PCA-DDReach: Efficient Statistical Reachability Analysis of Stochastic Dynamical Systems via Principal Component Analysis

Navid Hashemi NAVIDHAS@USC.EDU
Department of Computer Science, University of Southern California, Los Angeles, USA

Lars Lindemann LLINDEMA@USC.EDU
Department of Computer Science, University of Southern California, Los Angeles, USA

Jyotirmoy V. Deshmukh JDESHMUK@USC.EDU
Department of Computer Science, University of Southern California, Los Angeles, USA

Editors: G. Pappas, P. Ravikumar, S. A. Seshia

Abstract

This paper proposes a scalable data-driven algorithm for reachability analysis of complex cyber-physical systems (CPS) without requiring parametric models. Traditional methods rely on known physical dynamics, which are often unavailable due to system complexity or variability. Instead, we treat such systems as black boxes and use trajectory data to learn predictive models. To quantify prediction uncertainty and ensure safety, we integrate conformal inference (CI) — a statistical tool for probabilistic guarantees — with Principal Component Analysis (PCA) to reduce conservatism and enhance scalability. Our method constructs probabilistic reachable sets that are less conservative under distribution shifts compared to prior CI-based methods. We validate the approach on high-dimensional systems, including a 12D quadcopter and a 27D powertrain model, demonstrating improved accuracy and computational efficiency over existing techniques.

Keywords: Reachable set estimation, Conformal Inference, Principal Component Analysis

1. Introduction

System verification tools are crucial for ensuring correctness prior to testing, implementation, or deployment, particularly in expensive, high-risk, or safety-critical systems Zhang et al. (2023); Schilling et al. (2022); Komendera et al. (2012). In real-world implementations, we face two significant challenges for system verification: (1) assuming access to an underlying model for the system is often prohibitive, and (2) the presence of noise and uncertainty results in stochastic systems, requiring us to perform statistical verification to obtain probabilistic safety certificates. Data-driven statistical reachability analysis is a well-established tool for statistical verification, and has been applied to safety-critical problems across various domains, including, autonomy, medical imaging and CPS. Devonport et al. (2021); Fisac et al. (2018); Fan et al. (2017); Hashemi et al. (2024b).

In the context of data-driven statistical reachability analysis, given a user-specified threshold $\delta \in (0, 1)$, the goal is to produce a set that ensures that any trajectory during deployment lies within this set with a probability of no less than δ. As explained in the "*Related Work*" section, our work is motivated by the method proposed in Hashemi et al. (2024b). This methodology involves three main steps: (1) learning a deterministic surrogate model from sampled trajectories, (2) performing reachability analysis on the surrogate model, and (3) using robust conformal inference Cauchois et al. (2024) to calculate an inflating hypercube. This inflating hypercube quantifies the necessary expansion of the surrogate model's reachable set to provide provable probabilistic guarantees on the flowpipe that is obtained for the black-box oracle.

Nonetheless, the utilized technique in Hashemi et al. (2024b) to integrate robust conformal inference, can be adjusted to reduce the level of conservatism. Our first contribution in this paper is to address this issue, by combining robust conformal inference with Principal Component Analysis (PCA), and we show this adjustment results in tighter inflating hypercubes and probabilistic reachable sets. In Hashemi et al. (2024b), the authors assume a single model that maps the initial state to the entire trajectory[1]. However, this results in a large model, which limits the scalability of reachability analysis for extended horizons. Training such a large model becomes inefficient. Furthermore, the model's large size makes accurate methods for surrogate reachability infeasible, necessitating the use of conservative over approximation techniques. In response, our second contribution here is to address these challenges by presenting a new training strategy that effectively resolves for all of these listed scalability issues. We propose training a set of independent small models, each mapping the initial state to a separate sub-partition of the trajectory. This approach eliminates the need for iterating a model over time while maintaining smaller models that are more suitable for efficient training and efficient surrogate reachability analysis.

Related Work. In the context of data-driven reachability analysis, typically, to obtain the reachable set, a dataset of system trajectories needs to be available, e.g., from a black box simulator. In Devonport et al. (2021), the authors use Christoffel functions[2] to learn the model of the system from such a dataset and generate probabilistic reachable sets. This methodology has been extended in Tebjou et al. (2023) by incorporating conformal inference Vovk (2012) to improve its data efficiency. In Devonport and Arcak (2020a), a Gaussian process-based classifier is employed to learn a model that distinguishes between reachable and unreachable states to approximate the reachable set. In Devonport and Arcak (2020b); Dietrich et al. (2024) scenario optimization is used to generate probabilistic reachable sets. The method in Fisac et al. (2018) assumes partial knowledge of the system and leverages the dataset to perform statistical reachability analysis. Similarly, the work presented in Fan et al. (2017) uses the dataset to learn an exponential discrepancy function that estimates trajectory's sensitivity to uncertainty, enabling the computation of probabilistic reachable sets. Of our particular interest is the method proposed by Hashemi et al. (2024b), which suggests learning a system model on this dataset via ReLU neural networks and then conduct statistical reachability analysis via conformal inference. By leveraging the ability of neural networks to model complex, high-dimensional relationships alongside the data efficiency of conformal inference, this approach establishes an efficient and structured framework for statistical reachability analysis. Additionally, the probabilistic guarantees proposed by Hashemi et al. (2024b) remain valid even when there is a distribution shift between the training and deployment environments. This is the main reason we focus on extending this work instead of building on other existing methodologies.

Conformal inference (CI) is a data-efficient method for formally providing guarantees on the δ-quantile of distributions. This method involves sampling an i.i.d. scalar dataset, sorting the samples in ascending order, and demonstrating that one of the sorted samples represents the δ-quantile. The integration of CI with formal verification techniques has recently received noticeable interest, that is primarily due to its accuracy and level of scalability. For instance, Bortolussi et al. (2019) merges CI with neural state classifiers to develop a stochastic runtime verification algorithm. Lindemann et al. (2023); Zecchin et al. (2024) employ CI to guarantee safety in MPC control using a trained model. Tonkens et al. (2023) applies CI for planning with probabilistic safety guarantees, and Hashemi et al.

1. This approach is motivated by the fact that training a one-step model and iterating it over time leads to the well-known issue of cumulative errors
2. See Lasserre and Pauwels (2019); Marx et al. (2021) for more details about the Christoffel functions.

(2024b) integrates CI with existing neural network reachability techniques and provides a scalable reachability analysis on stochastic systems, see Lindemann et al. (2024) for a recent survey article.

Notation. We use bold letters to represent vectors and vector-valued functions, while caligraphic 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 $\mathcal{X}$. We present the structure of a feedforward neural network (FFNN) with ℓ hidden layers as an array $[n_0, n_1, \ldots n_{\ell+1}]$, 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 $\mathbb{R}^n$. We also denote $\lceil x \rceil$ as the smallest integer greater than $x \in \mathbb{R}$.

2. Preliminaries

2.1. Stochastic Dynamical Systems

Consider a set of random vectors $S_0, \ldots, S_{\mathrm{K}} \in \mathcal{S}$ indexed at times $0, \ldots, \mathrm{K}$ and with state space $\mathcal{S} \subseteq \mathbb{R}^n$. A realization of this stochastic process is a sequence of values $s_1, \ldots, s_{\mathrm{K}}$, denoted as system trajectory $\sigma_{s_0}^{\mathsf{real}}$. The joint distribution over $S_1, \ldots, S_{\mathrm{K}}$ is the trajectory distribution $\mathcal{D}_{S,\mathrm{K}}^{\mathsf{real}}$, while the marginal distribution of S_0 is known as the initial state distribution $\mathcal{W}$. It is assumed that $\mathcal{W}$ has support over a compact set of initial states $\mathcal{I}$, implying $\Pr[s_0 \notin \mathcal{I}] = 0$.

