2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Data-Driven Reachability Analysis of Stochastic Dynamical Systems with Conformal Inference

Navid Hashemi, Xin Qin, Lars Lindemann, and Jyotirmoy V. Deshmukh

***Abstract*— We consider data-driven reachability analysis of discrete-time stochastic dynamical systems using conformal inference. We assume that we are not provided with a symbolic representation of the stochastic system, but instead have access to a dataset of K-step trajectories. The reachability problem is to construct a probabilistic flowpipe such that the probability that a K-step trajectory can violate the bounds of the flowpipe does not exceed a user-specified failure probability threshold. The key ideas in this paper are: (1) to learn a surrogate predictor model from data, (2) to perform reachability analysis using the surrogate model, and (3) to quantify the surrogate model's incurred error using conformal inference in order to give probabilistic reachability guarantees. We focus on learning-enabled control systems with complex closed-loop dynamics that are difficult to model symbolically, but where state transition pairs can be queried, e.g., using a simulator. We demonstrate the applicability of our method on examples from the domain of learning-enabled cyber-physical systems.**

I. Introduction

Reachability analysis of stochastic nonlinear dynamical systems is a challenging problem that has been extensively studied in the literature [1]–[13]. Most of the prior work is *model-based*, i.e., it requires a symbolic model of the dynamical system which can then be over-approximated to obtain *flowpipes* or the set of reachable states of the system over a given time horizon. In this paper, we explore the notion of *model-free reachability analysis*, *i.e.*, to compute reachable sets of the stochastic dynamical system even when we *do not have the symbolic system dynamics*, but have access to a numeric simulator or actual behaviors sampled from the system. A significant advantage of such a data-driven technique is that we obtain not only (probabilistic) reachable sets for the system or the simulation model from which the trajectories are sampled, but we can get results over the possibly *infinite* set of models/systems consistent with the set of sampled trajectories. This provides us the opportunity for an analysis technique that is robust to model uncertainty.

There is growing literature on computing probabilistically approximate reachable sets directly from data. The authors in [14] utilize level sets of Christoffel functions and provide a technique to compute a high accuracy probabilistic reach-set for general nonlinear systems. On the other hand, as a comparison, we only have access to sampled trajectories. The authors in [15] propose specific parametric models (linear or polynomial), to identify the Markovian stochastic dynamics of the system from data, and then perform reachability analysis on the identified models. In contrast, we learn non-parameteric surrogates (using neural networks), which are not restricted to Markovian dynamics, and quantify uncertainty using conformal inference. In [16], the authors use an interesting approach based on a Gaussian process-based classifier to separate reachable states from unreachable states, and approximate the reach set by computing the sublevel set of the classifier. We note that this approach uses adaptive sampling of initial states where states are chosen to reduce uncertainty of the surrogate. This may require solving a high-dimensional optimization problem and does not give probability guarantees. The authors also propose an interval abstraction of the reach set, where sample complexity bounds are provided; however, this approach may suffer from conservatism and high computational cost in high-dimensional systems. In [17], the authors assume partial knowledge of the model, while using data to deal with Lipschitz-continuous state dependent uncertainty. The authors in [18] propose a method called DeepReach that uses a neural PDE solver to perform Hamilton-Jacobi method-based reachability analysis for high-dimensional systems. Here, the complexity of the flowpipe computation scales with the complexity of the flowpipe itself rather than the system dimension. While this method uses neural methods to perform reachability analysis, it still requires access to the system dynamics. In [19], the authors combine simulation-guided reachability analysis techniques with data-driven techniques. Here, the authors estimate a discrepancy function using system trajectories using a probably-approximately-correct learning algorithm. The discrepancy function bounds the distance between system trajectories as a function of the distance between their initial states. This function is then used to inflate the simulation trajectories to obtain reachtubes that are then used to compute the approximate reachable set of states. We remark that such an approach requires a parametric form of the discrepancy function which could be hard to obtain.

Our approach to compute probabilistic reachable sets for stochastic systems has the following main steps: (1) we sample a number of system trajectories according to the user-provided distribution on the set of initial states, (2) we learn a data-driven surrogate model that predicts the next K states of the system from a given state, (3) we perform traditional set propagation-based reachability analysis on the surrogate model, and (4) we inflate the resulting reachable

The authors are with the Thomas Lord Department of Computer Science, University of Southern California, Los Angeles, USA. Emails: (navidhas, xinqin, llindema, jdeshmuk)@usc.edu. The authors would like to thank the anonymous reviewers for their feedback. This work was supported by the National Science Foundation through the following grants: CAREER award (SHF-2048094), CNS-1932620, FMitF-1837131, CCF-SHF-1932620, the Airbus Institute for Engineering Research, and funding by Toyota R&D and Siemens Corporate Research through the USC Center for Autonomy and AI.

sets in a systematic fashion to account for the uncertainty induced by sampling from the user-provided distribution on the set of initial states and system stochasticity.

While our general technique can work with arbitrary regression-based surrogate models, in this paper, we choose feedforward neural networks as a data-driven model of the system dynamics[1]. A key reason for choosing feedforward neural networks is to use recently developed techniques to perform set-propagation through neural networks [21]. However, since the surrogate model is learned by sampling, we still need to account for this uncertainty. To do this, we rely on *conformal inference* (CI) or *conformal prediction* [22], [23]. In the machine learning community, CI is a well-known data-efficient framework for error analysis of predictive models. Recently, CI has been employed for data-driven stochastic verification [24]–[27]. For instance, in [25]–[27], the authors obtain confidence intervals on the robustness degree of system behaviors (for example based on the quantitative satisfaction of a given signal temporal logic (STL) property). A naïve adaptation of existing techniques for STL robustness verification to reachability analysis would require training surrogate models to predict each state variable independently (effectively discarding the rich correlation between the state variables) resulting in an analysis that is too conservative. Instead, our approach preserves the structure of the dynamics through our usage of a surrogate model, while still accounting for uncertainty in a systematic fashion using CI.

The layout of our paper is as follows. In Section II, we present the preliminaries and problem statement. Our algorithm for probabilistic reachable set computation is proposed in Section III. We demonstrate the efficacy of our method on several numerical examples in Section IV, and we finally conclude in Section V.

