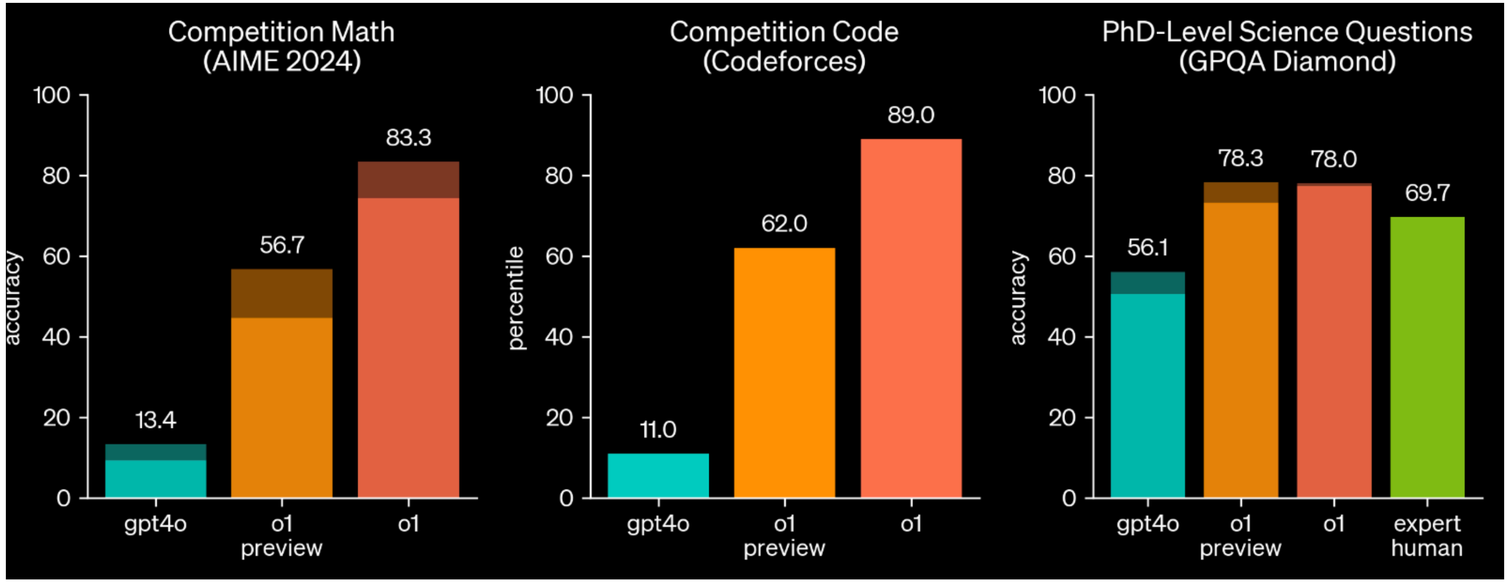
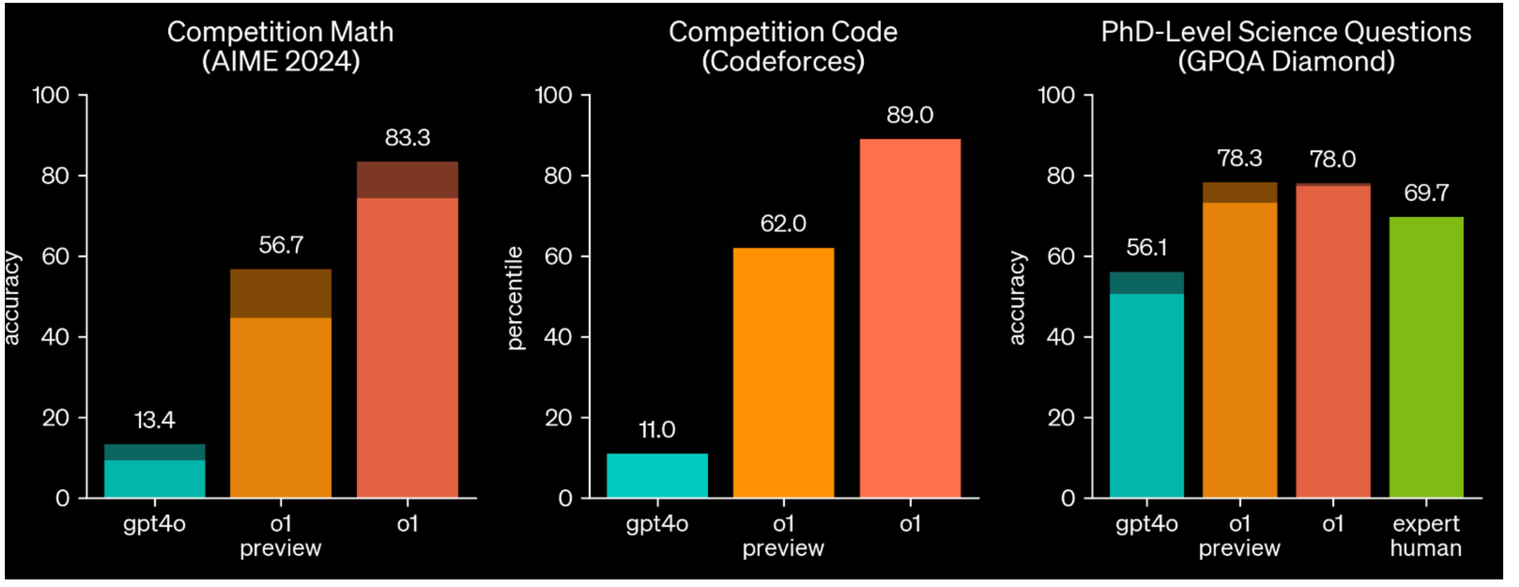
Foundations of Imitation Learning

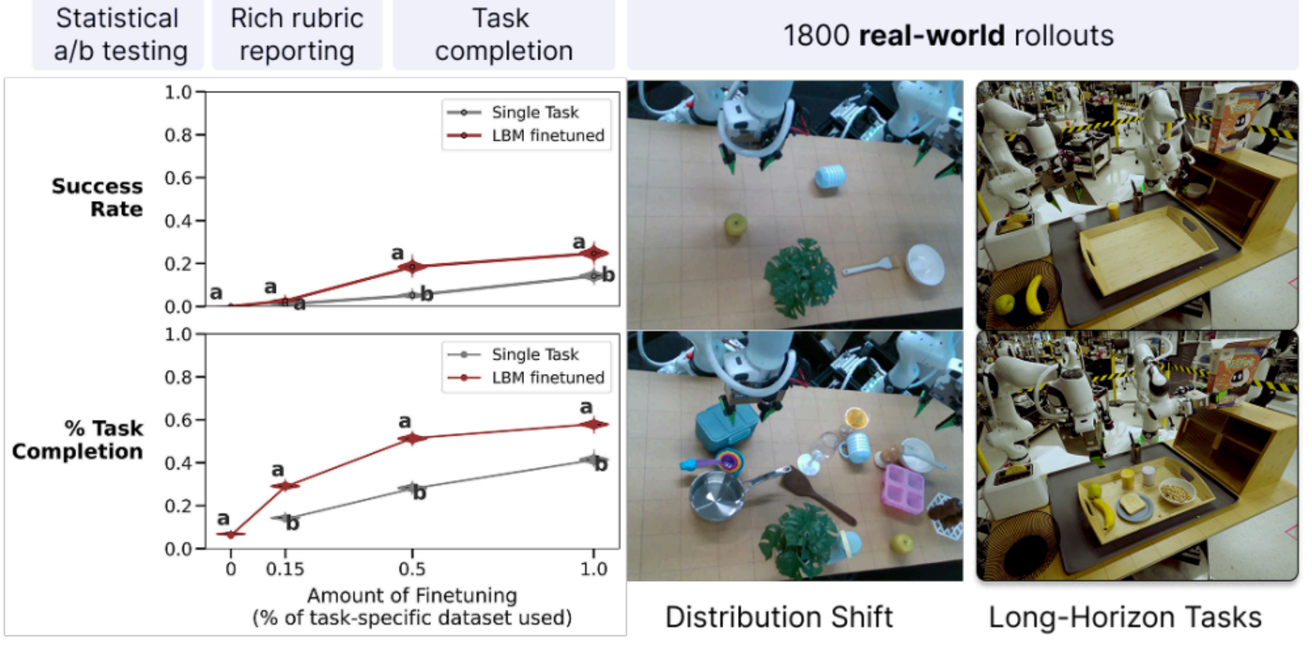
From Language Modeling to Continuous Control

Adam Block, Dylan J. Foster, and Max Simchowitz

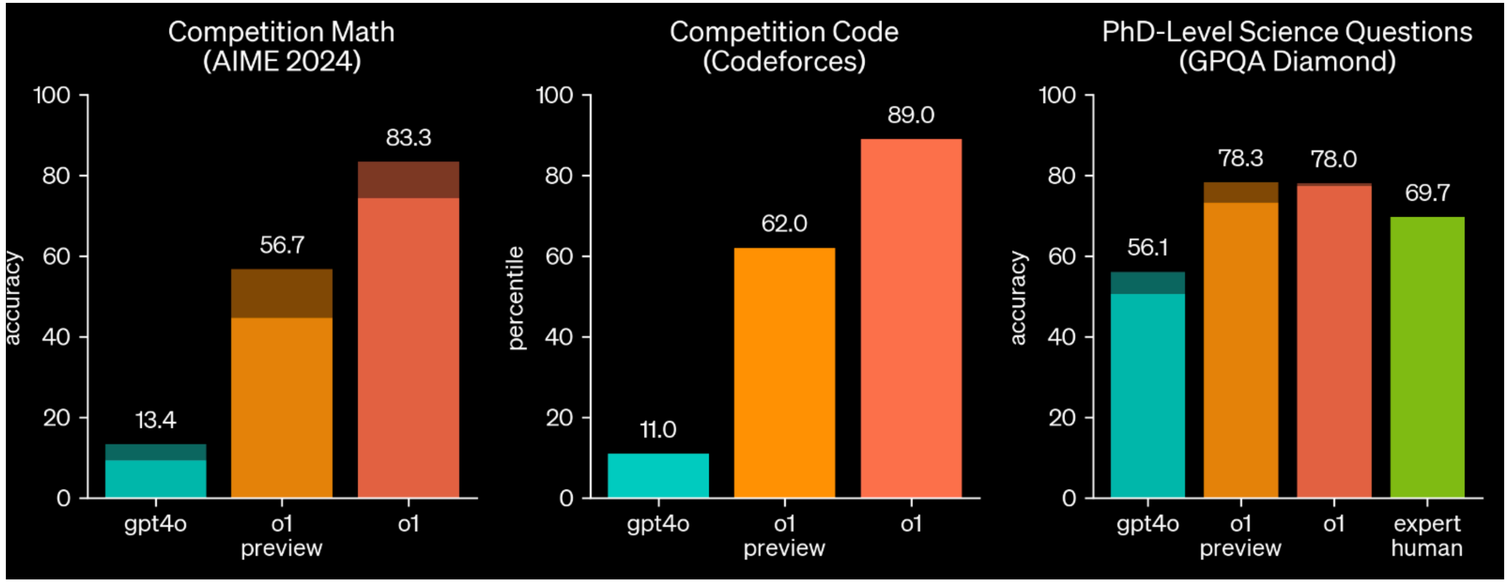


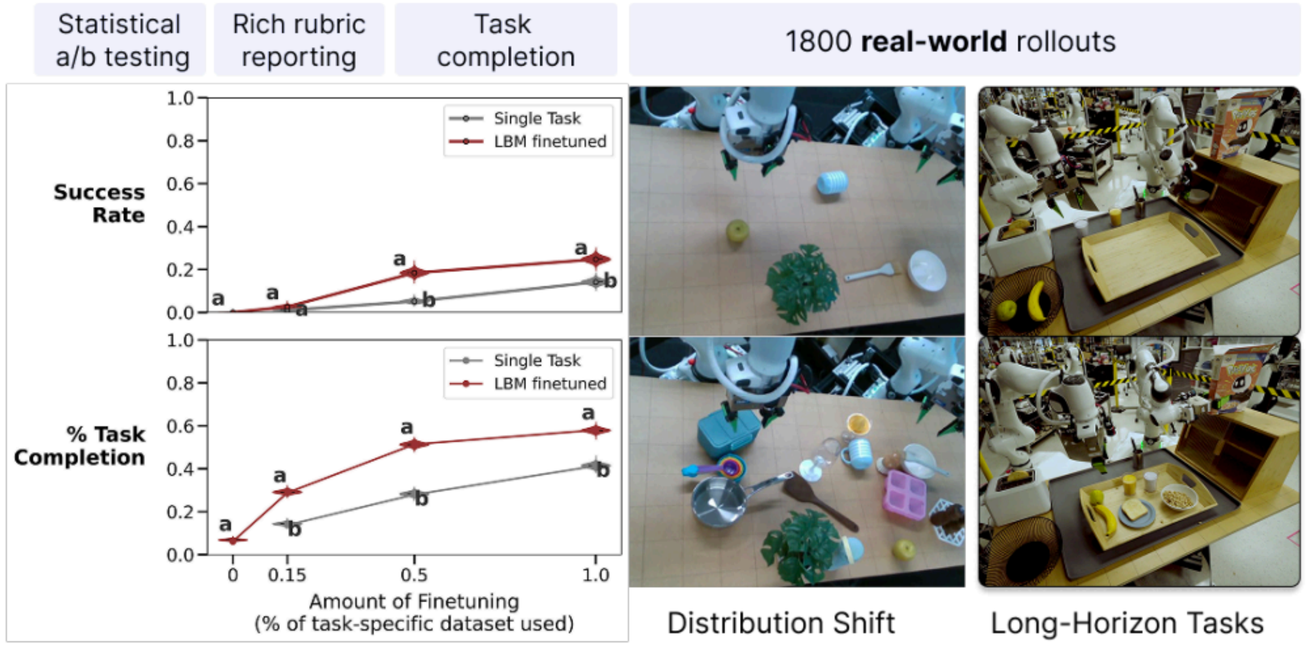
o1 Tech Report (2024); TRI LBM Team (2025); The Waymo Team (2022)

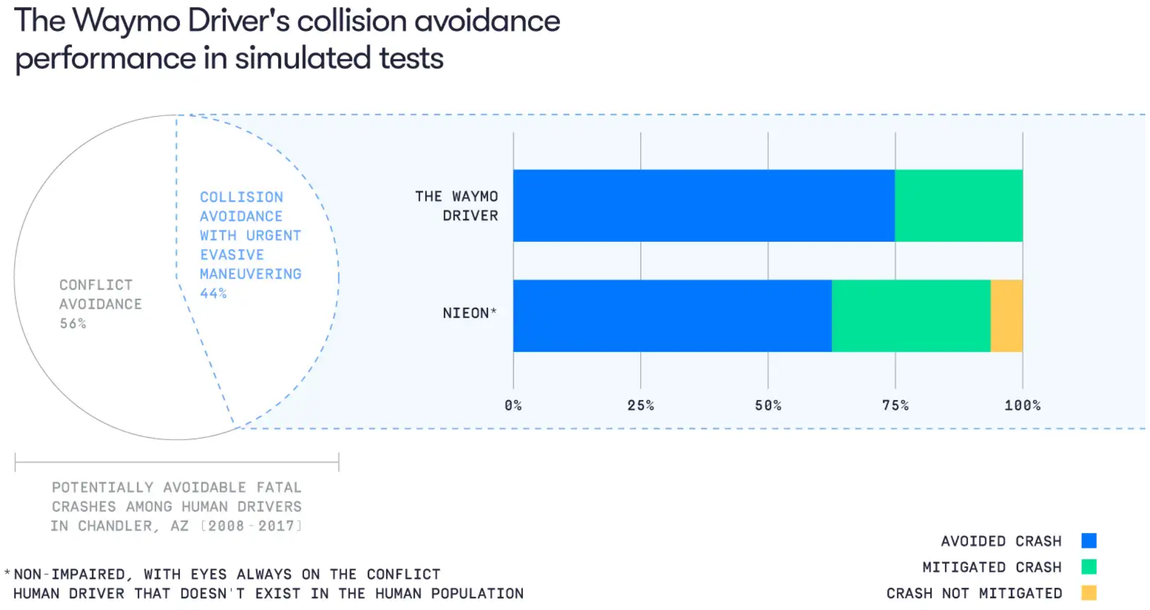




o1 Tech Report (2024); TRI LBM Team (2025); The Waymo Team (2022)







o1 Tech Report (2024); TRI LBM Team (2025); The Waymo Team (2022)

Pre-history

Pre-history

- Decision Trees

Pre-history

- Decision Trees
- Early NLP

Pre-history

- Decision Trees
- Early NLP
- Dynamic Programming

Pre-history

- Decision Trees
- Early NLP
- Dynamic Programming
- SVMs

Pre-history

Supervised and Reinforcement Learning

- Decision Trees
- Early NLP
- Dynamic Programming
- SVMs

Pre-history

Supervised and Reinforcement Learning

2012

- Decision Trees
- Early NLP
- Dynamic Programming
- SVMs

- AlexNet

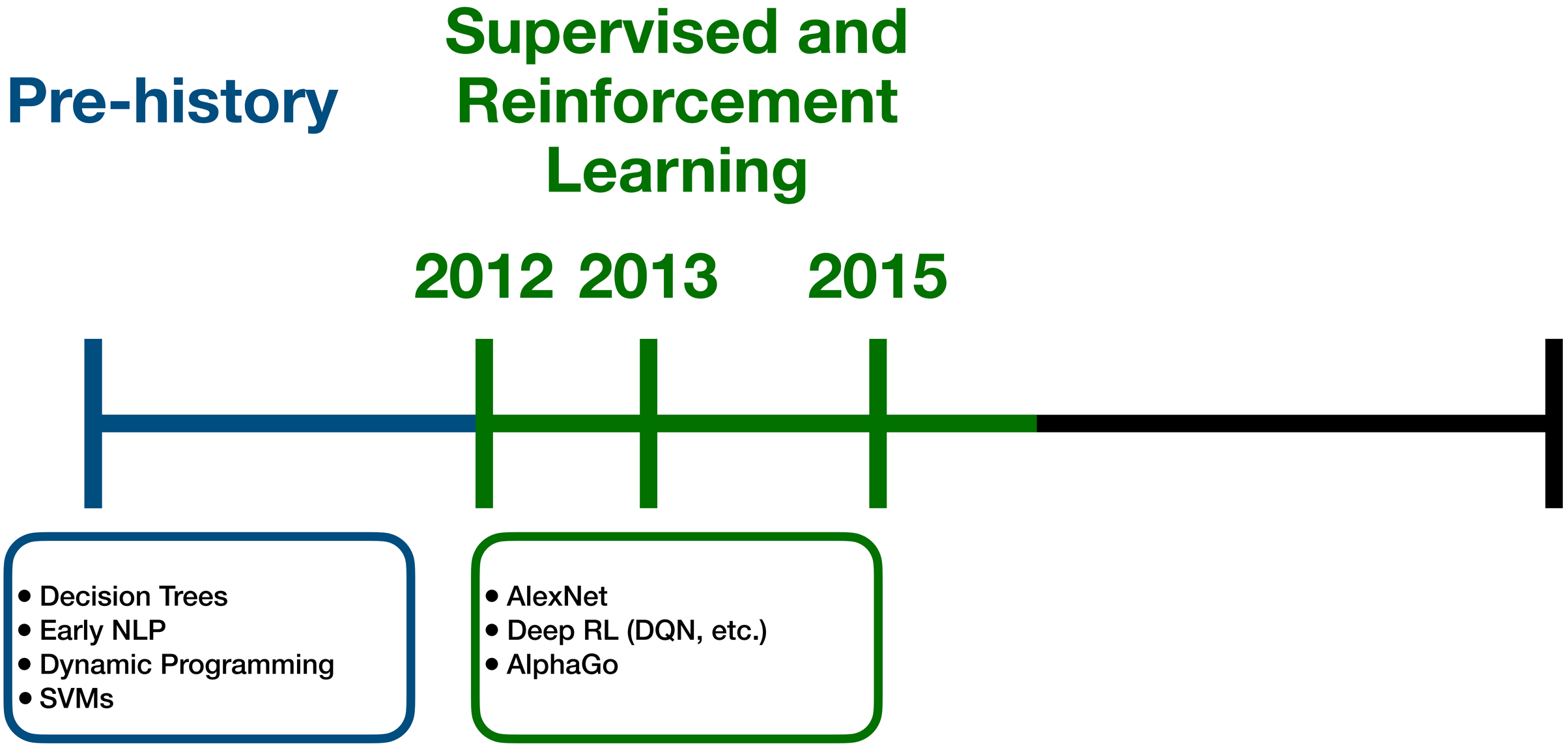
Pre-history

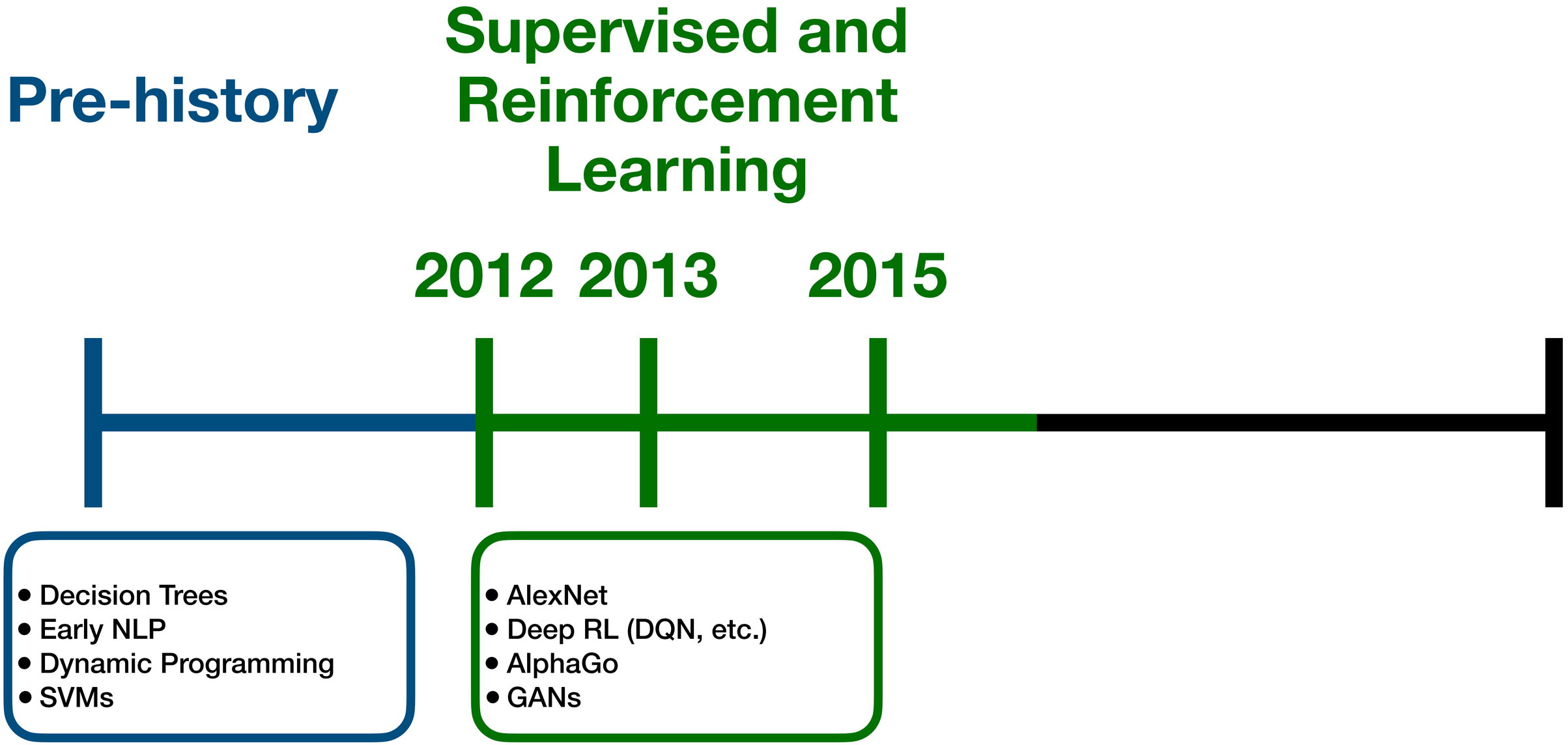
Supervised and Reinforcement Learning

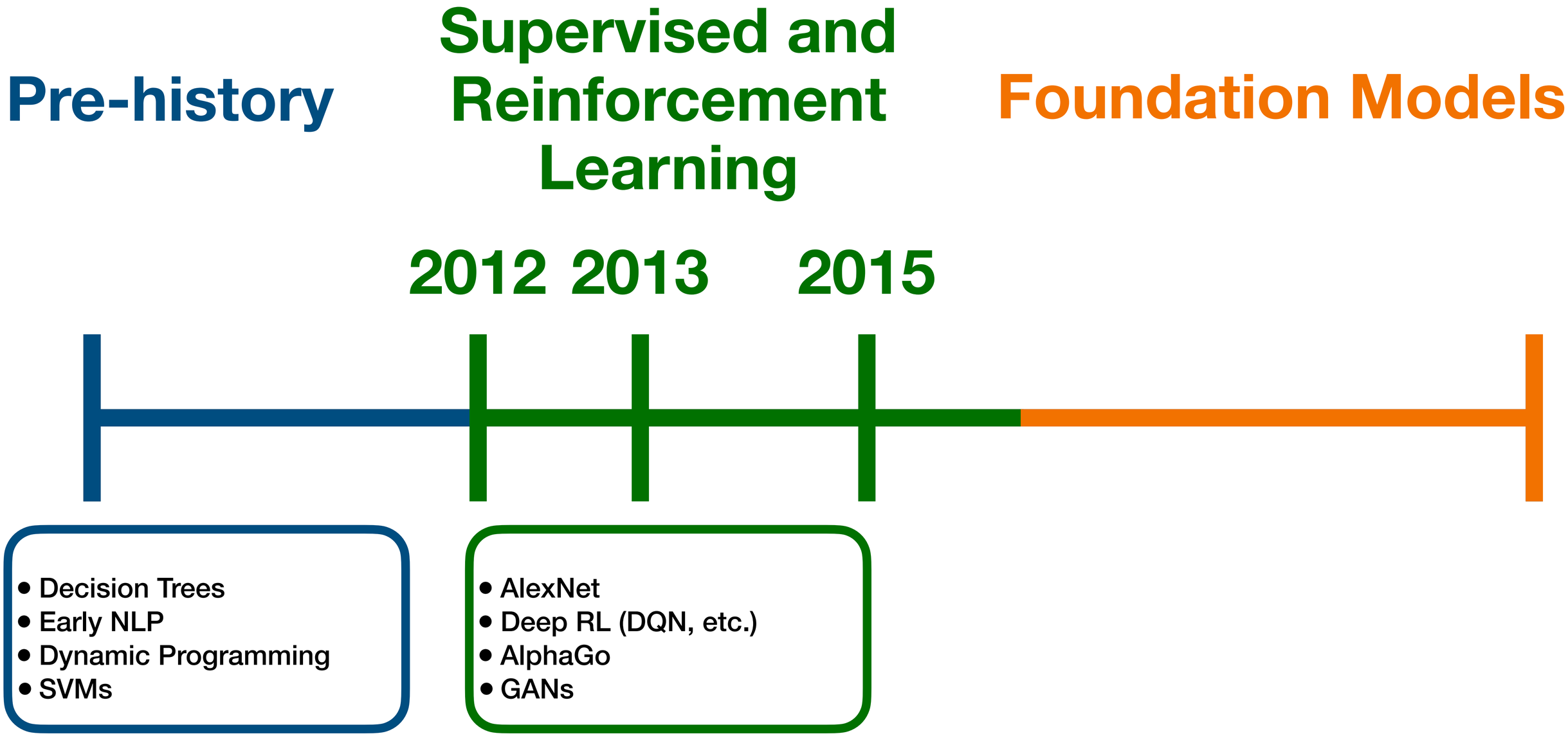
2012 2013

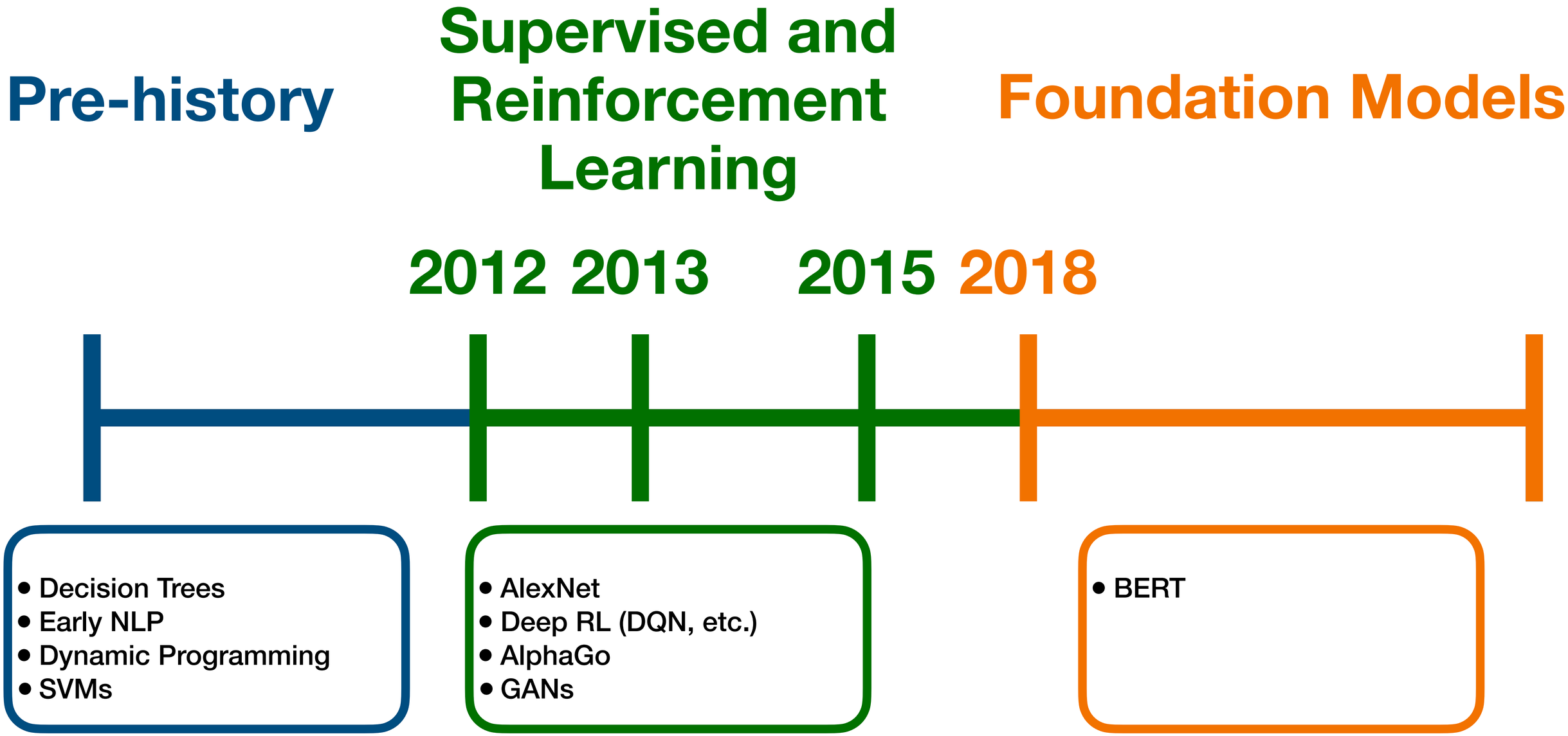
- Decision Trees
- Early NLP
- Dynamic Programming
- SVMs

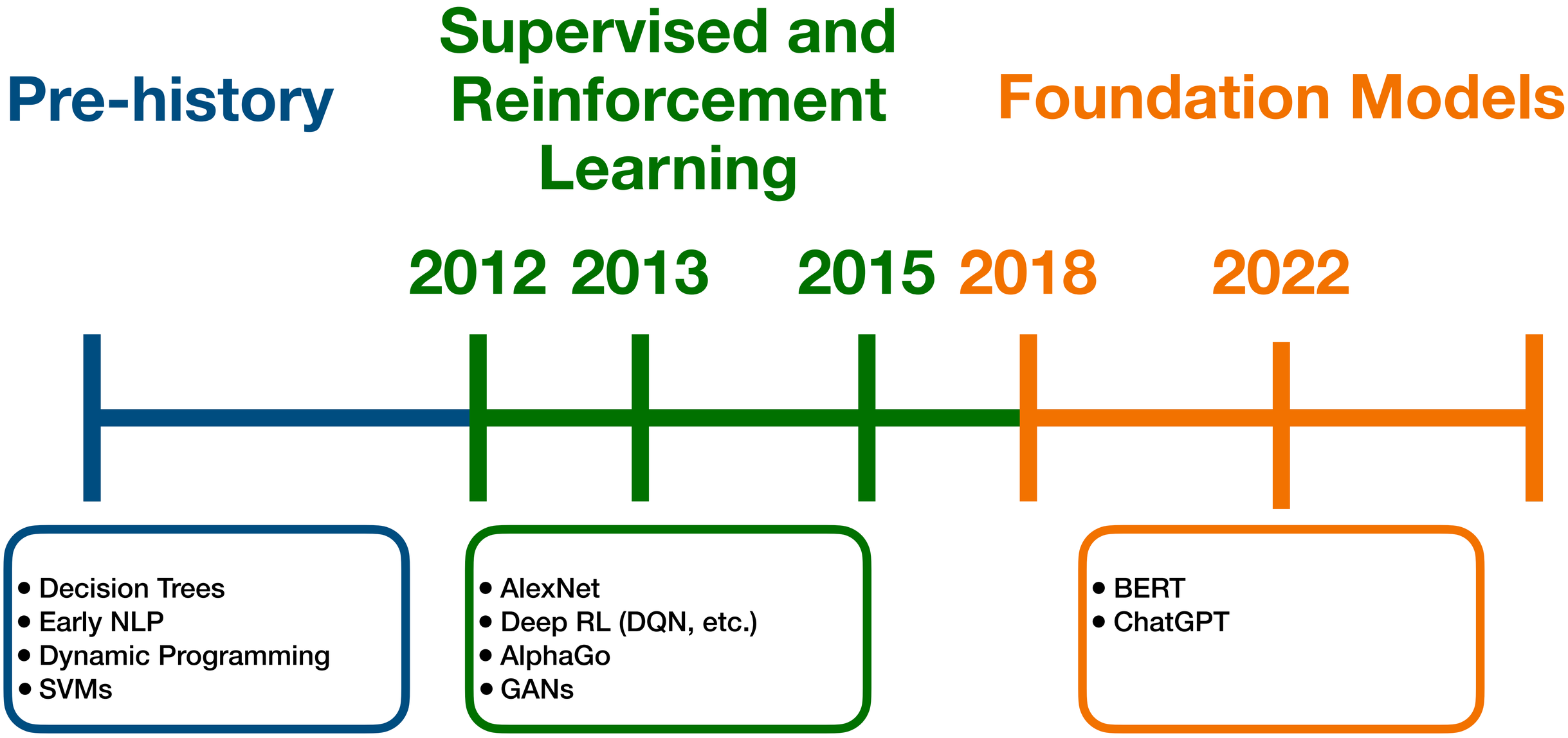
- AlexNet
- Deep RL (DQN, etc.)

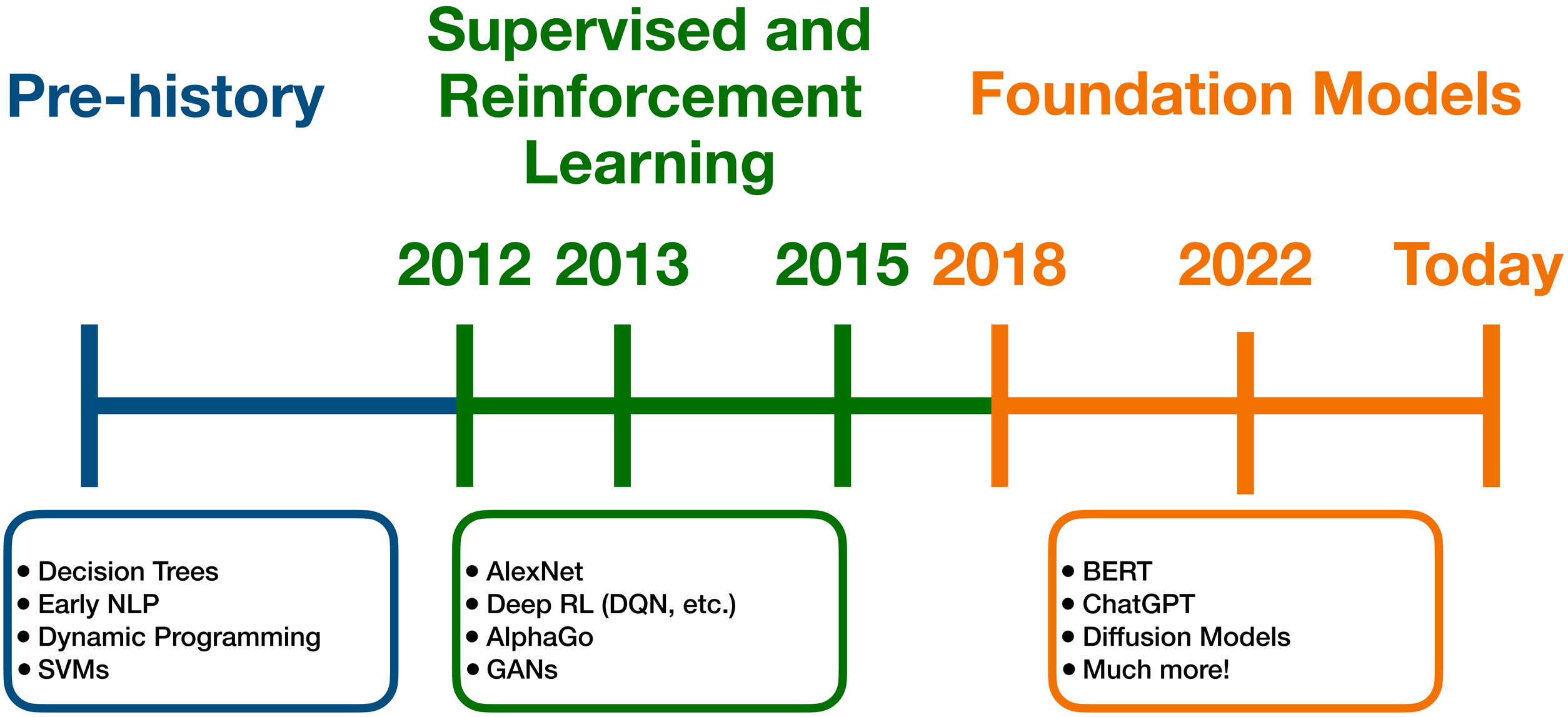

Pre-history
Supervised and Reinforcement Learning
2012
2013
2015
• Decision Trees
• Early NLP
• Dynamic Programming
• SVMs
• AlexNet
• Deep RL (DQN, etc.)
• AlphaGo


Pre-history
Supervised and Reinforcement Learning
2012
2013
2015
• Decision Trees
• Early NLP
• Dynamic Programming
• SVMs
• AlexNet
• Deep RL (DQN, etc.)
• AlphaGo
• GANs


Pre-history
Supervised and Reinforcement Learning
Foundation Models
2012
2013
2015
• Decision Trees
• Early NLP
• Dynamic Programming
• SVMs
• AlexNet
• Deep RL (DQN, etc.)
• AlphaGo
• GANs


Pre-history
Supervised and Reinforcement Learning
Foundation Models
2012
2013
2015
2018
• Decision Trees
• Early NLP
• Dynamic Programming
• SVMs
• AlexNet
• Deep RL (DQN, etc.)
• AlphaGo
• GANs
• BERT


Pre-history
Supervised and Reinforcement Learning
Foundation Models
2012
2013
2015
2018
2022
• Decision Trees
• Early NLP
• Dynamic Programming
• SVMs
• AlexNet
• Deep RL (DQN, etc.)
• AlphaGo
• GANs
• BERT
• ChatGPT

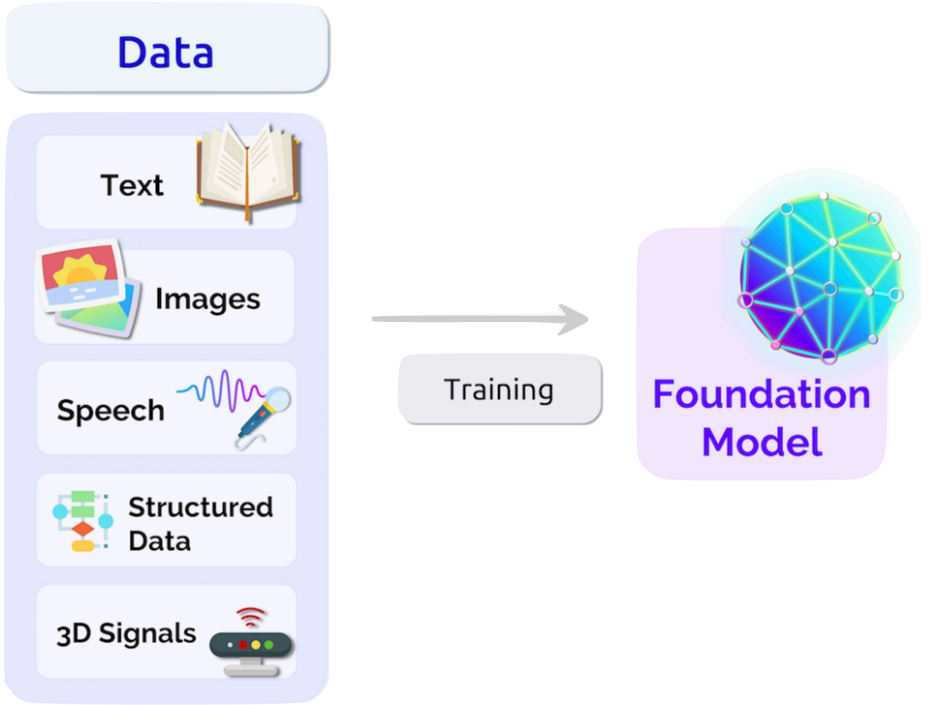

Pre-history
Supervised and Reinforcement Learning
Foundation Models
2012
2013
2015
2018
2022
Today
• Decision Trees
• Early NLP
• Dynamic Programming
• SVMs
• AlexNet
• Deep RL (DQN, etc.)
• AlphaGo
• GANs
• BERT
• ChatGPT
• Diffusion Models
• Much more!

On the Opportunities and Risks of Foundation Models

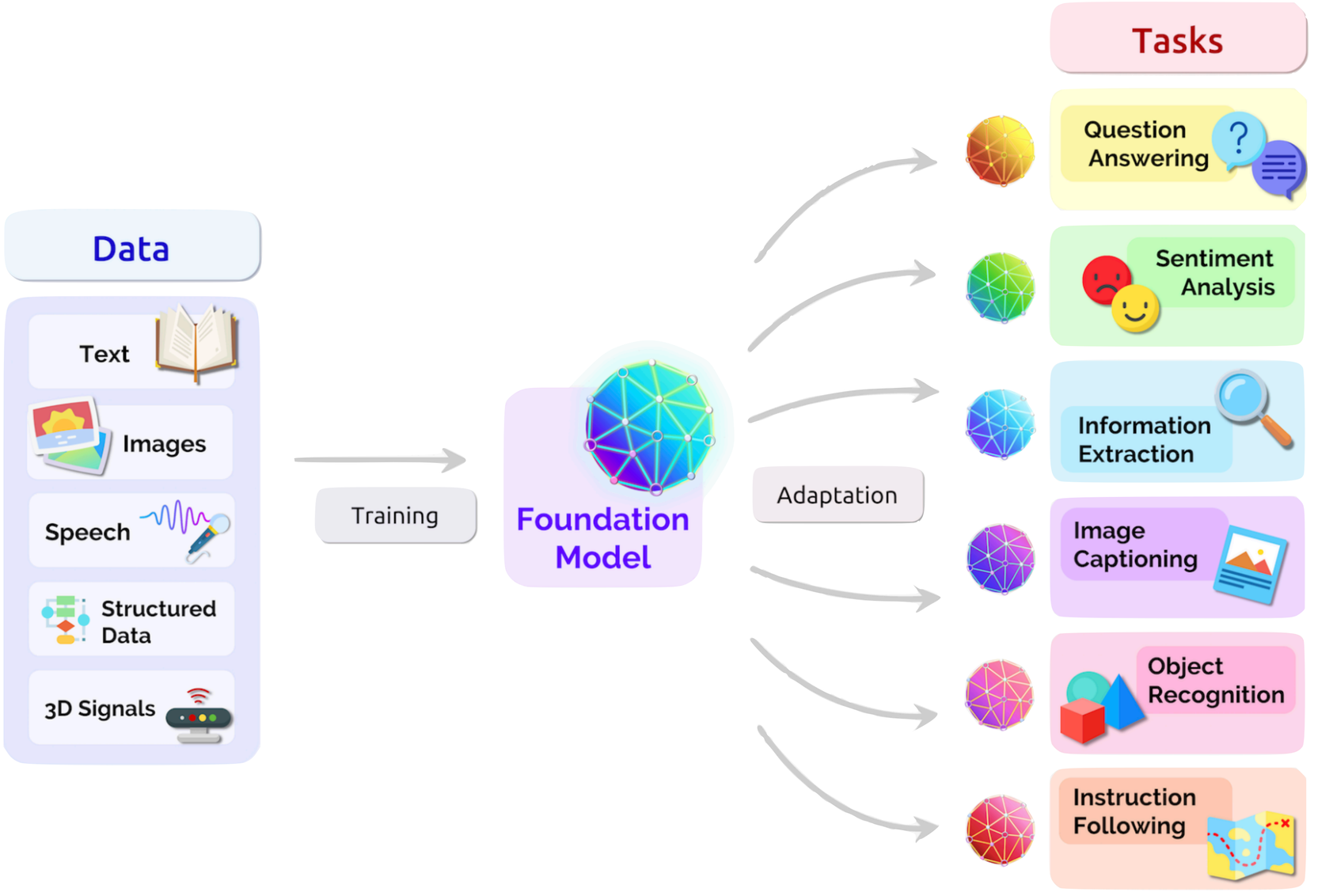
Rishi Bommasani* Drew A. Hudson Ehsan Adeli Russ Altman Simran Arora
Sydney von Arx Michael S. Bernstein Jeannette Bohg Antoine Bosselut Emma Brunskill
Erik Brynjolfsson Shyamal Buch Dallas Card Rodrigo Castellon Niladri Chatterji
Annie Chen Kathleen Creel Jared Quincy Davis Dorottya Demszky Chris Donahue
Moussa Doumbouya Esin Durmus Stefano Ermon John Etchemendy Kawin Ethayarajh
Li Fei-Fei Chelsea Finn Trevor Gale Lauren Gillespie Karan Goel Noah Goodman
Shelby Grossman Neel Guha Tatsunori Hashimoto Peter Henderson John Hewitt
Daniel E. Ho Jenny Hong Kyle Hsu Jing Huang Thomas Icard Saahil Jain
Dan Jurafsky Pratyusha Kalluri Siddharth Karamcheti Geoff Keeling Fereshte Khani
Omar Khattab Pang Wei Koh Mark Krass Ranjay Krishna Rohith Kuditipudi
Ananya Kumar Faisal Ladhak Mina Lee Tony Lee Jure Leskovec Isabelle Levent
Xiang Lisa Li Xuechen Li Tengyu Ma Ali Malik Christopher D. Manning
Suvir Mirchandani Eric Mitchell Zanele Munyikwa Suraj Nair Avanika Narayan
Deepak Narayanan Ben Newman Allen Nie Juan Carlos Niebles Hamed Nilforoshan
Julian Nyarko Giray Ogut Laurel Orr Isabel Papadimitriou Joon Sung Park Chris Piech
Eva Portelance Christopher Potts Aditi Raghunathan Rob Reich Hongyu Ren
Frieda Rong Yusuf Roohani Camilo Ruiz Jack Ryan Christopher Ré Dorsa Sadigh
Shiori Sagawa Keshav Santhanam Andy Shih Krishnan Srinivasan Alex Tamkin
Rohan Taori Armin W. Thomas Florian Tramèr Rose E. Wang William Wang Bohan Wu
Jiajun Wu Yuhuai Wu Sang Michael Xie Michihiro Yasunaga Jiaxuan You Matei Zaharia
Michael Zhang Tianyi Zhang Xikun Zhang Yuhui Zhang Lucia Zheng Kaitlyn Zhou
Percy Liang*[1]

Center for Research on Foundation Models (CRFM)
Stanford Institute for Human-Centered Artificial Intelligence (HAI)
Stanford University

Bomasanni et al (2021)

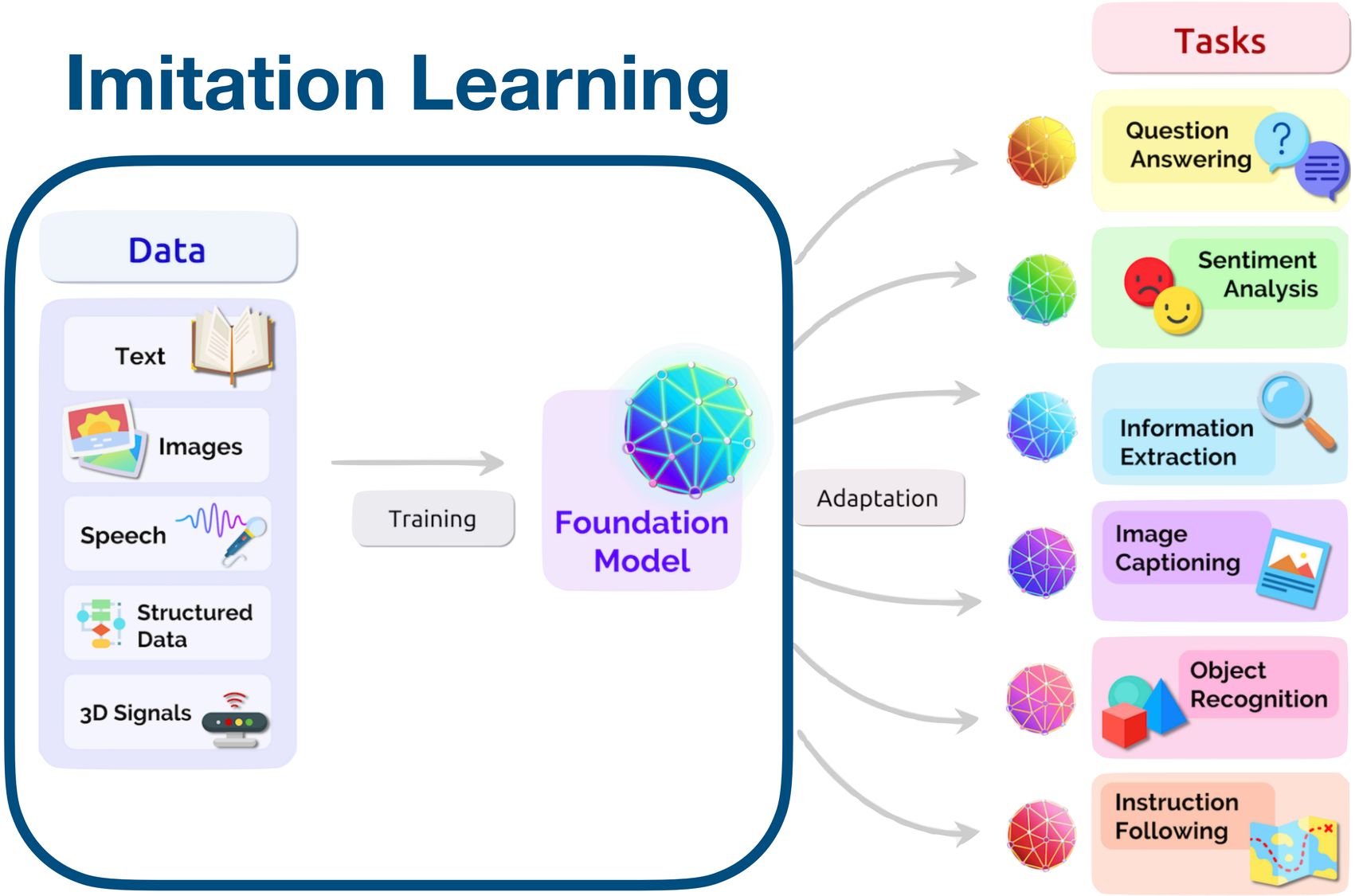


Bomasanni et al (2021)



Bomasanni et al (2021)

Imitation Learning



Bomasanni et al (2021)

1. Why have we converged on such a **unified paradigm** to address different **problem settings**?

1. Why have we converged on such a **unified paradigm** to address different **problem settings**?

2. When does **Imitation Learning** succeed and how is it different from **supervised learning**?

1. Why have we converged on such a **unified paradigm** to address different **problem settings**?

2. When does **Imitation Learning** succeed and how is it different from **supervised learning**?

3. How can we leverage **theoretical principles** to (empirically) **improve** Imitation Learning?

Outline

Outline

I. Background and Introduction.

Outline

I. Background and Introduction.

II. Classical Imitation Learning.

Outline

I. Background and Introduction.

II. Classical Imitation Learning.

III. Modern IL in discrete settings.

Outline

I. Background and Introduction.

II. Classical Imitation Learning.

III. Modern IL in discrete settings.

IV. Modern IL in continuous settings.

Outline

I. Background and Introduction.

II. Classical Imitation Learning.

III. Modern IL in discrete settings.

IV. Modern IL in continuous settings.

V. Opening the black box of optimization.

Outline

I. Background and Introduction.

II. Classical Imitation Learning.

III. Modern IL in discrete settings.

IV. Modern IL in continuous settings.

V. Opening the black box of optimization.

Reinforcement Learning

Goal: Learn a policy π that gets good reward over a trajectory.

Reinforcement Learning

Goal: Learn a policy π that gets good reward over a trajectory.

Reinforcement Learning: Observe states s, actions a, and rewards $r(s, a)$.

Reinforcement Learning

Goal: Learn a policy π that gets good reward over a trajectory.

Reinforcement Learning: Observe states s, actions a, and rewards $r(s, a)$.

Explore environments *tabula rasa*.

Reinforcement Learning

Goal: Learn a policy π that gets good reward over a trajectory.

Reinforcement Learning: Observe states s, actions a, and rewards $r(s, a)$.

Explore environments *tabula rasa*.

Capable of learning new behaviors and improving rewards.

Reinforcement Learning

Goal: Learn a policy π that gets good reward over a trajectory.

Reinforcement Learning: Observe states s, actions a, and rewards $r(s, a)$.

Explore environments *tabula rasa*.

Capable of learning new behaviors and improving rewards.

Reward design can be challenging.

Reinforcement Learning

Goal: Learn a policy π that gets good reward over a trajectory.

Reinforcement Learning: Observe states s, actions a, and rewards $r(s, a)$.

Explore environments *tabula rasa*.

Capable of learning new behaviors and improving rewards.

Reward design can be challenging.

Exploration is hard in large environments.

Imitation Learning

Goal: Learn a policy π that gets good reward over a trajectory.

Imitation Learning

Goal: Learn a policy π that gets good reward over a trajectory.

Imitation Learning: Observe states s, actions a from **good policy** $\pi^\star$.

Imitation Learning

Goal: Learn a policy π that gets good reward over a trajectory.

Imitation Learning: Observe states s, actions a from **good policy** $\pi^\star$.

No need to explore.

Imitation Learning

Goal: Learn a policy π that gets good reward over a trajectory.

Imitation Learning: Observe states s, actions a from **good policy** $\pi^\star$.

No need to explore.

Leverages expert assistance to remove need for rewards.

Imitation Learning

Goal: Learn a policy π that gets good reward over a trajectory.

Imitation Learning: Observe states s, actions a from **good policy** $\pi^\star$.

No need to explore.

Leverages expert assistance to remove need for rewards.

Reduces decision making to a supervised learning problem.

