SReachTools Kernel Module: Data-Driven Stochastic Reachability Using Hilbert Space Embeddings of Distributions

Anonymous Author(s)

ABSTRACT

We present algorithms for performing data-driven stochastic reachability as an addition to `SReachTools`, an open-source stochastic reachability toolbox. Our method leverages a class of machine learning techniques known as kernel embeddings of distributions to approximate the safety probabilities for a wide variety of stochastic reachability problems. By representing the probability distributions of the system state as elements in a reproducing kernel Hilbert space, we can learn the “best fit” distribution via a simple regularized least-squares problem, and then compute the stochastic reachability safety probabilities as simple linear operations. This technique admits finite sample bounds and has known convergence in probability. We implement these methods as part of `SReachTools`, and demonstrate their use on a double integrator system, on a million-dimensional repeated planar quadrotor system, and a cart-pole system with a black-box neural network controller.

CCS CONCEPTS

• **Computing methodologies** → **Computational control theory**; *Kernel methods*; • **Theory of computation** → *Stochastic control and optimization.*

KEYWORDS

Stochastic Reachability, Machine Learning, Stochastic Optimal Control

ACM Reference Format:
Anonymous Author(s). 2021. SReachTools Kernel Module: Data-Driven Stochastic Reachability Using Hilbert Space Embeddings of Distributions. In *Proceedings of Hybrid Systems: Computation and Control (HSCC '21).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Modern control systems incorporate a wide variety of elements which are resistant to traditional modeling techniques. For instance, systems with autonomous or learning components, human-in-the-loop elements, or poorly characterized stochasticity provide a significant challenge for model-based analysis methods. In practice, model assumptions can be overly simplistic or fail to capture uncertain behavior, and in some cases are simply wrong. As such, these scenarios have brought about a need for algorithms which can provide probabilistic guarantees of safety, even when a comprehensive model of the system is unavailable. Thus, data-driven techniques for safety analysis are widely applicable for systems with poorly characterized dynamics or uncertainties, and provide an inroad for computing the probability of safety for stochastic systems in a model-free manner.

We present an addition to `SReachTools` [24] which enables data-driven stochastic reachability, based on a machine learning technique known as conditional distribution embeddings [12, 17]. As a nonparametric technique, kernel distribution embeddings use principles from functional analysis to embed probability distributions as elements in a high-dimensional Hilbert space. These techniques have applications to Markov models [7], partially-observable models [13, 16], and have recently been used to solve stochastic reachability problems [20–22]. We incorporate an implementation of the algorithms presented in [20–22] into `SReachTools`, enabling data-driven solutions for stochastic reachability problems that are model-free and distribution agnostic.

These algorithms have several advantages over comparable model-based techniques. First, the techniques admit finite sample bounds [21] and provide convergence guarantees in the infinite sample case [17]. Second, the algorithms largely avoid the curse of dimensionality [1], which can be a significant computational roadblock when solving dynamic programs. Without modification, [20] computes the safety probabilities for a stochastic chain of integrators up to ten thousand dimensions, which is beyond the scope of many existing toolsets. However, the techniques are also amenable to several approximative speedup techniques [6, 14, 28], which have been shown to reduce the computational complexity down to log-linear time. These techniques generally rely upon Fourier approximations of kernel functions and random sampling in the frequency domain, as well as approximations of Gaussian random matrices to alleviate the computational burden. In [22], the authors present an application of one of these techniques, known as random Fourier features [14], to solve a stochastic reachability problem for a million-dimensional system.

Several point-based stochastic reachability techniques are already implemented in `SReachTools`, based on chance constraints [26], Fourier transforms [25], and particle-based approaches [10], among others. Several existing toolboxes, including Faust^2 [18], PRISM [9], STORM [4], and multiple preexisting algorithms in `SReachTools`, present solutions for stochastic reachability problems and model-checking of continuous and discrete-time Markov chains. Unlike most traditional approaches, however, our algorithms are data-driven, meaning we treat the system as a black box, and do not rely upon gridding-based solutions. Because of this, we are able to perform stochastic reachability on systems with arbitrary disturbances, as well as systems with autonomous elements such as neural network controllers (Figure 1). Effectively, this means we can also compute the safety probabilities for autonomous systems and perform neural network verification in a model-free

HSCC '21, May 19-21, 2021, Nashville, USA
© 2021 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

environment. Several existing toolboxes, such as Sherlock [5], NNV [23], and Marabou [8] tackle the problem of neural network verification, but to the best of our knowledge, our toolbox is the first to be able to compute safety probabilities for stochastic, autonomous systems using backward reachability.

The paper is organized as as follows: In Section 2, we present the system model and outline the stochastic reachability problems our algorithms are designed to solve. The contribution to `SReachTools` is presented in Section 3. We present a brief outline of the theory of conditional distribution embeddings and random Fourier features. Then, we describe the algorithms and their use as part of `SReachTools`. In Section 4, we present several numerical examples demonstrating the algorithms, including a stochastic integrator system, a repeated planar quadrotor system, and a cart-pole system with a black-box neural network controller. Concluding remarks are presented in Section 5.

