

Nonparametric Analysis and Control of Dynamical Systems

Stability, Safety, and Policy Improvement

Enrique Mallada



JOHNS HOPKINS
UNIVERSITY

Tsinghua University

Department of Automation

July 21th, 2025

Acknowledgements



Yue Shen



Roy Siegelmann



Agustin Castellano



Sohrab Rezaei



Fernando Paganini



Maxim Bichuch



Hussein Sibai



Jared Markowitz



A Dream World of Success Stories

2017 Google DeepMind's DQN



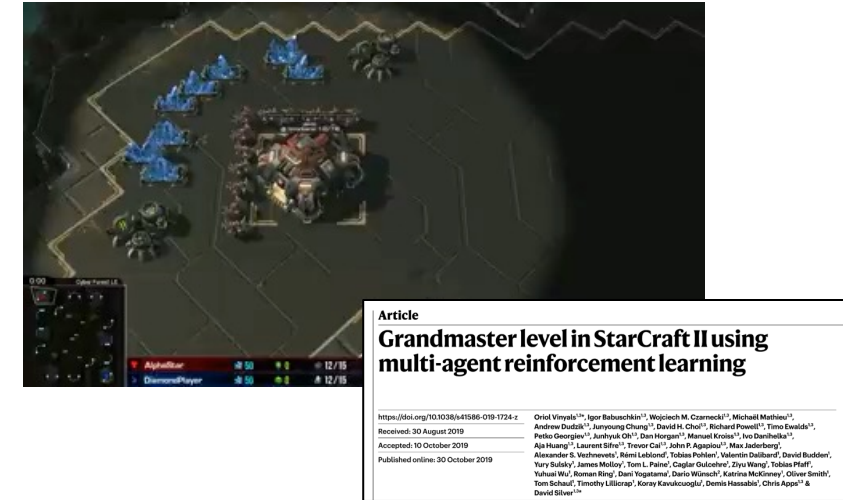
2017 AlphaZero – Chess, Shogi, Go



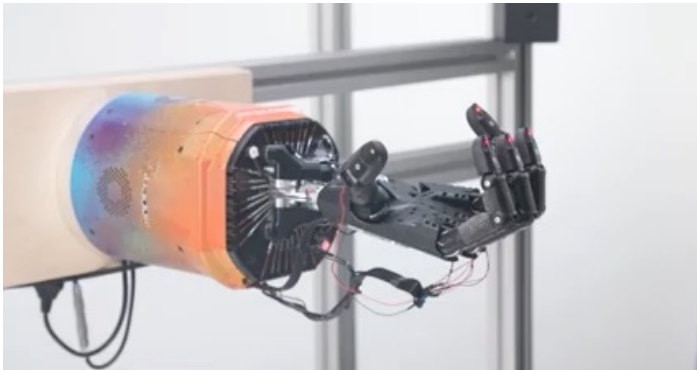
Boston Dynamics



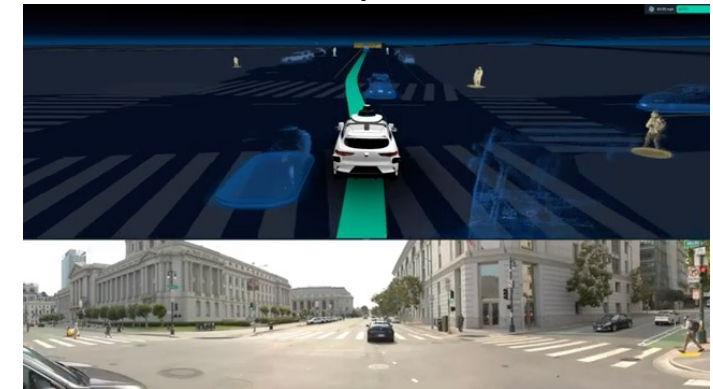
2019 AlphaStar – Starcraft II



OpenAI – Rubik's Cube



Waymo



Reality Kicks In

Angry Residents, Abrupt Stops: Waymo Vehicles Are Still Causing Problems in Arizona

RAY STERN | MARCH 31, 2021 | 8:26AM

DeepMind's Losses and the Future of Artificial Intelligence

Alphabet's DeepMind unit, conqueror of Go and other games, is losing lots of money. Continued deficits could imperil investments in AI.

AARIAN MARSHALL | BUSINESS | 12.07.2020 04:06 PM

Uber Gives Up on the Self-Driving Dream

The ride-hail giant invested more than \$1 billion in autonomous vehicles. Now it's selling the unit to Aurora, which makes self-driving tech.

Tesla Recalls Nearly All Vehicles Due to Autopilot Failures

Tesla disagrees with feds' analysis of glitches

BY LINA FISHER, 2:54PM, WED. DEC. 13, 2023

CRUISE KNEW ITS SELF-DRIVING CARS HAD PROBLEMS RECOGNIZING CHILDREN — AND KEPT THEM ON THE STREETS

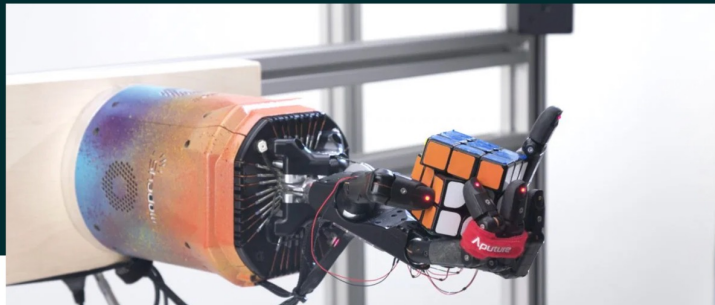
According to internal materials reviewed by The Intercept, Cruise cars were also in danger of driving into holes in the road.



OpenAI disbands its robotics research team

Kyle Wiggers | @Kyle_L_Wiggers | July 16, 2021 11:24 AM

f t in



Self-driving Uber car that hit and killed woman did not recognize that pedestrians jaywalk

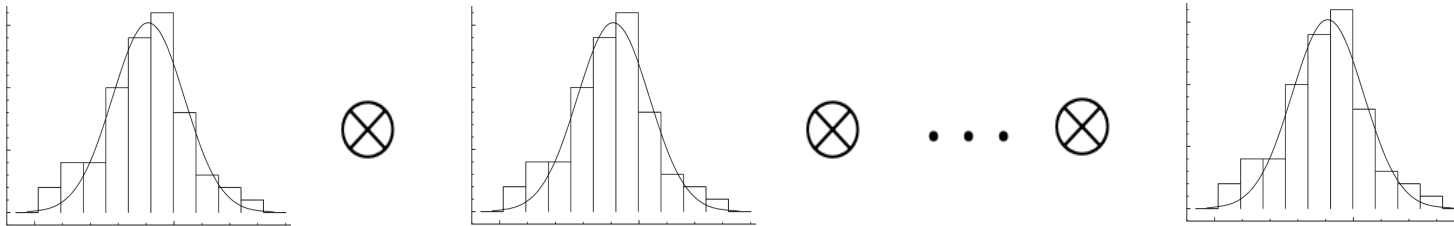
The automated car lacked "the capability to classify an object as a pedestrian unless that object was near a crosswalk," an NTSB report said.



Fundamental challenge: The curse of dimensionality

■ Statistical: No clear inductive bias

Sampling in d dimension with resolution ϵ :



Sample complexity:

$$O(\epsilon^{-d})$$

For $\epsilon = 0.1$ and $d = 100$, we would need 10^{100} points.
Atoms in the universe: 10^{78}

■ Computational: Verifying non-negativity of polynomials

Copositive matrices:

$$\begin{bmatrix} x_1^2 & \dots & x_d^2 \end{bmatrix} A \begin{bmatrix} x_1^2 & \dots & x_d^2 \end{bmatrix}^T \geq 0$$

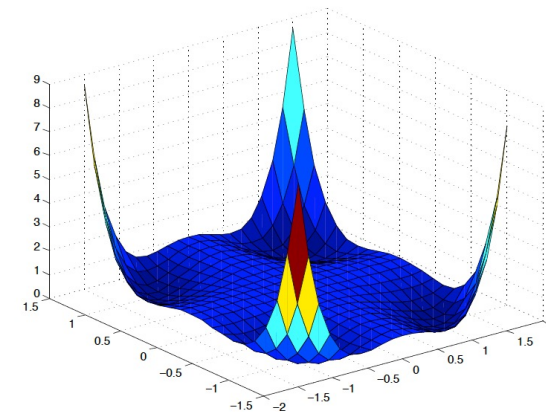
Murty&Kadabi [1987]: Testing co-positivity is NP-Hard

Sum of Squares (SoS):

$$z(x)^T Q z(x) \geq 0, \quad z_i(x) \in \mathbb{R}[x], \quad x \in \mathbb{R}^d, Q \succcurlyeq 0$$

Artin [1927] (Hilbert's 17th problem):

Non-negative polynomials are sum of square of *rational* functions



Motzkin [1967]:

$$p = x^4y^2 + x^2y^4 + 1 - 3x^2y^2$$

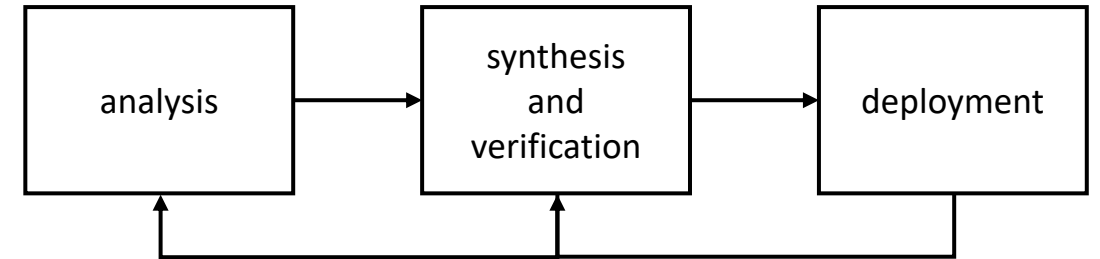
is nonnegative,

not a sum of squares,

but $(x^2 + y^2)^2 p$ is SoS

Methodological challenges

- Focused on a ***design-then-deploy*** philosophy
 - Most methods have a strict separation between control synthesis and deployment
- Synthesis usually aims for the ***best*** (optimal) controller
 - Lack of exploration of the benefits of designing sub-optimal controllers
- Policy ***parameters*** can ***drastically affect*** the system's ***behavior***
 - The params to behavior maps are highly sensitive to perturbations



RL:

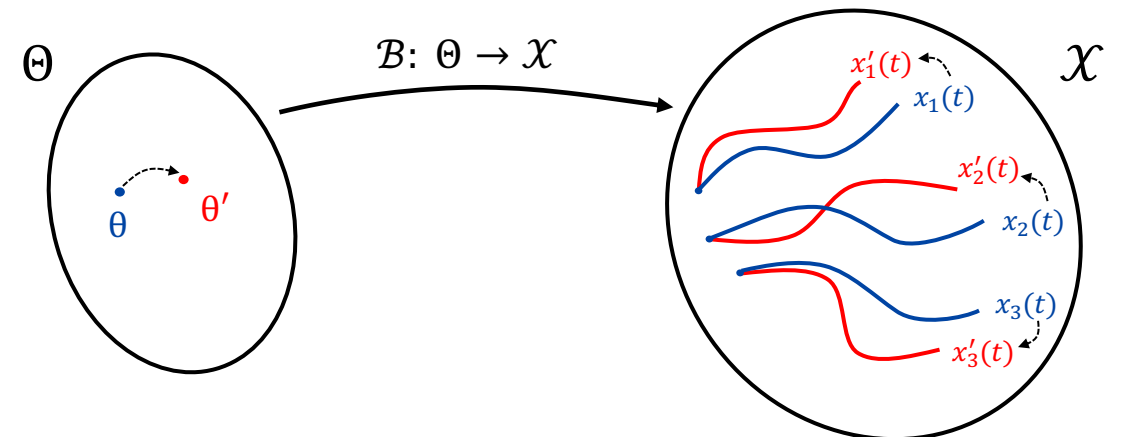
$$\max_{\pi} J(\pi) = \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right]$$

$$\text{s.t. } s_{t+1} \sim P(\cdot \mid s_t, a_t), \quad a_t \sim \pi(\cdot \mid s_t)$$

Optimal Control:

$$\min_{u(\cdot)} J = \int_0^T L(x(t), u(t), t) dt + \Phi(x(T))$$

$$\text{s.t. } \dot{x}(t) = f(x(t), u(t), t), \quad x(0) = x_0$$



Aspirational Goals

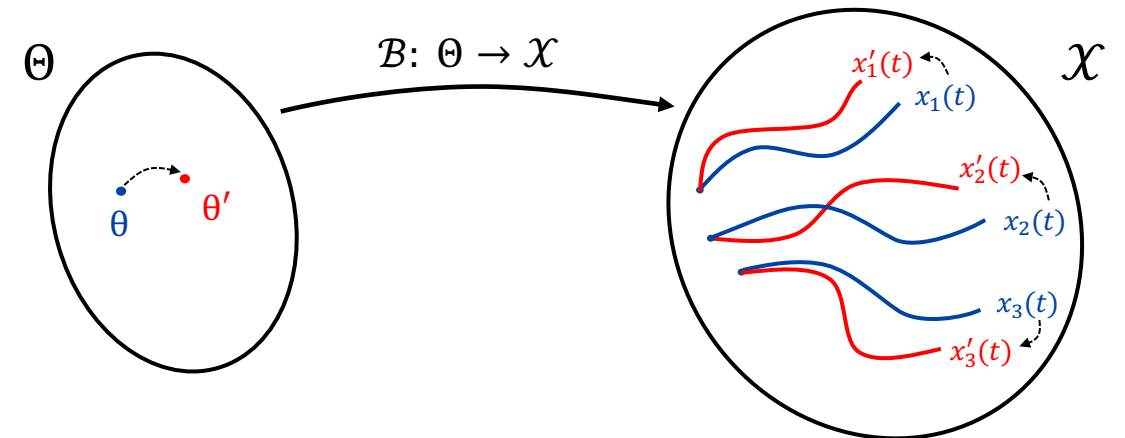
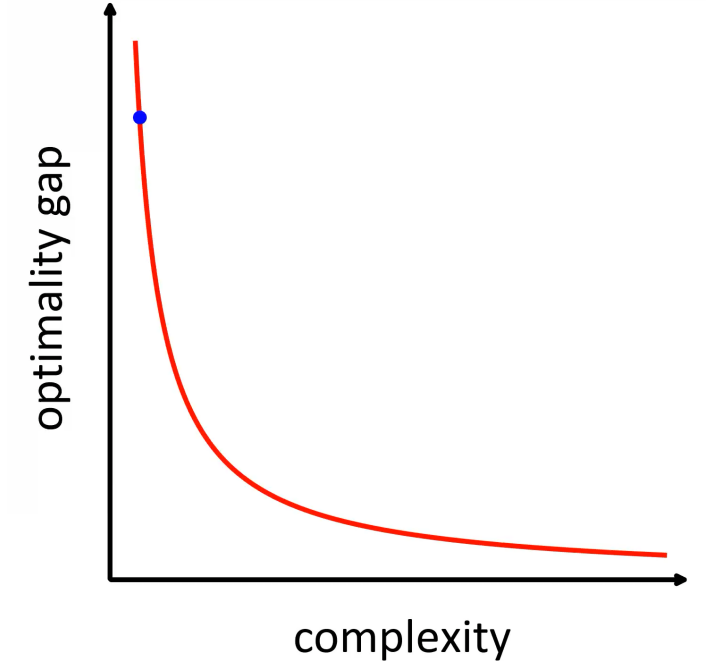
To design policies as nature does...



refining post deployment self improving, with each trial discarding poor decisions reinforcing good ones

Research Goals

- To develop analysis and design methods that *trade off complexity and performance*.
- To allow for *continual improvement*, without the need for redesign, retune, or retrain
- To design control policies with controlled sensitivity to parameter changes



Outline

- **Relaxing Invariance: Merits and trade offs**
 - *Recurrent Sets*: Letting thing go and come back
- **Nonparametric Analysis via Recurrent Sets**
 - *Stability analysis*: Recurrent Lyapunov Functions (RLFs)
 - *Safety verification*: Recurrent Barrier Functions (RBFs)
- **Self-Improving via Nonparametric Control Policies**
 - Policy Improvement using Expert Demonstrations

Outline

- **Relaxing Invariance: Merits and trade offs**
 - *Recurrent Sets*: Letting thing go and come back
- **Nonparametric Analysis via Recurrent Sets**
 - *Stability analysis*: Recurrent Lyapunov Functions (RLFs)
 - *Safety verification*: Recurrent Barrier Functions (RBFs)
- **Self-Improving via Nonparametric Control Policies**
 - Policy Improvement using Expert Demonstrations

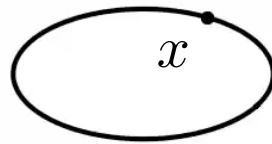
Problem setup

Continuous time dynamical system: $\dot{x}(t) = f(x(t))$

- Initial condition $x_0 = x(0)$, solution at time t : $\phi(t, x_0)$.

Asymptotic behavior: Ω -Limit Set $\Omega(f)$

$$x \in \Omega(f) \iff \exists x_0, \{t_n\}_{n \geq 0}, \text{ s.t. } \lim_{n \rightarrow \infty} t_n = \infty \text{ and } \lim_{n \rightarrow \infty} \phi(t_n, x_0) = x$$



Problem setup

Continuous time dynamical system: $\dot{x}(t) = f(x(t))$

- Initial condition $x_0 = x(0)$, solution at time t : $\phi(t, x_0)$.

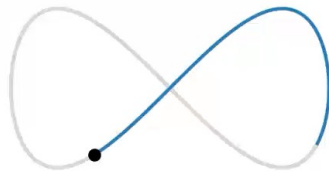
Ω -Limit Set $\Omega(f)$:

$$x \in \Omega(f) \iff \exists x_0, \{t_n\}_{n \geq 0}, \text{ s.t. } \lim_{n \rightarrow \infty} t_n = \infty \text{ and } \lim_{n \rightarrow \infty} \phi(t_n, x_0) = x$$

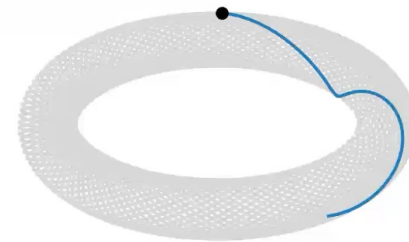
Types of Ω -limit set



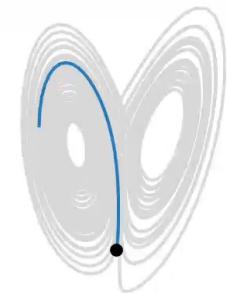
equilibrium



limit cycle



limit torus



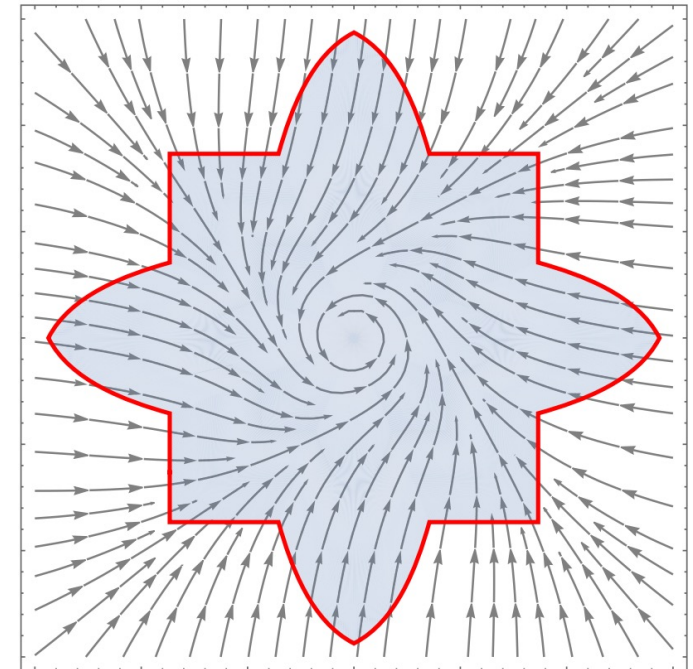
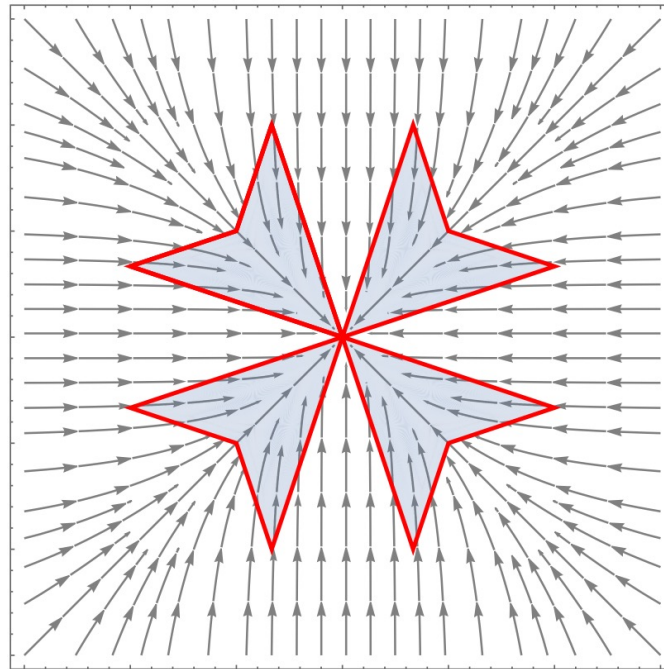
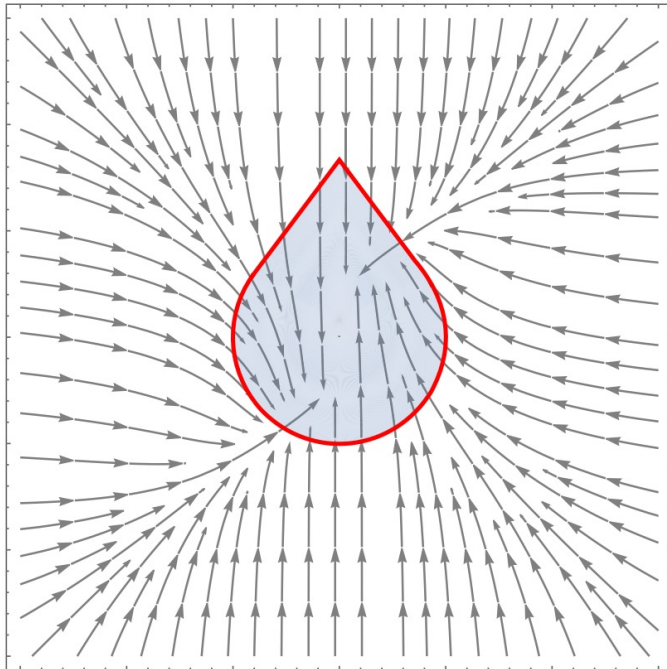
chaotic attractor

Remark: invariance is a shared property, thus a natural tool for analysis

Invariant sets

A set $\mathcal{S} \subseteq \mathbb{R}^d$ is **positively invariant** if and only if: $x_0 \in \mathcal{S} \rightarrow \phi(t, x_0) \in \mathcal{S}, \forall t \geq 0$

