Statistical Reachability Analysis of Stochastic Cyber–Physical Systems Under Distribution Shift

Navid Hashemi, Lars Lindemann, *Member, IEEE*, and Jyotirmoy V. Deshmukh

***Abstract*—Reachability analysis is a popular method to give safety guarantees for stochastic cyber–physical systems (SCPSs) that takes in a symbolic description of the system dynamics and uses set-propagation methods to compute an overapproximation of the set of reachable states over a bounded time horizon. In this article, we investigate the problem of performing reachability analysis for an SCPS that does not have a symbolic description of the dynamics, but instead is described using a digital twin model that can be simulated to generate system trajectories. An important challenge is that the simulator implicitly models a probability distribution over the set of trajectories of the SCPS; however, it is typical to have a sim2real gap, i.e., the actual distribution of the trajectories in a deployment setting may be shifted from the distribution assumed by the simulator. We thus propose a statistical reachability analysis technique that, given a user-provided threshold $1-\epsilon$, provides a set that guarantees that any trajectory during deployment lies in this set with probability not smaller than this threshold. Our method is based on three main steps: 1) learning a deterministic surrogate model from sampled trajectories; 2) conducting reachability analysis over the surrogate model; and 3) employing robust conformal inference (CI) using an additional set of sampled trajectories to quantify the surrogate model's distribution shift with respect to the deployed SCPS. To counter conservatism in reachable sets, we propose a novel method to train surrogate models that minimizes a quantile loss term (instead of the usual mean squared loss), and a new method that provides tighter guarantees using CI using a normalized surrogate error. We demonstrate the effectiveness of our technique on various case studies.**

***Index Terms*—Sim2real gap, statistical reachability analysis, stochastic cyber–physical systems (SCPSs).**

I. Introduction

SAFETY-CRITICAL cyber–physical systems operate in highly dynamic and uncertain environments. It is common to model such systems as *stochastic dynamical systems* where given an initial configuration (or state) of the system, system parameter values, and a sequence of exogenous inputs to the system, a *simulator* can provide a system trajectory. Several executions of the simulator can generate a sample distribution of the system trajectories, and such a distribution can then be studied with the goal of analyzing safety and performance specifications of the system. In safety verification analysis, we are interested in checking if any system trajectory can reach an *unsafe* state. A popular approach for safety verification considers only bounded-time safety properties using (bounded-time) *reachability analysis* [1], [2], [3], [4], [5]. Here, the typical assumption is that the symbolic dynamics of the simulator (i.e., the equations it uses to provide the updated state from a previous state and stimuli) are known. Most reachability analysis methods rely on a deterministic description of the symbolic dynamics and use set-propagation methods to compute a *flowpipe* or an overapproximation of the set of states reachable over a specified time horizon. Other methods allow the system dynamics to be stochastic, but rely on linearity of the dynamics to propagate distributions over initial states/parameters to compute probabilistic reach sets [6], [7], [8], [9].

However, for complex cyber–physical systems, dynamical models may be highly nonlinear or hybrid with artifacts, such as look-up tables, learning-enabled components, and proprietary closed-box functions making the symbolic dynamics either unavailable, or difficult for existing (symbolic) reachability analysis tools to analyze them. To address this issue, we pursue the idea of *model-free analysis*, where the idea is to compute reachable sets for the system from only sampled system trajectories [10], [11]. The main idea of data-driven reachability analysis in [10] consists of the following main steps: *Step 1:* sample system trajectories based on a user-specified distribution on a parametric set of system uncertainties (such as the set of initial states); *Step 2:* train a data-driven surrogate model to predict the next K states from a given state (for example, a neural network (NN)-based model); *Step 3:* perform set-propagation-based reachability analysis using the surrogate dynamics; and *Step 4:* inflate the computed flowpipe with a surrogate error term that guarantees that any actually reached state is within the inflated reach set with probability not smaller than a user-provided threshold.

There are three main challenges in this overall scheme: 1) in [10], a simple training loss based on minimizing the mean square error between the surrogate model and the actual system is used. This may lead to the error distribution to have a heavy tail, which in turn leads to conservatism in the inflated reach set; 2) the approach in [10] uses the uncertainty quantification technique of conformal inference (CI) to construct the inflated flowpipes, but quantifies surrogate error per trajectory component (i.e., per state dimension and per

Manuscript received 28 July 2024; accepted 29 July 2024. Date of current version 6 November 2024. This work was supported in part by the National Science Foundation through the CAREER Award under Grant SHF-2048094, Grant CNS-1932620, Grant FMitF-1837131, and Grant CCF-SHF-1932620; in part by the Airbus Institute for Engineering Research; in part by the Toyota Research and Development; and in part by the Siemens Corporate Research through the USC Center for Autonomy and AI. This article was recommended by Associate Editor S. Dailey. *(Corresponding author: Jyotirmoy V. Deshmukh.)*

The authors are with the Department of Computer Science, University of Southern California, Los Angeles, CA 90089 USA (e-mail: navidhas@usc.edu; llindema@usc.edu; jdeshmuk@usc.edu).

Digital Object Identifier 10.1109/TCAD.2024.3438072

trajectory time-step). These per-component-wise probabilistic guarantees are then combined using union bounding, i.e., using that $P(A \cup B) \leq P(A) + P(B)$, leading to conservatism. This is because requiring a $1-\epsilon$ probability threshold on the inflated reach set requires stricter probability thresholds in the CI step per component, i.e., thresholds $1-\epsilon'$ with $\epsilon' = (\epsilon/nK)$, where n is the number of dimensions and K is the number of time-steps in the trajectory. A stricter probability threshold induces a larger uncertainty set, which implies greater conservatism; and 3) the most significant real-world challenge is that the surrogate model is usually learned based on the trajectories sampled from the simulator, and thus distributed according to the assumptions on stochasticity made by the simulator. However, the actual trajectory distribution in the deployed system may change. Typically, such distribution shifts can be quantified using divergence measures, such as an f-divergence or the Wasserstein distance [12].

To address these challenges, we propose a robust and efficient approach to computing probabilistic reach sets for stochastic systems, with the following main contributions: 1) we propose novel training algorithms to obtain surrogate models to forecast trajectories from sampled initial states (or other model parameters). Instead of minimizing the mean square loss between predicted trajectories and the training trajectories, we allow minimizing an arbitrary quantile of the loss function. This provides our models with better overall predictive performance over the entire trajectory space (i.e., over different state dimensions and time steps); 2) similar to [10], we utilize CI to quantify prediction uncertainty. However, inspired by work in [13], we compute the maximum of the weighted residual errors to compute the nonconformity score to use with CI which has the effect of normalizing component-wise residuals. In contrast to [13], which solves a linear complementarity problem to compute these weights, we obtain these weights when training the surrogate model using gradient descent and backpropagation; and 3) finally, to address distribution shifts, we use techniques from robust CI [14]. Our analysis is motivated by [15] and valid for all trajectory distributions corresponding to real-world environments that are close to the original trajectory distribution used for training the surrogate model; here, the proximity is measured by a certain f-divergence metric [16].

We show that our training procedure and the use of the max-based nonconformity score noticeably enhances data efficiency and significantly improves the conservatism in reachability analysis. This improvement in data efficiency is the key factor that enables us to efficiently incorporate robust CI in our reachability analysis. We empirically validate our algorithms on challenging benchmark problems from the cyber–physical systems community [17], and demonstrate considerable improvement over prior work.

Related Work:

Reachability Analysis for Stochastic Systems With Known Dynamics: Reachability analysis is a widely studied topic and typically assumes access to the system's underlying dynamics, and the proposed guarantees are valid only on the given model dynamics. Lin and Bansal [18] proposed DeepReach, a method using neural PDE solvers for Hamilton–Jacobi method-based reachability analysis in high-dimensional systems. While it incorporates neural methods for reachability analysis, it still requires access to the system dynamics. Alanwar et al. [19] identified Markovian stochastic dynamics from data through specific parametric models, such as linear or polynomial, followed by reachability analysis on the identified models. In contrast, our method employs NNs, which are not confined to Markovian dynamics. The approach in [20] is an algorithm that sequentially linearizes the dynamics and uses constrained zonotopes for set representation and computation. Bortolussi and Sanguinetti [21] developed a method utilizing Gaussian Processes and statistical techniques to compute reachable sets of dynamical systems with uncertain initial conditions or parameters, providing confidence bounds for the reconstruction and bounding the reachable set with probabilistic confidence, extending to uncertain stochastic models.

Huang et al. [22] introduced a scalable method utilizing Fourier transforms to compute forward stochastic reach probability measures and sets for uncontrolled *linear systems* with affine disturbances. Similar approaches are explored in [6] and [23] for stochastic reachability analysis of linear, potentially time-varying, discrete-time systems. A constructive method utilizing convex optimization to determine and compute probabilistic reachable and invariant sets for linear discrete-time systems under stochastic disturbances is introduced in [24]. We note that most existing techniques are for systems with linear dynamics, while we permit arbitrary stochastic dynamics. In Thorpe et al. [25], a method utilizing conditional distribution embeddings and random Fourier features is presented to efficiently compute stochastic reachability safety probabilities for high-dimensional stochastic dynamical systems without prior knowledge of system structure. We note that this work does not provide finite-data probability guarantees as we do, but asymptotically converge to the exact reachset.

Probabilistic Guarantees and Reachability Analysis for Unknown Stochastic Systems: Recent work has studied computation of reachable sets with probabilistic guarantees directly from data. Devonport et al. [26] employed level sets of Christoffel functions [27], [28] to achieve probabilistic reach sets for general nonlinear systems. Specifically, let $v_d(\mathbf{x})$ denote the vector of monomials up to degree d, and let M denote the empirical moment matrix obtained by computing the expected value of $v_d(\mathbf{x})^\top v_d(\mathbf{x})$ by sampling over the set of reachable states. An empirical inverse Christoffel function $\Lambda^{-1}(\mathbf{x})$ is then defined as $v_d(\mathbf{x})^\top M^{-1} v_d(\mathbf{x})$. The main idea in [29] and [30] is to empirically determine $\Lambda^{-1}(\mathbf{x})$ and give probabilistic bounds using the volume of the actual reachset contained in the sublevel sets of $\Lambda^{-1}(\mathbf{x})$. Tebjou et al. [30] extended the method proposed in [26] by including CI. A key challenge of this approach is estimating the moment matrix M from data, which may not scale with increasing state dimension n and user-selected degree d, as the dimension of M is $\binom{n+d}{d}$, and the approach requires inverting M.

Devonport and Arcak [29] used a Gaussian process-based classifier to distinguish reachable from unreachable states and approximate the reachset. However, the approach requires adaptive sampling of initial states, which may require solving

high-dimensional optimization problems. They also propose an interval abstraction of the reachset, which, though it provides sample complexity bounds, can be overly conservative and computationally costly in high-dimensional systems. The method in [31] assumes partial knowledge of the model and leverages data to handle Lipschitz-continuous state-dependent uncertainty; their reachability analysis combines probabilistic and worst-case analysis. Finally, the work presented [32] combines simulation-guided reachability analysis with data-driven techniques, utilizing a discrepancy function estimated from system trajectories, which can be challenging to obtain.

