Efficient Interaction-Aware Interval Analysis of Neural Network Feedback Loops

Saber Jafarpour*, *Member, IEEE*, Akash Harapanahalli*, *Graduate Student Member, IEEE*, and Samuel Coogan, *Senior Member, IEEE*

***Abstract*— In this paper, we propose a computationally efficient framework for interval reachability of systems with neural network controllers. Our approach leverages inclusion functions for the open-loop system and the neural network controller to embed the closed-loop system into a larger-dimensional embedding system, where a single trajectory over-approximates the original system's behavior under uncertainty. We propose two methods for constructing closed-loop embedding systems, which account for the interactions between the system and the controller in different ways. The interconnection-based approach considers the worst-case evolution of each coordinate separately by substituting the neural network inclusion function into the open-loop inclusion function. The interaction-based approach uses novel Jacobian-based inclusion functions to capture the first-order interactions between the open-loop system and the controller by leveraging state-of-the-art neural network verifiers. Finally, we implement our approach in a Python framework called `ReachMM` to demonstrate its efficiency and scalability on benchmarks and examples ranging to 200 state dimensions.**

***Index Terms*— Reachability analysis, Inclusion functions, Neural networks, Interconnected systems.**

I. Introduction

Problem Description and Motivations: Recent advances in machine learning are bringing learning algorithms into the heart of safety-critical control systems, such as autonomous vehicles, manufacturing sectors, and robotics. For control systems, these learning algorithms often act in the closed-loop setting, as direct feedback controllers [1] or motion planners [2]. Learning-based closed-loop controllers can improve the performance of systems while reducing their computational burden for online implementations as compared with more traditional optimization-based approaches. However, despite their impressive performance, learning models are known to be overly sensitive to input disturbances [3]: a small input perturbation can lead to comparatively large changes in their output. This issue amplifies in feedback settings, as disturbances can accumulate in the closed-loop. As a result, ensuring reliability of learning algorithms is an essential challenge in their integration into safety-critical systems.

Typical learning architectures involve high-dimensional nonlinear function approximators, such as neural networks, necessitating special tools for their input-output analysis. The machine learning and control community have made significant progress in analyzing the safety of neural networks in isolation, including efficient and sound input-output bounds, worst-case adversarial guarantees, and sampling-based stochastic guarantees (cf. [4]). However, most of these existing frameworks for standalone neural networks do not address the unique challenges for closed-loop analysis—namely information propagation, non-stationarity of the bounds, and complex interactions between the system and the controller. In these cases, it is essential to understand and capture the nature of the interactions between the system and the neural network. Recently, several frameworks have emerged for safety verification of learning algorithms in the closed-loop. These frameworks usually incorporate neural network verification algorithms into the closed-loop safety analysis by studying their interactions with the open-loop system. However, most of these existing frameworks are either limited to a specific class of systems or learning algorithms, or they impose significant computational burdens, rendering them unsuitable for runtime safety verification.

Literature Review: Safety verification of nonlinear dynamical systems without learning-enabled systems is a classical problem and is typically solved using reachability analysis tools, although significant challenges remains even in this setting. Reachability of nonlinear systems has been studied using optimization-based methods such as the Hamilton-Jacobi approach [5] and the level set approach [6]. Several computationally tractable approaches including the ellipsoidal method [7] and the zonotope method [8] have been developed for reachability analysis of dynamical systems. We refer to [9] for a recent review of state-of-the-art reachability analysis techniques. Interval analysis is a classical framework for computing interval functional bounds [10] which has been successfully used for reachability analysis of dynamical systems [11]–[13]. A closely-related body of literature is the mixed monotone theory, which extends the classical monotone system theory by separating the cooperative and competitive effect of the dynamics [14], [15]. Recently, mixed monotone theory has been used as an efficient framework for reachability analysis of dynamical systems [16]–[19].

Rigorous verification of standalone neural networks has

* These authors contributed equally.

This work was supported in part by the National Science Foundation under grants 1749357 and 2219755 and the Air Force Office of Scientific Research under Grant FA9550-23-1-0303.

Saber Jafarpour is with Department of Electrical, Computer, and Energy Engineering, University of Colorado Boulder, Boulder, Colorado, USA (e-mail: saber.jafarpour@colorado.edu})

Akash Harapanahalli and Samuel Coogan are with School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA (e-mail: {aharapan, sam.coogan}@gatech.edu})

been studied using abstract interpretation techniques such as Reluplex [20] and Neurify [21], using interval bound propagation methods [22], [23], and convex-relaxation approaches such as LipSDP [24] and CROWN [25] and its variants [26]. The simplest method for studying reachability of neural network controlled systems is via an *input-output approach*: using existing techniques for estimating the output of the neural network, the input-output approach performs reachability analysis of the closed-loop system by substituting the full output range of the neural network as the input set of the open-loop system. Examples of this approach include NNV [27], and simulation-guided interval analysis [28]. This approach is generally computationally efficient and applicable to general forms of neural networks and control systems, but it can lead to overly-conservative estimates of reachable sets [29, Section 2.1]. The reason for this over-conservatism is that this approach essentially treats neural networks controllers as adversaries and, thus, cannot capture the beneficial interactions between the controller and the system. An alternative framework is the *functional approach*, which is based on composing function approximations of the system and the neural network controller. Examples of this approach for linear systems include using linear programming in ReachLP [30], using semidefinite programming in Reach-SDP [31], and using mixed integer programming in [32]. For nonlinear systems, the functional approach has been used in ReachNN [33], Sherlock [29], Verisig 2.0 [34], POLAR [35] and JuliaReach [36], [37]. Functional methods are able to capture beneficial interactions between the neural network controller and the system, but they are often computationally burdensome and generally do not scale well to large-scale systems.

Contributions: In this paper, we introduce a general and computationally efficient framework for studying reachability of continuous-time closed-loop systems with nonlinear dynamics and neural networks controllers. First, we review the notion of inclusion function from classical interval analysis to over-approximate the input-output behavior of functions. Our first minor result states that if an inclusion function is chosen to be minimal, then it provides tighter over-approximations of the output behavior than a Lipschitz bound of the function. For a given function, we study different approaches for constructing inclusion functions. In particular, we propose a novel Jacobian-based cornered inclusion function, which is specially amenable to analysis of closed-loop systems. We further show that this class of inclusion functions plays a critical role in integrating the existing off-the-shelf neural network verification algorithms into our framework.

Second, using the notion of inclusion functions, we develop a general approach for interval analysis of continuous-time dynamical systems. The key idea is to use the inclusion functions to embed the original dynamical system into a higher dimensional embedding system. Then trajectories of the embedding system can be used to provide hyper-rectangular over-approximation for reachable sets of the original system. Our approach unifies and extends several existing approaches for interval analysis of dynamical systems. Notably, our novel proof technique, based on the classical monotone comparison Lemma, allows for accuracy comparison between different inclusion functions.

As the culminating contribution of this paper, we develop a computationally efficient framework for reachability analysis of nonlinear systems with neural network controllers. The key in our framework is a careful combination of the inclusion function for the open-loop system and the inclusion function for the neural network controller to obtain an inclusion function for the closed-loop system. We propose two methods for combining the open-loop inclusion function and the neural network inclusion function. The second method, called the interaction-based approach, constructs a closed-loop embedding system using a Jacobian-based inclusion function for the open-loop system and local affine bounds for the neural network. Compared to the interconnection-based approach, the interaction-based method completely captures the first order interactions between the system and the neural network.

Finally, we implement our approach in a Python toolbox called `ReachMM` and perform several numerical experiments to show the efficiency and accuracy of our framework. For several linear and nonlinear benchmarks, we show that our method beats the state-of-the-art methods. Moreover, we study scalability of our approach and compare it with the state-of-the-art methods on a platoon of vehicles with up to 200 state variables. Compared to our conference paper [38], this paper provides a more general framework for interval reachability using inclusion functions, introduces the interaction-based approach to capture the first order interactions between the system and the neural network controller, and includes a suite of numerical experiments. We would like to highlight that, in this paper, we focus on continuous-time dynamical systems. However, with minor modifications, our framework can be used to analyze reachability of discrete-time systems.

II. Mathematical Preliminary and Notations

For every two vectors $v, w \in \mathbb{R}^n$ and every subset $S \subseteq \{1, \dots, n\}$, we define the vector $v_{[S:w]} \in \mathbb{R}^n$ by $\left(v_{[S:w]}\right)_j = \begin{cases} v_j & j \notin S \\ w_j & j \in S. \end{cases}$. Given a matrix $B \in \mathbb{R}^{n \times m}$, we denote the nonnegative part of B by $[B]^+ = \max(B, 0)$ and the nonpositive part of B by $[B]^- = \min(B, 0)$. Given a square matrix $A \in \mathbb{R}^{n \times n}$ the Metzler and non-Metzler part of A are denoted by $[A]^{\mathrm{M}}$ and $[A]^{\mathrm{nM}}$, respectively, where $([A]^{\mathrm{M}})_{ij} = \begin{cases} A_{ij} & A_{ij} \geq 0 \text{ or } i = j \\ 0 & \text{otherwise,} \end{cases}$ and $[A]^{\mathrm{nM}} = A - [A]^{\mathrm{M}}$. Given a map $g : \mathbb{R}^n \to \mathbb{R}^m$, the ℓ_∞-norm Lipschitz bound of g is the smallest $L \in \mathbb{R}_{\geq 0}$ such that

$$\|g(x) - g(y)\|_\infty \leq L \|x - y\|_\infty, \qquad \text{for all } x, y \in \mathbb{R}^n.$$

We denote the ℓ_∞-norm Lipschitz constant of g by $\mathrm{Lip}_\infty(g)$. For a differentiable map $g : \mathbb{R}^n \to \mathbb{R}^m$, we denote its Jacobian at point $x \in \mathbb{R}^n$ by $Dg(x)$. Consider the dynamical system

$$\dot{x} = f(x, w), \qquad \text{for all } t \in \mathbb{R}_{\geq 0}, \tag{1}$$

where $x \in \mathbb{R}^n$ is the state of the system and $w \in \mathbb{R}^q$ is the disturbance. Given a piecewise continuous disturbance $t \mapsto w(t)$ and an initial condition $x_0 \in \mathbb{R}^n$, the trajectory of (1)

starting from x_0 is denoted by $\phi_f(t, x_0, w(t))$. Let $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{W} \subseteq \mathbb{R}^q$, then the reachable set of (1) starting from the initial set $\mathcal{X}$ and with the disturbance set $\mathcal{W}$ is defined by:

$$\mathcal{R}_f(t, \mathcal{X}, \mathcal{W}) = \left\{ \begin{matrix} \phi_f(t, t_0, x_0, w),\ \forall x_0 \in \mathcal{X}, \\ w : \mathbb{R} \to \mathcal{W} \text{ piecewise cont.} \end{matrix} \right\} \quad (2)$$

The set $\mathcal{X}$ is *robustly forward invariant* if $\mathcal{R}_f(t, \mathcal{X}, \mathcal{W}) \subseteq \mathcal{X}$ for all $t \geq 0$. The system (1) is continuous-time monotone on $\mathcal{X} \subseteq \mathbb{R}^n$ if, for every $i \in \{1, \ldots, n\}$, every $x \leq y \in \mathcal{X}$ with $x_i = y_i$, and every $u \leq v$, we have $f_i(x, u) \leq f_i(y, v)$. If a dynamical system (1) is monotone on $\mathcal{X}$ and $\mathcal{X}$ is robustly forward invariant, then its trajectories preserve the standard partial order by time, i.e., for every two trajectories $t \mapsto x(t)$ and $t \mapsto y(t)$ of the systems (1), if $x(0) \leq y(0)$, then $x(t) \leq y(t)$ for all $t \in \mathbb{R}_{\geq 0}$ [39]. The *southeast* partial order $\leq_{\text{SE}}$ on $\mathbb{R}^{2n}$ is defined by $\left[\begin{smallmatrix} x \\ \widehat{x} \end{smallmatrix}\right] \leq_{\text{SE}} \left[\begin{smallmatrix} y \\ \widehat{y} \end{smallmatrix}\right]$ if and only if $x \leq y$ and $\widehat{y} \leq \widehat{x}$. We also define $\mathcal{T}^{2n}_{\geq 0} = \{\left[\begin{smallmatrix} x \\ \widehat{x} \end{smallmatrix}\right] \in \mathbb{R}^{2n} \mid x \leq \widehat{x}\}$ and $\mathcal{T}^{2n}_{\leq 0} = \{\left[\begin{smallmatrix} x \\ \widehat{x} \end{smallmatrix}\right] \in \mathbb{R}^{2n} \mid x \geq \widehat{x}\}$ and $\mathcal{T}^{2n} = \mathcal{T}^{2n}_{\geq 0} \bigcup \mathcal{T}^{2n}_{\leq 0}$.

III. Problem Statement

We consider a system described by:

$$\dot{x} = f(x, u, w), \qquad t \in \mathbb{R}_{\geq 0}, \quad (3)$$

where $x \in \mathbb{R}^n$ is the state of the system, $u \in \mathbb{R}^p$ is the control input, and $w \in \mathbb{R}^q$ is the disturbance. We assume that $f : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}^n$ is a parameterized vector field and the state feedback is parameterized by a k-layer feed-forward neural network controller $N : \mathbb{R}^n \to \mathbb{R}^p$ defined by:

$$\begin{aligned} \xi^{(0)} &= x \\ \xi^{(i)} &= \phi^{(i-1)}(W^{(i-1)}\xi^{(i-1)} + b^{(i-1)}), \quad i \in \{1, \ldots, k\} \\ u &= W^{(k)}\xi^{(k)}(x) + b^{(k)} := N(x), \end{aligned} \quad (4)$$

where n_i is the number of neurons in the ith layer, $W^{(i-1)} \in \mathbb{R}^{n_i \times n_{i-1}}$ is the weight matrix of the ith layer, $b^{(i-1)} \in \mathbb{R}^{n_i}$ is the bias vector of the ith layer, $\xi^{(i)}(y) \in \mathbb{R}^{n_i}$ is the ith layer hidden variable, and $\phi^{(i-1)} : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i}$ is the ith layer diagonal activation function satisfying $0 \leq \frac{\phi_j^{(i-1)}(x) - \phi_j^{(i-1)}(y)}{x - y} \leq 1$ for every $j \in \{1, \ldots, n_i\}$. One can show that a large class of activation functions including ReLU, leaky ReLU, sigmoid, and tanh satisfies this condition (after a possible re-scaling of their co-domain). With this neural network feedback controller, the closed-loop system is given by:

$$\dot{x} = f(x(t), N(x(t)), w) := f^{\text{c}}(x(t), w). \quad (5)$$

We assume that $\mathcal{X}_0 \subseteq \mathbb{R}^n$ is the initial set of states for the closed-loop system (5). This set can represent the uncertainties in the starting state of the system or localization errors in estimating the current state of the system. Moreover, we assume that the disturbance w belongs to the set $\mathcal{W} \subseteq \mathbb{R}^n$ representing the model uncertainty or exogenous disturbances on the system. In this paper, we focus on the target-avoid problem as defined below:

Target-Avoid Problem. Given a target set $\mathcal{G} \subset \mathbb{R}^n$, and an unsafe set $\mathcal{S}_{\text{unsafe}} \subset \mathbb{R}^n$ and a time interval $[0, T]$, check if the closed-loop system (5) reaches the target $\mathcal{G}$ at time T while avoiding the unsafe region $\mathcal{S}_{\text{unsafe}}$. Mathematically,

(i) $\mathcal{R}_{f^c}(T, \mathcal{X}_0, \mathcal{W}) \subseteq \mathcal{G}$,
(ii) $\mathcal{R}_{f^c}(t, \mathcal{X}_0, \mathcal{W}) \bigcap \mathcal{S}_{\text{unsafe}} = \emptyset$, for all $t \in [0, T]$.

The target-avoid problem is a classical and prevalent notion of safety in many engineering applications, especially those involving safety-critical systems [9]. Moreover, diverse objectives including multiagent coordination, complex planning specified with formal languages and logics, and classical control-theoretic criteria such as robust invariance and stability can be achieved by concatenating and combining different instantiations of the target-avoid problem.

In general, computing the exact reachable sets of the closed-loop system (5) is not computationally tractable. Our approach to solve the target-avoid safety problem is based on constructing a computationally efficient over-approximation $\overline{\mathcal{R}}_{f^c}(t, \mathcal{X}_0, \mathcal{W})$ of the reachable set of the closed-loop system. Then, reaching the target is guaranteed by $\overline{\mathcal{R}}_{f^c}(T, \mathcal{X}_0, \mathcal{W}) \subseteq \mathcal{G}$ and avoiding the unsafe region is guaranteed when $\overline{\mathcal{R}}_{f^c}(t, \mathcal{X}_0, \mathcal{W}) \bigcap \mathcal{S}_{\text{unsafe}} = \emptyset$, for every $t \in [0, T]$.

IV. Interval Analysis and Inclusion Functions

In this section, we develop a theoretical framework for interval reachability analysis of arbitrary mappings. The key element in our framework is the notion of inclusion function from interval arithmetic [10], [40].