Training and Deployment Environments. In the training environment, we pre-record or simulate datasets to conduct reachability analysis. Conversely, the deployment environment refers to the real world where we apply our reachable sets. There is typically a difference between the distribution of trajectories in the training and deployment environments. We refer to this difference as distribution shift. In this paper, we assume that for a predefined distribution on initial states $s_0 \sim \mathcal{W}$, the real-world trajectories $\sigma_{s_0}^{\mathsf{real}}$ are sampled from $\sigma_{s_0}^{\mathsf{real}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{real}}$, whereas the simulated trajectories $\sigma_{s_0}^{\mathsf{sim}}$ are sampled from $\sigma_{s_0}^{\mathsf{sim}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{sim}}$.

2.2. Surrogate Model: Reachability & Error Analysis

A surrogate model $\mathcal{F} : \mathcal{I} \times \Theta \to \mathcal{S}^{\mathrm{K}}$, with trainable parameters $\theta \in \Theta$, can be trained by sampling K-step trajectories $\sigma_{s_0}^{\mathsf{sim}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{sim}}$ from the simulator to predict the trajectory $\sigma_{s_0}^{\mathsf{sim}} \in \mathcal{S}^{\mathrm{K}}$ given its initial state $s_0 \in \mathcal{I}$. We call this dataset $\mathcal{T}^{\mathsf{trn}}$, and we denote the predicted trajectory by,

$$\bar{\sigma}_{s_0} = \mathcal{F}(s_0\,; \theta), \text{ where, } \mathcal{F}(s_0\,; \theta) = \left[\mathsf{F}^1(s_0), \ldots, \mathsf{F}^n(s_0), \ldots, \mathsf{F}^{(\mathrm{K}-1)n+1}(s_0), \ldots, \mathsf{F}^{n\mathrm{K}}(s_0)\right]^\top \tag{1}$$

where, $\mathsf{F}^{(k-1)n+\ell}(s_0)$ is the ℓ^{th} state component at the k^{th} time-step in the trajectory[3]. Let $e_\ell \in \mathbb{R}^n$ denote the ℓ-th basis vector of $\mathbb{R}^n$. For a trajectory $s_1, \ldots s_{\mathrm{K}}$, and $s_0 \sim \mathcal{W}$, for $j = (k-1)n + \ell$, we define the prediction errors as,

$$R^j = e_\ell^\top s_k - \mathsf{F}^j(s_0), \quad k \in [\mathrm{K}], \ell \in [n]. \tag{2}$$

In this paper, we introduce the residual $\rho : \mathbb{R}^{n\mathrm{K}} \to \mathbb{R}_{\geq 0}$ as a function of the prediction errors R^j, where $j \in [n\mathrm{K}]$. In this section, we present a previously proposed example of such a function and later introduce an adjustment to reduce the conservatism in probabilistic reachability.

3. Here, the dimension and time steps are stacked into a single vector.

Definition 1 (Simulation & Real Residual Distribution) *If the residual is generated by* $\sigma_{s_0}^{\mathsf{sim}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{sim}}$*, we denote the residual distribution as* $\rho \sim \mathcal{J}_{S,\mathrm{K}}^{\mathsf{sim}}$ *where* $\mathcal{J}_{S,\mathrm{K}}^{\mathsf{sim}}$ *is the simulation residual distribution. Conversely, if the residual is generated by* $\sigma_{s_0}^{\mathsf{real}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{real}}$*, we denote it as* $\rho \sim \mathcal{J}_{S,\mathrm{K}}^{\mathsf{real}}$ *where* $\mathcal{J}_{S,\mathrm{K}}^{\mathsf{real}}$ *is the real residual distribution.*

We also utilize total variation, Takezawa (2005) as a metric to quantify their distribution shift, $\tau \geq 0$. In other word, $\tau = \mathsf{TV}(\mathcal{J}_{S,\mathrm{K}}^{\mathsf{sim}}, \mathcal{J}_{S,\mathrm{K}}^{\mathsf{real}})$, where TV refers to the total variation.

Surrogate Flowpipe and Star-Set. The surrogate flowpipe $\bar{X} \subset \mathbb{R}^{n\mathrm{K}}$ 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 $\mathcal{F}(s_0\,;\theta) \in \bar{X}$. Due to the recent achievements in verifying neural networks with ReLU activations, we limit ourselves to the choice of ReLU neural networks as our surrogate models, and we rely on the NNV toolbox from Tran et al. (2020) to compute the surrogate flowpipe. Although other activation functions can be used, we anticipate more conservative results if non-ReLU activation functions are utilized. We choose to use NNV in our analysis here, because it can yield accurate reachability results in settings where neural networks with ReLU activation function are used. The approach in Tran et al. (2020) employs star-sets (an extension of zonotopes) to represent the reachable set and utilizes two main methods: (1) the exact-star method, which performs precise but slow computations, and (2) the approx-star method, which is faster but it is more conservative.

Definition 2 (Star set Bak and Duggirala (2017)) *A star set* $Y \subset \mathbb{R}^d$ *is a tuple* $\langle c, V, P\rangle$ *where* $c \in \mathbb{R}^d$ *is the center,* $V = \{v_1, v_2, \ldots, v_m\}$ *is a set of* m *vectors in* $\mathbb{R}^d$ *called basis vectors, and* $P : \mathbb{R}^m \to \{\top, \bot\}$ *is a predicate. The basis vectors are arranged to form the star's* $d \times m$ *basis matrix. Given variables* $\mu_\ell \in \mathbb{R}, \ell = 1, \ldots, m$*, the set of states represented by the star is given as:*

$$Y = \left\{ y \mid y = c + \sum_{\ell=1}^{m} (\mu_\ell v_\ell) \text{ s.t. } P(\mu_1, \ldots, \mu_m) = \top \right\}. \tag{3}$$

2.3. Conformal Inference & Probabilistic Reachability

A key step toward probabilistic reachability is to provide a provable δ-quantile for the residual. Let $\rho_1 < \rho_2 < \ldots < \rho_L$ represent L different i.i.d. residuals sampled from $\mathcal{J}_{S,\mathrm{K}}^{\mathsf{sim}}$, and sorted in ascending order. Given a confidence probability, $\delta \in (0, 1)$, a provable tight upper bound for the δ-quantile of the residuals $\rho \sim \mathcal{J}_{S,\mathrm{K}}^{\mathsf{real}}$ is computable from samples $\rho_i \sim \mathcal{J}_{S,\mathrm{K}}^{\mathsf{sim}}, i \in [L]$, using robust conformal inference proposed in Cauchois et al. (2024) that is an extension of CI, proposed in Vovk (2012).

The theory of conformal inference states that for a new sample $\rho \sim \mathcal{J}_{S,\mathrm{K}}^{\mathsf{sim}}$, the rank $\ell := \lceil (L+1)\delta \rceil \leq L$ satisfies $\Pr[\rho < \rho_\ell] \geq \delta$. This implies that ρ_ℓ serves as a provable upper bound for the δ-quantile of $\mathcal{J}_{S,\mathrm{K}}^{\mathsf{sim}}$. However, this result does not extend to another residual $\rho \sim \mathcal{J}_{S,\mathrm{K}}^{\mathsf{real}}$, which is drawn from a different distribution. To address this distribution shift, the theory of robust conformal inference introduces an adjustment to conformal inference. It establishes that for any random variable $\rho \sim \mathcal{J}_{S,\mathrm{K}}^{\mathsf{real}}$ satisfying $\mathsf{TV}(\mathcal{J}_{S,\mathrm{K}}^{\mathsf{real}}, \mathcal{J}_{S,\mathrm{K}}^{\mathsf{sim}}) \leq \tau$ with a threshold $\tau > 0$, we have $\Pr[\rho < \rho_{\ell^*}] > \delta$, where

$$\ell^* := \lceil (L+1)(1 + 1/L)(\delta + \tau) \rceil, \quad \ell^* \leq L. \tag{4}$$

Thus, ρ_{ℓ^*} serves as an upper bound for the δ-quantile of $\mathcal{J}_{S,\mathrm{K}}^{\mathsf{real}}$.

