# Interaction-aware Reachability of Neural Network Controlled Systems: A Mixed Monotone Approach

**Saber Jafarpour**

University of Colorado **Boulder**

Georgia Tech

September 28, 2023

# Acknowledgment

Akash Harapanahalli
Georgia Tech

Samuel Coogan
Georgia Tech

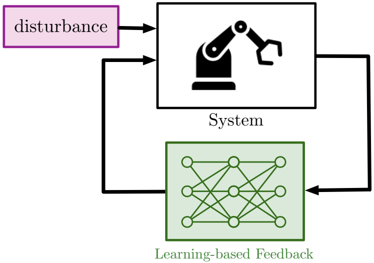
**SJ** and A. Harapanahalli and S. Coogan. Efficient Interaction-Aware Interval Analysis of Neural Network Feedback Loops. arXiv, 2023.

A. Harapanahalli an **SJ** and S. Coogan. A Toolbox for Fast Interval Arithmetic in `numpy` with an Application to Formal Verification of Neural Network Controlled Systems. 2nd ICML workshop on Formal Verification of Machine Learning, 2023

A. Harapanahalli and **SJ** and S. Coogan. Forward Invariance in Neural Network Controlled Systems. arXiv, 2023.

- Neural Networks as controllers in safety-critical applications (examples: autonomous vehicles and mobile robots)

**Goal**: ensure and verify *safety* of the closed-loop system




**Issues with neural network controllers:**

- large # of parameters with nonlinearity
- sensitive wrt to input perturbations
- limited closed-loop safety guarantees

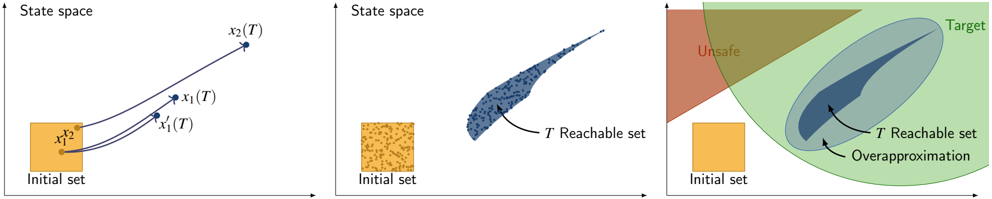
**Challenges**

Rigorous verification and computational efficiency vs. accuracy.

# Safety Verification via Reachability Analysis

## Problem Statement

**System** : $\dot{x} = f(x, w)$ **State** : $x \in \mathbb{R}^n$ **Disturbance** : $w \in \mathcal{W} \subseteq \mathbb{R}^m$




- reachable sets characterize evolution of the system

$$\mathcal{R}^f(t, \mathcal{X}_0) = \{x_w(t) \mid x_w(\cdot) \text{ is a traj of the system for some } w \text{ with } x_0 \in \mathcal{X}_0\}$$

- over-approximation of reachable sets for safety and verification
- reachability of dynamical system is an old problem with several classical approaches

Classical approaches are not scalable to large-scale nonlinear systems

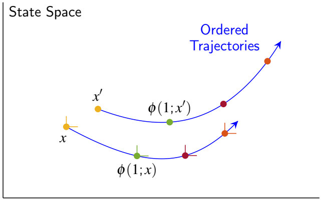
A dynamical system $\dot{x} = f(x, w)$ is monotone[1](with respect to cones $K, C$) if

$$x_u(0) \preceq_K y_w(0) \quad \text{and} \quad u \preceq_C w \quad \Longrightarrow \quad x_u(t) \preceq_K y_w(t) \quad \text{for all time}$$

where $\preceq_K$ ($\preceq_C$) is the partial order with induced by the cone $K$ (cone $C$).