2 STOCHASTIC REACHABILITY

We utilize the following notation throughout the paper. Let E be an arbitrary nonempty space. The indicator function $1_A : E \to \{0, 1\}$ of $A \subseteq E$ is defined such that $1_A(x) = 1$ if $x \in A$, and $1_A(x) = 0$ if $x \notin A$. Let $\mathcal{E}$ denote the σ-algebra on E. If E is a topological space [3], the σ-algebra generated by the set of all open subsets of E is called the Borel σ-algebra, denoted by $\mathscr{B}(E)$. Let $(\Omega, \mathcal{F}, \mathbb{P})$ denote a probability space, where $\mathcal{F}$ is the σ-algebra on Ω and $\mathbb{P} : \mathcal{F} \to [0, 1]$ is a *probability measure* on the measurable space $(\Omega, \mathcal{F})$. A measurable function $X : \Omega \to E$ is called a *random variable* taking values in $(E, \mathcal{E})$. The image of $\mathbb{P}$ under X, $\mathbb{P}(X^{-1}A)$, $A \in \mathcal{E}$ is called the *distribution* of X.

2.1 System Model

Consider a Markov control process $\mathcal{H}$ as defined in [19].

Definition 1. *A Markov control process $\mathcal{H} = (\mathcal{X}, \mathcal{U}, Q)$ is comprised of:*

- *$\mathcal{X} \subseteq \mathbb{R}$ a measurable Borel space called the state space;*
- *$\mathcal{U} \subset \mathbb{R}$, a compact Borel space called the control space;*
- *$Q : \mathscr{B}(\mathcal{X}) \times \mathcal{X} \times \mathcal{U} \to [0, 1]$, a stochastic kernel that assigns a probability measure $Q(\cdot \mid x, u)$ on $(\mathcal{X}, \mathscr{B}(\mathcal{X}))$ to every $(x, u) \in \mathcal{X} \times \mathcal{U}$.*

The system evolves from an initial condition $x_0 \in \mathcal{X}$ over a finite time horizon $k = 0, 1, \ldots, N$ with control inputs chosen according to a Markov control policy π.

Definition 2 (Markov Policy). *A Markov control policy $\pi = \{\pi_0, \pi_1, \ldots, \pi_{N-1}\}$ is a sequence of universally measurable maps $\pi_k : \mathcal{X} \to \mathcal{U}$, $k = 0, 1, \ldots, N-1$. The set of all admissible Markov policies is denoted as $\mathcal{M}$.*

2.2 Problem Definitions

We define the stochastic reachability problems the algorithms can solve as in [19] using the stochastic reachability tube definition from [27].

Definition 3 (Stochastic Reachability Tube). *Given a finite time horizon $N \in \mathbb{N}$, a stochastic reachability tube is a sequence $\mathcal{A} = \{\mathcal{A}_0, \ldots, \mathcal{A}_N\}$ of nonempty sets $\mathcal{A}_k$, where $k = 0, 1, \ldots, N$.*

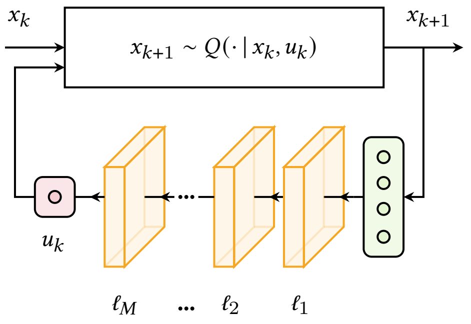

Figure 1: Feedback control diagram for a stochastic system with a neural network controller.

2.2.1 Terminal-Hitting Time Problem.

Given a target tube $\mathcal{T}$, the terminal-hitting time safety probability $\hat{p}^{\pi}_{x_0}$ is defined as the probability that a system following a policy π will reach the target set $\mathcal{T}_N$ at time $k = N$ while remaining within the constraint tube $\mathcal{T}$ for all time $k < N$ from an initial condition x_0.

$$\hat{p}^{\pi}_{x_0}(\mathcal{T}) = \mathbb{E}^{\pi}_{x_0}\left[\left(\prod_{i=0}^{N-1} 1_{\mathcal{T}_i}(x_i)\right) 1_{\mathcal{T}_N}(x_N)\right] \tag{1}$$

For a fixed policy $\pi \in \mathcal{M}$, we define the terminal-hitting value functions $W^{\pi}_k : \mathcal{X} \to \mathbb{R}$, $k = 0, 1, \ldots, N$ as

$$\begin{aligned} W^{\pi}_N(x) &= 1_{\mathcal{T}_N}(x) \\ W^{\pi}_k(x) &= 1_{\mathcal{T}_k}(x) \int_{\mathcal{X}} W^{\pi}_{k+1}(y) Q(\mathrm{d}y \mid x, \pi_k(x)) \end{aligned} \tag{2}$$

Then $W^{\pi}_0(x_0) = \hat{p}^{\pi}_{x_0}(\mathcal{T})$.

