Reachability Analysis of Neural Network Dynamical Systems

Mahyar Fazlyab

Johns Hopkins University

mahyarfazlyab@jhu.edu

Workshop on Data-Driven Verification and Control of Cyber-Physical Systems

IFAC World Congress 2023

July 9, 2023

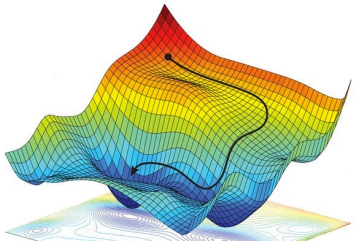
JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Deep Learning in Safety Critical Domains

computation (hardware)

big data

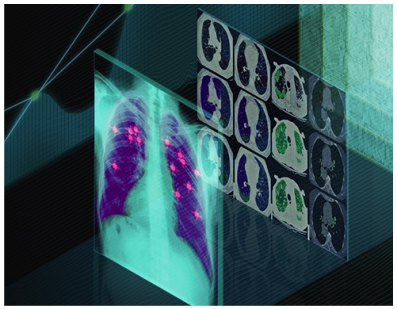
algorithms (software)



deep learning

autonomous systems

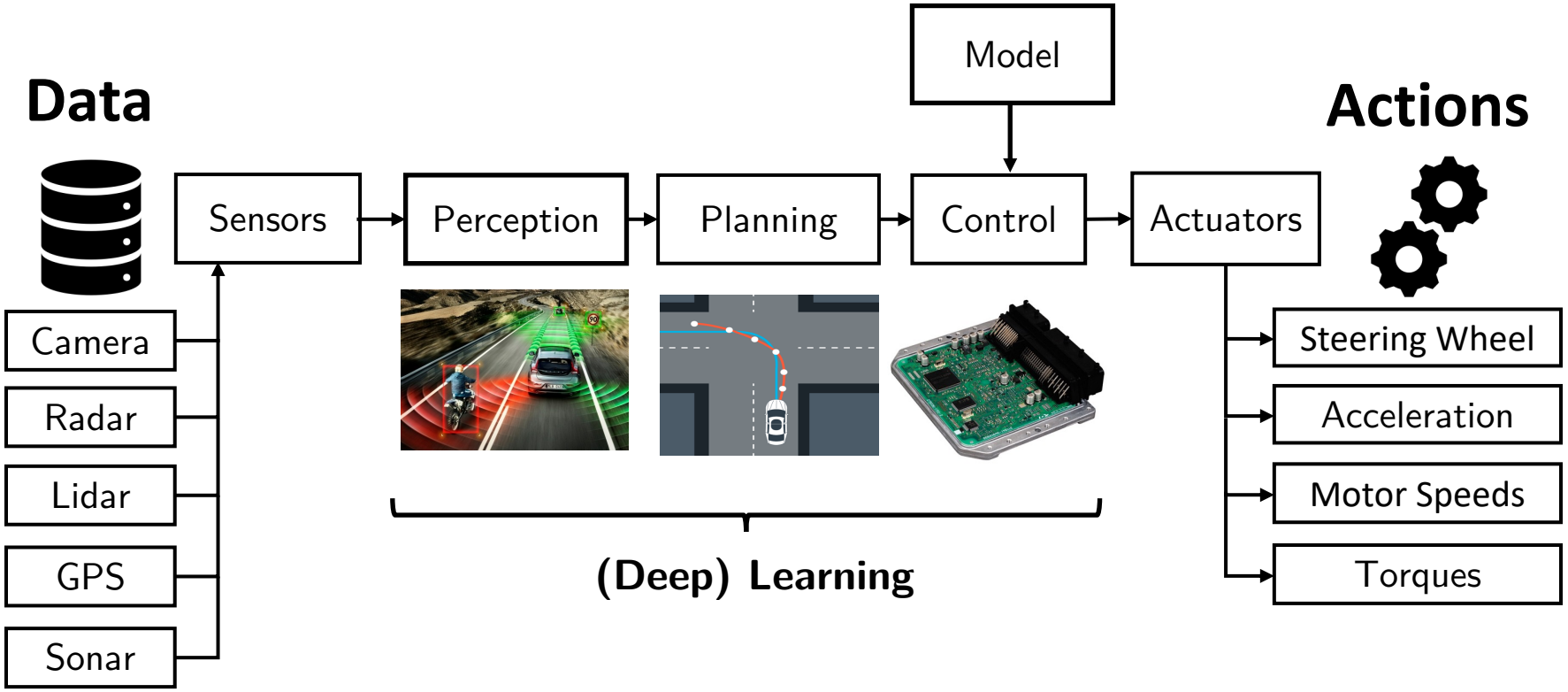
automated healthcare



conversational AI

critical infrastructure

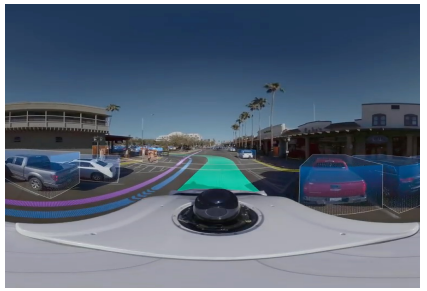
Deep Learning in Autonomous Systems



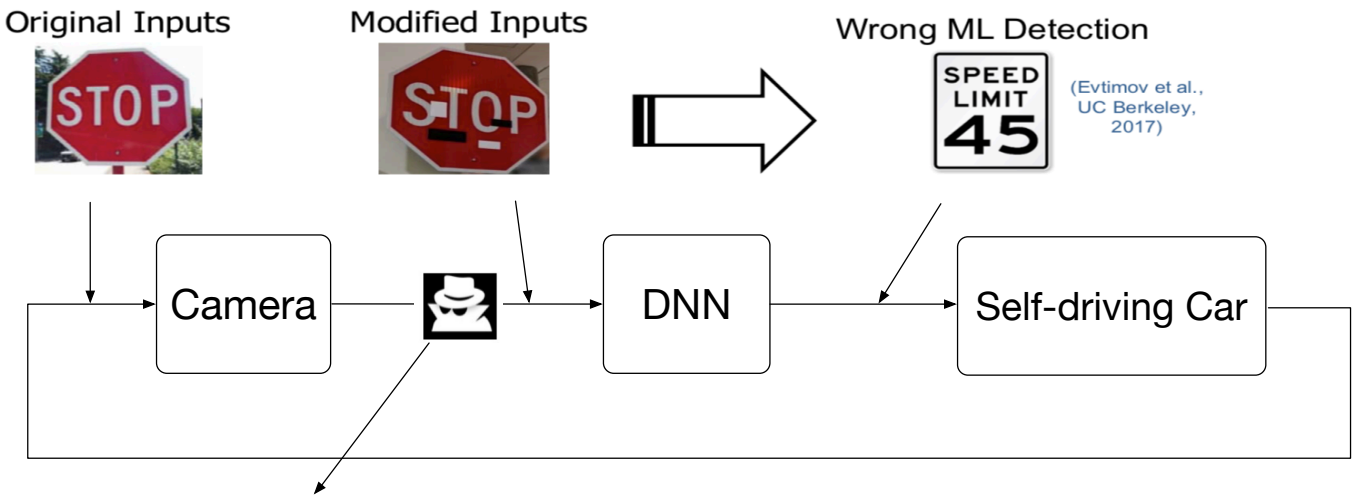
Goldberg lab

DeepMind

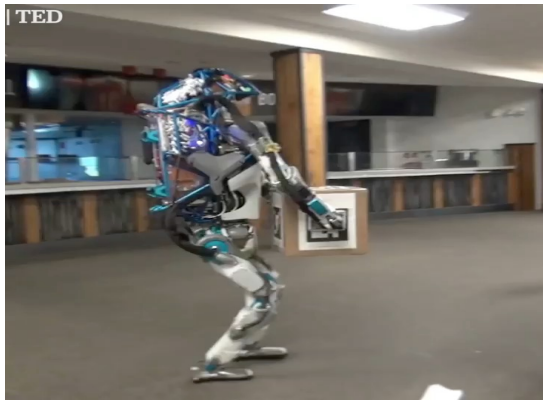
Waymo

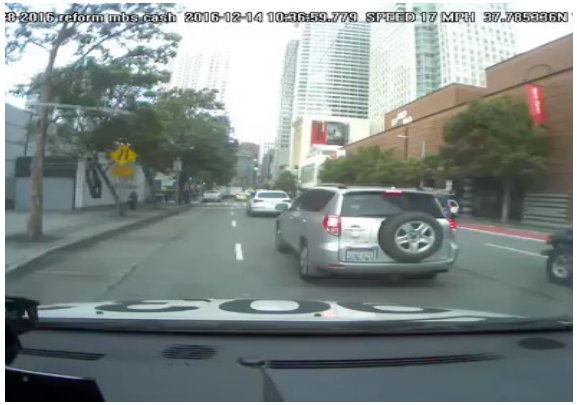


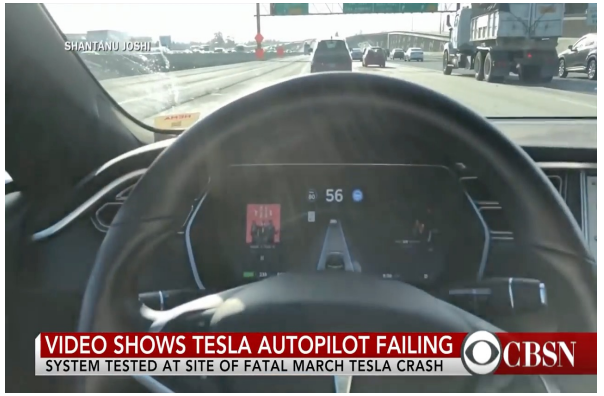
Current Limitation: Lack of Robustness



- Adversarial attacks
- Camera noise



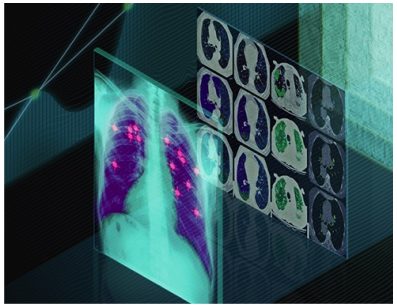




Research Overview

autonomous systems

automated healthcare

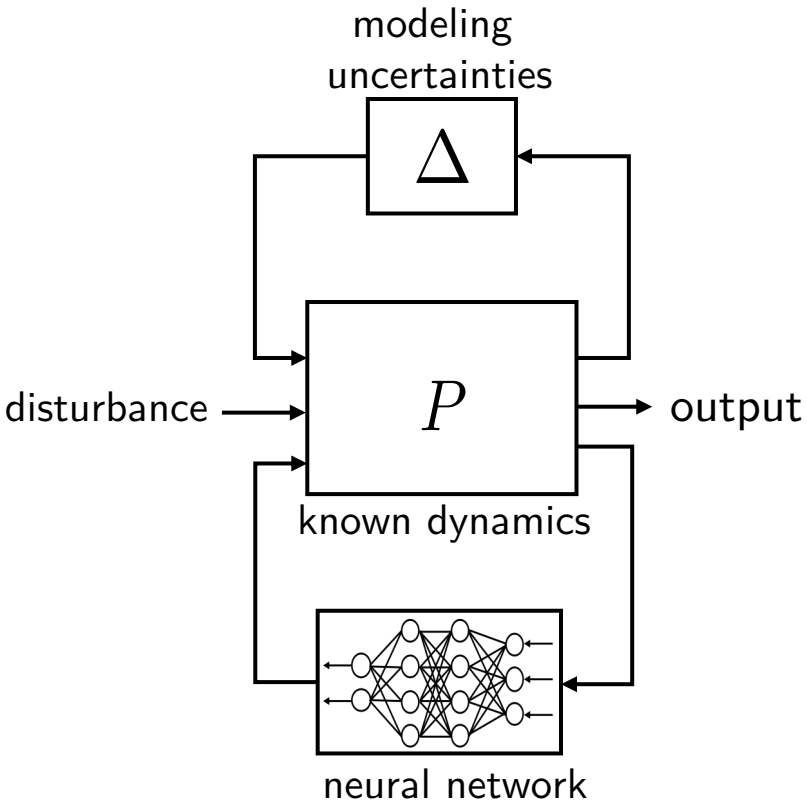


conversational AI

critical infrastructure

Adoption of deep learning in safety-critical systems requires strict guarantees of robustness, stability and safety

