

Representation of Reinforcement Learning Policies in Reproducing Kernel Hilbert Spaces

Bogdan Mazoure^{*12}Thang Doan^{*12}Tianyu Li¹²Vladimir Makarenkov³Joelle Pineau¹²⁴⁵Doina Precup¹²⁵⁶Guillaume Rabusseau²⁵⁷

Abstract

We propose a general framework for policy representation for reinforcement learning tasks. This framework involves finding a low-dimensional embedding of the policy on a reproducing kernel Hilbert space (RKHS). The usage of RKHS based methods allows us to derive strong theoretical guarantees on the expected return of the reconstructed policy. Such guarantees are typically lacking in black-box models, but are very desirable in tasks requiring stability. We conduct several experiments on classic RL domains. The results confirm that the policies can be robustly embedded in a low-dimensional space while the embedded policy incurs almost no decrease in return.

1 Introduction

In the reinforcement learning (RL) framework, the goal of a rational agent consists in maximizing the expected rewards in a dynamical system by finding a suitable conditional distribution known as *policy*. This conditional distribution can be found using policy iteration and any suitable function approximator, ranging from linear models to neural networks [Sutton and Barto, 2018, Lillicrap et al., 2015]. Albeit neural networks

being the state-of-the-art learner for most performance-based tasks [Schulman et al., 2015, Bellemare et al., 2017], this class of blackbox models provide few guarantees on collected rewards, which should be imposed on a policy in risk-sensitive tasks [Garcia and Fernández, 2015].

In many application domains such as self-driving cars or robotic controllers [Sadigh et al., 2016, Akametalu et al., 2014, Berkenkamp et al., 2017, Katz et al., 2017], it is often crucial to have a robust policy. One criteria of such policy is that the variance of the expected returns should be as small as possible. Unfortunately, due to the non-convexity of most deep learning methods, it is difficult to compute the exact variance. Moreover, the high complexity of these models often leads to high variance and has been shown empirically [Pinto et al., 2017].

One approach to mitigate this issue is to represent complex policies using decision rules [Bastani et al., 2018, Katz et al., 2017, Aswani et al., 2013]. However, their major downsides are non-convex loss functions (preventing in-depth error analysis), and access to external oracle information (e.g. unbiased gradients).

In order to address the need for performance guarantees, we propose a principled way to represent a broad class of policy density functions as a point in a Reproducing Kernel Hilbert Space (RKHS). One advantage of such method is that it is straight-forward to truncate over the RKHS, which leads to a reduction in variance of the expected returns. In addition, the theoretical framework of RKHS offers neat tools for analyzing the error introduced by the truncation. In general, our framework achieves two desirable properties: (i) our method tends to produce policies with lower return variance than in the original policy; (ii) the embedding dimensionality of the studied policies can vary without large performance drops. We show both of these properties theoretically as well as empirically.

The use of RKHS and kernel methods in reinforcement

¹McGill University ²Mila - Quebec Artificial Intelligence Institute ³Université du Québec à Montréal ⁴Facebook AI Research ⁵CIFAR AI chair ⁶DeepMind ⁷Université de Montréal. ^{*}Equal contribution. Correspondence to: Bogdan Mazoure <bogdan.mazoure@mail.mcgill.ca>

learning problems is not uncommon. Non-parametric kernel density estimators have previously been used to represent fundamental RL quantities such as the transition dynamics [Grunewalder et al., 2012, Nishiyama et al., 2012, Lever et al., 2016, Barreto et al., 2016] and predictive representations of states [Littman and Sutton, 2002] known as PSRs. Moreover, some works leverage spectral learning and RKHS to extend the classical PSR setting [Boots et al., 2013, Li et al., 2020]. One can even recover the original distribution from the kernel mean estimator through the tools provided by Kanagawa and Fukumizu [2014], which turns out to be useful in message passing [Song et al., 2008].

Our contributions are the following:

- We propose a decomposition of policy densities within a separable Hilbert space and derive a set of performance bounds which, when used together, provide a heuristic to pick the dimensionality of the embedding.
- The truncation of the RKHS embedding is shown to reduce variance of returns with respect to the initial conditions, when second order moment conditions are satisfied.
- We interpret the performance error bounds through three separate terms: truncation (Thm. 3.4), discretization (Thm. 3.5), and pruning (Thm. 3.6).
- Empirical evidence on continuous control tasks supports our theoretical findings.

To the best of our knowledge, this is the first work proposing a general framework to learn an RKHS policy embedding with performance and long-term behaviour guarantees.

2 Preliminary

In this section, we provide a brief introduction to reinforcement learning, density approximation, as well as singular value decomposition.

2.1 Reinforcement learning

We consider a problem modeled by a discounted Markov Decision Process (MDP) $\mathcal{M} := \langle \mathcal{S}, \mathcal{A}, r, P, S_0, \gamma \rangle$, where \mathcal{S} is the state space; \mathcal{A} is the action space; given states $s, s' \in \mathcal{S}$, action $a \in \mathcal{A}$, $\mathbb{P}(s'|s, a)$ is the transition probability of transferring s to s' under the action a and $r(s, a)$ is the reward collected at state s after executing the action a ; $S_0 \subseteq \mathcal{S}$ is the set of initial states and γ is the discount factor. Through the paper, we assume that r is a bounded function.

The agent in RL environment often execute actions according to some policies. We define a stochastic policy $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ such that $\pi(a|s)$ is the conditional probability that the agent takes action $a \in \mathcal{A}$ after observing the state $s \in \mathcal{S}$. The objective is to find a policy π which maximizes the expected discounted reward:

$$\eta(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right]. \quad (1)$$

The state value function V_π and the state-action value function Q_π are defined as:

$$V_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=t}^{\infty} \gamma^{k-t} r(s_k, a_k) \middle| s_t = s \right],$$

$$Q_\pi(s, a) = \mathbb{E}_\pi \left[\sum_{k=t}^{\infty} \gamma^{k-t} r(s_k, a_k) \middle| s_t = s, a_t = a \right]$$

The gap between Q_π and V_π is known as the advantage function:

$$A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s),$$

where $a \sim \pi(\cdot|s)$. The coverage of policy π is defined as the stationary state visitation probability $\rho_\pi(s) = \mathbb{P}(s_t = s)$ under π .

2.2 Function approximation in inner product spaces

The theory of inner product spaces has been widely used to characterize approximate learning of Markov decision processes [Puterman, 2014, Tsitsiklis and Van Roy, 1999]. In this subsection, we provide a short overview of inner product spaces of square-integrable functions on a closed interval I denoted $L^2(I)$.

The $L^2(I)$ space is known to be separable for $I = [0, 1]$, i.e., it admits a countable orthonormal basis of functions $\{\omega_k\}_{k=1}^\infty$, such that $\langle \omega_i, \omega_j \rangle = \delta_{ij}$ for all i, j and δ being Kronecker's delta. This property makes possible computations with respect to the inner product

$$\langle f, g \rangle = \int_{t \in I} f(t) \bar{g}(t) dt, \quad (2)$$

and corresponding norm $\|f\| = \sqrt{\langle f, f \rangle}$, for $f, g \in L^2(I)$.

That is, for any $f \in L^2(I)$, there exist scalars $\{\xi_k^f\}_{k=1}^\infty$ such that its representation in the RKHS

$$\hat{f}(t) = \sum_{k=1}^{\infty} \xi_k^f \omega_k(t), \quad \xi_k^f = \langle f, \omega_k \rangle. \quad (3)$$

If $\hat{f}_K(t) = \sum_{k=1}^K \xi_k^f \omega_k(t)$, then adding $\xi_{K+1} \omega_{K+1}$ to \hat{f}_K can be thought of as adding a new orthogonal basis.

Eq. equation 3 suggests a simple embedding rule, where one would project a density function onto a fixed basis and store only the first $\{\xi_k^f\}_{k=1}^K$ coefficients. Which components to pick will depend on the nature of the basis: harmonic and wavelet bases are ranked according to amplitude, while singular vectors are sorted by corresponding singular values.

The choice of the basis $\{\omega_k\}_{k=1}^\infty$ plays a crucial role in the quality of approximation. The properties of the function f (e.g. periodicity, continuity) should also be taken into account when picking ω . Some examples of well-known orthonormal bases of $L^2(I)$ are *Fourier* $\omega_k(t) = e^{2\pi i k t}$ [Stein and Shakarchi, 2003] and *Haar* $\omega_{k,k'}(t) = 2^{k/2}\omega(2^k t - k')$ [Haar, 1909]. Other examples include but are not limited to Hermite and Daubechies bases [Daubechies, 1988]. Moreover, it is known that a mixture model with countable number of components is a universal density approximator [McLachlan and Peel, 2004]; in such case, the basis consists of Gaussian density functions and is not necessarily orthonormal.

Moreover, matrix decomposition algorithms such as the truncated singular value decomposition (SVD) are popular projection methods due to their robustness and extension to finite-rank bounded operators [Zhu, 2007], since they learn both the basis vectors and the corresponding linear weights. Although SVD has previously been used in embedding schemes [Lu et al., 2017, Goetschalckx et al., 2018], unlike us, most works apply it on the weights of the function rather than the outputs. Similarly to the density approximators above, a key appeal of SVD is that the error rate of truncation can be controlled efficiently through the magnitude of the minimum truncated singular values¹.

While convergence guarantees are mostly known for the closed interval $I = [0, 1]$, multiple works have studied properties of the previously discussed basis functions on the whole real line and in multiple dimensions [Egozcue et al., 2006]. Weaker convergence results in Hilbert spaces can be stated with respect to the space's inner product. If, for every $g \in L^2(I)$ and sequence of functions $f_1, \dots, f_n \in L^2(I)$,

$$\lim_{n \rightarrow \infty} \langle f_n, g \rangle \rightarrow \langle f, g \rangle, \quad (4)$$

then the sequence f_n is said to converge weakly to f as $n \rightarrow \infty$. Stronger theorems are known for specific bases but require stronger assumptions on f .

In order to allow our learning framework to have strong convergence results in the inner product sense as well as a countable set of basis functions, we restrict ourselves to separable Hilbert spaces.

¹see Eckart-Young-Mirsky theorem

3 Framework

In this section, we first introduce a framework to represent policies in an RKHS with desirable error guarantees. The size of this embedding can be varied through the coefficient truncation step. We show that this truncation has the property to reduce the return variance without drastically affecting the expected return. Next, we propose a discretization and pruning steps as a tractable alternative to working directly in the continuous space. Finally, we derive a practical algorithm which leverages all aforementioned steps.

3.1 Formulation as RKHS problem

By Mercer's theorem [Mercer, 1909], any symmetric positive-definite kernel function κ with associated integral operator T_κ can be decomposed as:

$$\begin{aligned} \kappa(x, y) &= \sum_{k=1}^{\infty} (\sqrt{\lambda_k} e_k(x)) (\sqrt{\lambda_k} e_k(y)) \\ &= \sum_{k=1}^{\infty} \omega_k(x) \omega_k(y), \end{aligned} \quad (5)$$

where λ_k and e_k are the eigenvalues and eigenfunctions of T_κ , respectively.

It follows by Moore–Aronszajn theorem [Aronszajn, 1950] that there exists a unique Hilbert space \mathcal{H} of functions for which κ is the kernel.

Moreover, the inner product associated with \mathcal{H} is:

$$\langle f, g \rangle_{\mathcal{H}} := \sum_{k=1}^{\infty} \frac{\langle f, e_k \rangle_{L_2} \langle g, e_k \rangle_{L_2}}{\lambda_k} = \sum_{k=1}^{\infty} \frac{\xi_k^f \xi_k^g}{\lambda_k}. \quad (6)$$

This allows us to state performance bounds between policies embedded in an RKHS in terms of their projection weights, as shown in the next section.

Consider the conditional distribution $\pi : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$. For a fixed s , the function $\pi(\cdot|s)$ belongs to a Hilbert space \mathcal{H}_s of functions $f : \mathcal{A} \rightarrow \mathbb{R}$.

