Conservative Value Priors: A Bayesian Path to Offline Reinforcement Learning

Filippo Valdettaro, Yingzhen Li, A. Aldo Faisal

Keywords: Offline reinforcement learning, Bayesian inference.

Summary

Offline reinforcement learning (RL) seeks to improve a policy using only a fixed dataset of past interactions, without further environment exploration. To avoid overly optimistic decisions on uncertain or out-of-distribution actions, the learned policy should be supported by the data. Existing methods typically enforce this via heuristic modifications to objectives or value functions that encourage more conservative action selection. In contrast, we propose a principled alternative by introducing a conservative value prior, thereby modelling the belief that policies are expected to perform poorly unless the behavioural data provides evidence to the contrary. This yields a posterior that assigns high value only to supported actions, guiding the agent toward policies grounded in the data. Our approach thereby unifies Bayesian decision-making, uncertainty quantification and value regularisation while effectively mitigating distributional shift in offline RL. We develop this framework in a model-free setting with theoretical analysis on deterministic environments. We then present an exact inference algorithm for small-scale problems and finally extend it to a scalable deep learning variant compatible with standard off-policy algorithms. Our method achieves strong performance on benchmark locomotion tasks, outperforming comparable model-free baselines.

Contribution(s)

1. We formalises the conservative value prior (CVP) framework for offline reinforcement learning and establish theoretical guarantees on its behaviour in deterministic settings. Compared to mainline offline RL literature, this works takes an inference-based perspective to the conservatism necessary for offline RL, with the required conservatism dropping out naturally from expected posterior value maximisation.
   Context: Builds on and extends a previous proof-of-concept formulation (Valdettaro & Faisal, 2024).
2. We provide an exact-inference implementation of CVPs with Bayesian uncertainty quantification suitable for small datasets with Gaussian processes. For simple, interpretable tasks we observe qualitatively that the learned posterior value estimates and uncertainties behave consistently and incorporate the necessary conservatism as required.
   Context: Previous uncertainty-based offline RL methods typically use the uncertainty for action-selection, whereas in our framework decision-making is based on the posterior mean and does not rely explicitly on uncertainty.
3. We develop a scalable deep offline RL implementation of CVPs that reduces to adding a simple, easy-to-compute term to the critic loss of standard off-policy algorithms. We instantiate this with a Soft Actor-Critic (SAC)-based variant, CVP-SAC, and demonstrate strong performance on standard locomotion benchmarks compared to similar methods.
   Context: To isolate the effect of our Bayesian-derived value regularisation, in the main text we compare against methods that use a standard off-policy backbone with slight modifications to the losses, without ensembles or generative models for the data-generating process. In the appendix, we also compare against higher-performing algorithms that include additional engineering components and observe competitive performance. We do not compare to algorithms with substantially greater computational overhead, such as those relying on ensembles.

Conservative Value Priors: A Bayesian Path to Offline Reinforcement Learning

Filippo Valdettaro[1], **Yingzhen Li**[1,2], **A. Aldo Faisal**[1,3,4]
{filippo.valdettaro20, yingzhen.li, a.faisal}@imperial.ac.uk

[1]**Department of Computing, Imperial College London**
[2]**College of Computing & Data Science, Nanyang Technological University**
[3]**Department of Bioengineering, Imperial College London**
[4]**Chair in Digital Health & Data Science, University of Bayreuth**

Abstract

Offline reinforcement learning (RL) seeks to improve a policy using only a fixed dataset of past interactions, without further environment exploration. To avoid overly optimistic decisions on uncertain or out-of-distribution actions, the learned policy should be supported by the data. Existing methods typically enforce this via heuristic modifications to objectives or value functions that encourage more conservative action selection. In contrast, we propose a principled alternative by introducing a conservative value prior, thereby modelling the belief that policies are expected to perform poorly unless the behavioural data provides evidence to the contrary. This yields a posterior that assigns high value only to supported actions, guiding the agent toward policies grounded in the data. Our approach thereby unifies Bayesian decision-making, uncertainty quantification and value regularisation while effectively mitigating distributional shift in offline RL. We develop this framework in a model-free setting with theoretical analysis on deterministic environments. We then present an exact inference algorithm for small-scale problems and finally extend it to a scalable deep learning variant compatible with standard off-policy algorithms. Our method achieves strong performance on benchmark locomotion tasks, outperforming comparable model-free baselines.

1 Introduction

Offline reinforcement learning (RL) can harness datasets of suboptimal demonstrations to learn effective policies without direct interaction or additional exploration on an environment. This extends the potential for the application of RL in domains where simulation is not realistically possible, where a large amount of operational data have already been collected and where direct suboptimal interaction with the underlying environment can be dangerous or costly, such as healthcare (Gottesman et al., 2019; Komorowski et al., 2018), robotics (Sinha et al., 2022; Kalashnikov et al., 2018; Dasari et al., 2020; Kendall et al., 2019) or recommender systems (Huang et al., 2022; Xiao & Wang, 2021). However, a lack of interaction with the environment comes with additional challenges. In particular, traditional off-policy actor-critic algorithms can perform very poorly when naively applied to an offline dataset: without appropriate regularisation, uncontrolled extrapolation can cause the critic to assign high values to actions that it has never encountered in the dataset, leading to the actor selecting policies that are entirely unsupported by the data.

A flexible strategy to mitigate this issue is to train a critic that learns a *conservative* or *pessimistic* value function by modifying the value function estimate: one approach is to incorporate a penalty on state-action pairs with high uncertainty into the value estimate, often measured by the disagreement among an ensemble of critics (Ghasemipour et al., 2022; An et al., 2021). While ensemble methods

can be effective, they tend to be computationally intensive, and the extent of the uncertainty penalty may need to be determined in an *ad hoc* manner (Buckman et al., 2020). Other approaches have brought about methods that bypass explicit uncertainty quantification by training the critic to estimate a lower bound on the value function based solely on the observed data, aiming to provide robustness against worst-case outcomes (Kumar et al., 2020; Cheng et al., 2022). However, such worst-case formulations may be excessively pessimistic, potentially hindering overall performance (Xu & Mannor, 2006).

Here, we explore an alternative approach to learning a conservative value function estimation for offline RL rooted in Bayesian inference. We build on the proof-of-concept work in Valdettaro & Faisal (2024), extend it to continuous control, and provide a scalable deep learning variant of the method. Our approach is inspired by Bayesian decision theory: the actor is trained to optimise a posterior expectation of utility (value), which is a provably optimal and principled decision criterion (Robert et al., 2007), and is in general preferable to worst-case robustness as the latter can significantly harm average performance (Xu & Mannor, 2006). Choosing a prior belief over value functions that is conservative will naturally induce conservatism in the posterior value function estimate of actions outside the dataset's support. This can be interpreted as a less extreme form regularisation than in algorithms that learn worst-case lower bounds of value functions. Thus, we train a critic with a conservative value prior (CVP), so that after inference only those state-actions that are supported by the data will be assigned high values. We achieve this in an off-policy model-free way in deterministic environments by structuring inference so that it respects the temporal-difference (TD) structure of a Markovian environment to get consistent value functions after conditioning on the observed data. As a natural by-product of our inference scheme, we obtain posterior epistemic uncertainty estimates, although we keep these decoupled from decision-making.

Our main contributions are threefold and summarised as follows: 1. We formalise the CVP formulation with theoretical analysis for in deterministic environments (Section 4). 2. We provide a way to implement the framework with exact inference and continuous control and showcase the qualitative results and uncertainty visualisations on a a classical control environment in the low-data regime (Section 5). 3. We then propose a scalable, deep learning-based variant of CVP for offline RL, achieved by adding an easily-computable term to the critic's loss, which constitutes a simple modification applicable to any off-policy algorithm and evaluate our method on the D4RL (Fu et al., 2020) robotics locomotion tasks, on which we find it performs better than previous comparable methods and recovers similar performance to more algorithmically complex methods (Section 6.2).

2 Related Work

In the following, we introduce and motivate the elements of conservative value estimation, existing approaches, and model-free inference in RL. Additionally, we cover function-space inference, which enables Bayesian reasoning over function outputs rather than parameters, which is necessary to introduce a conservative prior over value.

Conservative value estimation. A core challenge in offline RL is avoiding distributional shift between the learned and behaviour policies. One approach adds constraints to keep the actor close to the behaviour policy, either explicitly (Fujimoto & Gu, 2021) or implicitly (Nair et al., 2020), but this is sensitive to suboptimal data. Instead, faithful generative models of the behaviour policy can constrain actions to merely by in the dataset's support (Fujimoto et al., 2019; Zhou et al., 2021), though they rely on accurately modelling the behaviour policy, which is itself challenging. Unlike behaviour regularisation, conservative value estimation methods are less sensitive to poor-quality data. Uncertainty-based approaches (An et al., 2021; Ghasemipour et al., 2022; Yang et al., 2024) use pessimistic targets from ensembles to prevent value overestimation in uncertain regions. However, ensembles are computationally expensive and lack strong theoretical guarantees despite empirical success (Lakshminarayanan et al., 2017; D' Angelo & Fortuin, 2021). Beyond uncertainty-based methods, critic regularisation can also use action discrepancy penalties (Wu et al., 2019; Tarasov et al., 2024; Kostrikov et al., 2021), or directly learn conservative value lower bounds robust to

worst-case scenarios (Kumar et al., 2020; Lyu et al., 2022; Cheng et al., 2022). Our work builds on Valdettaro & Faisal (2024), where conservatism arises from Bayesian inference with a conservative value prior, ensuring high values are assigned only to supported regions, thus naturally avoiding out-of-distribution (OOD) actions without explicit robustness.

Bayesian inference in RL. Bayesian methods offer an approach to the exploration-exploitation challenge of online RL, either through uncertainty-guided heuristics (O'Donoghue et al., 2018; Agrawal & Goyal, 2017) or formalised as an adaptive expected value maximisation in belief-space MDPs (Duff, 2002; Poupart et al., 2006; Zhang et al., 2025). Recent work (Ghosh et al., 2022) revisits this adaptive formulation, investigating how offline datasets can inform non-Markovian policies that adapt to newly observed online data. Bayesian approaches have also been proposed for the offline-to-online setting, where uncertainty is used to balance online exploration in the presence of an existing offline dataset (Hu et al., 2024) These approaches apply Bayesian reasoning primarily to tackle the challenge of acquiring and incorporating new information during online interaction. In contrast, we consider the fully offline setting, where a Markovian, non-adaptive policy must be learned from a fixed dataset without further interaction. Existing uncertainty-based offline RL methods focus primarily on uncertainty estimation and conservative value learning, while a principled formulation of fully offline RL as posterior expected value maximisation remains an open challenge. Our work takes a step towards such a formulation for offline, model-free RL that retains the key Bayesian principle of posterior expected value optimisation.

A central challenge in off-policy model-free Bayesian inference setting is that value targets are not observed directly. Unlike supervised learning, or even on-policy Bayesian RL (e.g., GP-SARSA (Engel et al., 2005)), the agent must infer values by leveraging the Markov structure and combine together information across multiple transitions. While ensemble-based methods (Lakshminarayanan et al., 2017; Ovadia et al., 2019; Osband et al., 2018) have been used to approximate epistemic uncertainty in RL, their theoretical grounding and uncertainty quantification capabilities specifically in the context of TD learning remains unsettled (Buckman et al., 2020; Osband et al., 2018; Touati et al., 2020). In contrast, we adopt a fully probabilistic approach, placing a prior over value functions and performing inference consistent with the Bellman equation and the data.

Inference in function space Our method relies on placing a prior directly on the value function. Thus, we need to carry out Bayesian inference in function space rather than parameter space, which comes with an additional set of complexities. While some nonparametric approaches, such as Gaussian processes (GPs), naturally carry out inference in function space, these require approximations to be scalable to large datasets (Hensman et al., 2013; Titsias, 2009) and classically rely on hand-crafted kernels which is undesirable for complex tasks or environments. The field of functional variational inference (Burt et al., 2020; Ma & Hernández-Lobato, 2021) seeks to approximate inference in value-function space while still employing expressive parametric models. The deep learning variant of our approach is closest to the framework for functional variational inference proposed in Hafner et al. (2020), where the prior knowledge is injected into training by sampling pseudo-data points from a prior distribution.

3 Problem Setting and Probabilistic Model

Setting and notation. We work with the discounted Markov Decision Process (MDP) formalism (Sutton et al., 1998) with states $s \in \mathcal{S}$, actions $a \in \mathcal{A}$, discount factor $\gamma \in [0, 1)$, rewards $r \in \mathbb{R}$ sampled from an unknown reward function with mean $R(s, a)$ and next states $s' \in \mathcal{S}$ sampled from an unknown transition kernel $P(s'|s, a)$. We refer to environments with finite state-action spaces as *tabular*. For tractability, we assume deterministic transitions in the analysis. The objective is to use a fixed dataset of transitions $(s_i, a_i, r_i, s'_i) \in \mathcal{D}$ to learn a policy $\pi(a|s)$ mapping states to distributions over actions (with shorthand $a = \pi(s)$ for deterministic policies) that maximises expected cumulative discounted returns $\mathbb{E}(\sum_{t=0}^{\infty} \gamma^t r_t)$ where r_t is the reward sampled at step t, with $s_{t+1} \sim P(\cdot|s_t, a_t), a_{t+1} \sim \pi(s_t), s_0 \sim \eta_0(s)$ with η_0 being some unknown initial state distribution.

We refer to the (discounted) state distribution under π as $\eta^\pi(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \Pr(s_t = s \mid \pi)$. For a given policy π we have the value function $V^\pi(s) = \mathbb{E}(\sum_t \gamma^t r_t | s_0 = s)$ and action-values $Q^\pi(s,a) = R(s,a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}(V^\pi(s'))$. We denote with $\mathbf{Q}^\pi(s,a)$ the random variables associated with the belief in $Q^\pi(s,a)$ and refer to the random variables $\mathbf{Q}^\pi(s,a)$ across state-actions collectively as $\mathbf{Q}^\pi$. under a prior over value functions. Expectations and variances involving $\mathbf{Q}^\pi$ are therefore taken with respect to the probability measure induced by this prior, or by the corresponding posterior after conditioning on the dataset $\mathcal{D}$ through the probabilistic model introduced below. Finally, it will sometimes be convenient to refer to state-actions jointly, for which we will use the letter $x = (s,a)$ belonging to the set $X = \mathcal{S}\times\mathcal{A}$.

Probabilistic model. A key conceptual challenge in Bayesian model-free off-policy evaluation is that values are not observed directly. They must instead be inferred from information obtained in individual transitions and the knowledge that the environment is Markovian. By including the Bellman equation directly into the inference process, we ensure that posterior samples of the value function respect the environment's Markov structure and consistently combine information about value across multiple transitions. For a deterministic policy π and a deterministic environment, the action-values at any s, a satisfy the Bellman equation

$$Q^\pi(s,a) = R(s,a) + \gamma Q^\pi(s', \pi(s')),$$

where s' is the state deterministically reached from state s after taking action a. We place a probabilistic model directly on the (unknown) action-value function. Let $Q(s,a)$ be a generic realisation of $\mathbf{Q}^\pi(s,a)$. Then, our model assumes that the observed rewards r_i and the latent action-values Q corresponding to the transitions in the dataset $(s_i, a_i, r_i, s_i') \in \mathcal{D}$ are related through

$$r_i = Q(s_i,a_i) - \gamma_i Q(s_i', \pi(s_i')) + \varepsilon_i, \tag{1}$$

where $\gamma_i = \gamma$ if s_i' is not terminal and 0 otherwise, and ε_i a small zero-mean independent Gaussian noise term with variance σ_r^2.