Inflating Hypercube. In reachability analysis, the main purpose for the definition of the residual is to achieve a bounding region that will cover the random sequence of prediction errors, $\mathsf{PE} =$

$[R^1, R^2, \ldots, R^{n\mathrm{K}}]$ with a confidence $\delta \in (0, 1)$. In this case, as suggested by Cleaveland et al. (2024), the $\max()$ operator over the absolute value of all errors is a suitable choice. The authors in Hashemi et al. (2024b), for some positive constants $\alpha_j, j \in [n\mathrm{K}]$ (details on the choice of α_j can be found in Hashemi et al. (2024b)), define the residual

$$\rho := R = \max(\alpha_1|R^1|, \alpha_2|R^2|, \ldots, \alpha_{n\mathrm{K}}|R^{n\mathrm{K}}|), \tag{5}$$

and show that such a bounding region is achievable by computing an upper bound for the δ-quantile of residual, R. In other words, assuming R^* as the mentioned upper bound, we have,

$$\Pr[R < R^*] \geq \delta \Longleftrightarrow \Pr[P^*(R^1, \ldots, R^{n\mathrm{K}}) = \top] \geq \delta, \;\; P^*(R^1, \ldots, R^{n\mathrm{K}}) = \bigwedge_{j=1}^{n\mathrm{K}} (|R^j| < \frac{R^*}{\alpha_j}) \tag{6}$$

where R^* is efficiently obtainable via robust conformal inference. As defined in (6), the predicate P^* implies that for every component $R^j, j = 1, \ldots, n\mathrm{K}$ of the vector PE we have $-R^*/\alpha_j \leq R^j \leq R^*/\alpha_j$. This describes a hypercube which can be formulated as the following star set:

$$\delta X = \langle\, 0_{n\mathrm{K}\times 1},\; I_{n\mathrm{K}},\; P^*(R^1, \ldots, R^{n\mathrm{K}}) \,\rangle \subset \mathbb{R}^{n\mathrm{K}}. \tag{7}$$

Since $\Pr[P^* = \top] \geq \delta$, this star set serves as such a bounding region for PE. We refer to this bounding region as inflating hypercube.

δ-Confident Flowpipe & Probabilistic Reachability. For a given confidence probability $\delta \in (0,\ 1)$, and $s_0 \sim \mathcal{W}$, we say that $X \subseteq \mathbb{R}^{n\mathrm{K}}$ is a δ-confident flowpipe if for any random trajectory $\sigma_{s_0}^{\mathsf{real}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{real}}$, we have $\Pr[\sigma_{s_0}^{\mathsf{real}} \in X] \geq \delta$. In this paper, our ultimate goal is to propose a δ-confident flowpipe. To compute such a flowpipe, the authors in Hashemi et al. (2024b) suggest simulating a set of trajectories, $\sigma_{s_0}^{\mathsf{sim}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{sim}}, s_0 \sim \mathcal{W}$ and training a ReLU NN surrogate model $\mathcal{F}(s_0; \theta)$ on this dataset. This model will be utilized to compute for its surrogate reachset, $\bar{X} \subset \mathbb{R}^{n\mathrm{K}}$. They also sample a new set of trajectories $\sigma_{s_0}^{\mathsf{sim}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{sim}}, s_0 \sim \mathcal{W}$ for error analysis on $\mathcal{F}(s_0; \theta)$ through robust conformal inference to compute for another hypercube $\delta X \subset \mathbb{R}^{n\mathrm{K}}$, known as the inflating hypercube, that covers the prediction errors $R^j, j \in [n\mathrm{K}]$ for trajectories $\sigma_{s_0}^{\mathsf{real}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{real}}$, with a provable probabilistic guarantee, and finally they propose the following lemma to compute for the δ-confident flowpipe on $\sigma_{s_0}^{\mathsf{real}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{real}}, s_0 \sim \mathcal{W}$. See Hashemi et al. (2024b) for the proof.

Lemma 3 *Let $\bar{X}$ be a surrogate flowpipe of the surrogate model $\mathcal{F}$ for the set of initial conditions $\mathcal{I}$. Let $\mathsf{PE} := [R^1, R^2, \ldots, R^{n\mathrm{K}}]$ be the sequence of prediction errors for $\sigma_{s_0}^{\mathsf{real}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{real}}$, where $s_0 \sim \mathcal{W}$, and let δX be the inflating hypercube for PE such that $\Pr[\mathsf{PE} \in \delta X] > \delta$. Then the inflated reachset $X = \bar{X} \oplus \delta X$ is a δ-confident flowpipe for $\sigma_{s_0}^{\mathsf{real}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{real}}$ where $s_0 \sim \mathcal{W}$.*

2.4. Problem Definition

We are interested in computing a δ-confident flowpipe X from a set of trajectories $\sigma_{s_0}^{\mathsf{sim}}$ collected from $\mathcal{D}_{S,\mathrm{K}}^{\mathsf{sim}}$ so that X is also valid for all trajectories $\sigma_{s_0}^{\mathsf{real}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{real}}$ when the total variation between $\mathcal{J}_{S,\mathrm{K}}^{\mathsf{real}}$ and $\mathcal{J}_{S,\mathrm{K}}^{\mathsf{sim}}$ is less than $\tau > 0$. While we are motivated by the results in Hashemi et al. (2024b) which propose a solution to the stated problem, we note that their solution lacks scalability and accuracy that results in sometimes large levels of conservatism, i.e., the set X is unnecessarily large.

The primary sources of conservatism and inaccuracy in the methodology described in Hashemi et al. (2024b) stem from the training process for the surrogate model $\mathcal{F}(s_0\,; \theta)$ and the method used to compute the inflating hypercube δX. In the following sections, we address both issues and propose solutions to improve the accuracy and scalability of this approach for the reachability analysis.

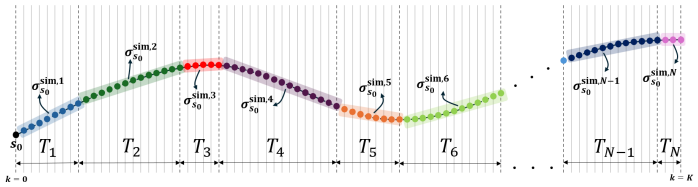

Figure 1: This figure shows the division of the trajectory into N different segments $\sigma_{s_0}^{\mathsf{sim},q}, q \in [N]$

3. Scalable and Accurate Data Driven Reachability Analysis

In this section, we introduce two key adjustments to the methodology of Hashemi et al. (2024b) to enhance scalability and reduce conservatism.