Behavior Cloning: 1988

ALVINN:
AN AUTONOMOUS LAND VEHICLE IN A NEURAL NETWORK

Dean A. Pomerleau
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

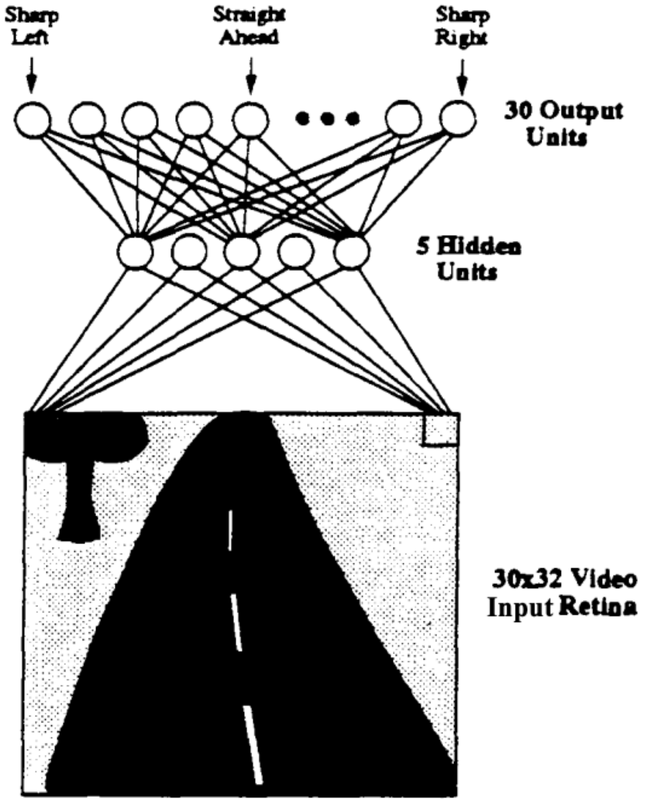


Figure 2: ALVINN Architecture

Behavior Cloning

Given: Expert trajectories, supervised learning loss, and policy class.

Behavior Cloning

Given: Expert trajectories, supervised learning loss, and policy class.

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathrm{BC}}(\pi)$.

Behavior Cloning

Given: Expert trajectories, supervised learning loss, and policy class.

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathrm{BC}}(\pi)$ Supervised learning loss

Behavior Cloning

Given: Expert trajectories, supervised learning loss, and policy class.

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathrm{BC}}(\pi)$ Supervised learning loss

Idea: Predict expert actions given states using classification or regression.

Behavior Cloning: Language Models

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$.

Cross-entropy

Behavior Cloning: Language Models

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$.

Cross-entropy

$$L_{\mathsf{BC}}(\pi) = -\sum_{i=1}^{n} \sum_{h=1}^{H} \log\left(\pi\left(a_h^{(i)} \mid s_h^{(i)}\right)\right)$$

Behavior Cloning: Language Models

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$.

Cross-entropy

$$L_{\mathsf{BC}}(\pi) = -\sum_{i=1}^{n} \sum_{h=1}^{H} \log\left(\pi\left(a_h^{(i)} \mid s_h^{(i)}\right)\right)$$

The quick brown fox jumps over the lazy

State: $s_h^{(i)}$

Behavior Cloning: Language Models

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$.

Cross-entropy

$$L_{\mathsf{BC}}(\pi) = -\sum_{i=1}^{n} \sum_{h=1}^{H} \log\left(\pi\left(a_h^{(i)} \mid s_h^{(i)}\right)\right)$$

The quick brown fox jumps over the lazy dog.

State: $s_h^{(i)}$ **Action:** $a_h^{(i)}$

Behavior Cloning: Robotics and Control

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$.

Mean Square Error

Behavior Cloning: Robotics and Control

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$.

Mean Square Error

$$L_{\mathsf{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \|\pi(s_h^{(i)}) - a_h^{(i)}\|^2$$

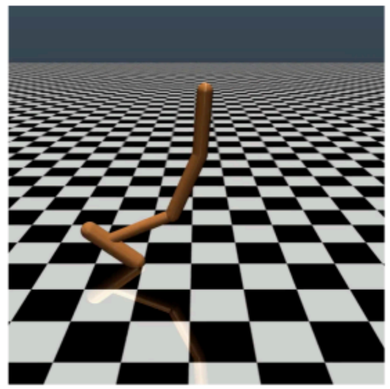
Behavior Cloning: Robotics and Control

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$.

Mean Square Error

$$L_{\mathsf{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \|\pi(s_h^{(i)}) - a_h^{(i)}\|^2$$



State: $s_h^{(i)}$

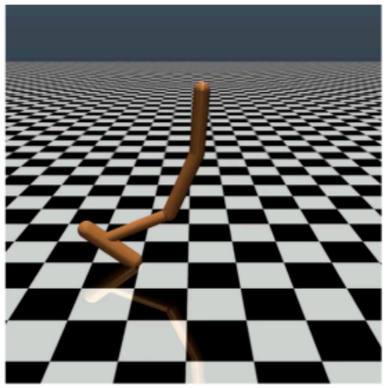
Behavior Cloning: Robotics and Control

Behavior Cloning (BC) [P'89;RB'10]

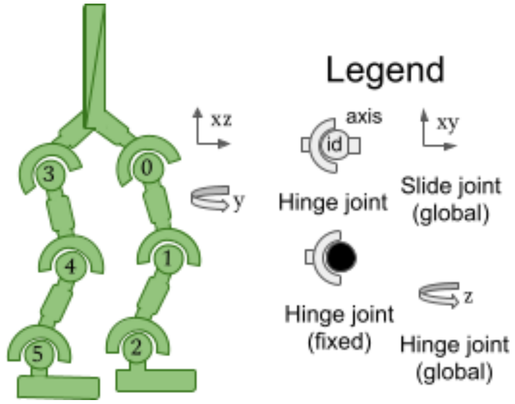
Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathrm{BC}}(\pi)$.

Mean Square Error

$$L_{\mathrm{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \|\pi(s_h^{(i)}) - a_h^{(i)}\|^2$$



State: $s_h^{(i)}$



Action: $a_h^{(i)}$

Behavior Cloning

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$. Supervised learning loss

Idea: Hope that

$$\text{Suboptimality of } \hat{\pi} \lesssim L_{\mathsf{BC}}(\hat{\pi}).$$

Behavior Cloning

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$. Supervised learning loss

Idea: Hope that

$$\text{Suboptimality of } \hat{\pi} \lesssim L_{\mathsf{BC}}(\hat{\pi}).$$

Supervised learning ensures $L_{\mathsf{BC}}(\hat{\pi}) \ll 1$.

Behavior Cloning

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$. Supervised learning loss

Idea: Hope that

$$\text{Suboptimality of } \hat{\pi} \lesssim L_{\mathsf{BC}}(\hat{\pi}).$$

Supervised learning ensures $L_{\mathsf{BC}}(\hat{\pi}) \ll 1$.

How is this different from supervised learning?

Behavior Cloning

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$. Supervised learning loss

Idea: Hope that

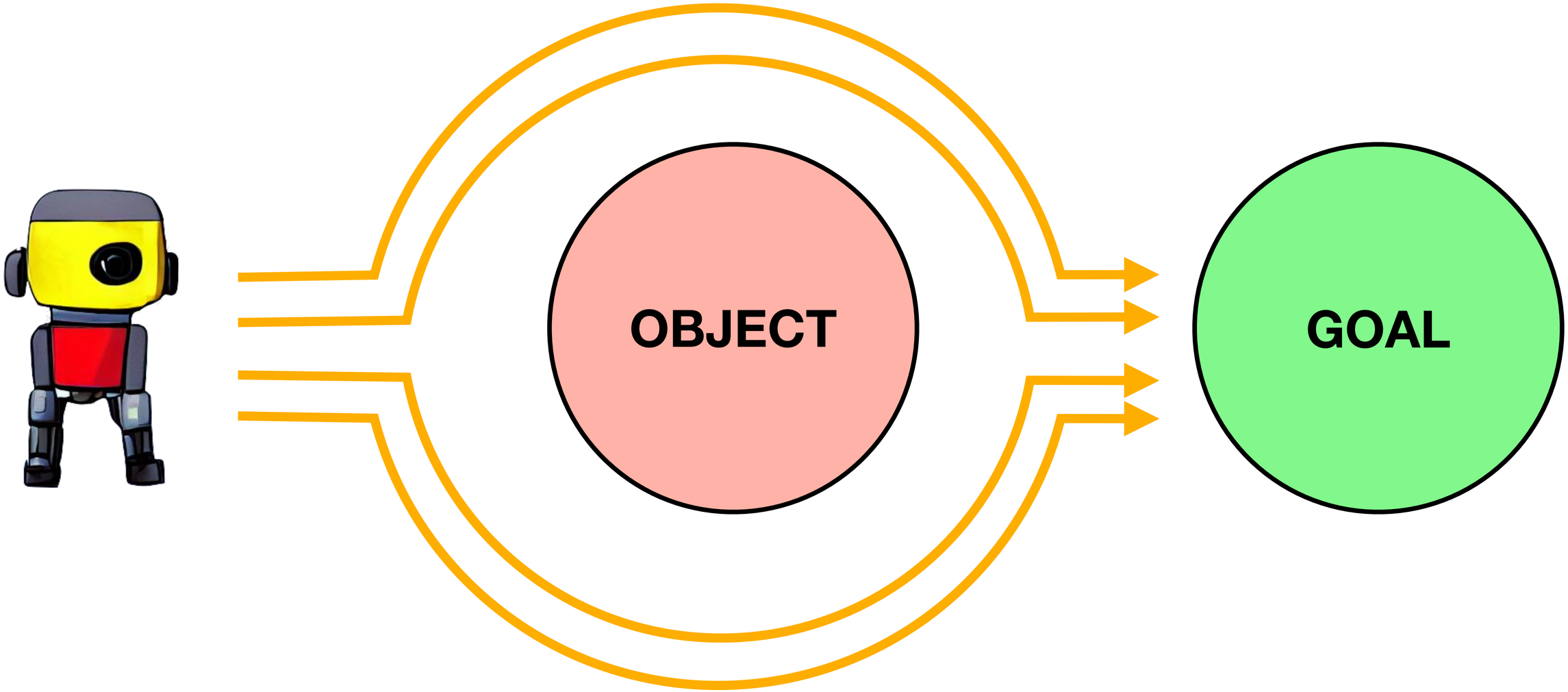
$$\text{Suboptimality of } \hat{\pi} \lesssim L_{\mathsf{BC}}(\hat{\pi}).$$

Supervised learning ensures $L_{\mathsf{BC}}(\hat{\pi}) \ll 1$.

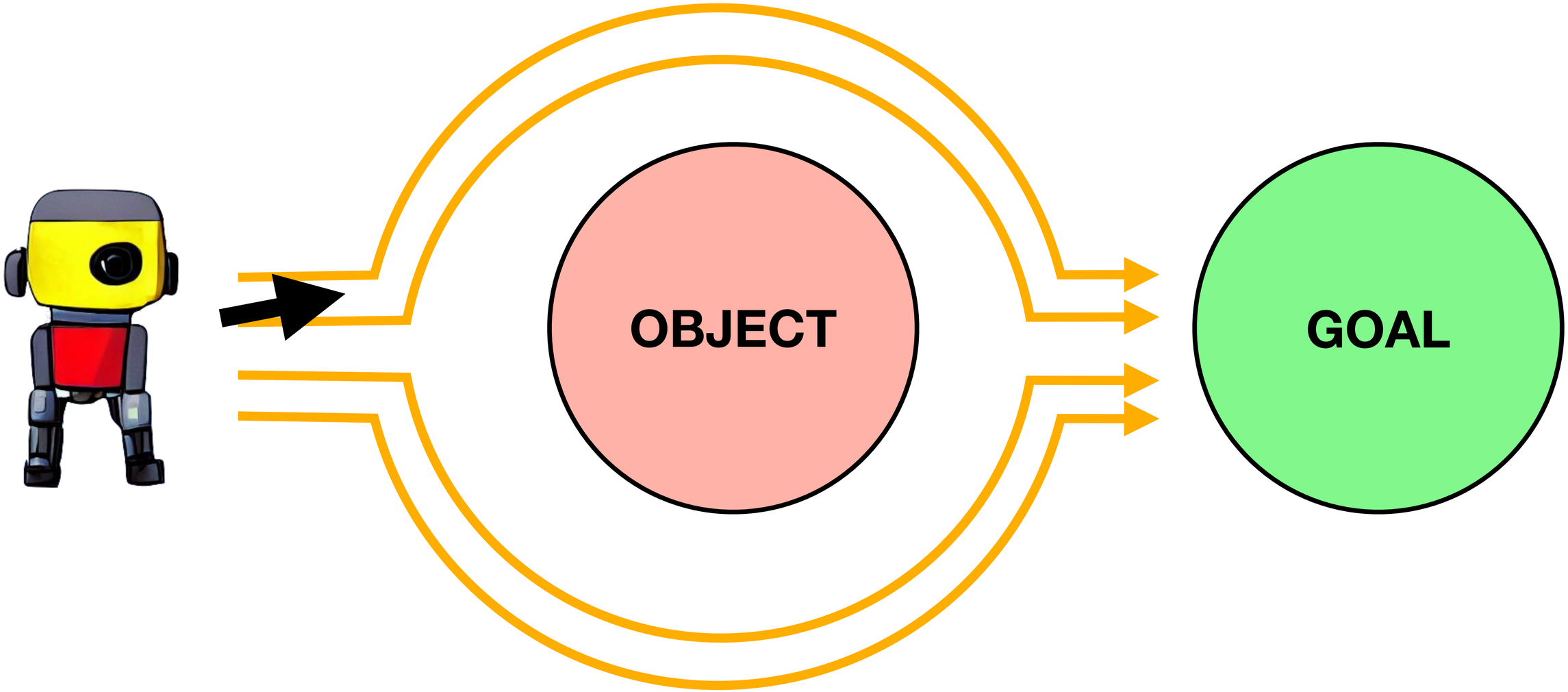
How is this different from supervised learning?

Error amplification!

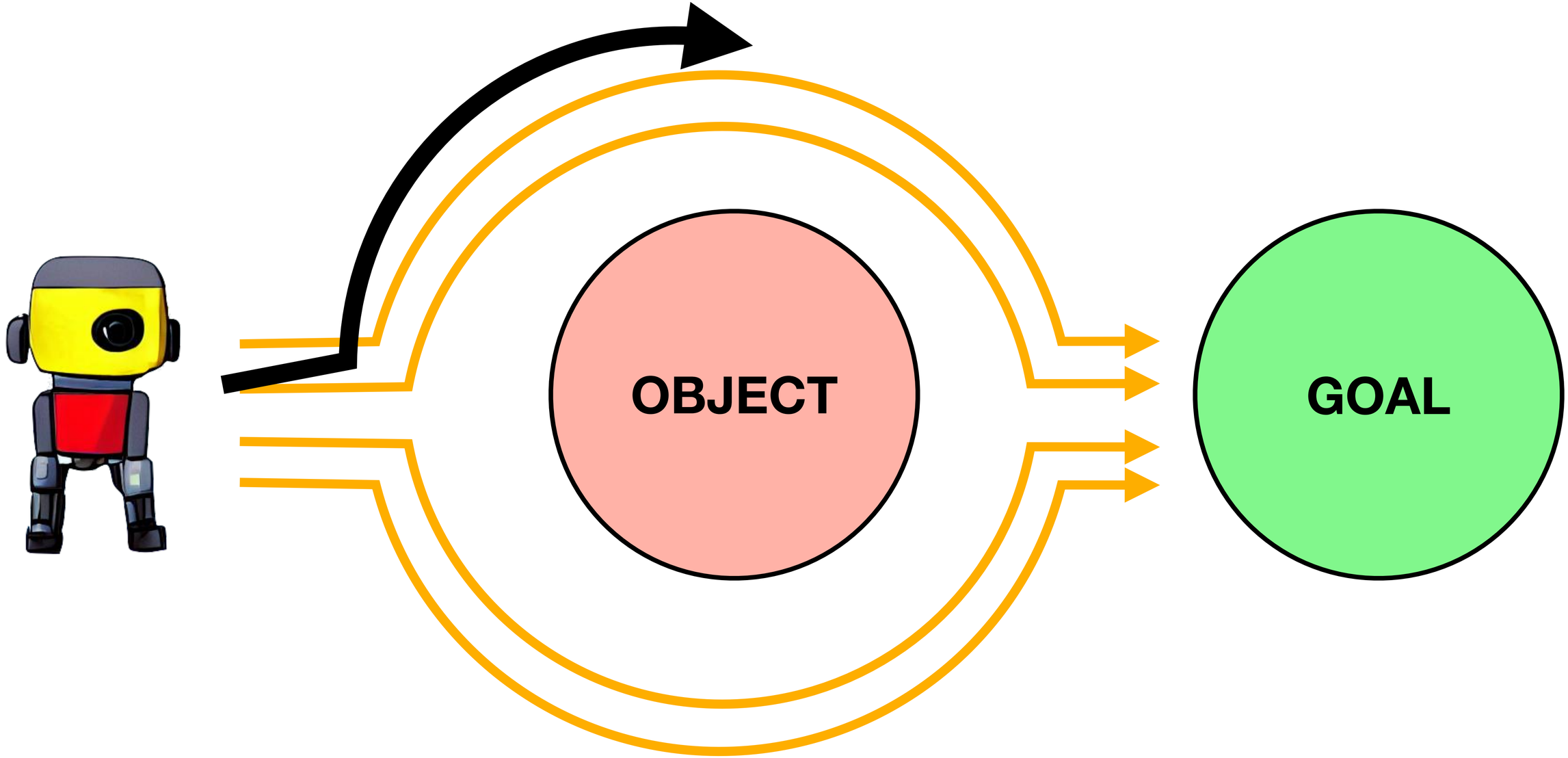
Error amplification



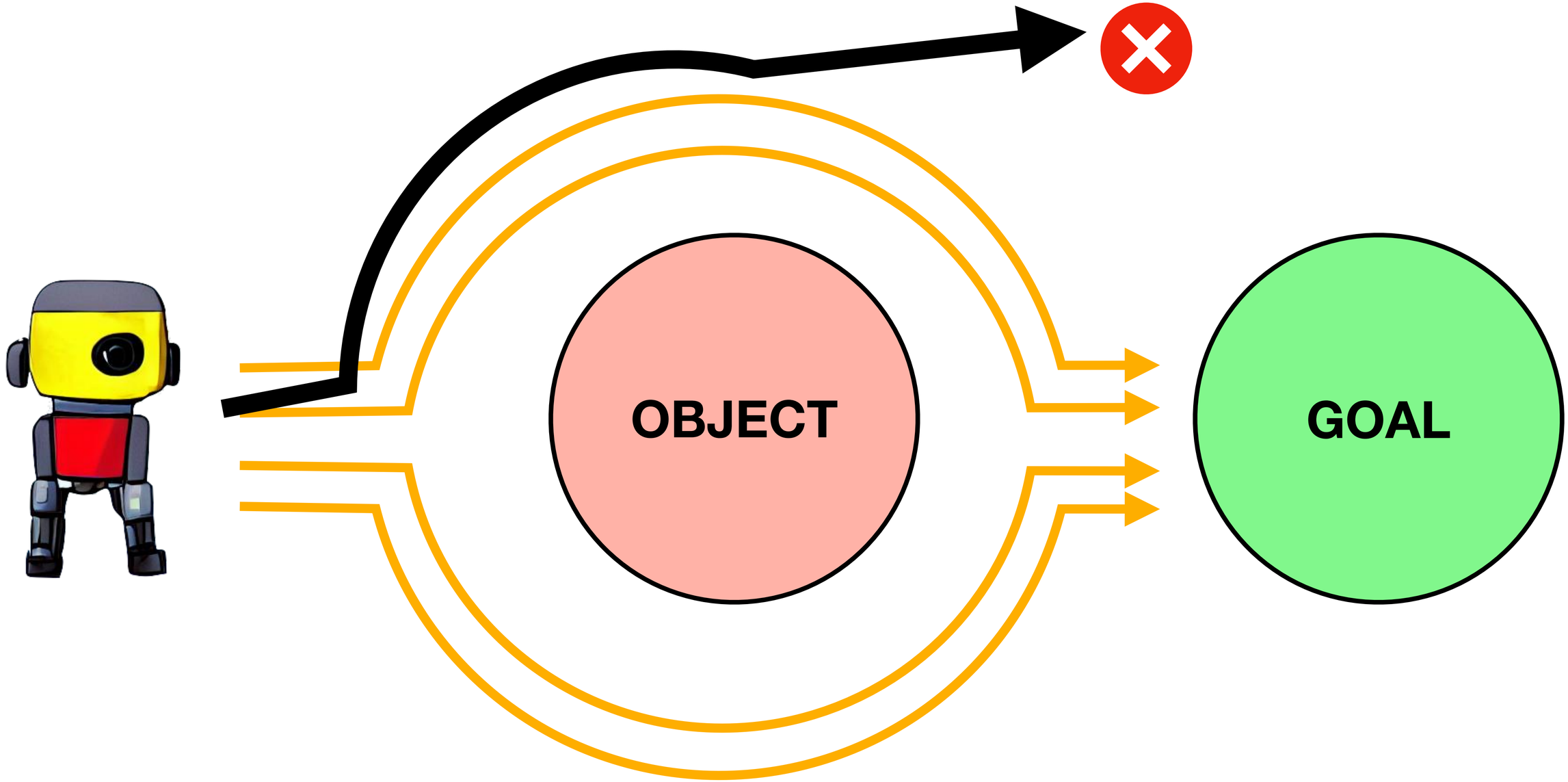
Error amplification



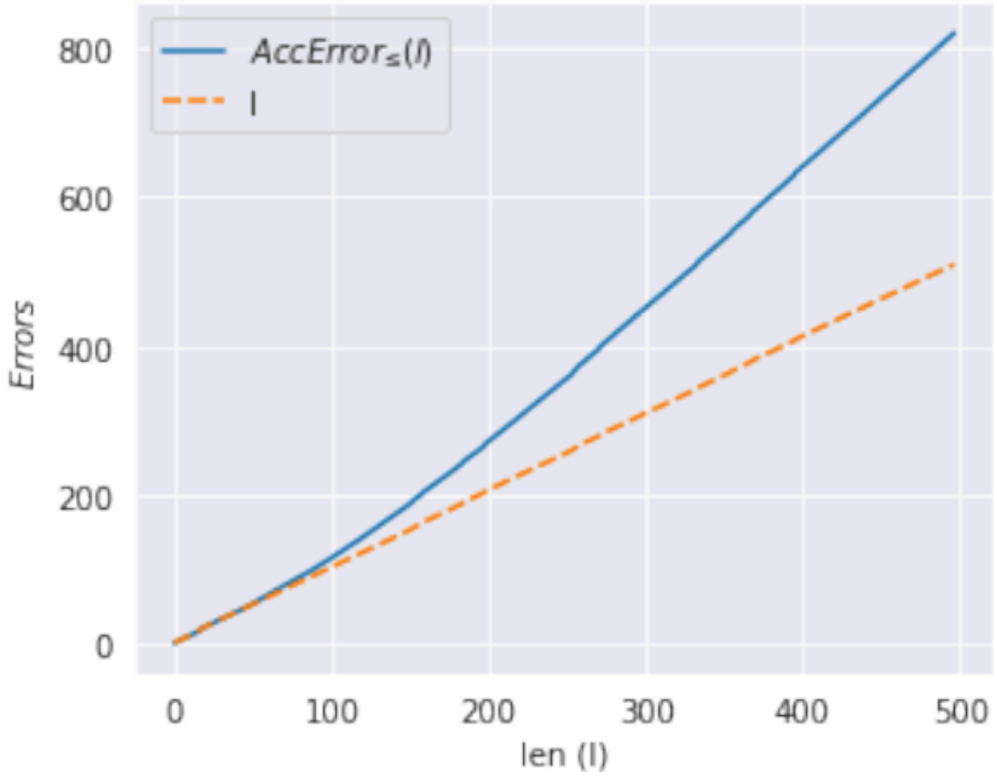
Error amplification

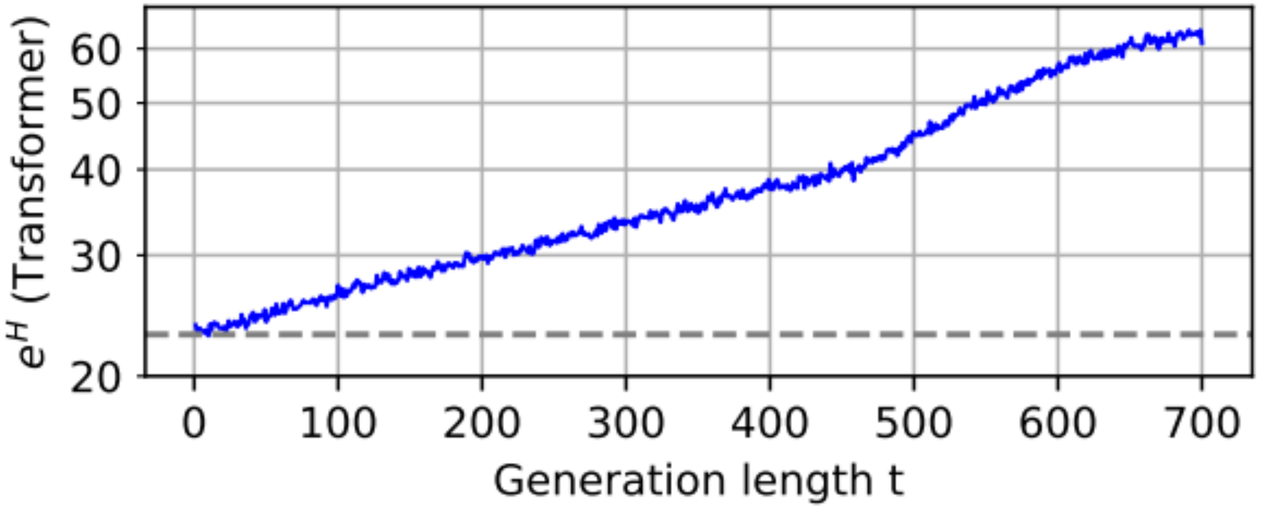


Error amplification



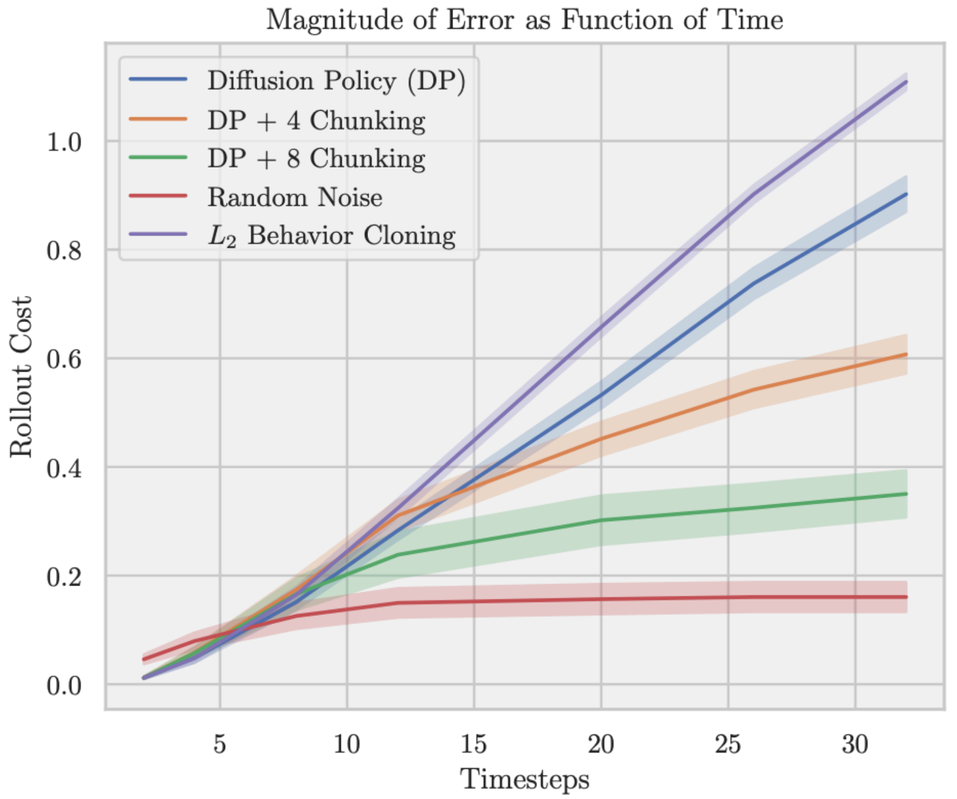
Error Amplification in LLMs





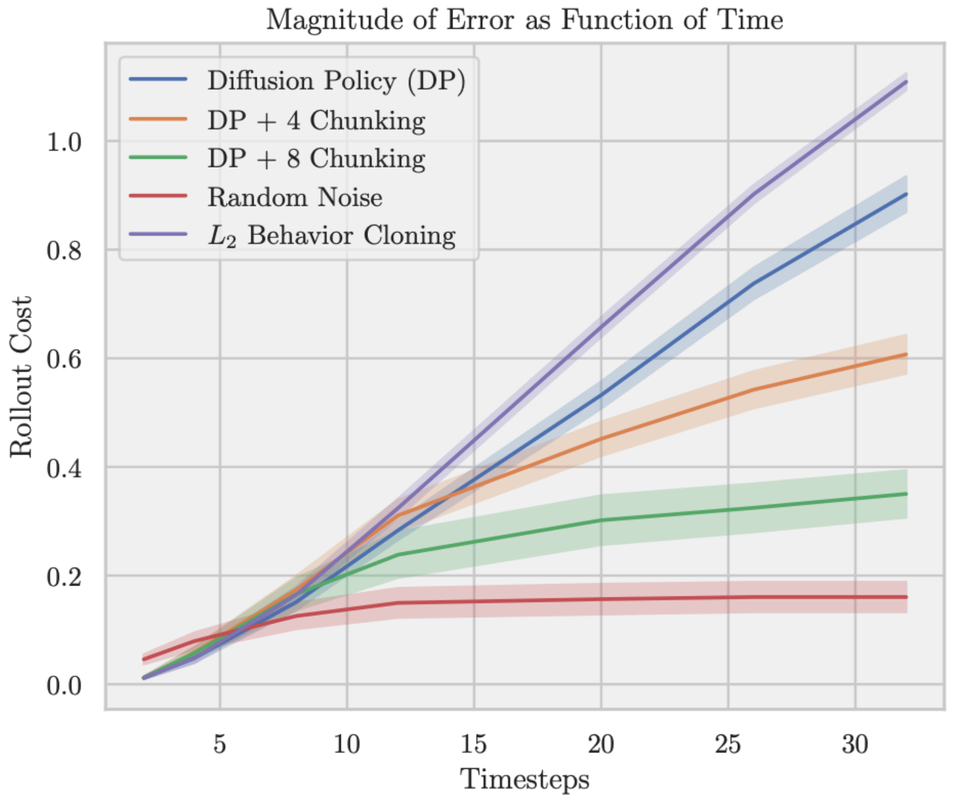
Arora et al (2023); Braverman et al (2019)

Error Amplification in Robotics and Control

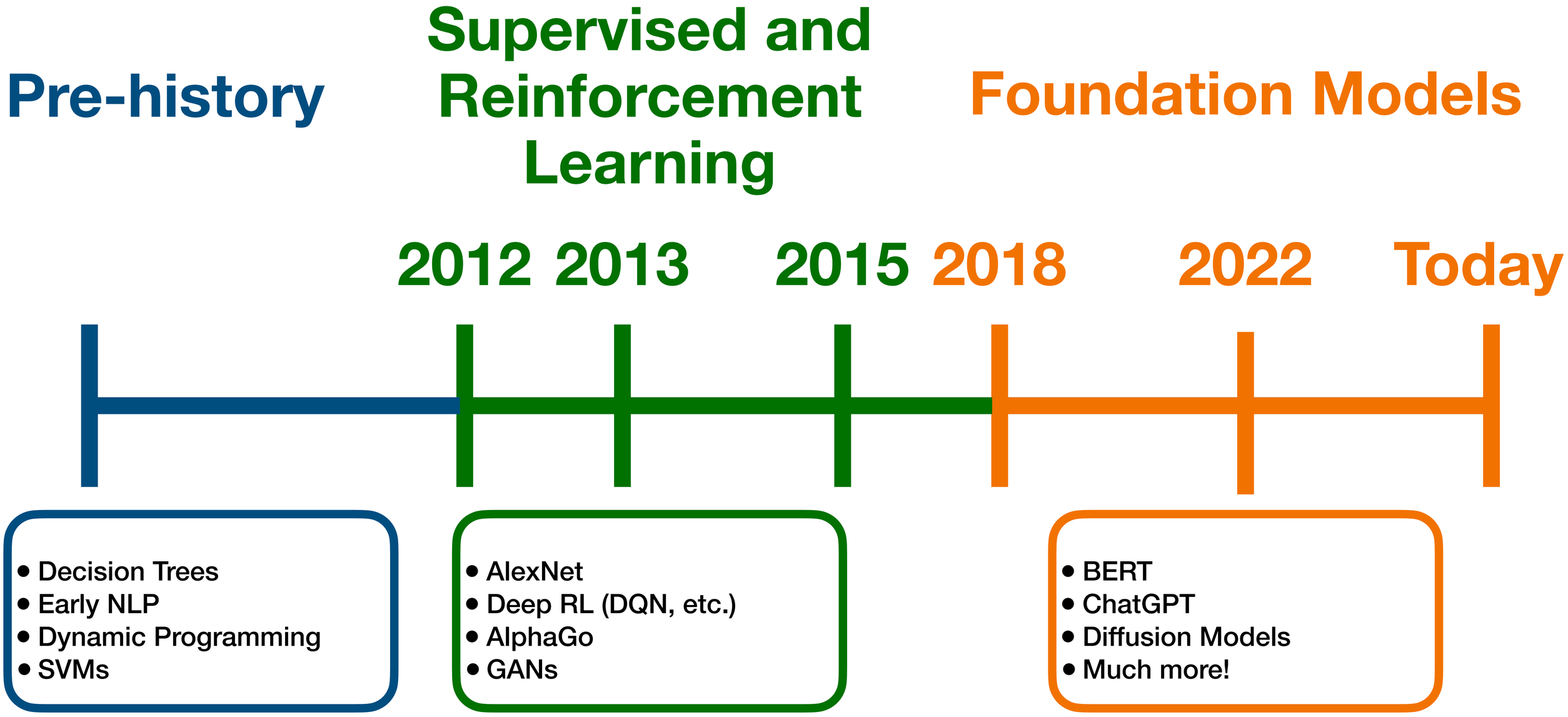


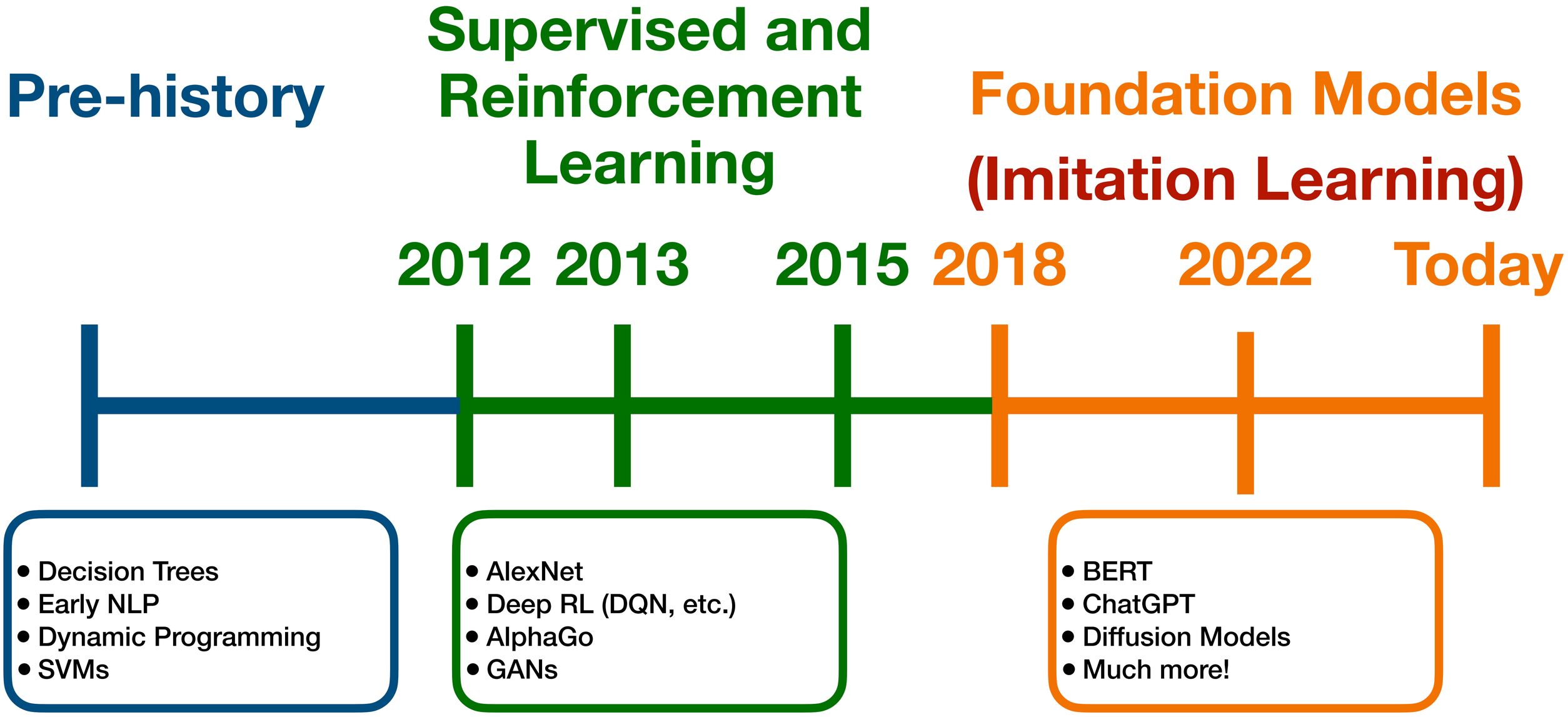
Pomerleau (1988), Laskey et al (2017); **Simchowitz** et al (2025)

Error Amplification in Robotics and Control



Pomerleau (1988), Laskey et al (2017); **Simchowitz** et al (2025)


Pre-history
Supervised and Reinforcement Learning
Foundation Models
2012
2013
2015
2018
2022
Today
• Decision Trees
• Early NLP
• Dynamic Programming
• SVMs
• AlexNet
• Deep RL (DQN, etc.)
• AlphaGo
• GANs
• BERT
• ChatGPT
• Diffusion Models
• Much more!


Pre-history
Supervised and Reinforcement Learning
Foundation Models (Imitation Learning)
2012
2013
2015
2018
2022
Today
• Decision Trees
• Early NLP
• Dynamic Programming
• SVMs
• AlexNet
• Deep RL (DQN, etc.)
• AlphaGo
• GANs
• BERT
• ChatGPT
• Diffusion Models
• Much more!

1. Why have we converged on such a **unified paradigm** to address different **problem settings**?

2. When does **Imitation Learning** succeed and how is it different from **supervised learning**?

3. How can we leverage **theoretical principles** to (empirically) **improve** Imitation Learning?

Outline

I. Background and Introduction.

II. Classical Imitation Learning.

III. Modern IL in discrete settings.

IV. Modern IL in continuous settings.

V. Opening the black box of optimization.

Outline

I. Background and Introduction.

II. Classical Imitation Learning.

III. Modern IL in discrete settings.

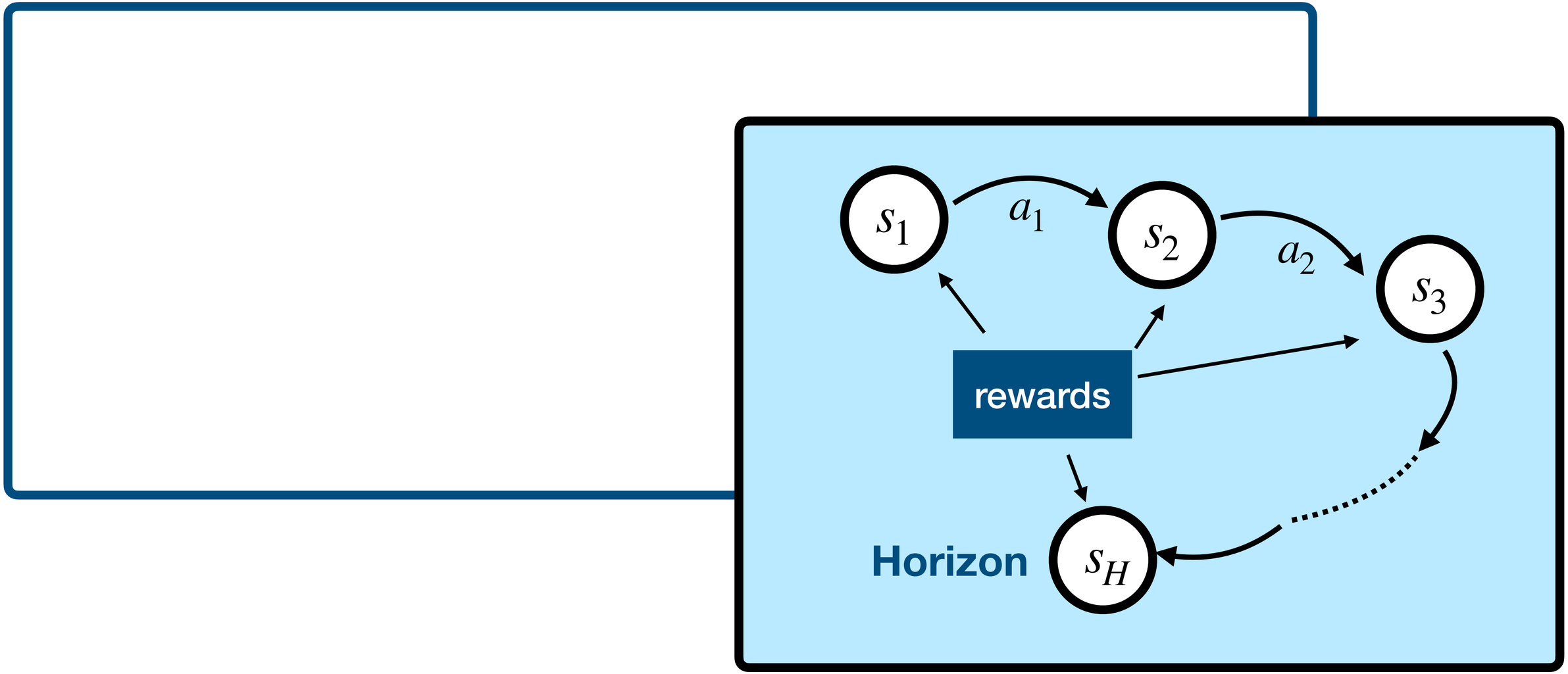
IV. Modern IL in continuous settings.

V. Opening the black box of optimization.

The Classical Formalism of Imitation Learning

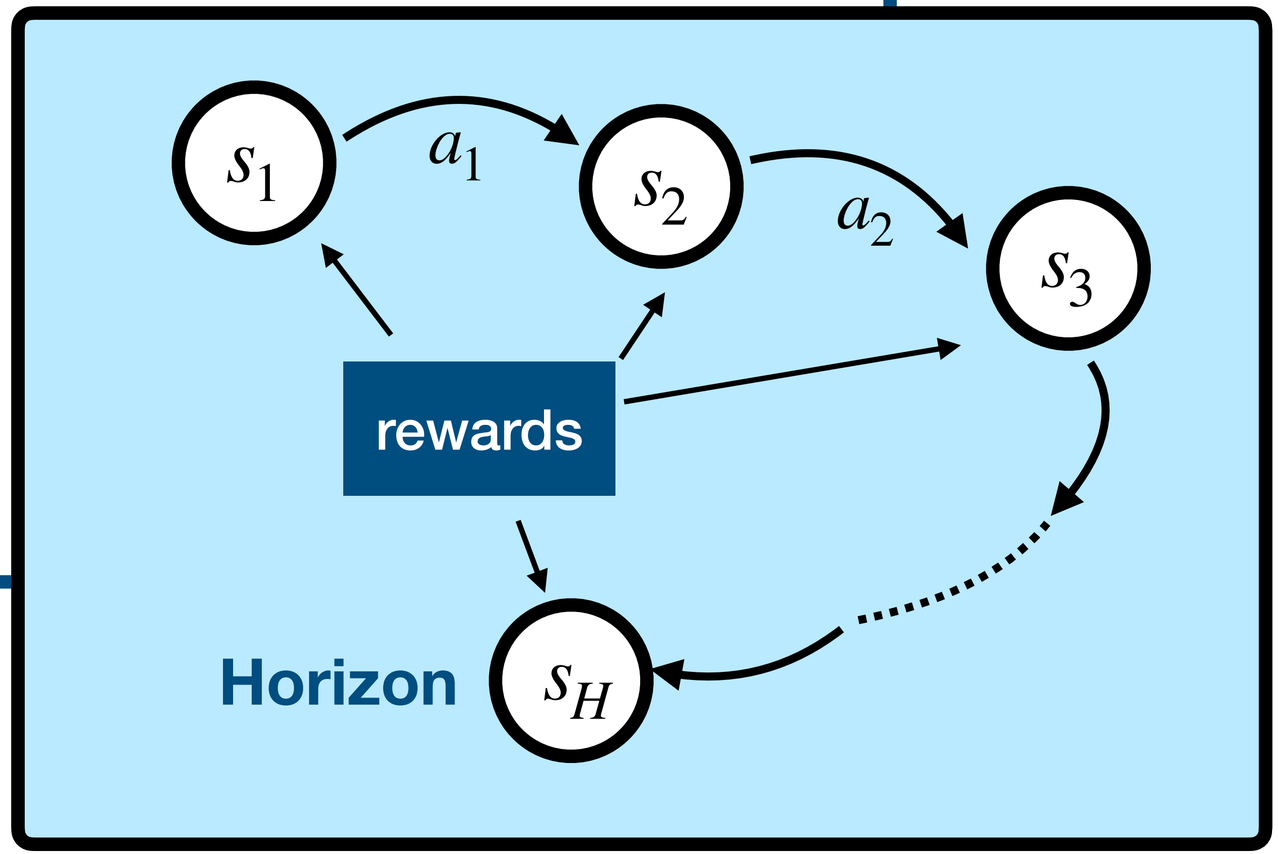
1. Formalizing Imitation Learning

Decision Making: A Formal Model



Decision Making: A Formal Model

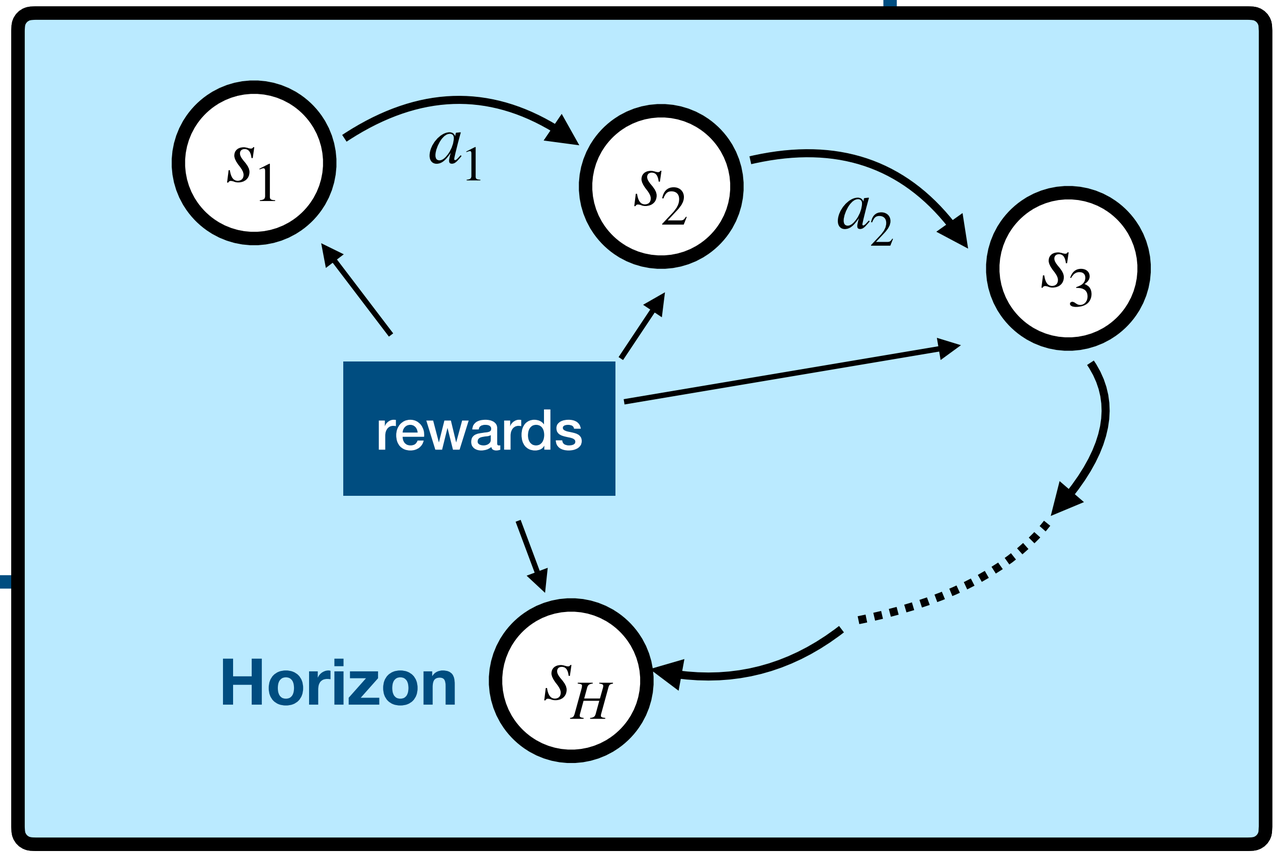
Definition: A Markov Decision Process (MDP) is specified by



Decision Making: A Formal Model

Definition: A Markov Decision Process (MDP) is specified by

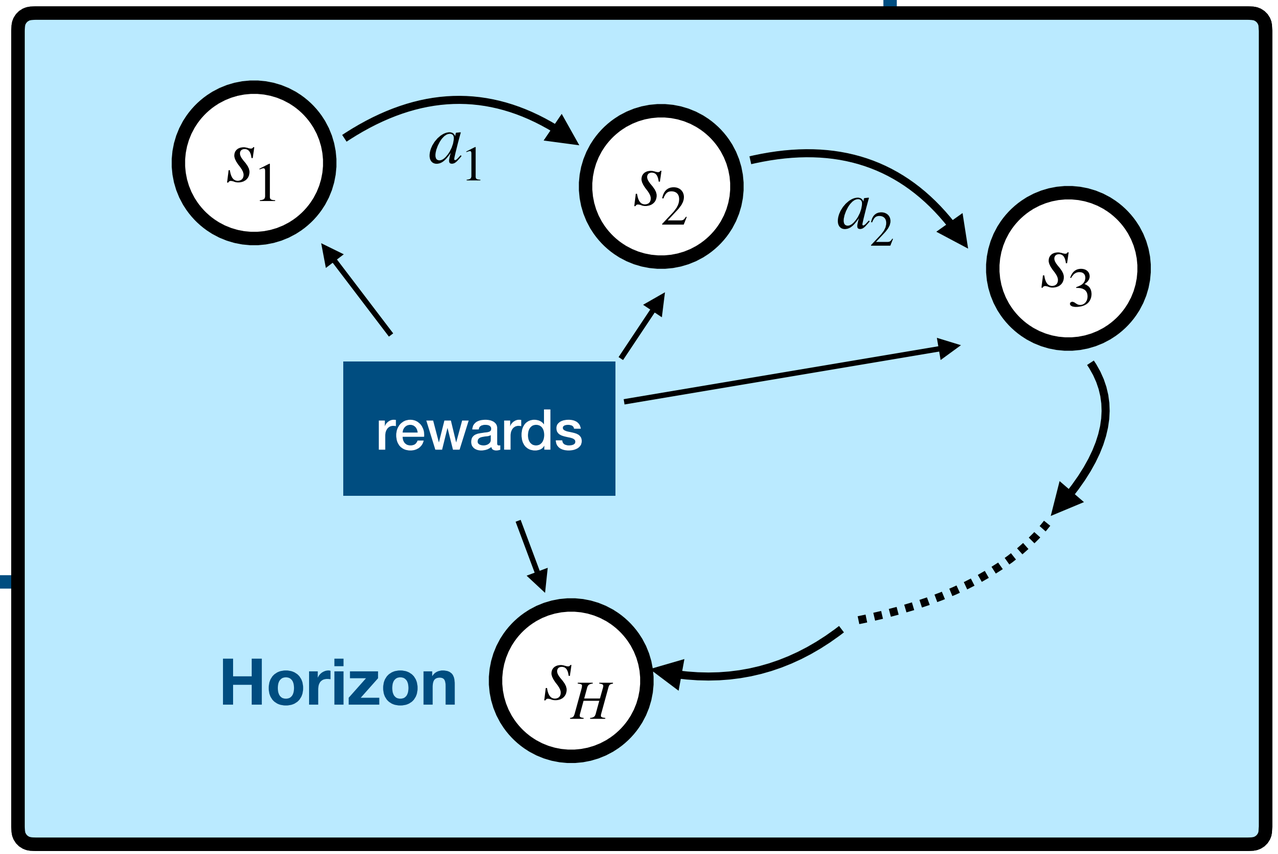
- **states** $s \in S$



Decision Making: A Formal Model

Definition: A Markov Decision Process (MDP) is specified by

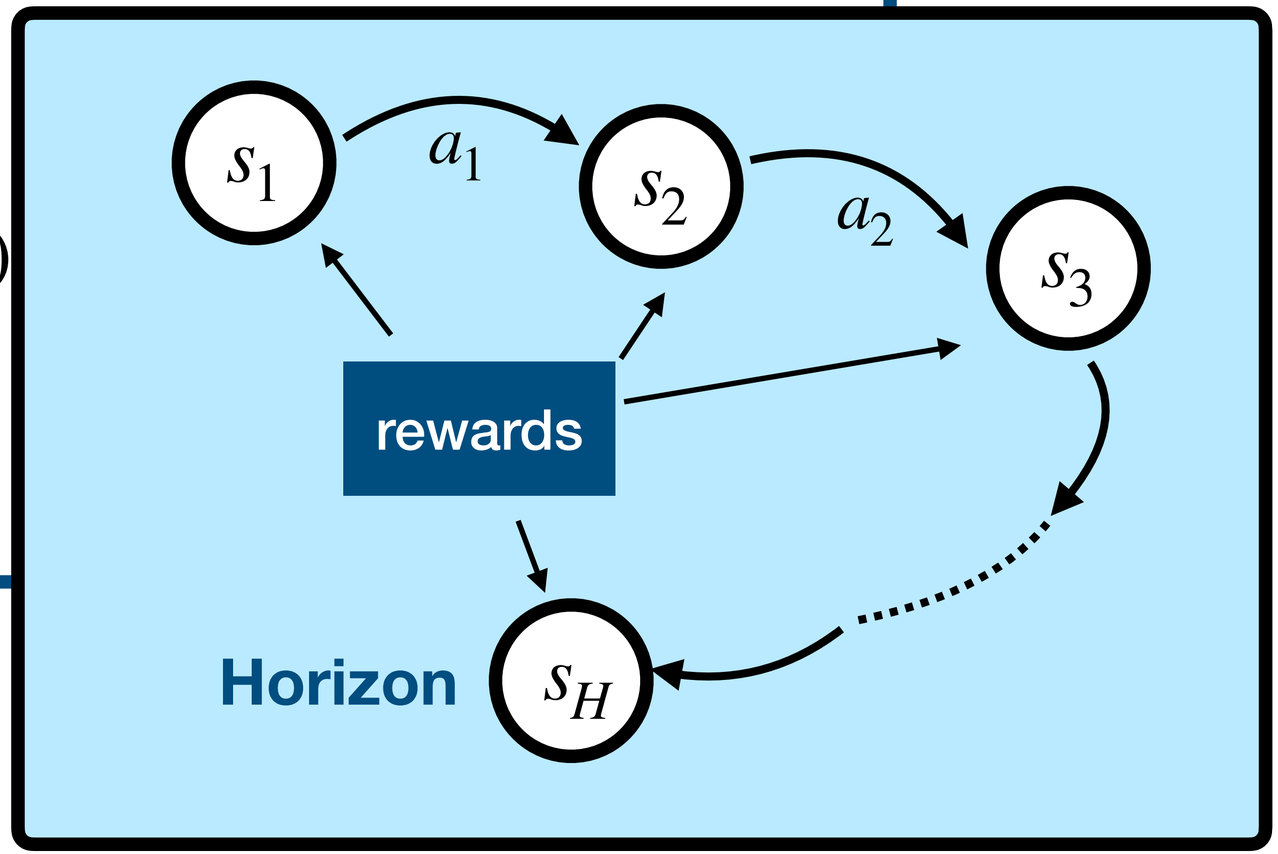
- **states** $s \in S$
- **actions** $a \in A$