Notation. We use bold letters to indicate vectors and vector-valued functions, and calligraphic letters to denote sets. We denote the set $\{1, 2, \cdots, n\}$ with $[n]$. We present the structure of a feedforward neural network (FFNN) with ℓ hidden layers as an array $[n_0, n_1, \cdots n_{\ell+1}]$, where n_0 denotes the number of inputs $n_{\ell+1}$ is the number of outputs and $n_i, i \in [\ell]$ denotes the width of the i-th hidden layer. We denote $e_i \in \mathbb{R}^n$ as the i-th base vector of $\mathbb{R}^n$. The expression $x \sim \mathcal{X}$ means, the random variable x follows the distribution $\mathcal{X}$. We denote the cardinality of a set A with $|A|$. $\mathsf{Zonotope}(b, A)$ denotes a zonotope [28] with center b and array of base vectors A. The operator $\oplus$ denotes the Minkowski sum. We also denote $\lceil x \rceil$ as the smallest integer greater than $x \in \mathbb{R}$.

II. Problem Statement and Preliminaries

Stochastic Dynamical System. Estimating the set of reachable sets of stochastic dynamical systems starting from a compact set of initial states $\mathcal{I} \subset \mathbb{R}^n$ is a well-studied problem. While this problem has been studied largely with model-based techniques, we focus on *bounded-time, model-free reachability* analysis of a black-box discrete-time stochastic dynamical system M. A random trajectory of M can be modelled as a sequence of time-stamped states $\sigma_{s_0} = [s_0^\top, \cdots, s_{\mathrm{K}}^\top]^\top \subset \mathbb{R}^{(\mathrm{K}+1)n}$. We denote the distribution over the set of trajectories consistent with M as $\mathcal{S}$, and use $\sigma_{s_0} \sim \mathcal{S}$ to denote that σ_{s_0} is sampled from $\mathcal{S}$. The distribution $\mathcal{S}$ could be induced due to a distribution on the set of initial states of the system as well as stochastic uncertainties in its dynamics. For example, the transition dynamics could be Markovian, i.e., $s_k \sim T(s' \mid s = s_{k-1})$. However, we remark that the techniques proposed in this paper can work for systems with non-Markovian dynamics. For convenience, we denote the distribution over the set of initial states $\mathcal{I}$ as $\mathcal{W}$, and assume that $(\Pr[s_0 \notin \mathcal{I}] = 0)$. The notation $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$ is used to denote that the state s_0 is sampled from $\mathcal{I}$ using the distribution $\mathcal{W}$. A practical choice of $\mathcal{W}$ may be to uniformly sample $\mathcal{I}$.

Trajectory Datasets. We assume that we have access to a dataset of independently sampled trajectories of M, where the initial states are sampled with $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$. We assume that each trajectory has K time-steps. We split the trajectory dataset into two separate datasets, denoted as D_{train} and D_{test}. Here D_{train} is our training dataset and is to be used for training a surrogate model of M. On the other hand, D_{test} is the test dataset, and is to be utilized to generate the calibration dataset for deriving statistical properties. Let $D_{\text{test}} = \{s_{0,i}, \sigma_{s_0,i}\}_{i=1}^{L}$ where L is the number of test trajectories. Our goal is now to compute a probabilistic reach set such that σ_{s_0} is included inside with a probability not lower than a user-defined threshold $1 - \varepsilon$, where $\varepsilon \in (0, 1)$.

We remark that although the data-points within a single trajectory are not independent and identically distributed (i.i.d.), we observe that the trajectory σ_{s_0} can be seen as a vector $\sigma_{s_0} \in \mathbb{R}^{n(\mathrm{K}+1)}$ so that entire trajectories within D_{train} and D_{test} can be viewed as i.i.d. samples in the $\mathbb{R}^{n(\mathrm{K}+1)}$-space. This observation is crucially used later when we quantify uncertainty using conformal prediction.

Surrogate Model. After obtaining the trajectory dataset, we train a surrogate model $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}^{(\mathrm{K}+1)n}$ that maps a given initial state to the predicted K-step trajectory of the system. For instance, the surrogate model can be a feedforward neural network with n inputs and $(\mathrm{K}+1)n$ outputs. Let $\bar{\sigma}_{s_0}$ denote the predicted trajectory, then, $\bar{\sigma}_{s_0} = \mathcal{F}(s_0)$. Training such a K-step predictive model can be difficult, especially when the dynamics are nonlinear. Hence, in practice, we train models that take as input trajectory fragments to predict future trajectory fragments and then sequentially compose such models to obtain a longer predicted trajectory. We remark that this does not affect the probabilistic reasoning that we perform later in the paper as the entire predictive model can still be treated as a deterministic map that takes as input the initial state s_0 and outputs a K-step predicted trajectory.

Conformal Inference. Conformal inference [22], [23], [29] is a statistical tool for uncertainty quantification that has

[1]There are many techniques to identify probably approximately correct surrogate models, for example, using concentration bounds for system identification [20]. In this paper, we eschew such methods, instead preferring the data-efficient approach for uncertainty quantification provided by conformal inference.

recently been used for analysing the uncertainty in the predictions performed by complex machine learning models [30]–[32]. Conformal inference/prediction performs error quantification without making any assumption on the underlying data-generating distribution or the machine learning model. Consider a regression model μ and the random variables $z_1, z_2, ..., z_{m+1}$ where $z_i = (x_i, y_i) \in \mathbb{R}^n \times \mathbb{R}$ with $i \in [m+1]$ which are all independently sampled from the same distribution. Given a miscoverage level $\alpha \in (0,1)$, conformal inference enables us to compute a prediction interval $C(x_{m+1}) = [\mu(x_{m+1}) - R^*,\ \mu(x_{m+1}) + R^*] \subset \mathbb{R}$ from $z_1, z_2, ..., z_m$ such that,

$$\Pr[y_{m+1} \in C(x_{m+1})] \geq 1 - \alpha.$$

More formally, define the residual $R_i = \mid y_i - \mu(x_i) \mid$ for all z_i with $i \in [m+1]$. Since the random variables $z_1, z_2, ..., z_{m+1}$ are independent and identically distributed, the same applies to the residual $R_1, \ldots, R_{m+1}$. Under the assumption that m satisfies $\ell = \lceil (m+1)(1-\alpha) \rceil \leq m$, it holds that R^* can be chosen to be the ℓ-th smallest residual [33, Lemma 1]. Without loss of generality, if $R_1, \ldots, R_m$ are sorted in non-decreasing order, then $R^* = R_\ell$ and it holds that $\Pr[R_{m+1} \leq R^*] \geq (1-\alpha)$. By the choice of the residual, it hence holds that

$$\Pr\left[y_{m+1} \in [\mu(x_{m+1}) - R^*,\ \mu(x_{m+1}) + R^*]\right] \geq 1 - \alpha.$$

We also refer to $\delta = 1 - \alpha$ as the confidence probability.