2.2.2 First-Hitting Time Problem.

Let $\mathcal{K}$ denote the constraint tube and $\mathcal{T}$ denote the target tube. The first-hitting time safety probability $p^{\pi}_{x_0}$ is defined as the probability that a system following a policy π will reach the target tube $\mathcal{T}$ at some time $j \leq N$ while remaining within the constraint tube $\mathcal{K}$ for all time $k < j$ from an initial condition x_0.

$$p^{\pi}_{x_0}(\mathcal{K}, \mathcal{T}) = \mathbb{E}^{\pi}_{x_0}\left[\sum_{j=0}^{N}\left(\prod_{i=0}^{j-1} 1_{\mathcal{K}_i \setminus \mathcal{T}_i}(x_i)\right) 1_{\mathcal{T}_N}(x_j)\right] \tag{3}$$

For a fixed policy $\pi \in \mathcal{M}$, we define the first-hitting value functions $V^{\pi}_k : \mathcal{X} \to \mathbb{R}$, $k = 0, 1, \ldots, N$ as

$$\begin{aligned} V^{\pi}_N(x) &= 1_{\mathcal{T}_N}(x) \\ V^{\pi}_k(x) &= 1_{\mathcal{T}_k}(x) + 1_{\mathcal{K}_k \setminus \mathcal{T}_k}(x) \int_{\mathcal{X}} V^{\pi}_{k+1}(y) Q(\mathrm{d}y \mid x, \pi_k(x)) \end{aligned} \tag{4}$$

Then $V^{\pi}_0(x_0) = p^{\pi}_{x_0}(\mathcal{K}, \mathcal{T})$.

3 DATA-DRIVEN STOCHASTIC REACHABILITY

We consider the case where the stochastic kernel Q is unknown, but observations taken from the system evolution are available. Thus, we have no prior knowledge of the system dynamics or the structure of the disturbance. Because Q is unknown, we cannot compute the safety probabilities in (4) or (2) directly. Instead, we seek to compute an approximation of the safety probabilities by embedding the expectation operator with respect to the stochastic kernel Q in a reproducing kernel Hilbert space and estimating the operator in Hilbert space.

Definition 4 (RKHS). *Let E be an arbitrary, nonempty space and $\mathscr{H}_E$ be a Hilbert space over E, which is a linear space of functions of the form $f : E \to \mathbb{R}$. The Hilbert space $\mathscr{H}_E$ is a* reproducing kernel Hilbert space *(RKHS) if there exists a positive definite kernel function $k_E : E \times E \to \mathbb{R}$, and it obeys the following properties:*

$$k_E(x, \cdot) \in \mathscr{H}_E \qquad \forall x \in E \tag{5a}$$

$$f(x) = \langle f, k_E(x, \cdot)\rangle_{\mathscr{H}_E} \qquad \forall f \in \mathscr{H}_E, \forall x \in E \tag{5b}$$

where (5b) *is called the* reproducing property, *and for any $x, x' \in E$, we denote $k_E(x, \cdot)$ in the RKHS $\mathscr{H}_E$ as a function on E such that $x' \mapsto k_E(x, x')$.*

We define an RKHS $\mathscr{H}_\mathcal{X}$ over $\mathcal{X}$ and $\mathscr{H}_{\mathcal{X}\times\mathcal{U}}$ over $\mathcal{X} \times \mathcal{U}$ with corresponding kernel functions $k_\mathcal{X}$ and $k_{\mathcal{X}\times\mathcal{U}}$, and define the conditional distribution embedding of Q as:

$$m_{Y|x,u} := \int_\mathcal{X} k_\mathcal{X}(y, \cdot) Q(\mathrm{d}y \mid x, u) \tag{6}$$

By the reproducing property, for any function $f \in \mathscr{H}_\mathcal{X}$, we can compute the expectation of f with respect to the distribution $Q(\cdot \mid x, u)$, where $(x, u) \in \mathcal{X} \times \mathcal{U}$, as an inner product in Hilbert space with the embedding $m_{Y|x,u}$ [20]. Thus, we can compute the safety probabilities for the first-hitting time problem and the terminal-hitting time problem by computing the expectations in (4) and (2) as Hilbert space inner products.

3.1 Kernel Distribution Embeddings

However, we typically do not have access to $m_{Y|x,u}$ directly since the distribution is typically unknown. Instead, we compute an estimate $\hat{m}_{Y|x,u}$ of $m_{Y|x,u}$ using a sample $\mathcal{S} = \{(x_i{}', x_i, u_i)\}_{i=1}^M$ of $M \in \mathbb{N}$ observations taken i.i.d. from Q, where $u_i = \pi(x_i)$ and $x_i{}' \sim Q(\cdot \mid x_i, u_i)$. The estimate $\hat{m}_{Y|x,u}$ can be found as the solution to a regularized least squares problem:

$$\min_{\hat{m}} \frac{1}{M} \sum_{i=1}^M \| k_\mathcal{X}(x_i{}', \cdot) - \hat{m}_{Y|x_i,u_i} \|_{\mathscr{H}_\mathcal{X}}^2 + \lambda \|\hat{m}\|_\Gamma^2 \tag{7}$$

where $\lambda > 0$ is the regularization parameter and Γ is a vector-valued RKHS. The solution to (7) is unique and has the following form:

$$\hat{m}_{Y|x,u} = \beta^\top \Psi \tag{8}$$

where $\beta \in \mathscr{H}_\mathcal{X}$ and Ψ is known as a *feature vector*, with elements $\Psi_i = k_{\mathcal{X}\times\mathcal{U}}((x_i, u_i), (x, u))$. The coefficients β are the unique solution to the system of linear equations