A **polyhedral cone** has the form

$$K = \underbrace{\{y \in \mathbb{R}^n \mid H_K y \geq \mathbb{0}_p\}}_{\text{halfspace rep}} = \underbrace{\{V_K y \mid y \geq \mathbb{0}_p\}}_{\text{vertex rep}}$$




## Monotonicity test

1. $H_K(\frac{\partial f}{\partial x}(x, w) + \alpha(x, w) I_n) V_K \geq \mathbb{0}$ for some $\alpha(x, w)$
2. $H_K \frac{\partial f}{\partial w}(x, w) V_C \geq \mathbb{0}$

[1] D. Angeli and E. Sontag, "Monotone control systems", IEEE TAC, 2003

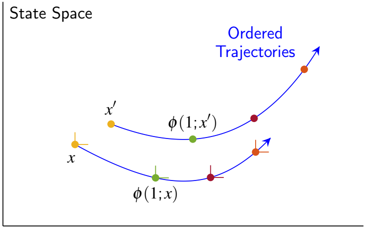
A dynamical system $\dot{x} = f(x,w)$ is monotone[1](with respect to cones $K, C$) if

$$x_u(0) \preceq_K y_w(0) \quad \text{and} \quad u \preceq_C w \quad \implies \quad x_u(t) \preceq_K y_w(t) \quad \text{for all time}$$

where $\preceq_K$ ($\preceq_C$) is the partial order with induced by the cone $K$ (cone $C$).

**Monotonicity test (for the standard cone $\mathbb{R}^n_{\geq 0}$)**

1. $\frac{\partial f}{\partial x}(x,w)$ is Metzler (off-diag $\geq 0$)
2. $\frac{\partial f}{\partial w}(x,w) \geq \mathbb{0}_m$




**In this talk**: monotone system theory for reachability analysis

[1]D. Angeli and E. Sontag, "Monotone control systems", IEEE TAC, 2003

**Theorem (classical result)**

For a monotone system with $\mathcal{W} = [\underline{w}, \overline{w}]$

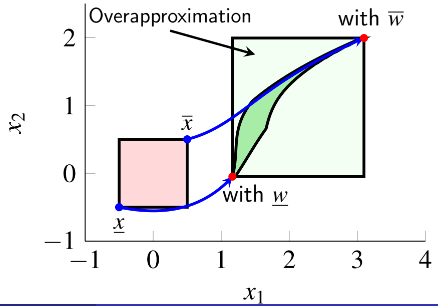
$$\mathcal{R}^f(t, [\underline{x}_0, \overline{x}_0]) \subseteq [x_{\underline{w}}(t), x_{\overline{w}}(t)]$$

where $x_{\underline{w}}(\cdot)$ (resp. $x_{\overline{w}}(\cdot)$) is the trajectory with disturbance $\underline{w}$ (resp. $\overline{w}$) starting at $\underline{x}_0$ (resp. $\overline{x}_0$)

**Example:**

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2^3 - x_1 + w \\ x_1 \end{bmatrix}$$

$$\mathcal{W} = [2.2\ ,\ 2.3] \quad \mathcal{X}_0 = \left[\begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix}, \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}\right]$$

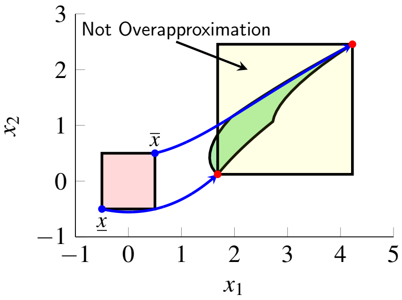


- For non-monotone dynamical systems the extreme trajectories do not provide any over-approximation of reachable sets

**Example:**

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2^3 - x_2 + w \\ x_1 \end{bmatrix}$$

$$\mathcal{W} = [2.2\ ,\ 2.3] \quad \mathcal{X}_0 = \left[\begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix}, \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}\right]$$



- **Key idea:** embed the dynamical system on $\mathbb{R}^n$ into a dynamical system on $\mathbb{R}^{2n}$
- Assume $\mathcal{W} = [\underline{w}, \overline{w}]$ and $\mathcal{X}_0 = [\underline{x}_0, \overline{x}_0]$

**Original system**

$$\dot{x} = f(x, w)$$

**Embedding system**

$$\dot{\underline{x}} = \underline{d}(\underline{x}, \overline{x}, \underline{w}, \overline{w}),$$
$$\dot{\overline{x}} = \overline{d}(\underline{x}, \overline{x}, \underline{w}, \overline{w})$$

$\underline{d}, \overline{d}$ are **decomposition functions** s.t.

