Kernel-Based Safe Exploration in Deep Reinforcement Learning

Rupak Majumdar RUPAK@MPI-SWS.ORG
Nikhil Singh NIKSINGH@MPI-SWS.ORG
Sadegh Soudjani SADEGH@MPI-SWS.ORG
Max Planck Institute for Software Systems, Germany

Editors: G. Sukhatme, L. Lindemann, S. Tu, A. Wierman, N. Atanasov

Abstract

Safety has been a major concern when deploying deep reinforcement learning algorithms in the real world. A promising direction that ensures that the learned policy does not visit unsafe regions is to learn a *barrier function* along with the policy. A barrier is a function from states to reals that assigns low values to the initial states, high values to the unsafe states, and decreases in expectation on each transition; such a function can be used to bound the probability of reaching unsafe states. Previous attempts learned a barrier function directly from exploration data, but this required either large amounts of data or restrictions on the system dynamics. In this paper, we show how kernel embeddings can be used to learn barrier functions during deep reinforcement learning for stochastic systems with unknown dynamics. Our algorithm, *kernel-based safe exploration (KBSE)*, learns an optimal policy and a barrier simultaneously during exploration. The barriers are computed iteratively, represented as conditional mean embeddings, and provide better probabilistic safety guarantees with more exploration. The exploration algorithm uses the learned barrier functions to identify safety violations. In the case of violation, it intervenes to modify the unsafe action to a safe action, thereby ensuring that the exploration is restricted to actions that bound the probability of reaching unsafe states. We evaluate KBSE on several complex continuous control benchmarks. Experimental results establish our new algorithm to be suitable for synthesizing control policies that are probabilistically safe without degradation in reward accumulation.

1. Introduction

In recent years, many challenging problems in optimal policy synthesis, including solving Atari (Mnih et al., 2015), Go (Silver et al., 2017), biped walking (Levine et al., 2016) and Language Models (Naveed et al., 2025), have been solved using reinforcement learning (RL). This is attributed to RL's ability to maximize cumulative rewards through online exploration, without requiring a model of the underlying dynamics.

For safety-critical, high-dimensional, control problems, an obstacle to deploying RL-based controllers is the possibility that a reward-maximizing controller may lead the system into unsafe states. Thus, a challenging research direction is to learn an optimal policy that is constrained by safety requirements. While there already exists a rich literature on this problem, existing solutions are still unsatisfactory: either because they do not provide certificates of correctness, or because they require data-intensive techniques, or because the learning process is subject to oscillations or high approximation errors.

In this paper, we propose an online, kernel-based approach to learning safe control policies under unknown dynamics. Our algorithm, *kernel-based safe exploration (KBSE)*, learns an optimal policy and a barrier function simultaneously. The policy optimizes reward accumulation while being constrained by a barrier function. The barrier function (Prajna, 2006) provides a certificate that the

system violates the safety specification with a bounded probability with high confidence (Schön et al., 2024). In particular, the exploration is constrained to use actions that do not violate the safety specification. Simultaneously, the exploration is used to update the barrier function to provide better safety guarantees over time.

Intuitively, a barrier function maps states to reals such that the function (a) assigns a low value to initial states, (b) assigns a high value to unsafe states, and (c) reduces in expectation on each step of the dynamics—the existence of such a function implies an upper bound on the probability that an unsafe state may be reached. The technical challenge in learning barrier functions is the conditional expectation in requirement (c). The key to our technique is the representation of barrier functions using conditional mean embeddings (CMEs) (Song et al., 2009; Klebanov et al., 2020) in reproducing kernel Hilbert spaces (RKHS) (Schölkopf and Smola, 2002; Berlinet and Thomas-Agnan, 2004; Steinwart and Christmann, 2008). An RKHS is a Hilbert space of functions characterized by the property that pointwise evaluation is a continuous linear functional. This property renders RKHS a powerful framework for studying and manipulating functions in high-dimensional spaces. A CME embeds conditional probability distributions into an RKHS, allowing for the computation of conditional expectations via simple inner products. In particular, using CMEs, we can synthesize barrier functions using a simple linear optimization problem.

We evaluate KBSE on several challenging continuous control benchmarks available in the Gym (Brockman et al., 2016) classical control and Mujoco-based environments. For each of these benchmarks, we consider safety specifications with requirements more stringent than those used in Gym environments. Our approach generalizes the safety requirements by allowing the violation of safety constraints with a certain probability threshold. This enables us to study systems for which no policy exists to satisfy safety almost surely. We compare the performance of the control policy synthesized by KBSE against the most popular off-policy safe RL algorithms. Experimental results show that the KBSE algorithm is superior to the baseline algorithms in terms of reward accumulation and safety cost. In addition, KBSE also provides the safety probability for the learned control policies which is not possible in the case of baseline algorithms. We have relegated the proofs of statements and additional details to the supplementary material.

2. Related Work

Constrained RL. Learning policies subject to constraints has been studied widely using a range of techniques including constrained policy optimization (Achiam et al., 2017), Lagrangian multipliers (Ray et al., 2019; Stooke et al., 2020), and penalty-based methods (Liu et al., 2020). While these approaches are subject to inherent oscillations, initial value dependency, large approximation errors, or large computational costs, they are not able to encode safety probability thresholds in their policy learning. Our approach integrates directly the bounds on safety probabilities in the learning through the concept of barrier functions.

Barrier functions in RL. Recent strategies in safe RL use various types of safety certificates to ensure constraint satisfaction during exploration. The work by Li and Belta (2019); Yang et al. (2023) uses barrier and Lyapunov functions combined with RL. Cheng et al. (2019) propose RL-CBF, providing safety guarantees with high probability assuming deterministic dynamics (known nominal dynamics and partially unknown dynamics). The approach by Wang et al. (2023) uses barrier functions for safe RL in stochastic dynamics using generative models. These approaches are subject to handling deterministic systems, being limited to low-dimensional systems, or requiring

large amounts of data to learn parameters of deep neural networks (DNNs) representing the barrier certificates and shields. In contrast, our approach studies unknown stochastic systems using non-parametric kernel-based methods (Hofmann et al., 2008; Meanti et al., 2020), which makes it applicable to large-dimensional systems.

Kernel methods in RL. Kernel methods have been explored for policy synthesis (Ormoneit and Sen, 2002; Barreto et al., 2016). The work by Romao et al. (2023) uses kernel mean embedding in value iteration to find the optimal policy, thus being applicable to small-dimensional benchmarks. The work by Domingues et al. (2021) provides a model-based algorithm and uses kernel smoothing to estimate rewards and transitions and obtains a regret bound for kernel-based RL. Thorpe and Oishi (2021); Thorpe et al. (2023) use conditional distribution embeddings to represent a model-free controller synthesis problem as a linear program in RKHS. However, this approach does not take into account safety constraints, which we address here.

Novelty of our formulation. Our problem formulation is distinct from the standard safe RL approaches in the literature. In safe RL with shielding (Alshiekh et al., 2018; Reed and Lahijanian, 2024), the safety requirement is specified as the level set of a function, and it disables actions that lead the system to an unsafe state. This means that the designed policy must satisfy the safety constraint almost surely (with probability 1). In contrast, our approach generalizes the safety requirements by allowing the violation of safety constraints with a certain probability threshold. In our problem formulation, we intentionally permit a certain level of safety violation due to the constraints in the definition of the barrier function. Our aim is to learn a control policy that gives safety with a certain probability without degradation in performance.

Our approach is also distinct from the available constrained RL techniques (Achiam et al., 2017; Liu et al., 2020; Stooke et al., 2020) as they consider constraints in the form of a bound on an additive or average objective. The probabilistic safety constraint in our formulation cannot be written in the form of additive or average objectives. Thus, these approaches are not applicable to our problem formulation.