$$(G + \lambda M I)\beta = \Phi \tag{9}$$

where $G = (g_{ij}) \in \mathbb{R}^{M\times M}$ is known as the Gram or kernel matrix, with elements $g_{ij} = k_{\mathcal{X}\times\mathcal{U}}((x_i, u_i), (x_j, u_j))$, and Φ is a feature vector with elements $\Phi_i = k_\mathcal{X}(x_i{}', \cdot)$. Thus, we can approximate the expectation of a function $f \in \mathscr{H}_\mathcal{X}$ using an inner product $\langle \hat{m}_{Y|x,u}, f\rangle_{\mathscr{H}_\mathcal{X}}$. As shown in [20, 22], we can substitute the inner product into the backward recursion in (4) and (2) to approximate the safety probabilities. We have implemented this in `SReachTools` as `KernelEmbeddings`. We present the algorithm for the first-hitting time problem as Algorithm 1.

Algorithm 1: `KernelEmbeddings`

```
Input: sample $\mathcal{S}$, policy $\pi$, horizon $N$
Output: value function estimate $V_0^\pi(x)$
1: Initialize $V_N^\pi(x) \leftarrow 1_{\mathcal{T}_N}(x)$
2: Compute $\hat{m}_{Y|x,\pi(x)}$ (8) using $\mathcal{S}$
3: for $k \leftarrow N-1$ to $0$ do
4:     $\mathcal{Y} \leftarrow [V_{k+1}^\pi(x_1{}'), \dots, V_{k+1}^\pi(x_M{}')]^\top$
5:     $V_k^\pi(x) \leftarrow 1_{\mathcal{T}_k}(x) + 1_{\mathcal{K}_k\setminus\mathcal{T}_k}(x)\mathcal{Y}^\top(G + \lambda M I)^{-1}\Psi$
6: end for
7: Return $V_0^\pi(x) \approx p_x^\pi(\mathcal{K}, \mathcal{T})$
```

Algorithm 2: `KernelEmbeddingsRFF`

```
Input: sample $\mathcal{S}$, policy $\pi$, horizon $N$, sample $\Omega$
Output: value function estimate $V_0^\pi(x)$
1: Initialize $V_N^\pi(x) \leftarrow 1_{\mathcal{T}_N}(x)$
2: Compute RFF approximation of $\hat{m}_{Y|x,\pi(x)}$ (11) using $\mathcal{S}$ and $\Omega$
3: for $k \leftarrow N-1$ to $0$ do
4:     $\mathcal{Y} \leftarrow [V_{k+1}^\pi(x_1{}'), \dots, V_{k+1}^\pi(x_M{}')]^\top$
5:     $V_k^\pi(x) \leftarrow 1_{\mathcal{T}_k}(x) + 1_{\mathcal{K}_k\setminus\mathcal{T}_k}(x)\mathcal{Y}^\top(ZZ^\top + \lambda M I)^{-1}Z$
6: end for
7: Return $V_0^\pi(x) \approx p_x^\pi(\mathcal{K}, \mathcal{T})$
```

3.2 Random Fourier Features

Note that in order to compute β in (9), we must solve a system of linear equations that scales with the number of observations M in the sample $\mathcal{S}$. When M is prohibitively large, [14] shows that by exploiting Bochner's theorem [15], we can approximate the kernel function by computing its Fourier transform and approximating the Fourier integral in feature space using random realizations of the frequency variable.

Bochner's Theorem. *[15] A continuous, translation-invariant kernel function $k(x, x') = \varphi(x - x')$ is positive definite if and only if $\varphi(x - x')$ is the Fourier transform of a non-negative Borel measure Λ.*

Following [14], we construct an estimate of the Fourier integral using a set of D realizations $\Omega = \{\omega_i\}_{i=1}^D$, such that ω_i is drawn i.i.d. from the Borel measure Λ according to $\omega_i \sim \Lambda(\cdot)$. Then, we can efficiently compute an approximation of the kernel as a sum of cosines.

$$k(x, x') \approx \frac{1}{D} \sum_{i=1}^D \cos(\omega_i^\top (x - x')) \tag{10}$$

According to [22], we can approximate the estimate using (10) as:

$$\hat{m}_{Y|x,u} \approx \gamma^\top Z \tag{11}$$

where Z is a feature vector computed using the RFF approximation of $k_{\mathcal{X}\times\mathcal{U}}$ and the coefficients γ are the solution to the system of linear equations

$$(ZZ^\top + \lambda M I)\gamma = \Phi \tag{12}$$

This enables a more computationally efficient approximation of the safety probabilities [22] when the number of observations M is large, or when the dimensionality of the system is high. We implement this in `SReachTools` as `KernelEmbeddingsRFF` and present the algorithm for the first-hitting time problem as Algorithm 2.

Table 1: Computation times of SReachPoint algorithms for a stochastic double integrator system with $N = 5$.