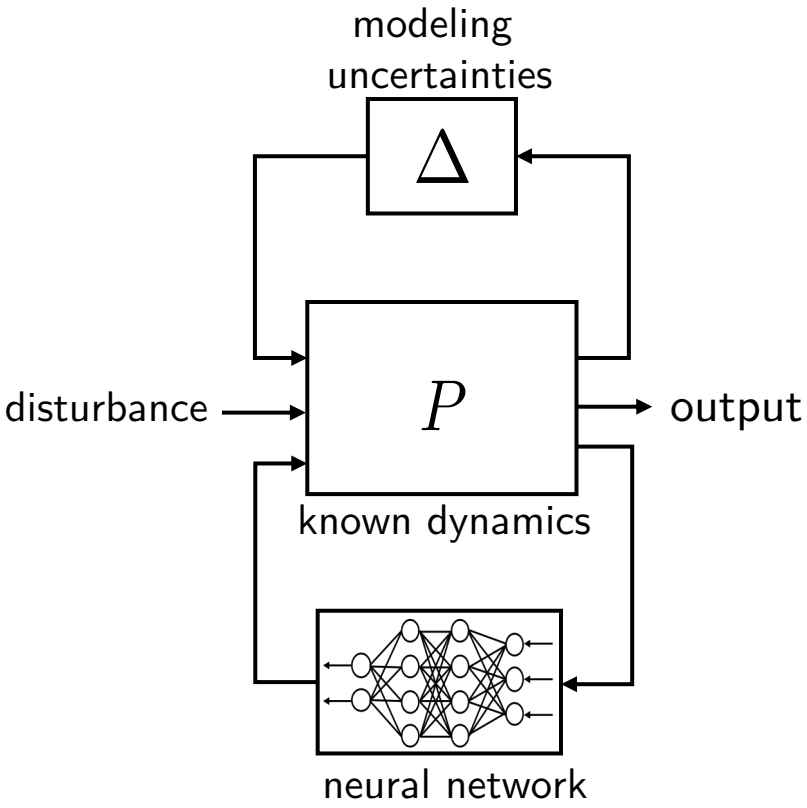
Neural Network Driven Dynamical Systems



Goal: verify a *property* about the closed-loop system for all admissible disturbances and modeling uncertainties

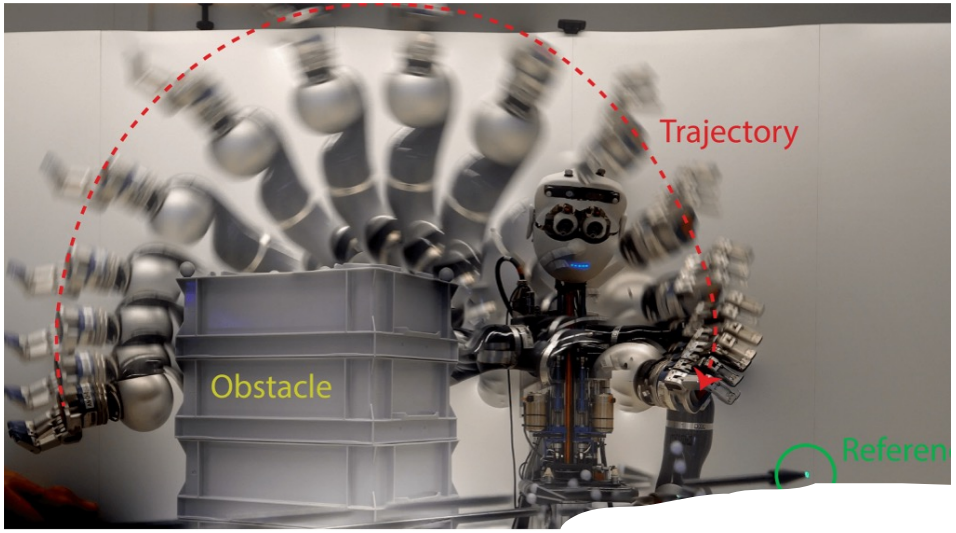
- stability
- robustness
- safety

Neural Network Driven Dynamical Systems



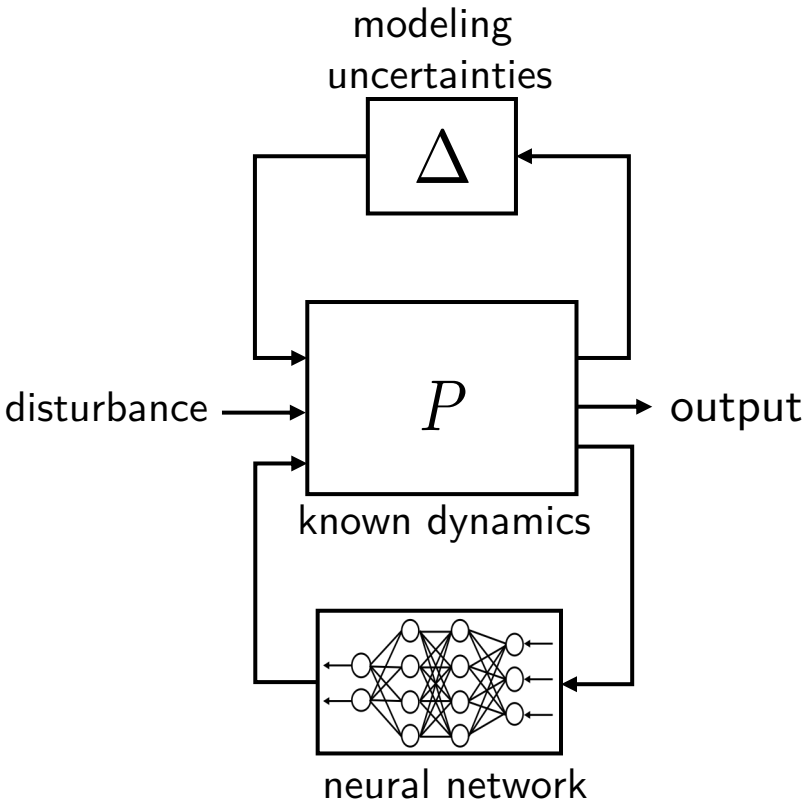
Goal: verify a *property* about the closed-loop system for all admissible disturbances and modeling uncertainties

- stability
- robustness
- **safety**



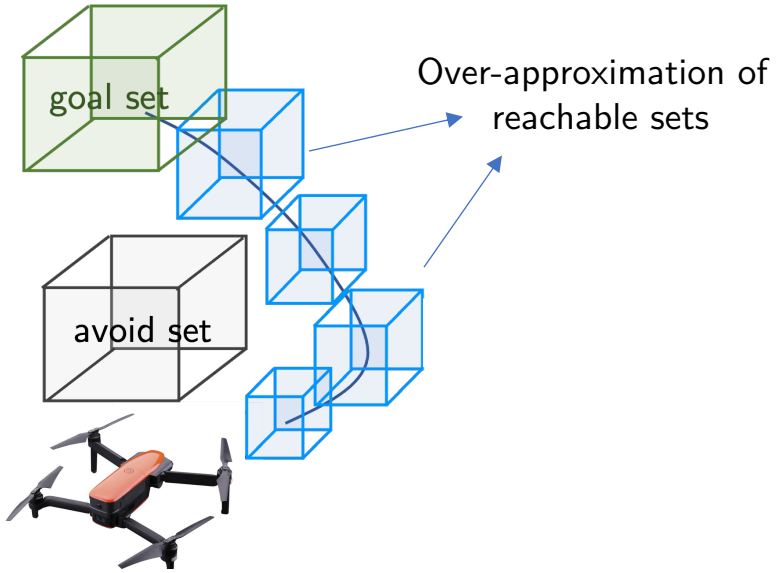
Nubert, Julian, et al. "Safe and fast tracking on a robot manipulator: Robust mpc and neural network control." IEEE Robotics and Automation Letters 5.2 (2020): 3050-3057.

Neural Network Driven Dynamical Systems

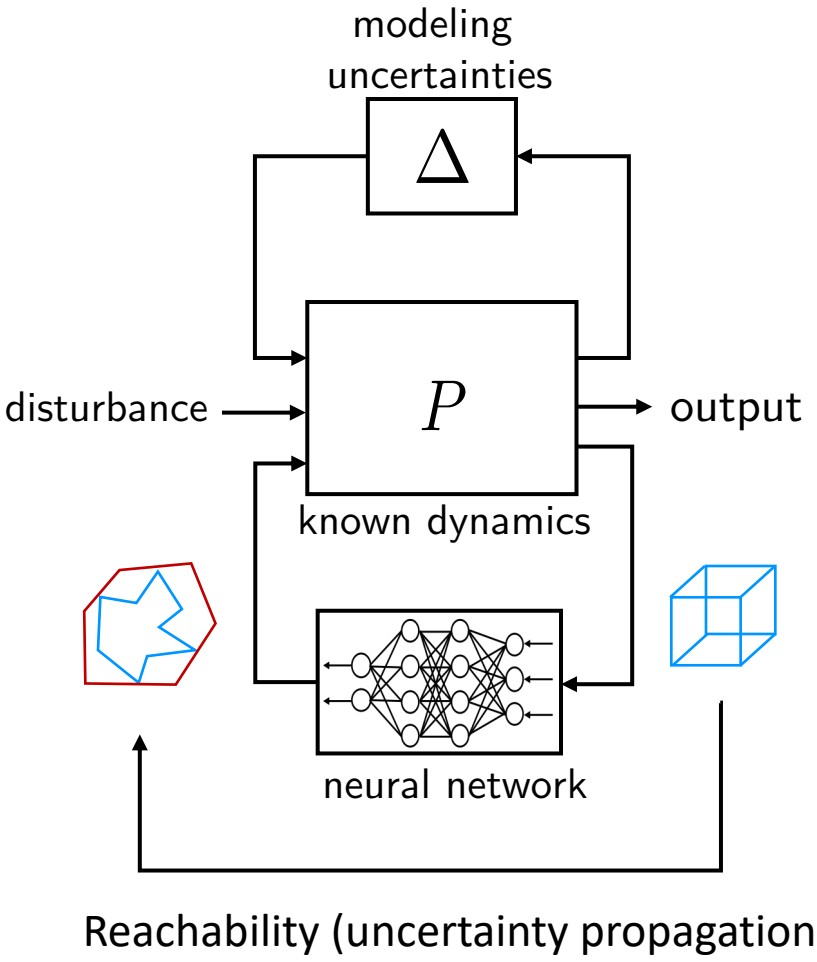


Goal: verify a *property* about the closed-loop system for all admissible disturbances and modeling uncertainties