3. Preliminaries

Notation. The set of reals, non-negative reals, and non-negative integers are denoted respectively by $\mathbb{R}$, $\mathbb{R}_{\geq 0}$ and $\mathbb{N}$. Consider a Polish sample space $\mathbb{X}$ with underlying probability space $(\mathbb{X}, \mathcal{B}(\mathbb{X}), \mathbb{P})$ endowed with a Borel σ-algebra $\mathcal{B}(\mathbb{X})$ and a probability measure $\mathbb{P}$. We denote a random variable by X and the instantiations are denoted by x. For a random variable X, let p_X be pushforward probability measure of $\mathbb{P}$ under X such that $X \sim p_X(\cdot)$. For a function $f(X)$, the expected value on $\mathbb{X}$ is given by $\mathbb{E}_{p_X}[f(X)]$. For a measurable space $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$, the set of all probability measures is denoted by $\mathcal{P}(\mathbb{X})$. For two measurable spaces $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$, $(\mathbb{Y}, \mathcal{B}(\mathbb{Y}))$, a probability kernel is defined as the mapping $p : \mathbb{X} \times \mathcal{B}(\mathbb{Y}) \to [0, 1]$ such that $p(x, \cdot) : \mathcal{B}(\mathbb{Y}) \to [0, 1]$ is a probability measure for all $x \in \mathbb{X}$ and $p(\cdot, B) : \mathbb{X} \to [0, 1]$ is measurable for all $B \in \mathcal{B}(\mathbb{Y})$. For each $x \in \mathbb{X}$, the conditional probability measure $p(x, \cdot)$ is also denoted by $p(\cdot|x)$.

Constrained Reinforcement Learning. We represent the environment $\mathcal{M}$ as a Markov Decision Process (MDP) $\mathcal{M} = \langle S, A, \mathcal{T}, R \rangle$, where $S \subset \mathbb{R}^p$ denotes the set of continuous states of the system and $A \subset \mathbb{R}^q$ denotes the set of continuous control actions. The function $\mathcal{T} : S \times A \times \mathcal{B}(S) \to [0, 1]$ is a probability kernel, where $\mathcal{T}(s, a, \cdot)$ is the probability measure for transitioning to the next state from the current state $s \in S$ under action $a \in A$. The probability kernel $\mathcal{T}$ is assumed to be unknown. The function $R : S \to \mathbb{R}$ represents the reward function for $\mathcal{M}$. A transition (or sample)

at time $t \in \mathbb{N}$ is denoted by $\langle s_t, a_t, r_t, s_{t+1} \rangle$, where $s_t \in S$ is a state, $a_t \in A$ is an action, $r_t \in \mathbb{R}$, such that $r_t = R(s_t)$, is the reward obtained by the agent by performing action a_t at state s_t, and $s_{t+1} \in S$ is the next state selected randomly according to the probability measure $\mathcal{T}(\cdot|s_t, a_t)$.

A Constrained Markov Decision Process (CMDP) is a tuple $\langle S, A, \mathcal{T}, R, \varphi \rangle$, which is an MDP $\langle S, A, \mathcal{T}, R \rangle$ augmented with a *safety specification* $\varphi = \langle S_u, T \rangle$, in which $S_u \subset S$ is the set of unsafe states and $T \in \mathbb{N}$ is a time horizon.

A parameterized *policy* is a probability kernel $\pi : S \times \Theta \times \mathcal{B}(A) \to [0, 1]$, where Θ is the set of parameters, and for each $\theta \in \Theta$, $\pi_\theta(\cdot|s) = \pi(s, \theta, \cdot)$ is a conditional probability measure for selecting the action a at state s, given parameters θ. A probability measure $\rho_0 : \mathcal{B}(S) \to [0, 1]$ for the initial state s_0 and a policy π_θ together determines a probability measure χ over trajectories $\omega = \langle s_0, a_0, s_1, \ldots, s_T \rangle$, given by $\chi(\rho_0, \theta)(\omega) = \rho_0(ds_0) \prod_{t=0}^{T} \pi_\theta(da_t|s_t) \cdot \mathcal{T}(ds_{t+1}|s_t, a_t)$.

We denote the probability of event with respect to this measure by $\mathbb{P}^\theta_{\rho_0}[\cdot]$ and expectations of random variables by $\mathbb{E}^\theta_{\rho_0}[\cdot]$. When the initial state is fixed to a single state s_0, we replace ρ_0 with s_0.

The safe reinforcement learning problem asks to compute an optimal policy π_{θ^*} that maximizes the expected discounted sum of rewards up to horizon T while ensuring that the probability that the MDP reaches an unsafe state within the horizon is at most a given bound δ. Formally,

Problem 1 (Constrained Policy Synthesis) *Given a CMDP $\langle S, A, \mathcal{T}, R, \langle S_u, T \rangle \rangle$ with unknown $\mathcal{T}$, initial state $s_0 \in S$, parametrized policy π_θ, discount factor $\gamma \in [0, 1)$, and a parameter δ, synthesize an optimal policy π_{θ^*} that maximizes the expected discounted sum of rewards while ensuring that the probability of reaching S_u is bounded above by δ. Mathematically, find an optimal policy π_{θ^*} such that*

$$\theta^* = \arg\max_{\theta \in \Theta} \mathbb{E}^\theta_{s_0} \left[\sum_{t=0}^{T} \gamma^t R(s_t) \right] \quad \text{s.t.} \quad \mathbb{P}^\theta_{s_0}\{\text{some state in } \omega \text{ is in } S_u\} \leq \delta. \tag{1}$$

The use of probability threshold enables us to generalize the safety requirements by allowing the violation of safety constraints up to a threshold and improve the optimal objective. This also enables us to study systems for which no policy exists to satisfy safety almost surely. Solving Problem 1 requires handling two challenges: (a) ensuring the probabilistic safety constraint, and (b) finding the policy under the unknown probability kernel $\mathcal{T}$. We utilize the concepts of barrier functions and Conditional Mean Embedding (CME) to tackle these challenges, as described below after giving a motivating example.

A motivating example. We motivate our technique using the classical example of controlling an inverted pendulum (Brockman et al., 2016) to maintain its upright position.

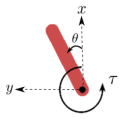

Figure 1: Pendulum

A pendulum freely hangs in the downward position, and the default goal is to balance it vertically upward. Normally, the control policy learns to balance upright by swinging the pendulum by one full round to gain sufficient momentum to bring it to the vertical position. However, if the starting configuration of the pendulum is in upward direction (above horizontal), a full swing might not be needed, and it can balance it upright with less effort. In this case, our safety specification requires that the pendulum should not go for a full swing. Given that the starting state of the

pendulum is feasible (sufficiently above horizontal position), our goal is to learn a control policy that balances the pendulum in the vertical upward position without going for a full swing.

Barrier functions. The notion of Barrier function provides a technique to define locally checkable conditions that guarantee satisfaction of the safety requirements in CMDPs. Fix a CMDP $\langle S, A, \mathcal{T}, R, \langle S_u, T\rangle\rangle$, a policy π_θ, and an initial state s_0. A function $B : S \to \mathbb{R}_{\geq 0}$ is a *barrier function* w.r.t. the unsafe set S_u if it satisfies the following conditions for some constants $\nu > \eta \geq 0$ and $c \geq 0$:

(i) $B(s_0) \leq \eta$; **(ii)** $\forall s \in S_u : \ B(s) \geq \nu$; and **(iii)** $\forall s \in S : \ \mathbb{E}_s^\theta[B(s^+)|s] - B(s) \leq c$;
where s^+ is the next state of the CMDP from the current state s for the policy π_θ.

Theorem 1 (Kushner (1967)) *If there is a non-negative barrier function $B : S \to \mathbb{R}_{\geq 0}$ for the CMDP $\langle S, A, \mathcal{T}, R, \langle S_u, T\rangle\rangle$, then $\mathbb{P}^\theta_{s_0}[$some state in ω is in $S_u] \leq \frac{\eta + cT}{\nu}$.*

Theorem 1 gives a bound on safety using barrier certificates and assuming full knowledge of the model. Problem 1 assumes the model of the system is unknown and requires solving a safety-constrained optimization. Our proposed approach fills the gap between Theorem 1 and the solution to Problem 1.

Reproducing Kernel Hilbert Spaces (RKHS). Let $\mathcal{H}$ be a Hilbert space of functions $f : S \to \mathbb{R}$ with inner product $\langle \cdot, \cdot \rangle_\mathcal{H}$ and a kernel $k : S \times S \to \mathbb{R}$. Each RKHS is equipped with a dot product which satisfies the reproducing property:

$$\langle f(\cdot), k(s, \cdot)\rangle_\mathcal{H} = f(s) \quad \text{and} \quad \langle k(s, \cdot), k(s', \cdot)\rangle_\mathcal{H} = k(s, s') \ .$$

Intuitively, a function evaluation at any point $s \in S$ can be expressed as an inner product. The function k is called the *reproducing kernel* since it reproduces the value of f at s via the inner product. For any set Y, we will use the notations k_Y and $\mathcal{H}_{k_Y}$ to denote respectively the kernel and its RKHS on Y.