Reachability Analysis for NNs: Recent approaches have tackled the challenge of determining the output range of an NN. These methods aim to compute an interval or a box (a vector of intervals) that encompasses the outputs of a given NN. Katz et al. [33] introduced Reluplex, an SMT-based approach that extends the simplex algorithm to handle ReLU constraints. Huang et al. [22] employed a refinement-by-layer technique to verify the presence or absence of adversarial examples in the vicinity of a specific input. Dutta et al. [4] proposed an efficient method using mixed-integer linear programming to compute the range of an NN featuring only ReLU activation functions. Tran et al. [34] proposed star-sets that offer similar expressiveness as hybrid zonotopes and are used to provide approximate and exact reachability of feed-forward ReLU NNs. In our setting, this method was the most applicable.

II. Problem Statement and Preliminaries

Notation: We use bold letters to represent vectors and vector-valued functions, while calligraphic letters denote sets and distributions. The set $\{1, 2, \ldots, n\}$ is denoted as $[n]$. The Minkowski sum is indicated by $\oplus$. We use $x \sim \mathcal{X}$ to denote that the random variable x is drawn from the distribution $\mathcal{X}$.

Stochastic Dynamical Systems: We consider discrete-time stochastic dynamical systems. While it is typical to describe such systems using symbolic equations that describe how the system evolves over time, we instead simply model the system as a *stochastic process*. In other words, let $S_0, \ldots, S_K$ be a set of $K+1$ random vectors indexed by times $0, \ldots, K$. We assume that for all times k, each S_k takes values from the set of states $\mathcal{S} \subseteq \mathbb{R}^n$. A realization of the stochastic process, or the *system trajectory* is a sequence of values $s_0, \ldots, s_K$, denoted as $\sigma_{s_0}^{\text{real}}$. The joint distribution over $S_0, \ldots, S_K$ is called the *trajectory distribution* $\mathcal{D}_{S,K}^{\text{real}}$ of the system, and the marginal distribution of S_0 is called the *initial state distribution* $\mathcal{W}$. We assume that the initial state distribution $\mathcal{W}$ has support over a compact set of initial states $\mathcal{I}$, i.e., we assume that $\mathcal{W}$ is such that $\Pr[s_0 \notin \mathcal{I}] = 0$. For example, such a stochastic dynamical system could describe a Markovian process, where for any $k \geq 1$, the distribution of S_k only depends on the realization of S_{k-1} and not the values taken at any past time. However, it is worth noting that the techniques presented in this article can be applied to systems with non-Markovian dynamics.

In the remainder of this article, we largely focus on just the system trajectories, so we abuse notation to denote $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$ to signify that s_0 is a value sampled from $\mathcal{I}$ using the initial state distribution $\mathcal{W}$.[1] Similarly, $\sigma_{s_0}^{\text{real}} \sim \mathcal{D}_{S,K}^{\text{real}}$ is used to denote the sampling of a trajectory from the trajectory distribution.

Quantification of Distribution Shift: In practice, we usually do not have knowledge of the distribution $\mathcal{D}_{S,K}^{\text{real}}$. However, one may have access to trajectories sampled from a distribution $\mathcal{D}_{S,K}^{\text{sim}}$ that is close to $\mathcal{D}_{S,K}^{\text{real}}$, e.g., a simulator. Given a distribution $\mathcal{D}$, we use the notation $\mathcal{P}(\mathcal{D})$ to denote a set of distributions *close* to $\mathcal{D}$, where the notion of proximity is defined using a suitable divergence measure or metric quantifying distance between distributions. Common examples include f-divergence measures (such as KL-divergence and total variation distance) and metrics, such as the Wasserstein distance [12], [35]. In this article, we assume that $\mathcal{D}_{S,K}^{\text{sim}}$ comes from the ambiguity set $\mathcal{P}(\mathcal{D}_{S,K}^{\text{sim}})$ that is centered at $\mathcal{D}_{S,K}^{\text{sim}}$ using f-divergence balls around $\mathcal{D}_{S,K}^{\text{sim}}$ [35].[2] Given a convex function $f : \mathbb{R} \to \mathbb{R}$ satisfying $f(1) = 0$ and $f(z) = +\infty$ for $z < 0$, the f-divergence [16] between the probability distributions $\mathcal{D}_{S,K}^{\text{sim}}$ and $\mathcal{D}_{S,K}^{\text{real}}$ that both have support $\mathcal{Z}$ is

$$D_f\left(\mathcal{D}_{S,K}^{\text{real}} \| \mathcal{D}_{S,K}^{\text{sim}}\right) = \int_{\mathcal{Z}} f\left(\frac{\mathbf{d}\mathcal{D}_{S,K}^{\text{real}}}{\mathbf{d}\mathcal{D}_{S,K}^{\text{sim}}}\right) \mathbf{d}\mathcal{D}_{S,K}^{\text{sim}}.$$

Here, the argument of f is the Radon–Nikodym derivative of $\mathcal{D}_{S,K}^{\text{sim}}$ w.r.t. $\mathcal{D}_{S,K}^{\text{real}}$. We define the set $\mathcal{P}_{f,\tau}(\mathcal{D}_{S,K}^{\text{sim}})$ as an f-divergence ball of radius $\tau \geq 0$ around $\mathcal{D}_{S,K}^{\text{sim}}$ as

$$\mathcal{P}_{f,\tau}\left(\mathcal{D}_{S,K}^{\text{sim}}\right) = \left\{\mathcal{D}_{S,K}^{\text{real}} \mid D_f\left(\mathcal{D}_{S,K}^{\text{real}} \| \mathcal{D}_{S,K}^{\text{sim}}\right) \leq \tau\right\}.$$

The radius τ and the function f are both user-specified parameters that quantify the distribution shift between $\mathcal{D}_{S,K}^{\text{real}}$ and $\mathcal{D}_{S,K}^{\text{sim}}$ that we have to account for in our reachability analysis. Specifically, we have to perform reachability analysis for random trajectories $\sigma_{s_0}^{\text{real}} \sim \mathcal{D}_{S,K}^{\text{real}}$ for all $\mathcal{D}_{S,K}^{\text{real}} \in \mathcal{P}_{f,\tau}(\mathcal{D}_{S,K}^{\text{sim}})$.

CI: CI [36], [37], [38] is a data-efficient statistical tool proposed for quantifying uncertainty, particularly valuable for assessing the uncertainty in predictions made by machine learning models [39], [40].

Consider a set of random variables $z_1, z_2, \ldots, z_{m+1}$ where $z_i = (x_i, y_i) \in \mathbb{R}^n \times \mathbb{R}$ for $i \in [m+1]$. Assume that $z_1, z_2, \ldots, z_{m+1}$ are independent and identically distributed (i.i.d.). Let $\mu(x_i)$ be a predictor that estimates outputs y_i from inputs x_i. With a predefined miscoverage level $\epsilon \in (0, 1)$, CI enables computation of a threshold $d > 0$ and a probabilistic prediction interval $C(x_{m+1}) = [\mu(x_{m+1}) - d,\ \mu(x_{m+1}) + d] \subseteq \mathbb{R}$ for y_{m+1} that guarantees that $\Pr[y_{m+1} \in C(x_{m+1})] \geq 1 - \epsilon$. To compute the threshold d, we reason over the *empirical distribution of the residual errors* between the predictor and the ground truth data. Let $R_i \coloneqq |y_i - \mu(x_i)|$ be the residual error between y_i and $\mu(x_i)$ for $i \in [m+1]$. Since the random variables $z_1, z_2, \ldots, z_{m+1}$ are i.i.d., the residuals $R_1, \ldots, R_{m+1}$ are also i.i.d. If m satisfies $\ell \coloneqq \lceil (m+1)(1-\epsilon) \rceil \leq m$, then we take the ℓ^{th} smallest error among these m values which is equivalent to

$$R_{1-\epsilon}^{*} = \text{Quantile}_{1-\epsilon}^{c}\{R_1, \ldots, R_m, \infty\} \tag{1}$$

i.e., the $(1-\epsilon)$-quantile over $R_1, \ldots, R_m, \infty$, see [41].

[1] $\mathcal{W}$ is assumed to be uniform or truncated Gaussian distributed in practice.

[2] Examples of f include $f(z) = z\log(z)$, which induces the KL-divergence and $f(z) = (1/2) \mid z - 1 \mid$, which induces the total variation distance.

CI uses this quantile to obtain the probability guarantee $\Pr[R_{m+1} \le R^*_{1-\epsilon}] \ge (1-\epsilon)$, see [36], [41]. For the choice of $R_i := |y_i - \mu(x_i)|$, this can be rewritten as

$$\Pr\big[y_{m+1} \in \big[\mu(x_{m+1}) - R^*_{1-\epsilon}, \mu(x_{m+1}) + R^*_{1-\epsilon}\big]\big] \ge 1-\epsilon. \quad (2)$$

The guarantees in (2) are marginal,[3] i.e., over the randomness in $R_{m+1}, R_1, R_2, \ldots, R_m$. Note that $R^*_{1-\epsilon}$ is a provable upper bound for the $(1-\epsilon)$-quantile[4] of the error distribution.

Robust CI: Unlike CI, which assumes the data-point z_{m+1} is sampled from the same distribution as the calibration samples $z_i, i \in [m]$, robust CI relaxes this assumption and allows z_{m+1} to be sampled from a different distribution. Let us denote the distribution of z_i for $i \in [m]$ as U and the distribution of z_{m+1} as V. As illustrated before, the residual R_i is a distribution and defined as a function of z_i. Let us denote the distribution of R_i for $i \in [m]$ with P and the distribution of R_{m+1} with Q. Further, assume Q is in $\mathcal{P}_{f,\tau}(P)$. Utilizing the results from [14] that assumes the distribution of residual R_{m+1} is within an f-divergence ball of the distributions for $R_1, \ldots, R_m$ with radius $\tau \ge 0$, for the miscoverage level $\epsilon \in (0,1)$, we obtain

$$\Pr\big[R_{m+1} \le R^*_{\mathbf{1}-\epsilon,\tau}\big] \ge 1-\epsilon$$

where $R^*_{1-\epsilon,\tau} = \text{Quantile}^c_{(1-\bar\epsilon)}\{R_1, \ldots, R_m, \infty\}$ is a *robust* $(1-\epsilon)$-quantile that is equivalent to the $(1-\bar\epsilon)$-quantile. We refer to $\bar\epsilon$ as the adjusted miscoverage level which is computed as $\bar\epsilon = 1 - g^{-1}_{f,\tau}(1-\epsilon_m)$ where ϵ_m is obtained as the solution of a series of convex optimizations problems as[5]

$$\begin{aligned}
\epsilon_m &= 1 - g_{f,\tau}\left(\left(1+\frac{1}{m}\right) g^{-1}_{f,\tau}(1-\epsilon)\right),\\
g_{f,\tau}(\beta) &= \inf\left\{z \in [0,1] \,\middle|\, \beta f\left(\frac{z}{\beta}\right) + (1-\beta) f\left(\frac{1-z}{1-\beta}\right) \le \tau\right\}\\
g^{-1}_{f,\tau}(\gamma) &= \sup\big\{\beta \in (0,1) \,\big|\, g_{f,\tau}(\beta) \le \gamma\big\}. \qquad (3)
\end{aligned}$$

Computation of $g_{f,\tau}$ and $g^{-1}_{f,\tau}$ is efficient since they are both solutions to 1-D convex optimization and therefore admit efficient binary search procedures. In some cases, we have also access to a closed form solution [14].

