PARAMETERIZED PROJECTED BELLMAN OPERATOR

Anonymous authors
Paper under double-blind review

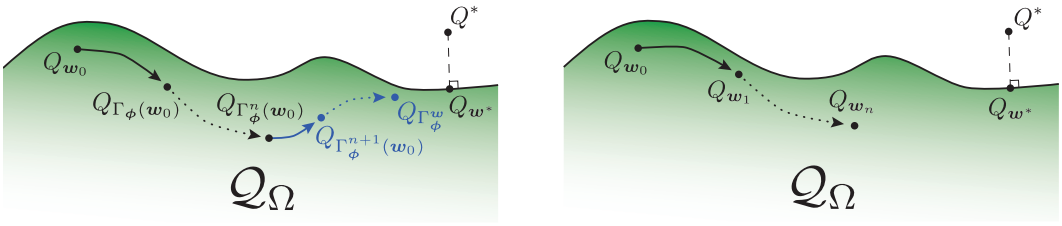
ABSTRACT

The Bellman operator is a cornerstone of reinforcement learning, widely used in a plethora of works, from value-based methods to modern actor-critic approaches. In problems with unknown models, the Bellman operator requires transition samples that strongly determine its behavior, as uninformative samples can result in negligible updates or long detours before reaching the fixed point. In this work, we introduce the novel idea of obtaining an approximation of the Bellman operator, which we call projected Bellman operator (PBO). Our PBO is a parametric operator on the parameter space of a given value function. Given the parameters of a value function, PBO outputs the parameters of a new value function and converges to a fixed point in the limit, as a standard Bellman operator. Notably, our PBO can approximate repeated applications of the true Bellman operator at once, as opposed to the sequential nature of the standard Bellman operator. We prove the important consequences of this finding for different classes of problems by analyzing PBO in terms of stability, convergence, and approximation error. Eventually, we propose an approximate value-iteration algorithm to show how PBO can overcome the limitations of classical methods, opening up multiple research directions as a novel paradigm in reinforcement learning.

1 INTRODUCTION

Value-based reinforcement learning (RL) is a popular class of algorithms for solving sequential decision-making problems with unknown dynamics (Sutton & Barto, 2018). For a given problem, value-based algorithms aim at obtaining the most accurate estimate of the expected return from each state, i.e., a value function. For instance, the well-known value-iteration algorithm computes value functions by iterated applications of the Bellman operator (Bellman, 1966), of which the true value function is the fixed point. Although the Bellman operator can be applied exactly in dynamic programming, it needs to be estimated from samples *at each application* when dealing with unknown models of RL problems, i.e., empirical Bellman operator (Watkins, 1989; Bertsekas, 2019). Intuitively, the dependence of value iteration on the samples has an impact on the efficiency of the algorithms and on the quality of the obtained estimate of the value function, which becomes accentuated when solving continuous problems that require value-based methods with function approximation, e.g., approximate value iteration (AVI) (Munos, 2005; Munos & Szepesvári, 2008).

In this paper, we tackle the original research problem of obtaining a function resembling the Bellman operator. In an ideal scenario, such a function should behave exactly like the true Bellman operator, being a contraction-mapping ending up to the same fixed point. Reasonably, the lack of knowledge of the perfect model of the environment limits us to using the empirical Bellman operator instead of the true one. Thus, we introduce the novel approach of using samples from the empirical Bellman operator to obtain a new operator, which we call *projected Bellman operator* (PBO). Our PBO is a parametric function $\Gamma_{\boldsymbol{\phi}} : \Omega \to \Omega$ defined on parameters $\boldsymbol{\omega} \in \Omega$ of the value function. Similarly to the Bellman operator that computes an updated value function $Q_{\boldsymbol{\omega}_{k+1}}$ given one $Q_{\boldsymbol{\omega}_k}$, our PBO computes the updated value function parameters $\boldsymbol{\omega}_{k+1} = \Gamma_{\boldsymbol{\phi}}(\boldsymbol{\omega}_k)$. The parametric nature of our PBO enables to apply it for an arbitrary number of iterations without using samples, as opposed to the standard Bellman operator. The crucial advantage of our approach is visualized in Figure 1. Starting from initial parameters $\boldsymbol{\omega}_0$, AVI approaches obtain consecutive approximations of the value function $Q_{\boldsymbol{\omega}_k}$ by applying the Bellman operator iteratively over samples, e.g., for n iterations (Figure 1b). Notably, starting from initial parameters $\boldsymbol{\omega}_0$, our PBO can produce a chain of updated parameters of arbitrary length (blue lines in Figure 1a). This means that an accurate PBO can compute the optimal value function parameters starting from any initial parameters in the chosen space. In the following,



(a) PBO (ours) (b) AVI

Figure 1: Behavior of our PBO and approximate value iteration (AVI) in the space of value functions. Q^* and $Q_{\boldsymbol{\omega}^*}$ are respectively the optimal value function and its projection on the parametric space.

after formally introducing PBO and a novel algorithm for value estimation based on it, we analyze its advantageous properties under a theoretical and empirical lens for different classes of problems. Thus, our contribution is threefold: (i) we introduce the notion of projected Bellman operator for a given policy and for the greedy one; (ii) we show how to derive different PBOs according to the class of problems at hand; (iii) we develop a novel algorithm for value estimation based on PBO and show its advantages over related baselines on several RL problems.

2 Related Works

Our work is, to the best of our knowledge, the first attempt to obtain an approximated parametric version of the Bellman operator. Nevertheless, several works in the literature proposed variants of the standard Bellman operator to induce some desired behavior.

Variants of the Bellman Operator. For instance, variants of the Bellman operator are widely studied for entropy-regularized MDPs (Neu et al., 2017; Geist et al., 2019; Belousov & Peters, 2019). The softmax (Haarnoja et al., 2017; Song et al., 2019), mellowmax (Asadi & Littman, 2017), and optimistic (Tosatto et al., 2019) operators are all examples of variants of the Bellman operator to obtain maximum-entropy exploratory policies. Besides favoring exploration, other approaches address the limitations of the standard Bellman operator. For instance, the consistent Bellman operator (Bellemare et al., 2016) is a modified operator that addresses the problem of inconsistency of the optimal action-value functions for suboptimal actions. The distributional Bellman operator (Bellemare et al., 2017) enables to operate on the whole return distribution, instead of its expectation, i.e., the value function (Bellemare et al., 2023). Furthermore, the logistic Bellman operator uses a logistic loss to solve a convex linear programming problem to find optimal value functions (Bas-Serrano et al., 2021). Finally, the Bayesian Bellman operator is a method in Bayesian RL to infer a posterior over Bellman operators centered on the true Bellman operator (Fellows et al., 2021). We point out that our PBO can be seamlessly applied on an arbitrary variant of the standard Bellman operator with just minor adaptations.

Operator Learning. Literature in operator learning is mostly focused on supervised learning, with methods for learning operators over vector spaces (Micchelli & Pontil, 2005) and parametric approaches for learning non-linear operators (Chen & Chen, 1995), with a resurgence of recent contributions in deep learning. For example, Kovachki et al. (2021) learn mappings between infinite function spaces with deep neural networks, or Kissas et al. (2022) apply an attention mechanism to learn correlations in the target function for efficient operator learning. The literature about operator learning is much larger, but we consider it out of the scope of this work that, to the best of our knowledge, is the first to deal with the original problem of operator learning in RL.

3 Preliminaries

We consider discounted Markov decision processes (MDPs) $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$ is a measurable state space, $\mathcal{A}$ is a measurable action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$[1] is the transition kernel of the dynamics of the system, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a reward function and $\gamma \in [0, 1)$ is a discount factor (Puterman, 1990). A deterministic policy $\pi : \mathcal{S} \to \mathcal{A}$ is a function mapping a state

[1] $\Delta(\mathcal{X})$ denotes the set of probability measures over the set $\mathcal{X}$.

to an action, inducing a value function $V^\pi(s) \triangleq \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(S_t, \pi(S_t)) | S_0 = s\right]$, i.e., expected cumulative discounted reward following policy π. Similarly, the action-value function $Q^\pi(s,a) \triangleq \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(S_t, A_t) | S_0 = s, A_0 = a, A_t = \pi(S_t)\right]$ is the expected discounted cumulative reward executing action a in state s, following policy π thereafter. RL aims to find an optimal policy π^* yielding the optimal value function $V^*(s) \triangleq \max_{\pi:\mathcal{S}\to\mathcal{A}} V^\pi(s), \forall s \in \mathcal{S}$ (Puterman, 1990).
The Bellman operator is a fundamental tool in RL for obtaining optimal policies. Given an arbitrary action-value function Q and policy π, the Bellman operator Γ^π is defined as:

$$(\Gamma^\pi Q)(s,a) \triangleq \mathcal{R}(s,a) + \gamma \int_{\mathcal{S}} \mathcal{P}(\mathrm{d}s'|s,a) V^\pi(s'), \quad \forall s \in \mathcal{S}. \tag{1}$$

Similarly, the optimal Bellman operator Γ^* is defined as:

$$(\Gamma^* Q)(s,a) \triangleq \mathcal{R}(s,a) + \gamma \int_{\mathcal{S}} \mathcal{P}(\mathrm{d}s'|s,a) V^*(s'), \quad \forall s \in \mathcal{S}. \tag{2}$$

The value function induced by a policy π is the fixed point of the Bellman operator $\Gamma^\pi Q^\pi = Q^\pi$. Furthermore, the fixed point for the optimal value function is $\Gamma^* Q^* = Q^*$. It is well-known that Bellman operators are contraction mappings in L_∞-norm, such that iterating over a value function leads to the fixed point in the limit (Bertsekas, 2015). We consider the use of function approximation to represent value functions and denote Ω the space of their parameters. Additionally, we define $\mathcal{Q}_\Omega = \{Q_{\boldsymbol{\omega}} : \mathcal{S} \times \mathcal{A} \to \mathbb{R} | \boldsymbol{\omega} \in \Omega\}$ as the set of value functions representable by parameters of Ω.

4 Approximating the Bellman operator

The application of the Bellman operator in RL requires transition samples (Equation 1 and 2) (Munos, 2003; 2005; Munos & Szepesvári, 2008). The impact of this requirement is twofold: (i) the behavior of the Bellman operator strictly depends on the quality of the samples, and, thus, a poor sample selection can result in negligible updates and long detours before convergence; (ii) the direction of the update is determined exclusively by the current samples, meaning that the empirical Bellman operator has no memory of previously observed samples (Bellemare et al., 2016; Agarwal et al., 2019; Fellows et al., 2021). Ideally, we want to obtain an operator that mirrors the behavior of the true Bellman operator while being independent of the transition samples and extrapolating from previous experience. Thus, we introduce the idea of a parametric approximation of the true Bellman operator which we call *projected Bellman operator* (PBO).

Definition 1. *Let $\mathcal{Q}_\Omega = \{Q_{\boldsymbol{\omega}} : \mathcal{S} \times \mathcal{A} \to \mathbb{R} | \boldsymbol{\omega} \in \Omega\}$ be a function approximation space for the action-value function, induced by the parameter space Ω. A projected Bellman operator (PBO) is a function $\Gamma_{\boldsymbol{\phi}} : \Omega \to \Omega$ with parameters $\boldsymbol{\phi} \in \Phi$ acting on the parameters $\boldsymbol{\omega} \in \Omega$.*

We call *π-PBO* the PBO $\Gamma^\pi_{\boldsymbol{\phi}}$ w.r.t. a given policy π and *optimal PBO* the PBO $\Gamma^*_{\boldsymbol{\phi}}$ w.r.t. to the greedy policy. Note that we use the subscript $\boldsymbol{\phi}$ to distinguish PBO from the standard Bellman operator Γ.

4.1 Learning projected Bellman operators

We formalize the problem of learning the closest approximation of the π-PBO and the optimal PBO with parametric functions $\Gamma^\pi_{\boldsymbol{\phi}}$ and $\Gamma^*_{\boldsymbol{\phi}}$, respectively, for a generic RL setting.

Definition 2. *Let π be a policy, we define the **π-PBO** as the operator*

$$\Gamma^\pi_{\boldsymbol{\phi}} \in \underset{\Gamma_{\boldsymbol{\phi}}:\Omega\to\Omega}{\arg\min}\, \mathbb{E}_{(s,a)\sim\rho, \boldsymbol{\omega}\sim\nu} \left[\left(\Gamma^\pi Q_{\boldsymbol{\omega}}(s,a) - Q_{\Gamma_{\boldsymbol{\phi}}(\boldsymbol{\omega})}(s,a) \right)^2 \right], \tag{3}$$

for some state-action distribution ρ and parameter distribution ν.

Note that the symbol Γ^π is the standard Bellman operator w.r.t. policy π on $\mathcal{Q}_\Omega$. Conversely, π-PBO is a parametric operator $\Gamma^\pi_{\boldsymbol{\phi}}$ acting on the space of parameters Ω. The existence of π-PBO is not guaranteed for every setting and has to be studied for each parameterization of the value function. An interesting case is when $\mathcal{Q}_\Omega$ is *stable* by the Bellman operator, i.e., $\Gamma^\pi \mathcal{Q}_\Omega \subset \mathcal{Q}_\Omega$. In this case, π-PBO exists for any $\boldsymbol{\omega} \in \Omega$, and there exist parameters $\boldsymbol{\omega}^*$ such that $\Gamma^\pi Q_{\boldsymbol{\omega}} = Q_{\boldsymbol{\omega}^*}$. The operator defined as $\boldsymbol{\omega} \mapsto \boldsymbol{\omega}^*$ is optimal since it has a value of 0 when evaluated on the function. In addition, π-PBO can be computed in closed form if it exists for $\Gamma^\pi Q_{\boldsymbol{\omega}}$.