Definition 1 (Mean embedding (ME)) *Consider a kernel $k_S : S \times S \to \mathbb{R}$ and the RKHS $\mathcal{H}_{k_S}$ on S. The mean embedding (ME) of a probability measure $p : \mathcal{B}(S) \to [0, 1]$ is given via the mean map $\mu_{k_S} : \mathcal{P}(S) \to \mathcal{H}_{k_S}$:*

$$\mu_{k_S}(p) = \mathop{\mathbb{E}}_{s\sim p}[k_S(\cdot, s)] = \int_S k_S(\cdot, s) \cdot dp(s) \ . \tag{2}$$

The reproducing property carries on to the ME, allowing to compute the expected value of a function $f \in \mathcal{H}_{k_S}$ via its inner product with the ME : $\mathbb{E}_{s\sim p}[f(s)] = \langle f, \mu_{k_S}(p)\rangle_{\mathcal{H}_{k_S}}$. Using ME, we define the maximum mean discrepancy (MMD) to be the distance between two probability measures p, p' in $\mathcal{H}_{k_S}$ as $\| \mu_p(\cdot) - \mu_{p'}(\cdot) \|_{\mathcal{H}_{k_S}}$ (Gretton et al., 2012).

Definition 2 (Conditional mean embedding (CME)) *Given the RKHS $\mathcal{H}_{k_{SA}}$ on the space $S \times A$ with a kernel k_{SA}, and the RKHS $\mathcal{H}_{k_S}$ on the space S with a kernel k_S, The CME of a conditional probability measure $p : S \times A \times \mathcal{B}(S) \to [0, 1]$ is an $S \times A$-measurable random variable taking values in $\mathcal{H}_{k_S}$ given by*

$$\mu(p)(\cdot, s, a) = \mathop{\mathbb{E}}_{s^+\sim p(s,a,\cdot)}[k_S(\cdot, s^+)|s, a] \ . \tag{3}$$

Analogous to the non-conditional case, we can compute the expected value of a function $f \in \mathcal{H}_{k_S}$ via its inner product with the CME: $\mathop{\mathbb{E}}_{s^+\sim p(s,a,\cdot)}[f(s^+)|s,a] = \langle f(\cdot), \mu(p)(\cdot,s,a)\rangle_{\mathcal{H}_{k_S}}$. To establish barrier functions, we start by reformulating the left-hand side of condition 3 of the barrier function definition as an inner product with the CME, i.e., we have

$$\mathbb{E}_{\mathcal{T}}[B(s^+)|s,a] = \langle B, \mu(s,a)\rangle_{\mathcal{H}_{k_S}} . \tag{4}$$

Remark 3 *For mathematical consistency, we restrict the unknown model of the system to a known function space. In particular, we assume that the CME of the probability kernel $\mathcal{T}$ lives in a vector-valued RKHS of functions from $S \times A$ to $\mathcal{H}_{k_s}$, denoted by $\mathcal{G}$ (Park and Muandet, 2020).*

We utilize the above observation and empirical computation of the CME to provide a solution for Problem 1, as described in the next section.

4. Kernel-based Safe Exploration

Our goal is to solve Problem 1 by simultaneously learning an optimal policy and a barrier function using data from the unknown CMDP. However, learning the barrier function is a data-intensive process (Kordabad et al., 2024; Salamati et al., 2024; Dai et al., 2023) and estimating the conditional expectation in the condition (iii) of the barrier function is difficult. To tackle these challenges, we use RKHS and CME to learn a valid barrier function. We show in this section that the CME transforms the problem of finding a valid barrier function into a linear program which can be solved efficiently. We also show that the barrier function learned via the approximated CME converges to the true barrier function as the size of the dataset increases (cf. Theorem 3).

The overall algorithm, called *kernel-based Safe Exploration* (*KBSE*), learns a controller online in a Deep-RL setting for the CMDP $\mathcal{M}$ with an unknown probability kernel $\mathcal{T}$. In the following subsections, we describe the main steps of the KBSE algorithm in detail, with the algorithm included in Appendix A.

4.1. Data collection

We use data generated during exploration, denoted as $\mathcal{R} = \{\langle \hat{s}_i, \hat{a}_i, \hat{r}_i, \hat{s}_i^+\rangle, i = 1, 2, \ldots, N\}$. We can also denote the dataset alternatively as $\mathcal{R} = (\hat{S}, \hat{A}, \hat{R}, \hat{S}^+)$, with

$$\hat{S} = [\hat{s}_1, \ldots, \hat{s}_N]^T \ , \ \hat{A} = [\hat{a}_1, \ldots, \hat{a}_N]^T \ , \ \hat{R} = [\hat{r}_1, \ldots, \hat{r}_N]^T \ , \ \hat{S^+} = [\hat{s}_1^+, \ldots, \hat{s}_N^+]^T \ , \tag{5}$$

where $\hat{r}_i = R(\hat{s}_i)$ and $s_i^+ \sim \mathcal{T}(\hat{s}_i, \hat{a}_i, \cdot)$ for all i. As an off-policy RL method that stores all the transitions collected during training in a replay buffer, we randomly sample from this buffer to generate independent and identically distributed (iid) samples.

For the CME and approximation of the expectation, we select the kernel k_S to be a radial basis function (RBF) of the form $k_S(s_i, s_j) = \exp\left[\frac{-\|s_i - s_j\|^2}{2\cdot\sigma^2}\right]$, for all $s_i, s_j \in S$, and similarly for the kernel k_{SA} of the product space $S \times A$. We define the corresponding *Gram matrices* $K_{\hat{S}}$ and K_+ as $K_{\hat{S}} := [k_S(\hat{s}_i, \hat{s}_j)]_{ij}$ and $K_+ := [k_S(\hat{s}_i^+, \hat{s}_j^+)]_{ij}$. We also define the vector-valued functions $k_{\hat{S}}(s) := [k_S(s, \hat{s}_i)]_i$ and $k_+(s) := [k_S(s, \hat{s}_i^+)]_i$. For the product space $S \times A$, the Gram matrix $K_{\hat{S}\hat{A}}$ and the function $k_{\hat{S}\hat{A}}(s,a)$ are defined similarly.

The function `sample_data` used in Algorithm 1 in Appendix A implements the generation of iid samples from $\mathcal{R}$.

4.2. Generating a Valid Barrier Function

To get a valid barrier function, we learn a barrier function from the data and perform a validity check iteratively until a valid barrier function is generated.

Learning barrier function. We define the barrier B as a linear combination of the RBF kernel functions, given as

$$B(s) = \sum_{i=1}^{N} \alpha_i \cdot k_s(s, s_i) \; . \tag{6}$$

Using the safety specification $\varphi = \langle S_u, T \rangle$, we classify each state $\hat{s}_i \in S$ using a binary safe/unsafe label Y. We use linear regression with regularization (Montgomery et al., 2006) to solve for α as

$$\min_{\alpha} \|\alpha \cdot K_{\hat{S}} - Y\|^2 + \lambda \|\alpha\|^2 \; . \tag{7}$$

We compute $\eta = B(s_0)$ and $\nu = \min_{s \in S_u} B(s)$. The function in (6) satisfies the conditions (i) and (ii) of being a barrier function if $\nu > \eta$.

