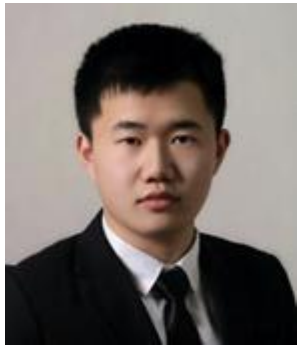
On the hardness of learning to stabilize linear systems

Xiong Zeng[1] Zexiang Liu[1] Zhe Du[1] **Necmiye Ozay[1]** Mario Sznaier[2]



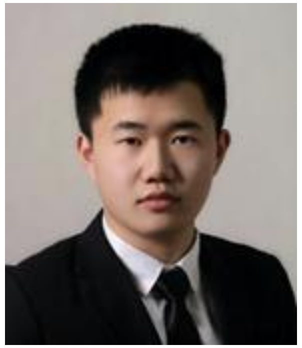
[1]Electrical Engineering and Computer Science
University of Michigan, Ann Arbor

[2]Electrical and Computer Engineering
Northeastern University

On the hardness of learning to stabilize linear systems

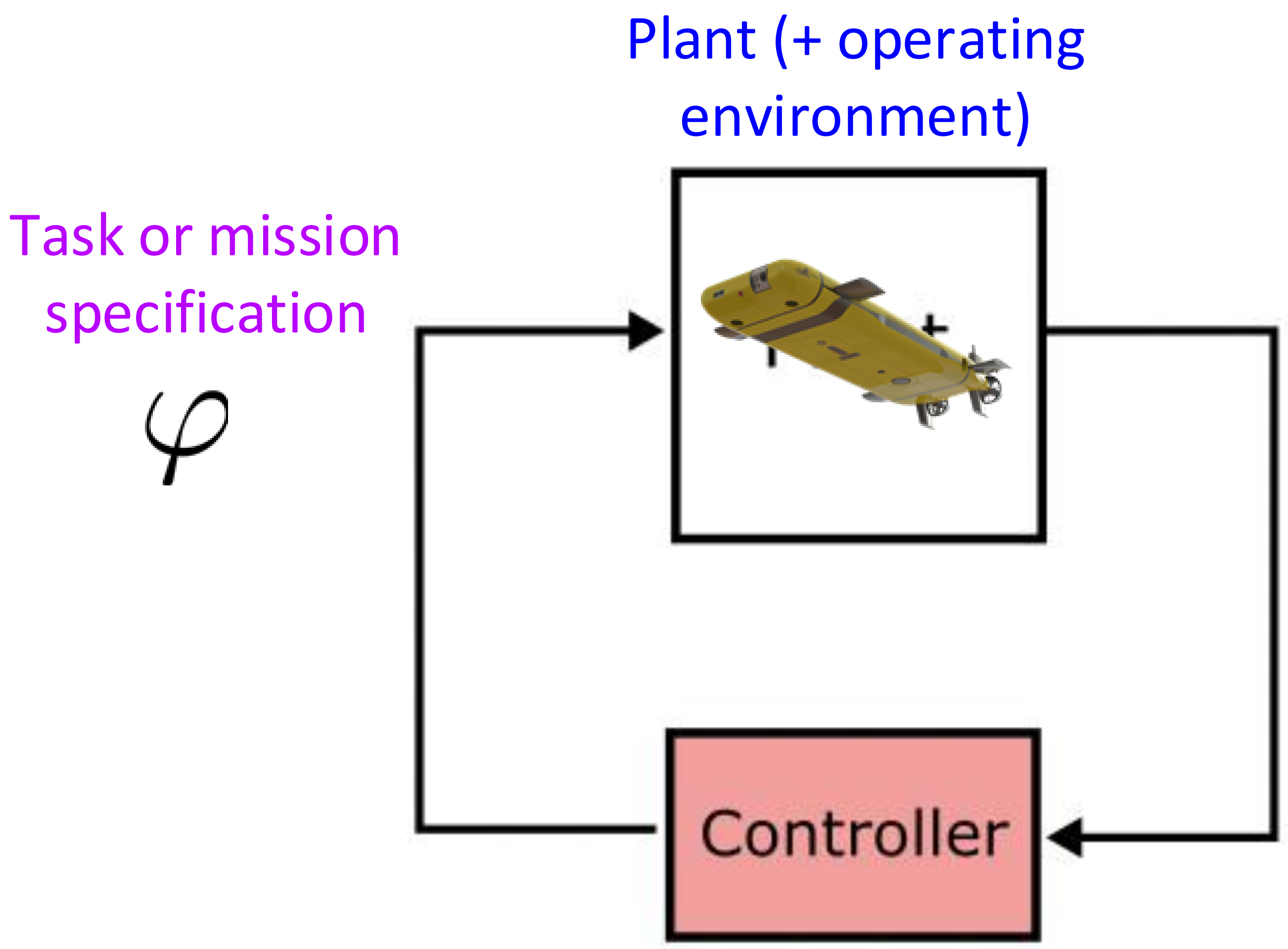
with a static linear state-feedback controller

