Reinforcement Learning Using Known Invariances

Alexandru Cioba[*1] **Aya Kayal**[*2] **Laura Toni**[3]

Sattar Vakili[1] **Alberto Bernacchia**[1]

[*]Equal contribution

[1]MediaTek Research [2]American University of Beirut [3]University College London

Abstract

In many real-world reinforcement learning (RL) problems, the environment exhibits inherent symmetries that can be exploited to improve learning efficiency. This paper develops a theoretical and algorithmic framework for incorporating known group symmetries into kernel-based RL. We propose a symmetry-aware variant of optimistic least-squares value iteration (LSVI), which leverages invariant kernels to encode invariance in both rewards and transition dynamics. Our analysis establishes new bounds on the maximum information gain and covering numbers for invariant RKHSs, explicitly quantifying the sample efficiency gains from symmetry. Empirical results on a customized Frozen Lake environment and a 2D placement design problem confirm the theoretical improvements, demonstrating that symmetry-aware RL achieves significantly better performance than their standard kernel counterparts. These findings highlight the value of structural priors in designing more sample-efficient reinforcement learning algorithms.

1 INTRODUCTION

Reinforcement Learning (RL) has achieved remarkable empirical success in diverse domains, including robotic manipulation, autonomous driving, game playing, and chip design (Kalashnikov et al., 2018; Silver et al., 2016; Kahn et al., 2017; Mirhoseini et al., 2021; Fawzi et al., 2022). However, despite these advances, the theoretical foundations of RL in complex environments—particularly those with continuous state-action spaces and nonlinear function approximation—remain less well understood. While recent work has established regret guarantees for tabular (Jin et al., 2018; Auer et al., 2008) and linear settings (Jin et al., 2020; Russo, 2019), there is an increasing need to analyze RL algorithms that employ expressive function classes under realistic structural assumptions.

A key structure prevalent in real-world RL problems is symmetry. Many environments exhibit invariance under transformations like rotations, translations, or permutations. For instance: Robotic arms often operate in rotationally symmetric workspaces; Autonomous driving scenarios may be invariant to reflections or road mirroring; Strategic games frequently contain symmetries due to interchangeable board configurations. By explicitly incorporating these invariances into RL algorithms, we can enhance sample efficiency and generalization by eliminating redundant learning.

How can known environment symmetries be systematically exploited to enhance sample efficiency and regret guarantees under nonlinear function approximation? To address this, we focus on kernel methods, which offer a theoretically tractable framework for modeling nonlinearity. Leveraging recent advances in invariant kernels for supervised and bandit learning (Brown et al., 2024; Haasdonk and Burkhardt, 2007; Mei et al., 2021), we introduce a rigorous theoretical and algorithmic framework for symmetry-aware RL via invariant Reproducing Kernel Hilbert Spaces (RKHSs). In this work, we assume that the relevant invariances in the reward and transition dynamics are known a priori, as is common in many structured domains (e.g., robotics, board games, or resource allocation). Our approach is complementary to methods that learn invariances from data, such as meta-learning symmetry structures (Zhou et al., 2020) or learning invariances via marginal-likelihood

Proceedings of the 29[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

optimization in Gaussian processes (van der Wilk et al., 2018), and can naturally incorporate such learned structures when available.

Contributions:

1. We introduce a symmetry-aware variant of the kernel-based optimistic least-squares value iteration (LSVI) algorithm (Yang et al., 2020a), incorporating group invariances in both reward and transition dynamics via a totally invariant kernel.
2. We provide a theoretical analysis of LSVI with totally invariant kernels, and give new bounds on the covering number of the function class of target Q-functions, which we translate into novel, symmetry-aware bounds on sample complexity. To our knowledge, this is the first theoretical analysis of the gains in sample complexity achieved in RL due to incorporating symmetries into the algorithmic design.
3. Finally, we validate our theoretical findings through a series of experiments where natural geometric symmetry occurs in the MDP, demonstrating that incorporating prior knowledge of invariance into the kernel yields substantial gains in sample complexity.

2 RELATED WORK

Regret Bounds/Sample Complexities in RL Numerous studies have explored the sample complexity problem within the episodic MDP framework. Both the tabular setting (Jin et al., 2018; Auer et al., 2008; Bartlett and Tewari, 2009) and the linear setting (Jin et al., 2020; Yao et al., 2014; Russo, 2019; Zanette et al., 2020; Neu and Pike-Burke, 2020) have been extensively investigated. In the tabular case, optimistic state-action value learning algorithms (Jin et al., 2018) achieve a regret bound of $\mathcal{O}\left(\sqrt{H^3|\mathcal{S}\times\mathcal{A}|T}\right)$, where H denotes the episode length, T is the number of episodes, and $\mathcal{S}$ and $\mathcal{A}$ represent the finite state and action spaces, respectively. In the linear setting, regret bounds scale as $\mathcal{O}\left(\sqrt{H^3d^3T}\right)$ (Jin et al., 2020), depending on the dimension d of the linear model rather than the size of the state-action space. Moreover, efforts have been made to extend these techniques to the kernelized setting (Yang et al., 2020a; Yang and Wang, 2020; Chowdhury and Gopalan, 2019; Domingues et al., 2021; Vakili and Olkhovskaya, 2023), though further refinements are required to achieve tighter regret bounds. The most notable advancement in this direction is Yang et al. (2020a), which establishes regret guarantees for an optimistic least-squares value iteration (LSVI) algorithm. The regret bounds reported in Yang et al. (2020a) are given by $\mathcal{O}\left(H^2\sqrt{(\Gamma(T)+\log\mathcal{N}(\epsilon))\Gamma(T)T}\right)$, where $\Gamma(T)$ and $\mathcal{N}(\epsilon)$ are kernel-related complexity measures—specifically, the maximum information gain and the ϵ-covering number of the state-action value function class. In this work, we adopt the same kernel-based framework as Yang et al. (2020a), but for the first time, we incorporate group symmetries by introducing a totally invariant kernel. This allows us to exploit problem structure directly in the learning algorithm. We analyze the corresponding LSVI algorithm and derive new theoretical results by bounding $\Gamma(T)$ and $\mathcal{N}(\epsilon)$ for invariant RKHSs, thereby establishing the first sample complexity guarantees for kernel-based RL methods that explicitly account for symmetry.

Invariant Kernel Methods Invariant kernel methods incorporate prior knowledge of invariances into kernel-based learning algorithms. The theoretical framework for group-invariant kernels was first introduced by Haasdonk and Burkhardt (2007); Kondor (2008), and subsequent work has leveraged these ideas in Gaussian Processes (GPs), kernel ridge regression, and bandit optimization. For example,van der Wilk et al. (2018) integrates invariances directly into the GP model structure rather than relying on data augmentation and develops a variational inference scheme to learn these invariances by maximizing the marginal likelihood. Other works, such as Mei et al. (2021) and Bietti et al. (2024), explore kernelized regression with translation- and permutation-invariant kernels, respectively, analyzing the benefits of invariance in terms of improved sample complexity. A related study by Tahmasebi and Jegelka (2023) provides an exact characterization of sample complexity gains when the target function is invariant under the action of an arbitrary Lie group, linking these gains to the dimensionality of the underlying group structure. In the context of kernelized bandit optimization, Brown et al. (2024) derive upper and lower bounds on sample complexity for Bayesian Optimization (BO) with invariant kernels when optimizing invariant objective functions. In this work, we extend these sample complexity results to the RL setting under symmetry assumptions on both the reward and transition dynamics.

RL with Symmetry Classical work on symmetry in RL is closely connected to model minimization and MDP homomorphisms (Ravindran and Barto, 2001, 2002; Ravindran, 2004), which formalize structure-preserving mappings between MDPs under which solving an abstract MDP yields an optimal policy for the original. More generally, state abstraction—mappings that reduce the state space while preserving selected reward or transition structure (Ravindran and Barto,

2003; Li et al., 2006)—and bisimulation, which defines equivalence classes of states that are behaviorally indistinguishable (Givan et al., 2003; Taylor, 2008), provide principled ways of exploiting structural equivalences in MDPs.

Building on these ideas, symmetries in Markov Decision Processes (MDPs) have also been studied from a modern learning perspective. Several works apply data augmentation techniques in deep RL to improve sample efficiency and generalization on well-known benchmarks (Yarats et al., 2021; Laskin et al., 2020; Lee et al., 2020; Cobbe et al., 2019; Weissenbacher et al., 2022; Sinha et al., 2022). Beyond data augmentation, a number of studies incorporate symmetry into neural network architectures to reduce sample complexity (Van der Pol et al., 2020; Simm et al., 2021; Wang et al., 2022; Mondal et al., 2022; Nguyen et al., 2023; Zhu et al., 2023), including (Van der Pol et al., 2020), which designs equivariant networks that respect symmetry-induced homomorphisms and reduce the solution space.

These ideas have also been extended to the multi-agent RL setting, where symmetries can help reduce policy space complexity, enable the discovery of equivalent strategies, and facilitate coordination between agents. Such approaches often leverage graph-based methods or symmetry-aware architectures (van der Pol et al., 2022; Liu et al., 2020; Jiang et al., 2020; Sukhbaatar et al., 2016; Muglich et al., 2022, 2025; Yu et al., 2024; Tian et al., 2024; McClellan et al., 2024; Yu et al., 2023; Shi et al., 2025; Chen and Zhang, 2024).

In contrast to both classical abstraction methods and modern parametric approaches, our work focuses on single-agent RL and provides a non-parametric, kernel-based framework for incorporating known symmetries. Rather than learning abstractions or designing equivariant neural networks, we encode invariance directly into the function class via group-invariant kernels. Conceptually, the induced equivalence relation resembles the state aggregation underlying abstractions and homomorphisms, but is implemented implicitly through the kernel rather than an explicit mapping. This enables new theoretical insights: we analyze how invariance reduces the effective hypothesis space through tighter bounds on the maximum information gain and covering numbers of the invariant RKHS, thereby quantifying sample-efficiency gains. Finally, while modern deep RL symmetry methods such as data augmentation or equivariant architectures enforce invariances heuristically or parametrically, our non-parametric, kernel-based formulation provides formal guarantees of improved sample complexity.

3 PRELIMINARIES AND PROBLEM FORMULATION

In this section we introduce the problem formulation that we require for our theoretical results. Similar background can be found in Yang et al. (2020a); Vakili and Olkhovskaya (2023).

3.1 Episodic Markov Decision Processes

An episodic MDP is defined by the tuple $M = (\mathcal{S}, \mathcal{A}, H, P, r)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, H is the episode length, $r = \{r_h\}_{h=1}^H$ is the sequence of deterministic reward functions $r_h : \mathcal{Z} \to [0,1]$, and $P = \{P_h\}_{h=1}^H$ is the sequence of transition distributions $P_h(\cdot \mid s,a)$ over next states, given $(s,a) \in \mathcal{Z} = \mathcal{S} \times \mathcal{A}$. A policy $\pi = \{\pi_h\}_{h=1}^H$ maps states to actions at each step h, with $\pi_h : \mathcal{S} \to \mathcal{A}$. At the start of episode t, the environment provides s_1^t, and the agent selects policy π^t. At step h, the agent observes s_h^t, takes $a_h^t = \pi_h^t(s_h^t)$, receives $r_h(s_h^t, a_h^t)$, and transitions to $s_{h+1}^t \sim P_h(\cdot \mid s_h^t, a_h^t)$. The episode ends after H steps.