Barrier function validation. In order to compute the constant c in the condition (iii) of the barrier function and validate the function in (6), we compute a data-driven empirical CME $\hat{\mu}(\cdot, s, a)$ for the CME μ in (3) applied to the probability kernel $\mathcal{T}$. Using data $\mathcal{R} = (\hat{S}, \hat{A}, \hat{R}, \hat{S}^+)$, the empirical CME (Grünewälder et al., 2012; Thorpe and Oishi, 2021) is

$$\hat{\mu}(\cdot, s, a) = k_{\hat{S}\hat{A}}(s, a)^T [K_{\hat{S}\hat{A}} + \lambda N \mathbb{I}_N]^{-1} k_+(\cdot) \; , \tag{8}$$

where $\lambda \geq 0$ is a regularization constant, $\mathbb{I}_N$ is the identity matrix with dimension N. Using the reproducing property, we have the following approximation for the conditional expectation in (4):

$$\langle B, \hat{\mu}(\cdot, s, a) \rangle_{\mathcal{H}_{k_S}} = k_{\hat{S}\hat{A}}(s, a)^T [K_{\hat{S}\hat{A}} + \lambda N \mathbb{I}_N]^{-1} B(\hat{S}^+) \; . \tag{9}$$

For simplicity, we denote

$$W(s, a)^T = k_{\hat{S}\hat{A}}(s, a)^T [K_{\hat{S}\hat{A}} + \lambda N \mathbb{I}_N]^{-1} \; . \tag{10}$$

The empirical CME $\hat{\mu}$ deviates from the true CME μ by at most ϵ in the $\mathcal{G}$-norm with probability $1 - \zeta$, where the approximation precision ϵ is an error bound to represent the Maximum Mean Discrepancy (MMD) (Nemmour et al., 2022) radius of the RKHS ambiguity set centered at the empirical CME. We therefore need the barrier condition to hold robustly over all CMEs within this ϵ-ball. We define an ambiguity set $\mathcal{C}_\epsilon$, with MMD ϵ, centered at the empirical CME $\hat{\mu}(\cdot, s, a)$ such that the true CME $\mu(\cdot, s, a)$ lies within the ambiguity set with probability at least $(1 - \zeta)$, i.e.,

$$\mathbb{P}(\mu(s, a) \in \mathcal{C}_\epsilon) \geq 1 - \zeta, \text{ where } \mathcal{C}_\epsilon = \{\mu \in \mathcal{G} | \parallel \mu - \hat{\mu} \parallel_{\mathcal{G}} \leq \epsilon\}. \tag{11}$$

Theorem 2 *Consider the dataset $\mathcal{R} = (\hat{S}, \hat{A}, \hat{R}, \hat{S}^+)$ with the kernels k_S and k_A and the Gram matrices $K_{\hat{S}}$, $K_{\hat{A}}$ and K_+. Consider the ambiguity set $\mathcal{C}_\epsilon$ centered at the empirical CME $\hat{\mu}$ in (8) with confidence $(1 - \zeta)$. If there exists a function $B : S \to \mathbb{R}_{\geq 0}$, $B \in \mathcal{H}_{k_S}$ with $\bar{B} \geq \|B\|_{\mathcal{H}_{k_S}}$, such that for all $s \in S$ there is an $a \in A$ satisfying*

$$W(s, a)^T \cdot B(\hat{S}^+) - B(s) \leq c - \epsilon \cdot \sqrt{k_{SA}((s, a), (s, a))} \cdot \bar{B}, \tag{12}$$

for some $c \geq 0$ with $W(s, a)$ in (10), then B satisfies the condition (iii) of the barrier function with constant c and confidence $(1 - \zeta)$.

Proof Refer Appendix B. ■

To compute the constant c for the barrier function B, we convert the constraint in (12) into a min-max optimization problem, as follows:

$$c = \min_{s \in S} \max_{a \in A} \left[W(s,a)^T \cdot B(\hat{S}^+) - B(s) + \epsilon \cdot \sqrt{k_{SA}((s,a),(s,a))} \cdot \bar{B} \right]. \tag{13}$$

The function `compute_BC` in Algorithm 1 in Appendix A generates a valid barrier function.

4.3. Finding the Safe Actions

Once we compute a valid barrier function, we use the inequalities of the barrier function to identify and reduce the number of safety violations. To handle these safety violations and guide locally the reinforcement learning towards safe states, we need the local dynamics of the system. We learn a local linear dynamics of the CMDP, represented by matrices P and Q, using state transition data $\langle \hat{s}_t, \hat{a}_t, \hat{s}_{t+1} \rangle$. We use regression and solve the following optimization problem (Montgomery et al., 2006):

$$\min_{P,Q} \sum_{i=t-H}^{t-1} \|\hat{s}_{i+1} - P \cdot \hat{s}_i - Q \cdot \hat{a}_i\|_2^2 , \tag{14}$$

where H denotes the number of local transitions used to learn the matrices P and Q.

Assuming safety specification violation at the state s_t, we solve a quadratic optimization problem that modifies the current action a_t to find the action $\bar{a}$ closest to a_t leading to a safe state s_{t+1}, given as

$$\bar{a}_t = \arg\min_{\bar{a}} \|\bar{a} - a_t\|^2 \qquad \text{s.t.} \quad B(s_{t+1}) \leq \nu, \quad s_{t+1} = P \cdot s_t + Q \cdot \bar{a} \, . \tag{15}$$

The above optimization ensures that we find the safe action $\bar{a}_t$ close to the action a_t generated by the control policy. This ensures that the learning of the control policy π_θ by the reinforcement learning algorithm remains stable.

The function `get_local_dynamics` in step 12 of Algorithm 1 in Appendix A solves the Equation (14). The function `get_safe_action` in line 13 of Algorithm 1 in Appendix A solves the Equation (15).

4.4. Properties of KBSE

With more exploration, the estimate of the conditional expectation in condition (iii) of the barrier function improves. In the following theorem, we provide a bound on the convergence of the approximation of the conditional expectation in the barrier function to the true expectation as the sample size increases.

Theorem 3 *For a given confidence parameter ζ, the approximation precision ϵ, representing the MMD radius, decreases as the number of samples increase with the upper bound $\sqrt{\frac{C}{N}}(1 + \sqrt{2 \cdot \texttt{log}(\frac{1}{\zeta})})$, where C is a constant such that $\texttt{sup}(k_S(s,s)) \leq C \leq \infty$ and N is the number of samples. This signifies the convergence of empirical CME $\hat{\mu}$ to the true CME μ with probability at least $(1-\zeta)$. Moreover, for a fixed ϵ, the confidence in the approximation improves with the increase in the sample size N according to $\zeta \leq \texttt{exp}\left[-\frac{1}{2}\left(\frac{\epsilon \cdot \sqrt{N}}{\sqrt{C}} - 1\right)^2\right]$.*

Env.	p	q	Safety specification	Training Time (s)					# Safety	90^{th}	Safety
				DLag	SLag	DPID	SD	KBSE	Viol.	$\%ile$	Prob.
SafetyPendulum	3	1	$(\vartheta > -0.8)$	406	704	423	718	835	4211	81.60	0.643
SafetyMCar	2	1	$(p > -1.0)$	443	682	486	662	683	342	82.14	0.572
SafetyInvPendulum	4	1	$(\|p\| < 0.3)$	2970	3395	2924	3362	3592	24	7.05	0.972
SafetyHopper	11	3	$((z > 0.8) \wedge (\|a\| < 0.2))$	9577	10492	9312	10520	12054	5250	68.66	0.958
SafetyWalker	17	6	$((z > 0.9) \wedge (\|a\| < 1))$	9391	10682	9458	10508	14196	7289	27.33	0.915
SafetyAnt	27	8	$(z > 0.25)$	9438	10472	9637	10608	24208	105113	52.60	0.729
SafetyHumanoid	376	17	$(z > 1.05)$	15838	17585	15791	17848	30452	923	20.60	0.872

Table 1: State dimensions (p), action dimensions (q), Safety specifications, computation time and number of safety violations for different environments (Env). For safety specifications p: position of the robot, z: height of the center of mass, a: pedal angle.

Env.	Algo.	Avg. Reward	Avg. Cost	Avg. Length	Env.	Algo.	Avg. Reward	Avg. Cost	Avg. Length
Safety Pendulum	DDPGLag	−143.62	6.8	200	Safety MountainCar	DDPGLag	−0.06	0.0	1000.0
	SACLag	−120.15	4.2	200		SACLag	93.38	12.0	137.2
	DDPGPID	−145.75	6.4	200		DDPGID	−0.01	0.0	1000.0
	SACPID	−162.89	8.6	200		SACPID	93.96	12.8	144.4
	KBSE	−164.68	7.8	200		KBSE	93.04	18.2	150.4
Safety Inverted-Pendulum	DDPGLag	1000.0	6.6	1000.0	Safety Hopper	DDPGLag	1204.37	3.4	404.0
	SACLag	1000.0	0.0	1000.0		SACLag	2991.61	0.0	1000.0
	DDPGID	1000.0	6.6	1000.0		DDPGPID	2110.11	0.8	613.4
	SACPID	1000.0	0.0	1000.0		SACPID	1035.41	0.0	1000.0
	KBSE	1000.0	0.0	1000.0		KBSE	3203.76	0.0	1000.0
Safety Walker	DDPGLag	2042.48	21.2	612.4	Safety Ant	DDPGLag	257.69	164.0	252.4
	SACLag	1734.94	9.0	558.60		SACLag	−283.28	411.8	867.0
	DDPGID	935.50	7.6	322.2		DDPGID	1403.84	50.0	748.4
	SACPID	450.34	83.8	577.2		SACPID	−1257.27	315.8	1000.0
	KBSE	3819.62	0.0	1000.0		KBSE	5819.89	0.0	1000.0
Safety Humanoid	DDPGLag	1362.88	5.6	275.8					
	SACLag	5469.56	0.0	1000.0					
	DDPGID	2217.06	5.2	432.0					
	SACPID	5169.28	0.0	1000.0					
	KBSE	5483.47	0.0	1000.0					

Table 2: Comparing the average reward, cost, and length of the KBSE w.r.t baseline algorithms for all benchmarks.