3.1. Improved Scalabilty and Accuracy for Training Surrogate Models

In this section, we introduce a new training strategy for the model $\mathcal{F}(s_0\,;\theta)$ that avoids the scalability issues arising from the surrogate model's large size when handling long time horizons, as encountered in Hashemi et al. (2024b). Figure 1 illustrates a realization of a trajectory $\sigma_{s_0}^{\mathsf{sim}} := s_1, \ldots, s_{\mathrm{K}}$ over the horizon K. In this figure, we divide the time horizon into N segments, each with length T_q, where $q \in [N]$. We denote each trajectory segment as $\sigma_{s_0}^{\mathsf{sim},q},\ q \in [N]$, defined as:

$$\sigma_{s_0}^{\mathsf{sim},q} := s_{t_q+1}, s_{t_q+2}, \ldots, s_{t_q+T_i}, \quad t_q = \sum_{\ell=1}^{q-1} T_\ell,\ t_1 = 0. \tag{8}$$

The key idea is to directly link each trajectory segment $\sigma_{s_0}^{\mathsf{sim},q},\ q \in [N]$ to its initial state $s_0 \in \mathcal{I}$. Thus, we can train an independent model $\mathcal{F}_q(s_0\,;\theta_q),\ q \in [N]$ for each segment, which predicts $\sigma_{s_0}^{\mathsf{sim},q}$ directly based on the initial state s_0. This model is also used to compute surrogate flowpipes for the trajectory segments $\bar{X}_q, q \in [N]$, representing the image of set $\mathcal{I}$ through the model $\mathcal{F}_q(\mathcal{I}\,;\theta_q)$.

Here are the reasons why this new training strategy for the trajectory $\sigma_{s_0}^{\mathsf{sim}}$ resolves all the scalability issues, we listed for Hashemi et al. (2024b), in the Introduction section. First and foremost, since all surrogate models $\mathcal{F}_q(s_0\,;\theta_q), q \in [N]$ are directly connected to the initial state, we do not need to iterate them sequentially over the time horizon for prediction of states in $\sigma_{s_0}^{\mathsf{sim},q}, q \in [N]$, thus eliminating the problem of cumulative errors over the time horizon. Furthermore in this setting, the size of the models $\mathcal{F}_q(s_0\,;\theta_q), q \in [N]$ can be small. The small size of the models allows for efficient computation of surrogate flowpipes $\bar{X}_q := \mathcal{F}_q(\mathcal{I}\,;\theta_q)$ for each segment via exact-star reachability analysis[4]. Additionally, the smaller models $\mathcal{F}_q(s_0\,;\theta_q), q \in [N]$ enable efficient training of accurate models for each trajectory segment. Furthermore, although we have to train more models using this technique, we can train them in parallel as they are totally independent processes.

Once the surrogate flowpipes $\bar{X}_q = \langle \bar{c}_q, \bar{V}^q, \bar{P}_q \rangle$, for $q \in [N]$, are obtained as star sets, the surrogate flowpipe for the entire trajectory forms another star set, $\bar{X} = \langle \bar{c}, \bar{V}, \bar{P} \rangle$. This global surrogate flowpipe is constructed by concatenating all individual star sets $\bar{X}_q$, for $q \in [N]$, which implies: $\bar{c} = \left[\bar{c}_1^\top, \ldots, \bar{c}_N^\top\right]^\top$, $\bar{V} = \mathbf{diag}\left(\bar{V}^1, \ldots, \bar{V}^N\right)$ and $\bar{P} = \bigwedge_{q=1}^{N} \bar{P}_q$.

3.2. Accurate Inflating Hypercubes via Principal Component Analysis

Principal Component Analysis (PCA) is a mathematical technique used to identify the principal directions of variation in a dataset. Given a dataset of L data points, $x_i \in \mathbb{R}^n, i \in [L]$, PCA estimates

4. However, if the set of initial states $\mathcal{I}$ is large and the partitioning of $\mathcal{I}$ is not scalable (high dimensional states), we remain limited to using approx star. Nevertheless, even for this case, the small size of the model significantly reduces the conservatism of approx star.

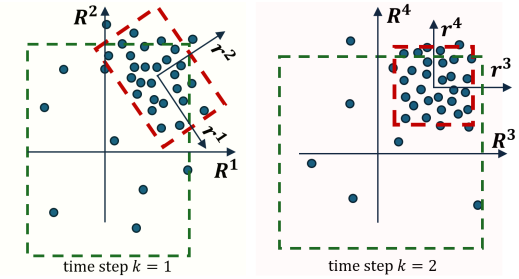

Figure 2: The figure shows the projection of prediction errors for two-dimensional states over a horizon of $\mathrm{K} = 2$. The left figure illustrates the projection on the (R^1, R^2) axes (e.g., $k = 1$), and the right figure displays the projection on the (R^3, R^4) axes (e.g., $k = 2$). This figure provides a comparison between the inflating hypercubes for a confidence level $\delta \in (0, 1)$, generated by the PCA approach (red hypercubes) and the method proposed in Hashemi et al. (2024b) (green hypercubes). It clearly demonstrates the superior accuracy of the PCA technique compared to the other method. The principal axes for $k = 1, 2$ are (r^1, r^2) and (r^3, r^4), respectively.

the covariance matrix, $\Sigma \succeq 0, \Sigma \in \mathbb{R}^{n \times n}$ of data points $x_i, i \in [L]$. The eigenvectors of Σ, known as **principal components**, define the directions along which the data exhibits the highest variance, while the corresponding eigenvalues quantify the magnitude of variance along each direction. These principal components form an orthonormal basis that aligns with the natural structure of the data, providing key insights into its intrinsic geometric properties.

The authors in Hashemi et al. (2024b) use the function in (5) to define the residual ρ and compute the corresponding inflating hypercube δX. However, their technique imposes two conservative constraints. First, the center of the hypercube is always located at the origin. Second, the edges of the hypercube are restricted to be aligned with the direction of the trajectory state components.

To address these limitations, we propose an adjustment on the definition of the residual ρ in equation (5), that enables us to overcome these issues. We integrate the concepts of Conformal Inference (CI) and Principal Component Analysis (PCA) in our new definition for the residual. This approach provides the principal axes as the orientation of inflating hypercube and noticeably reduces its size. In other words, our approach enhances the accuracy of conformal inference by manipulating the coordinate system, inspired by PCA. However, in the context of CI, altering the coordinate system has also been addressed in other works, such as Tumu et al. (2024); Sharma et al. (2024).

To obtain the principal axes, given the simulated trajectories from the training dataset $\sigma^{\mathsf{sim}}_{s_{0,i}} \in \mathcal{T}^{\mathsf{trn}}, i \in [|\mathcal{T}^{\mathsf{trn}}|]$, for each segment $q \in [N]$, we use the trajectory segment $\sigma^{\mathsf{sim},q}_{s_{0,i}}$ and its corresponding surrogate model $\mathcal{F}_q(s_{0,i}; \theta_q)$ to compute the corresponding set of prediction errors. Specifically, for each segment $q \in [N]$ and data index $i \in [|\mathcal{T}^{\mathsf{trn}}|]$, we collect:

$$\mathsf{PE}^q_i = \left[R_i^{t_q n+1}, R_i^{t_q n+2}, \ldots, R_i^{(t_q+T_q)n}\right] \tag{9}$$

and approximate the average and covariance as follows:

$$\overline{\mathsf{PE}}^q = \frac{\sum_{i=1}^{|\mathcal{T}^{\mathsf{trn}}|} \mathsf{PE}^q_i}{|\mathcal{T}^{\mathsf{trn}}|}, \ \Sigma^q = \frac{\sum_{i=1}^{|\mathcal{T}^{\mathsf{trn}}|} \left(\mathsf{PE}^q_i - \overline{\mathsf{PE}}^q\right)^\top \left(\mathsf{PE}^q - \overline{\mathsf{PE}}^q\right)}{|\mathcal{T}^{\mathsf{trn}}|}. \tag{10}$$

We then apply spectral decomposition on the covariance matrix Σ^q to obtain the array of eigenvectors $\mathsf{V}^q \in \mathbb{R}^{T_q n \times T_q n}$. Here, the principal axes for the trajectory segment $q \in [N]$ are centered on $\overline{\mathsf{PE}}^q$, and are aligned with the eigenvectors $\mathsf{V}^q_\ell, \ell \in [T_q n]$, which are the ℓ-th columns of the matrix V^q.

