Near-Optimal Sample Complexity in Reward-Free Kernel-Based Reinforcement Learning

Aya Kayal[1] **Sattar Vakili**[2] **Laura Toni**[1] **Alberto Bernacchia**[2]
[1]University College London [2]MediaTek Research

Abstract

Reinforcement Learning (RL) problems are being considered under increasingly more complex structures. While tabular and linear models have been thoroughly explored, the analytical study of RL under nonlinear function approximation, especially kernel-based models, has recently gained traction for their strong representational capacity and theoretical tractability. In this context, we examine the question of statistical efficiency in kernel-based RL within the reward-free RL framework, specifically asking: *how many samples are required to design a near-optimal policy?* Existing work addresses this question under restrictive assumptions about the class of kernel functions. We first explore this question by assuming a *generative model*, then relax this assumption at the cost of increasing the sample complexity by a factor of H, the length of the episode. We tackle this fundamental problem using a broad class of kernels and a simpler algorithm compared to prior work. Our approach derives new confidence intervals for kernel ridge regression, specific to our RL setting, which may be of broader applicability. We further validate our theoretical findings through simulations.

1 INTRODUCTION

Reinforcement Learning (RL) with nonlinear function approximation is a powerful method for learning general Markov Decision Processes (MDPs) through interactions with the environment. Kernel ridge regression for the prediction of the expected value function is perhaps one of the most versatile methods that has gained traction in recent years (Yang et al., 2020; Vakili and Olkhovskaya, 2023; Chowdhury and Oliveira, 2023), and lends itself to theoretical analysis. As a burgeoning research area, there are still numerous open problems and challenges in this topic.

We focus our work on statistical aspects of RL within the reward-free RL framework (Jin et al., 2020a; Wang et al., 2020; Qiu et al., 2021), which involves an exploration phase and a planning phase. In the exploration phase, the reward is unknown; the algorithm interacts with the environment to gather information about the underlying MDP, in the form of a dataset of transitions. In the planning phase, the reward is revealed; the algorithm uses the knowledge of the reward and the dataset gathered in the exploration phase to design a near-optimal policy. The planning phase is thus akin to offline RL (Precup, 2000; Antos et al., 2008; Munos and Szepesvári, 2008; Levine et al., 2020; Xie et al., 2021; Chen and Jiang, 2019). In this paper, we answer the following fundamental question: *Under some reasonable assumptions on the underlying MDP, what is the minimum number of samples required to enable designing a near-optimal policy?*

We refer to the number of samples as *sample complexity* and measure the optimality of the eventual policy in terms of error in the value function. In particular we refer to a policy as ϵ-optimal if its value function is at most a small $\epsilon > 0$ away from that of the optimal policy for all states.

The reward-free RL framework has been studied in tabular (Jin et al., 2020a) and linear (Wang et al., 2020; Hu et al., 2022; Wagenmaker et al., 2022) settings. Under the tabular setting, it has been shown that $\mathcal{O}(|\mathcal{S}|^2|\mathcal{A}|H^5/\epsilon^2)$ samples are sufficient to achieve an ϵ-optimal policy, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, respectively, and H represents the length of episode. In the linear setting, a sample complexity of $\mathcal{O}(d^3H^6/\epsilon^2)$ has been established that does not scale with the size of the state-action space, but the ambient dimension d of the linear model representing the transition structure of the MDP. With the limita-

Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258. Copyright 2025 by the author(s).

tions of the linear model (e.g., as shown in Lee and Oh, 2023), recent works have considered non-linear function approximation in RL. The work of Qiu et al. (2021) considered the reward-free RL framework with kernel-based function approximation. However, their results only apply to very smooth kernels with exponential eigendecay, such as Squared Exponential (SE), but fail to provide finite sample complexity applicable to a large class of kernels of interest with polynomial eigendecay (see Definition 2), such as Matérn family or Neural Tangent (NT) kernels. This shortcoming arises from the bias in the collected samples. Specifically in the exploration phase of Qiu et al. (2021), the samples are adaptively collected to achieve a high value with respect to a hypothetical reward —proportional to the uncertainties of the kernel ridge regression— introducing bias to the samples and inflating confidence intervals.

Another closely related work on reward-free RL in the kernel setting is Vakili et al. (2024), which, like Qiu et al. (2021), uses a hypothetical reward proportional to the uncertainty of kernel ridge regression. However, it improves upon Qiu et al. (2021) by providing order-optimal sample complexities for kernels with polynomially decaying eigenvalues, where Qiu et al. (2021)'s results are unbounded. This is achieved via an adaptive domain partitioning procedure inspired by Vakili and Olkhovskaya (2023). In this method, the state-action domain is adaptively divided into multiple subdomains as samples are collected, with kernel-based value function estimates constructed based on samples from the same subdomain, while discarding previous observations from other subdomains. Although their approach offers theoretical advantages, it is tedious to implement in practice due to complex domain partitioning structure. Moreover, discarding samples may degrade the empirical performance, a concern that is not addressed in Vakili et al. (2024). Additionally, their theoretical results depend on specific assumptions about the relationship between kernel eigenvalues and domain size, which limits generality of their work. A detailed comparison between our work and the two closely related works of Qiu et al. (2021) and Vakili et al. (2024) is provided in Appendix A.1 along with a more comprehensive literature review in Appendix A.2.

In contrast to the existing work, this paper establishes near-optimal sample complexities for the reward-free kernel-based RL framework over a general class of kernels, without relying on restrictive assumptions. This is accomplished via a simple algorithm and a novel confidence interval for unbiased samples, broadly applicable to other RL settings (offline RL, model-based, infinite horizon), and supported by empirical evidence. Specifically, we start with a case where a *generative model* (Kakade, 2003) is present and it permits the algorithm to sample state-actions of its choice during the exploration phase, not limiting the algorithm to stay on the Markovian trajectory. This setting has been extensively considered in previous work on statistical efficiency of RL (see, e.g., Kearns and Singh, 1998a; Azar et al., 2013; Sidford et al., 2018a,b; Agarwal et al., 2020; Yang and Wang, 2019). In the presence of a generative model, we propose a simple algorithm that collects *unbiased* samples by choosing the state-actions with highest kernel-based regression uncertainty at each step. We derive order-optimal sample complexities for this algorithm in terms of $\frac{1}{\epsilon}$, while Vakili et al. (2024) do not offer any particular advantages in the generative model case. Generative models are applicable in scenarios like games where the algorithm can manipulate the current state, offering insights into the statistical aspects of RL. However, this may not be the case in other scenarios. Inspired by the analysis of the exploration algorithm with a generative model, we propose a second online exploration algorithm that collects samples adhering to the Markovian trajectory. We prove that this relaxing of generative model requirement incurs merely an H factor increase in the sample complexity.

To highlight the significance of our results, we consider kernels with polynomial eigendecay that are of practical and theoretical interest (Srinivas et al., 2010; Jacot et al., 2018; Vakili et al., 2023). When the eigenvalues of the kernel decay polynomially as $\mathcal{O}(m^{-p})$ —see Definition 2—the results of Qiu et al. (2021) lead to possibly vacuous (infinite) sample complexities, while we prove an $\tilde{\mathcal{O}}((\frac{H^3}{\epsilon})^{2+\frac{2}{p-1}})$ sample complexity for the generative setting and $\tilde{\mathcal{O}}(H(\frac{H^3}{\epsilon})^{2+\frac{2}{p-1}})$ for the online setting. Our sample complexity results are comparable to those of Vakili et al. (2024). In a technical comparison, their approach requires a specific assumption on the dependence between kernel eigenvalues and domain size (see, Vakili et al., 2024, Definition 4.1), which we do not. Additionally, they employ a sophisticated domain partitioning algorithm that is more difficult to implement and possibly inefficient in practice, whereas our algorithm is simpler and more straightforward. In the case of Matérn kernel with smoothness parameter ν on a d-dimensional domain, where $p = 1 + \frac{2\nu}{d}$, our results translate to a sample complexity of $\tilde{\mathcal{O}}(H(\frac{H^3}{\epsilon})^{2+\frac{d}{\nu}})$, that matches the $\Omega((\frac{1}{\epsilon})^{2+\frac{d}{\nu}})$ lower bound proven in Scarlett et al. (2017) for the degenerate case of bandits with $H = 1$. Our sample complexities thus are not generally improvable in their scaling with $\frac{1}{\epsilon}$.

To achieve these results, we establish a confidence interval applicable to kernel ridge regression in our RL setting that may be of broader interest. The key technical novelties of this confidence interval involves leveraging the structure of RKHS and the properties of unbiased, independent samples. The main results regarding the confidence interval and sample complexities of the two exploration algorithms, with and without the generative model, are presented in Theorems 1, 2 and 3, respectively, in Section 4. We empirically validate our analytical findings through numerical experiments comparing the performance of our proposed exploration algorithms with that of Qiu et al. (2021), as detailed in Section 5. Section 2 provides an overview of episodic MDPs, the reward-free RL framework, and kernel-based models. Section 3 presents our algorithms for both the exploration and planning phases. Detailed proofs of theorems, along with the details of experiments and further experimental results, are included in the appendix due to space limitations.

2 PRELIMINARIES AND PROBLEM FORMULATION

In this section, we introduce the episodic MDP setting, describe the reward-free RL framework, provide background on kernel methods, and outline our technical assumptions.

2.1 Episodic MDP

An episodic MDP can be described by the tuple $M = (\mathcal{S}, \mathcal{A}, H, P, r)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ the action space, and the integer H the length of each episode. Here, $r = \{r_h\}_{h=1}^H$ represents the reward functions, and $P = \{P_h\}_{h=1}^H$ the transition probability distributions.[1] The state-action space is denoted by $\mathcal{Z} = \mathcal{S} \times \mathcal{A}$. The notation $z = (s, a)$ is used throughout the paper for state-action pairs. For each step $h \in [H]$, the reward function $r_h : \mathcal{Z} \to [0, 1]$ is supposed to be deterministic for simplicity, and $P_h(\cdot|s, a)$ is the unknown transition probability distribution on $\mathcal{S}$ for the next state given the current state-action pair (s, a).

A policy $\pi = \{\pi_h : \mathcal{S} \to \mathcal{A}\}_{h=1}^H$ determines the action $\pi_h(s)$ —possibly random— taken by the agent at state s during each step h. At the beginning of each episode, the environment picks an arbitrary initial state s_1. The agent adopts a policy $\pi = \{\pi_h\}_{h=1}^H$. For each step $h \in [H]$, the agent observes the current state $s_h \in \mathcal{S}$, and selects an action $a_h = \pi_h(s_h)$. The subsequent state, s_{h+1}, is then drawn from the transition probability distribution $P_h(\cdot|s_h, a_h)$. The episode ends when the agent receives the final reward $r_H(s_H, a_H)$.

We are interested in maximizing the expected total reward in the episode, starting at step h. This is quantified by the value function, which is defined as follows:

$$V_h^\pi(s) = \mathbb{E}\left[\sum_{h'=h}^H r_{h'}(s_{h'}, a_{h'}) \middle| s_h = s\right], \forall s \in \mathcal{S}, h \in [H], \tag{1}$$

where the expectation is taken with respect to the randomness in the trajectory $\{(s_h, a_h)\}_{h=1}^H$ obtained by the policy π. It can be shown that under mild assumptions (e.g., continuity of P_h, compactness of $\mathcal{Z}$, and boundedness of r), there exists an optimal policy $\pi^\star$ which attains the maximum possible value of $V_h^\pi(s)$ at every step and at every state (e.g., see, Puterman, 2014). We use the notation $V_h^\star(s) = \max_\pi V_h^\pi(s)$, $\forall s \in \mathcal{S}, h \in [H]$. By definition $V_h^{\pi^\star} = V_h^\star$. An ϵ-optimal policy is defined as follows.

Definition 1. *(ϵ-optimal policy) For $\epsilon > 0$, a policy π is called ϵ-optimal if it achieves near-optimal values from any initial state as follows: $V_1^\pi(s) \geq V_1^\star(s) - \epsilon, \quad \forall s \in \mathcal{S}$.*

Policy design often relies on the expected value of a value function with respect to the transition probability distribution, presented using the following notation:

$$[P_h V](s, a) := \mathbb{E}_{s' \sim P_h(\cdot|s,a)}[V(s')]. \tag{2}$$

We also define the state-action value function $Q_h^\pi : \mathcal{Z} \to [0, H]$ as follows:

$$Q_h^\pi(s, a) = \mathbb{E}_\pi\left[\sum_{h'=h}^H r_{h'}(s_{h'}, a_{h'}) \middle| s_h = s, a_h = a\right], \tag{3}$$

where the expectation is taken with respect to the randomness in the trajectory $\{(s_h, a_h)\}_{h=1}^H$ obtained by the policy π. The Bellman equation associated with a policy π is then represented as

$$Q_h^\pi(s, a) = r_h(s, a) + [P_h V_{h+1}^\pi](s, a),$$
$$V_h^\pi(s) = \mathbb{E}[Q_h^\pi(s, \pi_h(s))], \quad V_{H+1}^\pi = \mathbf{0}.$$

The notation $V = \mathbf{0}$ is used for $V(s) = 0$, for all $s \in \mathcal{S}$. We may specify the reward function in $V^\pi, Q^\pi, V^\star, Q^\star$ notations for clarity, for example, $V^\pi(s; r)$ and $Q^\star(z; r)$.

2.2 Reward-Free RL Framework

We aim to learn ϵ-optimal policies while minimizing the samples collected during exploration. Specifically, we employ the reward-free RL framework, which consists of two phases: exploration and planning. In the exploration phase, we collect a dataset $\mathcal{D}_N = \{\mathcal{D}_{h,N}\}_{h\in[H]}$, where each $\mathcal{D}_{h,N} = \left\{\left(s_{h,n}, a_{h,n}, s'_{h+1,n} \sim P_h(\cdot|s_{h,n}, a_{h,n})\right)\right\}_{n\in[N]}$ consists of N transition samples at step h. Then, in the planning phase, once the reward r is revealed, we design a

[1]We deliberately do not use the standard term transition kernel for P_h, to avoid confusion with kernel in kernel-based learning.

policy specific to reward r using the data collected during the exploration phase. The number N denotes the sample complexity required to design an ϵ-optimally performing policy. A critical question arises: *How many exploration episodes are necessary to achieve ϵ-optimal policies?* We provide an answer in this work.