A. Inclusion Functions

Given a nonlinear input-output map $g : \mathbb{R}^n \to \mathbb{R}^m$ and an interval input, the inclusion function of g provides interval bounds on the output of g.

Definition 1 (Inclusion function [10])**.** Given a map $g : \mathbb{R}^n \to \mathbb{R}^m$, the function $\mathsf{G} = \left[\begin{smallmatrix} \underline{\mathsf{G}} \\ \overline{\mathsf{G}} \end{smallmatrix}\right] : \mathcal{T}^{2n}_{\geq 0} \to \mathcal{T}^{2m}_{\geq 0}$ is

(i) an *inclusion function* for g if, for every $x \leq \widehat{x}$ and every $z \in [x, \widehat{x}]$, we have

$$\underline{\mathsf{G}}(x, \widehat{x}) \leq g(z) \leq \overline{\mathsf{G}}(x, \widehat{x}). \quad (6)$$

(ii) a $[y, \widehat{y}]$-*localized inclusion function* if (6) holds for every $[x, \widehat{x}] \subseteq [y, \widehat{y}]$ and every $z \in [x, \widehat{x}]$.

Moreover, an inclusion function G for g is

(iii) *monotone* if, for every $\left[\begin{smallmatrix} x \\ \widehat{x} \end{smallmatrix}\right] \leq_{\text{SE}} \left[\begin{smallmatrix} y \\ \widehat{y} \end{smallmatrix}\right]$, we have $\mathsf{G}(x, \widehat{x}) \leq_{\text{SE}} \mathsf{G}(y, \widehat{y})$;

(iv) *thin* if, for every $x \in \mathbb{R}^n$, $\underline{\mathsf{G}}(x, x) = \overline{\mathsf{G}}(x, x) = g(x)$;

(v) *minimal* if, for every $x \leq \widehat{x}$, the interval $[\underline{\mathsf{G}}(x, \widehat{x}), \overline{\mathsf{G}}(x, \widehat{x})]$ is the smallest interval containing $g([x, \widehat{x}])$.

Given a nonlinear map $g : \mathbb{R}^n \to \mathbb{R}^m$, one can show that its minimal inclusion function $\mathsf{G}^{\min} = \left[\begin{smallmatrix} \underline{\mathsf{G}}^{\min} \\ \overline{\mathsf{G}}^{\min} \end{smallmatrix}\right]$ is given by:

$$\underline{\mathsf{G}}^{\min}_i(x, \widehat{x}) = \inf_{z \in [x, \widehat{x}]} g_i(z), \quad \overline{\mathsf{G}}^{\min}_i(x, \widehat{x}) = \sup_{z \in [x, \widehat{x}]} g_i(z), \quad (7)$$

for every $i \in \{1, \ldots, n\}$. The next theorem shows that the minimal inclusion function provides sharper over-approximations compared to Lipschitz bounds.

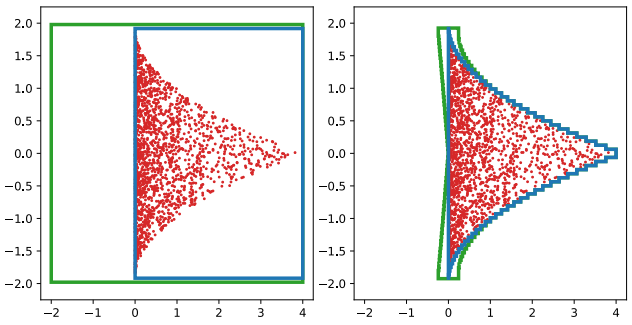

Fig. 1. **Left:** `npinterval` is used to generate interval approximations for a function $g(x_1, x_2) = [(x_1 + x_2)^2, 4\sin((x_1 - x_2)/4)]^\top$ using two different natural inclusion functions. Blue: using the inclusion functions for elementary functions $x \mapsto x^2$ and $x \mapsto \sin(x)$ in [41, Appendix A, Table 1], Green: rewriting $g(x_1, x_2) = [x_2^2 + 2x_1x_2 + x_2^2, 4\sin(x_1/4)\cos(x_2/4) - 4\cos(x_1/4)\sin(x_2/4)]^\top$ and obtaining a different natural inclusion function based on composition of elementary inclusion functions in [41, Appendix A, Table 1]. The approximations are generated using the initial set $[-1, 1] \times [-1, 1]$, and 2000 uniformly sampled ouptuts are shown in red. **Right:** The same function is analyzed, with the same two natural inclusion functions, but the initial set is partitioned into 1024 uniform sections, and the union of the interval approximations are shown.

Theorem 1 (Minimal inclusion functions)**.** *Consider the function $g : \mathbb{R}^n \to \mathbb{R}^m$ and the mapping $\mathsf{G}^{\min} = \begin{bmatrix} \underline{\mathsf{G}}^{\min} \\ \overline{\mathsf{G}}^{\min} \end{bmatrix}$ defined by (7). Then, for every $x \leq \widehat{x}$, we have*

$$\|\underline{\mathsf{G}}^{\min}(x, \widehat{x}) - \overline{\mathsf{G}}^{\min}(x, \widehat{x})\|_\infty \leq \mathrm{Lip}_\infty(g)\|x - \widehat{x}\|_\infty.$$

Proof. Let $i \in \{1, \ldots, k\}$ be such that $\|\overline{\mathsf{G}}^{\min}(x, \widehat{x}) - \underline{\mathsf{G}}^{\min}(x, \widehat{x})\|_\infty = \left|\overline{\mathsf{G}}_i^{\min}(x, \widehat{x}) - \underline{\mathsf{G}}_i^{\min}(x, \widehat{x})\right|$. Note that since g is continuous and the box $[x, \widehat{x}]$ is compact, there exist $\eta^*, \xi^* \in [x, \widehat{x}]$ such that

$$\max_{y \in [x, \widehat{x}]} g_i(y) = g_i(\eta^*), \qquad \min_{y \in [x, \widehat{x}]} g_i(y) = g_i(\xi^*).$$

This implies that $\|\overline{\mathsf{G}}^{\min}(x, \widehat{x}) - \underline{\mathsf{G}}^{\min}(x, \widehat{x})\|_\infty = |g_i(\eta^*) - g_i(\xi^*)| \leq \|g(\xi^*) - g(\eta^*)\|_\infty \leq \mathrm{Lip}_\infty(g)\|\xi^* - \eta^*\|_\infty \leq \mathrm{Lip}_\infty(g)\|x - \widehat{x}\|_\infty$. □

B. Designing Inclusion Functions

Given a function g, finding a suitable inclusion function for g is a central problem in interval analysis. We review three approaches for constructing inclusion functions.

Natural inclusion function: As we showed in the previous section, the minimal inclusion function can be computed using the optimization problem (7). However, for general functions, solving this optimization problem is not tractable. One feasible approach is to find the minimal inclusion function for a handful of elementary functions for which the optimization problem (7) is solvable. Then, for an arbitrary function, a natural inclusion function can be obtained by expressing it as a finite composition of operators and elementary functions [10, Theorem 2] [41, Theorem 2]. Several software packages exist for automatically computing natural inclusions functions. In [41], we introduce the software package `npinterval`[1] that implements intervals as a native data type within the Python toolbox `numpy` [42], yielding flexibility, efficiency through vectorization across arrays, and canonical constructions of natural inclusion functions. An example of an inclusion function computed using `npinterval` [41] is illustrated in Figure 1.

Jacobian-based inclusion functions: Given a continuously differentiable function $g : \mathbb{R}^n \to \mathbb{R}^m$, one can use upper and lower bounds on the Jacobian of g to construct inclusion functions for g. These inclusion functions are often derived from the Taylor expansion of the function g around a certain point. Most commonly, this point is taken as the midpoint of an interval, leading to, in particular, the centered inclusion function [10, §2.4.3] and the mixed-centered inclusion function [10, §2.4.4]. In the next theorem, we develop a Jacobian-based inclusion function obtained by linearization around corners of an interval. As demonstrated in the sequel, such cornered inclusion functions turn out to be particularly amenable to analysis of closed-loop control systems.

Proposition 1 (Jacobian-based cornered inclusion function)**.** *Given a continuously differentiable function $g : \mathbb{R}^n \to \mathbb{R}^m$ such that $Dg(z) \in [\underline{J}_{[x,\widehat{x}]}, \overline{J}_{[x,\widehat{x}]}]$, for every $z \in [x, \widehat{x}]$. The function $\mathsf{G}^{\mathrm{jac}} : \mathcal{T}_{\geq 0}^{2n} \to \mathcal{T}_{\geq 0}^{2m}$ defined by*

$$\mathsf{G}^{\mathrm{jac}}(x, \widehat{x}) = \begin{bmatrix} -[\underline{J}_{[x,\widehat{x}]}]^- & [\underline{J}_{[x,\widehat{x}]}]^- \\ -[\overline{J}_{[x,\widehat{x}]}]^+ & [\overline{J}_{[x,\widehat{x}]}]^+ \end{bmatrix} \begin{bmatrix} x \\ \widehat{x} \end{bmatrix} + \begin{bmatrix} g(x) \\ g(x) \end{bmatrix}$$

is a inclusion function for g.

Proof. Since g is continuously differentiable, for every $z \in [x, \widehat{x}]$, we have $g(z) = g(x) + \int_0^1 Dg(tz + (1-t)x)(z - x)dt$. Thus, for every $x \in [x, \widehat{x}]$,

$$\begin{aligned} g(z) &\leq g(x) + \overline{J}_{[x,\widehat{x}]}(z - x) \\ &= g(x) + [\overline{J}_{[x,\widehat{x}]}]^+(z - x) + [\overline{J}_{[x,\widehat{x}]}]^-(z - x) \\ &\leq g(x) + [\overline{J}_{[x,\widehat{x}]}]^+(\widehat{x} - x) = \overline{\mathsf{G}}^{\mathrm{jac}}(x, \widehat{x}) \end{aligned}$$

where the first inequality holds because $z - x \geq \mathbb{0}_n$, the second equality holds by $\overline{J}_{[x,\widehat{x}]} = [\overline{J}_{[x,\widehat{x}]}]^+ + [\overline{J}_{[x,\widehat{x}]}]^-$, and the last inequality holds because $z - x \leq \widehat{x} - x$ and $z - x \geq \mathbb{0}_n$. Similarly, one can show that, for every $z \in [x, \widehat{x}]$, $g(z) \geq g(x) + [\underline{J}_{[x,\widehat{x}]}]^-(\widehat{x} - x) = \underline{\mathsf{G}}^{\mathrm{jac}}(x, \widehat{x})$, completing the proof. □

Remark 1. (The role of corner points) The proof of Proposition 1 is based on Taylor expansion of g around the corner point x. The proposition extends straightforwardly to the Taylor expansion of g around other corner points, i.e., around any point y such that $y_i \in \{x_i, \widehat{x}_i\}$, resulting in different inclusion functions. Both the Jacobian-based cornered inclusion function (Proposition 1) and the Jacobian-based centered inclusion function [10, §2.4] are obtained using the Taylor expansion of the original function. In general, they are not comparable (cf. Example 1 and Table I). However, for functions that are monotone in their entries, the Jacobian-based cornered inclusion function will lead to the minimal inclusion function while

[1]The code for `npinterval` is available at https://github.com/gtfactslab/npinterval

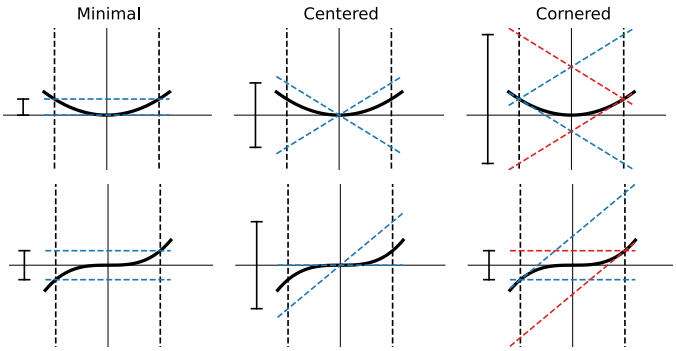

Fig. 2. Pictorial comparison between the minimal (equation (7)), the Jacobian-based centered ([10, §2.4.3]), and the Jacobian-based cornered (Proposition 1) inclusion functions for $\boldsymbol{f(x) = x^2}$ and $\boldsymbol{f(x) = x^3}$ on the interval $\boldsymbol{[-1, 1]}$. Note that, for the monotone function $\boldsymbol{f(x) = x^3}$, the intersection of the two Jacobian-based cornered inclusion functions (shown by blue and red dashed lines) leads to the minimal inclusion function.

the Jacobian-based centered inclusion function is often non-minimal. Figure 2 provides a pictorial comparison between the minimal, the centered, and the cornered inclusion functions.

The accuracy of the Jacobian-based cornered inclusion function can be improved at a cost of a slightly more complicated formulation using mixed differentiation in the same way that the Jacobian-based centered inclusion function is improved with the mixed-centered inclusion function [10, §2.4.4]. The idea is to use a step-by-step differentiation of the function with respect to each variable.

Proposition 2 (Mixed Jacobian-based cornered inclusion function)**.** *Given a continuously differentiable function $g : \mathbb{R}^n \to \mathbb{R}^m$ such that $Dg(z) \in [\underline{J}_{[x,\widehat{x}]}, \overline{J}_{[x,\widehat{x}]}]$, for every $z \in [x, \widehat{x}]$. The function $\mathsf{G}^{\mathrm{jac-m}} : \mathcal{T}^{2n}_{\geq 0} \to \mathcal{T}^{2m}_{\geq 0}$ defined by*

$$\mathsf{G}^{\mathrm{jac-m}}(x,\widehat{x}) = \begin{bmatrix} -[\underline{M}]^- & [\underline{M}]^- \\ -[\overline{M}]^+ & [\overline{M}]^+ \end{bmatrix} \begin{bmatrix} x \\ \widehat{x} \end{bmatrix} + \begin{bmatrix} g(x) \\ g(x) \end{bmatrix}$$

is an inclusion function for g, where the ith columns of $\underline{M}$ and $\overline{M}$ are defined by

$$\underline{M}_i = (\underline{J}_{[x,\widehat{x}_{[R_i:x]}]})_i, \qquad \overline{M}_i = (\overline{J}_{[x,\widehat{x}_{[R_i:x]}]})_i,$$

with $R_i = \{i+1, \ldots, n\}$, for every $i \in \{1, \ldots, n\}$. Moreover, for every $x \leq z \leq \widehat{x}$, we have

$$\underline{\mathsf{G}}^{\mathrm{jac}}(x,\widehat{x}) \leq \underline{\mathsf{G}}^{\mathrm{jac-m}}(x,\widehat{x}), \qquad \overline{\mathsf{G}}^{\mathrm{jac-m}}(x,\widehat{x}) \leq \overline{\mathsf{G}}^{\mathrm{jac}}(x,\widehat{x}).$$

Proof. Since g is continuously differentiable, for every $z \in [x, \widehat{x}]$, we have $g(z_{[R_1:x]}) = g(x) + \int_0^1 D_{x_1} g(t z_{[R_1:x]} + (1-t)x)(z_1 - x_1)dt$. Thus, for every $x \in [x, \widehat{x}]$, $g(z_{[R_1:x]}) \leq g(x) + \overline{J}_{[x,\widehat{x}_{[R_1:x]}]}(z_1 - x_1) \leq g(x) + [\overline{J}_{[x,\widehat{x}_{[R_1:x]}]}]^+(\widehat{x}_1 - x_1)$. where the first inequality holds because $z_1 - x_1 \geq \mathbb{0}_n$ and the second inequality holds because $z_1 - x_1 \leq \widehat{x}_1 - x_1$ and $z_1 - x_1 \geq \mathbb{0}_n$. By successively applying the above procedure on $g(z_{[R_1:x]})$, one can show that $g(z) \leq \overline{\mathsf{G}}^{\mathrm{jac-m}}(x,\widehat{x})$. Similarly, one can show that $g(z) \geq \underline{\mathsf{G}}^{\mathrm{jac-m}}(x,\widehat{x})$ and thus $\mathsf{G}^{\mathrm{jac-m}}$ is an inclusion function for g. Finally, one can show that $\underline{J}_{[x,\widehat{x}]} \leq \underline{M}$ and $\overline{M} \leq \overline{J}_{[x,\widehat{x}]}$. This implies that $-[\underline{M}]^- x + [\underline{M}]^- \widehat{x} \leq -[\underline{J}_{[x,\widehat{x}]}]^- x + [\overline{J}_{[x,\widehat{x}]}]^- \widehat{x}$. As a result $\underline{\mathsf{G}}^{\mathrm{jac}}(x,\widehat{x}) \leq \underline{\mathsf{G}}^{\mathrm{jac-m}}(x,\widehat{x})$. Similarly, one can show that $\overline{\mathsf{G}}^{\mathrm{jac-m}}(x,\widehat{x}) \leq \overline{\mathsf{G}}^{\mathrm{jac}}(x,\widehat{x})$. □

Example 1 (Cornered inclusion functions)**.** Consider the function $f : \mathbb{R}^2 \to \mathbb{R}^2$ defined by $f(x_1, x_2) = \begin{bmatrix} (x_1 + x_2)^2 \\ x_1 + x_2 + 2x_1 x_2 \end{bmatrix}$. We compute the natural inclusion function, the (mixed) Jacobian-based cornered inclusion function, and the (mixed) Jacobian-based centered inclusion function for f on the interval $[-0.1, 0.1] \times [-0.1, 0.1]$. The results are shown in Table (I), for runtimes averaged over 10 000 runs. We observe that the output interval estimated by the mixed Jacobian-based centered (cornered) is tighter than the interval obtained from the Jacobian-based centered (cornered) inclusion function. This observation is consistent with the comparison between mixed and non-mixed Jacobian-based cornered inclusion functions in Proposition 2. However, there is generally no ranking between other methods, as each can perform better in different scenarios.