Definition 3. *We define the **optimal PBO** as the operator*

$$\Gamma^*_{\boldsymbol{\phi}} \in \arg\min_{\Gamma_{\boldsymbol{\phi}}:\Omega\to\Omega} \mathbb{E}_{(s,a)\sim\rho,\boldsymbol{\omega}\sim\nu}\left[\left(\Gamma^* Q_{\boldsymbol{\omega}}(s,a) - Q_{\Gamma_{\boldsymbol{\phi}}(\boldsymbol{\omega})}(s,a)\right)^2\right], \tag{4}$$

for some state-action distribution ρ and parameter distribution ν.

Again, the symbol Γ^* appears twice to represent two different objects. It is important to note that the optimal Bellman operator is not defined for every function $Q \in \mathcal{Q}_\Omega$ because, for some parameters $\boldsymbol{\omega}$, a state s', and an action a', $\max_{a'\in\mathcal{A}} Q_{\boldsymbol{\omega}}(s',a')$ in Equation (2) might not be defined in the case of continuous actions. This means that the existence of the optimal PBO is not guaranteed and that Ω has to be carefully chosen. Similarly to π-PBO, the case where $\mathcal{Q}_\Omega$ is stable by the optimal Bellman operator, i.e., $\Gamma^*\mathcal{Q}_\Omega \subset \mathcal{Q}_\Omega$, guarantees the existence of the optimal PBO and can be computed in closed form if $\Gamma^* Q_{\boldsymbol{\omega}}$ has one.
The parametric nature of our PBO gives the flexibility of designing different PBOs according to the problem at hand. In the following, we define different PBOs, characterizing their properties for increasingly complex classes of RL problems (all proofs in Appendix A), and we further empirically analyze them in Section 6.

4.2 PBOs in finite Markov decision processes

Let us consider finite state and action spaces of cardinality N and M respectively, and a tabular setting with $\Omega = \mathbb{R}^{N\cdot M}$. Intuitively, there is a trivial bijection between $\mathcal{Q}_\Omega$ and Ω, which allows us to write the parameters of the value function as Q instead of $\boldsymbol{\omega}$. We notice that the space of Q functions is stable by the Bellman operators since any Bellman operator brings a table in $\mathbb{R}^{N\cdot M}$ to a table in $\mathbb{R}^{N\cdot M}$; thus, π-PBO and the optimal PBO exist.
For $\Omega \in \mathbb{R}^N$, where N is the number of parameters, we can derive a class of linear PBOs defined as

$$\{\Gamma_{\boldsymbol{\phi}}(\boldsymbol{\omega}) = \mathbf{A}\boldsymbol{\omega} + \mathbf{b} | \mathbf{A} \in \mathbb{R}^{N\times N}, \mathbf{b} \in \mathbb{R}^N, \boldsymbol{\omega} \in \Omega\}, \tag{5}$$

where $\boldsymbol{\phi}$ is the concatenated vector of $\mathbf{A}$ and $\mathbf{b}$. For any linear PBO parameterized by $\mathbf{A}$ and $\mathbf{b}$ with $\|\mathbf{A}\|_2 < 1$, infinite iterations of this operator lead to the fixed point $(\mathbf{I}-\mathbf{A})^{-1}\mathbf{b}$ by the Banach fixed-point theorem (Khamsi & Kirk, 2011)[2]. This fixed point can be computed in closed form, and it is differentiable w.r.t. the parameters $\boldsymbol{\phi}$. The π-PBO exists and it is equal to the Bellman operator

$$\Gamma^\pi_{\boldsymbol{\phi}}(Q) = R + \gamma P \Pi Q. \tag{6}$$

This clearly shows that π-PBO is a linear function $\Gamma_{\boldsymbol{\phi}}$ parameterized by $\mathbf{A} = \gamma P\Pi \in \mathbb{R}^{N\cdot M\times N\cdot M}$ and $\mathbf{b} = R \in \mathbb{R}^{N\cdot M}$. Notably, π-PBO is a γ-contraction mapping for the sup-norm since the Bellman operator is a γ-contraction mapping.

Proposition 1. *The optimal PBO exists and it is equal to the optimal Bellman operator*

$$\Gamma^*_{\boldsymbol{\phi}}(Q) = R + \gamma P \max_{a\in\mathcal{A}} Q(\cdot, a). \tag{7}$$

The non-linearity caused by the maximum operator requires the introduction of another class of PBOs that can be optimal in this setting. We call this class `max-lin` PBOs, defined as $\Gamma_{\boldsymbol{\phi}}(Q) = \mathbf{b} + \mathbf{A}\max_{a\in\mathcal{A}} Q$, with parameters $\mathbf{b} \in \mathbb{R}^{N\cdot M}$ and $\mathbf{A} \in \mathbb{R}^{N\cdot M\times N}$. Then, the optimal PBO is `max-lin` with parameters $\mathbf{A} = \gamma P$ and $\mathbf{b} = R$. Finally, since the optimal Bellman operator is a γ-contraction mapping for the sup-norm, the optimal PBO is also a γ-contraction mapping.

4.3 PBOs in low-rank MDPs

Low-rank MDPs is a class of problems with two feature maps $\boldsymbol{\sigma} : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^d$ and $\boldsymbol{\mu} : \mathcal{S} \to \mathbb{R}^d$, such that $\mathcal{P}(s'|s,a) = \langle \boldsymbol{\sigma}(s,a), \boldsymbol{\mu}(s')\rangle$ and $\mathcal{R}(s,a) = \langle \boldsymbol{\sigma}(s,a), \boldsymbol{\theta}\rangle$, for all $(s,a,s') \in \mathcal{S}\times\mathcal{A}\times\mathcal{S}$ and for $\boldsymbol{\theta} \in \mathbb{R}^d$. We assume, without loss of generality, that for all $(s,a) \in \mathcal{S}\times\mathcal{A}$, $\|\boldsymbol{\sigma}(s,a)\|_1 \le 1$ and $\max\{\|\boldsymbol{\mu}(s)\|_1, \|\boldsymbol{\theta}\|_1\} \le \sqrt{d}$. Provably, the space of value functions is the space of linear functions in the parameters, i.e., $\mathcal{Q}_\Omega = \{\langle \boldsymbol{\sigma}(\cdot,\cdot), \boldsymbol{\omega}\rangle | \boldsymbol{\omega} \in \mathbb{R}^d\}$. This space is stable by the Bellman

[2] $\Gamma_{\boldsymbol{\phi}}(\boldsymbol{\omega}) = \boldsymbol{\omega} \iff \boldsymbol{\omega} = (\mathbf{I}-\mathbf{A})^{-1}\mathbf{b}$ and $\mathbf{I}-\mathbf{A}$ is invertible since $\|\mathbf{A}\|_2 < 1$.

operators and the closed form can be obtained, as explained in Section 4.1. In case of continuous and unbounded state and action spaces, for a policy π and for parameters $\boldsymbol{\omega} \in \mathbb{R}^d$, the π-PBO is

$$\Gamma^{\pi}_{\boldsymbol{\phi}}(\boldsymbol{\omega}) = \boldsymbol{\theta} + \mathbf{A}_{\pi}\boldsymbol{\omega}, \tag{8}$$

where $\mathbf{A}_{\pi} = \gamma \int_{\mathcal{S}} \boldsymbol{\mu}(s') \mathbb{E}_{a' \sim \pi(\cdot|s')} \left[\boldsymbol{\sigma}(s', a')\right]^T \mathrm{d}s'$. This result shows that π-PBO is a linear PBO; thus, under the latent variable representation assumption, π-PBO is a γ-contraction.

Proposition 2. *In case of continuous state and action spaces and for $\boldsymbol{\omega} \in \mathbb{R}^d$, the optimal PBO is*

$$\Gamma^{*}_{\boldsymbol{\phi}}(\boldsymbol{\omega}) = \boldsymbol{\theta} + \gamma \int_{\mathcal{S}} \max_{a' \in \mathcal{A}} \langle \boldsymbol{\sigma}(s', a') | \boldsymbol{\omega} \rangle \boldsymbol{\mu}(s') \mathrm{d}s'. \tag{9}$$

The optimal PBO is not a linear PBO. However, its class can be understood from its closed form. To make the study easier, we first come back to finite state and action spaces of size N and M respectively. We define a new class of operator that we call `lin-max-lin`, parameterized by $\boldsymbol{A}_1 \in \mathbb{R}^{N \cdot M \times d}$, $\boldsymbol{A}_2 \in \mathbb{R}^{N \times d}$, and $\boldsymbol{b} \in \mathbb{R}^d$, with the following form: $\Gamma_{\boldsymbol{\phi}}(\boldsymbol{w}) = \boldsymbol{b} + \boldsymbol{A}_2^T \max_{a' \in \mathcal{A}} \boldsymbol{A}_1 \boldsymbol{w}$, where $\boldsymbol{\phi}$ is the concatenation of all the parameters. The optimal PBO is a `lin-max-lin` PBO (proof in Appendix A) with parameters:

$$\boldsymbol{A}_1 = \begin{bmatrix} \cdots \\ \boldsymbol{\sigma}(s,a)^T \\ \cdots \end{bmatrix}, \quad \boldsymbol{A}_2 = \gamma \begin{bmatrix} \cdots \\ \boldsymbol{\mu}(s)^T \\ \cdots \end{bmatrix} \quad \text{and} \quad \boldsymbol{b} = \boldsymbol{\theta}. \tag{10}$$

For infinite state and action spaces, the optimal PBO is a `lin-max-lin` operator with parameters having an infinite number of dimensions and, in both finite and infinite cases, the optimal PBO is a γ-contraction mapping under the assumption that the MDP has a latent variable representation.

4.4 PBOs FOR LINEAR QUADRATIC REGULATOR

Eventually, we consider the continuous MDPs class of linear quadratic regulator (LQR) with $\mathcal{S} = \mathcal{A} = \mathbb{R}$. The transition model $\mathcal{P}(s,a) = As + Ba$ is deterministic and the reward function $\mathcal{R}(s,a) = Qs^2 + 2Ssa + Ra^2$ is quadratic, where A, B Q, S and R, are constants inherent to the MDP. As commonly done for LQR, we consider an undiscounted setting, thus we do not analyze the contraction property of the PBOs. We choose to parameterize the action-value functions with a 3-dimensions parameter vector $\mathcal{Q}_{\Omega} = \{(s,a) \mapsto Gs^2 + 2Isa + Ma^2 | (G, I, M) \in \mathbb{R}^3\}$, for visualization purposes. Thus, the optimal action-value function is $Q^*(s,a) = (Q + A^2P)s^2 + 2(S + ABP)sa + (R + B^2P)a^2$, where P is the solution of the Riccati equation (proof in Appendix A). The space of value functions $\mathcal{Q}_{\Omega}$ is not stable for all the Bellman operators, and it depends on the policy π (see Appendix A). Nevertheless, the stability of $\mathcal{Q}_{\Omega}$ can be shown for the policies that are greedy with respect to a value function belonging to $\mathcal{Q}_{\Omega}$. Note that the greedy policy of a value function only exists if the parameters $\boldsymbol{w}$ parameterizing the value function belong to $\mathbb{R}^2 \times \mathbb{R}^-_* \cup \mathbb{R} \times \{0\} \times \{0\}$. Indeed, for $\boldsymbol{w} = (G, I, M) \in \mathbb{R}^3$ and $s \in \mathcal{S}, \arg\max_{a \in \mathcal{A}} Gs^2 + 2Isa + Ma^2$ exists if and only if $M < 0$ or $I = M = 0$. In the case where the greedy policy exists, π-PBO exists, is linear and can be derived in closed form. For a value function parameterized by $\overline{\boldsymbol{w}} = (\overline{G}, \overline{I}, \overline{M}) \in \mathbb{R}^2 \times \mathbb{R}^-_* \cup \mathbb{R} \times \{0\} \times \{0\}$, π-PBO is

$$\Gamma^{\pi^{Q_{\overline{\boldsymbol{w}}}}}_{\boldsymbol{\phi}} : \boldsymbol{w} \mapsto \boldsymbol{b} + \boldsymbol{A}^{\pi^{Q_{\overline{\boldsymbol{w}}}}} \cdot \boldsymbol{w}, \tag{11}$$

where $\pi^{Q_{\overline{\boldsymbol{w}}}}$ is the greedy policy with respect to $Q_{\overline{\boldsymbol{w}}}$, $\boldsymbol{b} = \begin{bmatrix} Q & S & R \end{bmatrix}^T$, $\boldsymbol{A}^{\pi^{Q_{\overline{\boldsymbol{w}}}}} = \begin{bmatrix} A^2 & AB & B^2 \end{bmatrix}^T \cdot \begin{bmatrix} 1 & 2C & C^2 \end{bmatrix}$ and $C = -\frac{\overline{I}}{\overline{M}}$. A geometrical interpretation of the behavior of this PBO is also available in the proof (Appendix A).

The space of value functions $\mathcal{Q}_{\Omega}$ is stable by the optimal Bellman operator when the optimal Bellman operator is defined, i.e., for parameters in $\mathbb{R}^2 \times \mathbb{R}^-_* \cup \mathbb{R} \times \{0\} \times \{0\}$, as for π-PBO.