2.3 Kernel Ridge Regression

A main step in RL with function approximation, keeping the Bellman equation in mind, is to derive statistical predictions and bounds for the expected value function $[PV] : \mathcal{Z} \to \mathbb{R}$, for some given value function $V : \mathcal{S} \to \mathbb{R}$ and conditional probability distribution $P(\cdot|z)$. Let us use the notation $f = [PV]$. Suppose that we are given n noisy observations of f, represented as $\{(z_i, y_i)\}_{i\in[n]}$, where $y_i = f(z_i) + \varepsilon_i$, and ε_i denotes zero-mean random noise. Provided a positive definite kernel $k : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ and employing kernel ridge regression, we can make the following prediction for f:

$$\hat{f}_n(z) = k_n^\top(z)(K_n + \tau^2 I)^{-1}\boldsymbol{y}_n, \tag{4}$$

where $k_n(z) = [k(z, z_1), k(z, z_2), \cdots, k(z, z_n)]^\top$ is the pairwise kernel values between z and observation points, $K_n = [k(z_i, z_j)]_{i,j\in[n]}$ is the Gram matrix, $\boldsymbol{y}_n = [y_1, y_2, \cdots, y_n]^\top$ is the vector of observations, $\tau > 0$ is a free parameter, and I is the identity matrix, appropriately sized to match the dimensions of K_n. In addition, the following uncertainty estimate can be utilized to bound the prediction error:

$$\sigma_n^2(z) = k(z, z) - k_n^\top(z)(K_n + \tau^2 I)^{-1}k_n(z) \tag{5}$$

In particular, various $1 - \delta$ confidence intervals of the form $|f(z) - \hat{f}_n(z)| \leq \beta(\delta)\sigma_n(z)$, under various assumptions, are proven, where $\beta(\delta)$ is a confidence interval width multiplier that depends on the setting and assumptions (Abbasi-Yadkori, 2013; Chowdhury and Gopalan, 2017; Vakili et al., 2021a; Whitehouse et al., 2023). One of our primary contributions is establishing a novel confidence intervals for $f = [PV]$, applicable to our RL setting. Equipped with the confidence intervals, we are able to design policies using least squares value iteration or its *optimistic* variant.

Reproducing Kernel Hilbert Spaces and Mercer Representation: Mercer theorem provides a representation of a positive definite kernel function k using an infinite-dimensional feature map: $k(z, z') = \sum_{m=1}^\infty \gamma_m \varphi_m(z)\varphi_m(z')$, where $\gamma_m > 0$ are referred to as Mercer eigenvalues, and φ_m are the corresponding eigenfunctions. The Reproducing Kernel Hilbert Space (RKHS) associated with k, denoted $\mathcal{H}_k$, is defined as: $\mathcal{H}_k = \{f : f = \sum_{m=1}^\infty w_m\phi_m, \quad w_m \in \mathbb{R}, \ \|\boldsymbol{w}\| < \infty\}$, where $\phi_m := \sqrt{\gamma_m}\varphi_m$ form an orthonormal basis of $\mathcal{H}_k$. Here, $\boldsymbol{w} = [w_1, w_2, \cdots]^\top$ represents a possibly infinite-dimensional weight vector. The RKHS norm of f is defined as $\|f\|_{\mathcal{H}_k} = \|\boldsymbol{w}\|$, the ℓ^2 norm of the weight vector. Formal statements and further details are provided in Appendix F.

To effectively use the confidence intervals established by the kernel-based models on f, we require the following assumption.

Assumption 1. *We assume $P_h(s|\cdot,\cdot) \in \mathcal{H}_k$, for some positive definite kernel k, and $\|P_h(s|\cdot,\cdot)\|_{\mathcal{H}_k} \leq 1$, for all $s \in \mathcal{S}$ and $h \in [H]$.*

Consequently, for all $V : \mathcal{S} \to [0, H]$, we have $\|[PV]\|_{\mathcal{H}_k} = \mathcal{O}(H)$. See Yeh et al. (2023), Lemma 3, for a proof.

Information Gain and Eigendecay: The analytical results in kernel-based RL and bandits are often given in terms of a kernel specific complexity term referred to as maximum information gain, defined as follows (Srinivas et al., 2010; Vakili et al., 2021b):

$$\Gamma(n) = \sup_{\{z_i\}_{i=1}^n \subset \mathcal{Z}} \frac{1}{2}\log\det(I + \frac{K_n}{\tau^2}), \tag{6}$$

Maximum information gain depends on the eigendecay defined as follows.

Definition 2. *A kernel k is said to have a polynomial (resp. exponential) eigendecay if $\gamma_m = \mathcal{O}(m^{-p})$ (resp. $\gamma_m = \mathcal{O}(c^m)$), for some $p > 1$ ($c < 1$), where γ_m are the Mercer eigenvalues in decreasing order.*

For kernels with polynomial and exponential eigendecays, $\Gamma(n) = \tilde{\mathcal{O}}(n^{\frac{1}{p}})$ and $\Gamma(n) = \mathcal{O}(\text{polylog}(n))$, respectively (Vakili et al., 2021b).

3 ALGORITHM DESCRIPTION

We now present our algorithms for both the exploration and planning phases. We begin by presenting the algorithm for the planning phase, as it remains unchanged across various exploration algorithms.

Algorithm 1 Planning Phase

```
Input: $\tau$, $\beta$, $\delta$, $k$, $M(\mathcal{S}, \mathcal{A}, H, P, r)$, and exploration dataset $\mathcal{D}_N$.
for $h = H, H-1, \cdots, 1$, do
    Compute the prediction $\hat{g}_h$ according to (8);
    Let $Q_h(\cdot,\cdot) = \Pi_{[0,H]}[\hat{g}_h(\cdot,\cdot) + r_h(\cdot,\cdot)]$;
    $V_h(\cdot) = \max_{a\in\mathcal{A}} Q_h(\cdot, a)$;
    $\pi_h(\cdot) = \arg\max_{a\in\mathcal{A}} Q_h(\cdot, a)$;
end for
Output: $\{\pi_h\}_{h\in[H]}$.
```

Algorithm 2 Exploration Phase **with** Generative Model

```
Require: $\tau$, $k$, $\mathcal{S}$, $\mathcal{A}$, $H$, $P$, $N$;
1: Initialize $\mathcal{D}_{h,0} = \{\}$, for all $h \in [H]$;
2: for $n = 1, 2, \cdots, N$ do
3:     for $h = 1, 2, \cdots, H$ do
4:         Let $s_{h,n}, a_{h,n} = \arg\max_{s,a\in\mathcal{A}} \sigma_{h,n-1}(s, a)$;
5:         Observe $s'_{h+1,n} \sim P_h(\cdot|s_{h,n}, a_{h,n})$;
6:         Update $\mathcal{D}_{h,n} = \mathcal{D}_{h,n-1} \bigcup \{s_{h,n}, a_{h,n}, s'_{h+1,n}\}$.
7:     end for
8: end for
9: Output: $\mathcal{D}_N$.
```

Algorithm 3 Exploration Phase **without** Generative Model

```
Require: $\tau$, $k$, $\beta$, $\delta$, $\mathcal{S}$, $\mathcal{A}$, $H$, $P$, $N$;
  Initialize $\mathcal{D}_{h,0} = \{\}$, for all $h \in [H]$;
  for $n = 1, 2, \cdots, N$ do
    for $h_0 = 1, 2, \cdots, H$ do
      Initialize $V_{h_0+1,n} = \mathbf{0}$
      for $h = h_0, h_0 - 1, \cdots, 1$ do
        Obtain $\hat{f}_{h,(n,h_0)}$; $Q_{h,(n,h_0)}$, and $V_{h,(n,h_0)}(\cdot)$ according to (11) and (12), respectively.
      end for
      for $h = 1, 2, \cdots, h_0$ do
        Observe $s_{h,n}$; Take action $a_{h,n} = \arg\max_{a\in\mathcal{A}} Q_{h,n}(s_{h,n}, a)$;
      end for
      Update $\mathcal{D}_{h_0,n} = \mathcal{D}_{h_0,n-1} \bigcup \{s_{h_0,n}, a_{h_0,n}, s_{h_0+1,n}\}$
    end for
  end for
```

3.1 Planning Phase

In the planning phase, the reward function r is revealed to the learner. In addition, a dataset $\mathcal{D}_N = \{\mathcal{D}_{h,N}\}_{h\in[H]}$ is available, with $\mathcal{D}_{h,N} = \{s_{h,n}, a_{h,n}, s'_{h+1,n} \sim P_h(\cdot|s_{h,n}, a_{h,n})\}_{n\in[N]}$ for each step $h \in [H]$. The objective is to leverage the knowledge of the reward function and utilize the dataset to design a near-optimal policy. As mentioned in the introduction, the planning phase comprises of an offline RL design without further interaction with the environment.

In the planning phase of our algorithm, we derive a policy using least squares value iteration. Specifically, at step h, we compute a prediction, $\hat{g}_h$, for the expected value function in the next step $[P_hV_{h+1}]$. We then define

$$Q_h(\cdot, \cdot) = \Pi_{[0,H]} \left[r_h(\cdot, \cdot) + \hat{g}_h(\cdot, \cdot) \right], \tag{7}$$

where $\Pi_{[a,b]}$ denotes projection on $[a, b]$ interval. The policy π is then obtained as a greedy policy with respect to Q. For each $h \in [H]$,

$$\pi_h(\cdot) = \arg\max_{a\in\mathcal{A}} Q_h(\cdot, a).$$

We now detail the computation of $\hat{g}_h$. Keeping the Bellman equation in mind and starting with $V_{H+1} = \mathbf{0}$, $\hat{g}_h$ is the kernel ridge predictor for $[P_hV_{h+1}]$. This prediction uses N observations $\boldsymbol{y}_h = [V_{h+1}(s'_{h+1,1}), V_{h+1}(s'_{h+1,2}), \cdots, V_{h+1}(s'_{h+1,N})]^\top$ at points $\{z_{h,n}\}_{n=1}^N$. Recall that $\mathbb{E}_{s'\sim P(\cdot|z_{h,n})} [V_{h+1}(s')] = [P_hV_{h+1}](z_{h,n})$. The observation noise $V_{h+1}(s'_{h+1,n}) - [P_hV_{h+1}](z_{h,n})$ is due to random transitions and is bounded by $H - h \leq H$. Specifically,

$$\hat{g}_h(z) = k_{h,N}^\top(z)(\tau^2 I + K_{h,N})^{-1}\boldsymbol{y}_h, \tag{8}$$

where $k_{h,N}(z) = [k(z, z_{h,1}), k(z, z_{h,2}), \cdots, k(z, z_{h,N})]^\top$ is the pairwise kernel values between z and observation points and $K_{h,N} = [k(z_{h,i}, z_{h,j})]_{i,j\in[N]}$ is the Gram matrix. We then define Q_h according to (7) and also set

$$V_h(s) = \max_{a\in\mathcal{A}} Q_h(s, a).$$

The values of $\hat{g}_h$, Q_h and V_h are obtained recursively for $h = H, H - 1, \cdots, 1$. For a pseudocode, see Algorithm 1.

3.2 Exploration Phase

In the exploration phase, the algorithm collects a dataset $\mathcal{D}_N = \{\mathcal{D}_{h,N}\}_{h\in[H]}$, where $\mathcal{D}_{h,N} = \{s_{h,n}, a_{h,n}, s'_{h+1,n}\}_{h\in[H],n\in[N]}$ for each $h \in [H]$, later used in the planning phase to design a near-optimal policy. The primary goal during this phase is to gather the most informative observations.

Initially, we consider a preliminary case where a *generative model* (Kakade, 2003) is present that can produce transitions for the state-actions selected by the algorithm. Under this setting, we demonstrate that a simple rule for data collection leads to improved and desirable sample complexities. Inspired by these results, we introduce a novel algorithm that completely relaxes the requirement for a generative model, at the price of increasing the number of exploration episodes by a factor of H. The key aspect of our algorithms is the *unbiasedness*–statistical independence of the collected samples, which means that the observation points do not depend on previous transitions.

3.2.1 Exploration with a Generative Model

In this section, we outline the exploration phase when a generative model is present. At each step h of the current exploration episode, uncertainties derived from kernel ridge regression are employed to guide exploration. Specifically, let

$$\sigma_{h,n}^2(z) = k(z, z) - k_{h,n}^\top(z)(\tau^2 I + K_{h,n})^{-1}k_{h,n}(z) \tag{9}$$

where $k_{h,n}(z) = [k(z, z_{h,1}), k(z, z_{h,2}), \cdots, k(z, z_{h,n})]^\top$ is the vector of kernel values between the state-action of interest and past observations in $\mathcal{D}_{h,n}$, and $K_{h,n} = [k(z_{h,i}, z_{h,j})]_{i,j=1}^n$ is the Gram matrix of pairwise kernel values between past observations in $\mathcal{D}_{h,n}$. Equipped with $\sigma_{h,n}(z)$, at step h, we select

$$s_{h,n}, a_{h,n} = \underset{s\in\mathcal{S}, a\in\mathcal{A}}{\arg\max}\, \sigma_{h,n-1}(s,a), \tag{10}$$

and observe the next state $s'_{h+1,n} \sim P_h(\cdot|s_{h,n}, a_{h,n})$. We then add this data point to the dataset and update $\mathcal{D}_{h,n} = \mathcal{D}_{h,n-1} \cup \{(s_{h,n}, a_{h,n}, s'_{h+1,n})\}$. For a pseudocode, see Algorithm 2.

We highlight that the selection rule (10) relies on the generative model that allows the algorithm to deviate from the Markovian trajectory and move to a state of its choice. Since observations $(s_{h,n}, a_{h,n})$ are selected based on maximizing $\sigma_{h,n-1}$, which by definition (9) does not depend on previous transitions $\{s'_{h+1,i}\}_{i=1}^{n-1}$, the statistical independence conveniently holds. The generative model setting is feasible in contexts such as games, where the player can manually set the current state. However, this may not always be possible in other scenarios. Next, we introduce our online algorithm, which strictly stays on the Markovian trajectory.

3.2.2 Exploration without Generative Models

In this section, we show that a straightforward algorithm, in contrast to existing approaches, achieves near-optimal performance in an online setting without requiring a generative model. Compared to the scenario with a generative model, the sample complexity of this algorithm increases by a factor of H. For a detailed and technical comparison with existing work, please refer to Appendix A.1.