Method	Output Interval	Runtime (s)
Natural	$[0, 0.04] \times [-0.22, 0.22]$	1.93×10^{-6}
Centered	$[-0.08, 0.08] \times [-0.24, 0.24]$	1.64×10^{-5}
Mixed Centered	$[-0.06, 0.06] \times [-0.22, 0.22]$	8.38×10^{-5}
Cornered	$[-0.12, 0.16] \times [-0.18, 0.22]$	6.24×10^{-5}
Mixed Cornered	$[-0.08, 0.08] \times [-0.18, 0.22]$	2.28×10^{-4}

TABLE I
COMPARISON OF INCLUSION FUNCTION METHODS.

Decomposition-based inclusion functions: Another approach for constructing inclusion functions is by decomposing the function into the sum of increasing and decreasing parts [15], [43]. This separation is often realized using a decomposition function.

Definition 2 (Decomposition function)**.** Given a function $g : \mathbb{R}^n \to \mathbb{R}^m$, a *decomposition function* for g is a map $d : \mathcal{T}^{2n} \to \mathbb{R}^m$ that satisfies

(a) $g(x) = d(x, x)$, for every $x \in \mathbb{R}^n$;
(b) $d(x, y) \leq d(\widehat{x}, y)$, for every $y \in \mathbb{R}^n$ and every $x \leq \widehat{x}$;
(c) $d(x, \widehat{y}) \leq d(x, y)$, for every $x \in \mathbb{R}^n$ and every $y \leq \widehat{y}$.

One can show that every continuous function has at least one decomposition function. Given a map $g : \mathbb{R}^n \to \mathbb{R}^m$, we define the tight decomposition function for g as the mapping $d^{\mathrm{tight}} : \mathcal{T}^{2n} \to \mathbb{R}^{2m}$ given by

$$d_i^{\mathrm{tight}}(x,\widehat{x}) = \begin{cases} \min_{z \in [x,\widehat{x}]} g_i(z) & x \leq \widehat{x} \\ \max_{z \in [\widehat{x},x]} g_i(z) & \widehat{x} \leq x \end{cases} \tag{8}$$

for every $i \in \{1, \ldots, n\}$. In general, the decomposition function for function g is not unique. The next theorem, whose proof is straightforward, shows how to construct inclusion functions from decomposition functions.

Proposition 3 (Decomposition-based inclusion functions)**.** *Consider a map $g : \mathbb{R}^n \to \mathbb{R}^m$ with a function $d : \mathcal{T}^{2n} \to \mathbb{R}^m$. Then, the map $\mathsf{G}^d : \mathcal{T}^{2n} \to \mathbb{R}^{2m}$ defined by*

$$\mathsf{G}^d(x,\widehat{x}) = \begin{bmatrix} d(x,\widehat{x}) \\ d(\widehat{x},x) \end{bmatrix}, \quad \textit{for all } x \leq \widehat{x},$$

is a thin monotone inclusion function for g if and only if d is a decomposition function for g. Moreover, G^d is the minimal inclusion function (7) if and only if d is the tight decomposition function (8).

Remark 2 (Comparison with the literature)**.** For a vector field $f : \mathbb{R}^n \to \mathbb{R}^n$, the decomposition function has already been studied in mixed monotone theory (cf. [44], [16, Definition 2], [45, Definition 4], and [46]). Definition 2 extends this classical notion to arbitrary functions.

One can combine two inclusion functions to obtain another inclusion function with tighter estimates for the output of the original function.

Proposition 4 (Intersection of inclusion functions)**.** *Given a map $g : \mathbb{R}^n \to \mathbb{R}^m$ with two inclusion functions $\mathsf{G}_1 = \begin{bmatrix} \underline{\mathsf{G}}_1 \\ \overline{\mathsf{G}}_1 \end{bmatrix} : \mathcal{T}^{2n}_{\geq 0} \to \mathcal{T}^{2m}_{\geq 0}$ and $\mathsf{G}_2 = \begin{bmatrix} \underline{\mathsf{G}}_2 \\ \overline{\mathsf{G}}_2 \end{bmatrix} : \mathcal{T}^{2n}_{\geq 0} \to \mathcal{T}^{2m}_{\geq 0}$. Define the mapping $\mathsf{G}_1 \wedge \mathsf{G}_2 = \begin{bmatrix} \underline{\mathsf{G}}_1 \wedge \underline{\mathsf{G}}_2 \\ \overline{\mathsf{G}}_1 \wedge \overline{\mathsf{G}}_2 \end{bmatrix} : \mathcal{T}^{2n}_{\geq 0} \to \mathcal{T}^{2m}_{\geq 0}$ by*

$$[\underline{\mathsf{G}}_1 \wedge \underline{\mathsf{G}}_2](x, \widehat{x}) = \max\{\underline{\mathsf{G}}_1(x, \widehat{x}), \underline{\mathsf{G}}_2(x, \widehat{x})\},$$
$$[\overline{\mathsf{G}}_1 \wedge \overline{\mathsf{G}}_2](x, \widehat{x}) = \min\{\overline{\mathsf{G}}_1(x, \widehat{x}), \overline{\mathsf{G}}_2(x, \widehat{x})\}.$$

Then $\mathsf{G}_1 \wedge \mathsf{G}_2$ is an inclusion function for g and it provides over-approximations of the output of g which are tighter than both G_1 and G_2.

Proof. Note that both G_1 and G_2 are inclusion functions for g. Thus, for every $x \leq z \leq \widehat{x}$, we have $\mathsf{G}_1(x, \widehat{x}) \leq g(z)$ and $\mathsf{G}_2(x, \widehat{x}) \leq g(z)$. This implies that $\max\{\underline{\mathsf{G}}_1(x, \widehat{x}), \underline{\mathsf{G}}_2(x, \widehat{x})\} = [\underline{\mathsf{G}}_1 \wedge \underline{\mathsf{G}}_2](x, \widehat{x}) \leq g(z)$. Similarly, one can show that $g(z) \leq [\overline{\mathsf{G}}_1 \wedge \overline{\mathsf{G}}_2](x, \widehat{x})$. □

C. Convergence of inclusion functions

We now define the convergence rate of an inclusion function which can be used to capture the accuracy of the interval estimations.

Definition 3 (Convergence rate of inclusion functions)**.** Consider the mapping $g : \mathbb{R}^n \to \mathbb{R}^m$ with an inclusion function $\mathsf{G} = \begin{bmatrix} \underline{\mathsf{G}} \\ \overline{\mathsf{G}} \end{bmatrix} : \mathcal{T}^{2n}_{\geq 0} \to \mathcal{T}^{2m}_{\geq 0}$. The rate of convergence of G is the largest $\alpha \in \mathbb{R}_{\geq 0} \cup \{\infty\}$ such that

$$\|\overline{\mathsf{G}}(x, \widehat{x}) - \underline{\mathsf{G}}(x, \widehat{x})\|_\infty - \max_{z,y \in [x,\widehat{x}]} \|g(z) - g(y)\|_\infty \leq \beta \|\widehat{x} - x\|^\alpha_\infty \quad (9)$$

for some $\beta \in \mathbb{R}_{\geq 0}$ and for every $x \leq \widehat{x}$.

Roughly speaking, the rate of convergence of an inclusion function G shows how accurately it estimates the range of the original function g. Indeed, for the minimal inclusion function, the interval $[\underline{\mathsf{G}}(x, \widehat{x}), \overline{\mathsf{G}}(x, \widehat{x})]$ is the tightest interval containing $g([x, \widehat{x}])$. Thus the left hand side of inequality (9) is always zero and the rate of convergence of the minimal inclusion function is $\alpha = \infty$. It is shown that, in general, the convergence rate of the natural inclusion functions is $\alpha = 1$ [40, Lemma 4.1] and the convergence rate of the Jacobian-based inclusion functions are $\alpha = 2$ [10, §2.4.5].

D. Inclusion functions for Neural Networks

Given a neural network N of the form (4) and any interval $[y, \widehat{y}] \subseteq \mathbb{R}^n$, as described in the sequel, our framework proposes using existing neural network verification algorithms to construct a $[y, \widehat{y}]$-localized inclusion function of the form $\mathsf{N}_{[y,\widehat{y}]} = \begin{bmatrix} \underline{\mathsf{N}}_{[y,\widehat{y}]} \\ \overline{\mathsf{N}}_{[y,\widehat{y}]} \end{bmatrix} : \mathcal{T}^{2n}_{\geq 0} \to \mathcal{T}^{2p}_{\geq 0}$ for N satisfying

$$\underline{\mathsf{N}}_{[y,\widehat{y}]}(x, \widehat{x}) \leq N(x) \leq \overline{\mathsf{N}}_{[y,\widehat{y}]}(x, \widehat{x}), \quad (10)$$

for any $x \in [x, \widehat{x}] \subseteq [y, \widehat{y}]$. A large number of the existing neural network verification algorithms can provide bounds of the form (10) for the output of the neural networks, including CROWN [25], LipSDP [24], and IBP [23]. In particular, some neural network verification algorithms can provide *affine* $[y, \widehat{y}]$-localized inclusion functions for N. Examples of these neural network verification algorithms include CROWN and its variants [25]. Given an interval $[y, \widehat{y}]$, these algorithms provide a tuple $(\underline{C}, \overline{C}, \underline{d}, \overline{d})$ defining affine upper and lower bounds for the output of the neural network

$$\underline{C}(y, \widehat{y})x + \underline{d}(y, \widehat{y}) \leq N(x) \leq \overline{C}(y, \widehat{y})x + \overline{d}(y, \widehat{y}), \quad (11)$$

for every $x \in [y, \widehat{y}]$. Using these linear bounds, we can construct the affine $[y, \widehat{y}]$-localized inclusion function for N:

$$\begin{aligned} \underline{\mathsf{N}}_{[y,\widehat{y}]}(x, \widehat{x}) &= [\underline{C}(y, \widehat{y})]^+ x + [\underline{C}(y, \widehat{y})]^- \widehat{x} + \underline{d}(y, \widehat{y}), \\ \overline{\mathsf{N}}_{[y,\widehat{y}]}(x, \widehat{x}) &= [\overline{C}(y, \widehat{y})]^+ \widehat{x} + [\overline{C}(y, \widehat{y})]^- x + \overline{d}(y, \widehat{y}) \end{aligned} \quad (12)$$

for any $[x, \widehat{x}] \subseteq [y, \widehat{y}]$.

Remark 3.

(i) *(Computational complexity):* the computational complexity of finding the $[y, \widehat{y}]$-localized inclusion function $\mathsf{N}_{[y,\widehat{y}]} = \begin{bmatrix} \underline{\mathsf{N}}_{[y,\widehat{y}]} \\ \overline{\mathsf{N}}_{[y,\widehat{y}]} \end{bmatrix}$ depends on the neural network verification algorithm. For instance, for a neural network with k-layer and M neurons per layer, the computational complexity of CROWN [25] in finding the bounds of the form (10) is $\mathcal{O}(k^2 M^3)$.

(ii) *(Inclusion function):* Since the $[y, \widehat{y}]$-localized inclusion function $\mathsf{N}_{[y,\widehat{y}]} = \begin{bmatrix} \underline{\mathsf{N}}_{[y,\widehat{y}]} \\ \overline{\mathsf{N}}_{[y,\widehat{y}]} \end{bmatrix}$ can be obtained for every $[y, \widehat{y}]$, we can define an inclusion function $\mathsf{N} = \begin{bmatrix} \underline{\mathsf{N}} \\ \overline{\mathsf{N}} \end{bmatrix}$ for the neural network N by $\mathsf{N}(x, \widehat{x}) = \mathsf{N}_{[x,\widehat{x}]}(x, \widehat{x})$, for all $x \leq \widehat{x}$.

V. Interval Reachability of Dynamical Systems

In this section, we study reachability of the dynamical system (1) using interval analysis. Our key idea is to embed the original dynamical system (1) into an *embedding system* on $\mathbb{R}^{2n}$ and use trajectories of the embedding system to study the propagation of the state bounds with time. Our main goal is to use the notion of inclusion function (cf. Definition 1) to design embedding systems for dynamical systems. Consider the dynamical system (1) with an inclusion function $\mathsf{F} = \begin{bmatrix} \underline{\mathsf{F}} \\ \overline{\mathsf{F}} \end{bmatrix} : \mathcal{T}^{2n}_{\geq 0} \times \mathcal{T}^{2q}_{\geq 0} \to \mathcal{T}^{2n}_{\geq 0}$ for f. We define the associated *embedding system* on $\mathbb{R}^{2n}$:

$$\begin{aligned} \dot{x}_i &= \underline{\mathsf{F}}_i(x, \widehat{x}_{[i:x]}, w, \widehat{w}), \\ \dot{\widehat{x}}_i &= \overline{\mathsf{F}}_i(x_{[i:\widehat{x}]}, \widehat{x}, w, \widehat{w}), \end{aligned} \quad (13)$$

for every $i \in \{1,\dots,n\}$. For instance, using the minimal inclusion function $\mathsf{F}^{\min} = \begin{bmatrix}\underline{\mathsf{F}}^{\min}\\ \overline{\mathsf{F}}^{\min}\end{bmatrix}$ for f, the associated embedding system is given by:

$$\begin{aligned}\dot{x}_i &= \min_{\substack{z\in[x,\widehat{x}],\ z_i=x_i,\\ u\in[w,\widehat{w}]}} f_i(z,u),\\ \dot{\widehat{x}}_i &= \max_{\substack{z\in[x,\widehat{x}],\ z_i=\widehat{x}_i,\\ u\in[w,\widehat{w}]}} f_i(z,u),\end{aligned} \tag{14}$$

for every $i \in \{1,\dots,n\}$. Given the disturbance set $\mathcal{W} = [\underline{w},\overline{w}]$ and the initial set $\mathcal{X}_0 = [\underline{x}_0,\overline{x}_0]$, one can use a single trajectory of the embedding system (13) to obtain over-approximations of reachable sets of the system (1). Moreover, the embedding system obtained from the minimal inclusion function (i.e., the dynamical system (14)) provides the best reachable set over-approximations at any time $t \geq 0$, compared to any other embedding system constructed using an inclusion function for vector field f in the dynamical system (1).