Proof Refer Appendix B. ■

5. Evaluation

We now describe the setup and results for our experiments.

Experimental setup. All experiments are carried out on an Ubuntu22.04 machine with Intel(R) Xeon(R) Gold 6226R @2.90 GHz×16 CPU, NVIDIA RTX A4000 32GB Graphics card, and 48 GB RAM using the OMNISAFE (Ji et al., 2024) framework.

Baseline for comparison. Since KBSE belongs to the class of off-policy safe RL algorithms, we consider the most popular off-policy safe RL algorithms as baseline for comparison, the Lagrangian and PID-Lagrangian approaches, i.e., DDPGLagrangian, SACLagrangian (Ray et al., 2019) and DDPGPID, SACPID (Stooke et al., 2020). We have not reported the comparison results with other

safe RL algorithms such as CPO (Achiam et al., 2017), PPO-Lag (Ray et al., 2019), or CUP (Yang et al., 2022) as these are *on-policy* algorithms and are known to suffer from high variance and slow convergence compared to the off-policy methods (Chung et al., 2021).

Benchmarks. Our experiments include classical and Gym-Mujoco benchmarks (Brockman et al., 2016) involving safety constraints. See Table 1 with the details of the benchmarks described in the supplementary material and the hyper-parameters in Appendix E.

5.1. Results

Training. The training time needed to learn the control policy and the number of safety violations encountered during training for the KBSE algorithm are presented in Table 1. The column $90^{th}\%ile$ provides the time-step by which 90% of the total number of safety violations have occurred for the KBSE algorithm, as a percentage of the training horizon. This indicates that the number of safety violations decreases significantly in the later stages of training. Given that the baselines do not permit quantitative safety violations in their problem formulation, the metrics on safety violations (e.g., 90th percentile) are not applicable in these cases.

We also observe that the training time for KBSE algorithm increases with increasing number of safety violations. This is most apparent in case of the SafetyAnt benchmark where the training time is twice than that of the baselines. The training time depends on the number of safety violations encountered during exploration (cf. Appendix D.3). This, in turn, depends on the shape of the barrier function - one with narrow "safe set" (i.e. stricter safety specifications) would cause more violations. The last column of Table 1 provides the lower bound on the safety probability guaranteed by the KBSE algorithm for each benchmark, a feature that is not integrated in the available safe RL algorithms.

Testing. Table 2 compares the control policies learned by the KBSE algorithm w.r.t. the baselines. For testing policies, we use the metrics *Average Episodic Reward*, *Average Episodic Cost*, and *Average Episodic Length*. In general, the results demonstrate that the KBSE algorithm achieves higher reward and lower cost compared to the baselines. In benchmarks with high dimension, the KBSE algorithm shows superior performance that can be attributed to the larger exploration space for these high-dimensional benchmarks, providing a significant margin for improvement. On the other hand, in low-dimensional benchmarks such as SafetyPendulum and Safety-MountainCar, the performance of KBSE is similar to the baselines as the training occurs for a shorter duration and hence the margin of improvement is smaller.

Further experimental results are available in the appendix, showing synthesized barrier functions (Appendix D.1), plots for ϵ and $\bar{B}$ (Appendix D.2), the relationship between training time and safety violations (Appendix D.3).

6. Conclusion

We have proposed a novel safe exploration algorithm using online barrier functions constructed using kernel mean embeddings (CME). Our approach does not require knowledge of the system dynamics and uses barrier functions that involve chance constraints that allow the violation of the safety specification up to a given probability threshold. Future work includes enhancing the performance using sparse CME and considering temporal behaviors beyond safety.

7. Acknowledgments

This work was supported by the European Research Council (ERC, grant 101089047), the European Innovation Council (EIC, grant 101070802), and the Deutsche Forschungsgemeinschaft (DFG, project 389792660, TRR 248—CPEC).

References

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 22–31. PMLR, 06–11 Aug 2017.

Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. *Proceedings of the AAAI Conference on Artificial Intelligence*, Apr. 2018.

André M.S. Barreto, Doina Precup, and Joelle Pineau. Practical kernel-based reinforcement learning. *Journal of Machine Learning Research*, 17(67):1–70, 2016.

Alain Berlinet and Christine Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic, Boston, 2004.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai Gym. *CoRR*, 2016.

Richard Cheng, Gábor Orosz, Richard M. Murray, and Joel W. Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference*, AAAI'19, 2019.

Wesley Chung, Valentin Thomas, Marlos C. Machado, and Nicolas Le Roux. Beyond variance reduction: Understanding the true impact of baselines on policy optimization. In *Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 1999–2009. PMLR, 18–24 Jul 2021.

Efrén Cruz Cortés and Clayton Scott. Sparse approximation of a kernel mean. *IEEE Transactions on Signal Processing*, 65(5):1310–1323, 2017.

Bolun Dai, Heming Huang, Prashanth Krishnamurthy, and Farshad Khorrami. Data-efficient control barrier function refinement. In *American Control Conference, ACC 2023*, pages 3675–3680, 2023.

Omar Darwiche Domingues, Pierre Menard, Matteo Pirotta, Emilie Kaufmann, and Michal Valko. Kernel-based reinforcement learning: A finite-time analysis. In *Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 2783–2792. PMLR, 18–24 Jul 2021.

D. K. Duvenaud. *Automatic Model Construction with Gaussian Processes*. PhD thesis, University of Cambridge, 2014.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012.

S. Grünewälder, G. Lever, L. Baldassarre, S. Patterson, A. Gretton, and M. Pontil. Conditional mean embeddings as regressors. In *Proceedings of the 29th International Conference on Machine Learning*, pages 1823–1830, New York, NY, USA, 2012. Omnipress.

Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola. Kernel methods in machine learning. *Annals of Statistics*, 36(3):1171–1220, 2008.

Boya Hou, Sina Sanjari, Nathan Dahlin, and Subhonmesh Bose. Compressed decentralized learning of conditional mean embedding operators in reproducing kernel hilbert spaces. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence*, AAAI'23, 2023.

Jiaming Ji, Jiayi Zhou, Borong Zhang, Juntao Dai, Xuehai Pan, Ruiyang Sun, Weidong Huang, Yiran Geng, Mickel Liu, and Yaodong Yang. Omnisafe: An infrastructure for accelerating safe reinforcement learning research. *Journal of Machine Learning Research*, 25(285):1–6, 2024.

Ilja Klebanov, Ingmar Schuster, and T. J. Sullivan. A rigorous theory of conditional mean embeddings. *SIAM Journal on Mathematics of Data Science*, 2(3):583–606, 2020.

Arash Bahari Kordabad, Maria Charitidou, Dimos V Dimarogonas, and Sadegh Soudjani. Control barrier functions for stochastic systems under signal temporal logic tasks. In *2024 European Control Conference (ECC)*, pages 3213–3219. IEEE, 2024.

Harold Joseph Kushner. Stochastic stability and control. *Academic Press, Inc.*, 1967.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.*, 17(1):1334–1373, 2016.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *CoRR*, 2020.

Xiao Li and Calin Belta. Temporal logic guided safe reinforcement learning using control barrier functions. In *arXiv*, 2019.

Zhu Li, Dimitri Meunier, Mattes Mollenhauer, and Arthur Gretton. Optimal learning rates for regularized conditional mean embedding. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NeurIPS '22, Red Hook, NY, USA, 2022.

Yongshuai Liu, Jiaxin Ding, and Xin Liu. IPO: Interior-point policy optimization under constraints. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):4940–4947, Apr. 2020.

Giacomo Meanti, Luigi Carratino, Lorenzo Rosasco, and Alessandro Rudi. Kernel methods through the roof: Handling billions of points efficiently. In *Advances in Neural Information Processing Systems*, volume 33, pages 14410–14422. NeurIPS, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen,

Charlie Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

Douglas C. Montgomery, Elizabeth A. Peck, and Geoffrey G. Vining. *Introduction to Linear Regression Analysis (4th ed.)*. Wiley & Sons, 2006.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models. *ACM Trans. Intell. Syst. Technol.*, 16(5), August 2025.