Example 1: For the total variation, $f(z) = (1/2)|z-1|$, we have $g_{f,\tau}(\beta) = \max(0, \beta - \tau)$, $g^{-1}_{f,\tau}(\gamma) = \gamma + \tau$, $\gamma \in (0, 1-\tau)$. This implies that given radius $\tau \in [0,1]$ an adjusted miscoverage level $\bar\epsilon$ is infeasible if $\epsilon \le \tau$, and $\bar\epsilon$ is computed as

$$\bar\epsilon = 1 - \left(1+\frac{1}{m}\right)(1-\epsilon+\tau), \ \epsilon \in [\tau, \ 1], \tau \in [0,1]. \quad (4)$$

Problem Definition: We are given a closed-box stochastic dynamical system as the training environment with the trajectory distribution $\mathcal{D}^{\mathsf{sim}}_{S,K}$. We assume that when this system is deployed in the real world, the trajectories satisfy $\sigma^{\mathsf{real}}_{s_0} \sim \mathcal{D}^{\mathsf{real}}_{S,K} \in \mathcal{P}_{f,\tau}(\mathcal{D}^{\mathsf{sim}}_{S,K})$. Given a user-specified failure probability $\varepsilon \in (0,1)$ and an i.i.d. dataset of trajectories sampled from $\mathcal{D}^{\mathsf{sim}}_{S,K}$, the problem is to obtain a probabilistically guaranteed flowpipe X that contains $\sigma^{\mathsf{real}}_{s_0} \sim \mathcal{D}^{\mathsf{real}}_{S,K}$ for all $\mathcal{D}^{\mathsf{real}}_{S,K} \in \mathcal{P}_{f,\tau}(\mathcal{D}^{\mathsf{sim}}_{S,K})$ with a confidence of $1-\varepsilon$. Formally

$$\left.\begin{aligned} &s_0 \overset{\mathcal{W}}{\sim} \mathcal{I},\\ &\sigma^{\mathsf{real}}_{s_0} \sim \mathcal{D}^{\mathsf{real}}_{S,K} \in \mathcal{P}_{f,\tau}(\mathcal{D}^{\mathsf{sim}}_{S,K}) \end{aligned}\right\} \implies \Pr\Big[\sigma^{\mathsf{real}}_{s_0} \in X\Big] \ge 1-\varepsilon. \quad (5)$$

In other words, we are interested in computing a probabilistically guaranteed flowpipe X from a set of trajectories collected from $\mathcal{D}^{\mathsf{sim}}_{S,K}$ so that X is valid for all trajectories $\mathcal{D}^{\mathsf{real}}_{S,K} \in \mathcal{P}_{f,\tau}(\mathcal{D}^{\mathsf{sim}}_{S,K})$, i.e., despite a potential distribution shift.

III. Learning Surrogate Model Suitable for Probabilistic Reachability Analysis

As we do not have access to the system dynamics in symbolic form, our approach to characterize the trajectory distribution is to use a predictor, called the *surrogate model*.

Definition 1: A surrogate model $\mathcal{F} : \mathcal{X} \times \Theta \to \mathcal{Y}$ is a function that approximates a given function $f : \mathcal{X} \to \mathcal{Y}$. Let $d_{\mathcal{Y}}$ be some metric on $\mathcal{Y}$, then the surrogate model guarantees that for some value of $\theta \in \Theta$, and for any x sampled from a distribution over $\mathcal{X}$, the induced distribution over the random variable $d_{\mathcal{Y}}(\mathcal{F}(x;\theta), f(x))$ has good approximation properties, such as bounds on the moments of the distribution (e.g., mean value) or bounds on the quantile of the distribution.

In our setting, the set $\mathcal{X}$ is the set of states $\mathcal{S}$ with the distribution over $\mathcal{X}$ being $\mathcal{D}^{\mathsf{sim}}_{S,K}$ and $\mathcal{Y}$ is the set of K-step trajectories $\mathcal{S}^K$, i.e., $\mathcal{F}$ maps a given initial state (or an uncertain model parameter) to the predicted K-step trajectory of the system. The metric $d_{\mathcal{Y}}$ can be any metric on the trajectory space. One example surrogate model is a feedforward NN with n inputs and Kn outputs, represented as $\bar\sigma_{s_0} = \mathcal{F}(s_0;\theta)$ where θ is the set of trainable parameters. To train the surrogate model, we need to define a specific residual error between a set of sampled trajectories and those predicted by the model. While most surrogate models are trained using the cumulative squared loss across a training dataset [42], we consider a loss function that helps us reduce conservatism in computing the probabilistic reach set of the system.

Training a Lipschitz-Bounded NN-Based Surrogate Model: Training is a procedure to identify the parameter value θ which makes the surrogate model a good approximation; we use backpropagation to train the surrogate by sampling K-step trajectories from the simulator of the original model. We call this dataset $\mathcal{T}^{\mathsf{trn}}$. The surrogate model predicts the trajectory $\sigma^{\mathsf{sim}}_{s_0}$ starting from an initial state sampled from $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$. We denote the predicted trajectory $\bar\sigma_{s_0}$ corresponding to $\sigma^{\mathsf{sim}}_{s_0}$ as

$$\bar\sigma_{s_0} = \Big[s_0^\top, \ \mathcal{F}(s_0\ \theta)\Big], \quad \text{where, } \mathcal{F}(s_0\ \theta) = \Big[\mathsf{F}^1(s_0), \ldots, \mathsf{F}^n(s_0), \ldots, \mathsf{F}^{(K-1)n+1}(s_0), \ldots, \mathsf{F}^{nK}(s_0)\Big]^\top.$$

Here, $\mathsf{F}^{(i-1)n+r}(s_0)$ is the rth state component at the ith time-step in the trajectory. In other words, we stack the

[3]The guarantees from CI are marginal over all potentially sampled calibration sets. The guarantees over some fixed calibration set can be shown to be a random variable that has distribution **Beta**$(\ell, m+1-\ell)$ [39]. For example, if $m = 10^4$, we get tight probabilistic guarantees for any $\epsilon \in (0,1)$ as the variance of the **Beta** distribution is bounded by 2.5×10^{-5}.

[4]For any $\epsilon \in (0,1)$, the $(1-\epsilon)$-quantile of a random variable R is defined as $\inf\{z \in \mathbb{R} | \Pr[R \le z] \ge 1-\epsilon\}$.

[5]Following [14, Lemma A.2], we note that $g_{f,\tau}$ is related to the worst-case CDF of any distribution with at most τ distribution shift, and g^{-1} is related to the inverse worst-case CDF.

dimension and time in the trajectory into a single vector.[6] We remark that a trained surrogate model with a nonrestricted Lipschitz constant is problematic for reachability analysis, as approximation errors can get uncontrollably magnified resulting in trivial bounds. As a result, we use techniques from [43] to penalize the Lipschitz constant of the trained NN over the course of the training process.

Residual Error: For training NN surrogate models, a common practice is to minimize a loss function, representing the difference between the trajectory predicted by the surrogate model and the actual trajectory. To formulate this difference, we formally define the notion of the residual error as follows.

Definition 2 (Residual Error): Let $e_i \in \mathbb{R}^n$ denotes the ith basis vector of $\mathbb{R}^n$. For a trajectory $(s_0, \sigma_{s_0}^{\mathsf{sim}})$ with $\sigma_{s_0}^{\mathsf{sim}}$ sampled from $\mathcal{D}_{S,K}^{\mathsf{sim}}$, and $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$, we define

$$R^j = \left| e_{j+n}^\top \sigma_{s_0}^{\mathsf{sim}} - \mathsf{F}^j(s_0) \right|, \quad j \in [nK]. \tag{6}$$

Note that R^j is a non-negative prediction error between the $(j+n)^{th}$ component[7] of $\sigma_{s_0}^{\mathsf{sim}}$ and its prediction $\mathsf{F}^j(s_0), j \in [nK]$. The trajectory residual R is then defined as the largest among all scaled, component-wise prediction errors with scaling factors $\alpha_j > 0, j \in [nK]$, i.e., R is defined as

$$R = \max\left(\alpha_1 R^1, \alpha_2 R^2, \ldots, \alpha_{nK} R^{nK}\right). \tag{7}$$

Note that this definition is inspired by [13].[8] Compared to [10], utilizing the maximum of weighted errors obviates the need to union bound component-wise probability guarantees to obtain a trajectory-level guarantee. Let $R_i = \max(\alpha_1 R_i^1, \ \alpha_2 R_i^2, \ldots, \alpha_{nK} R_i^{nK})$ for $i \in [|\mathcal{T}^{\mathsf{trn}}|]$ denotes the trajectory residual as in (7) for the training dataset $\mathcal{T}^{\mathsf{trn}}$.

Training Using a $\bar{\delta}$-Quantile Loss: Let $\bar{\delta} = 1 - \bar{\epsilon}$ where $\bar{\epsilon}$ is the adjusted miscoverage level as defined previously. The ultimate goal from training a surrogate model is to achieve a higher level of accuracy in our reachability analysis. The mean squared error (MSE) loss function is a popular choice to train surrogate models; however, we later show that our proposed flowpipe is generated based on the quantile of the trajectory residual error. Although the MSE loss function is popular and efficient, it may result in a heavy tailed distribution for the residual error which can imply a noticeably larger quantile and result in conservative flowpipes. Thus, to improve overall statistical guarantees, we are interested in minimizing the $\bar{\delta}$-quantile of the trajectory-wise residuals, for an appropriate $\bar{\delta} \in [0, 1)$; toward that end, we add a new trainable parameter q. We can also setup the training process such that the scaling factors $\alpha_1, \ldots, \alpha_{nK}$ become decision variables for the optimization problem. Thus, the set of trainable parameters includes the NN parameters θ, the scaling factors $\alpha_1, \ldots, \alpha_{nK}$ and the parameter q that approximates the $\bar{\delta}$-quantile of the residual loss. We define two loss functions.