Given the initial state, $s_0 \sim \mathcal{W}$, assume a trajectory $s_1, \ldots, s_{\mathrm{K}}$ that is not necessarily sampled from $\mathcal{D}^{\mathsf{sim}}_{S,\mathrm{K}}$ and also is not necessarily a member of the training dataset. For any segment $q \in [N]$ of this trajectory, we map its vector of prediction errors $\mathsf{PE}^q = \left[R^{t_q n+1}, R^{t_q n+2}, \ldots, R^{(t_q+T_q)n}\right]$ to the principal axes. We do this with a linear map, as,

$$\left[r^{t_q n+1}, r^{t_q n+2}, \ldots, r^{(t_q+T_q)n}\right] = \mathsf{V}^{q\top}(\mathsf{PE}^q - \overline{\mathsf{PE}}^q), \tag{11}$$

and utilize the parameters $r^j,\ t_q n+1 \leq j \leq (t_q+T_q)n$ to define the residual. Collecting the mapped prediction errors for all segments $q \in [N]$, we propose our definition for residual as follows:

$$\rho := \max\left(\frac{|r^1|}{\omega_1},\ \frac{|r^2|}{\omega_2},\ \ldots\ ,\ \frac{|r^{n\mathrm{K}}|}{\omega_{n\mathrm{K}}}\right) \tag{12}$$

where the scaling factors $\omega_j, j \in [n\mathrm{K}]$ are the maximum magnitude of parameters $r^j_i, i \in |\mathcal{T}^{\mathsf{trn}}|$, that are obtained from the training dataset. In other words,

$$\omega_j = \max(|r^j_1|,\ |r^j_2|,\ \ldots,\ |r^j_{|\mathcal{T}^{\mathsf{trn}}|}|),\ \ j \in [n\mathrm{K}]. \tag{13}$$

Although we use the training dataset $\mathcal{T}^{\mathsf{trn}}$ to determine hyperparameters, V^q, $\overline{\mathsf{PE}}^q$, and ω_j for defining the residual, reusing $\mathcal{T}^{\mathsf{trn}}$ to generate the inflating hypercube with robust conformal inference violates CI rules. Thus, in order to generate the inflating hypercube, we first sample a new i.i.d. set of trajectories from the training environment $\mathcal{D}^{\mathsf{sim}}_{S,\mathrm{K}}$, which we denote as the calibration dataset.

Definition 4 (Calibration Dataset) *The calibration dataset $\mathcal{R}^{\mathsf{calib}}$ is defined as:*

$$\mathcal{R}^{\mathsf{calib}} = \left\{ (s_{0,i}, \rho_i) \;\middle|\; \begin{array}{l} s_{0,i} \sim \mathcal{W},\ \sigma^{\mathsf{sim}}_{s_{0,i}} \sim \mathcal{D}^{\mathsf{sim}}_{S,\mathrm{K}}, \\ \rho_i = \max(\frac{|r^1_i|}{\omega_1}, \ldots, \frac{|r^{n\mathrm{K}}_i|}{\omega_{n\mathrm{K}}}) \end{array} \right\}. \tag{14}$$

Here, $\sigma^{\mathsf{sim}}_{s_{0,i}}, i \in |\mathcal{R}^{\mathsf{calib}}|$ refers to the trajectory starting at the i^{th} initial state sampled from $\mathcal{W}$, generated from $\mathcal{D}^{\mathsf{sim}}_{S,\mathrm{K}}$. The parameters r^j_i are also as defined in equation (11).

Consider sorting the i.i.d. residuals $\rho_i \sim \mathcal{J}^{\mathsf{sim}}_{S,\mathrm{K}}$ collected in the calibration dataset $\mathcal{R}^{\mathsf{calib}}$ by their magnitude: $\rho_1 < \rho_2 < \ldots < \rho_{|\mathcal{R}^{\mathsf{calib}}|}$. Our goal is to provide a provable upper bound for the δ-quantile of a residual $\rho \sim \mathcal{J}^{\mathsf{real}}_{S,\mathrm{K}}$, given knowledge of a radius $\tau > 0$ such that the total variation $\mathsf{TV}(\mathcal{J}^{\mathsf{real}}_{S,\mathrm{K}}, \mathcal{J}^{\mathsf{sim}}_{S,\mathrm{K}}) < \tau$. In this case, robust conformal inference Cauchois et al. (2024) suggests using the rank ℓ^* from equation (4) and selecting $\rho^*_{\delta,\tau} := \rho_{\ell^*}$ as an upper bound for the residual's δ-quantile. In other words, for a residual $\rho \sim \mathcal{J}^{\mathsf{real}}_{S,\mathrm{K}}$, we have $\Pr[\rho < \rho^*_{\delta,\tau}] > \delta$.

Proposition 5 *Assume $\rho^*_{\delta,\tau}$ is the δ-quantile of $\rho \sim \mathcal{J}^{\mathsf{real}}_{S,\mathrm{K}}$, computed over the residuals $\rho_i \sim \mathcal{J}^{\mathsf{sim}}_{S,\mathrm{K}}$ from the calibration dataset $\mathcal{R}^{\mathsf{calib}}$ where $\mathsf{TV}(\mathcal{J}^{\mathsf{real}}_{S,\mathrm{K}}, \mathcal{J}^{\mathsf{sim}}_{S,\mathrm{K}}) < \tau$. For the residual $\rho = \max\left(\frac{|r^1|}{\omega_1}, \frac{|r^2|}{\omega_2}, \ldots, \frac{|r^{n\mathrm{K}}|}{\omega_{n\mathrm{K}}}\right)$ sampled from the distribution $\mathcal{J}^{\mathsf{real}}_{S,\mathrm{K}}$, and the trajectory division setting, $T_q, q \in [N]$, it holds that, $\Pr\left[P(r^1, \ldots, r^{n\mathrm{K}}) = \top\right] > \delta$, where,*

$$P(r^1, \ldots, r^{n\mathrm{K}}) = \bigwedge_{q=1}^{N} P_q(r^{t_q n+1}, \ldots, r^{(t_q+T_q)n}),\ \ P_q(r^{t_q n+1}, \ldots, r^{(t_q+T_q)n}) := \bigwedge_{j=t_q n+1}^{(t_q+T_q)n} \left(-\omega_j \rho^*_{\delta,\tau} \leq r^j \leq \omega_j \rho^*_{\delta,\tau}\right), \tag{15}$$

and r^j is the mapped version of prediction errors $R^j,\ j \in [n\mathrm{K}]$ on principal axes.