Problem Definition. Our probabilistic reachability analysis can be formally stated as follows. Given the stochastic dynamical system M with initial state $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$ and trajectory distribution $\mathcal{S}$, training and test datasets D_{train} and D_{test} consisting of K-step trajectories independently sampled from M, and a user provided failure probability threshold $\varepsilon \in (0,1)$, compute a probabilistic reach set (also called flowpipe) X such that:

$$\Pr\left[\sigma_{s_0} \in X\right] \geq 1 - \varepsilon \tag{1}$$

III. Scalable Data-Driven Reachability

In the setting described in the previous section, we now show how to compute a reach set or a flowpipe $X \subset \mathbb{R}^{n(\mathrm{K}+1)}$ using reachability analysis and conformal inference. This flowpipe will contain the trajectory σ_{s_0} of M sampled with initial state $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$ with the confidence level of $\Delta = 1 - \varepsilon$. We hence denote this flowpipe as Δ-confident flowpipe and define it as follows.

Definition 1 (Δ-confident flowpipe): For a given confidence probability $\Delta \in (0,\ 1)$ and a random trajectory $\sigma_{s_0} \sim \mathcal{S}$ with initial state $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$, we say that $X \subset \mathbb{R}^{n(\mathrm{K}+1)}$ is a Δ-confident flowpipe if

$$\Pr[\sigma_{s_0} \in X] \geq \Delta \tag{2}$$

In this work, we are interested in computing X while our access to M is limited to the datasets D_{train} and D_{test}. We will show that we can compute X with valid guarantees by employing reachability analysis on the surrogate model trained from D_{train} and error analysis of this model by applying conformal prediction on D_{test}.

A. Computing Reachsets for Surrogate Models

1) ReLU Surrogate Model: We start with training a neural network surrogate model $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}^{n(\mathrm{K}+1)}$ over the training dataset D_{train}. The surrogate model is trained to approximate the trajectory $\sigma_{s_0} \in \mathbb{R}^{n(\mathrm{K}+1)}$ of the stochastic system M sampled with initial state $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$. We denote the prediction $\bar{\sigma}_{s_0} \in \mathbb{R}^{n(\mathrm{K}+1)}$ of σ_{s_0} as,

$$\bar{\sigma}_{s_0} = \mathcal{F}(s_0) = \begin{bmatrix} s_0^\top & \mathsf{F}^1(s_0) & \cdots & \mathsf{F}^n(s_0) & \cdots \\ & \mathsf{F}^{(n-1)\mathrm{K}}(s_0) & \cdots & \mathsf{F}^{n\mathrm{K}}(s_0) & \end{bmatrix}^\top$$

where $\mathsf{F}^j(s_0)$ is the $(j+n)$-th component of the vector $\mathcal{F}(s_0)$.

Recent works in the literature have had great success on obtaining accurate bounds for the reachability analysis of ReLU neural networks using polyhedral sets [21], [34], [35]. The accuracy of these techniques motivates us to use ReLU activation functions for training neural networks as the surrogate models. These surrogate models will be used for deterministic reachability analysis which provides surrogate flowpipes which we formally define next.

Definition 2 (Surrogate flowpipe): The surrogate flowpipe $\bar{X} \subset \mathbb{R}^{n(\mathrm{K}+1)}$ contains the image of $\mathcal{F}(\mathcal{I})$. Formally, for all, $s_0 \in \mathcal{I}$ it has to hold that $\mathcal{F}(s_0) \in \bar{X}$.

The reachability analysis methodology for ReLU neural networks in [21] introduces two different approaches known as the exact-star and approx-star techniques. These are used to compute a surrogate flowpipe $\bar{X}$. The exact-star technique proposes exact reachability analysis using star sets, but can be slower due to its inherent computational complexity. On the other hand, the approx-star technique computes over-approximation of the flowpipe and is thus runtime-efficient, although it may make the surrogate model reachable set estimation conservative. The computational complexity of the exact-star technique and the conservatism of the approx-star technique can both be noticeably reduced using the idea of set partitioning [21]. In this approach, we partition the set of initial conditions $\mathcal{I}$ into N different sub-partitions,

$$\mathcal{I}_i \subset \mathcal{I}, \quad \bigcup_{i=1}^{N} \mathcal{I}_i = \mathcal{I},$$

and perform reachability analysis on every single sub-region with parallel computing. The inclusion of set-partitioning results in noticeable improvement in the computational efficiency and helps us compute more accurate Δ-confident flowpipes.

B. Computation of a guaranteed Δ-confident flowpipe

When we train neural network surrogate models, typically we minimize a loss function defined as the difference between the K-step trajectory predicted by the surrogate model and the actual trajectory. Depending on the dynamics of the underlying stochastic system M and depending on how well the surrogate model is trained, there is potential for error when predicting the trajectory from a previously unseen initial state. To give probabilistic bounds on this error, we utilize conformal prediction. We formally define the notion of the residual error as follows.

Definition 3 (Residual Error): Given a realization $(s_0,\ \sigma_{s_0})$ sampled from the stochastic black-box system M, the residual $R^j \in \mathbb{R}_{>0}$ is the distance between the $(j+n)$-th component of the trajectory and $\mathsf{F}^j(s_0)$[2] Formally, we define

$$R^j = \left| e_{j+n}^\top \sigma_{s_0} - \mathsf{F}^j(s_0) \right|$$

where, $e_j \in \mathbb{R}^{n(\mathrm{K}+1)}$ is the j^{th} base vector of $\mathbb{R}^{n(\mathrm{K}+1)}$. We consider the component-wise residual R^j since every single component $e_{j+n}^\top \sigma_{s_0}$ in σ_{s_0} may represent a different quantity at a different time. For example, it would not make sense to define a joint error of the position and velocity of a system at some time, which motivates the definition of the component-wise error. Now that we have defined the residual R^j for a component j, let us compute this residual for all calibration trajectories from D_{test}, i.e., for $\sigma_{s_{0,i}}, i \in [L]$.

Definition 4 (Calibration Dataset): For a given test dataset D_{test}, the calibration dataset $\mathcal{R}^j_{D_{\text{test}}},\ j \in [n\mathrm{K}]$, is a collection of pairs $(s_{0,i}, R_i^j), i \in [L]$, such that

$$\mathcal{R}^j_{D_{\text{test}}} = \Big\{ \Big(s_{0,i}, R_i^j \Big) \mid (s_{0,i}, \sigma_{s_{0,i}}) \in D_{\text{test}},\ i \in [L] \Big\}.$$

where $R_i^j = \mid e_{j+n}^\top \sigma_{s_{0,i}} - \mathsf{F}^j(s_{0,i}) \mid$.