Proposition 5 (Reachability using embedding systems)**.** *Consider the dynamical system (1) with the disturbance set $[\underline{w},\overline{w}]$ and the initial set $\mathcal{X}_0 = [\underline{x}_0,\overline{x}_0]$. Suppose that $\mathsf{F} = \begin{bmatrix}\underline{\mathsf{F}}\\ \overline{\mathsf{F}}\end{bmatrix}$ is an inclusion function for f, and $t \mapsto \begin{bmatrix}\underline{x}(t)\\ \overline{x}(t)\end{bmatrix}$ and $t \mapsto \begin{bmatrix}\underline{x}^{\min}(t)\\ \overline{x}^{\min}(t)\end{bmatrix}$ are the trajectories of embedding systems (13) and (14), starting from $\begin{bmatrix}\underline{x}_0\\ \overline{x}_0\end{bmatrix}$ with fixed disturbance $\begin{bmatrix}\underline{w}\\ \widehat{w}\end{bmatrix} = \begin{bmatrix}\underline{w}\\ \overline{w}\end{bmatrix}$. Then, for every $t \in \mathbb{R}_{\geq 0}$, we have*

$$\mathcal{R}_f(t,[\underline{x}_0,\overline{x}_0],[\underline{w},\overline{w}]) \subseteq [\underline{x}^{\min}(t),\overline{x}^{\min}(t)] \subseteq [\underline{x}(t),\overline{x}(t)].$$

Proof. We first show that the dynamical system (14) is a monotone dynamical system with respect to the southeast partial order $\leq_{\mathrm{SE}}$. Let $\begin{bmatrix}x\\ \widehat{x}\end{bmatrix} \leq_{\mathrm{SE}} \begin{bmatrix}y\\ \widehat{y}\end{bmatrix}$ be such that $x_i = y_i$ and let $\begin{bmatrix}w\\ \widehat{w}\end{bmatrix} \leq_{\mathrm{SE}} \begin{bmatrix}v\\ \widehat{v}\end{bmatrix}$. Then, we have $[y,\widehat{y}] \subseteq [x,\widehat{x}]$ and $[v,\widehat{v}] \subseteq [w,\widehat{w}]$, and we can compute

$$\begin{aligned}\underline{\mathsf{F}}_i^{\min}(x,\widehat{x},w,\widehat{w}) &= \min_{\substack{z\in[x,\widehat{x}],\ z_i=x_i,\\ u\in[w,\widehat{w}]}} f_i(z,u)\\ &\leq \min_{\substack{z\in[y,\widehat{y}],\ z_i=y_i,\\ u\in[v,\widehat{v}]}} f_i(z,u) = \underline{\mathsf{F}}_i^{\min}(y,\widehat{y},v,\widehat{v}),\end{aligned}$$

for every $i \in \{1,\dots,n\}$. Similarly, let $\begin{bmatrix}x\\ \widehat{x}\end{bmatrix} \leq_{\mathrm{SE}} \begin{bmatrix}y\\ \widehat{y}\end{bmatrix}$ be such that $\widehat{x}_i = \widehat{y}_i$ and let $\begin{bmatrix}w\\ \widehat{w}\end{bmatrix} \leq_{\mathrm{SE}} \begin{bmatrix}v\\ \widehat{v}\end{bmatrix}$. Then, we have $[y,\widehat{y}] \subseteq [x,\widehat{x}]$ and $[v,\widehat{v}] \subseteq [w,\widehat{w}]$, and similar reasoning implies $\overline{\mathsf{F}}_i^{\min}(y,\widehat{y},v,\widehat{v}) \leq \overline{\mathsf{F}}_i^{\min}(x,\widehat{x},w,\widehat{w})$, for every $i \in \{1,\dots,n\}$. As a result, the embedding system (14) is monotone with respect to $\leq_{\mathrm{SE}}$. Let $x_0 \in [\underline{x}_0,\overline{x}_0]$ and $u \in [\underline{w},\overline{w}]$ and $t \mapsto x_u(t)$ be the trajectory of the system (1) starting from x_0 with disturbance u. Note that $t \mapsto \begin{bmatrix}x_u(t)\\ x_u(t)\end{bmatrix}$ is a solution of the embedding system (14) with the initial condition $\begin{bmatrix}x_0\\ x_0\end{bmatrix}$ and the disturbance $\begin{bmatrix}u\\ u\end{bmatrix}$. Moreover, we know that $\begin{bmatrix}\underline{x}_0\\ \overline{x}_0\end{bmatrix} \leq_{\mathrm{SE}} \begin{bmatrix}x_0\\ x_0\end{bmatrix}$ and $\begin{bmatrix}\underline{w}\\ \overline{w}\end{bmatrix} \leq_{\mathrm{SE}} \begin{bmatrix}u\\ u\end{bmatrix}$. Therefore, by monotonicity of the embedding system (14) with respect to $\leq_{\mathrm{SE}}$, we get $\begin{bmatrix}\underline{x}^{\min}(t)\\ \overline{x}^{\min}(t)\end{bmatrix} \leq_{\mathrm{SE}} \begin{bmatrix}x_u(t)\\ x_u(t)\end{bmatrix}$, for every $t \in \mathbb{R}_{\geq 0}$. This implies that $\mathcal{R}_f(t,[\underline{x}_0,\overline{x}_0],[\underline{w},\overline{w}]) \subseteq [\underline{x}^{\min}(t),\overline{x}^{\min}(t)]$, for every $t \in \mathbb{R}_{\geq 0}$. On the other hand, $\mathsf{F}^{\min}$ is the minimal inclusion function for f and therefore, for every $x \leq \widehat{x}$ and every $w \leq \widehat{w}$, we have $\begin{bmatrix}\underline{\mathsf{F}}(x,\widehat{x},w,\widehat{w})\\ \overline{\mathsf{F}}(x,\widehat{x},w,\widehat{w})\end{bmatrix} \leq_{\mathrm{SE}} \begin{bmatrix}\underline{\mathsf{F}}^{\min}(x,\widehat{x},w,\widehat{w})\\ \overline{\mathsf{F}}^{\min}(x,\widehat{x},w,\widehat{w})\end{bmatrix}$. Thus, for every $i \in \{1,\dots,n\}$, every $x \leq \widehat{x}$, and every $w \leq \widehat{w}$,

$$\begin{aligned}\underline{\mathsf{F}}_i(x,\widehat{x}_{[i:x]},w,\widehat{w}) &\leq \min_{\substack{z\in[x,\widehat{x}],\ z_i=x_i,\\ \xi\in[w,\widehat{w}]}} f_i(z,\xi),\\ \overline{\mathsf{F}}_i(x_{[i:\widehat{x}]},\widehat{x},w,\widehat{w}) &\geq \max_{\substack{z\in[x,\widehat{x}],\ z_i=\widehat{x}_i,\\ \xi\in[w,\widehat{w}]}} f_i(z,\xi).\end{aligned}$$

Now, we can use the classical monotone comparison Lemma [47, Theorem 3.8.1] to obtain $\begin{bmatrix}\underline{x}(t)\\ \overline{x}(t)\end{bmatrix} \leq_{\mathrm{SE}} \begin{bmatrix}\underline{x}^{\min}(t)\\ \overline{x}^{\min}(t)\end{bmatrix}$, for every $t \in \mathbb{R}_{\geq 0}$. This implies that $[\underline{x}^{\min}(t),\overline{x}^{\min}(t)] \subseteq [\underline{x}(t),\overline{x}(t)]$, for every $t \in \mathbb{R}_{\geq 0}$, and completes the proof. □

Remark 4.

(i) It is straightforward to extend Proposition 5 to the case when the mapping F is a $[y,\widehat{y}] \times \mathbb{R}^q$-localized inclusion function for f. In this setting, the results of Proposition 5 hold as long as we have $[\underline{x}(t),\overline{x}(t)] \subseteq [y,\widehat{y}]$.

(ii) Proposition 5 can be considered as a generalization of [48, Proposition 6] and [18, Proposition 1]. Indeed, in the special case that F is a decomposition-based inclusion function constructed from the decomposition function d, one can recover the embedding system

$$\begin{aligned}\dot{x}_i &= d(x,\widehat{x}_{[i:x]},w,\widehat{w}),\\ \dot{\widehat{x}}_i &= d(\widehat{x},x_{[i:\widehat{x}]},\widehat{w},w)\end{aligned}$$

for every $i \in \{1,\dots,n\}$. This embedding system is identical to [48, Equation (7)] and [18, Equation (3)]. We highlight that this extension is crucial for our framework as the inclusion functions we obtain for learning-based components are neither thin nor decomposition-based.

(iii) Proposition 5 can alternatively be proved using [12, Theorem 2]. However, our proof of Proposition 5 is different in that it uses monotone system theory [49], [50] and classical monotone comparison Lemma [47, Theorem 3.8.1]. Indeed, compared to [12, Theorem 2], our proof techniques allow to compare accuracy of different over-approximations of reachable sets.

(iv) For the dynamical system (1) with locally Lipschitz vector field f, the minimal inclusion function for f as defined in equation (7) exists. Therefore, the reachability framework from Proposition 5 is applicable to locally Lipschitz nonlinear systems, regardless of monotonicity of the original dynamics.

We now study the accuracy of the over-approximations provided in Proposition 5 using the notion of convergence rate of embedding systems.

Definition 4 (Convergence rate of embedding systems)**.** Consider the dynamical system (1) with the embedding system (13). Then the convergence rate of the embedding system (13) is the largest value of $\alpha \in \mathbb{R}_{\geq 0} \cup \{\infty\}$ such that

$$\begin{aligned}\|\overline{x}(t) - \underline{x}(t)\|_\infty - \max_{y,z\in[\underline{x}_0,\overline{x}_0]} \|\phi_f(t,y) - \phi_f(t,z)\|_\infty\\ \leq M(t)\|\overline{x}_0 - \underline{x}_0\|_\infty^\alpha\end{aligned}$$

for some $M : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ and for $t \mapsto \begin{bmatrix}\underline{x}(t)\\ \overline{x}(t)\end{bmatrix}$ being the trajectory of the embedding system (13) starting from any $\begin{bmatrix}\underline{x}_0\\ \overline{x}_0\end{bmatrix}$ with fixed disturbance $\begin{bmatrix}\underline{w}\\ \widehat{w}\end{bmatrix} = \mathbb{0}_{2n}$.

Definition 4 raises an important question: what is the connection between the convergence rate of the embedding system (13) and the convergence rate of its associated inclusion function F (as in Definition 3)? In the special case when f is a discrete-time monotone vector field, one can choose the inclusion function $\mathsf{F}(\underline{x}, \overline{x}) = \begin{bmatrix} f(\underline{x}) \\ f(\overline{x}) \end{bmatrix}$ for f. In this case, it can be shown that the convergence rate of both the embedding system and the inclusion function are the same and equal to $\alpha = \infty$. Given a Lipschitz and monotone inclusion function F for f, one can show that the convergence rate of the embedding system (13) satisfies $\alpha \geq 1$ [40, Theorem 4.1]. However, in general, the inclusion function F does not reveal more information about the convergence rate of the embedding system (13). Even when the convergent rate of the inclusion function F is $\alpha = \infty$, i.e., the embedding system is given by equations (14), one might get a convergence rate of $\alpha = 1$ for the embedding system. Thus, even the reachable set estimates from the embedding system associated to the minimal inclusion function can incur over-approximation errors proportional to the size of uncertainty set.

VI. Interval Reachability of Learning-Based Systems

In this section, we develop a computationally efficient framework for studying reachability of neural network controlled systems. The key idea of our approach is to employ the framework of Section V and to capture the interactions between the open-loop system and the neural network controller using a suitable closed-loop embedding system. However, finding a closed-loop embedding system (or, equivalently, computing an inclusion function for the closed-loop vector field) using the approaches in Section IV is often not tractable, as the neural networks can have a large number of parameters. Instead, we propose to use a compositional approach based on an inclusion function for the open-loop dynamics and an inclusion function for the neural network controller. We develop two different approaches for designing the closed-loop inclusion function from the open-loop vector field f and the localized inclusion function of the neural network.

First, we propose an *interconnection-based* approach where the key idea is to consider the closed-loop embedding system as the feedback interconnection of the open-loop embedding system and the neural network inclusion function. This method, as elaborated below, can capture some of the stabilizing effects of the neural network controller by approximating the input-output behavior of the neural network separately on the faces of a hyperrectangle. Thus, we consider this approach a semi-input-output approach. The advantages of the interconnection-based approach are its computational speed and that it is amenable to any open-loop inclusion function.

Second, we propose an *interaction-based* approach where the key idea is to use the Jacobian-based cornered inclusion function (developed in Proposition 1) to more fully capture the interaction between the neural network controller and the system. The advantages of this approach are reduced conservatism compared to the interconnection-based inclusion function while retaining much of the computational efficiencies of interval reachability analysis.

Our methods are applicable to general nonlinear systems with neural network controllers. Nonetheless, the essence of our approaches, and their advantages over a naive input-output approach, are evident even in the simple setting of a linear system with a linear feedback controller, as shown in the following illustrative example.

Example 2 (Invariant intervals)**.** Consider the control system $\dot{x} = Ax + Bu$ where $x \in \mathbb{R}^2$ and $A = \left[\begin{smallmatrix} -2 & 1 \\ 1 & -2 \end{smallmatrix}\right]$ and $B = \left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right]$. Note that A is Hurwitz so the origin is the globally asymptotically stable equilibrium point of the open-loop system when $u \equiv 0$. Consider a state feedback controller $u = \pi(x) = Kx = k_1 x_1 + k_2 x_2$ with $k_1 = k_2 = -3$ designed to, *e.g.*, achieve faster convergence to the origin. As a special case of reachability analysis, our goal is to characterize forward invariant intervals of the closed-loop system around the origin. We compare three different approaches for finding invariant intervals.

Naive input-output approach: this approach decouples the system from the controller and only uses knowledge of the output range of the controller. In particular, this approach seeks an interval $S = [-\xi, \xi]$, $\xi \in \mathbb{R}^2_{\geq 0}$ such that S is robustly forward invariant for the open loop system under any $u \in \pi(S)$. However, such a robustly forward invariant set does not exist. By contradiction, suppose $S = [-\xi, \xi]$ is forward invariant and consider the point $x = \left[\begin{smallmatrix} 0 \\ \xi_2 \end{smallmatrix}\right]$ on the boundary of this set and pick $u = \pi(-x) = 3\xi_2 \in \pi(S)$, for which $\dot{x}_2 = -2\xi_2 + u = \xi_2 > 0$ and the resulting trajectory immediately leaves S. Despite the stability of the open-loop system and the stabilizing effect of the controller, this approach fails to capture the stability of the closed-loop system.

Interconnection-based approach: this approach also decouples the system from the controller and only assumes knowledge of the output of the controller to characterize forward invariant intervals $S = [-\xi, \xi]$, for some $\xi \in \mathbb{R}^2_{\geq 0}$. However, it considers the interconnection between the system and the controller on the edges of the interval S. Let $S_1^-, S_1^+, S_2^-, S_2^+$ denote the edges of S defined by $S_i^{\pm} = \{x \in S \mid x_i = \pm\xi_i\}$ for $i \in \{1, 2\}$. Then a sufficient condition for forward invariance of S is that $\dot{x}_i \geq 0$ for every $x \in S_i^-$, and $\dot{x}_i \leq 0$ for every $x \in S_i^+$. Note that, if $\xi_1/\xi_2 \geq \frac{1}{2}$, then we have $\dot{x}_1 \leq 0$ on S_1^+ and we have $\dot{x}_1 \geq 0$ on S_1^-. On S_2^+, we have $\dot{x}_2 \in [-\xi_1, \xi_1] - 2\xi_2 + u$ and $u \in -3[-\xi_1, \xi_1] - 3\xi_2$. This implies that $\dot{x}_2 \in 4[-\xi_1, \xi_1] - 5\xi_2$, and, therefore, we need $\xi_1/\xi_2 \leq \frac{5}{4}$ for $\dot{x}_2 \leq 0$. A similar condition is required to ensure $\dot{x}_2 \geq 0$ on S_2^-. This implies that, using this interconnection-based approach, we certify $S = [-\xi, \xi]$ is forward invariant for any $\xi \in \mathbb{R}^2_{>0}$ satisfying $\frac{1}{2} \leq \xi_1/\xi_2 \leq \frac{5}{4}$.

Interaction-based approach: this approach uses the knowledge of the controller functional dependency $\pi(x) = Kx$ and the open-loop dynamics to obtain the closed-loop system $\dot{x} = (A + BK)x = \left[\begin{smallmatrix} -2 & 1 \\ -2 & -5 \end{smallmatrix}\right] x$. One can show that the interval $S = [-\xi, \xi]$ is forward invariant for this closed-loop system for any $\xi \in \mathbb{R}^2_{\geq 0}$ satisfying $\frac{1}{2} \leq \xi_1/\xi_2 \leq \frac{5}{2}$.

Now, we develop a mathematical framework for reachability analysis of the learning-based closed-loop system (5). Given an open-loop system of the form (1), we use approaches developed in Section IV-B to construct an inclusion function

$\mathsf{F}^{\mathrm{o}} = \begin{bmatrix} \underline{\mathsf{F}}^{\mathrm{o}} \\ \overline{\mathsf{F}}^{\mathrm{o}} \end{bmatrix} : \mathcal{T}^{2n}_{\geq 0} \times \mathcal{T}^{2p}_{\geq 0} \times \mathcal{T}^{2q}_{\geq 0} \to \mathcal{T}^{2n}_{\geq 0}$ for the open-loop vector field f. We also assume that we have access to a neural network inclusion function N as developed in Section IV-D. We construct two classes of inclusion functions for the closed-loop vector field f^c in (5).

A. Interconnection-based approach

In the first approach, which we refer to as interconnection-based approach, we obtain the closed-loop inclusion function from a feedback interconnection of the open-loop embedding system and the neural network inclusion function.