1) The first loss function $\mathcal{L}_1$ is to set the trainable parameter q as the $\bar{\delta}$-quantile of trajectory-wise residuals. This loss function is inspired from literature on quantile regression [44], and it is a well-known result that minimizing this function yields q to be the $\bar{\delta}$-quantile of $R_1, \ldots, R_{|\mathcal{T}^{\mathsf{trn}}|}$. Thus, given a batch of training data points of size $M < |\mathcal{T}^{\mathsf{trn}}|$, let

$$\mathcal{L}_1 = \sum_{i=1}^{M} \bar{\delta}\ \mathsf{ReLU}(R_i - q)\ + \left(1 - \bar{\delta}\right) \mathsf{ReLU}(q - R_i). \tag{8}$$

2) Assuming q as the $\bar{\delta}$-quantile of the i.i.d. residuals R_i, we let the second loss function $\mathcal{L}_2$ minimize

$$\mathcal{L}_2 = q\left(\frac{1}{\alpha_1} + \frac{1}{\alpha_2} + \cdots + \frac{1}{\alpha_{nK}}\right). \tag{9}$$

This is motivated by the fact that, for all $j \in [nK]$, $R_i^j \leq R_i/\alpha_j$ by the definition of R_i. Thus, the sum of errors over the trajectory components is upper bounded by

$$\mathbf{UB}_i = R_i\left(\frac{1}{\alpha_1} + \frac{1}{\alpha_2} + \cdots + \frac{1}{\alpha_{nK}}\right) \tag{10}$$

and the $\bar{\delta}$-quantile of $\mathbf{UB}_i, i \in [|\mathcal{T}^{\mathsf{trn}}|]$ is nothing but $\mathcal{L}_2$.[9]

Therefore, we define the loss function as

$$\mathcal{L} = c\mathcal{L}_1 + \mathcal{L}_2 \tag{11}$$

where c is a large number that penalizes $\mathcal{L}_1$ to make sure that q serves as a good approximation for the $\bar{\delta}$-quantile. The training itself uses standard backpropagation methods for computing the gradient of the loss function, and uses stochastic gradient descent to train the surrogate model.

Properties of Surrogate Model: We pick NNs as surrogate models due to their computational advantages and the ability to fit arbitrary nonlinear functions with low effort in tuning hyper-parameters. We note that the input layer of the NN is always of size n (the state dimension), and the output layer is of size nK (the dimension of the predicted trajectory over K time-steps.) In our experiments, we choose NNs with two to three hidden layers for which we observed good results; picking more hidden layers will give better training accuracy, but may cause overfitting. In each hidden layer we pick an increasing number of neurons between n and nK.

IV. Scalable Data-Driven Reachability Analysis

In this section, we show how we can compute a robust probabilistically guaranteed reach set or flowpipe $X \subset \mathbb{R}^{n(K+1)}$ for a stochastic dynamical system. Given a miscoverage level ϵ, we wish to be at least $(1-\epsilon)$-confident about the reach-set that we compute. For brevity, we introduce $\delta = (1 - \epsilon)$. In

[6]The main advantage of training the trajectory as a long vector in one shot is that this approach eliminates the problem of compounding errors in time series prediction; however, this comes with higher training runtimes.

[7]There is offset of n as the first n components of $\sigma_{s_0}^{\mathsf{sim}}$ are the initial state.

[8]In this definition, we consider component-wise residual for R^j instead of a state-wise residual as the component $e_{j+n}^\top \sigma_{s_0}^{\mathsf{sim}}$ in $\sigma_{s_0}^{\mathsf{sim}}$ may represent different quantities like velocity or position. State-wise residuals may lead to a higher level of conservatism in robust CI, as the magnitude of error in different components of a state may be noticeably different.

[9]In case we replace $\mathcal{L}_2$ with q, the trivial solution for scaling factors is $\alpha_j = 0, j \in [nK]$. Therefore, the proposed secondary loss function $\mathcal{L}_2$ also results in avoiding the trivial solution for scaling factors.

the procedure that we describe, we compute a probabilistically guaranteed δ-confident flowpipe, defined as follows.

Definition 3 (δ-Confident Flowpipe): For a given confidence probability $\delta \in (0,\ 1)$, a distribution $\mathcal{D}^{\mathsf{sim}}_{S,K}$, the radius τ, and an f-divergence ball $\mathcal{P}_{f,\tau}(\mathcal{D}^{\mathsf{sim}}_{S,K})$, we say that $X \subseteq \mathbb{R}^{n(K+1)}$ is a δ-confident flowpipe if we have $\Pr[\sigma^{\mathsf{real}}_{s_0} \in X] \geq \delta$ for any random trajectory $\sigma^{\mathsf{real}}_{s_0} \sim \mathcal{D}^{\mathsf{real}}_{S,K} \in \mathcal{P}_{f,\tau}(\mathcal{D}^{\mathsf{sim}}_{S,K})$ with $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$.

Our objective is to compute X while being limited to sample trajectories from the training environment $\mathcal{D}^{\mathsf{sim}}_{S,K}$. We will demonstrate that we can compute X with formal probabilistic guarantees by combining reachability analysis on the surrogate model trained from $\mathcal{T}^{\mathsf{trn}}$ and error analysis on this model via robust CI.

Deterministic Reachsets for the Surrogate Models: Using the surrogate model from Section III, we show how to perform deterministic reachability analysis to get *surrogate flowpipes*.

Definition 4 (Surrogate Flowpipe): The surrogate flowpipe $\bar{X} \subset \mathbb{R}^{n(K+1)}$ is defined as a superset of the image of $\mathcal{F}(\mathcal{I}\ \theta)$. Formally, for all $s_0 \in \mathcal{I}$, we need that $[s_0^\top,\ \mathcal{F}(s_0\ \theta)] \in \bar{X}$.

Thus, to compute the surrogate flowpipe, we essentially need to compute the image of $\mathcal{I}$ w.r.t. the $\mathcal{F}$. This can be accomplished by performing reachability analysis for NNs, e.g., using tools, such as [3], [34], [45], and [46].

Robust δ-Confident Flowpipes: In spite of training the surrogate model to maximize prediction accuracy, it is still possible that a predicted trajectory is not accurate, especially when predicting the system trajectory from a previously unseen initial state. Note also that we trained the surrogate model on trajectory data from $\mathcal{D}^{\mathsf{sim}}_{S,K}$. We thus cannot expect the predictor to always perform well on trajectories drawn from $\mathcal{D}^{\mathsf{real}}_{S,K}$. We now show how to quantify this prediction uncertainty using robust CI. To do so, we first sample an i.i.d. set of trajectories from the training environment $\mathcal{D}^{\mathsf{sim}}_{S,K}$, which we again denote as the calibration dataset.

Definition 5 (Calibration Data Set): The calibration dataset $\mathcal{R}^{\mathsf{calib}}$ is defined as

$$\mathcal{R}^{\mathsf{calib}} = \left\{ (s_{0,i}, R_i) \;\middle|\; \begin{matrix} s_{0,i} \overset{\mathcal{W}}{\sim} \mathcal{I},\ \sigma^{\mathsf{sim}}_{s_{0,i}} \sim \mathcal{D}^{\mathsf{sim}}_{S,K} \\ R_i = \max\left(\alpha_1 R_i^1, \ldots, \alpha_{nK} R_i^{nK}\right) \end{matrix} \right\}.$$

Here, $\sigma^{\mathsf{sim}}_{s_{0,i}}$ refers to the trajectory starting at the ith initial state sampled from $\mathcal{W}$ and the resulting trajectory from $\mathcal{D}^{\mathsf{sim}}_{S,K}$, and R_i^j is as defined in (6).

Remarks 1: It is worth noting that although the data points within a single trajectory may not be i.i.d., the trajectory $\sigma^{\mathsf{sim}}_{s_0}$ can be treated as an i.i.d. random vector in the $\mathbb{R}^{n(K+1)}$-space, and subsequently the residuals are also i.i.d. This is crucial to apply robust CI, which requires that the calibration set is exchangeable (a weaker form of i.i.d.).

Let $\mathcal{J}^{\mathsf{sim}}_{S,K}$ be the distribution over trajectory-wise residuals for trajectories from $\sigma^{\mathsf{sim}}_{s_0} \sim \mathcal{D}^{\mathsf{sim}}_{S,K}$. However, we wish to get information about the trajectory-wise residual R for a trajectory sampled from $\mathcal{D}^{\mathsf{real}}_{S,K} \in \mathcal{P}_{f,\tau}(\mathcal{D}^{\mathsf{sim}}_{S,K})$. Let the distribution of R induced by $\mathcal{D}^{\mathsf{real}}_{S,K}$ be denoted by $\mathcal{J}^{\mathsf{real}}_{S,K}$. As a direct result from the data processing inequality [47], the distribution shift between $\mathcal{D}^{\mathsf{real}}_{S,K}$ and $\mathcal{D}^{\mathsf{sim}}_{S,K}$ is larger than the distribution shift between $\mathcal{J}^{\mathsf{real}}_{S,K}$ and $\mathcal{J}^{\mathsf{sim}}_{S,K}$ so that we have $\mathcal{J}^{\mathsf{real}}_{S,K} \in \mathcal{P}_{f,\tau}(\mathcal{J}^{\mathsf{sim}}_{S,K})$.

Knowing that $\mathcal{J}^{\mathsf{real}}_{S,K} \in \mathcal{P}_{f,\tau}(\mathcal{J}^{\mathsf{sim}}_{S,K})$, we can utilize robust CI in [14] to find a guaranteed upper bound for the δ-quantile of R. We call this guaranteed upper bound as robust conformalized δ-quantile, and we denote it with $R^*_{\delta,\tau}$, where, $\Pr[R \leq R^*_{\delta,\tau}] \geq \delta$. Specifically, we utilize (3) to compute $R^*_{\delta,\tau}$ from the calibration dataset $\mathcal{R}^{\mathsf{calib}}$.

Next we show that our definition of residual error introduced in (7) allows us to use a single trajectory-wise nonconformity score for applying robust CI (instead of the component-wise CI as in [10]).

Lemma 1: Assume $R^*_{\delta,\tau}$ is the $\bar{\delta}$-quantile computed over the residuals R_i from the calibration dataset $\mathcal{R}^{\mathsf{calib}}$. For the residual $R = \max(\alpha_1 R^1, \alpha_2 R^2, \ldots, \alpha_{nK} R^{nK})$ sampled from the distribution $\mathcal{J}^{\mathsf{real}}_{S,K} \in \mathcal{P}_{f,\tau}(\mathcal{J}^{\mathsf{sim}}_{S,K})$, it holds that

$$\Pr\left[\bigwedge_{j=1}^{nK}\left[R^j \leq R^*_{\delta,\tau}/\alpha_j\right]\right] \geq \delta$$

where R^j is again the component-wise residual for $j \in [nK]$.

Proof: The proof follows as the residual R is the maximum of the scaled version of component-wise residuals so that:

$$R = \max\left(\alpha_1 R^1, \alpha_2 R^2, \ldots, \alpha_{nK} R^{nK}\right) \iff \bigwedge_{j=1}^{nK}\left[R^j \leq \frac{R}{\alpha_j}\right].$$

Now, since $\Pr[R \leq R^*_{\delta,\tau}] \geq \delta$ as well as $R < R^*_{\delta,\tau} \iff R^j < R^*_{\delta,\tau}/\alpha_j$ for all $j \in [nK]$, we can claim that $\Pr[\bigwedge_{j=1}^{nK}[R^j \leq R^*_{\delta,\tau}/\alpha_j]] \geq \delta$ ■

Next, we introduce the notion of an inflating zonotope to define the inflated flowpipe from the surrogate flowpipe.