1. $f(x, w) = \underline{d}(x, x, w, w)$ for every $x, w$
2. cooperative: $(\underline{x}, \underline{w}) \mapsto \underline{d}(\underline{x}, \overline{x}, \underline{w}, \overline{w})$
3. competitive: $(\overline{x}, \overline{w}) \mapsto \underline{d}(\underline{x}, \overline{x}, \underline{w}, \overline{w})$
4. the same properties for $\overline{d}$

The embedding system is a monotone dynamical system on $\mathbb{R}^{2n}$ with respect to the **southeast** partial order $\leq_{\mathrm{SE}}$:

$$\begin{bmatrix} x \\ \widehat{x} \end{bmatrix} \leq_{\mathrm{SE}} \begin{bmatrix} y \\ \widehat{y} \end{bmatrix} \quad \Longleftrightarrow \quad x \leq y \quad \text{and} \quad \widehat{y} \leq \widehat{x}$$

In terms of cones, $\leq_{\mathrm{SE}}$ is induced by the cone $\mathbb{R}^n_{\geq 0} \times -\mathbb{R}^n_{\geq 0}$.

- $f$ locally Lipschitz $\implies$ a decomposition function exists

The best (tightest) decomposition function is given by

$$\underline{d}_i(\underline{x}, \overline{x}, \underline{w}, \overline{w}) = \min_{\substack{z \in [\underline{x}, \overline{x}], z_i = \underline{x}_i \\ u \in [\underline{w}, \overline{w}]}} f_i(z, u), \qquad \overline{d}_i(\underline{x}, \overline{x}, \underline{w}, \overline{w}) = \max_{\substack{z \in [\underline{x}, \overline{x}], z_i = \overline{x}_i \\ u \in [\underline{w}, \overline{w}]}} f_i(z, u)$$

**A short (and incomplete) history:**

J-L. Gouze and L. P. Hadeler. Monotone flows and order intervals. Nonlinear World, 1994

G. Enciso, H. Smith, and E. Sontag. Nonmonotone systems decomposable into monotone systems with negative feedback . Journal of Differential Equations, 2006.

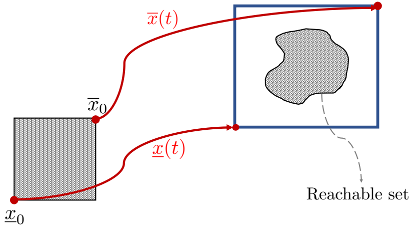
H. Smith. Global stability for mixed monotone systems. Journal of Difference Equations and Applications, 2008

**Theorem**[2]

Assume $\mathcal{W} = [\underline{w}, \overline{w}]$ and $\mathcal{X}_0 = [\underline{x}_0, \overline{x}_0]$ and

$$\begin{aligned} \dot{\underline{x}} &= \underline{d}(\underline{x}, \overline{x}, \underline{w}, \overline{w}), \qquad & \underline{x}(0) &= \underline{x}_0 \\ \dot{\overline{x}} &= \overline{d}(\overline{x}, \underline{x}, \overline{w}, \underline{w}), \qquad & \overline{x}(0) &= \overline{x}_0 \end{aligned}$$

Then $\mathcal{R}^f(t, \mathcal{X}_0) \subseteq [\underline{x}(t), \overline{x}(t)]$




**(Scalable)** a single trajectory of embedding system provides **lower bound** $(\underline{x})$ and **upper bound** $(\overline{x})$ for the trajectories of the original system.

[2] Coogan and Arcak, "Efficient finite abstraction of mixed monotone systems", HSCC, 2015.