Proof The proof follows as the residual ρ is the maximum of the normalized version of parameters $r^j, j \in n\mathrm{K}$ so that

$$\rho = \max\left(\frac{|r^1|}{\omega_1}, \frac{|r^2|}{\omega_2}, \ldots, \frac{|r^{n\mathrm{K}}|}{\omega_{n\mathrm{K}}}\right) \iff \bigwedge_{j=1}^{n\mathrm{K}} \left[|r^j| \leq \rho\omega_j\right]. \tag{16}$$

Now, since $\Pr[\rho \leq \rho^*_{\delta,\tau}] \geq \delta$ as well as $\rho < \rho^*_{\delta,\tau} \iff |r^j| < \rho^*_{\delta,\tau}\omega_j$ for all $j \in [n\mathrm{K}]$, we can claim that $\Pr[\bigwedge_{j=1}^{n\mathrm{K}}[|r^j| \leq \rho^*_{\delta,\tau}\omega_j]] \geq \delta$. The guarantee proposed in (15) is the reformulation of this results in terms of the division setting, $T_q, q \in [N]$. ■

Referring to Def. 2, and using the predicates $P_q(r^{t_q n+1}, \ldots, r^{(t_q+T_q)n}), q \in [N]$ from Proposition 5 we can introduce the inflating hypercubes, $\delta X_q, q \in [N]$ as star sets. In other words, from equation (11), for any $q \in [N]$ we can compute the prediction errors as,

$$\mathsf{PE}^q = \overline{\mathsf{PE}}^q + \mathsf{V}^q \left[r^{t_q n+1}, r^{t_q n+2}, \ldots, r^{(t_q+T_q)n}\right] \tag{17}$$

which implies $\delta X_q = \langle \overline{\mathsf{PE}}^q, \mathsf{V}^q, P_q(r^{t_q n+1}, \ldots, r^{(t_q+T_q)n})\rangle$, and thus the concatenation of the star sets δX_q, serves as an inflating hypercube for PE. Therefore, we denote the inflating hypercube of the entire trajectory by $\delta X = \langle \overline{\mathsf{PE}}, V, P\rangle$ where $V = \mathbf{diag}\left(V^1, \ldots, V^N\right)$ and $P = \bigwedge_{q=1}^{N} P_q$.

Finally, based on Lemma 3, the δ-confident flowpipe on the entire trajectory X can be obtained through the inflation of surrogate flowpipe $\bar{X}$ with the inflating hypercube δX, i.e., $X = \bar{X} \oplus \delta X$.

Remark 6 *Figure 2 shows the advantage of the PCA approach by illustrating prediction errors of a 2-dimensional state over 2 consecutive time steps*[5]*. This figure also provides a schematic of the inflating hypercubes generated by our residual definition and those generated by the definition proposed in (5). The primary function of the vectors $\overline{\mathsf{PE}}^q, q \in [N]$ is to reposition the surrogate reachsets $\bar{X}_q$ to locations that require minimal inflation, and the main role of V^q is to further reduce the necessary level of inflation.*

4. Numerical Evaluation

To simulate real-world systems capable of producing actual trajectory data, we employ stochastic difference equation-based models with additive Gaussian noise to account for uncertainties in observations, dynamics, and potential modeling errors. Our theoretical guarantees apply to any real-world distribution $\sigma^{\mathsf{real}}_{s_0} \in \mathcal{D}^{\mathsf{real}}_{S,\mathrm{K}}$, provided that the residual distribution shift $\mathsf{TV}(\mathcal{J}^{\mathsf{sim}}_{S,\mathrm{K}}, \mathcal{J}^{\mathsf{real}}_{S,\mathrm{K}})$ is below a given threshold τ. Here we evaluate our results on three different case studies. The first two experiments involve a 12-D quadcopter with $\tau = 0$, while the final experiment focuses on a 27-D powertrain model where the distribution shift is upper-bounded at $\tau = 4\%$. In Experiment 1, we compare our approach with Hashemi et al. (2024b). However, since that methodology does not scale well for trajectories with large number of time-steps, K, we address Experiments 2 and 3, only using the methodology proposed in this paper. The next two sections provide a general overview of the experiments, with detailed information deferred to the Appendix. Table 1 also presents the details of the numerical results.

5. Division setting: $\mathrm{K} = 2,\ n = 2,\ N = 2$, and $T_1 = T_2 = 1$.

	Specification			Training		Surrogate Reachability		Inflating Hypercube	
Exp #:	δ	τ	#	avg runtime	$\lvert \mathcal{T}^{\mathsf{trn}} \rvert$	#	avg runtime(method)	runtime	$\lvert \mathcal{R}^{\mathsf{calib}} \rvert$
1	99.99%	0	100	39.6 sec	42,000	100	1.43 sec (E)	2.08 sec	20,000
2	99.99%	0	451	33.65 sec	20,000	4501	0.030 sec (E)	116.58 sec	20,000
3	95%	4%	400	40.6 sec	10,000	4000	0.064 sec (A)	142.02 sec	10,000

Table 1: Shows details of the experiments. The models are trained in parallel with 18 CPU workers. Thus, the average training runtime may vary by selecting different number of workers. The words E, and A represent exact-star and approx-star, respectively.

4.1. 12-Dimensional Quadcopter

We consider the 12-dimensional quadcopter system under stochastic conditions for two different case studies. Trajectories are simulated using two ODE models from Hashemi et al. (2024b) and Hashemi et al. (2024a) as our simulators. The state variables include the quadcopter's position (x_1, x_2, x_3), velocity (x_4, x_5, x_6), Euler angles (x_7, x_8, x_9) representing roll, pitch, and yaw angles, and angular velocities (x_{10}, x_{11}, x_{12}). We also include zero mean additive Gaussian process noise $v \sim \mathcal{N}(0_{12\times1}, \Sigma_v)$ to the simulators with covariance $\Sigma_v = \mathbf{diag}\left([0.05 \times \mathbf{1}_{1\times6},\ 0.01 \times \mathbf{1}_{1\times6}]^2\right)$. In both examples, the set of initial states $s_0 \in \mathcal{I}$ is taken from the cited papers, with the distribution $s_0 \sim \mathcal{W}$ being uniform.

4.2. 27-Dimensional Powertrain

We use the powertrain system proposed by Althoff and Krogh (2012) as our simulator, which is a hybrid system with three modes. To introduce stochastic conditions, we add zero-mean Gaussian process noise, $v \sim \mathcal{N}(\mathbf{0}_{27\times1}, \Sigma_v)$, where $\Sigma_v = \mathbf{diag}\left(10^{-5} \times \mathbf{1}_{1\times27}\right)$, to their simulator, defining the distribution $\sigma^{\mathsf{sim}}_{s_0} \sim \mathcal{D}^{\mathsf{sim}}_{S,\mathrm{K}}$. This system is highly sensitive to noise, which is a key reason we addressed it in this paper. For example, Figure 6 shows the angular velocity of the last rotating mass, x_{27}, both with and without noise. Following Althoff and Krogh (2012), we simulate trajectories with a sampling time of $\delta t = 0.0005$ over a horizon of 2 seconds ($\mathrm{K} = 4000$), and consider their set of initial states $\mathcal{I}$ [6]. We also define the trajectory division setting as $N = 4000$, $T_q = 1, q \in [N]$. The ReLU NN models are with structure $[27, 54, 27]$. To reduce the training runtime, we again follow the analytical interpolation strategy we introduced for Experiment 2.

5. Acknowledgements

This work was partially supported by the National Science Foundation through the following grants: CAREER award (SHF-2048094), CNS-1932620, CNS-2039087, FMitF-1837131, CCF-SHF-1932620, IIS-SLES-2417075, funding by Toyota R&D and Siemens Corporate Research through the USC Center for Autonomy and AI, an Amazon Faculty Research Award, and the Airbus Institute for Engineering Research. This work does not reflect the views or positions of any organization listed.

6. Conclusion

We introduced a scalable technique for reachability in real-world settings. Our results demonstrate that integrating PCA with Conformal inference significantly enhances the accuracy of error analysis. We validated the effectiveness of our approach across three distinct high-dimensional environments.

6. In this case the set $\mathcal{I}$ proposed in Althoff and Krogh (2012) is a large and high dimensional set, thus the exact star does not scale, and we are restricted to utilized approx star for surrogate reachability.