Any trajectory starting in the set remains in inside it for all times



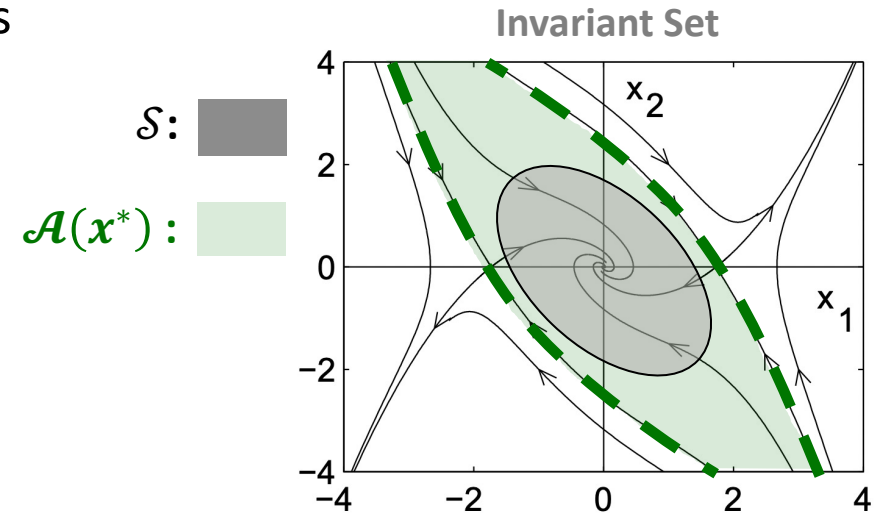
Invariant sets: Merits

A set $\mathcal{S} \subseteq \mathbb{R}^d$ is **positively invariant** if and only if: $x_0 \in \mathcal{S} \rightarrow \phi(t, x_0) \in \mathcal{S}, \forall t \geq 0$

Any trajectory starting in the set remains in inside it for all times

- Invariant sets approximate regions of attraction**

Compact invariant set $\mathcal{S}$, containing **only** $\{x^*\} = \Omega(f) \cap \mathcal{S}$ must be in the region of attraction $\mathcal{A}(x^*)$ ($\mathcal{S} \subset \mathcal{A}(x^*)$)



Invariant sets: Merits

A set $\mathcal{S} \subseteq \mathbb{R}^d$ is **positively invariant** if and only if: $x_0 \in \mathcal{S} \rightarrow \phi(t, x_0) \in \mathcal{S}, \forall t \geq 0$

Any trajectory starting in the set remains in inside it for all times

- Invariant sets approximate regions of attraction**

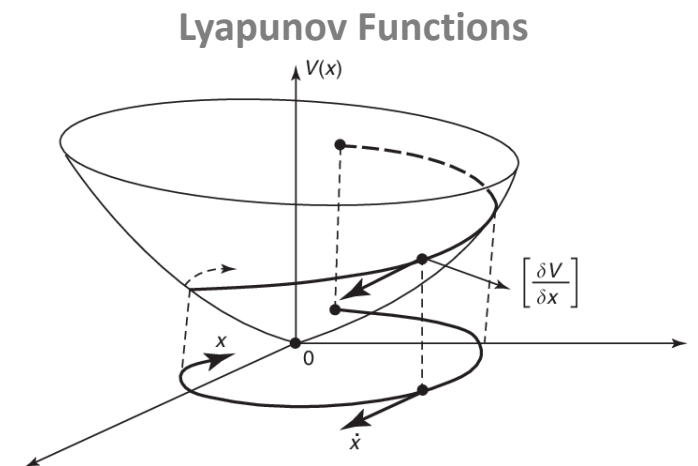
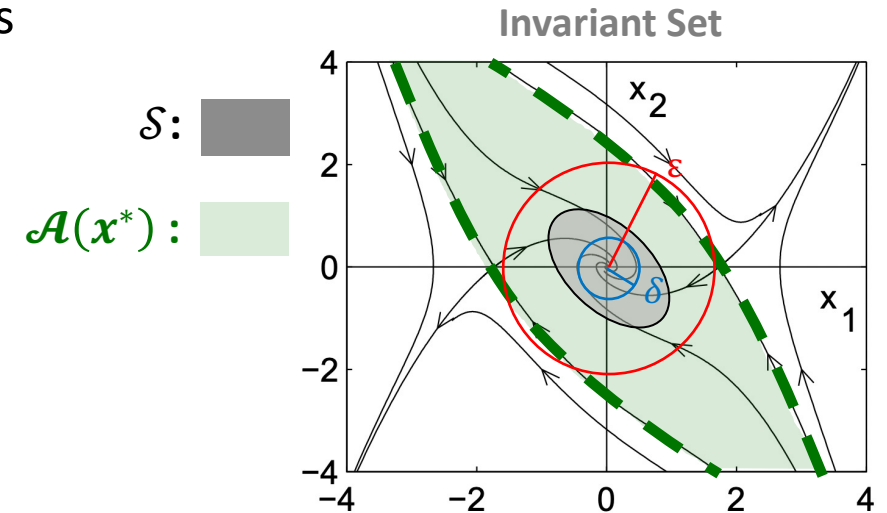
Compact invariant set $\mathcal{S}$, containing **only** $\{x^*\} = \Omega(f) \cap \mathcal{S}$ must be in the region of attraction $\mathcal{A}(x^*)$ ($\mathcal{S} \subset \mathcal{A}(x^*)$)

- Invariant sets guarantee stability**

Lyapunov stability: solutions starting "close enough" to the equilibrium (within a distance δ) remain "close enough" forever (within a distance ε)

- Invariant sets further certify asymptotic stability via Lyapunov's direct method**

Asymptotic stability: solutions that start close enough, remain close enough, and eventually converge to equilibrium.



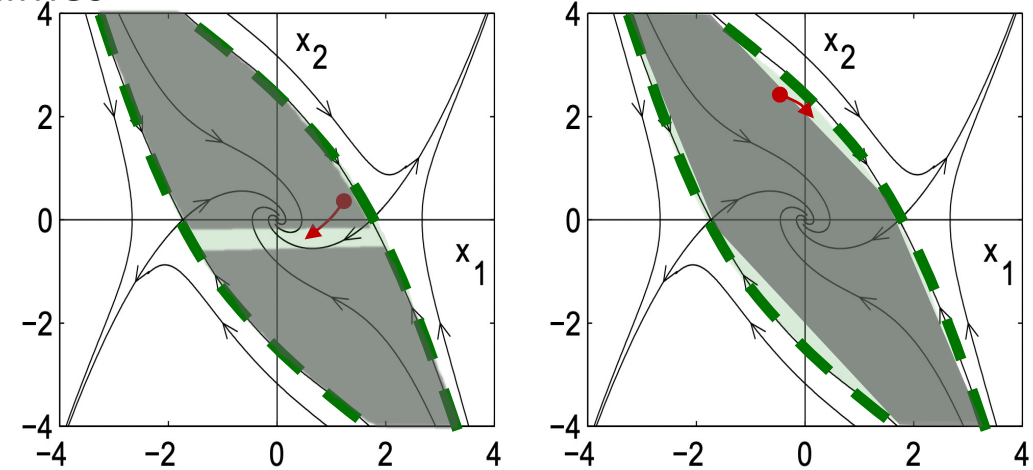
Invariant sets: Challenges

A set $\mathcal{S} \subseteq \mathbb{R}^d$ is **positively invariant** if and only if: $x_0 \in \mathcal{S} \rightarrow \phi(t, x_0) \in \mathcal{S}, \forall t \geq 0$

Any trajectory starting in the set remains in inside it for all times

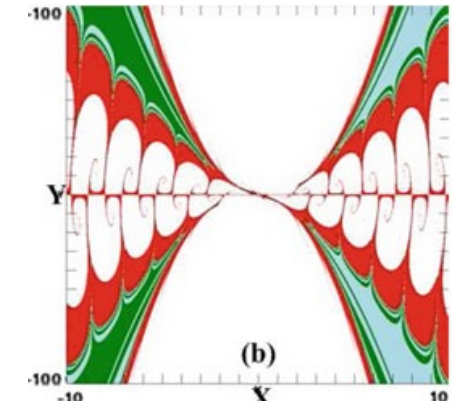
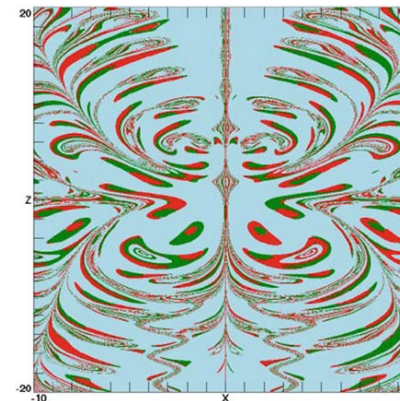
- $\mathcal{S}$ is topologically constrained
 - If $\mathcal{S} \cap \Omega(f) = \{x^*\}$, then $\mathcal{S}$ is connected
- $\mathcal{S}$ is geometrically constrained
 - f should not point outwards for $x \in \partial\mathcal{S}$
- $\mathcal{S}$ geometry can be wild
 - $\mathcal{A}(\Omega(f))$ can be fractal

$\mathcal{S}$: 
 $\mathcal{A}(x^*)$: 



A not invariant trajectory: 

Basin of $\Omega(f)$



Outline

- **Relaxing Invariance: Merits and trade offs**
 - *Recurrent Sets*: Letting thing go and come back
- **Nonparametric Analysis via Recurrent Sets**
 - *Stability analysis*: Recurrent Lyapunov Functions (RLFs)
 - *Safety verification*: Recurrent Barrier Functions (RBFs)
- **Self-Improving via Nonparametric Control Policies**
 - Policy Improvement using Expert Demonstrations

Outline

- **Relaxing Invariance: Merits and trade offs**
 - *Recurrent Sets*: Letting thing go and come back
- **Nonparametric Analysis via Recurrent Sets**
 - *Stability analysis*: Recurrent Lyapunov Functions (RLFs)
 - *Safety verification*: Recurrent Barrier Functions (RBFs)
- **Self-Improving via Nonparametric Control Policies**
 - Policy Improvement using Expert Demonstrations

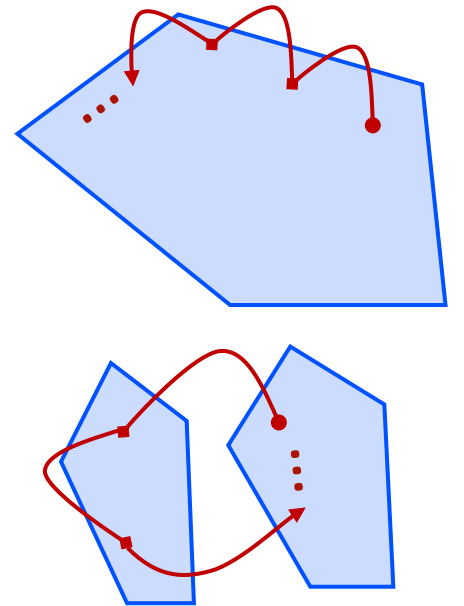
Recurrent sets: Letting things go, and come back

A set $\mathcal{R} \subseteq \mathbb{R}^d$ is **recurrent** if for any $x_0 \in \mathcal{R}$ and $t \geq 0$, $\exists t' \geq t$ s.t. $\phi(t', x_0) \in \mathcal{R}$.

Property of Recurrent Sets

- $\mathcal{R}$ need **not** be **connected**
- $\mathcal{R}$ does **not** require f to **point inwards** on all $\partial\mathcal{R}$

Recurrent sets, while not invariant,
guarantee that solutions that start in this set,
will come back **infinitely often, forever!**



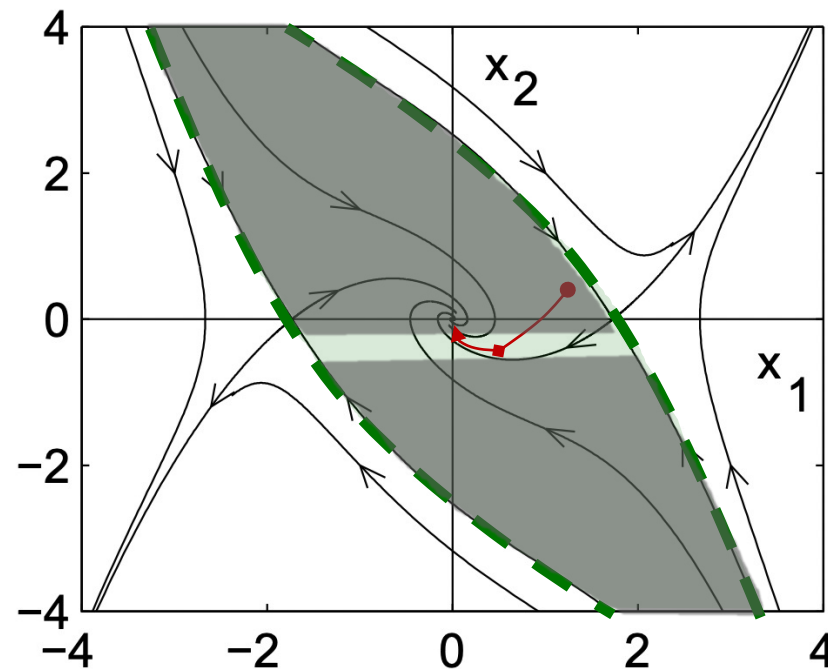
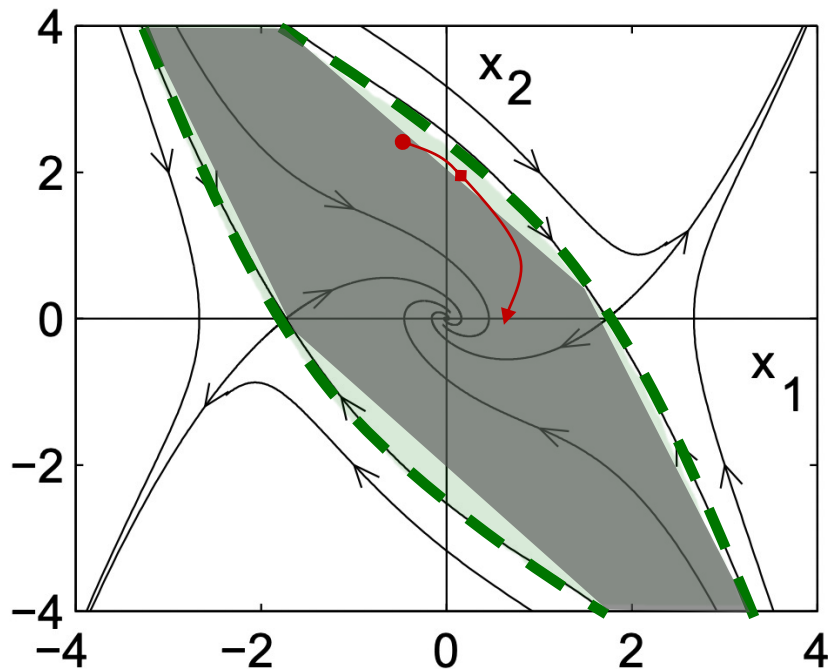
Recurrent set $\mathcal{R}$: 

A recurrent trajectory: 

Recurrent sets: Letting things go, and come back

A set $\mathcal{R} \subseteq \mathbb{R}^d$ is **recurrent** if for any $x_0 \in \mathcal{R}$ and $t \geq 0$, $\exists t' \geq t$ s.t. $\phi(t', x_0) \in \mathcal{R}$.

Previous two good inner approximations of $\mathcal{A}(x^*)$ are recurrent sets



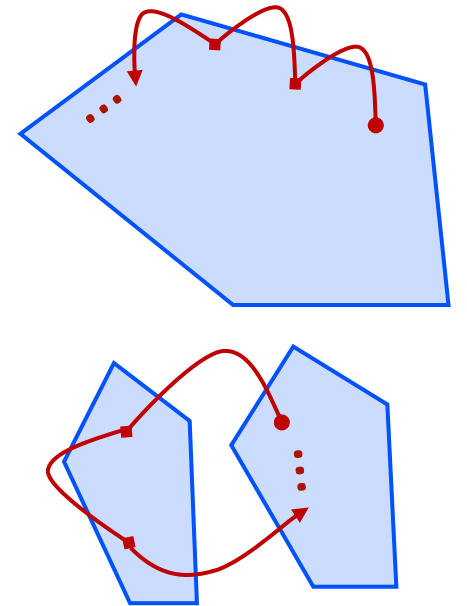
Recurrent sets: Letting things go, and come back

A set $\mathcal{R} \subseteq \mathbb{R}^d$ is **recurrent** if for any $x_0 \in \mathcal{R}$ and $t \geq 0$, $\exists t' \geq t$ s.t. $\phi(t', x_0) \in \mathcal{R}$.

Property of Recurrent Sets

- $\mathcal{R}$ need **not** be **connected**
- $\mathcal{R}$ does **not** require f to **point inwards** on all $\partial\mathcal{R}$

Recurrent sets, while not invariant,
guarantee that solutions that start in this set,
will come back **infinitely often, forever!**



Recurrent set $\mathcal{R}$: 

A recurrent trajectory: 

Question: Can we use recurrent sets as functional substitutes of invariant sets?

Outline

- **Relaxing Invariance: Merits and trade offs**
 - *Recurrent Sets*: Letting thing go and come back
- **Nonparametric Analysis via Recurrent Sets**
 - *Stability analysis*: Recurrent Lyapunov Functions (RLFs)
 - *Safety verification*: Recurrent Barrier Functions (RBFs)
- **Self-Improving via Nonparametric Control Policies**
 - Policy Improvement using Expert Demonstrations



Roy Siegelmann



Yue Shen



Fernando Paganini



Nonparametric Stability Analysis

R. Siegelmann, Y. Shen, F. Paganini, and E. Mallada, “A recurrence-based direct method for stability analysis and GPU-based verification of non-monotonic Lyapunov functions”, CDC 2023

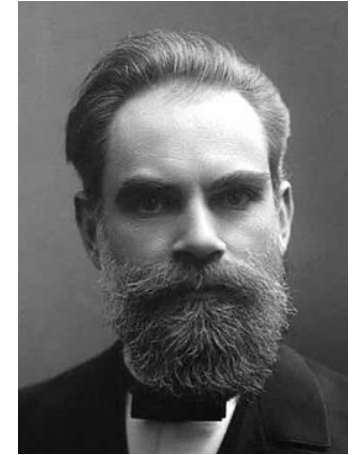
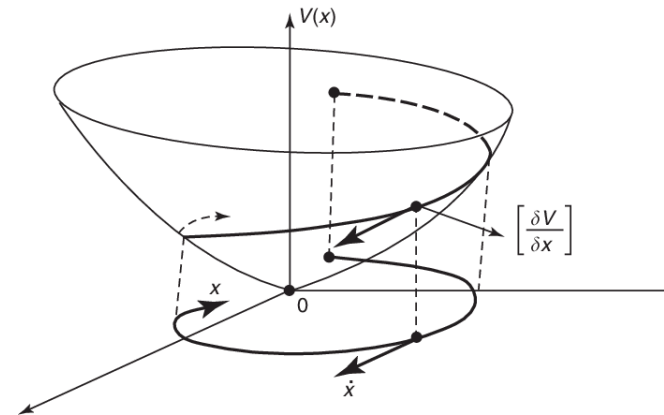
R. Siegelmann, Y. Shen, F. Paganini, and E. Mallada, “Recurrent Lyapunov Functions”, TAC 2025, submitted

Lyapunov's Direct Method

Key idea: Make sub-level sets invariant to trap trajectories

Theorem [Lyapunov '1892]. Given $V: \mathbb{R}^d \rightarrow \mathbb{R}_{\geq 0}$, with $V(x) > 0, \forall x \in \mathbb{R}^d \setminus \{x^*\}$, then:

- $\dot{V} \leq 0 \rightarrow x^*$ stable
- $\dot{V} < 0 \rightarrow x^*$ as. stable



Challenge: Couples shape of V and vector field f

- Towards decoupling the $V - f$ geometry
 - Controlling regions where $\dot{V} \geq 0$ [Karafyllis '09, Liu et al '20]
 - Higher order conditions: $g(V^{(q)}, \dots, \dot{V}, V) \leq 0$ [Butz '69, Gunderson '71, Ahmadi '06, Meigoli '12]
 - Discretization approach: $V(x(T)) \leq V(x(0))$ [Coron et al '94, Aeyels et. al '98, Karafyllis '12]
 - Multiple Lyapunov Functions: $\{V_j: j \in [k]\}$ [Ahmadi et al '14]

A Butz. Higher order derivatives of Lyapunov functions. IEEE Transactions on automatic control, 1969

Gunderson. A comparison lemma for higher order trajectory derivatives. Proceedings of the American Mathematical Society, 1971

Coron, Lionel Rosier. A relation between continuous time-varying and discontinuous feedback stabilization. J. Math. Syst., Estimation, Control, 1994

Aeyels, Peuteman. A new asymptotic stability criterion for nonlinear time-variant differential equations. IEEE Transactions on automatic control, 1998

Ahmadi. Non-monotonic Lyapunov functions for stability of nonlinear and switched systems: theory and computation, 2008

Karafyllis, Kravaris, Kalogerakis. Relaxed Lyapunov criteria for robust global stabilisation of non-linear systems. International Journal of Control, 2009

Meigoli, Nikravesh. Stability analysis of nonlinear systems using higher order derivatives of Lyapunov function candidates. Systems & Control Letters, 2012

Karafyllis. Can we prove stability by using a positive definite function with non sign-definite derivative? IMA Journal of Mathematical Control and Information, 2012

Ahmadi, Jungers, Parrilo, Roozbehani. Joint spectral radius and path-complete graph Lyapunov functions. SIAM Journal on Control and Optimization, 2014

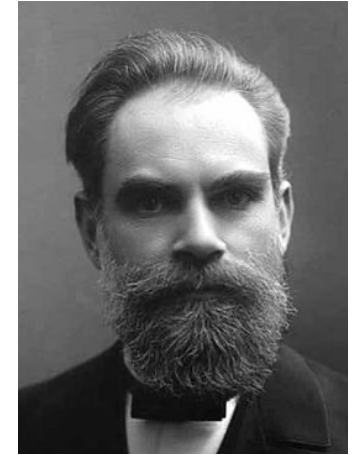
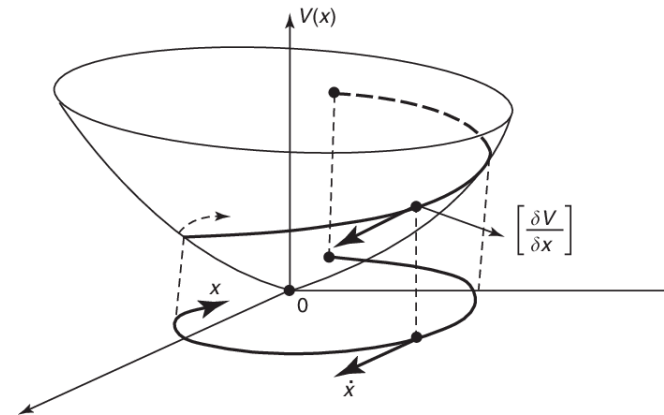
Liu, Liberzon, Zharnitsky. Almost Lyapunov functions for nonlinear systems. Automatica, 2020

Lyapunov's Direct Method

Key idea: Make sub-level sets invariant to trap trajectories

Theorem [Lyapunov '1892]. Given $V: \mathbb{R}^d \rightarrow \mathbb{R}_{\geq 0}$, with $V(x) > 0, \forall x \in \mathbb{R}^d \setminus \{x^*\}$, then:

- $\dot{V} \leq 0 \rightarrow x^*$ stable
- $\dot{V} < 0 \rightarrow x^*$ as. stable



Challenge: Couples shape of V and vector field f

- Towards decoupling the $V - f$ geometry
 - Controlling regions where $\dot{V} \geq 0$ [Karafyllis '09, Liu et al '20]
 - Higher order conditions: $g(V^{(q)}, \dots, \dot{V}, V) \leq 0$ [Butz '69, Gunderson '71, Ahmadi '06, Meigoli '12]
 - Discretization approach: $V(x(T)) \leq V(x(0))$ [Coron et al '94, Aeyels et. al '98, Karafyllis '12]
 - Multiple Lyapunov Functions: $\{V_j: j \in [k]\}$ [Ahmadi et al '14]

Question: Can we provide stability conditions based on recurrence?