Xiong Zeng[1] Zexiang Liu[1] Zhe Du[1] **Necmiye Ozay[1]** Mario Sznaier[2]



[1]Electrical Engineering and Computer Science
University of Michigan, Ann Arbor

[2]Electrical and Computer Engineering
Northeastern University



Many components in a feedback loop can be learned:

- **Plant model:** model-based learning, system identification
- **Task specification:** inverse optimal control, inverse reinforcement learning
- **Controller:** model-free learning
- **Perception/estimation modules**

What are the fundamental limits in learning-based control?

What is complexity?

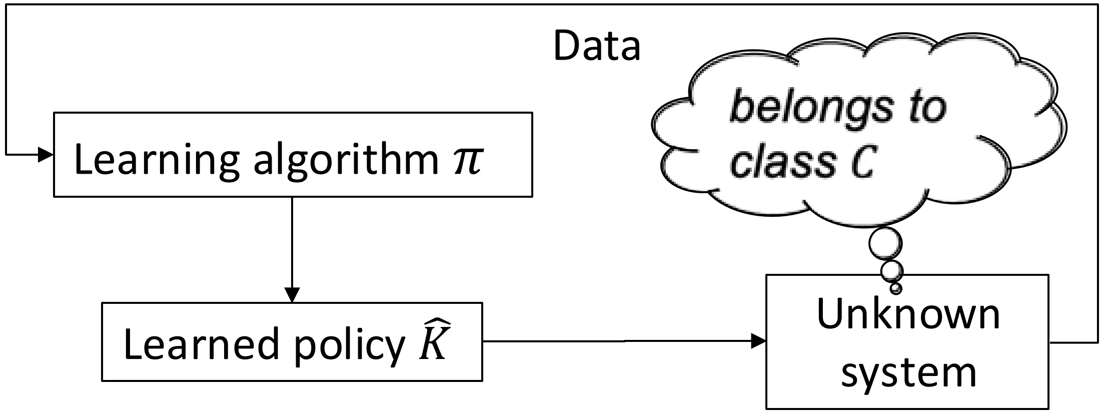
- Computational complexity: how the **computational cost** of an algorithm depends on the **input size**?

$$\textbf{Comp. Cost} = \mathcal{O}(\textbf{input size})$$

- Sample complexity: how a **performance metric** of a *learning* algorithm depends on the **sample size** (# of data points)?

$$\textbf{Performance} = \mathcal{O}(\textbf{sample size})$$

Examples of performance metric:
approximation error,
accumulative cost or regret,
probability of stability (this work)



complexity of a problem ≈ complexity of the best algorithm

Setup

Given

- A discrete-time linear time-invariant (LTI) system S from **a class $\boldsymbol{C_n}$**:

$$x_{t+1} = Ax_t + Bu_t + w_t,$$

where $x_t \in \mathbb{R}^n$ and $w_t \sim \mathcal{N}(0, \sigma_w^2 \boldsymbol{I})$