References

Matthias Althoff and Bruce H Krogh. Avoiding geometric intersection operations in reachability analysis of hybrid systems. In *Proceedings of the 15th ACM international conference on Hybrid Systems: Computation and Control*, pages 45–54, 2012.

Stanley Bak and Parasara Sridhar Duggirala. Simulation-equivalent reachability of large linear systems with inputs. In *International Conference on Computer Aided Verification*, pages 401–420. Springer, 2017.

Luca Bortolussi, Francesca Cairoli, Nicola Paoletti, Scott A Smolka, and Scott D Stoller. Neural predictive monitoring. In *Runtime Verification: 19th International Conference, RV 2019, Porto, Portugal, October 8–11, 2019, Proceedings 19*, pages 129–147. Springer, 2019.

Maxime Cauchois, Suyash Gupta, Alnur Ali, and John C Duchi. Robust validation: Confident predictions even when distributions shift. *Journal of the American Statistical Association*, pages 1–66, 2024.

Matthew Cleaveland, Insup Lee, George J Pappas, and Lars Lindemann. Conformal prediction regions for time series using linear complementarity programming. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 20984–20992, 2024.

Alex Devonport and Murat Arcak. Data-driven reachable set computation using adaptive gaussian process classification and monte carlo methods. In *Proc. of ACC*, pages 2629–2634, 2020a.

Alex Devonport and Murat Arcak. Estimating reachable sets with scenario optimization. In *Learning for dynamics and control*, pages 75–84. PMLR, 2020b.

Alex Devonport, Forest Yang, Laurent El Ghaoui, and Murat Arcak. Data-driven reachability analysis with christoffel functions. In *Proc. of CDC*, pages 5067–5072, 2021.

Elizabeth Dietrich, Alex Devonport, and Murat Arcak. Nonconvex scenario optimization for data-driven reachability. In *6th Annual Learning for Dynamics & Control Conference*, pages 514–527. PMLR, 2024.

Chuchu Fan, Bolun Qi, Sayan Mitra, and Mahesh Viswanathan. Dryvr: Data-driven verification and compositional reasoning for automotive systems. In *International Conference on Computer Aided Verification*, pages 441–461. Springer, 2017.

Jaime F Fisac, Anayo K Akametalu, Melanie N Zeilinger, Shahab Kaynama, Jeremy Gillula, and Claire J Tomlin. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control*, 64(7):2737–2752, 2018.

Navid Hashemi, Bardh Hoxha, Danil Prokhorov, Georgios Fainekos, and Jyotirmoy V Deshmukh. Scaling learning-based policy optimization for temporal logic tasks by controller network dropout. *ACM Transactions on Cyber-Physical Systems*, 8(4):1–28, 2024a.

Navid Hashemi, Lars Lindemann, and Jyotirmoy V Deshmukh. Statistical reachability analysis of stochastic cyber-physical systems under distribution shift. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 43(11):4250–4261, 2024b.

Erik Komendera, Daniel Scheeres, and Elizabeth Bradley. Intelligent computation of reachability sets for space missions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 2299–2304, 2012.

Jean B Lasserre and Edouard Pauwels. The empirical christoffel function with applications in data analysis. *Advances in Computational Mathematics*, 45:1439–1468, 2019.

Lars Lindemann, Matthew Cleaveland, Gihyun Shim, and George J Pappas. Safe planning in dynamic environments using conformal prediction. *IEEE Robotics and Automation Letters*, 2023.

Lars Lindemann, Yiqi Zhao, Xinyi Yu, George J Pappas, and Jyotirmoy V Deshmukh. Formal verification and control with conformal prediction. *arXiv preprint arXiv:2409.00536*, 2024.

Swann Marx, Edouard Pauwels, Tillmann Weisser, Didier Henrion, and Jean Bernard Lasserre. Semi-algebraic approximation using christoffel–darboux kernel. *Constructive Approximation*, pages 1–39, 2021.

Christian Schilling, Marcelo Forets, and Sebastián Guadalupe. Verification of neural-network control systems by integrating taylor models and zonotopes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8169–8177, 2022.

Apoorva Sharma, Sushant Veer, Asher Hancock, Heng Yang, Marco Pavone, and Anirudha Majumdar. Pac-bayes generalization certificates for learned inductive conformal prediction. *Advances in Neural Information Processing Systems*, 36, 2024.

Kunio Takezawa. *Introduction to nonparametric regression*. John Wiley & Sons, 2005.

Abdelmouaiz Tebjou, Goran Frehse, et al. Data-driven reachability using christoffel functions and conformal prediction. In *Conformal and Probabilistic Prediction with Applications*, pages 194–213. PMLR, 2023.

Sander Tonkens, Sophia Sun, Rose Yu, and Sylvia Herbert. Scalable safe long-horizon planning in dynamic environments leveraging conformal prediction and temporal correlations. In *Long-Term Human Motion Prediction Workshop, International Conference on Robotics and Automation*, 2023.

Hoang-Dung Tran, Xiaodong Yang, Diego Manzanas Lopez, Patrick Musau, Luan Viet Nguyen, Weiming Xiang, Stanley Bak, and Taylor T Johnson. Nnv: the neural network verification tool for deep neural networks and learning-enabled cyber-physical systems. In *Proc. of CAV*, pages 3–17, 2020.

Renukanandan Tumu, Matthew Cleaveland, Rahul Mangharam, George Pappas, and Lars Lindemann. Multi-modal conformal prediction regions by optimizing convex shape templates. In *6th Annual Learning for Dynamics & Control Conference*, pages 1343–1356. PMLR, 2024.

Vladimir Vovk. Conditional validity of inductive conformal predictors. In *Asian conference on machine learning*, pages 475–490. PMLR, 2012.

Matteo Zecchin, Sangwoo Park, and Osvaldo Simeone. Forking uncertainties: Reliable prediction and model predictive control with sequence models via conformal risk control. *IEEE Journal on Selected Areas in Information Theory*, 2024.

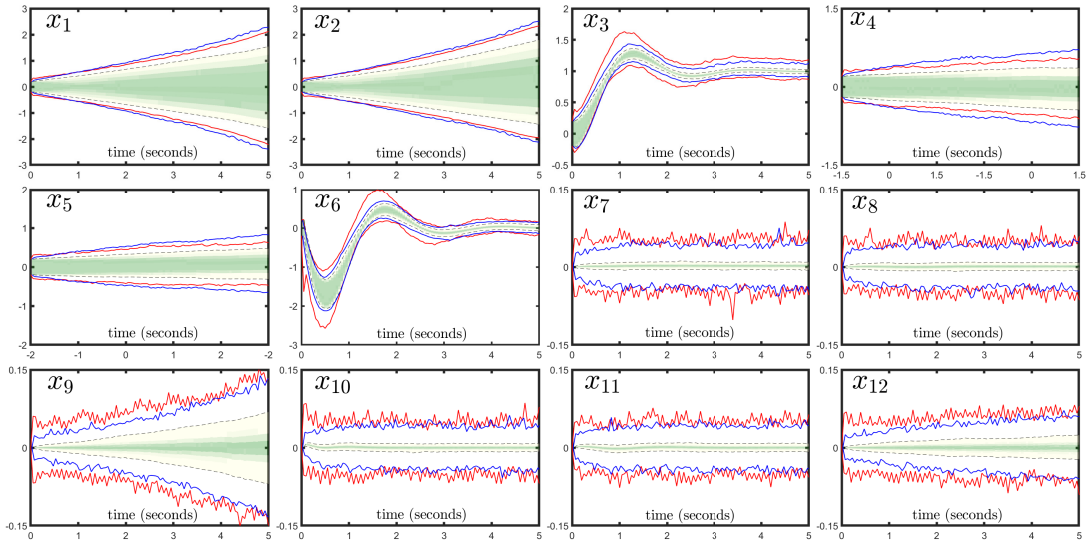

Figure 3: Shows the comparison with Hashemi et al. (2024b). The blue and red borders are projections of our and their δ-confident flowpipes respectively with $\delta = 99.99\%$. The shaded regions show the density of the trajectories from $\mathcal{T}^{\mathsf{trn}}$.