Algorithm	Sample Size	Computation Time [s]
KernelEmbeddings	$M = 2500$	0.72 s
KernelEmbeddingsRFF	$M = 2500$, $D = 5000$	2.78 s
ChanceAffine		3.71 s
ChanceAffineUniform		2.22 s
ChanceOpen		0.25 s
GenzpsOpen		0.53 s
ParticleOpen		0.34 s
VoronoiOpen		1.69 s

3.3 SReachTools Kernel Module

We incorporate the algorithms into SReachTools as modular components, meaning they can be applied to any closed-loop system for which observations are available, and can be applied to either the first-hitting time problem, the terminal-hitting time problem, or the reach-avoid problem [24], which can be viewed as a simplification of the terminal-hitting time problem.

In the implementation, we restrict ourselves to a Gaussian kernel function since the Fourier transform is easy to compute, and to enable a more direct comparison between the two algorithms. The kernel has the form $k(x, x') = \exp(-\|x - x'\|_2^2/2\sigma^2)$, where σ is known as the bandwidth parameter. In order to use the algorithms, we must specify the parameter σ, the regularization parameter λ (7), the sample $\mathcal{S}$, and the problem, which is either the first-hitting time problem or the terminal hitting time problem, parameterized by the stochastic reachability tubes $\mathcal{K}$ and $\mathcal{T}$ (Definition 3).

The algorithm parameters σ and λ are typically chosen via cross-validation, though [2] suggests a method to select a more optimal rate. The default values are chosen to be $\sigma = 0.1$ and $\lambda = 1$. We represent a sample $\mathcal{S} = \{(x_i', x_i, u_i)\}_{i=1}^M$ of size $M \in \mathbb{N}$ taken i.i.d. from a Markov control process $\mathcal{H}$ as a set of matrices, (X, U, Y), where the i^{th} columns of X and U are the observations x_i and u_i, where $u_i = \pi(x_i)$, and the i^{th} column of Y is x_i', where $x_i' \sim Q(\cdot \mid x_i, u_i)$. For KernelEmbeddingsRFF, we are also required to specify the size D of the frequency sample Ω. The stochastic reachability tubes $\mathcal{K}$ and $\mathcal{T}$ are specified as in [24], which is typically defined as a sequence of polyhedra representing the constraints. However, we also include the possibility of representing the tubes via a function-based approach, where the constraints are user-specified indicator functions.

We then use the sample, the constraint and target tube definitions, and the kernel parameters as inputs to the algorithm and compute the safety probabilities for a point $x_0 \in \mathcal{X}$ using SReachPoint [24], which returns the safety probability $p_{x_0}^{\pi}(\mathcal{K}, \mathcal{T}) \in [0, 1]$.

4 NUMERICAL EXPERIMENTS

We present several examples to showcase the capabilities of the proposed methods. Numerical experiments were performed in Matlab on a Intel Xeon CPU with 32 GB RAM, and computation times were obtained using Matlab's Performance Testing Framework.

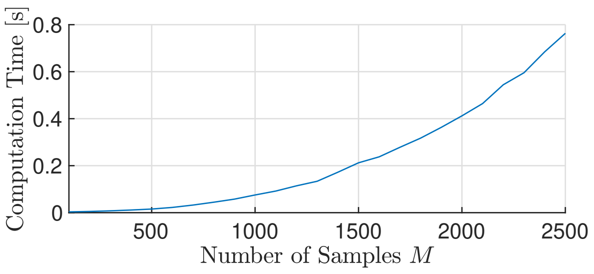

Figure 2: Computation time of KernelEmbeddings as a function of the sample size M.

4.1 Stochastic Chain of Integrators

We first consider a toy example to demonstrate the use of the algorithms and compare the output against known results. Consider a n-dimensional stochastic chain of integrators [22, 27], in which the input appears at the n^{th} derivative and each element of the state vector is the discretized integral of the element that follows it. The dynamics with sampling time T are given by:

$$x_{k+1} = \begin{bmatrix} 1 & T & \cdots & \frac{T^{n-1}}{(n-1)!} \\ 0 & 1 & \cdots & \frac{T^{n-2}}{(n-2)!} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} \frac{T^n}{n!} \\ \frac{T^{n-1}}{(n-1)!} \\ \vdots \\ T \end{bmatrix} u_k + w_k \tag{13}$$

For the purpose of comparison against other algorithms, we restrict ourselves to the 2-dimensional case and Gaussian disturbances. We compare the computation time of the algorithms with a single evaluation point $x_0 = 0$ against several other algorithms present in SReachTools. The results are displayed in Table 1. Note that the kernel-based algorithms do not compute the *optimal* value functions unless the policy is the *maximally safe* Markov policy π^* [19]. This makes direct comparison of the algorithms in SReachTools difficult, since the existing algorithms also perform controller synthesis to obtain the maximal reach probability. We then seek to quantify the effect of the sample size M on the computation time. We vary M and plot the computation time as a function of M in Figure 2. This shows that as we increase the sample size M, the computation time increases exponentially.

In order to validate the approach, we compute the safety probabilities using dynamic programming $V_0^{\text{DP}}(x_0)$. We then generate observations of the system by choosing $x_i \in \mathcal{X}$, $i = 1, \ldots, 2500$, uniformly in the range $[-1.1, 1.1]^2$ and collect a sample $\mathcal{S}$ of size $M = 2500$. The dynamic programming results are shown in Figure 3(a). Figure 3(b) shows the safety probabilities $V_0^{KM}(x_0)$ computed for a 2-dimensional integrator system using KernelEmbeddings. We then treat the dynamic programming solution as a truth model, and plot the absolute error $|V_0^{\text{DP}}(x_0) - V_0^{KM}(x_0)|$ in Figure 3(c). In order to evaluate KernelEmbeddingsRFF, we then collect a sample of frequency realizations of size $D = 15000$ and compute the safety probabilities $V_0^{RFF}(x_0)$. The results are shown in Figure 3(d), where Figure 3(e) shows the absolute error, $|V_0^{\text{DP}}(x_0) - V_0^{RFF}(x_0)|$.