Lemma 3.1. *Let s be a state in \mathcal{S} . Let \mathcal{H}_s be an RKHS, the integral operator of which has eigenvalues $\{\lambda_k\}_{k=1}^\infty$ and eigenfunctions $\{e_k\}_{k=1}^\infty$. Let $\xi_k^{\pi_i} = \langle \pi_i, e_k \rangle$ and $\pi_1, \pi_2 \in \mathcal{H}_s$ such that $\pi_1 = \pi_1(\cdot|s)$, $\pi_2 = \pi_2(\cdot|s)$. Then, the following holds*

$$\|\pi_1 - \pi_2\|_{\mathcal{H}_s}^2 = \sum_{k=1}^{\infty} \frac{(\xi_k^{\pi_1} - \xi_k^{\pi_2})^2}{\lambda_k}. \quad (7)$$

Lemma 3.2. *Let $(\xi_1^{\pi_1}, \xi_2^{\pi_1}), \dots, (\xi_K^{\pi_1}, \xi_K^{\pi_2})$ be the projection weights of π_i onto \mathcal{H}_s such that, for $p, q \in \mathbb{N}^+$, if $p > q$ then $\xi_p^{\pi_i} < \xi_q^{\pi_i}$. If there exists a $K \in \mathbb{N}^+$ such*

that for all $k > K$, $|\xi_k^{\pi_1} - \xi_k^{\pi_2}| < \varepsilon^k \sqrt{\lambda_k}$ for some real $\varepsilon > 0$, then the following holds

$$\sum_{k=K+1}^{\infty} \frac{(\xi_k^{\pi_1} - \xi_k^{\pi_2})^2}{\lambda_k} \leq \frac{\varepsilon^{2(K+1)}}{1 - \varepsilon^2}. \quad (8)$$

The truncated embedding of size K can be formed by setting the sequence of coefficients $\{\xi_k^{\pi}\}_{k=K}^{\infty}$ to zero, obtaining

$$\hat{\pi}_K(t) = \sum_{k=1}^K \xi_k \omega_k(t). \quad (9)$$

As shown in Experiments 7.5, truncating the embedding at a given rank can have desirable properties on the returns and on their variance.

3.2 Truncating RKHS embeddings can reduce variance of returns

In this subsection, we show under which conditions one can expect a reduction in variance of returns of the RKHS policy, thus helping in sensitive tasks such as medical interventions or autonomous vehicles [Garcia and Fernández, 2015].

Any policy π in the RKHS can be decomposed as a finite sum:

$$\pi(a|s) = \underbrace{\sum_{k=1}^K \xi_k \omega_k(s, a)}_{\hat{\pi}_K} + \underbrace{\sum_{k=K}^{\infty} \xi_k \omega_k(s, a)}_{\varepsilon_K}, \quad \forall s, a \in \mathcal{S} \times \mathcal{A}. \quad (10)$$

Our framework allows to derive variance guarantees by first considering the *random return* variable, Z of the policy π given some state and action:

$$Z(\pi|s_0, a_0) = r(S_0, A_0) + \sum_{t=1}^{\infty} \gamma^t r(S_t, A_t), \quad (11)$$

with $S_t \sim P(\cdot|s_{t-1}, a_{t-1})$, $A_t \sim \pi(\cdot|s_t)$ and $S_0, A_0 \sim \beta$ for some initial distribution β .

Suppose we have access to some entry-wise normalization function $\sigma : \mathbb{R} \rightarrow [0, 1]$. We are not concerned with the exact form of σ , but, as an example, we use the softmax function $\sigma(x_i) = \frac{e^{x_i}}{\sum_{j=1}^d e^{x_j}}$.

The variance of $Z(\pi)$ over the entire trajectory is hard to compute [Tamar et al., 2016] since it requires to solve a Bellman equation for $\mathbb{E}[Z^2(\pi)]$. Instead, we look at the variance over initial state-action pairs sampled from β . Since we do not know the distribution of $\pi(a|s)$ but only that of (a, s) under β , we compute all expectations using the Law of the Unconscious Statistician [Ringnér, 2009, LOTUS].

Lemma 3.3. Let $\mathbb{V}_{\beta}[X] = \mathbb{E}_{\beta}[X^2] - (\mathbb{E}_{\beta}[X])^2$ be the variance operator with respect to β , and let σ be a positive-value, differentiable function. If the following conditions are satisfied

1. $\sigma'(\eta(\hat{\pi}_K)) \leq \sigma'(\eta(\pi))$ (monotonically decreasing σ' on $(0, \infty)$);
2. $\sqrt{\frac{\mathbb{E}_{\beta}[\varepsilon_K^2]}{3}} \geq \mathbb{E}_{\beta}[\varepsilon_K]$ (second moment condition);
3. $\mathbb{V}_{\beta}[\pi] \approx (\sigma'(\eta(\pi)))^2 \mathbb{V}_{\beta}[Q^{\pi}]$ (Second order Taylor residual is small),

then the following holds:

$$\mathbb{V}_{\beta}[Q^{\pi}] \geq \mathbb{V}_{\beta}[Q^{\hat{\pi}_K}] \quad (12)$$

Lemma 3.3 tells us the relation between the variance of returns collected by either the true or truncated policies (detailed analysis in Appendix).

In practice, Condition 1 is satisfied when σ' is decreasing on $(0, \infty)$, rewards are strictly positive and $\eta(\hat{\pi}_K) \leq \eta(\pi)$. Condition 2 is not restrictive, and is, for example, satisfied by the Student's t distribution with $\nu > 1$ degrees of freedom. Condition 3 is a classical assumption in variance analysis [Benaroya et al., 2005] and holds if the expansion is done near the expectation of the Q function. As we show in the next section, even after the truncation step, our framework still provides performance important guarantees.

3.3 Truncation bound

Storing an approximation of the ground truth policy implies a trade-off between performance and number of basis functions. Our framework allows to control the difference in collected reward using the following theorem of projecting π onto the first K basis functions.

Theorem 3.4. Let $s \in \mathcal{S}$ and \mathcal{H}_s be the associated RKHS. Let $\pi_1, \pi_2 \in \mathcal{H}_s$ be represented by $\{\xi_k^{\pi_1}\}_{k=1}^{\infty}$ and $\{\xi_k^{\pi_2}\}_{k=1}^{\infty}$ and let ε such that Lemma 3.2 holds. Let $M_s > 0$ be such that $|\pi_1(a|s) - \pi_2(a|s)| \leq M_s \|\pi_1 - \pi_2\|_{\mathcal{H}}$ for all $a \in \mathcal{A}$. Let $\Delta_{\mathcal{H}_s}^K = \sum_{k=1}^K \frac{(\xi_k^{\pi_1} - \xi_k^{\pi_2})^2}{\lambda_k}$ and $\bar{\epsilon}_i = \max_{s,a} |A_{\pi_i}(s, a)|$. Then

$$|\eta(\pi_2) - \eta(\pi_1)| \leq \frac{4|\mathcal{A}|^2 \bar{\epsilon} \gamma}{(1 - \gamma)^2} \left\{ \begin{aligned} & \max_{s \in \mathcal{S}} (M_s \Delta_{\mathcal{H}_s}^K)^2 + O(\varepsilon^{2K} \max_{s \in \mathcal{S}} M_s \Delta_{\mathcal{H}_s}^K) \\ & + \max(\bar{\epsilon}_1, \bar{\epsilon}_2) \end{aligned} \right\}$$

This theorem implies that representing any policy within the top K bases of the Hilbert space yields an approximation error polynomial in ε due to how the linear coefficients were picked. Since $M_s = \|\kappa(\cdot, s)\|_{\mathcal{H}} \leq \sup_{s \in \mathcal{S}} \|\kappa(\cdot, s)\|_{\mathcal{H}} < \infty$ when all function in \mathcal{H} are bounded (see Sun and Wu [2009]), the right hand side of Theorem 3.4 is finite.

While Theorem 3.4 holds in the continuous case, most projection algorithms such as Fast Fourier Transform, wavelet transform and SVD operate on a discretized version of the function. The next section describes the error made during this discretization step.

3.4 Discretization error bound

So far, the theoretical formulation of our algorithm was in the continuous space of states and actions. However, most efficient algorithms to project a function onto an orthonormal basis operate in the discrete space [Stein and Shakarchi, 2003, Daubechies, 1988]. For this reason, we introduce the discretization step, which allows to leverage these highly optimized methods without sacrificing the approximation guarantees from the continuous domain.

An important step in our algorithm is the component-wise grouping of similar states. This step is crucial, since it allows us to greatly simplify the calculations of the error bounds and re-use existing discrete projection algorithms such as Fast Fourier Transform. However, when the discretization is done naively, it can result in slower computation times due to a blowup in state-action space dimensions (to mitigate this, we propose the pruning step in next section). We use an approach known as *quantile binning* [Naeini et al., 2015]. We assume that state components assigned to the same bin have a similar behaviour, a somewhat limiting condition which is good enough for simple environments and greatly simplifies our proposed algorithm.

Recall that the cumulative distribution function $\Pi_X(x) = \mathbb{P}[X \leq x]$ and the corresponding quantile function $\mathcal{Q}_X(p) = \inf\{x \in \mathbb{R} : p \leq \Pi_X(x)\}$. In practice, we approximate \mathcal{Q}, Π with the empirical quantile and density functions $\hat{\mathcal{Q}}, \hat{\Pi}$, respectively.

Theorem 3.5. *Let Π_S, Π_A be cumulative policy functions over \mathcal{S}, \mathcal{A} and let $[\Pi]_S, [\Pi]_A$ be their empirical estimates discretized using quantile binning over states and actions into b_S and b_A clusters, respectively. Let $\mathcal{Q}_S, \mathcal{Q}_A$ be the corresponding empirical quantile functions. Then, the volume of the discretization error can*

be written as

$$\delta_{\Pi, [\Pi]} = \sum_{i=1}^{b_S} \sum_{j=1}^{b_A} \left\{ \left| \int_{q_{i-1}^S}^{q_i^S} [\hat{\Pi}]_S(s) ds - \frac{i}{b_S} (q_i^S - q_{i-1}^S) \right| \right. \\ \left. \left| \int_{q_{i,j-1}^A}^{q_{i,j}^A} [\hat{\Pi}]_A(a) da - \frac{j}{b_A} (q_{i,j}^A - q_{i,j-1}^A) \right| \right\}, \quad (13)$$

where $q_i^S = \lceil \hat{\mathcal{Q}}_S \rceil \left(\frac{i}{b_S} \right)$ and $q_{i,j}^A = \lceil \hat{\mathcal{Q}}_{A|i} \rceil \left(\frac{j}{b_A} \right)$.

Note that $\delta_{\Pi, [\Pi]}$ is similar to $\|\Pi, [\Pi]\|_1$, with the only difference that the volume of each error rectangle is considered. Therefore, an argument identical to Theorem 3.4 can be used to map Theorem 3.5 into the error space of returns.

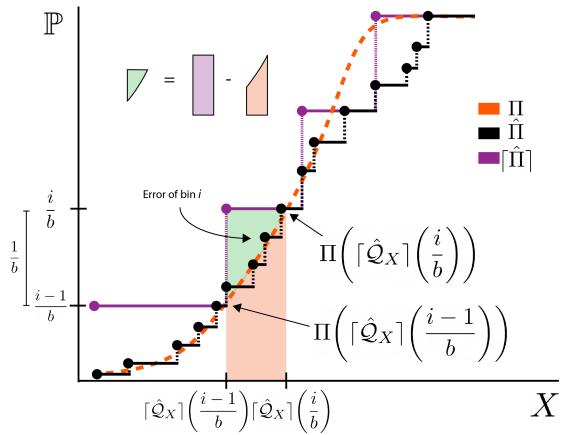


Figure 1: Schematic view of the discretization error

Theorem 3.5 and Figure 1 show how the discretization error induced by Algorithm 1 can be computed across all states and actions in the training set. Note that samples used to compute both distribution functions are i.i.d. inter-, but not intra-trajectories. This means that the classical result from Dvoretzky et al. [1956] might not hold.

The discretization step enables us to use powerful discrete projection algorithms, but as a side effect can drastically expand the state space. However, we observed empirically that most policies tend to cover a small subset of the state space. Using this information, we introduce a state-space pruning method based on the steady state distribution, which is addressed in the followed section.

3.5 Pruning error bound

The discretization step from the previous section greatly simplifies the calculations of the projection weights. However, it can potentially make the new state-action space quite large. In this section, we introduce the

pruning step based on the long-run distribution of the policy, and then quantify the impact of removing unvisited states both on the performance and coverage of the policy.

3.5.1 Pruning unvisited states

Since policies tend to concentrate around the optimum as training progresses [Ahmed et al., 2018], pruning those states would not significantly hinder the overall performance of the embedded policy. Let $\rho_\pi(s)$ denote the number of times π visits state s in the long run and $N(s, a)$ the number of times action a is taken from s over N trajectories.

This leads us to define π_{pruned} as a policy which acts randomly under some state visitation threshold. Precisely, if $S_0 = \{s : \rho_\pi(s) = 0\}$ and $\mathbb{1}(\cdot)$ is the indicator function, then

$$\pi_{\text{pruned}}(a|s) = \{1 - \mathbb{1}_{S_0}(s)\} \frac{N(s, a)}{\rho_\pi(s)} + \mathbb{1}_{S_0}(s) \frac{1}{b_A}. \quad (14)$$

Pruning rarely visited states can drastically reduce the number of parameters, while maintaining high probability performance guarantees for π_{pruned} . The following theorem quantifies the effect of pruning states on the performances of the policy.

Theorem 3.6 (Policy pruning, Simão et al. [2019]). *Let N be the number of rollout trajectories of the policy π , r_M be the largest reward. With probability $1 - 2\delta$, the following holds:*

$$|\eta(\pi) - \eta(\pi_{\text{pruned}})| \leq \frac{2r_M}{1 - \gamma} \sqrt{\frac{3|\mathcal{S}||\mathcal{A}| + 4 \log \frac{1}{\delta}}{2N}}. \quad (15)$$

This theorem allows one to discretize only the subset of frequently visited states while still ensuring a strong performance guarantee with high probability.