Definition 6 (Inflating Zonotope): A zonotope $\mathsf{Zonotope}(b, A)$ is defined as a centrally symmetric polytope with $b \in \mathbb{R}^k$ as its center, and $A = \{g_1, \ldots, g_p\}$ is a set of generators, where $g_i \in \mathbb{R}^k$, that represents the set $\{b + \mu_i g_i \mid \mu_i \in [-1, 1]\}$. Here, we introduce the inflating zonotope with base vector

$$A = \mathbf{diag}\left(0_{1\times n}, \frac{R^*_{\delta,\tau}}{\alpha_1}, \ldots, \frac{R^*_{\delta,\tau}}{\alpha_{nK}}\right)$$

and center, b is the vector $\mathbf{0}$ of length $(n+1)K$; the notation $\mathbf{diag}(v)$ represents a diagonal matrix with the elements of v along its diagonal and off-diagonal elements being zero.

Including this inflating zonotope in our probabilistic reachability analysis leads to the following result.

Theorem 1: Let $\bar{X}$ be a surrogate flowpipe of the surrogate model $\mathcal{F}$ for the set of initial conditions $\mathcal{I}$. Let $R^*_{\delta,\tau}$ be computed from the calibration dataset $\mathcal{R}^{\mathsf{calib}}$, as shown before. If we use $R^*_{\delta,\tau}$ to construct the inflated surrogate flowpipe

$$X = \bar{X} \oplus \mathsf{Zonotope}(0, \mathbf{diag}([0_{1\times n}, e]))$$
$$e = \left[R^*_{\delta,\tau}/\alpha_1, \ldots, R^*_{\delta,\tau}/\alpha_{nK}\right]$$

then, it holds that X is a δ-confident flowpipe for any $\sigma^{\mathsf{real}}_{s_0} \sim \mathcal{D}^{\mathsf{real}}_{S,K} \in \mathcal{P}_{f,\tau}(\mathcal{D}^{\mathsf{sim}}_{S,K})$ with $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$.

Proof: Assume again that $\sigma^{\mathsf{real}}_{s_0} \sim \mathcal{D}^{\mathsf{real}}_{S,K} \in \mathcal{P}_{f,\tau}(\mathcal{D}^{\mathsf{sim}}_{S,K})$ with $s_0 \overset{\mathcal{W}}{\sim} \mathcal{I}$, and recall that $R = \max[\alpha_1 R^1, \ldots, \alpha_{nK} R^{nK}]$

where $R^j = |e_{j+n}^\top \sigma_{s_0}^{\mathsf{real}} - \mathsf{F}^j(s_0)|$. Applying Lemma 1 results in $\Pr[\bigwedge_{j=1}^{nK}(R^j \leq R^*_{\delta,\tau}/\alpha_j)] \geq \delta$. We rephrase this as

$$\Pr\left[\bigwedge_{j=1}^{nK}\left(|\, e_{j+n}^\top \sigma_{s_0}^{\mathsf{real}} - \mathsf{F}^j(s_0)\,| \leq R^*_{\delta,\tau}/\alpha_j\right)\right] \geq \delta.$$

Next, we define the interval $C_j(s_0)$ as

$$C_j(s_0) := \left[\mathsf{F}^j(s_0) - R^*_{\delta,\tau}/\alpha_j \;\; \mathsf{F}^j(s_0) + R^*_{\delta,\tau}/\alpha_j\right]$$

and accordingly obtain the guarantee that

$$\Pr\left[\bigwedge_{j=1}^{nK}\left(e_{j+n}^\top \sigma_{s_0}^{\mathsf{real}} \in C_j(s_0)\right)\right] \geq \delta.$$

Based on this representation, we can now see that

$$\Pr\left[\sigma_{s_0}^{\mathsf{real}} \in \mathsf{Zonotope}\left(\left[s_0^\top, \mathcal{F}(s_0\;\theta)\right], \mathbf{diag}\left(\left[0_{1\times n},\; e\right]\right)\right)\right] \geq \delta. \quad (12)$$

Finally, since $\Pr[s_0 \notin \mathcal{I}] = 0$ and $\bar{X}$ is a surrogate flowpipe for the surrogate model $\mathcal{F}$ on $\mathcal{I}$, i.e., $s_0 \in \mathcal{I}$ implies $[s_0^\top, \mathcal{F}(s_0\;\theta)] \in \bar{X}$, we can conclude

$$\mathsf{Zonotope}\left(\left[s_0^\top, \mathcal{F}(s_0\;\theta)\right], \mathbf{diag}\left(\left[0_{1\times n},\; e\right]\right)\right) \subset \bar{X} \oplus \mathsf{Zonotope}\left(0, \mathbf{diag}\left(\left[0_{1\times n},\; e\right]\right)\right) = X. \quad (13)$$

Consequently, we know that $\Pr[\sigma_{s_0}^{\mathsf{real}} \in X] \geq \delta$ holds. ■

We note that the surrogate reachability, and also use of the Minkowski sum in the reachability analysis, results in some level of conservatism.

Remarks 2: We note that we can even compute the minimum size of the calibration dataset required to achieve a desired confidence probability $\delta \in (0, 1)$. Robust CI [14] imposes two constraints in this regard. The first constraint specifies a relation between the adjusted miscoverage level $\bar{\epsilon}$ and the size of the calibration dataset as $\lceil (L+1)(1-\bar{\epsilon})\rceil \leq L$. The second constraint is that the ranges of $g_{f,\tau}$ and $g_{f,\tau}^{-1}$ have to be within $[0, 1]$. Thus, we can impose $(1+1/L)g_{f,\tau}^{-1}(\delta) < 1$, or in other words $L > \lceil g_{f,\tau}^{-1}(\delta)/(1 - g_{f,\tau}^{-1}(\delta))\rceil$.

Tightening the Surface Area of the Flowpipe: The scaling factors α_j are trained to minimize the sum of errors over the trajectory components, see (9). The expression $R^*_{\delta,\tau}\sum_{j=1}^{nK} 1/\alpha_j$ arising from (9) can also be interpreted as the surface area of the inflating zonotope, see Definition 6. We now show how we can update scaling factors after training to reduce the surface area to tighten the δ-confident flowpipe further. Let us sample a new trajectory dataset $\mathcal{T}^{\mathsf{LP}}$ and compute the prediction errors R_i^j and residuals R_i for $i \in [|\mathcal{T}^{\mathsf{LP}}|]$, and also their conformalized robust $\bar{\delta}$-quantile $R^*_{\delta,\tau}$, using the trained scaling factors α_j and surrogate model.

The main idea for an efficient update of the trained scaling factors is as follows. Assume α_j' is the updated version of α_j. If this update is such that the updated trajectory residuals $\max(\alpha_1' R_1^i, \ldots, \alpha_{nK}' R_{nK}^i)$, $i \in [|\mathcal{T}^{\mathsf{LP}}|]$ are the same as the trajectory residuals R_i under α_j, then $R^*_{\delta,\tau}$ under the updated α_j' remains the same. By defining $\omega_j' = 1/\alpha_j'$, we see that the surface area $R^*_{\delta,\tau}\sum_{j=1}^{nK}\omega_j'$ of the inflating zonotope depends linearly on ω_j'. On the other hand the constraint $R_i = \max(R_i^1/\omega_1', \ldots, R_i^{nK}/\omega_{nK}')$, is a linear constraint. This constraint can be equivalently represented as

$$\forall i \in \left[|\mathcal{T}^{\mathsf{LP}}|\right], j \in [nK]\;\; R_i\omega_j' \geq R_i^j$$

under the additional assumption that the updated scaling factors ω_j' are minimized. This means an efficient update on scaling factors to reduce the surface area can be done via linear programming with decision variables $\omega_j', j \in [nK]$, i.e.,

$$\textbf{minimize} \sum_{j=1}^{nK}\omega_j' \quad \textbf{s.t.} \quad \forall i \in \left[|\mathcal{T}^{\mathsf{LP}}|\right], j \in [nK]\;\; \omega_j' \geq R_i^j/R_i \quad (14)$$

which has the analytical solution $\omega_j' = \max_i [R_i^j/R_i]$.

V. Experimental Results

To mimic real-world systems that can produce actual trajectory data, we use stochastic difference equation-based models derived from dynamical system models. In these difference equations, we assume additive Gaussian noise that models uncertainty in observation, dynamics, or even modeling errors.

Our theoretical guarantees depend on knowledge of the distribution shift τ. In practice, however, τ is usually not known *a priori* but can be estimated from the data. For the purpose of providing an empirical examination of our results, we fix τ *a priori* to compute the δ-confident flowpipe and construct a system $\mathcal{D}_{S,K}^{\mathsf{real}}$ from $\mathcal{D}_{S,K}^{\mathsf{sim}}$ by varying system parameters such that $\mathcal{J}_{S,K}^{\mathsf{real}} \in \mathcal{P}_{f,\tau}(\mathcal{J}_{S,K}^{\mathsf{sim}})$. We ensure that this holds by estimating the distribution shift, denoted by $\tilde{\tau}$, as the f-divergence between $\mathcal{J}_{S,K}^{\mathsf{sim}}$ and $\mathcal{J}_{S,K}^{\mathsf{real}}$ and by making sure that $\tilde{\tau} \leq \tau$. In our experiments, we used the total variation distance for f, and used 3×10^5 trajectories to estimate $\tilde{\tau}$.

We use ReLU activation functions in our surrogate NN-based models motivated by recent advances in NN verification with ReLU activations. We specifically use the NNV toolbox from [34] for reachability analysis of the surrogate model. While other activation functions could be used, we expect more conservative results in case we utilize non-ReLU activation functions. The approach in [34] uses star-sets (an extension of zonotopes) to represent the reachable set and employs two main methods: 1) the exact-star method that performs exact but slow computations and 2) the approx-star method that is conservative but faster. To mitigate the runtime of the exact-star technique and the conservatism of the approx-star technique, set partitioning can be utilized [48], where initial states are partitioned into subregions and reachability is done on each subregion in parallel.

As per Theorem 1, our results are guaranteed to be valid with a confidence of δ. To determine how tight this bound is, we will empirically examine the computed probabilistic flowpipes. We do so by sampling i.i.d. trajectories from $\mathcal{D}_{S,K}^{\mathsf{real}}$[10] and computing the ratio of the trajectories that are included in the probabilistic flowpipes, which we denote by $\tilde{\Delta}$. In addition, to check the coverage guarantee δ for $R^*_{\delta,\tau}$ directly, we also report the ratio of the trajectories that provide a residual less

[10]We use trajectories close to the worst case where $\tilde{\tau}$ is close to τ.