Theorem 2 (Closed-loop interconnection-based inclusion function)**.** *Consider the open-loop system (3) with an inclusion function F^{o} and a neural network controller of the form (4) with a $[y,\widehat{y}]$-localized inclusion function $\mathsf{N}_{[y,\widehat{y}]} = \begin{bmatrix} \underline{\mathsf{N}}_{[y,\widehat{y}]} \\ \overline{\mathsf{N}}_{[y,\widehat{y}]} \end{bmatrix}$. Then, the map $\mathsf{F}^{\mathrm{con}}_{[y,\widehat{y}]} : \mathcal{T}^{2n}_{\geq 0} \times \mathcal{T}^{2q}_{\geq 0} \to \mathcal{T}^{2n}_{\geq 0}$ defined by*

$$\begin{aligned} \underline{\mathsf{F}}^{\mathrm{con}}_{[y,\widehat{y}]}(x,\widehat{x},w,\widehat{w}) &= \underline{\mathsf{F}}^{\mathrm{o}}(x,\widehat{x},\xi,\widehat{\xi},w,\widehat{w}) \\ \overline{\mathsf{F}}^{\mathrm{con}}_{[y,\widehat{y}]}(x,\widehat{x},w,\widehat{w}) &= \overline{\mathsf{F}}^{\mathrm{o}}(x,\widehat{x},\eta,\widehat{\eta},w,\widehat{w}) \end{aligned} \tag{15}$$

is a $[y,\widehat{y}]\times\mathbb{R}^q$-localized inclusion function for the closed-loop vector field f^c, where

$$\begin{aligned} \xi &= \underline{\mathsf{N}}_{[y,\widehat{y}]}(x,\widehat{x}), \quad \widehat{\xi} = \overline{\mathsf{N}}_{[y,\widehat{y}]}(x,\widehat{x}), \\ \eta &= \overline{\mathsf{N}}_{[y,\widehat{y}]}(x,\widehat{x}), \quad \widehat{\eta} = \underline{\mathsf{N}}_{[y,\widehat{y}]}(x,\widehat{x}). \end{aligned}$$

Proof. Let $z \in [x,\widehat{x}] \subseteq [y,\widehat{y}]$ and $v \in [w,\widehat{w}]$. Since $\mathsf{N}_{[y,\widehat{y}]} = \begin{bmatrix} \underline{\mathsf{N}}_{[y,\widehat{y}]} \\ \overline{\mathsf{N}}_{[y,\widehat{y}]} \end{bmatrix}$ is a $[y,\widehat{y}]$-localized inclusion function for the neural network controller N, we have $\underline{\mathsf{N}}_{[y,\widehat{y}]}(x,\widehat{x}) \leq N(z) \leq \overline{\mathsf{N}}_{[y,\widehat{y}]}(x,\widehat{x})$. Therefore, we can compute $\underline{\mathsf{F}}^{\mathrm{con}}_{[y,\widehat{y}]}(x,\widehat{x},w,\widehat{w}) = \underline{\mathsf{F}}^{\mathrm{o}}(x,\widehat{x},\xi,\widehat{\xi},w,\widehat{w}) \leq f(z,N(z),v) = f^c(z,v)$, where the second inequality holds because F^{o} is an inclusion function for the open-loop vector field f. Similarly, one can show that $\overline{\mathsf{F}}^{\mathrm{con}}_{[y,\widehat{y}]}(x,\widehat{x},w,\widehat{w}) \geq f^c(z,v)$. This implies that $\mathsf{F}^{\mathrm{con}}_{[y,\widehat{y}]}$ is a $[y,\widehat{y}]$-localized inclusion function for f^c. □

B. Interaction-based approach

The second approach, which we refer to as interaction-based approach, uses a Jacobian-based cornered inclusion function for the open-loop system and an affine neural network bound to capture the first order interaction between the neural network controller and the system.

Theorem 3 (Closed-loop interaction-based inclusion function)**.** *Consider the closed-loop system (5) and assume the open-loop vector field f is continuously differentiable and the neural network controller (4) satisfies affine bounds of the form (11) on the interval $[y,\widehat{y}]$. Assume that, for every $z \in [x,\widehat{x}] \subseteq [y,\widehat{y}]$, every $\xi \in [u,\widehat{u}]$, and every $\eta \in [w,\widehat{w}]$,*

$$\begin{aligned} D_z f(z,\xi,\eta) &\in [\underline{J}_{[x,\widehat{x}]}, \overline{J}_{[x,\widehat{x}]}], \\ D_\xi f(z,\xi,\eta) &\in [\underline{J}_{[u,\widehat{u}]}, \overline{J}_{[u,\widehat{u}]}], \\ D_\eta f(z,\xi,\eta) &\in [\underline{J}_{[w,\widehat{w}]}, \overline{J}_{[w,\widehat{w}]}]. \end{aligned} \tag{16}$$

Then, the map $\mathsf{F}^{\mathrm{act}}_{[y,\widehat{y}]} : \mathcal{T}^{2n}_{\geq 0} \times \mathcal{T}^{2q}_{\geq 0} \to \mathcal{T}^{2n}_{\geq 0}$ defined by

$$\mathsf{F}^{\mathrm{act}}_{[y,\widehat{y}]}(x,\widehat{x},w,\widehat{w}) = \begin{bmatrix} [\underline{H}]^+ & -\underline{J}_{[x,\widehat{x}]} & [\underline{H}]^- \\ [\overline{H}]^- & -\overline{J}_{[x,\widehat{x}]} & [\overline{H}]^+ \end{bmatrix} \begin{bmatrix} x \\ \widehat{x} \end{bmatrix} + L \begin{bmatrix} w \\ \widehat{w} \end{bmatrix} + Q, \tag{17}$$

is a $[y,\widehat{y}]\times\mathbb{R}^q$-localized inclusion function for the closed-loop vector field f^c, where

$$\begin{aligned} \underline{H} &= \underline{J}_{[x,\widehat{x}]} + [\underline{J}_{[u,\widehat{u}]}]^+ \underline{C}(y,\widehat{y}) + [\underline{J}_{[u,\widehat{u}]}]^- \overline{C}(y,\widehat{y}) \\ \overline{H} &= \overline{J}_{[x,\widehat{x}]} + [\overline{J}_{[u,\widehat{u}]}]^+ \overline{C}(y,\widehat{y}) + [\overline{J}_{[u,\widehat{u}]}]^- \underline{C}(y,\widehat{y}), \\ L &= \begin{bmatrix} -[\underline{J}_{[w,\widehat{w}]}]^- & [\underline{J}_{[w,\widehat{w}]}]^- \\ -[\overline{J}_{[w,\widehat{w}]}]^+ & [\overline{J}_{[w,\widehat{w}]}]^+ \end{bmatrix}, \\ Q &= \begin{bmatrix} -\underline{J}_{[u,\widehat{u}]}u + [\underline{J}_{[u,\widehat{u}]}]^+ \underline{d}(y,\widehat{y}) + [\underline{J}_{[u,\widehat{u}]}]^- \overline{d}(y,\widehat{y}) + f(x,u,w) \\ -\overline{J}_{[u,\widehat{u}]}u + [\overline{J}_{[u,\widehat{u}]}]^+ \overline{d}(y,\widehat{y}) + [\overline{J}_{[u,\widehat{u}]}]^- \underline{d}(y,\widehat{y}) + f(x,u,w) \end{bmatrix}. \end{aligned}$$

Proof. Let $z \in [x,\widehat{x}]$ and $\eta \in [w,\widehat{w}]$. Since f is continuously differentiable, the fundamental theorem of Calculus gives

$$\begin{aligned} f^c(z,\eta) &= f(z,N(z),\eta) = f(x,u,w) + M(z,u,w)(z-x) \\ &\quad + P(z,u,w)(N(z)-u) + R(z,\eta,w)(\eta - w), \end{aligned}$$

where $M(z,u,w) = \int_0^1 D_x f(\tau z + (1-\tau)x,u,w)d\tau$, $P(z,u,w) = \int_0^1 D_u f(z,\tau N(z) + (1-\tau)u,w)d\tau$, and $R(z,\eta,w) = \int_0^1 D_w f(z,N(z),\tau\eta + (1-\tau)w)d\tau$. Using the bounds (16), we get $M(z,u,w) \geq \underline{J}_{[x,\widehat{x}]}$, $P(z,u,w) \geq \underline{J}_{[u,\widehat{u}]}$, and $R(z,\eta,w) \geq \underline{J}_{[w,\widehat{w}]}$. Moreover, we have $N(z) - u \geq \mathbb{0}_p$. Therefore, using the affine bounds (11) for the neural network, $P(z,u,w)(N(z)-u) \geq [P(z,u,w)]^+(\underline{C}(y,\widehat{y})z + \underline{d}(y,\widehat{y}) - u) + [P(z,u,w)]^-(\overline{C}(y,\widehat{y})z + \overline{d}(y,\widehat{y}) - u)$. Using the fact that $z - x \geq \mathbb{0}_n$ and $\eta - w \geq \mathbb{0}_q$, we have

$$\begin{aligned} f^c(z,\eta) &\geq f(x,u,w) + M(z,u,w)(z-x) \\ &\quad + [P(z,u,w)]^+(\underline{C}(y,\widehat{y})z + \overline{d}(y,\widehat{y}) - u) \\ &\quad + [P(z,u,w)]^-(\overline{C}(y,\widehat{y})z + \overline{d}(y,\widehat{y}) - u) \\ &\quad + R(z,\eta,w)(\eta - w) \\ &\geq \underline{H}z - \underline{J}_{[x,\widehat{x}]}x + \underline{J}_{[w,\widehat{w}]}\eta - \underline{J}_{[w,\widehat{w}]}w \\ &\quad + [\underline{J}_{[w,\widehat{w}]}]^+ \underline{d}(y,\widehat{y}) + [\underline{J}_{[w,\widehat{w}]}]^- \overline{d}(y,\widehat{y}) + f(x,u,w), \end{aligned}$$

where the second inequality follows from the Jacobian bounds. Now, we note that $x \leq z \leq \widehat{x}$ and $w \leq \eta \leq \widehat{w}$ and therefore $\underline{H}z \geq [\underline{H}]^+ x + [\underline{H}]^+ \widehat{x}$ and $\underline{J}_{[w,\widehat{w}]}\eta \geq [\underline{J}_{[w,\widehat{w}]}]^+ w + [\underline{J}_{[w,\widehat{w}]}]^- \widehat{w}$. As a result, we get

$$\begin{aligned} f^c(z,\eta) &\geq [\underline{H}]^+ x + [\underline{H}]^- \widehat{x} - \underline{J}_{[x,\widehat{x}]}x - [\underline{J}_{[w,\widehat{w}]}]^- w \\ &\quad + [\underline{J}_{[w,\widehat{w}]}]^- \widehat{w} + [\underline{J}_{[w,\widehat{w}]}]^+ \underline{d}(y,\widehat{y}) + [\underline{J}_{[w,\widehat{w}]}]^- \overline{d}(y,\widehat{y}) \\ &\quad + f(x,u,w). \end{aligned}$$

This implies that $f^c(z,\eta) \geq \underline{\mathsf{F}}^{\mathrm{con}}_{[y,\widehat{y}]}(x,\widehat{x},w,\widehat{w})$. Similarly, one can show that $f^c(z,\eta) \leq \overline{\mathsf{F}}^{\mathrm{con}}_{[y,\widehat{y}]}(x,\widehat{x},w,\widehat{w})$. □

Remark 5.

(i) *(Jacobian-based cornered inclusion function):* Theorem 3 provides a localized inclusion function for the closed-loop vector field f^c using the Jacobian-based cornered inclusion function (as in Proposition 1) for the corner point (x,u,w). First, one can obtain different but similar localized inclusion functions using Taylor expansion of f around other corner points such as $(\widehat{x},u,w)$, $(\widehat{x},\widehat{u},w)$, etc. Our numerical results show that, in some cases, considering a few corner points and combining the resulting localized inclusion functions using Proposition 4 significantly reduces conservatism at low computational

expense. Second, as it is clear from the proof of Theorem 3, the Jacobian-based cornered inclusion function is particularly well-suited for integrating the neural network bounds into the closed-loop inclusion function and for studying the interaction between the system and the neural network controller.

(ii) *(Mixed Jacobian-based inclusion functions):* Theorem 3 uses the Jacobian-based cornered inclusion function (as in Proposition 1) for the open-loop vector field f. Alternatively, one can use the mixed Jacobian-based cornered inclusion function (as in Proposition 2) for the open-loop vector f and obtain a different class of inclusion functions for f^c. This class of closed-loop inclusion functions can be obtained from (17) by replacing $\underline{J}_{[x,\widehat{x}]}, \overline{J}_{[x,\widehat{x}]}, \underline{J}_{[u,\widehat{u}]}, \overline{J}_{[u,\widehat{u}]}, \underline{J}_{[w,\widehat{w}]}$, and $\overline{J}_{[w,\widehat{w}]}$ with their counterparts $\underline{M}_{[x,\widehat{x}]}, \overline{M}_{[x,\widehat{x}]}, \underline{M}_{[u,\widehat{u}]}, \overline{M}_{[u,\widehat{u}]}, \underline{M}_{[w,\widehat{w}]}$, and $\overline{M}_{[w,\widehat{w}]}$ as defined in Proposition 2 and is demonstrated in the numerical experiments below. Our numerical results show that, in most cases, mixed Jacobian-based cornered inclusion functions significantly reduces the conservatism of the over-approximation as compared to their non-mixed counterparts.

(iii) *(Comparison):* The interconnection-based approach provides a general and computationally efficient framework for constructing closed-loop inclusion functions. This approach can be applied using any inclusion function for the open-loop system and any inclusion function of the form (10) for the neural network. The computational efficiency of the interconnection-based approach is a direct consequence of its compositional structure. On the other hand, the interaction-based approach fully captures the first-order interaction between the system and the neural networks using a Jacobian-based decomposition. However, this approach requires affine bounds of the form (11) for the neural network and Jacobian bounds of the form (16) for the open-loop system. As a result, computing the closed-loop inclusion functions using the interaction-based approach is generally more computationally expensive.

When the open-loop system (3) is linear, one can significantly simplify the expression for the closed-loop inclusion functions obtained from the interconnection-based and the interaction-based approaches.

Corollary 1 (Linear open-loop systems)**.** *Consider the closed-loop system (5) and assume that the open-loop vector field f is linear with the form $f(x,u,w) = Ax + Bu + Dw$, where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times p}$ and $D \in \mathbb{R}^{n\times q}$. Then the following statements hold:*

(i) using the minimal inclusion function for the open-loop vector field f and affine neural network inclusion functions of the form (12), the closed-loop inclusion function $\mathsf{F}^{\mathrm{con}}_{[y,\widehat{y}]}$ defined in (15) is given by:

$$\mathsf{F}^{\mathrm{con}}_{[y,\widehat{y}]} = \begin{bmatrix} [A]^+ + \big[[B]^+\underline{C} + [B]^-\overline{C}\big]^+ & [A]^- + \big[[B]^+\underline{C} + [B]^-\overline{C}\big]^- \\ [A]^- + \big[[B]^+\overline{C} + [B]^-\underline{C}\big]^- & [A]^+ + \big[[B]^+\overline{C} + [B]^-\underline{C}\big]^+ \end{bmatrix} \begin{bmatrix} x \\ \widehat{x} \end{bmatrix} + \begin{bmatrix} [D]^+ & [D]^- \\ [D]^- & [D]^+ \end{bmatrix} \begin{bmatrix} w \\ \widehat{w} \end{bmatrix} + \begin{bmatrix} [B]^+\underline{d} + [B]^-\overline{d} \\ [B]^-\underline{d} + [B]^+\overline{d} \end{bmatrix}. \tag{18}$$

(ii) the inclusion function $\mathsf{F}^{\mathrm{act}}_{[y,\widehat{y}]}$ defined in (17) is given by

$$\mathsf{F}^{\mathrm{act}}_{[y,\widehat{y}]} = \begin{bmatrix} [A+B^+\underline{C}+B^-\overline{C}]^+ & [A+B^+\underline{C}+B^-\overline{C}]^- \\ [A+B^+\overline{C}+B^-\underline{C}]^- & [A+B^+\overline{C}+B^-\underline{C}]^+ \end{bmatrix} \begin{bmatrix} x \\ \widehat{x} \end{bmatrix} + \begin{bmatrix} [D]^+ & [D]^- \\ [D]^- & [D]^+ \end{bmatrix} \begin{bmatrix} w \\ \widehat{w} \end{bmatrix} + \begin{bmatrix} [B]^+\underline{d} + [B]^-\overline{d} \\ [B]^-\underline{d} + [B]^+\overline{d} \end{bmatrix}. \tag{19}$$

Proof. Regarding part (i), it is easy to see that the minimal inclusion function of the linear open-loop vector field f is

$$\mathsf{F}^{\mathrm{o}}(x,\widehat{x},u,\widehat{u},w,\widehat{w}) = \begin{bmatrix} [A]^+ & [A]^- \\ [A]^- & [A]^+ \end{bmatrix} \begin{bmatrix} x \\ \widehat{x} \end{bmatrix} + \begin{bmatrix} [B]^+ & [B]^- \\ [B]^- & [B]^+ \end{bmatrix} \begin{bmatrix} u \\ \widehat{u} \end{bmatrix} + \begin{bmatrix} [D]^+ & [D]^- \\ [D]^- & [D]^+ \end{bmatrix} \begin{bmatrix} w \\ \widehat{w} \end{bmatrix}.$$

One can then replace the above minimal inclusion function and the affine bounds (12) into equation (15) and use Theorem 2 to obtain the result. Regarding part (ii), we use the bounds $\underline{J}_{[x,\widehat{x}]} = \overline{J}_{[x,\widehat{x}]} = A$, $\underline{J}_{[u,\widehat{u}]} = \overline{J}_{[u,\widehat{u}]} = B$, and $\underline{J}_{[w,\widehat{w}]} = \overline{J}_{[w,\widehat{w}]} = D$ in Theorem 3. Thus, using the notation of Theorem 3, we compute

$$\underline{H} = A + [B]^+\underline{C}(y,\widehat{y}) + [B]^-\overline{C}(y,\widehat{y}),$$
$$\overline{H} = A + [B]^+\overline{C}(y,\widehat{y}) + [B]^-\underline{C}(y,\widehat{y}),$$
$$L = \begin{bmatrix} -[D]^- & [D]^- \\ -[D]^+ & [D]^+ \end{bmatrix}, \qquad Q = \begin{bmatrix} Ax+Dw+[B]^+\underline{d}(y,\widehat{y})+[B]^-\overline{d}(y,\widehat{y}) \\ Ax+Dw+[B]^+\overline{d}(y,\widehat{y})+[B]^-\underline{d}(y,\widehat{y}) \end{bmatrix}.$$

The result then follows by replacing the above terms into equation (17) and using Theorem 3. □

Remark 6 (The role of interactions)**.** The closed-form expressions (18) and (19) demonstrate the difference between the interconnection-based and the interaction-based approach. In both cases, one can interpret the term A as the effect of open-loop dynamics and the terms $B^+\underline{C} + B^-\overline{C}$ and $B^+\overline{C}+B^-\underline{C}$ as the effect of the neural network controller. The interconnection-based approach considers the interconnection of the cooperative and competitive effect of the open-loop system and the neural network controller, separately. This leads to the terms $[A]^\pm + [B^+\underline{C}+B^-\overline{C}]^\pm$ and $[A]^\pm + [B^+\overline{C}+B^-\underline{C}]^\pm$. The interaction-based approach instead considers the interaction between the open-loop system and the neural network controller via the terms $A + B^+\underline{C} + B^-\overline{C}$ and $A + B^+\overline{C} + B^-\underline{C}$ and then the cooperative and competitive effect of these interactions are separated.