Our online algorithm operates as follows: in each exploration episode, only one data point specific to a step h is collected —this accounts for the H scaling in sample complexity. This observation however is collected in an unbiased way, which eventually leads to tighter performance guarantees. Specifically, at episode $nH + h_0$, where $n \in [N]$ and $h_0 \in [H]$, the algorithm collects an informative sample for the transition at step h_0. This results in a total of N samples at each step over NH episodes. The algorithm initializes $V_{h_0+1,(n,h_0)} = \mathbf{0}$. Let $\hat{f}_{h,(n,h_0)}$ and $\sigma_{h,n}$ represent the predictor and uncertainty estimator for $[P_h V_{h+1,(n,h_0)}]$, respectively. These are derived from the historical data $\mathcal{D}_{h,n-1}$ of observations at step h. Specifically,

$$\hat{f}_{h,(n,h_0)}(z) = k_{h,n}^\top(z)(K_{h,n} + \tau^2 I)^{-1}\boldsymbol{y}_{h,n},$$
$$\sigma_{h,n}^2(z) = k(z,z) - k_{h,n}^\top(z)(K_{h,n} + \tau^2 I)^{-1}k_{h,n}(z), \tag{11}$$

where $k_{h,n}(z) = [k(z, z_{h,1}), k(z, z_{h,2}), \cdots, k(z, z_{h,n})]^\top$ is the vector of kernel values between the state-action of interest and past observations in $\mathcal{D}_{h,n}$, $K_{h,n} = [k(z_{h,i}, z_{h,j})]_{i,j=1}^n$ is the Gram matrix of pairwise kernel values between past observations in $\mathcal{D}_{h,n}$, and

$$\boldsymbol{y}_{h,(n,h_0)} = [V_{h+1,(n,h_0)}(s_{h+1,1}), V_{h+1,(n,h_0)}(s_{h+1,2}), \cdots, V_{h+1,(n,h_0)}(s_{h+1,n})]^\top$$

is the vector of observations. We then have

$$Q_{h,(n,h_0)} = \Pi_{0,H}\left[\hat{f}_{h,(n,h_0)} + \beta(\delta)\sigma_{h,n}\right],$$
$$V_{h,(n,h_0)}(\cdot) = \max_{a\in\mathcal{A}} Q_{h,(n,h_0)}(\cdot, a).$$

The values of $Q_{h,(n,h_0)}$ and $V_{h,(n,h_0)}$ are obtained recursively for all $h \in [h_0]$. The exploration policy at episode $nH + h_0$ is then the greedy policy with respect to $Q_{h,(n-1,h_0)}$. The dataset is updated by adding the new observation to the dataset for step h_0, such that $\mathcal{D}_{h_0,n} = \mathcal{D}_{h_0,n-1} \cup \{(s_{h_0,n}, a_{h_0,n}, s_{h_0+1,n})\}$, while datasets for all other steps remain unchanged: $\mathcal{D}_{h,n} = \mathcal{D}_{h,n-1}$ for all $h \neq h_0$. This specific update ensures that the collected samples are unbiased. More specifically, the sample collected at h_0 solely relies on the uncertainty $\sigma_{h_0,n}$, due to the initialization $V_{h_0+1,(n,h_0)} = \mathbf{0}$ which implies $\hat{f}_{h_0,(n,h_0)} = \mathbf{0}$. Since $\sigma_{h_0,n}$ does not depend on previous transitions $s_{(h_0+1,i)}$ for any $i \leq n$, the samples at $h = h_0$ are unbiased. However, for $h < h_0$, the samples depend on both the uncertainty $\sigma_{h,n}$ and the prediction $\hat{f}_{h,(n,h_0)}$ (11). Since the prediction depends on the transitions $s_{(h+1,i)}$ for $i \leq n$, these samples are biased. As a result, we discard them and only retain the unbiased samples at $h = h_0$. This approach improves the rates in our analysis, albeit at the cost of a factor of H.

3.3 Computational Complexity

The main computational bottleneck is the matrix inversion in kernel ridge regression, which incurs a cost of $\mathcal{O}(n^3)$, leading to a total complexity of $\mathcal{O}(N^4)$ for our algorithms. This is comparable to the complexities in related work, such as Vakili and Olkhovskaya (2023) and Qiu et al. (2021). Notably, the $\mathcal{O}(n^3)$ cost of matrix inversion is not unique to RL but is common across kernel-based supervised learning and bandit literature. Sparse approximation methods, such as Sparse Variational Gaussian Processes (SVGP) and the Nyström method, can significantly reduce this complexity (in some cases, to linear time) while preserving kernel-based confidence intervals and corresponding rates (e.g., Vakili et al., 2022). However, since these methods are broadly applicable rather than specific to our setting, we chose to maintain a clear, notation-light presentation focused on our main contributions.

4 ANALYSIS OF THE SAMPLE COMPLEXITY

In this section, we present our main results on the sample complexity of the algorithms. We first establish a novel confidence interval that is applicable to the unbiased samples collected by our exploration algorithms. We then provide theorems detailing the performance of these algorithms.

4.1 Confidence Intervals

We introduce a novel confidence interval that is tighter than existing ones in our RL setting and can also be applied to other RL problems such as offline RL and infinite-horizon settings.

Theorem 1 (Confidence Bounds)**.** *Consider compact sets $\mathcal{S} \subset \mathbb{R}^{d_s}, \mathcal{A} \subset \mathbb{R}^{d_a}$, and define $\mathcal{Z} = \mathcal{S} \times \mathcal{A}$, $d = d_a + d_s$. Consider two Mercer kernels $k_\varphi : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ and $k_\psi : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$. Assume that functions $f : \mathcal{Z} \to \mathbb{R}$ and $V : \mathcal{S} \to \mathbb{R}$, and for each $z \in \mathcal{Z}$, a conditional probability distribution $P(\cdot|z)$ over $\mathcal{S}$, are given such that $f(z) = \mathbb{E}_{s\sim P(\cdot|z)}[V(s)]$, $\|f\|_{\mathcal{H}_{k_\varphi}} \le B_1$, $\|V\|_{\mathcal{H}_{k_\psi}} \le B_2$, and $\max_{s\in\mathcal{S}} V(s) \le v_{\max}$, for some $B_1, B_2, v_{\max} > 0$. Assume a dataset of $\{z_i, s'_i\}_{i=1}^n$ is provided, where each z_i is independent of the set $\{s'_j\}_{j=1}^n$, and $s'_i \sim P(\cdot|z_i)$. Let $\hat{f}^n$ and σ^n be the kernel ridge predictor and uncertainty estimator of f using the observations:*

$$\begin{aligned}\hat{f}_n(z) &= k^\top_{\varphi_n}(z)(\tau^2 I + K_{\varphi_n})^{-1}\boldsymbol{y}_n,\\ \sigma^2_n(z) &= k_\varphi(z,z) - k^\top_{\varphi_n}(z)(\tau^2 I + K_{\varphi_n})^{-1}k_{\varphi_n}(z),\end{aligned} \quad (12)$$

where $\boldsymbol{y}_n = [V(s'_1), V(s'_2), \cdots, V(s'_n))]^\top$. In addition, let λ_m, $m = 1, 2, \cdots$ represent the Mercer eigenvalues of k_ψ in a decreasing order, and ψ_m the corresponding Mercer eigenfunctions. Assume $\psi_m \le \psi_{\max}$ for some $\psi_{\max} > 0$. Fix $M \in \mathbb{N}$, and let C be a constant such that $C \ge \sum_{m=1}^M \lambda_m$.

Then, for a fixed $z \in \mathcal{Z}$, and for all V, with $\|V\|_{\mathcal{H}_{k_\psi}} \le B_2$, we have, the following each hold, with probability at least $1 - \delta$,

$$|f(z) - \hat{f}_n(z)| \le \beta(\delta)\sigma_n(z)$$

with $\beta(\delta) =$

$$B_1 + \frac{CB_2\psi_{\max}}{\tau}\sqrt{2\log(\frac{M}{\delta})} + \frac{2B_2\psi_{\max}}{\tau}\sqrt{n\sum_{m=M+1}^{\infty}\lambda_m} .$$

Theorem 1 provides a confidence bound for kernel ridge regression that is applicable to our RL setting, and is a key result in deriving our sample complexities.

Proof sketch To derive our confidence bounds, we use the Mercer representation of V and decompose the prediction error $f(z) - \hat{f}_n(z)$ into error terms corresponding to each Mercer eigenfunction ψ_m. We then divide these terms into two groups: the first M elements, corresponding to eigenfunctions with the largest eigenvalues, and the remainder. For the top M eigenfunctions, we establish high-probability bounds using standard kernel-based confidence intervals from Vakili et al. (2021a). The remaining terms are bounded based on eigendecay, and we sum over all m to obtain $\beta(\delta)$.

Remark 1. *Under some mild conditions, for example, the polynomial eigendecay given in Definition 2, the following expression can be derived for β:*

$$\beta(\delta) = \mathcal{O}\left(B_1 + \frac{B_2\psi_{\max}}{\tau}\sqrt{\log(\frac{n}{\delta})}\right). \quad (13)$$

With polynomial eigendecay, the remark follows from setting M to $\lceil n^{\frac{1}{p-1}} \rceil$ in the expression of β in Theorem 1.

The confidence interval presented in Theorem 1 is applicable to a fixed $z \in \mathcal{Z}$. Over a discrete domain this can be easily extended to all $z \in \mathcal{Z}$ using a probability union bound and replacing δ with $\frac{\delta}{|\mathcal{Z}|}$ in the expression of $\beta(\delta)$. Using standard discretization techniques, we can also prove a variation of the confidence interval that holds true uniformly over continuous domains. In particular, under the following assumption, we present a variation of the theorem over continuous domains.

Assumption 2. *For each $n \in \mathbb{N}$, there exists a discretization $\mathbb{Z}$ of $\mathcal{Z}$ such that, for any $f \in \mathcal{H}_k$ with $\|f\|_{\mathcal{H}_k} \le B_1$, we have $f(z) - f([z]) \le \frac{1}{n}$, where $[z] = \operatorname{argmin}_{z'\in\mathbb{Z}} ||z' - z||_{l^2}$ is the closest point in $\mathbb{Z}$ to z, and $|\mathbb{Z}| \le cB_1^d n^d$, where c is a constant independent of n and B_1.*

Assumption 2 is a mild technical assumption that holds for typical kernels (Srinivas et al., 2010; Chowdhury and Gopalan, 2017; Vakili et al., 2021a).

Corollary 1. *Under the setting of Theorem 1, and under Assumption 2, the following inequalities each hold uniformly in $z \in \mathcal{Z}$ and $V : \|V\|_{\mathcal{H}_{k_\psi}} \le B_2$, with probability at least $1 - \delta$*

$$\begin{aligned} f(z) &\le \hat{f}_n(z) + \frac{2}{n} + \tilde{\beta}(\delta)(\sigma_n(z) + \frac{2}{\sqrt{n}}),\\ f(z) &\ge \hat{f}_n(z) - \frac{2}{n} - \tilde{\beta}(\delta)(\sigma_n(z) + \frac{2}{\sqrt{n}}),\end{aligned}$$

with $\tilde{\beta}(\delta) = \beta(\frac{\delta}{2c_n})$, $c_n = c(u_n(\frac{\delta}{2}))^d n^d$, and $u_n(\delta) = \mathcal{O}(\sqrt{n + \log(\frac{1}{\delta})})$.

Remark 2. *Under some mild conditions, for example, the polynomial eigendecay given in Definition 2, the* *following expression can be derived for $\tilde{\beta}$:*

$$\tilde{\beta}(\delta) = \mathcal{O}\left(B_1 + \frac{CB_2\psi_{\max}}{\tau}\sqrt{d\log(\frac{n}{\delta})}\right). \tag{14}$$

4.2 Sample Complexities

We have the following theorem on the performance of Algorithm 2. The weakest assumption one can pose on the value functions is realizability, which asserts that the optimal value functions $V_h^\star$ for $h \in [H]$ belong to the RKHS H_{k_ψ} for some kernel $k_\psi : \mathcal{S}\times\mathcal{S}\to\mathbb{R}$, or at least can be well-approximated by H_{k_ψ}. For stateless MDPs or multi-armed bandits where $H=1$, realizability alone is enough to guarantee provably efficient algorithms (Srinivas et al., 2010; Chowdhury and Gopalan, 2017; Vakili et al., 2021a). However, when $H>1$, this assumption appears insufficient (Weisz et al., 2021; Du et al., 2019), and stronger assumptions are typically made in these settings (Jin et al., 2020b; Wang et al., 2019; Chowdhury and Oliveira, 2023). Following these works, our main assumption is a closure property for all value functions in the following class:

$$\mathcal{V} = \left\{ s \mapsto \min\left\{ H, \max_{a\in\mathcal{A}}\left\{ r(s,a) + \varphi^\top(s,a)\boldsymbol{w} + \beta\sqrt{\varphi^\top(s,a)\Sigma^{-1}\varphi(s,a)} \right\}\right\}\right\}, \tag{15}$$

where $0<\beta<\infty$, $\|\boldsymbol{w}\|\le\infty$

and Σ is an $\infty\times\infty$ matrix with $\Sigma \succ \tau^2 I$.

Assumption 3 (Optimistic Closure)**.** *For any $V\in\mathcal{V}$, for some positive constant c_v, we have $\|V\|_{H_{k_\psi}} \leqslant c_v$.*

This is the same assumption as Assumption 1 in Chowdhury and Oliveira (2023) and can be relaxed to value functions ϵ away from this class as described in Section 4.3 of Chowdhury and Oliveira (2023). The assumption ensures that the proxy value functions $(V_{h,n})$ lie within the RKHS of a suitable kernel k_ψ. Notably, the RKHS of widely used kernels, such as Matérn and NT kernels, can uniformly approximate any continuous function over compact subsets of $\mathbb{R}^d$ (Srinivas et al., 2010).

We have the following theorem on the sample complexity of the exploration algorithm with a generative model.

Theorem 2. *Consider the reward-free RL framework described in Section 2. Assume the existence of a generative model in the exploration phase that allows the algorithm to select state-action pairs of its choice at each step. Let N_0 be the smallest integer satisfying*

$$2H\beta(\delta)\sqrt{\frac{2\Gamma(N_0)}{N_0\log(1+1/\tau^2)}} + \frac{4\beta(\delta)H}{\sqrt{N_0}} + \frac{4H}{N_0} \le \epsilon,$$

with $\beta(\delta)=\mathcal{O}(\frac{H}{\tau}\sqrt{d\log(\frac{NH}{\delta})})$ with a sufficiently large constant. Run Algorithm 2 for $N\ge N_0$ episodes to obtain the dataset $\mathcal{D}_N$. Then, use the obtained samples to design a policy π using Algorithm 1 with $\beta(\delta)=\mathcal{O}(\frac{H}{\tau}\sqrt{d\log(\frac{NH}{\delta})})$ with a sufficiently large constant. Then, under Assumptions 1, 2 and 3, with probability at least $1-\delta$, π is guaranteed to be an ϵ-optimal policy.

The following theorem presents the sample complexity for exploration without generative models.

Theorem 3. *Consider the reward free RL framework described in Section 2. Let N_0 be the smallest integer satisfying*

$$3H^2\beta(\delta)\sqrt{\frac{2\Gamma(N_0)}{N_0\log(1+1/\tau^2)}} + \frac{8\beta(\delta)H^2}{\sqrt{N_0}} + \frac{4H^2(\log(N_0)+1)}{N_0} + 2H^2\sqrt{2N_0\log(\frac{3N_0}{\delta})} \le \epsilon \tag{16}$$

with $\beta(\delta)=\mathcal{O}(\frac{H}{\tau}\sqrt{d\log(\frac{NH}{\delta})})$ with a sufficiently large constant. Run Algorithm 3 for $NH\ge N_0H$ episodes to obtain the dataset $\mathcal{D}_N$. Then, use the obtained samples to design a policy π using Algorithm 1. Then, under Assumptions 1, 2 and 3, with probability at least $1-\delta$, π is guaranteed to be an ϵ-optimal policy.

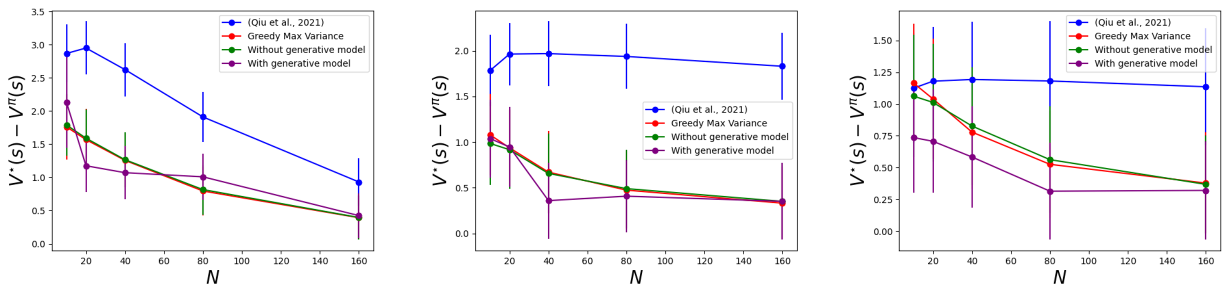
The proof of theorems are provided in Appendix C and D.

