a noise standard deviation σ, we update the parameters of the critic by minimizing the following loss:

$$\begin{aligned} \theta_{k+1} \leftarrow \arg\min_{\theta} L_Q(\theta) = \mathbb{E}_{(s,a,s')\sim D}[(Q(s, a; \theta) - r(s, a) - \gamma \mathbb{E}_{a'\sim\pi(\cdot|s';\phi_k)} Q'(s', a'; \theta'_k))^2] \\ + \alpha \mathbb{E}_{s\sim D, a\sim\pi_\sigma(\cdot|s;\psi)} \left[\left(1 - \frac{\beta(a \mid s; \phi_k)}{C_s}\right) (Q(s, a; \theta) - Q_{\min})^2 \right]. \end{aligned} \tag{11}$$

When β is the Gaussian model, we have $C_s = (\sigma(s)\sqrt{2\pi})^{-1}$. We set $Q_{\min} = r_{\min}/(1 - \gamma)$, where $r_{\min}$ is the minimum reward in the dataset. Similar to SVR, the sampling distribution in the penalty $\pi_\sigma(\cdot|s, \phi_k)$ is derived from the current policy with a fixed (larger) variance term: $\pi_\sigma(\cdot|s, \phi_k) \sim N(\pi(s; \phi_k), \sigma^2)$ where $\pi(s; \phi_k)$ is an action sampled from the policy – σ is set to encourage sampling OOD actions more frequently. The target network parameters θ'_k are updated via Polyak averaging, that is, $\theta'_k \leftarrow \tau\theta_k + (1 - \tau)\theta'_k$ where $\tau \in (0, 1)$ is the interpolation factor.

Policy improvement. Using the current estimate of the Q-function $Q(s, a; \theta)$, we update the policy parameters ϕ by minimizing the following loss:

$$\phi_{k+1} \leftarrow \arg\min_{\phi} L_\pi(\phi) = -\mathbb{E}_{(s,a)\sim D, a'\sim\pi(\cdot|s;\phi_k)}[Q(s, a'; \theta_{k+1}) - \lambda \| a - a' \|^2], \tag{12}$$

which follows the standard policy update loss with a behavior cloning term $\lambda \| a - a' \|^2$ (Tarasov et al., 2023) which we turn on only for the vision datasets in our experiments.

5 Experiments

We evaluate the effectiveness of our DS algorithm and compare it to previous offline RL methods including the state-of-the-art SVR method. We benchmark our method on offline continuous control problems that

contain both vector observations and high-dimensional pixel observations. We also investigate and compare the robustness and numerical stability of our method to SVR by designing an experiment that modifies the behavior policy estimator. Finally, we study our method's sensitivity to the hyperparameter values.

5.1 Continuous control

D4RL. First, we benchmark a suite of continuous control tasks in D4RL (Fu et al., 2021). We compare our DS method with previous model-free baseline algorithms: BC, One-Step RL (Brandfonbrener et al., 2021), TD3+BC (Fujimoto & Gu, 2021), CQL (Kumar et al., 2020), IQL (Kostrikov et al., 2021a), and SVR (Mao et al., 2023). We derive the BC scores from (Mao et al., 2023). Here, we run our algorithm using only our critic regularization term, and do not use actor regularization (i.e. $\lambda = 0$). We tune the regularization strength α based on the difficulty of the environments. Full experimental details, including training parameters and hyperparameter choices, are provided in Appendix A.1.

Table 1 summarizes the average scores for each algorithm. In the Gym-Mujoco domains, DS is competitive with the state-of-the-art SVR across all tasks, slightly outperforming it on the total aggregrate score. In the Adroit domain, DS achieves the highest performance on the expert domain, but underperforms SVR in the human and cloned datasets. Overall, this demonstrates that the DS penalty is competitive with state-of-the-art methods, demonstrating that a smooth Density-Scaled penalty is comparable in performance to SVR, a support-constraint method.

Table 1: Normalized scores on the D4RL benchmarks. m = medium, m-r = medium-replay, m-e = medium-expert, e = expert, r = random. Results are averaged over 5 seeds.

Dataset	BC	OneStep	TD3BC	CQL	IQL	SVR	DS
halfcheetah-m	42	50.4	48.3	47	47.4	60.5	**63.1**
hopper-m	56.2	87.5	59.3	53	66.2	103.5	**103.7**
walker2d-m	71	84.8	83.7	73.3	78.3	**92.4**	90.0
halfcheetah-m-r	36.4	42.7	44.6	45.5	44.2	52.5	**54.6**
hopper-m-r	21.8	98.5	60.9	88.7	94.7	**103.7**	103.5
walker2d-m-r	24.9	61.7	81.8	81.8	73.8	95.6	**96.2**
halfcheetah-m-e	59.6	75.1	90.7	75.6	86.7	**94.2**	93.7
hopper-m-e	51.7	108.6	98	105.6	91.5	**111.2**	110.1
walker2d-m-e	101.2	111.3	110.1	107.9	109.6	109.3	**109.4**
halfcheetah-e	92.9	88.2	96.7	96.3	95	**96.1**	94.0
hopper-e	110.9	106.9	107.8	96.5	109.4	111.1	111.1
walker2d-e	107.7	110.7	110.2	108.5	109.9	110	**111.9**
halfcheetah-r	2.6	2.3	11	17.5	13.1	**27.2**	26.8
hopper-r	4.1	5.6	8.5	7.9	7.9	**31.0**	**31.0**
walker2d-r	1.2	**6.9**	1.6	5.1	5.4	2.2	2.4
gym-v2 total	784.2	1041.2	1013.2	1010.2	1033.1	1200.5	**1201.6**
pen-expert	85.1	61.6	111	107	110.2	138.9	**140.8**
pen-human	34.4	**73.7**	54.9	37.5	71.5	73.1	58.6
pen-cloned	56.9	31.8	63.8	39.2	37.3	**70.2**	43.1
adroit-v0 total	176.4	167.1	229.7	183.7	219	**282.2**	242.5

Vision D4RL. We also evaluate our algorithm on challenging visual domains using the V-D4RL dataset (Lu et al., 2023), which is a challenging pixel-based analogue of D4RL. We compare it against the results of the model-free algorithms investigated in the original paper. We run our algorithm with the actor regularization term (Eq. (12), $\lambda > 0$). Experimental details are provided in Appendix A.1.