- stability
- robustness
- **safety**



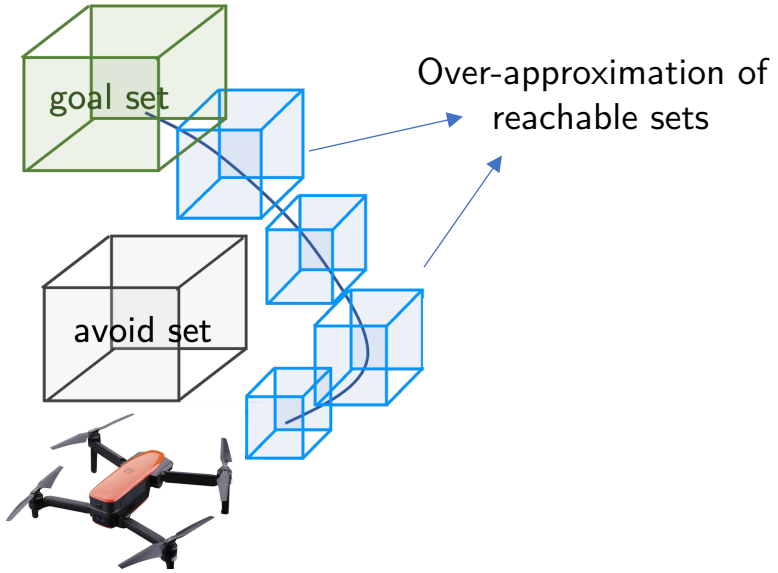
Neural Network Driven Dynamical Systems



Reachability (uncertainty propagation)

Goal: verify a *property* about the closed-loop system for all admissible disturbances and modeling uncertainties

- stability
- robustness
- **safety**

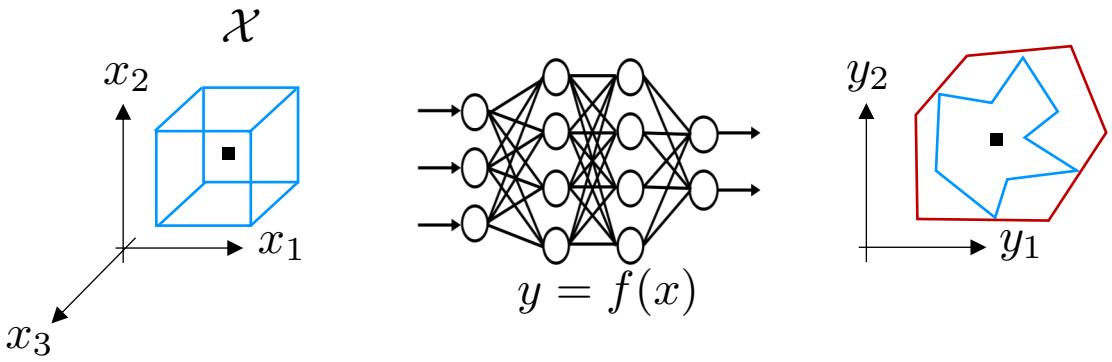


Part I

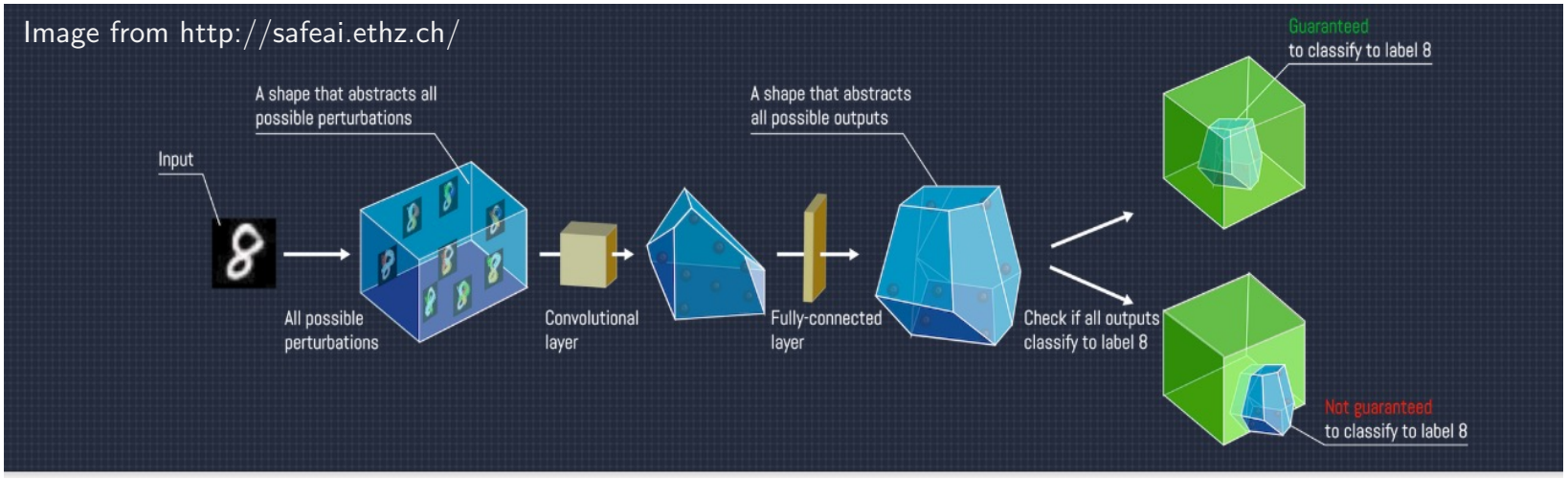
A Robust Control Perspective to Neural Network Verification