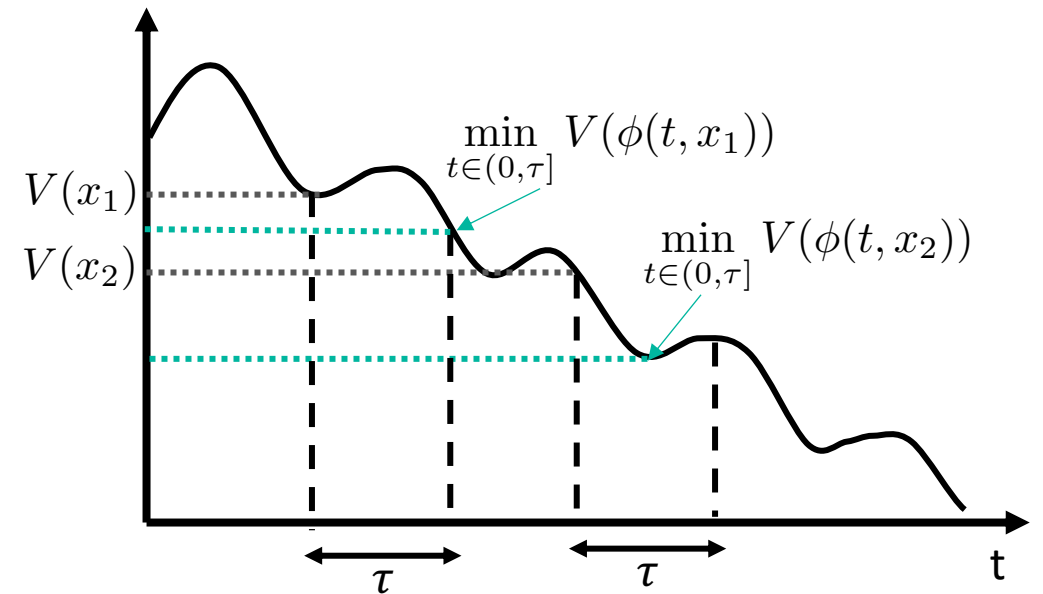
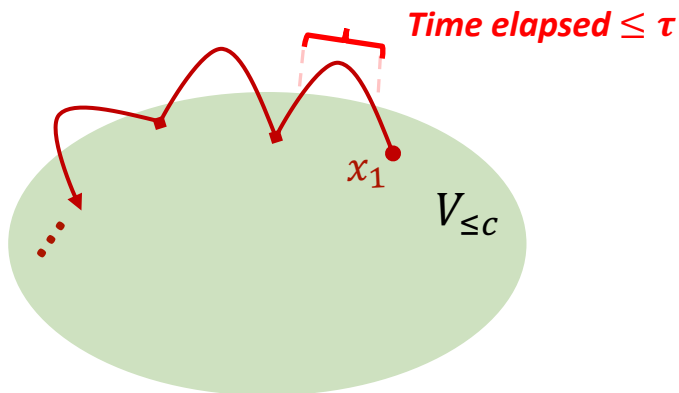
Recurrent Lyapunov Functions

A continuous function $V: \mathbb{R}^d \rightarrow \mathbb{R}_+$ is a **Recurrent Lyapunov Function** if

$$\mathcal{L}_f^{(0,\tau]} V(x) := \min_{t \in (0,\tau]} V(\phi(t, x)) - V(x) \leq 0 \quad \forall x \in \mathbb{R}^d$$

Preliminaries:

- Sub-level sets $\{V(x) \leq c\}$ are τ -recurrent sets.



Definition: A set $\mathcal{R} \subseteq \mathbb{R}^d$ is **τ -recurrent** if for any $x_0 \in \mathcal{R}$ and $t \geq 0$, $\exists t' \in (t, t + \tau]$ s.t. $\phi(t', x_0) \in \mathcal{R}$.

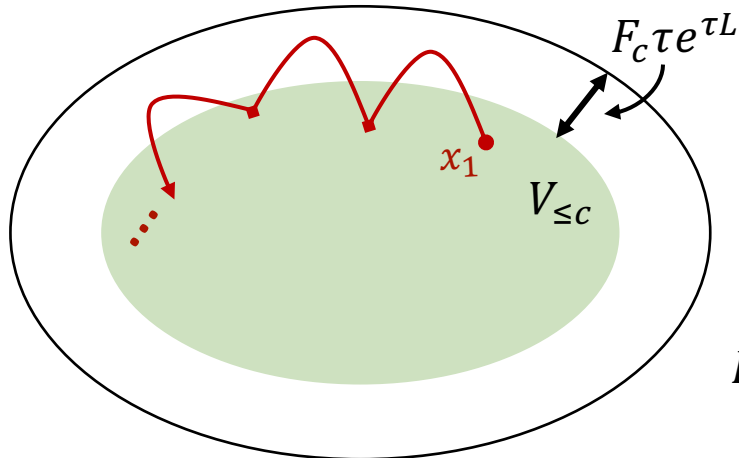
Recurrent Lyapunov Functions

A continuous function $V: \mathbb{R}^d \rightarrow \mathbb{R}_+$ is a **Recurrent Lyapunov Function** if

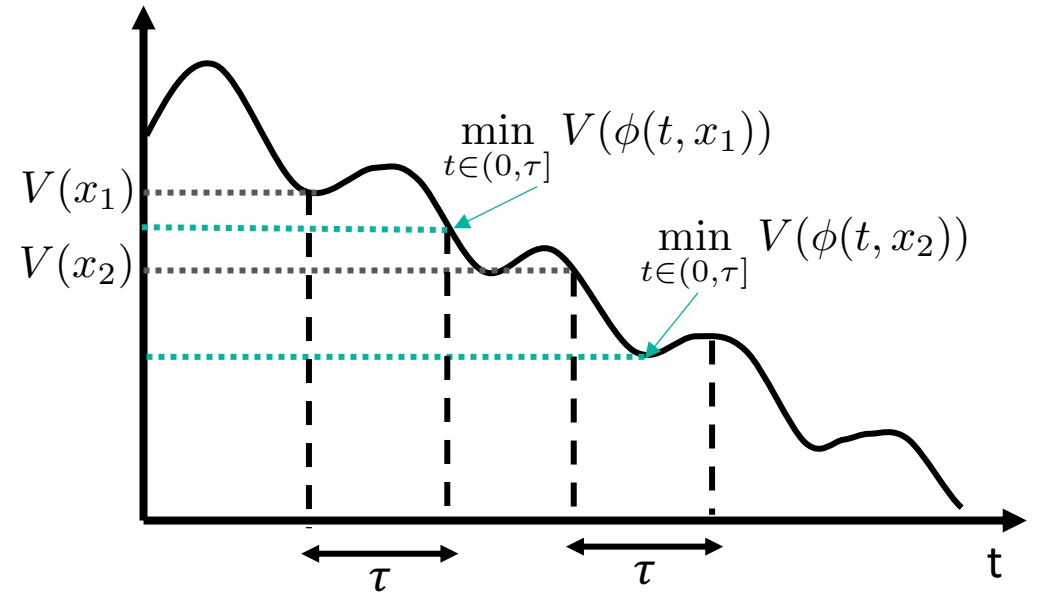
$$\mathcal{L}_f^{(0,\tau]} V(x) := \min_{t \in (0,\tau]} V(\phi(t, x)) - V(x) \leq 0 \quad \forall x \in \mathbb{R}^d$$

Preliminaries:

- Sub-level sets $\{V(x) \leq c\}$ are τ -recurrent sets.
- When f is L -Lipschitz, one can trap trajectories.



$$F_c = \max_{x \in V_{\leq c}} \|f(x)\|$$



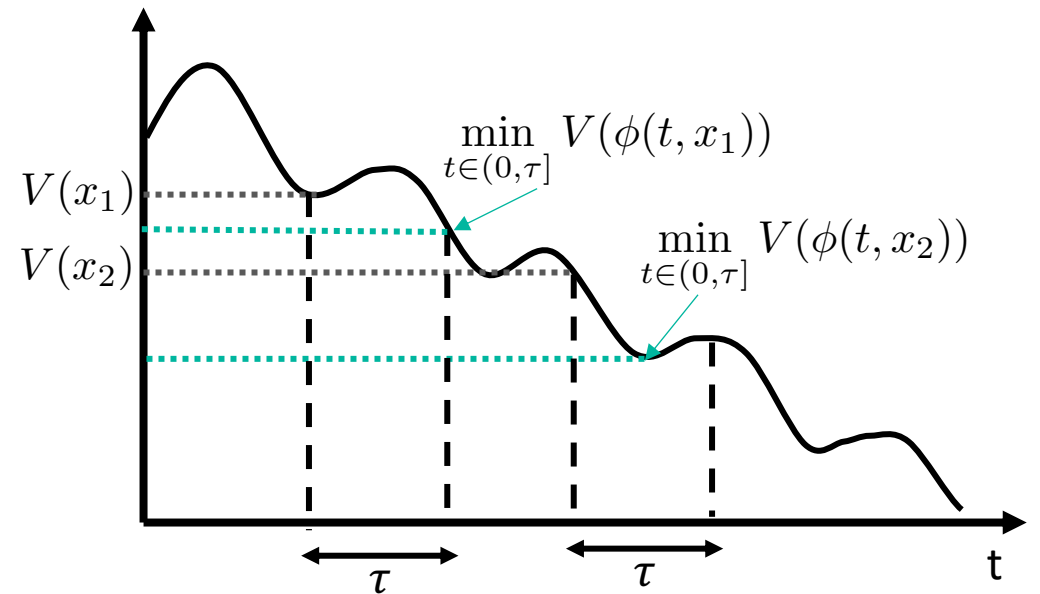
Recurrent Lyapunov Functions

A continuous function $V: \mathbb{R}^d \rightarrow \mathbb{R}_+$ is a **Recurrent Lyapunov Function** if

$$\mathcal{L}_f^{(0,\tau]} V(x) := \min_{t \in (0,\tau]} V(\phi(t, x)) - V(x) \leq 0 \quad \forall x \in \mathbb{R}^d$$

Theorem [CDC 23]: Let $V: \mathbb{R}^d \rightarrow \mathbb{R}_{\geq 0}$ be a Recurrent Lyapunov Function and let f be L -Lipschitz

- Then, the equilibrium x^* is stable.



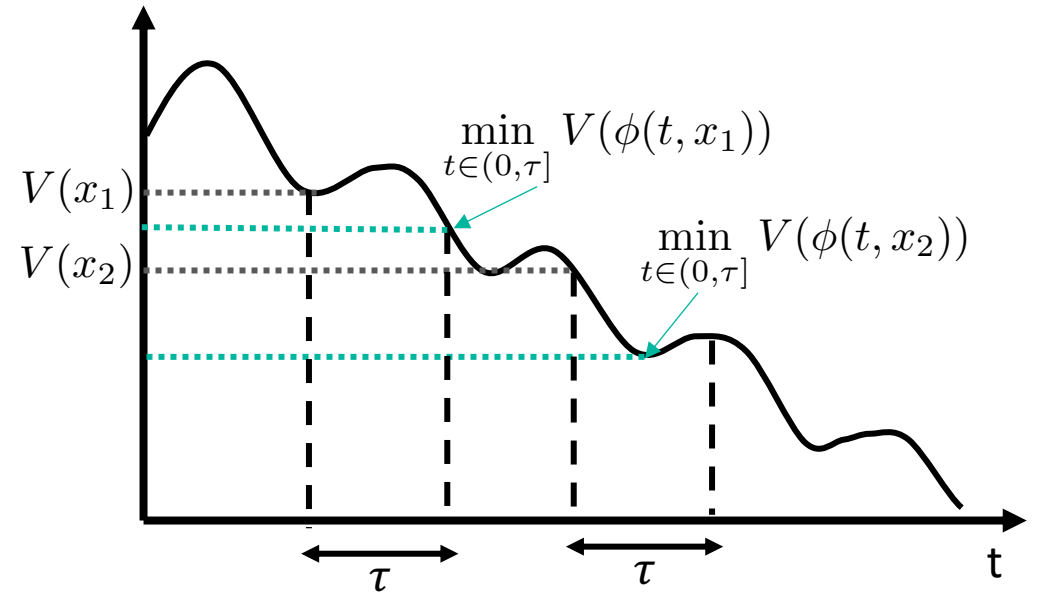
Recurrent Lyapunov Functions

A continuous function $V: \mathbb{R}^d \rightarrow \mathbb{R}_+$ is a **Recurrent Lyapunov Function** if

$$\mathcal{L}_f^{(0,\tau]} V(x) := \min_{t \in (0,\tau]} V(\phi(t, x)) - V(x) < 0 \quad \forall x \in \mathbb{R}^d$$

Theorem [CDC 23]: Let $V: \mathbb{R}^d \rightarrow \mathbb{R}_{\geq 0}$ be a Recurrent Lyapunov Function and let f be L -Lipschitz

- Then, the equilibrium x^* is stable.
- Further, if the **inequality is strict**, then x^* is asymptotically stable!



Exponential Stability Analysis

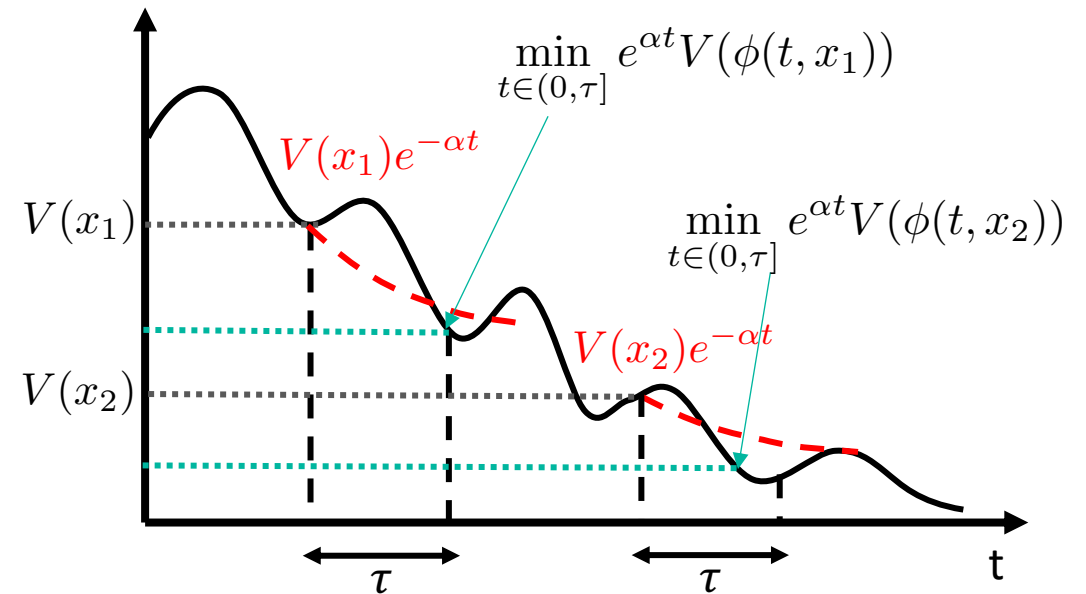
The function $V: \mathbb{R}^d \rightarrow \mathbb{R}_+$ is **α -Exponential Recurrent Lyapunov Function** if

$$\mathcal{L}_{f,\alpha}^{(0,\tau]} V(x) := \min_{t \in (0,\tau]} e^{\alpha t} V(\phi(t, x)) - V(x) \leq 0 \quad \forall x \in \mathbb{R}^d$$

Theorem [CDC 23]: Let $V: \mathbb{R}^d \rightarrow \mathbb{R}_{\geq 0}$ satisfy

$$\alpha_1 \|x - x^*\| \leq V(x) \leq \alpha_2 \|x - x^*\|.$$

Then, if V is **α -Exponential Recurrent Lyapunov Function**, x^* is **α -exponentially stable**.



Norm-based Converse Theorem

Theorem: Assume x^* is λ -exponentially stable: $\exists K, \lambda > 0$ such that:

$$||\phi(t, x) - x^*|| \leq K e^{-\lambda t} ||x - x^*||, \quad \forall x \in \mathbb{R}^d.$$

Then, $V(x) = ||x - x^*||$ is α -Exponential Recurrent Lyapunov Function, i.e.,

$$\min_{t \in (0, \tau]} e^{\alpha t} ||\phi(t, x) - x^*|| - ||x - x^*|| \leq 0, \quad \forall x \in \mathbb{R}^d,$$

whenever $\alpha < \lambda$ and $\tau \geq \frac{1}{\lambda - \alpha} \ln K$.

Remarks:

- The rate α must be strictly smaller than the rate of convergence λ (trading off optimality).
- Any norm is a Lyapunov function!

Question: Is the struggle for its search over?

Trajectory-based Verification

Proposition [CDC 23]: Let $||\cdot||$ be any norm and $x^* = 0$. Then, whenever

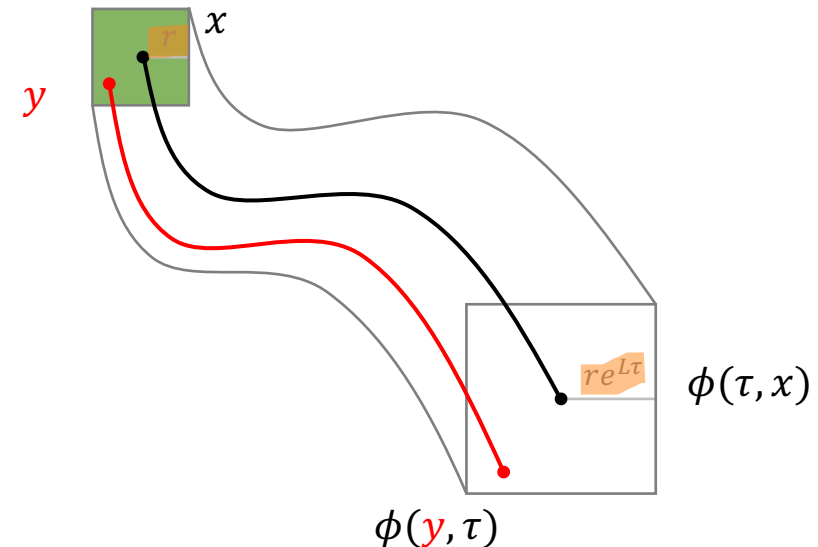
$$\min_{t \in (0, \tau]} e^{\alpha t} (||\phi(x, t)|| + r e^{L t}) \leq ||x|| - r$$

for all y with $||y - x|| \leq r$

$$\min_{t \in (0, \tau]} e^{\alpha t} ||\phi(y, t)|| \leq ||y||$$

Remarks:

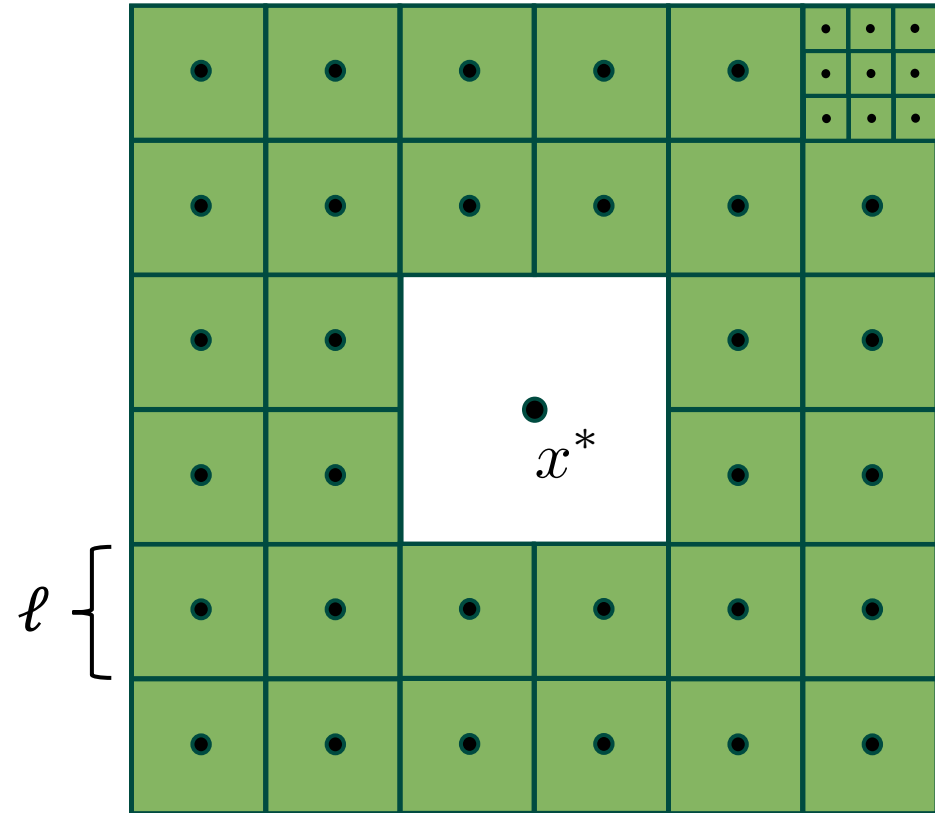
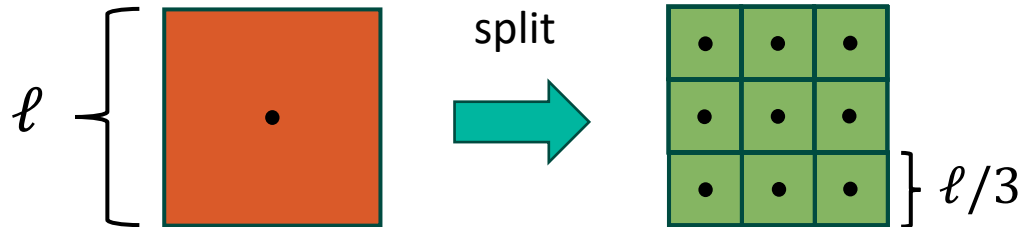
- Only requires a trajectory of length τ
- Trades off between **radius r** and verified **performance α**
- Amenable for parallel computations **using GPUs**



Nonparametric Stability Verification via GPUs

- **Basic Algorithm:**

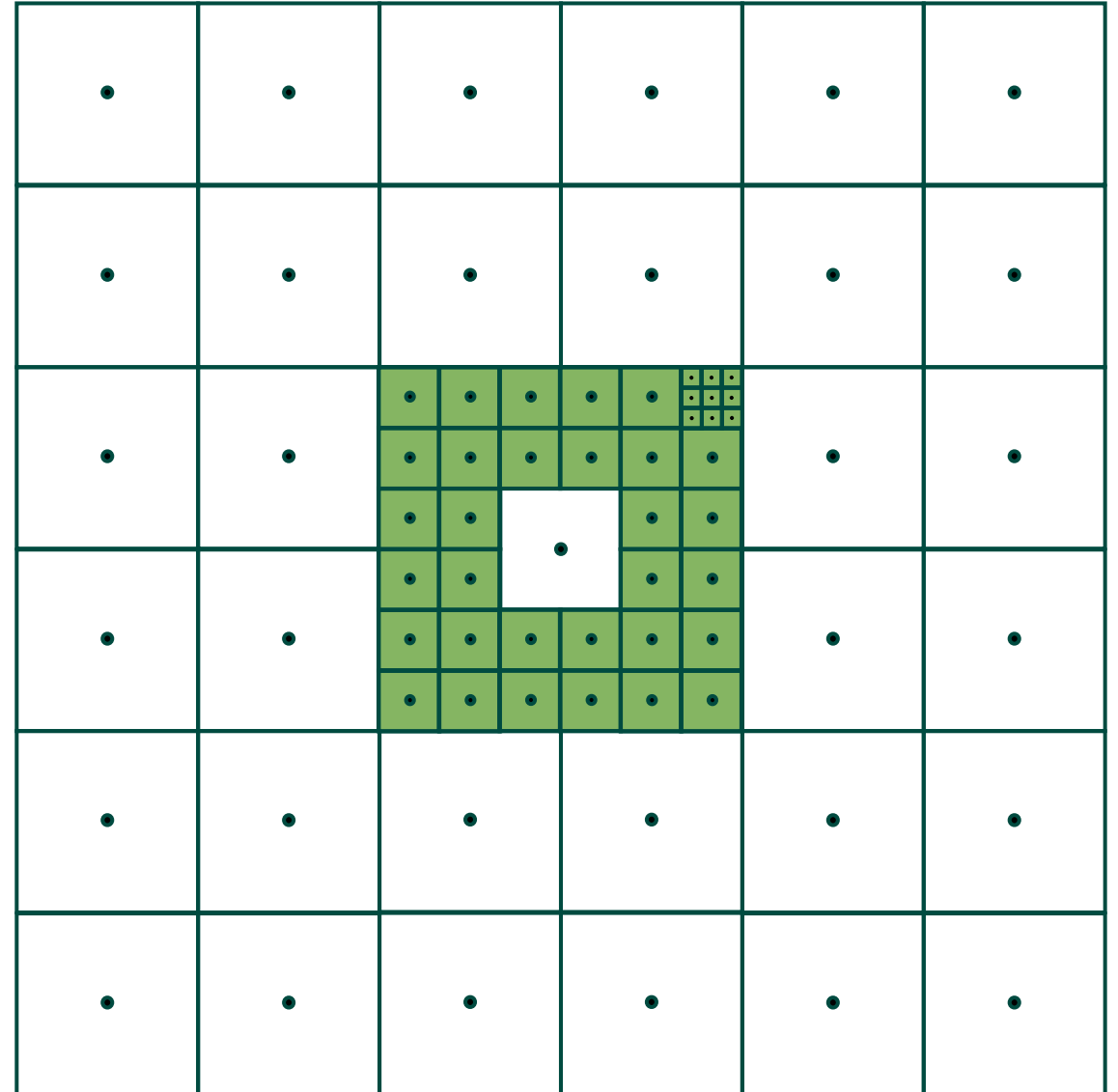
- Consider $V(x) = ||x - x^*||_\infty$
- Build a grid of hypercubes surrounding x^*
- Test grid center points:
 - Simulate trajectories of length τ
 - Find α s.t. the verified radius is $r \geq \ell/2$
- Hypercube **not verified**, **split in 3^d** parts
- Repeat testing of new points