As our access to the underlying system M is limited to a finite number of pre-recorded trajectories, it is not possible to exactly compute the distribution of the residual $R^j, j \in [n\mathrm{K}]$. Consequently, we cannot compute the δ-quantile of $R^j, j \in [n\mathrm{K}]$.[3] However, we can utilize the method of conformal prediction [22] to compute an upper bound on the δ-quantile of R^j. Based on the technique introduced in Section II, we sort the residuals R_i^j in non-decreasing order. For simplicity and without loss of generality, let us re-index the values of $R_1^j, \ldots, R_L^j$ so that $R_1^j \leq R_2^j \leq \cdots \leq R_L^j$. We can now apply conformal prediction and define $R^{j*} = R_\ell^j$ where $\ell = \lceil (L+1)\delta \rceil$. Consequently, we know that

$$\Pr[R^j \leq R^{j*}] \geq \delta.$$

In other words, the ℓ-th smallest residual R^{j*} is an upper bound for the δ-quantile of the random variable R^j.

Recall that our ultimate goal is to compute a Δ-confident flowpipe X for trajectories σ_{s_0} from M with distribution $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$. To that end, we compute the vector of upper bounds for all components of residual's δ-quantile from conformal inference, and we denote it by $R^* = \left[R^{1*}, \cdots, R^{n\mathrm{K}*}\right]^\top$.

Theorem 1: Let $\bar{X}$ be a surrogate flowpipe of the surrogate model $\mathcal{F}$ for the set of initial conditions $\mathcal{I}$. Let R^{j*} be computed from the calibration dataset $\mathcal{R}^j_{D_{\text{test}}},\ j \in [n\mathrm{K}]$ with confidence probability $\delta \in (0,1)$ where $\mathcal{R}^j_{D_{\text{test}}}$ is based on i.i.d. test trajectories in D_{test} from the stochastic system M with initial state $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$. Define the inflated surrogate flowpipe,

$$X = \bar{X} \oplus \mathsf{Zonotope}(0, \mathbf{diag}([0_{1\times n},\ R^*])),$$
$$R^* = \left[R^{1*}, \cdots, R^{n\mathrm{K}*}\right].$$

Then, it holds that X is a Δ-confident flowpipe with $\Delta = 1 - n\mathrm{K}(1-\delta)$ for σ_{s_0} from M with initial state $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$.

Proof: Based on the definition of the conformity score, we have $\Pr\left[R^j \leq R^{j*}\right] \geq \delta$. Therefore, we have that

$$\Pr[R^j > R^{j*}] < 1 - \delta.$$

By applying the union bound over probabilities, it follows that

$$\Pr\left[\bigvee_{j=1}^{n\mathrm{K}} \left(R^j > R^{j*}\right)\right] < n\mathrm{K}(1-\delta).$$

The negation of this statement implies that

$$\Pr\left[\bigwedge_{j=1}^{n\mathrm{K}} \left(R^j \leq R^{j*}\right)\right] \geq 1 - n\mathrm{K}(1-\delta). \tag{3}$$

We now denote $\Delta = 1 - n\mathrm{K}(1-\delta)$ so that we can rephrase the above statement as,

$$\Pr\left[\bigwedge_{j=1}^{n\mathrm{K}} \left(\mid e_{j+n}^\top \sigma_{s_0} - \mathsf{F}^j(s_0) \mid \leq R^{j*}\right)\right] \geq \Delta.$$

Next, we define the interval $C_j(s_0) = \left[\mathsf{F}^j(s_0) - R^{j*}\ ,\ \mathsf{F}^j(s_0) + R^{j*}\right]$. Accordingly, we have

$$\Pr\left[\bigwedge_{j=1}^{n\mathrm{K}} \left(e_{j+n}^\top \sigma_{s_0} \in C_j(s_0)\right)\right] \geq \Delta$$

Based on this, we can now see that

$$\Pr\left[\sigma_{s_0} \in \mathsf{Zonotope}\left(\mathcal{F}(s_0), \mathbf{diag}\left([0_{1\times n},\ R^*]\right)\right)\right] \geq \Delta \tag{4}$$

Since $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$ and $\bar{X}$ is a surrogate flowpipe for the surrogate model $\mathcal{F}$ on $\mathcal{I}$, i.e., $s_0 \in \mathcal{I}$ implies $\mathcal{F}(s_0) \in \bar{X}$, we can conclude,

$$\begin{aligned} &\mathsf{Zonotope}\left(\mathcal{F}(s_0), \mathbf{diag}\left([0_{1\times n},\ R^*]\right)\right) \\ &\quad \subset \bar{X} \oplus \mathsf{Zonotope}\left(0, \mathbf{diag}\left([0_{1\times n},\ R^*]\right)\right) = X \end{aligned} \tag{5}$$

This fact implies that $\Pr[\sigma_{s_0} \in X] \geq \Delta$, i.e., X is a Δ-confident flowpipe, which completes the proof. ■

Theorem 1 tells us how to obtain a Δ-confident flowpipe given the K-step datasets D_{train} and D_{test}. We can now compute a lower bound on the minimum size of the calibration dataset that we need given a confidence probability $\Delta \in (0,1)$. Specifically, we note that $\Delta = 1 - n\mathrm{K}(1-\delta)$ is equivalent to $\delta = 1 - \frac{1-\Delta}{n\mathrm{K}}$. The minimum required size L of the calibration dataset has to satisfy $\lceil (L+1)\delta \rceil \leq L$ which gives us the explicit lower bound $L \geq \lceil \frac{1+\delta}{1-\delta} \rceil$. In this work, we defined the residual component-wise, recall Definition 3. As observed in the proof of Theorem 1, we thus had to apply the union bound over all residuals R^j. This may in some cases be conservative, i.e., for large system dimension n or large trajectory horizon K. However, there are possible ways to define a residual in a way that removes this conservatism. For example, in our recent work [36] we show how to obtain tight conformal prediction regions for time series. Applying this method will also result in better data efficiency. We intend to explore this method in the context of this paper in future work and refer the reader to [36] for more details.