The expression of suboptimality gap after N samples, given in (16), can be simplified as

$$\mathcal{O}\left(H^3\sqrt{\frac{\Gamma(N)\log(NH/\delta)}{N}}\right).$$

Remark 3. *Replacing $\Gamma(N)=\tilde{\mathcal{O}}(N^{\frac{1}{p}})$ in the case of kernels with polynomial eigendecay, we obtain a sample complexity of $N=\tilde{\mathcal{O}}((\frac{H^3}{\epsilon})^{2+\frac{2}{p-1}})$. We also recall that without a generative model, we interact with H times more episodes to collect these samples. Specifically, the number of episodes in the exploration phase is $NH=\tilde{\mathcal{O}}\left(H(\frac{H^3}{\epsilon})^{2+\frac{2}{p-1}}\right)$.*

When specialized for the case of Matérn kernels with $p=1+\frac{2d}{\nu}$, we obtain $NH=\tilde{\mathcal{O}}(H(\frac{H^3}{\epsilon})^{2+\frac{d}{\nu}})$ that matches the lower bound for the degenerate case of bandits with $H=1$ proven in Scarlett et al. (2017). Our sample complexity is thus order optimal in terms of ϵ dependency. We also recall that the existing results lead to possibly vacuous (infinite) sample complexities for these kernels.



(a) Squared Exponential Kernel (b) Matérn Kernel with $\nu = 2.5$ (c) Matérn kernel with $\nu = 1.5$

Figure 1: Average suboptimality gap against N. The error bars indicate standard deviation.

5 EXPERIMENTS

We numerically validate our proposed algorithms and compare with the baseline algorithms. From the literature, we implement Qiu et al. (2021), in which the exploration aims at maximizing a hypothetical reward of $\beta\sigma_n/H$ over each episode n. The planning phase is similar to Algorithm 1, but with upper confidence bounds based on kernel ridge regression being used rather than the prediction $\hat{g}_h$. We also implement our exploration algorithms with and without a generative model: Algorithms 2 and 3 respectively. Additionally, we implement a heuristic variation of Algorithm 3, which collects the exploration samples in a greedy manner $a_{h,n} = \arg\max_{a\in\mathcal{A}} \sigma_{h,n}(s_{h,n}, a)$ while remaining on the Markovian trajectory by sampling $s_{h+1} \sim P_h(\cdot|s_h, a_h)$. We refer to this heuristic as *Greedy Max Variance*. For all these algorithms, we use Algorithm 1 to obtain a planning policy. In the experimental setting, we choose $H = 10$ and $\mathcal{S} = \mathcal{A} = [0, 1]$ consisting of 100 evenly spaced points. We choose r and P from the RKHS of a fixed kernel. For the detailed framework and hyperparameters, please refer to Appendix E. We run the experiment for three different kernels across all 4 algorithms for 80 independent runs, and plot the average suboptimality gap $V_1^\star(s) - V_1^\pi(s)$ for $N = 10, 20, 40, 80, 160$, as shown in Figure 1. Our proposed Algorithm 3, without generative model, demonstrates better performance compared to prior work (Qiu et al., 2021) across all three kernels, validating the improved sample efficiency. Notably, Qiu et al. (2021) performs poorly with nonsmooth kernels. Greedy Max Variance is a heuristic that in many of our experiments performs close to Algorithm 3. Furthermore, with access to a generative model, Algorithm 2 performs the best. This is anticipated, as the generative model provides the flexibility to select the most informative state-action pairs, unconstrained by Markovian transitions.

6 CONCLUSION

We proposed novel algorithms for the kernel-based reward-free RL problem, both with and without generative models. We demonstrated that, with a generative model, a simple algorithm can achieve near-optimal sample complexities. Without the generative model requirement, we showed that an online algorithm achieves the same sample complexity up to a factor of H, indicating that the cost of online sampling is a factor of H. Our results apply to a general class of kernels, including those with polynomial eigendecay, where existing methods may either lead to vacuous sample complexities (Qiu et al., 2021) or require additional assumptions and a sophisticated, difficult-to-implement domain partitioning method (Vakili et al., 2024). Our experimental results support these analytical findings. In comparison to the lower bounds proven for the degenerate case of bandits with $H = 1$ for the Matérn kernel, the order optimality of our results in terms of ϵ becomes clear.

References

Abbasi-Yadkori, Y. (2013). Online learning for linearly parametrized control problems.

Agarwal, A., Kakade, S., and Yang, L. F. (2020). Model-based reinforcement learning with a generative model is minimax optimal. In *Conference on Learning Theory*, pages 67–83. PMLR.

Antos, A., Szepesvári, C., and Munos, R. (2008). Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71:89–129.

Auer, P., Jaksch, T., and Ortner, R. (2008). Near-optimal regret bounds for reinforcement learning. *Advances in Neural Information Processing Systems*, 21.

Azar, M. G., Munos, R., and Kappen, H. J. (2013). Minimax pac bounds on the sample complexity of reinforcement learning with a generative model. *Machine learning*, 91:325–349.

Bartlett, P. L. and Tewari, A. (2012). Regal: A regularization based algorithm for reinforcement learning in weakly communicating MDPs. *arXiv preprint arXiv:1205.2661*.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. (2016). Unifying count-based exploration and intrinsic motivation. *Advances in Neural Information Processing Systems*, 29.

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

Chowdhury, S. R. and Gopalan, A. (2017). On kernelized multi-armed bandits. In *International Conference on Machine Learning*, pages 844–853. PMLR.

Chowdhury, S. R. and Gopalan, A. (2019). Online learning in kernelized Markov decision processes. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3197–3205. PMLR.

Chowdhury, S. R. and Oliveira, R. (2023). Value function approximations via kernel embeddings for no-regret reinforcement learning. In *Asian Conference on Machine Learning*, pages 249–264. PMLR.

Christmann, A. and Steinwart, I. (2008). *Support Vector Machines*. Springer New York, NY.

Domingues, O. D., Ménard, P., Pirotta, M., Kaufmann, E., and Valko, M. (2021). Kernel-based reinforcement learning: A finite-time analysis. In *International Conference on Machine Learning*, pages 2783–2792. PMLR.

Du, S. S., Kakade, S. M., Wang, R., and Yang, L. F. (2019). Is a good representation sufficient for sample efficient reinforcement learning? *arXiv preprint arXiv:1910.03016*.

Hazan, E., Kakade, S., Singh, K., and Van Soest, A. (2019). Provably efficient maximum entropy exploration. In *International Conference on Machine Learning*, pages 2681–2691. PMLR.

Hu, P., Chen, Y., and Huang, L. (2022). Towards minimax optimal reward-free reinforcement learning in linear mdps. In *The Eleventh International Conference on Learning Representations*.

Jacot, A., Gabriel, F., and Hongler, C. (2018). Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31.

Jin, C., Allen-Zhu, Z., Bubeck, S., and Jordan, M. I. (2018). Is Q-learning provably efficient? *Advances in Neural Information Processing Systems*, 31.

Jin, C., Krishnamurthy, A., Simchowitz, M., and Yu, T. (2020a). Reward-free exploration for reinforcement learning. In *International Conference on Machine Learning*, pages 4870–4879. PMLR.

Jin, C., Yang, Z., Wang, Z., and Jordan, M. I. (2020b). Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pages 2137–2143. PMLR.

Kakade, S. M. (2003). *On the sample complexity of reinforcement learning*. University of London, University College London (United Kingdom).

Kearns, M. and Singh, S. (1998a). Finite-sample convergence rates for q-learning and indirect algorithms. In *Advances in Neural Information Processing Systems*, volume 11. MIT Press.

Kearns, M. and Singh, S. (1998b). Finite-sample convergence rates for Q-learning and indirect algorithms. *Advances in Neural Information Processing Systems*, 11.

Lattimore, T. (2023). A lower bound for linear and kernel regression with adaptive covariates. In *The Thirty Sixth Annual Conference on Learning Theory*, pages 2095–2113. PMLR.

Lee, J. and Oh, M.-h. (2023). Demystifying linear mdps and novel dynamics aggregation framework. In *The Twelfth International Conference on Learning Representations*.

Levine, S., Kumar, A., Tucker, G., and Fu, J. (2020). Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *ArXiv*, abs/2005.01643.

Mercer, J. (1909). 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.

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

Neu, G. and Pike-Burke, C. (2020). A unifying view of optimism in episodic reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1392–1403.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. (2017). Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pages 2778–2787. PMLR.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Precup, D. (2000). Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80.

Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Qiu, S., Ye, J., Wang, Z., and Yang, Z. (2021). On reward-free rl with kernel and neural function approximations: Single-agent MDP and Markov Game. In *International Conference on Machine Learning*, pages 8737–8747. PMLR.

Russo, D. (2019). Worst-case regret bounds for exploration via randomized value functions. *Advances in Neural Information Processing Systems*, 32.

Scarlett, J., Bogunovic, I., and Cevher, V. (2017). Lower bounds on regret for noisy Gaussian process bandit optimization. In *Conference on Learning Theory*, pages 1723–1742. PMLR.

Sidford, A., Wang, M., Wu, X., Yang, L., and Ye, Y. (2018a). Near-optimal time and sample complexities for solving markov decision processes with a generative model. *Advances in Neural Information Processing Systems*, 31.

Sidford, A., Wang, M., Wu, X., and Ye, Y. (2018b). Variance reduced value iteration and faster algorithms for solving markov decision processes. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 770–787. SIAM.

Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. (2010). Gaussian process optimization in the bandit setting: No regret and experimental design. In *International Conference on Machine Learning*.

Vakili, S. (2024). Open problem: Order optimal regret bounds for kernel-based reinforcement learning. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 5340–5344. PMLR.

Vakili, S., Bouziani, N., Jalali, S., Bernacchia, A., and Shiu, D.-s. (2021a). Optimal order simple regret for Gaussian process bandits. *Advances in Neural Information Processing Systems*, 34:21202–21215.

Vakili, S., Bromberg, M., Garcia, J., Shiu, D.-s., and Bernacchia, A. (2023). Information gain and uniform generalization bounds for neural kernel models. In *2023 IEEE International Symposium on Information Theory (ISIT)*, pages 555–560. IEEE.

Vakili, S., Khezeli, K., and Picheny, V. (2021b). On information gain and regret bounds in gaussian process bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 82–90. PMLR.

Vakili, S., Nabiei, F., Shiu, D.-s., and Bernacchia, A. (2024). Reward-free kernel-based reinforcement learning. In *Forty-first International Conference on Machine Learning*.

Vakili, S. and Olkhovskaya, J. (2023). Kernelized reinforcement learning with order optimal regret bounds. *Advances in Neural Information Processing Systems*, 36.

Vakili, S., Scarlett, J., Shiu, D.-s., and Bernacchia, A. (2022). Improved convergence rates for sparse approximation methods in kernel-based learning. In *International Conference on Machine Learning*, pages 21960–21983. PMLR.

Wagenmaker, A. J., Chen, Y., Simchowitz, M., Du, S., and Jamieson, K. (2022). Reward-free rl is no harder than reward-aware rl in linear markov decision processes. In *International Conference on Machine Learning*, pages 22430–22456. PMLR.

Wang, R., Du, S. S., Yang, L., and Salakhutdinov, R. R. (2020). On reward-free reinforcement learning with linear function approximation. *Advances in Neural Information Processing Systems*, 33:17816–17826.

Wang, Y., Wang, R., Du, S. S., and Krishnamurthy, A. (2019). Optimism in reinforcement learning with generalized linear function approximation. *arXiv preprint arXiv:1912.04136*.

Weisz, G., Amortila, P., and Szepesvári, C. (2021). Exponential lower bounds for planning in mdps with linearly-realizable optimal action-value functions. In *Algorithmic Learning Theory*, pages 1237–1264. PMLR.

Whitehouse, J., Ramdas, A., and Wu, S. Z. (2023). On the sublinear regret of GP-UCB. *Advances in Neural Information Processing Systems*, 36.

Xie, T., Cheng, C.-A., Jiang, N., Mineiro, P., and Agarwal, A. (2021). Bellman-consistent pessimism for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34:6683–6694.

Yang, L. and Wang, M. (2019). Sample-optimal parametric q-learning using linearly additive features. In *International conference on machine learning*, pages 6995–7004. PMLR.

Yang, L. and Wang, M. (2020). Reinforcement learning in feature space: Matrix bandit, kernels, and regret bound. In *International Conference on Machine Learning*, pages 10746–10756. PMLR.

Yang, Z., Jin, C., Wang, Z., Wang, M., and Jordan, M. (2020). Provably efficient reinforcement learning with kernel and neural function approximations. *Advances in Neural Information Processing Systems*, 33:13903–13916.

Yao, H., Szepesvári, C., Pires, B. A., and Zhang, X. (2014). Pseudo-MDPs and factored linear action models. In *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 1–9. IEEE.

Yeh, S.-Y., Chang, F.-C., Yueh, C.-W., Wu, P.-Y., Bernacchia, A., and Vakili, S. (2023). Sample complexity of kernel-based q-learning. In *International Conference on Artificial Intelligence and Statistics*, pages 453–469. PMLR.

Zanette, A., Brandfonbrener, D., Brunskill, E., Pirotta, M., and Lazaric, A. (2020). Frequentist regret bounds for randomized least-squares value iteration. In *International Conference on Artificial Intelligence and Statistics*, pages 1954–1964. PMLR.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes, we describe the mathematical setting, preliminaries, and problem formulation in Section 2. We present the proposed algorithms in Section 3 with their corresponding pseudocodes. All theorems and assumptions are stated in Section 4. Theorem 1 is self-contained, with all necessary assumptions explicitly mentioned in the body of the theorem. It is formulated to be broadly applicable to other problems. Theorems 2 and 3 rely on Assumptions 2 and 3, which are clearly stated in Section 4.]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes, detailed analysis about the confidence intervals and sample complexities of our proposed algorithms are provided in Theorems 1, 2 and 3 of Section 4. However, we do not provide analysis for time and space complexities.]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes, the code used to conduct our experiments is included in a zip file as supplementary material. It also contains a README file and a requirements file to facilitate the installation of all necessary packages.]
2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes, we clearly state the full set of assumptions (Assumptions 2 and 3) in Section 4.]
   (b) Complete proofs of all theoretical results. [Yes, we provide the detailed proofs of Theorems 1, 2 and 3 in Appendices B, C, D, respectively.]
   (c) Clear explanations of any assumptions. [Yes, we explicitly state the assumptions of Theorem 1 within the main body of the theorem, making it self-contained. Assumptions 2 and 3, which are necessary for Theorems 2 and 3, are clearly articulated and explained separately.]
3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes, we provide the code as an anonymized zip file in the supplementary material, along with a Readme file that instructs the user on how to run the code.]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes, the main paper provides a core explanation of the results, including the algorithms tested, kernels used, and the synthetic framework in Section 5. For comprehensive details (hyperparameter-tuning, visualizations of the reward and transition probability functions), please refer to Appendix E.]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to

the random seed after running experiments multiple times). [Yes, our results are accompanied by error bars indicating the standard deviation across 80 independent runs of our experiments.]