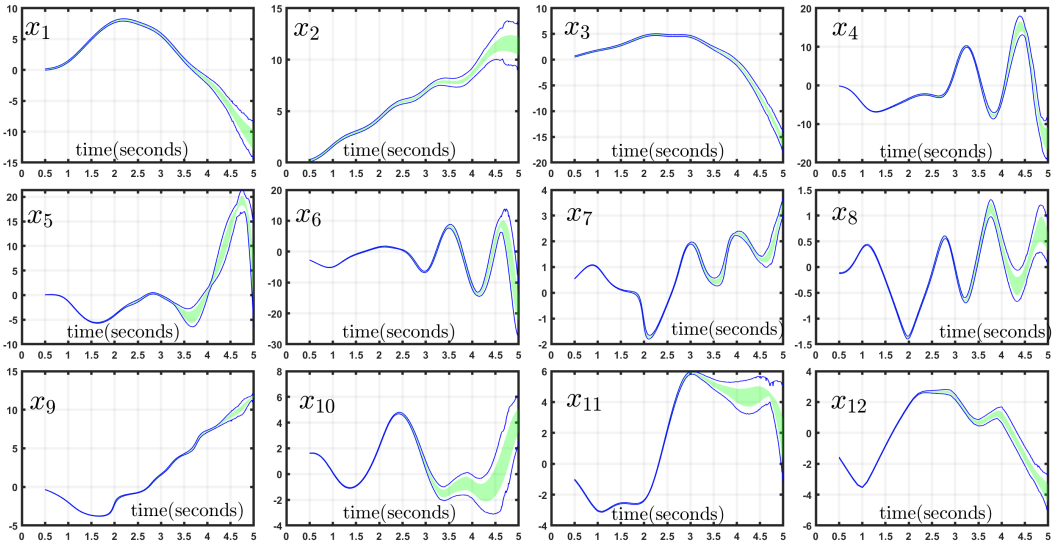

Figure 4: Shows the projection of our δ-confident flowpipe on each component of the trajectory state. The shaded area are the simulation of trajectories from $\mathcal{T}^{\mathsf{trn}}$.

Chi Zhang, Wenjie Ruan, and Peipei Xu. Reachability analysis of neural network control systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 15287–15295, 2023.

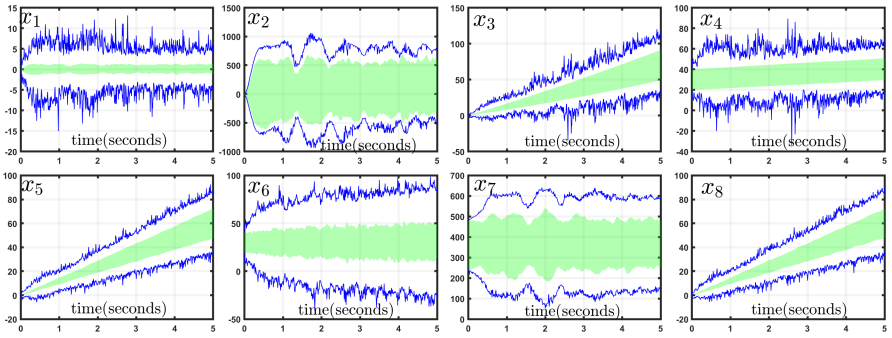

Figure 5: Shows the projection of our δ-confident flowpipe on the first 8 components of the trajectory state. There is a shift between the distribution of deployment and training environments. The shaded area are the trajectories sampled from the deployment environment.

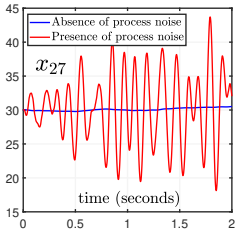

Figure 6: Shows the comparison of angular velocity of the last rotating mass in presence and absence of the process noise.

Appendix A. Detail of the Experiments

A.1. Experiment 1:[Comparison with Hashemi et al. (2024b)]

Here we address Experiment 2 from Hashemi et al. (2024b) for comparison of the results. In this experiment, a quadcopter hovers at a specific elevation, and its trajectories are simulated over a horizon of $\mathrm{K} = 100$ time steps, with a sampling time of $\delta t = 0.05$. The δ-confident flowpipe has a confidence level of $\delta = 99.99\%$. Compared to Hashemi et al. (2024b), our approach achieves a higher level of accuracy. This improvement is due to our training strategy, which allows us to use exact-star for surrogate reachability, and our PCA-based technique, which results in smaller inflating hypercubes. In this experiment, we use a trajectory division setting of $N = 100$, $T_q = 1$, for $q \in [N]$, with ReLU neural network surrogate models structured as $[12, 24, 12]$. Figure 3 shows the projection of the flowpipe on each state in comparison with the results of Hashemi et al. (2024b), and Table 1 shows the detail of the experiment.

A.2. Experiment 2: [Sequential Goal Reaching Task]

In this example, we consider the quadcopter scenario described in Hashemi et al. (2024a), where a controller is designed to ensure that the machine accomplishes a sequential goal-reaching task. We also include the previously mentioned process noise in the simulator to include stochasticity. Given the quadcopter's tendency for unpredictable behavior, we significantly reduce the sampling time in this instance. The trajectories are sampled at a frequency of 1 KHz over a 5-second horizon, resulting in 5000 time steps. Our objective is to perform reachability analysis for time steps 500 through 5000 with the level of confidence $\delta = 99.99\%$. We propose a trajectory division setting of $N = 5000$ with $T_q = 1$ for $q \in [N]$. To reduce the runtime for model training, we employ analytical interpolation. Specifically, we select every tenth time step for model training, and for $i \in 50, 51, \ldots, 500$, and $j \in [10]$, we regenerate all the ReLU neural network surrogate models using the following formula:

$$\mathcal{F}_{10i+j} = (1 - 0.1j)\mathcal{F}_{10i} + 0.1j\mathcal{F}_{10(i+1)} \tag{18}$$

where the models $\mathcal{F}_{10i}$ have a structure of $[12, 24, 12]$. We then utilize these regenerated ReLU neural network surrogate models for surrogate reachability through exact star reachability analysis, as well as error analysis using PCA-based conformal inference. Figure 4 shows the resulting flowpipe and Table 1 shows the detail of the experiment.

A.3. Experiment 3: [Reachability with Distribution shift]

Let's assume the real world trajectories $\sigma_{s_0}^{\mathsf{real}} \sim \mathcal{D}_{S,\mathrm{K}}^{\mathsf{real}}$ are such that its covariance of process noise is 20% larger than Σ_v. In this case, the threshold $\tau = 0.04$ is a valid upper-bound for $\mathsf{TV}(\mathcal{J}_{S,\mathrm{K}}^{\mathsf{sim}}, \mathcal{J}_{S,\mathrm{K}}^{\mathsf{real}})$. In this experiment, given the threshold, τ we generate a δ-confident flowpipe for $\sigma_{s_0}^{\mathsf{real}}$ with $\delta = 95\%$. Figure 5 shows the projection of our computed flowpipe on the first 8 components of states, and Table 1 shows the detail of the experiment.