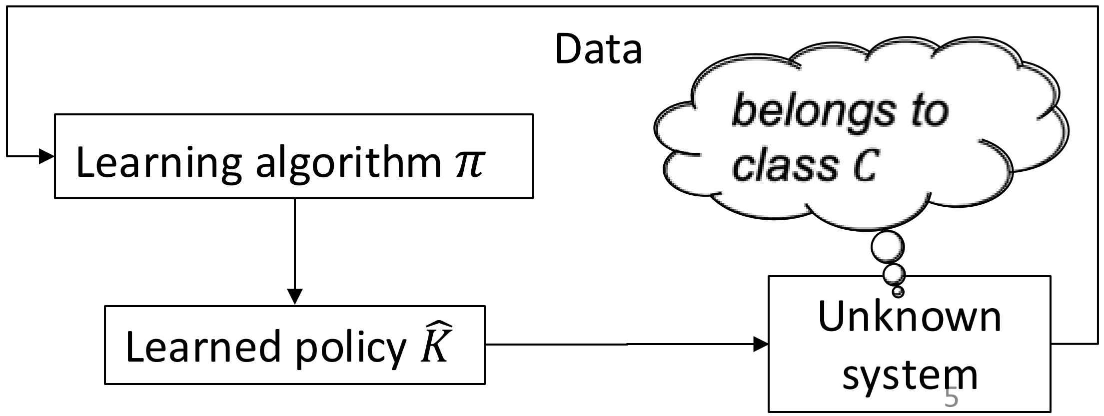
- A time budget N and an input budget $\mathbb{E}[\|u_t\|^2] \leq \sigma_u^2$

Consider a learning algorithm π that

(i) interacts with the system for N units of time, and

(ii) outputs a linear static state-feedback controller $\widehat{K}$ at time N

We want $\widehat{K}$ to stabilize the (unknown) system S.



Setup

Given

- A discrete-time linear time-invariant

$$x_{t+1} =$$

where $x_t \in \mathbb{R}^n$ and $w_t \sim \mathcal{N}(0, \sigma_w^2 \boldsymbol{I})$

- A time budget N and an input budge

Use your favorite algorithm π:

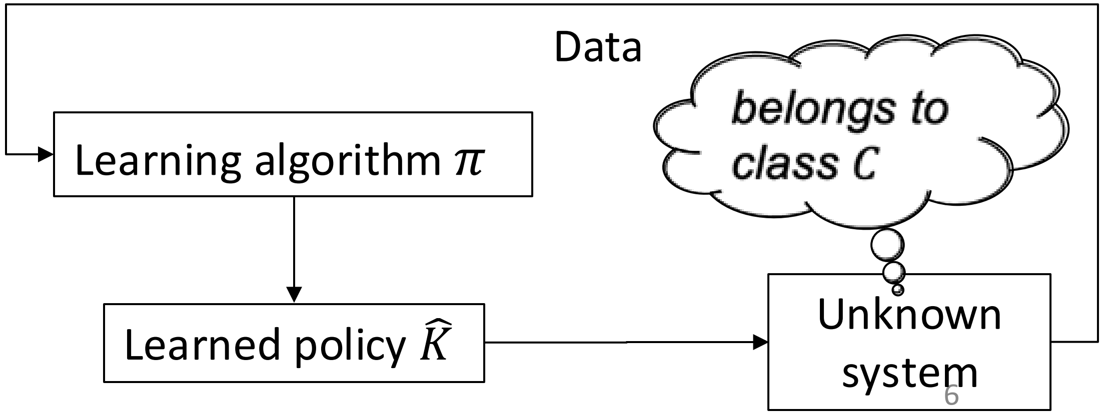
- Excite the system for N steps and do (direct) data-driven control
- Excite the system for N steps and do system id and certainty equivalent control
- Use any other online/active learning method

Consider a learning algorithm π that

(i) interacts with the system for N units of time, and

(ii) outputs a linear static state-feedback controller $\widehat{K}$ at time N

We want $\widehat{K}$ to stabilize the (unknown) system S.