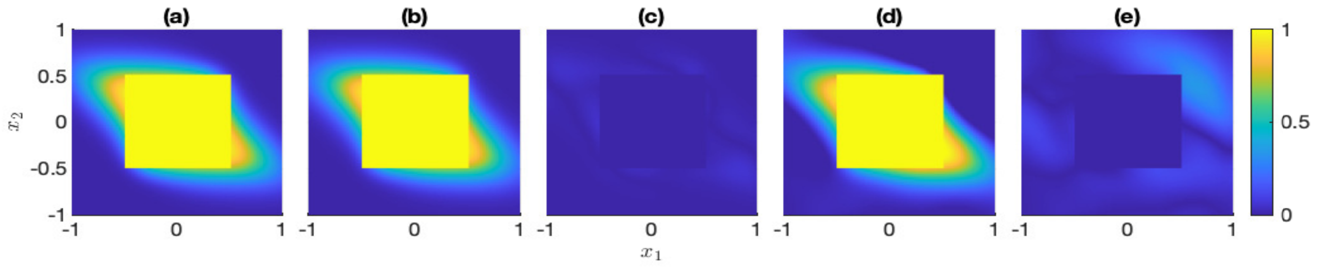

Figure 3: (a) Safety probabilities $V_0^{\mathrm{DP}}(x_0)$ for a 2-D stochastic chain of integrators computed using dynamic programming over a time horizon $N = 5$. (b) Safety probabilities $V_0^{KM}(x_0)$ computed using `KernelEmbeddings` (c) Absolute error $|V_0^{\mathrm{DP}}(x_0) - V_0^{KM}(x_0)|$. (d) Safety probabilities $V_0^{RFF}(x_0)$ computed using `KernelEmbeddingsRFF` (e) Absolute error $|V_0^{\mathrm{DP}}(x_0) - V_0^{RFF}(x_0)|$.

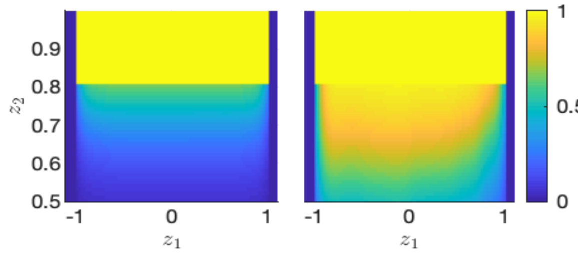

Figure 4: (left) First-hitting time safety probabilities for a single planar quadrotor system with a Gaussian disturbance and (right) with a beta distribution disturbance over the horizon $N = 5$.

4.2 Repeated Planar Quadrotor

This example is used to showcase the ability of the system to handle high-dimensional systems [22]. This problem can be interpreted as a simplification of formation control for a large swarm of quadrotors, where we compute the safety probabilities for the entire swarm as they are controlled to reach a particular elevation. The nonlinear dynamics of a single quadrotor are given by

$$\begin{aligned} m\ddot{x} &= -(u_1 + u_2)\sin(\theta) \\ m\ddot{y} &= (u_1 + u_2)\cos(\theta) - mg \\ \mathcal{I}\ddot{\theta} &= r(u_1 - u_2) \end{aligned} \tag{14}$$

where x is the lateral position, y is the vertical position, θ is the pitch, and we have the constants intertia $\mathcal{I} = 2$, length $r = 2$, mass $m = 5$, and $g = 9.8$ is the gravitational constant.

The safety probabilities for a single planar quadrotor with a Gaussian and non-Gaussian disturbance are shown in Figure 4. We then formulate the dynamics for a swarm of quadrotors by repeating the dynamics until we have over a million state variables. We then generate a sample $\mathcal{S}$ of size $M = 1000$ for the repeated system and compute the safety probabilities using `KernelEmbeddings`. Then, we collect a sample of frequency realizations of size $D = 15000$ and compute the safety probabilities using `KernelEmbeddingsRFF` over a single time step $N = 1$. The mean computation time for `KernelEmbeddings` was 1.23 hours, and for `KernelEmbeddingsRFF` the mean computation time was 44.63 seconds. This shows that `KernelEmbeddingsRFF` can be used to compute the safety probabilities for extremely high-dimensional systems.

4.3 Cart-Pole

The following examples are used to showcase the ability of the algorithms to compute the safety probabilities for systems with neural network controllers.