Yassine Nemmour, Heiner Kremer, Bernhard Schölkopf, and Jia-Jie Zhu. Maximum mean discrepancy distributionally robust nonlinear chance-constrained optimization with finite-sample guarantee. In *IEEE 61st Conference on Decision and Control (CDC)*, pages 5660–5667, 2022.

Dirk Ormoneit and Saunak Sen. Kernel-based reinforcement learning. *Machine Learning*, 49(2-3): 161–178, 2002.

Junhyung Park and Krikamol Muandet. A measure-theoretic approach to kernel conditional mean embeddings. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 21247–21259, 2020.

S. Prajna. Barrier certificates for nonlinear model validation. *Automatica*, 42(1):117–126, 2006.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. In *arXiv*, 2019.

Robert Reed and Morteza Lahijanian. Learning-based shielding for safe autonomy under unknown dynamics. In *arXiv*, 2024.

Licio Romao, Ashish R. Hota, and Alessandro Abate. Distributionally robust optimal and safe control of stochastic systems via kernel conditional mean embedding. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 2016–2021, 2023.

Ali Salamati, Abolfazl Lavaei, Sadegh Soudjani, and Majid Zamani. Data-driven verification and synthesis of stochastic systems via barrier certificates. *Automatica*, 159:111323, 2024.

Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

Oliver Schön, Zhengang Zhong, and Sadegh Soudjani. Data-driven distributionally robust safety verification using barrier certificates and conditional mean embeddings. In *2024 American Control Conference (ACC)*, pages 3417–3423, 2024.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy P. Lillicrap, Fan Hui, L. Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 2017.

Le Song, Jonathan Huang, Alex Smola, and Kenji Fukumizu. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 961–968, New York, NY, USA, 2009.

Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Springer Publishing Company, 1st edition, 2008.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by pid lagrangian methods. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR, 2020.

Adam J. Thorpe and Meeko M. K. Oishi. Stochastic optimal control via Hilbert space embeddings of distributions. In *2021 60th IEEE Conference on Decision and Control (CDC)*, page 904–911. IEEE Press, 2021.

Adam J. Thorpe, Jake A. Gonzales, and Meeko M. K. Oishi. Data-driven stochastic optimal control using kernel gradients. In *American Control Conference, ACC 2023, San Diego, CA, USA, May 31 - June 2, 2023*, pages 2548–2553. IEEE Press, 2023.

Yixuan Wang, Simon Sinong Zhan, Ruochen Jiao, Zhilu Wang, Wanxin Jin, Zhuoran Yang, Zhaoran Wang, Chao Huang, and Qi Zhu. Enforcing hard constraints with soft barriers: safe reinforcement learning in unknown stochastic environments. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR, 2023.

Long Yang, Jiaming Ji, Juntao Dai, Linrui Zhang, Binbin Zhou, Pengfei Li, Yaodong Yang, and Gang Pan. Constrained update projection approach to safe policy optimization. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NeurIPS '22, Red Hook, NY, USA, 2022.

Yujie Yang, Yuxuan Jiang, Yichen Liu, Jianyu Chen, and Shengbo Eben Li. Model-free safe reinforcement learning through neural barrier certificate. *IEEE Robotics and Automation Letters*, 8 (3):1295–1302, 2023.

Appendix

Appendix A. KBSE Algorithm

A.1. Algorithm

The overall algorithm, called *kernel-based Safe Exploration* (*KBSE*) and presented in Algorithm 1, learns a controller online in a Deep-RL setting for the CMDP $\mathcal{M}$ with an unknown probability kernel $\mathcal{T}$.

The initialization for the algorithm is performed in lines 3-6. The dataset $\mathcal{R}$, the initial state s_0, the bound on the norm of the barrier function $\bar{B}$ and the approximation precision ϵ are initialized in line 3. A random policy is instantiated in line 4. Under this policy, a collection of sampled transitions of the form $\langle s_t, a_t, r_t, s_{t+1} \rangle$ are collected in w in line 5, and a valid barrier function together with its associated constants (ν, η, c) are computed in line 6. The sampled transitions w are added to the dataset $\mathcal{R}$ in line 7.

The lines 8-28 describe the main loop of the algorithm with the training horizon T_h. In line 10, we check for a safety violation at state s_t. In case of safety violation, the algorithm invokes the function `get_local_dynamics` in line 12 to obtain the local dynamics of the system and use it to find the closest safe action using the function `get_safe_action` in line 13. In line 16, the system is simulated to generate a transition sample and appends it to the dataset $\mathcal{R}$ (line 17). In line 18, a batch of data samples from $\mathcal{R}$ is used to update the policy, similar to the policy update in offline reinforcement learning (Levine et al., 2020).

The lines 20-22 are executed after each episode to update the approximation error of the condition (iii) of the barrier function. This involves recomputing the upper bound of RKHS norm of the barrier function $\bar{B}$ (line 21) and the approximation precision ϵ (line 22) using the updated dataset (line 20) and confidence parameter ζ. The lines 25-26 are executed at the end of each epoch, which is over a time horizon longer than the episode length. An epoch contains a number of episodes. The condition `epoch ends` is used to compute the barrier function and `episode ends` is used to update the barrier function parameters. The function `compute_BC` in line 26 involves learning the barrier function over the latest dataset from line 25 and performing a validity check iteratively until a valid barrier function is obtained. Although the iterative check entails an overhead, but it is much less expensive and more scalable than computing the barrier function a priori or offline. The parameters of the learned barrier function will give the bound on the safety probability $(1-\delta)$ using the line 30.

Appendix B. Proofs of Theorems

Theorem 2 *Consider the dataset $\mathcal{R} = (\hat{S}, \hat{A}, \hat{R}, \hat{S}^+)$ with the kernels k_S and k_A and the Gram matrices $K_{\hat{S}}$, $K_{\hat{A}}$ and K_+. Consider the ambiguity set $\mathcal{C}_\epsilon$ centered at the empirical CME $\hat{\mu}$ in (8) with confidence $(1-\zeta)$. If there exists a function $B : S \to \mathbb{R}_{\geq 0}$, $B \in \mathcal{H}_{k_S}$ with $\bar{B} \geq \|B\|_{\mathcal{H}_{k_S}}$, such that for all $s \in S$ there is an $a \in A$ satisfying*

$$W(s,a)^T \cdot B(\hat{S}^+) - B(s) \leq c - \epsilon \cdot \sqrt{k_{SA}((s,a),(s,a))} \cdot \bar{B}, \tag{1}$$

for some $c \geq 0$ with $W(s,a)$ in (10), then B satisfies the condition (iii) of the barrier function with constant c and confidence $(1-\zeta)$.

Algorithm 1: KBSE algorithm for CMDP

procedure `main()`
  Input: CMDP $\mathcal{M} = \langle S, A, \mathcal{T}, R\rangle$ with unknown $\mathcal{T}$, safety specification $\varphi = \langle S_u, T\rangle$, parameterized policy π_θ, training horizon T_h, confidence parameter $\zeta \in (0,1)$ for the CME
  $\mathcal{R} \leftarrow []$; $s_0 \leftarrow \rho_0$; $\bar{B} \leftarrow$ `initialize()`; $\epsilon \leftarrow$ `initialize`(ζ);
  $\pi_\theta \leftarrow$ `initialize_policy()`
  $w \leftarrow$ `initial_rollout`(π_θ)
  $(B, \nu, \eta, c) \leftarrow$ `compute_BC`$(w, \epsilon, \bar{B})$
  $\mathcal{R} \leftarrow \mathcal{R} \cup w$
  while $t < T_h$ **do**
    $a_t \leftarrow \pi_\theta(s_t)$
    if $s_t \in S_u$ **then**
      /* Safety is violated */
      $P, Q \leftarrow$ `get_local_dynamics`$(\mathcal{R})$
      $\bar{a}_t \leftarrow$ `get_safe_action`(B, s_t, a_t, P, Q)
      $a_t \leftarrow \bar{a}_t$
    end
    s_{t+1} sampled from $\mathcal{T}(s_t, a_t, \cdot)$, $r_t = R(s_t)$
    $\mathcal{R} \leftarrow \mathcal{R} \cup \{\langle s_t, a_t, r_t, s_{t+1}\rangle\}$
    `update_policy`$(\pi_\theta, \mathcal{R})$
    if *episode ends* **then**
      $w \leftarrow$ `sample_data`$(\mathcal{R})$
      $\bar{B} \leftarrow$ `update_upper_bound` (B, w)
      $\epsilon \leftarrow$ `update_epsilon` (w, ζ)
    end
    if *epoch ends* **then**
      $w \leftarrow$ `sample_data`$(\mathcal{R})$
      $(B, \nu, \eta, c) \leftarrow$ `compute_BC` $(w, \epsilon, \bar{B})$
    end
    $t \leftarrow t + 1$
  end
  $\delta \leftarrow (\eta + cT)/\nu$
  return *Policy* π_θ *and barrier function* B *with safety probability* $(1-\delta)$ *guaranteed with confidence* $(1-\zeta)$