Neural Network Verification

- Given a set of inputs, compute/localize the output reachable set



- **Example:** robustness certification in NN-classifiers



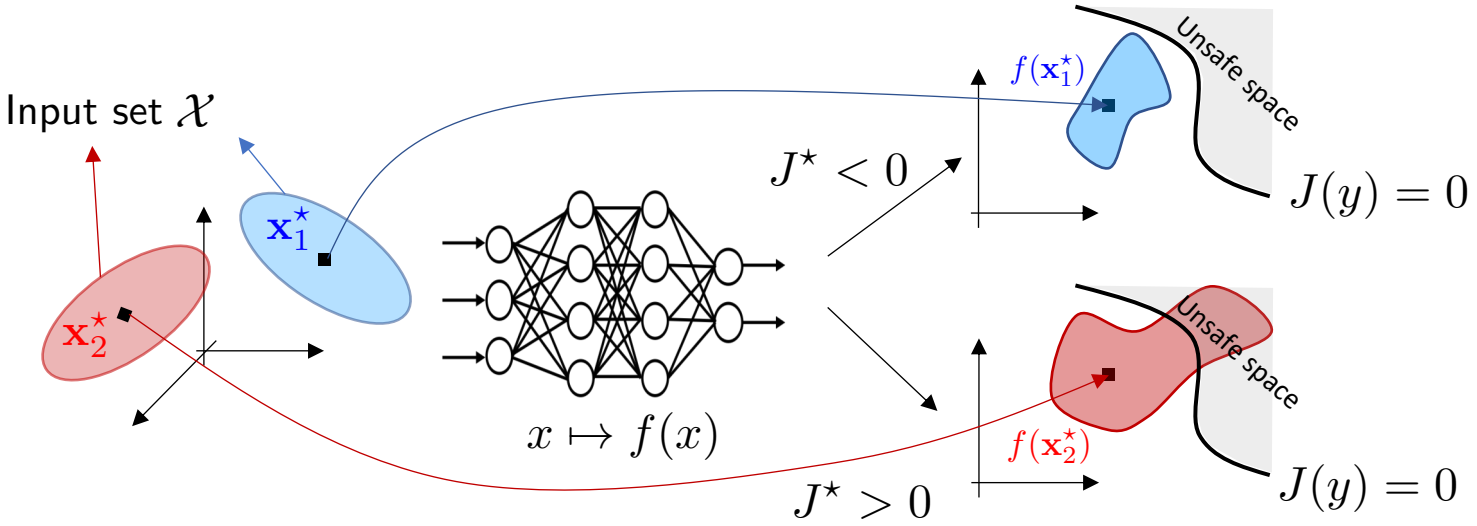
Neural Network Verification

- Verification = constraint satisfaction problem

$$J(f(x)) \leq 0 \quad \forall x \in \mathcal{X}$$

- Equivalent non-convex optimization problem

$$J^\star = \sup\{J(y) \mid y = f(x),\ x \in \mathcal{X}\}$$



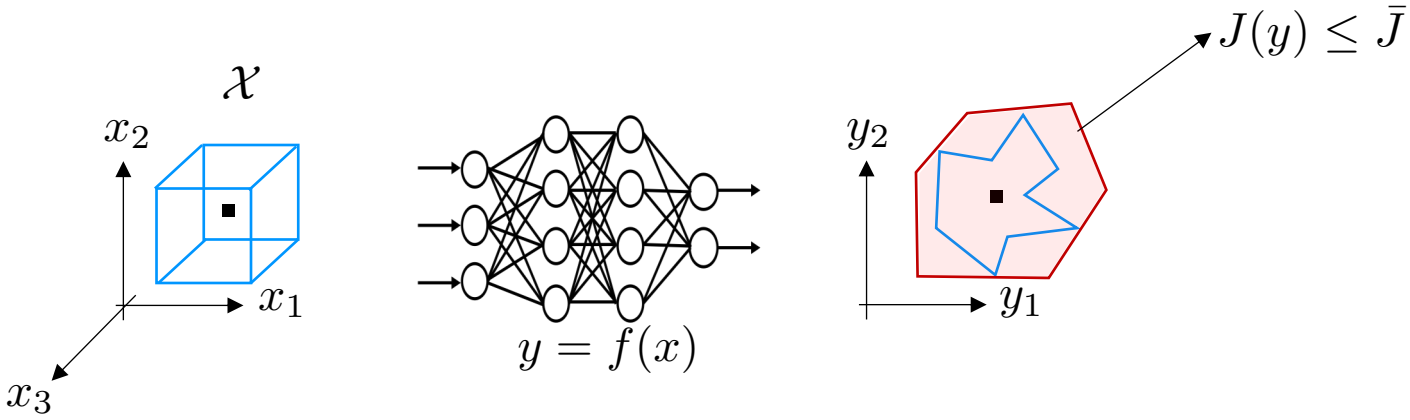
Neural Network Verification

- Non-convex optimization problem
  - MILP for ReLU activation functions

$$J^\star = \sup\{J(y) \mid y = f(x),\ x \in \mathcal{X}\}$$

- Convex relaxations: find guaranteed upper bounds in polynomial time

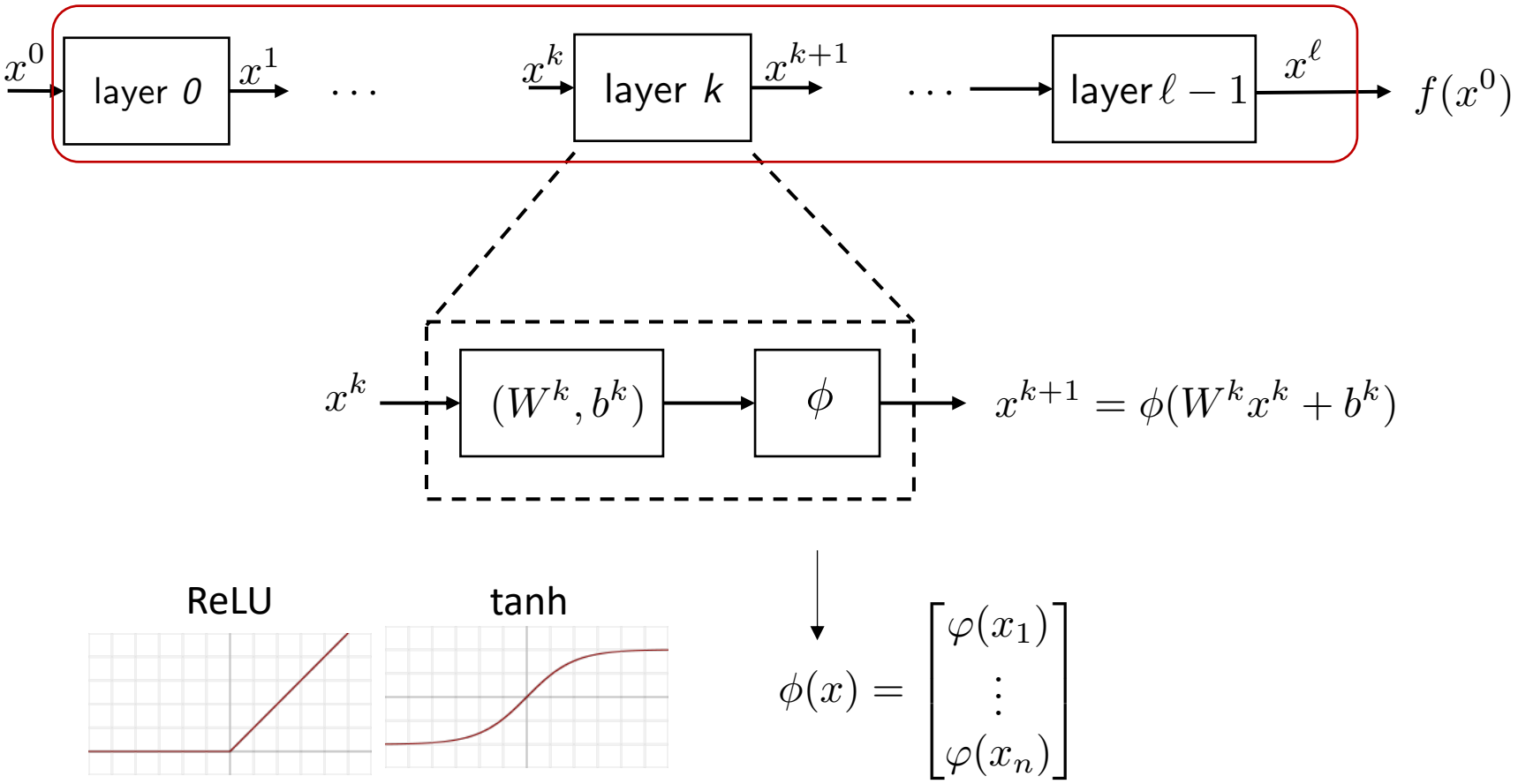
$$J^\star = \sup\{J(y) \mid y = f(x),\ x \in \mathcal{X}\} \leq \bar{J}$$



- The relaxation gap $\bar{J} - J^\star$ (propagation error) grows for large input sets and large neural networks

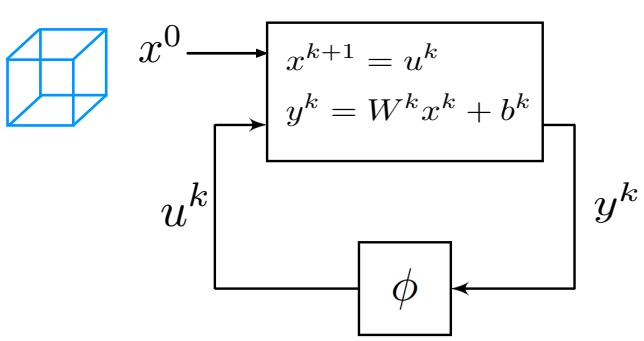
Deep Neural Networks

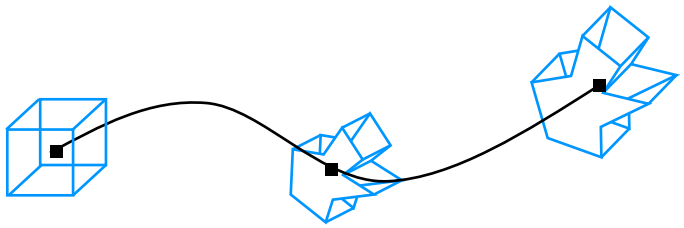
- Feed-forward fully-connected neural networks (FFC)



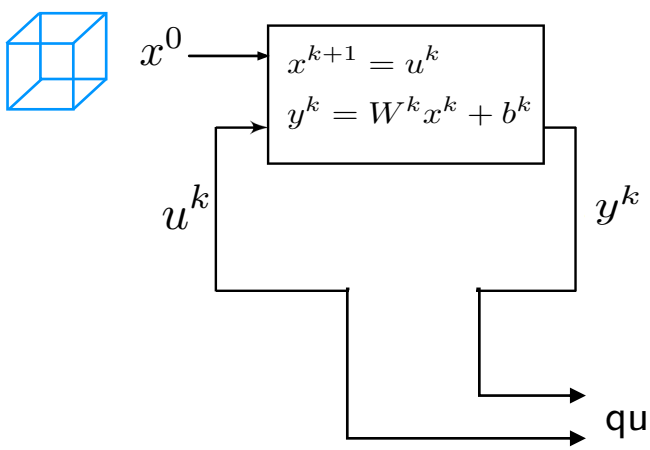
Robust Control Perspective

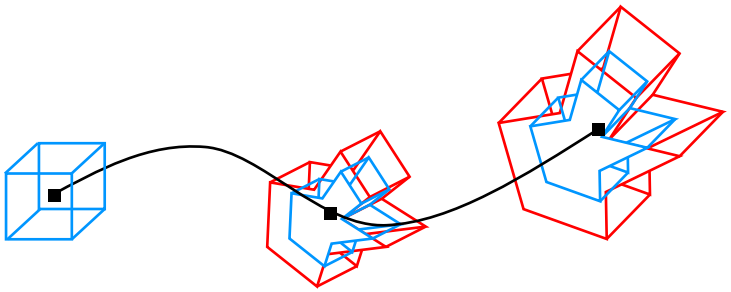
- Linear "time-varying" system with nonlinear feedback $k = 0, 1, \cdots, \ell - 1$





- Abstracted system: linear system subject to **quadratic constraints**





- Analyze via semidefinite programming (**DeepSDP**)

IEEE CSS IEEE TRANSACTIONS ON AUTOMATIC CONTROL, VOL. 67, NO. 1, JANUARY 2022 1

Safety Verification and Robustness Analysis of Neural Networks via Quadratic Constraints and Semidefinite Programming

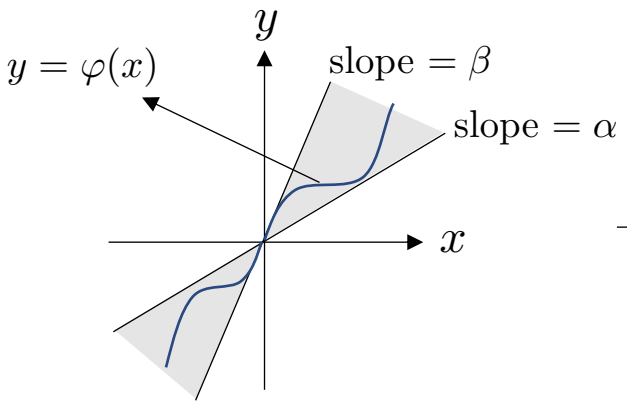
Mahyar Fazlyab, *Student Member, IEEE*, Manfred Morari, *Fellow, IEEE*, and George J. Pappas, *Fellow, IEEE*

Abstraction of Activation Functions via QCs

- Describe φ as QCs on its input-output pair

$$x \longrightarrow \boxed{\varphi} \longrightarrow \varphi(x) \xrightarrow{\text{QC}} \begin{bmatrix} x \\ \varphi(x) \\ 1 \end{bmatrix}^\top Q \begin{bmatrix} x \\ \varphi(x) \\ 1 \end{bmatrix} \geq 0 \quad \forall x \in \mathcal{X}$$

- Example: sector-bounded nonlinearities

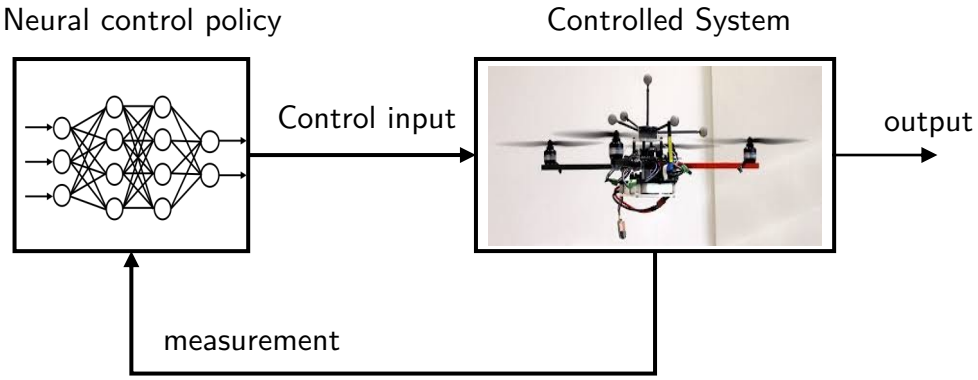


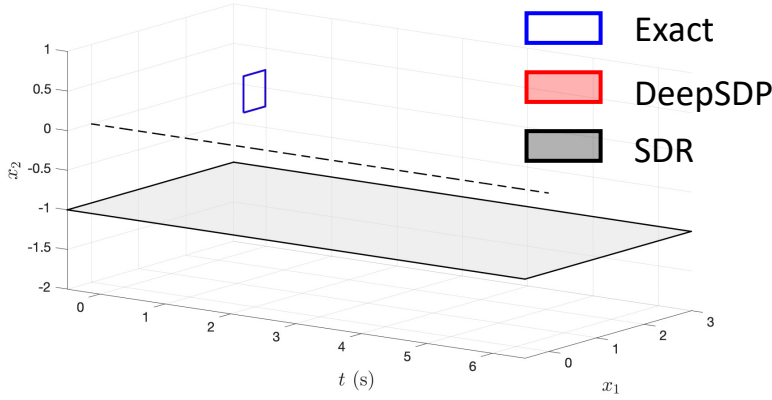
$$\longrightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}^\top \begin{bmatrix} -2\alpha\beta & \alpha+\beta & 0 \\ \alpha+\beta & -2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \geq 0$$

- We are over-approximating the graph of φ by QCs

single activation → layer of activations → neural network

Closed-Loop Verification via SDP





2020 59th IEEE Conference on Decision and Control (CDC)
Jeju Island, Republic of Korea, December 14-18, 2020