Instead of pruning based on samples from $\rho_{[\pi]}$, which induces a computational overhead since we need to perform an additional set of rollouts, we instead prune states based on ρ_π . The approximation error induced by this switch can be understood through the scope of Thm. 3.7.

3.5.2 Impact of pruning on state visitation

Any changes done to the ground truth policy such as embedding it into an RKHS will affect its stationary distribution and hence the long-run coverage of the policy. In this subsection, we provide a result on the error made on the stationary distributions as a function of error made in the original policies.

Under a fixed policy π , an MDP induces a Markov chain defined by the *expected transition model* $(\mathbf{P}_\pi)_{ss'} = \sum_{a \in \mathcal{A}} \pi(a|s) P(s'|s, a)$. In tensor form, it can be represented as $\mathbf{A}(\mathbf{T}_{(2)} \boldsymbol{\Pi})$, where $\boldsymbol{\Pi}_{as} = \pi(a|s)$ is the policy matrix, $\mathbf{A} = \mathbf{I} \odot \mathbf{I}$ is the Khatri-Rao product and $\mathbf{T}_{(2)}$ is the second matricization of the transition tensor $\mathbf{T}_{sas'} = P(s'|s, a)$.

If the Markov chain defined above is irreducible and homogeneous, then its *stationary distribution* corresponds to the state occupancy probability² given by the principal left eigenvector of \mathbf{P}_π .

The following theorem bridges the error made on the reconstruction of long-run distribution as a function of the system's transition dynamics and distance between policies.

Theorem 3.7 (Approximate policy coverage). *Let $\|\cdot\|_{S_p}$ be the Schatten p -norm and $\|\cdot\|_p$ be the vector p -norm. If $(\mathbf{P}_\pi, |\mathcal{S}|^{-1}\mathbf{1}_{|\mathcal{S}|})$ and $(\mathbf{P}_{\pi_{\text{pruned}}}, |\mathcal{S}|^{-1}\mathbf{1}_{|\mathcal{S}|})$ are irreducible, homogeneous Markov chains, then the following holds:*

$$\|\rho_\pi^\top - \rho_{\pi_{\text{pruned}}}^\top\|_{S_1} \leq \|\mathbf{Z}\|_{S_\infty} \|\boldsymbol{\Pi} - \boldsymbol{\Pi}_{\text{pruned}}\|_{S_1} \|\mathbf{T}_{(3)}\|_{S_2}, \quad (16)$$

where $\mathbf{Z} = (\mathbf{I} - \mathbf{P}_\pi + \mathbf{1}_{|\mathcal{S}|}\rho_\pi^\top)^{-1}$ and $\mathbf{1}_{|\mathcal{S}|}$ is a vector of all ones.

A detailed proof can be found in the Appendix. Note that the upper bound depends on both the environment's structure as well as on the policy reconstruction quality. It is thus expected that, for MDPs with particularly large singular values of $\mathbf{T}_{(2)}$, the bound converges less quickly than for those with smaller singular values. See Appendix for visualizations of the bound on empirical domains.

Equiped with these tools, we are ready to state the general policy embedding bound in the RKHS setting.

3.6 General policy embedding bound

We are finally ready to use the previous performance bounds to state the general performance result. The difference in performance of ground-truth policy and truncated embedded policy can be decomposed as a discretization error, a pruning error and a projection error:

Corollary 3.7.1. *Given ground truth policy π , discretized policy $[\pi]$, pruned policy π_{pruned} and embedded*

²The existence and uniqueness of ρ follow from the Perron-Frobenius theorem.

policy $\hat{\pi}$, the total policy embedding bound is

$$\begin{aligned} |\eta(\pi) - \eta(\hat{\pi})| &\leq \underbrace{|\eta(\pi) - \eta([\pi])|}_{\text{Theorem 3.5}} + \underbrace{|\eta([\pi]) - \eta(\pi_{\text{pruned}})|}_{\text{Theorem 3.6}} \\ &+ \underbrace{|\eta(\pi_{\text{pruned}}) - \eta(\hat{\pi})|}_{\text{Theorem 3.4}}. \end{aligned}$$

Tighter bounds of Thm. 3.4 can be found in the literature Giardina and Chirlian [1972], Rakhlin et al. [2005] for specific basis functions.

In the next section, we propose a practical algorithm which integrates all three steps.

4 Practical algorithm

We first discuss the construction of $[\pi]$ via quantile discretization. The idea consists in grouping together similar state-action pairs (in term of visitation frequency). To this end, we use the quantile state visitation function Q_S and the state visitation distribution function Π_S , as well as their empirical counterparts, \hat{Q}_S and $\hat{\Pi}_S$. Quantile and distribution functions for the action space are defined analogously.

Algorithm 1: Quantile discretization

Input: Policy π , number of state bins b_S , number of action bins b_A

Result: Discrete policy $[\pi]$

Collect a set of states $S \subseteq \mathcal{S}$ and set of actions $A \subseteq \mathcal{A}$ via rollouts of π ;

Build the empirical c.d.f. $\hat{\Pi}_S$ from set S ;

Build the empirical c.d.f. $\hat{\Pi}_A$ from set A ;

Find numbers i_1, \dots, i_{b_S} s.t. $\hat{\Pi}_S(i_l) - \hat{\Pi}_S(i_{l-1}) = \frac{1}{b_S}$ using \hat{Q}_S for all $s \in S, l = 1, \dots, b_S$;

Find numbers j_1, \dots, j_{b_A} s.t. $\hat{\Pi}_A(j_l) - \hat{\Pi}_A(j_{l-1}) = \frac{1}{b_A}$ using \hat{Q}_A for all $a \in A, l = 1, \dots, b_A$;

if $i_{l-1} \leq s \leq i_l$ and $j_{l'-1} \leq a \leq j_{l'}$ **then**

| Assign (s, a) to (l, l') ;

end

Set $[\pi]_{ll'}$ to $\pi(\frac{i_{l'-1}+i_l}{2}, \frac{j_{l'-1}+j_l}{2})$

Algorithm 1 outlines the proposed discretization process allowing one to approximate any continuous policy (e.g., computed by a neural network) by a 2-dimensional table indexed by discrete states and actions. We use quantile discretization in order to have maximal resolution in frequently visited areas. It also allows for slightly faster sampling during rollouts, since the probability of falling in each bin is uniform.

Algorithm 2: RKHS policy embedding

Input: Policy π , number of components K , basis $\omega = \{\omega_k\}_{k=1}^\infty$

Result: A set of coefficients ξ_1, \dots, ξ_K

1. Rollout π in the environment to estimate \hat{Q}_s, \hat{Q}_a (empirical quantile functions);
 2. Project π onto lattice to obtain $[\pi]$ using Alg. 1;
 3. Prune $[\pi]$ with \hat{Q}_s, \hat{Q}_a using Eq. 14 ;
 4. Project $[\pi]$ onto first K elements of ω using one of the projection algorithms described in Section 2.2;
-

Since in practice working with continuous RKHS projections is cumbersome, we use Algorithm 1 in order to project the continuous policy π onto a discrete lattice, which is then projected onto the Hilbert space³. A natural consequence of working with embedded policies means that "taking actions according to $\hat{\pi}_K$ " reduces to importance sampling of actions conditional on state bins (discussed in Appendix 7.4).

5 Experimental results

In this section, we consider a range of control tasks: bandit turntable, Pendulum and Continuous Mountain Car (CMC) from OpenAI Gym [Brockman et al., 2016]. All experimental details can be found in Appendix 7.9.3.

In Pendulum and CMC, we omit GMM and fixed basis GMM methods, since the expectation-maximization algorithm runs into stability problems when dealing with a high number of components. Fixed-basis GMM serves as a baseline to benchmark optimization issues in the Expectation-Maximization (EM) algorithm for GMMs. Moreover, we use SVD for low-rank matrix factorization and the fourth order Daubechies wavelet [Daubechies, 1988] (DB4) as an orthonormal basis alternative to Fourier (DFT).

5.1 Bandit turntable

We evaluate our framework in the multi-armed bandit [Slivkins, 2019, MAB] inspired from Fellows et al. [2018] (see Figure 2a). Results are reported in Figure 2b via the average Wasserstein-1 metric, defined as $\mathbb{E}_s[W_1(\pi(\cdot|s), \hat{\pi}(\cdot|s))]$. From the figure, we can observe that GMM has more trouble converging when we keep more than 60 components. This is due to the EM's stability issue when dealing with large number of components. While fixed basis GMM struggles to accurately approximate the policy, DFT shows a stable

³Python pseudocode can be found in Appendix 7.9.1

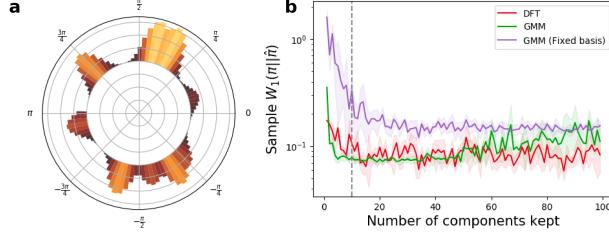


Figure 2: Bandit turtable environment in (a) polar coordinates. Magnitude of modes is proportional to the reward (lighter is higher). Actions are absolute angles in interval $[-\pi, \pi]$. (b) shows the W_1 distance between ground truth and order K approximations. Dotted line indicates the number of modes in the ground truth density.

performance after a threshold of 10 components. The appendix contains additional results in the MAB setting motivated by recent advances [Dudik et al., 2011, Foster et al., 2018], and a simplified truncation lemma for average rewards.

5.2 Continuous Mountain Car

We compare all embedding methods with a maximum likelihood estimate (MLE) of π , which consists of approximating $\mathbb{E}[\pi_K(\cdot|s)] \approx \frac{1}{K} \sum_{k=1}^K A_k$, $A_k \sim \pi(\cdot|s)$. The MLE rate of convergence to π is $O(\sqrt{K})$, which grounds the convergence rate of other methods. As shown in Figure 3 over 10 runs, DFT, SVD and DB4 reach same returns as the MLE method. Note that DB4 shows slightly better performance than DFT.

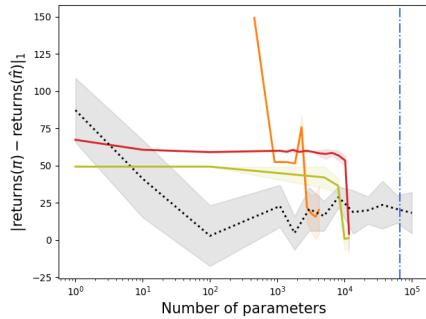


Figure 3: Difference in returns between embedded and ground truth policies for the Mountain Car task with respect of the number of parameters used.

Due to space constraints, Appendix Figure 15 contains insights on the pruning and discretization errors.

5.3 Pendulum

The same experimental setup as for Mountain Car applies to this environment. As shown in Figure 4 over 10 runs, when the true policy has converged to a degenerate (optimal) distribution (5,000 steps), all methods show comparable performance (in terms of convergence). DFT shows better convergence at early stages of training (2,000 steps), that is when the true policy has a large variance. Refer to Appendix Figure 9–11 for visualizations of the true policies.

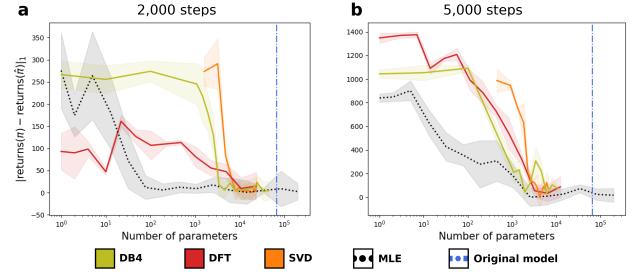


Figure 4: Absolute difference in returns collected by discretized and reconstructed Gaussian policies for (a) 2,000 steps and (b) 5,000 steps in the Pendulum task, averaged over 10 trials. Blue dots represent the number of parameters of the neural network policy. SVD, DFT and DB4 projections need an order of magnitude less in term of parameters to reconstruct the original policy.

Task	$\sqrt{\mathbb{V}[\eta(\pi)]}$	$\sqrt{\mathbb{V}[\eta(\hat{\pi}_K)]}$
Pendulum (2k)	114.51	84.75
Pendulum (3k)	333.06	191.83
Pendulum (4k)	363.6	273.1
Pendulum (5k)	182.2	163.3
CMC	28.1	11.0

Table 1: Variance of the collected returns in Pendulum-v0 under various policies over 100 trials, with K set to half of its maximum value (i.e. $K = b_S b_A / 2$).

Table 1 shows the standard deviation in returns collected by true and truncated policies over 100 trials. The truncated policies systematically has standard deviation at least as large as the one of the true policy.