Supported policies. Our analysis takes a complementary view to mainline offline RL theory. Theoretical guarantees in standard offline RL often require the learned policy to remain within the distribution of the dataset (Kostrikov et al., 2022; Fujimoto et al., 2019; Kumar et al., 2019). In contrast, we call a policy *supported* if the dataset contains sufficient information to infer its value. Being *in-distribution* and being *supported* (in our sense) can coincide for certain datasets, such as those from deterministic environments with complete trajectories, but they are not equivalent in general. For example, if the dataset was collected for a different task, demonstrations may terminate early; then a policy can be in-distribution while its long-term outcome remains undetermined by the data, and conservatism is still required. An inference-based perspective also lets us treat some OOD states as allowable. If we can bound the cost of visiting OOD states, then actions leading to them may be desirable when we are confident that sufficiently high reward will be obtained before leaving the well-understood region. Thus, rather than enforcing an in- or out-of-distribution distinction, an inference-based approach enables direct reasoning about this tradeoff and provides a more versatile way to represent the form of conservatism we seek in offline RL.

In the deterministic setting with finite state-action space considered in our theoretical analysis, it can be shown that the dataset $\mathcal{D}$ and a deterministic policy π induce a natural partition of state-action pairs into those whose values are identified by the data and those that are not. We denote by X_{sup} the set of *supported* state-actions, for which the value $Q^\pi(s,a)$ can be recovered exactly from $\mathcal{D}$ under the deterministic dynamics, and by X_{unsup} the remaining, *unsupported*, state-actions. These sets are policy dependent and are subsets of $\mathcal{S}\times\mathcal{A}$. A precise discussion and characterisation of this partition, in terms of the transition structure induced by π and $\mathcal{D}$, is provided in Appendix B. For a policy π to be *supported* we require it to choose supported actions at every state where those exist.

Establishing guarantees in a tractable setting. Before presenting the theoretical results, we note that the proposed evaluation scheme is formulated generally and applies to both tabular and

continuous environments. The tabular setting studied in Section 4 can be recovered as a special case of the continuous formulation in Section 5, obtained by using a kernel that assigns zero covariance between distinct state-actions. We focus our theoretical analysis on the tabular case because it allows for a tractable characterisation of supported and unsupported state-actions in deterministic environments. This enables us to analytically establish the desired properties of the evaluation scheme. While these guarantees do not directly apply to continuous state spaces, the continuous formulation preserves the same evaluation scheme and probabilistic interpretation analysed in the tabular analysis, providing the motivation for applying it in more complex environments.

4 Conservative Value Priors

In this section, we define and give motivation for why conservative value priors (CVPs) as a natural ingredient in a Bayesian treatment of offline RL. Our theoretical results rely on deterministic environment assumptions and concern tabular environments with a finite state-action space, which is in line with some other model-free offline RL theory (Fujimoto et al., 2019). We remark that compared to other theoretical contributions in the literature, we do not require our dataset to contain full trajectories terminating at terminal states, i.e. it is not required to be *coherent* as defined in Fujimoto et al. (2019).

Theoretical results. In the results that follows, we assume deterministic policies, a finite-state tabular environment with deterministic transitions and rewards, and employ Gaussian priors on the $\mathbf{Q}^\pi(s,a)$ random variables, with constant mean and standard deviation that are independent across state-actions. Having introduced X_{sup} and X_{unsup} as the sets containing supported and unsupported state-actions respectively in the previous section, we let $\hat{Q}^\pi(s,a) = \mathbb{E}(\mathbf{Q}^\pi(s,a)|\mathcal{D})$ denote the posterior mean and present Theorem 1.

Theorem 1. *For any dataset $\mathcal{D}$ and policy π, there exists μ_0, σ_0^2 such that if $\mathbf{Q}^\pi$ has prior mean $\mathbb{E}(\mathbf{Q}^\pi(s,a)) = \mu_Q < \mu_0$ and prior variance $\text{Var}(\mathbf{Q}^\pi(s,a)) = \sigma_Q^2 > \sigma_0^2$ at all $s \in \mathcal{S}, a \in \mathcal{A}$, then for any $x_u \in X_{\text{unsup}}$ and $x_s \in X_{\text{sup}}$, we have $\hat{Q}^\pi(x_u) < \hat{Q}^\pi(x_s)$.*

Theorem 1, proved in Appendix B separately for $\gamma = 0$ and $\gamma \in (0,1)$, establishes that choosing a suitable conservative prior assigns higher values to supported state-actions than unsupported ones. From the proof of Theorem 1, we also obtain the following result that bounds the distance between the posterior expected value to the ground-truth action-values $Q^\pi(s,a)$:

Corollary 2. *For any dataset $\mathcal{D}$, policy π and $\varepsilon > 0$, there exists a σ_0^2 such that if $\mathbf{Q}^\pi$ has prior variance $\text{Var}(\mathbf{Q}^\pi(s,a)) > \sigma_0^2$ at all $s \in \mathcal{S}, a \in \mathcal{A}$, then for all $x_s \in X_{\text{sup}}$, $\left|\hat{Q}^\pi(x_s) - Q^\pi(x_s)\right| < \varepsilon$.*

This result shows that, for state-actions supported by the dataset, one can choose a prior such that the posterior expected value is arbitrarily close to the ground truth.

Greedy improvement on supported state-actions. We can summarise the consequences of Theorem 1 and Corollary 2 to claim that for any deterministic policy π and any $\varepsilon > 0$, there exist a prior mean μ_Q and variance σ_Q^2 such that the posterior mean $\hat{Q}^\pi = \mathbb{E}(\mathbf{Q}^\pi|\mathcal{D})$ satisfies:

(i) For all $(s,a) \in X_{\text{sup}}$, $\left|\hat{Q}^\pi(s,a) - Q^\pi(s,a)\right| < \varepsilon$.

(ii) For all $x_s \in X_{\text{sup}}$ and all $x_u \in X_{\text{unsup}}$, $\hat{Q}^\pi(x_u) < \hat{Q}^\pi(x_s)$.

Consequently, any greedy policy π' with respect to $\hat{Q}^\pi$, $\pi'(s) \in \arg\max_a \hat{Q}^\pi(s,a)$, selects only supported state-actions whenever X_{sup} contains at least one action at s. Moreover, if at a state s the optimal supported state-action $a^* \in \arg\max_{a:(s,a)\in X_{\text{sup}}} Q^\pi(s,a)$ has an action gap larger than 2ε, i.e. $Q^\pi(s,a^*) \geq Q^\pi(s,a) + 2\varepsilon$ for all $a \neq a^*$ with $(s,a) \in X_{\text{sup}}$, then $\pi'(s) = a^*$, i.e. the greedy action under $\hat{Q}^\pi$ coincides with the greedy action under the true Q^π at s.

While the general form to find such a μ_Q from Theorem 1 is not tractable for larger scale problems, we can determine a *necessary* condition on μ_Q for greedy posterior selection to choose supported actions by considering the $\gamma = 0$ case. At the end of Appendix B we show that satisfying the following definition is a necessary for such a prior.

Definition 1. *A prior is defined as conservative with respect to a policy π if its prior mean satisfies* $\mu_Q(s,a) < Q^\pi(s,a) \quad$ *for all* $a \in \mathcal{A},\ s \in \text{supp}(\eta^\pi)$.

We distinguish between locally conservative (conservative with respect to a given policy) and globally conservative (conservative with respect to all policies) priors. For greedy posterior value action selection to remain in the support of the data, the prior must remain conservative for any π considered during training. We summary intuition for this in Figure 1. However, as noted when discussing supported policies in Section 3, if the prior adequately and conservatively reflects the belief of the outcome of a policy in OOD regions, we would still expect successful offline RL performance.

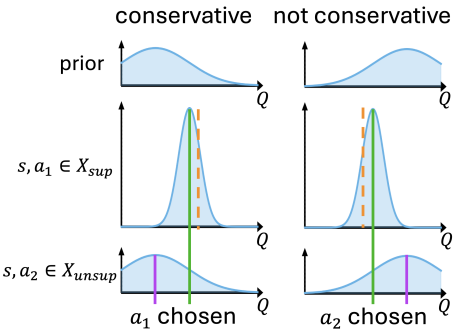


Figure 1: Effect of conservative (left) vs non-conservative (right) priors on action selection at a fixed state s with two candidate actions, a_1 and a_2. Action a_1 is supported by the dataset, whereas a_2 is unsupported. The dashed orange line denotes the value of observed data. For the supported action, the posterior mean (green) is close to the observed value. For the unsupported action, the posterior mean (purple) remains close to the prior mean. Consequently, if the prior is not conservative (Definition 1), maximising the posterior expected value selects the unsupported action a_2.

Choosing a suitable prior. Keeping this in mind, we propose two approaches for choosing a CVP in practice.

1. **Reward preprocessing (RP).** Consider translating all rewards in the dataset by a constant $r \to r - r_{\text{min}}$, with r_{min} being the smallest observed reward in the dataset. We show in Appendix C that, under some assumptions regarding the representativeness of the dataset, any prior with $\mu_Q < 0$ is then globally conservative by Definition 1 with the transformed rewards. For practical experiments, since μ_Q can be arbitrarily close to 0 with the same guarantee, we carry out RP and then set $\mu_Q = 0$.
2. **Task-specific prior design.** The Bayesian formulation allows us to inject task-specific prior knowledge when choosing the prior mean. Although this requires some domain insight, many tasks admit a natural baseline value (for example, the return of a "do-nothing" or default controller) that can be used as an initial choice for μ_Q. Even if such a baseline is not strictly conservative in the theoretical sense, we still expect good practical performance provided it adequately reflects the expected performance of a policy in regions where no informative data are available. This baseline thus provides a reasonable starting point around which μ_Q can be further tuned as a hyperparameter. See, for example, the robotics experiments in Section 6.2.

While RP provides a general recipe for finding a suitable prior mean, the drawback is that an excessively pessimistic prior may introduce excessive regularisation and harm performance, so the choice of approach should be considered on a task-by-task basis.

5 CVP via Gaussian Processes

We present here an algorithm that implements the CVP idea with GPs with exact inference, suitable for simple environments and small datasets. To tackle continuous states, we generalise to settings in which a kernel induces arbitrary prior dependencies across state-action pairs. The tabular setting with independent priors from the previous section is recovered as a special case corresponding to the

delta kernel. Through a qualitative evaluation of its decision-making and uncertainty quantification capabilities in the mountain car classic control task, we demonstrate the soundness of the CVP approach in its most direct form.

5.1 Off-policy Evaluation with GPs

The derivation we present here builds on the work in Valdettaro & Faisal (2024), which itself provides an off-policy generalisation of the line of work in Engel et al. (2005), to achieve TD off-policy inference in value-function space for continuous state and action spaces. Our objective is to find a policy that maximises the off-policy objective (Degris et al., 2012)

$$J(\pi) = \int \rho(s) V^\pi(s)\, ds \approx \frac{1}{|\mathcal{D}|} \sum_{s \in \mathcal{D}} Q^\pi(s, \pi(s)), \tag{2}$$

where the state distribution under the dataset policy ρ is approximated by the dataset's empirical state distribution. Since Q^π is an unknown quantity, J is also unknown, and Bayesian decision theory (Robert et al., 2007) dictates that we maximise its expected posterior value, so the task is now to find an expression for $\mathbb{E}(\mathbf{Q}^\pi|\mathcal{D})$, which we can then use to find a π that optimises Eq. (2).

We consider a (conservative) Gaussian prior on $\mathbf{Q}^\pi$ with constant prior mean μ_Q and covariance kernel that factorises across state and action spaces $k_Q((s_1, a_1), (s_2, a_2)) = k_s(s_1, s_2)k_a(a_1, a_2)$. We proceed from the setup established in Section 4 and in particular from Eq. (1). Since each r_i is itself a linear combination of Gaussian random variables, we can establish its prior mean and the covariance between any two distinct observed rewards:

$$\mathbb{E}(r_i) = (1 - \gamma_i)\mu_Q$$

$$\text{cov}(r_i, r_j) = k(x_i, x_j) - \gamma_j k(x_i, x'_j) - \gamma_i k(x'_i, x_j) + \gamma_i \gamma_j k(x'_i, x'_j),$$

where we used the shorthand notation $x_k = (s_k, a_k)$ and $x'_k = (s'_k, \pi(s'_k))$. We can also compute the prior covariance between the observed rewards and the action-value of any arbitrary state-action:

$$\text{cov}(Q^\pi(s, a), r_i) = k((s, a), x_i) - \gamma_i k((s, a), x'_i).$$

We can now find the posterior value distribution for any state-action s, a by using the Gaussian conditioning formulas for posterior mean $\mu_{Q|\mathcal{D}}$ and covariance $\mathbf{\Sigma}_{Q|\mathcal{D}}$ (Rasmussen et al., 2006):

$$\begin{aligned} \mu_{Q|\mathcal{D}} &= \mathbf{m}_r + \mathbf{K}_{QR}^\top (\mathbf{K}_R + \sigma_r^2 \mathbf{I})^{-1} (\mathbf{r} - \mathbf{m}_r) \\ \mathbf{\Sigma}_{Q|\mathcal{D}} &= \mathbf{K}_Q - \mathbf{K}_{QR}^\top (\mathbf{K}_R + \sigma_r^2 \mathbf{I})^{-1} \mathbf{K}_{QR}, \end{aligned} \tag{3}$$

where we can populate $\mathbf{m}_r, \mathbf{K}_R, \mathbf{K}_{QR}$ and $\mathbf{K}_Q$ with the entries $\mathbb{E}(r_i), \text{cov}(r_i, r_j), \text{cov}(Q^\pi(s, a), r_i)$ and $k((s, a), (s, a))$ respectively. Notice that the action-value posterior's dependence on policy is differentiable and present through the terms containing $x' = (s', \pi(s'))$.

This allows us to compute posterior mean (and variance) for $\mathbf{Q}^\pi$ at any (s, a), giving a differentiable expression for the posterior expectation of Eq. (2) in terms of π, which can then be optimised through gradient ascent. This is summarised in Alg. 1. In principle, the posterior variance could also be used for action selection, but keeping in line with Bayesian decision theory we only consider posterior mean as the objective to maximise.

5.2 Continuous Control with Small Datasets

We qualitatively evaluate the exact inference CVP approach on the classic mountain car control problem (Moore, 1990), demonstrating its ability to learn good offline policies and meaningful uncertainty estimates. This environment is chosen for its two-dimensional state space, which allows intuitive visualisation of posterior value and uncertainty. The task is set in a low-dimensional state-action space and small-data regime. A brief description of the environment is provided in Appendix D.

Algorithm 1 CVP for offline RL with GPs

Require: Dataset $\mathcal{D}$, conservative Gaussian prior on $\mathbf{Q}^{\pi}$, parametric actor $\pi_\theta : \mathcal{S} \to \mathcal{A}$, learning rate η

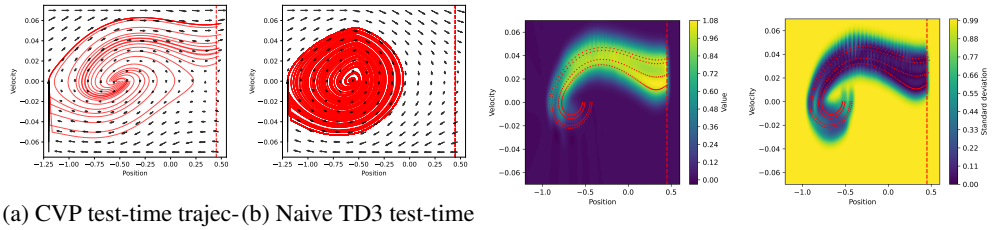
while not converged **do**

$\hat{Q} \leftarrow \mathbb{E}\left(\mathbf{Q}^{\pi_\theta} | \mathcal{D}\right)$ ▷ Evaluate $\mathbb{E}(\mathbf{Q}^{\pi_\theta}|\mathcal{D})$, Eq. (3)

$J(\theta) \leftarrow \frac{1}{|\mathcal{D}|} \sum_{s \in \mathcal{D}} \hat{Q}(s, \pi_\theta(s))$ ▷ Posterior mean of objective in Eq. (2)

$\theta \leftarrow \theta + \eta \nabla_{\theta'} J(\theta')$ ▷ Change π_θ in direction of increasing J.

end while



(a) CVP test-time trajectories (b) Naive TD3 test-time trajectories (c) Value (d) Standard deviation

Figure 2: Mountain car state-space visualisation. The black arrows in Figs. 2a and 2b show the transitions in state-space after following the learned policies for one timestep and we also include 15 trajectories that arise from uniformly spaced starting states (red continuous lines). Figs. 2c and 2d display learned posterior values and uncertainty (standard deviation) of the trained CVP agent with exact inference. The red dots indicate the states observed in the dataset. In all figures, crossing the vertical red dotted line corresponds to reaching the goal.