Table 2 provides the full experimental results. Overall, we find that our method is competitive with existing methods, slightly outperforming DRQ+BC and CQL, and only underperforming the BC method. We note that Lu et al. (2023) finds that V-D4RL is challenging for model-free actor critic algorithms, with BC outperforming these methods on most tasks. However, when taking both datasets (standard D4RL and

Table 2: Normalized scores on the V-D4RL benchmarks. The average score from 3 seeds is mapped from [0,1000] to [0,100].

Environment	DrQ+BC	CQL	BC	SVR	DS
walker-walk					
random	5.5	14.4	2.0	2.0	3.4
mixed	28.7	11.4	16.5	22.6	23.7
medium	46.8	14.8	40.9	37.8	42.9
medexp	86.4	56.4	47.7	43.1	42.5
expert	68.4	89.6	91.5	89.6	88.9
cheetah-run					
random	5.8	5.9	0.0	0.1	0.5
mixed	44.8	10.7	25	50.2	19.6
medium	53.0	40.9	51.6	30.5	43.5
medexp	50.6	20.9	57.5	56.2	50.0
expert	34.5	61.5	67.4	29.3	42.6
humanoid-walk					
random	0.1	0.2	0.1	0.1	0.1
mixed	15.9	0.1	18.8	13.9	22.6
medium	6.2	0.1	13.5	4.6	5.4
medexp	7.0	0.1	17.2	9.1	3.4
expert	2.7	1.6	6.1	5.1	6.9
total	300.6	328.6	455.8	394.1	396.0

V-D4RL) into consideration, these findings demonstrate that our method can achieve good performance consistently across both standard offline RL tasks and tasks involving complex visual observations.

5.2 Robustness study

In this section, we conduct an experiment to contrast our method with SVR, which also uses an explicit density estimator $\beta(a \mid s)$ of the behavior policy to compute its regularization term. We investigate how robust the methods are to (mis)-estimation of this behavior policy and their numerical stability in training. For the behavior model, both our DS method and SVR use a parameterized Gaussian model $a \mid s \sim N(\mu(s;\psi), \sigma(s;\psi))$; for the experiments in Table 1, a constant $\sigma(s) = 0.2$ is used. Here, we vary the σ parameter across a range of values and evaluate the algorithms for the different behavior models.

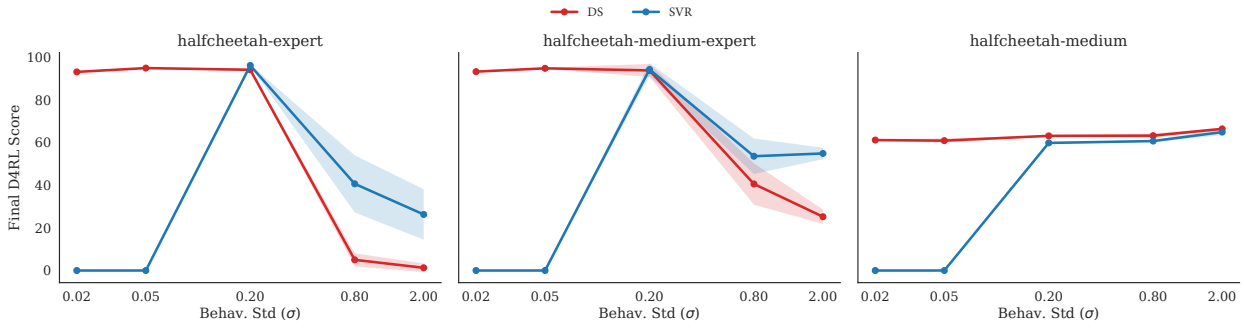

Figure 1: Results of varying the standard deviation parameter in the Gaussian behavior model β. Curves are averaged over 3 independent trials, where shading represents the standard deviation.

The results are displayed in Figure 1. We discover that for small values of $\sigma = \{0.02, 0.05\}$, SVR fails to train as the penalty term diverges, leading to numerical overflow during optimization. In contrast, the DS does not suffer from this issue for all values of σ considered, outperforming or remaining close to SVR on most of the different values.

We also train a Gaussian variance network that learns a variable state-dependent $\sigma(s)$, by minimizing the weighted negative log-likelihood (from Seitzer et al. (2022)):

$$L_{\mu,\sigma}(\psi) = \mathbb{E}_{(s,a)\sim D}\left[\lfloor\sigma(s;\psi)\rfloor\left(\log\sigma^2(s;\psi) + \frac{\|\mu(s;\psi)-a\|^2}{\sigma^2(s;\psi)}\right)\right] \tag{13}$$

where $\lfloor\cdot\rfloor$ denotes the stop-gradient operation. This produces a behavior policy $a \mid s \sim N(\mu(s;\psi), \sigma^2(s;\psi))$ where the variance varies with s, instead of being constant. Again, we train and evaluate the performance of SVR and DS on Gym-Mujoco using this behavior model. We find that SVR fails to train for all environments, due to the same issue of the penalty diverging, whereas DS trains under the varying $\sigma(s)$ successfully. We present the aggregrated results and comparison in Table 3, and provide the full results and training details in Appendix A.3.