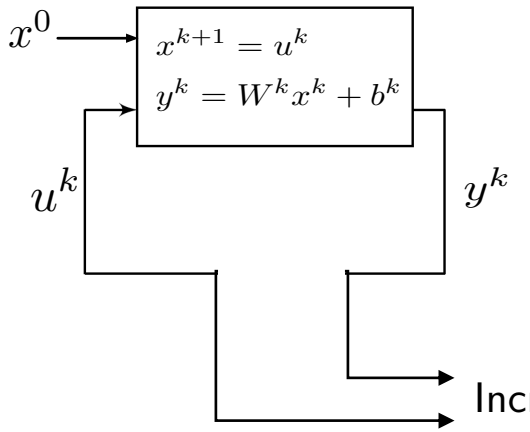
Reach-SDP: Reachability Analysis of Closed-Loop Systems with Neural Network Controllers via Semidefinite Programming

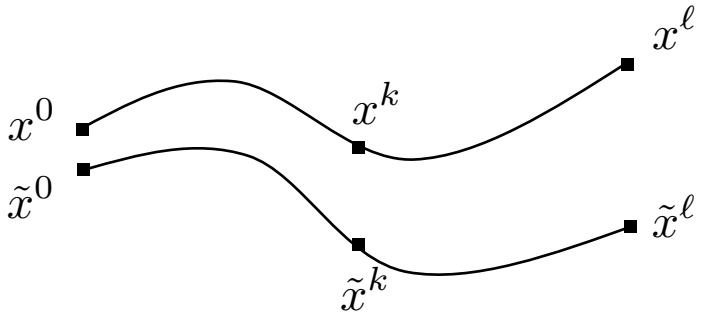
Haimin Hu, Mahyar Fazlyab, Manfred Morari, and George J. Pappas

SDR: Raghunathan, Aditi, Jacob Steinhardt, and Percy S. Liang. "Semidefinite relaxations for certifying robustness to adversarial examples." In Advances in Neural Information Processing Systems, pp. 10877-10887. 2018

Robust Control Perspective

- Abstract the nonlinearities via ***incremental* quadratic constraints**



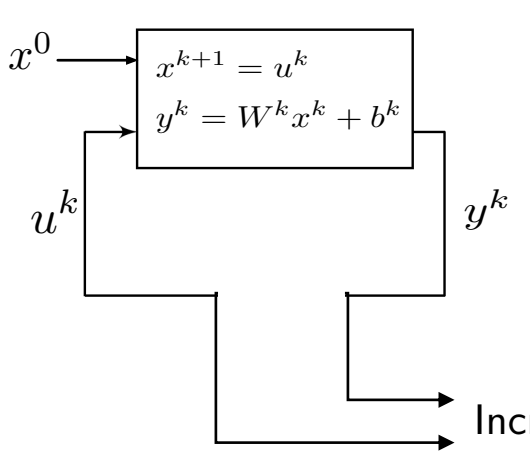


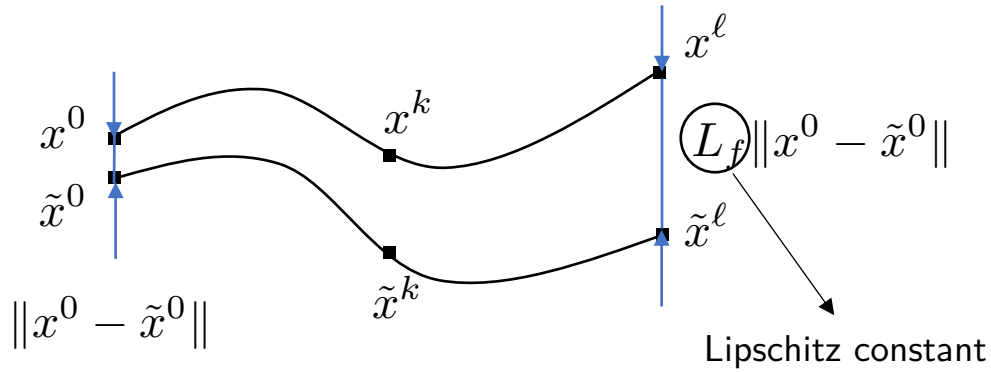
Incremental quadratic constraints on $(y^k - \tilde{y}^k, u^k - \tilde{u}^k)$

- Analyze via semidefinite programming

Bounding the Lipschitz Constant of NNs

- Abstract the nonlinearities via ***incremental*** **quadratic constraints**



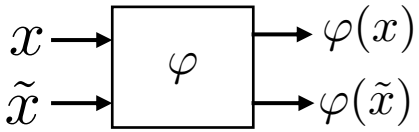


Incremental quadratic constraints on $(y^k - \tilde{y}^k, u^k - \tilde{u}^k)$

$$L_f = \sup_{x^0, \tilde{x}^0} \frac{\|f(x^0) - f(\tilde{x}^0)\|}{\|x^0 - \tilde{x}^0\|}$$

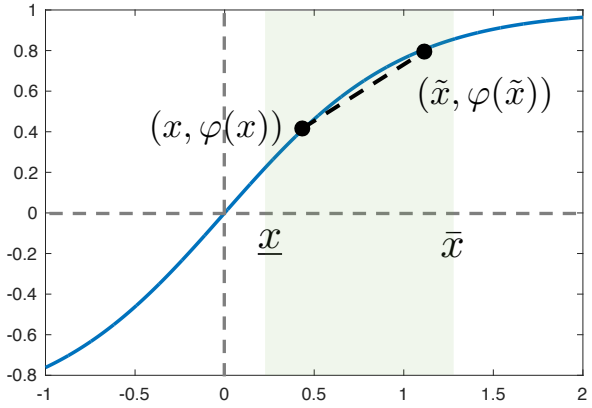
- Why important?
  - A measure of adversarial robustness/smoothness
  - Safety and stability guarantees for neural network controllers/deep RL methods
  - Generalization and approximation bounds

Abstraction of Activation Functions via Incremental QCs



Describe as incremental quadratic constraints on $(x, \varphi(x))$ and $(\tilde{x}, \varphi(\tilde{x}))$

- Example: slope-restricted nonlinearities



$$\alpha = \varphi'(\bar{x}) \leq \frac{\varphi(\tilde{x}) - \varphi(x)}{\tilde{x} - x} \leq \varphi'(\underline{x}) = \beta$$

$$\left[\star\right]^\top \begin{bmatrix} -2\alpha\beta & (\alpha+\beta) \\ (\alpha+\beta) & -2 \end{bmatrix} \begin{bmatrix} x - \tilde{x} \\ \varphi(x) - \varphi(\tilde{x}) \end{bmatrix} \geq 0$$

single activation → layer of activations → neural network

Lipschitz Constant Estimation via SDP

Theorem (One Layer): Consider the neural network $f(x^0) = W^1\phi(W^0x^0 + b^0) + b^1$
If $\exists\ \rho > 0$ and $\exists\lambda \in \mathbb{R}^n_+$ such that

$$M(\rho, \lambda) := \begin{bmatrix} -2\alpha\beta W^{0\top}\operatorname{diag}(\lambda)W^0 - \rho I_{n_x} & (\alpha+\beta)W^{0\top}\operatorname{diag}(\lambda) \\ (\alpha+\beta)\operatorname{diag}(\lambda)W^0 & -2\operatorname{diag}(\lambda) + W^{1\top}W^1 \end{bmatrix} \preceq 0$$

Then $\sqrt{\rho}$ is an upper bound on the Lipschitz constant of $f(x^0)$ on $\mathbb{R}^{n_x}$

- A semidefinite program for finding the best upper bound:

$$\text{minimize}_{\rho\geq 0, \lambda \geq 0} \quad \rho \quad \text{subject to } M(\rho, \lambda) \preceq 0$$

Efficient and Accurate Estimation of Lipschitz Constants for Deep Neural Networks

Mahyar Fazlyab
ESE Department
University of Pennsylvania
Philadephia , PA 19104
mahyarfa@seas.upenn.edu

Alexander Robey
ESE Department
University of Pennsylvania
Philadephia , PA 19104
arobey1@seas.upenn.edu

Hamed Hassani
ESE Department
University of Pennsylvania
Philadephia , PA 19104
hassani@seas.upenn.edu

Manfred Morari
ESE Department
University of Pennsylvania
Philadephia , PA 19104
morari@seas.upenn.edu

George J. Pappas
ESE Department
University of Pennsylvania
Philadephia , PA 19104
pappasg@seas.upenn.edu

Proceedings of Machine Learning Research vol 144:1–12, 2021

Certifying Incremental Quadratic Constraints for Neural Networks via Convex Optimization

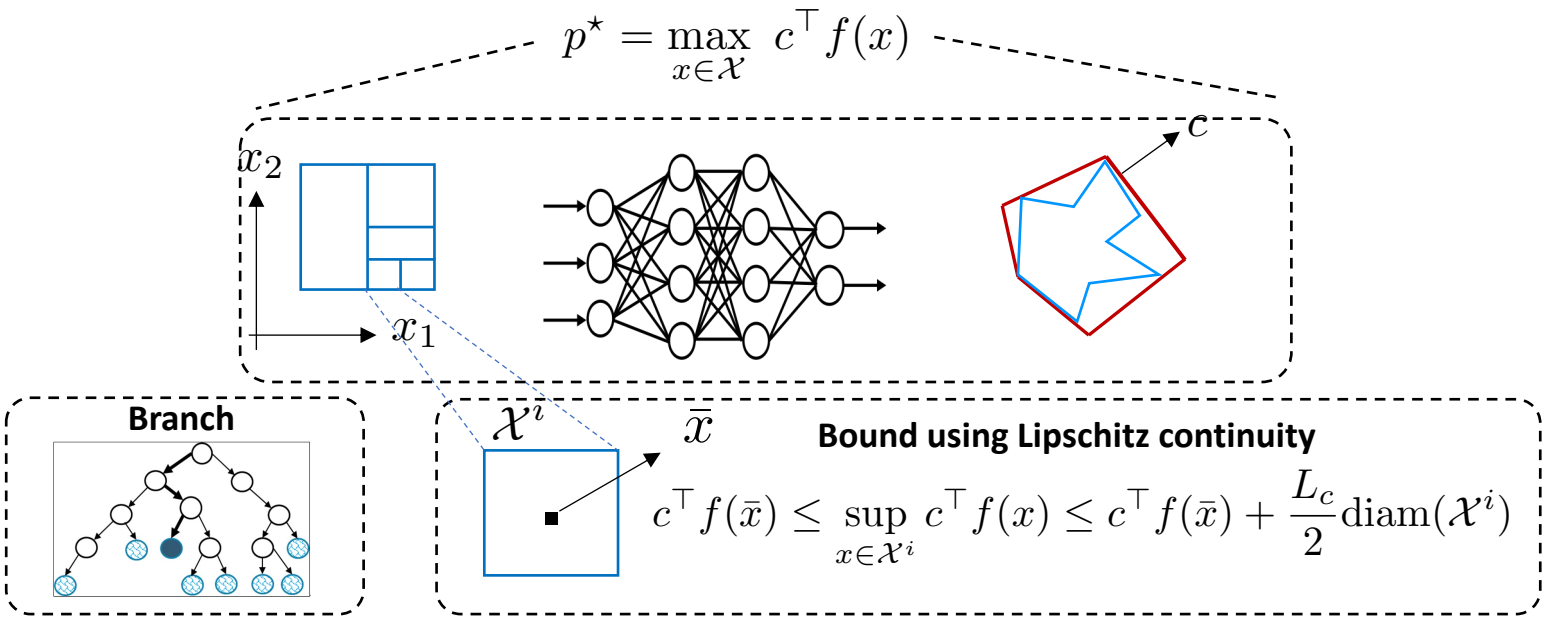
Navid Hashemi NAVID.HASHEMI@UTDALLAS.EDU
Mechanical Engineering, University of Texas at Dallas