[2]Note that we denote the trajectory as a single row vector where component-wise states at each time-stamp are concatenated. Thus, for the i^{th} state variable at time k, j will be equal to $i \cdot k + n$ (skipping the first n component-wise values corresponding to the initial state s_0).

[3]The δ-quantile of a random variable R^j is defined as $\inf\{z \in \mathbb{R} | \Pr[R^j \leq z] \geq \delta\}$ for $\delta \in (0,1)$.

Failure Probability	Conformal inference Run-time	Reachability Run-time
0.10	2.9439 sec	9.8059 sec
0.09	2.8511 sec	9.0679 sec
0.08	2.7367 sec	10.0502 sec
0.07	2.8785 sec	9.7935 sec
0.06	2.9341 sec	9.7988 sec
0.05	2.8783 sec	10.7457 sec
0.04	2.7901 sec	10.0033 sec
0.03	3.096 sec	9.4364 sec
0.02	3.3982 sec	10.0027 sec
0.01	2.8536 sec	9.7945 sec

TABLE I: Adaptive Cruise Control: Computation times of our method for different user-provided failure probabilities ε. The reachability run-time is the overall time for reachability analysis over the surrogate model and constructing the probabilistic flowpipe with conformal inference. The run-time for training a surrogate ReLU model was 2 hours and the run-time for test data-generation was 122 seconds.

IV. Experimental Results

We consider an adaptive cruise controller, a quadcopter, and the Laubloomis benchmarks in [6]. In all case studies, we train 1-step surrogate models that we combine into a K-step surrogate model for trajectory prediction. For reachability analysis of the surrogate model we use the approx-star algorithm from [21] for the Laubloomis case study, while we use the exact-star algorithm from [21] for the adaptive cruise controller and the quadcopter. The underlying system is described by an ordinary differential equation (ODE), potentially affected by noise, and we consider the system at discrete time points. Therefore, we let the control input and the noise be fixed over the sampling time $[k\delta t, (1+k)\delta t], k \in [\mathrm{K}]$. The star-sets that we obtain are $n > 3$ dimensional. Illustration of our results is thus demonstrated through their projection onto state components. This is important to keep in mind when assessing the tightness of the obtained star-sets.

Adaptive Cruise Control. We consider a 6-dimensional system consisting of a leader and a follower vehicle that is equipped with an adaptive cruise controller. The underlying black-box model is described by the ODE model

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\\ \dot{x}_5\\ \dot{x}_6\end{bmatrix} = \begin{bmatrix} x_2\\ x_3\\ -2x_3 - 4 - \mu x_2^2\\ x_5\\ x_6\\ -2x_6 + 2u - \mu x_4^2\end{bmatrix} + v, \quad \mathcal{I} = \left\{ s_0 \;\middle|\; \begin{bmatrix}90\\32\\0\\10\\30\\0\end{bmatrix} \le s_0 \le \begin{bmatrix}110\\32.2\\0\\11\\30.2\\0\end{bmatrix} \right\} \tag{6}$$

where $v \sim \mathcal{N}(0, d^2)$ with standard deviation $d = \mathbf{diag}([1, 0.1, 0.05, 1, 0.1, 0.05])$. The friction coefficient is $\mu = 10^{-4}$. The state components x_1, x_2, x_3 are the position, velocity, and hidden state of the lead vehicle, while the components x_4, x_5, x_6 are the position, velocity, and hidden state of the ego vehicle. The surrogate model is a neural network with ReLU activation functions and with layers $[6, 32, 32, 32, 6]$. The surrogate model is trained from a training dataset generated from Eq. (6) where the control input u is generated by the adaptive cruise controller. To perform conformal inference, we generate $L = 40000$ i.i.d. 50-step trajectories from Eq. (6) with sampling time $\delta t = 0.1\,\mathrm{sec}$ that we collect within the test dataset D_{test}. Fig. 1 shows the projections of our probabilistic flowpipe onto its state components by setting $\varepsilon = 0.01$. To visualize the tightness of the probabilistic flowpipe, we generate 100000 additional i.i.d. trajectories from Eq. (6) and include their projection on their state components in Fig. 1 as well. Table I shows the run time of our experimental results for the range $\varepsilon \in [0.01, 0.1]$.

Quadcopter. We consider a 12-dimensional Quadcopter model that was introduced in [6]. The ODE model for the Quadcopter is shown in Eq. (7). The additive noise $v = [v_1, v_2, \cdots, v_{12}]$ is Gaussian $v \sim \mathcal{N}(0, d^2)$ with standard deviation $d = \mathbf{diag}([0.05 \times \vec{1}_{1\times 6}, 0.01 \times \vec{1}_{1\times 6}])$. The controller is a neural network controller that was presented in [6]. From this model, we generate a test dataset with $L = 40000$ data-points by sampling 50-step trajectories with sampling time $\delta t = 0.1\,\mathrm{sec}$. We also train our surrogate model as a neural network with layers $[12, 20, 20, 20, 12]$ from an additional training dataset. The run time for our data-driven reachability analysis is shown in Table II and the projections of the probabilistic flowpipes onto its state components for $\varepsilon = 0.01$ is shown in Fig. 2.

Laubloomis. Finally, we consider a 7-dimensional system which is known as Laubloomis [6]. The ODE model for Laubloomis is shown in Eq. (8). To provide a comparison with deterministic reachability analysis techniques, we do not add noise to the system in this case so that the system is effectively deterministic. However, note that in our approach we need to sample the initial conditions. We thus generate a test dataset of size $L = 160000$ consisting of 200-step trajectories with sampling time $\delta t = 0.01$.

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\\ \dot{x}_5\\ \dot{x}_6\\ \dot{x}_7\end{bmatrix} = \begin{bmatrix} 1.4x_3 - 0.9x_1\\ 2.5x_5 - 1.5x_2\\ 0.6x_7 - 0.8x_3x_2\\ 2.0 - 1.3x_4x_3\\ 0.7x_1 - 1.0x_4x_5\\ 0.3x_1 - 3.1x_6\\ 1.8x_6 - 1.5x_7x_2\end{bmatrix}, \quad \mathcal{I} = \left\{ s_0 \;\middle|\; \begin{bmatrix}1.05\\0.9\\1.35\\2.25\\0.85\\-0.05\\0.3\end{bmatrix} \le s_0 \le \begin{bmatrix}1.35\\1.2\\1.65\\2.55\\1.15\\0.25\\0.6\end{bmatrix} \right\} \tag{8}$$

Again, we train a surrogate model as a neural network with ReLU activation functions and layers $[7, 20, 20, 20, 7]$. Since the horizon length of trajectories is long, we utilize the approx-star approach instead of the exact-star approach which is more efficient at the cost of conservatism. The simulation of the results for $\varepsilon = 0.01$ is also demonstrated in Fig.3. The reachability run time for our data driven probabilistic approach for a range of failure probabilities $\varepsilon \in [0.01, 0.1]$ is shown in Table III. Finally, we utilize the CORA toolbox [37] to compare to our results, which is shown in Fig.3. We emphasize that the CORA toolbox is only applicable to deterministic systems and assumes access to the model, while our approach only assumes availability of data.