**Original System:**

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} {\color{red}x_2^3}{\color{blue}-x_2} + {\color{red}w} \\ {\color{red}x_1} \end{bmatrix}$$

$$\mathcal{W} = [2.2\ ,\ 2.3] \quad \mathcal{X}_0 = \left[\begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix}, \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}\right]$$

red $=$ cooperative, blue $=$ competitive

**Embedding System:**

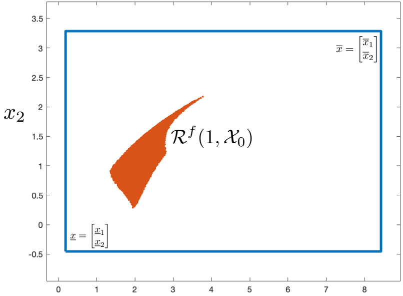
$$\frac{d}{dt}\begin{bmatrix} \underline{x}_1 \\ \underline{x}_2 \\ \overline{x}_1 \\ \overline{x}_2 \end{bmatrix} = \begin{bmatrix} \underline{x}_2^3 - \overline{x}_2 + \underline{w} \\ \underline{x}_1 \\ \overline{x}_2^3 - \underline{x}_2 + \overline{w} \\ \overline{x}_1 \end{bmatrix} \qquad \begin{bmatrix} \underline{w} \\ \overline{w} \end{bmatrix} = \begin{bmatrix} 2.2 \\ 2.3 \end{bmatrix}$$

$$\begin{bmatrix} \underline{x}_1(0) \\ \underline{x}_2(0) \end{bmatrix} = \begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix} \qquad \begin{bmatrix} \overline{x}_1(0) \\ \overline{x}_2(0) \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

**Decomposition function**

$$\underline{d}(\underline{x}, \overline{x}, \underline{w}, \overline{w}) = \begin{bmatrix} \underline{x}_2^3 + \underline{w} \\ \underline{x}_1 \end{bmatrix} + \begin{bmatrix} -\overline{x}_2 \\ 0 \end{bmatrix}$$

$$\overline{d}(\underline{x}, \overline{x}, \underline{w}, \overline{w}) = \begin{bmatrix} \overline{x}_2^3 + \overline{w} \\ \overline{x}_1 \end{bmatrix} + \begin{bmatrix} -\underline{x}_2 \\ 0 \end{bmatrix}$$

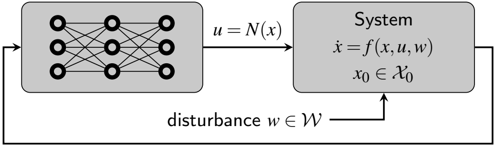


Given the open-loop nonlinear system with a neural network controller

$$\dot{x} = f(x, u, w),$$
$$u = N(x),$$

study reachability of the closed-loop system

$$\dot{x} = f(x, N(x), w) := f^c(x, w)$$




**Challenge:** find a decomposition function for closed-loop system

**Key observation**: **Interval bounds** for neural networks combines nicely with **mixed monotone theory** for the open-loop system!

- Interval bounds for NN using verification algorithms (CROWN, LipSDP, IBP, etc)

**Question**: How to capture the interaction between NN and the system

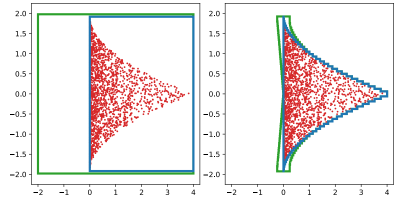
# Decomposition Functions for Systems

## A Jacobian-based Approach

**Jacobian-based**: $\dot{x} = f(x,w)$ such that $\frac{\partial f}{\partial x} \in [\underline{J}_{[\underline{x},\overline{x}]}, \overline{J}_{[\underline{x},\overline{x}]}]$ and $\frac{\partial f}{\partial w} \in [\underline{J}_{[\underline{w},\overline{w}]}, \overline{J}_{[\underline{w},\overline{w}]}]$, then