Decision Making: A Formal Model

Definition: A Markov Decision Process (MDP) is specified by

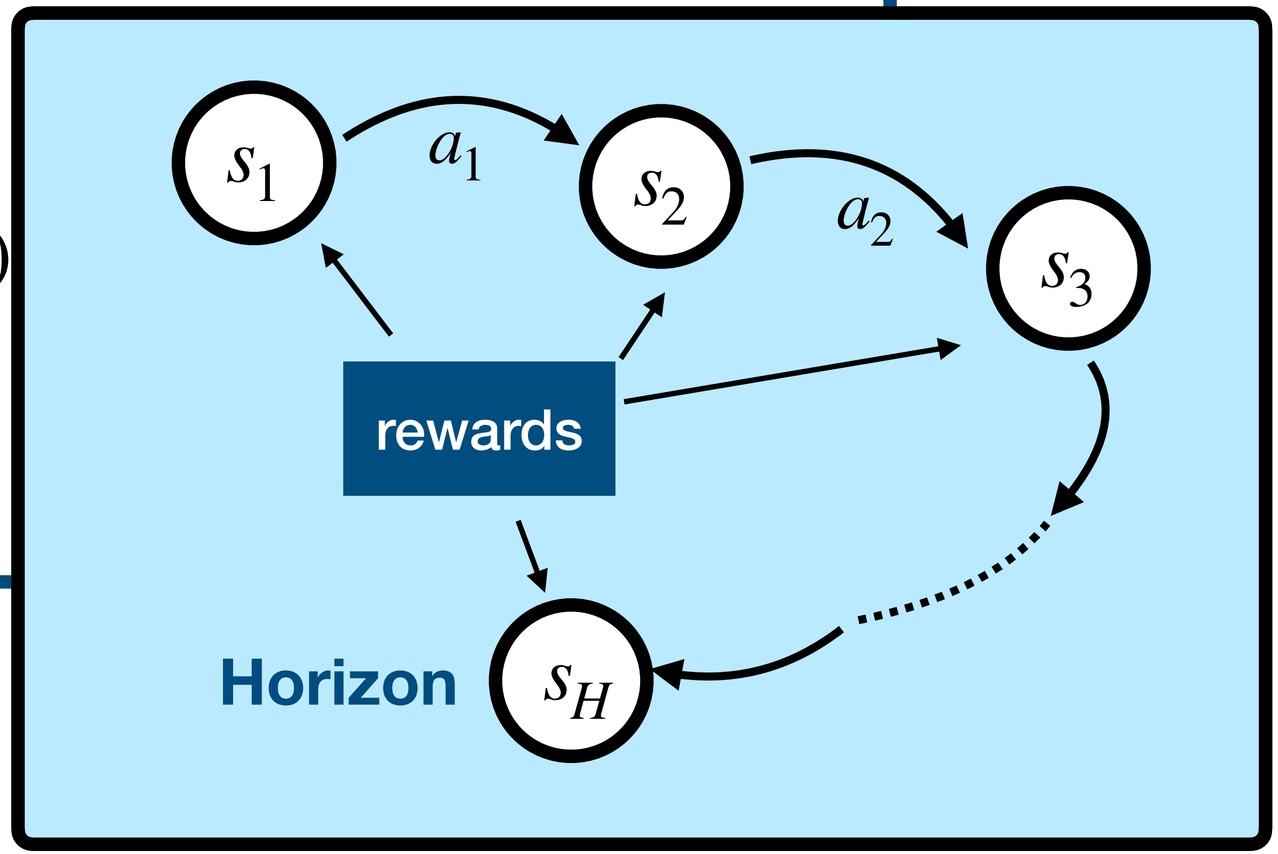
- **states** $s \in S$
- **actions** $a \in A$
- **transition probabilities** $s' \sim P(s, a)$



Decision Making: A Formal Model

Definition: A Markov Decision Process (MDP) is specified by

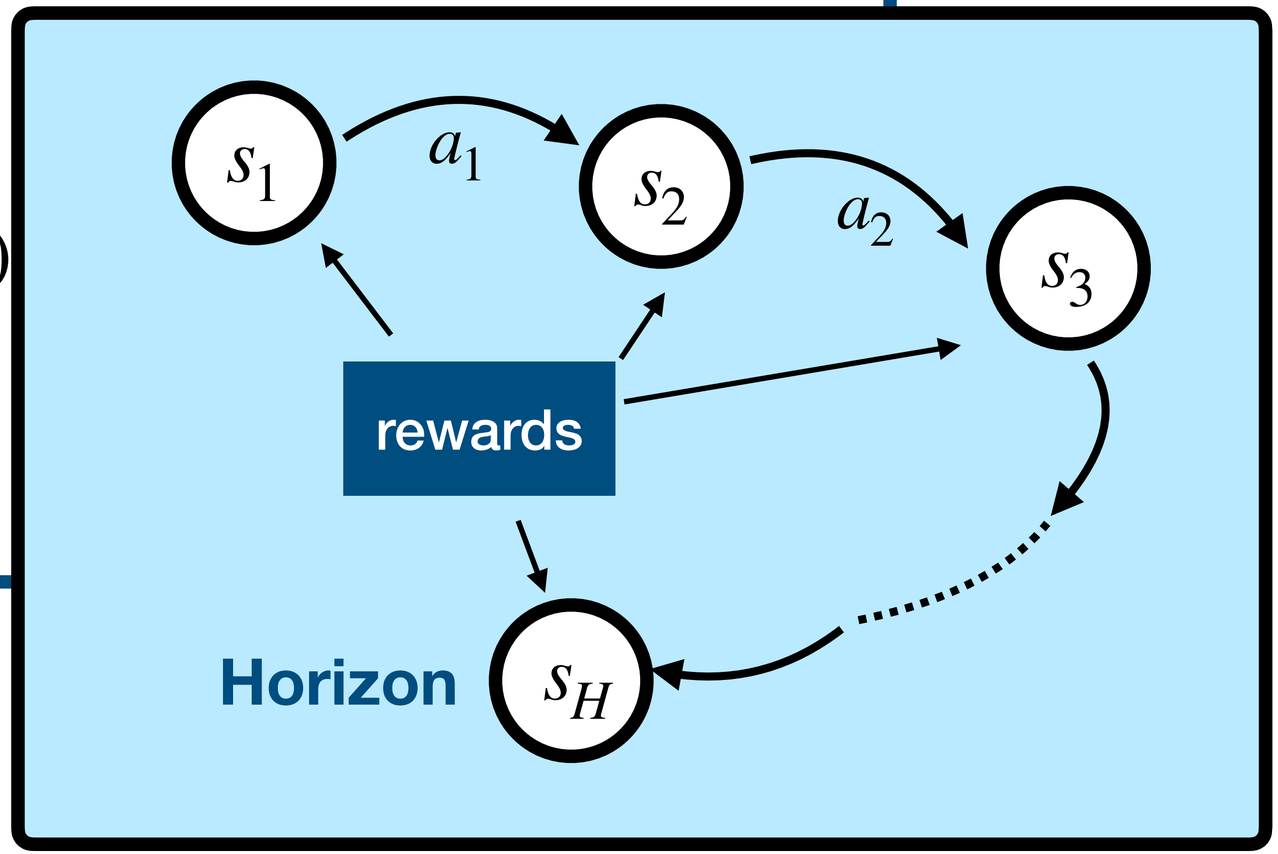
- **states** $s \in S$
- **actions** $a \in A$
- **transition probabilities** $s' \sim P(s, a)$
- **rewards** $r(s, a)$



Decision Making: A Formal Model

Definition: A Markov Decision Process (MDP) is specified by

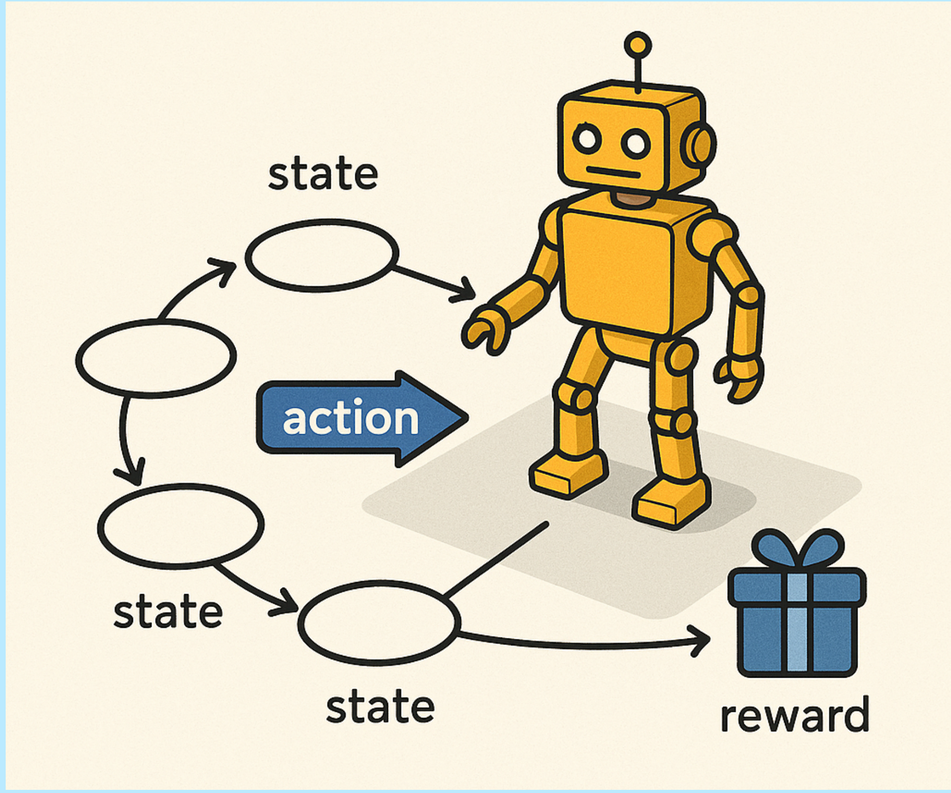
- **states** $s \in S$
- **actions** $a \in A$
- **transition probabilities** $s' \sim P(s, a)$
- **rewards** $r(s, a)$
- **initial state distribution** $s_1 \sim P_1$



Decision Making: A Formal Model

Definition: A Markov Decision Process (MDP) is specified by

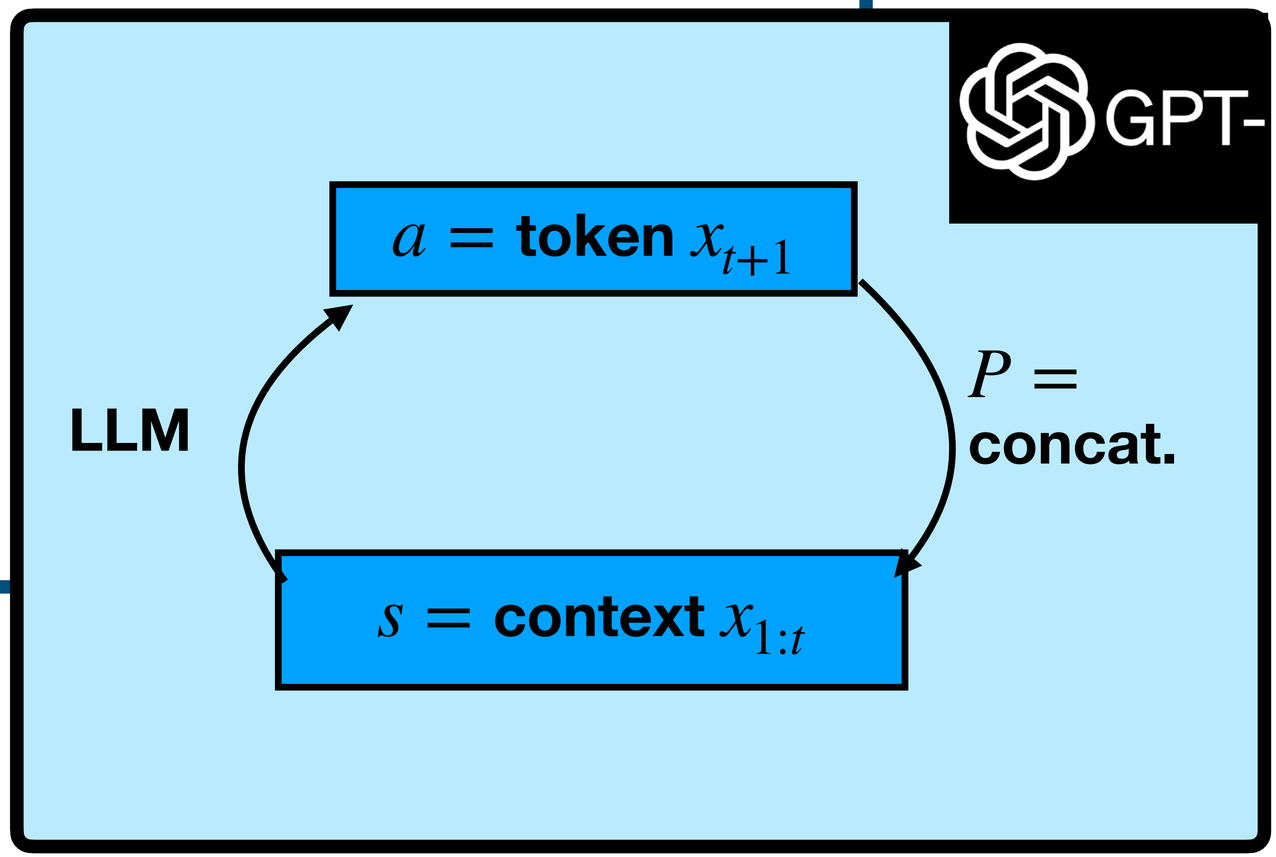
- **states** $s \in S$
- **actions** $a \in A$
- **transition probabilities** $s' \sim P(s, a)$
- **rewards** $r(s, a)$
- **initial state distribution** $s_1 \sim P_1$



Decision Making: A Formal Model

Definition: A Markov Decision Process (MDP) is specified by

- **states** $s \in S$
- **actions** $a \in A$
- **transition probabilities** $s' \sim P(s, a)$
- **rewards** $r(s, a)$
- **initial state distribution** $s_1 \sim P_1$



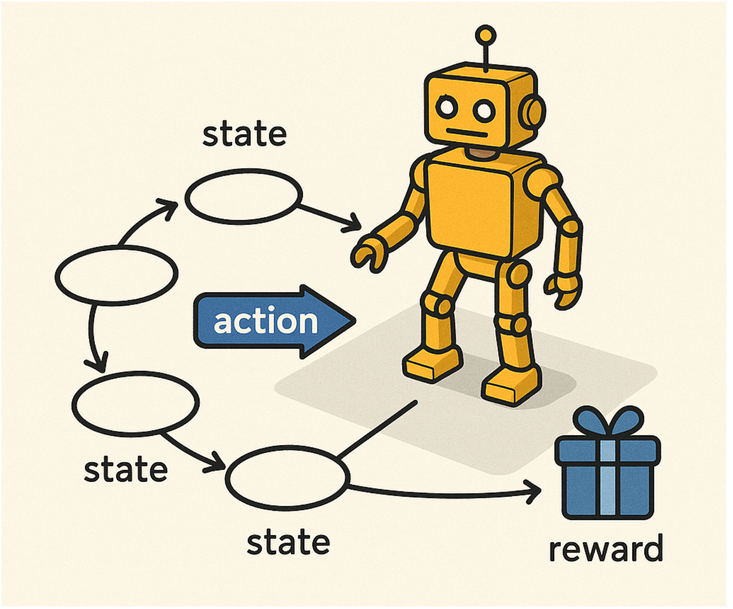
Policies

Definition: A **Policy** is a mapping $\pi : S \to \Delta(A)$

**Markovian policy*

Policies

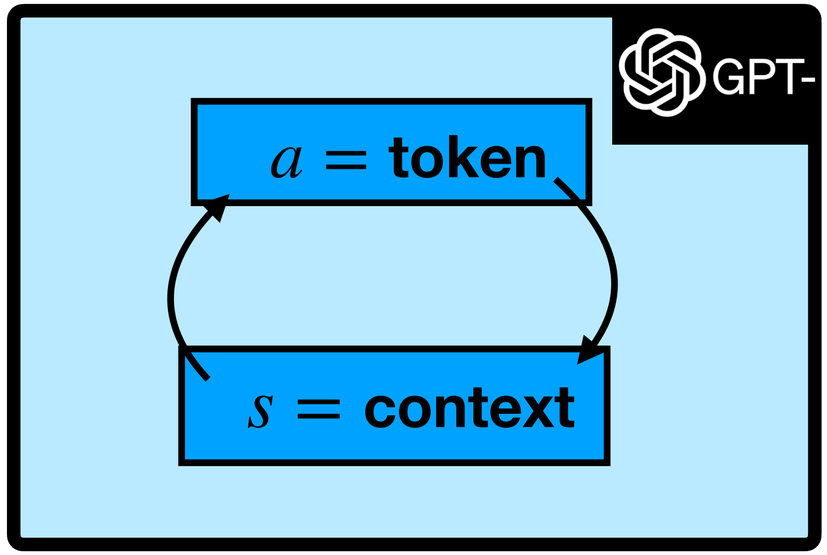
Definition: A **Policy** is a mapping $\pi : S \to \Delta(A)$

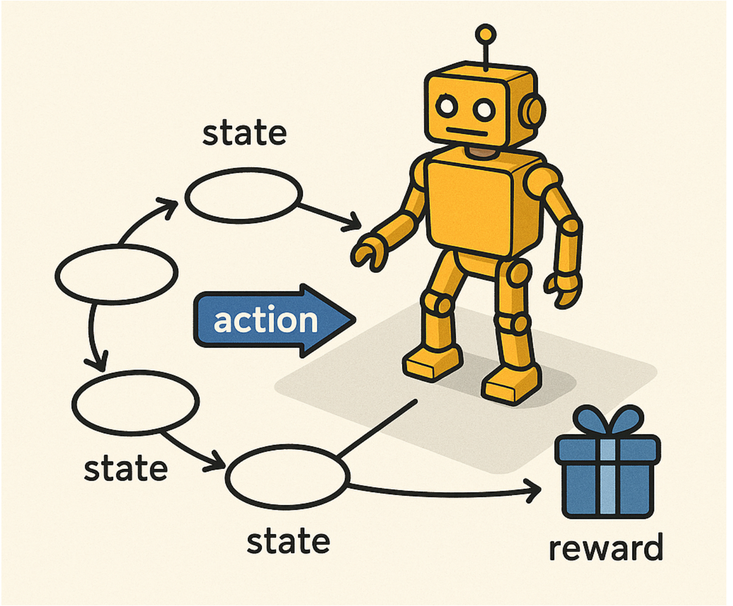


*Markovian policy

Policies

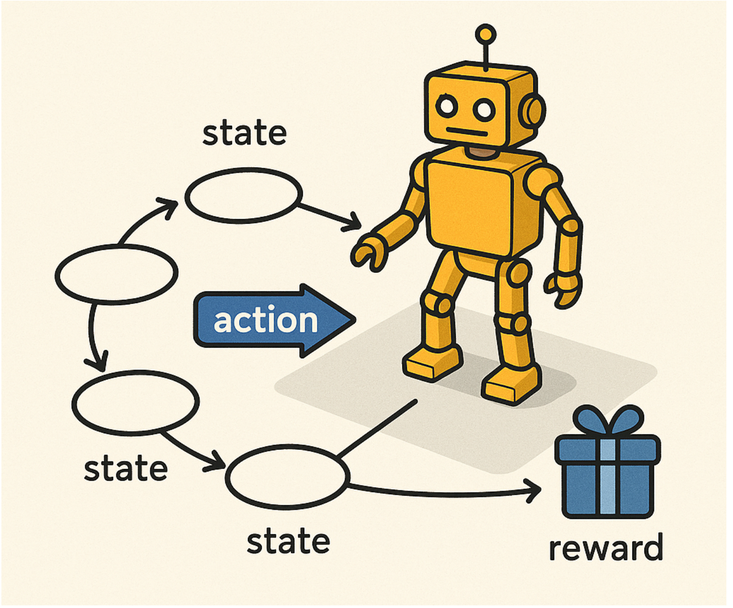
Definition: A **Policy** is a mapping $\pi : S \to \Delta(A)$





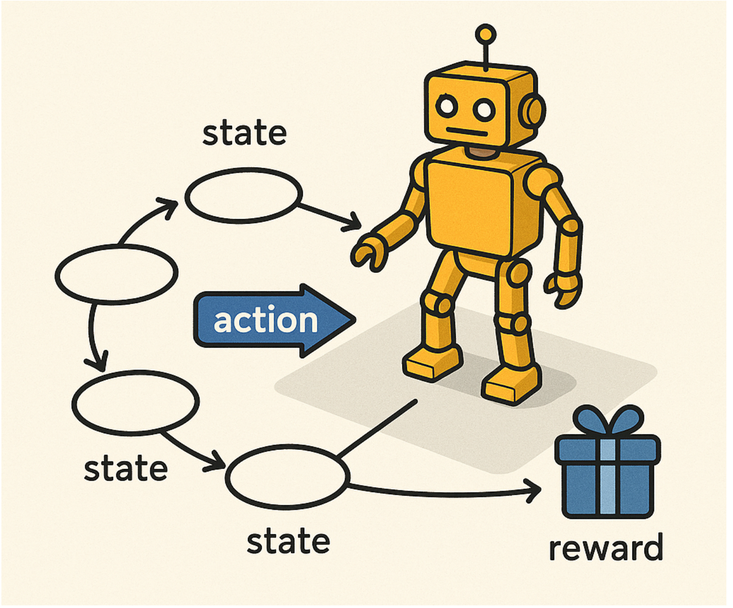
*Markovian policy

Expected Reward



Expected Reward

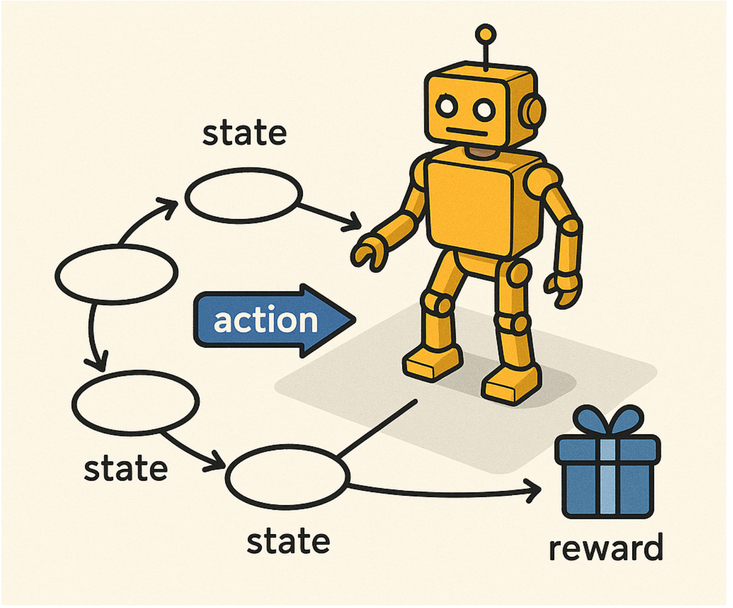
Definition: The **J-functional** is $J(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=1}^{H} r(s_t, a_t)\right]$



Expected Reward

Definition: The **J-functional** is $J(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=1}^{H} r(s_t, a_t)\right]$

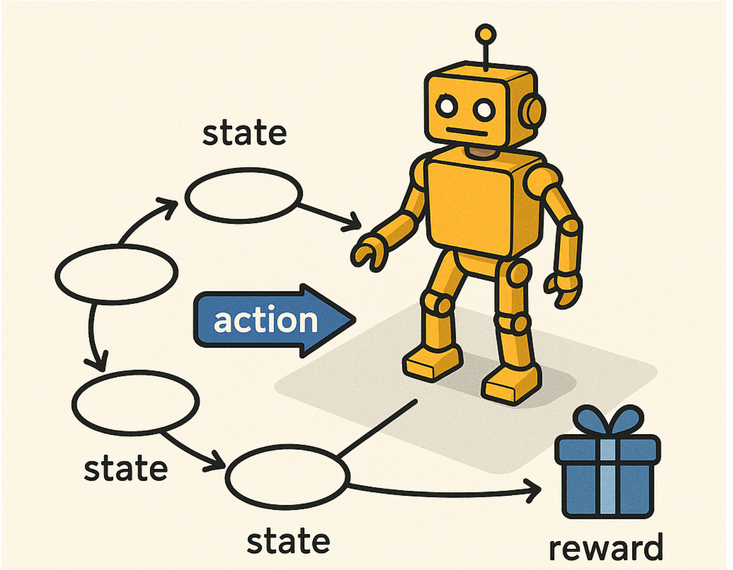
Expected cumulated reward of policy π



Expected Reward

Definition: The **J-functional** is $J(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=1}^{H} r(s_t, a_t)\right]$

Expected cumulated reward of policy π



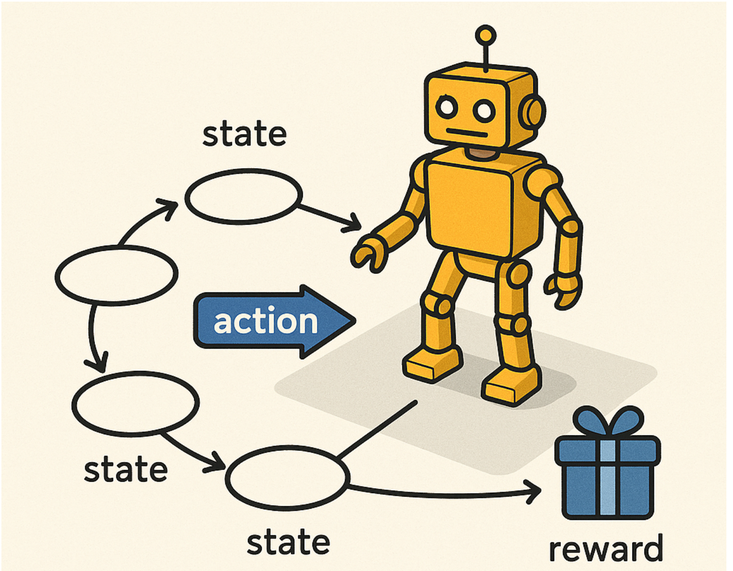
$\mathbb{E}^{\pi} : s_{t+1} \sim P(s_t, a_t),\ a_t \sim \pi(s_t), s_1 \sim P_1$

Expected Reward

Definition: The **J-functional** is $J(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=1}^{H} r(s_t, a_t)\right]$

Expected cumulated reward of policy π

Goal: Using data, maximize $J(\pi)$



$\mathbb{E}^{\pi} : s_{t+1} \sim P(s_t, a_t),\ a_t \sim \pi(s_t), s_1 \sim P_1$

Protocol: Offline Imitation Learning

Protocol: Offline Imitation Learning

1. An **expert** $\pi^\star$ collects $n \in \mathbb{N}$ i.i.d trajectories $(s_1, a_1, \ldots, s_H, a_H) \sim \mathbb{P}^{\pi^\star}$

Protocol: Offline Imitation Learning

1. An **expert** $\pi^\star$ collects $n \in \mathbb{N}$ i.i.d trajectories $(s_1, a_1, \ldots, s_H, a_H) \sim \mathbb{P}^{\pi^\star}$

2. Given trajectories, **imitator** attempts to return $\hat{\pi}$ such that

Protocol: Offline Imitation Learning

1. An **expert** $\pi^\star$ collects $n \in \mathbb{N}$ i.i.d trajectories $(s_1, a_1, \ldots, s_H, a_H) \sim \mathbb{P}^{\pi^\star}$

2. Given trajectories, **imitator** attempts to return $\hat{\pi}$ such that

$$J(\pi^\star) - J(\hat{\pi}) \leq \epsilon$$

Protocol: Offline Imitation Learning

1. An **expert** $\pi^\star$ collects $n \in \mathbb{N}$ i.i.d trajectories $(s_1, a_1, \ldots, s_H, a_H) \sim \mathbb{P}^{\pi^\star}$

2. Given trajectories, **imitator** attempts to return $\hat{\pi}$ such that

$$J(\pi^\star) - J(\hat{\pi}) \leq \epsilon$$

Crucially: learner does not see rewards

Protocol: Offline Imitation Learning

1. An **expert** $\pi^\star$ collects $n \in \mathbb{N}$ i.i.d trajectories $(s_1, a_1, \ldots, s_H, a_H) \sim \mathbb{P}^{\pi^\star}$

2. Given trajectories, **imitator** attempts to return $\hat{\pi}$ such that

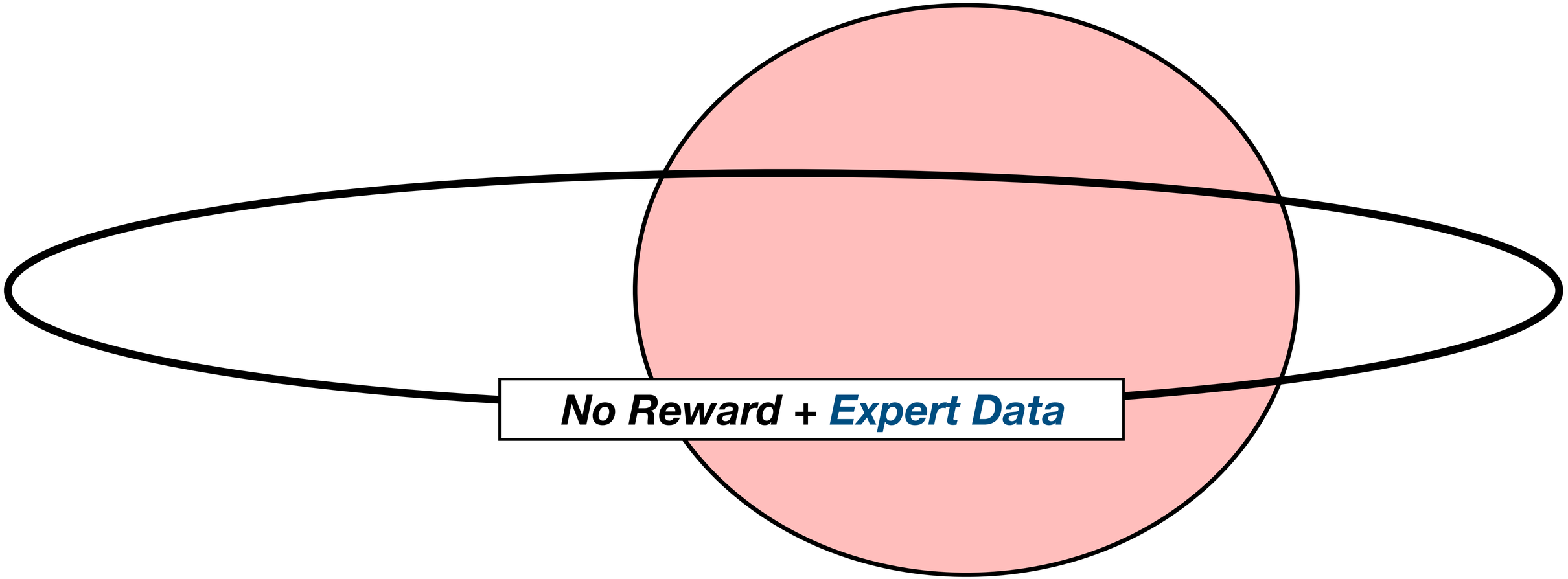
$$J(\pi^\star) - J(\hat{\pi}) \leq \epsilon$$

This tutorial focuses on this setting.

Summary: Offline Imitation Learning

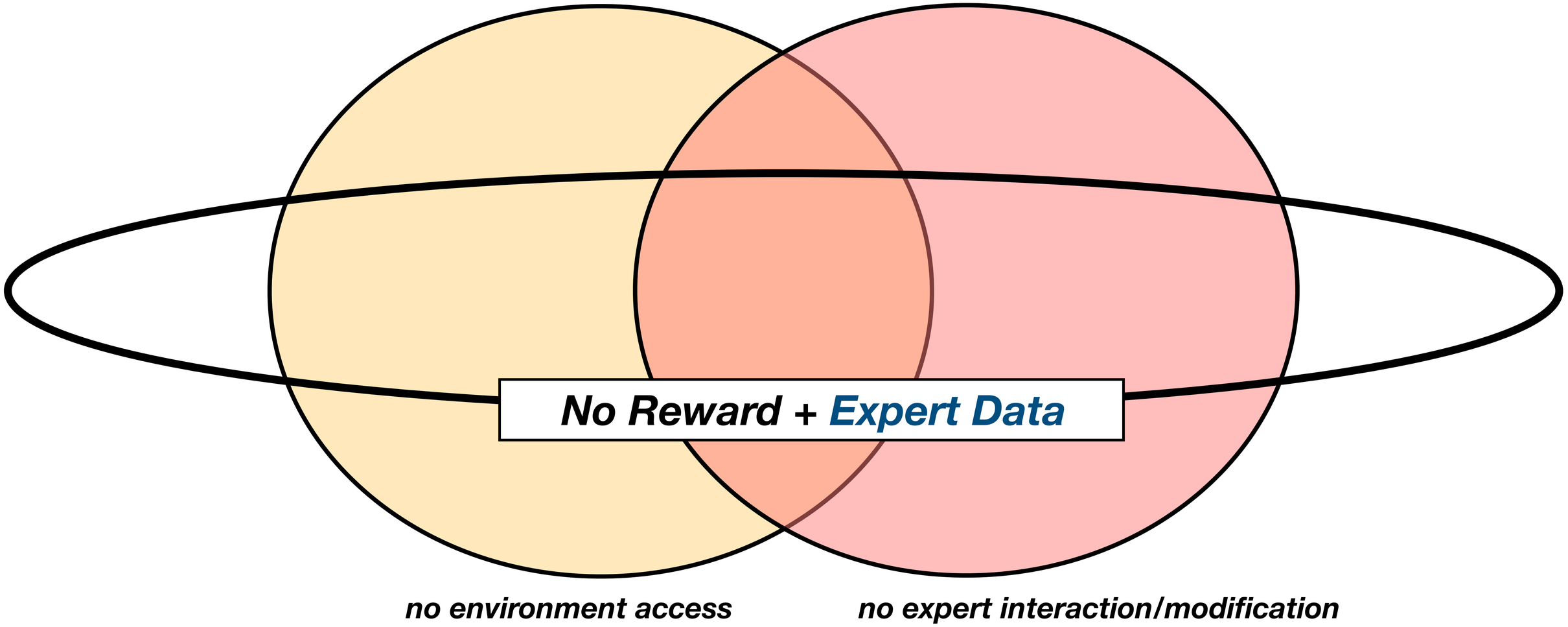
Summary: Offline Imitation Learning

Summary: Offline Imitation Learning

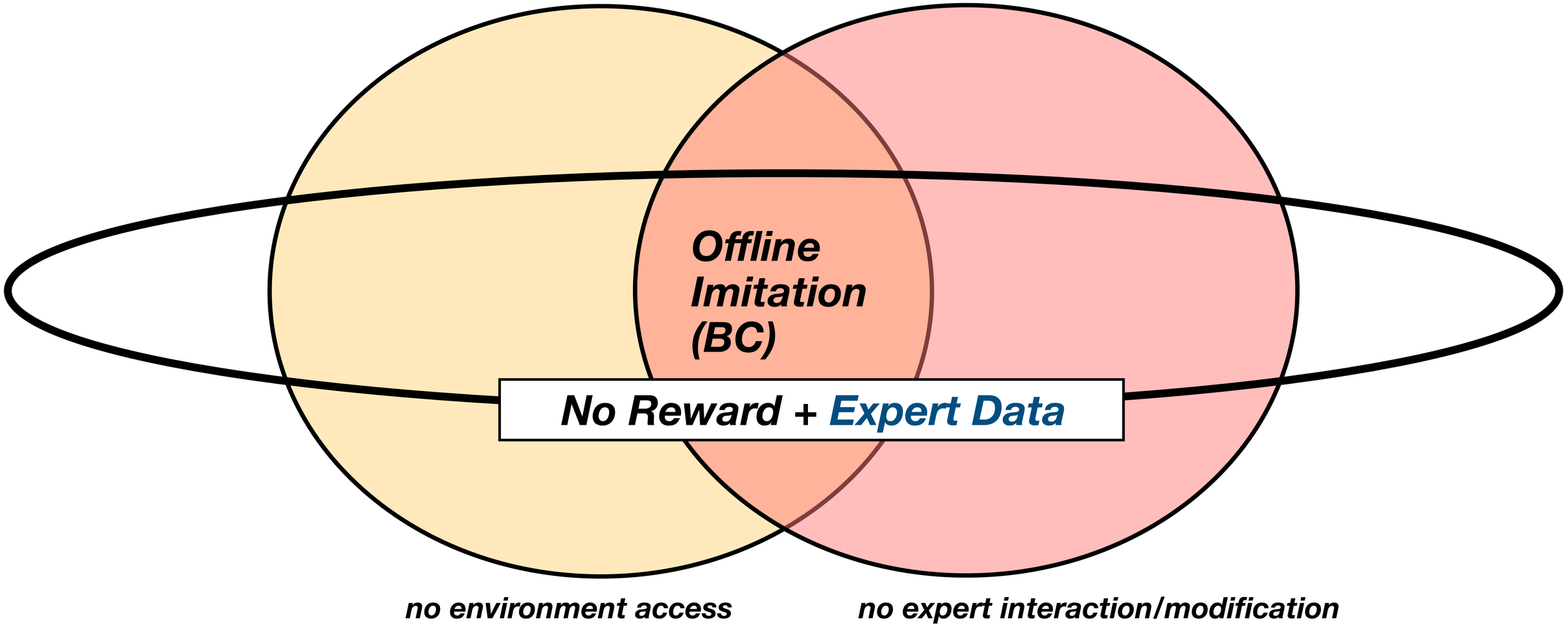


no expert interaction/modification

Summary: Offline Imitation Learning



Summary: Offline Imitation Learning



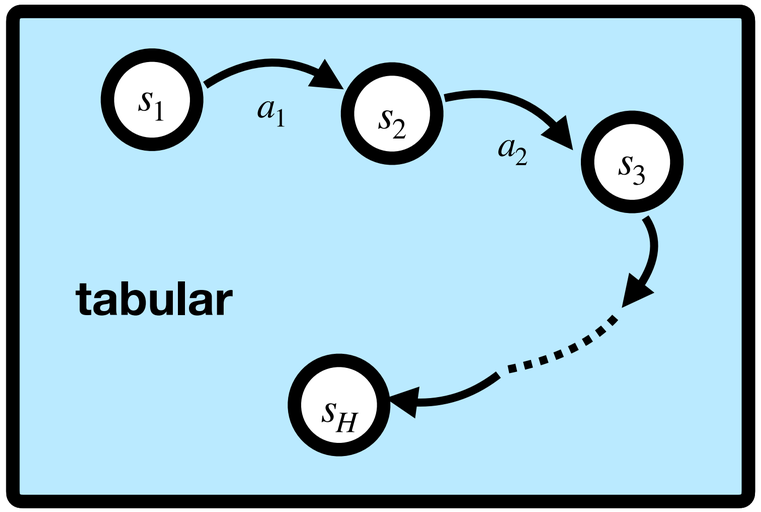
How Neural Networks Represent Policies

How Neural Networks Represent Policies

Definition: A **Policy*** is a mapping $\pi : S \to \Delta(A)$

How Neural Networks Represent Policies

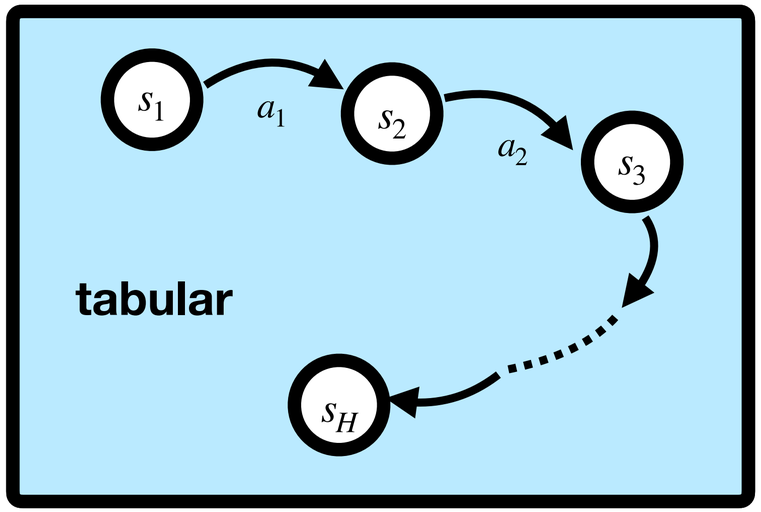
Definition: A **Policy*** is a mapping $\pi : S \to \Delta(A)$



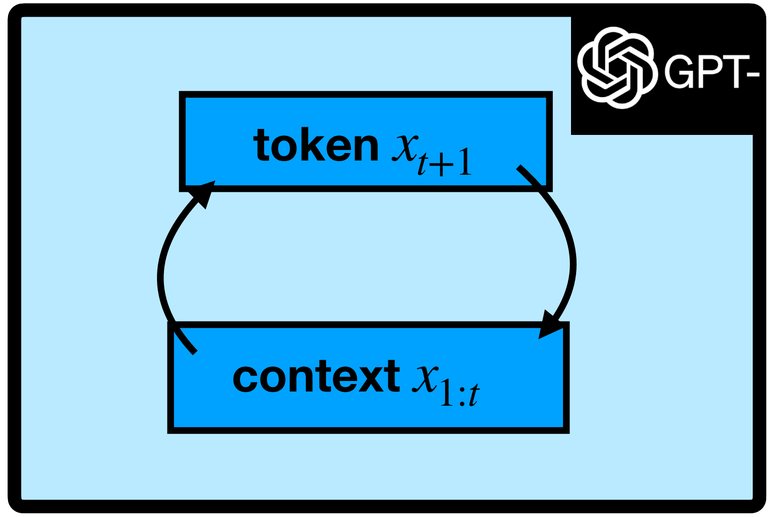
$$\pi_\theta(a_t = a \mid s_t)$$

How Neural Networks Represent Policies

Definition: A **Policy*** is a mapping $\pi : S \to \Delta(A)$



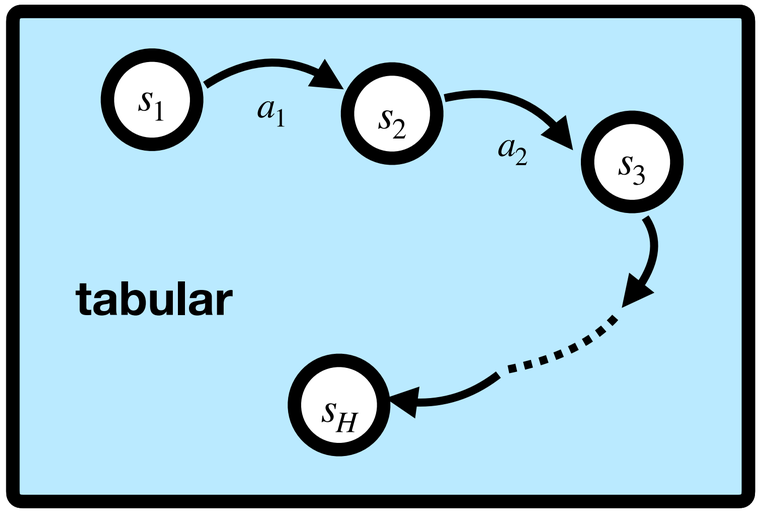
$\pi_\theta(a_t = a \mid s_t)$



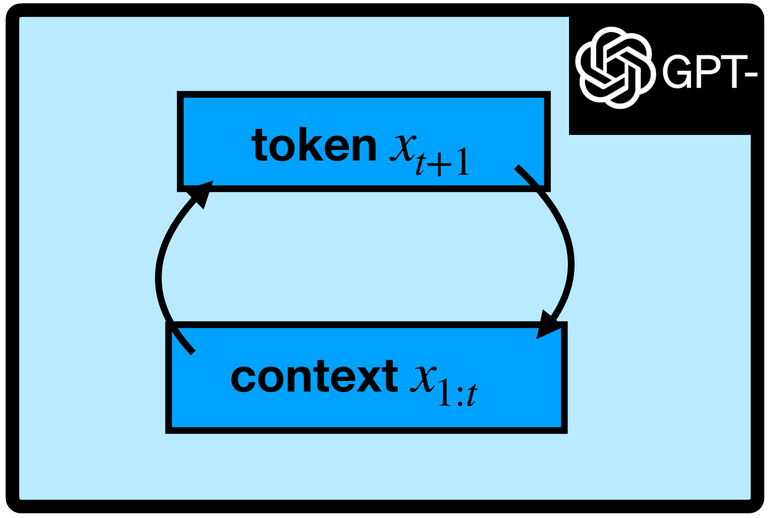
$\log \pi_\theta(x_{t+1} = x \mid x_{1:t})$

How Neural Networks Represent Policies

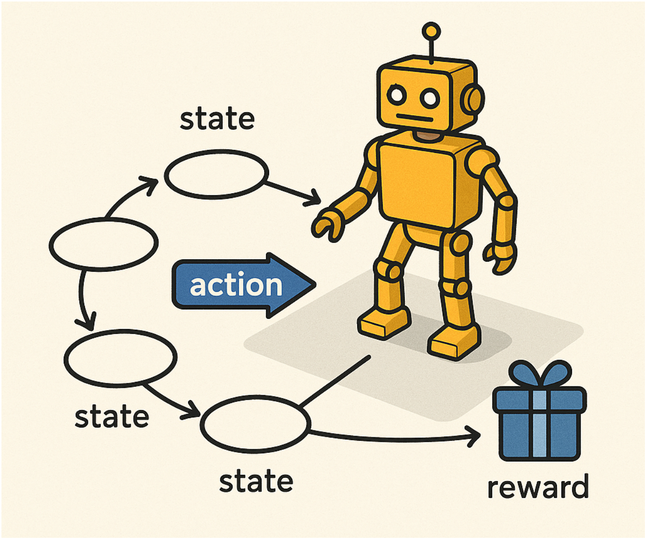
Definition: A **Policy*** is a mapping $\pi : S \to \Delta(A)$



$\pi_\theta(a_t = a \mid s_t)$



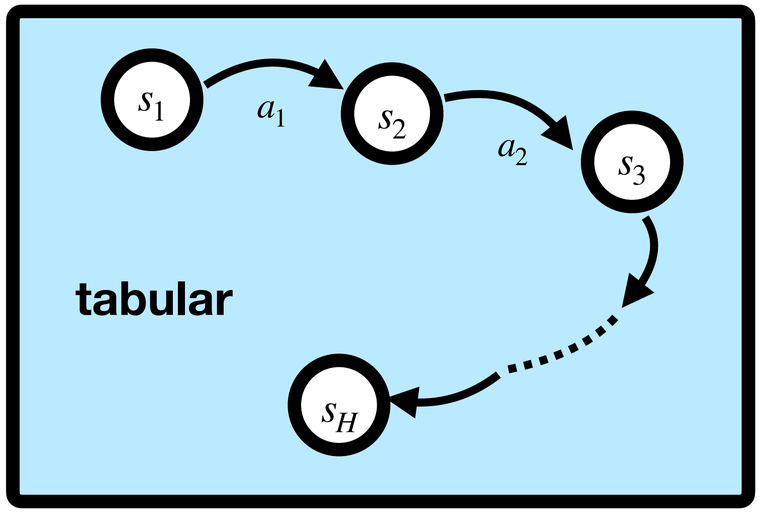
$\log \pi_\theta(x_{t+1} = x \mid x_{1:t})$



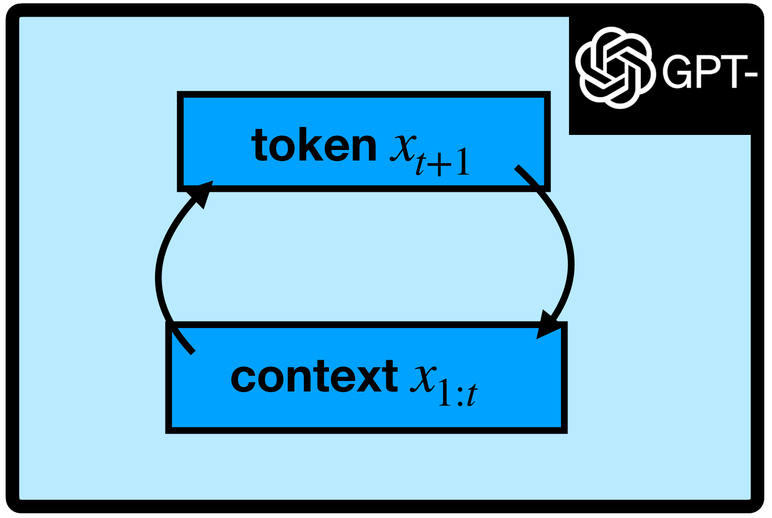
$\pi_\theta(a_t \mid s_t) = \mathrm{N}(\mu_\theta(s_t), \sigma^2 I)$

How Neural Networks Represent Policies

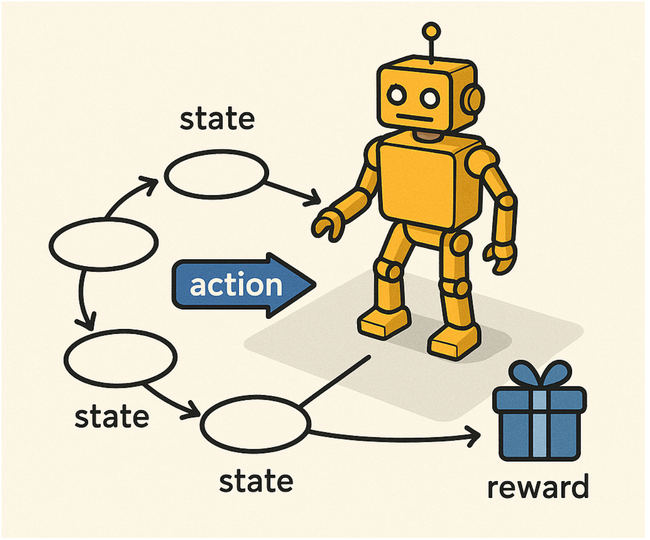
Definition: A **Policy*** is a mapping $\pi : S \to \Delta(A)$



$\pi_\theta(a_t = a \mid s_t)$



$\log \pi_\theta(x_{t+1} = x \mid x_{1:t})$

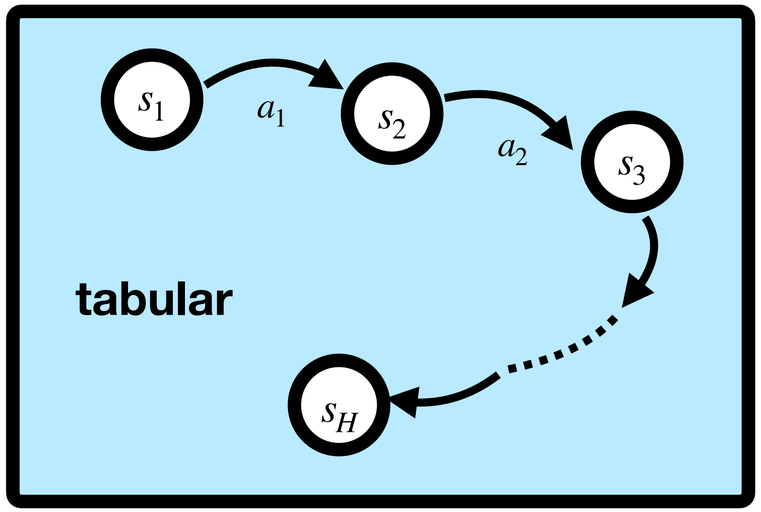


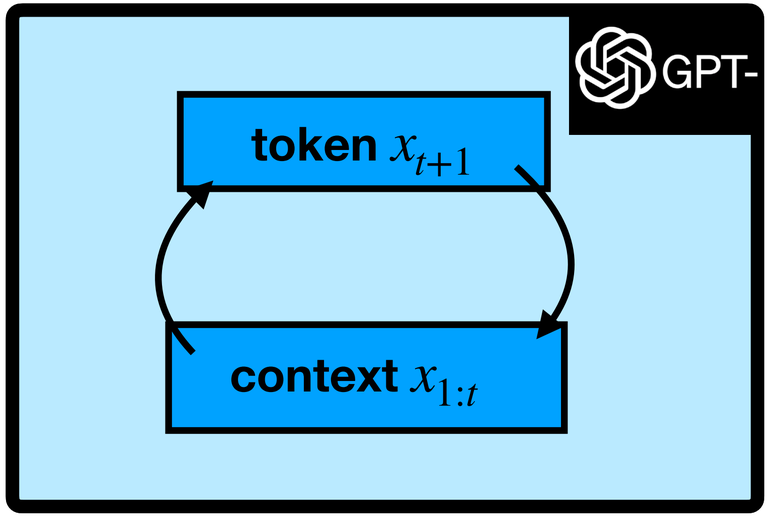
$\pi_\theta(a_t \mid s_t) = \mathrm{N}(\mu_\theta(s_t), \sigma^2 I)$

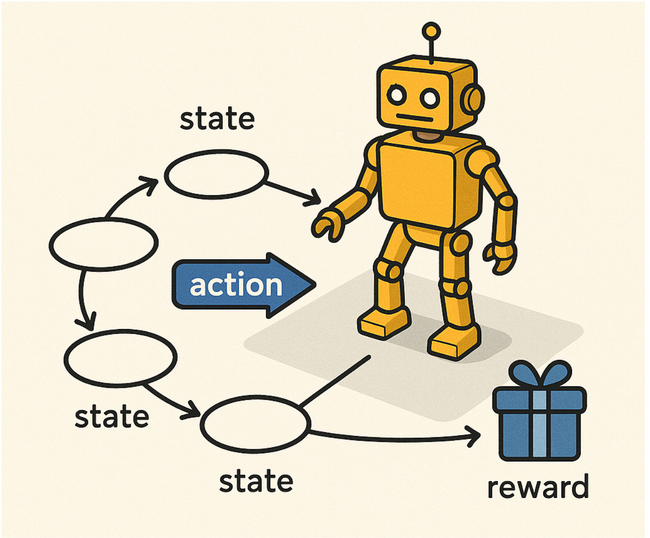
or **diffusion/score model**

How do I represent a policy?