4.3.1 Linearized Cart-Pole. The dynamics for the linearized cart-pole system [11] are given by:

$$\begin{aligned} \ddot{x} &= 0.0043\dot{\theta} - 2.75\theta + 1.94u - 10.95\dot{x} \\ \ddot{\theta} &= 28.58\theta - 0.044\dot{\theta} - 4.44u + 24.92\dot{x} \end{aligned} \tag{15}$$

We add an additional Gaussian disturbance $\mathcal{N}(0, \Sigma)$ with $\Sigma = 0.01I$ to the dynamical state equations, which can simulate dynamical uncertainty or minor system perturbations. The control input is computed via a feedforward neural network controller [11], which takes the current state and outputs a real number $u \in \mathbb{R}$, which can be interpreted as the input torque.

Figure 5(a) shows a cross-section of the safety probabilities for the system computed using `KernelEmbeddings` holding x and $\dot{x}$ constant at 0, and Figure 5(b) shows a 2-D projection. The algorithms are agnostic to the structure of the dynamics, which means we do not require any prior knowledge of the structure of the neural network controller. Because of this, the algorithms are able to perform verification of systems that incorporate learning enabled components.

4.3.2 Nonlinear Cart-Pole. We then analyzed a nonlinear cart-pole system with a neural network controller [11], with dynamics given by:

$$\begin{aligned} \ddot{x} &= \frac{u + ml\omega^2\sin(\theta)}{m_t} \\ &\quad - \frac{ml(g\sin(\theta) - \cos(\theta))(\frac{u+ml\omega^2\sin(\theta)}{m_t})}{l(\frac{4}{3} - m\frac{\cos^2(\theta)}{m_t})}\frac{\cos(\theta)}{m_t} \\ \ddot{\theta} &= \frac{g\sin(\theta) - \cos(\theta)(\frac{u+ml\omega^2\sin(\theta)}{m_t})}{l(\frac{4}{3} - m\frac{\cos^2(\theta)}{m_t})}\frac{\cos(\theta)}{m_t} \end{aligned} \tag{16}$$

where $g = 9.8$ is the gravitational constant, the pole mass is $m = 0.1$, half the pole's length is $l = 0.5$, and $m_t = 1.1$ is the total mass. The

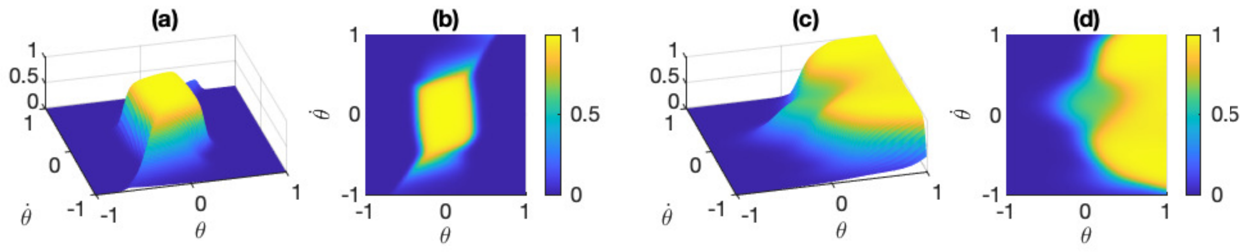

Figure 5: (a), (b) 3-D and 2-D cross-sections respectively, of the safety probabilities computed using `KernelEmbeddings` for linearized Cart-Pole system, and (c), (d) 3-D and 2-D cross-sections respectively, of safety probabilities computed using `KernelEmbeddings` for the nonlinear cart-pole system.

control input, $u \in \{-10, 10\}$, which affects the lateral position of the cart, is chosen by the neural network controller [11]. We add an additional Gaussian disturbance $\mathcal{N}(0, \Sigma)$ with $\Sigma = 0.01I$ to the dynamical state equations.

Figure 5(c) shows a 3-D representation of the safety probabilities for the system computed using `KernelEmbeddings`, and Figure 5(d) shows a 2-D projection. This example shows that we can handle systems which are traditionally very difficult to model and analyze, such as nonlinear systems and systems with neural network controllers.

5 CONCLUSION & FUTURE WORK

We presented a data-driven stochastic reachability module for `SReachTools` based on conditional distribution embeddings. These algorithms add to the suite of stochastic reachability algorithms already present in the toolbox and enable point-based stochastic reachability for a wide variety of stochastic systems. We would like to extend the kernel-based algorithms to enable model-free controller synthesis and broaden the capability of the algorithms to handle a wider variety of systems, such as hybrid system models.

REFERENCES

[1] Richard Bellman and Stuart Dreyfus. 2015. *Applied dynamic programming*. Princeton university press.

[2] A. Caponnetto and E. De Vito. 2007. Optimal Rates for the Regularized Least-Squares Algorithm. *Foundations of Computational Mathematics* 7, 3 (Jul 2007), 331–368.

[3] Erhan Çinlar. 2011. Probability and Stochastics. (2011).

[4] Christian Dehnert, Sebastian Junges, Joost-Pieter Katoen, and Matthias Volk. 2017. A Storm is Coming: A Modern Probabilistic Model Checker. In *Computer Aided Verification*, Rupak Majumdar and Viktor Kunčak (Eds.). Springer International Publishing, Cham, 592–600.

[5] Souradeep Dutta, Xin Chen, Susmit Jha, Sriram Sankaranarayanan, and Ashish Tiwari. 2019. Sherlock-A tool for verification of neural network feedback systems. In *International Conference on Hybrid Systems: Computation and Control*. 262–263.