$$\begin{bmatrix} \underline{d}(\underline{x},\overline{x},\underline{w},\overline{w}) \\ \overline{d}(\underline{x},\overline{x},\underline{w},\overline{w}) \end{bmatrix} = \begin{bmatrix} -[\underline{J}_{[\underline{x},\overline{x}]}]^- & [\underline{J}_{[\underline{x},\overline{x}]}]^- \\ -[\overline{J}_{[\underline{x},\overline{x}]}]^+ & [\overline{J}_{[\underline{x},\overline{x}]}]^+ \end{bmatrix} \begin{bmatrix} \underline{x} \\ \overline{x} \end{bmatrix} + \begin{bmatrix} -[\underline{J}_{[\underline{w},\overline{w}]}]^- & [\underline{J}_{[\underline{w},\overline{w}]}]^- \\ -[\overline{J}_{[\underline{w},\overline{w}]}]^+ & [\overline{J}_{[\underline{w},\overline{w}]}]^+ \end{bmatrix} \begin{bmatrix} \underline{w} \\ \overline{w} \end{bmatrix} + \begin{bmatrix} f(\underline{x},\underline{w}) \\ f(\underline{x},\underline{w}) \end{bmatrix}$$

- Interval arithmetic allows computing Jacobian bounds efficiently using *inclusion functions*.
- `npinterval`[3]: Toolbox that implements intervals as native data-type in `numpy`.




$g(x_1,x_2) = [(x_1+x_2)^2, 4\sin((x_1-x_2)/4)]^T$

vs.

$g(x_1,x_2) =$
$[x_2^2 + 2x_1x_2 + x_2^2, 4\sin(x_1/4)\cos(x_2/4) - 4\cos(x_1/4)\sin(x_2/4)]^T$

[3] Harapanahalli, Jafarpour, Coogan. "A Toolbox for Fast Interval Arithmetic in numpy with an Application to Formal Verification of Neural Network Controlled Systems", 2nd WFVML, ICML, 2023

**Input-output bounds:** Given a neural network controller $u = N(x)$

$$\underline{u}_{[\underline{x},\overline{x}]} \leq N(x) \leq \overline{u}_{[\underline{x},\overline{x}]}, \qquad \text{for all } x \in [\underline{x},\overline{x}]$$

**Functional bounds:** Given a neural network controller $u = N(x)$

$$\underline{N}_{[\underline{x},\overline{x}]}(x) \leq N(x) \leq \overline{N}_{[\underline{x},\overline{x}]}(x), \qquad \text{for all } x \in [\underline{x},\overline{x}]$$

- Example: CROWN[4] can provide both input-output and functional bounds.

CROWN functional bounds:

$$\begin{aligned}\underline{N}_{[\underline{x},\overline{x}]}(x) &= \underline{A}_{[\underline{x},\overline{x}]}x + \underline{b}_{[\underline{x},\overline{x}]},\\ \overline{N}_{[\underline{x},\overline{x}]}(x) &= \overline{A}_{[\underline{x},\overline{x}]}x + \overline{b}_{[\underline{x},\overline{x}]}\end{aligned}$$

CROWN input-output bounds:

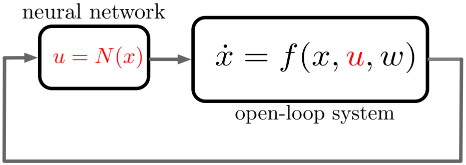
$$\begin{aligned}\underline{u}_{[\underline{x},\overline{x}]} &= \underline{A}^{+}_{[\underline{x},\overline{x}]}\overline{x} + \overline{A}^{-}_{[\underline{x},\overline{x}]}\underline{x} + \underline{b}_{[\underline{x},\overline{x}]},\\ \overline{u}_{[\underline{x},\overline{x}]} &= \overline{A}^{+}_{[\underline{x},\overline{x}]}\overline{x} + \underline{A}^{-}_{[\underline{x},\overline{x}]}\underline{x} + \overline{b}_{[\underline{x},\overline{x}]}\end{aligned}$$

[4] Zhang, Weng, Chen, Hsieh, Daniel. "Efficient neural network robustness certification with general activation functions." NeurIPS, 2018.