Definition: A **Policy*** is a mapping $\pi : S \to \Delta(A)$

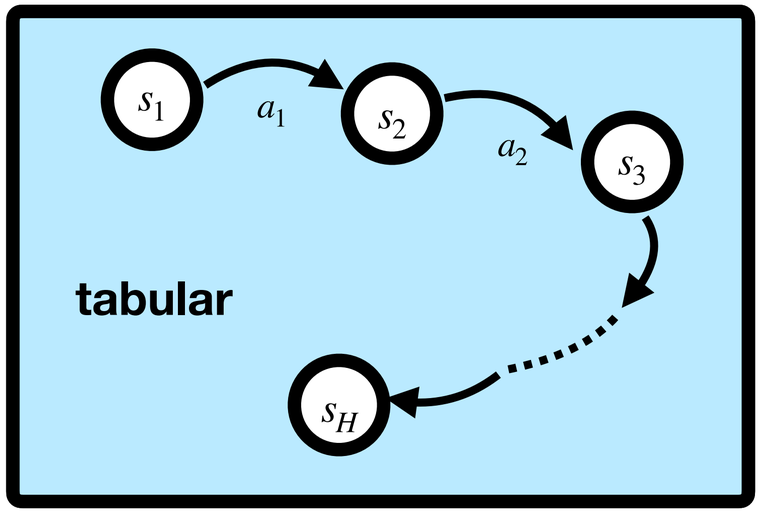


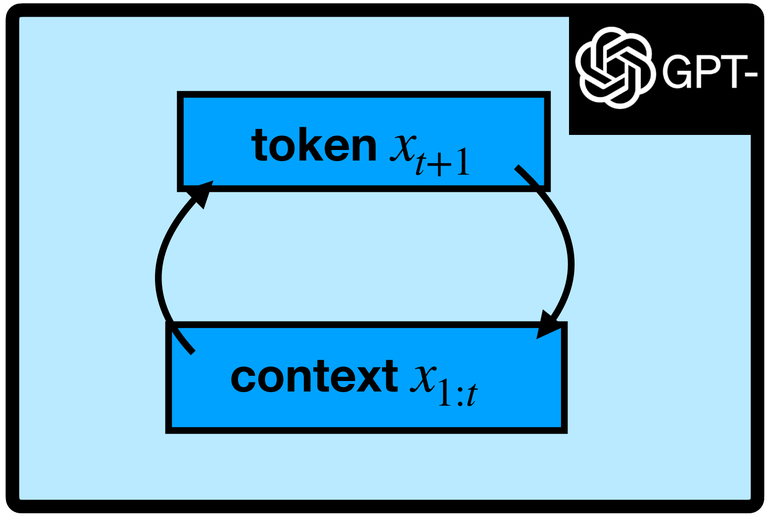


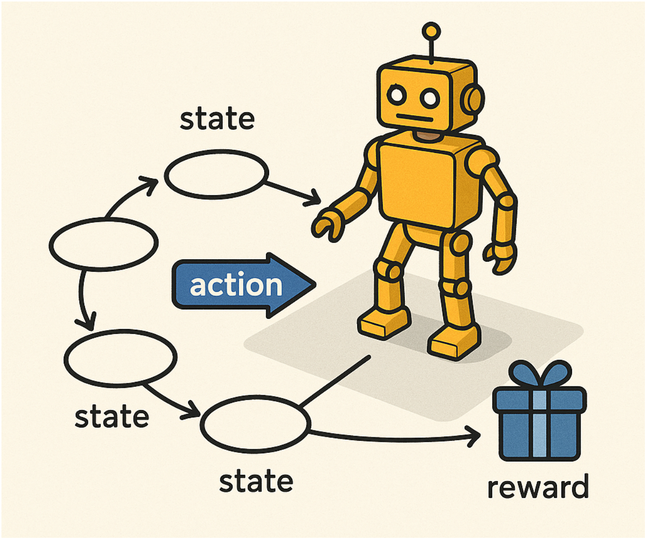


How do I represent a policy?

Definition: A **Policy*** is a mapping $\pi : S \to \Delta(A)$







Key Theme: policy parametrization will influence learning

Protocol: Offline Imitation Learning

1. An **expert** $\pi^\star$ collects $n \in \mathbb{N}$ i.i.d trajectories $(s_1, a_1, \ldots, s_H, a_H) \sim \mathbb{P}^{\pi^\star}$

2. Given trajectories, **imitator** attempts to return $\hat{\pi}$ such that

$$J(\pi^\star) - J(\hat{\pi}) \leq \epsilon$$

Crucially: learner does not see rewards

The Classical Formalism of Imitation Learning

2. Formalizing Imitation Learning and Behavior Cloning

Behavior Cloning

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

expert data

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

expert data

loss function

(related to policy parametrization)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

expert data

policy class

(e.g. neural network)

loss function

(related to policy parametrization)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Example 2: $\text{loss}_2(\pi, s, a) = \|a - \pi(s)\|^2$ ($\pi^\star$ is takes continuous values)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Example 2: $\text{loss}_2(\pi, s, a) = \|a - \pi(s)\|^2$ ($\pi^\star$ is takes continuous values)

Example 3: $\text{loss}_{\log}(\pi, s, a) = \log \pi(a \mid s)$

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Example 2: $\text{loss}_2(\pi, s, a) = \|a - \pi(s)\|^2$ ($\pi^\star$ is takes continuous values)

Example 3: $\text{loss}_{\log}(\pi, s, a) = \log \pi(a \mid s)$ ($\pi^\star$ is discrete, or $\pi^\star(s)$ has density)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Example 2: $\text{loss}_2(\pi, s, a) = \|a - \pi(s)\|^2$ ($\pi^\star$ is takes continuous values)

Example 3: $\text{loss}_{\log}(\pi, s, a) = \log \pi(a \mid s)$ ($\pi^\star$ is discrete, or $\pi^\star(s)$ has density)

Example 4: $\text{loss}(\pi, s, a) =$ (Score Matching)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Example 2: $\text{loss}_2(\pi, s, a) = \|a - \pi(s)\|^2$ ($\pi^\star$ is takes continuous values)

Example 3: $\text{loss}_{\log}(\pi, s, a) = \log \pi(a \mid s)$ ($\pi^\star$ is discrete, or $\pi^\star(s)$ has density)

Example 4: $\text{loss}(\pi, s, a) = (\text{Score Matching})$ (popular in robotics)

Behavior Cloning Generalization

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Behavior Cloning Generalization

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Define $L_{\text{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \text{loss}(s_t, a_t, \pi)\right]$.

Behavior Cloning Generalization

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Define $L_{\text{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \text{loss}(s_t, a_t, \pi)\right]$.

Then $L_{\text{bc}}(\hat{\pi}) - L_{\text{bc}}(\pi^\star) \leq$ **generalization error**

Behavior Cloning Generalization

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Define $L_{\text{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \text{loss}(s_t, a_t, \pi)\right]$.

Then $L_{\text{bc}}(\hat{\pi}) - L_{\text{bc}}(\pi^\star) \leq$ **generalization error** $\leq \frac{\log|\Pi|}{n}$ (deterministic policies)

Behavior Cloning Generalization

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Behavior Cloning Generalization

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Central Challenge: Mismatch between BC error and cumulative reward:

Behavior Cloning Generalization

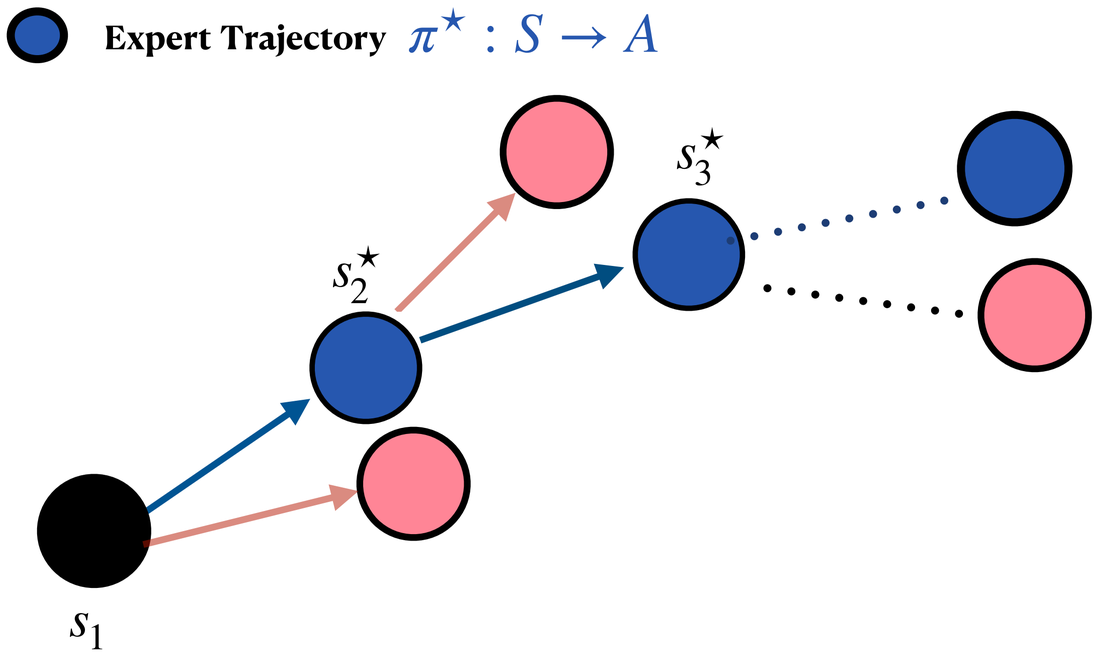
Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

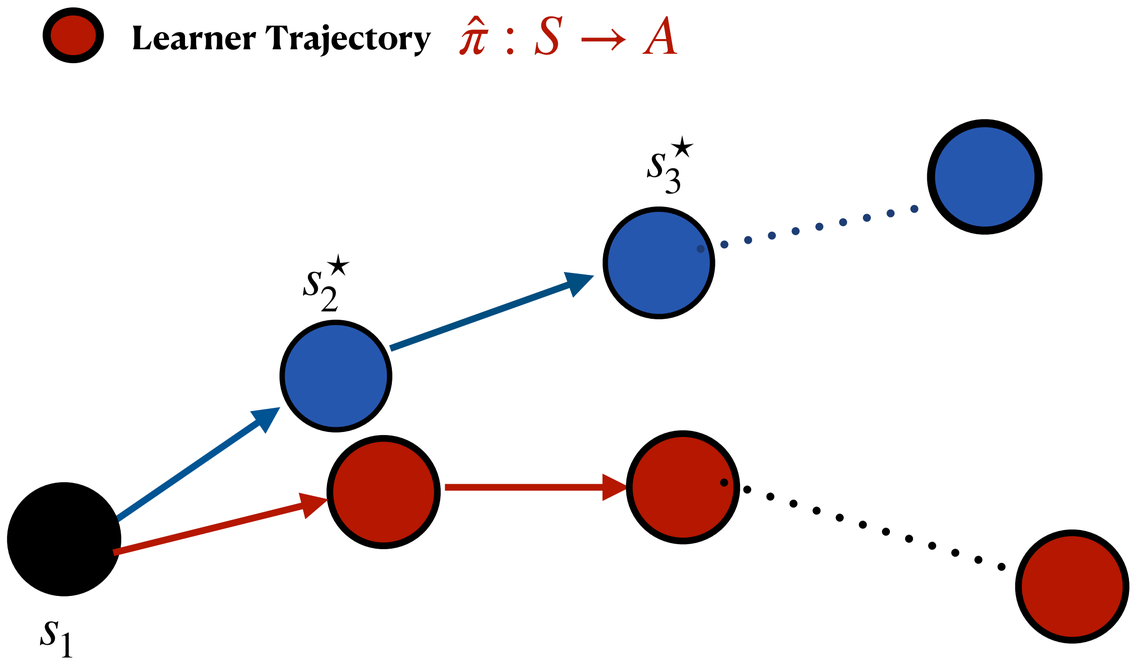
Central Challenge: Mismatch between BC error and cumulative reward:

Does $L_{\text{bc}}(\hat{\pi}) - L_{\text{bc}}(\pi^\star) \leq \text{small}$ **imply** $J(\pi^\star) - J(\hat{\pi}) \leq \text{small}$ **?**

BC Training v.s. Deployment



Learner selects $\hat{\pi}$ to minimize **supervised learning error** L_{bc}



But errors can accumulate!

aka '**curse of horizon**'

Summary: Imitation Learning / Behavior Cloning

Summary: Imitation Learning / Behavior Cloning

1. Imitation Learning (IL) learns from i.i.d expert data

Summary: Imitation Learning / Behavior Cloning

1. Imitation Learning (IL) learns from i.i.d expert data

2. Behavior Cloning (BC) solves IL via supervised learning

Summary: Imitation Learning / Behavior Cloning

1. Imitation Learning (IL) learns from i.i.d expert data

2. Behavior Cloning (BC) solves IL via supervised learning

3. Gap between train and test can lead to curse of horizon.

The Classical Formalism of Imitation Learning

3. Classical Perspectives on Behavior Cloning

Behavior Cloning in in the 0/1 loss

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Behavior Cloning in in the 0/1 loss

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

($\pi^\star$, $\hat{\pi}$ is deterministic)

Behavior Cloning in in the 0/1 loss

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

($\pi^\star$, $\hat{\pi}$ is deterministic)

Here, we focus on the 0/1 loss: $\text{loss}(\pi, s, a) = \mathbb{I}\{\pi(s) \neq a\}$

Behavior Cloning in in the 0/1 loss

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

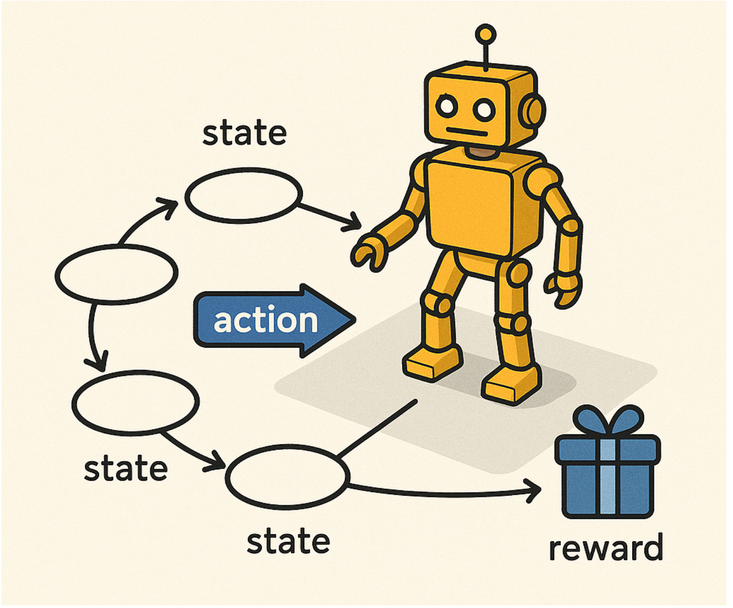
$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

($\pi^\star$, $\hat{\pi}$ is deterministic)

Here, we focus on the 0/1 loss: $\text{loss}(\pi, s, a) = \mathbb{I}\{\pi(s) \neq a\}$

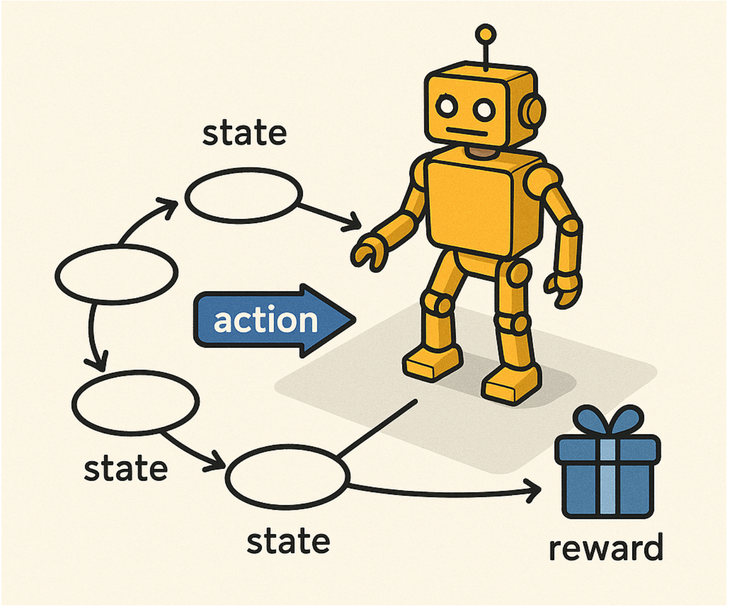
$$L_{\text{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \text{loss}(s_t, a_t, \pi)\right] = \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \mathbb{I}\{\pi(s_t) \neq \pi^\star(s_t)\}\right]$$

Values and Q-Functions



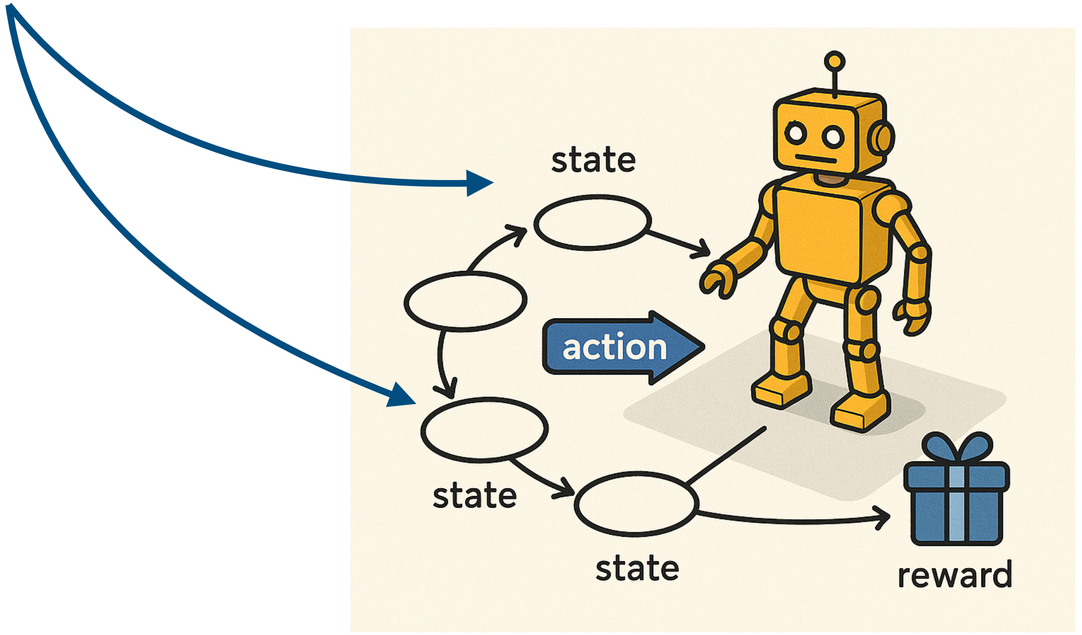
Values and Q-Functions

Definition: The **Q-function** is $Q_t^\pi(s,a) = \mathbb{E}^\pi \left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s \right]$



Values and Q-Functions

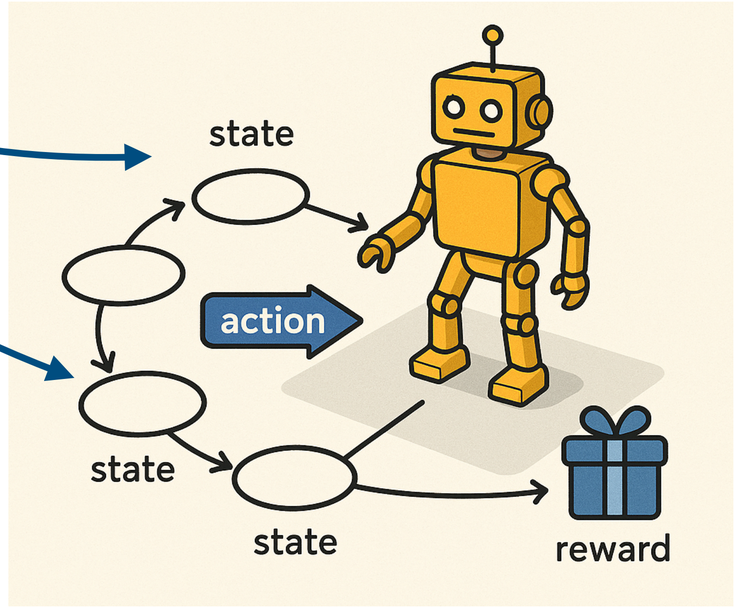
Definition: The **Q-function** is $Q_t^{\pi}(s,a) = \mathbb{E}^{\pi}\left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s\right]$



Values and Q-Functions

Definition: The **Q-function** is $Q_t^\pi(s,a) = \mathbb{E}^\pi \left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s \right]$

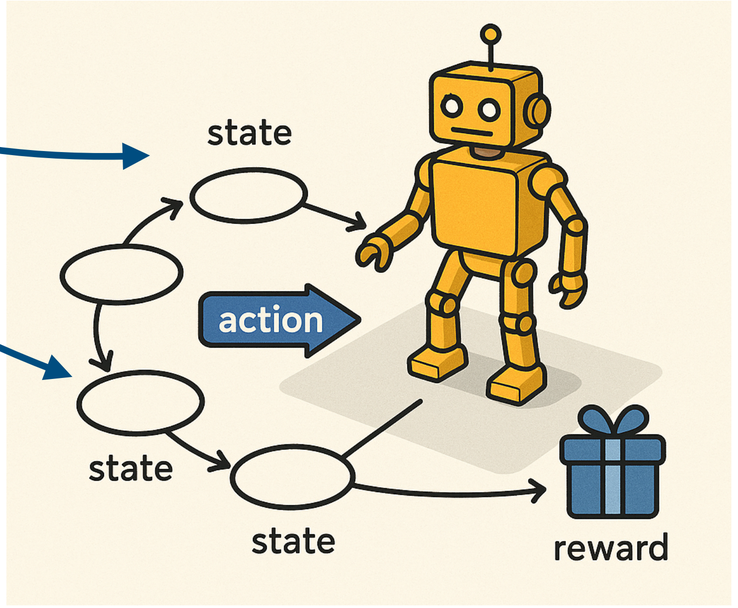
future reward from here



Values and Q-Functions

Definition: The **Q-function** is $Q_t^\pi(s,a) = \mathbb{E}^\pi \left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s \right]$

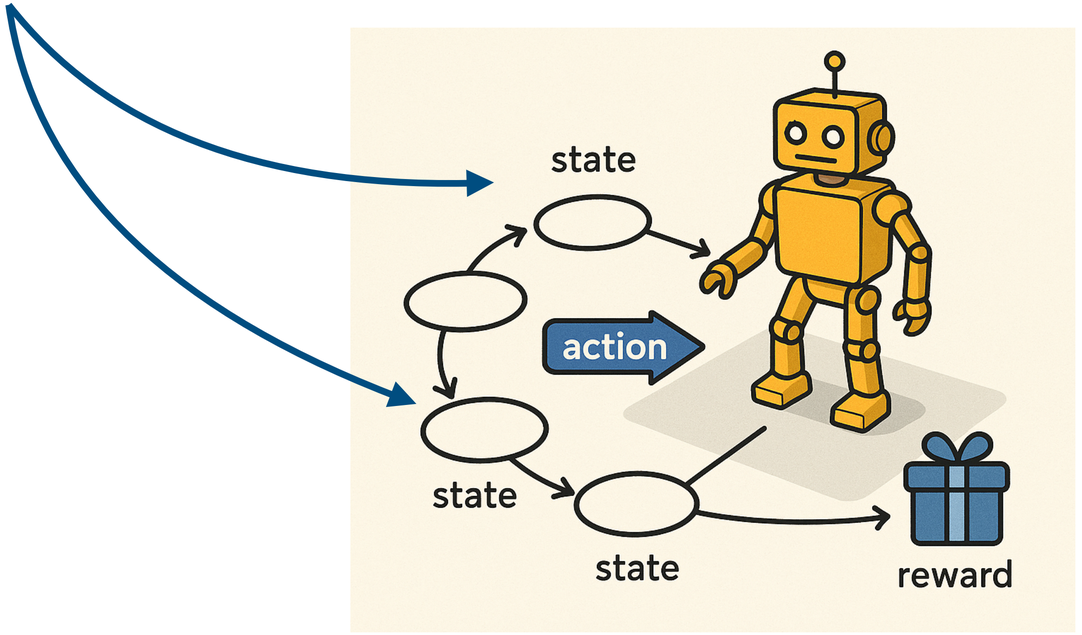
future reward from here



classical object from RL / Control

Values and Q-Functions

Definition: The **Q-function** is $Q_t^{\pi}(s,a) = \mathbb{E}^{\pi}\left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s\right]$

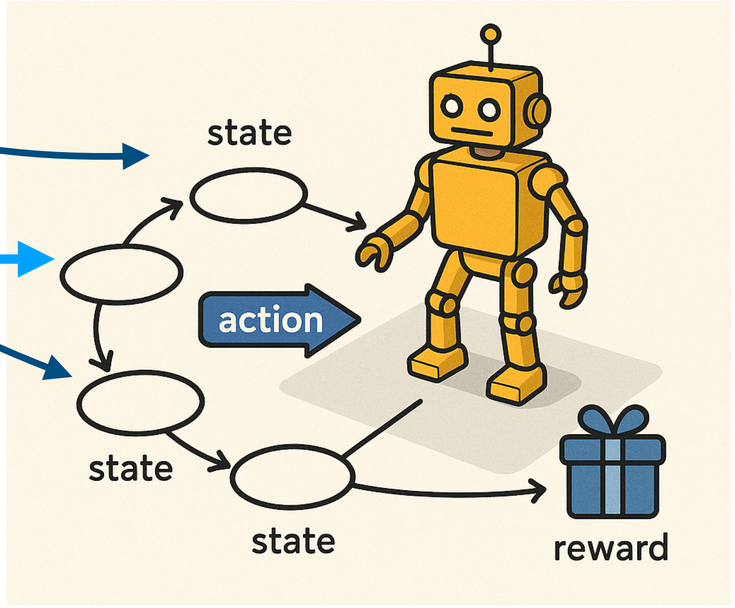


Values and Q-Functions

Definition: The **Q-function** is $Q_t^\pi(s,a) = \mathbb{E}^\pi \left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s \right]$

Definition: If π deterministic, **Value function**

$$V_t^\pi(s) = Q^\pi(s, \pi(s))$$

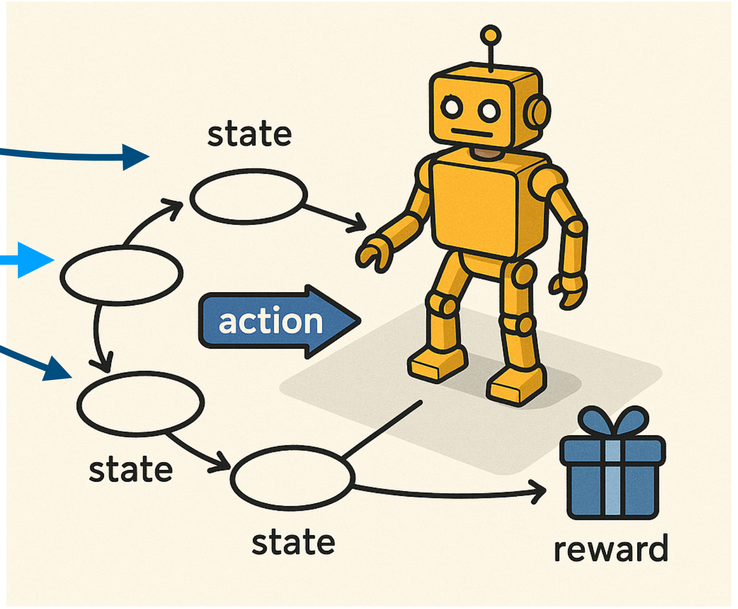


Values and Q-Functions

Definition: The **Q-function** is $Q_t^\pi(s,a) = \mathbb{E}^\pi \left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s \right]$

Definition: If π deterministic, **Value function**

$V_t^\pi(s) = Q^\pi(s, \pi(s))$



$J(\pi) = \mathbb{E}_{s \sim P_1} \left[V_1^\pi(s) \right]$

The Performance Difference Lemma

The Performance Difference Lemma

Lemma (Performance Difference, Kakade 2003)**:** Consider two policies, $\hat{\pi}$ and $\pi^\star$. Then, the difference in their cumulative return is

The Performance Difference Lemma

Lemma (Performance Difference, Kakade 2003)**:** Consider two policies, $\hat{\pi}$ and $\pi^\star$. Then, the difference in their cumulative return is

$J(\pi^\star) - J(\hat{\pi}) =$ **controlled by what?**

The Performance Difference Lemma

Lemma (Performance Difference, Kakade 2003)**:** Consider two policies, $\hat{\pi}$ and $\pi^\star$. Then, the difference in their cumulative return is

$$J(\pi^\star) - J(\hat{\pi}) = \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

The Performance Difference Lemma

Lemma (Performance Difference, Kakade 2003)**:** Consider two policies, $\hat{\pi}$ and $\pi^\star$. Then, the difference in their cumulative return is

$$J(\pi^\star) - J(\hat{\pi}) = \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

$(s_t, a_t) \sim \pi^\star$

The Performance Difference Lemma

Lemma (Performance Difference, Kakade 2003)**:** Consider two policies, $\hat{\pi}$ and $\pi^\star$. Then, the difference in their cumulative return is

$$J(\pi^\star) - J(\hat{\pi}) = \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

$(s_t, a_t) \sim \pi^\star$

Q-function under $\hat{\pi}$

The Performance Difference Lemma

Lemma (Performance Difference, Kakade 2003)**:** Consider two policies, $\hat{\pi}$ and $\pi^\star$. Then, the difference in their cumulative return is

$$J(\pi^\star) - J(\hat{\pi}) = \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

$(s_t, a_t) \sim \pi^\star$

Q-function under $\hat{\pi}$

$\hat{a}_t \mid s_t \sim \hat{\pi}$

Behavior Cloning in in the 0/1 loss

Behavior Cloning in in the 0/1 loss

Theorem: Suppose that $r(s,a) \geq 0, \ \sum_{t=1}^{H} r(s,a) \in [0,R]$.

Behavior Cloning in in the 0/1 loss

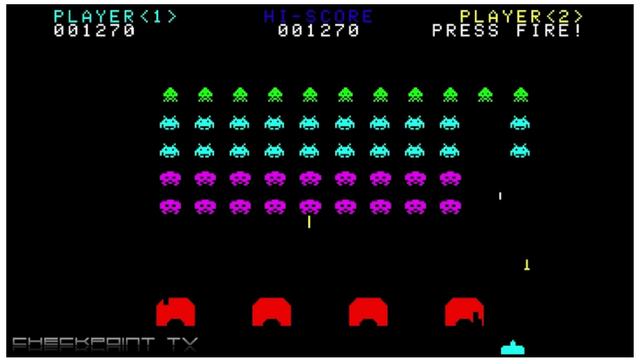
Theorem: Suppose that $r(s,a) \geq 0,\ \sum_{t=1}^{H} r(s,a) \in [0,R]$.

E.g: **Sparse Reward** $R \leq 1$

Behavior Cloning in in the 0/1 loss

Theorem: Suppose that $r(s,a) \geq 0$, $\sum_{t=1}^{H} r(s,a) \in [0,R]$.

E.g: **Sparse Reward** $R \leq 1$



Dense Reward $R \leq H$

Behavior Cloning in in the 0/1 loss

Behavior Cloning in in the 0/1 loss

Theorem: Suppose that $r(s,a) \geq 0,\ \sum_{t=1}^{H} r(s,a) \in [0,R]$. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

Behavior Cloning in in the 0/1 loss

Theorem: Suppose that $r(s,a) \geq 0,\ \sum_{t=1}^{H} r(s,a) \in [0,R]$. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) = \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

Behavior Cloning in in the 0/1 loss

Theorem: Suppose that $r(s,a) \geq 0,\ \sum_{t=1}^{H} r(s,a) \in [0,R]$. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) = \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

$$\leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} R \cdot \mathbb{I}(a_t \neq \hat{a}_t)\right]$$

Behavior Cloning in in the 0/1 loss

Theorem: Suppose that $r(s,a) \geq 0,\ \sum_{t=1}^{H} r(s,a) \in [0,R]$. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) = \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

$$\leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} R \cdot \mathbb{I}(a_t \neq \hat{a}_t)\right]$$

$$= H \cdot R \cdot L_{\text{bc}}(\hat{\pi})$$

Behavior Cloning in in the 0/1 loss

Theorem: Suppose that $(s,a) \in [0,1]$, $\sum_{t=1}^{H} r(s,a) \in [0,R]$. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq H \cdot R \cdot L_{\text{bc}}(\hat{\pi})$$

1. **General reduction** from supervised learning to low imitation error

Behavior Cloning in in the 0/1 loss

Theorem: Suppose that $(s,a) \in [0,1]$, $\sum_{t=1}^{H} r(s,a) \in [0,R]$. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq H \cdot R \cdot L_{\mathrm{bc}}(\hat{\pi})$$

1. **General reduction** from supervised learning to low imitation error
2. $L_{\mathrm{bc}}(\hat{\pi})$ can be bounded by generalization error, e.g. $\lesssim \frac{1}{n} \log |\Pi|$

(ERM with finite Π, deterministic policies)

Behavior Cloning in in the 0/1 loss

Theorem: Suppose that $(s,a) \in [0,1]$, $\sum_{t=1}^{H} r(s,a) \in [0,R]$. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq H \cdot R \cdot L_{\text{bc}}(\hat{\pi})$$

Seminal Result by Ross & Bagnell ’10

Error Compounding

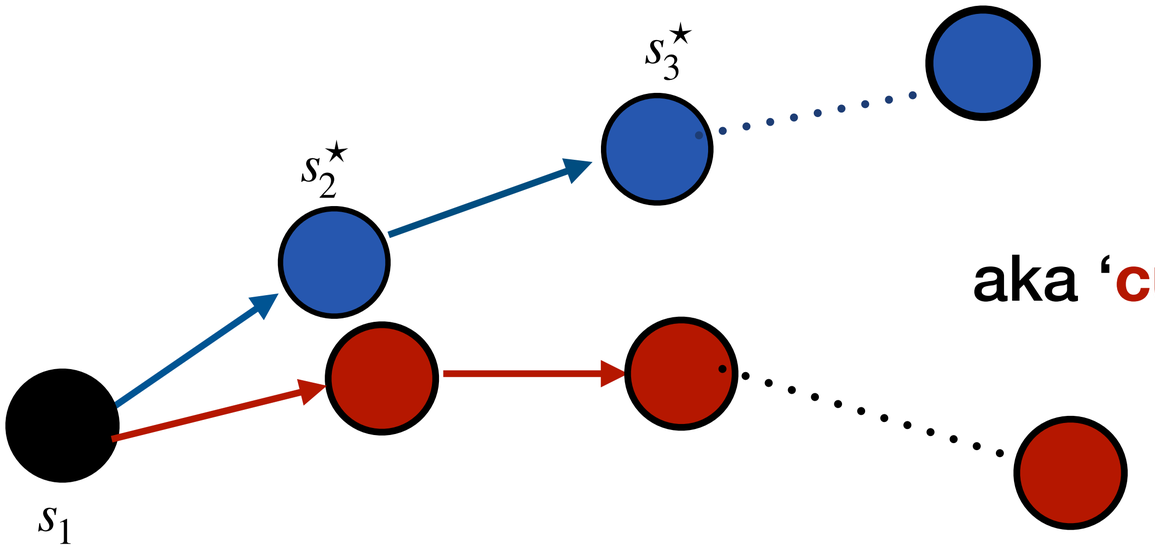
Theorem: Suppose that $(s,a) \in [0,1]$, $\sum_{t=1}^{H} r(s,a) \in [0,R]$. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq H \cdot R \cdot L_{\text{bc}}(\hat{\pi}) \leq H \cdot R \cdot \frac{1}{n} \log |\Pi|$$

Error Compounding

Theorem: Suppose that $(s,a) \in [0,1]$, $\sum_{t=1}^{H} r(s,a) \in [0,R]$. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq \underline{H} \cdot R \cdot L_{\text{bc}}(\hat{\pi}) \leq \underline{H} \cdot R \cdot \frac{1}{n} \log |\Pi|$$



aka '**curse of horizon**'

Error Compounding

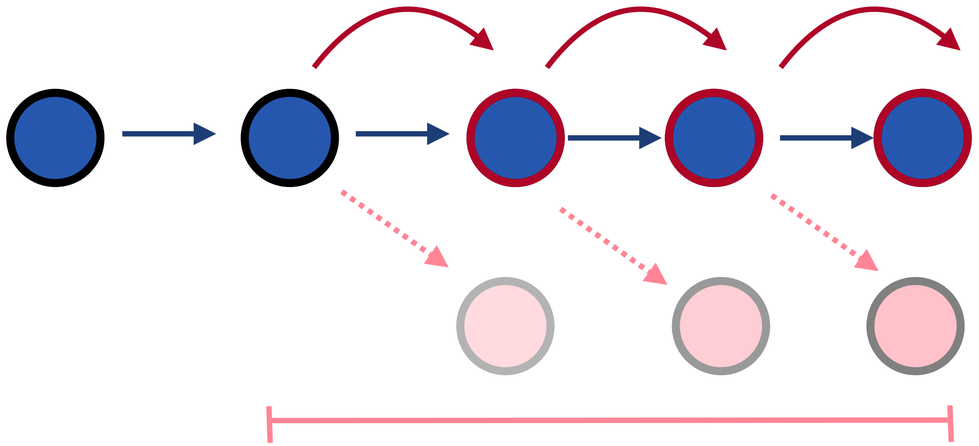
Theorem: Suppose that $(s,a) \in [0,1]$, $\sum_{t=1}^{H} r(s,a) \in [0,R]$. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq H \cdot R \cdot L_{\text{bc}}(\hat{\pi}) \leq H \cdot R \cdot \frac{1}{n} \log |\Pi|$$

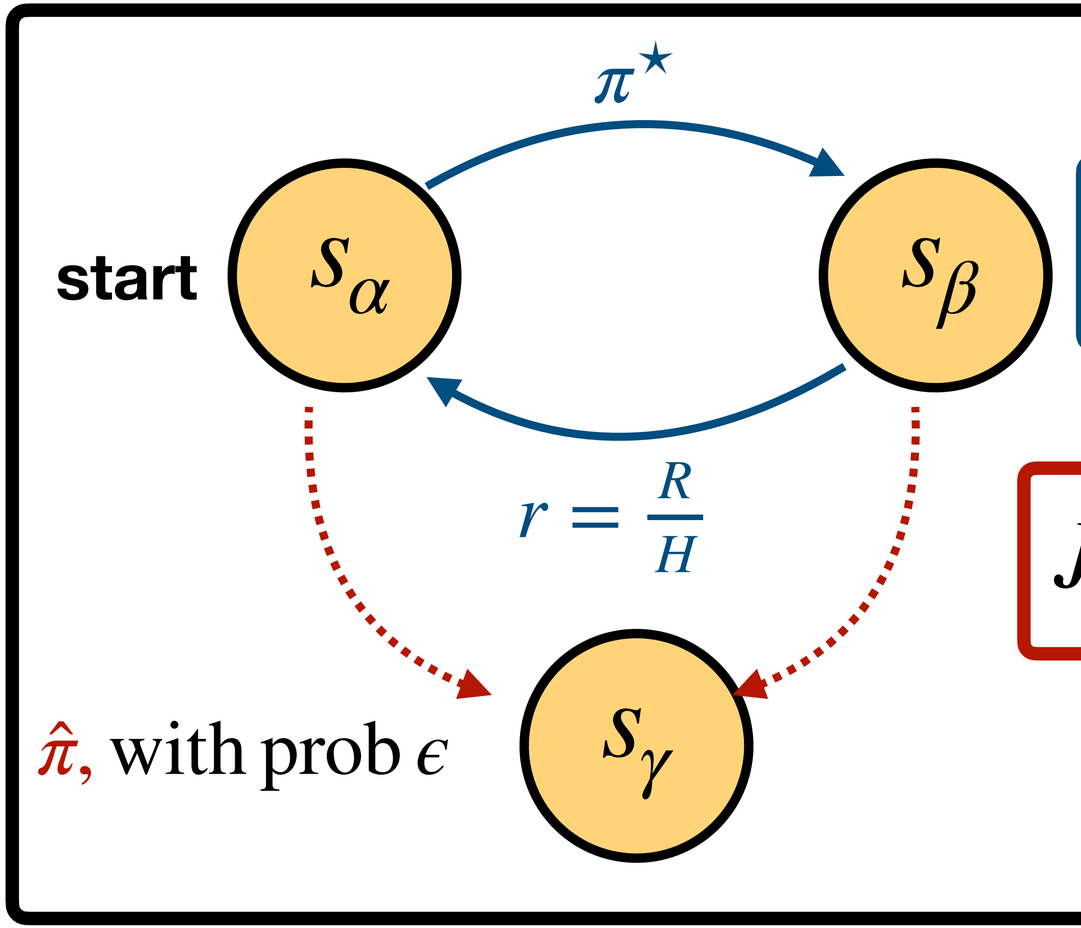
Error Compounding

Theorem: Suppose that $(s,a) \in [0,1]$, $\sum_{t=1}^{H} r(s,a) \in [0,R]$. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq H \cdot R \cdot L_{\text{bc}}(\hat{\pi}) \leq H \cdot R \cdot \frac{1}{n} \log |\Pi|$$



The curse of horizon is optimal



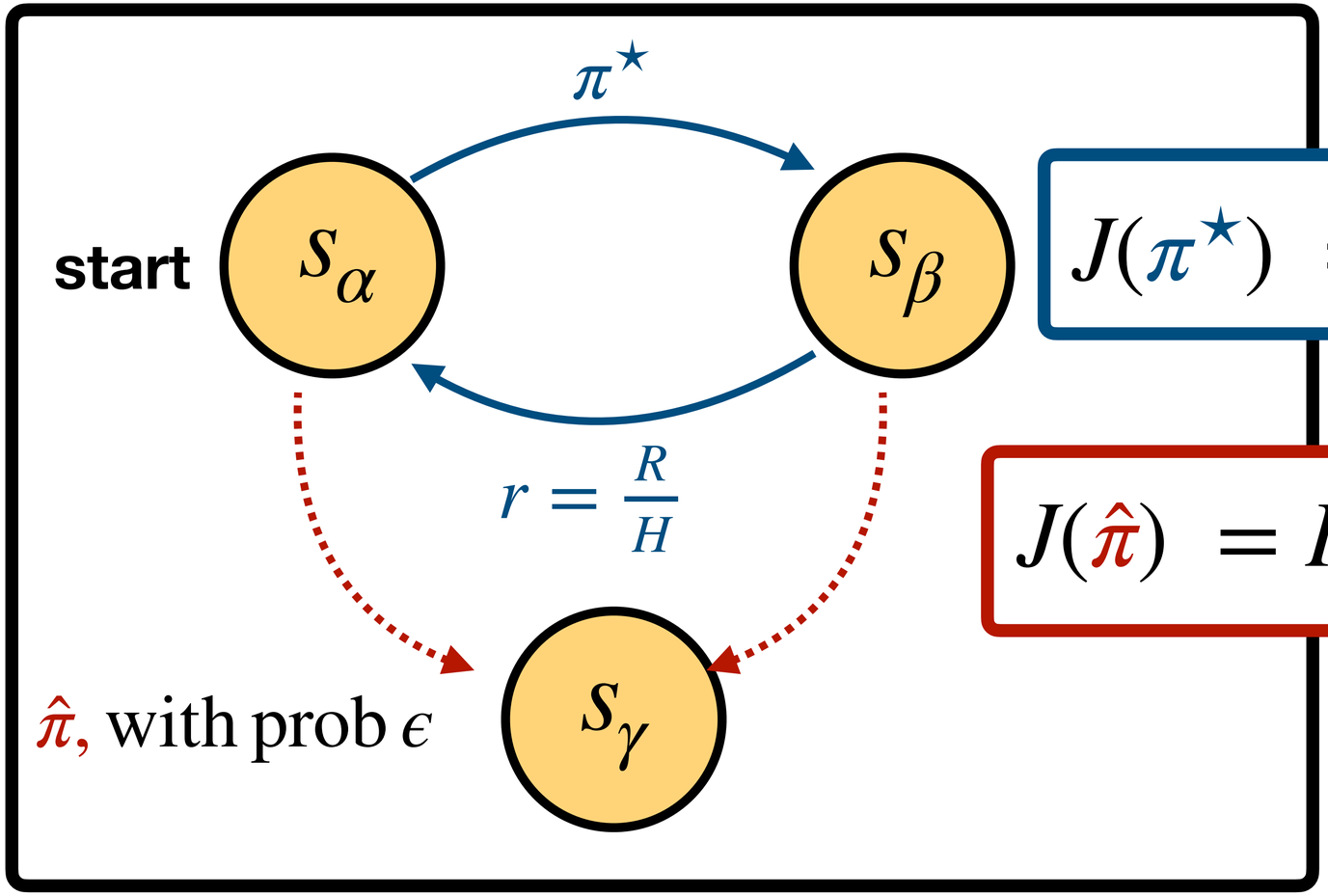
$$J(\pi^\star) = R$$

$$J(\hat{\pi}) = R - \Omega\left(\sum_{h=1}^{H} (H-h) \cdot \frac{R}{H} \cdot \epsilon\right)$$

$$= J(\pi^\star) - \Omega(H \cdot R) \cdot L_{\text{bc}}(\hat{\pi})$$

(Ross & Bagnell, 2010)

The curse of horizon is optimal



$$J(\pi^\star) = R$$

$$J(\hat{\pi}) = R - \Omega\left(\sum_{h=1}^{H} (H-h) \cdot \frac{R}{H} \cdot \epsilon\right)$$

$$= J(\pi^\star) - \Omega(H \cdot R) \cdot L_{\text{bc}}(\hat{\pi})$$

(Ross & Bagnell, 2010)

(at least for the indicator loss…)

Summary: Curse of Horizon

Summary: Curse of Horizon

1. There are many choices of loss for BC error

Summary: Curse of Horizon

1. There are many choices of loss for BC error

2. For the 0/1 loss, errors are magnified by a factor of horizon

Summary: Curse of Horizon

1. There are many choices of loss for BC error

2. For the 0/1 loss, errors are magnified by a factor of horizon

foreshadow: things may look very different for other losses!

Summary: Curse of Horizon

1. There are many choices of loss for BC error

2. For the 0/1 loss, errors are magnified by a factor of horizon

foreshadow: things may look very different for other losses!

Are there other interventions that attempt to mitigate this?

The Classical Formalism of Imitation Learning

4. Variational Perspective and Inverse RL

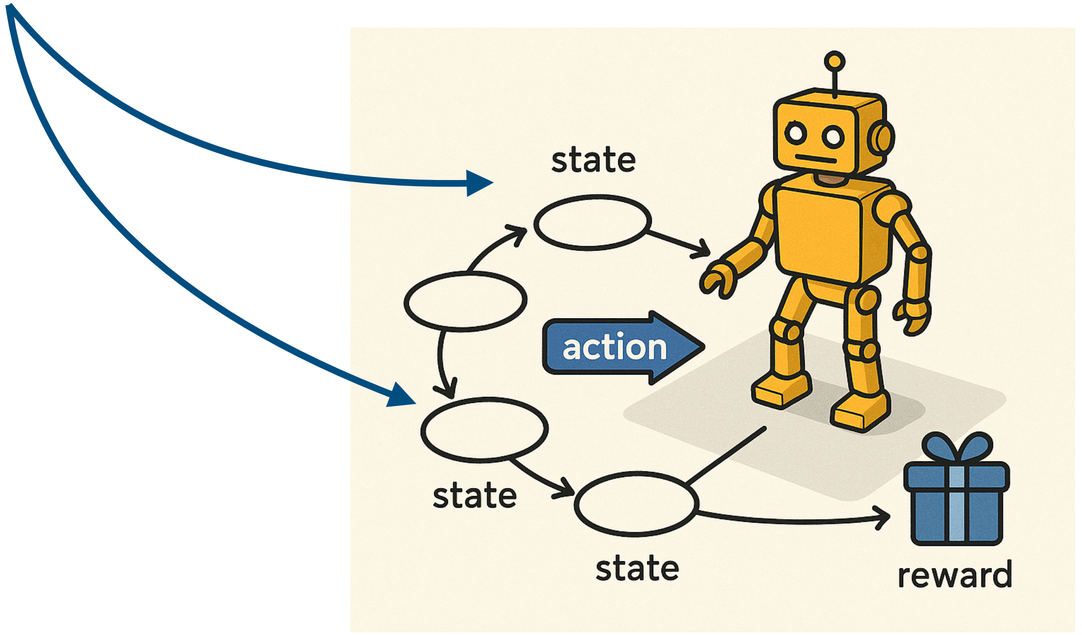
Variational BC Loss Function

Variational BC Loss Function

Motivation: Can we define a BC Loss that depends on MDP Structure?

Values and Q-Functions

Definition: The **Q-function** is $Q_t^{\pi}(s,a) = \mathbb{E}^{\pi}\left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s\right]$

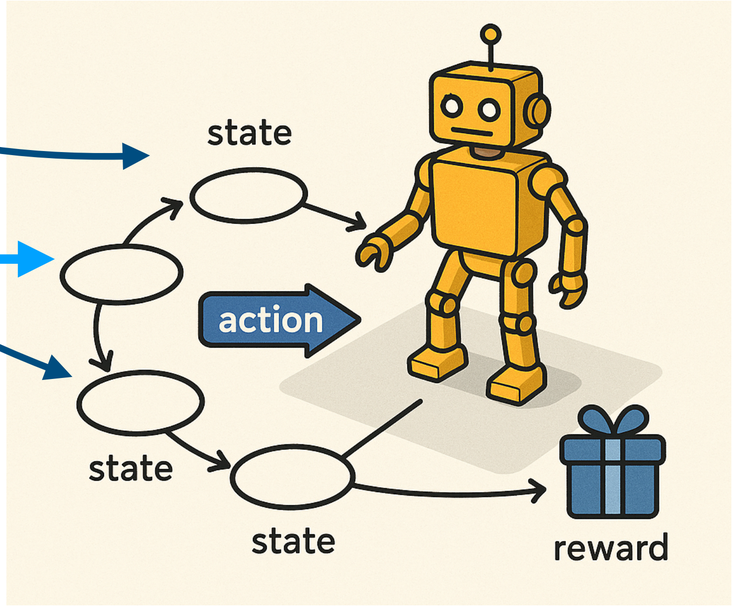


Values and Q-Functions

Definition: The **Q-function** is $Q_t^\pi(s,a) = \mathbb{E}^\pi \left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s \right]$

Definition: The **Value function** is

$$V_t^\pi(s) = \mathbb{E}^\pi \left[\sum_{s=t}^{H} r(s,a) \mid s_t = s \right]$$

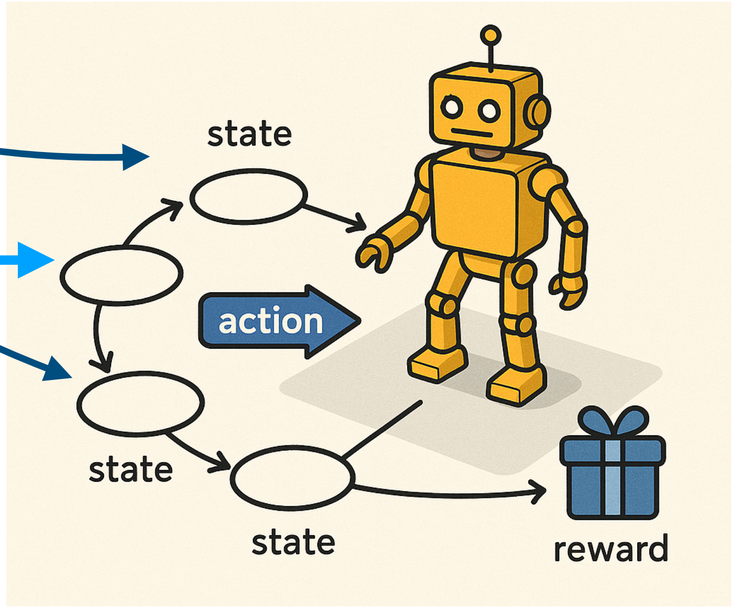


Values and Q-Functions

Definition: The **Q-function** is $Q_t^\pi(s,a) = \mathbb{E}^\pi \left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s \right]$

Definition: The **Value function** is

$$V_t^\pi(s) = \mathbb{E}^\pi \left[\sum_{s=t}^{H} r(s,a) \mid s_t = s \right]$$



Note: $V_t^\pi(s) = \mathbb{E}_{a \sim \pi(s)} \left[Q_t^\pi(s,a) \right]$

Variational BC Loss

$$J(\pi^\star) - J(\hat{\pi}) \leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

Variational BC Loss

$$J(\pi^\star) - J(\hat{\pi}) \leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

$$\leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} \sup_Q \left(\mathbb{E}_{a \sim \pi^\star(s_t)}[Q(s,a)] - \mathbb{E}_{a \sim \hat{\pi}(s_t)}[Q(s,a)]\right)\right]$$

Variational BC Loss

$$J(\pi^\star) - J(\hat{\pi}) \leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

$$\leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} \sup_Q \left(\mathbb{E}_{a\sim\pi^\star(s_t)}[Q(s,a)] - \mathbb{E}_{a\sim\hat{\pi}(s_t)}[Q(s,a)]\right)\right]$$

$$:= \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} L_{\text{IPM}}(\pi^\star(\cdot \mid s), \hat{\pi}(\cdot \mid s))\right]$$

Variational BC Loss

$$J(\pi^\star) - J(\hat{\pi}) \leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

$$\leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} \sup_Q \left(\mathbb{E}_{a\sim\pi^\star(s_t)}[Q(s,a)] - \mathbb{E}_{a\sim\hat{\pi}(s_t)}[Q(s,a)]\right)\right]$$

$$:= \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} L_{\text{IPM}}(\pi^\star(\cdot \mid s), \hat{\pi}(\cdot \mid s))\right]$$

‘Integral Probability Metric’ induced by Q functions

Variational BC Loss

$$J(\pi^\star) - J(\hat{\pi}) \leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

$$\leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} \sup_Q \left(\mathbb{E}_{a\sim\pi^\star(s_t)}[Q(s,a)] - \mathbb{E}_{a\sim\hat{\pi}(s_t)}[Q(s,a)]\right)\right]$$

$$:= \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} L_{\text{IPM}}(\pi^\star(\cdot \mid s), \hat{\pi}(\cdot \mid s))\right]$$

‘Integral Probability Metric’ induced by Q functions

gives a first-principles derivation of BC Losses from reward structure (see Swamy et al. ’21)

Variational BC Loss

$$J(\pi^\star) - J(\hat{\pi}) \leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

$$\leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} \sup_Q \left(\mathbb{E}_{a\sim\pi^\star(s_t)}[Q(s,a)] - \mathbb{E}_{a\sim\hat{\pi}(s_t)}[Q(s,a)]\right)\right]$$

$$:= \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} L_{\text{IPM}}(\pi^\star(\cdot \mid s), \hat{\pi}(\cdot \mid s))\right]$$

‘Integral Probability Metric’ induced by Q functions

gives a first-principles derivation of BC Losses from reward structure (see Swamy et al. ’21)

can specialize to 0/1 loss BC and Wasserstein as a special case

Variational BC Loss

$$J(\pi^\star) - J(\hat{\pi}) \leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} Q_t^{\hat{\pi}}(s_t, a_t) - Q_t^{\hat{\pi}}(s_t, \hat{a}_t)\right]$$

$$\leq \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} \sup_Q \left(\mathbb{E}_{a\sim\pi^\star(s_t)}[Q(s,a)] - \mathbb{E}_{a\sim\hat{\pi}(s_t)}[Q(s,a)]\right)\right]$$

$$:= \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} L_{\text{IPM}}(\pi^\star(\cdot \mid s), \hat{\pi}(\cdot \mid s))\right]$$

This represents BC as a variational (min-max) problem

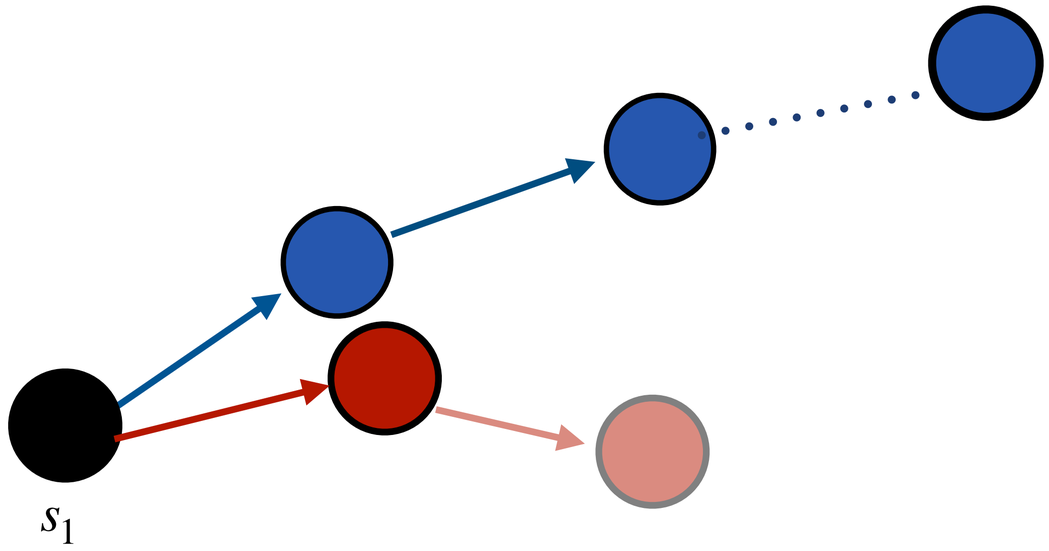
Protocol: Inverse RL

Protocol: Inverse RL

Motivation: Can interaction with the environment help?