Table 3: Total scores on the Gym-Mujoco domain, summed over (random, medium, medium-replay, medium-expert, expert) datasets. (var) refers to using the behavior model with a learnable variance. Null results arise from failure to train the algorithm.

Dataset	DS	DS (var)	SVR (var)
halfcheetah-v2 total	332.3	322.1	Null
hopper-v2 total	459.4	450.4	Null
walker-v2 total	409.9	404.1	Null

These results indicate that the DS penalty is more robust to evaluations of the estimated behavior density compared to SVR. SVR performs well only on a narrow range of σ values, and we find that the loss diverges when the behavior density takes small values in training, leading to early failure in optimization. This is because the support penalty used by SVR (Eq. (6)) contains a term that involves the importance ratio $\frac{\mu(a|s)}{\beta(a|s)}$. This term is unbounded for arbitrary densities and blows up for any action a such that $\beta(a \mid s)$ is close to zero, which can occur in practice when the Gaussian density is narrow. In contrast, the DS penalty uses a coefficient of the form $\left(1 - \frac{\beta(a|s)}{\max_a \beta(a|s)}\right)$, which is bounded in $[0, 1]$ for all a. This results in superior numerical stability and admits greater flexibility when estimating the behavior policy, allowing the use of a learnable variance parameter that matches the performance of the standard tuned model.

5.3 Hyperparameter sensitivity

We evaluate the sensitivity of the DS penalty to the regularization strength α. We focus on datasets of varying difficulty, ranging from medium to expert, and consider a range of values $\alpha \in [10^{-4}, 10^{-3}, ..., 1]$.

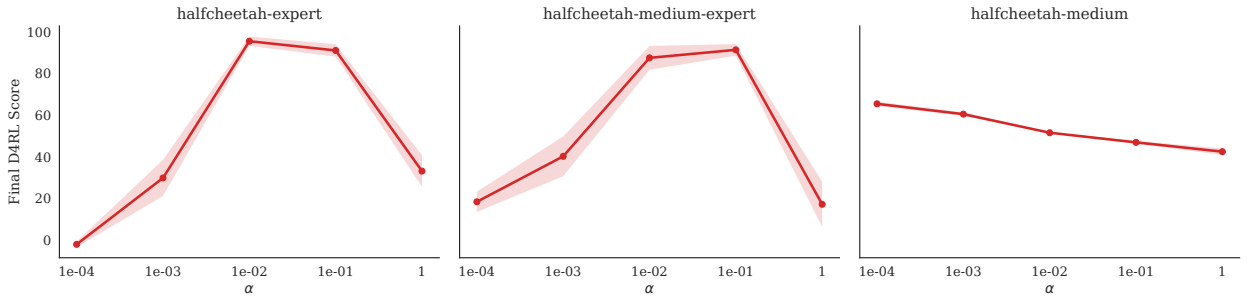

Figure 2: Results of varying the regularization parameter α for the DS penalty. Curves are averaged over 3 independent trials, where shading represents the standard deviation.

Figure 2 displays the performance of our method. We observe that for datasets with expert or mixed expert trajectories, there is a range spanning an order of magnitude of α where the DS penalty performs well. For suboptimal datasets, where coverage is more uniform, the DS penalty and performs well on a wide range of magnitudes. This indicates that the performance and strength of regularization is sensitive to the coverage

and optimality of the dataset, yet there is a reasonable range of values that can be chosen, which can perform well across different tasks of similar dataset type.

6 Conclusion

This work considers value-based offline RL and proposes a new regularization method, the Density-Scaled penalty, that reduces out-of-distribution errors by penalizing the policy evaluation step. The penalty is simple and practical to implement for value-based RL methods. We theoretically connect it to existing state-of-the-art work, SVR, and demonstrate that it is competitive on the D4RL dataset while being more robust and numerically stable.

We have demonstrated that the DS penalty provides a promising alternative for performing regularization on offline Q-learning algorithms. Like SVR, our method requires explicit density estimation of the behavior policy, and future work can investigate more refined models that could capture the dataset distribution more accurately or methods to compute the loss without explicitly constructing this estimator. In addition, our general idea of penalizing the Q-values in proportion to the behavior density could be extended to consider other forms beyond the quadratic, or having a state-action dependent minimum value in the term. This could be useful for improving the performance and increasing the flexibility of the method.

References

Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning, 2020. URL https://arxiv.org/abs/1907.04543.

Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble, 2021. URL https://arxiv.org/abs/2110.01548.

David Brandfonbrener, William F. Whitney, Rajesh Ranganath, and Joan Bruna. Offline rl without off-policy evaluation, 2021. URL https://arxiv.org/abs/2106.08909.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2021. URL https://arxiv.org/abs/2004.07219.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning, 2021. URL https://arxiv.org/abs/2106.06860.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods, 2018. URL https://arxiv.org/abs/1802.09477.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration, 2019. URL https://arxiv.org/abs/1812.02900.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. URL https://arxiv.org/abs/1801.01290.

Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog, 2019. URL https://arxiv.org/abs/1907.00456.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning, 2021a. URL https://arxiv.org/abs/2110.06169.

Ilya Kostrikov, Jonathan Tompson, Rob Fergus, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization, 2021b. URL https://arxiv.org/abs/2103.08050.

Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction, 2019. URL https://arxiv.org/abs/1906.00949.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning, 2020. URL https://arxiv.org/abs/2006.04779.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020. URL https://arxiv.org/abs/2005.01643.

Cong Lu, Philip J. Ball, Tim G. J. Rudner, Jack Parker-Holder, Michael A. Osborne, and Yee Whye Teh. Challenges and opportunities in offline reinforcement learning from visual observations, 2023. URL https://arxiv.org/abs/2206.04779.

Yixiu Mao, Hongchang Zhang, Chen Chen, Yi Xu, and Xiangyang Ji. Supported value regularization for offline reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=fze7P9oy6l.

Yixiu Mao, Qi Wang, Yun Qu, Yuhang Jiang, and Xiangyang Ji. Doubly mild generalization for offline reinforcement learning, 2024. URL https://arxiv.org/abs/2411.07934.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602.

Brendan O'Donoghue, Ian Osband, Remi Munos, and Volodymyr Mnih. The uncertainty bellman equation and exploration, 2018. URL https://arxiv.org/abs/1709.05380.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning, 2025. URL https://arxiv.org/abs/2502.02538.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning, 2019. URL https://arxiv.org/abs/1910.00177.

Maximilian Seitzer, Arash Tavakoli, Dimitrije Antic, and Georg Martius. On the pitfalls of heteroscedastic uncertainty estimation with probabilistic neural networks, 2022. URL https://arxiv.org/abs/2203.09168.

Yutaka Shimizu, Joey Hong, Sergey Levine, and Masayoshi Tomizuka. Strategically conservative q-learning, 2024. URL https://arxiv.org/abs/2406.04534.

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning, 2023. URL https://arxiv.org/abs/2305.09836.

Jialong Wu, Haixu Wu, Zihan Qiu, Jianmin Wang, and Mingsheng Long. Supported policy optimization for offline reinforcement learning, 2022. URL https://arxiv.org/abs/2202.06239.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning, 2019. URL https://arxiv.org/abs/1911.11361.

Yue Wu, Shuangfei Zhai, Nitish Srivastava, Joshua Susskind, Jian Zhang, Ruslan Salakhutdinov, and Hanlin Goh. Uncertainty weighted actor-critic for offline reinforcement learning, 2021. URL https://arxiv.org/abs/2105.08140.

Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Wai Kin Chan, and Xianyuan Zhan. Offline rl with no ood actions: In-sample learning via implicit value regularization, 2023. URL https://arxiv.org/abs/2303.15810.

A Appendix

A.1 Experimental details

In this section, we provide further details on the experiments conducted.

D4RL. Table 4 provides the hyperparameter choices for our DS method.

Table 4: Hyperparameters for implementation of DS algorithm for the D4RL datasets.

Hyperparameter	Value
Actor hidden dimension	[256, 256]
Critic hidden dimension	[256, 256]
Number of critics	4
Actor learning rate	3e-4
Critic learning rate	3e-4
Optimizer	Adam
Critic penalty coefficient α	{9e-4, 4e-2} (Gym-MuJoCo), 1 (Adroit)
Target update rate τ	0.005
Batch size	256
Discount factor γ	0.99
Number of iterations	10^6
Policy update frequency	2

For Gym-Mujoco datasets, we use a penalty coefficient $\alpha = 9\times10^{-4}$ for random, medium, and medium-replay environments, and $\alpha = 4 \times 10^{-2}$ for medium-expert and expert environments.

The behavior model is a network with the same architecture as the actor. It is trained with Adam (lr=1e-3) for 10^5 iterations. For the Gaussian model we used a fixed standard deviation of $\sigma = 0.2$.

V-D4RL. Our DS method uses the same critic and actor network and the same training hyperparameters as the CQL agent in the V-D4RL paper (Lu et al., 2023).

For our additional hyperparameters, we set $\alpha = 0.1$ for all walker-walk and humanoid-walk environments, and $\alpha = 1$ for cheetah-run. We set $\lambda = 1$ for all environments.

For the behavior network, we use same architecture and training procedure as the BC agent in the V-D4RL paper. For the Gaussian model we used a fixed standard deviation of $\sigma = 0.1$.

A.2 Proofs

In this section, we provide proofs of the main results in Section 4. We will at times use $\mathbb{E}_s$ as a shorthand for $\mathbb{E}_{s\sim D}$, $\mathbb{E}_{s'}$ for $\mathbb{E}_{s'\sim P(\cdot|s,a)}$, and $\mathbb{E}_{a\sim\beta}$ for $\mathbb{E}_{a\sim\beta(\cdot|s)}$ when the context is clear. First, we restate and provide expanded proofs of the update solutions to CQL, SVR, and the DS objective; the full proofs of the CQL and SVR solutions were not provided in the original papers.
We use the following fact about the derivative of the Bellman error. Given dataset $D = \{(s, a, s', r)\}$, where $a \sim \beta(\cdot \mid s)$ for each s, the mean squared Bellman error is given by

$$B(Q) = \frac{1}{2}\mathbb{E}_{s\sim D, a\sim\beta(\cdot|s)}[(Q(s,a) - T^\pi\hat{Q}(s,a))^2].$$

Note that here, we write Q' (a copy of Q) in the target $T^\pi\hat{Q}(s,a)$ since we are treating it as a fixed constant in optimization. The derivative of the error with respect to the current value $Q(s,a)$ is:

$$\frac{dB(Q)}{dQ(s,a)} = d_\beta(s)\beta(a \mid s)\left[Q(s,a) - T^\pi\hat{Q}(s,a)\right],$$

where $d_\beta(s)$ is the state-visitation distribution of $\beta(a \mid s)$. We assume throughout that $d_\beta(s) > 0$ for all s.