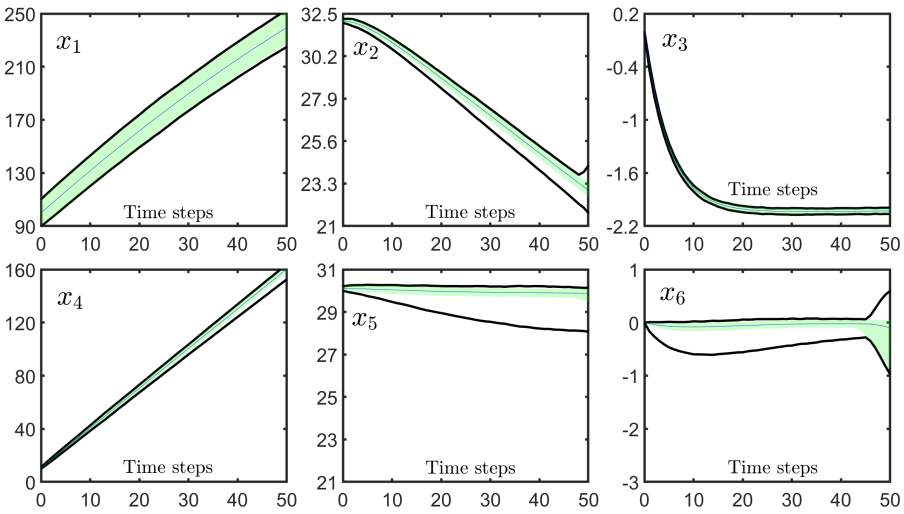

Fig. 1: Adaptive Cruise Control: The black lines indicate the probabilistic flowpipe computed using the exact-star technique on the learned surrogate model combined with conformal inference with failure probability $\varepsilon = 0.01$. The green shaded areas and blue lines are computed from 100000 random trajectories from the ODE model. The green area shows the maximum and minimum value of the trajectory components over this dataset, and the blue line represents their average value.

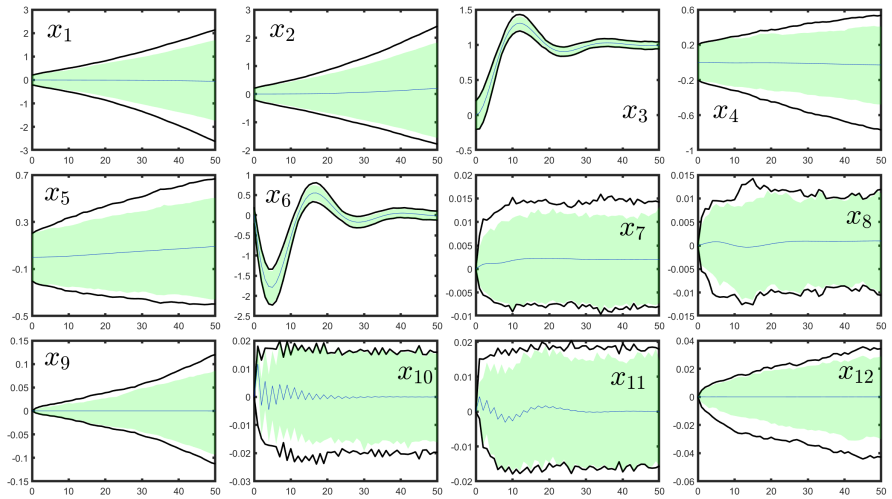

Fig. 2: Quadcopter: The black lines indicate the probabilistic flowpipe computed using the exact-star technique on the learned surrogate model combined with conformal inference with failure probability $\varepsilon = 0.01$. The green shaded areas and blue lines are computed from 100000 random trajectories from the ODE model. The green area shows the maximum and minimum value of trajectory components over this dataset, and the blue line represents their average value.

Failure Probability	Conformal inference Run-time	Reachability Run-time	Failure Probability	Conformal inference Run-time	Reachability Run-time
0.10	2.475 sec	7.3779 sec	0.05	2.4762 sec	7.5 sec
0.09	2.4754 sec	6.8911 sec	0.04	2.4822 sec	7.3542 sec
0.08	2.5894 sec	7.2825 sec	0.03	2.5316 sec	7.5548 sec
0.07	2.6046 sec	7.5339 sec	0.02	2.364 sec	7.2504 sec
0.06	2.4674 sec	7.1876 sec	0.01	3.5858 sec	7.4376 sec

TABLE II: Quadcopter: Computation times of our method for different user-provided failure probabilities ε. The reachability run-time is the overall time for reachability analysis over the surrogate model and constructing the probabilistic flowpipe with conformal inference. The data generation time for the test dataset is 273 sec and the run time for training the ReLU model on training dataset is 2 hours and 25 minutes.

V. Conclusion

We proposed a data-driven approach to analyze the reachability of stochastic dynamical systems. Particularly, we studied stochastic dynamical systems when no mathematical model of the system is available, and the only information available to us is data observed from the system. We showed how to compute probabilistic reach sets, so called probabilistic flowpipes, for this system from data. Probabilistic flowpipes ensure that the probability of a new trajectory not being in the flowpipe is upper bounded by a user-defined threshold. Our approach consists of first learning a surrogate model of the system from a training dataset. We then used the surrogate model to perform reachability analysis over it using existing tools for deterministic reachability analysis. To quantify

Failure Probability	Conformal inference Run-time	Reachability Run-time	Failure Probability	Conformal inference Run-time	Reachability Run-time
0.10	89.9305 sec	93.8766 sec	0.05	71.3700 sec	49.7159 sec
0.09	72.0818 sec	53.0224 sec	0.04	67.8528 sec	50.1378 sec
0.08	74.2623 sec	50.0363 sec	0.03	72.8819 sec	50.1828 sec
0.07	71.7105 sec	49.7770 sec	0.02	85.9235 sec	49.7792 sec
0.06	69.4149 sec	50.7159 sec	0.01	91.9276 sec	92.9743 sec