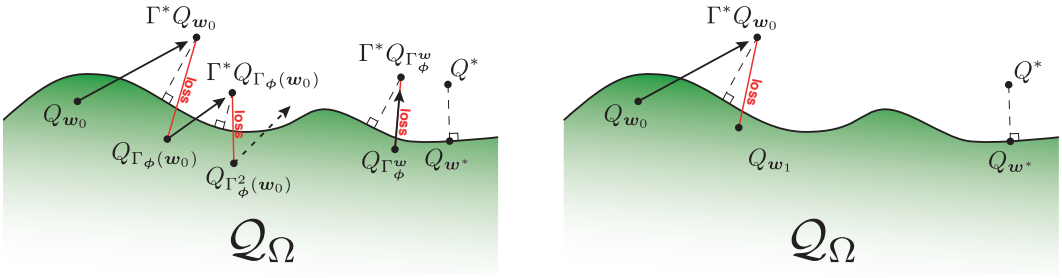
Proposition 3. *The optimal PBO exists and for any $\boldsymbol{w} \in \mathbb{R}^2 \times \mathbb{R}^-_*$ [3] its closed form is*

$$\Gamma^{*}_{\boldsymbol{\phi}} : \boldsymbol{\omega} = \begin{bmatrix} G \\ I \\ M \end{bmatrix} \mapsto \begin{bmatrix} Q + A^2(G - \frac{I^2}{M}) \\ S + AB(G - \frac{I^2}{M}) \\ R + B^2(G - \frac{I^2}{M}) \end{bmatrix}. \tag{12}$$

[3] If $\boldsymbol{w} \in \mathbb{R} \times \{0\} \times \{0\}$, the formula still holds with the convention $0/0 = 0$.

Table 1: Summary of properties of PBOs for different classes of problems.

	Finite MDPs		Low-rank MDPs		LQR	
	$\Gamma^\pi_{\boldsymbol{\phi}}$	$\Gamma^*_{\boldsymbol{\phi}}$	$\Gamma^\pi_{\boldsymbol{\phi}}$	$\Gamma^*_{\boldsymbol{\phi}}$	$\Gamma^\pi_{\boldsymbol{\phi}}$	$\Gamma^*_{\boldsymbol{\phi}}$
Ω	$\mathbb{R}^{N\cdot M}$		$\mathbb{R}^d$		$\mathbb{R}^3$	
$\mathcal{Q}_\Omega$	$\mathbb{R}^{N\cdot M}$		$\{\langle\boldsymbol{\phi}(\cdot,\cdot)\vert\boldsymbol{w}\rangle\vert\boldsymbol{w}\in\mathbb{R}^d\}$		$\{\langle\boldsymbol{f}(s,a)\vert\boldsymbol{w}\rangle\vert\boldsymbol{w}\in\mathbb{R}^3\}$	
Stability	✓	✓	✓	✓	✓	✓
Class	linear	`max-lin`	linear	`lin-max-lin`	linear	Λ`-lin`
Modulus	γ	γ	γ	γ	-	-



(a) PQI (b) FQI

Figure 2: Behavior of PQI and FQI in the space of all value functions for *one* iteration. The dashed lines represent the projections in $\mathcal{Q}_\Omega$.

The optimal PBO can be understood as a composition of the function: $\Lambda : \boldsymbol{w} \mapsto G - \frac{I^2}{M}$ and a linear operator with the following parameters: $\boldsymbol{A} = \begin{bmatrix} A^2 & AB & B^2 \end{bmatrix}^T$ and $\boldsymbol{b} = \begin{bmatrix} Q & S & R \end{bmatrix}^T$. We then have $\Gamma^*_{\boldsymbol{\phi}}(\boldsymbol{w}) = \Lambda(\boldsymbol{w}) \cdot \boldsymbol{A} + \boldsymbol{b}$. We call this class of operator Λ`-lin`. A geometrical interpretation of the behavior of this PBO is available in Appendix A.

Table 1 summarizes the theoretical results stated in this section. For every setting, we report the stability of $\mathcal{Q}_\Omega$ by the Bellman operator along with the class of PBO and its modulus of contraction for the sup-norm. The purple color specifies that the results only hold with the latent variable representation assumption. Moreover, we highlight in orange the result that only holds for greedy policies, and in brown the result that only holds when the optimal Bellman operator is defined.

5 Approximate Value Iteration with Projected Bellman Operators

We devise an algorithm to learn optimal PBOs for a chosen class of PBOs. This choice has to be made according to the parameterization of the value function, as discussed in Section 4. Since we define our PBOs and value functions as differentiable with respect to their parameters, we can optimize with gradient descent on a loss function. A first idea would be to learn an operator that minimizes the TD-error in the definition of the optimal PBO (Equation 4). While this approach has the benefit of allowing to sample as many parameters as needed from Ω, it has the major drawback of not leveraging the crucial fact that learning an operator allows us to iterate over it as many times as desired. Thus, our key insight is to include multiple iterations of the operator to the loss. The function to minimize becomes:

$$\mathcal{L}_{\text{PBO}}(\boldsymbol{\phi}) = \sum_{k=1}^{K} \left(\Gamma^* Q_{\Gamma^{k-1}_{\boldsymbol{\phi}}(\boldsymbol{w})} - Q_{\Gamma^k_{\boldsymbol{\phi}}(\boldsymbol{w})} \right)^2, \tag{13}$$

for all $\boldsymbol{\omega} \in \Omega$, $s \in \mathcal{S}$, and $a \in \mathcal{A}$, and where K is an arbitrary number of optimal Bellman operator iterations. An additional idea is to add a term that corresponds to an infinite number of iterations, i.e., the fixed point, when possible. Now, the loss becomes:

$$\mathcal{L}_{\text{PBO+}}(\boldsymbol{\phi}) = \mathcal{L}_{\text{PBO}}(\boldsymbol{\phi}) + \underbrace{\left(\Gamma^* Q_{\Gamma^{\boldsymbol{w}}_{\boldsymbol{\phi}}} - Q_{\Gamma^{\boldsymbol{w}}_{\boldsymbol{\phi}}} \right)^2}_{\text{Fixed point term}}, \tag{14}$$

Algorithm 1 Projected Q-Iteration (PQI)

Inputs:
samples $\mathcal{D} = \{\langle s_j, a_j, r_j, s'_j \rangle\}_{j=1}^{J}$, parameters $\mathcal{W} = \{\boldsymbol{\omega}_l\}_{l=1}^{L}$,
#Bellman iterations K, #training steps I, #fitting steps H,
operator $\Gamma_{\boldsymbol{\phi}}$ to learn with initial parameters $\boldsymbol{\phi} = \boldsymbol{\phi}_0$, learning rate λ
for $i \in 1 \dots I$ **do**
  # Compute the target T
  for $k \in 1 \dots K$ **do**
    for $(\boldsymbol{w}_l, (s_j, a_j, r_j, s'_j)) \in \mathcal{W} \times \mathcal{D}$ **do**
      $$T_l^k(s_j, a_j) = \underbrace{r_j + \gamma \max_{a \in \mathcal{A}} Q_{\Gamma^{k-1}_{\boldsymbol{\phi}_{i-1}}(\boldsymbol{w}_l)}(s'_j, a)}_{\text{Optimal Bellman iteration of } Q_{\Gamma^{k-1}_{\boldsymbol{\phi}_{i-1}}(\boldsymbol{w}_l)}}$$
    end for
  end for
  # Minimize the loss starting from the previous parameters
  $\boldsymbol{\phi}_i = \boldsymbol{\phi}_{i-1}$
  for $_ \in 1 \dots H$ **do**
    for $j \in 1 \dots J$ **do**
      $$\boldsymbol{\phi}_i \leftarrow \boldsymbol{\phi}_i - \lambda \sum_{l=1}^{L} \sum_{j=1}^{J} \sum_{k=1}^{K} \nabla_{\boldsymbol{\phi}} \left(T_l^k(s_j, a_j) - Q_{\Gamma^k_{\boldsymbol{\phi}_i}(\boldsymbol{w}_l)}(s_j, a_j) \right)^2$$
    end for
  end for
end for
return Parameters $\boldsymbol{\phi}_I$ of operator $\Gamma_{\boldsymbol{\phi}_I}$

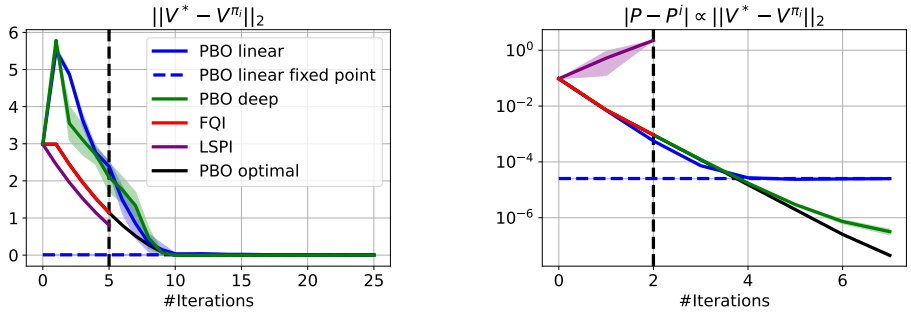
for all $\boldsymbol{\omega} \in \Omega$, $s \in \mathcal{S}$, and $a \in \mathcal{A}$, where $\Gamma^w_{\boldsymbol{\phi}}$ is the fixed point of the PBO. Note that the addition of the fixed point term is only possible for classes of PBOs where the fixed point can be computed and the computation is differentiable in the parameters of the PBO, such as for the class of linear PBOs described in Section 4.2. The pseudo-code of our algorithm is presented in Algorithm 1. The general structure of the algorithm closely resembles the one of Fitted Q-Iteration (Ernst et al., 2005), alternating one phase in which we compute the targets, and another one where we apply gradient descent to fit the parameters w.r.t. the targets. For this reason, we call our algorithm *projected Q-iteration* (PQI). When possible, the fixed point of the PBO is added to the loss. An interesting property of our method is that it tries to learn all the optimal Bellman iterations at once, as opposed to FQI, that learns each Bellman iteration sequentially: $Q_i \in \arg\inf_{Q_i \in \mathcal{Q}} \|Q_i - \Gamma^* Q_{i-1}\|$. Thus, we can conclude that PBO enables to build a richer loss compared to other AVI methods. Figure 2 illustrates this advantage by comparing the way PQI and FQI behave in the space of all possible action-value functions. The ability of iterating for an arbitrary number of times enables us to derive theoretical advantages of PQI by leveraging established results in approximation error for AVI.

Theorem 1. *(Theorem 3.4 of Farahmand (2011)) Let $K \in \mathbb{N}^*$, ρ, ν two distribution probabilities over $\mathcal{S} \times \mathcal{A}$. For any sequence $(Q_k)_{k=0}^{K} \subset B(\mathcal{S} \times \mathcal{A}, R_\gamma)$ where R_γ depends on the reward function and the discount factor, we have*

$$||Q^* - Q^{\pi_K}||_{1,\rho} \leq \inf_{r \in [0,1]} F(r; K, \rho, \gamma) \left(\overbrace{\sum_{k=1}^{K} \alpha_k^{2r} \underbrace{||\Gamma^* Q_{k-1} - Q_k||_{2,\nu}^2}_{FQI}}^{PQI} \right)^{\frac{1}{2}} + C_{K,\gamma,R_\gamma}, \tag{15}$$

where α_k and C_{K,γ,R_γ} do not depend on the sequence $(Q_k)_{k=0}^{K}$. $F(r; K, \rho, \gamma)$ relies on the concentrability coefficients of the greedy policies w.r.t. the value functions.

This theorem shows that the distance between the value function and the optimal value function depends on the approximation errors for all the iterations. The loss of FQI contains only one term of the sum, while the loss of PQI contains the entire sum. In PQI, the terms are summed up with equal weights, which is not the case for the error propagation. Future works could investigate how to weight those terms for increased performance.



(a) Chain-walk. (b) LQR.

Figure 4: ℓ_2-norm between the optimal value function and the value function of the greedy policy from the learned value function. Results averaged over 20 seeds with 95% confidence interval.

6 Experiments

We empirically analyze our PQI algorithm with different classes of PBOs w.r.t. FQI (Ernst et al., 2005) and, when possible, the additional baseline least-squares policy iteration (LSPI) (Lagoudakis & Parr, 2003). Note that the fixed point is always added to the loss for the linear PBO, unless stated otherwise. The pseudocodes of FQI and LSPI are provided in Appendix B.

Experimental details. Each algorithm receives a fixed dataset of samples $\mathcal{D}$ and has access to the same number of optimal Bellman iterations K. We define the initial parameters, which we call *validation parameters*, for both FQI and LSPI such that they result in a zero-function. We store the parameters after each iteration of the algorithms; then, we compute the action-value function for each iteration and the value function associated with the greedy policy computed from the action-value function at each iteration. PQI can access the same functions by iterating over the validation parameters for more than K iterations, thanks to our PBO. When possible, we include the parameters corresponding to an infinite number of iterations, i.e., the fixed point of the PBO. In the figures, the PBOs that use more complex function approximators than linear PBOs are denoted *PBO deep*. Additional details on the experiments are in Appendix C.