We use a variant with continuous actions and a sparse reward: 1 if the agent reaches the goal, and 0 otherwise. This experiment showcases three core capabilities of our method. First, it combines information from different transitions in a dataset to obtain policies that lead to in-distribution trajectories with high rewards. Secondly, it correctly assigns low values to policies that rely on OOD state-actions. Thirdly, it produces consistent Bayesian uncertainty estimates over multiple steps.

We choose the prior mean to be $\mu_Q = 0$, following the RP guideline from Section 4. The RBF covariance function's lengthscale was selected manually (see Appendix D for further details).

We apply Alg. 1 to a dataset consisting of 4 expert trajectories from a pre-trained agent (shown as red dots in Fig. 2). The resulting values, uncertainty and trajectories from the learned CVP policy are shown in Fig. 2. In Fig. 2a we display the dynamics under this policy and full episodic trajectories from uniformly spaced initial states, all of which successfully reach the goal. In contrast, a naive TD3 agent trained on the dataset as the replay buffer fails to do so (Fig. 2b. Further experimental details are given in Appendix D.

This behaviour illustrates the desired properties outlined at the start Section 5.2. The posterior value is correctly propagated across steps, being approximately equal to 1 close to the goal and slowly decaying as more steps become necessary to reach it. High values are restricted to in-distribution regions where data supports successful goal-reaching behaviour; elsewhere, values remain low. Interestingly, even in OOD regions where the policy successfully generalises to solve the task, it still assigns a low posterior value if it is out of the data distribution: here the agent 'tries its best' but does not expect a high value. In a complementary fashion, posterior uncertainty is low in the in-distribution regions and high OOD, as expected by a consistent quantification of epistemic uncertainty.

The focus of this section is to show that CVP can yield consistent value estimation and supported policies. We do not address the quantification of improvement over a suboptimal policy here, but we include at the end of Appendix F an example in an interpretable environment where the dataset contains suboptimal actions. This example provides evidence that, as expected, the exact-inference version of CVP remains effective in the presence of suboptimal actions in the dataset.

6 CVPs via Neural Networks

While exact Bayesian inference is suitable for small datasets and simple environments where a hand-crafted kernel is readily specified, sequential decision making from larger datasets on complex environments requires a more scalable approach. One option is to approximate exact inference with variational inference (Titsias, 2009), but this still leaves the difficult task of finding adequate prior kernels, and methods that try to learn these (Wilson et al., 2016; Ober et al., 2021; van Amersfoort et al., 2021) require introducing a highly non-trivial layer of algorithmic complexity on top of the already challenging task of offline RL. Therefore, here we opt instead to forego approximations to the full posterior distribution of a GP altogether and instead focus on approximately regressing the posterior mean only, as parametrised by a neural network. To implement this, we take inspiration from the approach in Hafner et al. (2020), where the inductive bias of inference in function space with a prior is introduced by sampling pseudo-data points from the prior.

6.1 A Scalable Implementation of CVPs

The key quantity that is optimised in Eq. (2) for decision making in Section 5 is the posterior mean of $\mathbf{Q}^\pi$. To maintain scalability to large datasets and complex environments, we employ a neural network $\hat{Q}$ to represent it. For tractability, we simplify the probabilistic model from Section 5 to employ independent Gaussian priors across state-actions, corresponding to choosing a delta covariance kernel $k_Q((s_1, a_1), (s_2, a_2)) = \delta_s(s_1 - s_2)\delta_a(a_1 - a_2)$. We can then find a training objective that ensures that $\hat{Q}$ regresses to this posterior mean. We show in Appendix E that the appropriate loss function to achieve this is given by

$$L(\theta) = L_B(\theta) + \alpha \mathbb{E}_{(s,a)\sim p_{\text{pseudo}}(s,a)}(\hat{Q}_\theta(s, a) - \mu_Q)^2, \tag{4}$$

where L_B is the standard Bellman TD critic loss

$$L_B(\theta) = \sum_{(s,a,r,s')\in\mathcal{D}} (\hat{Q}_\theta(s, a) - r - \gamma \hat{Q}_\theta(s', \pi(s')))^2,$$

μ_Q is the prior mean on Q, p_{pseudo} is a prior pseudo-data distribution used to regularise the learned value function and α is a hyperparameter. The result in Appendix E holds when p_{pseudo} is uniform, where we show the L_B term can be interpreted as a log-likelihood loss and the other term as the contribution from the prior, with the corollary that the regularisation strength α can be interpreted as being inversely proportional to the prior variance. An intuitive interpretation of Eq. (4) is that $\hat{Q}$ will be incentivised to fit to the data in the in-distribution regions where the L_B term dominates and will instead regress to the prior otherwise. We further note that the Eq. (4) can be interpreted as a TD analogue of Eq. 2 in Hafner et al. (2020), where the KL between two Gaussians recovers the same prior term proportional to the difference of the means squared.

While the assumption of independent state-actions would not lead to useful generalisation with non-parametric methods in continuous state-action spaces (such as GPs), when employing a parametric function approximator with strong generalisation capabilities, such as neural networks, this is not problematic: since hand-crafted kernels are unlikely to achieve good results in complicated environments anyway, we let the inductive bias implicit in the neural networks handle generalisation instead. On the other hand, the condition that p_{pseudo} is taken to be uniform will run into serious issues in high-dimensional environments. Thus, we instead restrict ourselves to regularising the most relevant state-actions towards decision-making. For a dataset state distribution ρ, a natural pseudo-data distribution for a stochastic policy would be $\rho(s)\pi(a|s)$ whereas for a deterministic policy we can sample so as to ensure regularisation around the learned policy's boundary

$$p_{\text{pseudo}}(s, a) = \rho(s)\mathcal{N}(a|\pi_\theta(s), \sigma_a^2),$$

for some constant σ_a^2 with similar intuition that this prioritises regularising the most relevant actions, which are those closest to the ones considered by the learned policy, as in Hafner et al. (2020). The

resulting critic loss is summarised in Algorithm 2, which we observe is a straightforward modification of standard off-policy online algorithms' loss that simply involves introducing a readily-computable regularisation term arising from the prior's contribution.

Algorithm 2 Scalable CVP critic update loss

Require: Batch of transitions $\mathcal{B}$ sampled from a dataset $\mathcal{D}$, online off-policy algorithm with deterministic or stochastic actor π, critic $\hat{Q}_\theta$ and critic loss $L_B(\theta)$, conservative prior mean μ_Q, prior regularisation strength α, pseudo-data number of samples n and noise scale σ_a.

for $(s, a, r, s') \in \mathcal{B}$ **do**

for $s_p \in \{s, s'\}$ **do**

Sample n actions $a_p^{(s_p)} \leftarrow \begin{cases} a_p^{(s_p)} = \pi(s_p) + \sigma_a z, z \sim \mathcal{N}(\cdot|0, \mathbf{I}_n) & \text{if } \pi \text{ deterministic} \\ a_p^{(s_p)} \sim \pi(\cdot|s_p) & \text{if } \pi \text{ stochastic} \end{cases}$

end for

end for

$\mathcal{S}_\mathcal{B} \leftarrow \{s, s' \mid (s, a, r, s') \in \mathcal{B}\}$ ▷ Use empirical state distribution to approximate $\rho(s)$

$$L \leftarrow L_B(\theta) + \alpha \frac{1}{|\mathcal{B}|n} \sum_{s_p \in \mathcal{S}_\mathcal{B}} \sum_{a_p^{(s_p)}} (\hat{Q}_\theta(s_p, a_p^{(s_p)}) - \mu_Q)^2$$

return loss L

Having established an approach to include the conservative prior's effect in the critic's evaluation, we can adapt any off-policy online algorithm by adding the new term in Eq. (4) to obtain a suitably regularised offline RL critic. In Appendix F we present a comparison between applying Alg. 1 and Alg. 2 for both a stochastic (SAC) and deterministic (TD3) base algorithm on a structured, easily interpretable toy environment, and observe that the key desirable behaviour is consistent across all three implementations. For the remainder of this work, we use the stochastic policy variant, with SAC as the base algorithm for experiments.

6.2 Offline Robotic Locomotion

Task	BC	SAC	CQL	TD3+BC	IQL	CVP-SAC (ours)
halfcheetah-r	2.2±0.0	29.7±1.4	17.5±1.5	11.0±1.1	13.1±1.3	**32.5±1.5**
hopper-r	3.7±0.6	9.9±1.5	7.9±0.4	8.5±0.6	7.9±0.2	**27.5±8.0**
walker2d-r	1.3±0.1	0.9±0.8	**5.1±1.3**	1.6±1.7	**5.4±1.2**	**6.9±3.0**
halfcheetah-m	43.2±0.6	55.2±27.8	47.0±0.5	48.3±0.3	47.4±0.2	**66.0±1.8**
hopper-m	**54.1±3.8**	0.8±0.0	**53.0±28.5**	**59.3±4.2**	**66.2±5.7**	**72.2±20.0**
walker2d-m	70.9±11.0	-0.3±0.2	73.3±17.7	**83.7±2.1**	78.3±8.7	**84.1±1.0**
halfcheetah-m-r	37.6±2.1	0.8±1.0	45.5±0.7	44.6±0.5	44.2±1.2	**57.2±0.9**
hopper-m-r	16.6±4.8	7.4±0.5	88.7±12.9	60.9±18.8	94.7±8.6	**100.5±4.2**
walker2d-m-r	20.3±9.8	-0.4±0.3	**81.8±2.7**	**81.8±5.5**	73.8±7.1	**81.5±11.1**
halfcheetah-m-e	44.0±1.6	28.4±19.4	75.6±25.7	90.7±4.3	86.7±5.3	**95.4±0.6**
hopper-m-e	53.9±4.7	0.7±0.0	**105.6±12.9**	98.0±9.4	91.5±14.3	**103.8±10.1**
walker2d-m-e	90.1±13.2	1.9±3.9	107.9±1.6	**110.1±0.5**	109.6±1.0	109.3±0.9
halfcheetah-e	91.8±1.5	-0.8±1.8	**96.3±1.3**	**96.7±1.1**	95.0±0.5	95.1±0.6
hopper-e	107.7±0.7	0.7±0.0	96.5±28.0	107.8±7	**109.4±0.5**	**110.7±1.5**
walker2d-e	106.7±0.2	0.7±0.3	108.5±0.5	**110.2±0.3**	109.9±1.2	**110.0±0.2**
Average	49.6	9.0	67.3	67.6	68.9	**76.9**

Table 1: D4RL results on random (-r), medium (-m), medium-replay (-m-r), medium-expert (-m-e) and expert (-e) datasets. Scores are normalised such that a random policy scores 0 and an expert policy scores 100. Baseline results are taken from Lyu et al. (2022). We bold those scores within one standard deviation of the method with the highest average.

We demonstrate our framework's scalability by evaluating the SAC-based CVP variant (Alg. 2) on D4RL locomotion tasks (Fu et al., 2020). We build our implementation on top of the CORL (Tarasov et al., 2022) CQL code, and do not modify any hyperparameters relating to the architecture or optimisation of the base SAC algorithm, with the only exception being the optimiser used for the automatic entropy tuning parameter loss (Haarnoja et al., 2018b): we found that ADAM (Kingma & Ba, 2014) updates on this single parameter led to similar performance but presented convergence issues after training stabilised, so we used stochastic gradient descent to optimise this loss instead. Full hyperparameter settings and other experimental details are provided in Appendix G.

Choosing CVP hyperparameters. The two main hyperparameters necessary for CVP are prior mean and α. For the choice of prior mean, the locomotion tasks have a reward of the form *movement reward* minus *control cost*, so a 'do nothing' policy would have value approximately equal to 0. Thus, setting $\mu_Q = 0$ provides a suitable conservative prior mean. While we cannot prove that this is a CVP in the strict sense of Definition 1, this is a reasonable base value to start from. We found that for one task (hopper-medium-expert), further tuning of μ_Q helped performance while using the RP heuristic (see Section 4) was instead beneficial for other tasks. Choosing α is less straightforward since, unlike μ_Q, there was no clear default, so we tuned it per task. We briefly discuss in Appendix H how training diagnostics can help guide the choice of α.

D4RL results. In Table 1 we display results on a variety of D4RL `v2` locomotion tasks (Fu et al., 2020). We report the results of evaluating the best-performing hyperparameters after evaluating on the ground-truth environments, reporting averages and sample standard deviations over 5 random seeds. The values for the other methods reported are taken from Lyu et al. (2022). We compare results with behaviour cloning (BC), SAC (Haarnoja et al., 2018a), CQL (Kumar et al., 2020), TD3+BC (Fujimoto & Gu, 2021) and IQL (Kostrikov et al., 2022). These, like ours, are straightforward adaptations of base off-policy model-free algorithms. We include in Appendix I additional results comparing to other two more sophisticated model-free algorithms that employ additional ingredients, MCQ (Lyu et al., 2022) and ATAC (Cheng et al., 2022), where we see that our simpler algorithm still recovers similarly good performance, achieving 97% and 102% of their averaged D4RL score respectively.

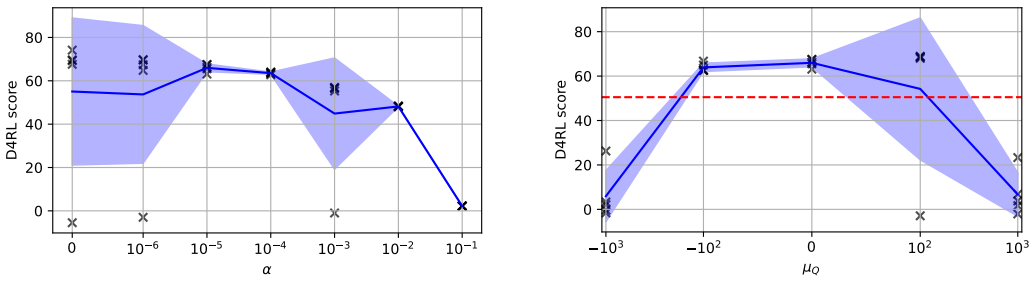
Algorithmically, the most closely related baseline to our approach in Table 1 is CQL. CVP-SAC shares the same base algorithm and the difference between the two boils down to the form of the critic regularisation term, a consequence of the two approaches' fundamentally different starting principles. We see a significant improvement in performance from these differences.

6.3 Hyperparameter Study

To illustrate the impact of α and the choice of prior mean on the performance of our algorithm, we present results of varying these on the exemplar halfcheetah-medium task. The overall trend we notice is that, generally speaking, decreasing regularisation strength (decreasing α or increasing μ_Q) tends to have a beneficial effect on performance up to a critical point, after which performance and training becomes unstable. On the other hand, adding excessive regularisation tends to have a more gradual impact on performance. These insights can help tune α from training diagnostics, as discussed in Appendix H.