Proposition 3 (CQL solution)**.** *For any two policies $\beta(a \mid s)$ and $\pi(a \mid s)$, the solution to*

$$\min_Q B(Q) + \alpha(\mathbb{E}_{s\sim D}[\mathbb{E}_{a\sim\pi(\cdot|s)}[Q(s,a)] - \mathbb{E}_{a\sim\beta(\cdot|s)}[Q(s,a)]])$$

is given by

$$Q_{\mathrm{CQL}}(s,a) = \begin{cases} T^\pi Q(s,a) - \alpha\left(\frac{\pi(a|s)}{\beta(a|s)} - 1\right) & \beta(a \mid s) > 0 \\ -\infty & \beta(a \mid s) = 0, \pi(a \mid s) > 0 \\ Q(s,a) & \beta(a \mid s) = \pi(a \mid s) = 0 \end{cases}$$

where we (arbitrarily) set the solution in the case $\beta(a \mid s) = \pi(a \mid s) = 0$ to be $Q(s,a)$, i.e. no update performed.

Proof. The CQL update is

$$\min_Q B(Q) + \alpha(\mathbb{E}_{s\sim D, a\sim\pi}[Q(s,a)] - \mathbb{E}_{s\sim D, a\sim\beta}[Q(s,a)])$$

and is convex in Q. For fixed (s,a), consider the following cases:

Case 1: $\beta(a \mid s) > 0$. Setting the derivative with respect to $Q(s,a)$ to zero, and dropping the $d_\beta(s)$ terms, we have

$$\beta(a \mid s)\left[Q(s,a) - T^\pi Q(s,a)\right] + \alpha\big(\pi(a \mid s) - \beta(a \mid s)\big) = 0.$$

Solving for Q gives

$$\begin{aligned} Q^*(s,a) &= T^\pi Q(s,a) - \alpha\frac{\pi(a \mid s) - \beta(a \mid s)}{\beta(a \mid s)} \\ &= T^\pi Q(s,a) - \alpha\left(\frac{\pi(a \mid s)}{\beta(a \mid s)} - 1\right). \end{aligned}$$

Case 2: $\beta(a \mid s) = 0$ and $\pi(a \mid s) > 0$. In this case, the terms involving expectations in β vanish, and the objective reduces to

$$\min_Q \alpha\mathbb{E}_{s\sim D, a\sim\pi}[Q(s,a)]$$

Since $\alpha > 0$ and $\pi(a \mid s) > 0$, the objective is minimized at $Q^*(s,a) = -\infty$.

Case 3: $\beta(a \mid s) = 0$ and $\pi(a \mid s) = 0$. Both the Bellman error term and the regularization term vanish. In this case, the solution is degenerate so we say $Q(s,a)$ is left unchanged.

Combining all cases, the CQL minimizer is given by

$$Q_{\mathrm{CQL}} = \begin{cases} T^\pi Q(s,a) - \alpha\left(\frac{\pi(a|s)}{\beta(a|s)} - 1\right) & \beta(a \mid s) > 0 \\ -\infty & \beta(a \mid s) = 0, \pi(a \mid s) > 0 \\ Q(s,a) & \beta(a \mid s), \pi(a \mid s) = 0 \end{cases}$$

□

Proposition 4 (SVR update)**.** *For any policies $\beta(a \mid s)$, $\mu(a \mid s)$, and $\pi(a \mid s)$, the solution to*

$$\min_Q B(Q) + \alpha\mathbb{E}_{s\sim D}\left(\mathbb{E}_{a\sim\mu(\cdot|s)}(Q(s,a) - Q_{\min})^2 - \mathbb{E}_{a\sim\beta(\cdot|s)}\frac{\mu(a \mid s)}{\beta(a \mid s)}(Q(s,a) - Q_{\min})^2\right)$$

is given by

$$Q_{\mathrm{SVR}}(s,a) = \begin{cases} T^\pi Q(s,a) & \beta(a \mid s) > 0 \\ Q_{\min} & \beta(a \mid s) = 0 \end{cases}$$

Proof. The objective is

$$\min_Q B(Q) + \alpha \left(\mathbb{E}_{s\sim D, a\sim\mu}(Q(s,a) - Q_{\min})^2 - \mathbb{E}_{s\sim D, a\sim\beta} \frac{\mu(a \mid s)}{\beta(a \mid s)} (Q(s,a) - Q_{\min})^2 \right)$$

which is convex in Q.

For a fixed (s, a), consider the first-order optimality conditions in the following cases.

Case 1: $\beta(a \mid s) > 0$.

Note that the derivative of the regularization term vanishes since $\beta(a \mid s) > 0$ and we have

$$\mu(a \mid s)(Q(s,a) - Q_{\min}) - \beta(a \mid s) \frac{\mu(a \mid s)}{\beta(a \mid s)} (Q(s,a) - Q_{\min}) = 0.$$

Thus the optimality condition is the same as the unregularized Bellman objective $\min_Q L(Q) = B(Q)$, the solution of which is $Q^*(s,a) = T^\pi Q(s,a)$.