6 Conclusion

In this work, we introduced a general framework for representing policies for reinforcement learning tasks. It allows to represent any continuous policy density by a projection onto the K first basis functions of a reproducing kernel Hilbert space. We showed theoretically that the performance of this embedded policy

depends on the discretization, pruning and projection algorithms, and we provide upper bounds for these errors. In addition, by projecting to a lower dimensional space, our framework provides a more stable and robust reconstructed policy. Finally, we conducted experiments to demonstrate the behaviour of SVD, DFT, DB4 and GMM basis functions on a set of classical control tasks, supporting our performance guarantees. Moreover, our experiments show a reduction in the variance of the returns of the reconstructed policies, resulting a more robust policy. A natural extension to our framework would be to directly operate in the continuous space, avoiding the discretization step procedure.

Acknowledgements

This research is supported by the Canadian Institute for Advanced Research (CIFAR AI chair program).

References

- Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. Understanding the impact of entropy on policy optimization. *arXiv preprint arXiv:1811.11214*, 2018.
- Anayo K Akametalu, Jaime F Fisac, Jeremy H Gillula, Shahab Kaynama, Melanie N Zeilinger, and Claire J Tomlin. Reachability-based safe learning with gaussian processes. In *53rd IEEE Conference on Decision and Control*, pages 1424–1431. IEEE, 2014.
- Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.
- Anil Aswani, Humberto Gonzalez, S Shankar Sastry, and Claire Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216–1226, 2013.
- André MS Barreto, Doina Precup, and Joelle Pineau. Practical kernel-based reinforcement learning. *The Journal of Machine Learning Research*, 17(1):2372–2441, 2016.
- Osbert Bastani, Yewen Pu, and Armando Solar-Lezama. Verifiable reinforcement learning via policy extraction. In *Advances in neural information processing systems*, pages 2494–2504, 2018.
- Bernhard Baumgartner. An inequality for the trace of matrix products, using absolute values. *arXiv preprint arXiv:1106.6189*, 2011.
- Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. *arXiv preprint arXiv:1707.06887*, 2017.
- Haym Benaroya, Seon Mi Han, and Mark Nagurka. *Probability models in engineering and science*, volume 193. CRC press, 2005.
- Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. In *Advances in neural information processing systems*, pages 908–918, 2017.
- Byron Boots, Geoffrey Gordon, and Arthur Gretton. Hilbert space embeddings of predictive state representations. *arXiv preprint arXiv:1309.6819*, 2013.
- Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016. URL <http://arxiv.org/abs/1606.01540>.
- Grace E Cho and Carl D Meyer. Comparison of perturbation bounds for the stationary distribution of a markov chain. *Linear Algebra and its Applications*, 335(1-3):137–150, 2001.
- Ingrid Daubechies. Orthonormal bases of compactly supported wavelets. *Communications on pure and applied mathematics*, 41(7):909–996, 1988.
- Luc Devroye. An automatic method for generating random variates with a given characteristic function. In *SIAM journal on applied mathematics*, 1986.
- Simon S Du, Yining Wang, and Aarti Singh. On the power of truncated svd for general high-rank matrix estimation problems. In *Advances in neural information processing systems*, pages 445–455, 2017.
- Miroslav Dudik, Daniel Hsu, Satyen Kale, Nikos Karampatziakis, John Langford, Lev Reyzin, and Tong Zhang. Efficient optimal learning for contextual bandits. *arXiv preprint arXiv:1106.2369*, 2011.
- Aryeh Dvoretzky, Jack Kiefer, and Jacob Wolfowitz. Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator. *The Annals of Mathematical Statistics*, pages 642–669, 1956.
- Juan José Egozcue, José Luis Díaz-Barrero, and Vera Pawlowsky-Glahn. Hilbert space of probability density functions based on aitchison geometry. *Acta Mathematica Sinica*, 22(4):1175–1182, 2006.
- Matthew Fellows, Kamil Ciosek, and Shimon Whiteson. Fourier Policy Gradients. In *ICML*, 2018.
- Dylan J Foster, Alekh Agarwal, Miroslav Dudík, Haipeng Luo, and Robert E Schapire. Practical contextual bandits with regression oracles. *arXiv preprint arXiv:1803.01088*, 2018.
- Javier Garcia and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

- C Giardina and P Chirlian. Bounds on the truncation error of periodic signals. *IEEE Transactions on Circuit Theory*, 19(2):206–207, 1972.
- Koen Goetschalckx, Bert Moons, Patrick Wambacq, and Marian Verhelst. Efficiently combining svd, pruning, clustering and retraining for enhanced neural network compression. In *Proceedings of the 2nd International Workshop on Embedded and Mobile Deep Learning*, pages 1–6, 2018.
- Steffen Grunewalder, Guy Lever, Luca Baldassarre, Massi Pontil, and Arthur Gretton. Modelling transition dynamics in mdps with rkhs embeddings. *arXiv preprint arXiv:1206.4655*, 2012.
- Alfred Haar. *Zur theorie der orthogonalen funktionsysteme*. Georg-August-Universitat, Gottingen., 1909.
- Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *CoRR*, abs/1801.01290, 2018. URL <http://arxiv.org/abs/1801.01290>.
- Dunham Jackson. *The theory of approximation*. The American mathematical society, 1930.
- Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning.
- Motonobu Kanagawa and Kenji Fukumizu. Recovering distributions from gaussian rkhs embeddings. In *Artificial Intelligence and Statistics*, pages 457–465, 2014.
- Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pages 97–117. Springer, 2017.
- Guy Lever, John Shawe-Taylor, Ronnie Stafford, and Csaba Szepesvári. Compressed conditional mean embeddings for model-based reinforcement learning. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.
- Tianyu Li, Bogdan Mazoure, Doina Precup, and Guillaume Rabusseau. Efficient planning under partial observability with unnormalized q functions and spectral learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2852–2862. PMLR, 2020.
- Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
- Michael L Littman and Richard S Sutton. Predictive representations of state. In *Advances in neural information processing systems*, pages 1555–1561, 2002.
- Mathias Lohne. The computational complexity of the fast fourier transform. Technical report, 2017.
- Yongxi Lu, Abhishek Kumar, Shuangfei Zhai, Yu Cheng, Tara Javidi, and Rogerio Feris. Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5334–5343, 2017.
- Geoffrey McLachlan and David Peel. *Finite mixture models*. John Wiley & Sons, 2004.
- J Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 209:415–446, 1909.
- Mahdi Pakdaman Naeini, Gregory F Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the... AAAI Conference on Artificial Intelligence. AAAI Conference on Artificial Intelligence*, volume 2015, page 2901. NIH Public Access, 2015.
- Yu Nishiyama, Abdeslam Boularias, Arthur Gretton, and Kenji Fukumizu. Hilbert space embeddings of pomdps. *arXiv preprint arXiv:1210.4887*, 2012.
- Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. *arXiv preprint arXiv:1703.02702*, 2017.
- Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.
- Alexander Rakhlin, Dmitry Panchenko, and Sayan Mukherjee. Risk bounds for mixture density estimation. *ESAIM: Probability and Statistics*, 9:220–229, 2005.
- Howard L Resnikoff, O Raymond Jr, et al. *Wavelet analysis: the scalable structure of information*. Springer Science & Business Media, 2012.
- Bengt Ringnér. Law of the unconscious statistician. *Unpublished Note*, 2009.
- Dorsa Sadigh, S Shankar Sastry, Sanjit A Seshia, and Anca Dragan. Information gathering actions over human internal state. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 66–73. IEEE, 2016.
- John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy

optimization. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1889–1897, Lille, France, 07–09 Jul 2015. PMLR. URL <http://proceedings.mlr.press/v37/schulman15.html>.

Paul J Schweitzer. Perturbation theory and finite markov chains. *Journal of Applied Probability*, 5(2):401–413, 1968.

Shayak Sen, Saikat Guha, Anupam Datta, Sriram K Rajamani, Janice Tsai, and Jeannette M Wing. Bootstrapping privacy compliance in big data systems. In *2014 IEEE Symposium on Security and Privacy*, pages 327–342. IEEE, 2014.

Thiago D. Simão, Romain Laroche, and Rémi Tachet des Combes. Safe policy improvement with an estimated baseline policy, 2019.

Aleksandrs Slivkins. Introduction to multi-armed bandits. *arXiv preprint arXiv:1904.07272*, 2019.

Le Song, Xinhua Zhang, Alex Smola, Arthur Gretton, and Bernhard Schölkopf. Tailoring density estimation via reproducing kernel moment matching. In *Proceedings of the 25th international conference on Machine learning*, pages 992–999, 2008.

E.M. Stein and R. Shakarchi. *Fourier Analysis: An Introduction*. Princeton University Press, 2003. ISBN 9780691113845. URL <https://books.google.ca/books?id=I6CJngEACAAJ>.

Hongwei Sun and Qiang Wu. Application of integral operator for regularized least-square regression. *Mathematical and Computer Modelling*, 49(1-2):276–285, 2009.

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

Aviv Tamar, Dotan Di Castro, and Shie Mannor. Learning the variance of the reward-to-go. *The Journal of Machine Learning Research*, 17(1):361–396, 2016.

John N Tsitsiklis and Benjamin Van Roy. Optimal stopping of markov processes: Hilbert space theory, approximation algorithms, and an application to pricing high-dimensional financial derivatives. *IEEE Transactions on Automatic Control*, 44(10):1840–1851, 1999.

Kirk Wolter. *Introduction to variance estimation*. Springer Science & Business Media, 2007.

Kehe Zhu. *Operator theory in function spaces*. Number 138. American Mathematical Soc., 2007.

7 Appendix

Reproducibility Checklist

We follow the reproducibility checklist "*The machine learning reproducibility checklist*" and point to relevant sections explaining them here.

For all algorithms presented, check if you include:

- **A clear description of the algorithm, see main paper and included codebase.** The proposed approach is completely described by Alg. 2.
- **An analysis of the complexity (time, space, sample size) of the algorithm.** The space complexity of our algorithm depends on the number of desired basis. It is $4K$ for a 1-dimension K -component GMM with diagonal covariance, $mK + K^2 + nK$ for a K -truncated SVD of an $m \times n$ matrix, and only K for the wavelet and Fourier bases. Note that a simple implementation of SVD requires to find the three matrices $\mathbf{U}, \mathbf{D}, \mathbf{V}^\top$ before truncation.

The time complexity depends on the reconstruction, pruning and discretization algorithms, which involves conducting rollouts w.r.t. policy π , training a quantile encoder on the state space trajectories, and finally embedding the pruned matrix using the desired algorithm. We found that the Gaussian mixture is the slowest to train, due to shortcomings of the expectation-maximization algorithm.

- **A link to a downloadable source code, including all dependencies.** The code is included with Supplemental Files as a zip file; all dependencies can be installed using Python's package manager. Upon publication, the code would be available on Github. Additionally, we include the model's weights as well as the discretized policy for Pendulum-v0 environment.

For all figures and tables that present empirical results, check if you include:

- **A complete description of the data collection process, including sample size.** We use standard benchmarks provided in OpenAI Gym (Brockman et al., 2016).
- **A link to downloadable version of the dataset or simulation environment.** Not applicable.
- **An explanation of how samples were allocated for training / validation / testing.** We do not use a training-validation-test split, but instead report the mean performance (and one standard deviation) of the policy at evaluation time across 10 trials.
- **An explanation of any data that were excluded.** The only data exclusion was done during policy pruning, as outlined in the main paper.
- **The exact number of evaluation runs.** 10 trials to obtain all figures, and 200 rollouts to determine ρ_π .
- **A description of how experiments were run.** See Section Experimental Results in the main paper and didactic example details in Appendix.
- **A clear definition of the specific measure or statistics used to report results.** Undiscounted returns across the whole episode are reported, and in turn averaged across 10 seeds. Confidence intervals shown in Fig. 3 were obtained using the pooled variance formula from a difference of means t -test.
- **Clearly defined error bars.** Confidence intervals are always mean \pm 1 standard deviation over 10 trials.
- **A description of results with central tendency (e.g. mean) and variation (e.g. stddev).** All results use the mean and standard deviation.
- **A description of the computing infrastructure used.** All runs used 1 CPU for all experiments with 8Gb of memory.

7.1 Bandit example

An N -armed bandit has a reward distribution such that $r(i) = \frac{1}{\sqrt{2\pi}} \exp(-(i - N/2)^2/2) + U_i$ for $i = 1, \dots, N$, and stationary i.i.d. measurement noise U , with the additional restriction that $\mathbb{V}(U) = \lambda^2$, i.e. the signal-to-noise ratio (SNR) simplifies to $\text{SNR} = \frac{N}{2\lambda}$.

Running any suitable policy optimization method Dudik et al. [2011], Foster et al. [2018] then produces a policy π which regresses on both the signal and the noise (blue curve in Figure 7). However, if the class of regressor functions is overparameterized, then the policy will also capture the noise, which in turn will lead to suboptimal collected rewards. If regularization is not an option (e.g. the training data was deleted due to privacy requirements as discussed in Sen et al. [2014]), then separating signal from noise becomes more challenging.