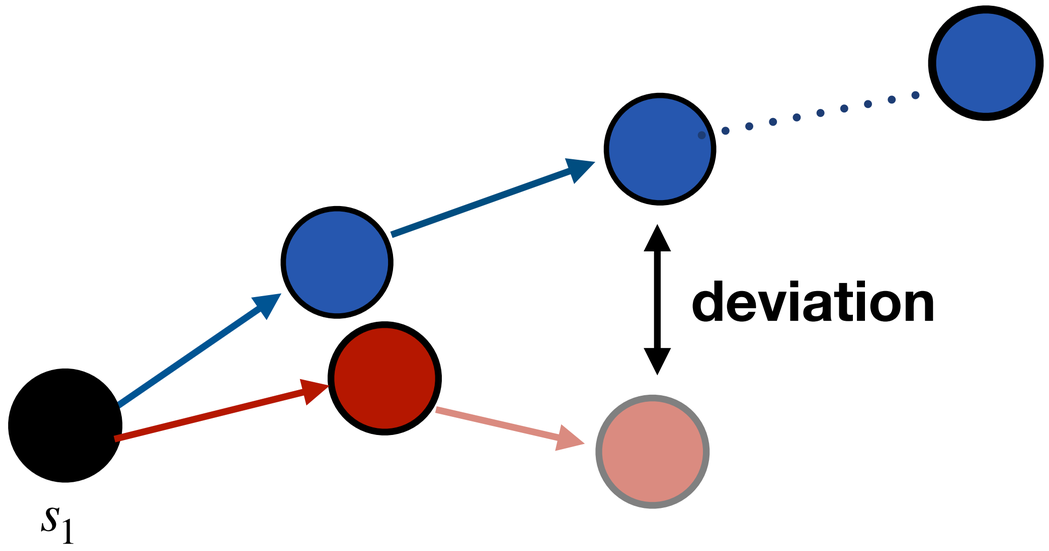
Protocol: Inverse RL

Motivation: Can interaction with the environment help?



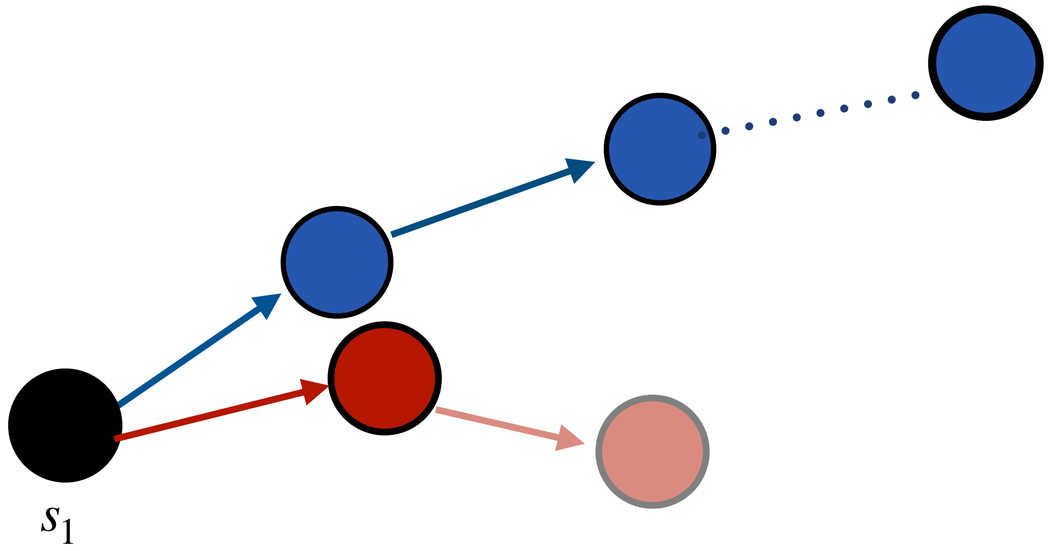
Protocol: Inverse RL

Motivation: Can interaction with the environment help?



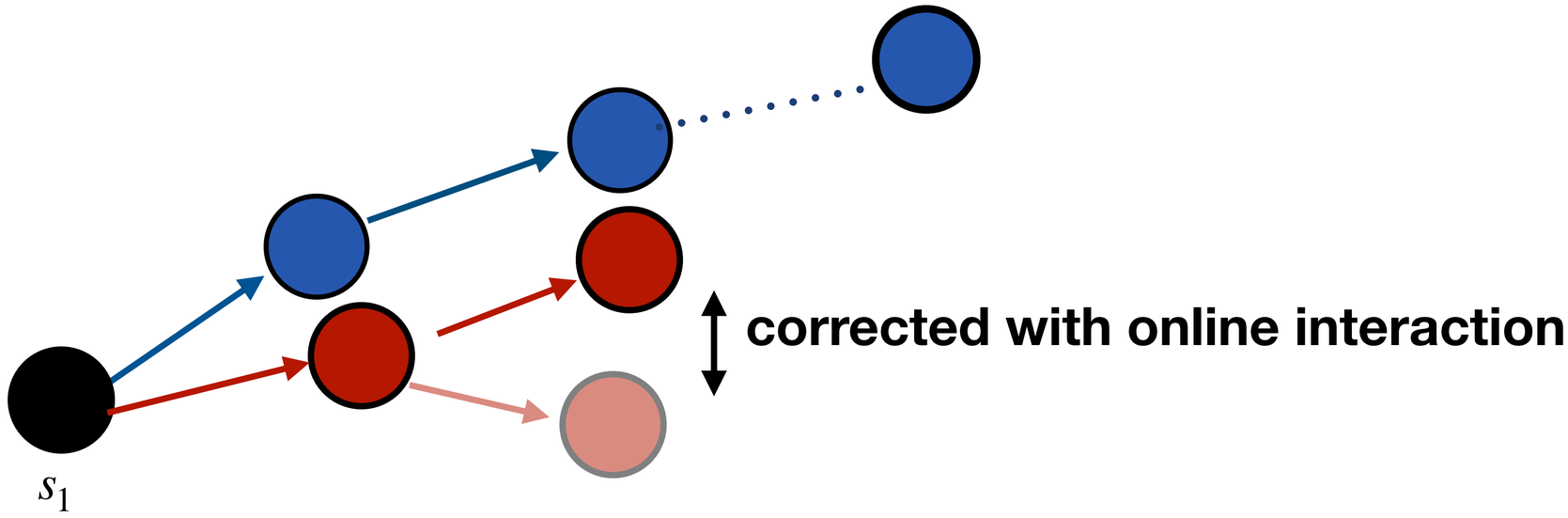
Protocol: Inverse RL

Motivation: Can interaction with the environment help?



Protocol: Inverse RL

Motivation: Can interaction with the environment help?



Protocol: Inverse RL

Protocol: Inverse RL

Definition: Inverse Reinforcement Learning (IRL) consists of 3 steps:

Protocol: Inverse RL

Definition: Inverse Reinforcement Learning (IRL) consists of 3 steps:

1. The expert $\pi^\star$ collects **n** demonstration trajectories (like IL/BC).

Protocol: Inverse RL

Definition: **Inverse Reinforcement Learning (IRL)** **consists of 3 steps:**

1. The expert $\pi^\star$ collects **n** demonstration trajectories (like IL/BC).

2. In rounds $k = 1,2,\ldots$ learner executes $\pi^{(k)}$ and sees trajectories **but no reward.**

Protocol: Inverse RL

Definition: Inverse Reinforcement Learning (IRL) consists of 3 steps:

1. The expert $\pi^\star$ collects **n** demonstration trajectories (like IL/BC).

2. In rounds $k = 1,2,\ldots$ learner executes $\pi^{(k)}$ and sees trajectories **but no reward.**

3. After collecting information, we return a policy $\hat{\pi}$

Protocol: Inverse RL

Definition: Inverse Reinforcement Learning (IRL) consists of 3 steps:

1. The expert $\pi^\star$ collects **n** demonstration trajectories (like IL/BC).

2. In rounds $k = 1,2,\ldots$ learner executes $\pi^{(k)}$ and sees trajectories **but no reward.**

3. After collecting information, we return a policy $\hat{\pi}$

Learner gets both expert data and online interaction

Protocol: Inverse RL

Definition: Inverse Reinforcement Learning (IRL) consists of 3 steps:

1. The expert $\pi^\star$ collects **n** demonstration trajectories (like IL/BC).

2. In rounds $k = 1,2,\ldots$ learner executes $\pi^{(k)}$ and sees trajectories **but no reward.**

3. After collecting information, we return a policy $\hat{\pi}$

Protocol: Inverse RL

Definition: Inverse Reinforcement Learning (IRL) consists of 3 steps:

1. The expert $\pi^\star$ collects **n** demonstration trajectories (like IL/BC).

2. In rounds $k = 1,2,\ldots$ learner executes $\pi^{(k)}$ and sees trajectories **but no reward.**

3. After collecting information, we return a policy $\hat{\pi}$

Algorithms formulated as min-max games over 'worst-case reward'

Protocol: Inverse RL

Definition: Inverse Reinforcement Learning (IRL) consists of 3 steps:

1. The expert $\pi^\star$ collects **n** demonstration trajectories (like IL/BC).

2. In rounds $k = 1,2,\ldots$ learner executes $\pi^{(k)}$ and sees trajectories **but no reward.**

3. After collecting information, we return a policy $\hat{\pi}$

Algorithms formulated as min-max games over 'worst-case reward'

Abeel and Ng 2004 *Ho & Ermon 2016*

Summary: Variational Approaches

Summary: Variational Approaches

1. Many ways to formulate imitation learning as a min-max game between cost and policy

Summary: Variational Approaches

1. Many ways to formulate imitation learning as a min-max game between cost and policy

2. The formulation based on performance difference lemma can relates BC error to the structure of the MDP

Summary: Variational Approaches

1. Many ways to formulate imitation learning as a min-max game between cost and policy

2. The formulation based on performance difference lemma can relates BC error to the structure of the MDP

3. In Inverse-RL, we can leverage interaction with the environment to further ensure closeness to expert

Summary: Variational Approaches

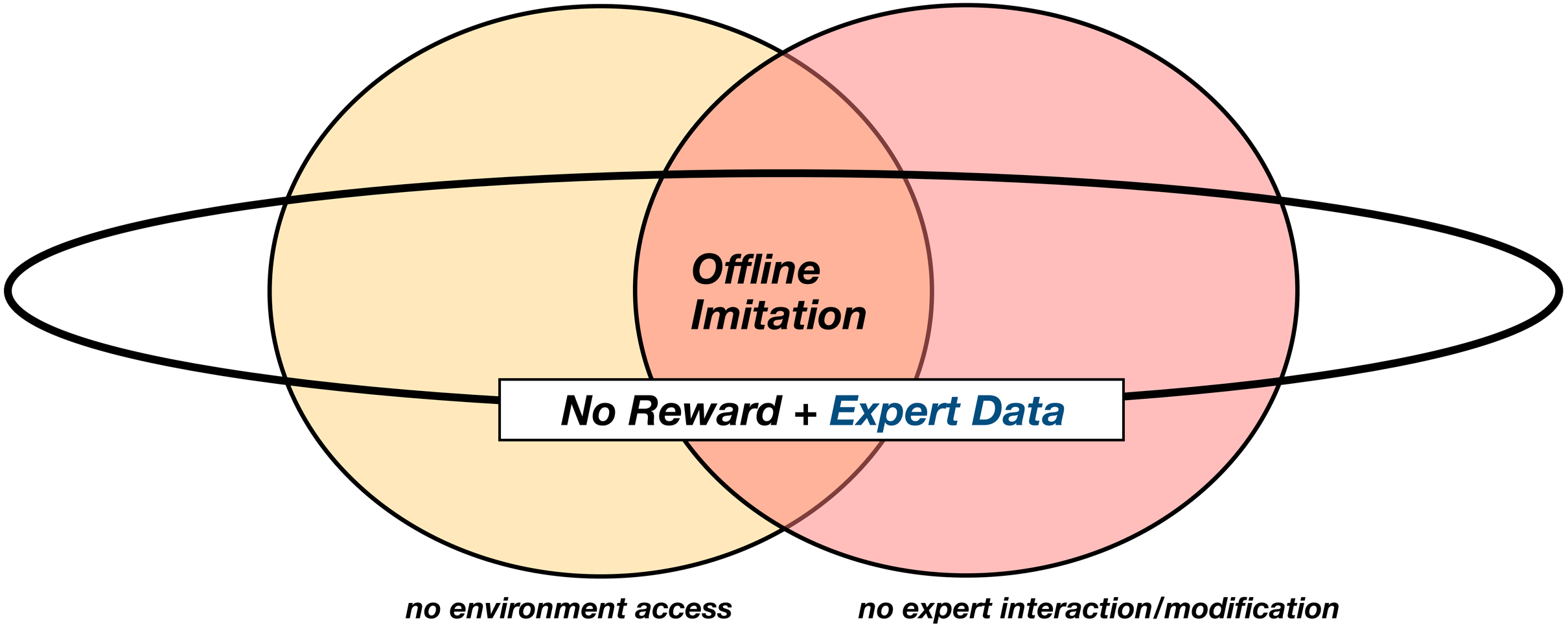
1. Many ways to formulate imitation learning as a min-max game between cost and policy

2. The formulation based on performance difference lemma can relates BC error to the structure of the MDP

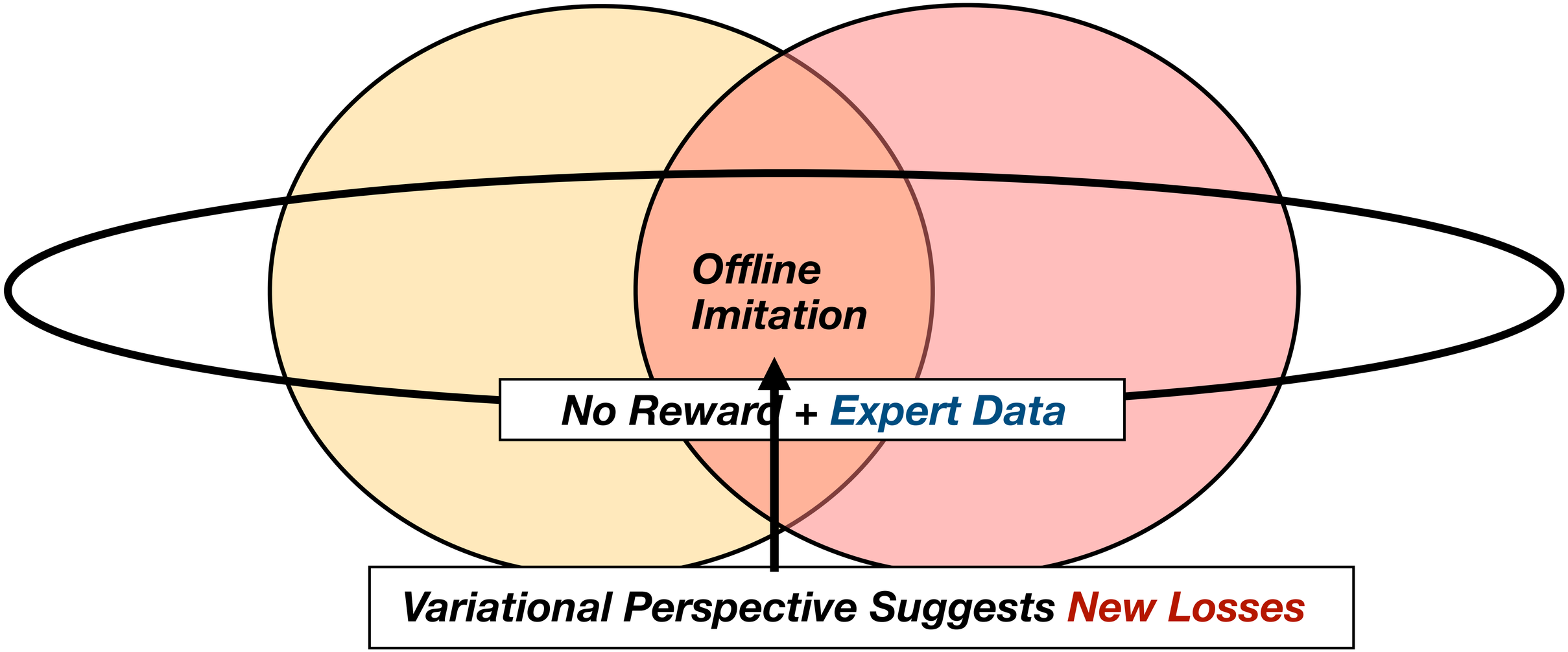
3. In Inverse-RL, we can leverage interaction with the environment to further ensure closeness to expert

open question: establishing concrete benefits of these perspectives

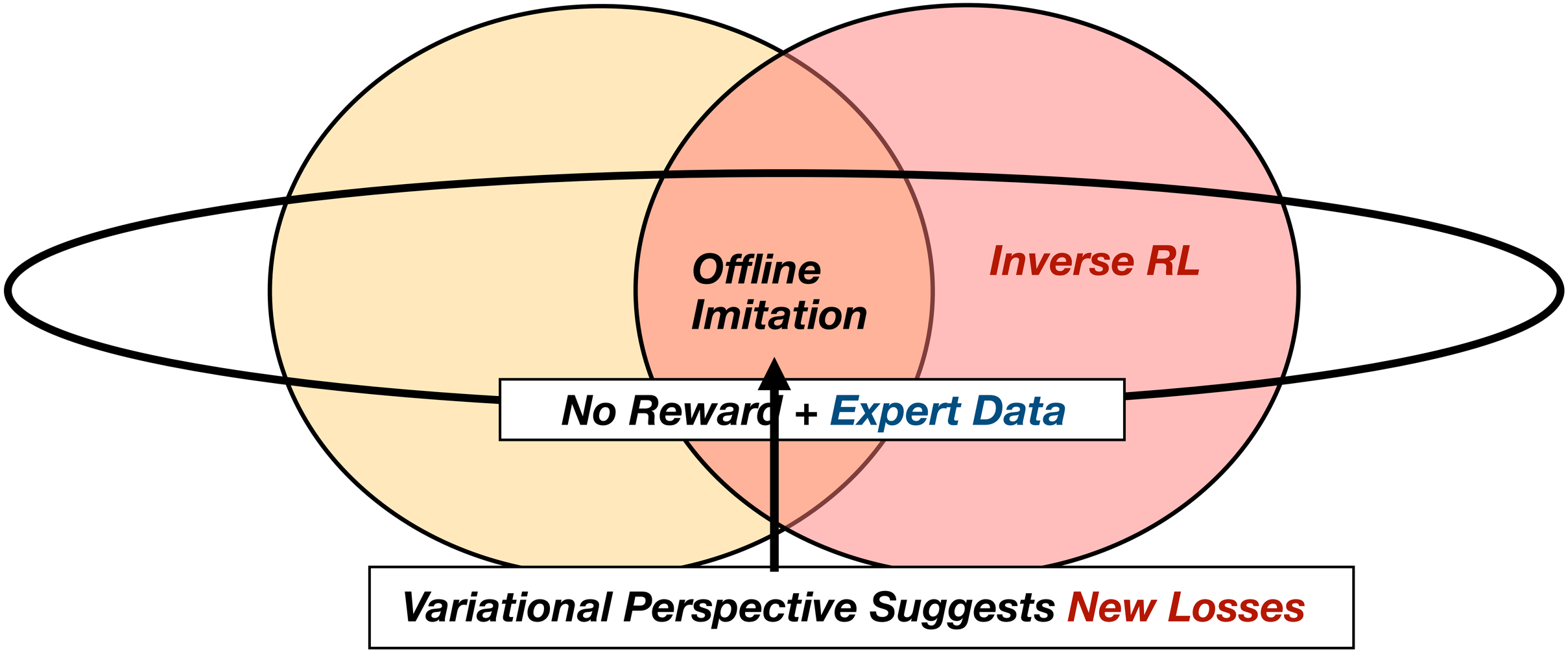
Summary: Variational Approaches



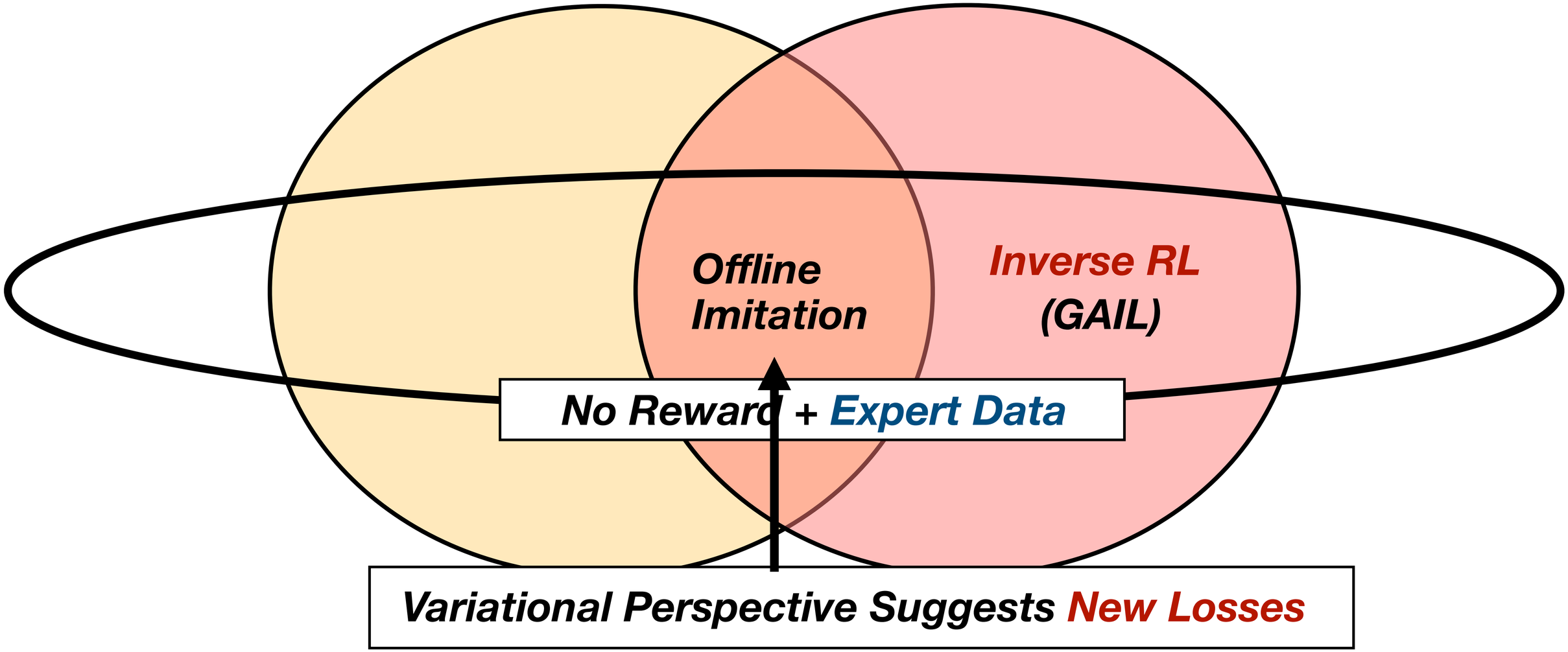
Summary: Variational Approaches



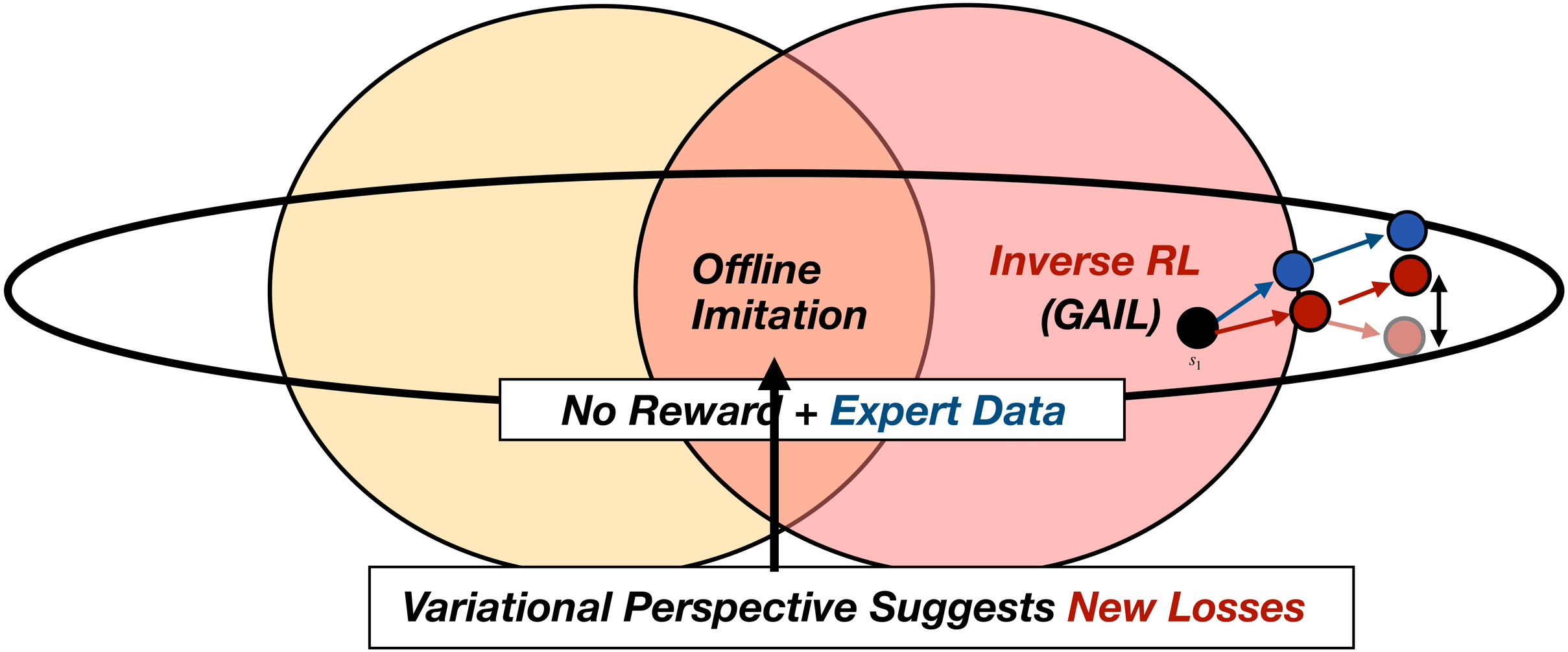
Summary: Variational Approaches



Summary: Variational Approaches



Summary: Variational Approaches



The Classical Formalism of Imitation Learning

5. Adaptive Data Collection (DAGGER, DART)

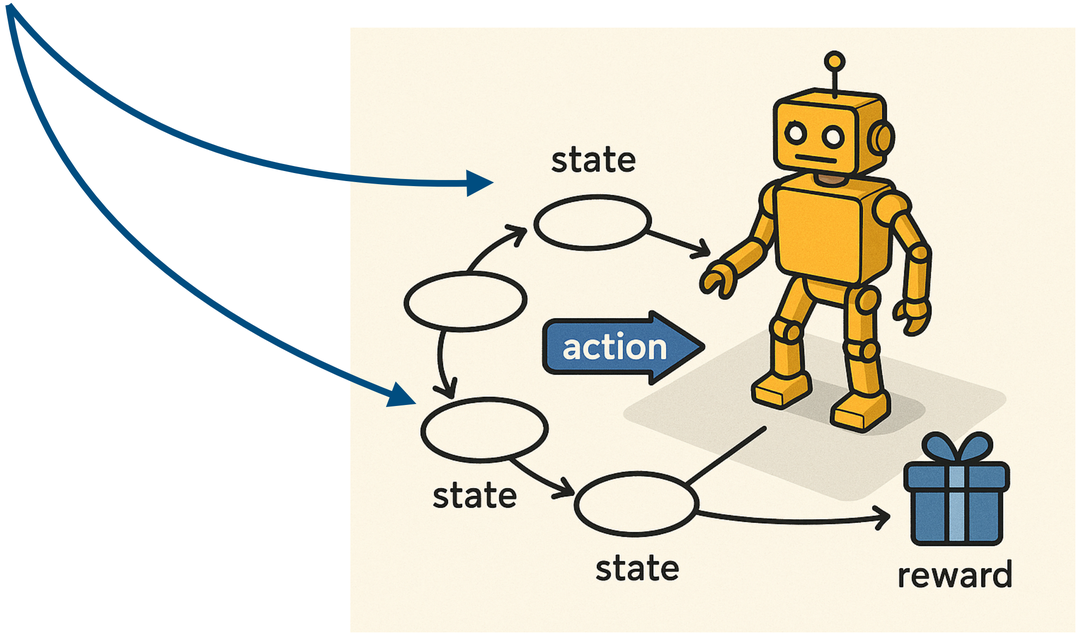
Expert Interaction

Expert Interaction

Motivation: Can interaction with expert or better data collection mitigate compounding error?

Values and Q-Functions

Definition: The **Q-function** is $Q_t^{\pi}(s,a) = \mathbb{E}^{\pi}\left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s\right]$

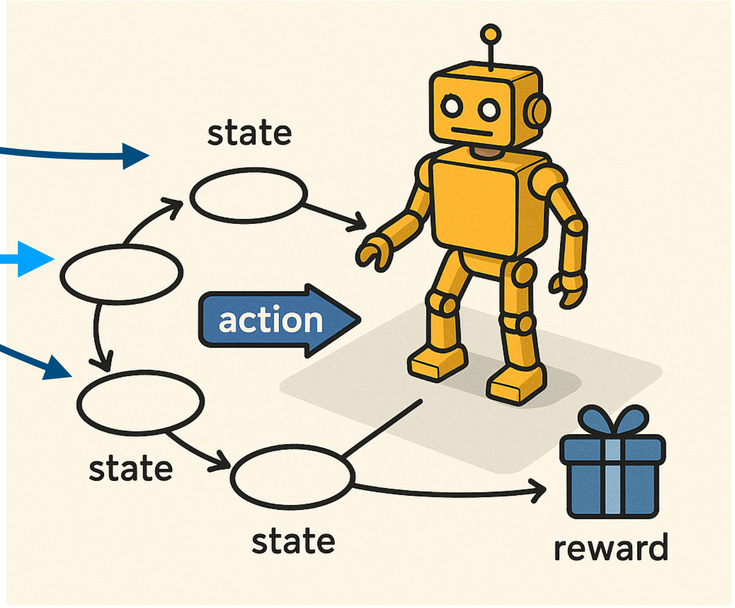


Values and Q-Functions

Definition: The **Q-function** is $Q_t^\pi(s,a) = \mathbb{E}^\pi \left[\sum_{s=t}^{H} r(s,a) \mid a_t = a, s_t = s \right]$

Definition: If π deterministic, **Value function**

$$V_t^\pi(s) = Q^\pi(s, \pi(s))$$



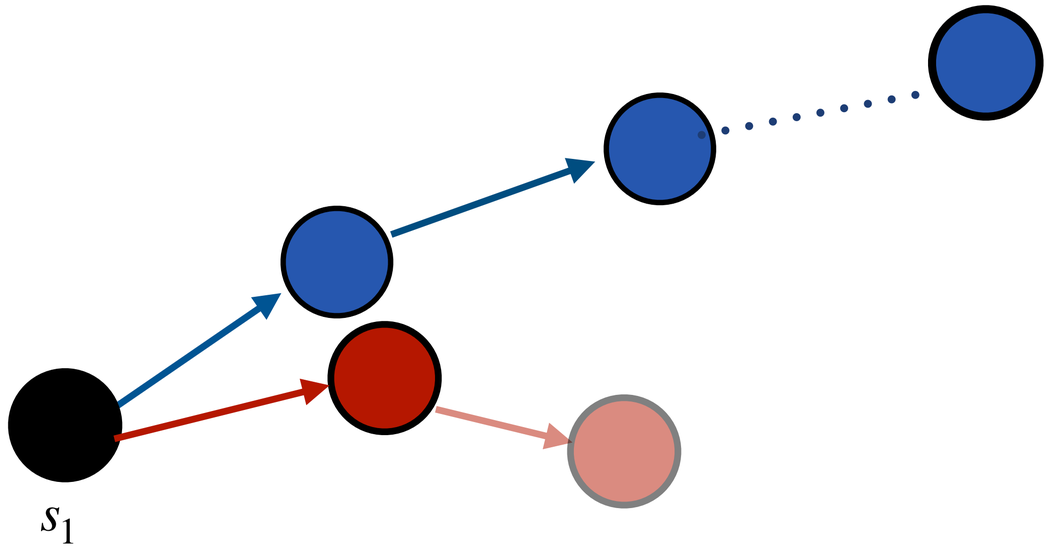
Recoverability

Recoverability

Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

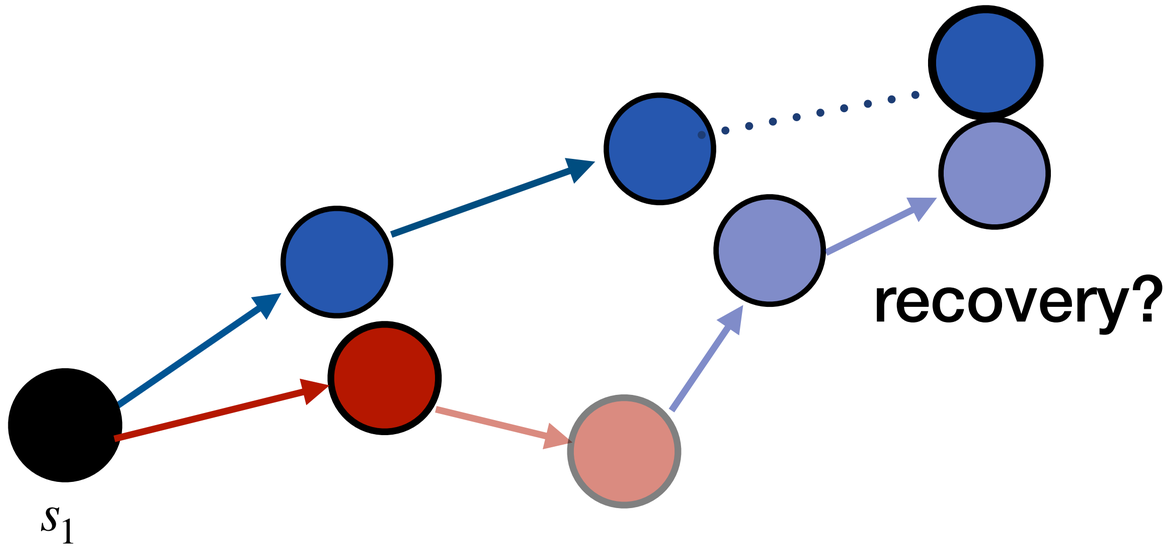
Recoverability

Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$



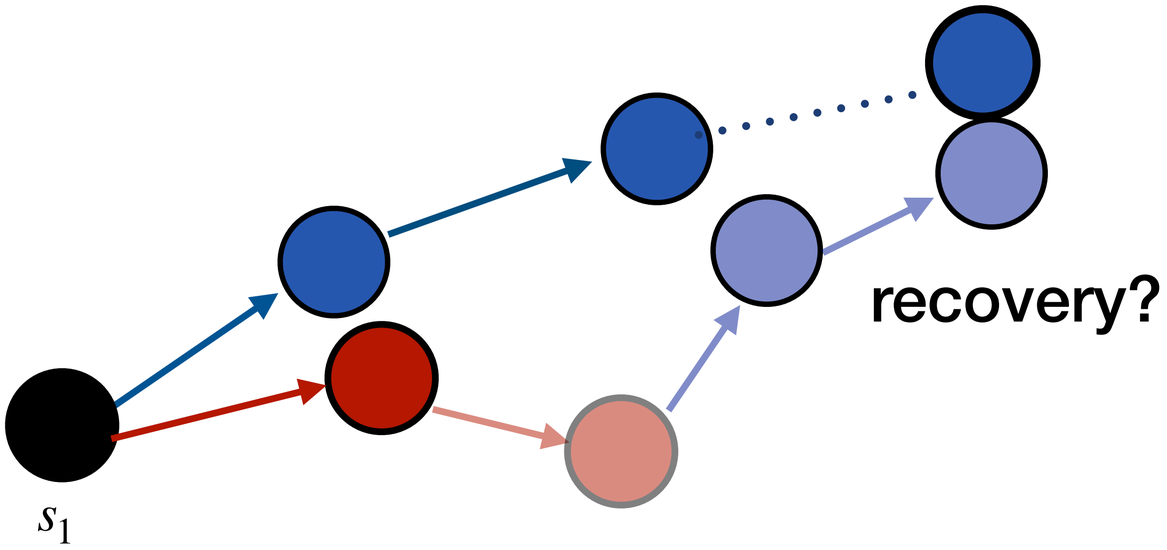
Recoverability

Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$



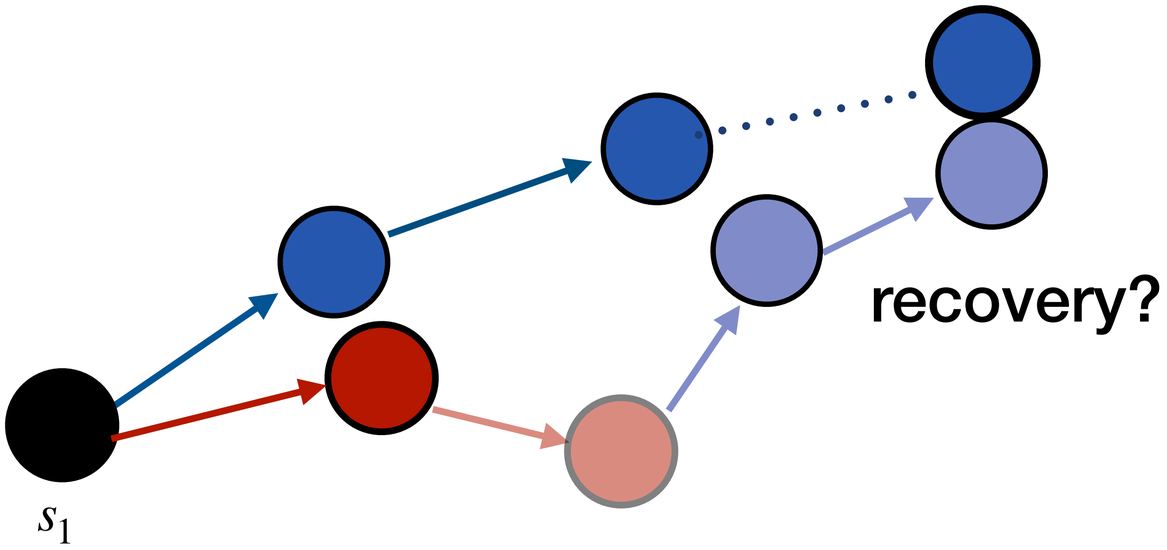
Recoverability

Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$



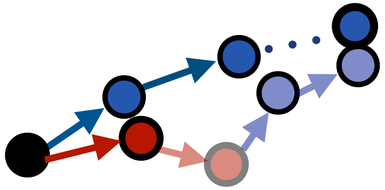
Recoverability

Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

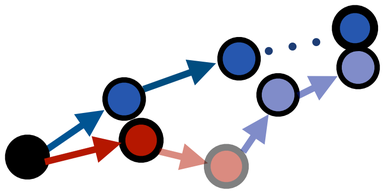


not recoverable :(

Recoverability

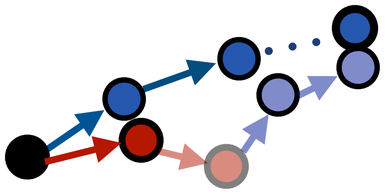


Recoverability



Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

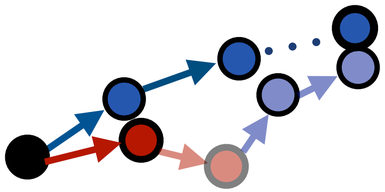
Recoverability



Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

Assume recoverability. Using Reverse Performance Difference

Recoverability

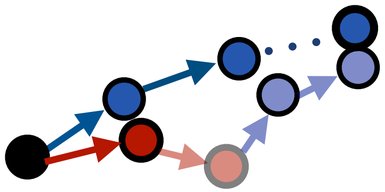


Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

Assume recoverability. Using Reverse Performance Difference

$$J(\pi^\star) - J(\hat{\pi}) = \mathbb{E}^{\hat{\pi}}\left[\sum_{t=1}^{H} Q_t^{\pi^\star}(s_t, a_t) - Q_t^{\pi^\star}(s_t, \hat{a}_t)\right]$$

Recoverability



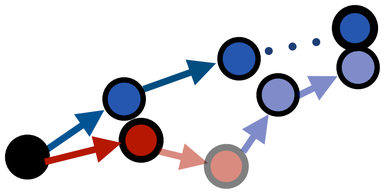
Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

Assume recoverability. Using Reverse Performance Difference

$$J(\pi^\star) - J(\hat{\pi}) = \mathbb{E}^{\hat{\pi}}\left[\sum_{t=1}^{H} Q_t^{\pi^\star}(s_t, a_t) - Q_t^{\pi^\star}(s_t, \hat{a}_t)\right]$$

$$\leq \mathbb{E}^{\hat{\pi}}\left[\sum_{t=1}^{H} \mu \cdot \mathbb{I}(a_t \neq \hat{a}_t)\right]$$

Recoverability



Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

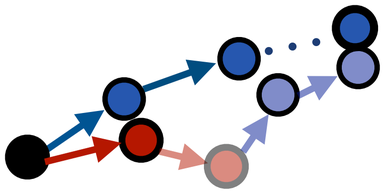
Assume recoverability. Using Reverse Performance Difference

$$J(\pi^\star) - J(\hat{\pi}) = \mathbb{E}^{\hat{\pi}}\left[\sum_{t=1}^{H} Q_t^{\pi^\star}(s_t, a_t) - Q_t^{\pi^\star}(s_t, \hat{a}_t)\right]$$

$$\leq \mathbb{E}^{\hat{\pi}}\left[\sum_{t=1}^{H} \mu \cdot \mathbb{I}(a_t \neq \hat{a}_t)\right]$$

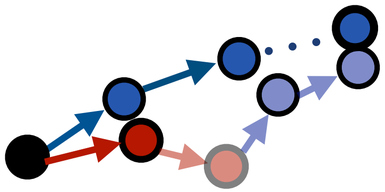
$$= H \cdot \mu \cdot \mathbb{E}^{\hat{\pi}}\left[\frac{1}{H}\sum_{t=1}^{H} \mathbb{I}\{\pi^\star(s_t) \neq \hat{\pi}(s_t)\}\right]$$

Recoverability



Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

Recoverability

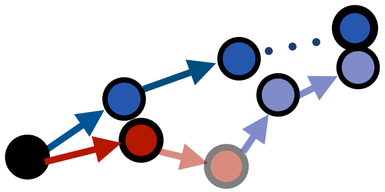


Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

Theorem: Suppose that expert is μ-recoverable. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq H \cdot \mu \cdot \mathbb{E}^{\hat{\pi}}\left[\frac{1}{H}\sum_{t=1}^{H} \mathbb{I}\{\pi^\star(s_t) \neq \hat{\pi}(s_t)\}\right]$$

Recoverability



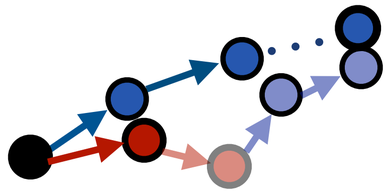
Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

Theorem: Suppose that expert is μ-recoverable. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq H \cdot \mu \cdot \mathbb{E}^{\hat{\pi}}\left[\frac{1}{H}\sum_{t=1}^{H} \mathbb{I}\{\pi^\star(s_t) \neq \hat{\pi}(s_t)\}\right]$$

replace R

Recoverability



Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

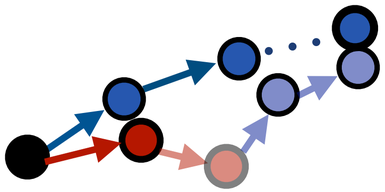
Theorem: Suppose that expert is μ-recoverable. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq H \cdot \mu \cdot \mathbb{E}^{\hat{\pi}}\left[\frac{1}{H}\sum_{t=1}^{H} \mathbb{I}\{\pi^\star(s_t) \neq \hat{\pi}(s_t)\}\right]$$

replace R

loss under $\hat{\pi}$

Recoverability



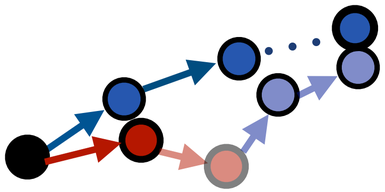
Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

Theorem: Suppose that expert is μ-recoverable. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq H \cdot \mu \cdot \mathbb{E}^{\hat{\pi}}\left[\frac{1}{H}\sum_{t=1}^{H} \mathbb{I}\{\pi^\star(s_t) \neq \hat{\pi}(s_t)\}\right]$$

replace R

Recoverability



Definition: We say $\pi^\star$ is μ-recoverable if $\max_a V^{\pi^\star}(s) - Q^{\pi^\star}(s,a) \leq \mu$

Theorem: Suppose that expert is μ-recoverable. Then, for the 0/1-loss, we have that **any** $\hat{\pi}$ satisfies

$$J(\pi^\star) - J(\hat{\pi}) \leq H \cdot \mu \cdot \mathbb{E}^{\hat{\pi}}\left[\frac{1}{H}\sum_{t=1}^{H} \mathbb{I}\{\pi^\star(s_t) \neq \hat{\pi}(s_t)\}\right]$$

replace R

we do not get this from BC!

Protocol: Expert Interaction

Protocol: Expert Interaction

Algorithm: For rounds $i = 1, \ldots, M$:

Protocol: Expert Interaction

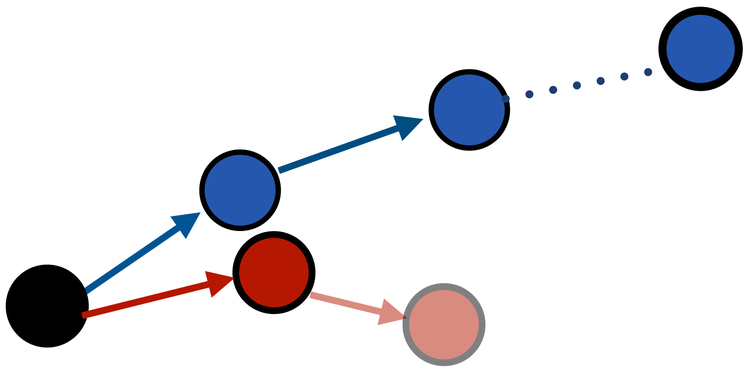
Algorithm: For rounds $i = 1, \ldots, M$:

1. **Collect n trajectories** $(s_1, \ldots, s_H)$**, with policy** $\hat{\pi}_i$

Protocol: Expert Interaction

Algorithm: For rounds $i = 1, \ldots, M$:

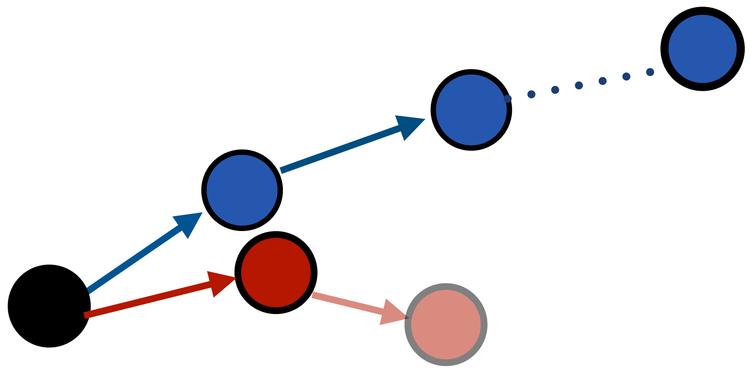
1. **Collect n trajectories** $(s_1, \ldots, s_H)$**, with policy** $\hat{\pi}_i$



Protocol: Expert Interaction

Algorithm: For rounds $i = 1, \ldots, M$:

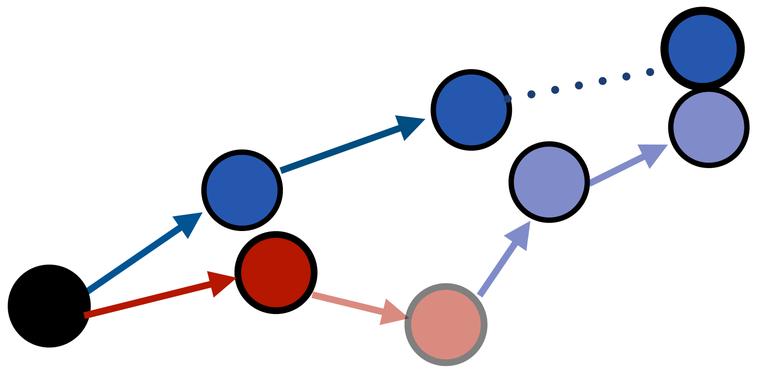
1. **Collect n trajectories** $(s_1, \ldots, s_H)$**, with policy** $\hat{\pi}_i$
2. **Label each trajectory with expert:** $(s_1, \pi^\star(s_1), \ldots, s_H, \pi^\star(s_H))$



Protocol: Expert Interaction

Algorithm: For rounds $i = 1, \ldots, M$:

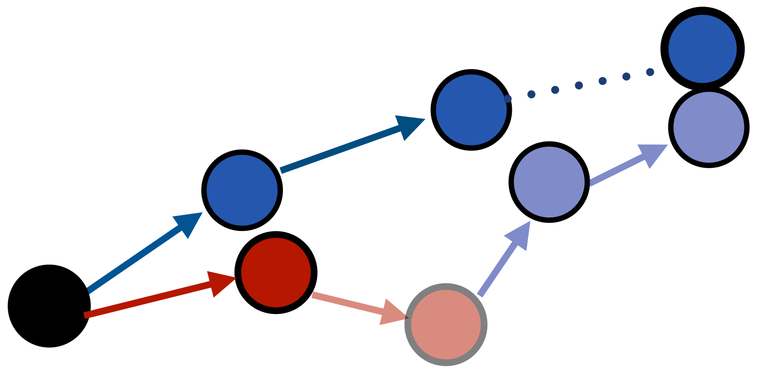
1. **Collect n trajectories** $(s_1, \ldots, s_H)$**, with policy** $\hat{\pi}_i$
2. **Label each trajectory with expert:** $(s_1, \pi^\star(s_1), \ldots, s_H, \pi^\star(s_H))$



Protocol: Expert Interaction

Algorithm: For rounds $i = 1, \ldots, M$:

1. **Collect n trajectories** $(s_1, \ldots, s_H)$**, with policy** $\hat{\pi}_i$
2. **Label each trajectory with expert:** $(s_1, \pi^\star(s_1), \ldots, s_H, \pi^\star(s_H))$
3. **Add trajectories to running dataset** D



DAGGER: Algorithm with Expert Interaction

Algorithm: For rounds $i = 1, \ldots, M$:

1. **Collect n trajectories** $(s_1, \ldots, s_H)$**, with policy** $\hat{\pi}_i$
2. **Label each trajectory with expert:** $(s_1, \pi^\star(s_1), \ldots, s_H, \pi^\star(s_H))$
3. **Add trajectories to running dataset** D
4. **train** $\hat{\pi}^{i+1} = \arg\min_{\pi \in \Pi} \sum_{i \in D} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$

DAGGER: Algorithm with Expert Interaction

Algorithm: For rounds $i = 1, \ldots, M$:

1. **Collect n trajectories** $(s_1, \ldots, s_H)$**, with policy** $\hat{\pi}_i$
2. **Label each trajectory with expert:** $(s_1, \pi^\star(s_1), \ldots, s_H, \pi^\star(s_H))$
3. **Add trajectories to running dataset** D
4. **train** $\hat{\pi}^{i+1} = \arg\min_{\pi \in \Pi} \sum_{i \in D} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$

Key Insight: Adaptive Interaction gives error control under $\hat{\pi}_i$

DAGGER: Algorithm with Expert Interaction

Algorithm: For rounds $i = 1, \ldots, M$:

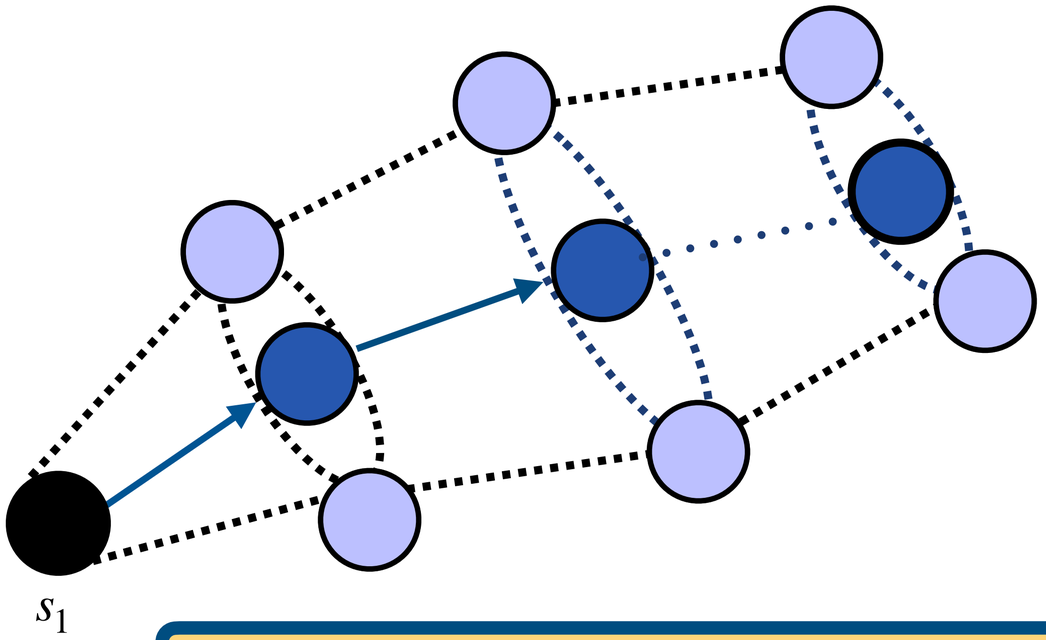
1. **Collect n trajectories** $(s_1, \ldots, s_H)$**, with policy** $\hat{\pi}_i$
2. **Label each trajectory with expert:** $(s_1, \pi^\star(s_1), \ldots, s_H, \pi^\star(s_H))$
3. **Add trajectories to running dataset** D
4. **train** $\hat{\pi}^{i+1} = \arg\min_{\pi \in \Pi} \sum_{i \in D} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$

Theorem: Given $M = \tilde{O}(1)$ rounds of interaction, deterministic policies,

$$J(\pi^\star) - \min_i J(\hat{\pi}_i) \leq H \cdot \mu \cdot \frac{1}{n} \log |\Pi| \qquad \textit{(Ross \& Bagnell, 2011)}$$

DART: Modifying Expert Data-Collection

Also propose an adaptive procedure, but not necessary.