Justin Ruths JRUTHS@UTDALLAS.EDU
Mechanical Engineering, University of Texas at Dallas

Mahyar Fazlyab MAHYARFAZLYAB@JHU.EDU
Mathematical Institute for Data Science, Johns Hopkins University

Black-Box Reachability

- The neural network model might not be available due to security or privacy reasons.
- Black-box reachability analysis using Lipschitz bounds



ReachLipBnB: A branch-and-bound method for reachability analysis of neural autonomous systems using Lipschitz bounds

Taha Entesari, Sina Sharifi, Mahyar Fazlyab

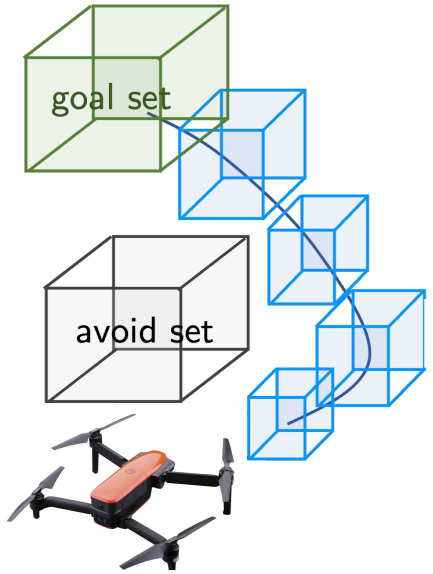
(Accepted in ICRA 2023)

Reachability Over Long Time Horizons

System dynamics (neural network)

$$x^{t+1} = f(x^t), \quad x^0 \in \mathcal{X}^0$$

Next state | Current state | Set of initial states

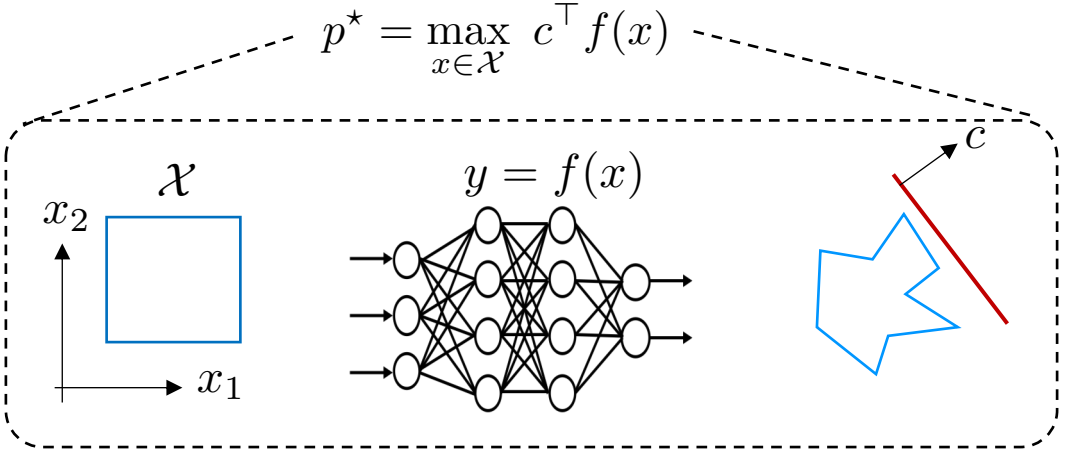


- Over-approximation error accumulates over time due to
  - **Propagation error:** underlying relaxations in the set propagation method
  - **Shape mismatch error:** potential mismatches between the shape of the over-approximator set and the true reachable set.

- Can we adapt the shape of the over-approximator sets to the shape of the reachable sets?

Automated Reachability Analysis

- **Choice of vector c:** How can we capture the shape of the reachable set as much as possible using polytopes?



- **Adaptive template polytopes:** Using the SVD of the weights to propose directions for a polytope on the output.

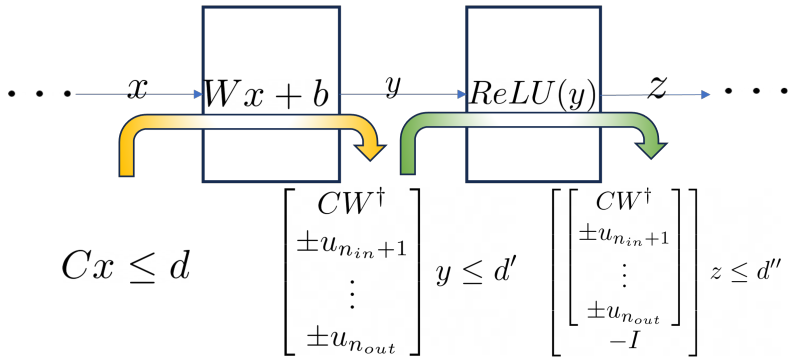
$$W = U\Sigma V^\top = \sum_{i=1}^{\min\{n_{in}, n_{out}\}} \sigma_i u_i v_i^\top$$

$$W^\dagger = V\Sigma^\dagger U^\top = \sum_{i=1}^{\min\{n_{in}, n_{out}\}} \sigma_i^{-1} v_i u_i^\top$$

$$Wx + b = y$$

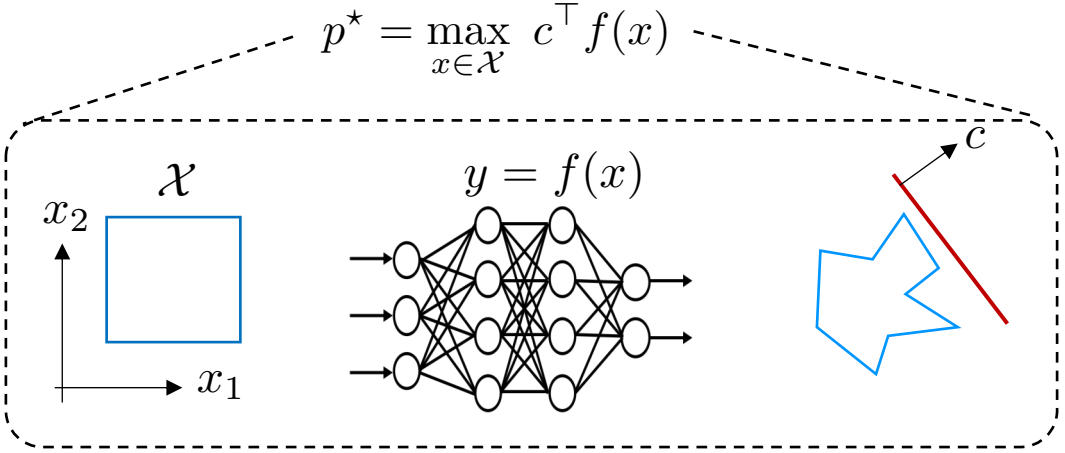
$$x = W^\dagger(y - b) + F\zeta$$

$$Cx \leq d \longrightarrow CW^\dagger y \leq d'$$

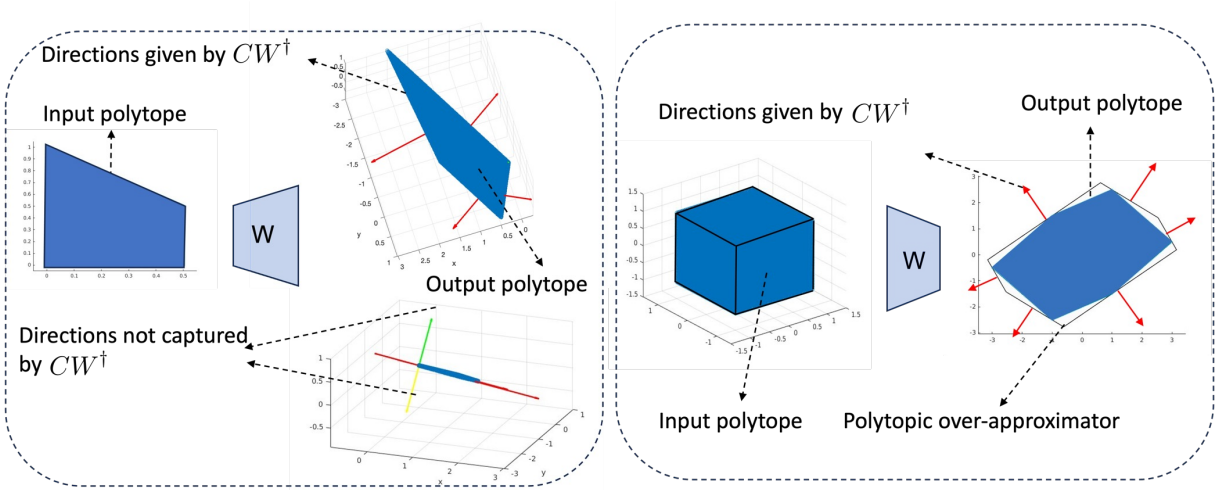


Automated Reachability Analysis

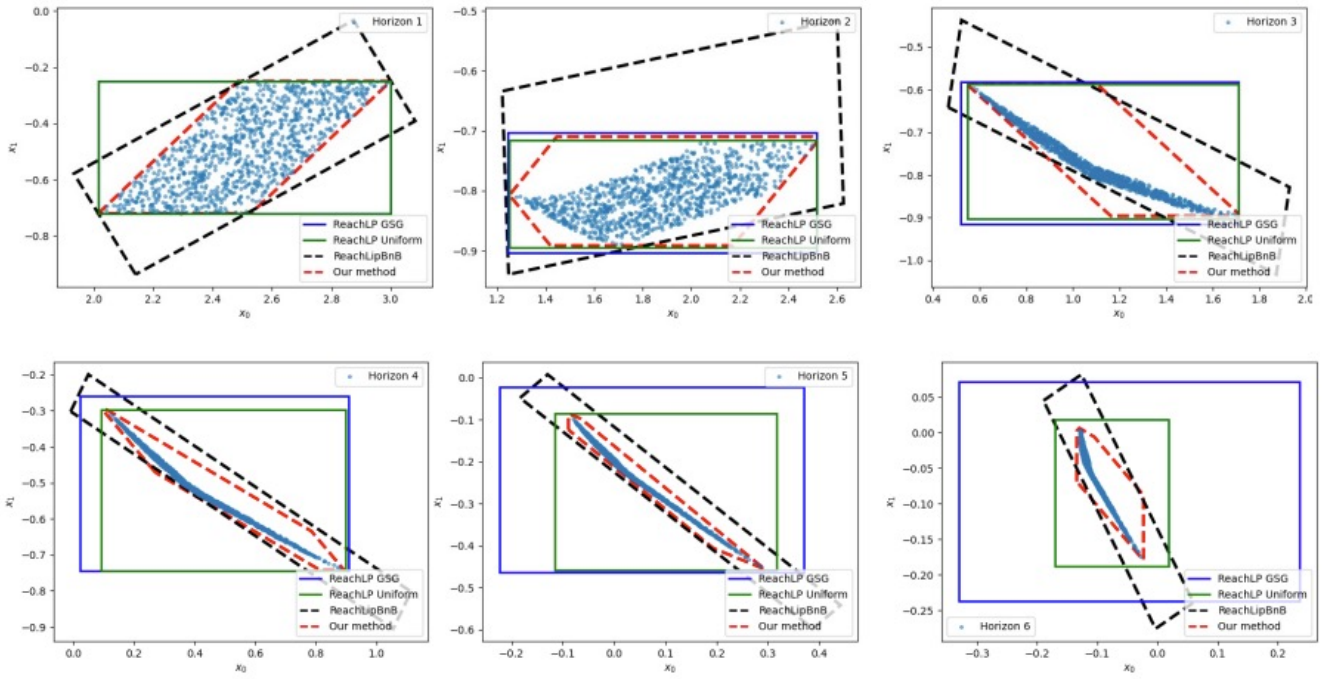
- **Choice of vector c:** How can we capture the shape of the reachable set as much as possible using polytopes?