C. Interval Reachability of Closed-loop Systems

Our culminating conclusion is that, by using the interconnection-based and the interaction-based inclusion functions, we obtain computationally efficient over-approximations of reachable sets of the closed-loop system.

Theorem 4 (Reachability of closed-loop system)**.** *Let* $\mathrm{c} \in \{\mathrm{con}, \mathrm{act}\}$*, the disturbance set $\mathcal{W} = [\underline{w}, \overline{w}]$, and the initial set $\mathcal{X}_0 = [\underline{x}_0, \overline{x}_0]$. For the closed-loop dynamical system (5) with $t \mapsto \begin{bmatrix} \underline{x}^{\mathrm{c}}(t) \\ \overline{x}^{\mathrm{c}}(t) \end{bmatrix}$ being the trajectory of the embedding system*

$$\dot{x}_i = \Big(\underline{\mathsf{F}}^{\mathrm{c}}_{[x,\widehat{x}]}(x, \widehat{x}_{[i:x]}, \underline{w}, \overline{w})\Big)_i,$$
$$\dot{\widehat{x}}_i = \Big(\overline{\mathsf{F}}^{\mathrm{c}}_{[x,\widehat{x}]}(x_{[i:\widehat{x}]}, \widehat{x}, \overline{w}, \underline{w})\Big)_i, \tag{20}$$

for every $i \in \{1,\dots,n\}$, starting from $\left[\begin{smallmatrix}\underline{x}_0\\ \overline{x}_0\end{smallmatrix}\right]$, the following statements hold, for every $t \geq 0$:

(i) $\mathcal{R}_{f^c}(t,[\underline{x}_0,\overline{x}_0],[\underline{w},\overline{w}]) \subseteq [\underline{x}^{\mathrm{c}}(t),\overline{x}^{\mathrm{c}}(t)]$;

(ii) moreover, if the open-loop system (3) is linear, then

$$[\underline{x}^{\mathrm{act}}(t),\overline{x}^{\mathrm{act}}(t)] \subseteq [\underline{x}^{\mathrm{con}}(t),\overline{x}^{\mathrm{con}}(t)].$$

Proof. Regarding part (i), using Theorem 2 and Theorem 3, we show that $\mathsf{F}^{\mathrm{con}}_{[y,\widehat{y}]}$ and $\mathsf{F}^{\mathrm{act}}_{[y,\widehat{y}]}$ are $[y,\widehat{y}]\times\mathbb{R}^q$-localized inclusion functions for the closed-loop vector field f^c. Thus, one can use the localized version of Proposition 5 (see Remark 4(i)) with $y = x$ and $\widehat{y} = \widehat{x}$ to obtain the result. Regarding part (ii), for linear open-loop systems, using Theorem 1, the embedding system associated to the interconnection-based inclusion function has the following form:

$$\begin{aligned}\tfrac{d}{dt}\left[\begin{smallmatrix}x\\ \widehat{x}\end{smallmatrix}\right] &= \begin{bmatrix}[A]^{\mathrm{M}}+[B^+\underline{C}+B^-\overline{C}]^{\mathrm{M}} & [A]^{\mathrm{nM}}+[B^+\underline{C}+B^-\overline{C}]^{\mathrm{nM}}\\ [A]^{\mathrm{nM}}+[B^+\underline{C}+B^-\overline{C}]^{\mathrm{nM}} & [A]^{\mathrm{M}}+[B^+\underline{C}+B^-\overline{C}]^{\mathrm{M}}\end{bmatrix}\left[\begin{smallmatrix}x\\ \widehat{x}\end{smallmatrix}\right]\\ &\quad+\begin{bmatrix}[D]^+ & [D]^-\\ [D]^- & [D]^+\end{bmatrix}\left[\begin{smallmatrix}w\\ \widehat{w}\end{smallmatrix}\right]+\begin{bmatrix}[B]^+\underline{d}+[B]^-\overline{d}\\ [B]^-\underline{d}+[B]^+\overline{d}\end{bmatrix}\\ &= \mathsf{F}^{\mathrm{con}}(x,\widehat{x},w,\widehat{w}),\end{aligned} \tag{21}$$

and the embedding system associated to the interaction-based inclusion function has the following form:

$$\begin{aligned}\tfrac{d}{dt}\left[\begin{smallmatrix}x\\ \widehat{x}\end{smallmatrix}\right] &= \begin{bmatrix}[A+[B]^+\underline{C}+[B]^-\overline{C}]^{\mathrm{M}} & [A+[B]^+\underline{C}+[B]^-\overline{C}]^{\mathrm{nM}}\\ [A+[B]^+\underline{C}+[B]^-\overline{C}]^{\mathrm{nM}} & [A+[B]^+\underline{C}+[B]^-\overline{C}]^{\mathrm{M}}\end{bmatrix}\left[\begin{smallmatrix}x\\ \widehat{x}\end{smallmatrix}\right]\\ &\quad+\begin{bmatrix}[D]^+ & [D]^-\\ [D]^- & [D]^+\end{bmatrix}\left[\begin{smallmatrix}w\\ \widehat{w}\end{smallmatrix}\right]+\begin{bmatrix}[B]^+\underline{d}+[B]^-\overline{d}\\ [B]^-\underline{d}+[B]^+\overline{d}\end{bmatrix}\\ &= \mathsf{F}^{\mathrm{act}}(x,\widehat{x},w,\widehat{w}).\end{aligned} \tag{22}$$

It is easy to check that both embedding systems (21) and (22) are continuous-time monotone. Moreover, we have

$$\begin{aligned}\left[A+[B]^+\underline{C}+[B]^-\overline{C}\right]^{\mathrm{M}} &\leq [A]^{\mathrm{M}}+\left[[B]^+\underline{C}+[B]^-\overline{C}\right]^{\mathrm{M}},\\ \left[A+[B]^+\underline{C}+[B]^-\overline{C}\right]^{\mathrm{nM}} &\geq [A]^{\mathrm{nM}}+\left[[B]^+\underline{C}+[B]^-\overline{C}\right]^{\mathrm{nM}}.\end{aligned}$$

Therefore, for every $x \leq \widehat{x}$, we have

$$\begin{aligned}&\left[A+[B]^+\underline{C}+[B]^-\overline{C}\right]^{\mathrm{M}}x+\left[A+[B]^+\underline{C}+[B]^-\overline{C}\right]^{\mathrm{nM}}\widehat{x}\\ &\quad\geq([A]^{\mathrm{M}}+[[B]^+\underline{C}+[B]^-\overline{C}]^{\mathrm{M}})x\\ &\qquad+([A]^{\mathrm{nM}}+[[B]^+\underline{C}+[B]^-\overline{C}]^{\mathrm{nM}})\widehat{x}.\end{aligned}$$

This implies that, $\mathsf{F}^{\mathrm{con}}(x,\widehat{x},w,\widehat{w}) \leq_{\mathrm{SE}} \mathsf{F}^{\mathrm{act}}(x,\widehat{x},w,\widehat{w})$, for every $x \leq \widehat{x}$ and every $w \leq \widehat{w}$. Now, we can use the classical monotone comparison Lemma [47, Theorem 3.8.1] to obtain $\left[\begin{smallmatrix}\underline{x}^{\mathrm{con}}(t)\\ \overline{x}^{\mathrm{con}}(t)\end{smallmatrix}\right] \leq_{\mathrm{SE}} \left[\begin{smallmatrix}\underline{x}^{\mathrm{act}}(t)\\ \overline{x}^{\mathrm{act}}(t)\end{smallmatrix}\right]$ for every $t \in \mathbb{R}_{\geq 0}$. As a result, we get $[\underline{x}^{\mathrm{act}}(t),\overline{x}^{\mathrm{act}}(t)] \subseteq [\underline{x}^{\mathrm{con}}(t),\overline{x}^{\mathrm{con}}(t)]$, for every $t \in \mathbb{R}_{\geq 0}$. □

Remark 7. The interval reachability framework developed in Sections V and VI has the following unique features.

(i) *(Computational efficiency)*: from a computational perspective, the framework presented in Theorem 4 consists of two main ingredients: (i) the neural network verification algorithm for computing the inclusion function for the neural network (either in the form (10) or in the form (12)), and (ii) integration of a single trajectory of the embedding system (13). Part (i) includes querying the neural network verification once per integration step and its runtime depends on the computational complexity of the associated algorithm. The runtime of part (ii) depends on the integration method and the form of the open-loop decomposition function F. Since our approach only requires integration of one trajectory of the embedding system, it is generally computationally efficient. In Section VII-D, we use a vehicle platooning example to demonstrate our approach's scalability to large dimensions where existing functional approaches suffer.

(ii) *(Nonlinearity of the dynamics):* For linear open-loop systems, a straightforward computation shows that Theorem 4 with $c = \mathrm{act}$ coincides with [30, Lemma IV.3] with $p = \infty$ and $q = 1$. Our interaction-aware inclusion function generalizes this efficient approach to locally Lipschitz nonlinear systems.

(iii) *(Flexibility of neural network verifiers):* The functional approaches from POLAR [35] and JuliaReach [36] are designed with specific neural network bounding algorithms. On the other hand, our framework can substitute arbitrary neural network interval verifiers for the interconnection-based approach, and functional linear bounds for the interaction-based approach.

VII. Numerical Experiments

We demonstrate the effectiveness of our reachability analysis, implemented as an open-source Python toolbox called `ReachMM`[2], using numerical experiments[3] for (i) a nonlinear bicycle model, (ii) the linear double integrator, (iii) several existing benchmarks in the literature, and (iv) a platoon of double integrator vehicles moving in a plane. We first briefly mention several algorithmic techniques that are implemented in the `ReachMM` toolbox and are used in several examples below to broaden the applicability and to improve the accuracy of our reachable set over-approximations.

Partitioning: The computational efficiency of our method makes it well-suited for partitioning to reduce compounding over-approximation error caused by the wrapping effect [10, Section 2.2.4]. In Section VII-B, we apply the partitioning methods proposed in [38], [51] to improve accuracy.

Redundant Variable Refinement: We accommodate a technique introduced in [13] to refine interval over-approximations of reachable sets using redundant state variables of the form $y = Ax+b$ for some $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. By augmenting the dynamical system (5) with the additional dynamics $\dot{y} = A\dot{x}$, we create a new dynamical system with the new states $z = \left[\begin{smallmatrix}x\\ y\end{smallmatrix}\right] \in \mathbb{R}^{n+m}$ and the affine constraints $Mz = b$ where $M = \begin{bmatrix}-A & I_m\end{bmatrix} \in \mathbb{R}^{m\times(n+m)}$. Following [13], we use the interval bounds on z to improve the accuracy of the interval bounds on x. We apply this technique in Section VII-C for the TORA benchmark.

Zero-order hold control: In some applications, we wish to explicitly account for the practical restriction that the control $u(t) = N(x(t))$ cannot be continuously updated and, instead, the control must be implemented via, *e.g.*, a zero-order hold strategy between sampling instances. It is straightforward to extend the interconnection-based approach to be able to

[2] https://github.com/gtfactslab/ReachMM_TAC2023

[3] All the experiments are performed using an AMD Ryzen 5 5600X CPU and 32 GB of RAM. Unless otherwise specified, runtimes are averaged over 100 runs, with mean and standard deviations reported.

capture the zero-order hold policy [51]. Adopting the zero-order hold policy in the interaction-based approach requires over-bounding the error and we omit this calculation for the sake of brevity. We refer to our code for details of this adaptation. We use this framework in Section VII-C.

Implementation: Our current implementation for all examples is in standard Python. For the interconnection-based approach, the inclusion function for the open-loop system is the natural inclusion functions computed using our package `npinterval` [41]. For neural networks, the affine inclusion functions are obtained from CROWN [25] and computed using `autoLiRPA` [52]. Further performance improvements of `ReachMM` are likely possible by, *e.g.*, implementing as compiled code, and are subject of ongoing and future work.

A. Nonlinear bicycle model

In the first experiment, we compare the interconnection-based and the interaction-based approach. Consider the nonlinear dynamics of a bicycle adopted from [53]:

$$\begin{aligned} \dot{p_x} &= v\cos(\phi+\beta(u_2)) & \dot{\phi} &= \frac{v}{\ell_r}\sin(\beta(u_2)) \\ \dot{p_y} &= v\sin(\phi+\beta(u_2)) & \dot{v} &= u_1 \end{aligned} \tag{23}$$

where $[p_x, p_y]^\top \in \mathbb{R}^2$ is the displacement of the center of mass in the $x-y$ plane, $\phi \in [-\pi, \pi)$ is the heading angle in the plane, and $v \in \mathbb{R}_{\geq 0}$ is the speed of the center of mass. Control input u_1 is the applied force, input u_2 is the angle of the front wheel, and $\beta(u_2) = \arctan\left(\frac{\ell_f}{\ell_f+\ell_r}\tan(u_2)\right)$ is the slip slide angle where the parameter ℓ_f (ℓ_r) is the distance between the center of mass and the front (rear) wheel. In this example, for the sake of simplicity, we set $\ell_f = \ell_r = 1$. Let $x = [p_x, p_y, \phi, v]^\top$ and $u = [u_1, u_2]^\top$. We apply the neural network controller ($4 \times 100 \times 100 \times 2$, ReLU activations) defined in [38], which was trained to mimic an MPC that stabilizes the vehicle to the origin while avoiding a circular obstacle centered at $(4, 4)$ with a radius of 2. The dynamics are simulated using Euler integration with a step size of 0.125. In Figure 3, we compare the accuracy and runtime of different reachability approaches for the bicycle model.

Discussion: Figure 3 shows that the naive input-output approach is the fastest approach with low accuracy of over-approximation. Using the interconnection-based approach with $\mathsf{F}^{\mathrm{con}}$ improves the accuracy with a slight increase in the runtime. In comparison, the interaction-based approach with the Jacobian-based cornered inclusion function $\mathsf{F}^{\mathrm{act}}$ is notably slower due to the computation of Jacobian bounds (16) in this approach. Over short time horizons, the interaction-based approach is more accurate, however, its accuracy deteriorates for longer horizons, which can be attributed to the decrease in accuracy of the Jacobian bounds (16). As has been shown in Proposition 4, the accuracy of the intersection $\mathsf{F}^{\mathrm{con}} \wedge \mathsf{F}^{\mathrm{act}}$ is better than both $\mathsf{F}^{\mathrm{con}}$ and $\mathsf{F}^{\mathrm{act}}$. Finally, using the mixed Jacobian-based cornered inclusion function (cf. Remark 5(ii)) significantly improves the accuracy of the interaction-based approach with little effect on its runtime.