Problem statement

Denote the probability that $\widehat{K}$ stabilizes an unknown system $S = (A, B) \in \mathcal{C}_n$ by

$$P^N_{S,\pi}\left((A + B\,\widehat{K}) \text{is stable}\right)$$

A learning algorithm π is evaluated by the **worst-case** stabilization probability over $\mathcal{C}_n$:

$$\inf_{S \in \mathcal{C}_n} P^N_{S,\pi}((A + B\,\widehat{K}) \text{ is stable})$$

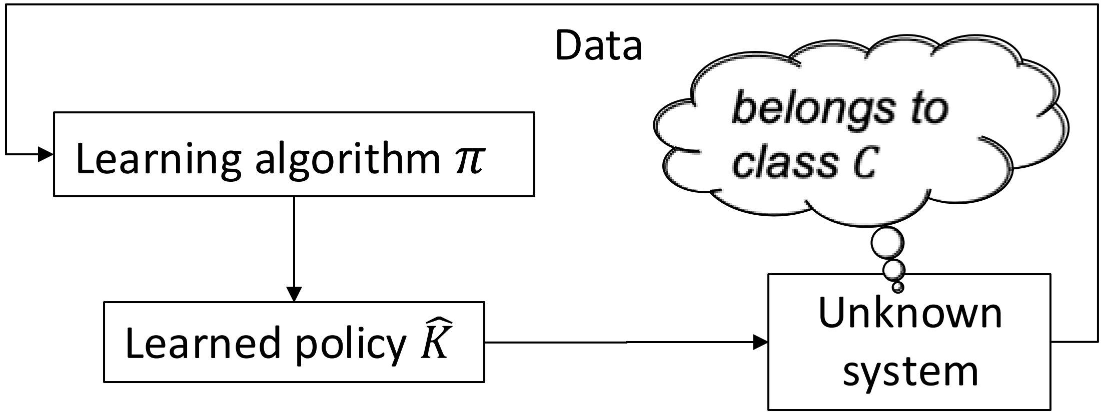
Sample complexity of π: Given a $\delta \in [0,1]$, the **minimum time/data budget N** such that

$$\boxed{\inf_{S \in \mathcal{C}_n} P^N_{S,\pi}((A + B\,\widehat{K}) \text{ is stable}) \geq 1 - \delta.}$$

Question: Given a class C of linear systems, how does the **sample complexity** for learning stabilizing static state-feedback controllers depend on the **system dimension n**?

Related work

Question: Given a class C of linear systems, how does the sample complexity for learning stabilizing static state-feedback controllers depend on the system dimension n?



Existing results (very incomplete list):

Dean et al. 2017, Mania et al. 2019	$N \geq C^{sys} poly(n) \Longrightarrow Stable$	Non-degenerate noise $(w_t \sim N(0,\Sigma), \Sigma > 0)$
Tsiamis et al. 2022	$Stable \Longrightarrow N \geq O(exp(n))$	Degenerate noise ($\Sigma \geq 0$ and $\Sigma \ngtr 0$): Hard to identify
Our Work	$Stable \Longrightarrow N \geq O(exp(n))$	Non-degenerate noise ($\Sigma > 0$): Not necessarily hard to identify but there is a *different obstruction*

Sufficient conditions for sample complexity

Necessary conditions for sample complexity

aka, lower bounds

Roadmap to establish hardness

Find a class $\mathcal{C}_n$ of systems such that for any algorithm π and for all $\delta \in [0,0.5]$, we have

$$\inf_{S \in \mathcal{C}_n} P^N_{S,\pi}((A + B\,\widehat{K}) \text{ is stable}) \geq 1 - \delta.$$

only if

$$N \geq O(exp(n)).$$

Key insight: if the class $\mathcal{C}_n$ contains a pair of systems (A_1, B_1) and (A_2, B_2) that are

- hard to distinguish, and
- not co-stabilizable

then learning to stabilize is hard.

Roadmap to establish hardness

Find a class $\mathcal{C}_n$ of systems such that for any algorithm π and for all $\delta \in [0,0.5]$, we have

$$\inf_{S \in \mathcal{C}_n} P_{S,\pi}^{N}((A + B\ \widehat{K}) \text{ is stable}) \geq 1 - \delta.$$

only if

$$N \geq O(exp(n)).$$

Key insight: if the class $\mathcal{C}_n$ contains a pair of systems (A_1, B_1) and (A_2, B_2) that are

- hard to distinguish, and

  when driven by the same policy π, the resulting state trajectories are similar

- not co-stabilizable

  there does not exist a single controller that simultaneously stabilizes the two systems

then learning to stabilize is hard. $\nexists\ K$ s.t. both $A_1 + B_1 K\ and\ A_2 + B_2 K$ are stable