The goal is to find a policy that maximizes the expected return from any state s and step h, defined by the value function:

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

where the expectation is over trajectories induced by π. Under standard assumptions, an optimal policy $\pi^\star$ exists such that $V_h^{\pi^\star}(s) = \max_\pi V_h^\pi(s)$ for all s and h (Puterman, 2014). We write $V_h^\star(s) := V_h^{\pi^\star}(s)$, and define the expected value under P_h as

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

The state-action value function $Q_h^\pi(s,a)$ is the expected return from (s,a) at step h, defined as $Q_h^\pi(s,a) = \mathbb{E}_\pi\left[\sum_{h'=h}^H r_{h'}(s_{h'}, a_{h'}) \mid s_h = s, a_h = a\right]$. The Bellman equation for π is $Q_h^\pi(s,a) = r_h(s,a) + P_h V_{h+1}^\pi$, with $V_h^\pi(s) = \mathbb{E}_{a \sim \pi_h(s)}[Q_h^\pi(s,a)]$ and $V_{H+1}^\pi := 0$. The Bellman optimality equations are $Q_h^\star(s,a) = r_h(s,a) + P_h V_{h+1}^\star$, and $V_h^\star(s) = \max_a Q_h^\star(s,a)$, with $V_{H+1}^\star := 0$. The regret of a policy sequence $\{\pi^t\}_{t=1}^T$ is defined by:

$$\mathcal{R}(T) = \sum_{t=1}^{T} \left(V_1^\star(s_1^t) - V_1^{\pi^t}(s_1^t)\right), \tag{3}$$

where s_1^t is the initial state at episode t, and measures the cumulative value loss relative to the optimal policy.

3.2 Symmetric RL

In many real-world RL problems, the environment exhibits symmetries: structured transformations of states and actions that leave the reward and dynamics unchanged. In this section, we formalize this notion and provide intuitions for how symmetry can be leveraged to reduce sample complexity. We defer detailed derivations to Appendix A.

Group Actions on MDPs. Let $(G, \cdot)$ be a group acting on a set $\mathcal{X}$. We denote the action map as $\circ : G \times \mathcal{X} \to \mathcal{X}$ and the induced action by $G \curvearrowright \mathcal{X}$. In the context of an MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, H, P, r)$, we say G acts on $\mathcal{M}$ if there are actions $G \curvearrowright \mathcal{S}$ and $G \curvearrowright \mathcal{A}$ such that, for all $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$:

$$\mathbb{P}(s'|g \circ_{\mathcal{S}} s, g \circ_{\mathcal{A}} a) = \mathbb{P}(g^{-1} \circ_{\mathcal{S}} s'|s, a) \tag{4}$$

$$r(g \circ_{\mathcal{S}} s, g \circ_{\mathcal{A}} a) = r(s, a) \tag{5}$$

The $\circ_{\mathcal{S}}$ and $\circ_{\mathcal{A}}$ actions allow us to define a coupling between the state and actions spaces. Functions that preserve this coupling are called *equivariant*. In particular this applies to policies.

Equivariant Policies. A policy $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ is called equivariant with respect to G if, for all g and s, $\pi(g \circ_{\mathcal{S}} s) = g^{-1} \circ_{\mathcal{A}} \pi(s)$. We are particularly interested in cases where such symmetries are known and can be used to constrain the hypothesis space of the learning algorithm. The following proposition which follows from first principles (see Appendix B for derivation), justifies restricting solutions to the space of invariant, respectively equivariant functions for all policy-value iteration (PVI) algorithms.

Proposition 1. *In finite-horizon MDPs, equivariant policies have invariant value functions $V_\pi(s)$ and $Q_\pi(s, a)$. Moreover, if $Q(s, a)$ is invariant under G and satisfies the Bellman equation, then the greedy policy $\pi(s) = \arg\max_a Q(s, a)$ is equivariant.*

This result implies that if we start PVI in the space of invariant functions, and we incur no approximation error in either policy evaluation or iteration, we are guaranteed to recover an optimum, and such optimum is equivariant. However, in practical settings these guarantees don't apply. The following two results, which we present intuitively here, are meant to bridge the gap to the realistic setting where function approximation is of importance.

Effective Reduction in the State Space The first simply states that in some cases symmetries can be factored cleanly out of the MDP, resulting in a smaller, simpler space where all algorithms benefit from reduced sample complexity guarantees. If the action on $\mathcal{A}$ is trivial ($g \circ a = a$), we can define a reduced MDP over state orbits $\mathcal{S}/G$. This induces a simpler equivalent problem whose optimal policy can be lifted back to the original space. See Proposition 2 in Appendix C.

Simple Sample Complexity Bounds for Q-learning The second is a variation on a celebrated result from Jin et al. (2018), where we explore how sample complexity varies in Q-learning in the tabular setting under data augmentation. In tabular Q-learning, regret bounds typically scale with $|\mathcal{S} \times \mathcal{A}| = SA$. In the presence of symmetry, the relevant complexity becomes $\widetilde{SA} = |(\mathcal{S} \times \mathcal{A})/G|$, the number of orbits. This motivates algorithmic design that explicitly incorporates symmetry, as we do in the rest of the paper. See Appendix D where we state and prove Theorem 3.

3.3 Kernel-Based RL with Symmetries

Kernel-based models are powerful tools for nonparametric function approximation in RL, offering flexible predictors and principled uncertainty estimates. Importantly, as seen in Brown et al. (2024), the pairing of the class of inner product kernels with orthogonal symmetries creates a powerful framework for incorporating symmetries naturally into predictors. See Appendix E for standard definitions in kernel methods.

Invariant Kernels. Let $k : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ be a positive definite inner product kernel. Assume $\mathcal{Z} \hookrightarrow \mathbb{R}^d$ is an embedding and the orthogonal group $O(d)$ restricts its action on $\mathcal{Z}$. Let G be a finite subgroup of $O(d)$. Then the following formula defines an invariant kernel:

$$k_G(z, z') = \frac{1}{|G|} \sum_{g \in G} k(g(z), z'), \tag{6}$$

The RKHS $\mathcal{H}_{k_G}$ induced by k_G consists of G-invariant functions. Invariant kernels constrain learning to a smaller, structured function class, offering both theoretical and practical benefits in symmetric environments. In RL, they yield value and transition models that respect MDP symmetries, enhancing generalization and sample efficiency.

3.4 Main Technical Assumptions

In our RL setting, we use a kernel-based model to predict the expected value function. For a given transition distribution $P(s'|\cdot, \cdot)$ and value function $v : \mathcal{S} \to \mathbb{R}$, we define $f = [Pv]$ and use past observations to construct predictions and uncertainty estimates for f via kernel ridge regression. Since the value functions evolve across steps due to the Markovian structure, we aim to control the estimation error at each step through confidence bounds. We will need the following standard assumption.

Assumption 1. *We assume* $r_h(\cdot,\cdot), P_h(s'|\cdot,\cdot) \in \mathcal{H}_{k_G}$ *for some invariant kernel* k_G*, and that* $\|r_h\|_{\mathcal{H}_{k_G}}, \|P_h(s'|\cdot,\cdot)\|_{\mathcal{H}_{k_G}} \leq 1$ *for all* $s' \in \mathcal{S}$ *and* $h \in [H]$.

Our analysis depends on the smoothness of the kernel k, which influences the function approximation properties of the associated RKHS. Specifically, we characterize the complexity of the kernel class through the decay of its Mercer eigenvalues.

Assumption 2 (Eigendecay Profile)**.** *Let* $\{\lambda_m\}_{m=1}^{\infty}$ *denote the Mercer eigenvalues of the invariant kernel* k_G*, ordered in decreasing magnitude. We assume the eigenvalues satisfy the decay condition* $\lambda_m = \mathcal{O}((m|G|)^{-p})$ *for some* $p > 0$.

This assumption is mild and satisfied by many commonly used kernels. For instance, in the case of the Matérn kernel with smoothness parameter ν in d dimensions, we have $p = 1 + \frac{2\nu}{d}$.

3.5 LSVI Algorithm

We consider the Kernel-based Optimistic Value Iteration (KOVI) algorithm (Yang et al. (2020a), see also Algorithm 1), which applies optimistic LSVI in the kernel regression setting. Least-Squares Value Iteration (LSVI) estimates $\widehat{Q}_h^t$ for the optimal $Q_h^\star$ at each step h of episode t via recursive Bellman updates. To encourage exploration, an upper confidence bonus $b_h^t : \mathcal{Z} \to \mathbb{R}$ is added:

$$Q_h^t = \min\{\widehat{Q}_h^t + \beta b_h^t,\ H - h + 1\}^+. \tag{7}$$

Here, $\beta > 0$ is a tunable parameter, and the term $\widehat{Q}_h^t + \beta b_h^t$ provides an optimistic estimate based on the principle of *optimism in the face of uncertainty*. Specifically, the function $\widehat{Q}_h^t$ is the kernel-based predictor for $r_h + [P_h V_{h+1}^t]$ using the dataset $\{r_h(z_h^\tau) + V_{h+1}^t(s_{h+1}^\tau)\}_{\tau=1}^{t-1}$ observed at inputs $\{z_h^\tau\}_{\tau=1}^{t-1}$. Since rewards are bounded by 1, the cumulative return from step h is at most $H - h + 1$. At episode t, the agent follows a greedy policy π^t based on the optimistic estimates $Q^t = \{Q_h^t\}_{h=1}^H$. Under Assumption 1, the estimates $\widehat{Q}_h^t$ and bonus b_h^t are computed via kernel ridge regression.

Algorithm 1 The KOVI Algorithm

Input: Regularization λ, confidence parameter $\beta_T(\delta)$, kernel k, MDP $M = (\mathcal{S}, \mathcal{A}, H, P, r)$
for episode $t = 1, 2, \ldots, T$ **do**
  Observe initial state s_1^t
  Set $V_{H+1}^t(s) = 0$ for all $s \in \mathcal{S}$
  for step $h = H, H-1, \ldots, 1$ **do**
    Compute $Q_h^t(z)$ and $V_h^t(z)$ via kernel ridge regression
  end for
  for step $h = 1, 2, \ldots, H$ **do**
    Take action $a_h^t \leftarrow \arg\max_{a \in \mathcal{A}} Q_h^t(s_h^t, a)$
    Observe reward $r_h(s_h^t, a_h^t)$ and next state s_{h+1}^t
  end for
end for

4 THEORETICAL ANALYSIS

4.1 Information Gain

We define a kernel-specific complexity term, referred to as the maximum information gain $\Gamma_k(T)$ from T observations, as follows (Srinivas et al., 2010):

$$\Gamma_k(T) = \max_{\{z_t\}_{t=1}^T \subset \mathcal{Z}} \log\det\left(I + \frac{1}{\lambda}[k(z_i, z_j)]_{i,j=1}^T\right). \tag{8}$$

This expression takes the maximum of the log determinant of a scaled and regularized kernel matrix, with the maximum over all input sequences. It quantifies the complexity of the kernel model and appears in related problems such as BO. In that context, Brown et al. (2024) provides a bound on the information gain for invariant kernels, formally stated below.