Case 2: $\beta(a \mid s) = 0$. In this case, the second expectation term has no contribution at (s, a), since a is never sampled under β. The objective becomes

$$\min_Q B(Q) + \alpha \mathbb{E}_{a\sim\mu}(Q(s,a) - Q_{\min})^2.$$

Taking the derivative with respect to $Q(s, a)$ and setting it to zero yields

$$\frac{dB(Q)}{dQ(s,a)} + 2\alpha\mu(a \mid s)(Q(s,a) - Q_{\min}) = 0.$$

Note that for $\beta(a \mid s) = 0$, $B(Q)$ does not depend on $Q(s, a)$ because the expectation is over in-distribution actions where $\beta(a \mid s) > 0$, so $\frac{dB(Q)}{dQ(s,a)} = 0$ and

$$2\alpha\mu(a \mid s)(Q(s,a) - Q_{\min}) = 0,$$

which implies $Q^\star(s,a) = Q_{\min}$. Combining the two cases, we have

$$Q_{\text{SVR}} = \begin{cases} T^\pi Q(s,a) & \beta(a \mid s) > 0 \\ Q_{\min} & \beta(a \mid s) = 0 \end{cases}$$

□

Theorem 1 (DS update)**.** *Given a policy $\pi(a \mid s)$, behavior policy $\beta(a \mid s)$, and regularization parameter α, let $C_s = \max_a \beta(a \mid s)$ for each s, and $k_{s,a} = \left(\frac{\pi(a|s)}{\beta(a|s)} - \frac{\pi(a|s)}{C_s} \right)$. For $a \in A$ and $s \in D$, the solution to*

$$\arg\min_Q B(Q) + \frac{\alpha}{2} \mathbb{E}_{s\sim D, a\sim\pi(\cdot|s)} \left[\left(1 - \frac{\beta(a \mid s)}{C_s} \right) (Q(s,a) - Q_{\min})^2 \right] \tag{8}$$

is given by

$$Q_{\text{DS}}(s,a) = \begin{cases} T^\pi Q(s,a) & \beta(a \mid s) = C_s \\ \frac{1}{1+\alpha k_{s,a}} T^\pi Q(s,a) + \frac{\alpha k_{s,a}}{1+\alpha k_{s,a}} Q_{\min} & 0 < \beta(a \mid s) < C_s \\ Q_{\min} & \beta(a \mid s) = 0 \end{cases} \tag{9}$$

Proof. For fixed (s, a), consider the following cases. We assume that $\pi(a \mid s) > 0$ throughout.

Case 1: $\beta(a \mid s) = C_s$. In this case, we have $(1 - \frac{\beta(a|s)}{C_s}) = 0$, so the regularization term vanishes, and the solution is the same as the unregularized objective: $Q^*(s,a) = T^\pi Q(s,a)$.

Case 2: $0 < \beta(a \mid s) < C_s$. Setting the derivative of the objective with respect to $Q(s,a)$ to zero, we have

$$\begin{aligned}
\frac{dB(Q)}{dQ(s,a)} + d_\beta(s)\alpha[\pi(a \mid s)\left(1 - \frac{\beta(a \mid s)}{C_s}\right)(Q(s,a) - Q_{\min})] &= 0 \\
\beta(s,a)[Q(s,a) - T^\pi Q(s,a)] + \alpha\,\pi(a \mid s)\left(1 - \frac{\beta(a \mid s)}{C_s}\right)(Q(s,a) - Q_{\min}) &= 0 \\
Q(s,a) - T^\pi Q(s,a) + \alpha\left(\frac{\pi(a \mid s)}{\beta(a \mid s)} - \frac{\pi(a \mid s)}{C_s}\right)(Q(s,a) - Q_{\min}) &= 0
\end{aligned}$$

Let $k_{s,a} = \frac{\pi(a|s)}{\beta(a|s)} - \frac{\pi(a|s)}{C_s}$. Then the above simplifies to

$$\begin{aligned}
(1 + \alpha k_{s,a})Q(s,a) &= \alpha k_{s,a} Q_{\min} + T^\pi Q(s,a) \\
Q(s,a) &= \frac{1}{1 + \alpha k_{s,a}} T^\pi Q(s,a) + \frac{\alpha k_{s,a}}{1 + \alpha k_{s,a}} Q_{\min}\,.
\end{aligned}$$

Thus $Q^*(s,a) = \frac{1}{1+\alpha k} T^\pi Q(s,a) + \frac{\alpha k_{s,a}}{1+\alpha k_{s,a}} Q_{\min}\,.$

Case 3: $\beta(a \mid s) = 0$. In this case, the terms in the objective involving expectations over $a \sim \beta$ vanish. The objective becomes

$$\min_Q \alpha \mathbb{E}_{s\sim D, a\sim\pi}(Q(s,a) - Q_{\min})^2$$

which has the solution $Q^*(s,a) = Q_{\min}$.

Combining the cases, we obtain the update solution for the DS objective:

$$Q_{\mathrm{DS}} = \begin{cases} T^\pi Q(s,a) & \beta(a \mid s) = C_s \\ \frac{1}{1+\alpha k_{s,a}} T^\pi Q(s,a) + \frac{\alpha k_{s,a}}{1+\alpha k_{s,a}} Q_{\min} & 0 < \beta(a \mid s) < C_s \\ Q_{\min}(s,a) & \beta(a \mid s) = 0 \end{cases}$$

□

Now, we prove the main properties of the DS operator. For simplicity, we rewrite the operator in a simpler way. Let $k_{s,a} = \left(\frac{\pi(a|s)}{\beta(a|s)} - \frac{\pi(a|s)}{C_s}\right)$ and $C_s = \max_a \beta(a \mid s)$. For simplicty, we can set $\delta = \frac{1}{1+\alpha k_{s,a}}$, leaving the dependence on s, a implicit. We thus have $1 - \delta = \frac{\alpha k_{s,a}}{1+\alpha k_{s,a}}$, and $0 < \delta < 1$. Then the operator $T^\pi_{\mathrm{DS}} Q(s,a)$ can be rewritten as

$$T^\pi_{\mathrm{DS}} Q(s,a) = \begin{cases} T^\pi Q(s,a) & \beta(a \mid s) = C_s \\ \delta T^\pi Q(s,a) + (1-\delta) Q_{\min} & 0 < \beta(a \mid s) < C_s \\ Q_{\min} & \beta(a \mid s) = 0 \end{cases}$$

Proposition 1 (Contraction)**.** *The operator $T^\pi_{\mathrm{DS}} Q(s,a)$ is a γ-contraction operator in the L_∞ norm.*

Proof. Let f_1 and f_2 be two arbitrary functions on $S \times A$. We consider the three cases induced by $\beta(a \mid s)$.