A pair of systems that are hard to learn to stabilize

$$\mathbf{A} = \begin{bmatrix} r & v & 0 & \cdots & 0 \\ 0 & 0 & v & \cdots & 0 \\ & & \ddots & \ddots & \\ 0 & 0 & 0 & \cdots & v \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{n\times n}, \ \mathbf{B} = \begin{bmatrix} b^{(1)} \\ 0 \\ \vdots \\ 0 \\ v \end{bmatrix} \in \mathbb{R}^{n}$$

$x_{t+1} = Ax_t + Bu_t + w_t,$ $\quad r > 1,\ 0 < v < \frac{r-1}{2}$, and $b^{(1)} \geq 0$.

First order system with actuation delay of length n-1

Take $\mathrm{B}_1 = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ v \end{bmatrix}, \mathrm{B}_2 = \begin{bmatrix} m \\ 0 \\ \vdots \\ v \end{bmatrix}$

A pair of systems that are hard to learn to stabilize

$$\mathbf{A} = \begin{bmatrix} r & v & 0 & \cdots & 0 \\ 0 & 0 & v & \cdots & 0 \\ & & \ddots & \ddots & \\ 0 & 0 & 0 & \cdots & v \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{n \times n}, \ \mathbf{B} = \begin{bmatrix} b^{(1)} \\ 0 \\ \vdots \\ 0 \\ v \end{bmatrix} \in \mathbb{R}^{n}$$

$x_{t+1} = Ax_t + Bu_t + w_t$, $r > 1$, $0 < v < \frac{r-1}{2}$, and $b^{(1)} \geq 0$.

First order system with actuation delay of length n-1

Take $B_1 = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ v \end{bmatrix}, B_2 = \begin{bmatrix} m \\ 0 \\ \vdots \\ v \end{bmatrix}$

Proposition 1: There exists a state-feedback controller K that stabilizes both (A, B_1) and (A, B_2) only if

$$m \leq \mathcal{O}\left(\frac{1}{\exp(n)}\right)$$

A pair of systems that are hard to learn to stabilize

Proposition 1: There exists a state-feedback controller K that stabilizes both (A, B_1) and (A, B_2) only if

$$m \leq \mathcal{O}(\frac{1}{\exp(n)})$$

Proof sketch: Use Ackerman's formula to analytically construct K to place the poles of system 1 arbitrarily inside the unit circle and check the stability of system 2 when controlled by K using Jury stability test.

A pair of systems that are hard to learn to stabilize

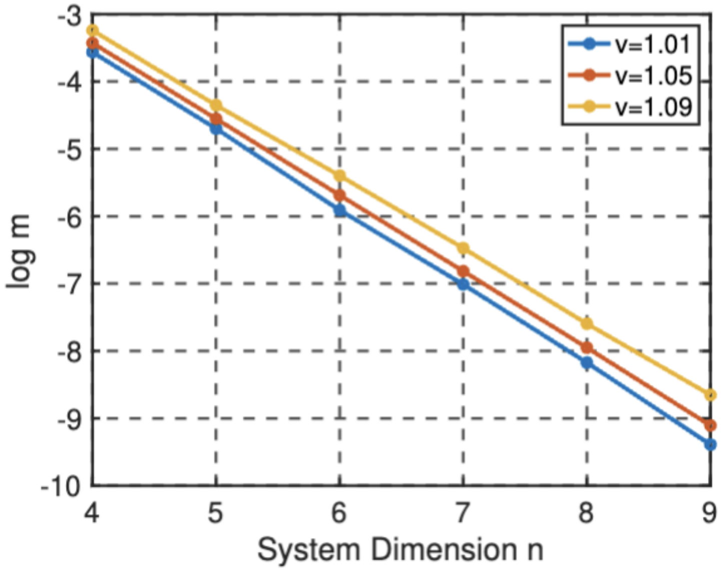
Proposition 1: There exists a state-feedback controller K that stabilizes both (A, B_1) and (A, B_2) only if