6.1 Chain-walk

We initially consider the chain-walk environment in Figure 3, with a chain of length $N = 20$. We parameterize value functions as tables to leverage our theoretical results on finite MDPs (Section 4.2), training a linear PBO and a `max-lin` PBO. In this simple setting, we also have access to the optimal PBO and we include it in our analysis. We plot the ℓ_2-norm between the iterated value functions and the optimal ones in Figure 4a. For all methods, the Bellman operator is iterated for $K = 5$ times during training. Notably, we can iterate the learned PBO for an arbitrary number of times. This property is crucial to nullify the distance between the value function computed with the PBOs and the optimal value function. The superior performance of PQI comes from the fact that both classes of operators are close to the class of the optimal PBO. Since in this setting we have access to the optimal Bellman operator through the optimal PBO[4], we can foresee that any algorithm minimizing the approximation error by learning parameters of value functions would perform in the best case as good as the optimal Bellman operator, meaning that their performance would coincide with the one of the optimal PBO at iteration 5. For this reason, any algorithm designed as such would perform worse than or equal to FQI in this setting. Note that LSPI is not aiming at reducing the approximation error, this is why the reasoning does not apply to this baseline.

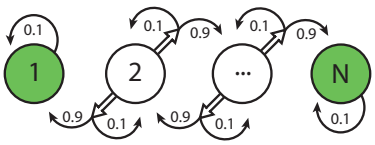
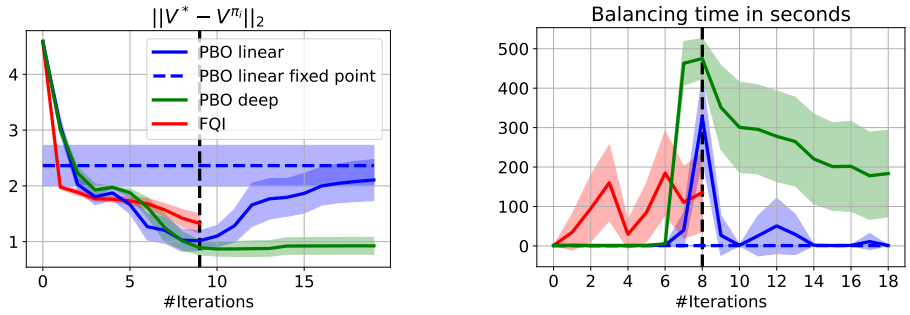


Figure 3: The chain-walk from Munos (2003). Reward is always 0, but 1 in green states.

[4] As shown in Section 4.2, $\forall \boldsymbol{\omega} \in \Omega, \Gamma^* Q_{\boldsymbol{\omega}} = Q_{\Gamma^*_{\boldsymbol{\phi}}(\boldsymbol{\omega})}$.



(a) Car-on-hill. (b) Bicycle.

Figure 5: Weighted ℓ_2-norm for the value function on car-on-hill (see Appendix C.3) and trajectories length in bicycle. Results averaged over 20 seeds with 95% confidence interval.

6.2 Linear quadratic regulator

We consider the LQR setting in Section 4.4, leveraging our theoretical analysis by parameterizing value functions accordingly. The theoretical analysis suggests using a Λ-`lin` PBO, and we also use a linear PBO. Figure 4b shows a metric proportional to the distance between the optimal value function and the value functions obtained at each iteration. For all methods, we set $K = 2$. As in the previous experiment, the chosen class of operators is close to the class of the optimal PBO. We see that, at iteration 2, the performance of FQI and the PQI are on par, while afterward, PQI keeps improving until the fixed point is reached. Again, we can state that any algorithm that learns parameters of value functions trying to minimize the approximation error, would perform worse than FQI. FQI follows the curve of the optimal PBO, meaning that the training cannot be improved. LSPI performs badly because the first policy for the evaluation step is a uniform policy, and the value function associated with the uniform policy is not a quadratic function.

6.3 Non-linear projected Bellman operators

We consider a more advanced setting where the value function is approximated by a neural network, thus the space of value functions is not stable by the optimal Bellman operator. We focus on the car-on-hill MDP (Ernst et al., 2005), as implemented in MushroomRL (D'Eramo et al., 2021), and the bicycle MDP in Randlov & Alstrøm (1998). We parameterize value functions with one hidden layer of 30 neurons, and we compare PQI only to FQI, since LSPI is not suitable for this setting. We train a linear PBO and, inspired by the theoretical analysis in Section 4.3, we train an operator close to being `lin-max-lin` – its architecture is `lin-ReLU-lin-max-lin-ReLU-lin`. We set $K = 9$ for car-on-hill and $K = 8$ for bicycle. Figure 5 shows that PBOs outperform FQI. Interestingly, iterating further produces a drop in performance for the linear PBO, indicating that the class of linear operator is too simple. Remarkably, deep PBO either shows no deterioration (Figure 5a), or a slight deterioration due to opposite performance in different seeds (Figure 5b and Appendix C.4). Further tuning should substantially reduce the number of failing seeds. This experiment clearly evinces the advantage of PQI. FQI is a greedy algorithm that follows a path to the optimal parameters step-by-step, thus, if an error is made at an iteration, it propagates until the last iteration (Theorem 1). Conversely, PQI can change the path to the optimal parameters at each optimization step.

7 Conclusion

We introduced the original idea of obtaining an approximation of the true Bellman operator, which we call projected Bellman operator (PBO). We defined PBO for different classes of problems, studied their theoretical properties of stability and convergence, and devised a novel approximate value iteration algorithm that provably reduces the error propagation over Bellman iterations. Finally, we have empirically shown the benefit of PBO over related baselines on a variety of problems. We consider this work a seminal study of PBO as a novel paradigm of RL, that can lead to multiple research directions, among which investigating its use in online settings is a first foreseeable avenue.

8 REPRODUCIBILITY STATEMENT

We aim to guarantee the reproducibility of our work by providing the proofs of our theoretical results and additional details of the conducted experiments in the appendix. Further, we provide the code for reproducing our results in the supplementary material, and we will make it public upon acceptance.

REFERENCES

Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, pp. 10–4, 2019.

Kavosh Asadi and Michael L Littman. An alternative softmax operator for reinforcement learning. In *International Conference on Machine Learning*, pp. 243–252. PMLR, 2017.

Joan Bas-Serrano, Sebastian Curi, Andreas Krause, and Gergely Neu. Logistic q-learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 3610–3618. PMLR, 2021.

Marc G Bellemare, Georg Ostrovski, Arthur Guez, Philip Thomas, and Rémi Munos. Increasing the action gap: New operators for reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, pp. 449–458. PMLR, 2017.

Marc G. Bellemare, Will Dabney, and Mark Rowland. *Distributional Reinforcement Learning*. MIT Press, 2023. `http://www.distributional-rl.org`.

Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

Boris Belousov and Jan Peters. Entropic regularization of markov decision processes. *Entropy*, 21 (7):674, 2019.

Dimitri Bertsekas. *Reinforcement learning and optimal control*. Athena Scientific, 2019.

Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control 4 th Edition , Volume II*. Athena Scientific, 2015.

Tianping Chen and Hong Chen. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 6(4):911–917, 1995.

Carlo D'Eramo, Davide Tateo, Andrea Bonarini, Marcello Restelli, and Jan Peters. Mushroomrl: Simplifying reinforcement learning research. *Journal of Machine Learning Research*, 22(131): 1–5, 2021.

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *JMLR*, 6:503–556, dec 2005. ISSN 1532-4435.

Amir-massoud Farahmand. Regularization in reinforcement learning. 2011.

Mattie Fellows, Kristian Hartikainen, and Shimon Whiteson. Bayesian bellman operators. *Advances in Neural Information Processing Systems*, 34:13641–13656, 2021.

Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized markov decision processes. In *International Conference on Machine Learning*, pp. 2160–2169. PMLR, 2019.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International conference on machine learning*, pp. 1352–1361. PMLR, 2017.

Mohamed A Khamsi and William A Kirk. *An introduction to metric spaces and fixed point theory*. John Wiley & Sons, 2011.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Georgios Kissas, Jacob H Seidman, Leonardo Ferreira Guilhoto, Victor M Preciado, George J Pappas, and Paris Perdikaris. Learning operators with coupled attention. *Journal of Machine Learning Research*, 23(215):1–63, 2022.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces. *arXiv preprint arXiv:2108.08481*, 2021.

Michail G Lagoudakis and Ronald Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003.

Charles A Micchelli and Massimiliano Pontil. On learning vector-valued functions. *Neural computation*, 17(1):177–204, 2005.

Rémi Munos. Error bounds for approximate policy iteration. In *ICML*, volume 3, pp. 560–567, 2003.

Rémi Munos. Error bounds for approximate value iteration. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, pp. 1006. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(5), 2008.

Gergely Neu, Anders Jonsson, and Vicenç Gómez. A unified view of entropy-regularized markov decision processes. *arXiv preprint arXiv:1705.07798*, 2017.

Martin L Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

Jette Randlov and Preben Alstrøm. Learning to drive a bicycle using reinforcement learning and shaping. In *Proceedings of the 15th International Conference on Machine Learning*, pp. 463–471, 01 1998.

Zhao Song, Ron Parr, and Lawrence Carin. Revisiting the softmax bellman operator: New benefits and new perspective. In *International conference on machine learning*, pp. 5916–5925. PMLR, 2019.

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

Samuele Tosatto, Carlo D'Eramo, Joni Pajarinen, Marcello Restelli, and Jan Peters. Exploration driven by an optimistic bellman equation. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2019.

Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.

A PROOFS

Proof for Section 4.2: existence of π-PBO for a finite MDP. As suggested in Section 4.1, we compute the Bellman iteration for a policy π over a table $Q \in \mathbb{R}^{N \cdot M}$

$$\begin{aligned}\Gamma^\pi Q(s,a) &= r(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)} \left[\mathbb{E}_{a' \sim \pi(\cdot|s')} [Q(s',a')] \right] \\ &= r(s,a) + \gamma \sum_{s'} p(s'|s,a) \left[\mathbb{E}_{a' \sim \pi(\cdot|s')} [Q(s',a')] \right] \\ &= r(s,a) + \gamma \sum_{s',a'} p(s'|s,a) \pi(a'|s') Q(s',a') \\ &= (R + \gamma P \Pi Q)(s,a), \end{aligned} \tag{16}$$

where $P \in \mathbb{R}^{N \cdot M \times N}$ is the transition probability matrix of the environment and $\Pi \in \mathbb{R}^{N \times N \cdot M}$ is the probability matrix if the policy π is followed. From these equations, we get that π-PBO exists and $\Gamma^\pi_{\boldsymbol{\phi}}(Q) = R + \gamma P \Pi Q$. □

Proof for Section 4.2: existence of the optimal PBO for a finite MDP. We proceed the same way as for π-PBO. By replacing the expectation over the next action by a $\max$ function in (16), we obtain

$$\Gamma^* Q(s,a) = (R + \gamma P \max_{a'} Q)(s,a).$$

We then get that the optimal PBO exists and $\Gamma^*_{\boldsymbol{\phi}}(Q) = R + \gamma P \max_{a'} Q$. □

Proof for Section 4.3: existence of π-PBO for a low-rank MDP. The proof considers continuous unbounded state and action spaces. For $(s,a) \in \mathcal{S} \times \mathcal{A}$, $Q_\omega \in \mathcal{Q}_\Omega$ and $\boldsymbol{w}$ the vector representing Q_ω, we write the Bellman iteration as

$$\begin{aligned}\Gamma^\pi Q_{\boldsymbol{w}}(s,a) &= r(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)} \left[\mathbb{E}_{a' \in \pi(\cdot|s')} \left[Q_{\boldsymbol{w}}(s',a') \right] \right] \\ &= r(s,a) + \gamma \int_{s'} E_{a' \in \pi(\cdot|s')} \left[\langle \boldsymbol{\sigma}(s',a') | \boldsymbol{w} \rangle \right] p(s'|s,a) \mathrm{d}s' \\ &= \langle \boldsymbol{\sigma}(s,a) | \boldsymbol{\theta} \rangle + \gamma \int_{s'} \mathbb{E}_{a' \in \pi(\cdot|s')} \left[\langle \boldsymbol{\sigma}(s',a') | \boldsymbol{w} \rangle \right] \langle \boldsymbol{\sigma}(s,a) | \boldsymbol{\mu}(s') \rangle \mathrm{d}s' \text{ from the low-rank MDP} \\ &\quad \text{assumption.} \\ &= \langle \boldsymbol{\sigma}(s,a) | \boldsymbol{\theta} + \gamma \int_{s'} \mathbb{E}_{a' \in \pi(\cdot|s')} \left[\langle \boldsymbol{\sigma}(s',a') | \boldsymbol{w} \rangle \right] \boldsymbol{\mu}(s') \mathrm{d}s' \rangle \text{ from the linearity of the scalar} \\ &\quad \text{product in its second variable.} \\ &= \langle \boldsymbol{\sigma}(s,a) | \boldsymbol{\theta} + \gamma \int_{s'} \langle \mathbb{E}_{a' \in \pi(\cdot|s')} \left[\boldsymbol{\sigma}(s',a') \right] | \boldsymbol{w} \rangle \boldsymbol{\mu}(s') \mathrm{d}s' \rangle \text{ from the linearity of the scalar} \\ &\quad \text{product in its first variable.} \\ &= \langle \boldsymbol{\sigma}(s,a) | \boldsymbol{\theta} + \gamma \int_{s'} \left(\mathbb{E}_{a' \in \pi(\cdot|s')} \left[\boldsymbol{\sigma}(s',a') \right]^T \boldsymbol{w} \right) \boldsymbol{\mu}(s') \mathrm{d}s' \rangle \\ &= \langle \boldsymbol{\sigma}(s,a) | \boldsymbol{\theta} + \gamma \int_{s'} \boldsymbol{\mu}(s') \left(\mathbb{E}_{a' \in \pi(\cdot|s')} \left[\boldsymbol{\sigma}(s',a') \right]^T \boldsymbol{w} \right) \mathrm{d}s' \rangle \\ &= \langle \boldsymbol{\sigma}(s,a) | \boldsymbol{\theta} + \gamma \left(\int_{s'} \boldsymbol{\mu}(s') \mathbb{E}_{a' \in \pi(\cdot|s')} \left[\boldsymbol{\sigma}(s',a') \right]^T \mathrm{d}s' \right) \boldsymbol{w} \rangle \text{ since } \boldsymbol{\omega} \text{ does not depend on } s'. \end{aligned}$$