Proof The inner product of the barrier function B and CME $\mu(\cdot, s, a)$ in $\mathcal{H}_{k_S}$ can be expressed as

$$\langle B, \mu(\cdot, s, a)\rangle_{\mathcal{H}_{k_S}} = \langle B, \mu(\cdot, s, a) - \hat{\mu}(\cdot, s, a)\rangle_{\mathcal{H}_{k_S}} + \langle B, \hat{\mu}(\cdot, s, a)\rangle_{\mathcal{H}_{k_S}}.$$

The second term gives us $W(s,a)^T B(\hat{S}^+)$ from Equation (9). To bound the first term, let $\Gamma : (S,A) \times (S,A) \to \mathcal{L}(\mathcal{H}_{k_S})$ be the operator-valued positive definite kernel of $\mathcal{G}$ (cf. Remark 3 and Li et al. (2022)), given by $\Gamma((s,a),(s',a')) := k_{SA}((s,a),(s',a')) \cdot Id_{\mathcal{H}_{k_S}}$, where $\mathcal{L}(\mathcal{H}_{k_S})$ is the banach space of bounded linear operators from $\mathcal{H}_{k_S}$ to $\mathcal{H}_{k_S}$ and $Id_{\mathcal{H}_{k_S}}$ is the identity operator

on $\mathcal{H}_{k_S}$. Using the reproducing property of Γ (cf. Li et al. (2022)), we get

$$\begin{aligned}\langle B, \mu - \hat{\mu}\rangle_{\mathcal{H}_{k_S}} &= \langle \Gamma(\cdot,(s,a))\cdot B, \mu - \hat{\mu}\rangle_{\mathcal{G}} \leq \epsilon \|\Gamma(\cdot,(s,a))\cdot B\|_{\mathcal{G}} \\ &= \epsilon\sqrt{\langle \Gamma(\cdot,(s,a))\cdot B, \Gamma(\cdot,(s,a))\cdot B\rangle_{\mathcal{G}}} \\ &= \epsilon\sqrt{\langle B, \Gamma((s,a),(s,a))\cdot B\rangle_{\mathcal{H}_{k_S}}} \\ &= \epsilon\bar{B}\sqrt{\|\Gamma((s,a),(s,a))\|} = \epsilon\bar{B}\sqrt{k_{SA}((s,a),(s,a))}.\end{aligned}$$

Combining the two terms concludes the proof. ■

Theorem 3 *For a given confidence parameter ζ, the approximation precision ϵ, representing the MMD radius, decreases as the number of samples increase with the upper bound $\sqrt{\frac{C}{N}}\left(1+\sqrt{2\cdot \texttt{log}(\frac{1}{\zeta})}\right)$, where C is a constant such that $\texttt{sup}(k_S(s,s)) \leq C \leq \infty$ and N is the number of samples. This signifies the convergence of empirical CME $\hat{\mu}$ to the true CME μ with probability at least $(1-\zeta)$. Moreover, for a fixed ϵ, the confidence in the approximation improves with the increase in the sample size N according to $\zeta \leq \texttt{exp}\left[-\frac{1}{2}\left(\frac{\epsilon\cdot\sqrt{N}}{\sqrt{C}}-1\right)^2\right]$.*

Proof From (Nemmour et al., 2022, Eqn. 4), the error bound of the MMD estimators certify that the true CME μ is contained in an MMD ball of radius ϵ around the empirical CME $\hat{\mu}$ with probability $(1-\zeta)$, where

$$\epsilon \leq \sqrt{\frac{C}{N}} + \sqrt{\frac{2\cdot C\cdot \texttt{log}(\frac{1}{\zeta})}{N}} = \sqrt{\frac{C}{N}}\left(1+\sqrt{2\cdot \texttt{log}(\frac{1}{\zeta})}\right) .$$

Rearranging the terms in the above inequality gives the reported bound for ζ. ■

Appendix C. Benchmarks

Pendulum. For the pendulum example, the safety specification requires that the pendulum never goes for a full swing to achive its goal (cf. Section 3).

MountainCarContinuous. In Mountain Car example, position $p \in [-1.2, 0.6]$, the safety constraint requires that the car does not go to the extreme left while gaining momentum to climb the hill, i.e. $p > -1.0$. Thus, the safety specification enforces a reduction on the maximum momentum that the car could use to reach the goal.

Inverted Pendulum. For the inverted pendulum, the safety specification requires that the inverted pendulum never goes too far in the horizontal direction, i.e. outside [-0.3,0.3].

Hopper. For Hopper environment, the safety specification requires that the height of its center of mass should always be greater than 0.8 meters and its pedal angle should always be in range $[-0.2, 0.2]$ radians. We construct a multi-dimensional safety specification defined as $\{z > 0.8 \wedge |a| < 0.2\}$.

Walker. For Walker environment, the safety specification requires that the height of its center of mass should always be greater than 0.9 meters and its pedal angle should always be in range

$[-1.0, 1.0]$ radians. We construct a multi-dimensional safety barrier defined as $\{z > 0.9 \wedge |a| < 1.0\}$.

Ant. The Ant environment belongs to the quadruped class with the safety specification requiring that the height of its center of mass should always be greater than 0.25 meters above the ground. This is challenging because the ant can achieve higher speed by bending its legs more (unsafe state), resulting in lower height of center of mass.

Humanoid. The humanoid is the most complex benchmark used in our experiments. The safety specification requires that the height of the center of mass is always above 1.1 meters above the ground.

Appendix D. Additional Experiments

D.1. Visualizing synthesized barriers

Figure 2 shows an intermediate barrier synthesized during the later stage of exploration via the KBSE algorithm. The barrier is one-dimensional (ϑ) for the Pendulum and two-dimensional (z, a) for the Hopper environment. The regions in red and blue correspond to large and small values for the barrier function, respectively. The yellow curve represents the contour for $B(s_t) = \nu$. In case of Pendulum (left), the contour is a line around $\vartheta = -0.8$ which corresponds to the boundary of the safety specification. This indicates that the barrier function successfully captures the safety for the Pendulum environment. Similar argument can be made for the Hopper environment (right).

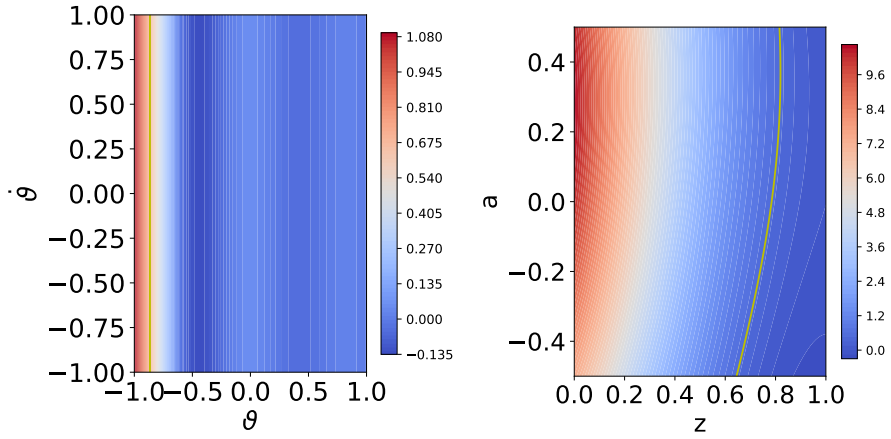

Figure 2: Plots showing an intermediate barrier synthesized for SafetyPendulum-v1 (left) and SafetyHopper-v1 (right) during the exploration phase for the KBSE algorithm..