Define exploratory expert $\tilde{\pi}^\star = \pi^\star + \sigma z$

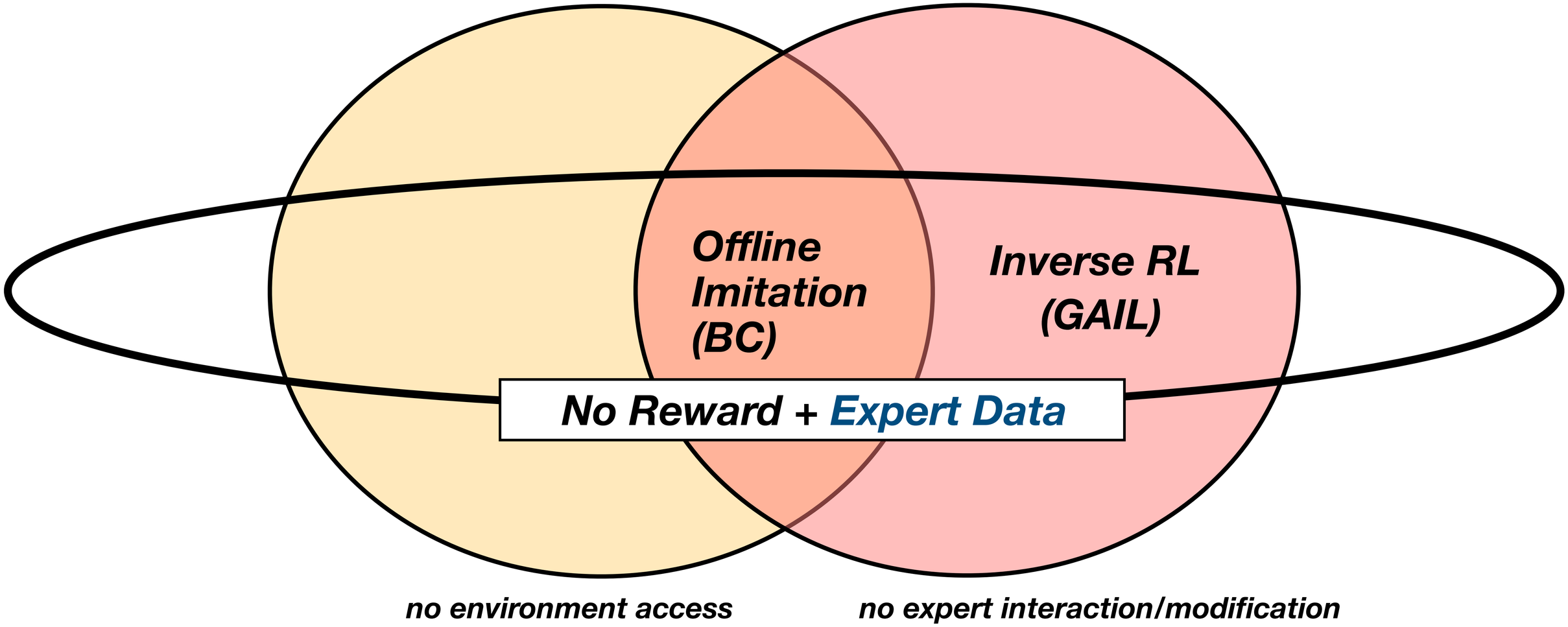
$$L_{\mathrm{bc},\sigma}(\pi) = \mathbb{E}^{\tilde{\pi}^\star}\left[\sum_{t=1}^{H} \|\pi(s_t) - \pi^\star(s_t)\|^2\right]$$

imitate expert policy

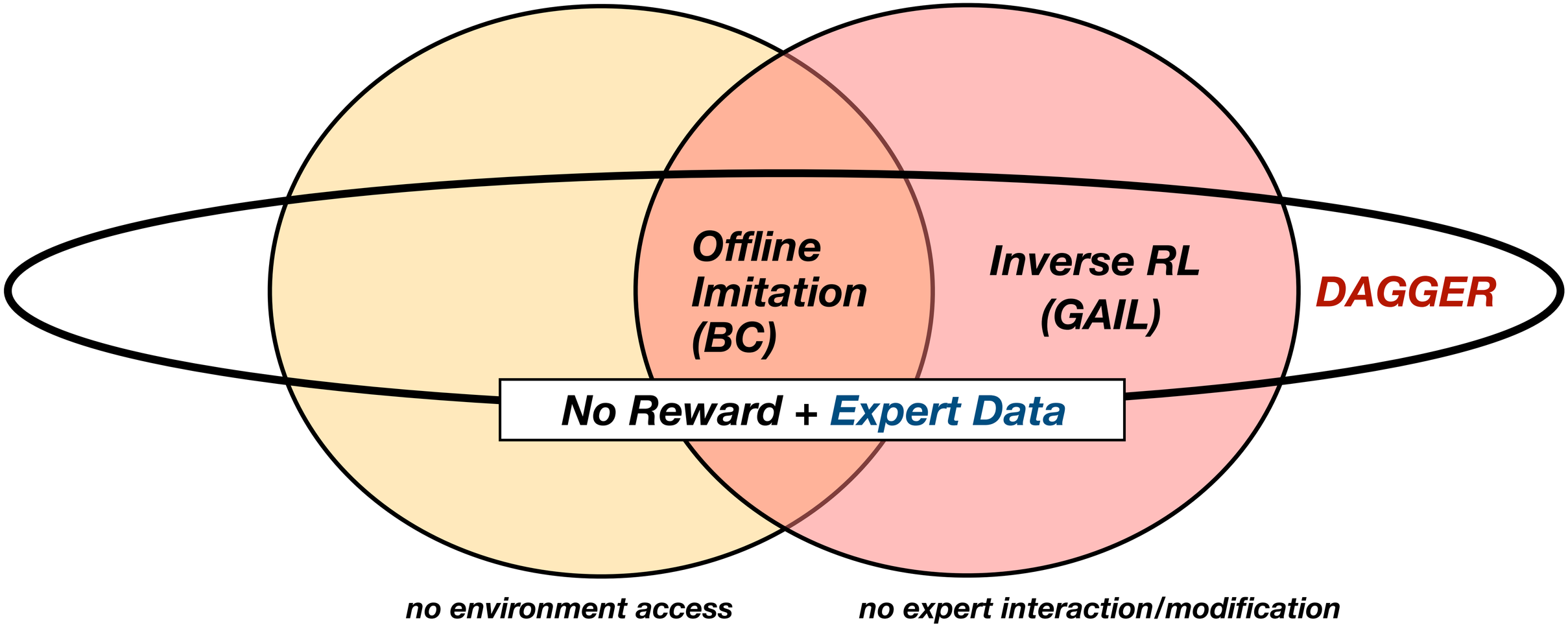
Proposition, Informal (Laskey ’17): Training and executing with Gaussian noise ensured bounded **KL**-divergence between trajectories.

Comparing these results to 0/1 loss is subtle, see part 4.

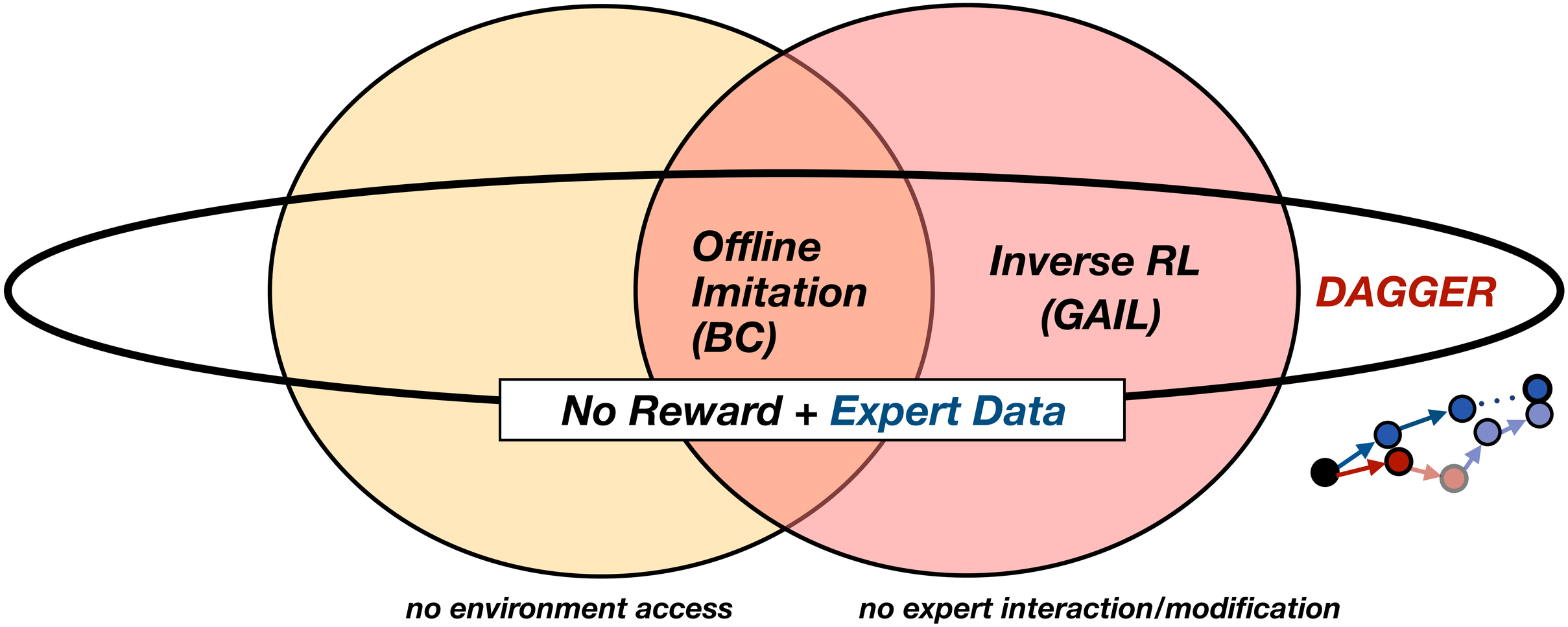
Summary: Expert Modification



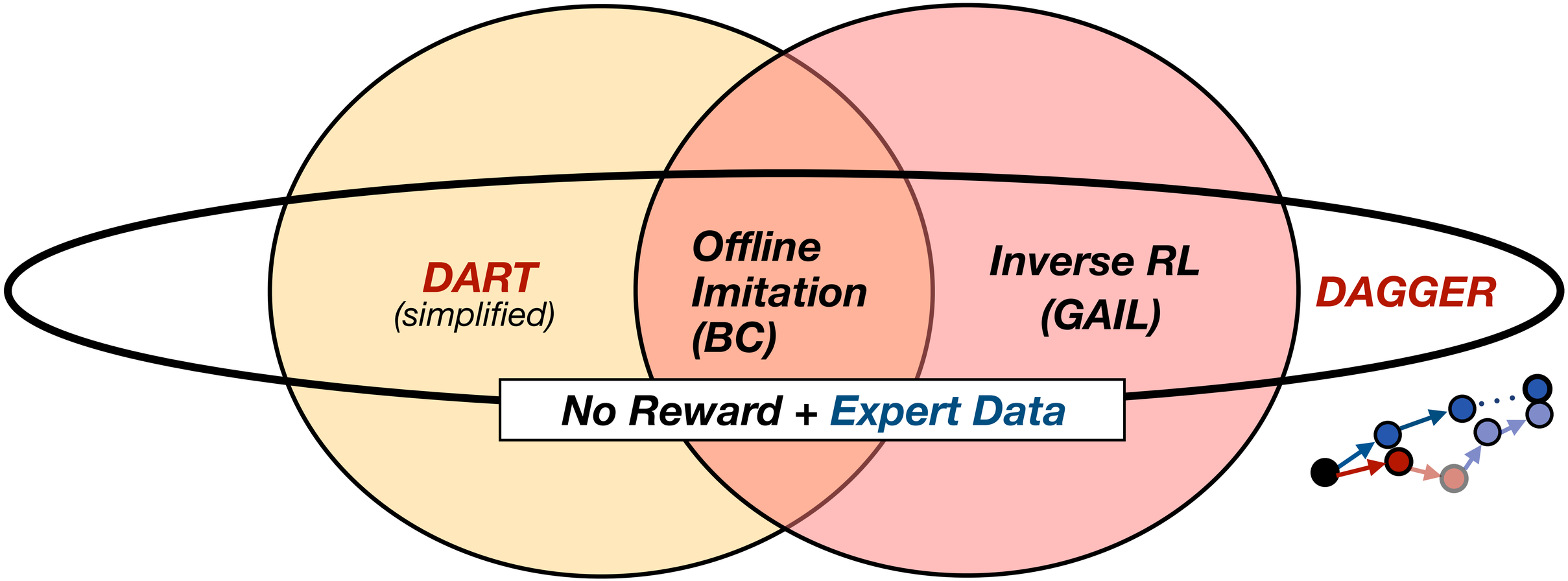
Summary: Expert Modification



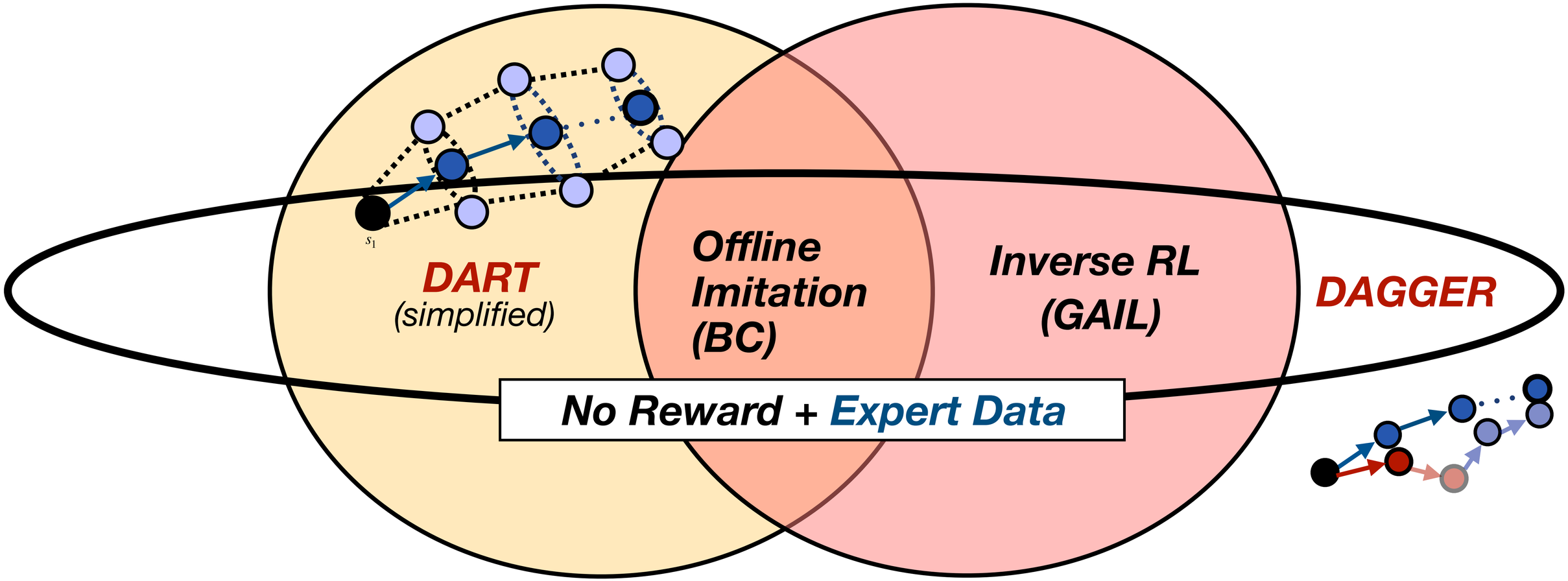
Summary: Expert Modification



Summary: Expert Modification



Summary: Expert Modification



Summary: Expert Interaction/Modification

Summary: Expert Interaction/Modification

1. DART and DAGGER both use either interactions and/or modifying the expert.

Summary: Expert Interaction/Modification

1. DART and DAGGER both use either interactions and/or modifying the expert.

2. These papers claim benefits in the form of upper bounds

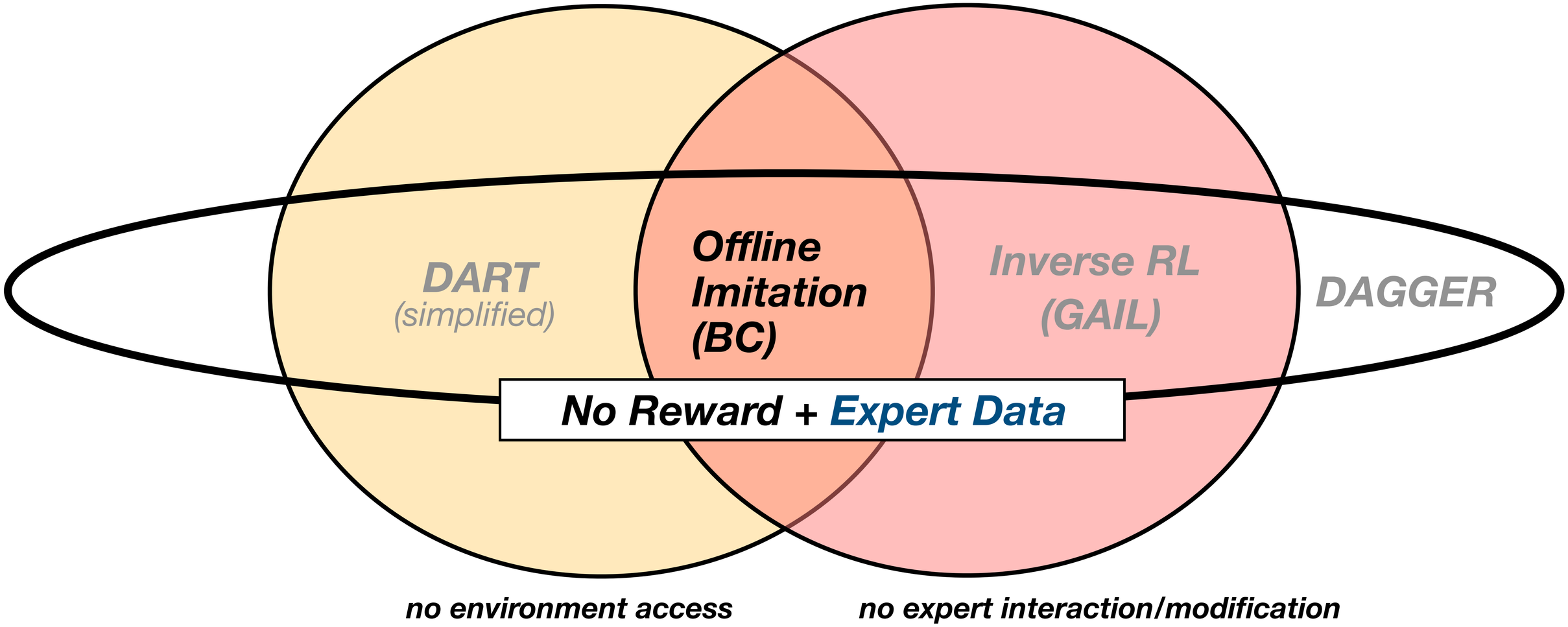
Summary: Expert Interaction/Modification

1. DART and DAGGER both use either interactions and/or modifying the expert.

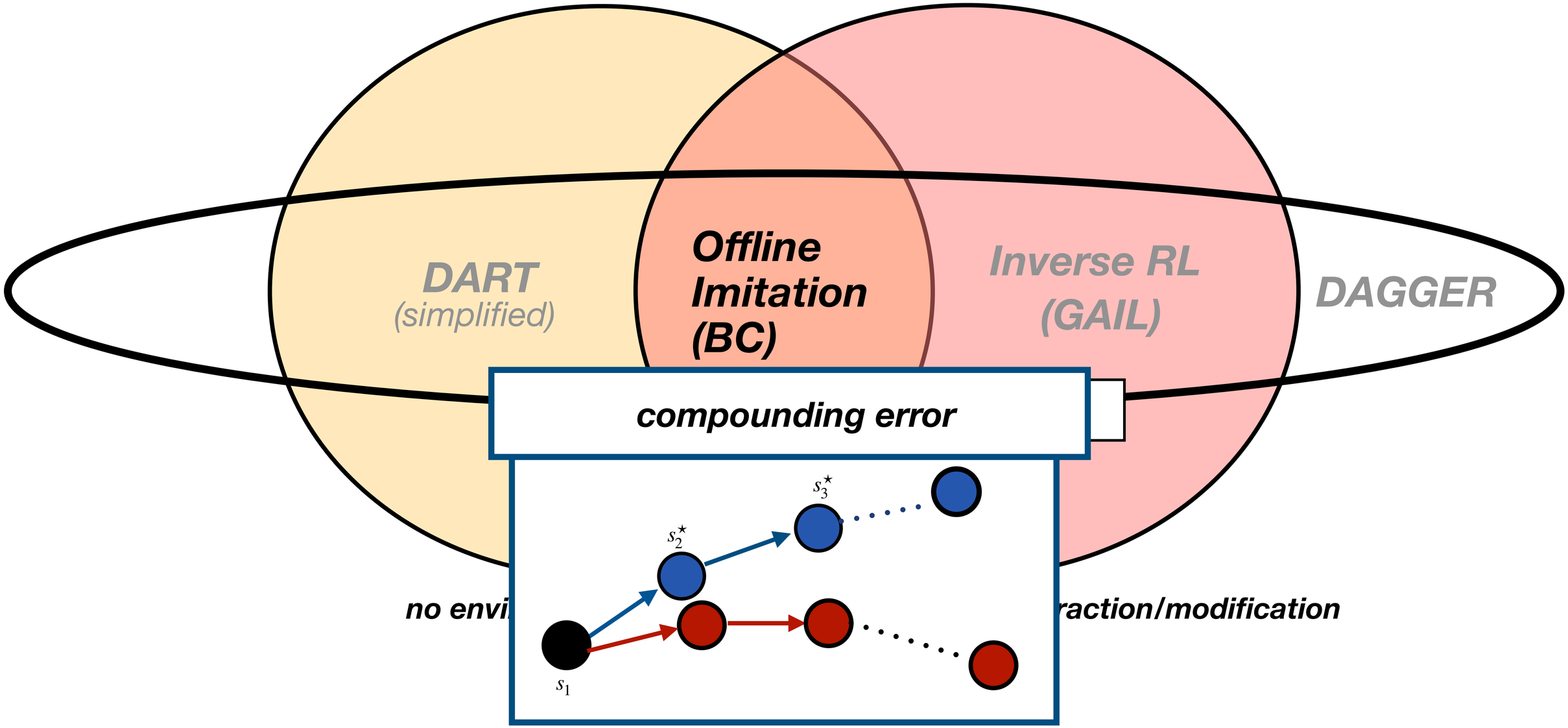
2. These papers claim benefits in the form of upper bounds

3. Stay tuned to understand if these interventions are necessary

Summary: Learning from Expert Demonstration



Summary: Learning from Expert Demonstration



Outline

I. Background and Introduction.

II. Classical Imitation Learning.

III. **Modern IL in discrete settings.**

IV. Modern IL in continuous settings.

V. Opening the black box of optimization.

Plan for remainder of tutorial

Part III. IL in discrete settings

Part IV. IL in continuous settings

Part V. The optimization perspective

Plan for remainder of tutorial

Part III. IL in discrete settings

Part IV. IL in continuous settings

Part V. The optimization perspective

Error Compounding

Discrete — Continuous

Better than we thought…

…or not

Part III

Imitation Learning in Discrete Spaces

Everything you ever wanted to know about the log loss

Dylan Foster

w/ Adam Block and Max Simchowitz

IL in Discrete Settings

Discrete setting: $\pi^\star$**(s) is discrete, or** $\pi^\star(s)$ **has density**

Implicit assumption:

$\pi^\star(a \mid s)$ **is well-defined and can be modeled by** Π **of reasonable capacity.**

IL in Discrete Settings

Discrete setting: $\pi^\star$(s) is discrete, or $\pi^\star(s)$ has density

Implicit assumption:

$\pi^\star(a \mid s)$ is well-defined and can be modeled by Π of reasonable capacity.

Behavior cloning with log-loss:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})}$$

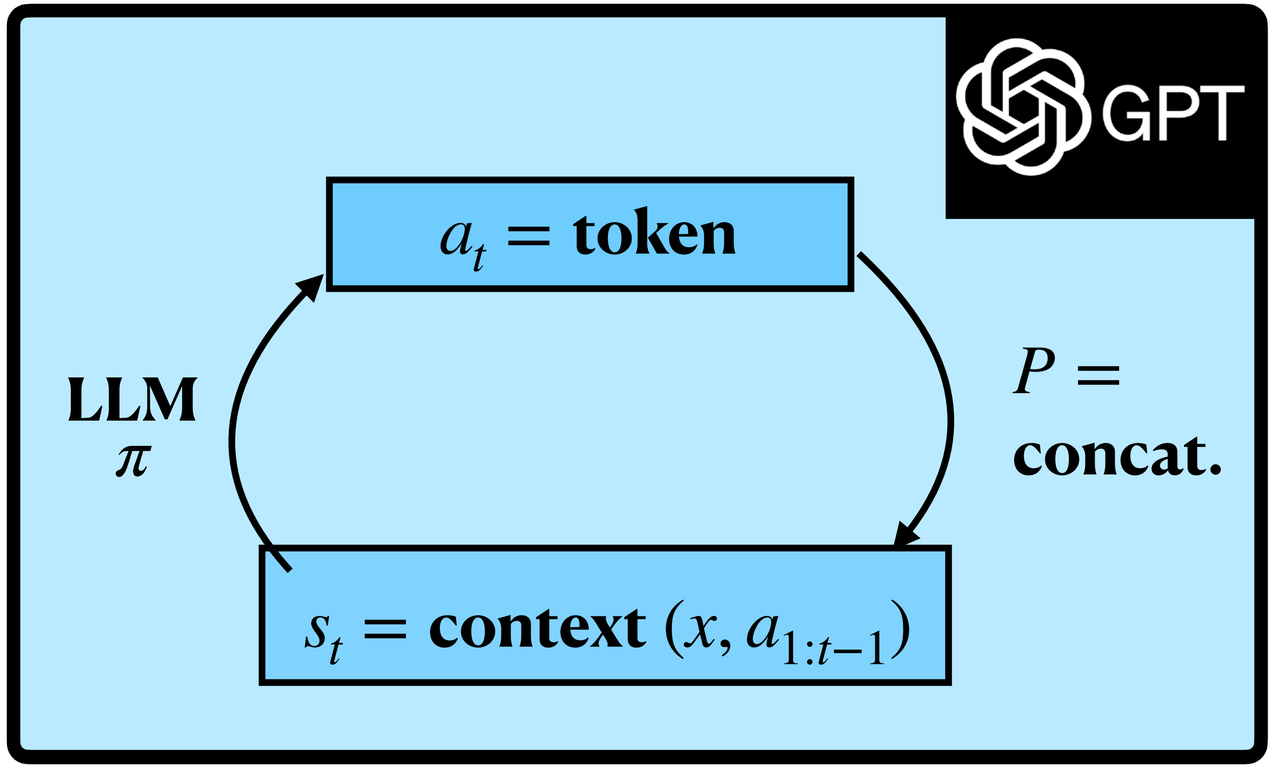
Example: Next-token prediction

Next-token prediction

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})}$$

Next-token prediction is log-loss behavior cloning in token-level MDP!

Expert $\pi^\star$ represents pre-training distribution.



Token-Level MDP

IL in Discrete Settings

Error compounding in LMS: (“exposure bias”)
[Holtzmann et al. ’19], [Braverman et al. ’20],
[Block et al. ’24]

Discrete setting: $\pi^\star$(s) is discrete, or $\pi^\star(s)$ has density

Behavior cloning with log-loss:

$$\hat{\pi} \approx \arg\max_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \pi(a_t^{(i)} \mid s_t^{(i)})$$

Overarching question:

Is error amplification truly a limitation in discrete settings?

Overview of Part III: IL in Discrete Spaces

3.1. Behavior cloning is horizon-independent

(as long as you use the log loss)

Overview of Part III: IL in Discrete Spaces

3.1. Behavior cloning is horizon-independent

(as long as you use the log loss)

3.2. Language models: The coverage principle

(moving beyond the classical IL perspective)

Part III

Imitation Learning in Discrete Spaces

1. Log-Loss Behavior Cloning: Revisiting Horizon

2. Beyond Classical IL: The Coverage Principle

Recap: Behavior Cloning with the {0,1}-loss

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \mathbf{1}\left\{\pi(s_t^{(i)}) \neq a_t^{(i)}\right\}$$

($\pi^\star$, $\hat{\pi}$ are deterministic)

$$L_{\mathrm{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \mathbb{I}\{\pi(s_t) \neq \pi^\star(s_t)\}\right]$$

1. $J(\pi^\star) - J(\hat{\pi}) \leq H \cdot R \cdot L_{\mathrm{bc}}(\hat{\pi})$ (performance difference lemma)

2. $L_{\mathrm{bc}}(\hat{\pi}) \lesssim \frac{1}{n}\log|\Pi|$ whenever $\pi^\star \in \Pi$. (generalization error)

3. Gives $J(\pi^\star) - J(\hat{\pi}) \lesssim \frac{RH\log|\Pi|}{n}$.

Recap: Behavior Cloning with the {0,1}-loss

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \mathbf{1}\left\{\pi(s_t^{(i)}) \neq a_t^{(i)}\right\}$$

($\pi^\star$, $\hat{\pi}$ are deterministic)

$$L_{\mathrm{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \mathbb{I}\{\pi(s_t) \neq \pi^\star(s_t)\}\right]$$

1. $J(\pi^\star) - J(\hat{\pi}) \leq H \cdot R \cdot L_{\mathrm{bc}}(\hat{\pi})$ (performance difference lemma)

2. $L_{\mathrm{bc}}(\hat{\pi}) \lesssim \frac{1}{n} \log|\Pi|$ whenever $\pi^\star \in \Pi$. (generalization error)

3. Gives $J(\pi^\star) - J(\hat{\pi}) \lesssim \frac{RH \log|\Pi|}{n}$.

Recap: Behavior Cloning with the {0,1}-loss

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \mathbf{1}\left\{\pi(s_t^{(i)}) \neq a_t^{(i)}\right\}$$

($\pi^\star$, $\hat{\pi}$ are deterministic)

$$L_{\mathrm{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \mathbb{I}\{\pi(s_t) \neq \pi^\star(s_t)\}\right]$$

1. $J(\pi^\star) - J(\hat{\pi}) \leq H \cdot R \cdot L_{\mathrm{bc}}(\hat{\pi})$ (performance difference lemma)

2. $L_{\mathrm{bc}}(\hat{\pi}) \lesssim \frac{1}{n}\log|\Pi|$ whenever $\pi^\star \in \Pi$. (generalization error)

3. Gives $J(\pi^\star) - J(\hat{\pi}) \lesssim \frac{RH\log|\Pi|}{n}$.

Recap: Behavior Cloning with the {0,1}-loss

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \mathbf{1}\left\{\pi(s_t^{(i)}) \neq a_t^{(i)}\right\}$$

($\pi^\star$, $\hat{\pi}$ are deterministic)

$$L_{\mathrm{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \mathbb{I}\{\pi(s_t) \neq \pi^\star(s_t)\}\right]$$

1. $J(\pi^\star) - J(\hat{\pi}) \leq H \cdot R \cdot L_{\mathrm{bc}}(\hat{\pi})$ (performance difference lemma)

2. $L_{\mathrm{bc}}(\hat{\pi}) \lesssim \frac{1}{n}\log|\Pi|$ whenever $\pi^\star \in \Pi$. (generalization error)

3. Gives $J(\pi^\star) - J(\hat{\pi}) \lesssim \frac{RH\log|\Pi|}{n}$.

Log-Loss BC: Supervised Learning Guarantee

Behavior cloning with log-loss:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})}$$

Define $\mathbb{P}^{\pi}[\,\cdot\,] = \mathbb{P}\left[(s_1, a_1), \ldots, (s_H, a_H) \mid \pi\right]$

Log-Loss BC: Supervised Learning Guarantee

Behavior cloning with log-loss:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})}$$

Define $\mathbb{P}^{\pi}[\,\cdot\,] = \mathbb{P}\left[(s_1, a_1), \ldots, (s_H, a_H) \mid \pi\right]$

Claim: BC is equivalent to maximum likelihood over set $\left\{\mathbb{P}^{\pi}\right\}_{\pi \in \Pi}$

$$\sum_{i=1}^{n} \log \left(\frac{1}{\mathbb{P}^{\pi}\left[s_1^{(i)}, a_1^{(i)}, \ldots, s_H^{(i)}, a_H^{(i)}\right]} \right) = \sum_{i=1}^{n} \sum_{t=1}^{H} \log \left(\frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})} \right) + \sum_{i=1}^{n} \sum_{t=1}^{H} \log \left(\frac{1}{P(s_{t+1}^{(i)} \mid s_t^{(i)}, a_t^{(i)})} \right)$$

MLE Objective

Log-Loss BC: Supervised Learning Guarantee

Behavior cloning with log-loss:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})}$$

Define $\mathbb{P}^{\pi}[\,\cdot\,] = \mathbb{P}\left[(s_1, a_1), \ldots, (s_H, a_H) \mid \pi\right]$

Claim: BC is equivalent to maximum likelihood over set $\left\{\mathbb{P}^{\pi}\right\}_{\pi \in \Pi}$

$$\sum_{i=1}^{n} \log \left(\frac{1}{\mathbb{P}^{\pi}\left[s_1^{(i)}, a_1^{(i)}, \ldots, s_H^{(i)}, a_H^{(i)}\right]} \right) = \sum_{i=1}^{n} \sum_{t=1}^{H} \log \left(\frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})} \right) + \sum_{i=1}^{n} \sum_{t=1}^{H} \log \left(\frac{1}{P(s_{t+1}^{(i)} \mid s_t^{(i)}, a_t^{(i)})} \right)$$

MLE Objective **BC Objective**

Log-Loss BC: Supervised Learning Guarantee

Behavior cloning with log-loss:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})}$$

Define $\mathbb{P}^{\pi}[\,\cdot\,] = \mathbb{P}\left[(s_1, a_1), \ldots, (s_H, a_H) \mid \pi\right]$

Claim: BC is equivalent to maximum likelihood over set $\left\{\mathbb{P}^{\pi}\right\}_{\pi \in \Pi}$

$$\sum_{i=1}^{n} \log \left(\frac{1}{\mathbb{P}^{\pi}\left[s_1^{(i)}, a_1^{(i)}, \ldots, s_H^{(i)}, a_H^{(i)}\right]} \right) = \sum_{i=1}^{n} \sum_{t=1}^{H} \log \left(\frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})} \right) + \sum_{i=1}^{n} \sum_{t=1}^{H} \log \left(\frac{1}{P(s_{t+1}^{(i)} \mid s_t^{(i)}, a_t^{(i)})} \right)$$

MLE Objective — **BC Objective** — **Independent of π**

Log-Loss BC: Supervised Learning Guarantee

Behavior cloning with log-loss:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})}$$

Key Property:

- We minimize a sum of **per-action** losses (BC)…
- Yet get control over a **trajectory-level** objective (MLE) for free!

Claim: BC is equivalent to maximum likelihood over set $\{\mathbb{P}^{\pi}\}_{\pi \in \Pi}$

$$\sum_{i=1}^{n} \log \left(\frac{1}{\mathbb{P}^{\pi}[s_1^{(i)}, a_1^{(i)}, \ldots, s_H^{(i)}, a_H^{(i)}]} \right) = \sum_{i=1}^{n} \sum_{t=1}^{H} \log \left(\frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})} \right) + \sum_{i=1}^{n} \sum_{t=1}^{H} \log \left(\frac{1}{P(s_{t+1}^{(i)} \mid s_t^{(i)}, a_t^{(i)})} \right)$$

MLE Objective — BC Objective — Independent of π

Log-Loss BC: Supervised Learning Guarantee

Behavior cloning with log-loss:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})}$$

Hellinger distance: $D_H^2(P, Q) := \sum_z \left(\sqrt{P(z)} - \sqrt{Q(z)} \right)^2$

Note: $D_{\mathrm{TV}}^2(P, Q) \lesssim D_H^2(P, Q) \lesssim D_{\mathrm{TV}}(P, Q)$

(at worst quadratic gap to TV)

Log-Loss BC: Supervised Learning Guarantee

Behavior cloning with log-loss:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})}$$

Hellinger distance: $D_H^2(P, Q) := \sum_z \left(\sqrt{P(z)} - \sqrt{Q(z)} \right)^2$

Proposition: If $\pi^\star \in \Pi$, then with probability at least $1 - \delta$, Log-Loss BC has:

$$D_H^2(\mathbb{P}^{\hat{\pi}}, \mathbb{P}^{\pi^\star}) \leq \frac{2 \log(|\Pi| \delta^{-1})}{n}$$

Idea: Don't use Hoeffding/Bernstein; MGF of log-loss is exactly Hellinger/Renyi divergence [Wong & Shen '95, van de Geer '00, Zhang '06].

Log-Loss BC: Supervised Learning Guarantee

Behavior cloning with log-loss:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})}$$

Hellinger distance: $D_H^2(P, Q) := \sum_z \left(\sqrt{P(z)} - \sqrt{Q(z)} \right)^2$

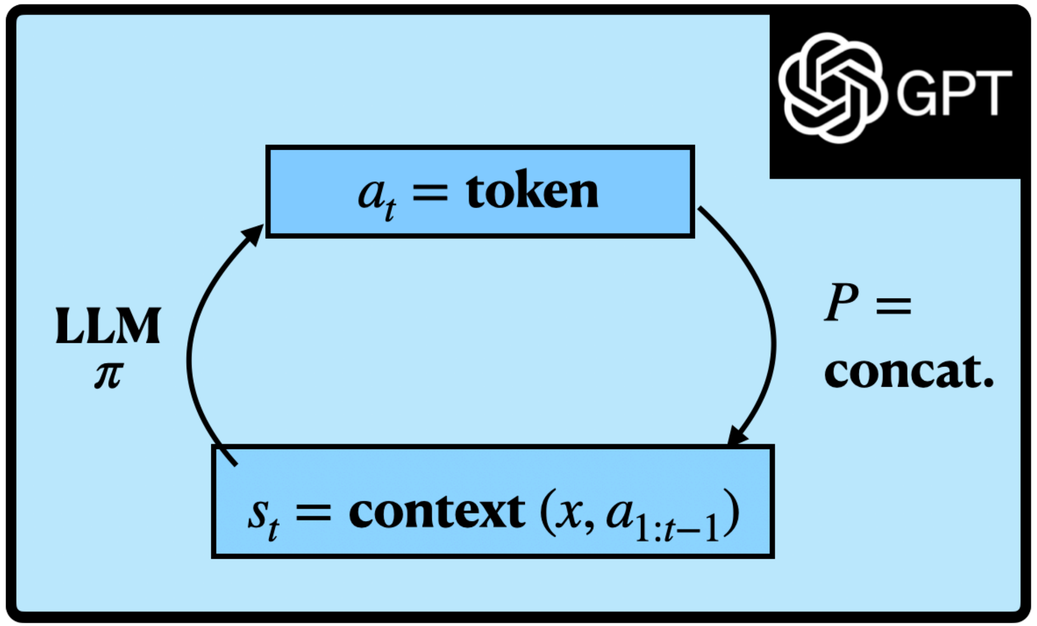
Proposition: If $\pi^\star \in \Pi$, then with probability at least $1 - \delta$, Log-Loss BC has:

$$D_H^2(\mathbb{P}^{\hat{\pi}}, \mathbb{P}^{\pi^\star}) \leq \frac{2 \log(|\Pi| \delta^{-1})}{n}$$

Idea: Don't use Hoeffding/Bernstein; MGF of log-loss is exactly Hellinger/Renyi divergence [Wong & Shen '95, van de Geer '00, Zhang '06].

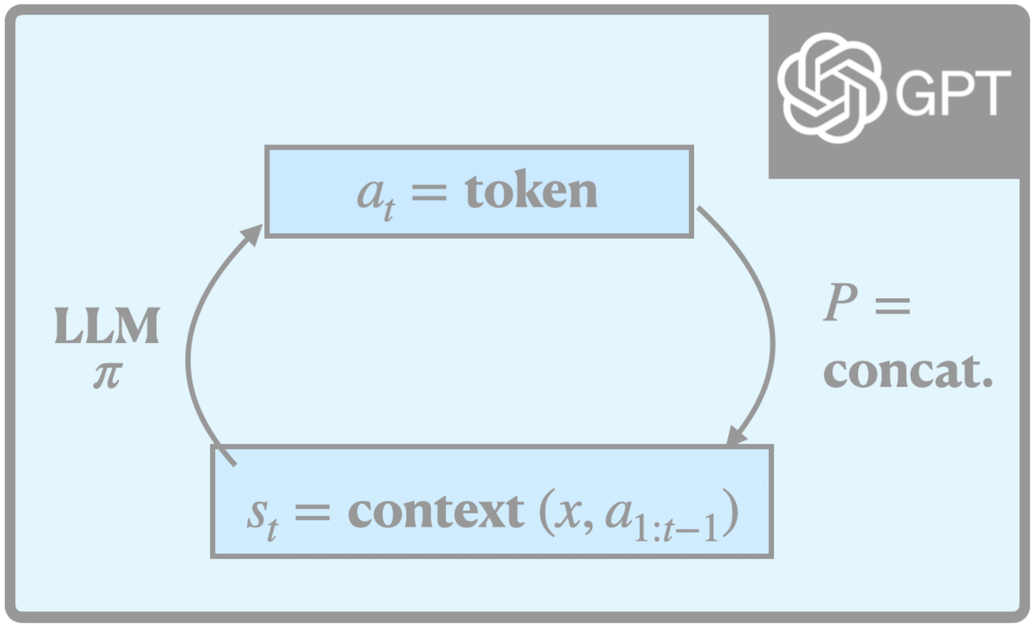
Holds irrespective of whether $\pi^\star$ is deterministic or stochastic.

Deterministic $\pi^\star$



Stochastic $\pi^\star$

Deterministic $\pi^\star$



Stochastic $\pi^\star$

Horizon-Independent Regret (Deterministic Case)

Is behavior cloning all you need? Understanding horizon in imitation learning (F, Block, Misra 2025)

Horizon-Independent Regret (Deterministic Case)

Theorem [F-Block-Misra '24]: For any $\pi^\star$ (deterministic) and imitator $\hat{\pi}$:

$$J(\pi^\star) - J(\hat{\pi}) \leq 4R \cdot D_H^2(\mathbb{P}^{\hat{\pi}}, \mathbb{P}^{\pi^\star})$$

Same spirit as performance difference lemma, but without explicit dependence on H.

Horizon-Independent Regret (Deterministic Case)

Theorem [F-Block-Misra '24]: For any $\pi^\star$ (deterministic) and imitator $\hat{\pi}$:

$$J(\pi^\star) - J(\hat{\pi}) \leq 4R \cdot D_H^2(\mathbb{P}^{\hat{\pi}}, \mathbb{P}^{\pi^\star})$$

Same spirit as performance difference lemma, but without explicit dependence on H.

Corollary [F-Block-Misra '24]: If $\pi^\star \in \Pi$, w/ probability at least $1 - \delta$, Log-Loss BC has:

$$J(\pi^\star) - J(\hat{\pi}) \leq \frac{8R \log(|\Pi| \delta^{-1})}{n}$$

Horizon-Independent Regret (Deterministic Case)

Corollary [F-Block-Misra '24]: If $\pi^\star \in \Pi$, w/ probability at least $1-\delta$, Log-Loss BC has:

$$J(\pi^\star) - J(\hat{\pi}) \leq \frac{8R\log(|\Pi|\delta^{-1})}{n}$$

Features:

- Removes H factor from bound we saw for indicator loss BC (R vs. RH).
- Also improves bound Max showed for Dagger (R vs. μH).
- Generalizes [Rajaraman et al. '20, '21] (tabular/linear policies).

Example: Stationary Linear Policies

Corollary [F-Block-Misra '24]: With probability at least $1-\delta$, Log-Loss BC has:

$$J(\pi^\star) - J(\hat{\pi}) \leq \frac{8R\log(|\Pi|\delta^{-1})}{n}$$

Linear policy with parameter sharing: $\pi_\theta(s_t) = \text{argmax}_{a\in\mathcal{A}}\theta^\intercal\phi(s_t,a)$, **where** $\theta, \phi \in \mathbb{R}^d$

[Rajaraman et al. '21]

Example: Stationary Linear Policies

Corollary [F-Block-Misra '24]: With probability at least $1-\delta$, Log-Loss BC has:

$$J(\pi^\star) - J(\hat{\pi}) \leq \frac{8R\log(|\Pi|\delta^{-1})}{n}$$

Linear policy with parameter sharing: $\pi_\theta(s_t) = \text{argmax}_{a\in\mathcal{A}}\theta^\intercal\phi(s_t,a)$, **where** $\theta,\phi\in\mathbb{R}^d$

[Rajaraman et al. '21]

Has $\log|\Pi| \approx \mathcal{O}(d) \Longrightarrow$ **Log-Loss BC gets** $J(\pi^\star) - J(\hat{\pi}) \lesssim Rd/n$.

→ **Beats the curse of horizon!**

Quick Remarks

Theorem [F-Block-Misra '24]: For any $\pi^\star$ (deterministic) and imitator $\hat{\pi}$:

$$J(\pi^\star) - J(\hat{\pi}) \leq 4R \cdot D_H^2(\mathbb{P}^{\hat{\pi}}, \mathbb{P}^{\pi^\star})$$

Corollary [F-Block-Misra '24]: If $\pi^\star \in \Pi$, w/ probability at least $1 - \delta$, Log-Loss BC has:

$$J(\pi^\star) - J(\hat{\pi}) \leq \frac{8R \log(|\Pi| \delta^{-1})}{n}$$

Quick Remarks

Theorem [F-Block-Misra '24]: For any $\pi^\star$ (deterministic) and imitator $\hat{\pi}$:

$$J(\pi^\star) - J(\hat{\pi}) \leq 4R \cdot D_H^2(\mathbb{P}^{\hat{\pi}}, \mathbb{P}^{\pi^\star})$$

Squared Hellinger is key; trivial to get similar result w/ TV, but gives wrong H dependence.

Corollary [F-Block-Misra '24]: If $\pi^\star \in \Pi$, w/ probability at least $1 - \delta$, Log-Loss BC has:

$$J(\pi^\star) - J(\hat{\pi}) \leq \frac{8R \log(|\Pi| \delta^{-1})}{n}$$

Quick Remarks

Theorem [F-Block-Misra '24]: For any $\pi^\star$ (deterministic) and imitator $\hat{\pi}$:

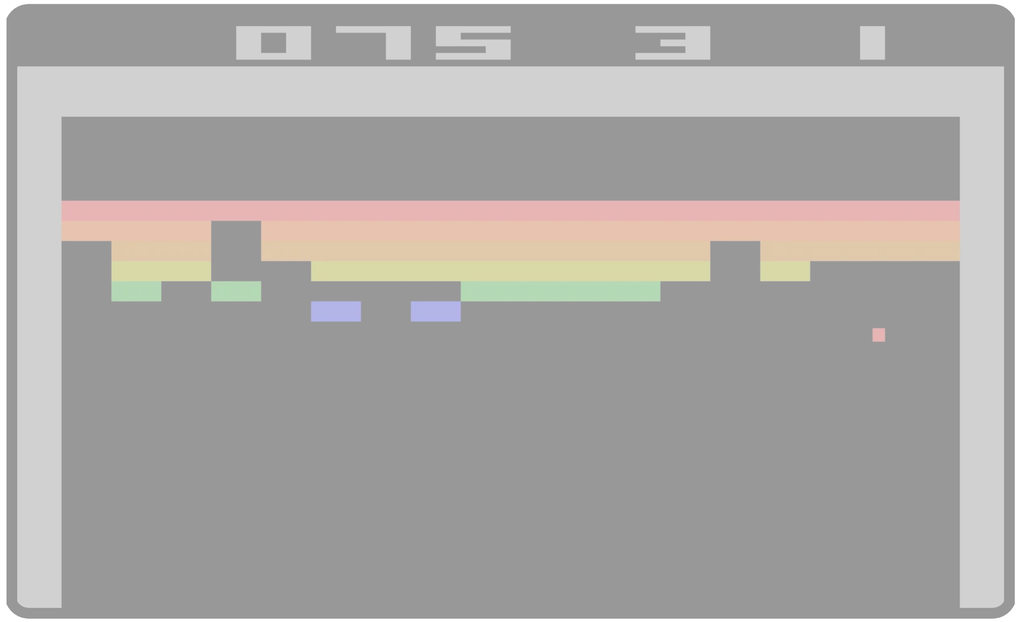
$$J(\pi^\star) - J(\hat{\pi}) \leq 4R \cdot D_H^2(\mathbb{P}^{\hat{\pi}}, \mathbb{P}^{\pi^\star})$$

Squared Hellinger is key; trivial to get similar result w/ TV, but gives wrong H dependence.

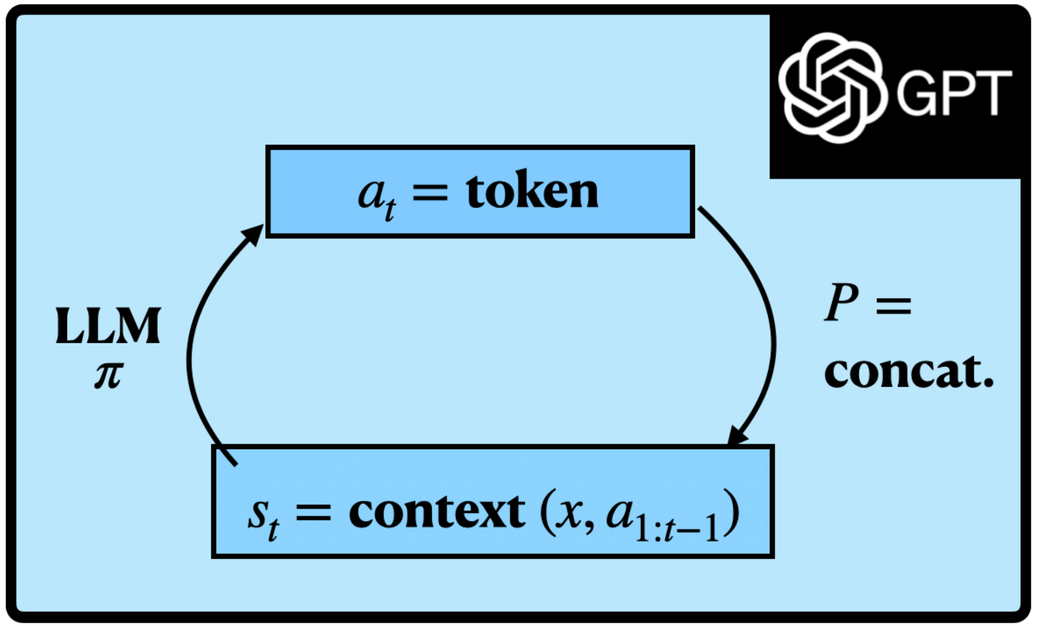
Corollary [F-Block-Misra '24]: If $\pi^\star \in \Pi$, w/ probability at least $1 - \delta$, Log-Loss BC has:

$$J(\pi^\star) - J(\hat{\pi}) \leq \frac{8R \log(|\Pi| \delta^{-1})}{n}$$

$R \log |\Pi| / n$ rate is un-improvable, even with interactive access and recoverability [FBM'24].



Deterministic $\pi^\star$



Stochastic $\pi^\star$

Horizon-Independent Regret (Stochastic Case)

Variance for the expert policy:

$$\sigma^2_{\pi^\star} := \sum_{t=1}^{H} \mathbb{E}^{\pi^\star}\left[\left(Q^{\pi^\star}(s_t, a_t) - V^{\pi^\star}(s_t)\right)^2\right]$$

Is behavior cloning all you need? Understanding horizon in imitation learning (F, Block, Misra 2025)

Horizon-Independent Regret (Stochastic Case)

Variance for the expert policy:

$$\sigma^2_{\pi^\star} := \sum_{t=1}^{H} \mathbb{E}^{\pi^\star}\left[\left(Q^{\pi^\star}(s_t, a_t) - V^{\pi^\star}(s_t)\right)^2\right]$$

Theorem [FBM '24]: If $\pi^\star \in \Pi$, w/ probability at least $1-\delta$, Log-Loss BC has:

$$J(\pi^\star) - J(\hat{\pi}) \lesssim \sqrt{\frac{\sigma^2_{\pi^\star} \cdot \log(|\Pi|\delta^{-1})}{n}} + \frac{R\log(|\Pi|\delta^{-1})}{n}$$

Is behavior cloning all you need? Understanding horizon in imitation learning (F, Block, Misra 2025)

Horizon-Independent Regret (Stochastic Case)

Variance for the expert policy:

$$\sigma^2_{\pi^\star} := \sum_{t=1}^{H} \mathbb{E}^{\pi^\star}\left[\left(Q^{\pi^\star}(s_t, a_t) - V^{\pi^\star}(s_t)\right)^2\right]$$

Theorem [FBM '24]: If $\pi^\star \in \Pi$, w/ probability at least $1-\delta$, Log-Loss BC has:

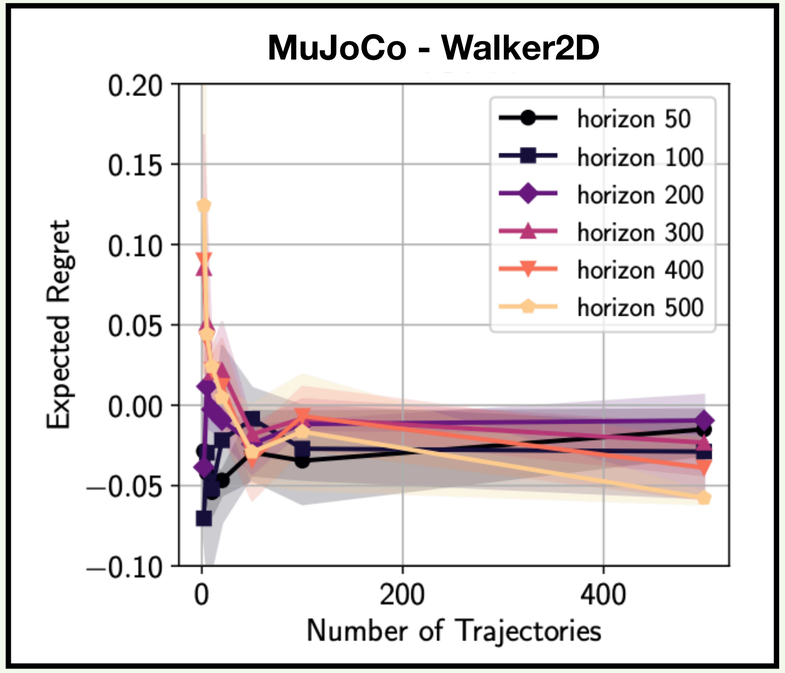
$$J(\pi^\star) - J(\hat{\pi}) \lesssim \sqrt{\frac{\sigma^2_{\pi^\star} \cdot \log(|\Pi|\delta^{-1})}{n}} + \frac{R\log(|\Pi|\delta^{-1})}{n}$$

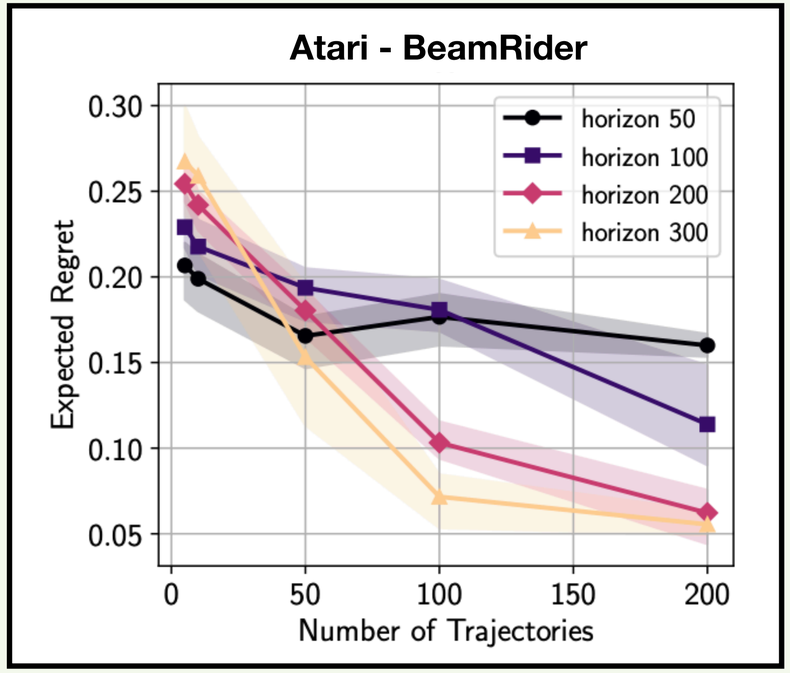
Still horizon-independent (can show $R \leq \sigma_{\pi^\star}$), but slower rate; this is fundamental [FBM'24].

Does this bear out empirically?