Nonparametric Stability Verification via GPUs

- **Basic Algorithm:**

- Consider $V(x) = ||x - x^*||_\infty$
- Build a grid of hypercubes surrounding x^*
- Test grid center points:
 - Simulate trajectories of length τ
 - Find α s.t. the verified radius is $r \geq \ell/2$
- Hypercube **not verified, split in 3^d** parts
- Repeat testing of new points
- **Exponentially expand** to outer layer
- Repeat testing in new layer



Nonparametric Stability Verification via GPUs

- **Basic Algorithm:**

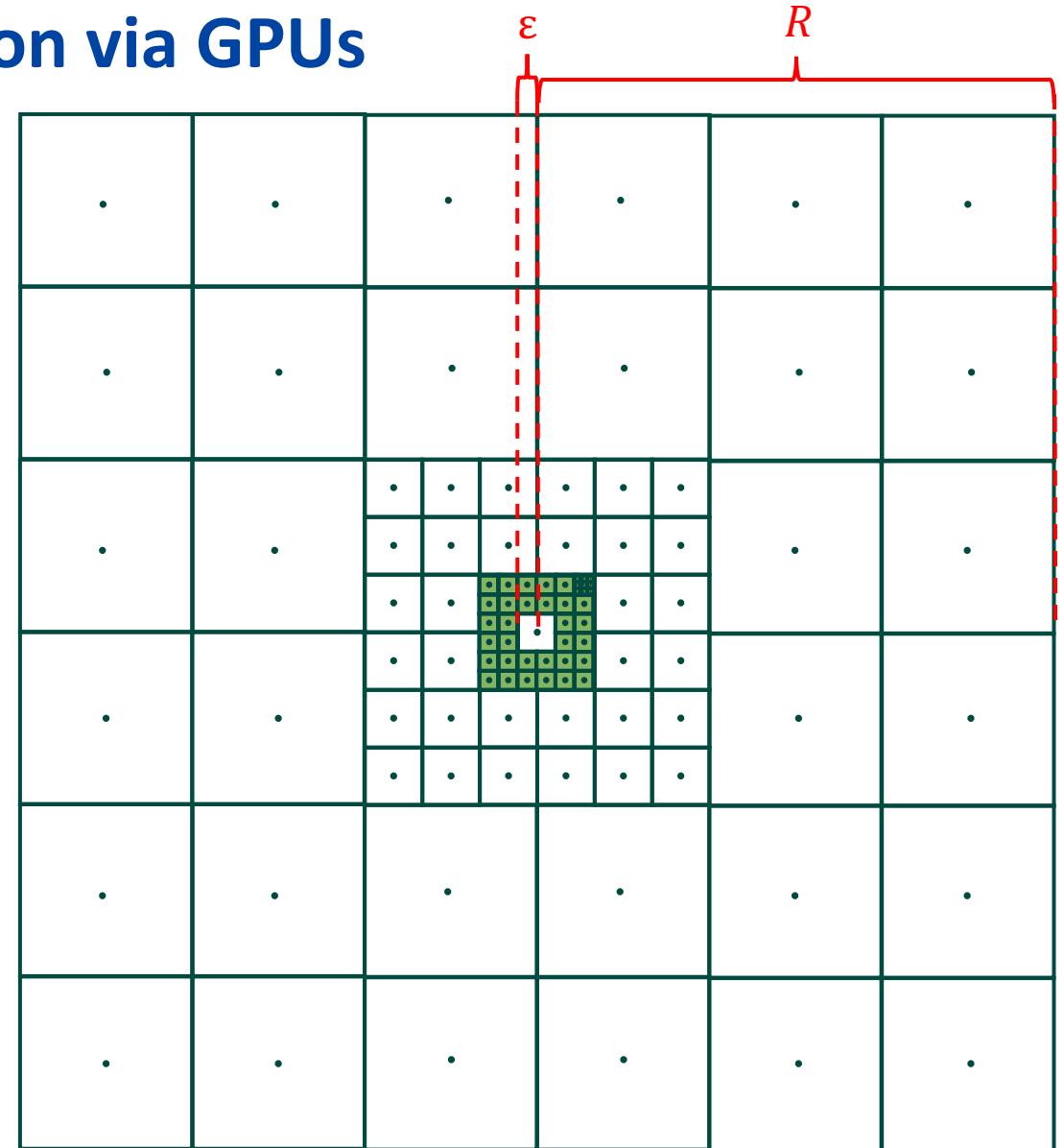
- Consider $V(x) = ||x - x^*||_\infty$
- Build a grid of hypercubes surrounding x^*
- Test grid center points:
 - Simulate trajectories of length τ
 - Find α s.t. the verified radius is $r \geq \ell/2$
- Hypercube **not verified**, **split in 3^d** parts
- Repeat testing of new points
- **Exponentially expand** to outer layer
- Repeat testing in new layer

Q: How many samples are needed?

If x^* is λ -exp. stable

$$\mathcal{O} \left(q^{-d} \log \left(\frac{R}{\varepsilon} \right) \right)$$

with $q = \frac{1 - Ke^{(\alpha - \lambda)\tau}}{1 + e^{(L + \alpha)\tau}} < 1$.



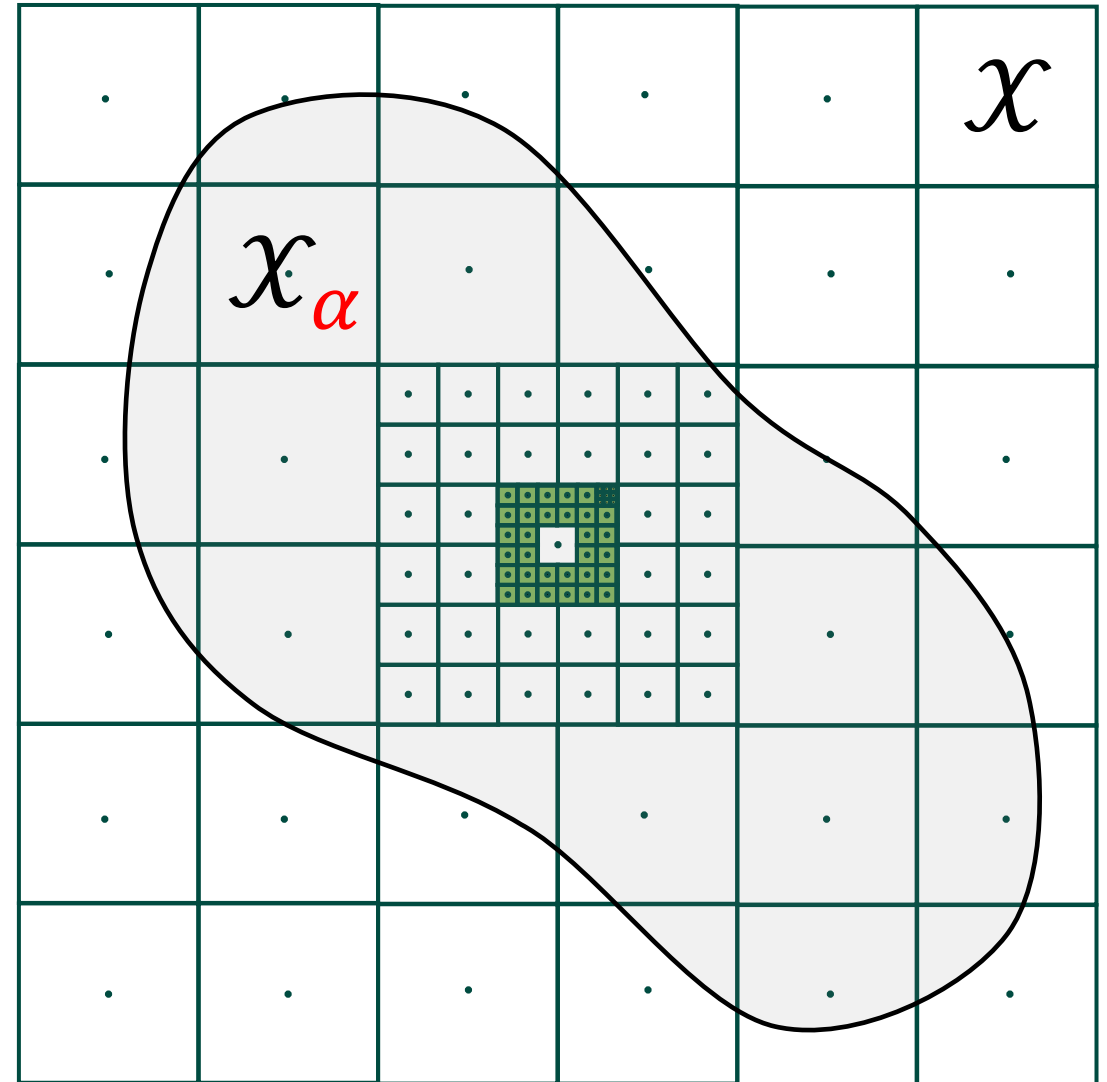
Nonparametric Stability Verification via GPUs

- **Basic Algorithm:**

- Consider $V(x) = ||x - x^*||_\infty$
- Build a grid of hypercubes surrounding x^*
- Test grid center points:
 - Simulate trajectories of length τ
 - Find α s.t. the verified radius is $r \geq \ell/2$
- Hypercube **not verified**, **split in 3^d** parts
- Repeat testing of new points
- **Exponentially expand** to outer layer
- Repeat testing in new layer

- **Two Alg. Variations:**

- Alg. 1: Find largest $\alpha_{\max}$ for region $\mathcal{X}$
- Alg. 2: Find region $\mathcal{X}_\alpha$ for given α



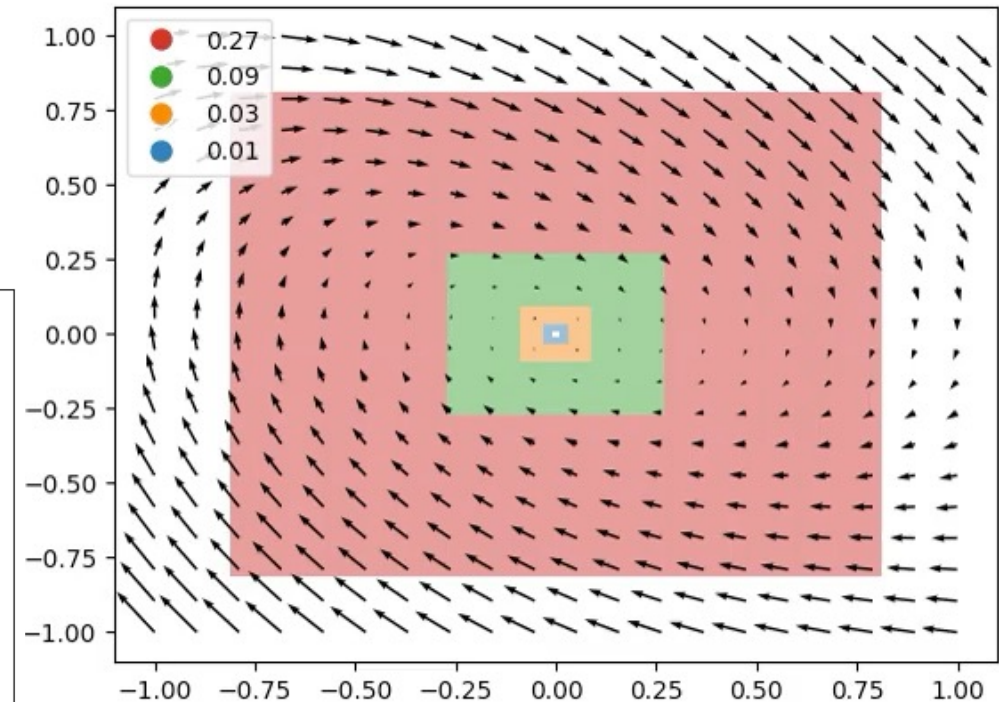
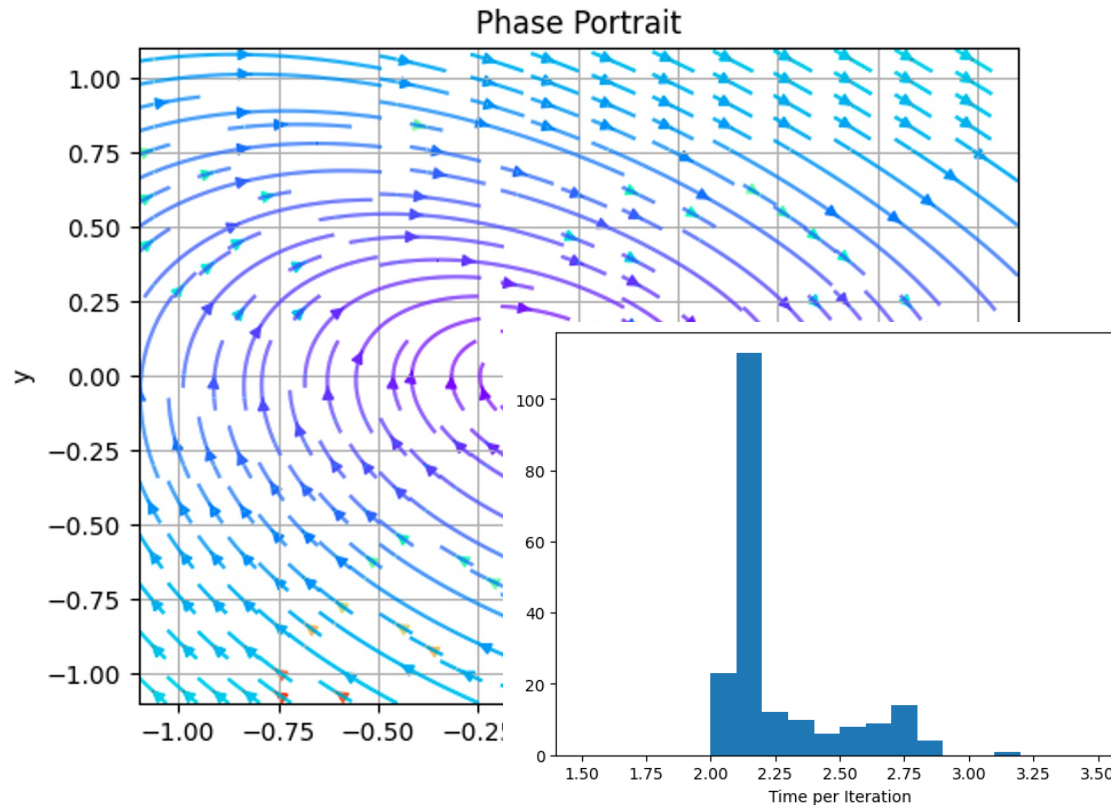
Numerical Illustration – Find Best α

Consider the 2-d non-linear system:
with $B_{ij} \sim \mathcal{N}(0, \sigma^2)$

$$\dot{x} = \begin{bmatrix} 0 & 2 \\ -1 & -1 \end{bmatrix} x + B \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix}$$

$$\sigma = 0.3$$

$$\alpha_{\max} = 0.470$$

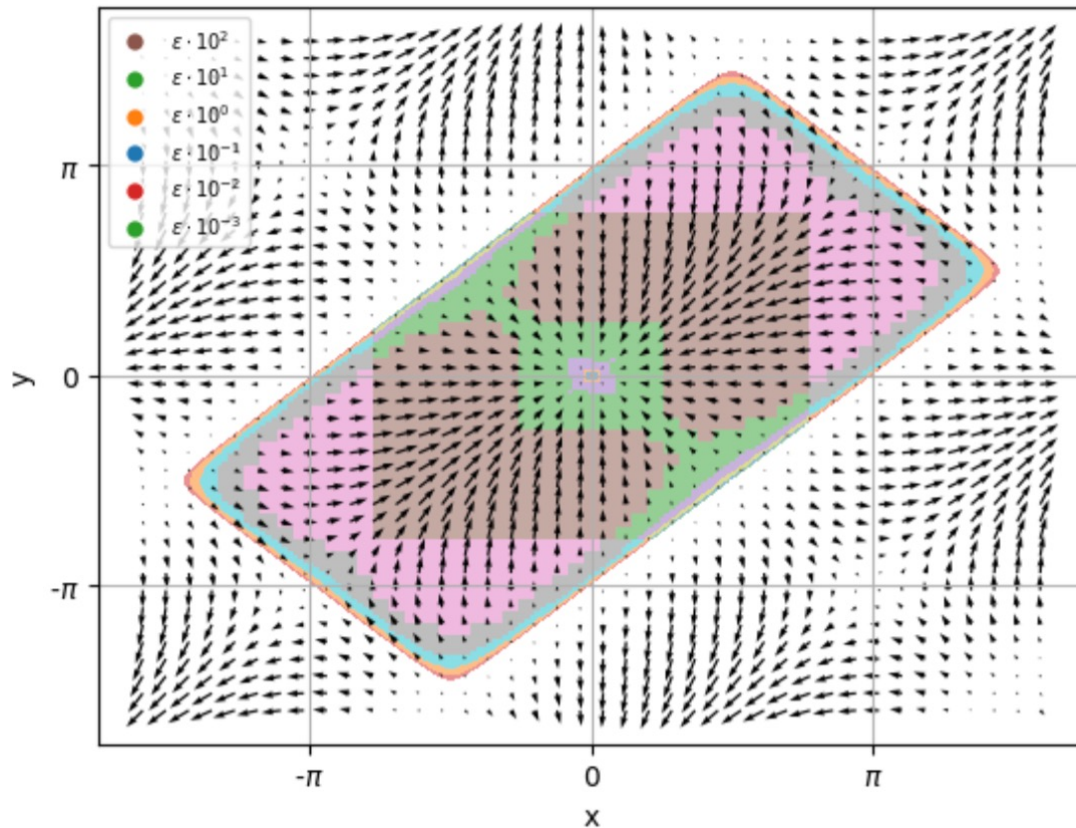


Numerical Illustration – Find region $\mathcal{X}_\alpha$

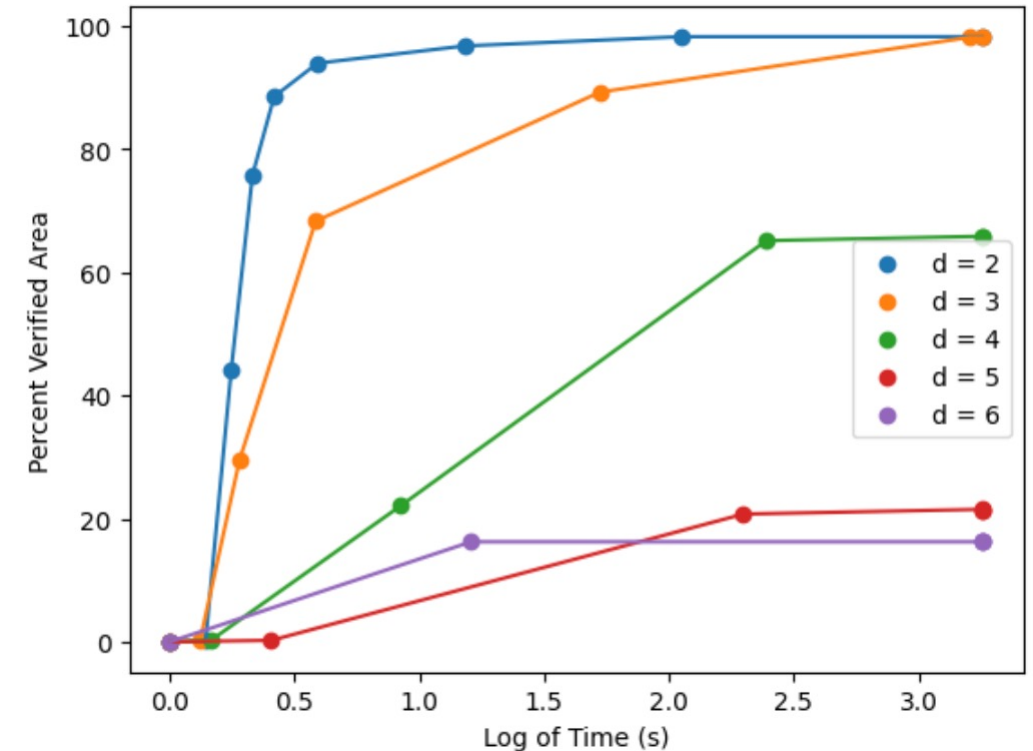
Consider the system of n Kuramoto oscillators:

$$\dot{\theta}_i = \frac{k}{n} \sum_{j=1}^n \sin(\theta_j - \theta_i)$$

Parameters: $n = 3$ and $\alpha = 1$



System dimension: $d = n - 1$



Outline

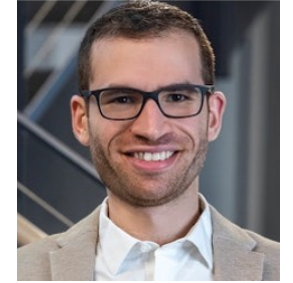
- **Relaxing Invariance: Merits and trade offs**
 - *Recurrent Sets*: Letting thing go and come back
- **Nonparametric Analysis via Recurrent Sets**
 - *Stability analysis*: Recurrent Lyapunov Functions (RLFs)
 - *Safety verification*: Recurrent Barrier Functions (RBFs)
- **Self-Improving via Nonparametric Control Policies**
 - Policy Improvement using Expert Demonstrations

Outline

- **Relaxing Invariance: Merits and trade offs**
 - *Recurrent Sets*: Letting thing go and come back
- **Nonparametric Analysis via Recurrent Sets**
 - *Stability analysis*: Recurrent Lyapunov Functions (RLFs)
 - *Safety verification*: Recurrent Barrier Functions (RBFs)
- **Self-Improving via Nonparametric Control Policies**
 - Policy Improvement using Expert Demonstrations