Lemma 1 (Brown et. al.)**.** *Consider the maximum information gain defined in equation 8. Under Assumption 2, we have*

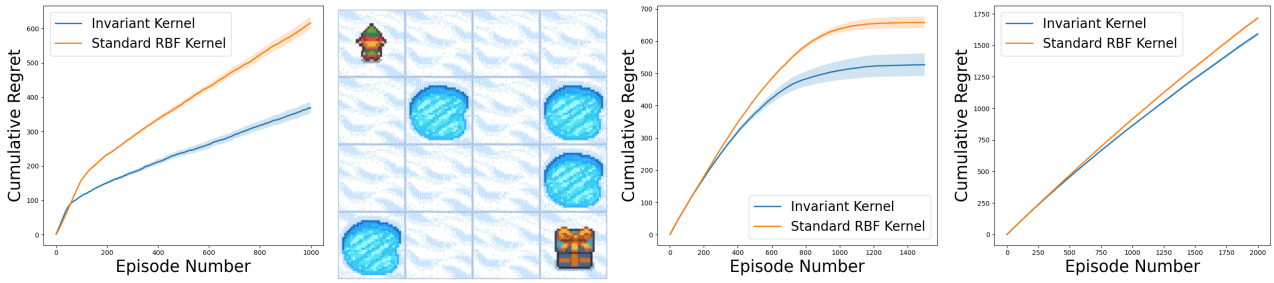
$$\Gamma_{k_G}(T) = \mathcal{O}\left(\frac{T^{\frac{1}{p}}}{|G|}\right). \tag{9}$$

4.2 Covering Number of Function Class

Following Yang et al. (2020a), recall the definition of the set of Q-functions in KOVI,

$$\mathcal{Q}(B, R) = \big\{Q_h^t \ \big|\ \forall\{z_j\}_{j=1}^t \subset \mathcal{Z},\ t \leq T, \\ ||Q_0||_{\mathcal{H}} \leq R,\ \beta \in [0, B]\big\}. \tag{10}$$

Covering Number: Consider a set of real functions $\mathcal{F}$. For $\epsilon > 0$, we define the minimum ϵ-covering set $\mathcal{C}(\epsilon)$ as the smallest subset of $\mathcal{F}$ that covers it up to an ϵ error in l_∞ norm. That is to say, for all $f \in \mathcal{F}$,



(a) Synthetic setting (b) Frozen Lake (c) Frozen Lake (Fixed) (d) Frozen Lake (Random)

Figure 1: Comparison of KOVI with invariant kernel vs. standard RBF kernel, across different settings. Cumulative regret is plotted against the number of episodes. (a) Regret for synthetic setting. (b) A rendered frame of Frozen Lake. (c,d) Regret for the Frozen Lake, fixed and random setting, respectively. Regret is averaged over 20 random seeds. Shaded area represents the standard error.

Table 1: Environments, state and action spaces, and group action used in our experiments.

Environment	Space	Group Action
Synthetic	S = [-1,1]	$\{x \to x, x \to -x\}$
	A = [-1,1]	$\{x \to x, x \to -x\}$
Frozen Lake	$S = \{z = (x_i, y_i)_{i=1}^6 \in \mathbb{C}^6\}$	$D_4 = \langle z \to e^{\frac{i\pi}{2}} z, z \to \bar{z} \rangle \curvearrowright \mathbb{C}^6$
	A = { (-1, 0), (1, 0), (0, -1), (0, 1) }	$D_4 \curvearrowright \mathbb{C}\vert_A$
Synpl	$S = \{z = (x_i, y_i)_{i=1}^{16} \in \mathbb{C}^{16}\}$	$D_4 = \langle z \to e^{\frac{i\pi}{2}} z, z \to \bar{z} \rangle \curvearrowright \mathbb{C}^{16}$
	$A = \{z = (x, y) \in \mathbb{C}\}$	$D_4 \curvearrowright \mathbb{C}\vert_A$

there exists a $g \in \mathcal{C}(\epsilon)$, such that $\|f - g\|_{l_\infty} \leq \epsilon$. We refer to the size of $\mathcal{C}(\epsilon)$ as the ϵ-covering number and use the notation $\mathcal{N}(\epsilon, B, R, G)$ to denote the ϵ-covering number of $\mathcal{Q}(B, R)$.

Theorem 1. *With the notation above, we have:*

$$\log \mathcal{N}(\epsilon, B, R, G) = \mathcal{O}\left(\left(\frac{R^2}{\epsilon^2 |G|^p}\right)^{\frac{1}{p-1}} \left(1 + \log \frac{R}{\epsilon}\right) + \left(\frac{B^2}{\epsilon^2 |G|^p}\right)^{\frac{2}{p-1}} \left(1 + \log \frac{B}{\epsilon}\right)\right). \tag{11}$$

This result provides the explicit dependence of the covering number on the number of symmetries, i.e. all other constants are independent of ϵ, B, R and $|G|$. See Appendix F for proof.

4.3 Regret Bounds and Discussions

Recall Theorem 4.2 and the associated notation in Yang et al. (2020a), which provides the regret bound in terms of the maximal information gain and covering number.

Theorem 2 (Yang et. al.)**.** *Consider the episodic MDP described in Section 3.1. Under Assumption 1, for the KOVI algorithm, there exist B and R such that for all $T \in \mathbb{N}$ and $\epsilon \in (0, 1)$, denoting $\Lambda(k_G; B, R, \epsilon, T) = (\Gamma_{k_G}(T))^{\frac{1}{2}}((\Gamma_{k_G}(T) + \log \mathcal{N}(\epsilon, B, R, G))T)^{\frac{1}{2}} + T\epsilon$:*

$$R(T) = \mathcal{O}\left(\Lambda(k_G; B, R, \epsilon, T)\right)$$

Under Assumption 2, we may follow the choices in Yang et al. (2020b) for $B := B_T$ and $\epsilon = \epsilon_T$, to extract the dependency of $\Lambda(k_G; B, R, \epsilon, T)$ on $|G|$ explicitly. Namely, Lemma 1 Theorem 1 give:

Corollary 1. *Under the assumptions above, $||Q_h^T||_{\mathcal{H}} \leq R_T(k_G) := 2H\sqrt{\Gamma_{k_G}}$, for $B_T := H \log(TH) \cdot T^{k^*}$ and $\epsilon = H/T$ we have:*

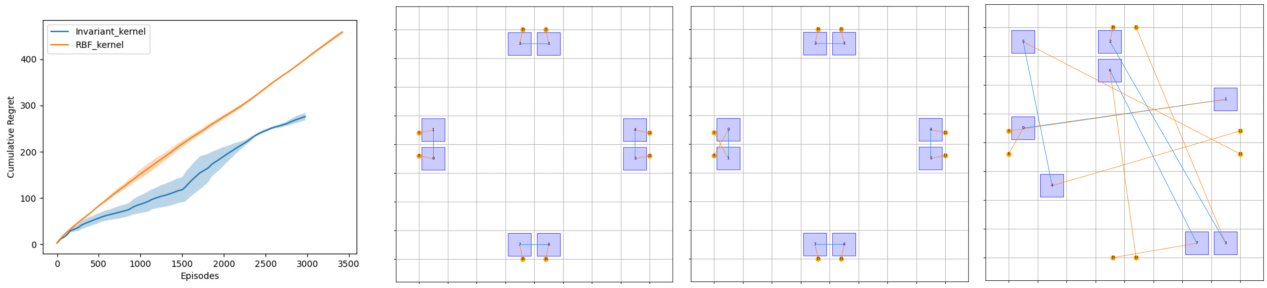
$$\Lambda(k_G; B_T, R_T(k_G), \epsilon, T) \leq \frac{1}{|G|}\Lambda(k; B_T, R_T(k), \epsilon, T) \tag{12}$$

$$R(T) = \mathcal{O}\left(\frac{1}{|G|}\Lambda(k; B_T, R_T, \epsilon, T)\right) \tag{13}$$

Remarks. Substituting $p = 1 + \frac{2\nu}{d}$ in Corollary 1, we have, for large d, and any $\nu > 0$:

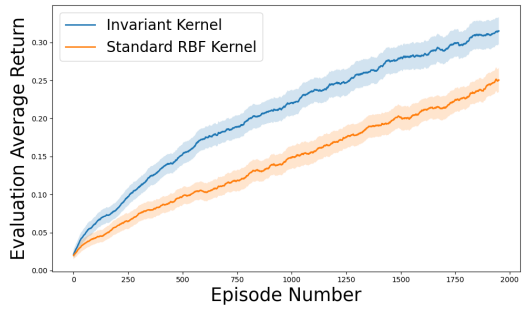
$$R(T) = \tilde{\mathcal{O}}\left(\frac{1}{|G|} H^2 \cdot T^{\frac{3d(d+1)+2\nu}{4\nu+2d(d+1)}}\right) \tag{14}$$

Contrast this result, with e.g. Theorem 3 or with regret in BO, for which $R_T \sim \sqrt{\Gamma_{k_G}(T)}$ (Wang et al. (2024)) which both pose a factor of $\frac{1}{|G|}$ in the bound. Why does KOVI achieve better improvement with $|G|$ over both these bounds? Compared with BO, the suboptimality comes for the acquisition function. Compared to KOVI in the tabular setting, however, the bound is consistent, since both $\mathcal{N}(\epsilon, B, R)$ and $\Gamma(T) \sim SA = |\mathcal{S} \times \mathcal{A}|$. Choosing $\beta = HSA$ in Theorem 2 gives: $R(T) = \tilde{\mathcal{O}}(H^2 SA\sqrt{T})$ which is consistent with the $\frac{1}{|G|}$ reduction seen above.

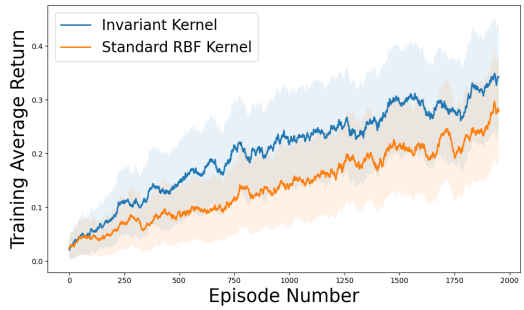


(a) Regret comparison (b) Optimal placement (c) Best placement (d) Random placement

Figure 2: Regret comparison of invariant vs. RBF kernel for SynPl (a); Optimal placement (b); Best placement achieved by KOVI (c); Baseline random placement from the random policy (d).