Experiments: Control

- Train stationary expert $\pi^\star$ with RL, train imitator $\hat{\pi}$ with Log Loss BC (same architecture).
- Repeat process for varying values of H, keeping env fixed and normalizing $R = \mathcal{O}(1)$.



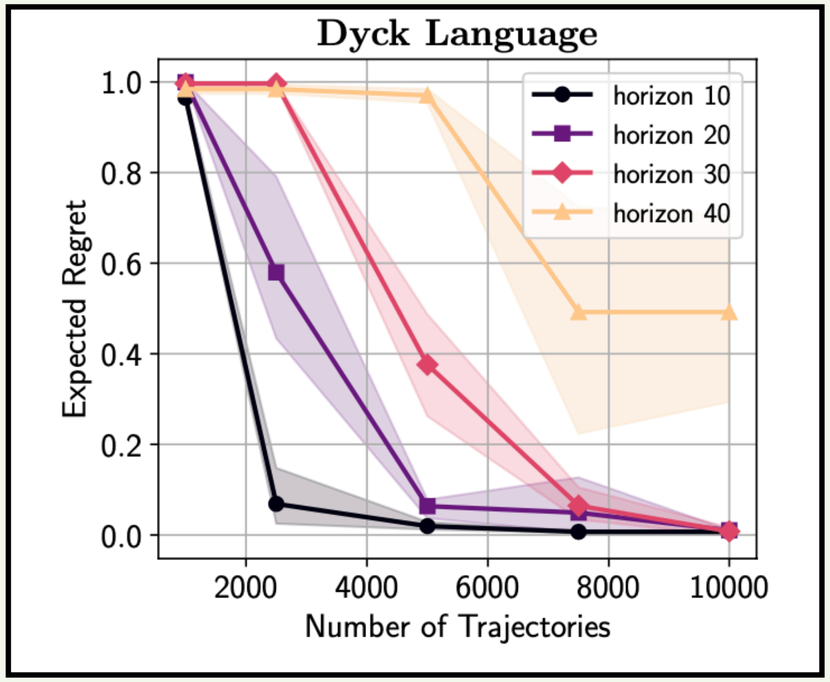


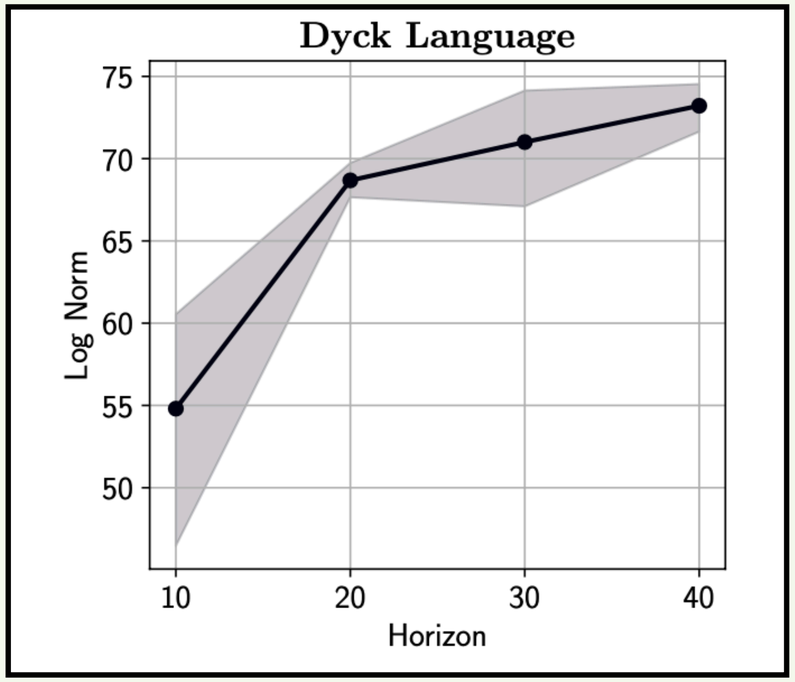
Learning curves are constant or improving with horizon H.

Experiments: Autoregressive Language

$\hat{\pi}$ is a transformed-based language model trained to complete Dyck language strings.

Ex: Complete "([(" to "([()]){}".





Regret grows with H—increased model capacity $\log|\Pi|$ needed to represent $\pi^\star$.

Summary: Part 3.1

See also:

- [Rajaraman et al. 2020]: Tight rates for tabular policies.
- [Joshi et al. 2025]: Implications for chain-of-thought learning.

- **For discrete settings, log-loss behavior cloning is horizon-independent.**

  $\rightarrow$ Error compounding/amplification is benign.

  $\rightarrow$ Seems to close gap between offline IL and online/interactive IL?

- **Relied on the assumption that $\pi^\star \in \Pi$.**

Summary: Part 3.1

See also:

- [Rajaraman et al. 2020]: Tight rates for tabular policies.
- [Joshi et al. 2025]: Implications for chain-of-thought learning.

- **For discrete settings, log-loss behavior cloning is horizon-independent.**

  → Error compounding/amplification is benign.

  → Seems to close gap between offline IL and online/interactive IL?

- **Relied on the assumption that $\pi^\star \in \Pi$.**

What if Π has insufficient model capacity to represent $\pi^\star$?

($\pi^\star \notin \Pi$)

Aside: Insufficient Model Capacity

Imitation learning under insufficient model capacity [RHBKF ’25]

- Log-Loss BC *can* suffer error compounding when $\pi^\star \notin \Pi$.

  Echoes empirical findings [Muller ’05, Bagnell ’15, De Haan ’19, …].

- In pure offline IL, no computationally efficient method can improve.

Ex: [Espinosa-Dice et al. ’25]

[Song et al. ’25]

Aside: Insufficient Model Capacity

Imitation learning under insufficient model capacity [RHBKF ’25]

- Log-Loss BC *can* suffer error compounding when $\pi^\star \notin \Pi$.

  Echoes empirical findings [Muller ’05, Bagnell ’15, De Haan ’19, …].

- In pure offline IL, no computationally efficient method can improve.

Interesting questions:

- Hardness result is for offline IL; can online interaction (Dagger, IRL) bypass?
- More fine-grained (policy class or reward specific) guarantees?

  Ex: [Espinosa-Dice et al. ’25]

  [Song et al. ’25]

Part III

Imitation Learning in Discrete Spaces

1. Log-Loss Behavior Cloning: Revisiting Horizon

2. Beyond Classical IL: The Coverage Principle

Recap: Next-token prediction as behavior cloning

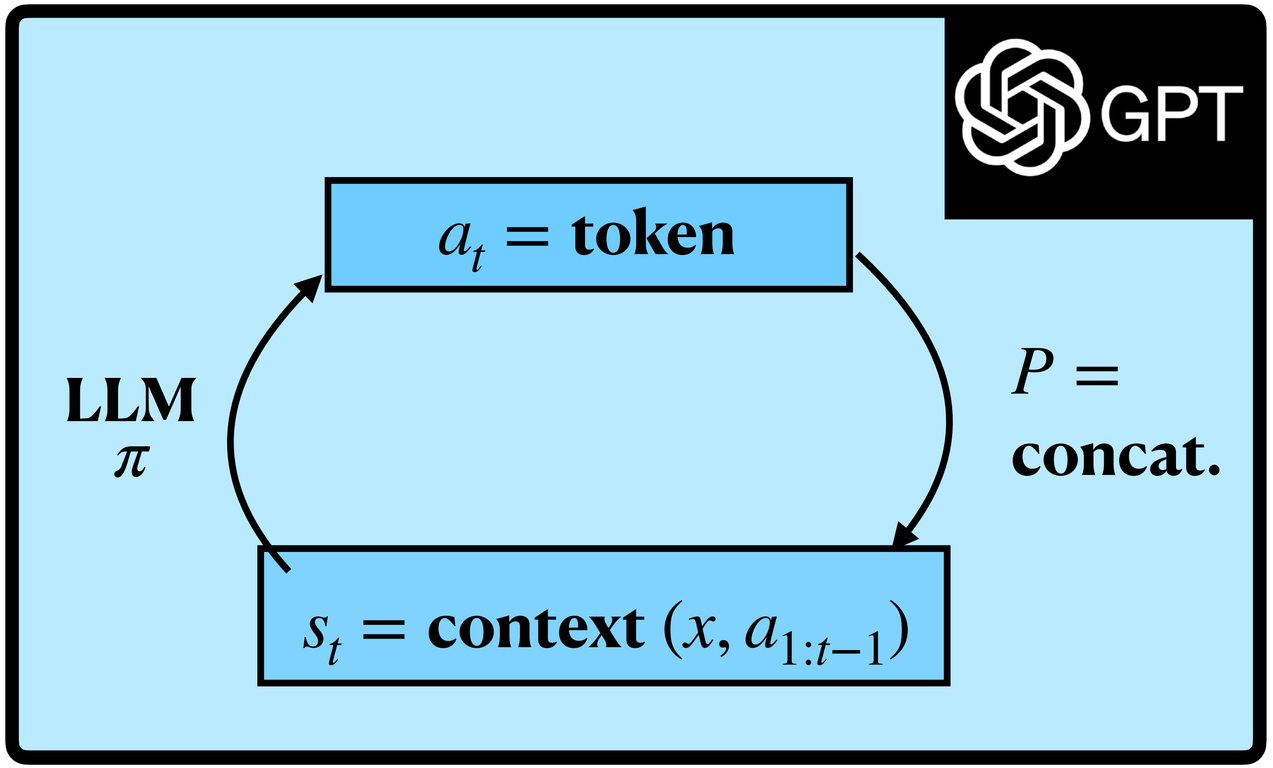
Next-token prediction for LM pre-training:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})}$$

Next-token prediction is log-loss behavior cloning in token-level MDP.

Expert $\pi^\star$ represents pre-training distribution.

Token-Level MDP



Language model training pipeline

Pre-Training

(Next-Token Prediction)

The quick brown fox

Language model training pipeline

Pre-Training

(Next-Token Prediction)

The quick brown fox

Post-Training

(Reinforcement Learning)

What is the solution to $\sqrt{1-\sqrt{1+x}} = x$?

$x = 0$

$x = 1$

Language model training pipeline

Pre-Training
(Next-Token Prediction)

The quick brown fox

Post-Training
(Reinforcement Learning)

What is the solution to $\sqrt{1-\sqrt{1+x}} = x$?

$x = 0$

$x = 1$

Inference-Time

- Chain of Thought
- Multi-shot prompting
- Model-as-Verifier
- Best-of-N

Language model training pipeline

Pre-Training
(Next-Token Prediction)

The quick brown fox

Post-Training
(Reinforcement Learning)

What is the solution to $\sqrt{1-\sqrt{1+x}} = x$?

$x = 0$

$x = 1$

Inference-Time

- Chain of Thought
- Multi-shot prompting
- Model-as-Verifier
- Best-of-N

log-loss behavior cloning

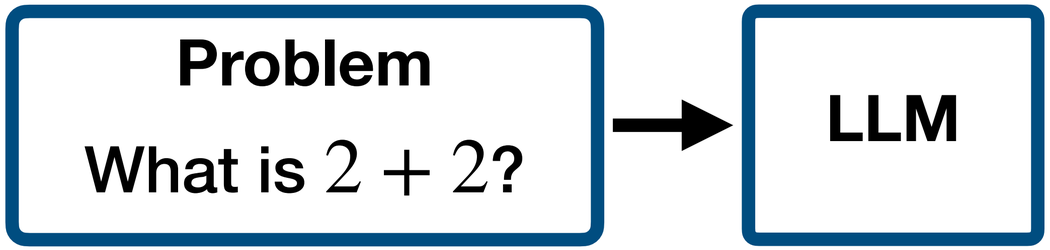
When and why does pre-training give us a model for which post-training and/or inference-time succeeds?

Beyond Classical IL: The Coverage Principle

Part I: Coverage governs downstream performance

Part II: Pre-training implicitly provides coverage

Best-of-N Sampling



Best-of-N Algorithm:

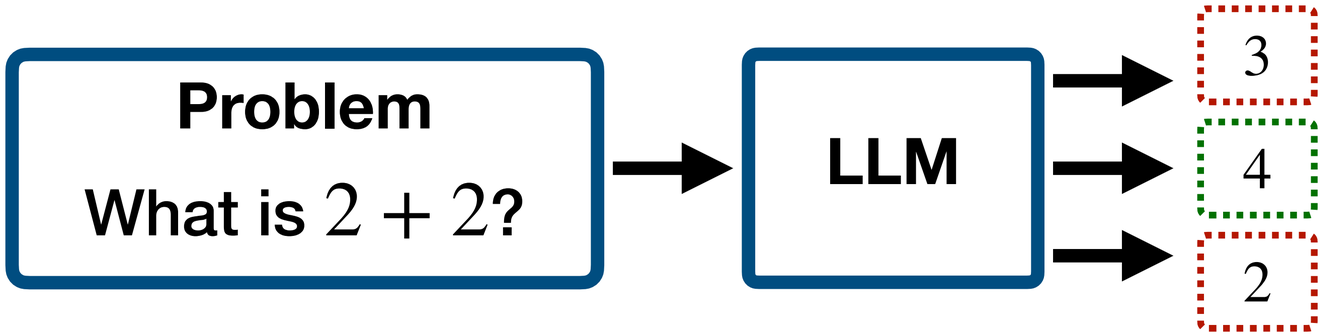
- Sample $y^1, \ldots, y^N \sim \pi(\cdot \mid x)$
- Return $y^i = \text{argmax}_i \, r(x, y^i)$

$y = a_{1:H}$ — response/trajectory

(full token sequence model generates when prompted)

$r(x, y)$ — task-specific reward/verifier

Best-of-N Sampling



Best-of-N Algorithm:

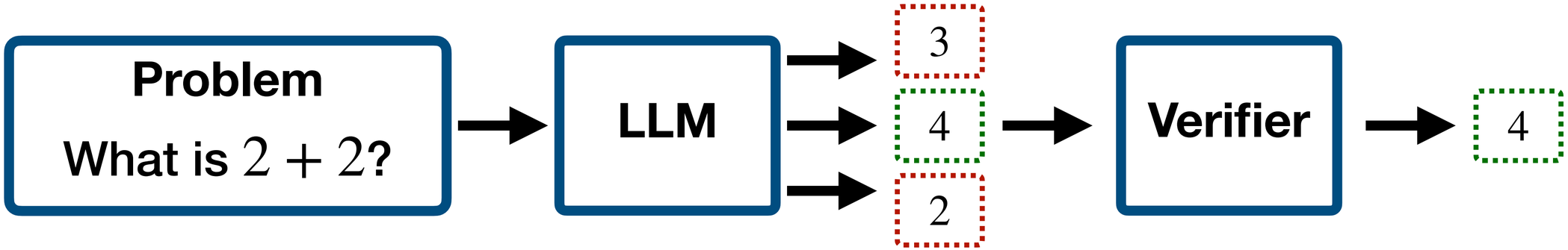
- Sample $y^1, \ldots, y^N \sim \pi(\cdot \mid x)$
- Return $y^i = \text{argmax}_i \, r(x, y^i)$

$y = a_{1:H}$ — response/trajectory

(full token sequence model generates when prompted)

$r(x, y)$ — task-specific reward/verifier

Best-of-N Sampling



Best-of-N Algorithm:

- Sample $y^1, \ldots, y^N \sim \pi(\cdot \mid x)$
- Return $y^i = \text{argmax}_i \, r(x, y^i)$

$y = a_{1:H}$ — response/trajectory

(full token sequence model generates when prompted)

$r(x, y)$ — task-specific reward/verifier

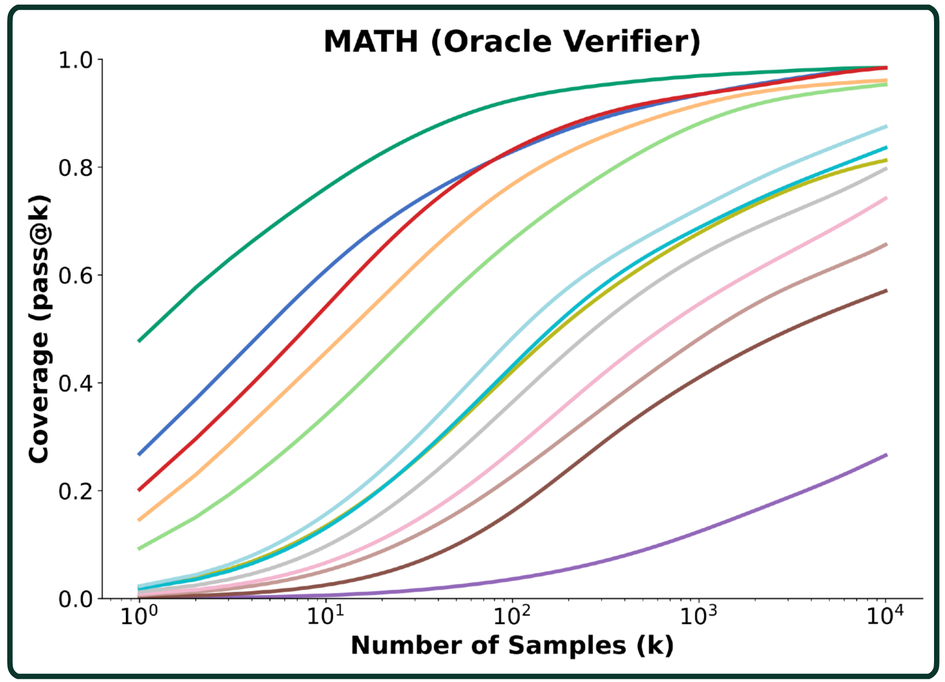
Understanding BoN: Binary Rewards

Example: Binary rewards

Suppose $r(x, y) = \mathbf{1}\{y = y^\star(x)\}$.

If $\pi(y^\star(x) \mid x) \gg 1/N$, $\mathrm{Pass@}N$ **contribution** $\approx$ **1.**

If $\pi(y^\star(x) \mid x) \ll 1/N$, $\mathrm{Pass@}N$ **contribution** $\approx$ **0.**



Brown et al 2024 (“Large Language Monkeys”)

The Coverage Profile

Definition: Coverage Profile

$$\mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi) := \mathbb{P}^{\pi_{\mathrm{T}}}\left[\frac{\pi_{\mathrm{T}}(y \mid x)}{\pi(y \mid x)} \geq N\right]$$

π — **Policy we sample from in Best-of-N**

π_{T} — **Task-specific policy with high reward**

Is best-of-N the best of them? Coverage, scaling, and optimality in inference-time alignment (Huang, Block, Liu, Jiang, Krishnamurthy, F 2025)

The Coverage Profile

Definition: Coverage Profile

$$\mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi) := \mathbb{P}^{\pi_{\mathrm{T}}}\left[\frac{\pi_{\mathrm{T}}(y \mid x)}{\pi(y \mid x)} \geq N\right]$$

π — Policy we sample from in Best-of-N

π_{T} — Task-specific policy with high reward

Proposition [HBLJKF ’25]

For any x, the Best-of-N distribution π_N^{BoN} satisfies:*

$$r(x, \pi_{\mathrm{T}}(x)) - r(x, \pi_N^{\mathrm{BoN}}(x)) \approx \mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi)$$

*Upper bound is for any [0,1]-valued r,
Lower bound is for worst-case [0,1]-valued r.

Is best-of-N the best of them? Coverage, scaling, and optimality in inference-time alignment (Huang, Block, Liu, Jiang, Krishnamurthy, F 2025)

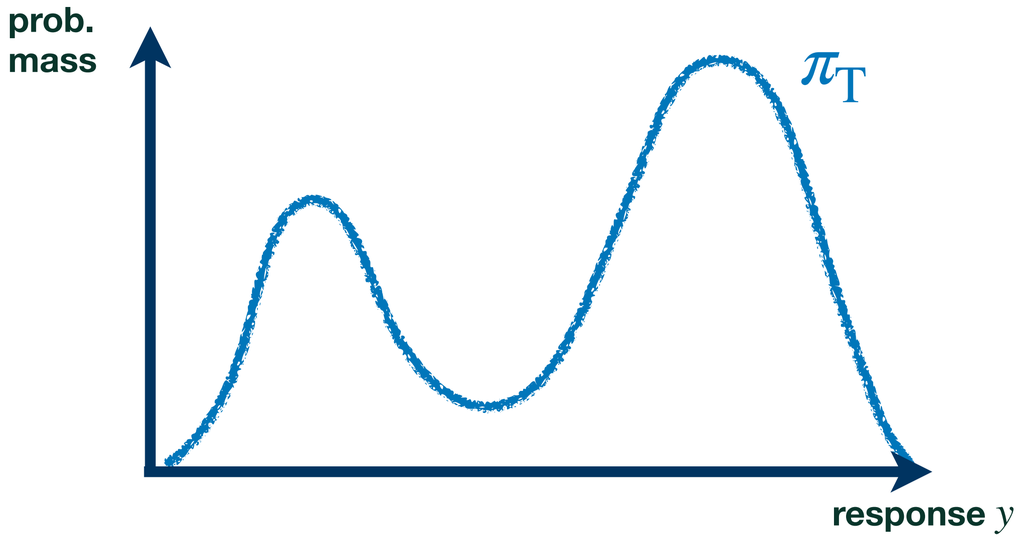
The Coverage Profile

Definition: Coverage Profile

$$\mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi) := \mathbb{P}^{\pi_{\mathrm{T}}}\left[\frac{\pi_{\mathrm{T}}(y \mid x)}{\pi(y \mid x)} \geq N\right]$$

π — **Policy we sample from in Best-of-N**

π_{T} — **Task-specific policy with high reward**



Proposition [HBLJKF '25]

For any x, the Best-of-N distribution π_N^{BoN} satisfies:*

$$r(x, \pi_{\mathrm{T}}(x)) - r(x, \pi_N^{\mathrm{BoN}}(x)) \approx \mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi)$$

*Upper bound is for any [0,1]-valued r,
Lower bound is for worst-case [0,1]-valued r.

Is best-of-N the best of them? Coverage, scaling, and optimality in inference-time alignment (Huang, Block, Liu, Jiang, Krishnamurthy, F 2025)

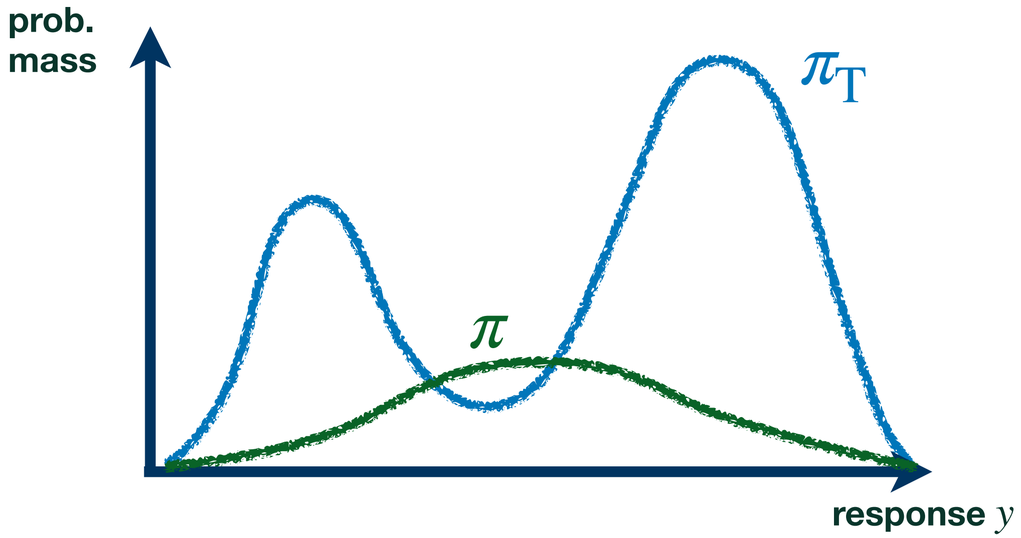
The Coverage Profile

Definition: Coverage Profile

$$\mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi) := \mathbb{P}^{\pi_{\mathrm{T}}}\left[\frac{\pi_{\mathrm{T}}(y \mid x)}{\pi(y \mid x)} \geq N\right]$$

π — **Policy we sample from in Best-of-N**

π_{T} — **Task-specific policy with high reward**



Proposition [HBLJKF '25]

For any x, the Best-of-N distribution π_N^{BoN} satisfies:*

$$r(x, \pi_{\mathrm{T}}(x)) - r(x, \pi_N^{\mathrm{BoN}}(x)) \approx \mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi)$$

*Upper bound is for any [0,1]-valued r,
Lower bound is for worst-case [0,1]-valued r.

Is best-of-N the best of them? Coverage, scaling, and optimality in inference-time alignment (Huang, Block, Liu, Jiang, Krishnamurthy, F 2025)

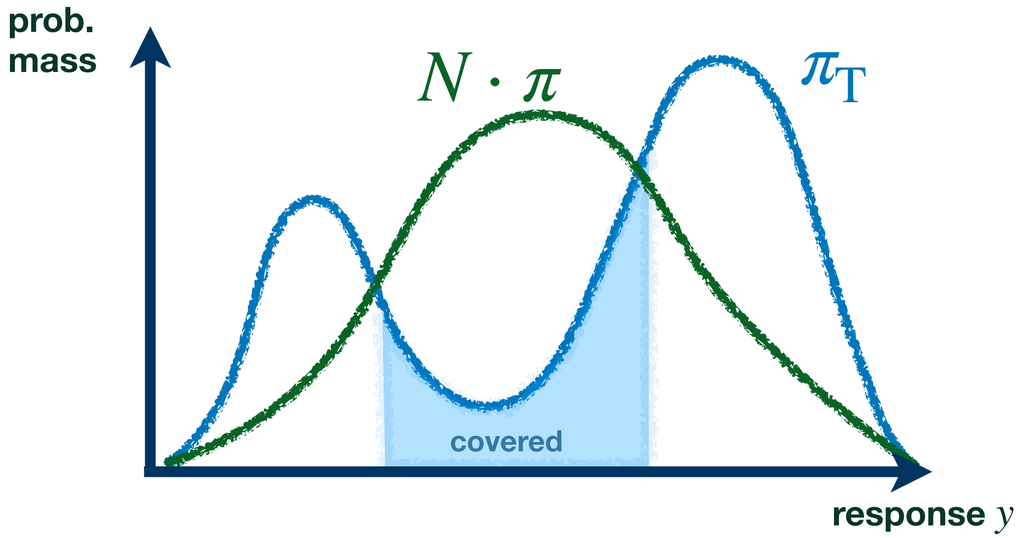
The Coverage Profile

Definition: Coverage Profile

$$\mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi) := \mathbb{P}^{\pi_{\mathrm{T}}}\left[\frac{\pi_{\mathrm{T}}(y \mid x)}{\pi(y \mid x)} \geq N\right]$$

π — **Policy we sample from in Best-of-N**

π_{T} — **Task-specific policy with high reward**



Proposition [HBLJKF '25]

For any x, the Best-of-N distribution π_N^{BoN} satisfies:*

$$r(x, \pi_{\mathrm{T}}(x)) - r(x, \pi_N^{\mathrm{BoN}}(x)) \approx \mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi)$$

*Upper bound is for any [0,1]-valued r,
Lower bound is for worst-case [0,1]-valued r.

Is best-of-N the best of them? Coverage, scaling, and optimality in inference-time alignment (Huang, Block, Liu, Jiang, Krishnamurthy, F 2025)

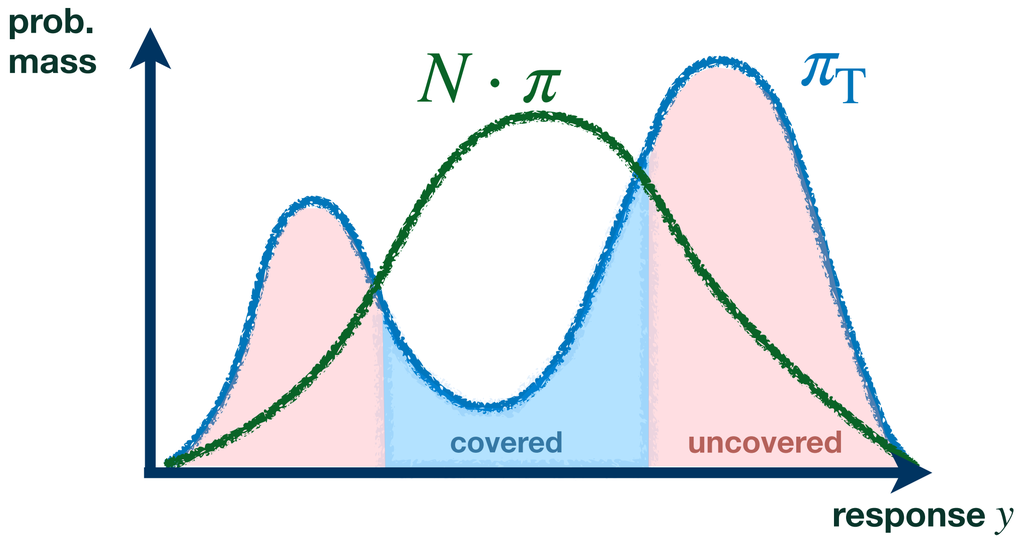
The Coverage Profile

Definition: Coverage Profile

$$\mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi) := \mathbb{P}^{\pi_{\mathrm{T}}}\left[\frac{\pi_{\mathrm{T}}(y \mid x)}{\pi(y \mid x)} \geq N\right]$$

π — **Policy we sample from in Best-of-N**

π_{T} — **Task-specific policy with high reward**



Proposition [HBLJKF '25]

For any x, the Best-of-N distribution π_N^{BoN} satisfies:*

$$r(x, \pi_{\mathrm{T}}(x)) - r(x, \pi_N^{\mathrm{BoN}}(x)) \approx \mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi)$$

*Upper bound is for any [0,1]-valued r,
Lower bound is for worst-case [0,1]-valued r.

Is best-of-N the best of them? Coverage, scaling, and optimality in inference-time alignment (Huang, Block, Liu, Jiang, Krishnamurthy, F 2025)

The Coverage Profile

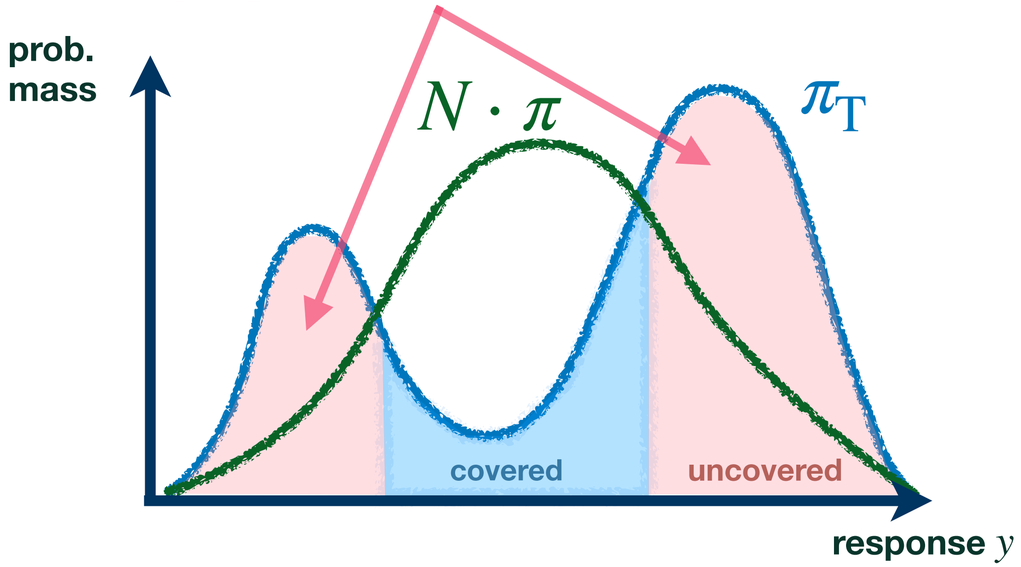
Definition: Coverage Profile

$$\mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi) := \mathbb{P}^{\pi_{\mathrm{T}}}\left[\frac{\pi_{\mathrm{T}}(y \mid x)}{\pi(y \mid x)} \geq N\right]$$

π — **Policy we sample from in Best-of-N**

π_{T} — **Task-specific policy with high reward**

$\mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi)$ **is mass of uncovered region.**



Proposition [HBLJKF '25]

For any x, the Best-of-N distribution π_N^{BoN} satisfies:*

$$r(x, \pi_{\mathrm{T}}(x)) - r(x, \pi_N^{\mathrm{BoN}}(x)) \approx \mathrm{Cov}_N(\pi_{\mathrm{T}} \,\|\, \pi)$$

*Upper bound is for any [0,1]-valued r,
Lower bound is for worst-case [0,1]-valued r.

Is best-of-N the best of them? Coverage, scaling, and optimality in inference-time alignment (Huang, Block, Liu, Jiang, Krishnamurthy, F 2025)

Coverage Profile: Summary

Definition: Coverage Profile

$$\mathrm{Cov}_N(\pi_T \,\|\, \pi) := \mathbb{P}^{\pi_T}\left[\frac{\pi_T(y \mid x)}{\pi(y \mid x)} \geq N\right]$$

- Coverage is necessary and sufficient for Best-of-N.
- Empirically, seems predictive of success for RL methods like GRPO as well [Yue et al. '25].
- Richer signal than cross-entropy or marginal $\mathrm{Pass}@1$.

log-loss behavior cloning

Does next-token prediction learn models with good coverage?

Beyond Classical IL: The Coverage Principle

Part I: Coverage governs downstream post-training performance

Part II: Pre-training implicitly provides coverage

Next-token prediction and coverage

Dataset satisfies $x^{(i)} \sim \rho,\ y^{(i)} \sim \pi^\star(\cdot \mid x^{(i)})$.

Next-token prediction for LM pre-training:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})} = \sum_{i=1}^{n} \log \frac{1}{\pi(y^{(i)} \mid x^{(i)})}$$

The coverage principle: How pre-training enables post-training (Chen, Huang, Golowich, Malladi, Block, Ash, Krishnamurthy, F 2025)

Next-token prediction and coverage

Dataset satisfies $x^{(i)} \sim \rho,\ y^{(i)} \sim \pi^\star(\cdot \mid x^{(i)})$.

Next-token prediction for LM pre-training:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})} = \sum_{i=1}^{n} \log \frac{1}{\pi(y^{(i)} \mid x^{(i)})}$$

Focus on covering $\pi^\star$ (assume $\pi^\star$ covers π_{T}; coverage is transitive).

The coverage principle: How pre-training enables post-training (Chen, Huang, Golowich, Malladi, Block, Ash, Krishnamurthy, F 2025)

Next-token prediction and coverage

Dataset satisfies $x^{(i)} \sim \rho,\ y^{(i)} \sim \pi^\star(\cdot \mid x^{(i)})$.

Next-token prediction for LM pre-training:

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \log \frac{1}{\pi(a_t^{(i)} \mid s_t^{(i)})} = \sum_{i=1}^{n} \log \frac{1}{\pi(y^{(i)} \mid x^{(i)})}$$

Focus on covering $\pi^\star$ (assume $\pi^\star$ covers π_{T}; coverage is transitive).

Proposition: Low cross-entropy implies coverage [CHGMBAKF '25]

$$\mathrm{Cov}_N(\pi^\star \,\|\, \hat{\pi}) \le \frac{D_{\mathrm{KL}}(\pi^\star \| \hat{\pi})}{\log(N/e)}$$

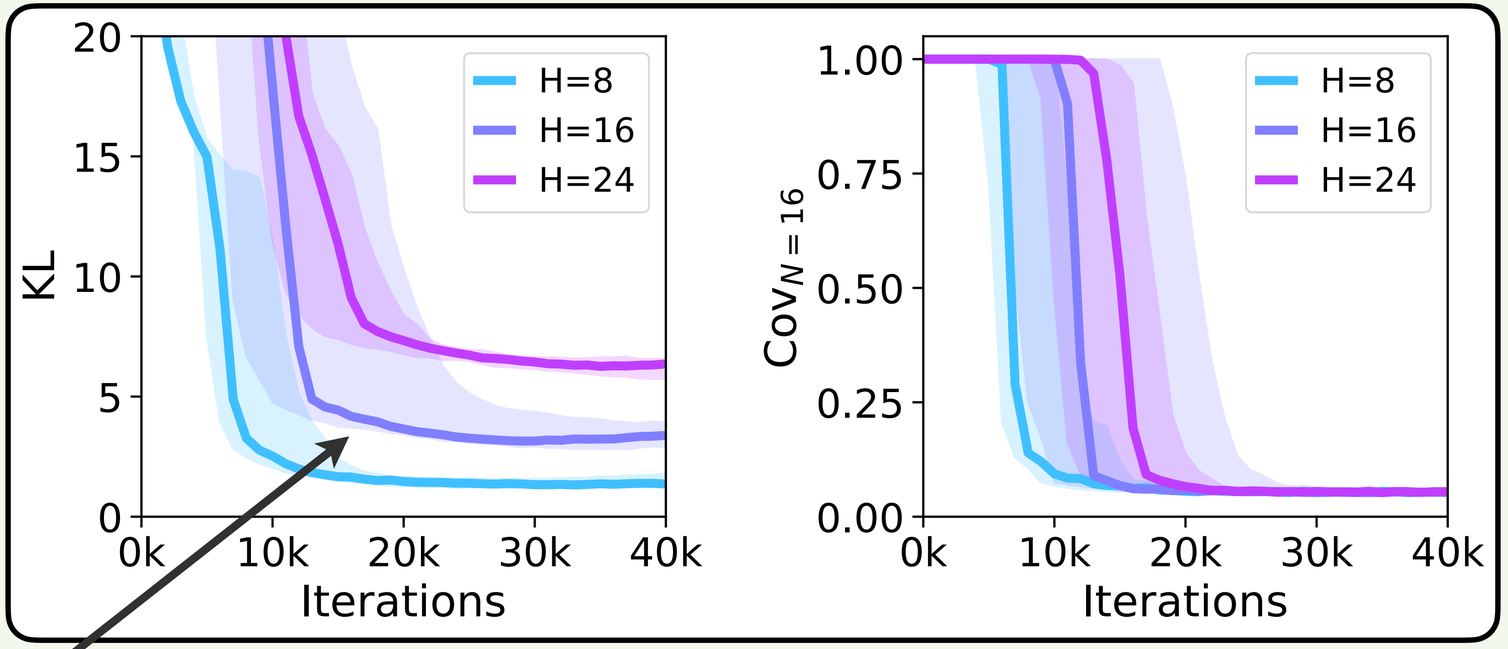
Cannot derive similar bound with Hellinger!

See also:

[Chatterjee & Diaconis '15]

[Block & Polyanskiy '23]

Cross-entropy is a loose upper bound



Task: Computing paths through graphs of depth H.
Model: 30M parameter GPT2-type transformer.

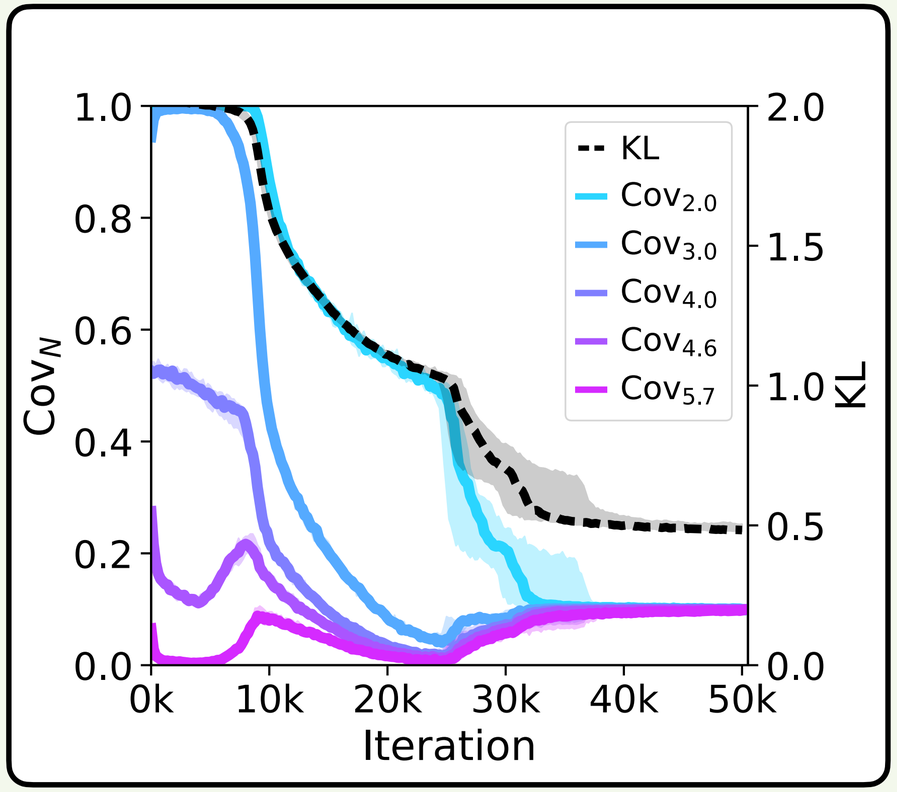
KL/CE scales with sequence length H, yet Cov_N does not!

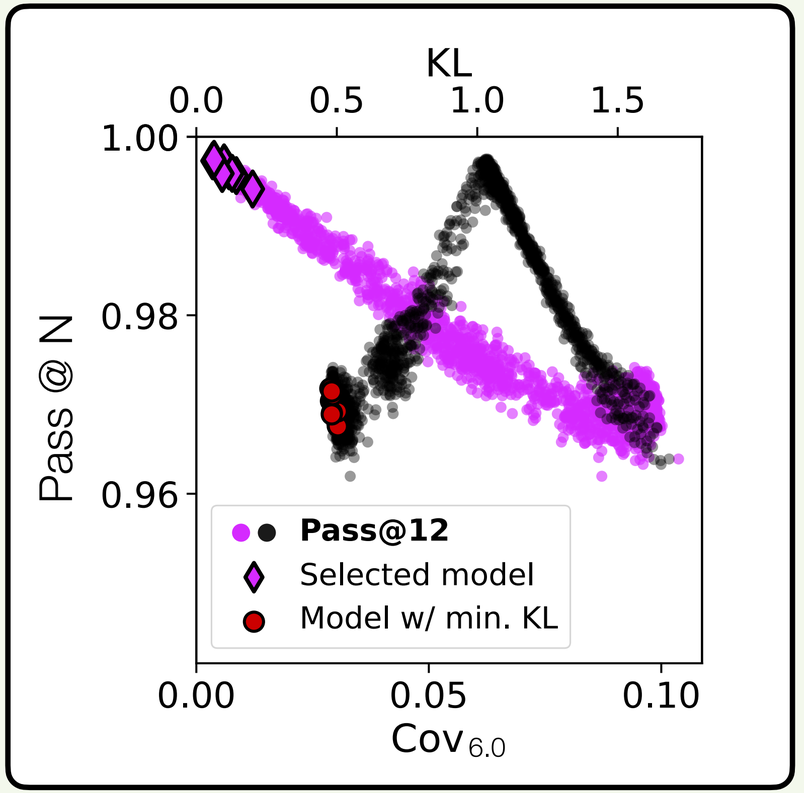
Proposition

$$\mathrm{Cov}_N(\pi^\star \,\|\, \hat{\pi}) \leq \frac{D_{\mathrm{KL}}(\pi^\star \| \hat{\pi})}{\log(N/e)}$$

The coverage principle: How pre-training enables post-training (Chen, Huang, Golowich, Malladi, Block, Ash, Krishnamurthy, F 2025)

Cross-entropy is a loose upper bound





Coverage profile (pink) correlates with BoN; KL (black) is anti-correlated.

KL/CE can be anti-correlated with downstream BoN performance!

See also: [Liu et al. '22], [Zeng et al. '25], [Chen et al. '25], [Lourie et al. '25], [Springer et al. '25]

Beyond Classical IL: The Coverage Principle

Part I: Coverage governs downstream post-training performance

Part II: Pre-training implicitly provides coverage

Testbed: Autoregressive linear models

Running example: Autoregressive linear model

$$\pi_\theta(a_t \mid x, a_{1:t-1}) \propto \exp\left(\left\langle \theta, \phi(x, a_{1:t}) \right\rangle\right)$$

with $\theta,\ \phi(x, a_{1:t}) \in \mathbb{R}^d$

Testbed: Autoregressive linear models

Running example: Autoregressive linear model

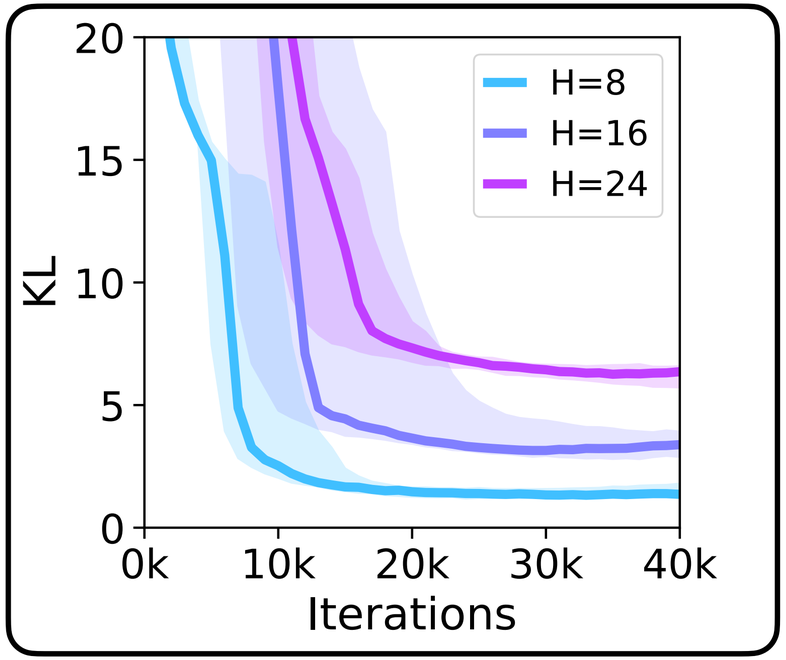
$$\pi_\theta(a_t \mid x, a_{1:t-1}) \propto \exp\left(\left\langle \theta, \phi(x, a_{1:t}) \right\rangle\right)$$

with $\theta,\ \phi(x, a_{1:t}) \in \mathbb{R}^d$

Proposition [CHGMBAKF '25]

Any algorithm (incl. next-token prediction) has:

$$D_{\mathrm{KL}}(\pi^\star \| \hat{\pi}) \gtrsim \frac{H}{n}$$



Holds even when $d = 1$ and $\|\theta\|,\ \|\phi(x, a_{1:t})\| \le 1$.

Next-token prediction implicitly provides coverage

Autoregressive linear model: $\pi_\theta(a_t \mid x, a_{1:t-1}) \propto \exp\left(\left\langle \theta, \phi(x, a_{1:t}) \right\rangle\right)$ with $\|\phi\|,\ \|\theta\| \le 1$.

Theorem [CHGMBAKF '25]

Next-token prediction satisfies (with high probability):

$$\mathrm{Cov}_N(\pi^\star \,\|\, \hat{\pi}) \lesssim \sqrt{\frac{\sigma^2}{n \log N}}$$

Where:

$$\sigma^2 := \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} \mathrm{Var}^{\pi^\star}[\phi_t \mid x, a_{1:t-1}]\right]$$

"sequence variance"

Through KL, can at best conclude:

$$\mathrm{Cov}_N(\pi^\star \,\|\, \hat{\pi}) \gtrsim \frac{H}{n \log N}$$

Next-token prediction implicitly provides coverage

Autoregressive linear model: $\pi_\theta(a_t \mid x, a_{1:t-1}) \propto \exp\left(\left\langle \theta, \phi(x, a_{1:t}) \right\rangle\right)$ with $\|\phi\|,\ \|\theta\| \le 1$.

Theorem [CHGMBAKF '25]

Next-token prediction satisfies (with high probability):

$$\mathrm{Cov}_N(\pi^\star \,\|\, \hat{\pi}) \lesssim \sqrt{\frac{\sigma^2}{n \log N}} + \frac{1}{n}.$$

Where:

$$\sigma^2 := \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} \mathrm{Var}^{\pi^\star}[\phi_t \mid x, a_{1:t-1}]\right]$$

"sequence variance"

Through KL, can at best conclude:

$$\mathrm{Cov}_N(\pi^\star \,\|\, \hat{\pi}) \gtrsim \frac{H}{n \log N}$$

The coverage principle: How pre-training enables post-training (Chen, Huang, Golowich, Malladi, Block, Ash, Krishnamurthy, F 2025)

Next-token prediction implicitly provides coverage

Autoregressive linear model: $\pi_\theta(a_t \mid x, a_{1:t-1}) \propto \exp\left(\left\langle \theta, \phi(x, a_{1:t})\right\rangle\right)$ with $\|\phi\|,\ \|\theta\| \le 1$.

Theorem [CHGMBAKF '25]

Next-token prediction satisfies (with high probability):

$$\mathrm{Cov}_N(\pi^\star \,\|\, \hat{\pi}) \lesssim \sqrt{\frac{\sigma^2}{n \log N}} + \frac{1}{n}.$$

Where:

$$\sigma^2 := \mathbb{E}^{\pi^\star}\left[\sum_{t=1}^{H} \mathrm{Var}^{\pi^\star}[\phi_t \mid x, a_{1:t-1}]\right]$$

"sequence variance"

Through KL, can at best conclude:

$$\mathrm{Cov}_N(\pi^\star \,\|\, \hat{\pi}) \gtrsim \frac{H}{n \log N}$$

Horizon-dependence: Implicit in σ^2 ("pivotal tokens").

Improves with N until lower-order $1/n$ term.

The coverage principle: How pre-training enables post-training (Chen, Huang, Golowich, Malladi, Block, Ash, Krishnamurthy, F 2025)

Takeaway:
Coverage can generalize w/ fewer samples than required for imitation.

Similar guarantees for general Π: larger $N \Longrightarrow$ weaker dependence on model capacity.

Next-token prediction can also learn models w/ good coverage under severe misspecification.

The coverage principle: How pre-training enables post-training (Chen, Huang, Golowich, Malladi, Block, Ash, Krishnamurthy, F 2025)

The coverage perspective: Interventions

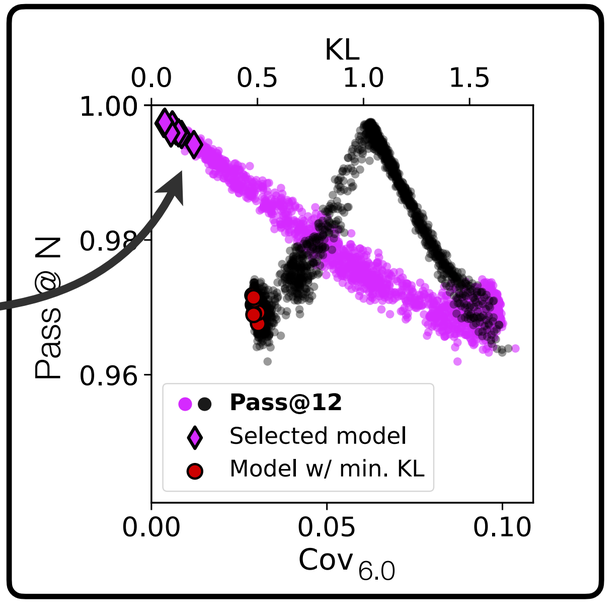
Optimization:

- Vanilla SGD has poor coverage (w.r.t H); **gradient normalization** (a-la Adam) fixes.
- **Test-time training** (SGD updates during autoregressive gen.) improves coverage.

[Krause et al. ’19, Sun et al. ’24, Akyürek et al. ‘25]

Checkpoint selection:

- Can select checkpoints with good coverage via **tournaments**; leads to better coverage than naively picking based on CE.



Summary: Part III

1. Error compounding is benign in discrete spaces (when $\pi^\star \in \Pi$)

2. …but can manifest under insufficient model capacity.

3. Coverage suffices for downstream post-training;
   requires fewer samples + assumptions than traditional imitation.

Imitation Learning in Continuous Action Spaces

The Peculiarities of Continuous Space

Max Simchowitz

w/ Adam Block and Dylan Foster

Outline

I. Background and Introduction.

II. Classical Imitation Learning.

III. Modern IL in discrete settings.

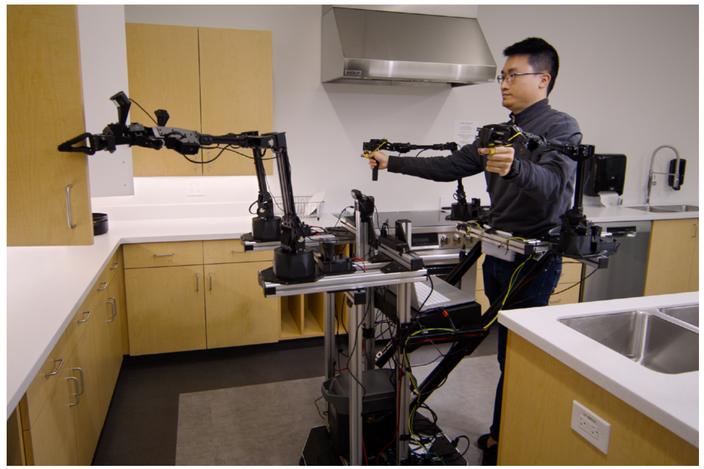
IV. Modern IL in continuous settings.