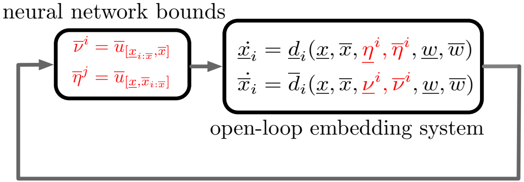
# Approach #1: Interconnection-based Approach

## A pictorial explanation

**Original system:**

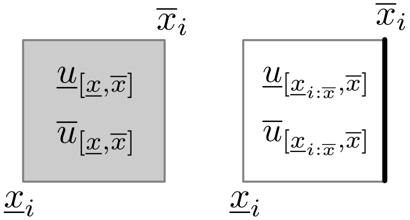



**Embedding system:**




How does the interconnection-based approach work?

- closed-loop embedding system = interconnection of NN interval bounds + open-loop embedding system
- NN bounds are evaluated on each edge instead of the whole box, i.e., we use $\underline{u}_{[\underline{x}_{i:\overline{x}}, \overline{x}]}$ and $\overline{u}_{[\underline{x}_{i:\overline{x}}, \overline{x}]}$ instead of $\underline{u}_{[\underline{x}, \overline{x}]}$ and $\overline{u}_{[\underline{x}, \overline{x}]}$.



## Theorem[5]

1. **Decomposition function** $\underline{d}, \overline{d}$ for the open-loop system $\dot{x} = f(x,u,w)$
2. **Interval input-output bounds** $\underline{u}_{[\underline{x},\overline{x}]}, \overline{u}_{[\underline{x},\overline{x}]}$ for the neural network controller $u = N(x)$,

Then

$$\begin{aligned} \underline{d}^c_i(\underline{x},\overline{x},\underline{w},\overline{w}) &= \underline{d}_i(\underline{x},\overline{x},\underline{\eta}^i,\overline{\eta}^i,\underline{w},\overline{w}) \\ \overline{d}^c_i(\underline{x},\overline{x},\underline{w},\overline{w}) &= \overline{d}_i(\underline{x},\overline{x},\underline{\nu}^i,\overline{\nu}^i,\underline{w},\overline{w}) \end{aligned}$$

where

$$\underline{\eta}^i = \underline{u}_{[\underline{x},\overline{x}_{i:\underline{x}}]} \quad \overline{\eta}^i = \overline{u}_{[\underline{x},\overline{x}_{i:\underline{x}}]}, \qquad \underline{\nu}^i = \underline{u}_{[\underline{x}_{i:\overline{x}},\overline{x}]} \quad \overline{\nu}^i = \overline{u}_{[\underline{x}_{i:\overline{x}},\overline{x}]},$$

is a **decomposition function for the closed-loop system** where $v_{i:w}$ is the vector $v$ with $i$th component replaced with $i$th component of $w$.

[5] Jafarpour, Harapanahalli, Coogan. "Efficient Interaction-aware Interval Reachability of Neural Network Feedback Loops", arXiv, 2003

**Original system:**

$$\dot{x} = f(x, \color{red}{N(x)}, w)$$

closed-loop system

**Embedding system:**

$$\begin{bmatrix} \dot{\underline{x}} \\ \dot{\overline{x}} \end{bmatrix} = \begin{bmatrix} [\color{red}{\underline{H}}]^+ - \underline{J}_{[\underline{x},\overline{x}]} & [\color{red}{\underline{H}}]^- \\ [\color{red}{\overline{H}}]^+ - \overline{J}_{[\underline{x},\overline{x}]} & [\color{red}{\overline{H}}]^- \end{bmatrix} \begin{bmatrix} \underline{x} \\ \overline{x} \end{bmatrix} + \begin{bmatrix} -[\underline{J}_{[\underline{w},\overline{w}]}]^- & [\underline{J}_{[\underline{w},\overline{w}]}]^+ \\ -[\overline{J}_{[\underline{w},\overline{w}]}]^- & [\overline{J}_{[\underline{w},\overline{w}]}]^+ \end{bmatrix} \begin{bmatrix} \underline{w} \\ \overline{w} \end{bmatrix} + Q$$

closed-loop embedding system

How does the interaction-based approach work?