Our approach involves projecting the policy into a Hilbert space in which noise is captured via fine-grained basis functions and therefore can be eliminated via truncation of the basis. Figure 7 illustrates the hypothetical bandit scenario, and the behavior of our approach. Executing the policy with 2 components will yield a larger reward signal than executing the original policy, because the measurement noise is removed via truncation.

7.2 Convergence results for Fourier approximation and GMM mixture

Rate of convergence of Fourier approximation Let π_K^{Fourier} denote the density approximated by the first K^{th} Fourier partial sums Stein and Shakarchi [2003], then a result from Jackson [1930] shows that

$$\|\pi - \pi_K^{\text{Fourier}}\|_1 = O(K^{-1}). \quad (17)$$

More recent results Giardina and Chirlian [1972] provide even tighter bounds for continuous and periodic functions with $m - 1$ continuous derivatives and bounded m^{th} derivative. In such case,

$$\|\pi - \pi_K^{\text{Fourier}}\|_\infty = O(K^{-(2m+1)}). \quad (18)$$

Rate of convergence of mixture approximation Let π_K^{MM} denote a (finite) K -component mixture model. A result from Rakhlin et al. [2005] shows the following result for π_K^{MM} in the class of mixtures with K marginally-independent scaled density functions and π in the class of lower-bounded probability density functions:

$$KL(\pi || \pi_K^{\text{MM}}) = O(K^{-1} + n^{-1}), \quad (19)$$

where n is the number of i.i.d. samples used in the learning of π_K^{MM} .

The constants hidden inside the big-O notation depend on the nature of the function and can become quite large, which can explain differences in empirical evaluation.

7.3 Discretization of continuous policies

Consider the case when f is a continuous, positive and decreasing function with a discrete sequence $a_n = f(n)$. For example, the reconstruction error $W_1(\Pi, \hat{\Pi})$ as well as difference in returns $|\eta(\pi) - \eta(\hat{\pi})|$ fall under this family. Then, $\int_0^\infty f(t)dt < \infty$ implies that $\sum_{i=0}^\infty a_n < \infty$. Under these assumptions, convergence guarantees (e.g. on monotonically decreasing reconstruction error) in continuous space imply convergence in the discrete (empirical) setting. Hence, we operate on discrete spaces rather than continuous ones.

All further computations of distance between two discretized policies will have to be computed over the corresponding discrete grid $(i, j), i = 1, \dots, b_S, j = 1, \dots, b_A$. One can look at the average action taken by two MDP policies at state s :

$$\left| \mathbb{E}_{a \sim \pi}[a|s] - \mathbb{E}_{a \sim \tilde{\pi}}[a|s] \right| \leq W_1(\Pi_s, \tilde{\Pi}_s) \quad (20)$$

This relationship is one of the motivations behind using W_1 to assess goodness-of-fit in the experiments section.

Naively discretizing some function f over a fixed interval $[a, b]$ can be done by n equally spaced bins. To pass from a continuous value $f(x)$ to discrete, one would find $i \in 1, \dots, n$ such that $\frac{b-a}{n}i \leq x \leq \frac{b-a}{n}(i+1)$. However, using a uniform grid for a non-uniform f wastes representation power. To re-allocate bins in low density areas, we use the

quantile binning method, which first computes the cumulative distribution function of f , called F . Then, it finds n points such that the probability of falling in each bin is equal. Quantile binning can be seen as uniform binning in probability space, and exactly corresponds to uniform discretization if f is constant (uniform distribution). Below, we present an example of discretizing 4,000 sample points taken from four $\mathcal{N}(\mu_i, 0.3)$ distributions, using the quantile binning.

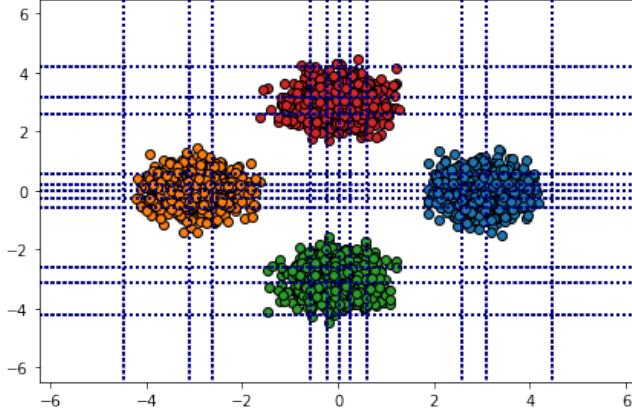


Figure 5: Quantile binning for the four Gaussians problem, using 10 bins per dimension.

In Figure 5, we are only allowed to allocate 10 bins per dimension. Note how the grid is denser around high-probability regions, while the furthermost bins are the widest. This uneven discretization, if applied to the policy density function, allows the agent to have higher detail around high-probability regions.

In Python, the function `sklearn.preprocessing.KBinsDiscretizer` can be used to easily discretize samples from the target density.

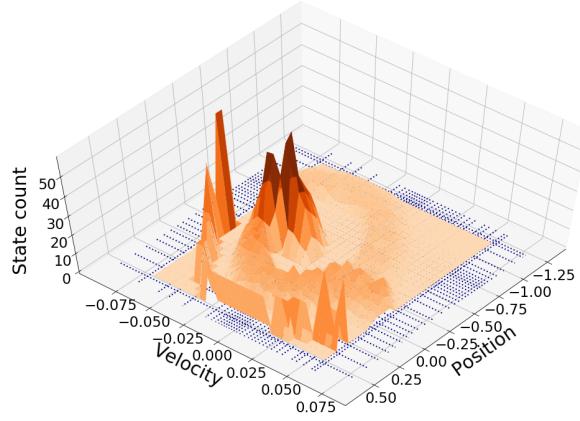


Figure 6: Quantile binning of unnormalized state visitation counts of DQN policy in the discrete Mountain Car task. Bins closer to the starting states are visited more often than those at the end of the trajectory. Blue dots represent the bins limits.

Since policy densities are expected to be more complex than the previous example, we analyze quantile binning in the discrete Mountain Car environment. To do so, we train a DQN policy until convergence (i.e. collected rewards at least -110), and perform 500 rollouts using the greedy policy. Trajectories are used to construct a 2-dimensional state visitation histogram. At the same time, all visited states are given to the quantile binning algorithm, which is used to assign observations to bins.

Figure 6 presents the state visitation histogram with $n_S = 30$ bins, where color represents the state visitation count. Bins are shown by dotted lines.

7.4 Acting with tabular policies

In order to execute the learned policy, one has to be able to (1) generate samples conditioned on a given state and (2) evaluate the log-probability of state-action pairs. Acting with a tabular density can be done via various sampling techniques. For example, importance sampling of the atoms corresponding to each bin is possible when b_A is small, while the inverse c.d.f. method is expected to be faster in larger dimensions. The process can be further optimized on a case-per-case basis for each method. For example, sampling directly from the real part of a characteristic function can be done with the algorithm defined in Devroye [1986]

Furthermore, it is possible to use any of the above algorithms to jointly sample (s, a) pairs under the assumption that states were discretized by quantiles. First, uniformly sample a state bin and then use any of the conditional sampling algorithms to sample action a . Optionally, one can add uniform noise (clipped to the range of the bin) to the sampled action. This naive trick would transform discrete actions into continuous approximations of the policy network.

7.5 Bandit motivating example

We observe that truncating the embedded policy can have a denoising effect, i.e. boosting the signal-to-noise ratio (SNR). As a motivation, consider a multi-armed bandit setting [Slivkins, 2019] with reward distribution $r(i) = \frac{1}{\sqrt{2\pi}} \exp(-(i - N/2)^2/2) + U_i$ for $i = 1, \dots, N$, and stationary measurement noise U_i . By running any suitable policy optimization method [Dudik et al., 2011, Foster et al., 2018], we obtain a policy π which regresses on both the signal and the noise. Figure 7 illustrates the hypothetical bandit scenario. Applying DFT followed by truncation produces π_K , some of which are shown in Figure 7a. Sampling actions from π_K yields a lower average regret up to $K=10$, because the measurement noise is truncated and the correct signal mode is recovered.

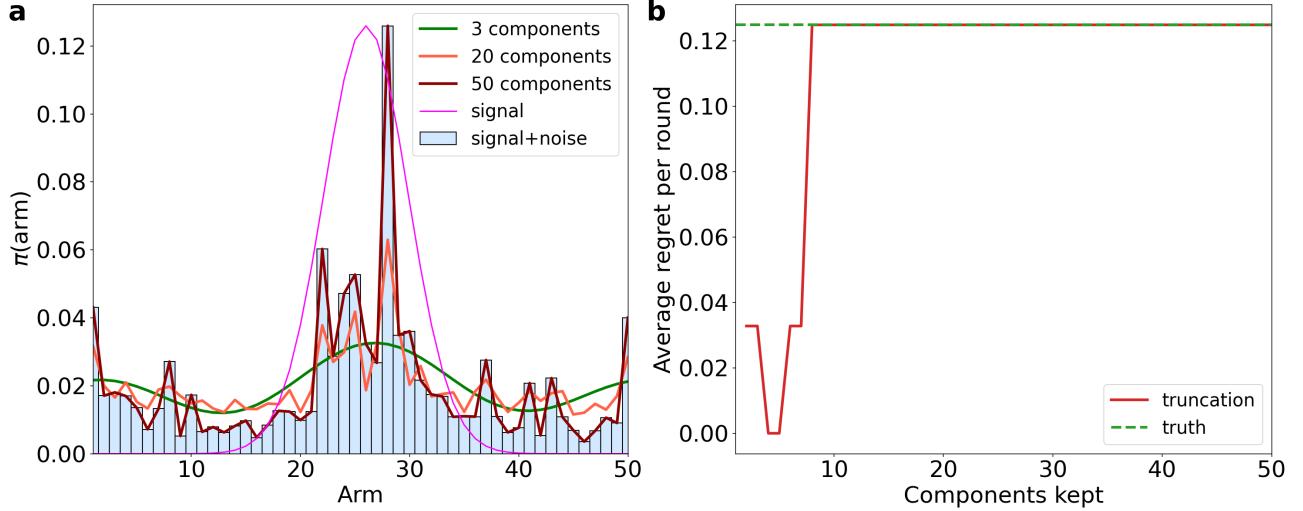


Figure 7: (a) Motivating bandit example with ground truth signal, noisy signal and truncated policy, and (b) rewards collected by greedy (i.e. the mode) truncated and true policies

The bandit performance bound can be stated as follows:

Lemma 7.1. Let \mathcal{H} be an RKHS, $\pi_1, \pi_2 \in \mathcal{H}$ be represented by $\{\xi_k^{\pi_1}\}_{k=1}^{\infty}$ and $\{\xi_k^{\pi_2}\}_{k=1}^{\infty}$ and $r : \mathcal{A} \rightarrow \mathbb{R}$ be the bandit's reward function. Let $M > 0$ be such that $|\pi_1(a) - \pi_2(a)| \leq M \|\pi_1 - \pi_2\|_{\mathcal{H}}$ for all $a \in \mathcal{A}$ and let $p, q > 0$ s.t. $\frac{1}{p} + \frac{1}{q} = 1$. Then

$$|\eta(\pi_1) - \eta(\pi_2)| \leq |\mathcal{A}|^{1/q} M \|r\|_p \sum_{k=1}^{\infty} \frac{(\xi_k^{\pi_1} - \xi_k^{\pi_2})^2}{\lambda_k}$$

Lemma 7.1 guarantees that rewards collected by π_2 will depend on the distance between π_1 and π_2 . The proof can be found in the next section.

7.6 Proofs

Proof. (Lemma 3.3) Throughout the proof, we will be using the expectation \mathbb{E}_β which can be seen as the classical inner product weighted by the initial state-action distribution

$$\langle f, g \rangle_\beta = \int_{s,a} f(s,a)g(s,a)\beta(s,a)d(sa)$$

over the domain of s, a .

If we take a uniform initialization density, i.e. $\beta(s,a) \propto (|\mathcal{S}||\mathcal{A}|)^{-1}$, then the inner product simplifies to

$$\langle f, g \rangle_\beta = (|\mathcal{S}||\mathcal{A}|)^{-1}\langle f, g \rangle,$$

which will be used further on to simplify the covariance term between orthonormal basis functions.

After a first-order Taylor expansion of variances around $\eta(\pi)$ and $\eta(\hat{\pi}_K)$ described in Benaroya et al. [2005], we obtain

$$\begin{aligned} \mathbb{V}_\beta[\pi(a|s)] &= \{\sigma'(\eta(\pi))\}^2 \mathbb{V}_\beta[Q^\pi(s,a)] + E_2^\pi \\ \mathbb{V}_\beta[\hat{\pi}_K(a|s)] &= \{\sigma'(\eta(\hat{\pi}_K))\}^2 \mathbb{V}_\beta[Q^{\hat{\pi}_K}(s,a)] + E_2^{\hat{\pi}_K}, \end{aligned} \quad (21)$$

where E_2^π is the Taylor residual of order 2 for policy π . This identity is very widely used in statistics, especially for approximating the variance of functions of statistics through the *delta method* [Wolter, 2007].