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes, we include in Appendix E the computational resources required for our experiments.]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

   (a) Citations of the creator If your work uses existing assets. [Yes, we have used the scikit-learn library to implement our kernel-based algorithms, and we have properly cited it (See Appendix E).]

   (b) The license information of the assets, if applicable. [Not applicable]

   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes, we submit the code generating our experimental results as a zip file in the supplementary material.]

   (d) Information about consent from data providers/curators. [Not Applicable]

   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

   (a) The full text of instructions given to participants and screenshots. [Not Applicable]

   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A Related Work

In this Section, we first provide a technical comparison between our work and that of Qiu et al. (2021) and Vakili et al. (2024). Following this, we present a more comprehensive literature review, including related works that were not covered in the main paper. We also include a summary table of the sample complexity results in the reward-free RL setting, highlighting our key contributions.

A.1 Comparison to the Existing Work

Here, we discuss the key differences between our approach and the closely related works of Qiu et al. (2021) and Vakili et al. (2024). In Qiu et al. (2021), they conduct exploration by accumulating standard deviation over an episode, then they apply a planning phase-like algorithm to maximize a reward proportional to $\beta(\delta)\sigma_{h,n}$ at each step of an episode. However, this approach can inflate the confidence interval width multiplier $\beta(\delta)$ by a factor of $\sqrt{\Gamma(n)}$, potentially leading to suboptimal or even trivial sample complexities when $\sqrt{\Gamma(n)}$ is large, as seen in Qiu et al. (2021). Specifically, their results are applicable to very smooth kernels like SE, with exponentially decaying Mercer eigenvalues, for which $\Gamma(n) = \mathcal{O}(\text{polylog}(n))$. For kernels with polynomial eigendecay, where $\Gamma(n) = \mathcal{O}(n^{\frac{1}{p+1}})$ grows polynomially with n, this algorithm possibly leads to trivial (infinite) sample complexities. Intuitively, the inflation of $\beta(\delta)$ is due to the adaptive sampling creating statistical dependencies among observations, specifically through next state transitions. When such dependencies exist, the best existing confidence intervals are based on a kernel adaptation of self-normalized vector values martingales (Abbasi-Yadkori, 2013). The $\sqrt{\Gamma(n)}$ term cannot be removed in general for adaptive samples that introduce bias, as was discussed in Vakili (2024) and Lattimore (2023).

Vakili et al. (2024) utilizes domain partitioning, relying on only a subset of samples to obtain confidence intervals. This approach achieves order-optimal sample complexity for kernels with polynomial eigendecay, offering an H-factor improvement compared to our work in the online setting. However, firstly, their results are limited by specific assumptions regarding the relationship between kernel eigenvalues and domain size, which reduces the generality of their findings. Secondly, their domain partitioning method is cumbersome to implement and lacks practical justification, as it requires dropping samples from other subdomains. In contrast, our algorithm achieves order-optimal results for general kernels with a simpler approach that leverages statistical independence. Moreover, our method is well-suited to the generative setting, where their approach offers no clear advantages.

A.2 Literature Review

Table 1: Existing sample complexities in reward-free RL. $\mathcal{S}$, $\mathcal{A}$, H, d and p represent the state space, action space, episode length, state-action space dimension and parameter of the kernel with polynomial eigendecay, respectively. Last two rows correspond to the performance guarantees for the algorithms proposed in this work.

Setting	Sample complexity
Tabular (Jin et al., 2020a)	$\mathcal{O}\left(\frac{\lvert\mathcal{S}\rvert^2\lvert\mathcal{A}\rvert H^5}{\epsilon^2}\right)$
Linear (Wang et al., 2020)	$\tilde{\mathcal{O}}\left(\frac{d^3H^6}{\epsilon^2}\right)$
Kernel-based (exponential eigendecay) (Qiu et al., 2021)	$\mathcal{O}\left(\frac{H^6\text{polylog}(\frac{1}{\epsilon})}{\epsilon^2}\right)$
Kernel-based (polynomial eigendecay) (Vakili et al., 2024)	$\tilde{\mathcal{O}}\left((\frac{H^3}{\epsilon})^{2+\frac{2}{p-1}}\right)$
Kernel-based (exponential eigendecay) (this work)	$\tilde{\mathcal{O}}\left(\frac{H^7\text{polylog}(\frac{1}{\epsilon})}{\epsilon^2}\right)$
Kernel-based (polynomial eigendecay) (this work)	$\tilde{\mathcal{O}}\left(H(\frac{H^3}{\epsilon})^{2+\frac{2}{p-1}}\right)$

Numerous studies have addressed the sample complexity problem in the discounted MDP framework with an infinite horizon, where the agent has sampling access to a generative model, such as (Kearns and Singh, 1998b; Azar et al., 2013; Agarwal et al., 2020). Alternatively, other research has focused on the episodic MDP framework, without reliance on a generative model or an exploratory policy. Both the tabular setting (Jin et al., 2018; Auer et al., 2008; Bartlett and Tewari, 2012) and the linear setting (Jin et al., 2020b; Yao et al., 2014; Russo, 2019; Zanette et al., 2020; Neu and Pike-Burke, 2020) have been thoroughly examined. Recent literature has extended these techniques to the kernel setting (Yang et al., 2020; Yang and Wang, 2020; Chowdhury and Gopalan, 2019; Domingues et al., 2021; Vakili and Olkhovskaya, 2023), although further improvements are needed in achieving better regret bounds. In contrast to these prior works which assume that the reward function is provided, we explore the episodic reward-free setting in this work, both with and without a generative model. This setting is significantly different from standard RL, rendering the existing sample complexity results inapplicable to our context.

In the context of reward-free RL, numerous empirical studies have proposed various exploration methods from a practical perspective, as demonstrated by works such as (Bellemare et al., 2016; Pathak et al., 2017; Hazan et al., 2019). Theoretically, researchers have explored the reward-free RL framework across different levels of complexity, ranging from tabular to linear, kernel-based, and deep learning-based models (Jin et al., 2020a; Wang et al., 2020; Qiu et al., 2021) (Table 1). Although the existing literature adequately covers the tabular and linear settings, it often provides only partial and incomplete findings when addressing the more intricate kernel-based and deep learning settings. The most relevant work in the kernel setting is Qiu et al. (2021), which provides a reward-free algorithm whose sample complexity is $\mathcal{O}\left(\frac{H^6\text{polylog}(\frac{1}{\epsilon})}{\epsilon^2}\right)$. Their results however are only applicable to very smooth kernels with exponentially decaying eigenvalues. The recent work of Vakili et al. (2024) proved a sample complexity of $\tilde{\mathcal{O}}\left((\frac{H^3}{\epsilon})^{2+\frac{2}{p-1}}\right)$ for kernels with polynomial eigendecay. However, they employ a niche domain partitioning technique that, despite its theoretical appeal, is cumbersome to implement and raises practical concerns, as mentioned earlier.

Finally, it's important to mention that the planning phase of our proposed algorithm is similar to the problem of learning a good policy from predefined datasets, typically called batch or offline RL (Levine et al., 2020). Many prior works on offline RL make the coverage assumption on the dataset, requiring it to sufficiently include any possible state-action pairs with a minimum probability (Precup, 2000; Antos et al., 2008; Chen and Jiang, 2019; Munos and Szepesvári, 2008). These works do not address the exploration needed to achieve such good coverage, which is where our reward-free approach significantly differs. Our goal is to demonstrate how to collect sufficient exploration data without any reward information, enabling the design of a near-optimal policy for any reward function during the planning phase.

B Proof of Theorem 1 and Corollary 1

For the proof of Theorem 1, we leverage the fact that V belongs to an RKHS. Specifically, we use the Mercer representation of V

$$V(s) = \sum_{m=1}^{\infty} w_m \lambda_m^{\frac{1}{2}} \psi_m(s). \tag{17}$$

We can also rewrite the observations in the observation vector $\boldsymbol{y}_n$ as the sum of a noise term and the expected value of the observation (noise free part).

$$V(s_i') = \underbrace{(V(s_i') - f(z_i))}_{\text{Observation noise}} + \underbrace{f(z_i)}_{\text{Noise-free observation}} \tag{18}$$

Using the notation $\overline{\psi}_m(z) = \mathbb{E}_{s'\sim P(\cdot|z)}\psi(s')$, we can rewrite $f(z_i)$ as follows

$$\begin{aligned} f(z_i) &= \mathbb{E}_{s\sim P(\cdot|z_i)}[V(s)] \\ &= \mathbb{E}_{s\sim P(\cdot|z_i)}\left[\sum_{m=1}^{\infty} w_m\lambda_m^{\frac{1}{2}}\psi_m(s)\right] \\ &= \sum_{m=1}^{\infty} w_m\lambda_m^{\frac{1}{2}}\mathbb{E}_{s\sim P(\cdot|z_i)}[\psi_m(s)] \\ &= \sum_{m=1}^{\infty} w_m\lambda_m^{\frac{1}{2}}\overline{\psi}_m(z_i) \end{aligned} \tag{19}$$

We then use the following notations, $\varepsilon_i = V(s_i') - f(z_i)$, $\boldsymbol{\varepsilon}_n = [\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_n]^\top$, $\boldsymbol{f}_n = [f(z_1), f(z_2), \cdots, f(z_n)]^\top$, to rewrite the prediction error

$$\begin{aligned} f(z) - \hat{f}_n(z) &= f(z) - k_n^\top(z)(\tau^2 I + K_n)^{-1}\boldsymbol{y}_n \\ &= f(z) - k_n^\top(z)(\tau^2 I + K_n)^{-1}(\boldsymbol{\varepsilon}_n + \boldsymbol{f}_n) \\ &= \underbrace{f(z) - k_n^\top(z)(\tau^2 I + K_n)^{-1}\boldsymbol{f}_n}_{\text{Prediction error from noise-free observations}} \underbrace{- k_n^\top(z)(\tau^2 I + K_n)^{-1}\boldsymbol{\varepsilon}_n}_{\text{The error due to noise}} \end{aligned}$$

The first term is deterministic (not random) and can be bounded following the standard approaches in kernel-based models, for example using the following result from Vakili et al. (2021a). Let us use the notations

$$\boldsymbol{\zeta}_n(z) = k_n^\top(z)(\tau^2 I + K_n)^{-1}$$

and $\zeta_i(z) = [\boldsymbol{\zeta}_n(z)]_i$.

Lemma 1 (Proposition 1 in Vakili et al. (2021a))**.** *We have*

$$\sigma_n^2(z) = \sup_{f:\|f\|_{\mathcal{H}}\leq 1} (f(z) - \boldsymbol{\zeta}_n^\top(z)\boldsymbol{f}_n)^2 + \tau^2\|\boldsymbol{\zeta}_n(z)\|_{\ell^2}^2.$$

Based on this lemma, the first term can be deterministically bounded by $B_1\sigma_n(z)$:

$$|f(z) - k_n^\top(z)(\tau^2 I + K_n)^{-1}\boldsymbol{f}_n| \leq B_1\sigma_n(z) \tag{20}$$

We next bound the second term, the error due to noise.

$$\begin{aligned} k_n^\top(z)(\tau^2 I + K_n)^{-1}\boldsymbol{\varepsilon}_n &= \sum_{i=1}^{n}\zeta_i(z)\varepsilon_i \\ &= \sum_{i=1}^{n}\zeta_i(z)(\sum_{m=1}^{\infty} w_m\lambda_m^{\frac{1}{2}}\psi_m(s_i') - \sum_{m=1}^{\infty} w_m\lambda_m^{\frac{1}{2}}\overline{\psi}_m(z_i)) \\ &= \sum_{m=1}^{\infty} w_m\lambda_m^{\frac{1}{2}}\sum_{i=1}^{n}\zeta_i(z)(\psi_m(s_i') - \overline{\psi}_m(z_i)) \\ &= \sum_{m=1}^{M} w_m\lambda_m^{\frac{1}{2}}\sum_{i=1}^{n}\zeta_i(z)(\psi_m(s_i') - \overline{\psi}_m(z_i)) + \sum_{m=M+1}^{\infty} w_m\lambda_m^{\frac{1}{2}}\sum_{i=1}^{n}\zeta_i(z)(\psi_m(s_i') - \overline{\psi}_m(z_i)) \end{aligned}$$

We note that $\psi_m(s_i') - \overline{\psi}_m(z_i)$ are bounded random variables with a range of $2\psi_{\max}$. Using Chernoff-Hoeffding inequality and the bound on the norm of $\boldsymbol{\zeta}_n$ provided in Lemma 1, we have that with probability at least $1 - \delta/M$

$$\sum_{i=1}^{n}\zeta_i(z)(\psi_m(s_i') - \overline{\psi}_m(z_i)) \leq \frac{\psi_{\max}\sigma_n(z)}{\tau}\sqrt{2\log(\frac{M}{\delta})}.$$

Using a probability union bound, with probability $1-\delta$

$$\begin{aligned}
&\sum_{m=1}^{M} w_m \lambda_m^{\frac{1}{2}} \sum_{i=1}^{n} \zeta_i(z)(\psi_m(s_i') - \overline{\psi}_m(z_i)) \\
&\leq \sum_{m=1}^{M} w_m \lambda_m^{\frac{1}{2}} \frac{\psi_{\max}\sigma_n(z)}{\tau} \sqrt{2\log\left(\frac{M}{\delta}\right)} \\
&\leq \left(\sum_{m=1}^{M} \lambda_m\right)^{\frac{1}{2}} \left(\sum_{m=1}^{M} w_m^2\right)^{\frac{1}{2}} \frac{\psi_{\max}\sigma_n(z)}{\tau} \sqrt{2\log\left(\frac{M}{\delta}\right)} \\
&\leq \left(\sum_{m=1}^{M} \lambda_m\right)^{\frac{1}{2}} B_2 \frac{\psi_{\max}\sigma_n(z)}{\tau} \sqrt{2\log\left(\frac{M}{\delta}\right)} \\
&\leq C B_2 \frac{\psi_{\max}\sigma_n(z)}{\tau} \sqrt{2\log\left(\frac{M}{\delta}\right)}
\end{aligned}$$

The second inequality is based on the Cauchy-Schwarz inequality. In the third inequality, we used that B_2 is the upper bound on the RKHS norm of V. In the last inequality, we used the observation that under both polynomial eigenvalue decay with $p > 1$ and exponential eigendecay, the sum of the eigenvalues is bounded by an absolute constant C, independent of M.