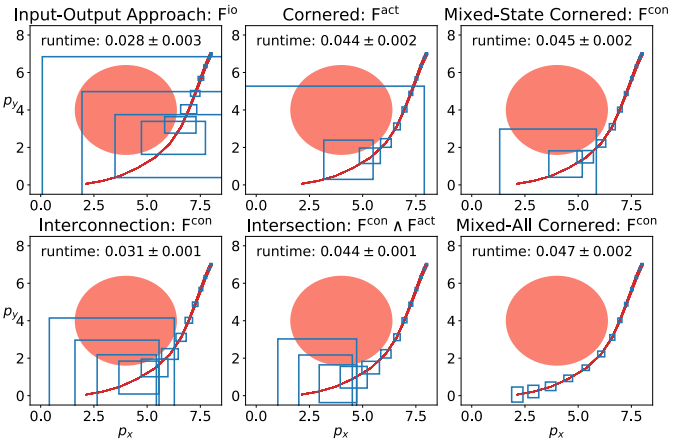

Fig. 3. Accuracy and runtime comparison between different reachability approaches for the bicycle model (23) from the initial set $[\mathbf{7.95, 8.05}] \times [\mathbf{6.95, 7.05}] \times [\mathbf{-2\pi/3 - 0.01, -2\pi/3 + 0.01}] \times [\mathbf{1.99, 2.01}]$ for the period $t \in [\mathbf{0, 1.5}]$: (a) a naive input-output approach combining the natural inclusion function for the open-loop dynamics and the affine inclusion function for the neural network, (b) the interconnection-based approach with the closed-loop inclusion function $\mathbf{F}^{\mathbf{con}}$ defined in (15) constructed from the natural inclusion function for the open loop dynamics and affine inclusion function for the neural network, (c) the interaction-based approach with Jacobian-based cornered inclusion function $\mathbf{F}^{\mathbf{act}}$ defined in (17), (d) the intersection of the interconnection-based inclusion and interaction-based approach $\mathbf{F}^{\mathbf{con}} \wedge \mathbf{F}^{\mathbf{act}}$, (e) the interaction-based approach with *mixed states* Jacobian-based cornered inclusion function defined in (17) and Remark 5(ii), and (f) the interaction-based approach with *mixed states and control* Jacobian-based cornered inclusion function defined (17) and Remark 5(ii). The blue boxes are hyper-rectangular over-approximation of reachable sets, **100** simulated trajectories of the system are shown in red, and the salmon colored region represents a circular obstacle.

B. Double Integrator Model

In the second experiment, we focus on reachability of linear open-loop systems with neural network controllers. We study the accuracy and efficiency of the interaction-based approach (19) for the double integrator benchmark system

$$x(t+1) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u(t). \tag{24}$$

For this example, we use the discrete-time version of our framework as derived in [51, Section VII.B]. We apply the neural network controller ($2 \times 10 \times 5 \times 1$, ReLU activations) defined in [30]. We consider the interaction-based approach with the closed-loop inclusion function $\mathsf{F}^{\mathrm{act}}$ defined in (19), with the contraction-guided adaptive partitioning algorithm in [51] to improve its accuracy. Additionally, we compare our proposed ReachMM to state-of-the-art algorithms for linear discrete-time systems: ReachLP [30] with uniform partitioning (ReachLP-Unif) and greedy simulation guided partitioning (ReachLP-GSG), and ReachLipBnB [54] (branch-and-bound using LipSDP [24]). Each algorithm is run with two different sets of hyper-parameters, aiming to compare their performances across various regimes. The setup for ReachMM is $(\varepsilon,\ D_p,\ D_{\mathrm{N}})$[4]; ReachLP-Unif is # initial partitions, ReachLP-GSG is # of total propogator calls, ReachLipBnB is ε. The

[4] D_p defines the depth of the partition tree, D_{N} defines how the depth of the partitions on the neural network, and ε describes the width at which to adaptively partition an interval. See [51] for the full details.

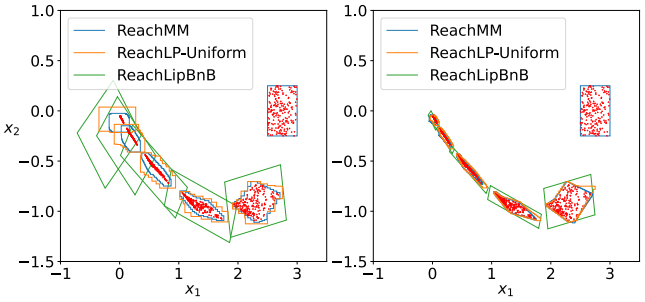

Fig. 4. The over-approximated reachable sets of the closed-loop double integrator model (24) are compared for three different algorithms on two different runtime regimes, for the initial set $[\mathbf{2.5}, \mathbf{3}] \times [-\mathbf{0.25}, \mathbf{0.25}]$ and final time $\boldsymbol{T = 5}$. The experiment setup and performances are reported in Table II. 200 true trajectories are shown in red. The horizontal axis is $\boldsymbol{x_1}$ and the vertical axis is $\boldsymbol{x_2}$.

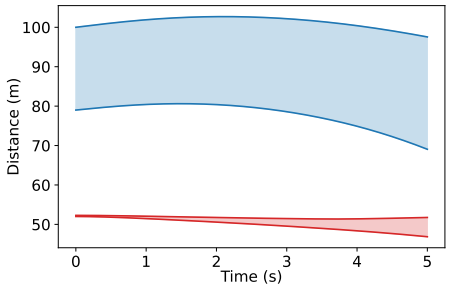

Fig. 5. The interval estimates of the safety specifications for the Adaptive Cruise Control benchmark obtained from the inclusion function $\mathbf{F^{con}}$. The upper bound and lower bound on $\boldsymbol{D}_{\text{rel}}$ are shown in blue and the upper and lower bounds on $\boldsymbol{D}_{\text{safe}}$ are shown in red. Our approach ensures that $\boldsymbol{D}_{\text{rel}} \geq \boldsymbol{D}_{\text{safe}}$ for $\boldsymbol{t} \in [\mathbf{0}, \mathbf{5}]$.

results are illustrated in Figure 4 and the runtime comparisons are provided in Table II.

Method	Setup	Runtime (s)	Area
ReachMM	(0.1, 3, 1)	$\mathbf{0.103 \pm 0.003}$	$\mathbf{6.2 \cdot 10^{-2}}$
	(0.05, 6, 2)	$\mathbf{1.762 \pm 0.026}$	$\mathbf{9.9 \cdot 10^{-3}}$
ReachLP-Unif	4	0.212 ± 0.002	$1.5 \cdot 10^{-1}$
	16	3.149 ± 0.004	$1.0 \cdot 10^{-2}$
ReachLP-GSG	55	0.913 ± 0.031	$5.3 \cdot 10^{-1}$
	205	2.164 ± 0.042	$8.8 \cdot 10^{-2}$
ReachLipBnB	0.1	0.956 ± 0.067	$5.4 \cdot 10^{-1}$
	0.001	3.681 ± 0.100	$1.2 \cdot 10^{-2}$

TABLE II
PERFORMANCE OF THEOREM 3 FOR THE DISCRETE-TIME DOUBLE INTEGRATOR MODEL.

Discussion: Figure 4 and Table II show that, when the open-loop system is linear, our interaction-based approach (Corollary 1(ii)) combined with a suitable partitioning scheme beats state-of-the-art approaches in both accuracy and runtime.

C. ARCH-COMP Benchmarks

In the third experiment, we analyze three of the benchmarks from ARCH-COMP22 [55].

Adaptive Cruise Control (ACC): We consider the Adaptive Cruise Control benchmark from [55, Equation (1)]. The neural network controller ($5 \times 20 \times 20 \times 20 \times 20 \times 20 \times 1$, ReLU activations) with the input $(x_{\text{lead}} - x_{\text{ego}}, v_{\text{lead}} - v_{\text{ego}}, a_{\text{lead}} - a_{\text{ego}}, v_{\text{set}}, T_{\text{gap}})^\top \in \mathbb{R}^5$ is applied with a zero-order holding of 0.1 seconds [55]. Our goal is to verify that from the initial set $\mathcal{X}_0 = [90, 110] \times [32, 32.2] \times [0, 0] \times [10, 11] \times [30, 30.2] \times [0, 0]$, the collision specification

$$x_{\text{lead}} - x_{\text{ego}} \geq D_{\text{default}} + T_{\text{gap}} x_{\text{ego}}$$

is never violated in the next 5 seconds, where $D_{\text{default}} = 10$ and $T_{\text{gap}} = 1.4s$. To verify the specification, we define $D_{\text{rel}} = x_{\text{lead}} - x_{\text{ego}}$ and $D_{\text{safe}} = D_{\text{default}} + T_{\text{gap}} x_{\text{ego}}$ and ensure that $D_{\text{rel}} - D_{\text{safe}} \geq 0$ for the given time horizon.

We consider the interconnection-based approach with the closed-loop inclusion function F^{con} defined in (15) constructed from the natural inclusion function for open-loop system and an affine inclusion function for the neural network. We use this interconnection-based inclusion function and Euler integration with a step-size of 0.01 to compute upper and lower bounds on the states of the system starting from the initial set $\mathcal{X}_0$. These bounds are then used to obtain upper and lower bounds on D_{rel} and D_{safe}. The results are shown in Figure 5 and the comparison between the runtime of our approach with POLAR [35] and JuliaReach [36] is provided in Table III.

2D Spacecraft Docking: We consider the 2D spacecraft docking model [55, Equation (13)]. The neural network controller ($4 \times 4 \times 256 \times 256 \times 4 \times 2$, $\tanh$ activations) with a zero-order holding of 1 second [55]. The goal is to verify that given an initial state set, the safety specification

$$(\dot{s}_x^2 + \dot{s}_x^2)^{\frac{1}{2}} \leq 0.2 + 2n(s_x^2 + s_y^2)^{\frac{1}{2}} \tag{25}$$

is never violated in the next 40 seconds. We define $H_L = (\dot{s}_x^2 + \dot{s}_x^2)^{\frac{1}{2}}$ and $H_R = 0.2 + 2n(s_x^2 + s_y^2)^{\frac{1}{2}}$ and our goal is to check that $H_L \leq H_R$ for the next 40 seconds.

We consider the interconnection-based approach with the closed-loop inclusion function F^{con} defined in (15) constructed from the natural inclusion function for the open-loop system and an affine inclusion function for the neural network. We use the closed-loop inclusion function F^{con} and Euler integration with a step-size 0.1 to obtain upper and lower bounds on the states of the system starting from four different initial sets $\mathcal{X}_0^i$ for $i \in \{0, 1, 2, 3\}$ as shown in Figure 6. These bounds are then used to provide upper and lower bounds on H_L and H_R. The results are shown in Figure 6 and the comparison between the runtime of our approach with POLAR [35] and JuliaReach [36] is provided in Table III.

Sherlock-Benchmark-9 (TORA): We consider the translational oscillations by a rotational actuator (TORA) benchmark from [55, Equation (2)]. The neural network controller ($4 \times 20 \times 20 \times 20 \times 1$ with ReLU activation functions and an additional layer with $\tanh$ activation function) is applied with a zero-order holding of 0.5 seconds [55]. The goal is to verify whether the system reaches the region $[-0.1, 0.2] \times [-0.9, -0.6] \times \mathbb{R}^2$ from the initial set $\mathcal{X}_0 = [-0.77, -0.75] \times [-0.45, -0.43] \times [0.51, 0.54] \times [-0.3, -0.28]$ within 5 seconds.

We add the affine redundant variables $y_1 = x_1 + x_2$ and $y_2 = x_1 - x_2$ and analyze the augmented system in $z = (x_1, x_2, y_1, y_2)^\top$. We consider the interaction-based approach with the mixed Jacobian-based cornered inclusion

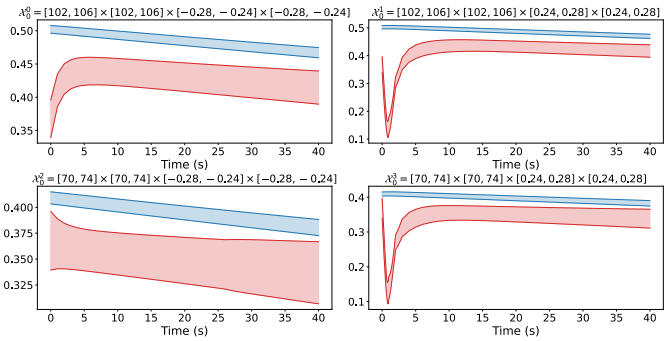

Fig. 6. The interval estimates of the safety specification for the 2D Spacecraft Docking benchmark obtained from the interconnection-based inclusion function $\mathbf{F^{con}}$ are pictured for four different initial sets $\{\boldsymbol{\mathcal{X}_0^i}\}_{i=0}^{3}$. The upper and lower bounds on $\boldsymbol{H_L}$ are shown in red and the upper and lower bounds on $\boldsymbol{H_R}$ are shown in blue. Our approach verifies that $\boldsymbol{H_L} \leq \boldsymbol{H_R}$ for $\boldsymbol{t} \in [\mathbf{0}, \mathbf{40}]$.

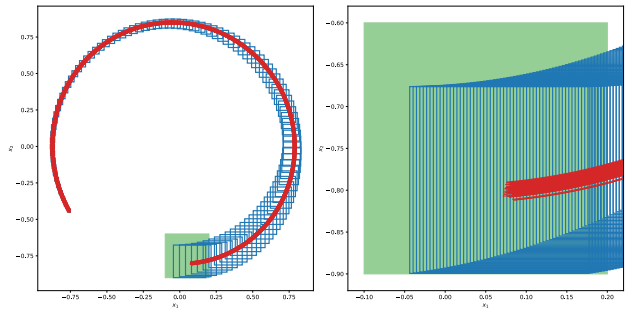

Fig. 7. The reachable set for the TORA benchmark is shown on the $\boldsymbol{x_1}$-$\boldsymbol{x_2}$ plane. The goal set is pictured in green, and 100 true trajectories of the system are pictured in red. **Left:** Every tenth over-approximation of the reachable set is pictured in blue. **Right:** The plot is zoomed into the goal set, and all over-approximations of the reachable set are shown in blue. We numerically verify that the final reachable set is fully contained inside of the goal region.

function $\mathsf{F}^{\mathrm{act}}$ as defined in (17) constructed for four corners $(z,u),(z,\widehat{u}),(\widehat{z},u),(\widehat{z},\widehat{u})$ (cf. Remark 5). We use the intersection of the two inclusion functions $\mathsf{F}^{\mathrm{con}} \wedge \mathsf{F}^{\mathrm{act}}$ with Euler integration step size of 0.005 to provide over-approximations of reachable sets of the system starting from $\mathcal{X}_0$. The results are shown in Figure 7 and the comparison between the runtime of our approach with POLAR [35] and JuliaReach [36] is provided in Table III.

System	ReachMM	POLAR	JuliaReach
ACC	0.398 ± 0.010	1.388 ± 0.022	0.255
Docking	0.243 ± 0.004	43.696 ± 0.141	N/A[5]
TORA	5.586 ± 0.029	0.139 ± 0.008	0.690

TABLE III
SUMMARY OF RUNTIMES (S) FROM THE ARCH-COMP BENCHMARKS.

Discussion: We showed that for verification of safety specification on a selected number of examples from ARCH-COMP [55], the runtime of our approach is comparable with the state-of-the-art methods POLAR [35] and JuliaReach [37]. On the other benchmarks in [55], our method is unable to verify the desired specification at this time, due to their large time horizons and excessive over-approximation buildup. In the next section, however, we show that our interconnection-based and interaction-based methods enjoy better scalability compared to POLAR and JuliaReach.

[5]JuliaReach did not attempt this specification.

N (units)	# of states	$\mathsf{F}^{\mathrm{con}}$	$\mathsf{F}^{\mathrm{act}}$	POLAR	JuliaReach
1	4	0.297	0.635	9.352	0.224
4	16	0.399	1.369	14.182	12.579
9	36	0.574	3.144	43.428	59.929
20	80	0.999	9.737	316.337	–[6]
50	200	2.420	46.426	4256.435	–

TABLE IV
THE RUNTIMES (S) FOR THE PLATOONING BENCHMARK.

D. Vehicle Platooning

Finally, we investigate the scalability of our method. We consider a platoon of N vehicles $\mathcal{V} = \{\mathcal{V}_j\}_{j=1}^{N}$, each with the two-dimensional double integrator dynamics

$$\begin{aligned} \dot{p}_x^j &= v_x^j, & \dot{v}_x^j &= \sigma(u_x^j) + w_x^j, \\ \dot{p}_y^j &= v_y^j, & \dot{v}_y^j &= \sigma(u_y^j) + w_y^j, \end{aligned} \tag{26}$$

where $p^j = (p_x^j, p_y^j) \in \mathbb{R}^2$ is the displacement of the center of mass of $\mathcal{V}_j$, $v^j = (v_x^j, v_y^j) \in \mathbb{R}^2$ is the velocity of the center of mass of $\mathcal{V}_j$, $(u_x^j, u_y^j) \in \mathbb{R}^2$ are desired acceleration inputs limited by the softmax operator $\sigma(u) = u_{\mathrm{lim}} \tanh(u/u_{\mathrm{lim}})$ with $u_{\mathrm{lim}} = 5$, and $w_x^j, w_y^j \sim \mathcal{U}([-0.001, 0.001])$ are uniformly distributed disturbances. We consider a leader-follower structure for the system, where the first vehicle $\mathcal{V}_1$ chooses its control $u = (u_x^1, u_y^1)$ as the output of a neural network ($4 \times 100 \times 100 \times 2$, ReLU activations), and the rest of the platoon $\{\mathcal{V}_j\}_{j=2}^{N}$ applies a PD tracking control input

$$u_{\mathbf{d}}^j = k_p \left(p_{\mathbf{d}}^{j-1} - p_{\mathbf{d}}^j - r \frac{v_{\mathbf{d}}^{j-1}}{\|v^{j-1}\|_2} \right) + k_v (v_{\mathbf{d}}^{j-1} - v_{\mathbf{d}}^j), \tag{27}$$

for each $\mathbf{d} \in \{x, y\}$ with $k_p = k_v = 5$ and $r = 0.5$. The neural network was trained using data from an offline MPC policy for the leader only ($N = 1$). The offline policy minimized a quadratic cost stabilizing to the origin while avoiding a circular obstacle centered at $(4, 4)$ with radius 2.25, implemented as a hard constraint with 33% padding and a slack variable.