(a) Test performance



(b) Training Performance

Figure 3: Average return computed on evaluation (test) data (a) and training data (b) vs. number of training episodes for the Random Layout Frozen Lake environment. Shaded areas represent standard error.

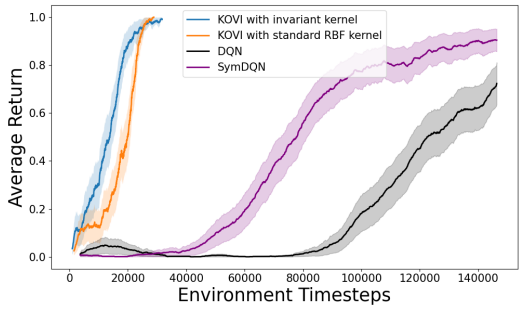


Figure 4: Comparison of KOVI with a standard RBF kernel, KOVI with an invariant kernel, DQN, and SymDQN on the FrozenLake (Fixed) environment. Average episodic return versus number of environment timesteps, averaged over 20 random seeds. The shaded area represents the standard error.

5 EXPERIMENTAL RESULTS

We demonstrate the performance improvement gained by incorporating known invariances in three settings: (1) a synthetic MDP setting with group symmetry, (2) a symmetrized Frozen Lake environment from the OpenAI Gym library and (3) a 2D placement problem. The code is available at: `https://github.com/mtkresearch/IQL`.

Synthetic Setting: We choose $H = 10$ and $\mathcal{S} = \mathcal{A} = [-1, 1]$ discretized into 10 evenly spaced points. The group we consider has size $|G| = 2$ and includes the identity and inversion transformations. The reward function r and transition function P are constructed to be invariant under G, by sampling functions from the RKHS of the invariant kernel k_G (eq. 6). We choose RBF as the base kernel k, and we train for a total of 1000 episodes. For a more detailed explanation of how r and P are generated and the hyperparameters used, please refer to Appendix G.1.

Frozen Lake: We build a navigation task based on OpenAI Gym's FrozenLake-v1 environment which is invariant to the 8-fold symmetry group of the square, D_4. The environment itself depends on a layout of holes, start and goal positions, see Figure 1b. Each state is represented as a concatenation of 2-dimensional coordinate vectors (x, y) specifying the positions of the agent, goal, and four holes, resulting in a 6×2 state vector. Actions are represented as 2D discrete coordinate vectors.

At each episode, the agent is tasked to solve the problem in any one of the 8 possible transformations of the

layout according to D_4, sampled uniformly at random. In Figure 1, this is denoted by *Fixed Layout*, denoting the fact that the only change in environment that the agent observes is due to the presence of symmetry in D_4. In order to exploit a more realistic scenario, where symmetry naturally occurs in the environment, we also train KOVI on a variant of Frozen Lake with fully randomized Layouts. In Figure 1 we refer to these experiments as *Random Layout*. In this setting, a new layout of the starting point and holes is randomly generated at the beginning of each episode. Four holes are placed randomly, and only configurations that allow a valid path from the start to the goal state are accepted. This setting is significantly more challenging due to the greater diversity of possible configurations. We train for a total of 2000 episodes. For details about the hyperparameters used, please refer to Appendix G.2.

In Figure 1 we plot the cumulative regret during training on the synthetic setting, the Fixed and Random Layout Frozen Lake. For the latter only, we also evaluate the episodic return on 40 randomly sampled test environments and report the average performance on these test environments in each training episode. Figure 3a shows the average return on these test environments over the course of training, averaged across 20 random seeds. While the return has not fully converged— due to the complexity introduced by the large number of random grid configurations—the invariant kernel consistently outperforms the standard RBF kernel. For reference, we also plot the average return on the training environment in Figure 3b.

Comparison vs Neural Networks In Figure 4 we present a comparison of relative sample complexity of our algorithm versus an Q-Network on FrozenLake implemented with equivariant MLP layers. The kernel regression enjoys consistent improvements in sample complexity versus the more data inefficient Q-network. See additional experimental details in Appendix G.3.

2-Dimensional Placement: The placement problem can be described as a 2D planar embedding of a graph of components under geometric constraints. This problem appears frequently in contexts such as smart-city design, building floorplanning, electronic design and VLSI. We build a simplified environment where we can study this task, denoted by *SynPl* in Figure 2. The task is to correctly place 8 units on an 8x8 grid with the following constraints: 1. No two units should overlap. 2. Each edge of the graph should be enclosed in a minimal area subset of the grid. While the placement task may be easy to solve for humans by inspection, leveraging our vast compendium of geometric reasoning and intuition, even simple cases remain difficult for machines without carefully tailored backend representations (see, e.g. GNNs used to encode the graphical information Mirhoseini et al. (2021), Chang et al. (2022)). 1-shot placement methods like BO perform poorly in this setting due to:

1. The subset of invalid configurations lies at the boundary of the space of optimal configurations (where the edge length is minimized). If overlaps are handled via penalties in the optimization objective, this requires modeling with progressively less smooth kernels, which rapidly decreases sample complexity guarantees. Forcing the acquisition function to only sample valid configurations (according to the posterior) is not without its own issues, with rejection sampling being the most straightforward, yet cumbersome approach.

2. The configuration space naturally lies on a $2n$ dimensional manifold in $\mathbb{R}^{2n}$. While lower-dimensional embeddings of this data exist, they don't benefit from the natural action of the isometry group on $\mathbb{R}^{2n}$, missing the crucial geometric information in the configuration space. Moreover, such embeddings represent a strenuous design decision on the experimenter. On the other hand, as n increases, this leads to high-dimensionality BO, which carries its own sample complexity issues (Eriksson and Jankowiak (2021)).

A better design decision is to leverage the sequential nature of the placement problem to produce valid configurations with progressively higher objective values $\Phi(s)$. The most general form of the environment would sample all canvas configurations uniformly however we consider a simplified setting and fix a single problem with a unique symmetric solution. Specifically, the environment places the units sequentially, with each action being the selection of one of the 64 grid cells. An action mask is applied to avoid overlaps. The state, at any one point consists of the position of the pre-placed units in Cartesian coordinated, with padding for the yet-to-be-placed units. The reward at each step t is a function $r(s_t, a_t)$, defined as follows. Let π^* be an oracle policy for all states, which is able to optimally place all remaining units in all states. Then $r(s_{t-1}, a_{t-1}) = \Phi(\pi^*(s_t)^{[H-t]}) - \Phi(\pi^*(s_{t-1})^{[H-t+1]})$, the potential difference due to selecting a_{t-1}. As this quantity is inaccessible, we replace the oracle policy π^* by a known reference policy π_0. As this need not be a deterministic policy, we sample several trajectories from π_0 and average the Φ estimates correspondingly. As the horizon for our problem is low ($H = 8$), in our experiments we take π_0 to be the random policy, and do not observe significant degradations due to this choice. In our experiments we observe that the invariant kernel consistently produces lower regret, even in

scenarios where symmetries are not inherently built into the environment. Empirical results differ from their theoretical counterparts in that the asymptotic expected gap isn't always recovered for small T. Our experiments thus strengthen the theoretical results by providing insight into the behavior of KOVI with and without invariant kernel in finite time. See Figure 2 for regret and placements.

5.1 Limitations

The KOVI algorithm suffers from general limitations which depend either on its theoretical formulation or on implementation. The two principal points are the choice of hyperparameters, primarily β and the overall complexity of the algorithm. Firstly, notice that since at every step H GP models are fitted to an experience buffer which contains all samples acquired to that point, the complexity of each step is $\mathcal{O}(Ht^3)$ due to inverting the kernel matrix, and summing over $t \in [0, T]$ produces a time complexity of $\mathcal{O}(HT^4)$. This is prohibitive for realistic environments for which more than a couple thousand samples need to be collected. This however can be remedied as follows: 1. The time complexity can be brought down to $\mathcal{O}(Ht^3)$, matching that of BO with exact kernel inversion, by caching previous calculations in the Cholesky decomposition of each GP model and performing only rank 1 updates. 2. Secondly, sparse GP methods can be applied (with or without optimizing the inducing point acquisition) to compress the collected samples into a manageable size for kernel inversion. In terms of algorithmic stability, the parameter β is empirically identified as the main contributor. In practice it is kept fixed throughout our runs and optimized as a hyperparameter, which is in contrast to the choice of $\beta = B_T$ in the theoretical formulation (which, in the tabular setting scales as HSA). This likely produces suboptimal results, however, more research needs to be done for adaptively setting β in terms of T (or $|G|$) in this setting. In terms of convergence properties of the algorithm, the $\beta \cdot \sigma_T(z)$ exploration bonus, induces a mode-hopping behavior, by over-estimating Q-values early in the runtime. Sometimes optimal solutions are found and all neighboring solutions are explored early on in the training process, however, the algorithm subsequently selects different, less explored configurations in an effort to reduce uncertainty there. This is not problematic in synthetic settings where the optimal $V^*(s)$ is known (s still hidden), however, in realistic settings, this behavior can contribute to a large proportion of the runtime.

6 CONCLUSION

Group symmetry in MDPs provides powerful structural constraints that, when incorporated effectively into the RL pipeline—via equivariant policies and invariant function spaces—can significantly reduce sample complexity. This work is the first to quantify the sample efficiency gains of incorporating group invariances into a fully invariant kernel within the kernel-based RL setting. We derive novel bounds on the maximum information gain and covering number for the invariant kernel, and demonstrate that our improved regret bounds are supported empirically through experiments on synthetic domains, classical RL benchmarks, and a practical 2D placement task, highlighting real-world applicability. For future work, it would be valuable to evaluate our symmetry-aware LSVI algorithm on more complex or high-dimensional input domains, as well as to extend our theoretical results to settings with partial or approximate symmetries.

References

Auer, P., Jaksch, T., and Ortner, R. (2008). Near-optimal regret bounds for reinforcement learning. *Advances in neural information processing systems*, 21.

Balandat, M., Karrer, B., Jiang, D., Daulton, S., Letham, B., Wilson, A. G., and Bakshy, E. (2020). Botorch: A framework for efficient monte-carlo bayesian optimization. *Advances in neural information processing systems*, 33:21524–21538.

Bartlett, P. L. and Tewari, A. (2009). Regal: a regularization based algorithm for reinforcement learning in weakly communicating mdps. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, page 35–42. AUAI Press.

Bietti, A., Venturi, L., and Bruna, J. (2024). On the sample complexity of learning under invariance and geometric stability. In *Proceedings of the 35th International Conference on Neural Information Processing Systems, NIPS'21*. Curran Associates Inc. Red Hook, NY, USA.