TABLE I

SHOWS THE DETAIL OF OUR COMPUTATION PROCESS TO PROVIDE PROBABILISTICALLY GUARANTEED FLOWPIPES. THE TIME HORIZON FOR EXPERIMENTS 1, 5, AND 6 IS $K = 50$ TIME-STEPS AND FOR THE EXPERIMENTS 2, 3, AND 4 IS $K = 100$ TIME-STEPS. THE SAMPLING TIME FOR QUADCOPTER AND TIME-REVERSED VAN DER POL (TRVDP) ARE 0.05 AND 0.02 S, RESPECTIVELY. WE EXAMINE THE RESULTS WITH A VALID DISTRIBUTION SHIFT (EXPLAINED IN DETAIL IN TABLE II) THAT IS, LESS THAN THE MAXIMUM SPECIFIED DISTRIBUTION SHIFT IN TERMS OF TOTAL VARIATION. THIS SHIFT IS ESTIMATED THROUGH THE COMPARISON BETWEEN 300 000 TRAJECTORIES FROM $\mathcal{D}^{\mathsf{real}}_{S,K}$ AND $\mathcal{D}^{\mathsf{sim}}_{S,K}$. WE ALSO UTILIZE 10 000 TRAJECTORIES (NUMBER OF TRIALS) FROM THIS SPECIFIC DISTRIBUTION $\mathcal{D}^{\mathsf{real}}_{S,K}$ TO EXAMINE THE COVERAGE OF FLOWPIPES AND 300 000 TRAJECTORIES FOR EXAMINATION OF THE COVERAGE LEVEL FOR $R^*_{\delta,\tau}$ (I.E., $\tilde{\Delta}$, $\tilde{\delta}$). TO EVALUATE THE CONTRIBUTION OF ROBUST CI, WE ALSO SOLVE FOR THE FLOWPIPES AGAIN NEGLECTING THE DISTRIBUTION SHIFT, I.E., $\bar{\epsilon} = \epsilon$, AND SHOW THE COVERAGE GUARANTEE FOR $R^*_{\delta,\tau}$ AND FLOWPIPES MAY GET VIOLATED, ($\tilde{\delta} < \delta$ OR $\tilde{\Delta} < \delta$), IN CASE THE SHIFTED DISTRIBUTION (DEPLOYMENT DISTRIBUTION) IS CONSIDERED. THE RUNTIMES WE REPORT FOR REACHABILITY ASSUMES NO PARALLEL COMPUTING

Specification			Reachability Analysis with Robust CI				
Experiment #: Underlying dynamics	Confidency of flowpipe, i.e. δ	Maximum distribution shift's radius	Number of Star-sets from NNV	Conformal inference runtime(sec)	Reachability technique	Overal reachability runtime(sec)	Size of calibration dataset ($\lvert\mathcal{R}^{\mathsf{calib}}\rvert$)
1: Periodic	$\delta = 95\%$	0	400	0.0892	approx-star	22.5233	10,000
2: Quadcopter	$\delta = 99.99\%$	0	64	1.4299	approx-star	160.0486	20,000
3: Quadcopter	$\delta = 80\%$	0.15	64	0.4971	approx-star	148.9815	10,000
4: Quadcopter	$\delta = 70\%$	0.25	64	0.4971	approx-star	148.9815	10,000
5: TRVDP	$\delta = 99.99\%$	0	1	4.8218	exact-star	1.5761	30,000
6: TRVDP	$\delta = 77\%$	0.225	1	0.2876	exact-star	0.1404	10,000

	Training			
	Training runtime	Size of training dataset ($\lvert\mathcal{T}^{\mathsf{trn}}\rvert$)	Linear programming runtime	Size of dataset for linear programming ($\lvert\mathcal{T}^{\mathsf{LP}}\rvert$)
Experiment 1:	41 minutes	100,000	1.7422 seconds	10,000
Experiment 2:	124 minutes	40,000	29.3255 seconds	2,000
Experiment 3,4:	112 minutes	40,000	21.1406 seconds	2,000
Experiment 5:	25 minutes	40,000	6.3559 seconds	50,000
Experiment 6:	27 minutes	40,000	2.8236 seconds	10,000

	Examination				
	Example of induced distribution shift's radius (i.e. $\tilde{\tau}$)	Coverage Estimation (i.e. $\tilde{\Delta}$) for flowpipe generated by:		Coverage Estimation (i.e. $\tilde{\delta}$) for $R^*_{\delta,\tau}$ generated by:	
		Robust CI	Vanilla CI	Robust CI	Vanilla CI
Experiment 1:	0	96.31%	96.31%	95.05%	95.05%
Experiment 2:	0	100%	100%	99.99%	99.99%
Experiment 3:	0.1445	100%	100%	88.58%	70.86%
Experiment 4:	0.2395	100%	100%	80.50%	49.64%
Experiment 5:	0	99.99%	99.99%	99.99%	99.99%
Experiment 6:	0.2085	95.91%	56.52%	95.87%	55.73%

than $R^*_{\delta,\tau}$, which we denote with $\tilde{\delta}$. We emphasize that $\tilde{\Delta}$ and $\tilde{\delta}$ are both expected to be greater than δ.

In the remainder, we first present a case study to compare between reachability with surrogate models using the MSE and our proposed quantile loss function in (11). We show that the quantile loss function results in tighter probabilistic flowpipes. After that, we present several case studies on a 12-D quadcopter and the time reversed van Der Pol dynamics. The results are also summarized in Table I. We visualize our flowpipes by their 2-D projection. Therefore, in case a trajectory is included in all the visualized bounds, it does not necessary mean the trajectory is covered. We instead, determine the inclusion of traces in our star-sets using the NNV toolbox which determines set inclusion by solving a linear programming feasibility problem.

Comparison Between MSE and Quantile Minimization: Experiment 1: Our first experiment will show the advantage of training a surrogate model with quantile loss function compared to training a surrogate model using the MSE loss function. Therefore, we model $\mathcal{D}^{\mathsf{sim}}_{S,K}$ as the nonlinear system

$$x_{k+1} = 0.985y_k + \sin(0.5x_k) - 0.6\sin(x_k + y_k) - 0.07 + 0.01v_1$$
$$y_{k+1} = 0.985x_k + \cos(0.5y_k) - 0.6\cos(x_k + y_k) - 0.07 + 0.01v_2$$

that generates a periodic motion. Here, v_1 and v_2 denote random variables sampled from a normal distribution. In this experiment, we do not consider a shifted stochastic system $\mathcal{D}^{\mathsf{real}}_{S,K}$, and instead sample trajectories from $\mathcal{D}^{\mathsf{sim}}_{S,K}$ for comparison of our two surrogate models. The first surrogate model is trained as proposed in Section III using quantile minimization, while the other surrogate model is trained with the MSE loss function. Our results are shown upfront in Fig. 1(a) where we compare the probabilistic reachable sets of these two models.

In more detail, recall that the scaling factors $\alpha_1, \ldots, \alpha_{nK}$ of our proposed method in Section III are jointly trained with the surrogate model. However, since we do not train these scaling factors jointly when we use the MSE loss function, we instead

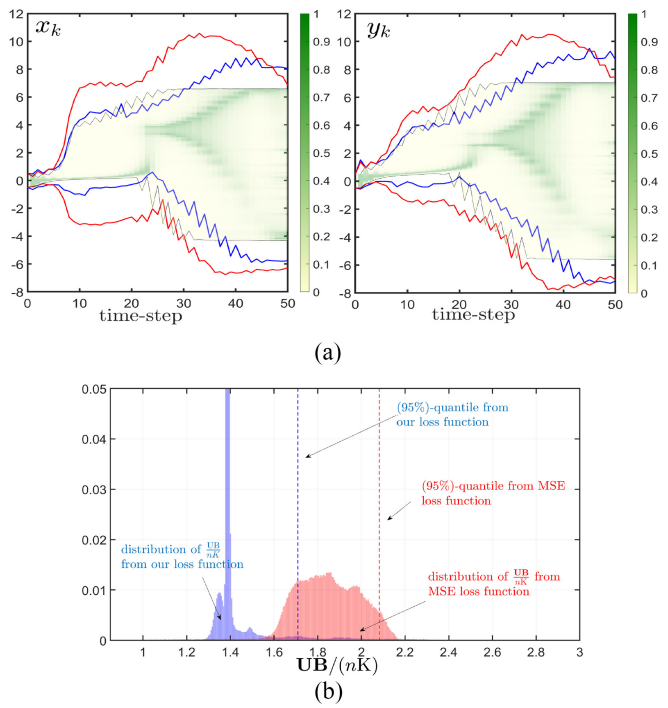

Fig. 1. (a) and (b) show a comparison between flowpipes and distributions of $\mathbf{UB}/(nK)$, respectively, for training via MSE and training via our proposed loss function (11). (a) Flowpipe for xk and yk over time steps. The red borders are for flowpipes generated by MSE loss function and the blue ones are for quantile based loss function. The shaded region shows an approximation of flowpipe by recording trajectories, and the darkness of the green color shows the density of the trajectories. The black lines are the borders for the shaded region. The shaded area is generated via 300000 trajectories. (b) Distribution of UB/(nK) for the MSE and the quantile-based NNs for 3 × 105 samples. The 95represents the surface area of the obtained inflating zonotope. The figure is cropped for better visibility.

compute them beforehand following [15]. In other words, we normalize the component-wise residuals as

$$\alpha_j = 1/\omega_j \text{ where } \omega_j = \max\left(R_1^j,\ R_2^j,\ \ldots,\ R_{|\mathcal{T}^{\text{trn}}|}^j\right)$$

for each $j \in [nK]$. We utilized $|\mathcal{T}^{\text{trn}}| = 10^5$ random trajectories with $K = 50$ for training the surrogate model. The initial states were uniformly sampled from the set of initial states $\mathcal{I}_1 = [-0.5, 0.5] \times [-0.5, 0.5]$. In both case, we trained a ReLU surrogate model with structure [2, 20, 50, 90, 100] and we applied approx-star from the NNV [34] toolbox for the reachability analysis. To lower the conservatism of approx-star, we partition the set of initial states into 400 partitions, and perform the surrogate reachability analysis for every partition separately. The flowpipe is also computed for the confidence level of $\delta \geq 95\%$. The details of the experiment via quantile minimization are also provided in Table I.

We additionally compare the surface area $R^*_{\delta,\tau} \sum_{j=1}^{nK} 1/\alpha_j$ of the inflating zonotopes, see Definition 6, for both surrogate models. Note that this surface area is the $\mathcal{L}_2$ loss in (9) when $q = R^*_{\delta,\tau}$, which we enforce during training. The $\bar{\delta}$-quantile of $\mathbf{UB}_i$ as defined in (10) is the $\mathcal{L}_2$ loss, and hence approximates the surface area of the inflating zonotope. To compare the distributions of $\mathbf{UB}_i$, we simulate 3×10^5 trajectories and compute $\mathbf{UB}_i/(nK)$ for both the MSE and the quantile loss-based NNs. We present the histograms of $\mathbf{UB}_i/(nK)$ for both loss functions in Fig. 1(b) where we see that the quantile of $\mathbf{UB}_i$ for MSE is larger. This emphasizes the advantage of training via quantile loss function.