Case 1: $\beta(a \mid s) = 0$. In this case,

$$|T^\pi_{\mathrm{DS}} f_1(s,a) - T^\pi_{\mathrm{DS}} f_2(s,a)| = |\, Q_{\min} - Q_{\min}\,| = 0 \le \gamma\ \ |\, f_1 - f_2\ \ |_\infty\,.$$

Case 2: $0 < \beta(a \mid s) < C_s$. We have

$$\begin{aligned}
|T^\pi_{\mathrm{DS}} f_1(s,a) - T^\pi_{\mathrm{DS}} f_2(s,a)| &= |\delta(T^\pi f_1(s,a) - T^\pi f_2(s,a))| \\
&= \delta\gamma\left|\mathbb{E}_{s'\sim P(\cdot|s,a), a'\sim\pi(\cdot|s)}\left[f_1(s',a') - f_2(s',a')\right]\right| \\
&\le \gamma \mathbb{E}_{s'\sim P(\cdot|s,a), a'\sim\pi(\cdot|s)}\left[|f_1(s',a') - f_2(s',a')|\,,\right] \\
&\le \gamma\ \ |\, f_1 - f_2\ \ |_\infty\,.
\end{aligned}$$

Case 3: $\beta(a \mid s) = 0$. In this case, T^π_{DS} coincides with the standard Bellman operator and we have

$$\begin{aligned}|T^\pi_{\mathrm{DS}} f_1(s,a) - T^\pi_{\mathrm{DS}} f_2(s,a)| &= |T^\pi f_1(s,a) - T^\pi f_2(s,a)| \\ &\leq \gamma \|f_1 - f_2\|_\infty.\end{aligned}$$

□

Proposition 2 (Fixed point)**.** *For any π, the fixed point of T^π_{DS}, denoted by f, exists and satisfies*

$$\begin{cases} Q_{\min} \leq f(s,a) \leq Q^\pi(s,a) & \beta(a \mid s) > 0 \\ f(s,a) = Q_{\min} & \beta(a \mid s) = 0 \end{cases}$$

Proof. Our proof is partially adapted from Mao et al. (2023). By Proposition 1 and the contraction mapping theorem there exists a unique fixed point, which we denote f. The fixed point satisfies $f(s,a) = T^\pi_{\mathrm{DS}} f(s,a)$. The equality $f(s,a) = Q_{\min}$ where $\beta(a \mid s) = 0$ follows directly from the definition of T^π_{DS}.

For $\beta(a \mid s) > 0$, we can combine the first two cases of T^π_{DS} and write $T^\pi_{\mathrm{DS}} f(s,a) = \delta T^\pi f(s,a) + (1-\delta) Q_{\min}$, where $0 < \delta \leq 1$. Note that taking $\delta = 1$ recovers the first case where $\beta(a \mid s) = C_s$.

We first prove the lower bound $f(s,a) \geq Q_{\min}$ for all (s,a).

Fix (s,a) with $\beta(s,a) > 0$. We have

$$\begin{aligned} f(s,a) &= T^\pi_{\mathrm{DS}} f(s,a) \\ &= \delta T^\pi f(s,a) + (1-\delta) Q_{\min} \\ &= \delta[r(s,a) + \gamma \mathbb{E}_{s'} \mathbb{E}_{a' \sim \pi(\cdot \mid s')} f(s',a')] + (1-\delta) Q_{\min} \\ &= \delta[r(s,a) + \gamma \mathbb{E}_{s'} \mathbb{E}_{a' \sim \pi(\cdot \mid s')} T^\pi f(s',a')] + (1-\delta) Q_{\min} \end{aligned}$$

We now analyze the term $\mathbb{E}_{s'} \mathbb{E}_{a' \sim \pi(\cdot \mid s')} T^\pi f(s',a')$. Denote $I(s) = \{a \in \mathcal{A} \mid \beta(a \mid s) > 0\}$, $J(s) = \{a \in \mathcal{A} \mid \beta(a \mid s) = 0\}$. The expectation over actions may be expanded as follows:

$$\begin{aligned} \mathbb{E}_{a' \sim \pi(\cdot \mid s')} T^\pi f(s',a') &= \sum_{a' \in I(s')} \pi(a' \mid s') T^\pi f(s',a') + \sum_{a' \in J(s')} \pi(a' \mid s') T^\pi f(s',a') \\ &= \sum_{a' \in I(s')} \pi(a' \mid s') f(s',a') + \sum_{a' \in J(s')} \pi(a' \mid s') Q_{\min} \\ &\geq \sum_{a' \in I(s')} \pi(a' \mid s') f_{\min}(s,a) + \sum_{a' \in J(s')} \pi(a' \mid s') Q_{\min} \end{aligned}$$

where $f_{\min} := \min_{s \in S, a \in I(s)} f(s,a)$. Combining this result with the previous result, we have