D.2. Plots for ϵ and $\bar{B}$

Figure 3 shows the values of ϵ and $\bar{B}$ during the exploration via the KBSE algorithm for the Hopper environment. We observe that ϵ and $\bar{B}$ decrease with time, leading to a reduction in the approximation error introduced due to the empirical CME.

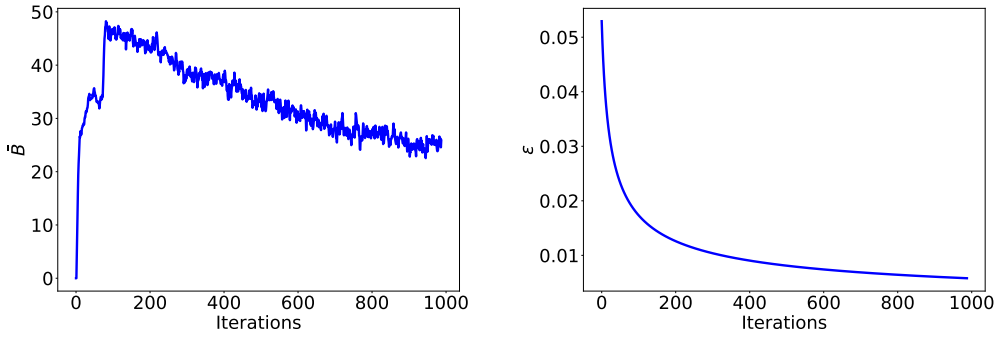

Figure 3: Values of $\bar{B}$ in the left and ϵ in the right during the exploration phase for the Hopper environment.

D.3. Training Time and Safety Violations

The training time depends on the number of safety violations encountered during exploration. Hence, relaxing the safety specification reduces training time. To demonstrate this, Table 3 reports the training time and the number of safety violations for different safety specifications in the SafetyHopper benchmark.

Safety Specification	Training Time	Safety Violations
$z > 0.8 \wedge \lvert a \rvert < 0.2$	20516.26	3745
$z > 0.75 \wedge \lvert a \rvert < 0.2$	17901.66	2155
$z > 0.70 \wedge \lvert a \rvert < 0.2$	14107.36	0

Table 3: Comparison of the training time and the number of safety violations for the SafetyHopper benchmark.

Appendix E. Hyper-parameters for KBSE Algorithm

In our experiments, we have used the same configuration as used by the OMNISAFE framework (Ji et al., 2024). We have used $H = 500$ in (14) to learn the local linear dynamics in order to find the closest safe action. The value of T is equal to the episode length for the corresponding environment. The values for Epoch and Barrier Size (N) is shown in Appendix E. For kernel bandwidth, we followed the same approach as described in Duvenaud (2014). The selected values for the hyper-parameters provide a good balance between accuracy and computational cost.

The term $\epsilon \cdot \sqrt{k_{SA}((s,a),(s,a))} \cdot \bar{B}$ in Equations (12)–(13) represents the conservatism introduced due to the worst-case analysis with respect to the unknown dynamics $\mathcal{T}$. The function `update_upper_bound` in Algorithm 1 computes the upper bound of B. This is given by the RKHS norm of B, i.e., $\|B\|_{\mathcal{H}_{k_S}} = \sqrt{\alpha^T \cdot K_{\hat{S}\hat{A}} \cdot \alpha}$, where $\alpha = [K_{\hat{S}\hat{A}} + \lambda N \mathbb{I}_N]^{-1} \cdot [B(s_i)]_{i=1}^N$ (Romao et al., 2023). For the KBSE algorithm, we have used the RBF kernel with $C = 1$ and assume $\zeta = 10^{-5}$, which means the function `update_epsilon` in Algorithm 1, updates the MMD radius ϵ at the rate of $\sqrt{\frac{1}{N}}\left(1 + \sqrt{10}\right)$.

For KBSE, the values of the hyper-parameters `Epoch`, `Barrier Size` (N), and the total number of training time-steps are shown in Table 4.

Env	Epoch	N	Training steps
SafetyPendulum	10000	500	50000
SafetyMountainCar	10000	500	50000
SafetyInvPendulum	10000	2000	200000
SafetyHopper	10000	2000	600000
SafetyWalker2d	10000	2000	600000
SafetyAnt	10000	2000	600000
SafetyHumanoid	50000	2000	1000000

Table 4: Hyper-parameters used in the KBSE algorithm.

Appendix F. Additional Discussions

Limitations. A general limitation of kernel-based methods is scalability and high memory requirements. Although the computational requirements of our algorithm increase as the dimension of state and action space increase, the increase in memory requirement is negligible. This is because the number of data samples used to construct the barrier function online is fixed (the sample set size grows with more data, thereby improving the barrier estimate). The computational overhead for KBSE comes from the quadratic optimization problem for `get_safe_action`. While our approach can handle benchmarks similar to SafetyHumanoid that has hundreds of states (cf. Table 1), scalability can be further improved by employing sparse CME (e.g., using the work by Cortés and Scott (2017)), which we plan to work on in the future.

Safety-performance tradeoff. Under the proposed KBSE algorithm, exploration can be guided away from highly rewarding but unsafe areas of the state space towards safer regions. The confidence level parameter ζ in the barrier formulation can be utilized to balance the trade-off between safety and performance.

Implications of Assumption in Remark 3. This is an assumption required for the theory of CME to hold and can be verified by having additional information on the class of system dynamics and the distribution of the noise. Under this additional information, one would need to check that the CME of the system dynamics belongs to the chosen Hilbert space. This restricts the conditional distribution to be "regular enough" so that it does not concentrate mass in a pathological way or have heavy tails. We refer to Park and Muandet (2020) for the technical details and remark that the requirements such as bounded kernel moments and RKHS norm integrability hold when selecting the kernel to be Gaussian or Radial Basis Functions (RBFs), as employed in our experiments.

Safety violation as indicator of safe exploration. Our problem formulation allows violation of safety up to a certain probability threshold even for the policy learned at the end of the exploration phase. We use safe exploration in the sense of satisfying the constraints characterizing the barrier function. The Safety violations during training occur due to the constraint optimization problem being infeasible. However, the training under the KBSE algorithm ensures that the collected data

samples satisfy the constraints, and thus the intermediate policies learned using these data samples would gradually satisfy these constraints.

Online vs. Off-policy. In this paper, the term "online" refers to the computation of the barrier function which is synthesized on-the-fly, i.e., during exploration in RL. The "off-policy" term refers to the RL based learning paradigm where policy is updated using the older (off-policy) data collected during exploration since the beginning.

Bias-Variance decomposition of error rates. As per Theorem 3, the bias decreases as the number of samples increases. The i.i.d. generation of data ensures that variance is bounded. A large value of the RBF kernel bandwidth makes the model less complex which may reduce variance but increase bias.

Assumption on bounds of the RKHS norm of B. The assumption of a bounded RKHS norm for the barrier function is a standard regularity condition commonly used in kernel-based learning methods to derive finite-sample generalization bounds (Li et al., 2022; Hou et al., 2023), and should be interpreted as such rather than as a claim that an arbitrarily complex safety boundary can always be captured. If the chosen kernel is insufficiently expressive, the theoretical safety guarantee holds with respect to the best approximation of the true dynamics and barrier within the RKHS, reflecting the standard trade-off between model expressiveness and statistical guarantees.

Solution to Problem 1. KBSE does not solve Problem 1 in the sense of finding the optimal policy. Instead, it produces a policy π_θ accompanied by a barrier B such that Theorem 1 certifies $\mathbb{P}\{\text{some state is in } S_u\} \leq \delta$ with confidence $1-\zeta$, where $\delta = \frac{\eta+cT}{\nu}$. The optimality of π_θ with respect to the reward objective under this safety constraint is not guaranteed; nevertheless, the learned policy is probabilistically safe with confidence $1-\zeta$ and accumulates competitive empirical reward across all benchmarks (Table 2).

Motivation for selecting the RBF kernel. We choose the RBF kernel for three reasons. First, it is a universal kernel on compact domains, ensuring that any continuous barrier function can be approximated within $\mathcal{H}_{k_S}$ to arbitrary precision. Second, it is a characteristic kernel, which guarantees that the MMD is a proper metric on distributions and that the ambiguity set $\mathcal{C}_\epsilon$ in Eq. (11) correctly captures uncertainty about the true CME. Third, it satisfies $\sup_s k_S(s,s) = 1$, which gives the tightest possible constant C in the convergence bound of Theorem 3, directly controlling the quality of the safety certificate.