$$m \leq \mathcal{O}(\frac{1}{\exp(n)})$$

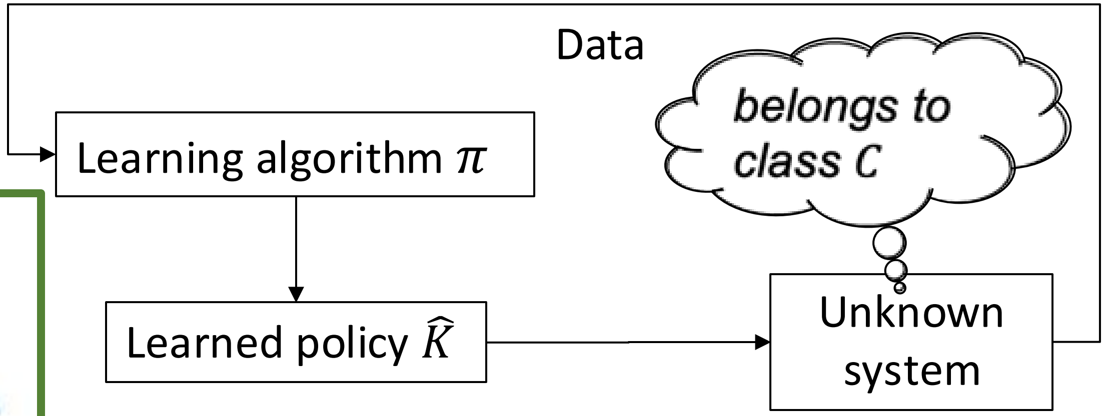
Proof sketch: Use Ackerman's formula to analytically construct K to place the poles of system 1 arbitrarily inside the unit circle and check the stability of system 2 when controlled by K using Jury stability test.

We can also do a sanity check numerically using LMI-based sufficient condition for co-stabilizability

$$\begin{aligned}
find \quad & K, P \\
s.t. \quad & (A + B_1 K)^T P (A + B_1 K) \prec P \\
& (A + B_2(m) K)^T P (A + B_2(m) K) \prec P \\
& P \succ 0
\end{aligned}$$



Main result



$$\mathbf{A} = \begin{bmatrix} r & v & 0 & \cdots & 0 \\ 0 & 0 & v & \cdots & 0 \\ & & \ddots & \ddots & \\ 0 & 0 & 0 & \cdots & v \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{n\times n}, \quad \mathbf{B} = \begin{bmatrix} b^{(1)} \\ 0 \\ \vdots \\ 0 \\ v \end{bmatrix} \in \mathbb{R}^{n}$$

$x_{t+1} = Ax_t + Bu_t + w_t$, $r > 1,\ 0 < v < \frac{r-1}{2}$, and $b^{(1)} \geq 0$.

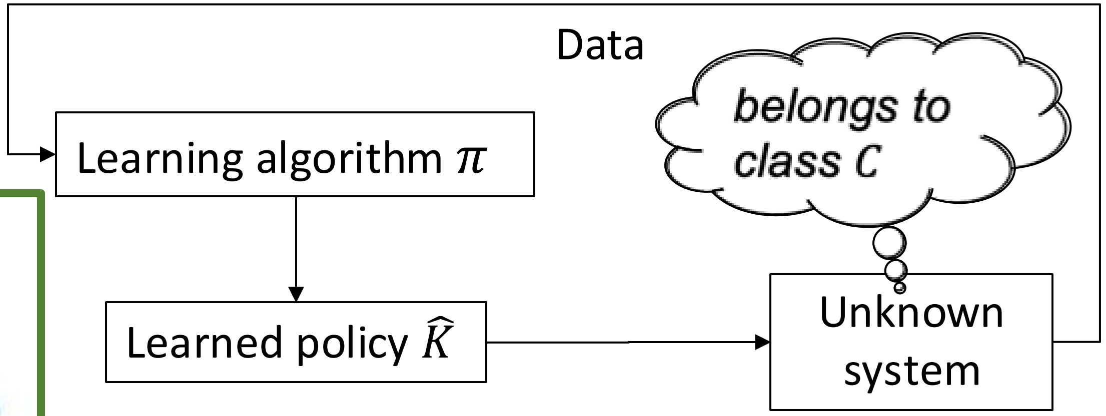
Take $B_1 = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ v \end{bmatrix}, B_2 = \begin{bmatrix} m \\ 0 \\ \vdots \\ v \end{bmatrix}$