Also, for the second term, we have

$$\begin{aligned}
\sum_{m=M+1}^{\infty} w_m \lambda_m^{\frac{1}{2}} \sum_{i=1}^{n} \zeta_i(z) \left(\psi_m(s_i') - \overline{\psi}_m(z_i)\right) &\leq 2\psi_{\max} \sum_{m=M+1}^{\infty} w_m \lambda_m^{\frac{1}{2}} \sum_{i=1}^{n} \zeta_i(z) \\
&\leq 2\psi_{\max} \sum_{m=M+1}^{\infty} w_m \lambda_m^{\frac{1}{2}} \left(n \sum_{i=1}^{n} \zeta_i^2(z)\right)^{\frac{1}{2}} \\
&\leq \frac{2\sigma_n(z)\psi_{\max}\sqrt{n}}{\tau} \sum_{m=M+1}^{\infty} w_m \lambda_m^{\frac{1}{2}} \\
&\leq \frac{2\sigma_n(z)\psi_{\max}\sqrt{n}}{\tau} \left(\left(\sum_{m=M+1}^{\infty} w_m^2\right)\left(\sum_{m=M+1}^{\infty} \lambda_m\right)\right)^{\frac{1}{2}} \\
&\leq \frac{2B_2\sigma_n(z)\psi_{\max}}{\tau} \left(n \sum_{m=M+1}^{\infty} \lambda_m\right)^{\frac{1}{2}}.
\end{aligned}$$

The first inequality holds by definition of $\psi_{\max}$. The second inequality is based on the Cauchy-Schwarz inequality. The third inequality uses Lemma 1. The fourth inequality utilizes the Cauchy-Schwarz inequality again, and the last inequality results from the upper bound on the RKHS norm of V.

Putting together, with probability $1-\delta$,

$$k_n^\top(z)(\tau^2 I + K_n)^{-1}\varepsilon_n \leq \frac{CB_2\psi_{\max}\sigma_n(z)}{\tau}\sqrt{2\log(\frac{M}{\delta})} + \frac{2B_2\sigma_n(z)\psi_{\max}}{\tau}\sqrt{n\sum_{m=M+1}^{\infty}\lambda_m}. \tag{21}$$

Proof of Corollary 1 To extend the confidence interval given in Theorem 1 to hold uniformly on $\mathcal{Z}$, we use a discretization argument. For this purpose, we apply Assumption 2 to f and $\hat{f}_n$, and also use Assumption 2 to bound the discrimination error in σ_n. The following lemma provides a high probability bound on $\|\hat{f}_n\|_{k_\varphi}$.

Lemma 2. *For function f defined in Theorem 1, the RKHS norm of $\hat{f}_n$ satisfies the following with probability at least $1-\delta$:*

$$\|\hat{f}_n\|_{\mathcal{H}_{k_\varphi}} \leq B_1 + \frac{v_{\max}}{\tau}\sqrt{2\Gamma_{k_\varphi}(n) + 1 + \log(\frac{1}{\delta})}. \tag{22}$$

For a proof see Lemma 5 in (Vakili and Olkhovskaya, 2023).

Let $B_3(\delta) = B_1 + \frac{v_{\max}}{\tau}\sqrt{2\Gamma_{k_\varphi}(n) + 1 + \log(\frac{1}{\delta})}$ denote the $1-\delta$ upper confidence bound on $\|\hat{f}_n\|_{H_{k_\varphi}}$. Let $\mathbb{Z}$ be the discretization of $\mathcal{Z}$ specified in Assumption 2 with RKHS norm bound $B_3(\frac{\delta}{2})$. That is for any $g \in \mathcal{H}_{k_\varphi}$ with $\|g\|_{\mathcal{H}_{k_\varphi}} \leq B_3(\frac{\delta}{2})$, we have $g(z) - g([z]) \leq \frac{1}{n}$, where $[z] = \arg\min_{z' \in \mathbb{Z}} \|z' - z\|$ is the closest point in $\mathbb{Z}$ to z, and $|\mathbb{Z}| \leq c_n$, where $c_n = c(B_3(\frac{\delta}{2}))^d n^d$. Applying Assumption 2 to f and $\hat{f}_n$ with this discretization, it holds for all $z \in \mathcal{Z}$ that

$$|f(z) - f([z])| \leq \frac{1}{n}. \tag{23}$$

In addition, by Lemma 2, with probability eat least $1-\delta$

$$|\hat{f}_n(z) - \hat{f}_n([z]) \leq \frac{1}{n}. \tag{24}$$

Furthermore, we have the following lemma, which can roughly be viewed as a Lipschitz continuity property for σ_n.

Lemma 3. *Under Assumption 2, with the discrimination $\mathbb{Z}$ described above, it holds for all $z \in \mathcal{Z}$ that*

$$\sigma_n(z) - \sigma_n([z]) \leq \frac{2}{\sqrt{n}}.$$

Proof of Lemma 3. Using the reproducing property of RKHS, we have

$$\|k_n^\top(z)(K_n + \tau^2 I)^{-1}k_n(\cdot)\|_{\mathcal{H}_k} \leq \frac{k_{\max}\sqrt{n}}{\tau}, \tag{25}$$

where $k_{\max}$ is the maximum value of the kernel. Let us define $q(\cdot,\cdot') = k_n^\top(\cdot)(K_n + \tau^2 I)^{-1}k_n(\cdot')$. We can write

$$\begin{aligned}
&|\sigma_n^2(z) - \sigma_n^2([z])| \\
&= \left|(k(z,z) - q(z,z)) - (k([z],[z]) - q([z],[z]))\right| \\
&= \left|(k(z,z) - q(z,z)) - (k(z,[z]) - q(z,[z])) + (k(z,[z]) - q(z,[z])) - (k([z],[z]) - q([z],[z]))\right| \\
&\leq |k(z,z) - k(z,[z])| + |k(z,[z]) - k([z],[z])| + |q(z,z) - q(z,[z])| - |q(z,[z]) - q([z],[z])| \\
&\leq \frac{4}{n}.
\end{aligned}$$

To obtain a discretization error bound for the standard deviation from that of the variance, we write

$$\begin{aligned}
(\sigma_n(z) - \sigma([z]))^2 &\leq |\sigma_n(z) - \sigma([z])|\,(\sigma_n(z) + \sigma([z])) \\
&= |\sigma_n^2(z) - \sigma^2([z])| \\
&\leq \frac{4}{n}.
\end{aligned}$$

Therefore,

$$|\sigma_n(z) - \sigma([z])| \leq \frac{2}{\sqrt{n}}.$$

□

Applying a probability union bound on the discretization $\mathbb{Z}$ to Theorem 1, and considering the error bounds in (23), (24) and Lemma 3, we arrive at Corollary 1.

C Proof of Theorem 2

First, we define the following high-probability event :

$$\mathcal{E} = \left\{ \forall h \in [H], |\hat{g}_h(z) - [P_h V_{h+1}](z)| \leq \beta(\delta) \left(\sigma_{h,N}(z) + \frac{2}{\sqrt{n}} \right) + \frac{2}{n} \right\}, \tag{26}$$

where $\beta(\delta) = \mathcal{O}\left(\frac{H}{\tau} \sqrt{d \log(\frac{NH}{\delta})} \right)$ as specified in Corollary 1 with $B_1 = \mathcal{O}(H)$ and $B_2 = c_v$. Using Corollary 1, we have $\Pr[\mathcal{E}] \geq 1 - \delta$.

We divide the rest of the analysis into several steps, as outlined next.

Step 1: Under $\mathcal{E}$, with reward r, we bound $V_1^\star(s) - V_1^\pi(s)$ using $V_1(s) - V_1^\pi(s)$, based on the following lemma. Recall that V_h^π and $V_h^\star$ are the value functions of policy π and the optimal policy, respectively, and V_h is the proxy value functions used in Algorithm 1.

Lemma 4. *Under $\mathcal{E}$, we have*

$$V_h^\star(s) - V_h(s) \leq (H + 1 - h)(\frac{2\beta(\delta)}{\sqrt{N}} + \frac{2}{N}). \tag{27}$$

Proof of Lemma 4. The lemma is proven by induction over h, starting from $V_{H+1}^\star = V_{H+1} = \mathbf{0}$. We have

$$\begin{aligned} Q_h^\star(s,a) - Q_h(s,a) &= r_h(s,a) + [P_h V_{h+1}^\star](s,a) - r_h(s,a) - \hat{g}_h(s,a) - \beta(\delta)\sigma_{h,N}(s,a) \\ &\leq [P_h V_{h+1}^\star](s,a) - [P_h V_{h+1}](s,a) + \frac{2\beta(\delta)}{\sqrt{N}} + \frac{2}{N} \\ &= [P_h(V_{h+1}^\star - V_{h+1})](s,a) + \frac{2\beta(\delta)}{\sqrt{N}} + \frac{2}{N} \\ &\leq (H + 1 - h)(\frac{2\beta(\delta)}{\sqrt{N}} + \frac{2}{N}) \end{aligned}$$

The first inequality holds by $\mathcal{E}$, and the second inequality by induction assumption. Then, we have

$$\begin{aligned} V_h^\star(s_h) - V_h(s_h) &= \max_{a \in \mathcal{A}} Q_h^\star(s,a) - \max_{a \in \mathcal{A}} Q_h(s,a) \\ &\leq \max_{a \in \mathcal{A}} \{Q_h^\star(s,a) - Q_h(s,a)\} \\ &\leq (H + 1 - h)(\frac{2\beta(\delta)}{\sqrt{N}} + \frac{2}{N}). \end{aligned}$$

That proves the lemma.

□

Step 2: We also bound $V_1(s) - V_1^\pi(s)$ using the sum of standard deviations for the trajectory generated by the policy.

Lemma 5. *Under $\mathcal{E}$, we have*

$$V_1(s_1) - V_1^\pi(s_1) \leq \mathbb{E}\left[\sum_{h=1}^{H} \left(\beta(\delta) \left(\sigma_{h,N}(s_h, a_h) + \frac{2}{\sqrt{N}} \right) + \frac{2}{N} \right) \right],$$

where the expectation is taken with respect to the trajectory generated by the policy.

Proof of Lemma 5. Note that $V_{H+1} = V_{H+1}^\pi = \mathbf{0}$. We next obtain a recursive relationship for the difference

$V_h(s) - V_h^\pi(s)$.

$$\begin{aligned} V_h(s_h) - V_h^\pi(s_h) &= Q_h\left(s_h, \pi(s_h)\right) - Q_h^\pi\left(s_h, \pi(s_h)\right) \\ &= r\left(s_h, \pi(s_h)\right) + \hat{g}_h\left(s_h, \pi(s_h)\right) + \beta(\delta)\sigma_{h,N}\left(s_h, \pi(s_h)\right) - r\left(s_h, \pi(s_h)\right) - \left[P_h V_{h+1}^\pi\right]\left(s_h, \pi(s_h)\right) \\ &\leq \left[P_h V_{h+1}\right]\left(s_h, \pi(s_h)\right) + 2\beta(\delta)\sigma_{h,N}\left(s_h, \pi(s_h)\right) + \frac{2\beta(\delta)}{\sqrt{N}} + \frac{2}{N} - \left[P_h V_{h+1}^\pi\right]\left(s_h, \pi(s_h)\right), \end{aligned}$$

where the inequality is due to $\mathcal{E}$. Recursive application of the above inequality over $h = H, H-1, \cdots, 1$, we obtain

$$V_1(s_1) - V_1^\pi(s_1) \leq \mathbb{E}_{s_{h+1}\sim P(\cdot|s_h,\pi(s_h)), h<H}\left[\sum_{h=1}^{H} 2\beta(\delta)\sigma_{h,N}\left(s_h, \pi(s_h)\right)\right] + \frac{2H\beta(\delta)}{\sqrt{N}} + \frac{2H}{N}.$$

□

Step 3: By definition, we have $V_1^\pi(s_1; \beta(\delta)\sigma_N) \leq V_1^\star(s_1; \beta(\delta)\sigma_N)$. Note that $V_1^\pi(s_1; \beta(\delta)\sigma_N) = \beta(\delta)\sum_{h=1}^{H}\sigma_{h,N}(s_h, \pi(s_h))$.

Step 4: We have $V_1^\star(s; \beta(\delta)\sigma_N) \leq V_1^\star(s; \beta(\delta)\sigma_n)$. This is due to the observation that $\sigma_{h,n}$ is decreasing in the number n of observations. We note that conditioning on observations only reduces the variance. That is seen from the positive definiteness of the Gram matrix and the formula for kernel ridge uncertainty estimator given in (5).

Step 5: Recall the selection rule in Algorithm 2: $s_{h,n}, a_{h,n} = \arg\max_{s,a}\sigma_{h,n-1}(s,a)$. When exploring with generative model, with this rule of selection, we have $V_1^\star(s_1; \beta(\delta)\sigma_{n-1}) \leq \beta(\delta)\sum_{h=1}^{H}\sigma_{h,n-1}(s_{h,n}, a_{h,n})$.

Step 6: Combining all previous steps, we conclude that, under $\mathcal{E}$,

$$V_1^\star(s) - V_1^\pi(s) \leq \frac{2\beta(\delta)}{N}\sum_{n=1}^{N}\sum_{h=1}^{H}\sigma_{h,n-1}(s_{h,n}, a_{h,n}) + \frac{4\beta(\delta)H}{\sqrt{N}} + \frac{4H}{N}. \tag{28}$$

Step 7: We bound the sum of standard deviations according to the following lemma that is a kernel based version of elliptical potential lemma (Abbasi-Yadkori, 2013).

Lemma 6. *For each h, we have*

$$\sum_{n=1}^{N}\sigma_{h,n-1}^2(s_{h,n}, a_{h,n}) \leq \frac{2\Gamma(N)}{\log(1+1/\tau^2)}. \tag{29}$$

See, e.g., Srinivas et al. (2010) for a proof. Using Cauchy–Schwarz inequality, we obtain

$$\sum_{n=1}^{N}\sigma_{h,n-1}(s_{h,n}, a_{h,n}) \leq \sqrt{\frac{2N\Gamma(N)}{\log(1+1/\tau^2)}}.$$

Step 8: From Steps 6 and 7, we conclude that, π is an ϵ-optimal policy with ϵ no larger than

$$V_1^\star(s) - V_1^\pi(s) \leq 2H\beta(\delta)\sqrt{\frac{2\Gamma(N)}{N\log(1+1/\tau^2)}} + \frac{4\beta(\delta)H}{\sqrt{N}} + \frac{4H}{N}.$$

A simpler expression can be given as

$$V_1^\star(s) - V_1^\pi(s) = \mathcal{O}\left(H^2\sqrt{\frac{\Gamma(N)\log(NH/\delta)}{N}}\right).$$

Now, let N_0 be the smallest integer suh that the right hand side less than ϵ. For any $N \geq N_0$ the suboptimality gap of the policy is at most ϵ. This completes the proof of Theorem 2.

D Proof of Theorem 3

We define the event $\mathcal{E}$ similar to the proof of Theorem 2. The first 4 steps related to the planning phase are exactly the same as in the proof of Theorem 2. The rest of the proof is different and we will present it here.

In addition to $\mathcal{E}$, we define another high-probability event $\mathcal{E}'$ where all the confidence intervals utilized in the exploration hold true. Specifically, we define the following:

$$\mathcal{E}' = \{\forall n \in [N], \forall h \in [H], |\hat{f}_{h,n}(z) - f_{h,n}(z)| \leq \beta(\delta)(\sigma_{h,n}(z) + \frac{2}{\sqrt{n}}) + \frac{2}{n}\}, \tag{30}$$

where $\beta(\delta) = \mathcal{O}\left(\frac{1}{\tau}\sqrt{d\log(\frac{NH}{\delta})}\right)$. Using Corollary 1, we have $\Pr[\mathcal{E}'] \geq 1-\delta$.