Brown, T., Cioba, A., and Bogunovic, I. (2024). Sample-efficient bayesian optimisation using known invariances. *Advances in Neural Information Processing Systems*, 37:47931–47965.

Chang, F.-C., Tseng, Y.-W., Yu, Y.-W., Lee, S.-R., Cioba, A., Tseng, I.-L., Shiu, D.-s., Hsu, J.-W., Wang, C.-Y., Yang, C.-Y., et al. (2022). Flexible chip placement via reinforcement learning: late breaking results. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*, pages 1392–1393.

Chen, D. and Zhang, Q. (2024). E(3)-equivariant actor-critic methods for cooperative multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 7640–7666. 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.

Cobbe, K., Klimov, O., Hesse, C., Kim, T., and Schulman, J. (2019). Quantifying generalization in reinforcement learning. In *International conference on machine learning*, pages 1282–1289. PMLR.

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.

Eriksson, D. and Jankowiak, M. (2021). High-dimensional bayesian optimization with sparse axis-aligned subspaces. In *Uncertainty in Artificial Intelligence*, pages 493–503. PMLR.

Fawzi, A., Balog, M., Huang, A., Hubert, T., Romera-Paredes, B., Barekatain, M., Novikov, A., R. Ruiz, F. J., Schrittwieser, J., Swirszcz, G., et al. (2022). Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610(7930):47–53.

Givan, R., Dean, T., and Greig, M. (2003). Equivalence notions and model minimization in markov decision processes. *Artificial intelligence*, 147(1-2):163–223.

Haasdonk, B. and Burkhardt, H. (2007). Invariant kernel functions for pattern analysis and machine learning. *Machine learning*, 68:35–61.

Jiang, J., Dun, C., Huang, T., and Lu, Z. (2020). Graph convolutional reinforcement learning. In *International Conference on Learning Representations*.

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

Kahn, G., Villaflor, A., Pong, V., Abbeel, P., and Levine, S. (2017). Uncertainty-aware reinforcement learning for collision avoidance. *arXiv preprint arXiv:1702.01182*.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. (2018). Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on robot learning*, pages 651–673. PMLR.

Kondor, I. R. (2008). *Group theoretical methods in machine learning*. Columbia University.

Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., and Srinivas, A. (2020). Reinforcement learning with augmented data. *Advances in neural information processing systems*, 33:19884–19895.

Lee, K., Lee, K., Shin, J., and Lee, H. (2020). Network randomization: A simple technique for generalization in deep reinforcement learning. In *International Conference on Learning Representations*.

Li, L., Walsh, T. J., and Littman, M. L. (2006). Towards a unified theory of state abstraction for mdps. *AI&M*, 1(2):3.

Liu, I.-J., Yeh, R. A., and Schwing, A. G. (2020). Pic: permutation invariant critic for multi-agent

deep reinforcement learning. In *Conference on Robot Learning*, pages 590–602. PMLR.

McClellan, J., Haghani, N., Winder, J., Huang, F., and Tokekar, P. (2024). Boosting sample efficiency and generalization in multi-agent reinforcement learning via equivariance. *arXiv preprint arXiv:2410.02581*.

Mei, S., Misiakiewicz, T., and Montanari, A. (2021). Learning with invariances in random features and kernel models. In *Conference on Learning Theory*, pages 3351–3418. PMLR.

Mirhoseini, A., Goldie, A., Yazgan, M., Jiang, J. W., Songhori, E., Wang, S., Lee, Y.-J., Johnson, E., Pathak, O., Nova, A., et al. (2021). A graph placement methodology for fast chip design. *Nature*, 594(7862):207–212.

Mondal, A. K., Jain, V., Siddiqi, K., and Ravanbakhsh, S. (2022). Eqr: Equivariant representations for data-efficient reinforcement learning. In *International Conference on Machine Learning*, pages 15908–15926. PMLR.

Muglich, D., Forkel, J., van der Pol, E., and Foerster, J. N. (2025). Expected return symmetries. In *The Thirteenth International Conference on Learning Representations*.

Muglich, D., Schroeder de Witt, C., van der Pol, E., Whiteson, S., and Foerster, J. (2022). Equivariant networks for zero-shot coordination. *Advances in Neural Information Processing Systems*, 35:6410–6423.

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.

Nguyen, H. H., Baisero, A., Klee, D., Wang, D., Platt, R., and Amato, C. (2023). Equivariant reinforcement learning under partial observability. In *Conference on Robot Learning*, pages 3309–3320. PMLR.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

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

Ravindran, B. (2004). *An algebraic approach to abstraction in reinforcement learning*. University of Massachusetts Amherst.

Ravindran, B. and Barto, A. G. (2001). Symmetries and model minimization in markov decision processes.

Ravindran, B. and Barto, A. G. (2002). Model minimization in hierarchical reinforcement learning. In *International symposium on abstraction, reformulation, and approximation*, pages 196–211. Springer.

Ravindran, B. and Barto, A. G. (2003). Smdp homomorphisms: an algebraic approach to abstraction in semi-markov decision processes. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, page 1011–1016. Morgan Kaufmann Publishers Inc.

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

Shi, R., Yu, X., Wang, Y., Tian, Y., Liu, Z., Wu, W., Zhang, X.-P., and Veloso, M. M. (2025). Symmetry-informed marl: A decentralized and cooperative uav swarm control approach for communication coverage. *IEEE Transactions on Mobile Computing*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.

Simm, G. N. C., Pinsler, R., Csányi, G., and Hernández-Lobato, J. M. (2021). Symmetry-aware actor-critic for 3d molecular design. In *International Conference on Learning Representations*.

Sinha, S., Mandlekar, A., and Garg, A. (2022). S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics. In *Conference on Robot Learning*, pages 907–917. PMLR.

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*, pages 1015–1022.

Sukhbaatar, S., Fergus, R., et al. (2016). Learning multiagent communication with backpropagation. *Advances in neural information processing systems*, 29.

Tahmasebi, B. and Jegelka, S. (2023). The exact sample complexity gain from invariances for kernel regression. *Advances in Neural Information Processing Systems*, 36.

Taylor, J. (2008). Lax probabilistic bisimulation. Master's thesis, McGill University.

Tian, Y., Yu, X., Qi, Y., Wang, L., Feng, P., Wu, W., Shi, R., and Luo, J. (2024). Exploiting hierarchical symmetry in multi-agent reinforcement learning. In *ECAI 2024*, pages 2202–2209. IOS Press.

Vakili, S. and Olkhovskaya, J. (2023). Kernelized reinforcement learning with order optimal regret bounds. *Advances in Neural Information Processing Systems*, 36:4225–4247.

van der Pol, E., van Hoof, H., Oliehoek, F. A., and Welling, M. (2022). Multi-agent MDP homomorphic networks. In *International Conference on Learning Representations*.

Van der Pol, E., Worrall, D., van Hoof, H., Oliehoek, F., and Welling, M. (2020). Mdp homomorphic networks: Group symmetries in reinforcement learning. *Advances in Neural Information Processing Systems*, 33:4199–4210.

van der Wilk, M., Bauer, M., John, S., and Hensman, J. (2018). Learning invariances using the marginal likelihood. *Advances in Neural Information Processing Systems*, 31.

Wang, D., Walters, R., and Platt, R. (2022). So(2)-equivariant reinforcement learning. In *International Conference on Learning Representations*.

Wang, J., Wang, H., Petra, C. G., and Chiang, N.-Y. (2024). On improved regret bounds in bayesian optimization with gaussian noise.

Weissenbacher, M., Sinha, S., Garg, A., and Yoshinobu, K. (2022). Koopman q-learning: Offline reinforcement learning via symmetries of dynamics. In *International conference on machine learning*, pages 23645–23667. 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. (2020a). Provably efficient reinforcement learning with kernel and neural function approximations. *Advances in Neural Information Processing Systems*, 33:13903–13916.

Yang, Z., Jin, C., Wang, Z., Wang, M., and Jordan, M. I. (2020b). On function approximation in reinforcement learning: Optimism in the face of large state spaces. *arXiv preprint arXiv:2011.04622*.

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.

Yarats, D., Kostrikov, I., and Fergus, R. (2021). Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International conference on learning representations*.

Yu, X., Shi, R., Feng, P., Tian, Y., Li, S., Liao, S., and Wu, W. (2024). Leveraging partial symmetry for multi-agent reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17583–17590.

Yu, X., Shi, R., Feng, P., Tian, Y., Luo, J., and Wu, W. (2023). Esp: Exploiting symmetry prior for multi-agent reinforcement learning. *arXiv preprint arXiv:2307.16186*.

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.

Zhou, A., Knowles, T., and Finn, C. (2020). Meta-learning symmetries by reparameterization. *arXiv preprint arXiv:2007.02933*.

Zhu, X., Wang, D., Su, G., Biza, O., Walters, R., and Platt, R. (2023). On robot grasp learning using equivariant models. *Auton. Robots*, 47(8):1175–1193.

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, the mathematical setting is introduced in Section 3, which covers Episodic Markov Decision Processes (Section 3.1), Symmetric RL (Section 3.2), and Kernel-Based RL with Symmetries (Section 3.3). In the same section, we also state our technical assumptions, namely Assumptions 1 and 2. The LSVI algorithm is then described in Section 3.5.]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes, a theoretical analysis of LSVI with a totally invariant kernel, leading to novel bounds on sample complexity, is provided in Section 4. In addition, we discuss the time complexity of our approach in the limitations section (Section 5.1).]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes, the code used for the experiments in the synthetic setting and on the Frozen Lake environment is publicly available at: `https://github.com/mtkresearch/IQL`. For the experiments on the 2D placement problem, we provide a clear and detailed description of the setup in Section 5, but we do not share the corresponding code, as it is confidential.]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes, we present in

Section 3 our technical assumptions, namely Assumptions 1 and 2.]

(b) Complete proofs of all theoretical results. [Yes, formal proofs for all theorems and propositions stated in the main text are provided in the appendix. The proof of Proposition 1 can be found in Appendix B, while Proposition 2 is stated and proved in Appendix 2. Theorem 3 is stated and proved in Appendix D, and Theorem 1 is proved in Appendix F. Where our work builds upon lemmas or theorems from prior literature, the original sources are clearly cited, and their relevance to our setting is explained (e.g., Theorem 2).]

(c) Clear explanations of any assumptions. [Yes, our main assumptions are clearly explained in Section 3.4.]

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, the code used for the experiments in the synthetic setting and on the Frozen Lake environment is publicly available at `https://github.com/mtkresearch/IQL`. However, since the 2D placement experiment relies on proprietary code and models, the corresponding implementation cannot be disclosed or made publicly available.]

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes, for comprehensive details—such as hyperparameter tuning, visualizations of the synthetic reward function, and computational resources used—please refer to Appendix G.]

(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, in all our experiments, we report relevant statistical information where applicable. This includes the sample sizes for experimental runs and error bars representing the standard error.]

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes, we make special mention of the type of compute and memory we use in Appendix G.]