- Closed-loop decomposition function = Jacobian based for $f(x, N(x), w)$.
- Neural Network affine functional bounds
  $\underline{N}_{[\underline{x},\overline{x}]} = \underline{A}_{[\underline{x},\overline{x}]} x + \underline{b}_{[\underline{x},\overline{x}]}$,
  $\overline{N}_{[\underline{x},\overline{x}]} = \overline{A}_{[\underline{x},\overline{x}]} x + \overline{b}_{[\underline{x},\overline{x}]}$
  are used to compute the interactions.

$$\color{red}{\underline{H}} = \underline{J}_{[\underline{x},\overline{x}]} + [\underline{J}_{[\underline{u},\overline{u}]}]^+ \underline{A}_{[\underline{x},\overline{x}]} + [\underline{J}_{[\underline{u},\overline{u}]}]^- \overline{A}_{[\underline{x},\overline{x}]}$$

$$\color{red}{\overline{H}} = \overline{J}_{[\underline{x},\overline{x}]} + [\underline{J}_{[\underline{u},\overline{u}]}]^+ \overline{A}_{[\underline{x},\overline{x}]} + [\underline{J}_{[\underline{u},\overline{u}]}]^- \underline{A}_{[\underline{x},\overline{x}]}$$

## Theorem[6]

Let $\frac{\partial f}{\partial x} \in [\underline{J}_{[\underline{x},\overline{x}]}, \overline{J}_{[\underline{x},\overline{x}]}]$, $\frac{\partial f}{\partial u} \in [\underline{J}_{[\underline{u},\overline{u}]}, \overline{J}_{[\underline{u},\overline{u}]}]$, and $\frac{\partial f}{\partial w} \in [\underline{J}_{[\underline{w},\overline{w}]}, \overline{J}_{[\underline{w},\overline{w}]}]$. Then

$$\begin{bmatrix} \underline{d}_i^c(\underline{x},\overline{x},\underline{w},\overline{w}) \\ \overline{d}_i^c(\underline{x},\overline{x},\underline{w},\overline{w}) \end{bmatrix} = \begin{bmatrix} [\underline{H}]^+ - \underline{J}_{[\underline{x},\overline{x}]} & [\underline{H}]^- \\ [\overline{H}]^+ - \overline{J}_{[\underline{x},\overline{x}]} & [\overline{H}]^- \end{bmatrix} \begin{bmatrix} \underline{x} \\ \overline{x} \end{bmatrix} + \begin{bmatrix} -[\underline{J}_{[\underline{w},\overline{w}]}]^- & [\underline{J}_{[\underline{w},\overline{w}]}]^+ \\ -[\overline{J}_{[\underline{w},\overline{w}]}]^- & [\overline{J}_{[\underline{w},\overline{w}]}]^+ \end{bmatrix} \begin{bmatrix} \underline{w} \\ \overline{w} \end{bmatrix} + Q$$

where

$$\begin{aligned} \underline{H} &= \underline{J}_{[\underline{x},\overline{x}]} + [\underline{J}_{[\underline{u},\overline{u}]}]^+ \underline{A}_{[\underline{x},\overline{x}]} + [\underline{J}_{[\underline{u},\overline{u}]}]^- \overline{A}_{[\underline{x},\overline{x}]} \\ \overline{H} &= \overline{J}_{[\underline{x},\overline{x}]} + [\underline{J}_{[\underline{u},\overline{u}]}]^+ \overline{A}_{[\underline{x},\overline{x}]} + [\underline{J}_{[\underline{u},\overline{u}]}]^- \underline{A}_{[\underline{x},\overline{x}]} \end{aligned}$$

is a **decomposition function for the closed-loop system**.

[6] Jafarpour, Harapanahalli, Coogan. "Efficient Interaction-aware Interval Reachability of Neural Network Feedback Loops", arXiv, 2003