- **Adaptive template polytopes:** Using the SVD of the weights to propose directions for a polytope on the output.



Automated Reachability Analysis



Proceedings of Machine Learning Research vol xxx:1–13, 2023

Automated Reachability Analysis of Neural Network-Controlled Systems via Adaptive Polytopes

Taha Entesari TENTESA1@JHU.EDU
Department of Electrical and Computer Engineering, Johns Hopkins University, USA

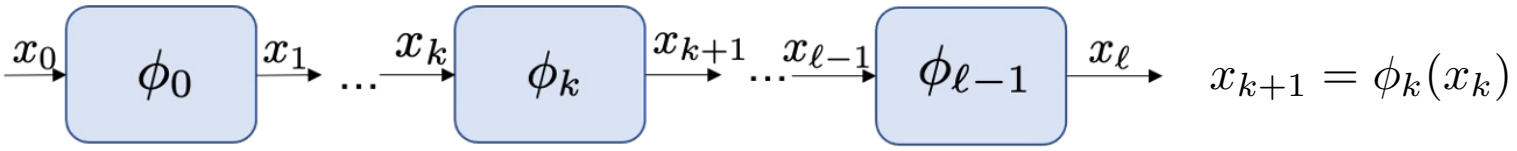
Mahyar Fazlyab MAHYARFAZLYAB@JHU.EDU
Department of Electrical and Computer Engineering, Johns Hopkins University, USA

Part II

Scalable Verification of Deep Neural Networks via Operator Splitting

Modular and Scalable Verification of NNs

- NNs as compositions of operators
  - linear (fully connected, convolutional) layers, max-pooling units, activation functions...

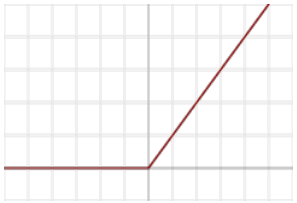


$$J^\star = \sup\{J(x_\ell) \mid x_\ell = f(x_0),\ x_0 \in \mathcal{X}\}$$

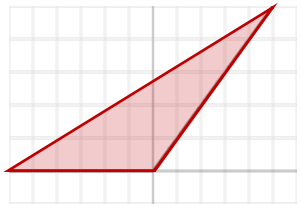
- Layer-wise convex relaxation

$$\begin{array}{ll} \text{maximize} & J(x_\ell) \\ \text{subject to} & (x_k, x_{k+1}) \in \text{gr}(\phi_k) \quad k = 0, \cdots, \ell-1 \\ & x_0 \in \mathcal{X}. \end{array}$$

non-convex

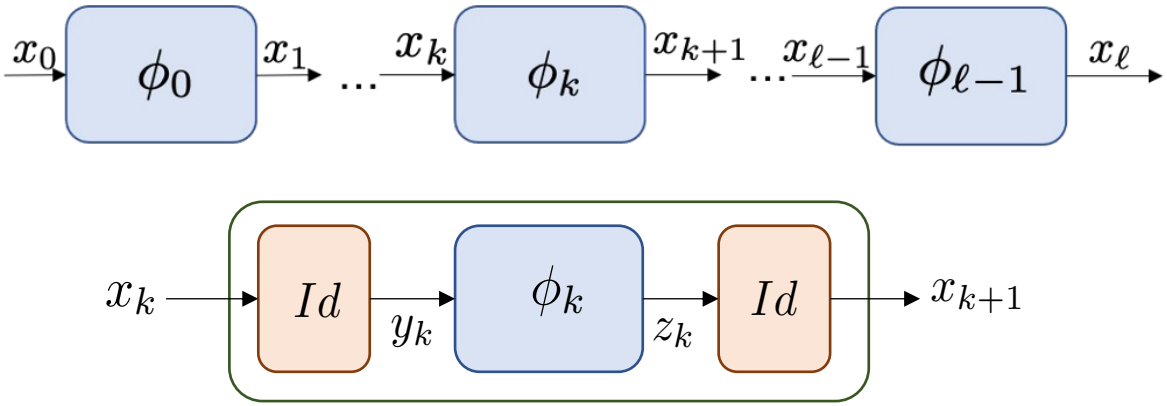


Convex relaxation



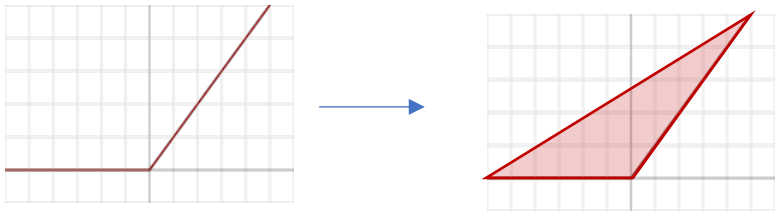
Splitting Layers

- Add virtual "identity layers" between actual layers



- Layer-wise convex relaxation

$$
\begin{aligned}
J^\star_{\text{relaxed}} \leftarrow \text{maximize} \quad & J(x_\ell) \\
\text{subject to} \quad & y_k = x_k, && k = 0, \cdots, \ell-1 \\
& (y_k, z_k) \in \text{gr}(\phi_k), && k = 0, \cdots, \ell-1 \\
& x_{k+1} = z_k, && k = 0, \cdots, \ell-1 \\
& x_0 \in \mathcal{X},
\end{aligned}
$$



Operator Splitting for NN Verification

$$\begin{array}{lll} \text{minimize} & J(x_\ell)+\mathbb{I}_{\mathcal{X}}(x_0) + \sum_{k=0}^{\ell-1} \mathbb{I}_{\mathcal{S}_{\phi_k}}(y_k, z_k) & \\ \text{subject to} & y_k = x_k & k = 0, \cdots, \ell-1 \\ & x_{k+1} = z_k & k = 0, \cdots, \ell-1 \end{array}$$

- This problem is of the form

$$\begin{array}{ll} \text{minimize} & f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) \\ \text{subject to} & A_1\mathbf{x}_1 + A_2\mathbf{x}_2 = 0 \end{array}$$

  - where $\mathbf{x}_1 = (x_0, \cdots, x_\ell)$ and $\mathbf{x}_2 = (y_0, \cdots, y_{\ell-1}, z_0, \cdots, z_{\ell-1})$
  - two sets of variables, with separable objective

- $L_\rho(\mathbf{x}_1, \mathbf{x}_2, \lambda) = f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) + \lambda^\top(A_1\mathbf{x}_1 + A_2\mathbf{x}_1) + (\rho/2)\|A_1\mathbf{x}_1 + A_2\mathbf{x}_2\|_2^2$

$$\begin{array}{lll} \mathbf{x}_1^+ & := \text{argmin}_{\mathbf{x}_1} L_\rho(\mathbf{x}_1, \mathbf{x}_2, \lambda) & //\mathbf{x}_1\text{-minimization} \\ \mathbf{x}_2^+ & := \text{argmin}_{\mathbf{x}_2} L_\rho(\mathbf{x}_1^+, \mathbf{x}_2, \lambda) & //\mathbf{x}_2\text{-minimization} \\ \lambda^+ & := \lambda + \rho(A_1\mathbf{x}_1^+ + A_2\mathbf{x}_2^+) & // \text{ dual update} \end{array}$$

DeepSplit Algorithm

- **First update**

$$x_0^+ = \mathrm{Proj}_{\mathcal{X}}(y_0 - \lambda_0)$$

$$x_k^+ = \frac{1}{2}(y_k - \lambda_k + z_{k-1} - \mu_{k-1}) \;\; k = 1, \cdots, \ell - 1,$$

$$x_\ell^+ = \arg\min_{x_\ell} \; J(x_\ell) + \frac{\rho}{2}\|x_\ell - z_{\ell-1} + \mu_{\ell-1}\|_2^2$$

- **Second update**

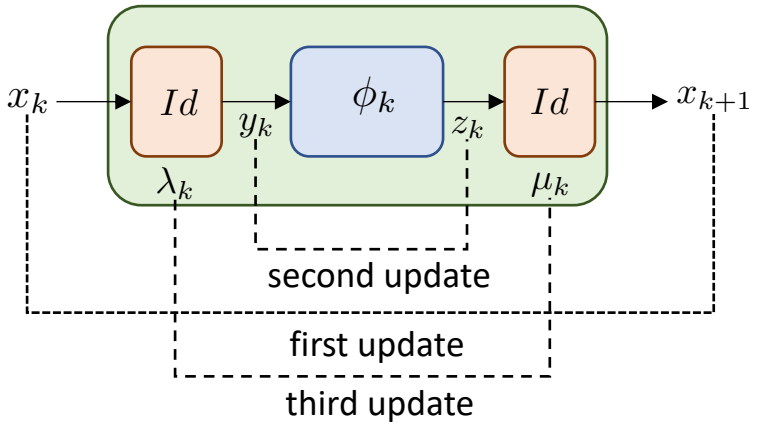
$$(y_k^+, z_k^+) = \mathrm{Proj}_{\mathrm{conv}(\mathrm{gr}(\phi_k))}(x_k^+ + \lambda_k, x_{k+1}^+ + \mu_k) \;\; k = 0, \cdots, \ell - 1,$$

- **Third update**

$$\lambda_k^+ = \lambda_k + (x_k^+ - y_k^+) \;\; k = 0, \cdots, \ell - 1,$$

$$\mu_k^+ = \mu_k + (x_{k+1}^+ - z_k^+) \;\; k = 0, \cdots, \ell - 1.$$

ADMM module for layer *k*



IEEE CSS

Received 11 March 2022; revised 28 May 2022; accepted 13 June 2022. Date of publication 30 June 2022; date of current version 2 August 2022. Recommended by Senior Editor Sonia Martinez.