Yue Shen



Hussein Sibai



Nonparametric Safety Verification using Recurrence

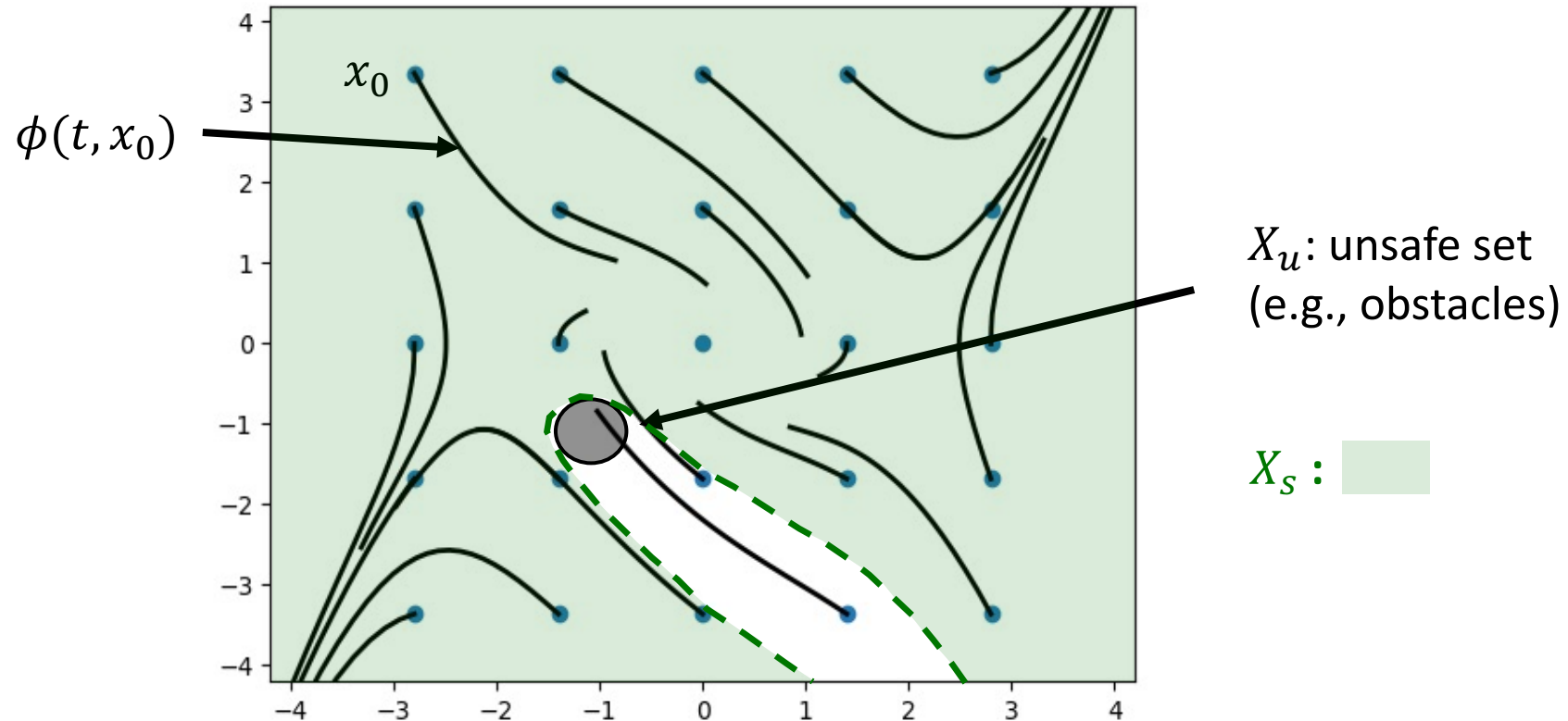
Y. Shen, H. Sibai, E. Mallada, “Generalized Barrier Functions: Integral Conditions and Recurrent Relaxations”, in 60th Allerton Conference on Communication, Control, and Computing 2024

Safety in Dynamical Systems

Consider the continuous-time dynamical system: $\dot{x} = f(x)$

- $\phi(t, x_0)$: solution at time t starting from x_0
- X_u : set of unsafe states

Goal: Find the safe set $\mathcal{X}_s := \{x_0 \in \mathbb{R}^d | \phi(t, x_0) \notin X_u, \forall t \geq 0\}$



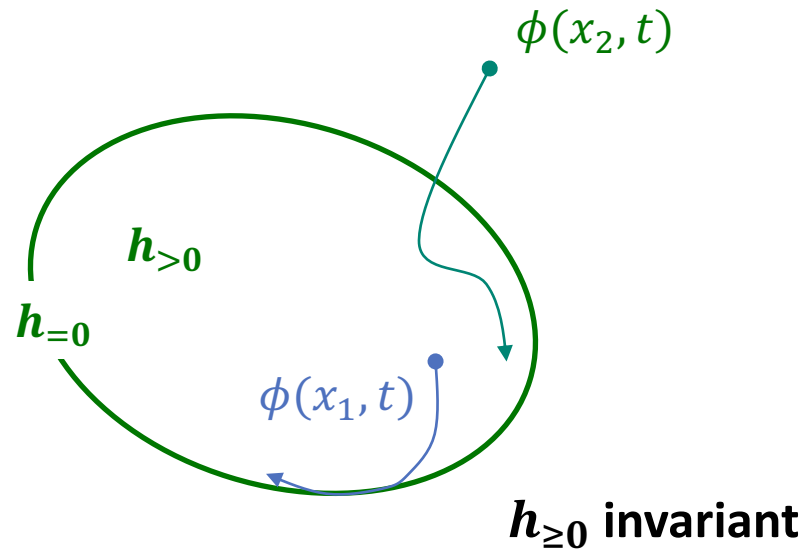
Barrier Functions

Barrier Function:

Let h be differentiable, $\gamma \in \mathcal{K}_e$, and

$$\dot{h}(x) \geq -\gamma(h(x))$$

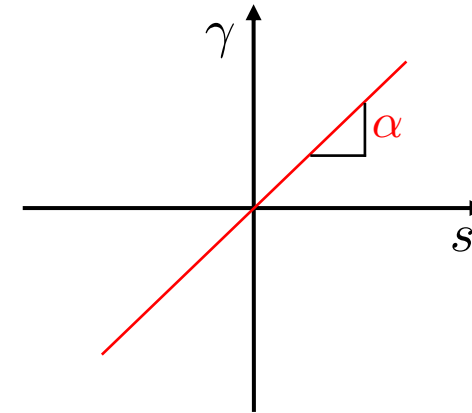
then, $h_{\geq 0}$ is invariant



Extended Class $\mathcal{K}_e$:

- $\gamma \in \mathcal{K}_e$ iff $\gamma'(s) \geq 0$ and $\gamma(0) = 0$

- Example:



Recurrent Barrier Functions

Barrier Function:

Let h be differentiable, $\gamma \in \mathcal{K}_e$, and

$$\dot{h}(x) + \gamma(h(x)) \geq 0$$

then, $h_{\geq 0}$ is invariant

Recurrent Barrier Function:

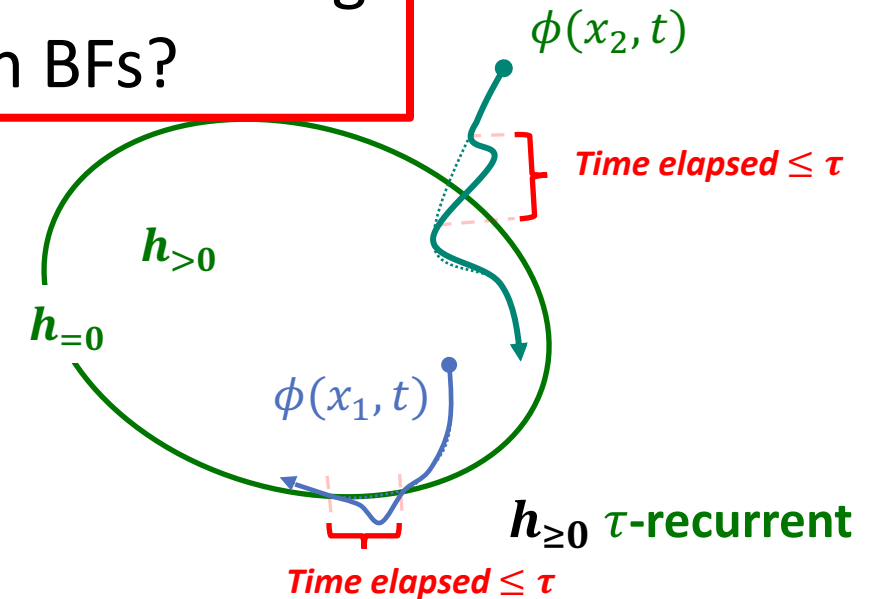
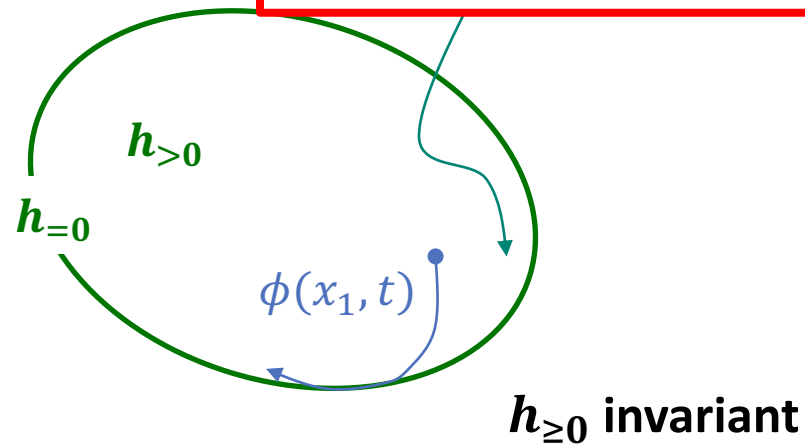
Let h be continuous, $\gamma \in \mathcal{K}_e$, and

$$\max_{t \in (0, \tau]} h(\phi(x, t)) + \int_0^t \gamma(h(\phi(x, s))) ds \geq h(x)$$

then, $h_{\geq 0}$ is **τ -recurrent**

As $\tau \rightarrow 0$
By definition

Question: Do we gain anything from relaxing the invariance condition in BFs?



Benefits of Recurrent Barrier Functions (RBFs)

Claim 1: Signed norms are RBFs!

Under mild* conditions:

- If there exists a **Barrier Function** h
- Then for any set S with $h_{\geq 0} \subseteq S \subseteq h_{\geq -c}$

$$\hat{h}(x) := -\text{sd}(x, S)$$

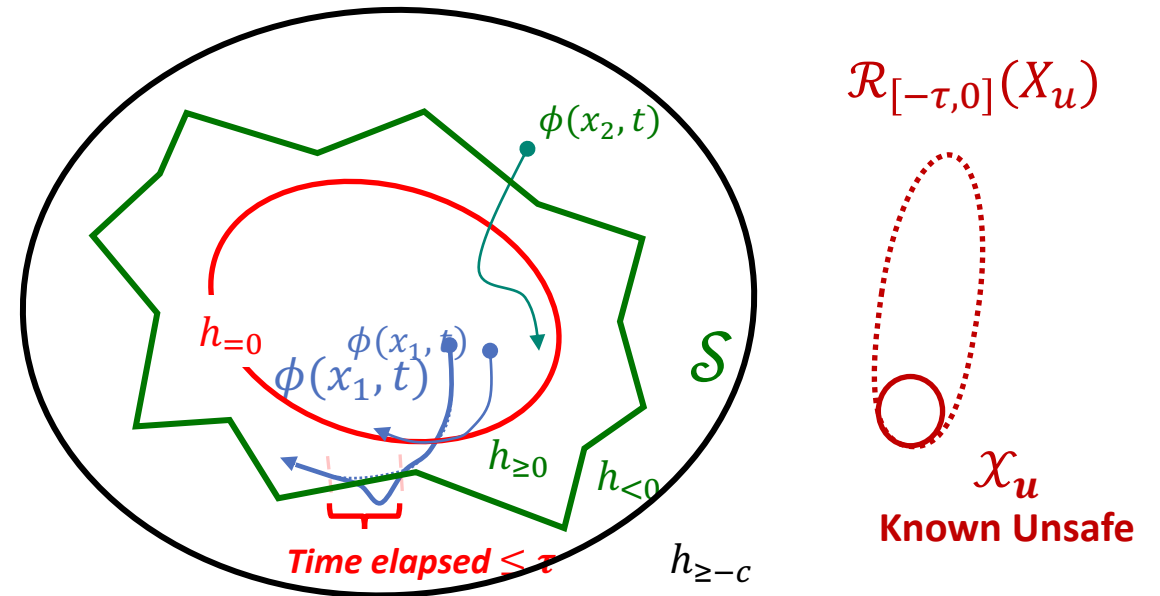
is a **Recurrent Barrier Function**

Claim 2: Guaranteeing Safety via RBFs

Whenever the following holds:

$$S \cap \mathcal{R}_{[-\tau, 0]}(X_u) = \emptyset$$

Then, S is a safe set



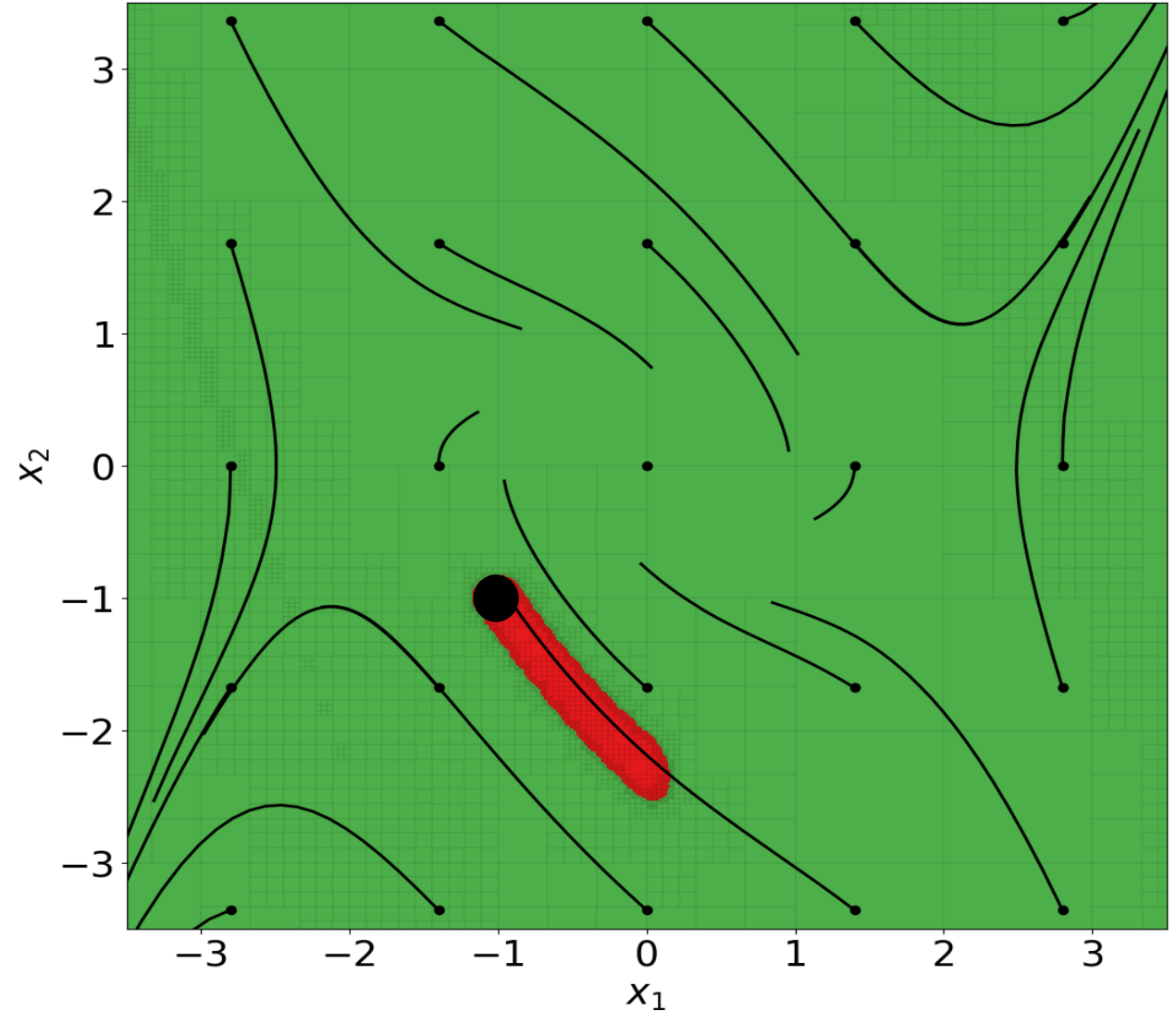
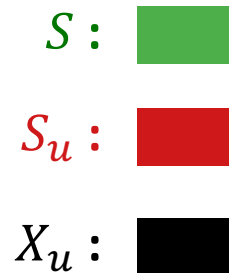
Data-Driven Safety Verification

Stage 1: τ —Backward reachability

- Find S_u with $\mathcal{R}_{[-\tau,0]}(X_u) \subset S_u$

Stage 2: RBF condition

- Check $h(x) = -\text{sd}(x, S)$ is RBF



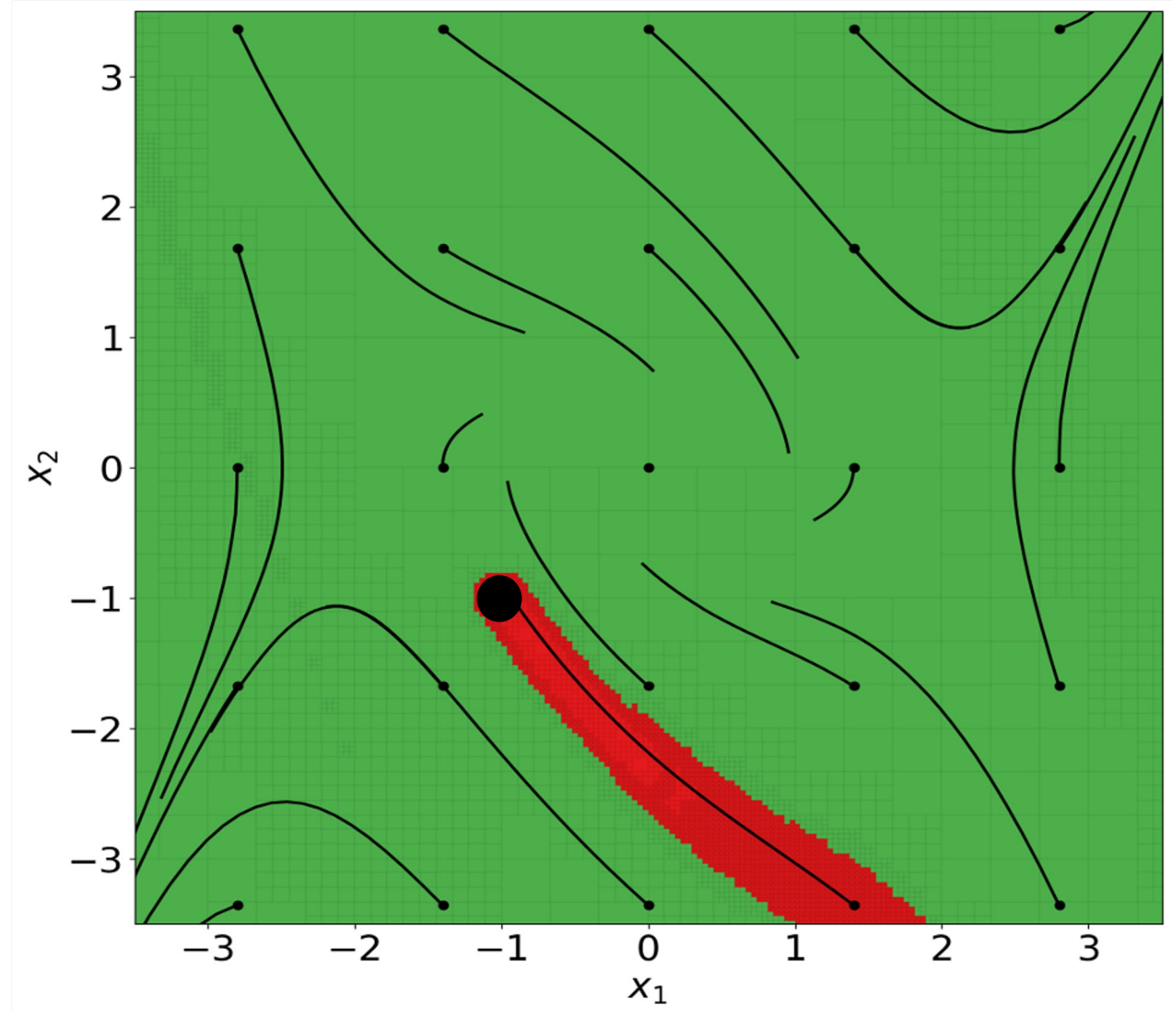
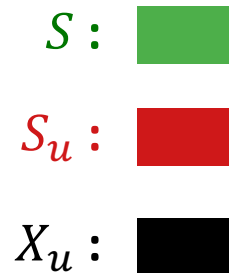
Data-Driven Safety Verification

Stage 1: τ —Backward reachability

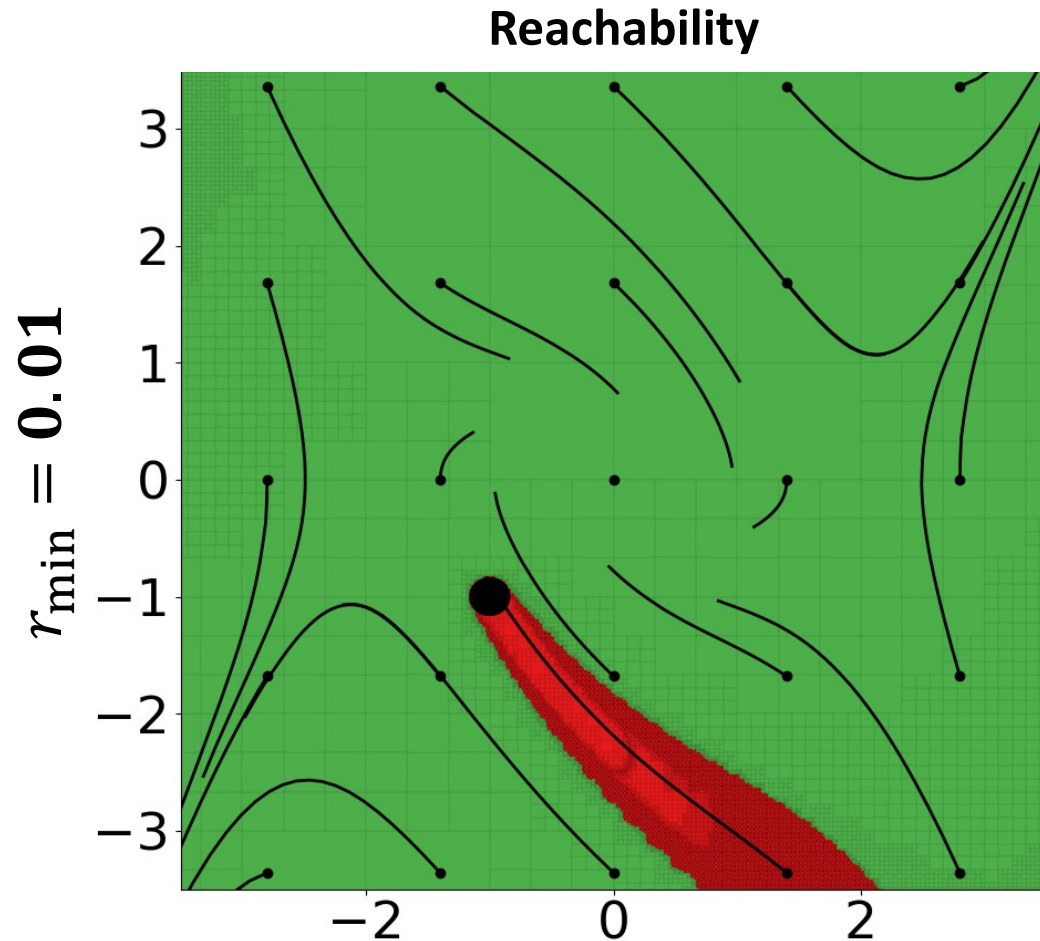
- Find S_u with $\mathcal{R}_{[-\tau,0]}(X_u) \subset S_u$