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 cited the Scikit-Learn and BoTorch libraries used in our experiments (see Appendix G).]

(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, the code generating our experimental results is publicly available at `https://github.com/mtkresearch/IQL`.]

(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 Group Action

We recall the standard definition. Let $(G, \cdot)$ be a group acting on a space $\mathcal{X}$ via a binary operation $\circ : G \times \mathcal{X} \to \mathcal{X}$ such that:

- $g \circ (h \circ x) = (g \cdot h) \circ x$ for all $g, h \in G$, $x \in \mathcal{X}$
- $e \circ x = x$ for identity element $e \in G$

B Proof of Proposition 1

Proof. Let H be the horizon of the MDP. We will superscript states by $t = 0, ...H$ to indicate the time-step it is observed in a particular trajectory. We'll show this by induction, starting backwards from a terminal state s^H, for which, by definition $Q_\pi(s^H, a) = V_\pi(s^H) = 0$. Now for an arbitrary state s^t, we have:

$$Q_\pi(gs^t, ga) = \sum_{s^{t+1} \in \mathcal{S}} \mathbb{P}(s^{t+1}|gs^t, ga)[r(gs^t, ga, s^{t+1}) + V_\pi(s^{t+1})] \tag{15}$$

$$= \sum_{s^{t+1} \in \mathcal{S}} \mathbb{P}(g^{-1}s^{t+1}|s^t, a)[r(s^t, a, g^{-1}s^{t+1}) + V_\pi(s^{t+1})] \tag{16}$$

$$= \sum_{s^{t+1} \in \mathcal{S}} \mathbb{P}(s^{t+1}|s^t, a)[r(s^t, a, s^{t+1}) + V_\pi(gs^{t+1})] \tag{17}$$

The value function itself satisfies:

$$V_\pi(gs^{t+1}) = \sum_{a \in \mathcal{A}} \pi(a|gs^{t+1}) Q_\pi(gs^{t+1}, a) = \tag{18}$$

$$= \sum_{a \in \mathcal{A}} \pi(g^{-1}a|s^{t+1}) Q_\pi(gs^{t+1}, a) \tag{19}$$

$$= \sum_{a \in \mathcal{A}} \pi(a|s^{t+1}) Q_\pi(gs^{t+1}, ga) \tag{20}$$

$$= \sum_{a \in \mathcal{A}} \pi(a|s^{t+1}) Q_\pi(s^{t+1}, a) = V_\pi(s^{t+1}), \tag{21}$$

where the final two equalities follow from the induction hypothesis.

The final element is showing greedy policies of invariant Q-functions are equivariant.

$$\pi_q(gs) = \arg\max_a q(gs, a) = g^{-1} \arg\max_a q(gs, ga) = g^{-1} \arg\max_a q(s, a) = g^{-1}\pi_q(s) \tag{22}$$

□

C Proposition 2

Denote the orbit of an element x under the action of G by the set $Orb(x) = \{g(x)|g \in G\}$. Under a couple of simplifying assumptions, the dynamics of the MDP reduces to that of a simpler MDP with strictly smaller state space.

Proposition 2. *Assume $\mathcal{M}$ is a G-invariant MDP. If the action on $\mathcal{A}$ is trivial, i.e.*

$$g \circ_{\mathcal{A}} a = a, \forall g \in G, \tag{23}$$

then the following quantity

$$\mathbb{P}(Orb(s')|Orb(s), a) := \mathbb{P}(Orb(s')|s, a) \tag{24}$$

is well-defined and $\mathcal{M}/G = (\mathcal{S}/G, \mathcal{A}, H, \mathbb{P}, r_G)$ given by $\mathcal{S}/G = \{Orb(s)|s \in \mathcal{S}\}$, dynamics $\mathbb{P}$ given by 24, and reward given by $r_G(Orb(s), a) = r(s, a)$ is a well-defined MDP and the optimal policy $\pi^*_{\mathcal{M}/G}$ extends to an optimal policy for $\mathcal{M}$, by

$$\pi^*_{\mathcal{M}}(s) = \pi^*_{\mathcal{M}/G}(Orb(s)). \tag{25}$$

Proof. Equation 24 effectively tells us that the probability of transitioning from one state to another, can be aggregated across orbits in such a way that the corresponding actions are unchanged. Namely given $s_1 = g(s_2)$:

$$\mathbb{P}(Orb(s')|Orb(s_1), a) = \mathbb{P}(Orb(s')|s_1, a) = \mathbb{P}(Orb(s')|g(s_2), a) = \\ = \mathbb{P}(g^{-1}(Orb(s'))|s_2, a) = \mathbb{P}(Orb(s')|s_2, a) = \mathbb{P}(Orb(s')|Orb(s_2), a)$$

Note that we have used Equation 23 here. Now let $\pi^*_{\mathcal{M}/G}$ be an optimal policy for $\mathcal{M}/G$, and $V^*_{\mathcal{M}}$ be the optimal value function for $\mathcal{M}$. Due to Proposition 1, $V^*_{\mathcal{M}}$ is invariant, so $V := V(Orb(s)) := V^*_{\mathcal{M}}(s)$ is well defined. Moreover, by construction, V is a value function corresponding to the policy $\pi^*_{\mathcal{M}}(Orb(s))$. Now by optimality of $\pi^*_{\mathcal{M}/G}$ for all s we have $V(Orb(s)) \leq V^{\pi^*_{\mathcal{M}/G}}(Orb(s))$. Conversely, $V^{\pi^*_{\mathcal{M}/G}}$ defines a valid value function on $\mathcal{M}$ by $V^{\pi^*_{\mathcal{M}/G}}(s) = V^{\pi^*_{\mathcal{M}/G}}(Orb(s))$, and hence, by optimality, $V^{\pi^*_{\mathcal{M}/G}}(s) \leq V^*_{\mathcal{M}}(s)$ for all s. This shows that the two value functions are equivalent, that $\pi^*_{\mathcal{M}/G}(s) := \pi^*_{\mathcal{M}/G}(Orb(s))$ is optimal for $\mathcal{M}$. □

D Tabular Q-learning with symmetry

In recent years, there has been a family of research outputs which study the complexity of Q-learning algorithms in the tabular setting. Typical results such as Jin et al. (2018), give complexity bounds in the tabular setting in terms of the size of the state-action space, $|\mathcal{S} \times \mathcal{A}| = SA$. In our setting, even though we cannot define a simpler dynamics on $\mathcal{S}/G \times \mathcal{A}/G$, we will see that the regret bounds given in terms of SA extend to regret bounds in terms of $\tilde{SA} = |(\mathcal{S} \times \mathcal{A})/G|$.

The gains in sample complexity in general will come from how new state action pairs are observed from unrolling equivariant policies. We will denote a trajectory under π by $\mathcal{T}$ to be a sequence $(s_i, a_i)|_{i=1}^H \in \mathcal{Z}$ with the property that $a_i \sim \pi(s_i)$ and $s_{i+1} \sim \mathbb{P}(s_i, a_i)$, The corresponding action of G on the space of $\mathcal{T}$'s is given by $g(\mathcal{T}) = (g(s_i), g(a_i))|_{i=1}^H$. We then have that if π is equivariant, the following holds:

1. The returns of $\mathcal{T}$ and $g\mathcal{T}$ are the same.
2. $\mathbb{P}_\pi(\mathcal{T}) = \mathbb{P}_\pi(g\mathcal{T})$.

We can use these properties to extend algorithms such as Algorithm 2.

Algorithm 2 Q-learning with UCB-Hoeffding & symmetric experience augmentation.

Initialize $Q_h(s,a) \leftarrow H$ and $N_h(s,a) \leftarrow 0$ for all $(s,a,h) \in S \times A \times [H]$.
for episode $k = 1, \ldots, K$ **do**
  Receive s_1.
  for step $h = 1, \ldots, H$ **do**
    Take action $a_h \leftarrow \arg\max_{a'} Q_h(s_h, a')$, and observe s_{h+1}.
    for $(s,a) \in G(s_h, a_h)$ **do**
      $t = N_h(s,a) \leftarrow N_h(s_h, a_h) + 1$; $b_t \leftarrow c\sqrt{\frac{H^3\iota}{t}}$.
      $Q_h(s,a) \leftarrow (1-\alpha_t)Q_h(s_h, a_h) + \alpha_t\left[r_h(s_h, a_h) + V_{h+1}(s_{h+1}) + b_t\right]$.
      $V_h(s) \leftarrow \min\{H, \max_{a' \in A} Q_h(s_h, a')\}$.
    end for
    end for
  end for

The regret bounds directly translate to:

Theorem 3. *There exists an absolute constant $c > 0$ such that, for any $p \in (0,1)$, if we choose $b_t = c\sqrt{\frac{H^3\iota}{t}}$, then with probability $1-p$, the total regret of Q-learning with UCB-Hoeffding and symmetric experience augmentation (see Algorithm 2) is at most $O(\sqrt{H^4\tilde{SA}T\iota})$, where $\iota := \log\left(\frac{SAT}{p}\right)$.*

Proof of Theorem 3. Proof: The update to the Q-function is now:

$$Q_h^{k+1}(s,a) = \begin{cases} (1-\alpha_t)Q_h^k(s,a) + \alpha_t\left[r_h(s,a) + V_{h+1}^k(s_{h+1}^k) + b_t\right] & \text{if } (s,a) \in G(s_h^k, a_h^k) \\ Q_h^k(s,a) & \text{otherwise} \end{cases} \tag{26}$$

so the Q-function remains constant on orbits of G. Because of this formulation, $Q_h^k(s,a)$ is updated once per episode - horizon pair, the same as in Jin et al. (2018). This means that all results and bounds that apply to a single (s,a) pair follow through. In particular this applies to Lemmas 4.2 and 4.3 from Jin et al. (2018).

Now, turning to their proof for the result, the quantities:

$$\delta_h^k := (V_h^k - V_h^{\pi^k})(gs_h^k) \tag{27}$$

$$\phi_h^k := (V_h^k - V_h^*)(gs_h^k) \text{ and} \tag{28}$$

$$t = n_h^k = N_h^k(gs_h^k, ga_h^k) \tag{29}$$

are still well defined and independent of $g \in G$.

We need to bound $\sum_{k=1}^K \delta_h^k$.

From:

$$\delta_h^k \leq \alpha_t^0 H + \sum_{i=1}^t \alpha_t^i \phi_{h+1}^{k_i} + \beta_t - \phi_{h+1}^k + \delta_{h+1}^k + \xi_{h+1}^k \tag{30}$$

We bound the summation of the first term:

$$\sum_{k=1}^K \alpha_{n_h^k}^0 H = H \cdot \sum_{k=1}^K \mathbb{I}[n_h^k = 0] \leq \tilde{SA}H \tag{31}$$

where the final inequality is due to the fact that $Q_h(s,a)$ gets updated on entire orbits of G on $\mathcal{S} \times \mathcal{A}$, so either $N_h^k(s,a) > 0$ for all $(s,a) \in G(s_h, a_h)$, or $N_h^k(s,a) = 0$ for all $(s,a) \in G(s_h, a_h)$ in which case $Q_h(s,a) = H$ and so $\arg\max_{a'} Q_h(s_h, a') = H$. By D, in the first case, we know that $N_{h-1}^k(s,a) > 0$ for all $(s,a) \in G(s_{h-1}, a_{h-1})$. Unrolling the trajectory backwards, we get $N_1^k(s_1, a) \geq 1$, which can only happen if all $a \in \mathcal{A}$ had been taken in s_1 prior to episode k.