TABLE II
INITIAL STATE DISTRIBUTION AND ADDED GAUSSIAN NOISE (MEAN: 0, COVARIANCE: Σ) FOR THE TRAINING AND THE SHIFTED ENVIRONMENTS; $\mathsf{uni}(\mathcal{I})$ DENOTES THE UNIFORM DISTRIBUTION OVER $\mathcal{I}$

	Expt.	Dist.	Σ for added noise Gaussian $\mathcal{N}(\mathbf{0}, \Sigma)$
$\mathcal{D}^{\text{sim}}_{S,\text{K}}$	1	$\mathsf{uni}(\mathcal{I}_1)$	$\mathbf{diag}([0.01, 0.01])^2$
	2	$\mathsf{uni}(\mathcal{I}_2)$	$\mathbf{diag}([0.05 \cdot \vec{1}_{1\times 6}, 0.01 \cdot \vec{1}_{1\times 6}])^2$
	3	$\mathsf{uni}(\mathcal{I}_2)$	$\mathbf{diag}([0.05 \cdot \vec{1}_{1\times 6}, 0.01 \cdot \vec{1}_{1\times 6}])^2$
	4	$\mathsf{uni}(\mathcal{I}_2)$	$\mathbf{diag}([0.05 \cdot \vec{1}_{1\times 6}, 0.01 \cdot \vec{1}_{1\times 6}])^2$
	5,6	$\mathsf{uni}(\mathcal{I}_3)$	$\mathbf{diag}([0.1, 0.1])^2$
$\mathcal{D}^{\text{real}}_{S,\text{K}} \in \mathcal{P}_{\tau,f}(\mathcal{D}^{\text{sim}}_{S,\text{K}})$	3	$\mathsf{uni}(\mathcal{I}_2)$	$\Sigma \times 1.8$
	4	$\mathsf{uni}(\mathcal{I}_2)$	$\Sigma \times 2.2$
	6	$\mathsf{uni}(\mathcal{I}_3)$	$\mathbf{diag}([0.1378, 0.1378])^2$

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

Fig. 2. Dynamics for the quadcopter. Here, initial set of states $\mathcal{I}_2 = \{s_0 | i \in [1, 6]: -0.2 \leq s_0(i) \leq 0.2, i \geq 7: s_0(i) = 0\}$.

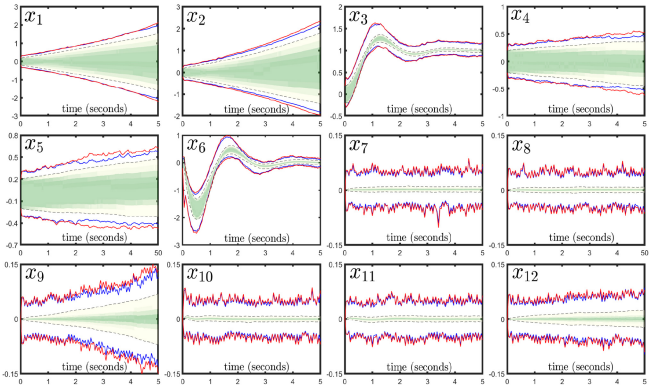

Fig. 3. This figure shows the proposed flowpipes computed for the quadcopter dynamics for each state component over the time horizon of 100 time steps with $\delta t = 0.05$ that means 5 s operation of quadcopter. The red borders show the flowpipe that contains trajectories from $\mathcal{D}^{\text{sim}}_{S,K}$ with provable coverage of $\delta \geq 99.99\%$. The green shaded area shows the density of a collection of 300 000 of these trajectories, and the darker color means the higher density of traces. The blue borders are also for a flowpipe that contains the trajectories from distribution $\mathcal{D}^{\text{sim}}_{S,K}$ with $\delta \geq 95\%$. The dotted black line also shows the border of collected simulated trajectories.

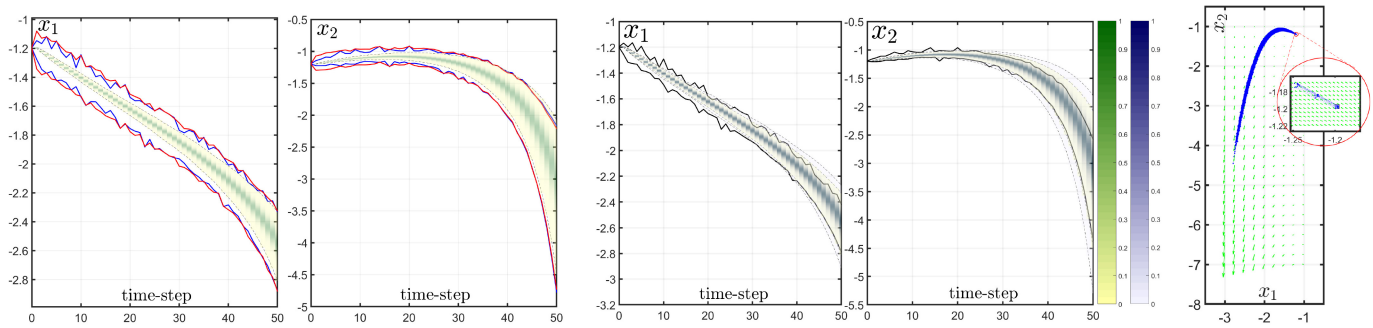

Fig. 4. Shows the density of trajectories starting from $\mathcal{I}_3$ versus their computed flowpipes. The green color-bar represents the density of traces from, $\mathcal{D}^{\mathsf{sim}}_{S,K}$ and the blue color-bar is for traces from $\mathcal{D}^{\mathsf{real}}_{S,K}$. The shaded areas are generated via 3×10^5 different trajectories, and the dotted lines represents their border. (a) shows two different flowpipes for TRVDP dynamics with confidence level of 0.9999 on $\mathcal{D}^{\mathsf{sim}}_{S,K}$. The tighter flowpipe (blue color) utilizes the linear programming (14) while the looser one (red color) does not. (b) shows a flowpipe that covers trajectories from $\mathcal{D}^{\mathsf{real}}_{S,K}$ with the confidence level of 77% and also covers the traces from $\mathcal{D}^{\mathsf{sim}}_{S,K}$ with the confidence level of 99.5%. The blue shaded area is for $\mathcal{D}^{\mathsf{real}}_{S,K}$ and the green shaded area is for $\mathcal{D}^{\mathsf{sim}}_{S,K}$. (c) shows the vector field of TRVDP dynamics that illustrates the instability of the system.

12-D Quadcopter: Next, we consider a 12-D quadcopter model from the benchmarks in [17] that is designed to hover around a prespecified elevation. The ODE model for this system is provided in Fig. 2, where the state consists of the position and velocity of the quadrotor x_1, x_2, x_3 and x_4, x_5, x_6, respectively, as well as the Euler angles x_7, x_8, x_9, i.e., roll, pitch, and yaw, and the angular velocities x_{10}, x_{11}, x_{12}.

We also add additive noise to the system that is detailed in Table II, and we generate data with time step $\delta t = 0.05$ s over 100 time steps (i.e., 5 s). The controller is an NN controller that was presented in [17]. We present three experiments on this model. Learning a surrogate model to map the 12-D initial state to a 1200-D trajectory is impractical. We thus use an interpolation technique to resolve this issue. To that end, we select only certain time-steps of the 1200-D trajectory in order to map the initial state to state values at the selected time steps, while we take care of the remaining time steps via interpolation. If the trajectories are smooth, as is the case in this case study, this is expected to work well. We here select every second time-step to extract a 600-D trajectory ($\delta t = 0.1, K = 50$) to train a surrogate model of structure [12, 200, 400, 600]. Finally we interpolate the sampled 600-D trajectory to approximate the original 1200-D trajectory ($\delta t = 0.05, K = 100$). This interpolation process is integrated in the model in an analytical way, and is done by multiplying a weight matrix, $W \in \mathbb{R}^{1200\times 600}$ to the last layer. This converts the model's structure to [12, 200, 400, 1200] which will be utilized for the surrogate reachability. The scaling factors $\omega_j, j \in [nK]$ will be also interpolated for un-sampled time-steps after the training and before the linear programming.

Experiment 2: In comparison with [10], we provide a higher level of data efficiency. Consider a confidence level of 99.99%, and no distribution shift. We assume a calibration dataset of size $|\mathcal{R}^{\mathsf{calib}}| = 2 \times 10^4$ to compute $R^*_{\delta,\tau}$ and the δ-confident flowpipe, and a ReLU NN of structure [12, 20, 400, 1200] to train the surrogate model. The methodology proposed in [10] requires a calibration dataset of at least 24×10^6 data-points[11] to provide the mentioned level of confidence. On the other hand, we only require 10^4 trajectories. Fig. 3 shows the proposed reach set and Table I presents the detail of the computation process. Our estimation shows that we achieve $\tilde{\delta} = 0.9999$ via 3×10^5 trials and $\tilde{\Delta} = 1$ via 10^4 trials, which aligns with our expectations.

Experiments 3 and 4: In this case study, we generate a 95% confident flowpipe for the trajectories from $\mathcal{D}^{\mathsf{sim}}_{S,K}$ and we utilize it to study the distribution shift on two different deployment environments $\mathcal{D}^{\mathsf{real}}_{S,K}$. This flowpipe is plotted in Fig. 3 and the details of the computation process is included in Tables I and II. For this generated flowpipe, given a maximum distribution shift radius $\tau \in [0, 1]$, the flowpipe's confidence level δ for trajectories from $\mathcal{D}^{\mathsf{real}}_{S,K}$ has to satisfy $\delta \geq 0.95 - \tau$. The bound $\delta \geq \bar{\delta} - \tau$ can be derived from (4). Therefore, we consider two different scenarios. In Experiment 3, we examine our flowpipe for the case $\tau = 0.15$. In this case, for a deployment environment with distribution shift, $\tilde{\tau} < 0.15$ we numerically show that $\tilde{\Delta}, \tilde{\delta} > 0.95 - 0.15 = 0.8$. In addition, in Experiment 4, we assume $\tau = 0.25$ and for a deployment environment with $\tilde{\tau} < 0.25$ we show that $\tilde{\Delta}, \tilde{\delta} > 0.95 - 0.25 = 0.7$. Tables I and II show the detail of the experiments and distribution shift, respectively.

TRVDP Oscillator Dynamics: The TRVDP dynamics is known for its inherent instability, which makes it a pernicious challenge for computing reach sets. The SDE model for TRVDP is

$$\left[\dot{x}_1\ \dot{x}_2\right]^\top = \left[x_2\ \mu x_2\left(1 - x_1^2\right) - x_1\right]^\top + v, \quad \mu = -1$$

here, v is an additive Gaussian noise, detailed in Table II. We generate data from this dynamics with sampling time $\delta t = 0.02$ s, and we target reachability for $K = 50$ time step. We use a limited set of initial states $\mathcal{I}_3 = \{s_0 \mid [-1.2, -1.2] \leq s_0 \leq [-1.195, -1.195]\}$ to investigate the instability of the system dynamics. Our analysis centers on discerning how this instability manifests as a divergence in trajectories originating from this restricted set of initial states. We also assume a model with structure [2, 50, 90, 100] to train the surrogate model. We perform two experiments on this system, explained below.