# Case Study: Bicycle Model

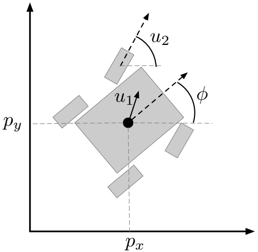
## Design of the neural network

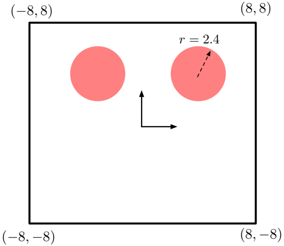
**Dynamics of bicycle**

$$\dot{p_x} = v\cos(\phi + \beta(u_2)) \qquad \dot{\phi} = \frac{v}{\ell_r}\sin(\beta(u_2))$$

$$\dot{p_y} = v\sin(\phi + \beta(u_2)) \qquad \dot{v} = u_1$$

$$\beta(u_2) = \arctan\left(\frac{l_r}{l_f + l_r}\tan(u_2)\right)$$



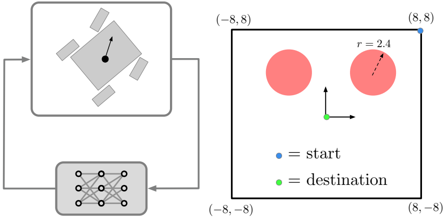

**Goal:** steer the bicycle to the origin avoiding the obstacles

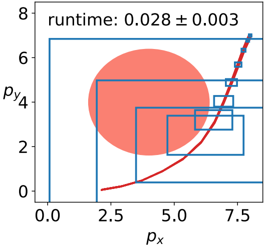
- **offline controller:** MPC with hard constraint to avoid the obstacles
- run MPC for $65000$ randomly chosen initial condition ($20$ sample per trajectory)
- train a feedforward neural network $4 \mapsto 100 \mapsto 100 \mapsto 2$ with this data

- start from $(8, 8)$ toward $(0, 0)$
- $\mathcal{X}_0 = [\underline{x}_0, \overline{x}_0]$ with

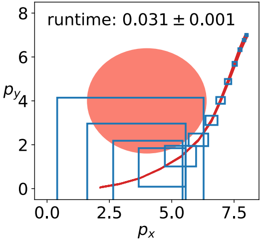
$$\underline{x}_0 = \begin{pmatrix} 7.95 & 6.95 & -\frac{2\pi}{3} - 0.01 & 1.99 \end{pmatrix}^\top$$
$$\overline{x}_0 = \begin{pmatrix} 8.05 & 7.05 & -\frac{2\pi}{3} + 0.01 & 2.01 \end{pmatrix}^\top$$

- CROWN for verification of neural network

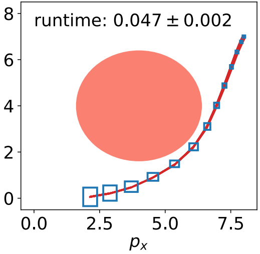



naive combination




interconnection




interaction

# Conclusions

- reachability using mixed monotone theory
- mixed monotone theory for reachability of NN controlled systems
- two methods for capturing the interaction between system and NN controller
  1. interconnection-based approach
  2. interaction-based approach

**Future directions:**

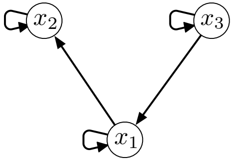
- forward invariance of systems with NN controllers
- design of suitable correction actions
- ensuring safety in the training of NN

- **Metzler/non-Metzler** decomposition: $A = \lceil A \rceil^{\mathrm{Mzl}} + \lfloor A \rfloor^{\mathrm{Mzl}}$
- Example: $A = \begin{bmatrix} 2 & 0 & -1 \\ 1 & -3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \implies \lceil A \rceil^{\mathrm{Mzl}} = \begin{bmatrix} 2 & 0 & 0 \\ 1 & -3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \lfloor A \rfloor^{\mathrm{Mzl}} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

## Linear systems

**Original system**

$$\dot{x} = Ax + Bw$$




**Embedding system**

$$\dot{\underline{x}} = \lceil A \rceil^{\mathrm{Mzl}} \underline{x} + \lfloor A \rfloor^{\mathrm{Mzl}} \overline{x} + B^{+} \underline{w} + B^{-} \overline{w}$$
$$\dot{\overline{x}} = \lceil A \rceil^{\mathrm{Mzl}} \overline{x} + \lfloor A \rfloor^{\mathrm{Mzl}} \underline{x} + B^{+} \overline{w} + B^{-} \underline{w}$$