Now, again, following the argument in Jin et al. (2018), we get:

$$\sum_{k=1}^K \delta_1^k \leq O\left(H^2 SA + \sum_{h=1}^H \sum_{k=1}^K (\beta_{n_h^k} + \xi_{h+1}^k)\right) \tag{32}$$

Subsequently:

$$\sum_{k=1}^K \beta_{n_h^k} \leq O(1) \cdot \sum_{k=1}^K \sqrt{\frac{H^3\iota}{n_h^k}} = O(1) \cdot \sum_{(\bar{s},\bar{a}) \in (\mathcal{S}\times\mathcal{A})/G} \sum_{n=1}^{N_h^K(s,a)} \sqrt{\frac{H^3\iota}{n}} \leq O\left(\sqrt{H^3 \tilde{SA} K\iota}\right) = O\left(\sqrt{H^2 \tilde{SA} T\iota}\right) \tag{33}$$

and the final inequality follows due to the fact that $\sum_{(\bar{s},\bar{a}) \in (\mathcal{S}\times\mathcal{A})/G} N_h^K(s,a) = K$ and the left hand side is maximized for $N_h^K(s,a) = K/\tilde{SA}$. □

E Kernel Methods

Mercer Theorem Let $k : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ be a positive definite kernel. Its associated reproducing kernel Hilbert space (RKHS), $\mathcal{H}_k$, consists of functions $f : \mathcal{Z} \to \mathbb{R}$ with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_k}$ and norm $|\cdot|_{\mathcal{H}_k}$, satisfying the reproducing property $f(z) = \langle f, k(\cdot, z) \rangle_{\mathcal{H}_k}$. By Mercer's theorem, k admits an eigenfunction expansion

$$k(z, z') = \sum_{m=1}^\infty \lambda_m \phi_m(z) \phi_m(z'), \tag{34}$$

where $\{\lambda_m\}_{m=1}^\infty$ are positive eigenvalues and $\{\phi_m\}_{m=1}^\infty$ are orthonormal eigenfunctions in $\mathcal{H}_k$. Any $f \in \mathcal{H}_k$ can be expressed as $f = \sum_{m=1}^\infty w_m \sqrt{\lambda_m} \phi_m$, with norm $\|f\|_{\mathcal{H}_k}^2 = \sum_{m=1}^\infty w_m^2$.

Kernel-Based Prediction. For $f \in \mathcal{H}_k$ and data $(z_i, y_i)_{i=1}^t$ with $y_i = f(z_i) + \varepsilon_i$ and zero-mean noise, kernel ridge regression yields posterior mean and variance:

$$\hat{f}_t(z) = k_t^\top(z)(K_t + \lambda I)^{-1}\boldsymbol{y}_t, \quad \sigma_t^2(z) = k(z,z) - k_t^\top(z)(K_t + \lambda I)^{-1}k_t(z), \tag{35}$$

where $k_t(z) = [k(z, z_1), \ldots, k(z, z_t)]^\top$, $K_t = [k(z_i, z_j)]_{i,j=1}^t$, and $\lambda > 0$ is a regularization parameter. Confidence bounds of the form $|f(z) - \hat{f}_t(z)| \leq \beta_t \sigma_t(z)$ hold with high probability under standard conditions.

F Proof of covering number

Since we only care about the dependence of $\log \mathcal{N}(\epsilon, B, R, G)$ on the newly introduced variable G, in this section we will simplify notation. Firstly, the definition of the state-action value function class,

$$\mathcal{Q} = \{Q = \hat{Q}_T(\cdot) + \beta\sigma_T(\cdot), \forall\{z_t\}_{t=1}^T \subset \mathcal{Z}\}.$$

and the notation $\mathcal{N}(\epsilon)$ for its ϵ-covering number. Let us use the notation $\mathcal{N}_{k,R}(\epsilon)$ for the ϵ-covering number of RKHS ball $\mathcal{B}_{k,R} = \{f : \|f\|_{\mathcal{H}_k} \leq R\}$, $\mathcal{N}_{[0,B]}(\epsilon)$ for the ϵ-covering number of interval $[0, B]$ with respect to Euclidean distance, and $\mathcal{N}_{k,\boldsymbol{b}}(\epsilon)$ for the ϵ-covering number of class of uncertainty functions $\boldsymbol{b}_k = \{b(z) = \left(k(z,z) - k_Z^\top(z)(K_Z + \lambda^2 I)^{-1}k_Z(z)\right)^{\frac{1}{2}}, |Z| \leq T\}$.

Consider $Q, \bar{Q} \in \mathcal{Q}$ where $Q(z) = \min\{Q_0(z) + \beta b(z), H - h + 1\}$ and $\bar{Q}(z) = \min\left\{\bar{Q}_0(z) + \bar{\beta}\bar{b}(z), H - h + 1\right\}$. We have

$$|Q(z) - \bar{Q}(z)| \leq |Q_0(z) - \bar{Q}_0(z)| + |\beta - \bar{\beta}| + B|b(z) - \bar{b}(z)|. \tag{36}$$

That implies

$$\mathcal{N} \leq \mathcal{N}_{k,R}(\frac{\epsilon}{3})\mathcal{N}_{[0,B]}(\frac{\epsilon}{3})\mathcal{N}_{k,\boldsymbol{b}}(\frac{\epsilon}{3B}). \tag{37}$$

For the ϵ-covering number of the $[0, B]$ interval, we simply have $\mathcal{N}_{[0,B]}(\epsilon/3) \leq 1 + 3B/\epsilon$. In the next lemmas, we bound the ϵ-covering number of the RKHS ball and the class of uncertainty functions.

Lemma 2 (RKHS Covering Number)**.** *Consider a positive definite kernel $k : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$. Under Assumption 2, the ϵ-covering number of R-ball in the RKHS satisfies*

$$\log \mathcal{N}_{k,R}(\epsilon) = \mathcal{O}\left(\left(\frac{R^2}{\epsilon^2|G|^p}\right)^{\frac{1}{p-1}} \log(1 + \frac{R}{\epsilon})\right). \tag{38}$$

Lemma 3 (Uncertainty Class Covering Number)**.** *Consider a positive definite kernel $k : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$. Under Assumption 2, the ϵ-covering number of the class of uncertainty functions satisfies*

$$\log \mathcal{N}_{k,\boldsymbol{b}}(\epsilon) = \mathcal{O}\left((\frac{1}{\epsilon^2|G|^p})^{\frac{2}{p-1}}(1 + \log(\frac{1}{\epsilon}))\right) \tag{39}$$

Combining equation 37 with Lemmas 2 and 3, we obtain

$$\log \mathcal{N}(\epsilon) = \mathcal{O}\left((\frac{R^2}{\epsilon^2|G|^p})^{\frac{1}{p-1}}(1 + \log(\frac{R}{\epsilon})) + (\frac{B^2}{\epsilon^2|G|^p})^{\frac{2}{p-1}}(1 + \log(\frac{B}{\epsilon}))\right), \tag{40}$$

that completes the proof of the theorem. Next, we provide the proof of two lemmas above on the covering numbers of the RKHS ball and the uncertainty function class.

Proof of Lemma 2. For f in the RKHS, recall the following representation

$$f = \sum_{m=1}^{\infty} w_m \sqrt{\lambda_m}\phi_m, \tag{41}$$

as well as its projection on the D-dimensional RKHS

$$\Pi_D[f] = \sum_{m=1}^{D} w_m \sqrt{\lambda_m}\phi_m. \tag{42}$$

We note that

$$\begin{aligned}
\|f - \Pi_D[f]\|_\infty &= \sum_{m=D+1}^{\infty} w_m \sqrt{\lambda_m}\phi_m \\
&\leq C_1 \sum_{m=D+1}^{\infty} |w_m|(m\theta)^{-\frac{p}{2}} \\
&\leq C_1 \left(\sum_{m=D+1}^{\infty} |w_m|^2 \right)^{\frac{1}{2}} \left(\sum_{m=D+1}^{\infty} (m\theta)^{-p} \right)^{\frac{1}{2}} \\
&\leq C_1 R \left(\int_D^{\infty} x^{-p} dx \right)^{\frac{1}{2}} \\
&= \frac{C_1 R \theta^{-\frac{p}{2}}}{\sqrt{p-1}} D^{\frac{-p+1}{2}}.
\end{aligned}$$

The second inequality follows form Cauchy–Schwarz inequality.

Now, let D_0 be the smallest D such that the right hand side is bounded by $\frac{\epsilon}{2}$. There exists a constant $C_2 > 0$, only depending on constants C_1 and p, such that

$$D_0 \leq C_2 \left(\frac{R\theta^{-p/2}}{\epsilon} \right)^{\frac{2}{p-1}}. \tag{43}$$

For a D-dimensional linear model, where the norm of the weights is bounded by R, the ϵ-covering is at most $C_3 D(1 + \log(\frac{R}{\epsilon}))$, for some constant C_3 (e.g., see, Yang et al., 2020a). Using an $\epsilon/2$ covering number for the space of $\Pi_D[f]$ and using the minimum number of dimensions that ensures $|f - \Pi_D[f]| \leq \epsilon/2$, we conclude that

$$\begin{aligned}
\log \mathcal{N}_{k,R}(\epsilon) &\leq C_3 D_0 (1 + \log(\frac{R}{\epsilon})) \\
&\leq C_2 C_3 \left(\frac{R\theta^{-p/2}}{\epsilon} \right)^{\frac{2}{p-1}} (1 + \log(\frac{R}{\epsilon})),
\end{aligned}$$

that completes the proof of the lemma.

□

Proof of Lemma 3. Let us define $\boldsymbol{b}_k^2 = \{b^2 : b \in \boldsymbol{b}_k\}$ and $\mathcal{N}_{k,\boldsymbol{b}^2}(\epsilon)$ to be its ϵ-covering number. We note that, for $b, \bar{b} \in \boldsymbol{b}$,

$$|b(z) - \bar{b}(z)| \leq \sqrt{|(b(z))^2 - (\bar{b}(z))^2|}. \tag{44}$$

Thus, an ϵ-covering number of $\boldsymbol{b}$ is bounded by an ϵ^2-covering of $\boldsymbol{b}^2$:

$$\mathcal{N}_{k,\boldsymbol{b}}(\epsilon) \leq \mathcal{N}_{k,\boldsymbol{b}^2}(\epsilon^2). \tag{45}$$

We next bound $\mathcal{N}_{k,\boldsymbol{b}^2}(\epsilon^2)$.