The following steps are specific to the exploration without the generative model. We define a reward sequence using $\tilde{\sigma}^{h_0}_{h,n}$ such that $\tilde{\sigma}^{h_0}_{h,n} = \sigma_{h,n}$ when $h = h_0$ and $\tilde{\sigma}^{h_0}_{h,n} = 0$ for all $h \neq h_0$.

Step 5b: We have $V^\star_1(s;\beta(\delta)\sigma_n) \leq \sum_{h_0=1}^{H} V^\star_1(s;\beta(\delta)\tilde{\sigma}^{h_0}_n)$. Note that the optimal policy with rewards $\tilde{\sigma}^{h_0}_{h,n}$ optimizes $\sigma_{h,n}$ at step $h = h_0$, while the optimal policy with rewards σ_n optimizes the sum of $\sigma_{h,n}$ over all steps.

Step 6b: We bound $V^\star_1(s;\beta(\delta)\tilde{\sigma}^{h_0}_n)$ using $V_{1,(n,h_0)}(s)$ where $V_{h,(n,h_0)}$ is the proxy for the value function used in Algorithm 3.

Lemma 7. *Under event $\mathcal{E}'$, for all $s \in \mathcal{S}$,*

$$V^\star_h(s;\beta(\delta)\tilde{\sigma}^{h_0}_n) \leq V_{h,(n,h_0)}(s) + (h_0+1-h)(\frac{2\beta(\delta)}{\sqrt{n}} + \frac{2}{n}).$$

Proof of Lemma 7. The lemma is proven by induction, starting from $V^\star_{h_0+1}(\cdot;\beta(\delta)\tilde{\sigma}^{h_0}_n) = V_{h_0+1,(n,h_0)} = \mathbf{0}$. We have, for $h \leq h_0$

$$\begin{aligned} V^\star_h(s;\beta(\delta)\tilde{\sigma}^{h_0}_n) - V_{h,(n,h_0)}(s) &= \max_{a\in\mathcal{A}} Q^\star_h(s,a;\beta(\delta)\tilde{\sigma}^{h_0}_n) - \max_{a\in\mathcal{A}} Q_{h,(n,h_0)}(s,a) \\ &\leq \max_{a\in\mathcal{A}} \left\{Q^\star_h(s,a;\beta(\delta)\tilde{\sigma}^{h_0}_n) - Q_{h,(n,h_0)}(s,a)\right\} \\ &\leq \max_{a\in\mathcal{A}} \left\{[P_hV^\star_{h+1}](s,a;\beta(\delta)\tilde{\sigma}^{h_0}_n) - [P_hV_{h+1,n}](s,a) + \frac{2\beta(\delta)}{\sqrt{n}} + \frac{2}{n}\right\} \\ &\leq (h_0+1-h)(\frac{2\beta(\delta)}{\sqrt{n}} + \frac{2}{n}). \end{aligned}$$

The fist inequality is due to rearrangement of max, the second inequality holds under $\mathcal{E}'$, and the third inequality is by the base of induction. We thus prove the lemma. □

Step 7b: Fix n and h_0. Let π_{n,h_0} denote the exploration policy in episode $nH + h_0$. We bound $V_{1,(n,h_0)}(s_1) - V^{\pi_{n,h_0}}_1(s_1;\beta(\delta)\tilde{\sigma}^{h_0}_{n-1})$ using $2\beta(\delta)\sum_{h=1}^{h_0}\sigma_{h,n-1}(s_h,a_h)$. Here for simplicity of notation, we use s_h and a_h for the state and action at step h of epsiode corresponding to n and h_0. In a richer notation, n and h_0 should be specified.

Lemma 8. *Under event $\mathcal{E}'$, we have*

$$\begin{aligned} V_{1,(n,h_0)}(s_1) - V^{\pi_{n,h_0}}_1(s_1;\beta(\delta)\tilde{\sigma}^{h_0}_{n-1}) &\leq 2\sum_{h=1}^{h_0}((\beta(\delta)\sigma_{h,n-1}(s_h,a_h) + \frac{2}{\sqrt{n}}) + \frac{2}{n}) \\ + \sum_{h=1}^{h_0}([P_hV_{h+1,n}](s_h,a_h) - V_{h+1,n}(s_{h+1})) &+ \sum_{h=1}^{h_0}(V^{\pi_{n,h_0}}_{h+1}(s_{h+1};\beta(\delta)\tilde{\sigma}^{h_0}_{n-1}) - [P_hV^{\pi_{n,h_0}}_{h+1}](s_h,a_h;\beta(\delta)\tilde{\sigma}^{h_0}_{n-1})). \end{aligned}$$

The second and third terms are martingale sums which can be bounded using Azuma-Hoeffding inequality, we refer to them as

$$\zeta_{h,(n,h_0)} = [P_hV_{h+1,n}](s_h,a_h) - V_{h+1,n}(s_{h+1})$$

$$\xi_{h,(n,h_0)} = V^{\pi_{n,h_0}}_{h+1}(s_{h+1};\beta(\delta)\tilde{\sigma}^{h_0}_{n-1}) - [P_hV^{\pi_{n,h_0}}_{h+1}](s_h,a_h;\beta(\delta)\tilde{\sigma}^{h_0}_{n-1})$$

Proof of Lemma 8. We obtain a iterative relation over h. In particular

$$\begin{aligned} V_{h,(n,h_0)}(s_h) - V_h^{\pi_{n,h_0}}(s_h;\beta(\delta)\tilde{\sigma}_{n-1}^{h_0}) &= Q_{h,(n,h_0)}(s_h,a_h) - Q_h^{\pi_{n,h_0}}(s_h,a_h;\beta(\delta)\tilde{\sigma}_{n-1}^{h_0}) \\ &\le [P_hV_{h+1,n}](s_h,a_h) - [P_hV_{h+1}^{\pi_{n,h_0}}](s_h,a_h;\beta(\delta)\tilde{\sigma}_{n-1}^{h_0}) + 2\beta(\delta)\sigma_{h,n-1}(s_h,a_h) + \frac{2\beta(\delta)}{\sqrt{n}} + \frac{2}{n} \\ &= V_{h+1,(n,h_0)}(s_h) - V_{h+1}^{\pi_{n,h_0}}(s_h;\beta(\delta)\tilde{\sigma}_{n-1}^{h_0}) + 2\beta(\delta)\sigma_{h,n-1}(s_h,a_h) + \frac{2\beta(\delta)}{\sqrt{n}} + \frac{2}{n} + \zeta_{h,(n,h_0)} + \xi_{h,(n,h_0)}. \end{aligned}$$

Iterating over h and noticing $V_{h_0+1,(n,h_0)} - V_{h_0+1}^{\pi_{n,h_0}}(\cdot;\beta(\delta)\tilde{\sigma}_{n-1}^{h_0}) = \mathbf{0}$ the lemma is proven.

□

Step 8b: We note that $V_{1,(n,h_0)}^{\pi_{n,h_0}}(s) = \beta(\delta)\sigma_{h,n-1}(s_{h_0,(n,h_0)}, a_{h_0,(n,h_0)})$.

Step 9b: Combining Steps 5b-8b we conclude that

$$V_1^\star(s,\beta(\delta)\sigma_{n-1}) \le \sum_{h_0=1}^{H}\sum_{h=1}^{h_0}\left(3\beta(\delta)\left(\sigma_{h,n-1}(s_{h,(n,h_0)},a_{h,(n,h_0)})\right) + \frac{4\beta(\delta)}{\sqrt{n}} + \frac{4}{n} + \zeta_{h,(n,h_0)} + \xi_{h,(n,h_0)}\right). \tag{31}$$

Step 10b: Combining with previous steps similar to the proof of Theorem 2, and using Azuma-Hoeffding inequality on $\zeta_{h,(n,h_0)}$ and $\xi_{h,(n,h_0)}$, we get, with probability $1-\delta$

$$V_1^\star(s) - V_1^\pi(s) \le 3H^2\beta(\delta)\sqrt{\frac{2\Gamma(N)}{N\log(1+1/\tau^2)}} + \frac{8\beta(\delta)H^2}{\sqrt{N}} + \frac{4H^2(\log(N)+1)}{N} + 2H^2\sqrt{2N\log(\frac{3N}{\delta})}.$$

The expression can be simplified as

$$V_1^\star(s) - V_1^\pi(s) = \mathcal{O}\left(H^3\sqrt{\frac{\Gamma(N)\log(N/\delta)}{N}}\right).$$

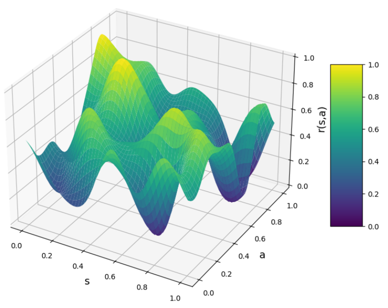
Now, let N_0 be the smallest integer suh that the right hand side less than ϵ. For any $N \ge N_0$ the suboptimality gap of the policy is at most ϵ. This completes the proof of Theorem 3.

E Experimental Details

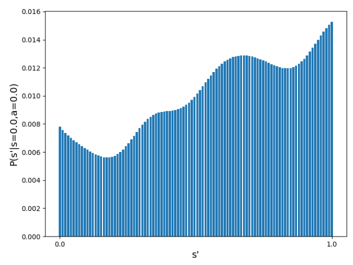
Here, we outline the procedure for generating r and P test functions from the RKHS, the computational resources utilized for the simulations, and the fine-tuning process of the confidence interval width multiplier β. Additionally, we present further experimental results when various samples of r and P are drawn from the RKHS.

E.1 Synthetic Test Functions from the RKHS

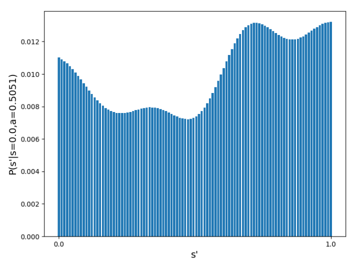
Our reward function r and transition probability P are arbitrarily chosen functions from an RKHS. For the reward function r, we draw a Gaussian Process (GP) sample on a subset of the domain $\mathcal{Z}$. This subset is generated by sampling a set of evenly spaced points on a 10×10 grid spanning the range $[0,1]$ in both dimensions. We then fit kernel ridge regression to these samples and scale the resulting predictions to fit the $[0,1]$ range to obtain r. For $P(s'|s,a)$, we similarly draw a GP sample on a subset of the domain $\mathcal{Z}\times\mathcal{S}$, fit kernel ridge regression to these samples, and then shift and rescale for each z to obtain $P(\cdot|z)$ as a conditional probability distribution. We use the same kernel as the one used in the algorithm. This is a common approach to create functions belonging to an RKHS (e.g., see, Chowdhury and Gopalan, 2017). Examples of r and P are visualized in Figures 2, 3 and 4 using Squared Exponential (SE) and Matérn kernels with parameter $\nu = 2.5$ and $\nu = 1.5$, respectively. For all kernels we use lengthscale of 0.1. With SE kernel we use $\tau = 0.01$, and with Metérn kernels we use $\tau = 0.5$.



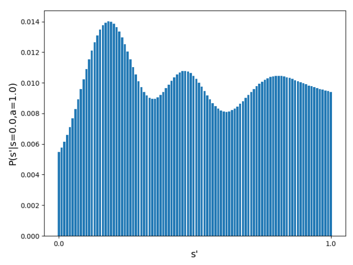
(a) Reward function $r(s,a)$



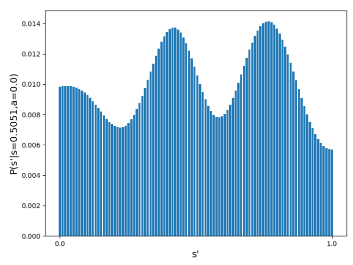
(b) $P(s'|(s=0,a=0)$



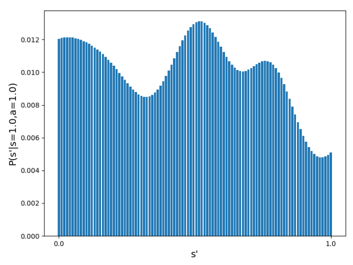
(c) $P(s'|(s=0,a=0.5051)$



(d) $P(s'|(s=0,a=1)$

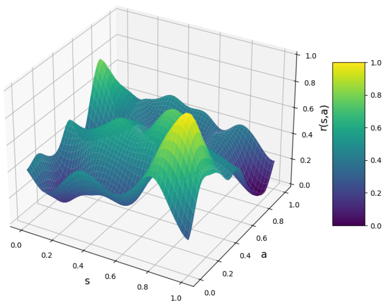


(e) $P(s'|(s=0.5051,a=0)$

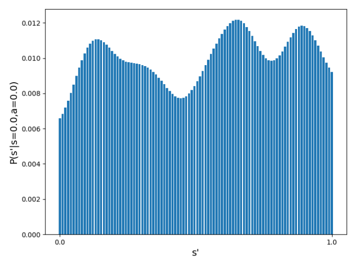


(f) $P(s'|(s=1,a=1)$

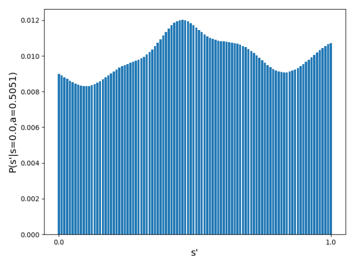
Figure 2: Reward and transition probability functions generated by kernel ridge regression using SE Kernel with lengthscale $= 0.1$ and $\tau = 0.01$



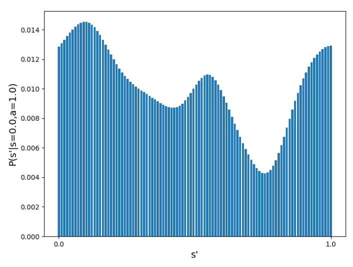
(a) Reward function $r(s,a)$



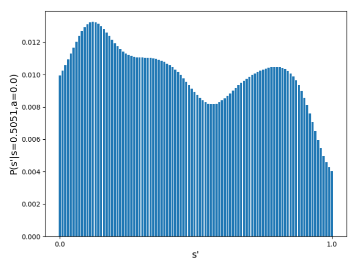
(b) $P(s'|(s=0,a=0)$



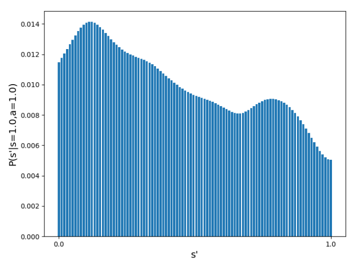
(c) $P(s'|(s=0,a=0.5051)$



(d) $P(s'|(s=0,a=1)$

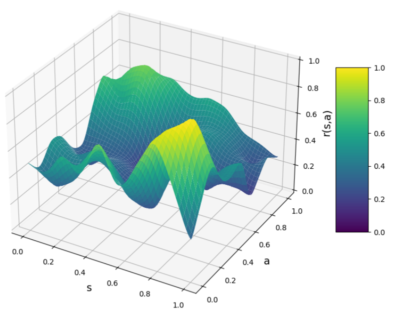


(e) $P(s'|(s=0.5051,a=0)$

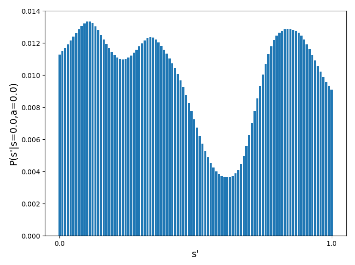


(f) $P(s'|(s=1,a=1)$

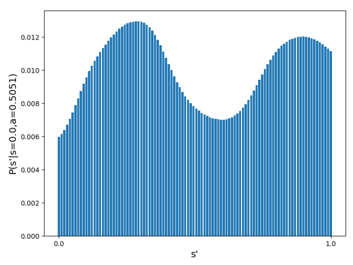
Figure 3: Reward and transition probability functions generated by kernel ridge regression using Matérn kernel with $\nu = 2.5$, lengthscale $= 0.1$ and $\tau = 0.5$