This means that $\Gamma^\pi Q_{\boldsymbol{w}} \in \mathcal{Q}_\Omega$. By setting $\boldsymbol{A}_\pi = \gamma \int_{s'} \boldsymbol{\mu}(s') \mathbb{E}_{a' \in \pi(\cdot|s')} \left[\boldsymbol{\sigma}(s',a') \right]^T \mathrm{d}s'$, we obtain that π-PBO exists and $\Gamma^\pi_{\boldsymbol{\phi}}(\boldsymbol{\omega}) = \boldsymbol{\theta} + \mathbf{A}_\pi \boldsymbol{\omega}$. □

Proof for Section 4.3: contraction of π-PBO for a low-rank MDP. We now assume that the MDP has a latent variable representation, which means that the vectors $\boldsymbol{\phi}(\cdot,\cdot)$ are probability distributions over $\{1,\dots,d\}$ and, for every $i \in \{1,\dots,d\}$, $\boldsymbol{\mu}(\cdot)_i$ is a probability distribution over $\mathcal{S}$. This proof is inspired by the proof of the contraction property of the Bellman Operator in Bertsekas (2015). To show the property, it is enough to show that for any vector $\boldsymbol{w} \in \mathbb{R}^d$ such that $\|\boldsymbol{w}\|_\infty = 1$,

$\|\boldsymbol{A}_\pi \boldsymbol{w}\|_\infty \leq \gamma$. If this is true, then for $(\boldsymbol{w}, \boldsymbol{w}') \in \mathbb{R}^d$ such that $\boldsymbol{w} \neq \boldsymbol{w}'$, $||\Gamma^\pi_{\boldsymbol{\phi}}(w) - \Gamma^\pi_{\boldsymbol{\phi}}(w')||_\infty = ||\boldsymbol{A}_\pi(\boldsymbol{w} - \boldsymbol{w}')||_\infty = ||\frac{\boldsymbol{A}_\pi(\boldsymbol{w}-\boldsymbol{w}')}{||\boldsymbol{w}-\boldsymbol{w}'||_\infty}||_\infty \cdot ||\boldsymbol{w} - \boldsymbol{w}'||_\infty \leq \gamma ||\boldsymbol{w} - \boldsymbol{w}'||_\infty$.

Considering $\boldsymbol{w} \in \mathbb{R}^d$, such that $||\boldsymbol{w}||_\infty = 1$, we have

$$
\begin{aligned}
||\boldsymbol{A}_\pi \boldsymbol{w}||_\infty &= ||\gamma \int_{s'} \boldsymbol{\mu}(s') \mathbb{E}_{a' \in \pi(\cdot|s')} \left[\boldsymbol{\sigma}(s', a')\right]^T \mathrm{d}s' \boldsymbol{w}||_\infty \\
&= \gamma \max_{i \in \{1,\dots,d\}} \left| \int_{s'} \boldsymbol{\mu}(s')_i \sum_{j \in \{1,\dots,d\}} \mathbb{E}_{a' \in \pi(\cdot|s')} \left[\boldsymbol{\sigma}(s', a')_j\right] \boldsymbol{w}_j \mathrm{d}s' \right| \\
&\leq \gamma \max_{i \in \{1,\dots,d\}} \int_{s'} \boldsymbol{\mu}(s')_i \sum_{j \in \{1,\dots,d\}} \mathbb{E}_{a' \in \pi(\cdot|s')} \left[\boldsymbol{\sigma}(s', a')_j\right] |\boldsymbol{w}_j| \mathrm{d}s' \text{ since } \boldsymbol{\mu}(\cdot) \text{ and} \\
&\boldsymbol{\phi}(\cdot,\cdot) \text{ are positive.} \\
&\leq \gamma \max_{i \in \{1,\dots,d\}} \int_{s'} \boldsymbol{\mu}(s')_i \sum_{j \in \{1,\dots,d\}} \mathbb{E}_{a' \in \pi(\cdot|s')} \left[\boldsymbol{\sigma}(s', a')_j\right] \cdot ||\boldsymbol{w}||_\infty \mathrm{d}s' \\
&\leq \gamma \max_{i \in \{1,\dots,d\}} \int_{s'} \boldsymbol{\mu}(s')_i \mathbb{E}_{a' \in \pi(\cdot|s')} \left[\sum_{j \in \{1\dots,d\}} \boldsymbol{\sigma}(s', a')_j\right] \mathrm{d}s' \text{ by linearity of the} \\
&\text{expectation and } ||\boldsymbol{w}||_\infty = 1. \\
&\leq \gamma \max_{i \in \{1,\dots,d\}} \int_{s'} \boldsymbol{\mu}(s')_i \mathrm{d}s' \text{ since } \boldsymbol{\phi}(\cdot,\cdot) \text{ are probability distributions.} \\
&\leq \gamma \text{ since } \boldsymbol{\mu}(\cdot)_i \text{ is a probability distribution for all } i.
\end{aligned}
$$

□

Proof for Section 4.3: existence of the optimal PBO for a low-rank MDP. The proof considers continuous unbounded state action spaces. For $Q_{\boldsymbol{w}} \in \mathcal{Q}_\Omega$, $\boldsymbol{w}$ the vector representing $Q_{\boldsymbol{w}}$, $\max_{a \in \mathcal{A}} Q_{\boldsymbol{w}}(s,a)$ is well defined (here $\max$ might not be attained, it should be interpreted as a supremum). We have $|Q_{\boldsymbol{w}}(s,a)| = |\langle \boldsymbol{\sigma}(s,a)|\boldsymbol{w}\rangle| \leq ||\boldsymbol{\sigma}(s,a)|| \cdot ||\boldsymbol{w}|| \leq ||\boldsymbol{w}||$, thus $\max_{a \in \mathcal{A}} Q_{\boldsymbol{w}}(s,a) < \infty$. Then, we write the optimal Bellman iteration on the function $Q_{\boldsymbol{w}}$ as

$$
\begin{aligned}
\Gamma^* Q_{\boldsymbol{w}}(s,a) &= r(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)} \left[\max_{a' \in \mathcal{A}} Q_{\boldsymbol{w}}(s', a')\right] \\
&= r(s,a) + \gamma \int_{s'} \max_{a'} \langle \boldsymbol{\sigma}(s', a')|\boldsymbol{w}\rangle p(s'|s,a) \mathrm{d}s' \\
&= \langle \boldsymbol{\sigma}(s,a)|\boldsymbol{\theta}\rangle + \gamma \int_{s'} \max_{a'} \langle \boldsymbol{\sigma}(s', a')|\boldsymbol{w}\rangle \langle \boldsymbol{\sigma}(s,a)|\boldsymbol{\mu}(s')\rangle \mathrm{d}s' \text{ from the definition of the transition} \\
&\text{probabilities.} \\
&= \langle \boldsymbol{\sigma}(s,a)|\boldsymbol{\theta} + \gamma \int_{s'} \max_{a'} \langle \boldsymbol{\sigma}(s', a')|\boldsymbol{w}\rangle \boldsymbol{\mu}(s') \mathrm{d}s'\rangle \text{ from the linear of the scalar product.}
\end{aligned}
$$

This derivation shows that $\Gamma^* Q_{\boldsymbol{w}} \in \mathcal{Q}_\Omega$ and that the optimal PBO exists and can be computed in closed form. □

Proof for Section 4.3: class of the optimal PBO for a low-rank MDP. In the finite case, the optimal PBO is

$$
\begin{aligned}
\Gamma^*_{\boldsymbol{\phi}}(\boldsymbol{w}) &= \boldsymbol{\theta} + \gamma \sum_{s'} \max_{a'} \langle \boldsymbol{\sigma}(s', a')|\boldsymbol{w}\rangle \boldsymbol{\mu}(s') \\
&= \boldsymbol{\theta} + \gamma \sum_{s'} \max_{a'} \left(\sum_{i \in \{0,\dots,d\}} \boldsymbol{\sigma}(s', a')_i \boldsymbol{w}_i \right) \boldsymbol{\mu}(s').
\end{aligned}
$$

We can see that, by defining $\boldsymbol{A}_1, \boldsymbol{A}_2$ and $\boldsymbol{b}$ as proposed in Section 4.3, the optimal PBO belongs to the class of `lin-max-lin` PBOs. □

Proof for Section 4.3: contraction of the optimal PBO for a low-rank MDP. We now assume that the MDP has a latent variable representation. This proof was inspired by the proof of the contraction property of the Bellman Operator in Bertsekas (2015). Considering $\boldsymbol{w}, \boldsymbol{w}' \in \mathbb{R}^d$, we have

$$
\begin{aligned}
||\Gamma^*_{\boldsymbol{\phi}}(\boldsymbol{w}) - \Gamma^*_{\boldsymbol{\phi}}(\boldsymbol{w}')||_\infty &= \gamma || \int_{s'} \max_{a'} \langle \boldsymbol{\sigma}(s',a') | (\boldsymbol{w} - \boldsymbol{w}') \rangle \boldsymbol{\mu}(s') \mathrm{d}s' ||_\infty \\
&= \gamma \max_{i \in \{1,\dots,d\}} \left| \int_{s'} \max_{a'} \langle \boldsymbol{\sigma}(s',a') | (\boldsymbol{w} - \boldsymbol{w}') \rangle \boldsymbol{\mu}(s')_i \mathrm{d}s' \right| \\
&\leq \gamma \max_{i \in \{1,\dots,d\}} \int_{s'} \max_{a'} |\langle \boldsymbol{\sigma}(s',a') | (\boldsymbol{w} - \boldsymbol{w}') \rangle| \, \boldsymbol{\mu}(s')_i \mathrm{d}s' \text{ since } \boldsymbol{\mu}(\cdot) \text{ is positive.} \\
&\leq \gamma \max_{i \in \{1,\dots,d\}} \int_{s'} \max_{a'} \sum_{j \in \{1,\dots,d\}} \boldsymbol{\sigma}(s',a')_j |(\boldsymbol{w} - \boldsymbol{w}')_j| \boldsymbol{\mu}(s')_i \mathrm{d}s' \text{ since } \boldsymbol{\phi}(\cdot,\cdot) \text{ are positive.} \\
&\leq \gamma \max_{i \in \{1,\dots,d\}} \int_{s'} \max_{a'} \sum_{j \in \{1,\dots,d\}} \boldsymbol{\sigma}(s',a')_j \boldsymbol{\mu}(s')_i \mathrm{d}s' \cdot ||\boldsymbol{w} - \boldsymbol{w}'||_\infty \\
&\leq \gamma \max_{i \in \{1,\dots,d\}} \int_{s'} \boldsymbol{\mu}(s')_i \mathrm{d}s' \cdot ||\boldsymbol{w} - \boldsymbol{w}'||_\infty \text{ since } \boldsymbol{\phi}(\cdot,\cdot) \text{ are probability distributions.} \\
&\leq \gamma ||\boldsymbol{w} - \boldsymbol{w}'||_\infty \text{ since } \boldsymbol{\mu}(\cdot)_i \text{ is a probability distribution for all } i.
\end{aligned}
$$

□

Proof for Section 4.4: optimal Q function for LQR MDP. First, note that the LQR setting provides a closed-form solution for the optimal value function: $V^*(s) = P \cdot s^2$ with P the solution of the Riccati equation. From there, we can compute the value function in closed-form

$$
\begin{aligned}
Q^*(s,a) &= r(s,a) + V^*(s') \\
&= r(s,a) + P \cdot s'^2 \\
&= Q \cdot s^2 + 2S \cdot sa + R \cdot a^2 + P \cdot (A \cdot s + B \cdot a)^2 \\
&= (Q + A^2 P) \cdot s^2 + 2(S + ABP) \cdot sa + (R + B^2 P) \cdot a^2.
\end{aligned}
$$

□

Proof for Section 4.4: non stability of $\mathcal{Q}_\Omega$ for LQR MDP. Let π be a policy, $\boldsymbol{w} \in \mathbb{R}^3$, the Bellman iteration over $Q_{\boldsymbol{w}}$ is

$$
\begin{aligned}
\Gamma^\pi Q_{\boldsymbol{w}}(s,a) &= r(s,a) + \int_{a'} \pi(a'|s') Q_{\boldsymbol{w}}(s',a') \mathrm{d}a' \\
&= Qs^2 + 2Ssa + Ra^2 + \int_{a'} \pi(a'|s') \left(Gs'^2 + 2Is'a' + Ma'^2 \right) \mathrm{d}a' \\
&= Qs^2 + 2Ssa + Ra^2 + Gs'^2 + 2Is' \mathbb{E}_{a' \sim \pi(\cdot|s')}[a'] + M \mathbb{E}_{a' \sim \pi(\cdot|s')}[a'^2] \\
&= Qs^2 + 2Ssa + Ra^2 + G(As + Ba)^2 + 2I(As + Ba) \mathbb{E}_{a' \sim \pi(\cdot|s')}[a'] + M \mathbb{E}_{a' \sim \pi(\cdot|s')}[a'^2] \\
&\text{by definition of } s'. \\
&= [Q + GA^2]s^2 + 2[S + GAB]sa + [R + GB^2]a^2 + 2I \mathbb{E}_{a' \sim \pi(\cdot|s')}[a'](As + Ba) \\
&+ M \mathbb{E}_{a' \sim \pi(\cdot|s')}[a'^2].
\end{aligned}
$$