Expanding the variance of sum of correlated random variables, we obtain

$$\mathbb{V}_\beta[\pi(a|s)] = \mathbb{V}_\beta[\hat{\pi}_K(a|s)] + \mathbb{V}_\beta[\varepsilon_K(a|s)] + 2\mathbb{V}_\beta[\hat{\pi}_K(a|s), \varepsilon_K(a|s)] \quad (22)$$

The covariance term $\mathbb{V}_\beta[\hat{\pi}_K(a|s), \varepsilon_K(a|s)]$ can be decomposed as $\mathbb{E}_\beta[\hat{\pi}_K(a|s)\varepsilon_K(a|s)] - \mathbb{E}_\beta[\hat{\pi}_K(a|s)]\mathbb{E}_\beta[\varepsilon_K(a|s)]$.

Then, the second moment expression of the covariance becomes zero with respect to this inner product, due to the use of orthonormal basis functions. We use the dominated convergence theorem which lets the order of summation be changed for $\sum_{j=K}^{\infty} \xi_j \omega_j(s,a)$, which greatly simplifies the calculations.

$$\begin{aligned} \mathbb{E}_\beta[\hat{\pi}_K(a|s)\varepsilon_K(a|s)] &= \mathbb{E}_\beta\left[\sum_{k=1}^K \xi_k \omega_k(s,a) \sum_{j=K}^{\infty} \xi_j \omega_j(s,a)\right] \\ &= \sum_{k=1}^K \sum_{j=K}^{\infty} \xi_k \xi_j \mathbb{E}_\beta[\omega_k(s,a)\omega_j(s,a)] \\ &= \sum_{k=1}^K \sum_{j=K}^{\infty} \xi_k \xi_j \langle \omega_k, \omega_j \rangle_\beta \\ &= 0 \quad (\omega_k(s,a) \text{ and } \omega_j(s,a) \text{ are orthogonal }) \end{aligned} \quad (23)$$

Combining both expressions yields the relationship between the variance of the truncated policy and that of the expected returns.

$$\begin{aligned} \mathbb{V}_\beta[Q^\pi] &= \left[\frac{\{\sigma'(\eta(\hat{\pi}_K))\}^2 \mathbb{V}_\beta[Q^{\hat{\pi}_K}] - 2\mathbb{E}_\beta[\hat{\pi}_K]\mathbb{E}_\beta[\varepsilon_K] + \mathbb{V}_\beta[\varepsilon_K] + E_2^{\hat{\pi}_K}}{\{\sigma'(\eta(\pi))\}^2} \right] + E_2^\pi \\ &= \left[\frac{\{\sigma'(\eta(\hat{\pi}_K))\}^2 \mathbb{V}_\beta[Q^{\hat{\pi}_K}] - 2\mathbb{E}_\beta[\hat{\pi}_K]\mathbb{E}_\beta[\varepsilon_K] + \mathbb{V}_\beta[\varepsilon_K]}{\{\sigma'(\eta(\pi))\}^2} \right] + E_2^\pi + \frac{E_2^{\hat{\pi}_K}}{\{\sigma'(\eta(\pi))\}^2} \end{aligned} \quad (24)$$

If the Taylor expansion is performed in a neighborhood of $\eta(\pi)$ and $\eta(\hat{\pi}_K)$, we can then consider the error terms as sufficiently small and neglect them, as is often the case in classical statistics [Benaroya et al., 2005].

$$\mathbb{V}_\beta[Q^\pi] \approx \left(\frac{\sigma'(\eta(\hat{\pi}_K))}{\sigma'(\eta(\pi))} \right)^2 \mathbb{V}_\beta[Q^{\hat{\pi}_K}] + \frac{\mathbb{V}_\beta[\varepsilon_K] - 2\mathbb{E}_\beta[\hat{\pi}_K]\mathbb{E}_\beta[\varepsilon_K]}{\{\sigma'(\eta(\pi))\}^2} \quad (25)$$

For the variance of the returns of the true policy to be not lower than the variance of returns of the truncated policy, the coefficient of the first term on the right hand side must be at most 1, and the second term must be positive. Note that these are sufficient but not necessary conditions.

Precisely, the following conditions must be satisfied:

1. $\sigma'(\eta(\hat{\pi}_K)) \leq \sigma'(\eta(\pi))$;
2. $\sqrt{\frac{\mathbb{E}_\beta[\varepsilon_K^2]}{3}} \geq \mathbb{E}_\beta[\varepsilon_K]$. This is an assumption on the second moment behavior of the policy and is satisfied by, for example, the Student's t distribution with degrees of freedom $\nu > 1$.
3. The Taylor error terms E_2^π and $E_2^{\hat{\pi}_K}$ are small since the expansion is performed around $\eta(\pi)$ and $\eta(\hat{\pi}_K)$, respectively.

That is, under various initializations, truncation of policy coefficients can be beneficial by reducing the variance. \square

Proof. (Theorem 3.5) We want to quantify the difference between the continuous cdf Π and the step-function defined by the quantile binning strategy, $[\Pi]$, with b bins. Notice that the error between Π and $[\Pi]$ in bin i can be written as

$$\delta_i = \int_{q_{i-1}}^{q_i} \Pi(x)dx - \frac{q_i - q_{i-1}}{b}, \quad (26)$$

where $q_i = \hat{Q}\left(\frac{i}{b}\right)$.

The total error across all bins is therefore

$$\delta = \sum_{i=1}^b \left[\int_{q_{i-1}}^{q_i} \Pi(x)dx - \frac{q_i - q_{i-1}}{b} \right], \quad (27)$$

The error δ is computed across both states and actions, meaning that the total volume of the error between the policies can be thought of as error in \mathcal{A} made for a single state group, which leads to the state approximation error being considered b_S times, akin to conditional probabilities.

The state-action total error can be computed as follows

$$\delta_{a,s} = \sum_{i=1}^{b_S} \left[\int_{q_{i-1}^S}^{q_i^S} \Pi_S(s)ds - \frac{i(q_i^S - q_{i-1}^S)}{b_S} \right] \sum_{j=1}^{b_A} \left[\int_{q_{i,j-1}^A}^{q_{i,j}^A} \Pi_A(a)da - \frac{j(q_{i,j}^A - q_{i,j-1}^A)}{b_A} \right], \quad (28)$$

where superscripts and subscripts A and S denotes dependence of the quantity on either action or state, respectively. The notation $q_{i,j}$ refers to the quantile $\hat{Q}\left(\frac{i}{b}\right)$ computed conditional on the state falling into bin i . \square

Proof. (Lemma 7.1)

$$\begin{aligned}
 |\eta(\pi_1) - \eta(\pi_2)| &= \left| \sum_{a \in \mathcal{A}} r(a) [\pi_1(a) - \pi_2(a)] \right| \\
 &= \left| \sum_{a \in \mathcal{A}} r(a) \Delta(a) \right| \\
 &\leq \sum_{a \in \mathcal{A}} |r(a) \Delta(a)| \\
 &= \|r\Delta\|_1 \\
 &\leq \|r\|_p \|\Delta\|_q \\
 &\leq |\mathcal{A}|^{1/q} M \|r\|_p \sum_{k=1}^{\infty} \frac{(\xi_k^{\pi_1} - \xi_k^{\pi_2})^2}{\lambda_k}
 \end{aligned}$$

where the before last line is a direct application of Holder inequality and the last line happens because evaluation is a bounded linear functional such that $|f(x)| \leq M\|f\|_{\mathcal{H}}$ for all $f \in \mathcal{H}$.

□

Proof. (Theorem 3.4) Recall the following result from Kakade and Langford, Schulman et al. [2015]:

$$|\eta(\pi_1) - L_{\pi_2}(\pi_1)| \leq \frac{4\alpha^2\gamma\bar{\epsilon}}{(1-\gamma)^2}, \quad (29)$$

where $\bar{\epsilon} = \max_{s,a} |Q_{\pi_1}(s, a) - V_{\pi_1}(s)|$ and $\alpha = \max_s TV(\pi_1(\cdot|s), \pi_2(\cdot|s))$.

Instead, suppose that for every fixed state $s \in \mathcal{S}$, there exists an RKHS of all square-integrable conditional densities of the form $\pi(\cdot|s)$. We examine the difference term between policies in this \mathcal{H}_s . Since $\pi_1, \pi_2 \in \mathcal{H}_s$, $\delta_{1,2} = \pi_1 - \pi_2 \in \mathcal{H}_s$ and hence there exist $M_s > 0$ such that $|\delta_{1,2}(a)| \leq M_s \|\delta_{1,2}\|_{\mathcal{H}_s}$ for all $a \in \mathcal{A}$. Then,

$$\begin{aligned}
 \sum_{a \in \mathcal{A}} |\pi_1(a) - \pi_2(a)| &\leq \sum_{a \in \mathcal{A}} |\delta_{1,2}(a)| \\
 &\leq M_s \sum_{a \in \mathcal{A}} \|\delta_{1,2}\|_{\mathcal{H}_s} \\
 &\leq |\mathcal{A}| M_s \sum_{k=1}^{\infty} \frac{(\xi_k^{\pi_1} - \xi_k^{\pi_2})^2}{\lambda_k} \\
 &= |\mathcal{A}| M_s \Delta_{\mathcal{H}_s}^{\infty},
 \end{aligned} \quad (30)$$

where we let $\Delta_{\mathcal{H}_s}^{\infty} = \sum_{k=1}^{\infty} \frac{(\xi_k^{\pi_1} - \xi_k^{\pi_2})^2}{\lambda_k}$ for shorter notation.

Then,

$$|\eta(\pi_1) - L_{\pi_2}(\pi_1)| \leq \frac{4|\mathcal{A}|^2\gamma\bar{\epsilon}}{(1-\gamma)^2} (\max_{s \in \mathcal{S}} M_s \Delta_{\mathcal{H}_s}^{\infty})^2, \quad (31)$$

We can obtain a performance bound by first expanding the left hand side

$$L_{\pi_2}(\pi_1) = \eta(\pi_1) + \sum_s \rho_{\pi_1}(s) \sum_a \pi_2(a|s) A_{\pi_1}(s, a)$$

Using Eq. (29), and using reverse triangle inequality, we have

$$\begin{aligned}
 |\eta(\pi_2) - \eta(\pi_1)| - \left| \sum_s \rho_{\pi_1}(s) \sum_a \pi_2(a|s) A_{\pi_1}(s, a) \right| &\leq \left| \eta(\pi_2) - \left\{ \eta(\pi_1) + \sum_s \rho_{\pi_1}(s) \sum_a \pi_2(a|s) A_{\pi_1}(s, a) \right\} \right| \\
 &\leq \frac{4\alpha^2\gamma\bar{\epsilon}}{(1-\gamma)^2}
 \end{aligned}$$

Therefore,

$$\begin{aligned} \left| \eta(\pi_2) - \eta(\pi_1) \right| &\leq \frac{4\alpha^2\gamma\bar{\epsilon}}{(1-\gamma)^2} + \left| \sum_s \rho_{\pi_1}(s) \sum_a \pi_2(a|s) A_{\pi_1}(s, a) \right| \\ &\leq \frac{4\alpha^2\gamma\bar{\epsilon}}{(1-\gamma)^2} + \bar{\epsilon} \end{aligned} \quad (32)$$

We combine Eq. (30) with Eq. (32) to obtain the following RKHS form on the improvement lower bound.

$$|\eta(\pi_2) - \eta(\pi_1)| \leq \frac{4|\mathcal{A}|^2\bar{\epsilon}\gamma}{(1-\gamma)^2} (\max_{s \in \mathcal{S}} M_s \Delta_{\mathcal{H}_s}^\infty)^2 + \bar{\epsilon} \quad (33)$$

If we suppose that weights after K are small enough, we can split the approximation error using Lemma 3.2 for $E_K = \frac{\varepsilon^{2(K+1)}}{1-\varepsilon^2}$:

$$M_s \Delta_{\mathcal{H}_s}^\infty \leq M_s \left(\Delta_{\mathcal{H}_s}^K + E_K \right), \quad (34)$$

Using this in Eq. (33) yields the final result:

$$|\eta(\pi_2) - \eta(\pi_1)| \leq \frac{4|\mathcal{A}|^2\bar{\epsilon}\gamma}{(1-\gamma)^2} \left(\max_{s \in \mathcal{S}} (M_s \Delta_{\mathcal{H}_s}^K)^2 + 2E_K \max_{s \in \mathcal{S}} M_s \Delta_{\mathcal{H}_s}^K + E_K^2 \max_{s \in \mathcal{S}} M_s^2 \right) + \bar{\epsilon} \quad (35)$$

$$|\eta(\pi_2) - \eta(\pi_1)| \leq \frac{4|\mathcal{A}|^2\bar{\epsilon}\gamma}{(1-\gamma)^2} \left\{ \max_{s \in \mathcal{S}} (M_s \Delta_{\mathcal{H}_s}^K)^2 + O(\varepsilon^{2K} \max_{s \in \mathcal{S}} M_s \Delta_{\mathcal{H}_s}^K) \right\} + \bar{\epsilon}. \quad (36)$$