Theorem: For any class C of systems containing such a pair with $m = 2\left(\frac{2v}{r-1}\right)^n$ and any learning algorithm, the least amount of data for learning to stabilize arbitrary systems in C grows exponentially with the state dimension!

$$\inf_{S\in C_n} P_{S,\pi}^{N}((A + B\,\hat{K}) \text{ is stable}) \geq 1 - \delta \quad \text{only if} \quad N \geq \underbrace{\frac{\sigma_w^2}{2\sigma_u^2}}_{\text{SNR}} \left(\frac{r-1}{2v}\right)^{2n} \log\left(\frac{1}{3\delta}\right)$$

exp in dimension n

Main result



$$\mathbf{A} = \begin{bmatrix} r & v & 0 & \cdots & 0 \\ 0 & 0 & v & \cdots & 0 \\ & & \ddots & \ddots & \\ 0 & 0 & 0 & \cdots & v \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{n \times n}, \quad \mathbf{B} = \begin{bmatrix} b^{(1)} \\ 0 \\ \vdots \\ 0 \\ v \end{bmatrix} \in \mathbb{R}^{n}$$

$$x_{t+1} = Ax_t + Bu_t + w_t, \quad r > 1,\ 0 < v < \tfrac{r-1}{2}, \text{ and } b^{(1)} \geq 0.$$

$$\text{Take } \mathrm{B}_1 = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ v \end{bmatrix}, \mathrm{B}_2 = \begin{bmatrix} m \\ 0 \\ \vdots \\ v \end{bmatrix}$$

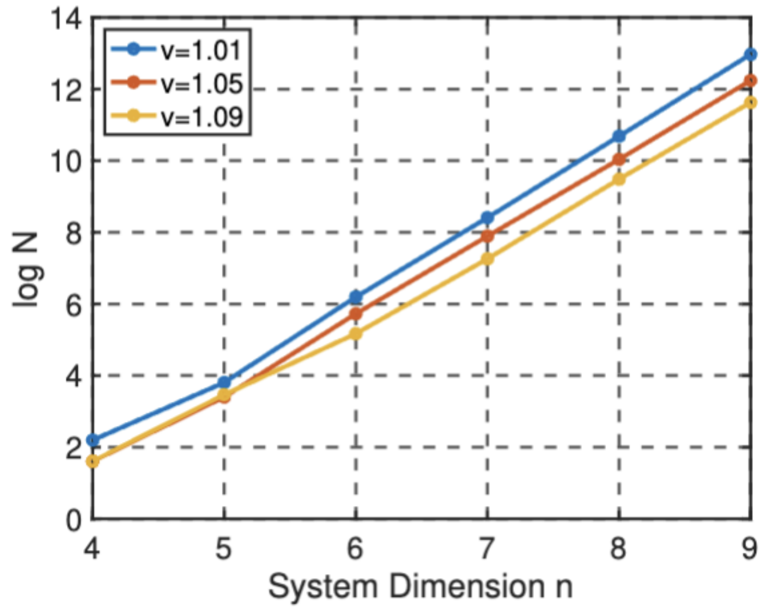
Theorem: For any class C of systems containing such a pair with $m = 2\left(\frac{2v}{r-1}\right)^n$ and any learning algorithm, the least amount of data for learning to stabilize arbitrary systems in C grows exponentially with the state dimension!

Proof Sketch

- Having $b^{(1)}$ entries of the pairs exponentially closer to each other as a function of n upper bounds the KL divergence of length-N trajectories generated by the two systems (indistinguishability)
- Co-stabilizability requires exponential closeness in parameters space due to Proposition 1
- Use Birge's inequality to establish a KL divergence lower bound

Numerical illustration

Number of samples required for 90% probability of finding a stabilizing controller using certainty equivalent LQR*:



y-axis: observed least sample size N for CE LQR being stabilizing with probability 0.9