Digital Object Identifier 10.1109/OJCSYS.2022.3187429

DeepSplit: Scalable Verification of Deep Neural Networks via Operator Splitting

SHAORU CHEN [1] (Graduate Student Member, IEEE), ERIC WONG [2], J. ZICO KOLTER [3], AND MAHYAR FAZLYAB [4] (Member, IEEE)

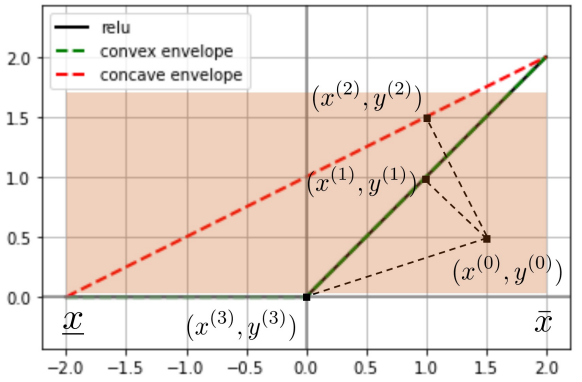
DeepSplit Algorithm—Second Update

- **Projection onto activation layers**

$$(y_k^+, z_k^+) = \mathrm{Proj}_{\mathcal{S}_{\phi_k}}(x_k^+ + \lambda_k, x_{k+1}^+ + \mu_k)\ k = 0, \cdots, \ell - 1$$

- Projection can be done coordinate wise

$$\begin{aligned}
x^{(1)} &= \min(\max(\frac{x^{(0)} + y^{(0)}}{2}, 0), \bar{x}), \quad y^{(1)} = x^{(1)}, \\
x^{(2)} &= \min(\max(\frac{x^{(0)} + sy^{(0)} + s(s\underline{x} - \underline{y})}{s^2 + 1}, \underline{x}), \bar{x}), \\
y^{(2)} &= \frac{s(x^{(0)} - \underline{x}) + s^2 y^{(0)} + \underline{y}}{s^2 + 1}, \\
x^{(3)} &= \min(\max(0, x^{(0)}), \underline{x}), \quad y^{(3)} = 0.
\end{aligned}$$



DeepSplit Algorithm—Second Update

- **Projection onto affine layers**

$$\phi_k(y_k) = W_k y_k + b_k$$

$$\mathrm{gr}(\phi_k) = \{(y_k, z_k) \mid z_k = W_k y_k + b_k\}$$

$$\begin{aligned} y_k^+ &= (I_{n_k} + W_k^\top W_k)^{-1}(x_k^+ + \lambda_k + W_k^\top (x_{k+1}^+ + \mu_k - b_k)), \\ z_k^+ &= W_k y_k^+ + b_k. \end{aligned}$$

- The matrix inverse $(I_{n_k} + W_k^\top W_k)^{-1}$ can be computed and cached for subsequent iterations
  - The inversion can be computed efficiently for convolutional layers

DeepSplit Algorithm- Scalability

- CNN trained with PGD-based adversarial training, ~60k ReLUs

TABLE 1: Certified test accuracy (%) of PGD-trained models on CIFAR10 through ADMM, the Lagrangian decomposition methods [13], [33], and fast dual/linear [19], [20] or interval bounds [43]. All the methods are given the same time budget.

	Exact	Lagrangian methods				Fast bounds	
ϵ	ADMM	Adam	Prox	Dual Adam	Dual Decomp Adam	Linear	IBP
1/255	**64.0**	60.5	62.4	59.8	60.3	59.8	42.8
1.5/255	**45.7**	41.2	43.5	40.5	41.1	36.8	16.8
2/255	**19.5**	17.3	18.2	16.9	17.1	13.2	3.6
2.5/255	**5.5**	4.6	4.9	4.5	4.6	3.3	0.7

[13] K. Dvijotham, R. Stanforth, S. Gowal, T. Mann, and P. Kohli, "A dual approach to scalable verification of deep networks," *arXiv preprint arXiv:1803.06567*, 2018.

[33] R. Bunel, J. Lu, I. Turkaslan, P. Kohli, P. Torr, and P. Mudigonda, "Branch and bound for piecewise linear neural network verification," *Journal of Machine Learning Research*, vol. 21, no. 2020, 2020.

[19] E. Wong, F. R. Schmidt, J. H. Metzen, and J. Z. Kolter, "Scaling provable adversarial defenses," *arXiv preprint arXiv:1805.12514*, 2018.

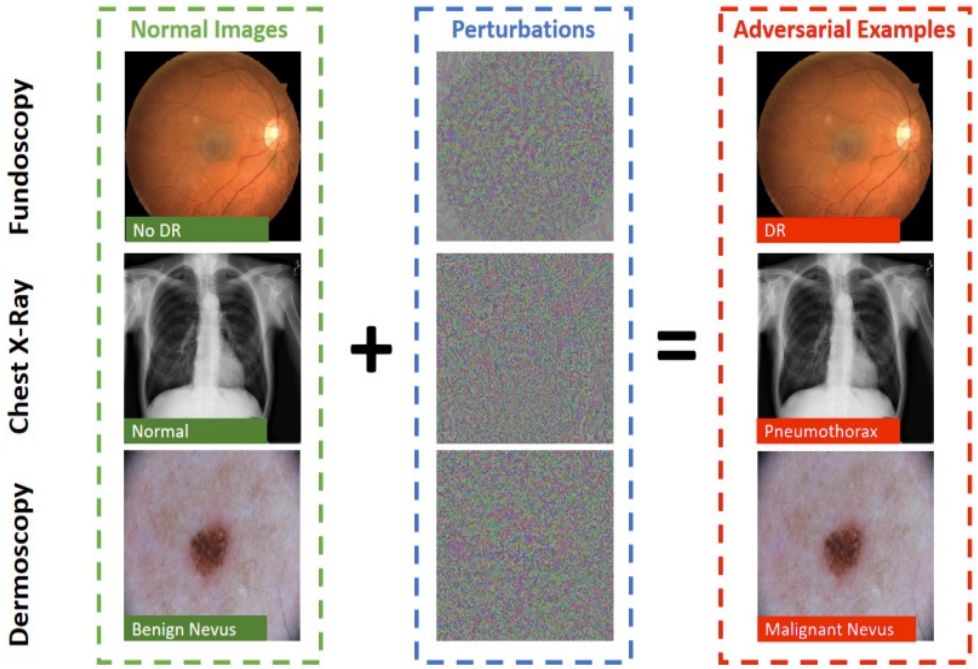
[20] K. Xu, Z. Shi, H. Zhang, Y. Wang, K.-W. Chang, M. Huang, B. Kailkhura, X. Lin, and C.-J. Hsieh, "Automatic perturbation analysis for scalable certified robustness and beyond," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[43] S. Gowal, K. Dvijotham, R. Stanforth, R. Bunel, C. Qin, J. Uesato, T. Mann, and P. Kohli, "On the effectiveness of interval bound propagation for training verifiably robust models," *arXiv preprint arXiv:1810.12715*, 2018.

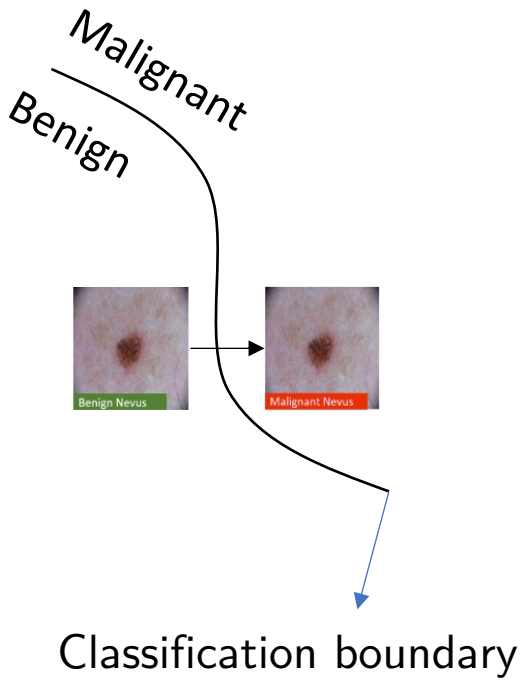
Part III
Final Thoughts

Beyond Control

- Robustness of classifiers (e.g., medical imaging)

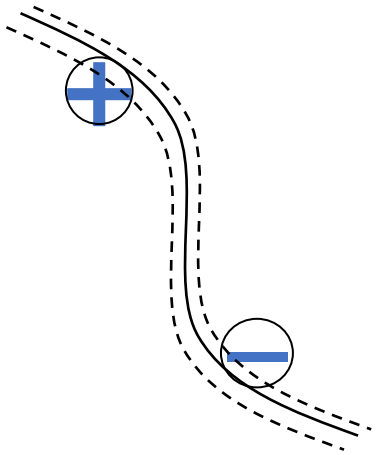


Ma, Xingjun, et al. (2021)

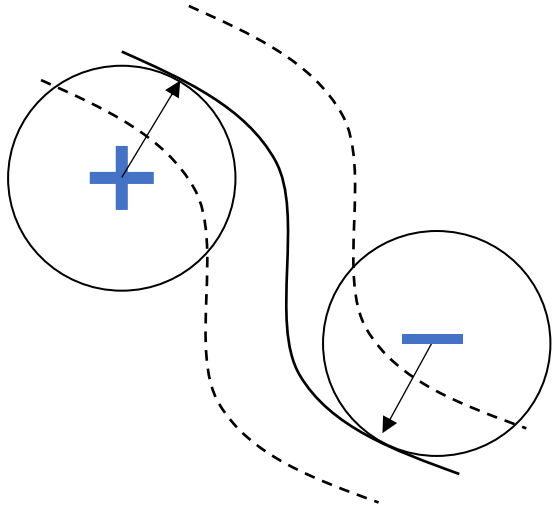


Promoting Robustness During Training

- **Ongoing work:** efficient and direct manipulation of decision boundary to increase margin in the feature space



Standard training (small margin)

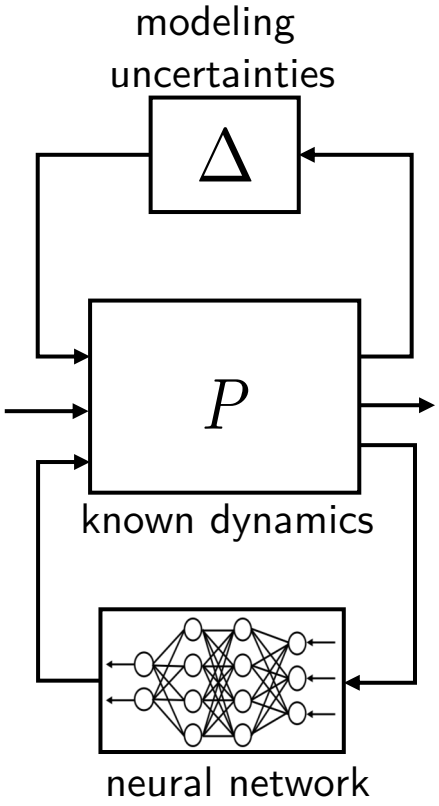


Robust training (large margin)

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathrm{CE}(x, y; \theta)]$$

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}[-\text{“margin”}]$$

Summary and Open Challenges



- **Open-loop analysis**
  - Reachability analysis via semidefinite programming
  - Stability and robustness analysis via Lipschitz constants
  - Verification of large-scale deep networks via operator splitting
- **Closed-loop analysis/synthesis**
  - Reachability analysis via semidefinite programming
  - Learning stability certificates for hybrid systems
  - Enforcing robustness guarantees within NN control policies
  - Black-box reachability analysis via Lipschitz bounds
- Code: https://github.com/mahyarfazlyab
  - https://github.com/o4lc

- **Open Challenges:**
  - Large-scale architectures (e.g., LLMs, perception neural networks)
  - Large *and* high dimensional uncertainties
  - Data-driven verification of complex systems + NNs
  - Scalable verification of other desirable properties (stability, fairness, consistency,…)