Stage 2: RBF condition

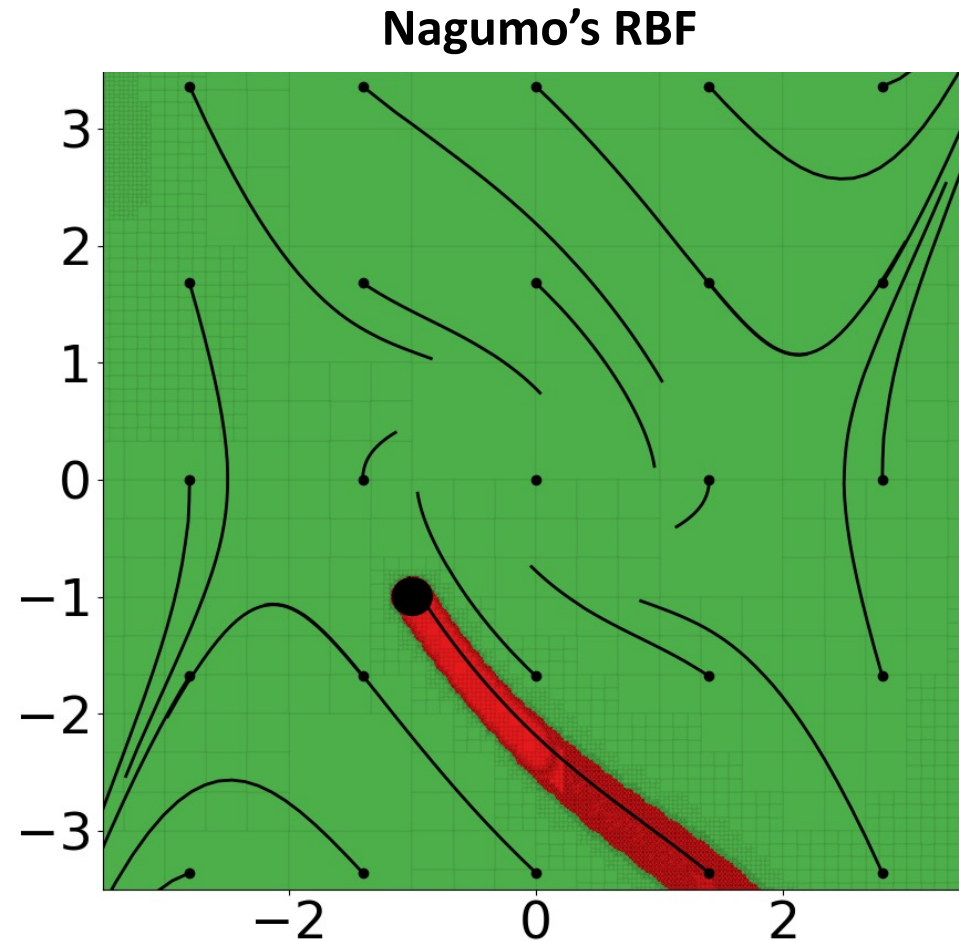
- Check $h(x) = -\text{sd}(x, S)$ is RBF



Numerical Validation: Reachability vs Recurrence



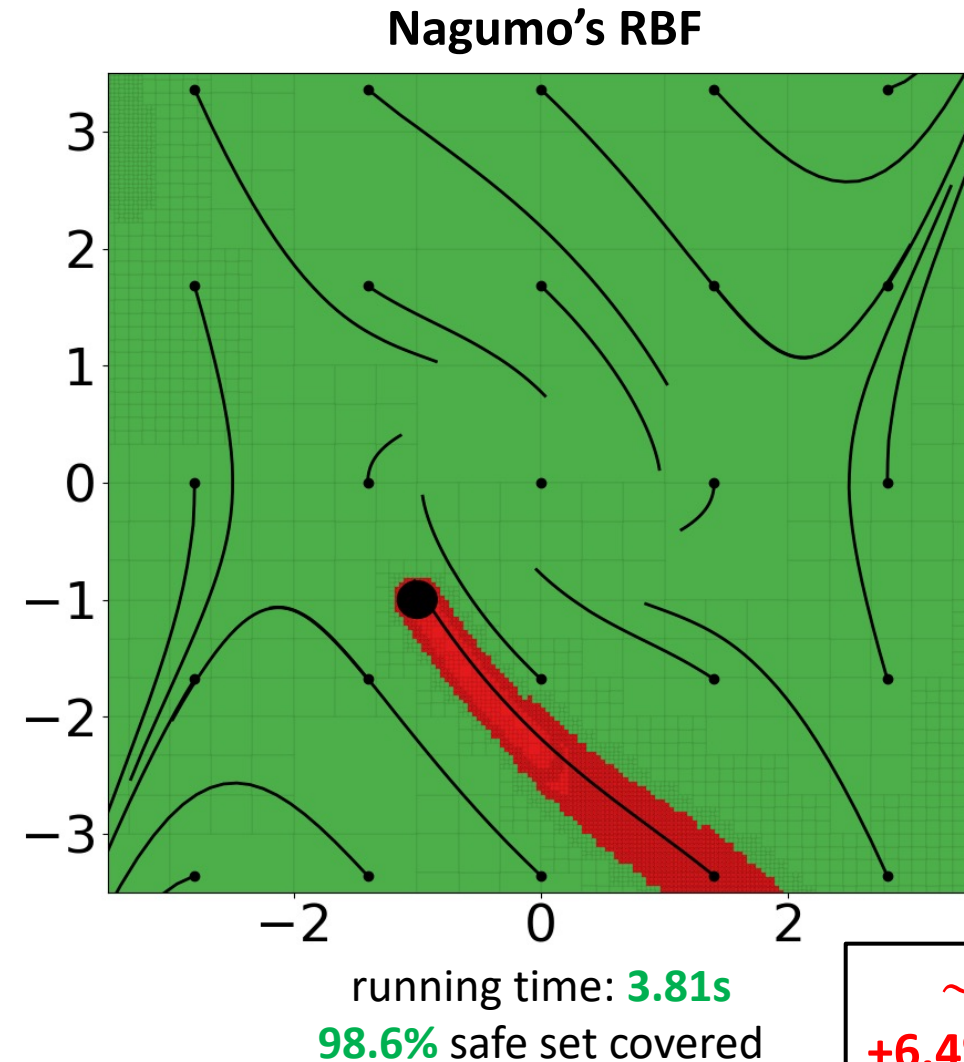
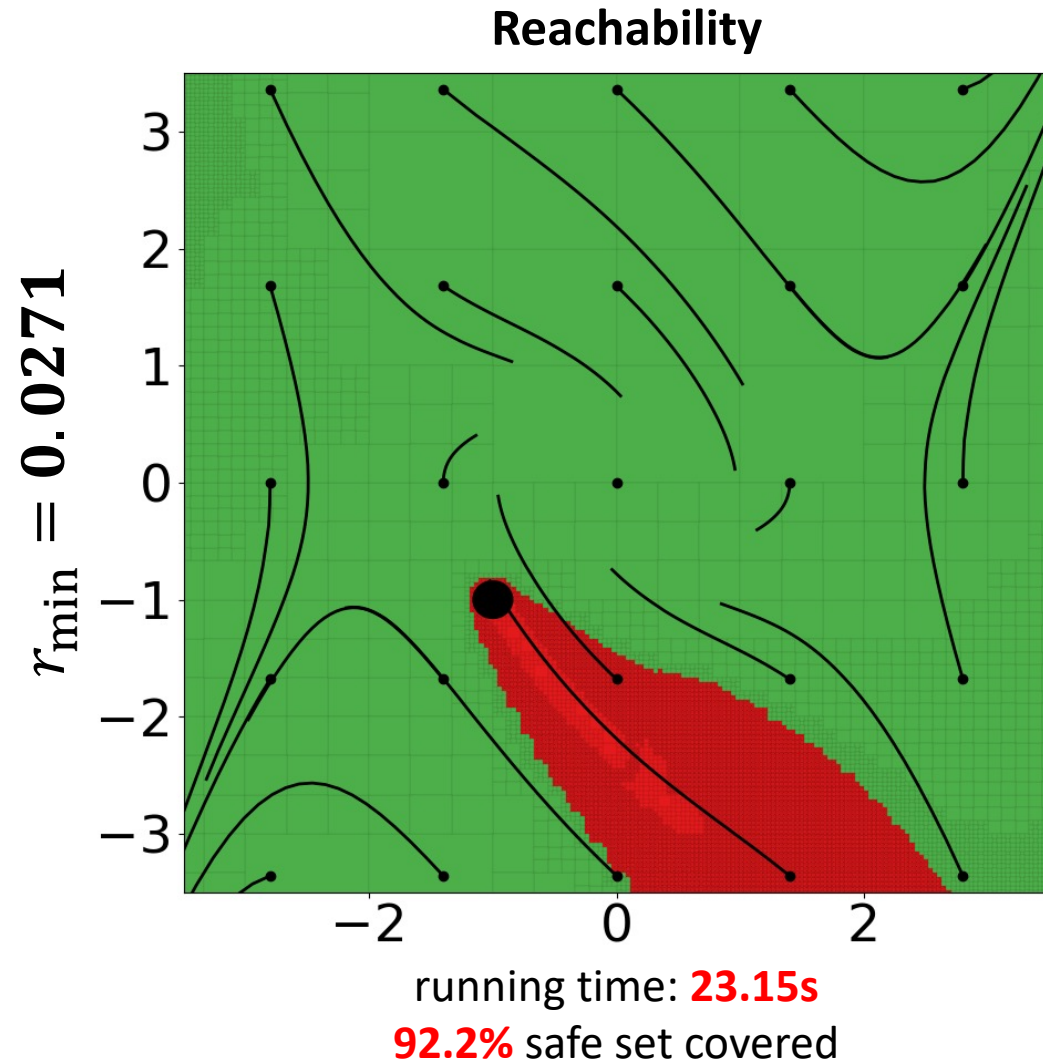
running time: **354.12s**
97.4% safe set covered



running time: **20.68s**
99.6% safe set covered

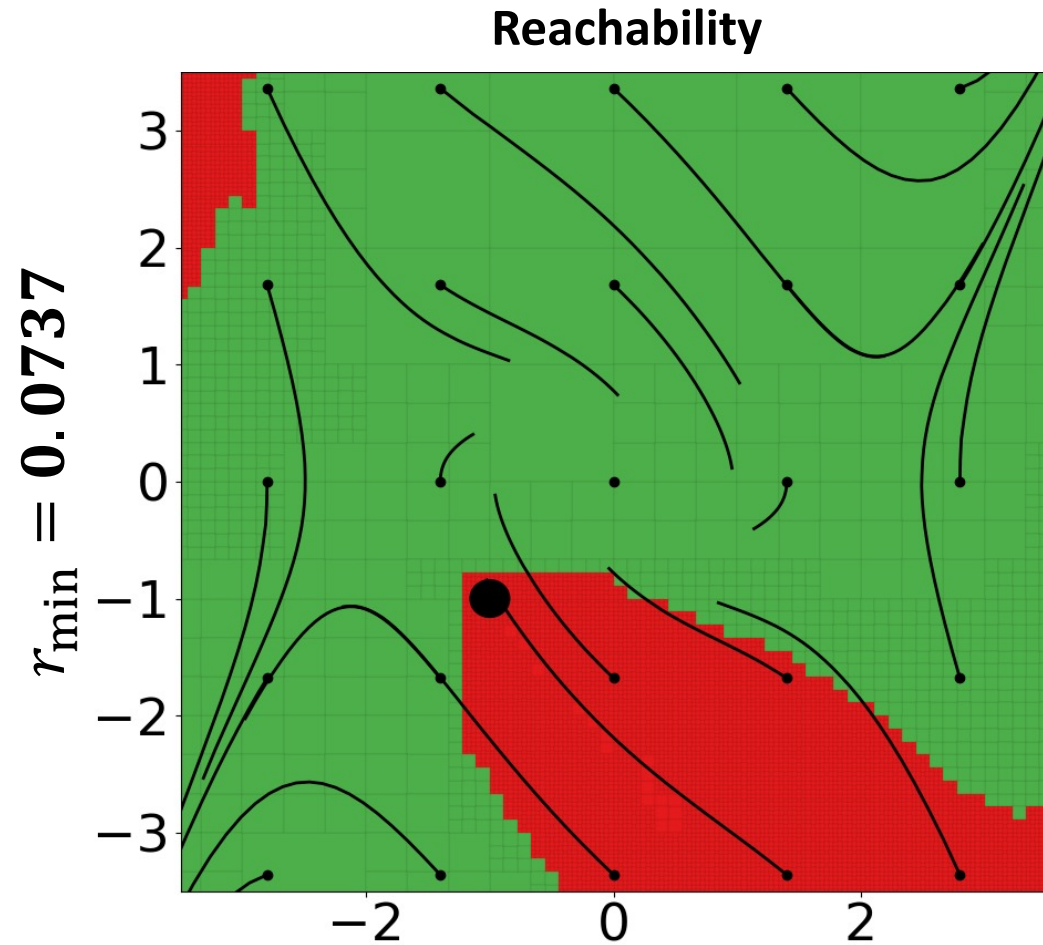
~17x faster
+2.2% more area

Numerical Validation: Reachability vs Recurrence

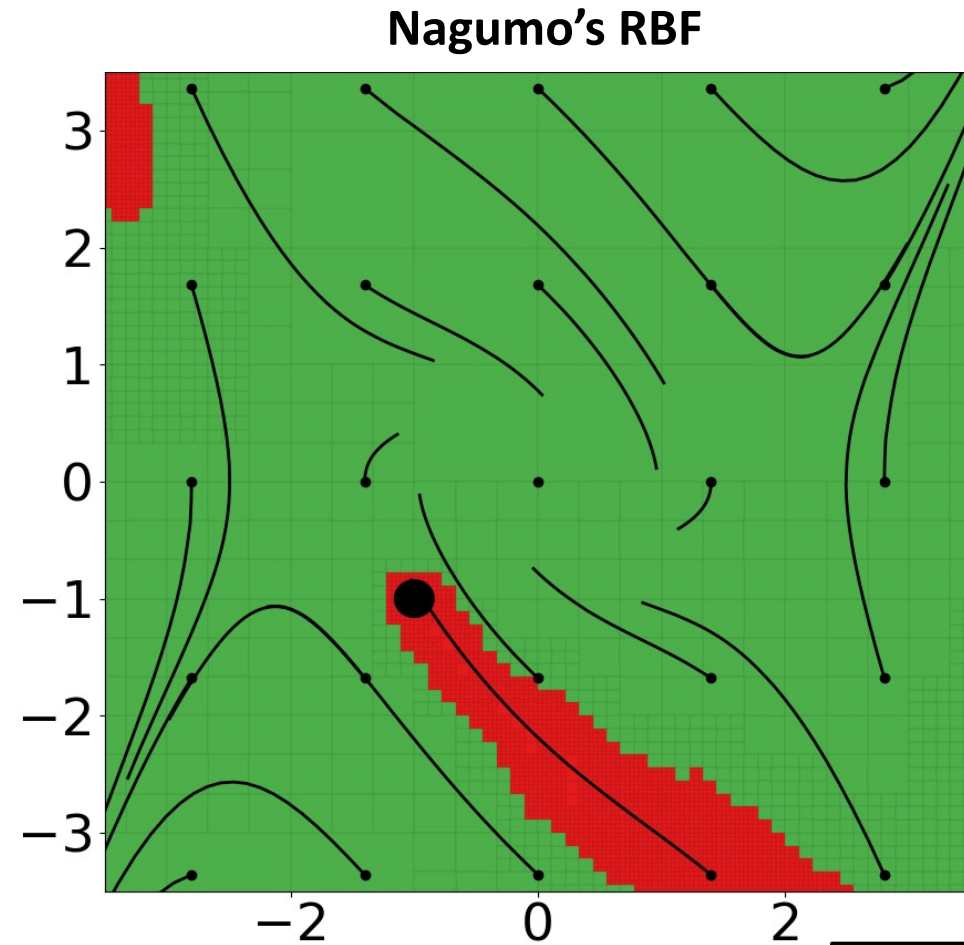


~6x faster
+6.4% more area

Numerical Validation: Reachability vs Recurrence



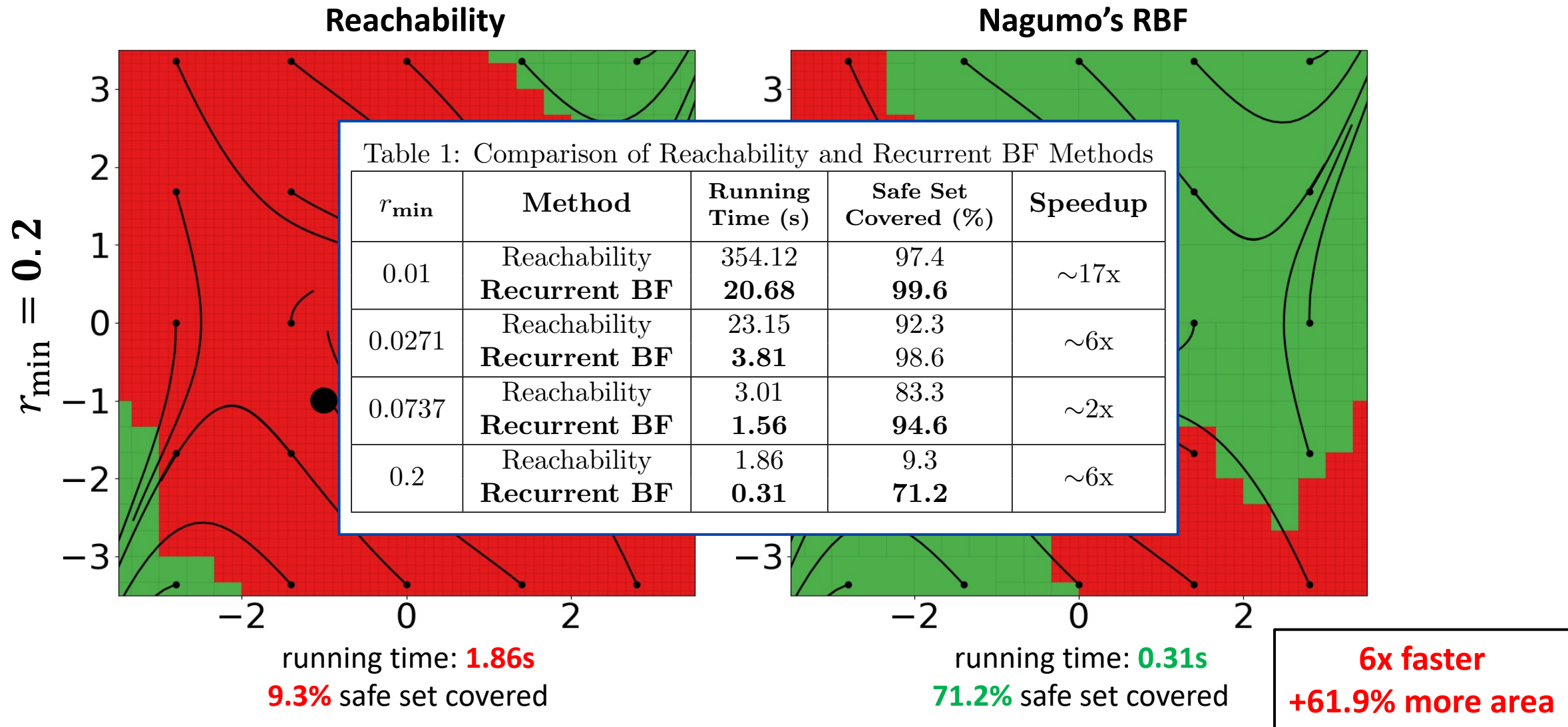
running time: **3.01s**
83.3% safe set covered



running time: **1.56s**
94.6% safe set covered

~2x faster
+11.2% more area

Numerical Validation: Reachability vs Recurrence



Outline

- **Relaxing Invariance: Merits and trade offs**
 - *Recurrent Sets*: Letting thing go and come back
- **Nonparametric Analysis via Recurrent Sets**
 - *Stability analysis*: Recurrent Lyapunov Functions (RLFs)
 - *Safety verification*: Recurrent Barrier functions (RBFs)
- **Self-Improving via Nonparametric Control Policies**
 - Policy Improvement using Expert Demonstrations

Outline

- **Relaxing Invariance: Merits and trade offs**
 - *Recurrent Sets*: Letting thing go and come back
- **Nonparametric Analysis via Recurrent Sets**
 - *Stability analysis*: Recurrent Lyapunov Functions (RLFs)
 - *Safety verification*: Recurrent Barrier functions (RBFs)
- **Self-Improving via Nonparametric Control Policies**
 - Policy Improvement using Expert Demonstrations

Reinforcement Learning

- **Agent:** at time t

- Receives state s_t and reward r_t
- Performs action a_t

Agent: $\pi_\theta(a|s)$

- **Environment:**

- Receives action a_t
- Provides state s_{t+1} and reward r_{t+1}

Environment

- **Goal:** Find a policy π_θ that maximizes

$$\max_{\theta} J(\theta) := E_{\pi_\theta, s_0 \sim \rho} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right]$$

- **RL Language:**

- **Value function:**

$$V^{\pi_\theta}(s_t) := E_{\pi_\theta} \left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r(s_{t'}, a_{t'}) \right]$$

Reinforcement Learning

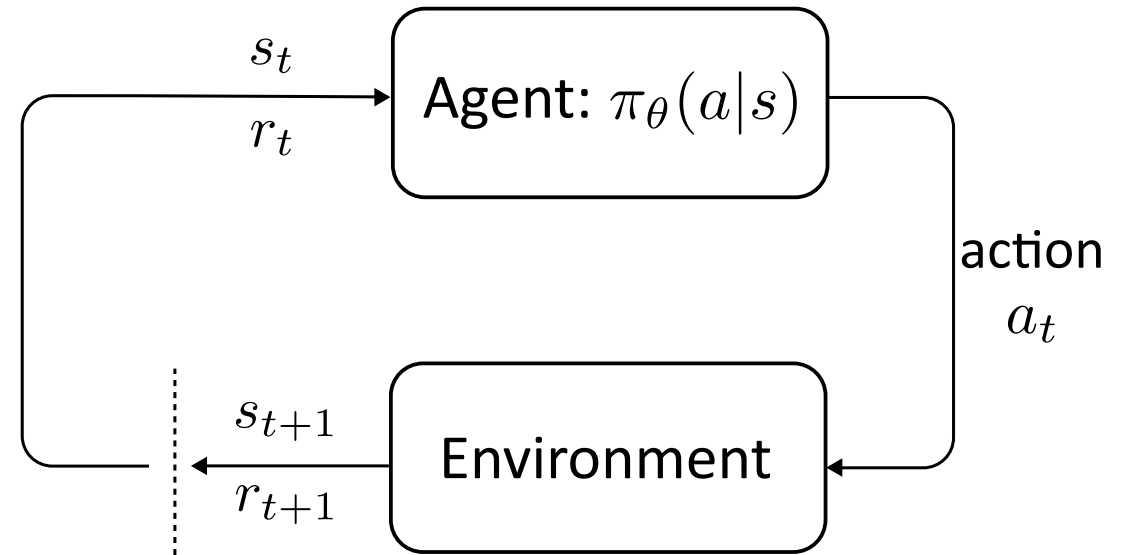
- **Agent:** at time t
 - Receives state s_t and reward r_t
 - Performs action a_t
- **Environment:**
 - Receives action a_t
 - Provides state s_{t+1} and reward r_{t+1}
- **Goal:** Find a policy π_θ that maximizes

$$\max_{\theta} J(\theta) := E_{s_0 \sim \rho} \left[V^{\pi_\theta}(s_0) \right]$$