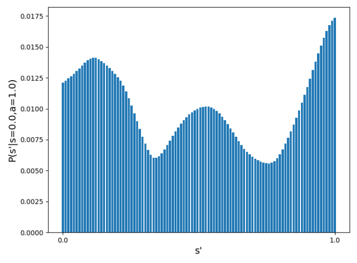
(a) Reward function $r(s, a)$



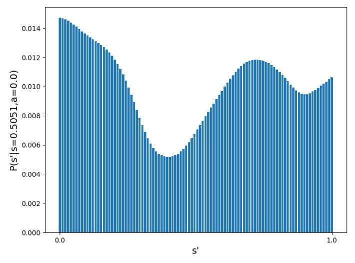
(b) $P(s'|(s = 0, a = 0)$



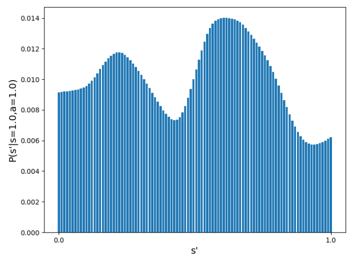
(c) $P(s'|(s = 0, a = 0.5051)$



(d) $P(s'|(s = 0, a = 1)$



(e) $P(s'|(s = 0.5051, a = 0)$



(f) $P(s'|(s = 1, a = 1)$

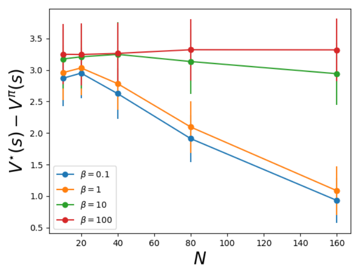
Figure 4: Reward and transition probability functions generated by kernel ridge regression using Matérn kernel with $\nu = 1.5$, lengthscale $= 0.1$ and $\tau = 0.5$

E.2 Implementation and Computational Resources

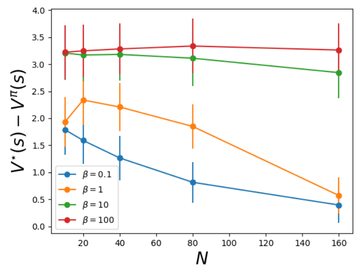
For kernel ridge regression, we used Sickit-Learn library (Pedregosa et al., 2011), which offers robust and efficient tools for implementing and tuning kernel-based machine learning models. The simulations were executed on a cluster which has 376.2 GiB of RAM, and an Intel(R) Xeon(R) Gold 5118 CPU running at 2.30 GHz. The algorithm by (Qiu et al., 2021), our algorithm without a generative model, and the Greedy Max Variance algorithm typically require approximately 2 minutes of CPU time on average per run. However, our algorithm with a generative model requires around 7 minutes per run due to the cost of increasing the number of exploration episodes by a factor of H.

E.3 Tuning the Confidence Interval Width Multiplier

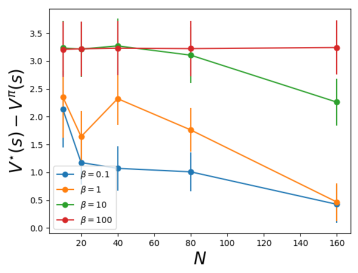
We perform hyperparameter tuning for the confidence interval width multiplier β. The theoretical analysis, especially in Qiu et al. (2021), leads to high values for β. To ensure a fair comparison between algorithms, we fine-tune β for Qiu et al. (2021), our algorithm without a generative model, our algorithm with generative model and Greedy Max Variance, selecting the best value for each. Figures 5, 6, 7 show the simulation results for various values of $\beta \in [0.1, 1, 10, 100]$ for several kernels. The value $\beta = 0.1$ yields the best performance consistently.



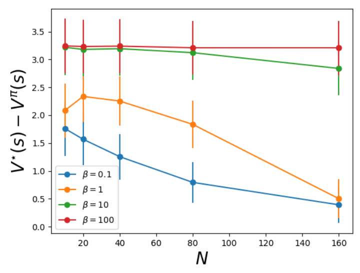
(a) (Qiu et al., 2021)



(b) Without generative model

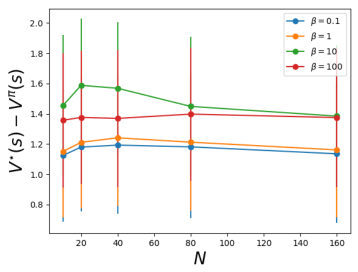


(c) With generative model

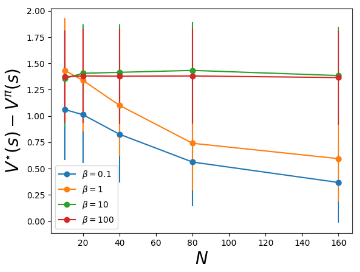


(d) Greedy Max Variance

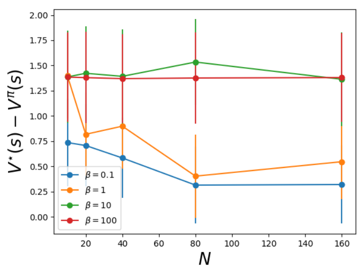
Figure 5: Average suboptimality gap plotted against the number of episodes N for different values of β in the case of SE kernel.



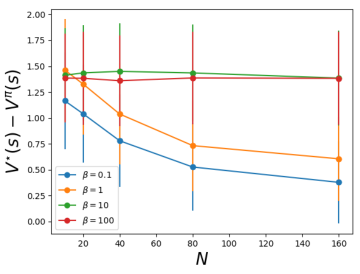
(a) (Qiu et al., 2021)



(b) Without generative model

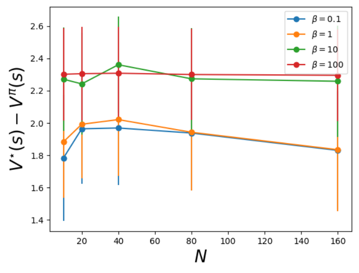


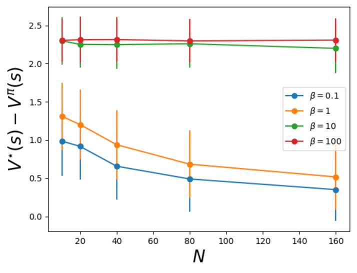
(c) With generative model

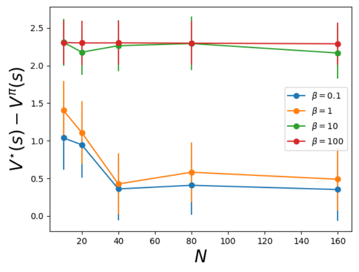


(d) Greedy Max Variance

Figure 7: Average suboptimality gap plotted against the number of episodes N for different values of β in the case of Matérn kernel with $\nu = 1.5$.


(a) (Qiu et al., 2021)


(b) Without generative model


(c) With generative model

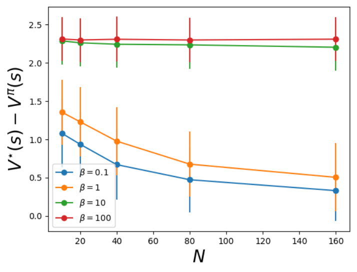
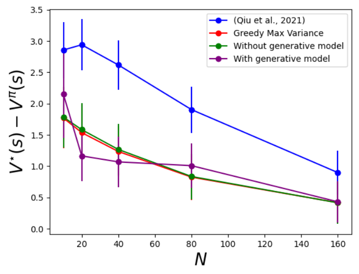
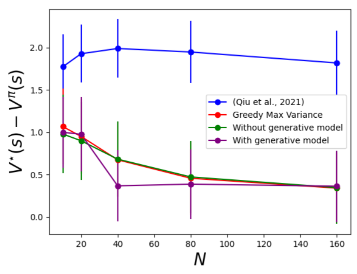

(d) Greedy Max Variance

Figure 6: Average suboptimality gap plotted against the number of episodes N for different values of β in the case of Matérn kernel with $\nu = 2.5$.

E.4 Repeated Experiments for Different Draws of r and P

To validate the robustness of the results against specific environment realizations, we ran the experiments three times, with each repetition using different reward and transition probability functions drawn from the RKHS. We kept the hyperparameters (lengthscale, τ, and β) identical to the ones used in the main paper. The results remained consistent across all repetitions, as shown in Figures 8, 9 and 10 for different kernels and algorithms.


(a) SE Kernel


(b) Matérn kernel with $\nu = 2.5$

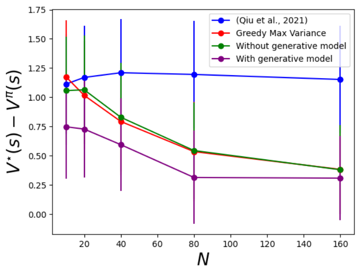
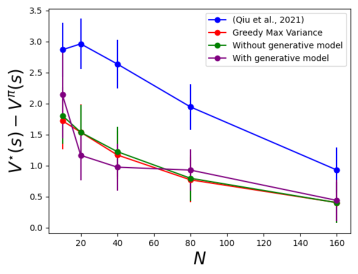
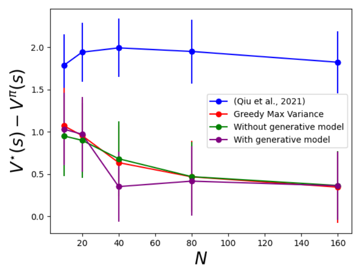

(c) Matérn kernel with $\nu = 1.5$

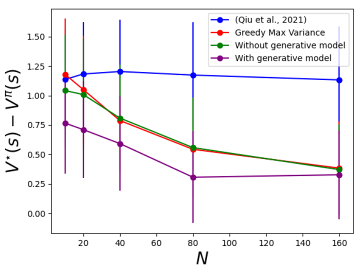
Figure 8: Average suboptimality gap plotted against N for repeated experiment 1



(a) SE Kernel

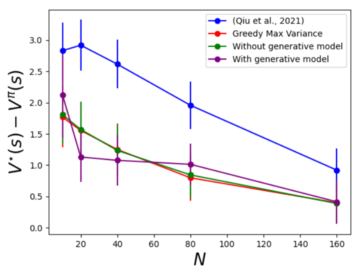


(b) Matérn kernel with $\nu = 2.5$

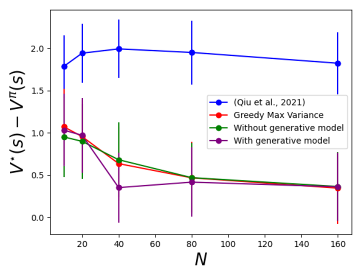


(c) Matérn kernel with $\nu = 1.5$

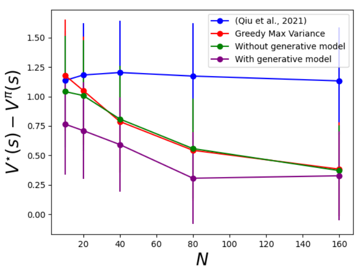
Figure 9: Average suboptimality gap plotted against N for repeated experiment 2



(a) SE Kernel



(b) Matérn kernel with $\nu = 2.5$



(c) Matérn kernel with $\nu = 1.5$

Figure 10: Average suboptimality gap plotted against N for repeated experiment 3

F RKHS and Mercer Theorem

Mercer theorem (Mercer, 1909) provides a representation of the kernel in terms of an infinite dimensional feature map (e.g., see, Christmann and Steinwart, 2008, Theorem 4.49). Let $\mathcal{Z}$ be a compact metric space and μ be a finite Borel measure on $\mathcal{Z}$ (we consider Lebesgue measure in a Euclidean space). Let $L^2_\mu(\mathcal{Z})$ be the set of square-integrable functions on $\mathcal{Z}$ with respect to μ. We further say a kernel is square-integrable if

$$\int_{\mathcal{Z}} \int_{\mathcal{Z}} k^2(z, z')\, d\mu(z) d\mu(z') < \infty.$$

Theorem 4. *(Mercer Theorem) Let $\mathcal{Z}$ be a compact metric space and μ be a finite Borel measure on $\mathcal{Z}$. Let k be a continuous and square-integrable kernel, inducing an integral operator $T_k : L^2_\mu(\mathcal{Z}) \to L^2_\mu(\mathcal{Z})$ defined by*

$$(T_k f)(\cdot) = \int_{\mathcal{Z}} k(\cdot, z') f(z')\, d\mu(z'),$$

where $f \in L^2_\mu(\mathcal{Z})$. Then, there exists a sequence of eigenvalue-eigenfeature pairs $\{(\gamma_m, \varphi_m)\}_{m=1}^\infty$ such that $\gamma_m > 0$, and $T_k \varphi_m = \gamma_m \varphi_m$, for $m \geq 1$. Moreover, the kernel function can be represented as

$$k(z, z') = \sum_{m=1}^{\infty} \gamma_m \varphi_m(z) \varphi_m(z'),$$

where the convergence of the series holds uniformly on $\mathcal{Z} \times \mathcal{Z}$.

According to the Mercer representation theorem (e.g., see, Christmann and Steinwart, 2008, Theorem 4.51), the RKHS induced by k can consequently be represented in terms of $\{(\gamma_m, \varphi_m)\}_{m=1}^\infty$.

Theorem 5. *(Mercer Representation Theorem) Let $\{(\gamma_m, \varphi_m)\}_{i=1}^\infty$ be the Mercer eigenvalue-eigenfeature pairs. Then, the RKHS of k is given by*

$$\mathcal{H}_k = \left\{ f(\cdot) = \sum_{m=1}^{\infty} w_m \gamma_m^{\frac{1}{2}} \varphi_m(\cdot) : w_m \in \mathbb{R}, \|f\|^2_{\mathcal{H}_k} := \sum_{m=1}^{\infty} w_m^2 < \infty \right\}.$$

Mercer representation theorem indicates that the scaled eigenfeatures $\{\sqrt{\gamma_m}\varphi_m\}_{m=1}^{\infty}$ form an orthonormal basis for $\mathcal{H}_k$.