7 Discussion and Limitations

We have presented a Bayesian formulation of the offline RL continuous control problem through conservative value priors. We demonstrated the qualitatively desirable behaviour of our approach on a classic control problem and strong quantitative performance on robotics tasks with larger datasets and complex environments through a simple modification to standard online RL algorithms, outperforming comparable prior model-free methods while remaining highly competitive with more sophisticated and algorithmically complex methods. On small datasets where exact inference is feasible, our method provides consistent epistemic uncertainty quantification. However, since uncertainty quantification is



(a) Varying α with constant $\mu_Q = 0$.

(b) Varying μ_Q with constant $\alpha = 1e-5$.

Figure 3: Hyperparameter study on halfcheetah-medium, with 5 seeds per configuration and shaded sample standard deviation. The red line in Fig. 3b corresponds to the average performance after carrying out RP as described in Section 4.

decoupled from decision-making in our framework, we can scale to larger problems without strictly requiring it. By unifying Bayesian decision theory, uncertainty quantification and the challenge of avoiding out-of-distribution actions in offline RL, we establish a foundation for a principled but scalable probabilistic approach to decision-making in offline RL.

Finally, we summarise the limitations of our approach. While in their current form the algorithms provided can also be applied to stochastic environments, our theoretical analysis and empirical evaluation is currently restricted to deterministic MDPs. A fully Bayesian approach to offline RL with stochastic environments would require a world model, which is beyond the scope of our model-free approach. The theoretical guarantees are only provided for tabular environments, and we defer formulation of these for the continuous setting to future work. Further, the choice of a suitable conservative prior, along with associated hyperparameters, is problem-specific and may be challenging in some contexts. The exact inference version of CVP does not scale well to large datasets and may face challenges in complex, high-dimensional environments. While GP-based approximate inference methods (Titsias, 2009) could in principle be applied in our setting to tackle scalability issues, it is unclear whether they are viable for accurately representing a critic function. A limitation towards the applicability of the deep learning, scalable variant of CVP is finding an appropriate regularisation strength α without ground-truth evaluation. Preliminary results suggest that it may be possible to tune α from training diagnostics (see Appendix H), but further work is necessary to verify this claim's robustness. We also observe relatively high variance in performance on certain tasks, such as hopper-medium, which is however not uncommon in offline RL algorithms due to the challenging nature of the task. Finally, while building scalable Bayesian uncertainty estimates is in principle possible from the formulation we have proposed, deep off-policy evaluation itself is difficult (Fu et al., 2021), so assigning accurate uncertainty measures is likely to be a challenging task, albeit one with huge potential for impact towards the practical applicability of offline RL.

Broader Impact Statement

This work develops a conceptual and methodological contribution to offline reinforcement learning and is evaluated on benchmark control tasks. We do not anticipate any direct negative societal impact, but as with any RL method, care should be taken when applying it in high-stakes real-world domains, where additional safety and fairness considerations are required.

Acknowledgements

FV was supported by a Imperial College London Department of Computing PhD scholarship. AAF acknowledges support of the UKRI AI programme, and the Engineering and Physical Sciences Research Council (EPSRC), for the AI Hub in Generative Models (EP/Y028805/1), and a UKRI Turing AI Fellowship (EP/V025449/1).

References

Shipra Agrawal and Navin Goyal. Near-optimal regret bounds for Thompson sampling. *Journal of the ACM (JACM)*, 64(5):1–24, 2017.

Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in Neural Information Processing Systems*, 34:7436–7447, 2021.

Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pp. 30–37. Morgan Kaufmann, San Francisco (CA), 1995. ISBN 978-1-55860-377-6. DOI: https://doi.org/10.1016/B978-1-55860-377-6.50013-X.

Jacob Buckman, Carles Gelada, and Marc G Bellemare. The importance of pessimism in fixed-dataset policy optimization. *arXiv preprint arXiv:2009.06799*, 2020.

David R Burt, Sebastian W Ober, Adrià Garriga-Alonso, and Mark van der Wilk. Understanding variational inference in function-space. *arXiv preprint arXiv:2011.09421*, 2020.

Ching-An Cheng, Tengyang Xie, Nan Jiang, and Alekh Agarwal. Adversarially trained actor critic for offline reinforcement learning. In *International Conference on Machine Learning*, pp. 3852–3878. PMLR, 2022.

Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. Robonet: Large-scale multi-robot learning. In *Conference on Robot Learning*, pp. 885–897. PMLR, 2020.

Thomas Degris, Martha White, and Richard Sutton. Off-policy actor-critic. *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, 1, 05 2012.

Francesco D' Angelo and Vincent Fortuin. Repulsive deep ensembles are Bayesian. In *Advances in Neural Information Processing Systems*, volume 34, pp. 3451–3465. Curran Associates, Inc., 2021.

Michael O'Gordon Duff. *Optimal Learning: Computational Procedures for Bayes-adaptive Markov Decision Processes*. University of Massachusetts Amherst, 2002.

Yaakov Engel, Shie Mannor, and Ron Meir. Reinforcement learning with Gaussian processes. In *Proceedings of the 22nd International Conference on Machine Learning*, pp. 201–208, 2005.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Justin Fu, Mohammad Norouzi, Ofir Nachum, George Tucker, Ziyu Wang, Alexander Novikov, Mengjiao Yang, Michael R Zhang, Yutian Chen, Aviral Kumar, et al. Benchmarks for deep off-policy evaluation. *International Conference on Learning Representations*, 2021.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34:20132–20145, 2021.

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

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

Kamyar Ghasemipour, Shixiang Shane Gu, and Ofir Nachum. Why so pessimistic? Estimating uncertainties for offline RL through ensembles, and why their independence matters. *Advances in Neural Information Processing Systems*, 35:18267–18281, 2022.

Dibya Ghosh, Anurag Ajay, Pulkit Agrawal, and Sergey Levine. Offline RL policies should be trained to be adaptive. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 7513–7530. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/ghosh22a.html.

Omer Gottesman, Fredrik Johansson, Matthieu Komorowski, Aldo Faisal, David Sontag, Finale Doshi-Velez, and Leo Anthony Celi. Guidelines for reinforcement learning in healthcare. *Nature Medicine*, 25(1):16–18, 2019.

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

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

Danijar Hafner, Dustin Tran, Timothy Lillicrap, Alex Irpan, and James Davidson. Noise contrastive priors for functional uncertainty. In *Uncertainty in Artificial Intelligence*, pp. 905–914. PMLR, 2020.

James Hensman, Nicolò Fusi, and Neil D. Lawrence. Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, UAI'13, pp. 282–290, Arlington, Virginia, USA, 2013. AUAI Press.

Hao Hu, Yiqin Yang, Jianing Ye, Chengjie Wu, Ziqing Mai, Yujing Hu, Tangjie Lv, Changjie Fan, Qianchuan Zhao, and Chongjie Zhang. Bayesian design principles for offline-to-online reinforcement learning. In *International Conference on Machine Learning*, pp. 19491–19515. PMLR, 2024.

Zhiyu Huang, Jingda Wu, and Chen Lv. Efficient deep reinforcement learning with imitative expert priors for autonomous driving. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pp. 651–673. PMLR, 2018.

Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda, John-Mark Allen, Vinh-Dieu Lam, Alex Bewley, and Amar Shah. Learning to drive in a day. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8248–8254. IEEE, 2019.

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

Matthieu Komorowski, Leo A Celi, Omar Badawi, Anthony C Gordon, and A Aldo Faisal. The artificial intelligence clinician learns optimal treatment strategies for sepsis in intensive care. *Nature Medicine*, 24(11):1716–1720, 2018.

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

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems*, 30, 2017.

Jiafei Lyu, Xiaoteng Ma, Xiu Li, and Zongqing Lu. Mildly conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2022.

Chao Ma and José Miguel Hernández-Lobato. Functional variational inference based on stochastic process generators. *Advances in Neural Information Processing Systems*, 34:21795–21807, 2021.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Andrew William Moore. Efficient memory-based learning for robot control. Technical report, University of Cambridge, 1990.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. AWAC: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

Sebastian W Ober, Carl E Rasmussen, and Mark van der Wilk. The promises and pitfalls of deep kernel learning. In *Uncertainty in Artificial Intelligence*, pp. 1206–1216. PMLR, 2021.

Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. *Advances in Neural Information Processing Systems*, 31, 2018.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in Neural Information Processing Systems*, 32, 2019.

Brendan O'Donoghue, Ian Osband, Remi Munos, and Volodymyr Mnih. The uncertainty Bellman equation and exploration. In *International conference on machine learning*, pp. 3836–3845, 2018.

Pascal Poupart, Nikos Vlassis, Jesse Hoey, and Kevin Regan. An analytic solution to discrete bayesian reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pp. 697–704, 2006.

Carl Edward Rasmussen, Christopher KI Williams, et al. *Gaussian Processes for Machine Learning*, volume 1. Springer, 2006.

Christian P Robert et al. *The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation*, volume 2. Springer, 2007.

Samarth Sinha, Ajay Mandlekar, and Animesh Garg. S4RL: Surprisingly simple self-supervision for offline reinforcement learning in robotics. In *Conference on Robot Learning*, pp. 907–917. PMLR, 2022.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement Learning: An Introduction*, volume 1. MIT press Cambridge, 1998.

Denis Tarasov, Alexander Nikulin, Dmitry Akimov, Vladislav Kurenkov, and Sergey Kolesnikov. CORL: Research-oriented deep offline reinforcement learning library. In *3rd Offline RL Workshop: Offline RL as a "Launchpad"*, 2022. URL https://openreview.net/forum?id=SyAS49bBcv.

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Andrei N. Tikhonov and Vasiliy Y. Arsenin. *Solutions of Ill-Posed Problems*. Winston and Sons, Washington, DC, 1977. ISBN 9780470991244.

Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial Intelligence and Statistics*, pp. 567–574. PMLR, 2009.

Ahmed Touati, Harsh Satija, Joshua Romoff, Joelle Pineau, and Pascal Vincent. Randomized value functions via multiplicative normalizing flows. In *Uncertainty in Artificial Intelligence*, pp. 422–432. PMLR, 2020.

Filippo Valdettaro and A. Aldo Faisal. Towards offline reinforcement learning with pessimistic value priors. In *Epistemic Uncertainty in Artificial Intelligence*, pp. 89–100, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-57963-9.

Joost van Amersfoort, Lewis Smith, Andrew Jesson, Oscar Key, and Yarin Gal. On feature collapse and deep kernel learning for single forward pass uncertainty. *arXiv preprint arXiv:2102.11409*, 2021.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial Intelligence and Statistics*, pp. 370–378. PMLR, 2016.

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

Teng Xiao and Donglin Wang. A general offline reinforcement learning framework for interactive recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4512–4520, 2021.

Huan Xu and Shie Mannor. The robustness-performance tradeoff in Markov decision processes. *Advances in Neural Information Processing Systems*, 19, 2006.

Kai Yang, Jian Tao, Jiafei Lyu, and Xiu Li. Exploration and anti-exploration with distributional random network distillation. In *International Conference on Machine Learning*, pp. 56397–56421. PMLR, 2024.

Shenao Zhang, Yaqing Wang, Yinxiao Liu, Tianqi Liu, Peter Grabowski, Eugene Ie, Zhaoran Wang, and Yunxuan Li. Beyond Markovian: Reflective exploration via Bayes-adaptive RL for LLM reasoning. *arXiv preprint arXiv:2505.20561*, 2025.

Wenxuan Zhou, Sujay Bajracharya, and David Held. PLAS: Latent action space for offline reinforcement learning. In *Conference on Robot Learning*, pp. 1719–1735. PMLR, 2021.

Supplementary Materials

The following content was not necessarily subject to peer review.

A Notation

Symbol	Meaning
$\mathcal{S}, \mathcal{A}$	State space and action space.
$s \in \mathcal{S}$, $a \in \mathcal{A}$	State and action.
$\gamma \in [0, 1)$	Discount factor.
$r \in \mathbb{R}$	Reward.
$R(s, a)$	Mean reward function.
$P(s' \mid s, a)$	Transition kernel.
$\mathcal{D}$	Offline dataset of transitions (s_i, a_i, r_i, s'_i).
$\pi(a \mid s)$	Policy; for deterministic policies, we write $a = \pi(s)$.
η_0	Initial state distribution.
$\eta^\pi(s)$	Discounted state visitation distribution under policy π.
$V^\pi(s)$	Value function under policy π.
$Q^\pi(s, a)$	Action-value function under policy π.
$\mathbf{Q}^\pi(s, a)$	Random variable representing the Bayesian belief about $Q^\pi(s, a)$.
$\mathbf{Q}^\pi$	Collection of random variables $\mathbf{Q}^\pi(s, a)$ over state-actions.
$x = (s, a)$	State-action pair.
$X = \mathcal{S} \times \mathcal{A}$	State-action space.
X_{sup}	Set of supported state-actions, whose values are identifiable from the dataset under policy π.
X_{unsup}	Set of unsupported state-actions, whose values are not identifiable from the dataset under policy π.
$\rho(s)$	State distribution induced by the dataset policy.
$J(\pi)$	Off-policy objective optimised by the actor.
μ_Q	Prior mean placed on action-values.
σ_r^2	Observation noise variance.
σ_Q^2	Prior variance on action-values.
$k_Q((s_1, a_1), (s_2, a_2))$	Covariance kernel on action-values in the GP formulation.
$\hat{Q}^\pi(s, a)$	Posterior mean of $\mathbf{Q}^\pi(s, a)$.
π_θ	Parametric actor.
$\hat{Q}_\theta(s, a)$	Neural-network critic used to approximate the posterior mean in the scalable formulation.
$L_B(\theta)$	Standard Bellman temporal-difference critic loss.
$L(\theta)$	CVP critic loss, consisting of the Bellman loss plus prior regularisation.
α	Prior regularisation strength in the scalable formulation.
$p_{\text{pseudo}}(s, a)$	Pseudo-data distribution used to regularise the critic toward the prior.

Table 2: Summary of notation used in the main text.

B Proof of Theorem 1

We provide a proof here of Theorem 1, starting from motivation for the definitions introduced in Section 4. Since the environment is deterministic, multiple observations of the same state–action pair necessarily yield the same outcome. Therefore, without loss of generality, we assume that the dataset contains at most one observation for each state–action pair, as duplicate observations do not provide additional information.