$$\begin{aligned} f(s,a) &= \delta[r(s,a) + \gamma \mathbb{E}_{s'} \mathbb{E}_{a' \sim \pi(\cdot \mid s')} T^\pi f(s',a')] + (1-\delta) Q_{\min} \\ &\geq \delta \left[r(s,a) + \gamma \mathbb{E}_{s'} \left[\sum_{a' \in I(s')} \pi(a' \mid s') f_{\min}(s,a) + \sum_{a' \in J(s')} \pi(a' \mid s') Q_{\min} \right] \right] + (1-\delta) Q_{\min} \\ &\geq \delta[r_{\min}(s,a) + \gamma \lambda f_{\min}(s,a) + \gamma(1-\lambda) Q_{\min}]] + (1-\delta) Q_{\min} \end{aligned}$$

where in the last line we have set $\lambda := \mathbb{E}_{s'} \left[\sum_{a' \in I(s')} \pi(a' \mid s') \right]$ and used the fact that

$$1 = \mathbb{E}_{s'} \sum_{a' \in I(s') \cup J(s')} \pi(a' \mid s') = \mathbb{E}_{s'} \sum_{a' \in I(s')} \pi(a' \mid s') + \sum_{a' \in J(s')} \pi(a' \mid s') = \lambda + (1-\lambda).$$

Note that the relation $f(s,a) \geq \delta[r_{\min}(s,a) + \gamma \lambda f_{\min}(s,a) + (1-\lambda) Q_{\min}]] + (1-\delta) Q_{\min}$ holds for all (s,a) such that $\beta(a \mid s) > 0$ and therefore also for the case where the LHS attains the minimum $f_{\min}$. Thus we

have

$$\begin{aligned} f_{\min}(s,a) &\geq \delta[r_{\min}(s,a) + \gamma\lambda f_{\min}(s,a) + (1-\lambda)Q_{\min}]] + (1-\delta)Q_{\min} \\ &= \frac{1}{1-\delta\gamma\lambda}[\delta(1-\gamma)Q_{\min} + \delta\gamma(1-\lambda)Q_{\min} + (1-\delta)Q_{\min}] \\ &= Q_{\min} \end{aligned}$$

where we have used the fact that $Q_{\min} = r_{\min}/(1-\gamma)$.

Therefore, we have $f(s,a) \geq f_{\min}(s,a) \geq Q_{\min}$ for all (s,a) such that $\beta(a \mid s) > 0$.

Now, we prove the upper bound $f(s,a) \leq Q^\pi(s,a)$. To this end, we first show that $T^\pi_{\mathrm{DS}} f(s,a) \leq T^\pi f(s,a)$. Since $T^\pi_{\mathrm{DS}} f(s,a) = \delta T^\pi f(s,a) + (1-\delta)Q_{\min}$ and $\delta \in [0,1]$ it suffices to show that $T^\pi f(s,a) \geq Q_{\min}$. From the definition of T, we have for any (s,a),

$$\begin{aligned} T^\pi f(s,a) &= r(s,a) + \gamma \mathbb{E}_{s'} \mathbb{E}_{a' \sim \pi(\cdot \mid s')} f(s',a') \\ &\geq r_{\min} + \gamma \mathbb{E}_{s'} \mathbb{E}_{a' \sim \pi(\cdot \mid s')} Q_{\min} \\ &= (1-\gamma)Q_{\min} + \gamma Q_{\min} \\ &= Q_{\min} \end{aligned}$$

we we have used the result that $f(s,a) \geq Q_{\min}$ for all (s,a) in the second inequality. Thus, we have $T^\pi_{\mathrm{DS}} f(s,a) \leq T^\pi f(s,a)$ for all (s,a). Then, by induction,

$$f(s,a) = T^\pi_{\mathrm{DS}} f(s,a) \leq T^\pi f(s,a) = T^\pi(T^\pi_{\mathrm{DS}} f(s,a)) \leq T^\pi(T^\pi f(s,a)) \ldots \leq (T^\pi)^n f(s,a).$$

Letting $n \to \infty$, and using the fact that Q^π is the fixed point of T^π, we have $f(s,a) \leq Q^\pi(s,a)$ for all (s,a).

□

A.3 Additional results

In this section, we provide the full results (Table 5) of the evaluation of the variable behavior model, as described in the main text. The behavior model has the same hidden dimension as the actor, with an extra final layer of size [256,1] that outputs the value Softplus$[\sigma^2_\psi(s)]$. It is optimized with Adam (lr=3e-4) for 1e5 iterations.

Table 5: Full results of learnable $\sigma(s)$ behavior model, denoted 'DS (var)', and comparison to fixed $\sigma = 0.2$ for the DS penalty. SVR results not shown, due to failure to train.

Dataset	DS	DS (var)
halfcheetah-medium-expert-v2	93.7	96.0
halfcheetah-expert-v2	94.0	84.2
hopper-medium-expert-v2	110.1	110.2
hopper-expert-v2	111.1	112.1
walker2d-medium-expert-v2	109.4	110.4
walker2d-expert-v2	111.9	111.4
halfcheetah-medium-v2	63.1	59.9
halfcheetah-medium-replay-v2	54.6	53.7
hopper-medium-v2	103.7	97.0
hopper-medium-replay-v2	103.5	99.9
walker2d-medium-v2	90.0	82.6
walker2d-medium-replay-v2	96.2	96.5
halfcheetah-random-v2	26.8	28.3
hopper-random-v2	31.0	31.1
walker2d-random-v2	2.4	3.2
gym-v2 total	1201.6	1176.6