[6] Steffen Grünewälder, Guy Lever, Luca Baldassarre, Sam Patterson, Arthur Gretton, and Massimilano Pontil. 2012. Conditional mean embeddings as regressors. In *International Conference on Machine Learning*. 1803–1810.

[7] S Grünewälder, G Lever, Baldassarre Ll, M Pontil, and A Gretton. 2012. Modelling transition dynamics in MDPs with RKHS embeddings. In *International Conference on Machine Learning*. Omnipress, 535–542.

[8] Guy Katz, Derek Huang, Duligur Ibeling, Kyle Julian, Christopher Lazarus, Rachel Lim, Parth Shah, Shantanu Thakoor, Haoze Wu, Aleksandar Zeljić, David L. Dill, Mykel Kochenderfer, and Clark Barrett. 2019. The Marabou Framework for Verification and Analysis of Deep Neural Networks. In *Computer Aided Verification*, Isil Dillig and Serdar Tasiran (Eds.). Springer International Publishing, Cham, 443–452.

[9] Marta Kwiatkowska, Gethin Norman, and David Parker. 2011. PRISM 4.0: Verification of probabilistic real-time systems. In *International conference on computer aided verification*. Springer, 585–591.

[10] K. Lesser, M. Oishi, and R. S. Erwin. 2013. Stochastic reachability for control of spacecraft relative motion. In *Conference on Decision and Control*. 4705–4712.

[11] Diego Manzanas Lopez, Patrick Musau, Hoang-Dung Tran, and Taylor Johnson. 2019. Verification of Closed-loop Systems with Neural Network Controllers. In *International Workshop on Applied Verification of Continuous and Hybrid Systems*, Vol. 61. 201–210.

[12] Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, Bernhard Schölkopf, et al. 2017. Kernel Mean Embedding of Distributions: A Review and Beyond. *Foundations and Trends in Machine Learning* 10, 1-2 (2017), 1–141.

[13] Y Nishiyama, A Boularias, A Gretton, and K Fukumizu. 2012. Hilbert Space Embeddings of POMDPs. In *Conference on Uncertainty in Artificial Intelligence*.

[14] Ali Rahimi and Benjamin Recht. 2008. Random features for large-scale kernel machines. In *Advances in neural information processing systems*. 1177–1184.

[15] Walter Rudin. 1962. *Fourier analysis on groups*. Vol. 121967. Wiley.

[16] Le Song, Byron Boots, Sajid M Siddiqi, Geoffrey Gordon, and Alex Smola. 2010. Hilbert space embeddings of hidden Markov models. In *International Conference on Machine Learning*. 991–998.

[17] Le Song, Jonathan Huang, Alex Smola, and Kenji Fukumizu. 2009. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *International Conference on Machine Learning*. 961–968.

[18] Sadegh Esmaeil Zadeh Soudjani, Caspar Gevaerts, and Alessandro Abate. 2015. FAUST 2 : Formal Abstractions of Uncountable-STate STochastic Processes. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Vol. 9035. Springer International Publishing, 272–286.

[19] Sean Summers and John Lygeros. 2010. Verification of discrete time stochastic hybrid systems: A stochastic reach-avoid decision problem. *Automatica* 46, 12 (Dec 2010), 1951–1961.

[20] Adam Thorpe and Meeko Oishi. 2019. Model-Free Stochastic Reachability Using Kernel Distribution Embeddings. *IEEE Control Systems Letters* 4, 2 (2019), 512–517.

[21] Adam Thorpe, Kendric Ortiz, and Meeko Oishi. 2020. Data-Driven Stochastic Reachability Using Hilbert Space Embeddings. arXiv:2010.08036 [math.OC]

[22] Adam Thorpe, Vignesh Sivaramakrishnan, and Meeko Oishi. 2020. Stochastic Reachability for Systems up to a Million Dimensions. arXiv:1910.10818 [eess.SY]

[23] Hoang-Dung Tran, Patrick Musau, Diego Manzanas Lopez, Xiaodong Yang, Luan Viet Nguyen, Weiming Xiang, and Taylor Johnson. 2020. NNV: A Tool for Verification of Deep Neural Networks and Learning-Enabled Autonomous Cyber-Physical Systems. In *International Conference on Computer-Aided Verification*.

[24] Abraham Vinod, Joseph Gleason, and Meeko Oishi. 2019. SReachTools: a MATLAB stochastic reachability toolbox. In *International Conference on Hybrid Systems: Computation and Control*. ACM, 33–38.

[25] Abraham Vinod and Meeko Oishi. 2017. Scalable Underapproximation for the Stochastic Reach-Avoid Problem for High-Dimensional LTI Systems Using Fourier Transforms. *IEEE Control Systems Letters* 1, 2 (Oct 2017), 316–321.

[26] Abraham Vinod and Meeko Oishi. 2019. Affine controller synthesis for stochastic reachability via difference of convex programming. In *Conference on Decision and Control*. IEEE, 7273–7280.

[27] Abraham Vinod and Meeko Oishi. 2020. Stochastic reachability of a target tube: Theory and computation. arXiv:1810.05217 [math.OC]

[28] Zichao Yang, Andrew Wilson, Alex Smola, and Le Song. 2015. A la carte–learning fast kernels. In *Artificial Intelligence and Statistics*. 1098–1106.