Posterior mean as a minimisation objective. The starting point for the proof is the statistical model introduced in Section 3. We denote a generic instantiation of $\mathbf{Q}$ as Q. Under the assumption of observation noise variance σ_r^2, independent Gaussian state-action priors with constant mean μ_Q and variance σ_Q^2, the posterior over $\mathbf{Q}$ has density of the form

$$
\begin{aligned}
&p(Q|\mathcal{D}) \propto p(Q)p(\mathcal{D}|Q) \\
&\propto \exp\left\{ \sum_{s\in\mathcal{S},a\in\mathcal{A}} -\frac{(Q(s,a)-\mu_Q)^2}{2\sigma_Q^2} \right\} \exp\left\{ -\frac{1}{2\sigma_r^2} \sum_{(s_i,a_i,r_i,s_i')\in\mathcal{D}} (Q(s_i,a_i)-r_i-\gamma Q(s_i',\pi(s_i'))^2 \right\} \\
&\propto \exp\left\{ -\frac{1}{2} \sum_{s\in\mathcal{S},a\in\mathcal{A}} \frac{(Q(s,a)-\mu_Q)^2}{\sigma_Q^2} - \frac{1}{2\sigma_r^2} \sum_{(s_i,a_i,r_i,s_i')\in\mathcal{D}} (Q(s_i,a_i)-r_i-\gamma Q(s_i',\pi(s_i'))^2 \right\} \\
&\propto \exp\left\{ -\frac{1}{2\sigma_r^2} \left(\sum_{s\in\mathcal{S},a\in\mathcal{A}} \sigma_r^2 \frac{(Q(s,a)-\mu_Q)^2}{\sigma_Q^2} + \sum_{(s_i,a_i,r_i,s_i')\in\mathcal{D}} (Q(s_i,a_i)-r_i-\gamma Q(s_i',\pi(s_i'))^2 \right) \right\}.
\end{aligned}
$$

Maximising $p(Q|\mathcal{D})$ is equivalent to minimising $-\log p(Q|\mathcal{D})$, so the MAP estimate of Q will minimise

$$
\sum_{(s_i,a_i,r_i,s_i')\in\mathcal{D}} (Q(s_i,a_i)-r_i-\gamma Q(s_i',\pi(s_i'))^2 + \sum_{s\in\mathcal{S},a\in\mathcal{A}} \frac{\sigma_r^2}{\sigma_Q^2}(Q(s,a)-\mu_Q)^2. \tag{5}
$$

Standard properties of Gaussians imply that the MAP estimate and posterior mean coincide, so the Q that optimise this objective will coincide with the posterior means $\mathbb{E}(\mathbf{Q}^\pi(s,a)|\mathcal{D})$. To simplify analysis, we don't model terminal states distinctly, and instead treat them as any other state that transitions to itself with zero reward.

Transition graph and supported state-actions. Given $\mathcal{D}$, one can build an empirical directed transition graph induced by π with nodes in state-action space, which we denote by G_π. G_π is constructed by connecting (s_1,a_1) to (s_2,a_2) if and only if the transition from (s_1,a_1) to s_2 is observed in $\mathcal{D}$ and $\pi(s_2)=a_2$. Each edge in G_π corresponds to a term in the sum over $\mathcal{D}$ in Eq. (5). Note that the condition that the environment and policy are deterministic imply that each edge has at most one outgoing node. Consequently, following the directed path from any vertex gives a unique trajectory, which either terminates at a sink (node with no outgoing edges) or eventually enters a directed cycle. Since such a path can enter at most one cycle, no weakly connected component can contain more than one directed cycle.

We identify those state-actions that are in components with cycles as *supported* and those in components without cycles as *unsupported*. Those components with a cycle will have number of edges equal to number of nodes, and one can treat the nodes of the component as a smaller Markov reward process (MRP) induced under π, where we have complete knowledge of the transitions, and are thus able to infer the action-values everywhere on the component. For example, a trajectory that leads to a terminal state would correspond to such a component, with the cycle being the terminal state looping back on itself. On the other hand, those components that have a sink and no cycle correspond to state-actions where the dataset does not contain sufficient information to infer the values of the nodes. For example, if a policy produces a trajectory that leads to a state where π is not observed in the dataset, while it is possible to infer relative values between the nodes, there is not sufficient information in the dataset to infer the exact action-values, and they remain unspecified up to a constant shift. Thus, these state-actions are *unsupported* and value estimation for these should be approached conservatively. We show this schematically in Figure 4.

The high-level approach of the proof is to show that, as $\lambda \to 0^+$, supported and unsupported components converge to different limiting values. On supported nodes, this limit is independent of the prior and tends to the ground-truth action-values, whereas on unsupported nodes the limit retains a dependence on the prior mean. We can exploit this additional degree of freedom on unsupported nodes to separate their values from those of supported nodes.

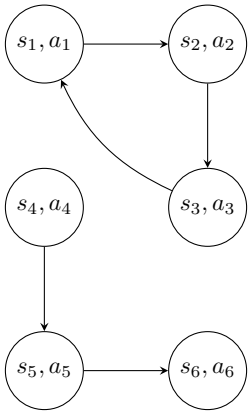


Figure 4: Schematic example of a transition graph G_π. The component $(s_1, a_1) \to (s_2, a_2) \to (s_3, a_3) \to (s_1, a_1)$ contains a cycle and is therefore supported. In this case, there are 3 nodes and 3 edges, corresponding to 3 unknowns and 3 linear constraints, so the value at each node in this component can be inferred. The component $(s_4, a_4) \to (s_5, a_5) \to s_6, a_6$ has a sink and no cycle, so its state-actions are unsupported. While we can infer the relative value difference between the nodes, we have an additional degree of freedom for the absolute value at these state-actions, and do not have sufficient information to infer the ground-truth values.

Linear algebraic setup and notation. Denote the set of supported and unsupported state-actions as X_{sup} and X_{unsup} respectively. Similarly, we call the components with cycles supported components and those with sink nodes unsupported components. The minimisation objective in Eq. (5) decomposes component-wise, so carry our analysis on each graph component individually. Denote the components of G_π by $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K$. Let $\mathcal{I}^{(k)}$ be the index set of all state-action pairs (s, a) that appear either as source nodes (s_i, a_i) or as successor nodes $(s'_i, \pi(s'_i))$ in $\mathcal{C}_k$. We enumnerate the elements of them as

$$\mathcal{I}^{(k)} = \{1, \ldots, |\mathcal{I}^{(k)}|\},$$

and identify an index $j \in \mathcal{I}^{(k)}$ with the corresponding pair (s_j, a_j). For each transition $(s_i, a_i, r_i, s'_i) \in \mathcal{D}$, we identify the corresponding component k and define the row vector $\phi_i \in \mathbb{R}^{|\mathcal{I}^{(k)}|}$ with elements

$$(\phi_i)_{(s,a)} = \begin{cases} +1, & \text{if } (s,a) = (s_i, a_i) \text{ and } (s_i, a_i) \neq (s'_i, \pi(s'_i)), \\ -\gamma, & \text{if } (s,a) = (s'_i, \pi(s'_i)) \text{ and } (s_i, a_i) \neq (s'_i, \pi(s'_i)), \\ 1-\gamma, & \text{if } (s,a) = (s_i, a_i) = (s'_i, \pi(s'_i)), \\ 0, & \text{otherwise.} \end{cases}$$

We then construct matrices $\Phi^{(k)}$ and vector $r^{(k)}$ for each component $\mathcal{C}_k$ by stacking the row-vectors of those i whose transitions are in component $\mathcal{C}_k$, to give design matrices $\Phi^{(k)} \in \mathbb{R}^{N_k \times |\mathcal{I}^{(k)}|}$, where $N_k \geq 0$ is the number of edges of component k, and corresponding reward vectors $r^{(k)} \in \mathbb{R}^{N_k}$. Let $Q^{(k)} \in \mathbb{R}^{|\mathcal{I}^{(k)}|}$ be the vectors of $Q(s, a)$ values stacked in the same order. Then, we are able to write the objective in Eq. (5) as

$$\mathcal{L}(Q) = \sum_k (\Phi^{(k)} Q^{(k)} - r^{(k)})^2 + \lambda (Q^{(k)} - \mu^{(k)})^2.$$

where $\mu^{(k)} = \mu_Q \mathbf{1}^{(k)} = (\mu_Q, \cdots, \mu_Q)^\top \in \mathbb{R}^{|\mathcal{I}^{(k)}|}$ and $\lambda = \sigma_r^2 / \sigma_Q^2 > 0$. Note that $\mathcal{L}$ decomposes as independent sums across components, so going forward we analyse the objective on a per-component basis, and drop k in the notation for notational convenience.

General minimiser expression. For each component, let $\widehat{Q}$ denote the minimiser of

$$\mathcal{L}_k(Q) = (\Phi Q - r)^2 + \lambda(Q - \mu)^2.$$

As this is a standard quadratic optimisation problem, we can find the minimiser in closed form, which is given by

$$\widehat{Q}_\lambda(\mu) = (\Phi^\top \Phi + \lambda I)^{-1}(\Phi^\top r + \lambda\mu). \tag{6}$$

Supported components As discussed above, we can treat connected components as independent MRPs. In cyclic components, each node has exactly one outgoing node, so by the deterministic nature of the transitions we have complete knowledge of the transition dynamics in this MRP. If we denote the transition matrix for such a component as P, with entries $P_{ij} = 1$ if node i contains an edge to node j and $P_{ij} = 0$ otherwise, then $\Phi = I - \gamma P$, which is a square invertible matrix, and the ground truth values (which coincide with the minimisation objective when $\lambda = 0$) are given by $Q_0 = \Phi^{-1} r$, and $\Phi Q_0 = r$. Then we can rewrite

$$\widehat{Q}_\lambda(\mu) = Q_0 - \lambda(\Phi^\top \Phi + \lambda I)^{-1}(\Phi r - \mu).$$

It is now straightforward to take the limit of $\lambda \to 0^+$, which is Q_0, independent of μ_Q, and we observe that the resulting minimisers are therefore a perturbation of Q_0. $\widehat{Q}_\lambda(\mu)$ is continuous in λ. So, with the notation $v_{(s,a)}$ representing the element of vector v corresponding to state-action (s, a), the following holds true:

Lemma 3. *For any supported component $\mathcal{C}_k$ and any $\varepsilon > 0$, there exists a $\lambda_c^{(k)}(\varepsilon, \mu)$ such that for all $0 < \lambda < \lambda_c^{(k)}$, $\left|\widehat{Q}_\lambda(\mu)_{(s,a)} - {Q_0^{(k)}}_{(s,a)}\right| < \varepsilon$ for all (s, a) in $\mathcal{C}_k$, where ${Q_0^{(k)}}_{(s,a)}$ is the true action-value of (s, a).*

In other words, on connected components the learned minimisers approximate all ground-truth values arbitrarily well if $\lambda = \sigma_r^2/\sigma_Q^2$ is sufficiently small. Corollary 2 can then immediately be established by choosing $\sigma_0^2 = \frac{\min_k \lambda_c^{(k)}(\varepsilon,\mu)}{\sigma_r^2}$ so that the inequality holds across all supported components.

Unsupported components. For unsupported components, we also analyse the limit of small λ, but need to account for the fact that Φ is not invertible and therefore the unregularised solution is not tied to the ground-truth values in the same way that it is for supported components. Starting from Eq. (6), we consider

$$\lim_{\lambda\to 0^+} \widehat{Q}_\lambda(\mu) = \lim_{\lambda\to 0^+} (\Phi^\top \Phi + \lambda I)^{-1}\Phi^\top r + \lim_{\lambda\to 0^+} \lambda(\Phi^\top \Phi + \lambda I)^{-1}\mu.$$

We can use standard arguments to express the first term in terms of the Moore-Penrose pseudoinverse Φ^+ as $\lim_{\lambda\to 0^+}(\Phi^\top \Phi + \lambda I)^{-1}\Phi^\top r = \Phi^+ r$, which is the solution with minimal L2 norm to the unregularised, overdetermined system $\Phi Q = r$ (Tikhonov & Arsenin, 1977).

Focusing then on the second term, we start by noticing that for an unsupported component with n nodes, we have $n - 1$ edges and $\Phi \in \mathbb{R}^{(n-1)\times n}$. Each edge induces an independent linear constraint on the system, so Φ must have kernel of dimension 1. We start by determining what the nullspace direction is. Let $z \in \ker(\Phi)$, then, by definition $\Phi z = 0$ and since the space is non-trivial, $z \neq 0$. Without loss of generality, set the unique element i_s corresponding to the sink $z_{i_s} = 1$. Then notice that for all nodes a distance 1 away from the sink, labelled with j, the structure of Φ implies $z_j - \gamma z_{i_s} = 0$, so $z_j = \gamma z_{i_s}$. In general, for any node k with successor l we have $z_k = \gamma z_l$. Therefore, by induction, we can conclude that a node k that is a distance $d(k)$ from the sink will have corresponding $z_k = \gamma^{d(k)}$. Thus,

$$\ker \Phi = \text{span}\{z\}, \text{ where } z \in \mathbb{R}^n \text{ and } z_i := \gamma^{d(i)}.$$

Notice in particular that z is element-wise positive. Next, decompose μ into orthogonal nullspace and row-space components so that $\mu = \mu_\mathcal{N} + \mu_\mathcal{R}, \mu_\mathcal{N} \in \ker \Phi, \mu_\mathcal{R} \in \text{Im}(\Phi^T)$ and define $B_\lambda :=$

$\Phi^\top\Phi + \lambda I$. Consider diagonalising the symmetric matrix $\Phi^\top\Phi$. By the definition of $\mu_\mathcal{N}$ and $\mu_\mathcal{R}$, $\mu_\mathcal{N} = v_0$ is an eigenvector with eigenvalue 0, whereas $\mu_\mathcal{R}$ can be decomposed into a weighted sum of eigenvectors v_i with eigenvalues $\rho_i > 0$. Then the eigenvalues of λB_λ^{-1} are

$$\rho_i^{(\lambda B_\lambda^{-1})} = \frac{\lambda}{\rho_i + \lambda} \to \begin{cases} 1 \text{ if } v_i \in \ker \Phi \\ 0 \text{ otherwise,} \end{cases}$$

as $\lambda \to 0^+$ so we conclude that

$$\lim_{\lambda \to 0^+} \lambda(\Phi^\top\Phi + \lambda I)^{-1}\mu = \mu_\mathcal{N}$$

and

$$\lim_{\lambda \to 0^+} \widehat{Q}_\lambda(\mu) = \Phi^+ r + \mu_\mathcal{N}.$$

As we previously determined the nullspace of Φ as the span of z, we can express the projection $\mu_\mathcal{N}$ of μ onto it explicitly as

$$\mu_\mathcal{N} = \frac{z^\top \mu}{z^2} z = \mu_Q \frac{z^\top \mathbf{1}}{z^2} z,$$

with each component is strictly increasing in μ_Q, with

$$(\mu_\mathcal{N})_i = \mu_Q \frac{\sum_j \gamma^{d(j)}}{\sum_k \gamma^{2d(k)}} \gamma^{d(i)}.$$

Thus, we have that on unsupported components, the limit

$$\widehat{Q}_{\text{unsup}}(\mu) := \lim_{\lambda \to 0^+} \widehat{Q}_\lambda(\mu) = \Phi^+ r + \mu_\mathcal{N}$$

converging for each element i of the vector $\widehat{Q}$ to, in terms of μ_Q,

$$\widehat{Q}_{\text{unsup}}(\mu_Q)_i = (\Phi^+ r)_i + \mu_Q C_i,\ C_i := \frac{\sum_j \gamma^{d(j)}}{\sum_k \gamma^{2d(k)}} \gamma^{d(i)} > 0.$$

Formally, we can make an analogous statement to Lemma 3.

Lemma 4. *For any unsupported component $\mathcal{C}_k$ and any $\varepsilon > 0$, there exists a $\lambda_u^{(k)}(\varepsilon, \mu)$ such that for all $0 < \lambda < \lambda_u^{(k)}$, $\left|\widehat{Q}_\lambda^{(k)}(\mu)_i - \left((\Phi^{(k)^+} r^{(k)})_i + \mu_Q C_i\right)\right| < \varepsilon$ for all $i \in \mathcal{I}_k$.*

Having established that supported and unsupported components tend to distinct values and that the limiting solution for each element is strictly increasing with μ_Q, it is straightforward to show that for small enough λ, we can pick μ_Q sufficiently negative (conservative) so that supported components tend to lower values than unsupported ones.