Note that we can take, for example, any policy with a constant momentum of order 2 for the state s' (i.e. $\mathbb{E}_{a' \sim \pi(\cdot|s')}[a'^2]$ is constant) and then $\Gamma^\pi Q_{\boldsymbol{w}} \notin \mathcal{Q}_\Omega$. This means that the space of value functions $\mathcal{Q}_\Omega$ is not stable by the Bellman operator associated with the policies that have this property. □

Proof for Section 4.4: existence of π-PBO for greedy policies for LQR MDP. Let $\overline{\boldsymbol{w}} = (\overline{G}, \overline{I}, \overline{M}) \in \mathbb{R}^2 \times \mathbb{R}_*^-$ [5] and let $\pi^{Q_{\overline{\boldsymbol{w}}}}$ be a greedy policy with respect to $Q_{\overline{\boldsymbol{w}}}$. Since $Q_{\overline{\boldsymbol{w}}}$ is a quadratic polynomial with a negative coefficient in front of the monomial of degree 2, for any $s \in \mathcal{S}$, $\pi^{Q_{\overline{\boldsymbol{w}}}}(s) = {}^{-\overline{I}}/_{\overline{M}} \cdot s$. We note $C = {}^{-\overline{I}}/_{\overline{M}}$. From the derivations of the previous proof, for $\boldsymbol{w} = (G, I, M) \in \mathbb{R}^2 \times \mathbb{R}_*^-$, the Bellman iteration over $Q_{\boldsymbol{w}}$ is

$$
\begin{aligned}
&\Gamma^{\pi^{Q_{\overline{\boldsymbol{w}}}}} Q_{\boldsymbol{w}}(s,a) = [Q + GA^2]s^2 + 2[S + GAB]sa + [R + GB^2]a^2 \\
&\quad + 2I\mathbb{E}_{a' \sim \pi^{Q_{\overline{\boldsymbol{w}}}}(\cdot|s')}[a'](As + Ba) + M\mathbb{E}_{a' \sim \pi^{Q_{\overline{\boldsymbol{w}}}}(\cdot|s')}[a'^2] \\
&\quad = [Q + GA^2]s^2 + 2[S + GAB]sa + [R + GB^2]a^2 + 2IC \cdot s'(As + Ba) + M(C \cdot s')^2 \text{since} \\
&\quad \pi^{Q_{\overline{\boldsymbol{w}}}} \text{ is deterministic.} \\
&\quad = [Q + GA^2]s^2 + 2[S + GAB]sa + [R + GB^2]a^2 + [2IC + MC^2](As + Ba)^2 \\
&\quad = [Q + A^2(G + 2IC + MC^2)]s^2 + 2[S + AB(G + 2IC + MC^2)]sa \\
&\quad + [R + B^2(G + 2IC + MC^2)]a^2 \\
&\quad = \left(\begin{bmatrix} Q \\ S \\ R \end{bmatrix} + \begin{bmatrix} A^2(G + 2IC + MC^2) \\ AB(G + 2IC + MC^2) \\ B^2(G + 2IC + MC^2) \end{bmatrix} \right) \cdot \begin{bmatrix} s^2 \\ 2sa \\ a^2 \end{bmatrix}.
\end{aligned}
$$

This shows the stability of $\mathcal{Q}_\Omega$ by the Bellman operator since $\Gamma^{\pi^{Q_{\overline{\boldsymbol{w}}}}} Q_{\boldsymbol{w}} \in \mathcal{Q}_\Omega$. We have

$$
\Gamma_{\boldsymbol{\phi}}^{\pi^{Q_{\overline{\boldsymbol{w}}}}}(\boldsymbol{w}) = \begin{bmatrix} Q \\ S \\ R \end{bmatrix} + \begin{bmatrix} A^2(G + 2IC + MC^2) \\ AB(G + 2IC + MC^2) \\ B^2(G + 2IC + MC^2) \end{bmatrix} = \boldsymbol{b} + \boldsymbol{A}^{\pi^{Q_{\overline{\boldsymbol{w}}}}} \cdot \boldsymbol{w} \overset{\text{or}}{=} \begin{bmatrix} Q \\ S \\ R \end{bmatrix} + \xi(\boldsymbol{w}) \begin{bmatrix} A^2 \\ AB \\ B^2 \end{bmatrix},
$$

with $\boldsymbol{A}^{\pi^{Q_{\overline{\boldsymbol{w}}}}} = \begin{bmatrix} A^2 & AB & B^2 \end{bmatrix}^T \begin{bmatrix} 1 & 2C & C^2 \end{bmatrix}$, $\boldsymbol{b} = [Q \quad S \quad R]^T$ and $\xi(\boldsymbol{w}) = \begin{bmatrix} 1 & 2C & C^2 \end{bmatrix} \boldsymbol{w}$. □

Remarks This result makes the evaluation of a greedy policy straightforward. One only needs to invert a matrix to get the value function corresponding to a greedy policy. Those parameters $\boldsymbol{w}^*$ satisfy the equation $\Gamma^{\pi^{Q_{\overline{\boldsymbol{w}}}}} Q_{\boldsymbol{w}^*}(s,a) = Q_{\boldsymbol{w}^*}(s,a)$, which means that $\Gamma_{\boldsymbol{\phi}}^{\pi^{Q_{\overline{\boldsymbol{w}}}}}(\boldsymbol{w}^*) = \boldsymbol{w}^*$ since $\Gamma^{\pi^{Q_{\overline{\boldsymbol{w}}}}} Q_{\boldsymbol{w}^*}(s,a) = Q_{\Gamma_{\boldsymbol{\phi}}^{\pi^{Q_{\overline{\boldsymbol{w}}}}}(\boldsymbol{w}^*)}(s,a)$. Thus, we have $Q^{\pi^{Q_{\overline{\boldsymbol{w}}}}} = Q_{(\boldsymbol{I}_3 - \boldsymbol{A}^{\pi^{Q_{\overline{\boldsymbol{w}}}}})^{-1}\boldsymbol{b}}$.
This operator can be understood in a geometrical way. By looking at the rightmost equation, we see that starting from any point in Ω, it projects the points along a line of direction vector $\begin{bmatrix} A^2 & AB & B^2 \end{bmatrix}^T$ with an offset $[Q \quad S \quad R]^T$. The iterations correspond to a linear transformation of the coefficient $\xi(\boldsymbol{w})$ in front of the direction vector. This means that all π-PBOs are iterating along the same line since the direction vector and the offset are not depending on the policy.

Proof for Section 4.4: existence of the optimal PBO for LQR MDP. Let $\boldsymbol{w} = (G, I, M) \in \mathbb{R}^2 \times \mathbb{R}_*^-$[6], be the optimal Bellman iteration over $Q_{\boldsymbol{w}}$ is $\Gamma^* Q_{\boldsymbol{w}}(s,a) = r(s,a) + \max_{a'} Q_{\boldsymbol{w}}(s',a')$. Before moving forward, we have to make sure this quantity is properly defined. The function $a' \mapsto Q_{\boldsymbol{w}}(s',a') = G \cdot s'^2 + 2I \cdot s'a' + M \cdot a'^2$ is a polynomial of degree 2 in a'. By choosing $M < 0$, we ensure that this function has a unique maximum of equation ${}^{-I}/_{M} \cdot s'$. This makes

$$
\max_{a'} Q_{\boldsymbol{w}}(s',a') = Q_{\boldsymbol{w}}(s', -\frac{I}{M} \cdot s') = G \cdot s'^2 - 2\frac{I^2}{M} \cdot s'^2 + \frac{I^2}{M} \cdot s'^2 = (G - \frac{I^2}{M}) \cdot s'^2.
$$

[5] For the case where $I = M = 0$, the greedy policy can be set to 0 everywhere since the value function is a constant with respect to the action a in that case. C is then set to 0 and the result still holds.

[6] If $\boldsymbol{w} \in \mathbb{R} \times \{0\} \times \{0\}$, the result still holds with the convention $0/0 = 0$.

By putting it in the Bellman equation, we get

$$\begin{aligned}\Gamma^* Q_{\boldsymbol{w}}(s,a) &= r(s,a) + (G - \frac{I^2}{M}) \cdot s'^2 \\ &= Q \cdot s^2 + 2S \cdot sa + R \cdot a^2 + (G - \frac{I^2}{M}) \cdot s'^2 \\ &= \left(Q + A^2(G - \frac{I^2}{M})\right) \cdot s^2 + 2\left(S + AB(G - \frac{I^2}{M})\right) \cdot sa \\ &+ \left(R + B^2(G - \frac{I^2}{M})\right) \cdot a^2.\end{aligned}$$

Then, the following operator is the optimal PBO

$$\Gamma^*_{\boldsymbol{\phi}}(\boldsymbol{w}) = \begin{bmatrix} Q + A^2(G - \frac{I^2}{M}) \\ S + AB(G - \frac{I^2}{M}) \\ R + B^2(G - \frac{I^2}{M}) \end{bmatrix} \overset{\text{or}}{=} \begin{bmatrix} Q \\ S \\ R \end{bmatrix} + \xi(\boldsymbol{w}) \begin{bmatrix} A^2 \\ AB \\ B^2 \end{bmatrix},$$

with $\xi(\boldsymbol{w}) = G - {}^{I^2}/_{M}$. We have $\Gamma^* Q_{\boldsymbol{w}} \in \mathcal{Q}_\Omega$, hence the stability of $\mathcal{Q}_\Omega$. □

Remarks The optimal PBO can also be understood in a geometrical way. Just like the π-PBO, it projects the parameters along a line of direction vector $\begin{bmatrix} A^2 & AB & B^2 \end{bmatrix}^T$ with an offset $\begin{bmatrix} Q & S & R \end{bmatrix}^T$. The only difference is that the iterations correspond to a non-linear transformation of the coefficient $\xi(\boldsymbol{w})$. This means that all π-PBOs and the optimal PBO are iterating on the same line. With this geometrical representation, we recover the fact that an optimal Bellman iteration on a value function corresponds to the Bellman iteration of the greedy policy of this value function. Since all the Bellman operators iterate on the same line, their fixed points are also on this line. Those fixed points correspond to the iteration of the optimal Bellman operator and we have just shown that the optimal Bellman operator iterates on this exact same line[7].

B Details of the baselines

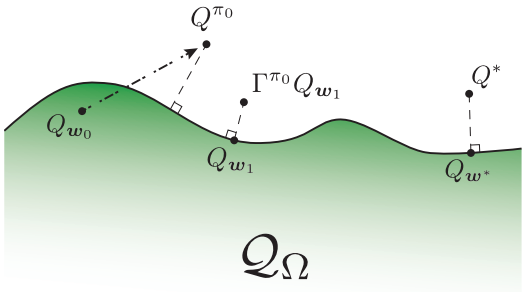


Figure 6: Behavior of LSPI in the space of all action-value functions for *one* iteration. The dotted lines are representing projections in the space of action-value functions.

[7]Here, we consider Bellman operators instead of the optimal PBOs because the iterations of those operators are equal. In the proofs we show $\Gamma^* Q_{\boldsymbol{w}} = Q_{\Gamma^*(\boldsymbol{w})}$ and $\Gamma^\pi Q_{\boldsymbol{w}} = Q_{\Gamma^\pi(\boldsymbol{w})}$.

Algorithm 2 FQI

Inputs:
  $\mathcal{D}$ # dataset of samples
  K # number of Bellman iterations
  fitting_steps
  λ # learning rate
Initialize:
  $\boldsymbol{w}_0$ # weight that give the null Q function
for i in $1 \dots$ K **do**
  # Compute the target T
  for $(s_j, a_j, r_j, s'_j) \in \mathcal{D}$ **do**
    $T(s_j, a_j) = r_j + \gamma \max_a Q_{\boldsymbol{w}_{i-1}}(s'_j, a)$
  end for

  # Minimize the loss starting from the previous weights
  $\boldsymbol{w}_i = \boldsymbol{w}_{i-1}$
  for _ in $1 \dots$ fitting_steps **do**
    for batch $\in \mathcal{D}$ **do**
      $\boldsymbol{w}_i \leftarrow \boldsymbol{w}_i - \lambda \sum_{j \in \text{batch}} \nabla_{\boldsymbol{w}} \left(T(s_j, a_j) - Q_{\boldsymbol{w}_i}(s_j, a_j)\right)^2$
    end for
  end for
end for
return $\boldsymbol{w}_K$

Algorithm 3 LSPI

Inputs:
  $\mathcal{D}$ # dataset of samples
  K # number of Bellman iterations
  $\boldsymbol{\sigma}$ # features
Initialize:
  $\boldsymbol{w}_0$ # weight that give the null Q function
for i in $1 \dots$ K **do**
  # Critic step: policy improvement
  for $(_, _, _, s'_j) \in \mathcal{D}$ **do**
    $\pi(s'_j) = \arg\max_a \boldsymbol{\sigma}(s'_j, a)^T \boldsymbol{w}_{i-1}$
  end for

  # Actor step: policy evaluation
  $\boldsymbol{A} \leftarrow 0$
  $\boldsymbol{b} \leftarrow 0$
  for $(s_j, a_j, r_j, s'_j) \in \mathcal{D}$ **do**
    $\boldsymbol{A} \leftarrow \boldsymbol{A} + \boldsymbol{\sigma}(s_j, a_j) \cdot \left(\boldsymbol{\sigma}(s_j, a_j) - \gamma \boldsymbol{\sigma}(s'_j, \pi(s'_j))\right)^T$
    $\boldsymbol{b} \leftarrow \boldsymbol{b} + r_j \boldsymbol{\sigma}(s_j, a_j)$
  end for
  $\boldsymbol{w}_i = \boldsymbol{A}^{-1} \cdot \boldsymbol{b}$
end for
return $\boldsymbol{w}_K$

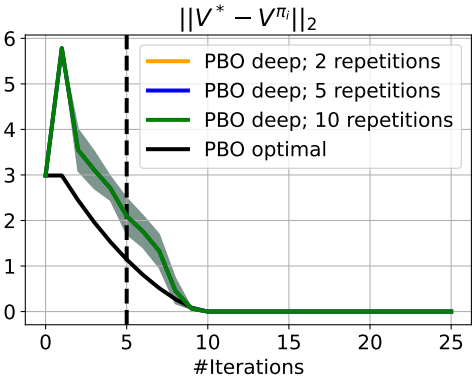
C ADDITIONAL RESULTS AND DETAILS OF THE EMPIRICAL ANALYSIS

We provide details of the experimental setting and additional empirical analysis to better understand the importance of some parameters. Table 2 summarizes the values of all parameters appearing in the experiments. For the additional empirical analysis, only the parameter at stake is changed, all

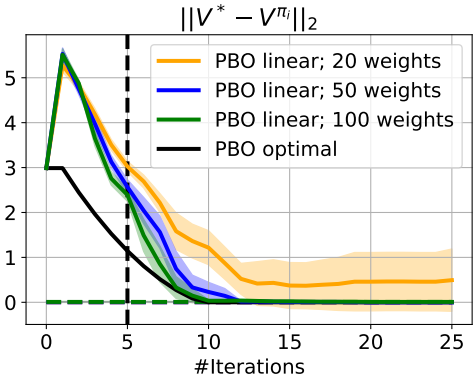
Table 2: Summary of all parameters used in the experiments.