Using the feature space representation of the kernel, we obtain

$$(b(z))^2 = \sum_{m=1}^{\infty} \gamma_m \lambda_m \phi_m^2(z), \tag{46}$$

for some $\gamma_m \in [0,1]$. Based on the GP interpretation of the model, γ_m can be understood as the posterior variances of the weights. Let D_0 be the smallest D such that $\sum_{m=D+1}^{\infty} \lambda_m \phi_m^2(z) \leq \epsilon^2/2$. Similar to the previous lemma, we can show that, for some constant C_4, only depending on constants C_1 and p,

$$D_0 \leq C_4 \left(\frac{1}{\theta^p \epsilon^2}\right)^{\frac{1}{p-1}}. \tag{47}$$

For $\sum_{m=1}^{D_0} \gamma_m \lambda_m \phi_m^2(z)$ on a finite D_0-dimensional spectrum, as shown in Lemma $D.3$ of Yang et al. (2020a), an $\epsilon^2/2$ covering number scales with D_0^2. Specifically, an $\epsilon^2/2$ covering number of the space of $\sum_{m=1}^{D_0} \gamma_m \lambda_m \phi_m^2(z)$ is bounded by

$$C_5 D_0^2 (1 + \log(\frac{1}{\epsilon})). \tag{48}$$

Combining Equations equation 47 and equation 48, we obtain

$$\begin{aligned} \mathcal{N}_{k,\boldsymbol{b}^2}(\epsilon^2) &\leq C_5 D_0^2 (1 + \log(\frac{1}{\epsilon})) \\ &\leq C_5 C_4^2 \left(\frac{1}{\theta^p \epsilon^2}\right)^{\frac{2}{p-1}} (1 + \log(\frac{1}{\epsilon})), \end{aligned}$$

that completes the proof of the lemma. □

Proof of Corollary 1. From the definition of k^*:

$$\kappa^* = \max\left(\xi^*, \frac{2d+p+1}{(d+p)\cdot[p-1]}, \frac{2}{p-3}\right),$$
$$\xi^* = \frac{d+1}{2(p+d)}.$$

Then:

$$\mathcal{N}(\epsilon(T), B_T, R_T, G) = \tilde{\mathcal{O}}\left(T^{\frac{4p^2+p-1}{2p^2-p}} \left(\frac{1}{|G|}\right)^{\frac{2p-1}{p-1}} + T^{k^*+\frac{4p}{p-1}} \left(\frac{1}{|G|}\right)^{\frac{2p}{p-1}}\right)$$

which is dominated by the $\Gamma_{k_G}(T)$ term in its dependence on $|G|$, hence the result. □

G Experimental details

Here, we outline the procedure of generating r and P test functions from the RKHS in the synthetic setting. We also detail the hyperparameters used in both the synthetic and Frozen Lake experiments, along with the computational resources required.

G.1 Synthetic Setting

The reward function r and transition probability P are chosen as arbitrary functions from the RKHS of the invariant kernel k_G. To generate r, we draw a Gaussian Process (GP) sample on a subset of the domain $\mathcal{Z}$. This subset consists of evenly spaced points on a 5×5 grid covering the range $[-1, 1]$ in both dimensions. Kernel ridge regression is then fitted to these samples, and the predictions are scaled to the $[0, 1]$ range to produce r (see Figure 5). To construct $P(s'|s, a)$, 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 yield a valid conditional probability distribution $P(\cdot|z)$. This is a common approach to create functions belonging to an RKHS (e.g., see, Chowdhury and Gopalan, 2017). We use RBF as the base kernel of k_G, with length scale $= 0.1$ and $\lambda = 0.01$.

For the KRVI algorithm, we use a length scale of 1, $\lambda = e^{-10}$, and a confidence interval width multiplier $\beta = 0.1$ for both the standard RBF and the invariant kernel. Results are averaged over 20 random seeds. Kernel ridge regression is implemented using the Scikit-Learn library (Pedregosa et al., 2011), which provides robust and efficient tools for kernel-based machine learning models. Simulations are run on a computing cluster with 512 GiB of RAM and two Intel(R) Xeon(R) Gold 6248 CPUs running at 2.5 GHz.

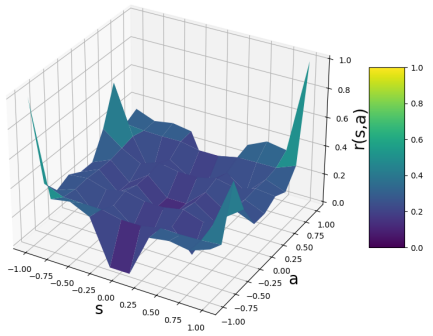


Figure 5: Reward function $r(s, a)$ generated by kernel ridge regression using an invariant kernel with lengthscale $= 0.1$ and $\lambda = 0.01$

G.2 Frozen Lake Experiments

For the Fixed Layout Frozen Lake and Random Layout Frozen Lake experiments, we use the BoTorch library (Balandat et al., 2020), specifically the SingleTaskGP model. Simulations are conducted on a node with 376.2 GiB of RAM and an Intel(R) Xeon(R) Gold 5118 CPU running at 2.30 GHz. We use the code for invariant kernels made publicly available by (Brown et al., 2024).

We tune the confidence interval width multiplier β for each setting (Fixed Layout and Random Layout) for both the invariant and standard RBF kernels by selecting the best-performing value from the grid $\{0.01, 0.1, 0.5, 1\}$. The kernel length scale is also tuned over the set $\{0.1, 0.5, 1\}$. The noise regularization parameter λ is initialized to 0.1 and optimized during training by maximizing the marginal log-likelihood, as handled by the BoTorch library.

To assess sensitivity to β, Table 2 reports cumulative returns for different values of β on the Frozen Lake (Fixed) environment for both the standard RBF and invariant kernels, while keeping all other hyperparameters fixed to their selected values.

Table 2: Cumulative returns for different β values on Frozen Lake (Fixed), with standard RBF and invariant kernels.

	RBF kernel				Invariant kernel			
Episodes	$\beta = 0.01$	$\beta = 0.1$	$\beta = 0.5$	$\beta = 1.0$	$\beta = 0.01$	$\beta = 0.1$	$\beta = 0.5$	$\beta = 1.0$
250	∼22	∼22	∼22	∼22	∼42	∼43	∼11	∼8
500	∼77	∼47	∼52	∼53	∼138	∼138	∼24	∼23
1000	∼318	∼174	∼85	∼83	∼500	∼371	∼41	∼46
1500	∼779	∼381	∼129	∼123	∼964	∼626	∼62	∼86

These results indicate that smaller values of β tend to perform better in practice, as larger values can lead to excessive exploration. This trend is consistent across both RBF and invariant kernels and motivates the selection of $\beta = 0.01$ in our experiments. We observe similar qualitative behavior across settings.

For **FrozenLake (Fixed)**, the selected hyperparameters are:

- RBF kernel: $\beta = 0.01$, length scale $= 0.1$, λ initialized to 0.1 (optimized during training).
- Invariant kernel: $\beta = 0.01$, length scale $= 0.5$, λ initialized to 0.1 (optimized during training).

For **FrozenLake (Random)**, the selected hyperparameters are:

- RBF kernel: $\beta = 0.01$, length scale = 1, λ initialized to 0.1 (optimized during training).
- Invariant kernel: $\beta = 0.01$, length scale = 0.5, λ initialized to 0.1 (optimized during training).

G.3 Additional Experiments: KOVI (with and without symmetry) vs. DQN (with and without symmetry)

To evaluate the performance of our kernel-based approach (KOVI with a standard RBF kernel) and its symmetry-aware variant (KOVI with an invariant kernel) against neural network–based methods, we compared them with DQN and its symmetrized counterpart (SymDQN) (see Figure 4). This comparison highlights the practical advantages of our kernel-based methods and provides insight into how they perform relative to deep RL approaches that explicitly incorporate symmetry (e.g., equivariant networks). Both DQN and SymDQN were implemented with the same base network architecture, with SymDQN using an equivariant policy network within the Stable-Baselines3 API, tailored to the rotational symmetries of the FrozenLake (Fixed) environment.

The results show that KOVI is substantially more sample-efficient than DQN, converging to the highest return with significantly fewer environment timesteps. Likewise, KOVI with an invariant kernel converges faster than SymDQN. Moreover, in both the kernel-based and neural approaches, the symmetry-aware variants consistently outperform their non-symmetric counterparts, highlighting the benefits of incorporating symmetry. Overall, kernel-based methods (with and without symmetry) achieve higher sample efficiency than their neural counterparts. These findings demonstrate that kernel methods can be more effective than neural networks, particularly in low-data regimes. While neural networks are known to be more scalable, they generally require large amounts of data to perform well. In contrast, kernel-based approaches tend to train faster and achieve superior performance when data are limited. This makes kernel methods especially valuable in structured, data-scarce environments where prior knowledge—such as symmetry—can be effectively exploited.

For completeness, DQN was run with light hyperparameter tuning over a grid of discrete values for the following parameters: `learning_starts` $\in \{0, 1000\}$, `train_freq` $\in \{1, 100\}$, `batch_size` $\in \{128, 256\}$, `exploration_fraction` $\in \{0.01, 0.05\}$, `target_update_interval` $\in \{500, 1000\}$, and `learning_rate` $\in \{1\text{e}{-4}, 5\text{e}{-5}\}$. The final configuration, which performed best, was: `learning_starts = 1000`, `train_freq = 1`, `gradient_steps = 1`, `batch_size = 256`, `exploration_fraction = 0.05`, `exploration_initial_eps = 1`, `exploration_final_eps = 0.05`, `target_update_interval = 500`, `learning_rate = 0.0001`, `buffer_size = 100000`.

G.4 SynPl experiments

We reuse most of the settings from the Frozen Lake experiments, but make relevant modifications to the environment to encode actions and states correctly. Since both actions and states represent positions on an 8x8 grid, we can produce a uniform representation for states and actions. Each position vector is a half integer in $[-4, 4]^2$ corresponding to the centers of squares of an 8x8 centered lattice at (0,0). Hyperparameter tuning is performed for β, the starting length scale and the starting noise regularization. We run experiments either with these values fixed or with optimizing these values in BoTorch and warm-starting the optimization in subsequent episodes. We perform hyperparameter tuning separately for the RBF baseline and for the invariant kernel, averaging performance over 5 seeds and then selecting the best configuration of hyperparameters. We then rerun the best configurations for 3 seeds for a total of 4000 episodes. Across both configurations, BoTorch optimization of the initial parameters provides improvement, and the optimal values found are $\beta = 0.1$ for the invariant kernel, $\beta = 0.05$ for the RBF kernel. For both kernels the optimal length scale was 1, and optimal noise regularization was 10^{-6}.