Choosing μ_Q to separate components type. Choose any $\delta \geq 0$ and define

$$\Delta_{ij} := (Q_0^{(k_i)})_i - (\Phi^{(k_j)^+} r^{(k_j)}) - C_j^{(k_j)} \mu_Q$$

for any supported node i in component k_i and any unsupported node j in component k_j. Then, if we choose any $\mu_Q < \mu_0$ with

$$\mu_0 := \min_{i \in X_{\text{sup}}, j \in X_{\text{unsup}}} \frac{\delta + (\Phi^{(k_j)^+} r^{(k_j)})_j - (Q_0^{(k_i)})_i}{C_j}$$

we obtain that $\Delta_{ij} > \delta$ for all $i \in X_{\text{sup}}, j \in X_{\text{unsup}}$. The definition of Δ_{ij} is such that

$$\Delta_{ij} = \lim_{\lambda \to 0^+} (\widehat{Q}_\lambda^{(k_i)}(\mu))_i - \lim_{\lambda \to 0^+} (\widehat{Q}_\lambda^{(k_j)}(\mu))_j > \delta \tag{7}$$

for any supported component $\mathcal{C}_{k_i}$ and unsupported component $\mathcal{C}_{k_j}$. We have just shown that with an appropriate choice of $\mu_Q < \mu_0$, this quantity can always made to be greater than any arbitrary $\delta > 0$. Equation (7) implies that for all i and j, there exists some $\lambda_0 > 0$ such that for all $0 < \lambda < \lambda_0$

$$(\widehat{Q}_\lambda^{(k_i)}(\mu_Q))_i - (\widehat{Q}_\lambda^{(k_j)}(\mu_Q))_j > 0,$$

thus proving Theorem 1.

Global choice of prior. We remark that the result in Theorem 1 that we just proved here is valid for a given policy, but can be easily extended to account for the fact that in practice a policy changes during training. Note that while the specific G_π structure and Φ will change with π. However, we can safely adapt the choice of μ_0 to account for this, so that

$$\mu_0 := \min_\pi \min_{i \in X_{\text{sup}}, j \in X_{\text{unsup}}} \frac{\delta + (\Phi^{(k_j)^+} r^{(k_j)})_j - (Q_0^{(k_i)})_i}{C_j}$$

then ensures that even with changing policies the same constant prior mean μ_Q (and corresponding derived λ) can be used to give a value gap between supported and unsupported components for any policy.

$\gamma = 0$ case. We treat the case $\gamma = 0$ separately. With such a γ, the state-action's value is exactly the immediate reward obtained, and the objective in Eq. (5) simplifies to

$$\sum_{(s_i, a_i, r_i, s_i') \in \mathcal{D}} (Q(s_i, a_i) - r_i)^2 + \sum_{s \in \mathcal{S}, a \in \mathcal{A}} \frac{\sigma_r^2}{\sigma_Q^2} (Q(s, a) - \mu_Q)^2.$$

The policy dependence of the minimisers drops out, resulting in a decoupled quadratic in each (s, a), so the minimisers, i.e. the learned posterior means, are equal to

$$\widehat{Q}(s, a) = \begin{cases} \dfrac{Q(s, a) + \lambda \mu_Q}{1 + \lambda}, & \text{if } (s, a, r, s') \in \mathcal{D}, \\ \mu_Q & \text{otherwise.} \end{cases} \qquad \lambda = \frac{\sigma_r^2}{\sigma_Q^2}.$$

Notice that we have used the fact that for deterministic environments with $\gamma = 0$ we necessarily have $Q = r$.

We note the following two consequences:

- For supported state-actions, there is a constant offset effect from the prior, so greedy action-selection using $\widehat{Q}$ is identical to using the ground-truth action-values, and therefore optimal within this set.
- a necessary condition to guarantee that the supported actions be selected by greedy action selection on $\widehat{Q}$ is that $\mu_Q < Q(s, a)$ for all $(s, a) \in X_{\text{unsup}}$. This needs to be true at any state that we may visit under π, so any state in $\text{supp}(\eta^\pi)$. Therefore, Definition 1 represents a necessary condition on the value prior to induce the selection of supported policies.

C Reward Preprocessing and Conservative Priors.

We show here that the reward preprocessing step in Section 4 results in a globally conservative prior for any $\mu_Q < 0$ under the assumption that the dataset $\mathcal{D}$ is representative of the smallest obtainable reward, i.e. the expected immediate reward R is bounded below by the smallest $r \in \mathcal{D}$ observed, which we denote by $r_{\min}$.

We work in the MDP which is equivalent to the original but with shifted rewards, $r' = r - r_{\min}$. For any policy π, the corresponding value function in the shifted MDP is $V'^\pi(s) = \mathbb{E}\left[\sum_{t=0}^\infty r_t' \gamma^t\right] = \mathbb{E}\left[\sum_{t=0}^\infty (r_t - r_{\min})\gamma^t\right] \geq 0$ since each term is non-negative. The corresponding action-values satisfy

$$Q'^\pi(s, a) = R'(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}\left[V^\pi(s')\right] = (R(s, a) - r_{\min}) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}\left[V^\pi(s')\right] \geq 0$$

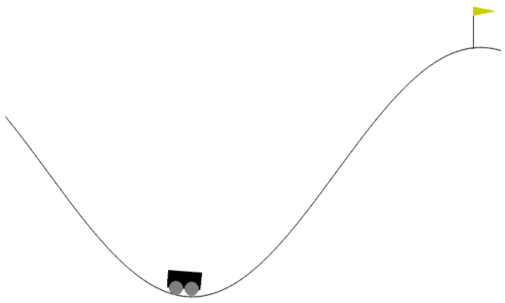


Figure 5: Mountain car environment. The agent's goal is to push the car starting from the bottom of the valley past the yellow flag on the right.

as we have a sum of two non-zero terms.

Thus, in the reward-shifted MDP the ground-truth action-values are non-negative for any policy π. Consequently, any constant prior mean $\mu_Q < 0$ defines a conservative prior in the sense of Definition 1, since $\mu_Q < Q'^{\pi}(s,a)$ holds for all (s,a) and all π. We could alternatively enforce conservatism in the original MDP by choosing $\mu_Q < \frac{r_{\min}}{1-\gamma}$, but this may lead to numerical instability when $\gamma \approx 1$, due to the resulting in large absolute values of μ_Q. The reward-shifting approach avoids this issue while still guaranteeing a globally conservative prior.

D Mountain Car Details

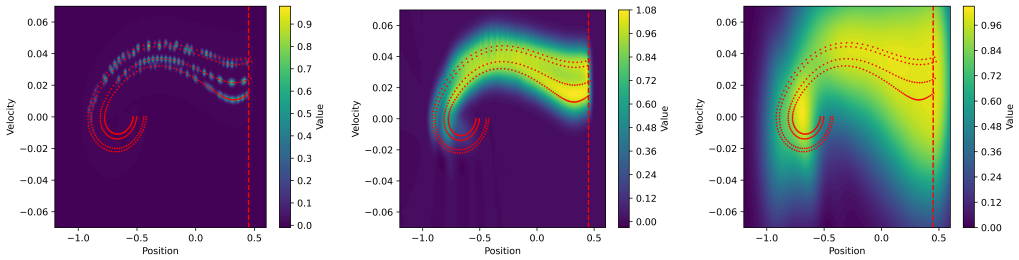
The mountain car (Moore, 1990) tackled in Section 5.2 is a deterministic classic control environment. The state-space is two dimensional (position and velocity) and the action space is continuous and one-dimensional $[-1, 1]$. The objective is to strategically apply impulses to a car so that it can escape the valley it starts and build up speed to reach the goal at the top of the mountain, visualised in Fig. 5. If the agent reaches the goal (which is to get the position coordinate to cross past the 0.45 threshold) it receives a reward of 1 and the episode terminates. Otherwise, the reward at each timestep is 0.

The exact-inference CVP agent we train in Section 5.2 has an actor architecture of two hidden layers with 256 neurons each. We train it for 500 gradient steps with Adam optimiser and learning rate of $1e-3$. In the gradient computation for the optimisation step, in analogy to target networks in standard RL (Mnih et al., 2015; Haarnoja et al., 2018a; Fujimoto et al., 2018), we do not pass gradients through the terms used to evaluate action-values at the next-states $Q^{\pi}(s', \pi(s'))$, which we treat as stop-gradients for this computation. We use a prior on the action-value function with zero mean, unit variance and RBF kernels, with position, velocity and action lengthscales chosen by hand to be 0.02, 0.008 and 0.5 respectively. We give more insight into how the state-space lengthscale was chosen at the end of this section. Due to the deterministic nature of rewards, we choose a small observation noise variance (the ε in Eq. (1)) of 1e-5.

The expert agent from which the dataset is gathered is taken from the publicly available model at this url: `https://huggingface.co/sb3/sac-MountainCarContinuous-v0/tree/main`.

The TD3 agent is trained for 50,000 training steps using full-batch gradient descent with the same actor network architecture as the CVP agent, an equivalent critic network architecture and default TD3-specific hyperparameters (target critic soft update coefficient 0.005, policy noise 0.2 clipped at 0.5 and delayed policy updates every 2 critic updates) and ADAM optimisers with learning rate of 3e-4 for both actor and critic. We find, as expected from naive offline application of TD3, that both actor and critic training losses diverge due to the absence of regularisation.

Choosing the lengthscale hyperparameter. The key hyperparameter for inference with an RBF prior is the lengthscale, which determines how quickly prior covariance decays with distance in



(a) $\kappa = 0.2$ (b) $\kappa = 1$ (c) $\kappa = 6$

Figure 6: Posterior value visualisation for different multiples of the base $(0.02, 0.008)$ lengthscale used in the main text. The red dots correspond to the observed states in the dataset and the red dotted line corresponds to the goal threshold.

state space. The state-space lengthscales reported in the main text, $(0.02, 0.008)$ (corresponding to the position and velocity dimensions respectively), was chosen by inspecting the data in state space, judging how far support should extend in the vertical and horizontal directions, and then making minor adjustments after running the experiment. One advantage of this parameter is its interpretability: in simple and easily visualised environments such as the one considered here, it is often visually apparent when the lengthscale is too small or too large. On the other hand, a broader challenge for Bayesian methods is specifying useful priors in settings where such quantities cannot be set by hand so easily.

In the CVP setting, a lengthscale that is too large causes the agent to treat distant states as in-distribution, and hence to regard OOD state-actions as supported. A lengthscale that is too small prevents the agent from leveraging useful information from neighbouring states, effectively treating each state as independent. Both failure modes degrade performance. In our experiments, however, tuning the lengthscale was straightforward, and inference behaved as expected as this hyperparameter varied.

In Fig. 6, we visualise how posterior value propagates from the observed data as we scale the state-space lengthscales used in the main text, i.e., lengthscales of the form $(0.02\kappa, 0.008\kappa)$. Throughout our experiments, we fixed the action lengthscale to 0.5, informed by the action range $[-1, 1]$, and did not tune it further. We report in Fig. 6 the values of κ for which performance began to deteriorate. For the lengthscale used in the main text, corresponding to $\kappa = 1$, the agent successfully reaches the goal from all 15 starting states. Decreasing κ down to 0.25 led to similar performance, while the first drop was observed at $\kappa = 0.2$, where the agent reached the goal from 11 of the 15 starting states. In Fig. 6a, we see that the posterior value of most states is overwhelmingly close to 0, with only very small neighbourhoods around the data having their beliefs updated by the observations. Increasing κ up to 5 also did not affect performance, while at $\kappa = 6$ the agent reached the goal from 13 of the 15 starting states. As expected, posterior value propagation then extends excessively beyond the data distribution, incorrectly treating distant parts of state space as supported, as can be observed in Fig. 6c. Fig. 6b revisits the posterior value recovered in the main text, which can be seen to strike a reasonable balance between these two regimes.

Overall, performance remained robust for $0.25 < \kappa < 5$, spanning slightly more than one order of magnitude. We note, however, that the horizontal-to-vertical lengthscale ratio of $0.02 : 0.008$ used in the main paper may not be optimal, and that a different ratio could be robust over an even wider range of κ.

E Neural Network CVP Loss

Here we provide full derivations to show that the loss in Eq. (4) is of the appropriate form for $\hat{Q}$ to regress to the posterior mean for independent state-action kernels as introduced in Section 5.1.

We start by considering a finite state-action space, with state space of size $|\mathcal{S}|$ and action space of size $|\mathcal{A}|$, and independent Gaussian priors over Q with mean μ_Q and state-action dependent variance $\sigma_Q^2(s,a)$. With this setup, the following claim holds:

Claim 5. *Using the definition*

$$L_B(Q) = \sum_{(s,a,r,s')\in\mathcal{D}} (Q(s,a) - r - \gamma Q(s', \pi(s'))^2,$$

the loss objective

$$L(Q) = L_B(Q) + \alpha \mathbb{E}_{(s,a)\sim p_{\text{pseudo}}(s,a)} (Q(s,a) - \mu_Q)^2 \tag{8}$$

with $p_{\text{pseudo}}(s,a) = \frac{1}{\alpha}\frac{\sigma_r^2}{\sigma_Q^2(s,a)}$ *is minimised when* $Q(s,a)$ *are the posterior mean.*

Proof. The proof starts by considering the form of the posterior as in Appendix B, but with a varying prior variance. Recall that the statistical model from Section 4 with observation noise variance σ_r^2 implies a posterior of the form

$$\begin{aligned}
&p(Q|\mathcal{D}) \propto p(Q)p(\mathcal{D}|Q) \\
&\propto \exp\left\{ \sum_{s\in\mathcal{S},a\in\mathcal{A}} -\frac{(Q(s,a)-\mu_Q)^2}{2\sigma_Q^2(s,a)} \right\} \exp\left\{ -\frac{1}{2\sigma_r^2} \sum_{(s_i,a_i,r_i,s_i')\in\mathcal{D}} (Q(s_i,a_i) - r_i - \gamma Q(s_i', \pi(s_i'))^2 \right\} \\
&\propto \exp\left\{ -\frac{1}{2} \sum_{s\in\mathcal{S},a\in\mathcal{A}} \frac{(Q(s,a)-\mu_Q)^2}{\sigma_Q^2(s,a)} - \frac{1}{2\sigma_r^2} \sum_{(s_i,a_i,r_i,s_i')\in\mathcal{D}} (Q(s_i,a_i) - r_i - \gamma Q(s_i', \pi(s_i'))^2 \right\} \\
&\propto \exp\left\{ -\frac{1}{2\sigma_r^2} \left(\sum_{s\in\mathcal{S},a\in\mathcal{A}} \sigma_r^2 \frac{(Q(s,a)-\mu_Q)^2}{\sigma_Q^2(s,a)} + \sum_{(s_i,a_i,r_i,s_i')\in\mathcal{D}} (Q(s_i,a_i) - r_i - \gamma Q(s_i', \pi(s_i'))^2 \right) \right\}.
\end{aligned}$$

We observe that maximising $p(Q|\mathcal{D})$ is equivalent to minimising $-\log p(Q|\mathcal{D})$, so the MAP estimate of Q will minimise

$$\sum_{(s_i,a_i,r_i,s_i')\in\mathcal{D}} (Q(s_i,a_i) - r_i - \gamma Q(s_i', \pi(s_i'))^2 + \sum_{s\in\mathcal{S},a\in\mathcal{A}} \frac{\sigma_r^2}{\sigma_Q^2(s,a)} (Q(s,a)-\mu_Q)^2.$$

We notice the first term is exactly L_B. Next, we rewrite the sum as an expectation over p_{pseudo} as given in Claim 5 so that the above becomes

$$\begin{aligned}
&L_B(Q) + \sum_{s\in\mathcal{S},a\in\mathcal{A}} p_{\text{pseudo}}(s,a) \underbrace{\frac{1}{p_{\text{pseudo}}(s,a)} \frac{\sigma_r^2}{\sigma_Q^2(s,a)}}_{\alpha} (Q(s,a)-\mu_Q)^2. \\
&= L_B(Q) + \alpha \mathbb{E}_{(s,a)\sim p_{\text{pseudo}}(s,a)} (Q(s,a)-\mu_Q)^2
\end{aligned}$$

after substituting α as defined in the claim. We have thus shown that the MAP Q-values will be recovered by minimising Eq. (8), which is quadratic in Q. The last step to prove Claim 5 is to notice that for Gaussian variables MAP and posterior mean coincide, since the mode of a Gaussian is equal to its mean. □

Having established that in the tabular case with finite state-action space the loss in Eq. (8) provably results in posterior mean Q-values, we consider increasing the state-action space to an intractably large, approximately continuous, number of state. Standard arguments (Mnih et al., 2015) suggest replacing the tabular $Q(s,a)$ with a parametrised $\hat{Q}_\theta(s,a)$, with training objective given by the same loss but replacing $Q(s,a)$ with $\hat{Q}_\theta(s,a)$, leading directly to the loss in Eq. (4).