		Chain-walk	LQR	Car-on-hill	Bicycle
horizon		$+\infty$	$+\infty$	100	50.000
γ		0.9	1	0.95	0.99
$\#\mathcal{D}$		$400^{(*)}$	$121^{(*)}$	5.000	70.000
batch size on $\mathcal{D}$		20	121	500	1.000
K		$5^{(*)}$	$2^{(*)}$	$9^{(*)}$	8
FQI	fitting step	400	800	1.200	1.200
	patience	-	-	7	7
	starting learning rate	10^{-2}	10^{-2}	10^{-3}	5×10^{-3}
	ending learning rate	10^{-5}	10^{-5}	10^{-5}	10^{-4}
PBOs	$\#\mathcal{W}$	$100^{(*)}$	1	1	1
	batch size on $\mathcal{W}$	100	1	1	1
	training step	400	3.200	4.000	500
	fitting step	4	1	1	15
	starting learning rate	5×10^{-3}	10^{-2}	10^{-3}	10^{-4}
	ending learning rate	10^{-5}	10^{-5}	5×10^{-6}	10^{-7}
	initial weight std	5×10^{-4}	$5 \times 10^{-4(*)}$	5×10^{-4}	5×10^{-9}

$^{(*)}$ Experimental analysis is available for these parameters.



(a) Performance for several sizes of samples dataset.



(b) Performance for several sizes of parameters dataset.

Figure 7: Analysis on the size of the different datasets given as input to PQI on chain-walk.

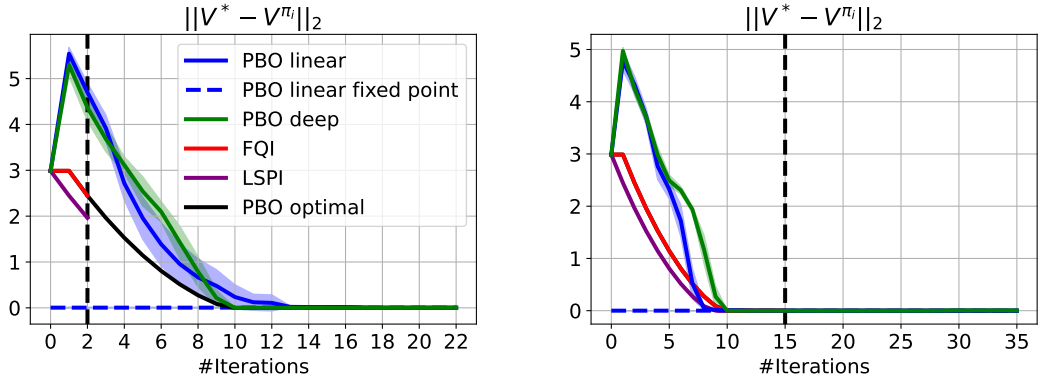
the others remain the ones shown in Table 2. FQI and PQI use an Adam optimizer (Kingma & Ba, 2015) with a linear annealing learning rate. For FQI, the optimizer is reset at each iteration.

C.1 CHAIN-WALK

We consider all the possible state-action pairs 10 times as the initial dataset of samples $\mathcal{D}$. These 10 repetitions help the algorithms to grasp the randomness of the environment. The set of parameters $\mathcal{W}$ is generated by taking a truncated normal distribution for different seeds. The validation parameter is a table composed of zeros. The following analyses show that the size of the dataset of samples is not important; conversely, the size of the dataset of parameters is crucial. The analysis also show that PQI behaves well when the number of Bellman iterations changes.

Analysis of the size of the dataset of samples $\mathcal{D}$. We consider `max-lin` operators, with the goal of studying how PQI behaves when the number of samples varies. The distance to the optimal value function is shown in Figure 7a. It can be seen that the distances are not visible, which means that the importance of this parameter is not high for this setting.

Analysis of the size of the dataset of parameters $\mathcal{W}$. Ideally, this dataset should contain parameters close to the initial parameter that we use in validation (the parameters with only zeros), others



(a) $K = 2$. (b) $K = 15$.

Figure 8: Analysis on the number of Bellman iterations K on chain-walk.

close to optimal parameters, and others in between. Since we only assume to know the validation parameters, we take parameters close to them, in the sense that the truncated normal distribution used for data collection is centered. Figure 7b shows the distance to the optimal value function for different sizes of $\mathcal{W}$. Clearly, the more parameters are given to the algorithm, the better it performs. This shows the importance of the number of parameters given to linear PBOs in this setting.

Analysis on the number of Bellman iterations K. We additionally analyze the behavior of PBOs for a different number of Bellman iterations than $K = 5$. We consider $K = 2$ and 15, observing that the PBOs still behave well as shown in Figure 8.

C.2 Linear quadratic regulator

The dynamics of the system is $\mathcal{P}(s, a) = 0.72s - 0.53a$ and the reward function is $\mathcal{R}(s, a) = -0.13s^2 + 0.26sa - 0.80a^2$. The set $\mathcal{D}$ is collected on a mesh of size 11 by 11 on the state-action space going from -4 to 4 in both directions, which means that samples belong to $[-4, 4] \times [-4, 4] \subset \mathcal{S} \times \mathcal{A}$. The set $\mathcal{W}$ is composed of only one weight which is the validation weight, i.e., $(0, 0, 0)$. This choice is made because during the optimization, the optimal Bellman operator might project the weights in an area where the coefficients M are positive which would lead the next optimal Bellman iterations to diverge. For PQI and FQI, we use the optimal Bellman operator, meaning that we should compute the maximum over a value function. We choose to parameterize the value functions in a quadratic way, thus allowing us to compute the maximum in closed-form. However, we assume that the algorithms does not have this knowledge since it is not the case in more general settings. For this reason, we discretize the action space in a set of 200 actions going from -8 to 8.

In the following, we explain how to obtain the metric used in Figure 4b. Then, we show that the size of $\mathcal{D}$ is not a crucial parameter, as we did for the chain-walk environment (see Appendix C.1). We also demonstrate that lowering the standard deviation of the initial parameters of PBO is important. The last analysis confirms that PQI behaves well when the number of Bellman iterations changes.

Metric. In this setting, the value functions associated to a greedy policy can be computed in closed form. For a weight $\boldsymbol{w} = (K, I, M) \in \mathbb{R}^2 \times \mathbb{R}_*^- \cup \mathbb{R} \times \{0\} \times \{0\}$ such that $A - {}^{BI}/_{M} \neq 1$, the value function associated to the greedy policy $\pi_{\boldsymbol{w}}$ computed from the value function $Q_{\boldsymbol{w}}$ has the following form, for any $s \in \mathcal{S}$:

$$V^{\pi_{\boldsymbol{w}}}(s) = P_{\boldsymbol{w}} \cdot s^2,$$

where $P_{\boldsymbol{w}} = \frac{Q - \frac{2SI}{M} + \frac{RI^2}{M^2}}{1 - (A - \frac{BI}{M})^2}$. Note that K does not influence the result, which is normal since the greedy policy is only influenced by the term linked with the actions.

Proof. Let $\boldsymbol{w} = (K, I, M) \in \mathbb{R}^2 \times \mathbb{R}_*^- \cup \mathbb{R} \times \{0\} \times \{0\}$ such that $A - {}^{BI}/_{M} \neq 1$, the greedy policy of the value function parameterized by $\boldsymbol{w}$ is $\pi^{\boldsymbol{w}}(s) = -{}^{I}/_{M} \cdot s$ where we take the convention ${}^{0}/_{0} = 0$. To show the result, we show that the function $s \mapsto P_{\boldsymbol{w}} \cdot s^2$ satisfies the optimal Bellman

equation[8]. We set $P_{\boldsymbol{w}} = \frac{Q - \frac{2SI}{M} + \frac{RI^2}{M^2}}{1-(A-\frac{BI}{M})^2}$, and we have

$$\begin{aligned}
&\forall s \in \mathcal{S}, P_{\boldsymbol{w}} \cdot s^2 = \Gamma^*(x \mapsto P_{\boldsymbol{w}} \cdot x^2)(s) \text{ optimal Bellman equation for the function } x \mapsto P_{\boldsymbol{w}} \cdot x^2. \\
&\iff \forall s \in \mathcal{S}, P_{\boldsymbol{w}} \cdot s^2 = r(s, \pi^{\boldsymbol{w}}(s)) + P_{\boldsymbol{w}} \cdot s'^2 \\
&\iff \forall s \in \mathcal{S}, P_{\boldsymbol{w}} \cdot s^2 = r(s, -\frac{I}{M}s) + P_{\boldsymbol{w}}(As - B\frac{I}{M}s)^2 \\
&\iff \forall s \in \mathcal{S}, P_{\boldsymbol{w}} \cdot s^2 = Qs^2 - 2S\frac{I}{M}s^2 + R\frac{I^2}{M^2}s^2 + P_{\boldsymbol{w}}(As - B\frac{I}{M}s)^2 \\
&\iff P_{\boldsymbol{w}} = Q - 2S\frac{I}{M} + R\frac{I^2}{M^2} + P_{\boldsymbol{w}}(A - B\frac{I}{M})^2 \\
&\iff P_{\boldsymbol{w}} = \frac{Q - \frac{2SI}{M} + \frac{RI^2}{M^2}}{1-(A-\frac{BI}{M})^2}.
\end{aligned}$$

Note that if we multiply s to both side of the assumption $A - {}^{BI}/_{M} \neq 1$, we get $s' \neq s$ where s' is the next state if the greedy policy is followed. This means that we cannot compute the value function of greedy policies that leads to steady states. This is normal since the value function is not defined for all the states because the sum of accumulated rewards diverges except in 0 where it has the value of 0. □

We leverage this result to state that the distance between the $P_{\boldsymbol{w}}$ of a weight $\boldsymbol{w}$ and the solution of the Riccati equation is proportional to the distance between the value function of the greedy policy and the optimal value function.

Proof. Let $b \in \mathbb{R}$. We consider the norm-2 over the functions going from$[-b, b]$ to $\mathbb{R}$. For $\boldsymbol{w} \in \mathbb{R}^2 \times \mathbb{R}_*^- \cup \mathbb{R} \times \{0\} \times \{0\}$, we have

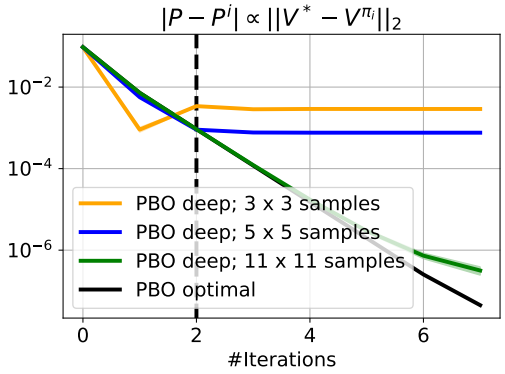
$$\begin{aligned}
||V^* - V^{\pi_{\boldsymbol{w}}}||_2 &= \sqrt{\int_{s\in[-b,b]} (V^*(s) - V^{\pi_{\boldsymbol{w}}}(s))^2 \mathrm{d}s} \\
&= |P - P_{\boldsymbol{w}}| \sqrt{\int_{s\in[-b,b]} s^4 \mathrm{d}s} \\
&= |P - P_{\boldsymbol{w}}| \cdot \sqrt{2\frac{b^5}{5}}.
\end{aligned}$$

The norm of the difference of the value functions is proportional to the difference of P coefficients. □

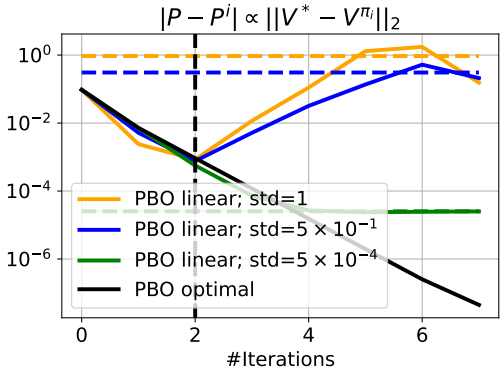
Analysis on the size of the dataset of samples $\mathcal{D}$. We focus on the Λ-`lin` PBO. Figure 9a shows the distance to the optimal value function for trainings with a varying number of samples. When the grid becomes thinner, the distances are decreasing. However this decrease is not strong. This is why the size of the dataset is not a crucial parameter for PBO.