TABLE III: Laubloomis: Computation times of our method for different user-provided failure probabilities ε. The reachability run-time is the overall time for reachability analysis over the surrogate model and constructing the probabilistic flowpipe with conformal inference. The data generation time for the test dataset is 19 minutes, and the run time for training the ReLU model on the training dataset is 2 hours and 30 minutes.

$$
\begin{cases}
\dot{x}_1 = \cos(x_8)\cos(x_9)x_4 + (\sin(x_7)\sin(x_8)\cos(x_9) - \cos(x_7)\sin(x_9))x_5 \\
\quad +(\cos(x_7)\sin(x_8)\cos(x_9) + \sin(x_7)\sin(x_9))x_6 + v_1 \\
\dot{x}_2 = \cos(x_8)\sin(x_9)x_4 + (\sin(x_7)\sin(x_8)\sin(x_9) + \cos(x_7)\cos(x_9))x_5 \\
\quad +(\cos(x_7)\sin(x_8)\sin(x_9) - \sin(x_7)\cos(x_9))x_6 + v_2 \\
\dot{x}_3 = \sin(x_8)x_4 - \sin(x_7)\cos(x_8)x_5 - \cos(x_7)\cos(x_8)x_6 + v_3 \\
\dot{x}_4 = x_{12}x_5 - x_{11}x_6 - 9.81\sin(x_8) + v_4 \\
\dot{x}_5 = x_{10}x_6 - x_{12}x_4 + 9.81\cos(x_8)\sin(x_7) + v_5 \\
\dot{x}_6 = x_{11}x_4 - x_{10}x_5 + 9.81\cos(x_8)\cos(x_7) - 9.81 - u_1/1.4 + v_6 \\
\dot{x}_7 = x_{10} + (\sin(x_7)(\sin(x_8)/\cos(x_8)))x_{11} + (\cos(x_7)(\sin(x_8)/\cos(x_8)))x_{12} + v_7 \\
\dot{x}_8 = \cos(x_7)x_{11} - \sin(x_7)x_{12} + v_8 \\
\dot{x}_9 = (\sin(x_7)/\cos(x_8))x_{11} + (\cos(x_7)/\cos(x_8))x_{12} + v_9 \\
\dot{x}_{10} = -0.9259x_{11}x_{12} + 18.5185u_2 + v_{10} \\
\dot{x}_{11} = 0.9259x_{10}x_{12} + 18.5185u_3 + v_{11} \\
\dot{x}_{12} = v_{12}
\end{cases}
$$

$$
\mathcal{I} = \left\{ s_0 \;\middle|\; \begin{bmatrix} -0.2 \\ -0.2 \\ -0.2 \\ -0.2 \\ -0.2 \\ -0.2 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \leq s_0 \leq \begin{bmatrix} 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right\} \tag{7}
$$

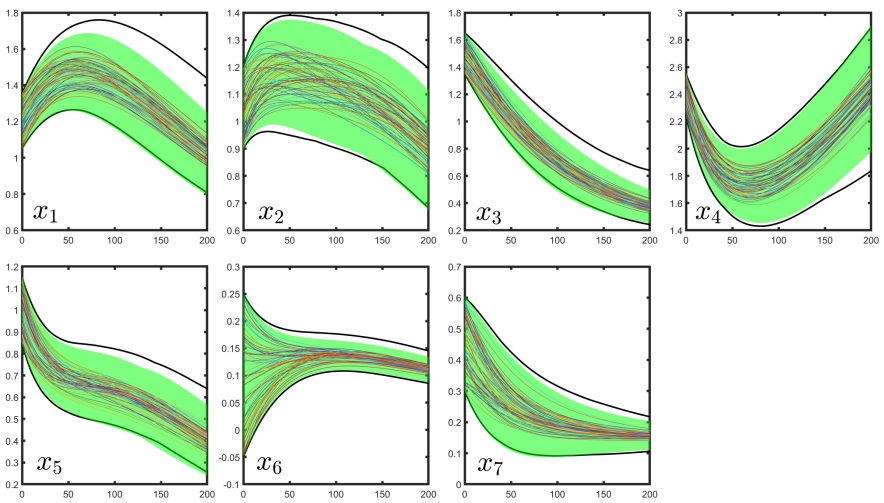

Fig. 3: Laubloomis: Shows our data driven reachability analysis from 200-step dataset along with 100 random trajectories generated from M. We also included the reachability analysis from CORA toolbox that is based on the ideally known model depicted as green regions. The bounds in black line shows the reachability analysis with approx-star technique combined with conformal inference with prescribed failure probability $\varepsilon = 0.01$.

the error between the surrogate model and the underlying unknown system, we finally use conformal inference on a test dataset. We illustrated our approach using three case studies.

REFERENCES

[1] A. P. Vinod and M. M. Oishi, "Stochastic reachability of a target tube: Theory and computation," *Automatica*, vol. 125, p. 109458, 2021.

[2] A. Abate, M. Prandini, J. Lygeros, and S. Sastry, "Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems," *Automatica*, vol. 44, no. 11, pp. 2724–2734, 2008.

[3] A. Abate, S. Amin, M. Prandini, J. Lygeros, and S. Sastry, "Computational approaches to reachability analysis of stochastic hybrid systems," in *International Workshop on Hybrid Systems: Computation and Control*. Springer, 2007, pp. 4–17.

[4] Y. Yang, J. Zhang, K.-q. Cai, and M. Prandini, "A stochastic reachability analysis approach to aircraft conflict detection and resolution," in *2014 IEEE Conference on Control Applications (CCA)*, 2014, pp. 2089–2094.

[5] C. Huang, J. Fan, W. Li, X. Chen, and Q. Zhu, "Reachnn: Reachability analysis of neural-network controlled systems," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 5s, pp. 1–22, 2019.

[6] C. Huang, J. Fan, X. Chen, W. Li, and Q. Zhu, "Polar: A polynomial arithmetic framework for verifying neural-network controlled systems," in *International Symposium on Automated Technology for Verification and Analysis*. Springer, 2022, pp. 414–430.

[7] X. Chen, E. Ábrahám, and S. Sankaranarayanan, "Flow*: An analyzer for non-linear hybrid systems," in *Computer Aided Verification: 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings 25*. Springer, 2013, pp. 258–263.

[8] S. Dutta, X. Chen, S. Jha, S. Sankaranarayanan, and A. Tiwari,

"Sherlock-a tool for verification of neural network feedback systems: demo abstract," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, 2019, pp. 262–263.