Notice that the relationship

$$\sigma_Q^2(s,a) = \frac{\sigma_r^2}{\alpha} \frac{1}{p_{\text{pseudo}}(s,a)} \tag{9}$$

implies that the regularisation strength constant α can be interpreted as being inversely proportional to the prior variance, as claimed in Section 6.1.

Finally, we comment on the specific choice of σ_Q^2 and, correspondingly, p_{pseudo}. A natural baseline, as done when applying Alg. 1 in practice, is to take a constant $\sigma_Q^2(s,a) = \sigma_Q^2$. However, this would entail using a uniform $p_{\text{pseudo}}(s,a) = \frac{1}{|\mathcal{S}||\mathcal{A}|}$, which is problematic for large or high-dimensional state-action spaces. As a consequence, as described in Section 6.1, we depart from using such a uniform p_{pseudo} in our experiments. Despite this modification, a Bayesian interpretation remains valid, but with a state-dependent prior variance given by Eq. (9). In the context of our decision-making task, this corresponds to setting narrower priors, thus stronger regularisation, on those state-actions that are most relevant to decision-making.

F Continuous Maze Environment

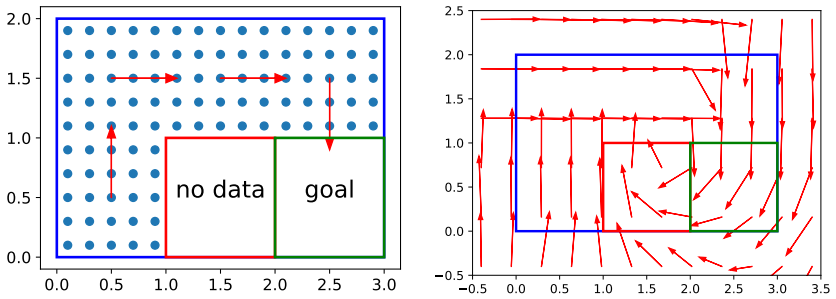
In this section, we introduce a structured toy environment that yields easily interpretable qualitative behaviour. We then use it to compare the exact-inference implementation (Algorithm 1) and the deep-learning critic variant from Section 6.1 (Algorithm 2) against two baseline online algorithms: TD3, which learns a deterministic policy (Fujimoto et al., 2018), and SAC, which learns a stochastic policy (Haarnoja et al., 2018a;b). We also consider their offline CVP-based counterparts, CVP-TD3 and CVP-SAC. Finally, at the end of this section, we present a modified-data example, distinct from the rest of the appendix, in which the agent must avoid suboptimal actions present in the dataset, as mentioned at the end of Section 1.

Continuous maze We consider an MDP in a continuous-space maze-like environment where only a part of the state-action space is adequately covered by the dataset. The state-space is $\mathcal{S} = \mathbb{R}^2$ and the action space is $\mathcal{A} = [-1, 1]^2$, visualised in Fig. 7a. The agent deterministically moves in state space by adding the action vector to its current state vector. It receives a reward of 1 upon reaching the goal region $\{(x, y) \in \mathbb{R}^2 : 2 < x < 3, 0 < y < 1\}$ and 0 otherwise, with episodes terminating at the goal.

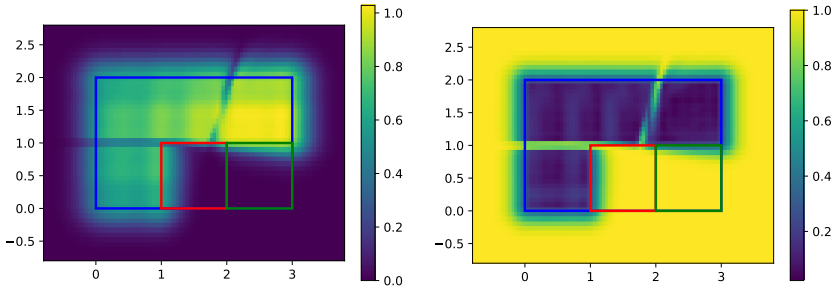
The static dataset we consider covers a limited part of the environment's state-space, and is formed by individual (s, a, r, s') transitions rather than full episodic traces. The datasets contains transitions starting from 100 uniformly spaced states in the region $\{(x, y) \in \mathbb{R}^2 : 2 < x < 3, 0 < y < 1)\} \cup \{(x, y) \in \mathbb{R}^2 : 0 < x < 3, 1 < y < 2)\}$. All actions in the dataset are in either the x or y directions and have size 0.6, with the action observed at each state being the one that makes progress towards the goal while remaining in the in-support region (see Fig. 7a).

Exact inference We first apply Alg. 1 to this toy dataset, with RBF kernels with length-scale 0.25 for both state and actions and an actor parametrised by an MLP with two hidden layers of 256 neurons trained for 1000 steps. The RBF state-space length-scale is chosen to match the step-size. In Fig 7b, we observe that the resulting policy reaches the goal while remaining in the in-distribution data regions. The values learned are displayed in Fig. 7c, where we observe that only the in-distribution regions are assigned high values, which gradually decrease with number of steps required to reach the goal (as expected due to the discount factor $\gamma = 0.9$). Similarly, the value uncertainty displayed in Fig. 7d is high in the OOD regions and low where the agent can confidently reach the goal while remaining in-distribution. The thin lines of slightly decreased expected value in some in-distribution regions of Fig. 7c are due to the continuous nature of the actor. In these regions that bridge a sharp policy change, continuity implies that the actor must employ actions that are slightly different to those on either side of the transition region, thus resulting in a lower covariance with the neighbouring states and lower posterior value.

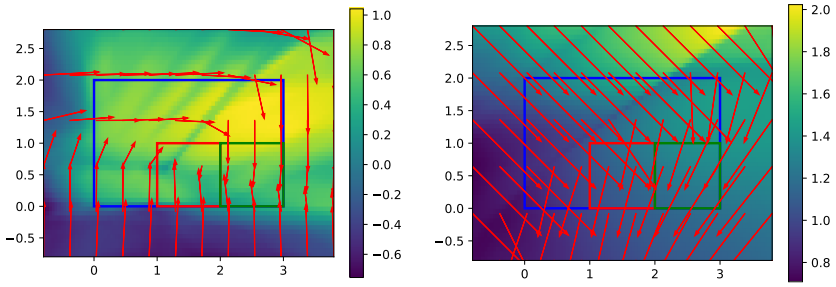
CVP-SAC We start by considering CVP with SAC as base algorithm. For consistency with the GP implementation, we use prior mean $\mu_Q = 0$, and employ $\alpha = 0.001$ for both methods. We implement our algorithm by modifying base implementations provided by Tarasov et al. (2022).



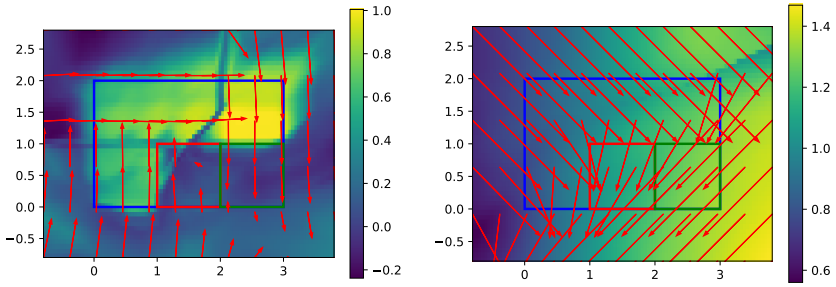
(a) Dataset visualisation (b) Learned policy



(c) Posterior expected value (d) Posterior value standard deviation



(e) CVP-SAC learned policy and MAP value (f) Naive SAC learned policy and MAP value



(g) CVP-TD3 learned policy and MAP value (h) Naive TD3 learned policy and MAP value

Figure 7: Toy environment for navigation task. The agent receives a reward of 1 for transitioning into the goal region (where the episode terminates) and 0 otherwise. The dataset consists of steps of size 0.6 in the cardinal direction that leads towards the goal while remaining in the supported (blue) region. Learned policies and value functions are visualised as arrows and colour-maps respectively. The length of the arrows corresponds to the size of the step taken.

Visualisations of the learned policy and values from applying Alg. 2 are shown in Fig. 7e. The scalable version produces a policy that avoids the no-data region while reaching the goal, with a value estimate consistent with the exact-inference case in the in-distribution region. Since the pseudo-data distribution regularises OOD actions rather than states, OOD states don't receive a signal to align with the prior. In contrast, Fig. 7f shows that applying the base SAC algorithm naively to the offline data (equivalent to setting $\alpha = 0$) leads to poor policies that do not remain in the support of the data and wander into regions of state-space that the agent has no knowledge of, leading to unsatisfactory offline policies.

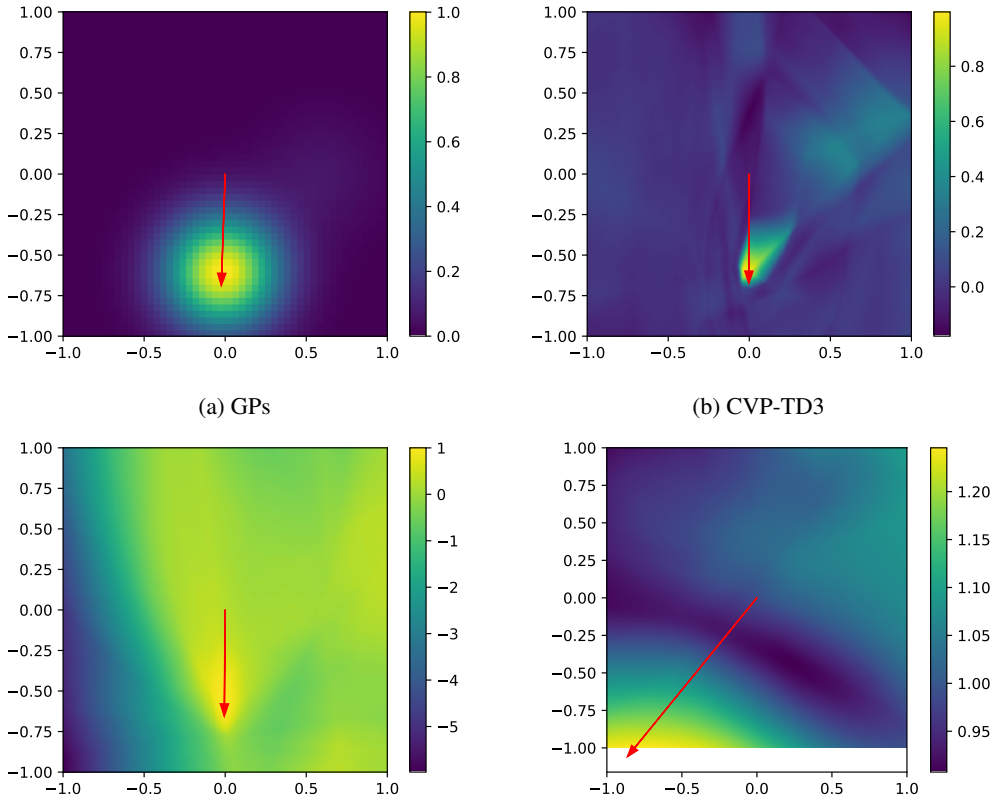
TD3-SAC. Next, we report analogous results for the the TD3-based CVP-TD3. In Fig. 7g, we observe how the CVP-TD3 policy successfully solves the offline RL task and learns similar values to those learned with exact inference CVP in the in-distribution region, whereas in Fig. 7h we see that the unregularised naive application of TD3 fails in this instance. This closely mirrors the behaviour shown with SAC.

Visualising CVP in action space. In Fig.8, we observe the effect of the conservative prior directly in action-space, where we visualise the action-values after training in the *action* space at state $(2.5, 1.1)$ just above the goal region. There is an action $(0, -0.6)$ in the dataset at this state that leads directly to the goal, which we expect the agent to follow. For all three methods, the neighbourhood around the observed action that leads to the goal is regularised as desired, and the algorithms correctly learn to choose the action similar to the one in the dataset that leads to the goal. As can be seen in Fig. 8a, due to the exact nature of the inference employed, the GPs smoothly regularise the whole action space around the observed action. Similarly, the actions that maximise value in Figs. 8b and 8c correctly correspond to following the observed action leading to the goal. While the extrapolation in regions far from the action the agent is considering can vary significantly across methods, this is because our approximate methods must focus on regularising those actions most towards the actor's decision making, and we do observe that values learned close to the actor's action are similar across methods, therefore ensuring that the appropriate action is learned. In contrast, Fig. 8d shows how a naive application of TD3 (setting $\alpha = 0$ in TD3-CVP) causes the actor to choose an unsupported action, where the increase in value is not supported by observed actions but rather entirely caused by unwarranted extrapolation.

Exact inference with suboptimal transitions Finally, we present results on a slightly different dataset. For the dataset considered so far, it is desirable for the agent to follow the observed actions in the dataset for the agents' trajectory to remain in-distribution. Here, we instead consider the behaviour of the exact inference scheme on a non-episodic dataset where certain actions lead the agent to OOD states that are not found anywhere else in the dataset. This example will also highlight the distinction between in-distribution and supported actions introduced in 3: although these actions appear in the dataset (and thus would not be penalised by a behaviour-regularised algorithm), their long-term values cannot be inferred from the available transitions, so they remain unsupported and a principled offline RL algorithm should avoid them.

The environment considered here is identical to the one introduced at the start of the section. The dataset is also similar, but the difference is that at each observed states, 4 actions in each of the cardinal directions are observed instead of just 1. That is, actions $[0.6, 0]$, $[0, 0.6]$, $[-0.6, 0]$ and $[0, -0.6]$ are observed at each state, thus resulting in a dataset consisting of 400 transitions overall. This is visualised in Fig. 9a. Notice this dataset is also not episodic, as it is only composed of individual transitions which the agent needs to learn to combine to produce high-reward trajectories, while avoiding suboptimal actions. We visualise this alternative dataset in Fig 9a.

We apply Alg. 1 to this dataset, with identical hyperparameters and architecture as at the start of this section. In Fig. 9b, we show the learned policy after 1000 steps of training. As expected, the resulting policy reaches the goal state and remains in the in-distribution region, avoiding the actions that lead it to OOD states. The value function displayed in Fig. 9c displays overall very similar behaviour to that in Fig. 7c. There is a small difference in the region just to the left of $x = 2, 0 < y < 1$, where



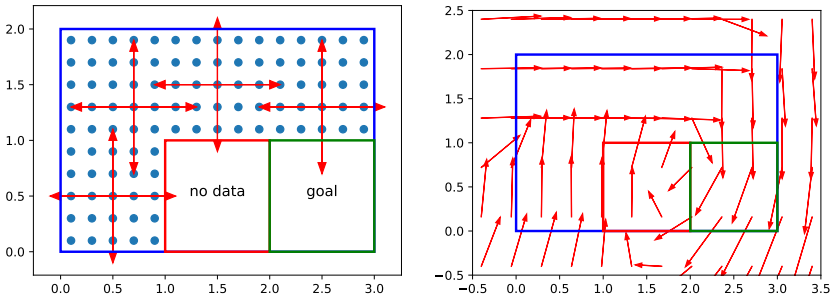
(a) GPs

(b) CVP-TD3

(c) CVP-SAC

(d) Naive TD3

Figure 8: Learned action-values at state $(2.5, 1.1)$, located just above the goal. The red arrow shows the learned action. The dataset contains an action $(0, -0.6)$ leading to the goal at this state.