- **RL Language:**

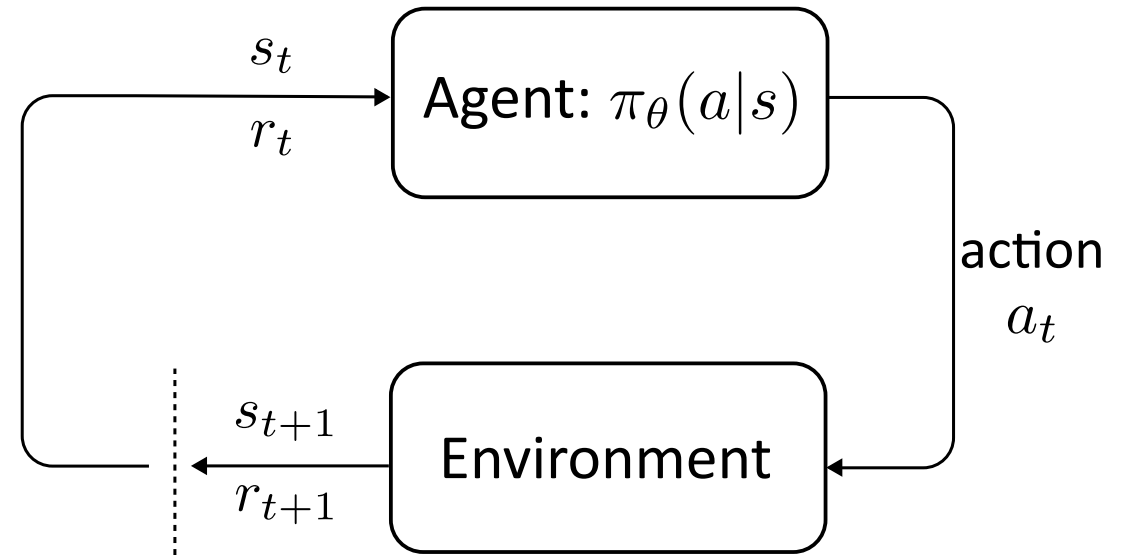
- **Value function:** $V^{\pi_\theta}(s_t) := E_{\pi_\theta} \left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r(s_{t'}, a_{t'}) \right]$

- **Action value function:** $Q^{\pi_\theta}(s_t, a_t) := E_{\pi_\theta} \left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r(s_{t'}, a_{t'}) \right]$



Reinforcement Learning

- **Agent:** at time t
 - Receives state s_t and reward r_t
 - Performs action a_t
- **Environment:**
 - Receives action a_t
 - Provides state s_{t+1} and reward r_{t+1}
- **Goal:** Find a policy π_θ that maximizes



$$\max_{\theta} J(\theta) := E_{s_0 \sim \rho, a_0 \sim \pi_\theta(s_0)} \left[Q^{\pi_\theta}(s_0, a_0) \right]$$

- **RL Language:**

- **Value function:**

$$V^{\pi_\theta}(s_t) := E_{\pi_\theta} \left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r(s_{t'}, a_{t'}) \right]$$

- **Action value function:**

$$Q^{\pi_\theta}(s_t, a_t) := E_{\pi_\theta} \left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r(s_{t'}, a_{t'}) \right]$$

Classical policy improvement works in discrete spaces

“Policy improvement” is a fundamental building block of classical RL

Policy iteration = Policy evaluation + Policy improvement

Policy evaluation

- Given π , evaluate it to find $Q^\pi(\cdot, \cdot)$

- Can evaluate “separately” for each (s, a)
- Can store Q in a table

Policy improvement

- Given $Q^\pi(\cdot, \cdot)$, define: $\pi' : \mathcal{S} \rightarrow \mathcal{A} : \pi'(s) \in \operatorname{argmax}_{a \in \mathcal{A}} Q^\pi(s, a)$

- Then:

$$V^{\pi'}(s) \geq V^\pi(s) \quad \forall s \in \mathcal{S}$$

- Given s , maximize an array of size $|\mathcal{A}|$

Rinse and repeat until $V^{\pi'} \equiv V^\pi \implies \pi = \pi' = \pi^*$

Policy Optimization in Continuous Action Spaces

$$\max_{\theta} J(\theta)$$

Based on Policy Gradient:

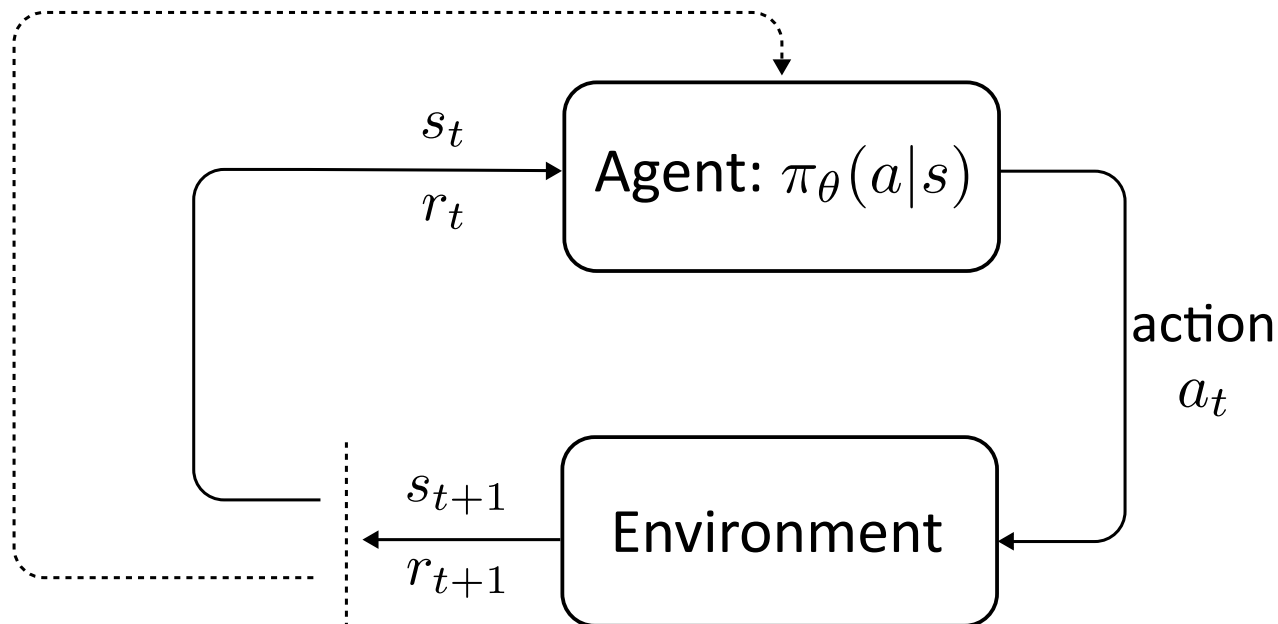
- Use experience to approximate $\nabla_{\theta} J(\theta) \approx \hat{g}$

$$\hat{g} := \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(a_t^{(i)} | s_t^{(i)}) \hat{R}_t^{(i)}$$

$$\text{update: } \theta_{k+1} = \theta_k + \eta \hat{g}$$

$$\hat{R}_t^{(i)} = \sum_{k=t}^T \gamma^{k-t} r_k^{(i)}$$

cumulative return



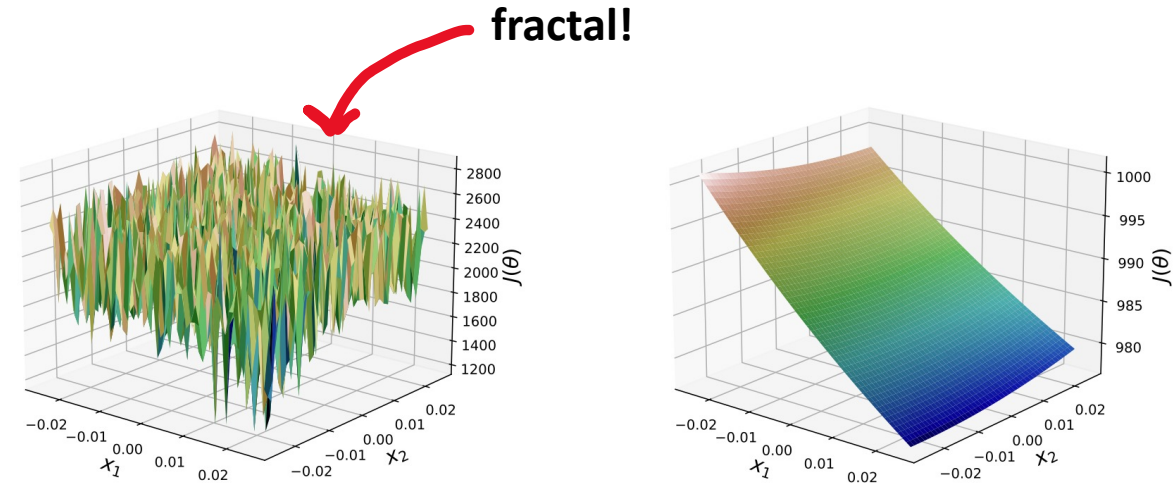
Many Challenges:

- Estimation variance
- Non-smoothness
- Fractal landscape
- Mollification
- ...

Fundamental challenges of Policy Optimization

Challenge: Fractal Optimization Landscapes

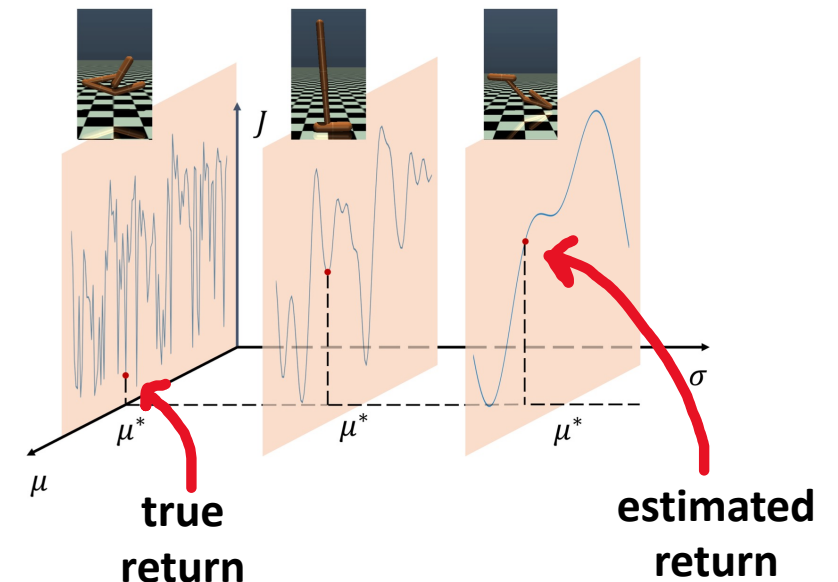
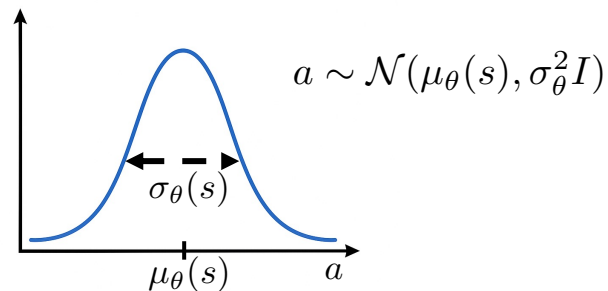
- **Goal:** $\max_{\theta} J(\theta) := E_{\pi_{\theta}, s_0 \sim \rho} [\sum_{t=0}^{\infty} \gamma^t r_t(s_t, a_t)]$
- **Approach:** $\theta_{k+1} = \theta_k + \eta \hat{\nabla}_{\theta} J(\theta)$



Challenge: Mollification of Policy Gradient

- **Goal:** $\max_{\theta} J(\theta) := E_{s_0 \sim \rho, a_0 \sim \pi_{\theta}(s_0)} [Q^{\pi_{\theta}}(s_0, a_0)]$

- **Policy:**





Agustin Castellano



Sohrab Rezaei



Jared Markowitz



Enrique Mallada



Nonparametric policy improvement in continuous action spaces

A. Castellano, S. Rezaei, J. Markovitz, and E. Mallada, Nonparametric Policy Improvement for Continuous Action Spaces via Expert Demonstrations, 2025, submitted to Reinforcement Learning Conference.

Problem Setup

Goal: find optimal policy

$$\max_{\theta} J(\theta) := E_{s_0 \sim \rho, a_0 \sim \pi_{\theta}(s_0)} \left[Q^{\pi_{\theta}}(s_0, a_0) \right]$$

Problem Setup

Goal: find optimal **nonparametric** policy

$$\max_{\mathcal{D}} J(\pi_{\mathcal{D}}) := E_{s_0 \sim \rho, a_0 \sim \pi_{\mathcal{D}}(s_0)} \left[Q^{\pi_{\mathcal{D}}}(s_0, a_0) \right]$$

Data set: $\mathcal{D} = \{(s_i, a_i, Q_i)\}_{i=1}^{|\mathcal{D}|}$ $Q_i := \sum_t \gamma^t r(s_t, a_t)$

Assumptions:

Optimal Q^* is smooth: $|Q^*(s, a) - Q^*(s', a')| \leq L(d_{\mathcal{S}}(s, s') + d_{\mathcal{A}}(a, a'))$

Deterministic dynamics: $s_{t+1} = f(s_t, a_t)$

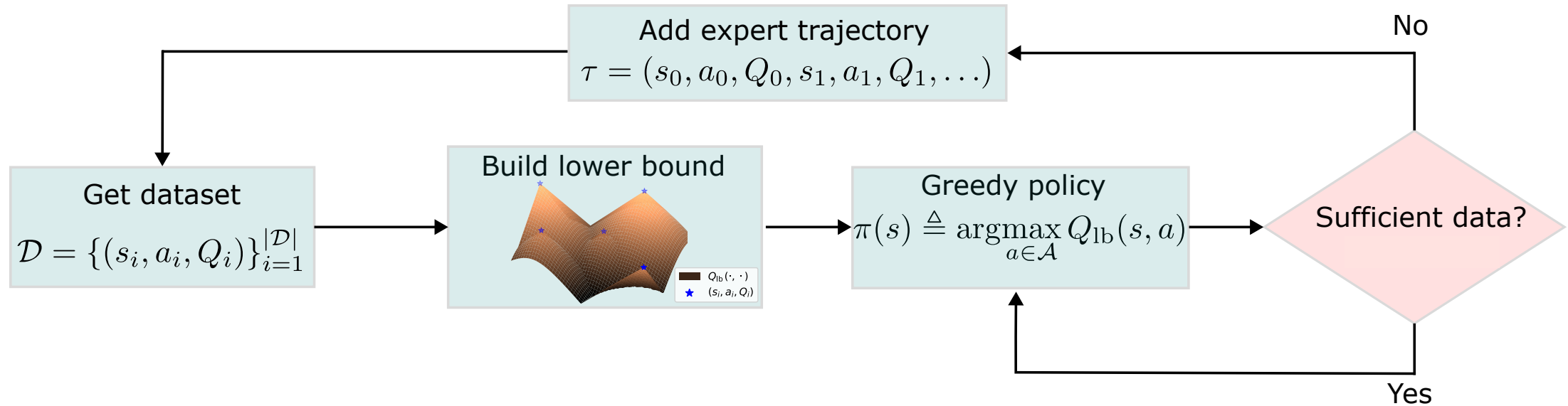
Expert data: we have $\mathcal{D} = \{(s_i, a_i, Q_i)\}_{i=1}^{|\mathcal{D}|}$, where $a_i = \pi^*(s_i); Q_i = Q^*(s_i, a_i)$



Expert data: we have $\mathcal{D} = \{(s_i, a_i, Q_i)\}_{i=1}^{|\mathcal{D}|}$, where $a_i = \pi^*(s_i)$; $Q_i = Q^*(s_i, a_i)$

1. **How** can we use these transitions to learn a nonparametric policy?
2. **What** guarantees can we get when we add more transitions?
3. **Where** should we add transitions to improve performance?

Overview of our method



1. How can we use these transitions to learn a nonparametric policy?

Building bounds & Nonparametric Policy

Expert data: we have $\mathcal{D} = \{(s_i, a_i, Q_i)\}_{i=1}^{|\mathcal{D}|}$, where $a_i = \pi^*(s_i)$; $Q_i = Q^*(s_i, a_i)$

- Use the data to define **lower** bounds on optimal values:

$$V_{\text{lb}}(s) \triangleq \max_{1 \leq i \leq |\mathcal{D}|} \{Q_i - L \cdot d_{\mathcal{S}}(s, s_i)\} \quad Q_{\text{lb}}(s, a) \triangleq \max_{1 \leq i \leq |\mathcal{D}|} \{Q_i - L \cdot (d_{\mathcal{S}}(s, s_i) + d_{\mathcal{A}}(a, a_i))\}$$

$$V_{\text{ub}}(s) \triangleq \min_{1 \leq i \leq |\mathcal{D}|} \{Q_i + L \cdot d_{\mathcal{S}}(s, s_i)\}$$

- **Nonparametric Policy:**

$$\pi(s) \triangleq \operatorname{argmax}_{a \in \mathcal{A}} Q_{\text{lb}}(s, a) = a_{i'}$$

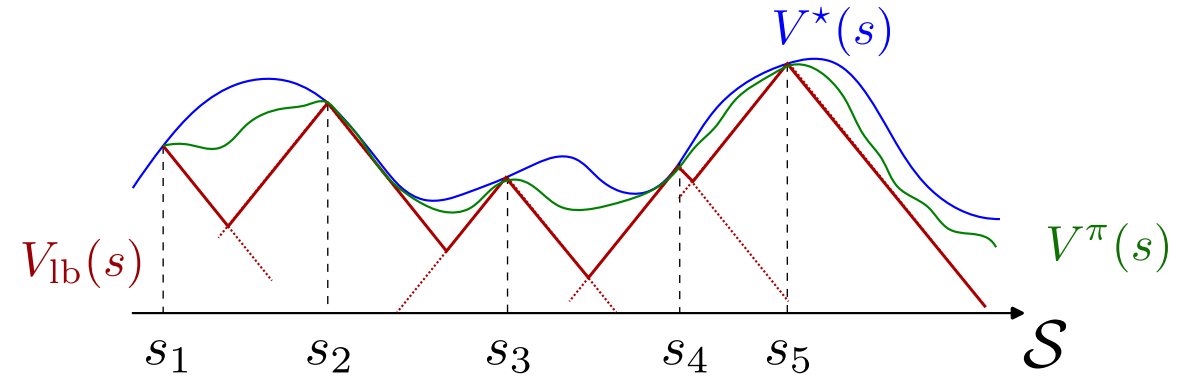
- **Remark:** Note argmax always gives actions in dataset $(s_{i'}, a_{i'}, Q_{i'})$
- **Question:** What can we say about $V^{\pi}(s)$?

Nonparametric policy *improves* over lower bound

Policy Evaluation:

- Nonparametric π satisfies $\forall s \in \mathcal{S}$:

$$V_{lb}(s) \leq V^\pi(s) \leq V^*(s)$$



Policy Improvement:

- Given data sets $\mathcal{D}, \mathcal{D}'$ with $\mathcal{D} \subset \mathcal{D}'$

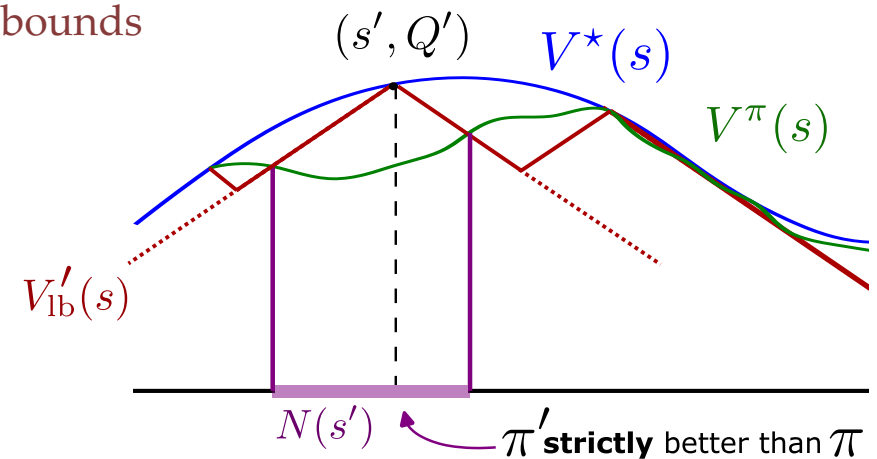
$$V_{lb}(s) \leq V'_{lb}(s) \quad \forall s \in \mathcal{S}$$

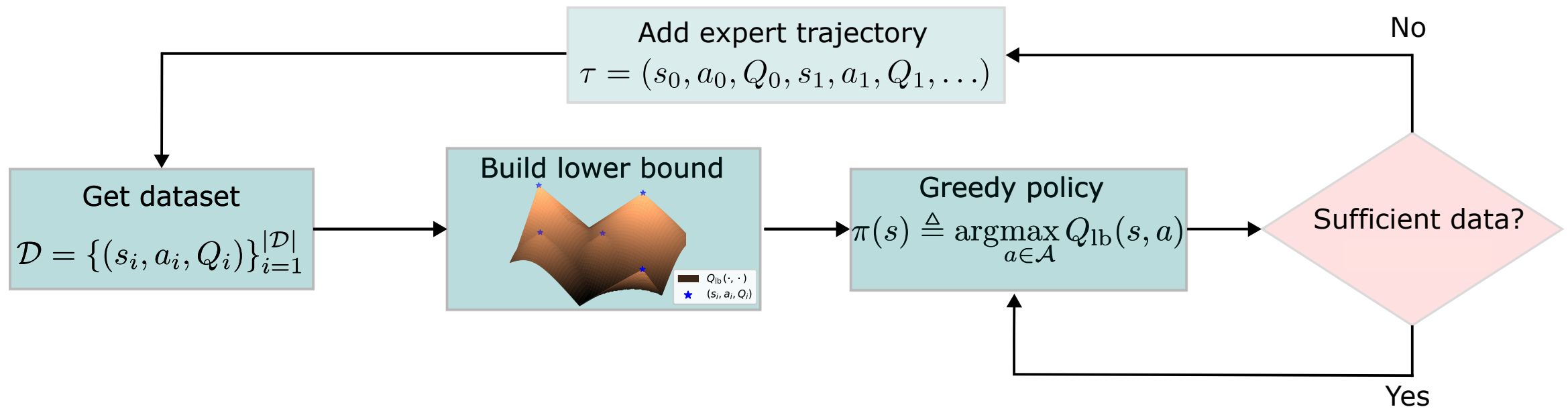
More data = better lower bounds

$$V^\pi(s') \leq V^{\pi'}(s') \quad \forall s' \in \mathcal{D}' \setminus \mathcal{D}$$

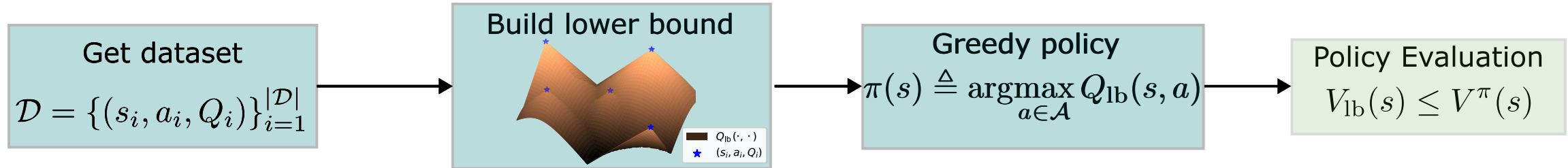
Improvement on added points

- Strict** on neighbors of new data: $\forall s \in N(s')$





1. How to learn a policy?



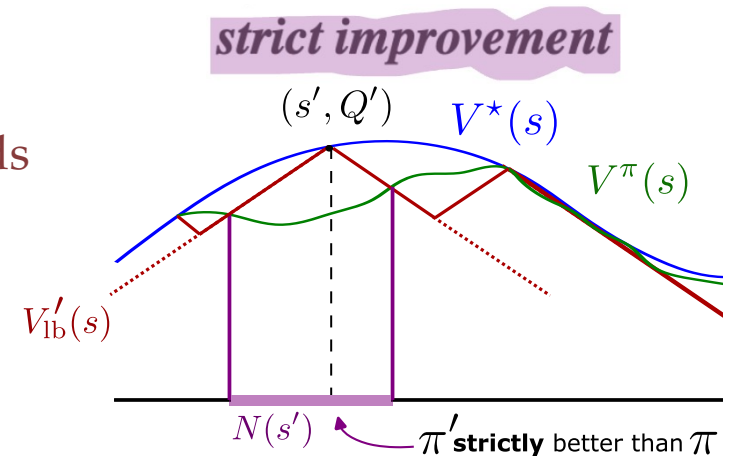
2. What guarantees with more transitions?