Now, if ε is close to zero (i.e. π_1 and π_2 have very similar $\xi_K, \xi_{K+1} \dots$), then the expression simplifies to

$$|\eta(\pi_2) - \eta(\pi_1)| \leq \frac{4|\mathcal{A}|^2\bar{\epsilon}\gamma}{(1-\gamma)^2} \max_{s \in \mathcal{S}} (M_s \Delta_{\mathcal{H}_s}^K)^2 + \bar{\epsilon}. \quad (37)$$

□

Proof. (Theorem 3.7) We first represent the approximate transition matrix $\mathbf{P}_{\tilde{\pi}}$ induced by the policy $\tilde{\pi}$ as a perturbation of the true transition:

$$\begin{aligned} \mathbf{P}_{\tilde{\pi}} &= \mathbf{P}_\pi + (\mathbf{P}_{\tilde{\pi}} - \mathbf{P}_\pi) \\ &= \mathbf{P}_\pi + \mathbf{E} \end{aligned} \quad (38)$$

Then, the difference between stationary distributions $\rho_{\tilde{\pi}}$ and ρ_π is equal to Schweitzer [1968], Cho and Meyer [2001]:

$$\rho_\pi^\top - \rho_{\tilde{\pi}}^\top = \rho_{\tilde{\pi}}^\top \mathbf{E} \mathbf{Z}, \quad (39)$$

where $\mathbf{Z} = (\mathbf{I} - \mathbf{P}_\pi + \mathbf{1}_{|\mathcal{S}|} \rho_\pi^\top)^{-1}$ is the *fundamental matrix* of the Markov chain induced by \mathbf{P}_π and $\mathbf{1}_{|\mathcal{S}|}$ is a vector of ones.

In particular, the above result holds for Schatten norms Baumgartner [2011]:

$$\|\rho_\pi - \rho_{\tilde{\pi}}\|_2 \leq \|\rho_{\tilde{\pi}}\|_2 \|\mathbf{Z}\|_{S_\infty} \|\mathbf{E}\|_{S_\infty} \leq \|\mathbf{Z}\|_{S_\infty} \|\mathbf{E}\|_{S_1} \quad (40)$$

So far, this result is known for irreducible, homogeneous Markov chains and has no decision component. Consider the matrix \mathbf{E} , which is the difference between expected transition models of true and approximate policies. It can be expanded into products of matricized tensors:

$$\begin{aligned} \mathbf{E} &= \mathbf{P}_{\tilde{\pi}} - \mathbf{P}_\pi \\ &= \mathbf{A}(\tilde{\mathbf{\Pi}} \otimes \mathbf{I}) \mathbf{T}_{(3)} - \mathbf{A}(\mathbf{\Pi} \otimes \mathbf{I}) \mathbf{T}_{(3)}^\top \\ &= \mathbf{A}((\tilde{\mathbf{\Pi}} - \mathbf{\Pi}) \otimes \mathbf{I}) \mathbf{T}_{(3)}^\top \end{aligned} \quad (41)$$

The norm of \mathbf{E} can also be upper bounded as follows:

$$\begin{aligned}
 \|\mathbf{E}\|_{S_1} &\leq \|\mathbf{A}\|_{S_1} \|(\tilde{\Pi} - \Pi) \otimes \mathbf{I} \mathbf{T}_{(3)}^\top\|_{S_\infty} \\
 &\leq \|\mathbf{A}\|_{S_1} \|(\tilde{\Pi} - \Pi) \otimes \mathbf{I} \mathbf{T}_{(3)}^\top\|_{S_1} \\
 &\leq \|\mathbf{A}\|_{S_1} \|(\tilde{\Pi} - \Pi) \otimes \mathbf{I}\|_{S_2} \|\mathbf{T}_{(3)}^\top\|_{S_2} \\
 &\leq \|\mathbf{A}\|_{S_1} \|(\tilde{\Pi} - \Pi) \otimes \mathbf{I}\|_{S_1} \|\mathbf{T}_{(3)}^\top\|_{S_2} \\
 &\leq \|\mathbf{A}\|_{S_1} \|\tilde{\Pi} - \Pi\|_{S_1} \|\mathbf{I}\|_{S_\infty} \|\mathbf{T}_{(3)}^\top\|_{S_2} \\
 &= \|\tilde{\Pi} - \Pi\|_{S_1} \|\mathbf{T}_{(3)}\|_{S_2}
 \end{aligned} \tag{42}$$

Combining this result from that of Schweitzer [1968] yields

$$\|\rho_\pi - \rho_{\tilde{\pi}}\|_{S_1} \leq \underbrace{\|\mathbf{Z}\|_{S_\infty}}_{\substack{\text{Depends on} \\ \text{MDP}}} \underbrace{\|\tilde{\Pi} - \Pi\|_{S_1}}_{\substack{\text{Depends on} \\ \pi}} \underbrace{\|\mathbf{T}_{(3)}\|_{S_2}}_{\substack{\text{Depends on} \\ \text{MDP}}} \tag{43}$$

□

7.7 Time complexity comparison for different approximation methods

The time complexity for a k -truncated SVD decomposition of a matrix $A \in \mathbb{R}^{n \times m}$ is $O(mnk)$ as discussed in Du et al. [2017]. Fast Fourier transform can be done in $O(mn \log(mn))$ for the 2-dimensional case Lohne [2017]. Hence, SVD is expected to be faster whenever $k < \log(nm)$. The discrete wavelet transform's (DWT) time complexity depends on the choice of mother wavelets. When using Mallat's algorithm, the 2-dimensional DWT is known to have complexities ranging from $O(n^2 \log n)$ to as low as $O(n^2)$, for a square matrix of size $n \times n$ and depending on the choice of mother wavelet Resnikoff et al. [2012]. Finally, we expect FFT and Daubechies wavelets to have less parameters than SVD, since the former use pre-defined orthonormal bases, while the latter must store the left and right singular vectors.

7.8 Rate of convergence of stationary distribution under random policy

We validate the rate of convergence of Thm. 3.7 in the toy experiment described below. Consider a deterministic chain MDP with N states. A fixed policy in state i transitions to $i + 1$ with probability α and to $i - 1$ with probability $1 - \alpha$. If in states 1 or N , the agent remains there with probability $1 - \alpha$ and α , respectively. We consider the expected transition model \mathbf{P}_π obtained using the policy reconstructed through discrete Fourier transform. A visualization of Theorem 3.7 is shown in Fig. 8.

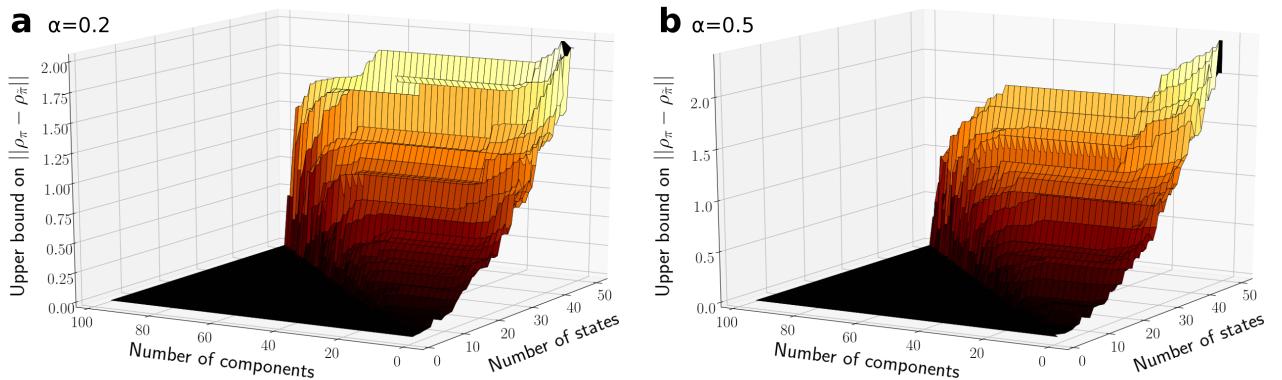


Figure 8: Upper bound on $\|\rho_\pi - \rho_{\tilde{\pi}}\|$ in the chain MDP task as a function of number of Fourier components and of the number of states.

7.9 Experiment details

This subsection covers all aspects of the conducted experiments, including pseudocode, additional results and details about the setup.

7.9.1 Python pseudocode

Below is the Python-like pseudocode for the case of Fourier bases (which can be easily adapted to all bases examined in this paper). This pseudocode is heavily simplified Python, and skips some bookkeeping steps such as edge cases, but convey the overall flow of the algorithm.

```
1 def RKHS_policy_step_1_and_2(cts_pi,env,n_trajectories,n_episode_steps,b_S,b_A):
2     """
3         cts_pi: Object with 2 methods:
4             - sample(state): samples an action from pi(.|state);
5             - log_prob(state,action): returns pi(a|s);
6         env: OpenAI Gym environment;
7         n_trajectories: Number of trajectories in rollouts (positive integer);
8         n_episode_steps: Number of maximum steps in each episode (positive integer);
9         b_S: number of state bins per dimension (positive integer);
10        b_A: number of action bins per dimension (positive integer).
11        """
12
13     from sklearn.preprocessing import KBinsDiscretizer
14     states,actions,rewards = rollout(cts_pi,env,n_trajectories,n_episode_steps)
15     # We use quantile in the paper but KMeans can also be used to find the bins
16     enc_A = KBinsDiscretizer(n_bins=b_A,encode='ordinal',strategy='quantile')
17     enc_S = KBinsDiscretizer(n_bins=b_S,encode='ordinal',strategy='quantile')
18     bins_A = enc_A.fit_transform(actions)
19     bins_S = enc_S.fit_transform(states)
20
21     dsc_pi = np.zeros(shape=(b_S,b_A))
22     for i in range(b_S):
23         for j in range(b_A):
24             s,a = enc_S.inverse_transform([i]), enc_A.inverse_transform([j])
25             dsc_pi[i,j] = np.exp(cts_pi.log_prob(s,a))
26             dsc_pi[i,:] /= np.sum(dsc_pi[i,:]) # re-normalize every conditional pmf
27
28     rho_pi = np.zeros(shape=(b_S,))
29     for bin in bins_S:
30         rho_pi[bin] += 1
31     rho_pi /= np.sum(rho_pi)
32
33     return dsc_pi, rho_pi
```

```
1 def RKHS_policy_step_3(dsc_pi,rho_pi):
2     """
3         dsc_pi: np.array of size b_S x b_A;
4         rho_pi: np.array of length b_S.
5         """
6         idx_to_keep = np.where(rho_pi>0)
7
8         pruned_dsc_pi = dsc_pi[idx_to_keep]
```

```

10     def pruned_agent(state_bin):
11         actions = np.arange(0,b_A)
12         if state_bin in idx_to_keep:
13             probs = pruned_dsc_pi[idx_to_keep==state_bin] # probabilities from the lattice
14         else:
15             probs = actions*0+1/actions.size # uniform probabilities
16         return np.random.choice(actions,1,p=probs)
17
18     return pruned_agent

```

```

1  def RKHS_policy_step_4(dsc_pi,K):
2      """
3          dsc_pi: np.array of size b_S x b_A;
4          K: number of components to keep (positive integer).
5      """
6
7      FS = np.fft.fftn(dsc_pi) # transform to Fourier domain
8      fshift = np.real(np.fft.fftshift(FS)).flatten() # vector of size b_S x b_A
9      # find threshold s.t. exactly K components are not zeroed out
10     threshold = sorted(flat_fshift,reverse=True)[K]
11
12     mask = np.real(fshift >= th).astype(int)
13     fshift *= mask # apply the binary mask to zero out |xi_K,|xi_{K+1},...
14     f_ishift = np.fft.ifftshift(fshift) # inverse shift of truncated spectrum
15     f_dft = np.real(np.fft.ifftn(f_ishift))

```

7.9.2 Bandit turntable

We consider a bandit problem in which rewards are spread in a circle as shown in Figure 2 a). Actions consist of angles in the interval $[-\pi, \pi]$. Given the corresponding Boltzmann policy, we compare the reconstruction quality of the discrete Fourier transform and GMM. In addition to Fourier and GMM, we also compare with fixed basis GMM, where the means are sampled uniformly over $[-\pi, \pi]$ and variance is drawn uniformly from $\{0.1, 0.5, 1, 2\}$; only mixing coefficients are learned. This provides insight into whether the reconstruction algorithms are learning the policy, or if π is so simple that a random set of Gaussian can already approximate it well.