(a) Dataset visualisation

(b) Learned policy

(c) Posterior expected value

(d) Posterior value standard deviation

Figure 9: Toy environment for navigation task with suboptimal actions in the dataset. The agent receives a reward of 1 for transitioning into the goal region (where the episode terminates) and 0 otherwise. The dataset consists of steps of size 0.6 in all four cardinal directions at all the uniformly spaced states (blue dots) in the blue region. Thus, some actions can lead the agent towards parts of state-space from which no further data is available. Learned policies and value functions are visualised as arrows and colour-naps respectively. The length of the arrows corresponds to the size of the step taken.

the value of the states of the small slice of OOD states is higher than expected. This is due to the smoothness assumption encoded by the RBF prior. This is in contrast to the discontinuous reward definition of the environment, and the artifact in the figure can be interpreted as a consequence of prior misspecification. For example, the value inferred at state $(2+\varepsilon, 1.3)$ (for a small $\varepsilon > 0$) for the action going down is approximately equal to 1, i.e. the model infers that with high probability

$$Q^\pi((2+\varepsilon, 1.3), (0, -0.6)) \approx 1,$$

since this transition directly reaches the terminal state and the terminal reward is observed. In contrast, transitions from states near $(2-\varepsilon, 1.3)$ with the same action terminate at the non-terminal state $(2-\varepsilon, 0.7)$, with reward $r = 0$. From the model's point of view, these observations therefore imply

$$Q^\pi((2-\varepsilon, 1.3), (0, -0.6)) \approx V^\pi(2-\varepsilon, 0.7)$$

with high probability Finally, the RBF prior couples nearby state-action pairs, so the model also places high probability on

$$Q^\pi((2+\varepsilon, 1.3), (0, -0.6)) \approx Q^\pi((2-\varepsilon, 1.3), (0, -0.6)).$$

Consequently, the model infers that $V^\pi(2-\varepsilon, 0.7)$ should also be close to 1 with high probability. In summary, this behaviour arises because the observed reward changes discontinuously at the $x = 2$ boundary, whereas the GP prior encodes a preference for smooth value functions.

The posterior standard deviation shown in Fig. 9d shows the same general pattern observed in Fig. 7d, with in-distribution states having lower posterior standard deviation and OOD states having higher posterior standard deviation. Interestingly, for those states that visited in the dataset but are left 'hanging', i.e. without a further transition departing from them while still not being terminal, the uncertainty is slightly decreased compared to those parts of state-space that are completely unexplored. This is because a transition from (s, a) to s' correlates the value functions: suppose the observed reward to a hanging state is 0, then the model infers that $Q^\pi(s, a) \approx V^\pi(s')$. If Q and V are independent Gaussians with prior variance σ^2, Gaussian conditioning (Rasmussen et al., 2006) implies that $p(V|Q = V)$ will have smaller variance than the prior $p(V)$. Specifically, the variance is $\sigma^2/2$, so the standard deviation is approximately 0.71σ. This mechanism explains the slight drop in uncertainty observed at these states, and the factor of 0.71 is consistent with what is observed at the hanging states in Fig. 9d. While their marginal posterior standard deviation is slightly decreased, it remains much higher than that of in-distribution states. Finally, we note that while the uncertainty is slightly decreased at hanging states, the posterior expected value, which is the criterion considered by the agent for decision-making, is just as conservative as for the other OOD states.

Overall, this experiment demonstrates that CVP distinguishes between merely observed actions and supported actions whose long-term values are identifiable, thereby recovering the intended conservative behaviour even from disconnected, suboptimal demonstrations and non-episodic datasets.

G D4RL Implementation and Compute Details

We base our implementation of CVP-SAC, which we train on the D4RL benchmarks in Section 6.2 on the SAC backbone of the CQL implementation in the CORL library (Tarasov et al., 2022), and the results presented do not require any additional hyperparameter tuning to the base algorithm beyond changing the optimiser of the entropy coefficient loss to Adam. The hyperparameter relevant to the base SAC algorithm, common to all tasks, are summarised in Table 3.

Next, we address the CVP-specific hyperparameters. We report the prior mean used for each task in Table 4 (either choosing μ_Q or carrying out reward preprocessing as explained in Section 4) and the choice of α for each task. Other methods that carry out critic value regularisation, such as CQL (Kumar et al., 2020) and MCQ (Lyu et al., 2022), both sample 10 actions at which to regularise, so we take the parameter $n = 10$ in Alg. 2 and do not tune it further.

Hyperparameter	**Value**
Batch size	256
Discount factor	0.99
Training steps	1e6
Actor hidden layers	3
Actor hidden size	256
Actor optimiser	Adam
Actor learning rate	3e-5
Critic hidden layers	3
Critic hidden size	256
Critic optimiser	Adam
Critic learning rate	3e-4
Target network update rate	0.005
Entropy coefficient optimiser	SGD
Entropy coefficient learning rate	3e-5

Table 3: Base SAC hyperparameters for the D4RL experiments.

Table 4: CVP hyperparameter settings for the D4RL experiments. RP in the μ_Q column refers to using a mean of 0 after carrying out reward preprocessing as described in Section 4.

Task	α	μ_Q
halfcheetah-random	1e-4	0
hopper-random	1e-4	RP
walker2d-random	1e-2	0
halfcheetah-medium	1e-5	0
hopper-medium	5e-4	0
walker2d-medium	5e-3	RP
halfcheetah-medium-replay	5e-5	0
hopper-medium-replay	5e-4	RP
walker2d-medium-replay	5e-3	RP
halfcheetah-medium-expert	1e-2	0
hopper-medium-expert	5e-3	-10
walker2d-medium-expert	1e-2	0
halfcheetah-expert	1e-2	0
hopper-expert	1e-1	0
walker2d-expert	1e-2	0

Hyperparameter tuning. We started by fixing $\mu_Q = 0$ as default and tuning α per-task. The heuristic described in Appendix H guided the α tuning, giving an indication of whether the next α to try should be greater, if the training was unstable due to regularisation being too small as can be inferred from training diagnostics, or smaller, to check if the regularisation can be decreased further, with possible performance benefits. We then tried RP with these values of α. For some tasks, such as hopper-medium-expert, we observed stable training diagnostics but higher variance in performance, suggesting that further tuning of μ_Q or prior mean selection could be beneficial, and used a different value for μ_Q if it had a beneficial effect on performance.

Compute resources. The experiments were carried out on standard general-purpose workstations provided by the authors' affiliated institution, rather than on dedicated high-performance computing systems. As the networks employed are small (three-layer MLPs with 256 neurons) the experiments were ran on a mixture of GPU and CPU-only machines, depending on availability. A typical runtime

for a full training run would be of approximately 10.5 hours on a machine with a NVIDIA GeForce GTX 1050 Ti GPU and a Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz CPU. This includes extensive evaluation in simulation and logging of training diagnostics during training, and we did not explicitly optimise our code for performance.

H Choosing Regularisation Strength From Training Diagnostics

Here we show how we can empirically guide the choice of regularisation strength, α in Alg. 2, off of training diagnostics - without relying on evaluations on the ground-truth environment. This points to the possibility of automated tuning of this hyperparameter, which we leave for future work.

We take as reference the halfcheetah-medium task. As mentioned in Section 6.3, we notice that reducing regularisation strength can be beneficial up to a certain critical point where training becomes unstable. A weak regularisation leads to inconsistent results, but the maximum achieved returns can be very high, even higher than that achieved by the α that gives the best average across 5 seeds. Meanwhile, regularisation that is too strong slowly degrades performance.

Further investigating the training diagnostics reveals that the failure modes causing this suboptimal performance arise from very different mechanisms. We find that strong regularisation generally leads to stable training (low losses) but poor fitting to the value function whereas small regularisation has greater potential for better performance but at the risk of very high instability. We display relevant losses for sample runs with different α that show this effect in Fig. 10. This strongly suggests that the optimal regularisation strength should be as small as possible while still ensuring stable training and losses. Indeed, there is strong evidence that the optimal value for α might depend on initialisation, as some seeds with the lowest regularisation setting still achieve strong performance with stable convergence, albeit more rarely than when a slightly higher α is employed, and some seeds with higher regularisation can still occasionally exhibit unstable training. This leads to the natural hypothesis that an automated tuning of α would be able to further increase the performance of CVP-SAC beyond what can be achieved by employing a constant α during training, which we leave for future work. We found the pattern of choosing the lowest α before training instability arises as generally leading to good ground-truth evaluation performance in the tasks considered, but further experimentation is required to determine the robustness of this empirical observation.

We show training diagnostics for exemplar runs in Fig. 10. We plot a diverging exemplar run for when α is too small ($\alpha = 10^{-6}$), where training is unstable and we observe training losses and Q-values diverge in a similar way as they would without regularisation. When α is too large ($\alpha = 10^{-2}$), training is stable but the returns achieved are suboptimal. This is in line with the notion that excessive regularisation can harm performance, and the ideal α should be the small but large enough to prevent the training from diverging. We remark that, in this example, this can be identified directly by the training diagnostics, without requiring evaluation on the ground truth environment.

I Additional D4RL Results

We compare here our results to two other model-free offline RL algorithms, that however require significant algorithmic extensions to standard off-policy algorithms achieve a performance similar to that of CVP-SAC, in contrast to those presented in the main Table 1. In this section, we also summarise the nature of these additional components. Overall, CVP-SAC achieves 97% and 102% of the scores attained by MCQ and ATAC, respectively, on the D4RL locomotion tasks.

MCQ comparison. We compare to the scores obtained by the Mildly Conservative Q Learning (MCQ) algorithm (Lyu et al., 2022) in Table 5. Similarly to our method, MCQ includes a L2 regularisation term in the critic loss, but the value against which the critic is regularised is learned during training, requiring the introduction of a generative model for the behaviour policy, thus significantly increasing the complexity of the algorithm. We see that CVP is able to recover 97% of MCQ's average performance without requiring such a generative model.

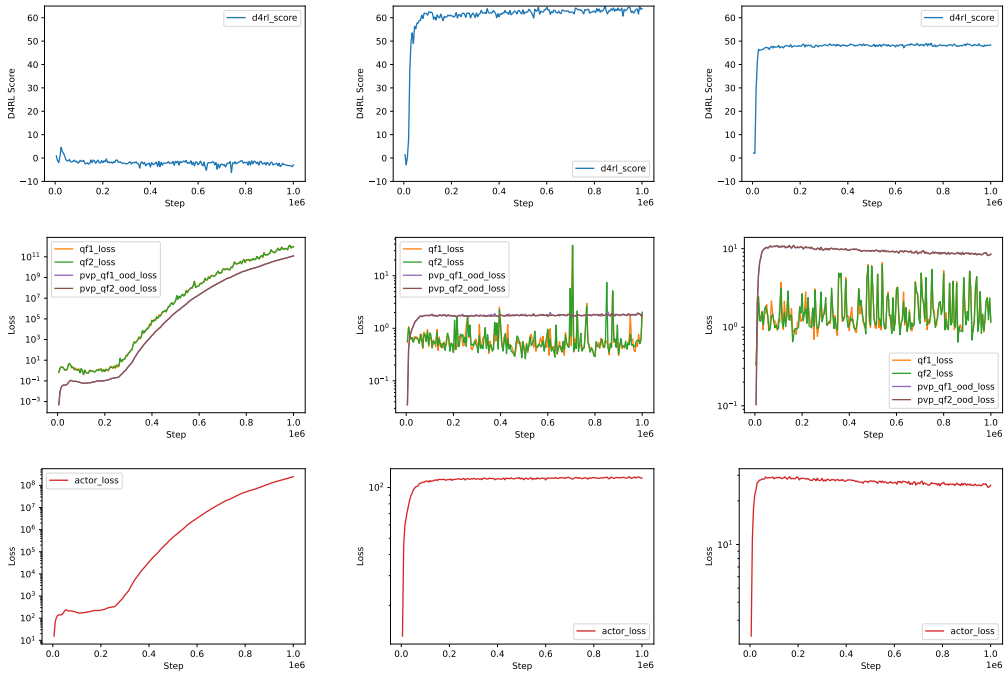


Figure 10: Comparison of training diagnostics and performance for different α values. The left column displays results for $\alpha = 1e-6$, the middle column $\alpha = 1e-4$ and the right column for $\alpha = 1e-2$. `qf_loss` is the standard RL Bellman loss of the 2 critics used whereas `ood_loss` is the CVP loss term.

Table 5: Comparison between MCQ (Lyu et al., 2022) and CVP results on D4RL benchmarks, indicating the percentage of MCQ's score that achieved by CVP-SAC, without having to explicitly model the behaviour policy. The MCQ results are taken from Lyu et al. (2022).

Task	MCQ	CVP	Percentage achieved
halfcheetah-random	28.5	32.5	114.0%
hopper-random	31.8	27.5	86.5%
walker2d-random	17.0	6.9	40.6%
halfcheetah-medium	64.3	66.0	102.6%
hopper-medium	78.4	72.3	92.2%
walker2d-medium	91.0	84.1	92.4%
halfcheetah-medium-replay	56.8	57.2	100.7%
hopper-medium-replay	101.6	100.5	98.9%
walker2d-medium-replay	91.3	81.5	89.3%
halfcheetah-medium-expert	87.5	95.4	109.0%
hopper-medium-expert	111.2	103.8	93.3%
walker2d-medium-expert	114.2	109.3	95.7%
halfcheetah-expert	96.2	95.1	98.9%
hopper-expert	111.4	110.7	99.4%
walker2d-expert	107.2	110.0	102.6%
Average	79.2	76.9	97.0%

ATAC comparison. We report an analogous set of results comparing to the Adversarially Trained Actor Critic (ATAC) for offline RL algorithm(Cheng et al., 2022) in Table 6. Compared to the other

baseline methods in Section 6.2, ATAC includes additional ingredients to ensure the adversarial approach they use has stable training. For example, they introduce updates on different timescales for actor and critic as well as a term in the critic loss, novel to offline RL, that combines a weighted double Q loss and a residual algorithm loss (Baird, 1995), which they found to be crucial for performance. Neither of these optimisation-related ingredients is strictly related to offline RL, and we did not implement these into CVP in our experiments. Nevertheless, we find that CVP still is able to achieve slightly better performance (102%) than ATAC in these tasks.

Table 6: Comparison between ATAC (Cheng et al., 2022) and CVP results on D4RL benchmarks, indicating the percentage of ATAC's score achieved by CVP-SAC. The ATAC results are taken from Cheng et al. (2022), where the expert datasets are not reported.

Task	ATAC	CVP	Percentage of score achieved
halfcheetah-random	3.9	32.5	833.3%
hopper-random	17.5	27.5	157.1%
walker2d-random	6.8	6.9	101.5%
halfcheetah-medium	53.3	66.0	123.8%
hopper-medium	85.6	72.3	84.4%
walker2d-medium	89.6	84.1	93.9%
halfcheetah-medium-replay	48	57.2	119.2%
hopper-medium-replay	102.5	100.5	98.0%
walker2d-medium-replay	92.5	81.5	85.2%
halfcheetah-medium-expert	94.8	95.4	98.8%
hopper-medium-expert	111.9	103.8	92.7%
walker2d-medium-expert	114.2	109.3	95.7%
Average	68.4	69.7	102.0%