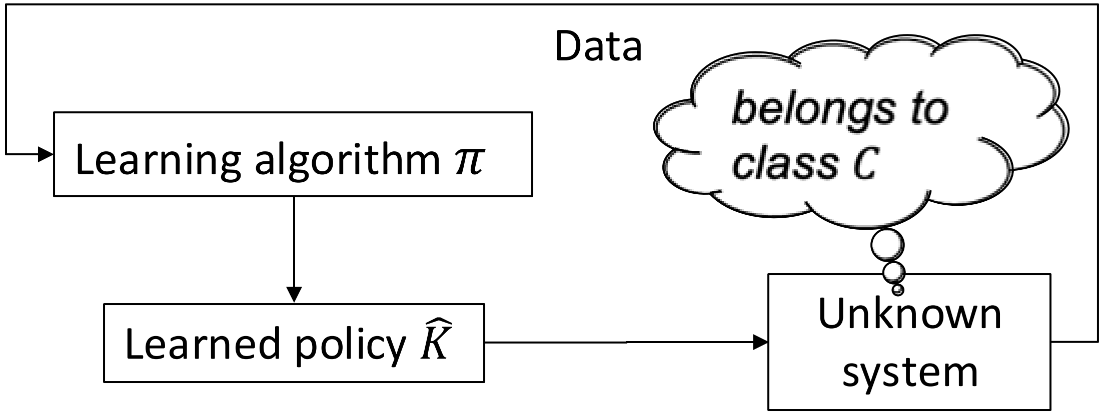
Certainty Equivalent LQR

- Excite the system with iid Gaussian input for N steps
- Learn system matrices
- Do LQR with learned model

$$\min_{u_0,u_1,\ldots} \lim_{T\to\infty} E\left[\frac{1}{T}\sum_{t=0}^{T}(x_t^T Q x_t + u_t^T R u_t)\right]$$
$$s.t.\ x_{t+1} = Ax_t + Bu_t + w_t$$

* By our result, same trend holds for any algorithm

Conclusions



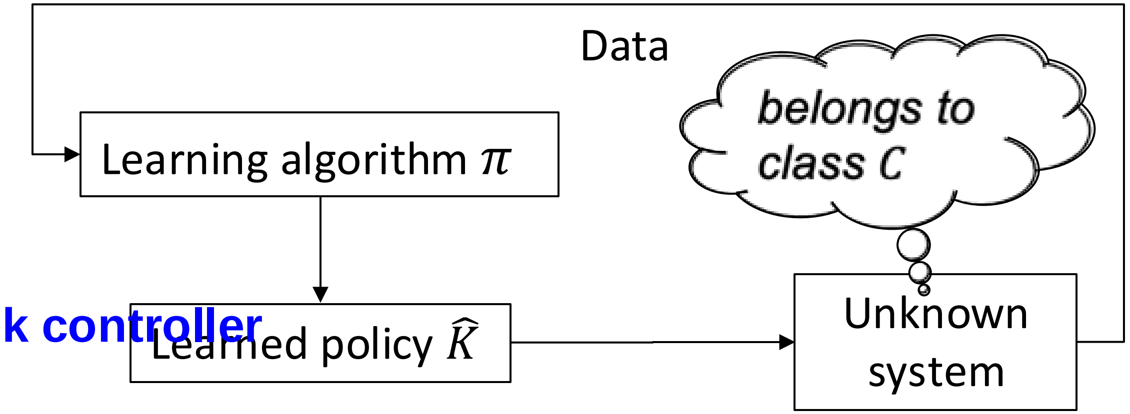
Key contributions:

- Learning to stabilize a linear system is hard if we have systems that are

  hard to distinguish AND hard to co-stabilize

- Compare with earlier work by Tsiamis et al., our analysis
  - reveals a new obstruction for the hardness of learning to stabilize;
  - covers a larger class of systems (including open-loop unstable ones)

Conclusions



Key contributions:

with a static linear state-feedback controller

- Learning to stabilize a linear system is hard if we have systems that are

  hard to distinguish AND hard to co-stabilize

- Compare with earlier work by Tsiamis et al., our analysis
  - reveals a new obstruction for the hardness of learning to stabilize;
  - covers a larger class of systems (including open-loop unstable ones)

Current work:

- <u>Observation:</u> Static linear state feedback controllers are not a rich enough class for co-stabilizing linear systems (memory helps!)
- Can we reduce sample complexity by using dynamic time-varying controllers?