7.9.3 Experimental parameters

All parameters were chosen using a memory-performance trade-off heuristic. For example, for Continuous Mountain Car, we based ourselves off Figure 15 to select $b_S = 35$.

Hyperparameters	Turntable	Pendulum	Continuous Mountain Car
b_A	100	15	10
b_S	N/A	35	35
Optimizer		Adam	
Architecture		256	
Learning rate		1e-03	
Hidden dimension		256	
# rollouts		100	
Torch seeds		0 to 9	

Table 2: Hyperparameters used across experiments, N/A: not applicable

7.9.4 Pendulum

This classical mechanic task consists of a pendulum which needs to swing up in order to stay upright. The actions represent the joint effort (torque) between -2 and 2 units. We train SAC until convergence and save snapshots of the actor and critic after $2k$ and $5k$ steps. The reconstruction task is to recover both the Gaussian policy (actor) and the Boltzmann-Q policy (critic, temperature set to 1).

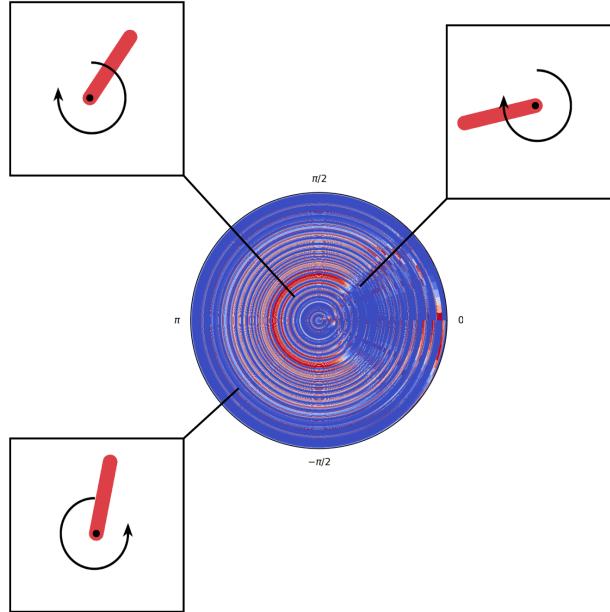


Figure 9: Plots of Gaussian Π after $5k$ training steps in polar coordinates. Each circle of different radius represents a states in the `Pendulum` environment as shown by the snapshots. Higher intensity colors (red) represent higher density mass on the given angular action.

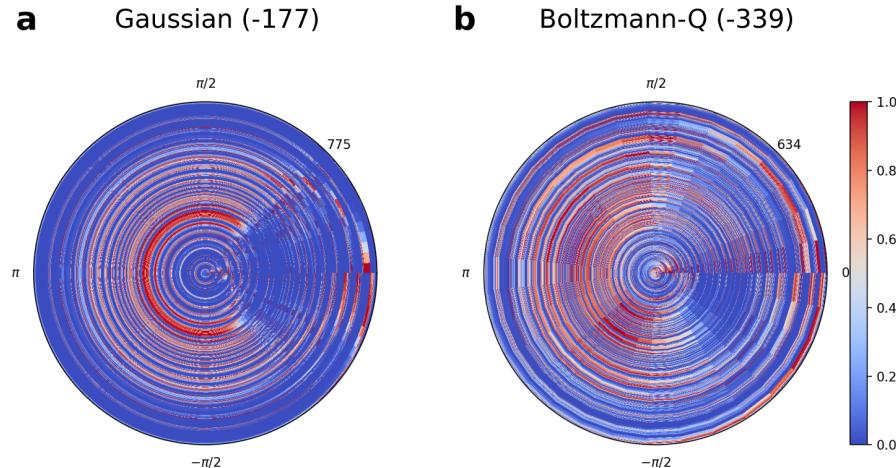


Figure 10: Plots of discretized and pruned (a) Gaussian policy and (b) Boltzmann-Q policy in polar coordinates for 5 thousands training steps of soft actor-critic. Rewards collected by their respective continuous networks are indicated in parentheses. Each circle of radius s corresponds to $\pi(\cdot|s)$ for a discrete $s = 1, 2, \dots, 750$. All densities are on the same scale.

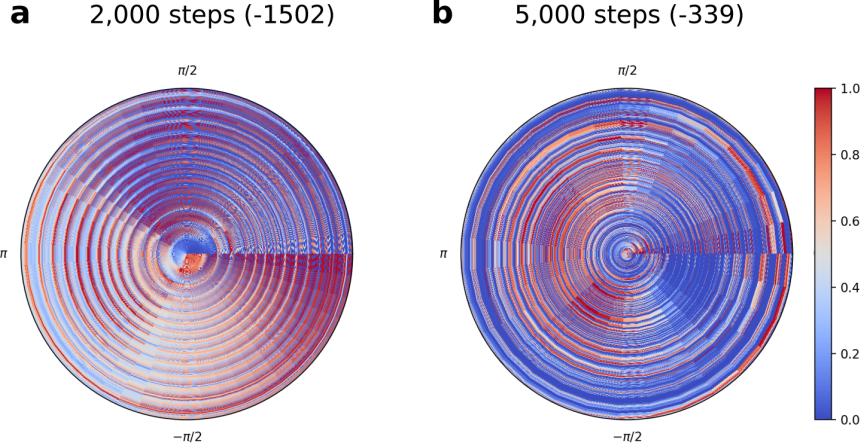


Figure 11: Plots of discretized and pruned Boltzmann policy in polar coordinates for 2 and 5 thousands training steps of soft actor-critic. Each circle of radius s corresponds to $\pi(\cdot|s)$ for a discrete $s = 1, 2, \dots, 750$.

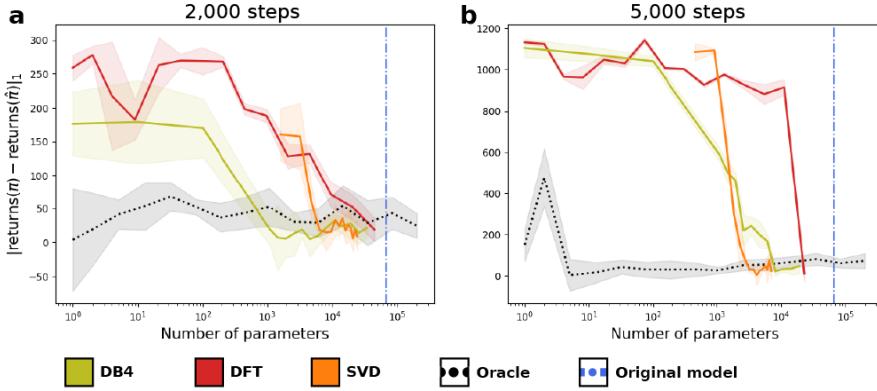


Figure 12: Absolute difference in returns collected by discretized and reconstructed Boltzmann-Q policies for *Pendulum-v0*, averaged over 10 trials. Blue dots represent the number of parameters of the neural network policy. SVD, DFT and DB4 projections need an order of magnitude less in term of parameters to reconstruct the original policy.

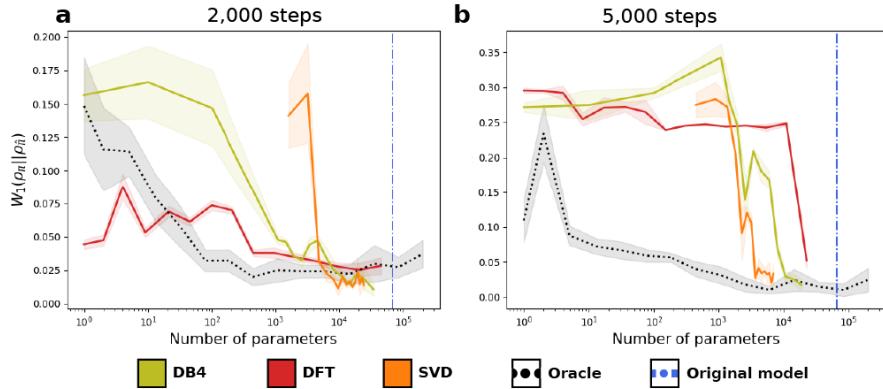


Figure 13: W_1 distance between the true and reconstructed stationary Boltzmann-Q distributions, averaged over 10 trials for *Pendulum-v0*.

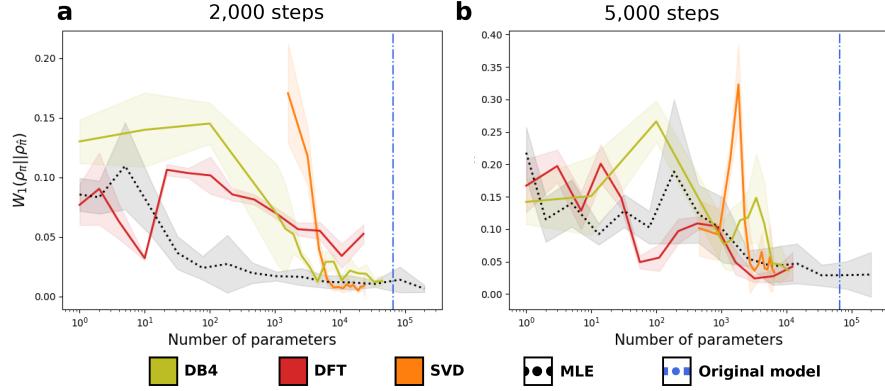


Figure 14: W_1 distance between the true and reconstructed stationary distributions, averaged over 10 trials. SVD, DFT and DB4 methods show a fast convergence to the oracle’s stationary distribution.

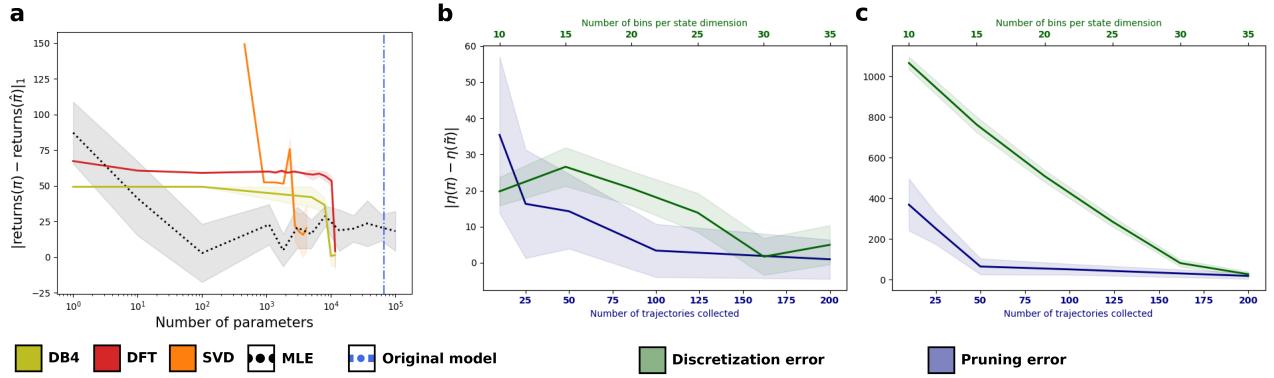


Figure 15: (a) Difference in returns between embedded and ground truth policies for the Mountain Car task with respect of the number of parameters used and the discretization and pruning errors for (b) the Gaussian policy of `ContinuousMountainCar-v0` and (c) the Boltzmann-Q policy of `Pendulum-v0`

7.9.5 Mountain Car

In this environment, the agent (a car) must reach the flag located at the top of a hill. It needs to go back and forth in order to generate sufficient momentum to reach the top. The agent is allowed to apply a speed motion in the interval $[-1, 1]$. We use Soft Actor-Critic (SAC, Haarnoja et al. [2018]) to train the agent until convergence (25k steps).

Figure 15a shows that the error between the true and truncated policies steadily decreases as a function of projection weights kept. Note that SVD’s parameters are computed as total entrie in the matrices U, D, V , and hence the smallest possible rank (rank 1) of D dictates the minimal number of parameters. Figure 15b-c show how the discretization and pruning performances behave as a function of trajectories (for pruning, since it relies on trajectories to estimate ρ_π), or bins per state dimension (for discretization).

7.10 Usefulness of the pruning step

In an environment where the state space is very large, or when policies are degenerate along a narrow path in the state space, the pruning step can turn out to be very useful.

Table 3 shows the proportion of parameters from the discretized policy that was pruned. Pruned states are ones which are visited with probability at most ε ; in all our experiments, ε was set to 0, so no visited state was ever pruned.

Task	Pruned %
Pendulum (2K)	95.36
Pendulum (3K)	94.49
Pendulum (4K)	94.59
Pendulum (5K)	98.45

Table 3: Percentage of pruned parameters for the Pendulum environment policies

The pruning step is extremely efficient in the Pendulum task, allowing to reduce the memory complexity between the discretization and the truncation step by up to 98%.

Note that as policies get trained on more samples, they concentrate along the optimal policy. This in turn means that less exploration is needed, and hence the policy coverage gets reduced, allowing us to prune more states.