More data = better lower bounds

$$V_{lb}(s) \leq V'_{lb}(s) \quad \forall s \in \mathcal{S}$$

Improvement on added points

$$V^\pi(s') \leq V^{\pi'}(s') \quad \forall s' \in \mathcal{D}' \setminus \mathcal{D}$$



3. Where to add transitions?

- Only *where* **sufficient improvement** is guaranteed: $\Delta(s) := V_{ub}(s) - V_{lb}(s) > \varepsilon$

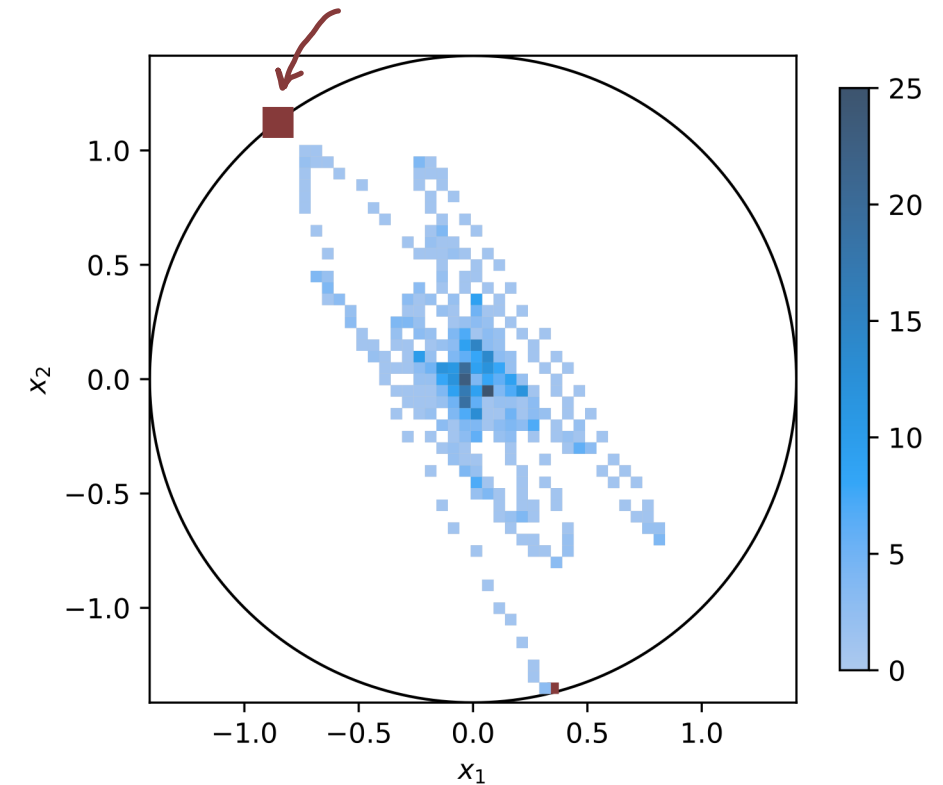
$$V_{lb}(s) \leq V^\pi(s) \leq V^*(s) \leq V_{ub}(s)$$

Algorithm

Input: Lipschitz constant L

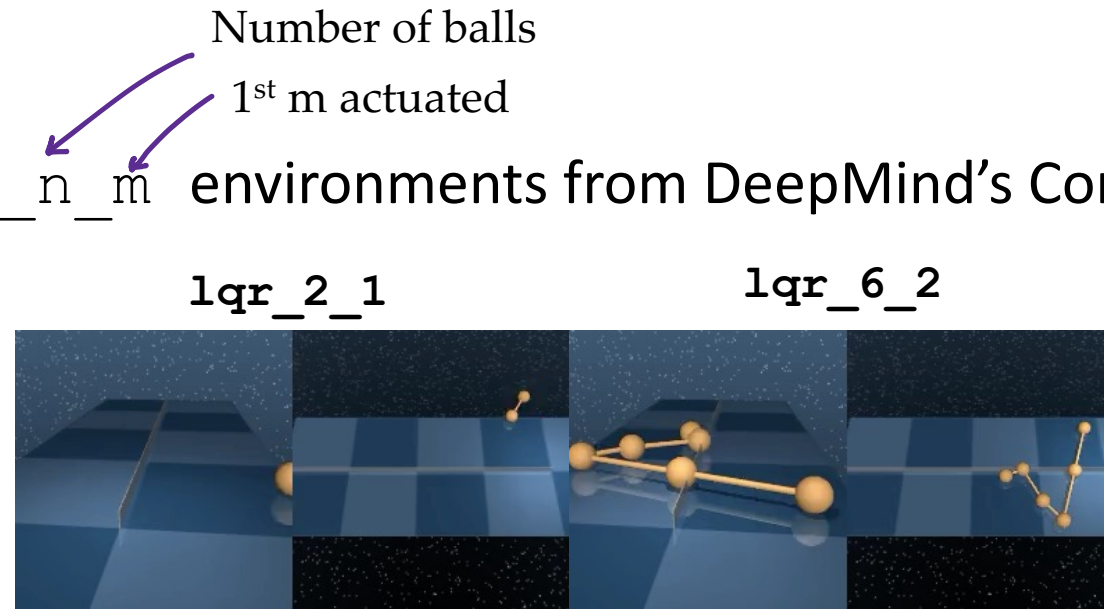
For each episode **do**:

1. Sample $s \sim \rho$
2. If $\Delta(s) > \varepsilon$:
 - Run optimal trajectory with π^*
 $\tau = (s_0, a_0, Q_0, s_1, a_1, Q_1, \dots)$
 - **Repeat:** add tuples to dataset (from $i = 0$)
 $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_i, a_i, Q_i)\}$
Until: $\Delta(s_i) \ll \varepsilon$.
3. Else:
 - **Continue**

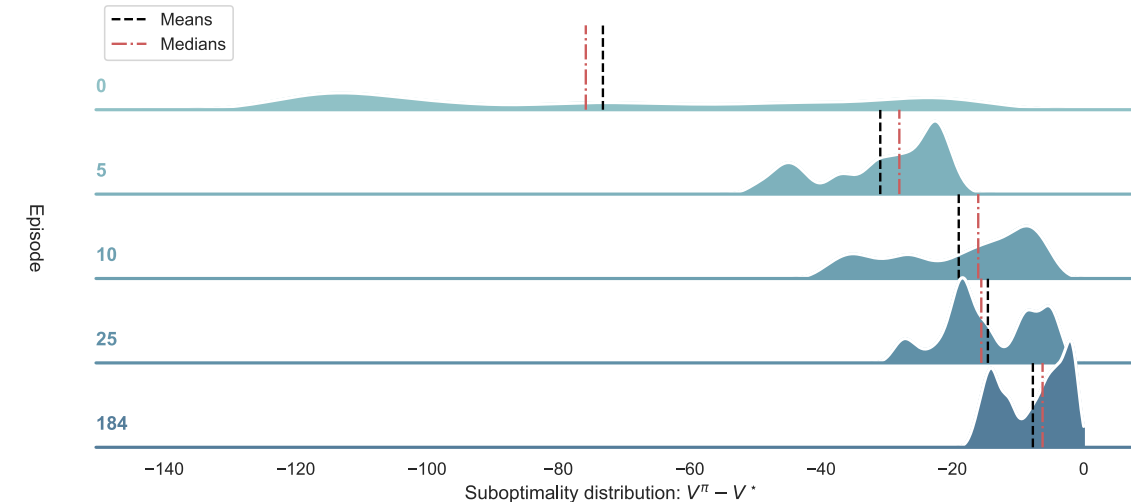
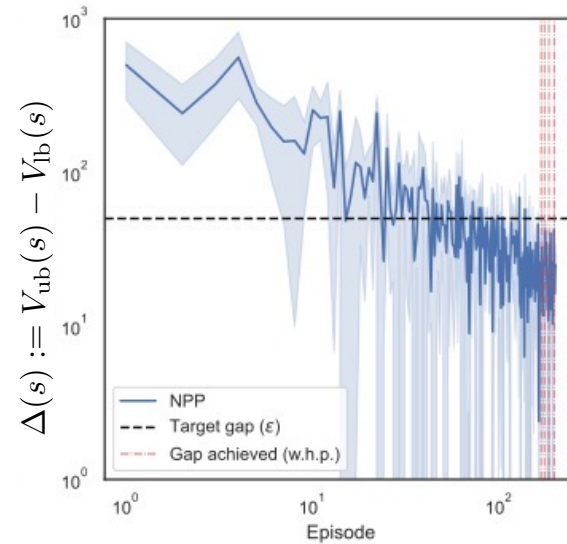
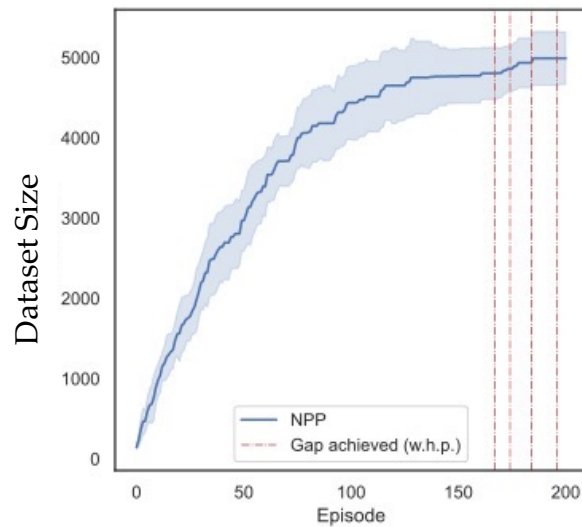


Experiments

- We use the `lqr_n_m` environments from DeepMind's Control Suite

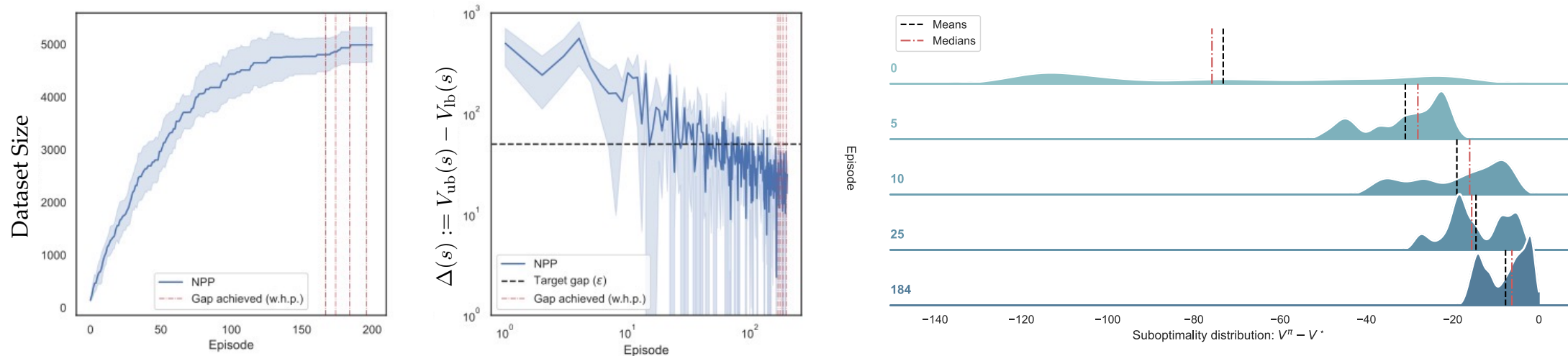


- Results on `lqr_2_1`:



Experiments

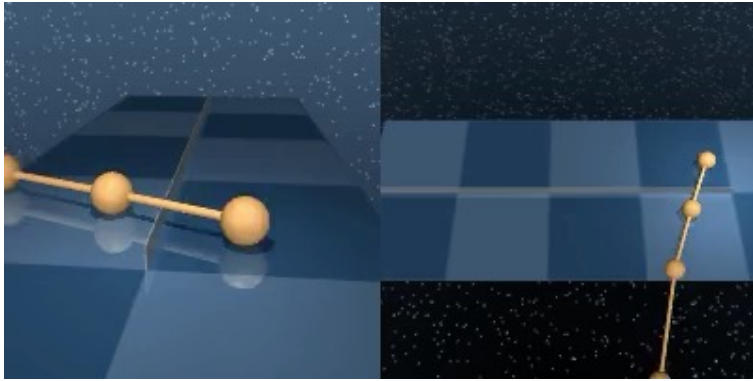
- We use the `lqr_n_m` environments from DeepMind's Control Suite
- Results on `lqr_2_1`:



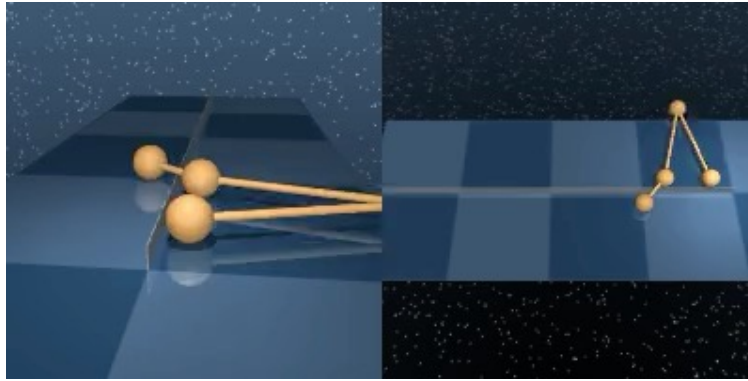
- **Remarks:**
 - **Incremental learning:** No catastrophic forgetting, or oscillations
 - Improvement across the entire state space (not in expectation)
 - Only valuable data is added (harder to find at times passes)

Incremental Learning

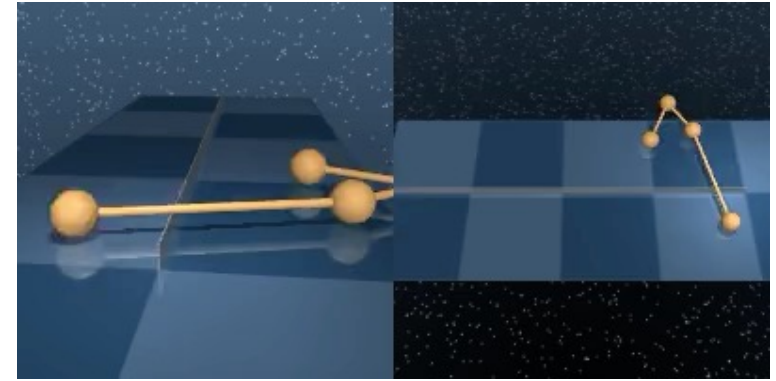
after 10 episode...



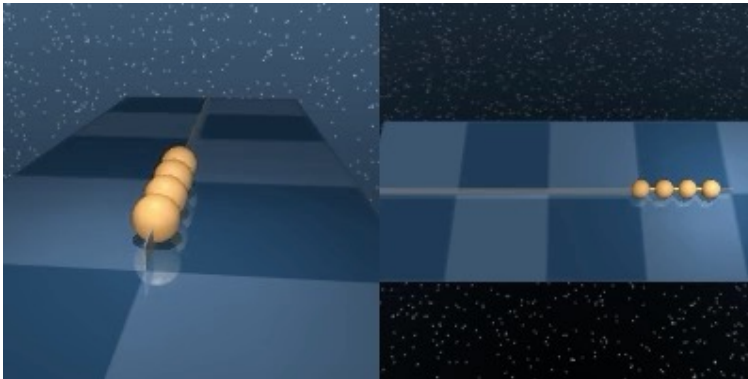
after 100 episode...



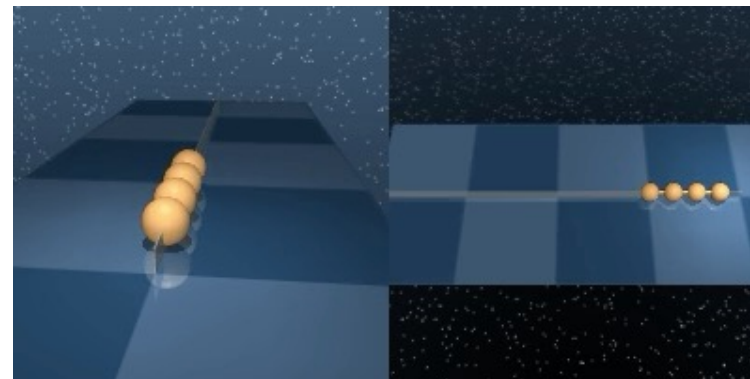
after 1000 episodes...



after 30K+

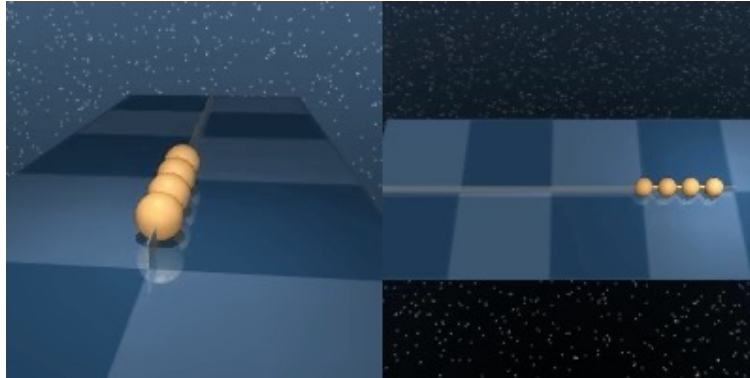


optimal control

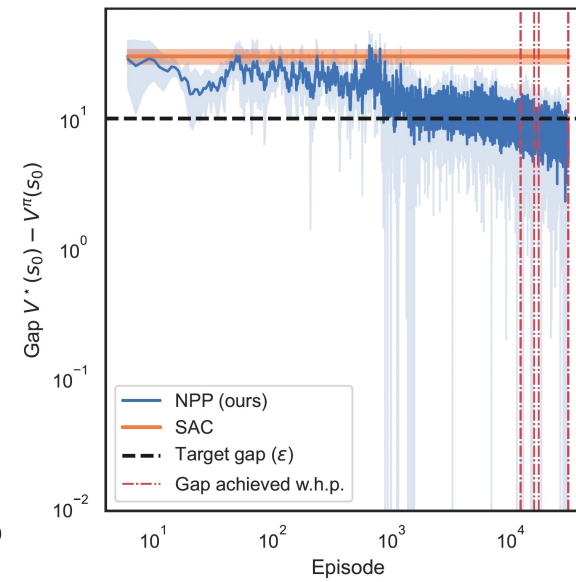
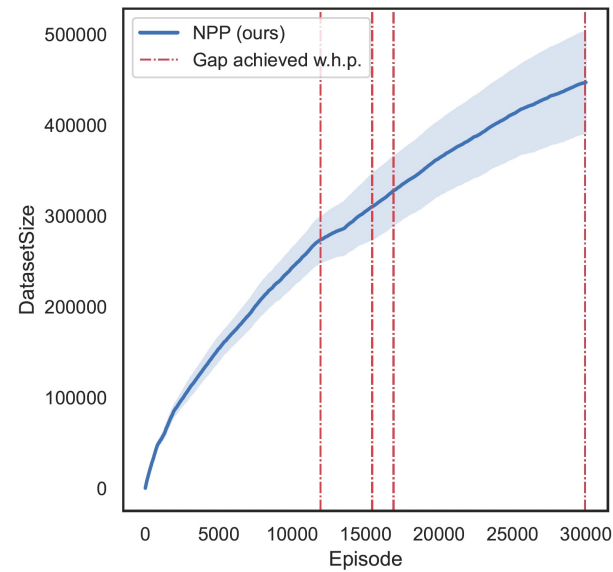


Incremental Learning

after 30K+

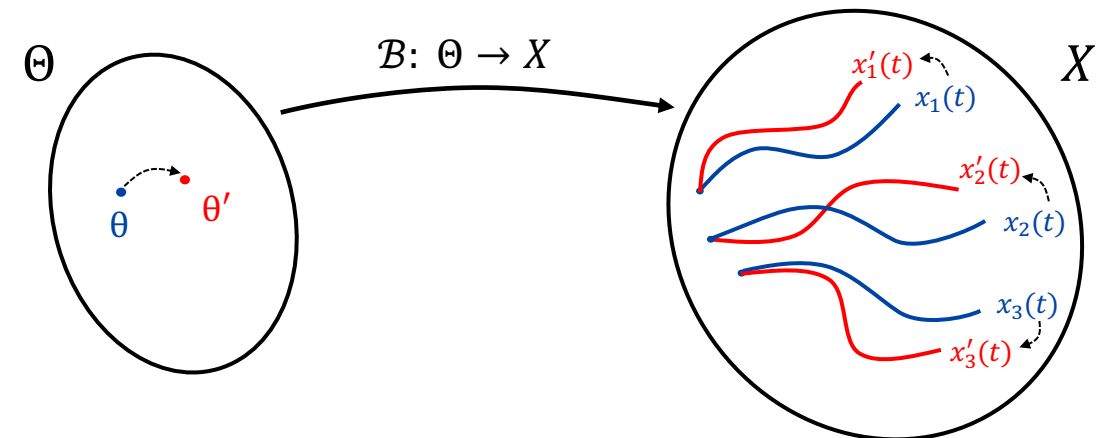
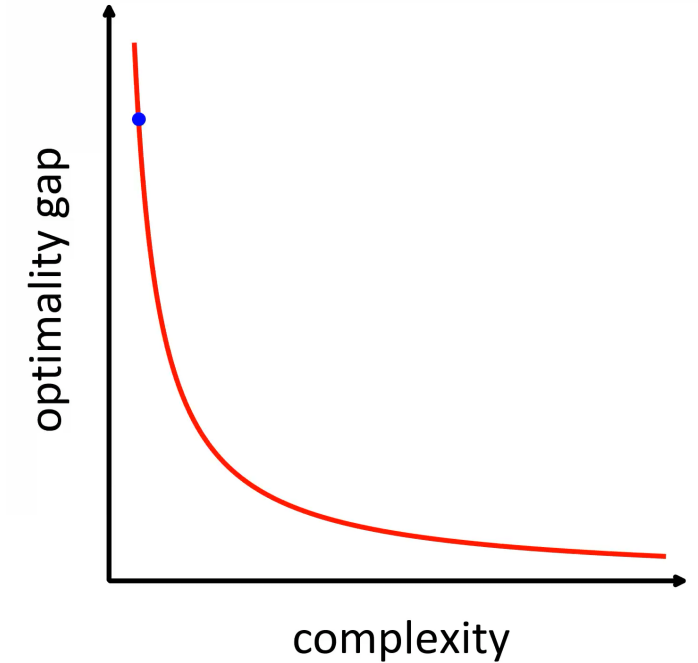


optimal control



Research Goals

- To develop analysis and design methods that *trade off complexity and performance*.
- To allow for *continual improvement*, without the need for redesign, retune, or retrain
- To design control policies with controlled sensitivity to parameter changes



Conclusions and Future work

- **Takeaways**

- Proposed a **relaxed notion of invariance: recurrence**.
- **Nonparametric theory for dynamical systems analysis** leading to:
 - General **Lyapunov** and **Barrier Function** conditions **satisfied by any norm!**
 - Algorithms that are **parallelizable and progressive/sequential**.
- **Nonparametric policies:** Guaranteed improvement with each demonstration.

- **Ongoing work**

- **Recurrence:** Information theoretical lower bounds of control recurrence sets
- **Lyapunov/CBF Theory:** Generalize other Lyapunov notions, Control Lyapunov Functions, Control Barrier Functions, Contraction
- **Nonparametric policies (NP):** NP policy iteration, enforcing safety and stability using NP, exploring alternative inductive biases (beyond Lipschitz)

Thanks!

Related Publications:

- [CDC 23] Siegelmann, Shen, Paganini, M, *A recurrence-based direct method for stability analysis and GPU-based verification of non-monotonic Lyapunov functions*, **CDC 2023, TAC submitted**
- [HSCC 24] Sibai, M, *Recurrence of nonlinear control systems: Entropy and bit rates*, **HSCC, 2024, NAHS under review**
- [Allerton 24] Shen, Sibai, M, *Generalized Barrier Functions: Integral conditions and recurrent relaxations*, **Allerton 2024**
- [RLC 25] Castellano, Rezaei, Markovitz, and M, *Nonparametric Policy Improvement for Continuous Action Spaces via Expert Demonstrations*, **RLC 2025**

Enrique Mallada
mallada@jhu.edu
<http://mallada.ece.jhu.edu>