We consider the interconnection-based approach with closed-loop inclusion function $\mathsf{F}^{\mathrm{con}}$ defined in (15) constructed from the natural inclusion function for open-loop system. We also consider the interaction-based approach with the closed-loop mixed Jacobian-based cornered inclusion function $\mathsf{F}^{\mathrm{act}}$ as defined in (17) constructed for four corners $(x,u),(u,\widehat{u}),(\widehat{x},u),(\widehat{x},\widehat{u})$ (cf. Remark 5). We perform reachability analysis for platoons of N vehicles with $N \in \{1, 4, 9, 20, 50\}$. For jth vehicle in the platoon, we compute the reachable sets for the time frame $t \in [0, 1.5]$ using the closed-loop inclusion functions $\mathsf{F}^{\mathrm{con}}$ and $\mathsf{F}^{\mathrm{act}}$ starting from the initial set $([7.225+0.5(j-1)\cos(\pi/3), 7.275+0.5(j-1)\cos(\pi/3)] \times [5.725 + 0.5(j-1)\sin(\pi/3), 5.775 + 0.5(j-1)\sin(\pi/3)] \times [-0.5, -0.5] \times [-5, -5])$. All integration is performed using Euler integration with a step size of 0.0125. Figure 8 visualizes

[6]No matter the choice of hyper-parameter, the package failed, either 1) citing a maximum number of validation steps reached or 2) timing out.

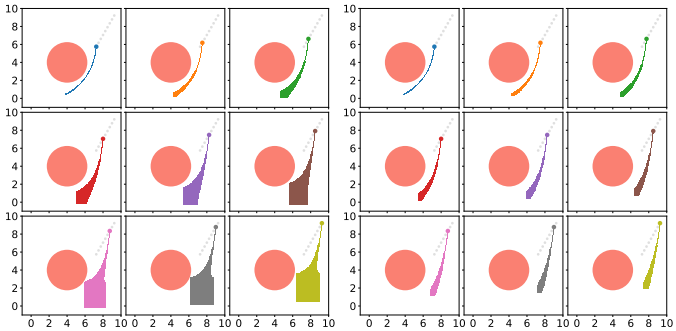

Fig. 8. **Left:** The over-approximation of the reachable sets computed using $\mathbf{F^{con}}$ for each individual unit in the platooning example with $N = 9$ are shown on p_x-p_y planes, with the circular obstacle in salmon. **Right:** The over-approximation of the reachable sets computed using $\mathbf{F^{act}}$ for each individual unit in the platooning example with $N = 9$ are shown on p_x-p_y planes, with the circular obstacle in salmon.

the result for $N = 9$. Runtimes of our approach as well as POLAR [35] and JuliaReach [37] for platoons of varying size are reported in Table IV. Note that the implementations in POLAR and JuliaReach omit the disturbances $w_{\mathbf{d}}^j$ and the softmax σ in equation (26).

Discussion: This experiment demonstrates one of the key features of our interval analysis framework—its scalability to large-scale systems. In general, Figure 8 shows how F^{act} outperforms F^{con} in accuracy of the reachable set over-approximation. However, Table IV shows how F^{con} outperforms F^{act} in terms of scalability with respect to state dimensions. Moreover, both F^{con} and F^{act} demonstrate better runtime scalability than POLAR and JuliaReach.

VIII. Conclusions

We present a framework based on interval analysis for safety verification of neural network controlled systems. The main idea is to embed the closed-loop system into a higher dimensional system whose over-approximation of reachable sets can be easily computed using its extreme trajectory. Using an inclusion function of the open-loop system and interval bounds for neural networks, we proposed an interconnection-based approach and an interaction-based approach for constructing the closed-loop embedding system. We show how these approaches can capture the interactions between the nonlinear plant and the neural network controllers and we provide several numerical experiments comparing them with the state-of-the-art algorithms in the literature.

References

[1] T. Zhang, G. Kahn, S. Levine, and P. Abbeel, "Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, p. 528–535.

[2] A. H. Qureshi, A. Simeonov, M. J. Bency, and M. C. Yip, "Motion planning networks," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 2118–2124.

[3] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *International Conference on Learning Representations*, 2014.

[4] C. Liu, T. Arnon, C. Lazarus, C. Strong, C. Barrett, M. J. Kochenderfer *et al.*, "Algorithms for verifying deep neural networks," *Foundations and Trends® in Optimization*, vol. 4, no. 3-4, pp. 244–404, 2021.

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

[6] I. Mitchell and C. J. Tomlin, "Level set methods for computation in hybrid systems," in *Hybrid Systems: Computation and Control (HSCC)*, 2000, pp. 310–323.

[7] A. B. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis," in *Proceedings of 3rd International Conference on Hybrid Systems: Computation and Control (HSCC)*, 2000, pp. 202–214.

[8] A. Girard, "Reachability of uncertain linear systems using zonotopes," in *Proceedings of the 8th International Conference on Hybrid Systems: Computation and Control (HSCC)*, 2005, p. 291–305.

[9] X. Chen and S. Sankaranarayanan, "Reachability analysis for cyber-physical systems: Are we there yet?" in *NASA Formal Methods: 14th International Symposium, NFM 2022, Pasadena, CA, USA, May 24–27, 2022, Proceedings*. Springer, 2022, pp. 109–130.

[10] L. Jaulin, M. Kieffer, O. Didrit, and É. Walter, *Applied Interval Analysis*. Springer London, 2001.

[11] M. Althoff, O. Stursberg, and M. Buss, "Reachability analysis of linear systems with uncertain parameters and inputs," in *2007 46th IEEE Conference on Decision and Control*, 2007, pp. 726–732.

[12] J. K. Scott and P. I. Barton, "Bounds on the reachable sets of nonlinear control systems," *Automatica*, vol. 49, no. 1, pp. 93–100, 2013.

[13] K. Shen and J. K. Scott, "Rapid and accurate reachability analysis for nonlinear dynamic systems by exploiting model redundancy," *Computers & Chemical Engineering*, vol. 106, pp. 596–608, 2017, eSCAPE-26.

[14] G. A. Enciso, H. L. Smith, and E. D. Sontag, "Nonmonotone systems decomposable into monotone systems with negative feedback," *Journal of Differential Equations*, vol. 224, no. 1, pp. 205–227, 2006.

[15] H. L. Smith, "Global stability for mixed monotone systems," *Journal of Difference Equations and Applications*, vol. 14, no. 10-11, pp. 1159–1164, 2008.

[16] S. Coogan and M. Arcak, "Efficient finite abstraction of mixed monotone systems," in *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, Apr. 2015, pp. 58–67.

[17] S. Coogan, "Mixed monotonicity for reachability and safety in dynamical systems," in *59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 5074–5085.

[18] M. Abate, M. Dutreix, and S. Coogan, "Tight decomposition functions for continuous-time mixed-monotone systems with disturbances," *IEEE Control Systems Letters*, vol. 5, no. 1, pp. 139–144, 2021.

[19] M. Khajenejad and S. Z. Yong, "Tight remainder-form decomposition functions with applications to constrained reachability and guaranteed state estimation," *IEEE Transactions on Automatic Control*, pp. 1–16, 2023.

[20] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient SMT solver for verifying deep neural networks," in *Computer Aided Verification*. Springer International Publishing, 2017, pp. 97–117.

[21] S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana, "Efficient formal safety analysis of neural networks," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, ser. NIPS'18, 2018, p. 6369–6379.

[22] M. Mirman, T. Gehr, and M. Vechev, "Differentiable abstract interpretation for provably robust neural networks," in *International Conference on Machine Learning*, vol. 80, 2018, pp. 3578–3586.

[23] S. Gowal, K. Dvijotham, R. Stanforth, R. Bunel, C. Qin, J. Uesato, R. Arandjelovic, T. A. Mann, and P. Kohli, "Scalable verified training for provably robust image classification," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 4841–4850.

[24] M. Fazlyab, M. Morari, and G. J. Pappas, "Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming," *IEEE Transactions on Automatic Control*, vol. 67, no. 1, pp. 1–15, 2022.

[25] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel, "Efficient neural network robustness certification with general activation functions," in *Advances in Neural Information Processing Systems*, vol. 31, 2018, p. 4944–4953.

[26] S. Wang, H. Zhang, K. Xu, X. Lin, S. Jana, C.-J. Hsieh, and J. Z. Kolter, "Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification," *Advances in Neural Information Processing Systems*, vol. 34, pp. 29 909–29 921, 2021.

[27] H.-D. Tran, X. Yang, D. Manzanas Lopez, P. Musau, L. V. Nguyen, W. Xiang, S. Bak, and T. T. Johnson, "NNV: The neural network verification tool for deep neural networks and learning-enabled cyber-

physical systems," in *Computer Aided Verification*. Springer International Publishing, 2020, pp. 3–17.
[28] W. Xiang, H.-D. Tran, X. Yang, and T. T. Johnson, "Reachable set estimation for neural network control systems: A simulation-guided approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 5, pp. 1821–1830, 2021.
[29] S. Dutta, X. Chen, and S. Sankaranarayanan, "Reachability analysis for neural feedback systems using regressive polynomial rule inference," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, 2019, p. 157–168.
[30] M. Everett, G. Habibi, C. Sun, and J. P. How, "Reachability analysis of neural feedback loops," *IEEE Access*, vol. 9, pp. 163 938–163 953, 2021.
[31] H. Hu, M. Fazlyab, M. Morari, and G. J. Pappas, "Reach-SDP: Reachability analysis of closed-loop systems with neural network controllers via semidefinite programming," in *59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 5929–5934.
[32] C. Sidrane, A. Maleki, A. Irfan, and M. J. Kochenderfer, "OVERT: An algorithm for safety verification of neural network control policies for nonlinear systems," *Journal of Machine Learning Research*, vol. 23, no. 117, pp. 1–45, 2022.
[33] C. Huang, J. Fan, W. Li, X. Chen, and Q. Zhu, "ReachNN: Reachability analysis of neural-network controlled systems," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 5s, pp. 1–22, 2019.
[34] R. Ivanov, T. Carpenter, J. Weimer, R. Alur, G. Pappas, and I. Lee, "Verisig 2.0: Verification of neural network controllers using taylor model preconditioning," in *International Conference on Computer Aided Verification*. Springer, 2021, pp. 249–262.
[35] C. Huang, J. Fan, X. Chen, W. Li, and Q. Zhu, "POLAR: A polynomial arithmetic framework for verifying neural-network controlled systems," in *Automated Technology for Verification and Analysis*. Springer International Publishing, 2022, pp. 414–430.
[36] S. Bogomolov, M. Forets, G. Frehse, K. Potomkin, and C. Schilling, "JuliaReach: a toolbox for set-based reachability," in *Proceedings of the 22nd International Conference on Hybrid Systems: Computation and Control (HSCC)*, 2019, pp. 39–44.
[37] C. Schilling, M. Forets, and S. Guadalupe, "Verification of neural-network control systems by integrating Taylor models and zonotopes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, pp. 8169–8177.
[38] S. Jafarpour, A. Harapanahalli, and S. Coogan, "Interval reachability of nonlinear dynamical systems with neural network controllers," in *Learning for Dynamics and Control Conference*. PMLR, 2023, pp. 12–25.
[39] H. L. Smith, *Monotone Dynamical Systems: An Introduction to the Theory of Competitive and Cooperative Systems*, ser. Mathematical Surveys and Monographs. American Mathematical Society, 1995.
[40] R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to Interval Analysis*. Society for Industrial and Applied Mathematics, 2009.
[41] A. Harapanahalli, S. Jafarpour, and S. Coogan, "A toolbox for fast interval arithmetic in `numpy` with an application to formal verification of neural network controlled system," in *2nd ICML Workshop on Formal Verification of Machine Learning*, 2023. [Online]. Available: https://arxiv.org/abs/2306.15340
[42] C. R. Harris *et al.*, "Array programming with NumPy," *Nature*, vol. 585, p. 357–362, 2020.
[43] J.-L. Gouze and K. P. Hadeler, "Monotone flows and order intervals," *Nonlinear World*, vol. 1, no. 1, pp. 23–34, 1994.
[44] D. Guo and V. Lakshmikantham, "Coupled fixed points of nonlinear operators with applications," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 11, no. 5, pp. 623–632, 1987.
[45] L. Yang, O. Mickelin, and N. Ozay, "On sufficient conditions for mixed monotonicity," *IEEE Transactions on Automatic Control*, vol. 64, no. 12, pp. 5080–5085, 2019.
[46] H. L. Smith, "The discrete dynamics of monotonically decomposable maps," *Journal of Mathematical Biology*, vol. 53, no. 4, pp. 747–758, 2006.
[47] A. N. Michel, L. Hou, and D. Liu, *Stability of dynamical systems: Continuous, discontinuous, and discrete systems*, ser. Systems & Control: Foundations & Applications. Birkhäuser Boston, 2008.
[48] P.-J. Meyer, A. Devonport, and M. Arcak, "TIRA: Toolbox for interval reachability analysis," in *Proceedings of the 22nd International Conference on Hybrid Systems: Computation and Control (HSCC)*, 2019, pp. 224–229.
[49] D. Angeli and E. D. Sontag, "Monotone control systems," *IEEE Transactions on Automatic Control*, vol. 48, no. 10, pp. 1684–1698, 2003.
[50] E. D. Sontag, "Monotone and near-monotone biochemical networks," *Systems and synthetic biology*, vol. 1, no. 2, pp. 59–87, 2007.
[51] A. Harapanahalli, S. Jafarpour, and S. Coogan, "Contraction-guided adaptive partitioning for reachability analysis of neural network controlled systems," in *2023 62nd IEEE Conference on Decision and Control (CDC)*, 2023, pp. 6044–6051.
[52] K. Xu, Z. Shi, H. Zhang, Y. Wang, K.-W. Chang, M. Huang, B. Kailkhura, X. Lin, and C.-J. Hsieh, "Automatic perturbation analysis for scalable certified robustness and beyond," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1129–1141, 2020.
[53] P. Polack, F. Altche, B. d Andrea-Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" in *IEEE Intelligent Vehicles Symposium (IV)*, 2017, pp. 812–818.
[54] T. Entesari, S. Sharifi, and M. Fazlyab, "ReachLipBnB: A branch-and-bound method for reachability analysis of neural autonomous systems using lipschitz bounds," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 1003–1010.
[55] D. M. Lopez *et al.*, "ARCH-COMP22 category report: Artificial intelligence and neural network control systems (AINNCS) for continuous and hybrid systems plants," in *Proceedings of 9th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH22)*, ser. EPiC Series in Computing, vol. 90. EasyChair, 2022, pp. 142–184.

Saber Jafarpour (Member, IEEE) received the Ph.D. degree from Department of Mathematics and Statistics, Queen's University, Kingston, ON, Canada, in 2016. He is currently a Research Assistant Professor at the University of Colorado Boulder, USA in the Department of Electrical, Computer, and Energy Engineering. Prior to joining CU Boulder in 2023, he was a Postdoctoral Researcher with Georgia Institute of Technology and with the Center for Control, Dynamical Systems, and Computation, University of California, Santa Barbara, Santa Barbara. His research interests include analysis and control of networks and learning-based systems.

Akash Harapanahalli (Graduate Student Member, IEEE) received the B.S. degree in Computer Engineering from the Georgia Institute of Technology, Atlanta, GA, USA in 2022. He is currently working towards his Ph.D. in Machine Learning and M.S. in Mathematics. His research interests include the intersection of machine learning and control theory, with a specialty in safety for learning-based control systems.

Samuel Coogan (Senior Member, IEEE) received the B.S. degree in electrical engineering from the Georgia Institute of Technology, Atlanta, GA, USA in 2010 and the M.S. and Ph.D. degrees in electrical engineering from the University of California, Berkeley, Berkeley, CA, USA in 2012 and 2015. He is currently an associate professor and the Demetrius T. Paris Junior Professor at the Georgia Institute of Technology, Atlanta, GA, USA in the School of Electrical and Computer Engineering and the School of Civil and Environmental Engineering. Prior to joining Georgia Tech in 2017, he was an assistant professor at the University of California, Los Angeles from 2015 to 2017. Prof. Coogan received a CAREER Award from the National Science Foundation in 2018, a Young Investigator Award from the Air Force Office of Scientific Research in 2019, and the Donald P Eckman Award from the American Automatic Control Council in 2020. He is a member of SIAM and a senior member of IEEE.