Analysis on the standard deviation of the PBO's initial parameters. Figure 9b shows the distance to the optimal value function for linear PBOs that have been initialized with different standard deviations. The linear PBO initialized with smaller standard deviation shows better performance. This behavior can be explained by the following two reasons. First, a linear PBO is very limited in terms of freedom, meaning that if it is initialized in such a way that it goes to the wrong direction, it is very likely that it will stay in the wrong direction during the training; second, by initializing the weights of the PBO with low value, the operator is already contractive at the beginning of the training. This helps the PBO to have a stable behavior when evaluating on iterations that were not seen during training.

Analysis on the number of Bellman iterations K. Since the convergence of all the algorithms is fast in term of iterations, we only show the performance with 4 Bellman iterations instead of 2. If we increase the number of Bellman iterations too much, numerical errors arise due to the small distances. Figures 10a confirms that PBOs perform well for a larger number of Bellman iterations.

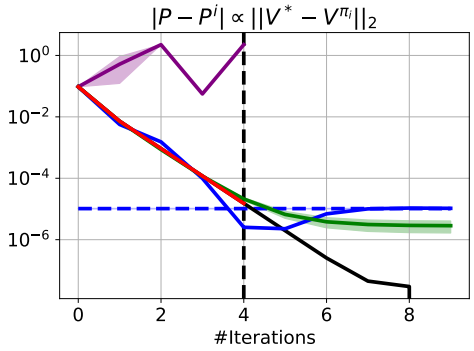


(a) Performance for different grid sizes.

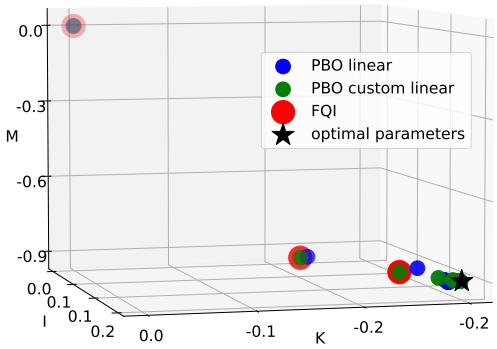


(b) Performance for initial parameters with different standard deviations.

Figure 9: Analysis on dataset $\mathcal{D}$ and on the standard deviation of the initial parameters on LQR.

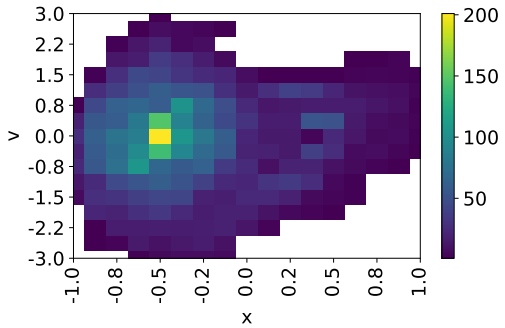


(a) Performance for different grid sizes..

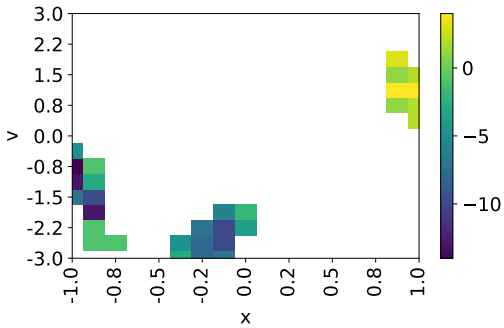


(b) Iterated parameters on the space Ω.

Figure 10: Analysis on number of Bellman iterations and visualization of iterated weights on LQR.

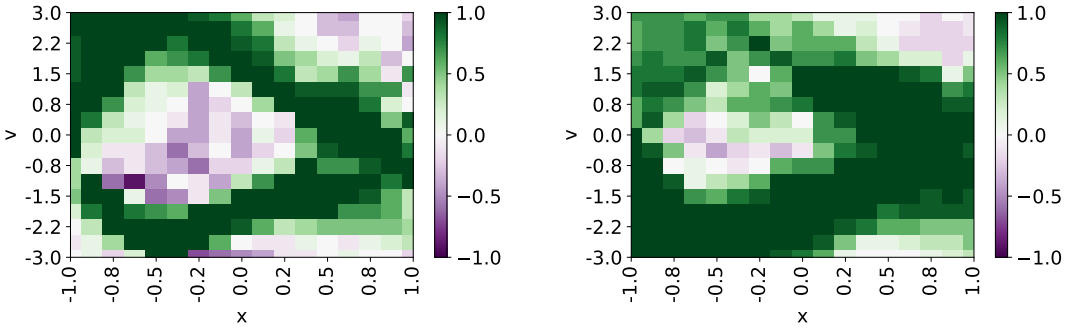


(a) Sample distribution.



(b) Reward distribution.

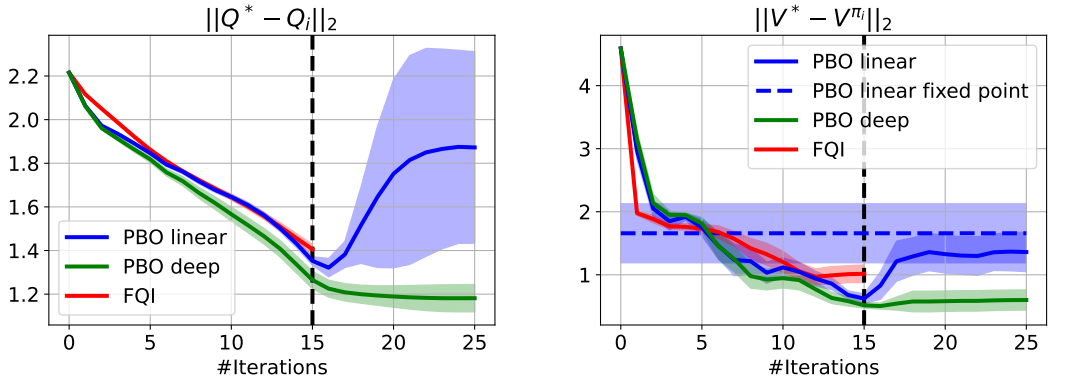
Figure 11: Composition of the dataset of samples $\mathcal{D}$ on car-on-hill.



(a) PQI's policy.

(b) FQI's policy.

Figure 12: Policies at iteration 9 on car-on-hill.



(a) Weighted distance between the optimal action-value function and the learned action-value function for every iteration.

(b) Weighted distance between the optimal value function and the learned value functions for every iteration.

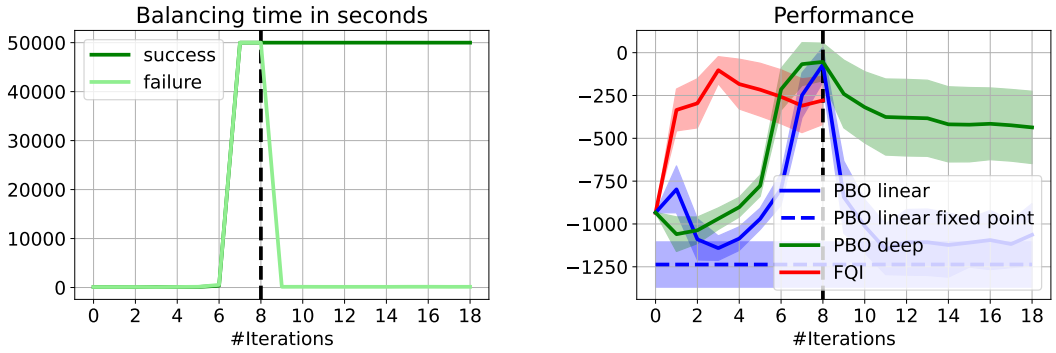
Figure 13: Performance with $K = 15$ on car-on-hill.

Visualization. This setting allow us to plot the parameters of the action-value functions in a 3D space. The parameters corresponding to the iterations of PQI and FQI are shown in Figure 10b. The training is done with $K = 2$. The first iteration for the three algorithms are on $(0, 0, 0)$ and then, the iterations goes towards the optimal parameters on the bottom right of the figure. In this case, being able to iterate after the training clearly makes the difference, as it is clear that PQI is able to get closer to the optimal parameters after the second iteration.

C.3 CAR-ON-HILL

The goal is to bring a car up a hill. The agent chooses between 2 actions: `left` or `right`. The state space is 2-dimensional: position in $[-1, 1]$ and velocity $[-3, 3]$. If the agent succeeds to bring the car up the hill – at position greater than 1 and velocity in between -3 and 3 – then the reward is 1, if the agent exceeds the state space, the reward is -1; otherwise, the reward is 0.
As our PQI is an offline algorithm, we need to make sure that all the necessary exploration has been done in the dataset of samples. For that reason, we first consider a uniform sampling policy to collect episodes starting from the lowest point in the map ($[-0.5, 0]$) with an horizon of 100. This sampling process is stopped when 4.500 samples are gathered. To get more samples with positive reward, we sample new episodes starting from a state located between $[0.1, 1.3]$ and $[0.5, 0.38]$ with a uniform policy as well. In total, 5.000 samples are collected. The sample and reward distributions over the state space is shown in Figure 11. The set of weights $\mathcal{W}$ is composed of only a single validation weight, because the complexity of the classes of PBOs used in the experiment is not high. The

[8]This only shows the result up to an additive constant, but since $\pi_{\boldsymbol{w}}(0) = 0$ then $V^{\pi_{\boldsymbol{w}}}(0) = 0$, so this constant is 0.



(a) Focus on two seeds with opposite outcomes.

(b) Performance of the algorithms.

Figure 14: Analysis for the Bicycle environment.

validation weight is a random weight where the weights for the last layer are only zeros, ensuring that the associated value function only outputs zeros. To help the training, the value functions are taking actions in $\{-1, 1\}$ instead of the usual $\{0, 1\}$. We also use a low standard deviation for the initial parameters of the PBOs as described in Appendix C.2.

Metric. Ernst et al. (2005) measure the performance of the agents by performing rollouts starting from different states located on a grid of size 17×17. Since the policies, the reward, and the dynamics, are deterministic, the discounted reward obtained at the end of the rollouts corresponds to the value function associated with the policy used during the rollouts. To compute the optimal value function, they run a brute force algorithm that tries every possible sequence of actions until it finds the shortest way to go up hill. In our case, the agents do not have all the transitions in $\mathcal{D}$ corresponding to every starting states of the grid they are using, as shown in Figure 11a. This is why we apply a mask on the grid to weight the starting states by the number of time a state has been visited in the dataset.

Policies. Figure 12 shows the policies for FQI and PQI `lin-max-lin`. Those policies reflect the average choice among the seeds. We believe that the reason why PQI `lin-max-lin` outperforms FQI is because, on average, the agent chooses to go more to the right when it is on the upper right corner of the state space. This can be seen in Figure 12a, where a darker green color is shown in that area. The agent has to choose action `right` to be able to climb up the hill.

Analysis on the number of Bellman iterations K. Figure 13 shows that PQI outperforms FQI at iteration 15, for $K = 15$. Similarly to the case $K = 9$, the linear PBO is too simple to behave well after the training iterations. Once again, the way the loss is shaped allows PQI to outperform FQI.

C.4 Bicycle

We consider the bicycle problem, as described in Randlov & Alstrøm (1998). The state space is composed of 4 dimensions: $(\omega, \dot{\omega}, \theta, \dot{\theta})$ where ω is the angle between the floor and the bike, and θ is the angle between the handle bar and the perpendicular axis to the bike. The goal is to ride a bicycle for 500 seconds (50.000 steps). The agent can apply a torque $T \in \{-2, 0, 2\}$ to the handle bar to make it rotate. The agent can also move its center of gravity in the direction $d \in \{-0.02, 0, 0.02\}$ perpendicular to the bike. As in Lagoudakis & Parr (2003), the agent chooses between applying a torque or moving its center of gravity, resulting in 5 actions instead of 9. Usually, a uniform noise in the interval $[-0.02, 0.02]$ is added to d. For the purpose of this work, we reduce the number of samples by making the magnitude of the noise 10 times smaller. A reward of -1 is given when the bike falls down, i.e., $|\omega| > 12°$. We also use reward shaping to have more informative samples, and we add a reward proportional to the change in ω, i.e., $10^4(|\omega_t| - |\omega_{t+1}|)$, as in Lagoudakis & Parr (2003). The dataset of samples is composed of 3.500 episodes starting from a position close to $(0, 0, 0, 0)$ and cut after 20 steps (Lagoudakis & Parr, 2003). To evaluate the value functions, we simulate 10 trajectories. We observed only two kinds of outcomes in our experiments: either the policies deteriorate the performance quickly, or they keep it stable, as shown in Figure 14a. This also explains the big confidence interval for deep PBO in Figure 5b. Moreover, Figure 14b shows the cumulative discounted reward obtained in the same experiment, where PQI outperforms FQI.