Experiment 5: In this experiment, we target the flowpipe computation for the TRVDP dynamics for the confidence

[11] Minimum data size in [10] is $|\mathcal{R}^{\mathsf{calib}}| > \lceil (1+\gamma/1-\gamma) \rceil$, where $\gamma = 1 - (1 - \delta/nK)$.

probability of $\delta \geq 99.99\%$ and no distribution shift. Fig. 4(a) shows the resulting flowpipe and Table I shows the details of the process. In this experiment, we also generate another 0.9999-confident flowpipe excluding the linear programming [proposed in (14)] from the process. Fig. 4(a) also compares these flowpipe and shows removing the linear programming increases the level of conservatism.

Experiment 6: We target an arbitrary confidence level of $\delta \geq 0.77$ for the flowpipe, despite distribution shifts within radius $\tau < 0.225$ measured in total variation. As suggested by robust CI, we should target a flowpipe with confidence level of $99.5\% = 77\% + 22.5\%$ on $\mathcal{D}^{\mathsf{sim}}_{S,K}$ to ensure the confidence level of 77% on $\mathcal{D}^{\mathsf{real}}_{S,K}$. Fig. 4(b) shows our probabilistically guaranteed flowpipe, and Tables I and II present the detail of the experiment. These tables also show that, in case we set $\bar{\epsilon} = \epsilon$ in reachability analysis (Vanilla CI) then our flowpipe, violates the guarantee (i.e., $\delta \geq 0.77$). This emphasizes on the contribution of robust CI.

Conclusion: This article addresses challenges in data-driven reachability analysis for stochastic dynamical systems, specifically focusing on distribution shifts between training and test environments. By leveraging a dataset of K-step trajectories, the approach constructs a probabilistic flowpipe, ensuring that the probability of trajectory violation remains below a user-defined threshold even in the presence of distribution shifts. We propose the reliable guarantees with higher data efficiency compared to the existing techniques assuming knowledge of an upper bound for distribution shift. The methodology relies on three key principles: 1) surrogate model learning; 2) reachability analysis using the surrogate model; and 3) robust CI for probabilistic guarantees. We illustrated the efficacy of our approach via reachability analysis on high-dimensional systems like a 12-D quadcopter and unstable systems like the TRVDP oscillator.

REFERENCES

[1] 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.

[2] A. Abate, S. Amin, M. Prandini, J. Lygeros, and S. Sastry, "Computational approaches to reachability analysis of stochastic hybrid systems," in *Proc. HSCC*, 2007, pp. 4–17.

[3] C. Huang, J. Fan, W. Li, X. Chen, and Q. Zhu, "ReachNN: Reachability analysis of neural-network controlled systems," *ACM Trans. Embed. Comput. Syst.)*, vol. 18, no. 5s, pp. 1–22, 2019.

[4] S. Dutta, X. Chen, S. Jha, S. Sankaranarayanan, and A. Tiwari, "Sherlock—A tool for verification of neural network feedback systems: Demo abstract," in *Proc. HSCC*, 2019, pp. 262–263.

[5] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton–Jacobi reachability: A brief overview and recent advances," in *Proc. CDC*, 2017, pp. 2242–2253.

[6] A. P. Vinod, J. D. Gleason, and M. M. Oishi, "SReachTools: A MATLAB stochastic reachability toolbox," in *Proc. HSCC*, 2019, pp. 33–38.

[7] A. P. Vinod, B. HomChaudhuri, and M. M. Oishi, "Forward stochastic reachability analysis for uncontrolled linear systems using fourier transforms," in *Proc. HSCC*, 2017, pp. 35–44.

[8] D. Adzkiya, B. De Schutter, and A. Abate, "Computational techniques for reachability analysis of max-plus-linear systems," *Automatica*, vol. 53, pp. 293–302, Mar. 2015.

[9] T. Gan, M. Chen, Y. Li, B. Xia, and N. Zhan, "Reachability analysis for solvable dynamical systems," *IEEE Trans. Autom. Control*, vol. 63, no. 7, pp. 2003–2018, Jul. 2018.

[10] N. Hashemi, X. Qin, L. Lindemann, and J. V. Deshmukh, "Data-driven reachability analysis of stochastic dynamical systems with conformal inference," in *Proc. CDC*, 2023, pp. 3102–3109.

[11] A. J. Thorpe and M. M. Oishi, "Model-free stochastic reachability using kernel distribution embeddings," *IEEE Control Syst. Lett.*, vol. 4, no. 2, pp. 512–517, Apr. 2020.

[12] S. Vallender, "Calculation of the Wasserstein distance between probability distributions on the line," *Theory Probab. Appl.*, vol. 18, no. 4, pp. 784–786, 1974.

[13] M. Cleaveland, I. Lee, G. J. Pappas, and L. Lindemann, "Conformal prediction regions for time series using linear complementarity programming," in *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 19, 2024, pp. 20984–20992.

[14] M. Cauchois, S. Gupta, A. Ali, and J. C. Duchi, "Robust validation: Confident predictions even when distributions shift," *J. Amer. Stat. Assoc.*, pp. 1–66, Feb. 2024.

[15] Y. Zhao, B. Hoxha, G. Fainekos, J. V. Deshmukh, and L. Lindemann, "Robust conformal prediction for STL runtime verification under distribution shift," 2023, *arXiv:2311.09482*.

[16] I. Csiszár, "A class of measures of informativity of observation channels," *Periodica Mathematica Hungarica*, vol. 2, no. 1–4, pp. 191–213, 1972.

[17] C. Huang, J. Fan, X. Chen, W. Li, and Q. Zhu, "POLAR: A polynomial arithmetic framework for verifying neural-network controlled systems," in *Proc. Int. Symp. Autom. Technol. Verif. Anal.*, 2022, pp. 414–430.

[18] A. Lin and S. Bansal, "Generating formal safety assurances for high-dimensional reachability," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2023, pp. 10525–10531.

[19] A. Alanwar, A. Koch, F. Allgöwer, and K. H. Johansson, "Data-driven reachability analysis from noisy data," *IEEE Trans. Autom. Control*, vol. 68, no. 5, pp. 3054–3069, May 2023.

[20] L. Yang, H. Zhang, J.-B. Jeannin, and N. Ozay, "Efficient backward reachability using the Minkowski difference of constrained zonotopes," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 11, pp. 3969–3980, Nov. 2022.

[21] L. Bortolussi and G. Sanguinetti, "A statistical approach for computing reachability of non-linear and stochastic dynamical systems," in *Proc. Int. Conf. Quant. Eval. Syst.*, 2014, pp. 41–56.

[22] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, "Safety verification of deep neural networks," in *Proc. CAV*, 2017, pp. 3–29.

[23] A. P. Vinod and M. M. Oishi, "Stochastic reachability of a target tube: Theory and computation," *Automatica*, vol. 125, Mar. 2021, Art. no. 109458.

[24] M. Fiacchini and T. Alamo, "Probabilistic reachable and invariant sets for linear systems with correlated disturbance," *Automatica*, vol. 132, Oct. 2021, Art. no. 109808.

[25] A. J. Thorpe, V. Sivaramakrishnan, and M. M. Oishi, "Approximate stochastic reachability for high dimensional systems," in *Proc. Amer. Control Conf. (ACC)*, 2021, pp. 1287–1293.

[26] A. Devonport, F. Yang, L. El Ghaoui, and M. Arcak, "Data-driven reachability analysis with Christoffel functions," in *Proc. CDC*, 2021, pp. 5067–5072.

[27] J. B. Lasserre and E. Pauwels, "The empirical Christoffel function with applications in data analysis," in *Proc. Adv. Comput. Math.*, vol. 45, 2019, pp. 1439–1468.

[28] S. Marx, E. Pauwels, T. Weisser, D. Henrion, and J. B. Lasserre, "Semi-algebraic approximation using Christoffel–Darboux kernel," *Constr. Approx.*, vol. 54, pp. 391–429, Dec. 2021.

[29] A. Devonport and M. Arcak, "Data-driven reachable set computation using adaptive Gaussian process classification and Monte Carlo methods," in *Proc. ACC*, 2020, pp. 2629–2634.

[30] A. Tebjou and G. Frehse, "Data-driven reachability using Christoffel functions and conformal prediction," in *Proc. Conform. Probab. Predict. Appl.*, 2023, pp. 194–213.

[31] 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 Trans. Autom. Control*, vol. 64, no. 7, pp. 2737–2752, Jul. 2019.

[32] C. Fan, B. Qi, S. Mitra, and M. Viswanathan, "DRYVR: Data-driven verification and compositional reasoning for automotive systems," in *Proc. Int. Conf. Comput. Aided Verif.*, 2017, pp. 441–461.

[33] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient SMT solver for verifying deep neural networks," in *Proc. CAV*, 2017, pp. 97–117.

[34] H.-D. Tran et al., "NNV: The neural network verification tool for deep neural networks and learning-enabled cyber–physical systems," in *Proc. CAV*, 2020, pp. 3–17.

[35] S. Shafieezadeh Abadeh, P. M. Mohajerin Esfahani, and D. Kuhn, "Distributionally robust logistic regression," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–9.

[36] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic Learning in a Random World*, vol. 29. New York, NY, USA: Springer, 2005.

[37] J. Lei and L. Wasserman, "Distribution-free prediction bands for non-parametric regression," *J. Roy. Stat. Soc., Ser. B, Stat. Methodol.*, vol. 76, no. 1, pp. 71–96, 2014.

[38] J. Lei, M. G'Sell, A. Rinaldo, R. J. Tibshirani, and L. Wasserman, "Distribution-free predictive inference for regression," *J. Amer. Stat. Assoc.*, vol. 113, no. 523, pp. 1094–1111, 2018.

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

[40] R. Luo et al., "Sample-efficient safety assurances using conformal prediction," in *Proc. Int. Workshop Algorithmic Found. Robot.*, 2022, pp. 149–169.

[41] R. J. Tibshirani, R. Foygel Barber, E. Candes, and A. Ramdas, "Conformal prediction under covariate shift," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.

[42] W. James and C. Stein, "Estimation with quadratic loss," in *Breakthroughs in Statistics: Foundations and Basic Theory*. New York, NY, USA: Springer, 1992, pp. 443–460.

[43] H. Gouk, E. Frank, B. Pfahringer, and M. J. Cree, "Regularisation of neural networks by enforcing Lipschitz continuity," *Mach. Learn.*, vol. 110, pp. 393–416, Feb. 2021.

[44] R. Koenker, *Quantile Regression*, vol. 38. Cambridge, U.K.: Cambridge Univ., 2005,

[45] H.-D. Tran et al., "Star-based reachability analysis of deep neural networks," in *Proc. Int. Symp. Formal Methods*, 2019, pp. 670–686.

[46] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel, "Efficient neural network robustness certification with general activation functions," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–10.

[47] N. J. Beaudry and R. Renner, "An intuitive proof of the data processing inequality," 2011, *arXiv:1107.0740*.

[48] 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 Trans. Embed. Comput. Syst.*, vol. 18, no. 5s, pp. 1–22, 2019.