[9] X. Koutsoukos and D. Riley, "Computational methods for reachability analysis of stochastic hybrid systems," in *Hybrid Systems: Computation and Control: 9th International Workshop, HSCC 2006, Santa Barbara, CA, USA, March 29-31, 2006. Proceedings 9*. Springer, 2006, pp. 377–391.

[10] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-jacobi reachability: A brief overview and recent advances," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 2242–2253.

[11] L. Bortolussi and G. Sanguinetti, "A statistical approach for computing reachability of non-linear and stochastic dynamical systems," in *International Conference on Quantitative Evaluation of Systems*. Springer, 2014, pp. 41–56.

[12] M. Kwiatkowska, G. Norman, and D. Parker, "Stochastic model checking," *Formal Methods for Performance Evaluation: 7th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2007, Bertinoro, Italy, May 28-June 2, 2007, Advanced Lectures 7*, pp. 220–270, 2007.

[13] A. Legay, A. Lukina, L. M. Traonouez, J. Yang, S. A. Smolka, and R. Grosu, "Statistical model checking," in *Computing and software science: state of the art and perspectives*. Springer, 2019, pp. 478–504.

[14] A. Devonport, F. Yang, L. El Ghaoui, and M. Arcak, "Data-driven reachability analysis with christoffel functions," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 5067–5072.

[15] A. Alanwar, A. Koch, F. Allgöwer, and K. H. Johansson, "Data-driven reachability analysis from noisy data," *IEEE Transactions on Automatic Control*, 2023.

[16] A. Devonport and M. Arcak, "Data-driven reachable set computation using adaptive gaussian process classification and monte carlo methods," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 2629–2634.

[17] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, "A general safety framework for learning-based control in uncertain robotic systems," *IEEE Transactions on Automatic Control*, vol. 64, no. 7, pp. 2737–2752, 2018.

[18] A. Lin and S. Bansal, "Generating formal safety assurances for high-dimensional reachability," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 10 525–10 531.

[19] C. Fan, B. Qi, S. Mitra, and M. Viswanathan, "Dryvr: Data-driven verification and compositional reasoning for automotive systems," in *International Conference on Computer Aided Verification*. Springer, 2017, pp. 441–461.

[20] N. Matni and S. Tu, "A tutorial on concentration bounds for system identification," in *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 2019, pp. 3741–3749.

[21] H.-D. Tran, X. Yang, D. Manzanas Lopez, P. Musau, L. V. Nguyen, W. Xiang, S. Bak, and T. T. Johnson, "Nnv: the neural network verification tool for deep neural networks and learning-enabled cyber-physical systems," in *Computer Aided Verification: 32nd International Conference, CAV 2020, Los Angeles, CA, USA, July 21–24, 2020, Proceedings, Part I*. Springer, 2020, pp. 3–17.

[22] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic learning in a random world*. Springer, 2005, vol. 29.

[23] J. Lei and L. Wasserman, "Distribution-free prediction bands for non-parametric regression," *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, pp. 71–96, 2014.

[24] L. Bortolussi, F. Cairoli, N. Paoletti, S. A. Smolka, and S. D. Stoller, "Neural predictive monitoring," in *Runtime Verification: 19th International Conference, RV 2019, Porto, Portugal, October 8–11, 2019, Proceedings 19*. Springer, 2019, pp. 129–147.

[25] F. Cairoli, N. Paoletti, and L. Bortolussi, "Conformal quantitative predictive monitoring of stl requirements for stochastic processes," in *Proceedings of the 26th ACM International Conference on Hybrid Systems: Computation and Control*, 2023, pp. 1–11.

[26] L. Lindemann, X. Qin, J. V. Deshmukh, and G. J. Pappas, "Conformal prediction for stl runtime verification," in *Proceedings of the ACM/IEEE 14th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2023)*, 2023, pp. 142–153.

[27] X. Qin, Y. Xia, A. Zutshi, C. Fan, and J. V. Deshmukh, "Statistical verification of cyber-physical systems using surrogate models and conformal inference," in *2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2022, pp. 116–126.

[28] A. B. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis," in *Hybrid Systems: Computation and Control: Third International Workshop, HSCC 2000 Pittsburgh, PA, USA, March 23–25, 2000 Proceedings*. Springer, 2002, pp. 202–214.

[29] J. Lei, M. G'Sell, A. Rinaldo, R. J. Tibshirani, and L. Wasserman, "Distribution-free predictive inference for regression," *Journal of the American Statistical Association*, vol. 113, no. 523, pp. 1094–1111, 2018.

[30] A. N. Angelopoulos and S. Bates, "A gentle introduction to conformal prediction and distribution-free uncertainty quantification," *arXiv preprint arXiv:2107.07511*, 2021.

[31] L. Lindemann, M. Cleaveland, G. Shim, and G. J. Pappas, "Safe planning in dynamic environments using conformal prediction," *IEEE Robotics and Automation Letters*, 2023.

[32] R. Luo, S. Zhao, J. Kuck, B. Ivanovic, S. Savarese, E. Schmerling, and M. Pavone, "Sample-efficient safety assurances using conformal prediction," in *Algorithmic Foundations of Robotics XV: Proceedings of the Fifteenth Workshop on the Algorithmic Foundations of Robotics*. Springer, 2022, pp. 149–169.

[33] R. J. Tibshirani, R. Foygel Barber, E. Candes, and A. Ramdas, "Conformal prediction under covariate shift," *Advances in neural information processing systems*, vol. 32, 2019.

[34] H.-D. Tran, D. Manzanas Lopez, P. Musau, X. Yang, L. V. Nguyen, W. Xiang, and T. T. Johnson, "Star-based reachability analysis of deep neural networks," in *Formal Methods–The Next 30 Years: Third World Congress, FM 2019, Porto, Portugal, October 7–11, 2019, Proceedings 3*. Springer, 2019, pp. 670–686.

[35] H.-D. Tran, F. Cai, M. L. Diego, P. Musau, T. T. Johnson, and X. Koutsoukos, "Safety verification of cyber-physical systems with reinforcement learning control," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 5s, pp. 1–22, 2019.

[36] M. Cleaveland, I. Lee, G. J. Pappas, and L. Lindemann, "Conformal prediction regions for time series using linear complementarity programming," *arXiv preprint arXiv:2304.01075*, 2023.

[37] M. Althoff, "An introduction to cora 2015." *ARCH@ CPSWeek*, vol. 34, pp. 120–151, 2015.