V. Opening the black box of optimization.

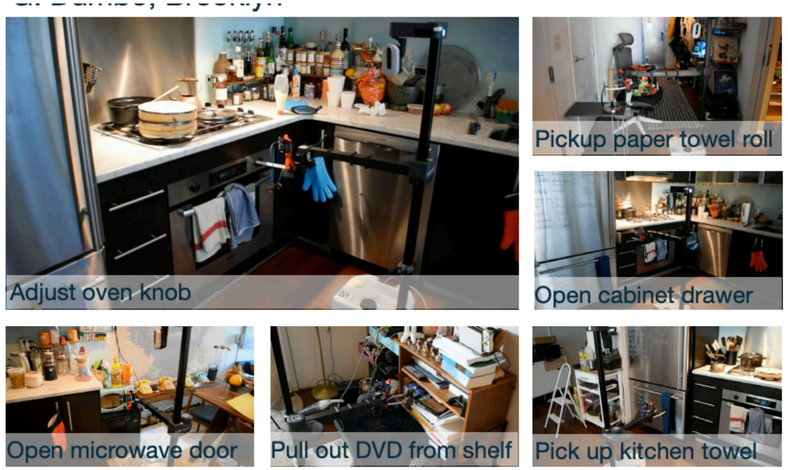
Imitation Learning in Continuous Action Spaces

1. Imitation Learning in Robot Learning

Behavior Cloning and Modern Robot Learning



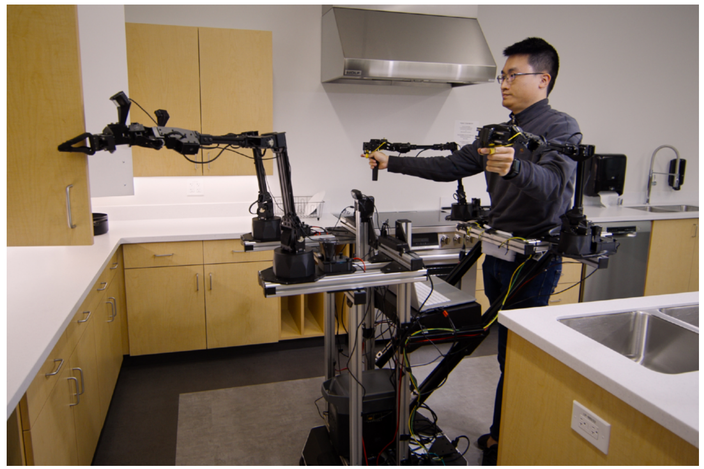
‘Aloha’



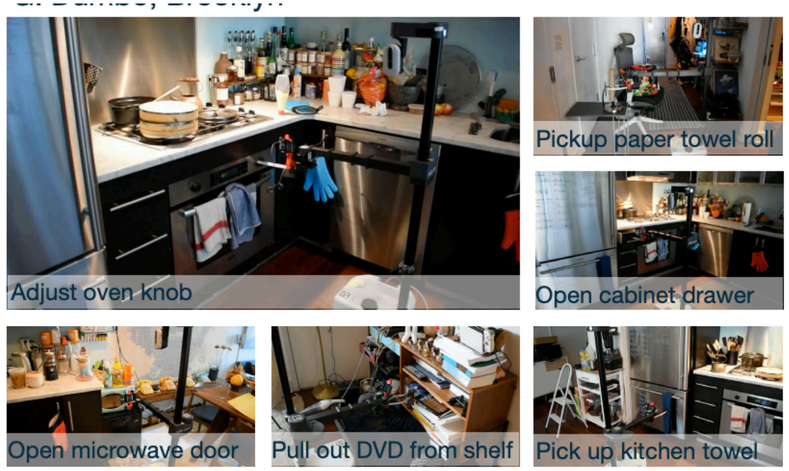
‘Bringing Robots Home’

TRI Teleoperation

Behavior Cloning and Modern Robot Learning



‘Aloha’

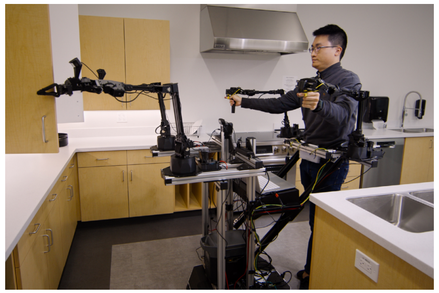


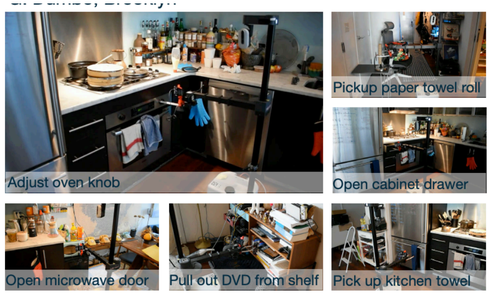
‘Bringing Robots Home’

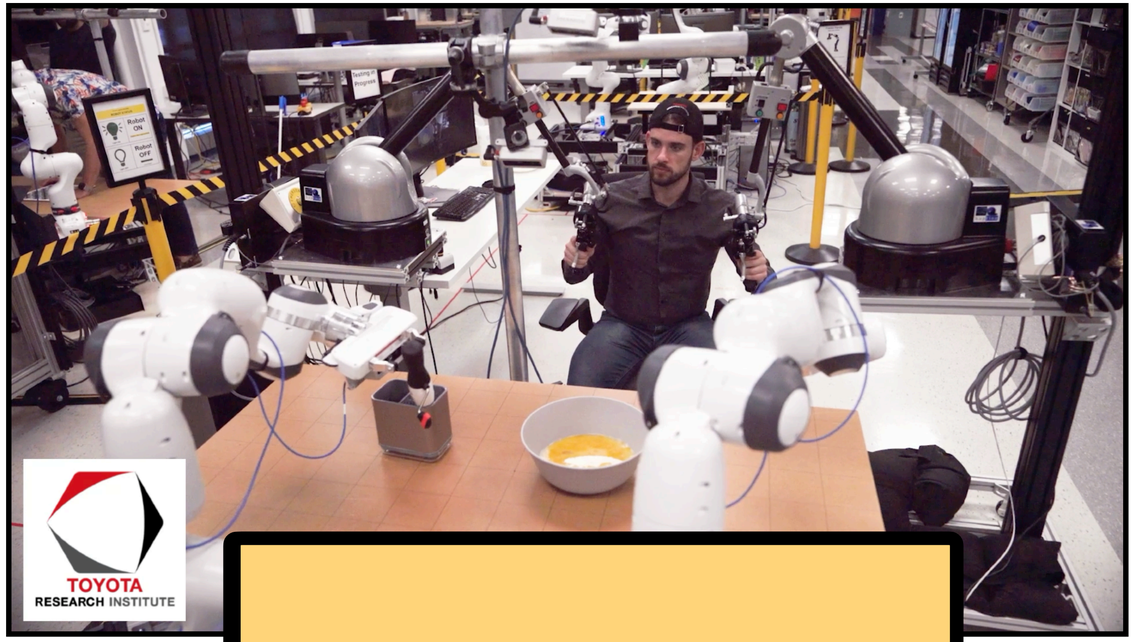
TRI Teleoperation

And a growing list of companies…

Behavior Cloning and Modern Robot Learning

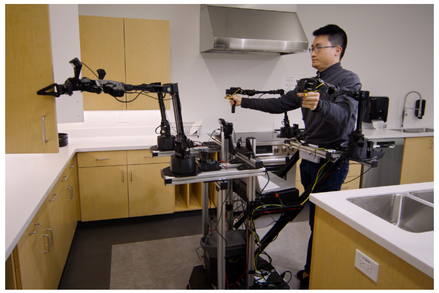


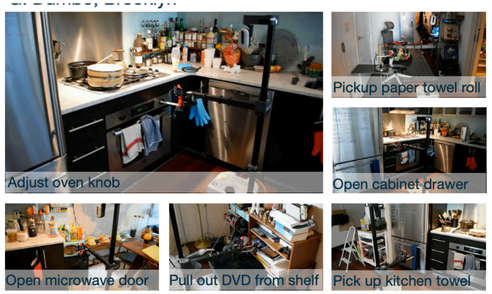


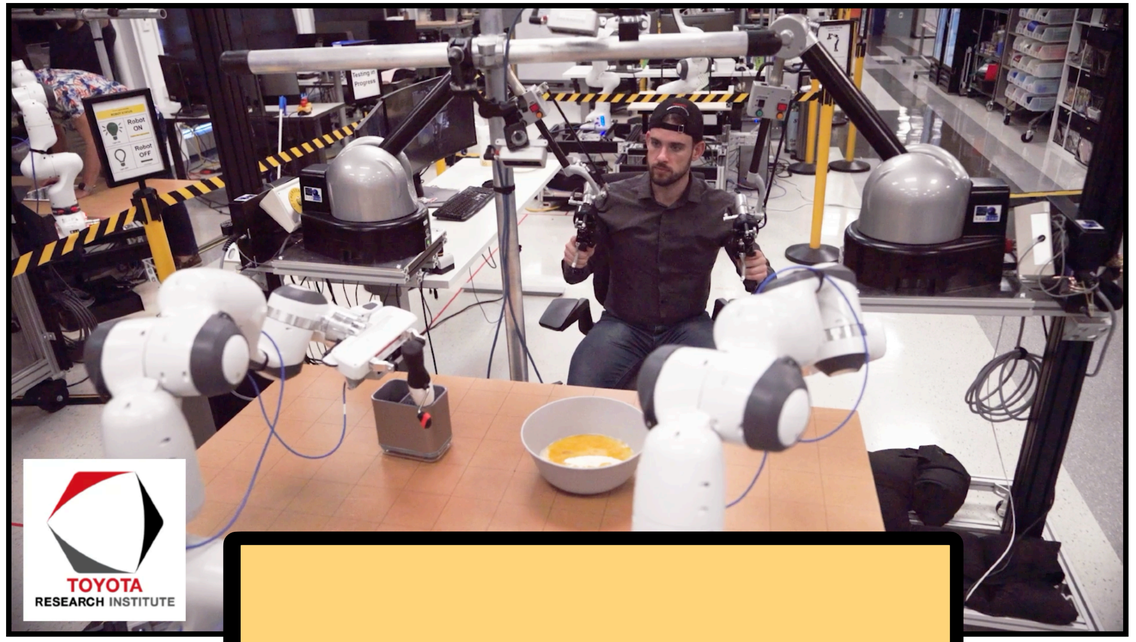


We use a human as the **expert demonstrator.**

Behavior Cloning and Modern Robot Learning

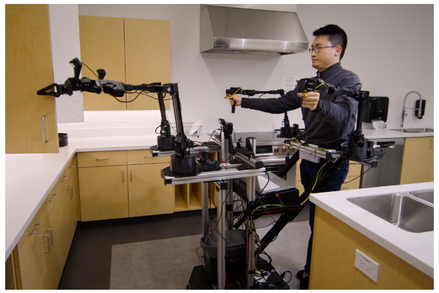


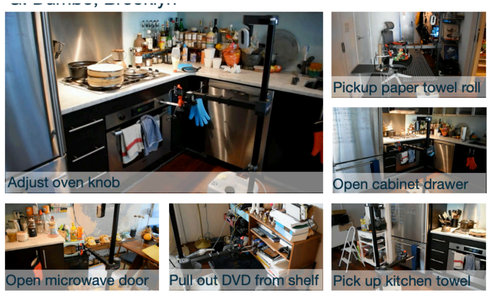


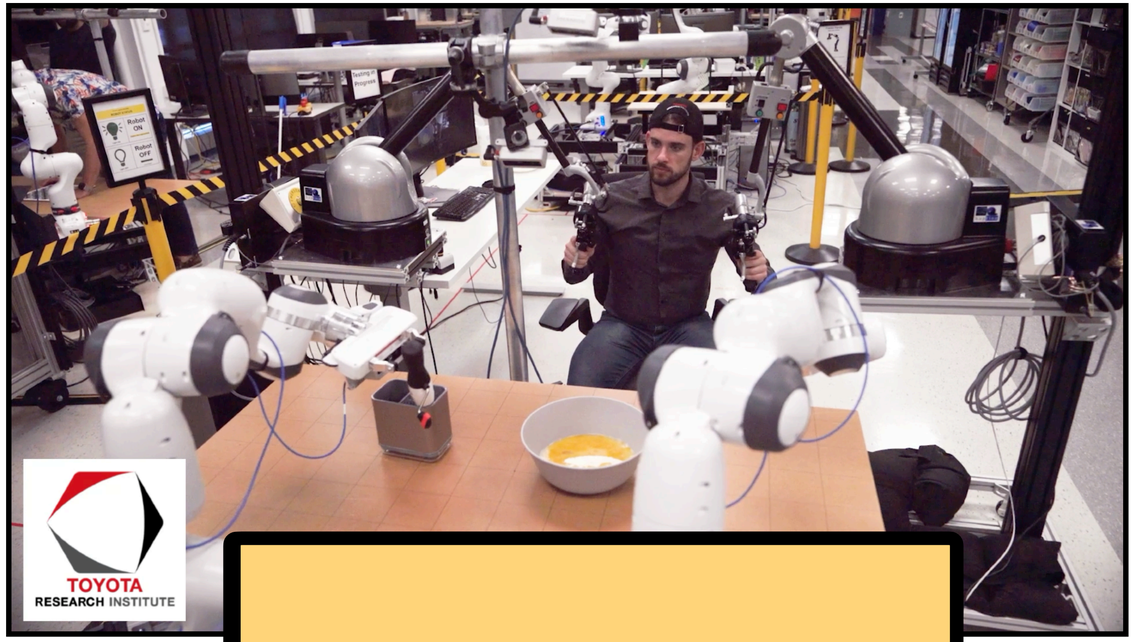


We use a human as the **expert demonstrator.**

Behavior Cloning and Modern Robot Learning



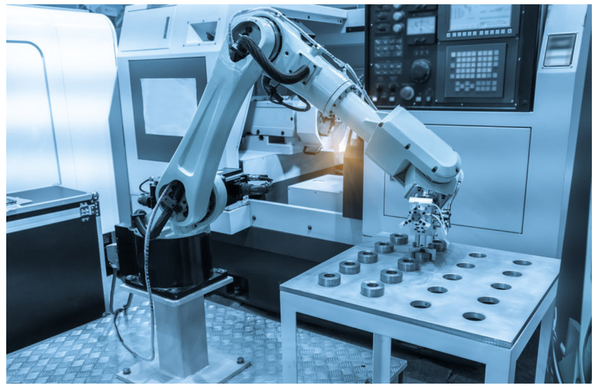




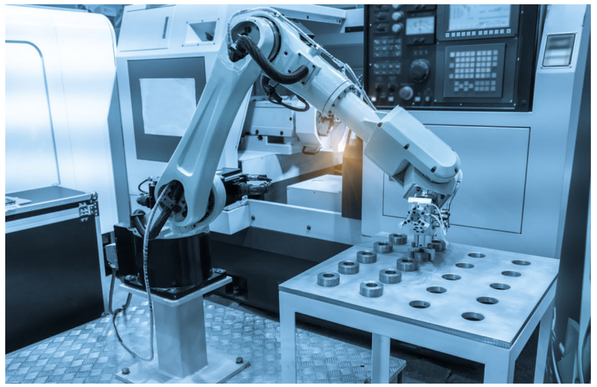
We use a human as the **expert demonstrator.**

Our aim is to predict a “**next action**” (robot action) from **observation** (pixels, tactile sensing.)

The Physical World 🤖 v.s. The Discrete World 📚

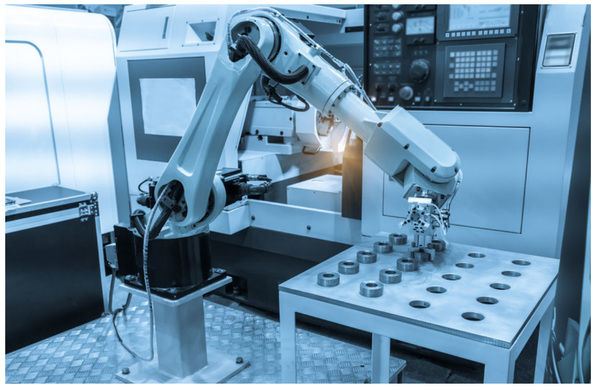


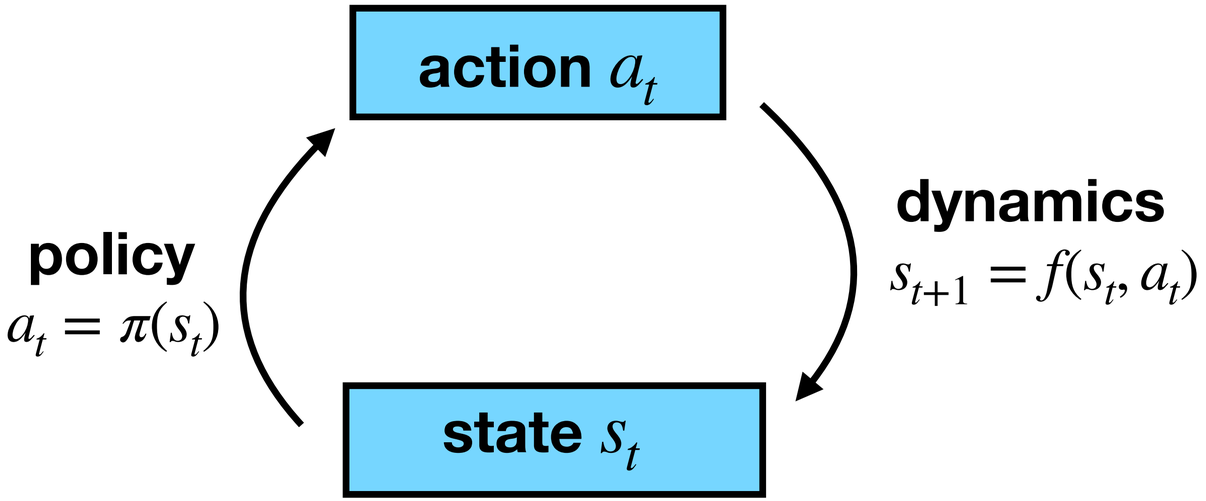
The Physical World 🤖 v.s. The Discrete World 📚



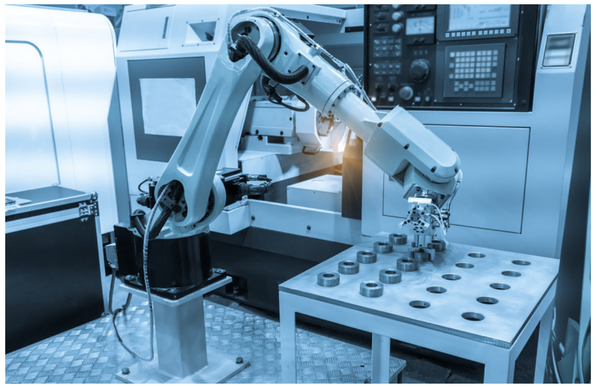
GPT-

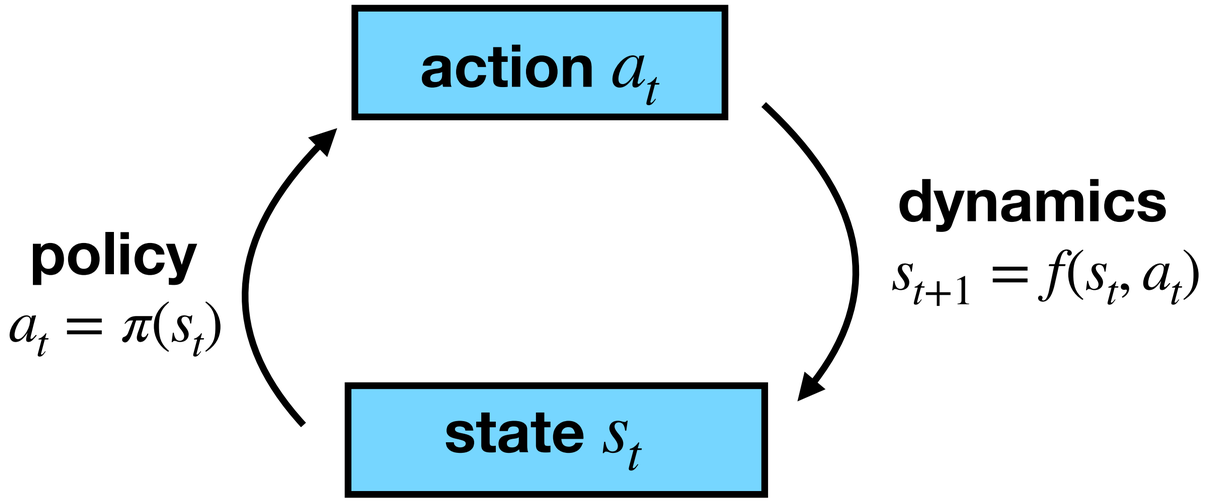
The Physical World 🤖 v.s. The Discrete World 📚



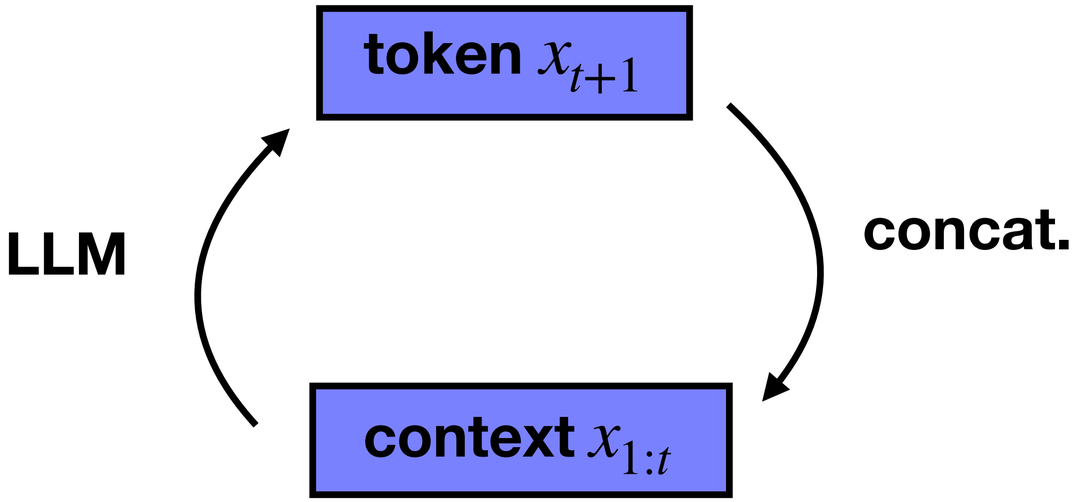


The Physical World 🤖 v.s. The Discrete World 📚

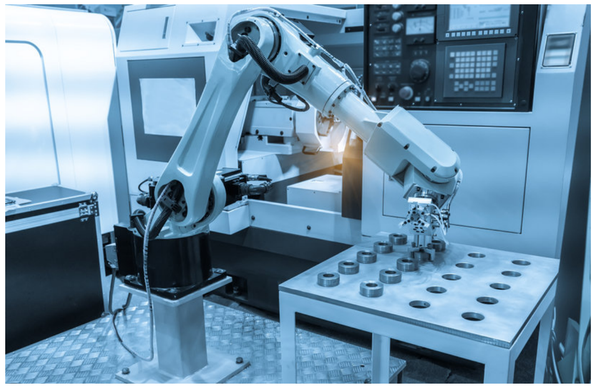


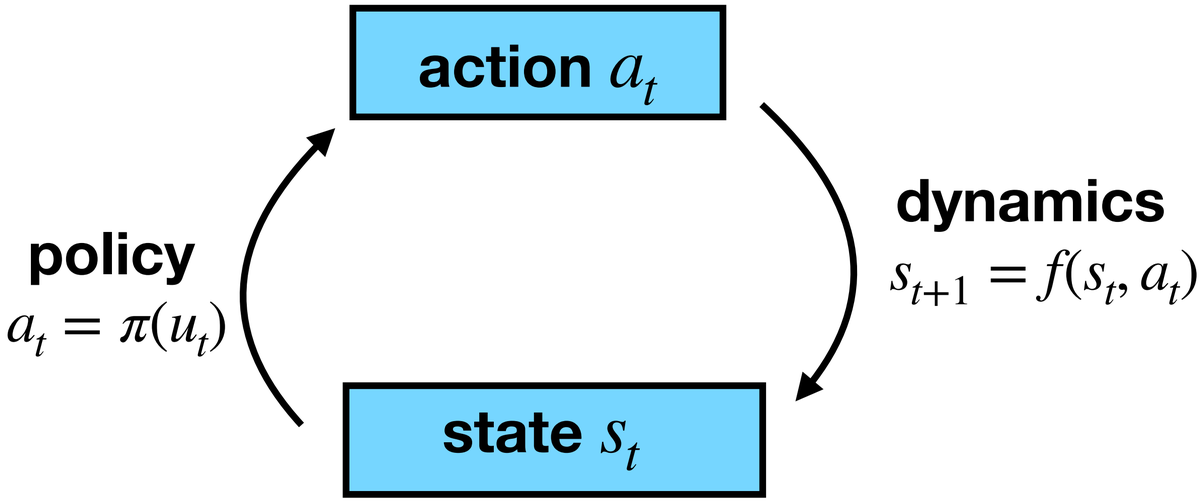


GPT-



The Physical World 🤖 v.s. The Discrete World 📚

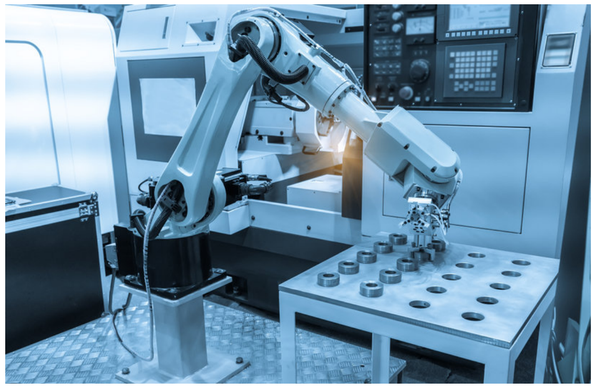


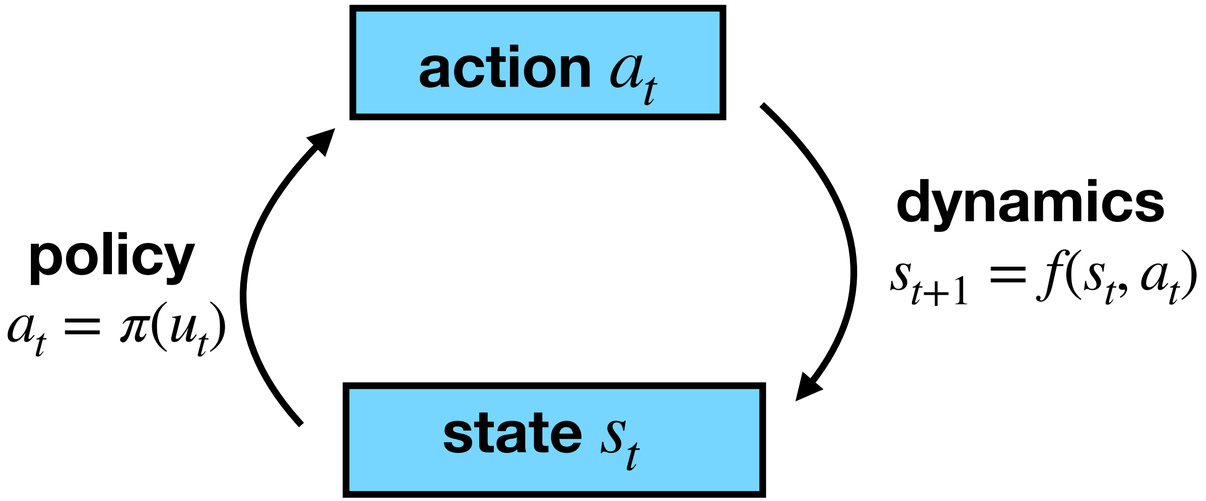


GPT-

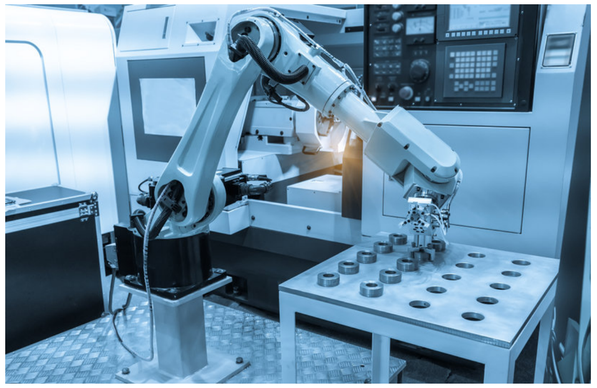
In Dylan's Part III, we saw this was easier than expected!

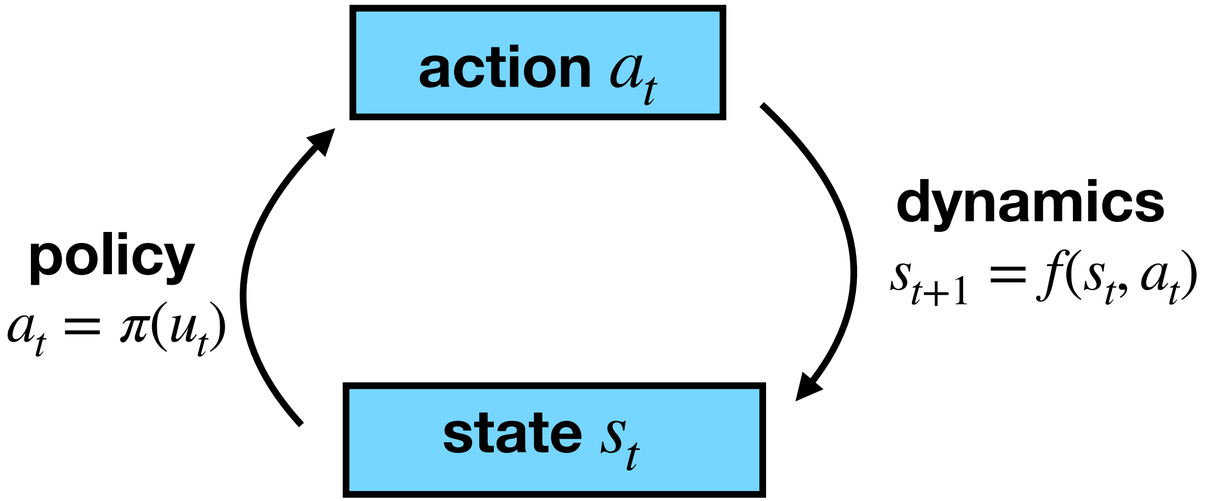
The Physical World 🤖 v.s. The Discrete World 📚





The Physical World 🤖 v.s. The Discrete World 📚





States and actions take continuous values

The Physical World 🤖 v.s. The Discrete World 📚

It is … easy to make computers exhibit adult level performance on intelligence tests …, and **difficult or impossible to give them the skills of a one-year-old when it comes to perception and mobility.**

Hans Moravec, ’98

The Physical World 🤖 v.s. The Discrete World 📚

It is … easy to make computers exhibit adult level performance on intelligence tests …, and **difficult or impossible to give them the skills of a one-year-old when it comes to perception and mobility.**

Hans Moravec, ’98

Part IV: A Mathematical Basis for Moravec's Paradox?

Part IV: A Mathematical Basis for Moravec's Paradox?

Part IV: A Mathematical Basis for Moravec's Paradox?

1. Imitation is **considerably more challenging** in the physical world 🤖 than in the discrete world 📚**.**

Part IV: A Mathematical Basis for Moravec's Paradox?

1. Imitation is **considerably more challenging** in the physical world 🤖 than in the discrete world 📚**.**

Part IV: A Mathematical Basis for Moravec's Paradox?

1. Imitation is **considerably more challenging** in the physical world 🤖 than in the discrete world 📚**.**

2. **Popular design decisions** from today's world of robotics are indispensable to avoiding these challenges.

In More Detail…

In More Detail…

In More Detail…

- Formalize BC in the continuous world via **continuous control**

In More Detail…

- Formalize BC in the continuous world via **continuous control**
- Explain why BC in the continuous world is **more challenging**

In More Detail…

- Formalize BC in the continuous world via **continuous control**
- Explain why BC in the continuous world is **more challenging**
- Describe how **action chunking** and **exploratory data collection** address these challenges.

In More Detail…

- Formalize BC in the continuous world via **continuous control**
- Explain why BC in the continuous world is **more challenging**
- Describe how **action chunking** and **exploratory data collection** address these challenges.
- Coda: **generative models** as controller parametrization

Imitation Learning in Continuous Action Spaces

2. Formalizing BC when Actions are Continuous

Recall: MDPs

Recall: A Markov Decision Process (MDP) is specified by

- **states** $s \in S$
- **actions** $a \in A$
- **transition probabilities** $s' \sim P(s, a)$
- **rewards** $r(s, a)$
- **initial state distribution** $s_1 \sim P_1$

Recall: MDPs

Recall: A Markov Decision Process (MDP) is specified by

- **states** $s \in S$
- **actions** $a \in A$
- **transition probabilities** $s' \sim P(s, a)$
- **rewards** $r(s, a)$
- **initial state distribution** $s_1 \sim P_1$

We will assume **observations are sufficient for state.** **Hence MDP.**

Formalizing BC in Continuous Action Spaces

Recall: A Markov Decision Process (MDP) is specified by

- **continuous states** $s \in \mathbb{R}^d$
- **continuous actions** $a \in \mathbb{R}^m$
- **transition dynamics** $s_{t+1} = f(s_t, a_t)$
- **reward** $r(s, a)$
- **initial state distribution** $s_1 \sim P_1$

Formalizing BC in Continuous Action Spaces

Recall: A Markov Decision Process (MDP) is specified by

- **continuous states** $s \in \mathbb{R}^d$
- **continuous actions** $a \in \mathbb{R}^m$
- **transition dynamics** $s_{t+1} = f(s_t, a_t)$
- **reward** $r(s, a)$
- **initial state distribution** $s_1 \sim P_1$

This formulation often called 'control system'
(return later)

Formalizing BC in Continuous Action Spaces

Assumption: f and $\pi^\star$ are deterministic and smooth*

***have continuous gradients:**

$$\|\nabla \pi^\star - \nabla \pi^\star(x')\| \leq L\|x - x'\|$$

Formalizing BC in Continuous Action Spaces

Recall: The **J-function** is $J(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=1}^{H} r(s_t, a_t)\right]$

Goal: $J(\pi^{\star}) - J(\hat{\pi}) \leq \text{small}$

Formalizing BC in Continuous Action Spaces

Recall: The **J-function** is $J(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=1}^{H} r(s_t, a_t)\right]$

Goal: $J(\pi^{\star}) - J(\hat{\pi}) \leq \text{small}$

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Example 2: $\text{loss}_2(\pi, s, a) = \|a - \pi(s)\|^2$ ($\pi^\star$ is takes continuous values)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Example 2: $\text{loss}_2(\pi, s, a) = \|a - \pi(s)\|^2$ ($\pi^\star$ is takes continuous values)

Example 3: $\text{loss}_{\log}(\pi, s, a) = \log \pi(a \mid s)$

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Example 2: $\text{loss}_2(\pi, s, a) = \|a - \pi(s)\|^2$ ($\pi^\star$ is takes continuous values)

Example 3: $\text{loss}_{\log}(\pi, s, a) = \log \pi(a \mid s)$ ($\pi^\star$ is discrete, or $\pi^\star(s)$ has density)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Example 2: $\text{loss}_2(\pi, s, a) = \|a - \pi(s)\|^2$ ($\pi^\star$ is takes continuous values)

Example 3: $\text{loss}_{\log}(\pi, s, a) = \log \pi(a \mid s)$ ($\pi^\star$ is discrete, or $\pi^\star(s)$ has density)

Example 4: $\text{loss}(\pi, s, a) =$ (Score Matching)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Example 1: $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$ ($\pi^\star$ is discrete)

Example 2: $\text{loss}_2(\pi, s, a) = \|a - \pi(s)\|^2$ ($\pi^\star$ is takes continuous values)

Example 3: $\text{loss}_{\log}(\pi, s, a) = \log \pi(a \mid s)$ ($\pi^\star$ is discrete, or $\pi^\star(s)$ has density)

Example 4: $\text{loss}(\pi, s, a) = (\text{Score Matching})$ (popular in robotics)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Recall $L_{\text{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \text{loss}(s_t, a_t, \pi)\right]$.

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Recall $L_{\text{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \text{loss}(s_t, a_t, \pi)\right]$.

Recall the classical result that, for $\text{loss}_{0/1}(\pi, s, a) = \mathbf{1}_{\pi(s) \neq a}$**,**

$$J(\pi^\star) - J(\hat{\pi}) \leq \text{poly}(H) \cdot L_{\text{bc}}(\hat{\pi})$$

(Ross & Bagnell '10)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Theorem (SPJ'25): There are BC problems in **continuous action spaces** where **learning in log-loss or 0/1 or Hellinger** is impossible:

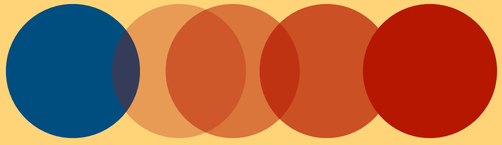
Expert

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Theorem (SPJ'25): There are BC problems in **continuous action spaces** where **learning in log-loss or 0/1 or Hellinger** is impossible:



Intuition: Zeno's Paradox

Expert

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Most of talk: $\text{loss}(s, a, \pi) = \|\pi(s) - a\|^2$ (assume deterministic expert)

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Most of talk: $\text{loss}(s, a, \pi) = \|\pi(s) - a\|^2$ (assume deterministic expert)

$$L_{\text{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \|\pi(s_t) - \pi^\star(s_t)\|^2\right]$$

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \text{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Most of talk: $\text{loss}(s, a, \pi) = \|\pi(s) - a\|^2$ (assume deterministic expert)

$$L_{\text{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \|\pi(s_t) - \pi^\star(s_t)\|^2\right]$$

something that is learnable.

Behavior Cloning

Behavior Cloning: Learner selects $\hat{\pi}$ to minimize **supervised learning error**

$$\hat{\pi} \approx \arg\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{t=1}^{H} \mathrm{loss}(s_t^{(i)}, a_t^{(i)}, \pi)$$

Most of talk: $\mathrm{loss}(s, a, \pi) = \|\pi(s) - a\|^2$ (assume deterministic expert)

$$L_{\mathrm{bc}}(\pi) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \|\pi(s_t) - \pi^\star(s_t)\|^2\right]$$

something that is learnable.

Dudley ’68, Van de Geer ‘00, Tsybakov ’06.

Are reductions to supervised learning possible?

Behavior Cloning: Suppose that the learner can ensure

$$L_{\mathrm{bc}}(\hat{\pi}) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \|\pi^\star(s_t) - \hat{\pi}(s_t)\|^2\right] \leq \epsilon$$

Are reductions to supervised learning possible?

Behavior Cloning: Suppose that the learner can ensure

$$L_{\mathrm{bc}}(\hat{\pi}) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \|\pi^\star(s_t) - \hat{\pi}(s_t)\|^2\right] \leq \epsilon$$

Can we guarantee that $J(\pi^\star) - J(\hat{\pi}) \leq \mathrm{poly}(H) \cdot \epsilon$**?**

sub-optimality

(By Analogy to Ross & Bagnell, '10)

Are reductions to supervised learning possible?

Behavior Cloning: Suppose that the learner can ensure

$$L_{\mathrm{bc}}(\hat{\pi}) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \|\pi^\star(s_t) - \hat{\pi}(s_t)\|^2\right] \leq \epsilon$$

Can we guarantee that $J(\pi^\star) - J(\hat{\pi}) \leq \mathrm{poly}(H) \cdot \epsilon$**?**

sub-optimality

In the discrete setting, the question was: do we pay for H at all?

(By Analogy to Ross & Bagnell, '10)

Are reductions to supervised learning possible?

Behavior Cloning: Suppose that the learner can ensure

$$L_{\mathrm{bc}}(\hat{\pi}) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \|\pi^\star(s_t) - \hat{\pi}(s_t)\|^2\right] \leq \epsilon$$

Question 1: **Can we guarantee that** $J(\pi^\star) - J(\hat{\pi}) \leq \mathrm{poly}(H) \cdot \epsilon$**?**

Are reductions to supervised learning possible?

Behavior Cloning: Suppose that the learner can ensure

$$L_{\mathrm{bc}}(\hat{\pi}) := \mathbb{E}^{\pi^\star}\left[\frac{1}{H}\sum_{t=1}^{H} \|\pi^\star(s_t) - \hat{\pi}(s_t)\|^2\right] \leq \epsilon$$

Question 1: **Can we guarantee that** $J(\pi^\star) - J(\hat{\pi}) \leq \mathrm{poly}(H) \cdot \epsilon$**?**

Question 2: **Is there some other algorithm which attains this?**

Summary: Formalizing Continuous Action BC

Summary: Formalizing Continuous Action BC

1. In continuous settings, 0/1 loss learning is impossible

Summary: Formalizing Continuous Action BC

1. In continuous settings, 0/1 loss learning is impossible

2. We study learning in the square loss

Summary: Formalizing Continuous Action BC

1. In continuous settings, 0/1 loss learning is impossible

2. We study learning in the square loss

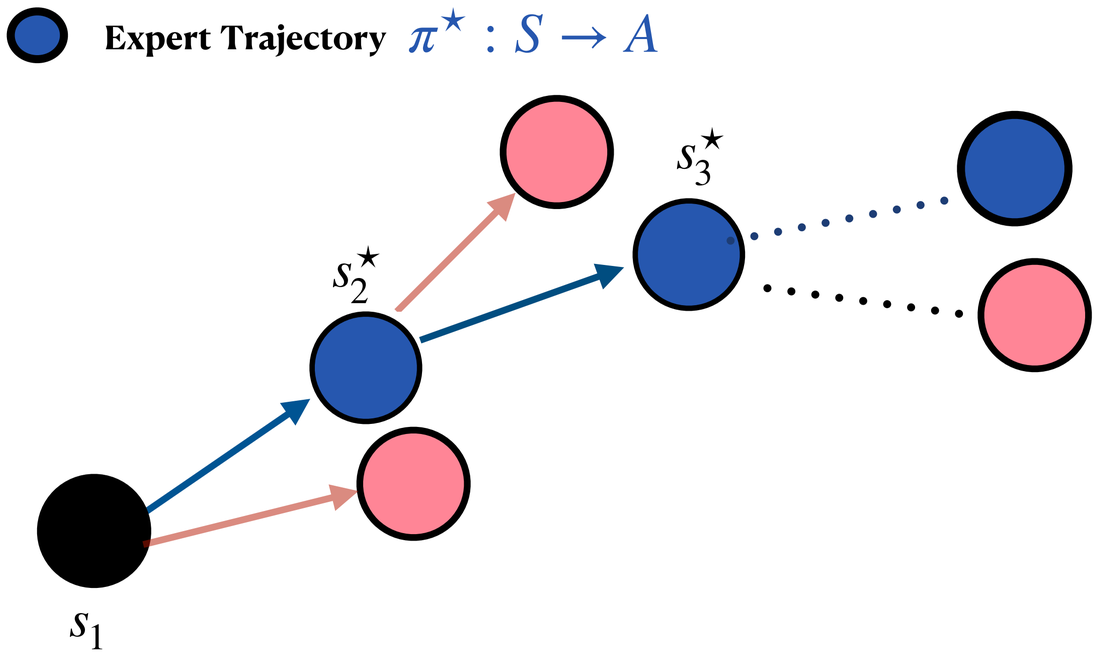
3. Can we recover at most poly-in-horizon compounding error?

Imitation Learning in Continuous Action Spaces

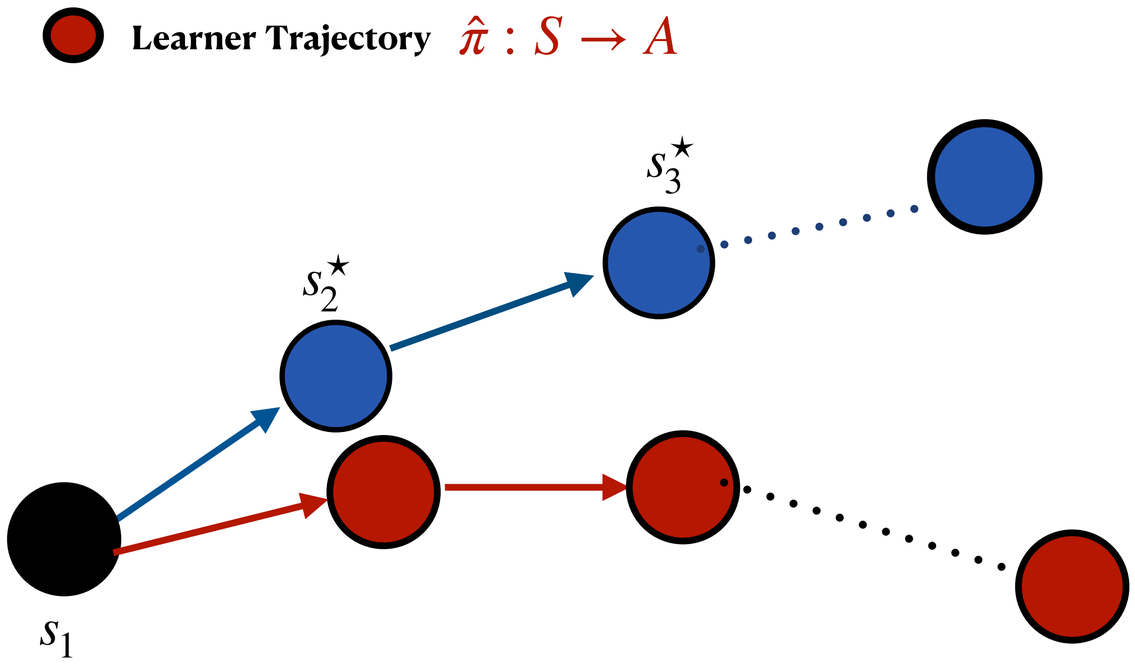
3. Pitfalls of Continuous Action Imitation Learning

“Pitfalls of Imitation Learning when the Actions Are Continuous,” Simchowitz, Pfrommer, Jadbabaie ’25.

BC Training v.s. Deployment



Learner selects $\hat{\pi}$ to minimize **supervised learning error** L_{bc}



But errors can accumulate!

aka '**curse of horizon**'

Continuous-action imitation is hard

Continuous-action imitation is hard

Theorem (SPJ ’25)**:** Consider the following setting

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathcal{J}$ of possible expert/dynamics

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

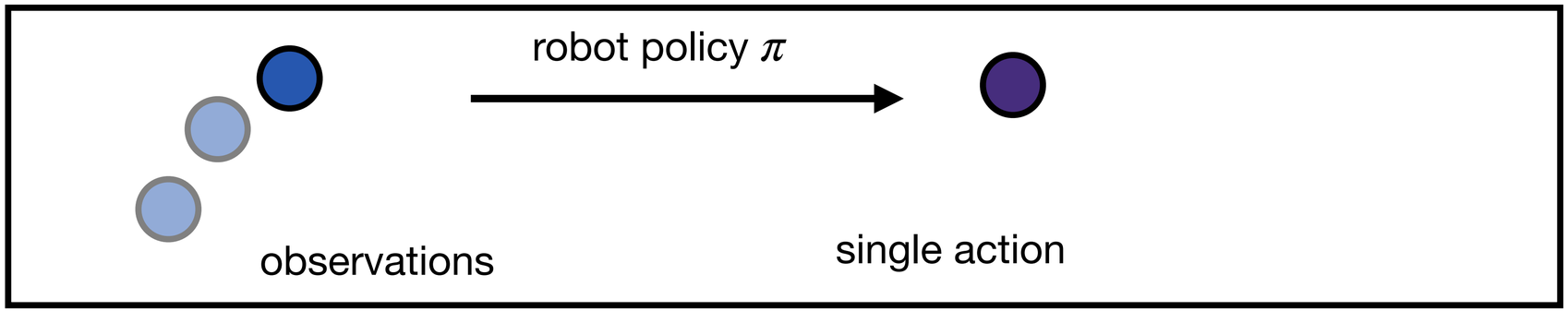
(2) Each $(\pi, f) \in \mathscr{I}$ is **smooth, deterministic, Markovian & stable***

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathscr{I}$ is **smooth, deterministic, Markovian & stable***

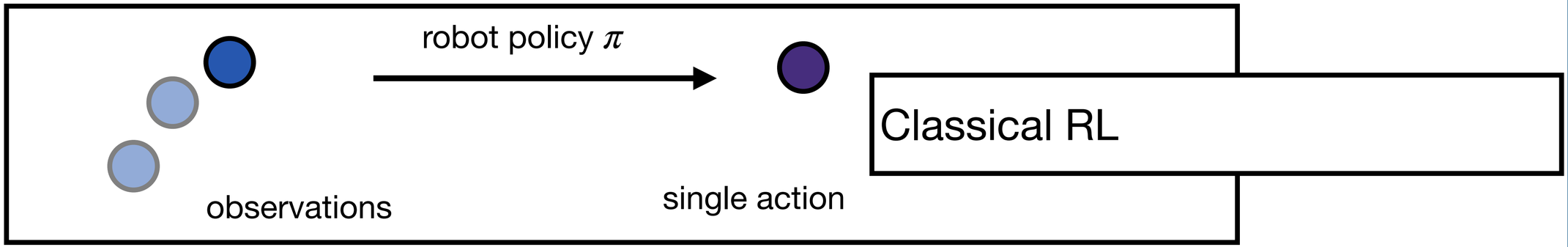


Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathscr{I}$ is **smooth, deterministic, Markovian & stable***



Continuous-action imitation is hard

Continuous-action imitation is hard

Theorem (SPJ ’25)**:** Consider the following setting

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathcal{J}$ of possible expert/dynamics

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathscr{I}$ is **smooth, deterministic, Markovian & stable***

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathscr{I}$ is **smooth, deterministic, Markovian & stable***

Informally, mistakes are forgotten exponentially quickly

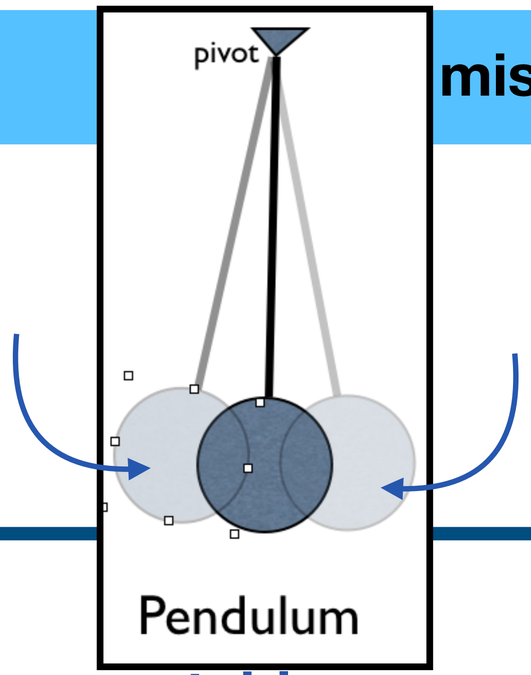
Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathcal{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathcal{I}$ is **smooth, deterministic, Markovian & stable***

mistakes are forgotten exponentially quickly



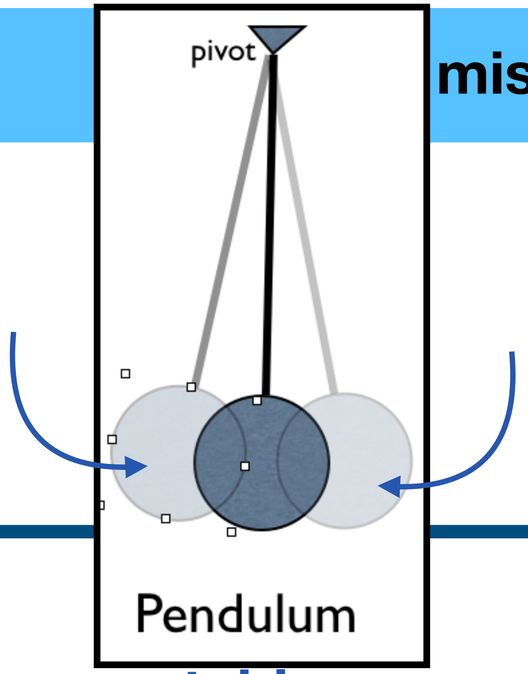
stable

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathscr{I}$ is **smooth, deterministic, Markovian & stable***



stable

mistakes are forgotten exponentially quickly

Incremental Stability, Agrachev ‘06

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathscr{I}$ is **smooth, deterministic, Markovian & stable***

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathscr{I}$ is **smooth, deterministic, Markovian & stable***

(3) The learner is **promised** that $(\pi^\star, f^\star) \in \mathscr{I}$ (well-specification)

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathscr{I}$ is **smooth, deterministic, Markovian & stable***

(3) The learner is **promised** that $(\pi^\star, f^\star) \in \mathscr{I}$ (well-specification)

Think of this as a “nice” behavior cloning instance

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathscr{I}$ is **smooth, deterministic, Markovian & stable***

(3) The learner is **promised** that $(\pi^\star, f^\star) \in \mathscr{I}$ (well-specification)

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathscr{I}$ is **smooth, deterministic, Markovian & stable***

(3) The learner is **promised** that $(\pi^\star, f^\star) \in \mathscr{I}$ (well-specification)

(4) The learner must return **smooth, Markovian, deterministic*** $\hat{\pi}$

(generalizations permitted)

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) The learner is given a set of pairs $(\pi, f) \in \mathcal{I}$ of possible expert/dynamics

(2) Each $(\pi, f) \in \mathcal{I}$ is **smooth, deterministic, Markovian & stable***

(3) The learner is **promised** that $(\pi^\star, f^\star) \in \mathcal{I}$ (well-specification)

(4) The learner must return **smooth, Markovian, deterministic*** $\hat{\pi}$

(generalizations permitted)

Markovian Matches the Expert

Continuous-action imitation is hard

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Fix any scaling law $\epsilon_n = n^{-\alpha}$. There exists a set of $(\pi, f) \in \mathcal{I}$ satisfying the **niceness** assumptions of the past slide such that

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Fix any scaling law $\epsilon_n = n^{-\alpha}$. There exists a set of $(\pi, f) \in \mathcal{I}$ satisfying the **niceness** assumptions of the past slide such that

ERM on the square loss has $L_{\mathrm{bc}}(\hat{\pi}) \lesssim n^{-\alpha}$

Continuous-action imitation is hard

Theorem (**S**PJ '25)**:** Fix any scaling law $\epsilon_n = n^{-\alpha}$. There exists a set of $(\pi, f) \in \mathcal{I}$ satisfying the **niceness** assumptions of the past slide such that

ERM on the square loss has $L_{\mathrm{bc}}(\hat{\pi}) \lesssim n^{-\alpha}$

Any algorithm satisfying (4) suffers $J(\pi^\star) - J(\hat{\pi}) \gtrsim \min\{2^H \cdot n^{-\alpha}\}$

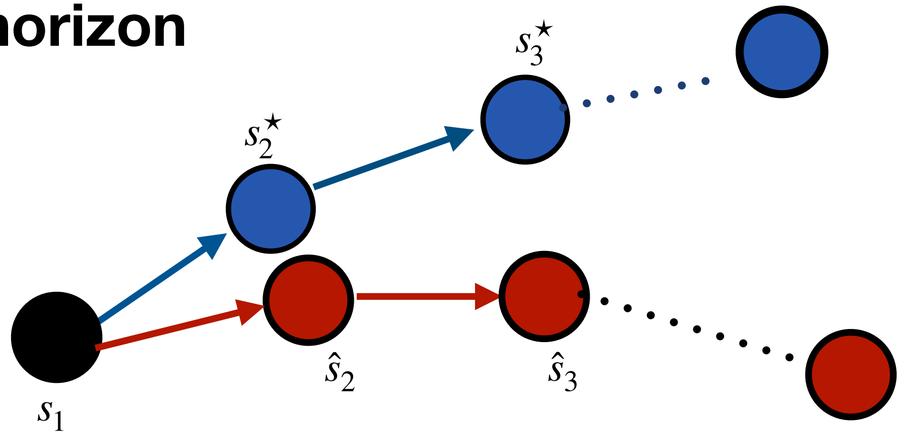
Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Fix any scaling law $\epsilon_n = n^{-\alpha}$. There exists a set of $(\pi, f) \in \mathcal{I}$ satisfying the **niceness** assumptions of the past slide such that

ERM on the square loss has $L_{\mathrm{bc}}(\hat{\pi}) \lesssim n^{-\alpha}$

Any algorithm satisfying (4) suffers $J(\pi^\star) - J(\hat{\pi}) \gtrsim \min\{2^H \cdot n^{-\alpha}\}$

1. Error compounds exponentially in problem horizon



Continuous-action imitation is hard

Theorem (**S**PJ '25)**:** Fix any scaling law $\epsilon_n = n^{-\alpha}$. There exists a set of $(\pi, f) \in \mathscr{I}$ satisfying the **niceness** assumptions of the past slide such that

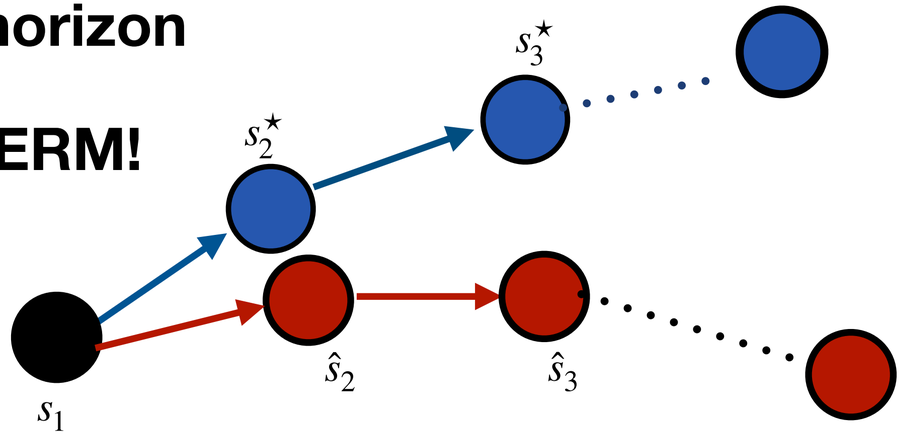
ERM on the square loss has $L_{\mathrm{bc}}(\hat{\pi}) \lesssim n^{-\alpha}$

Any algorithm satisfying (4) suffers $J(\pi^\star) - J(\hat{\pi}) \gtrsim \min\{2^H \cdot n^{-\alpha}\}$

1. Error compounds exponentially in problem horizon

2. The result applies to all algorithms, not just ERM!

(as long as policies satisfy (4)….)



Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Fix any scaling law $\epsilon_n = n^{-\alpha}$. There exists a set of $(\pi, f) \in \mathscr{I}$ satisfying the **niceness** assumptions of the past slide such that

ERM on the square loss has $L_{\mathrm{bc}}(\hat{\pi}) \lesssim n^{-\alpha}$

Any algorithm satisfying (4) suffers $J(\pi^\star) - J(\hat{\pi}) \gtrsim \min\{2^H \cdot n^{-\alpha}\}$

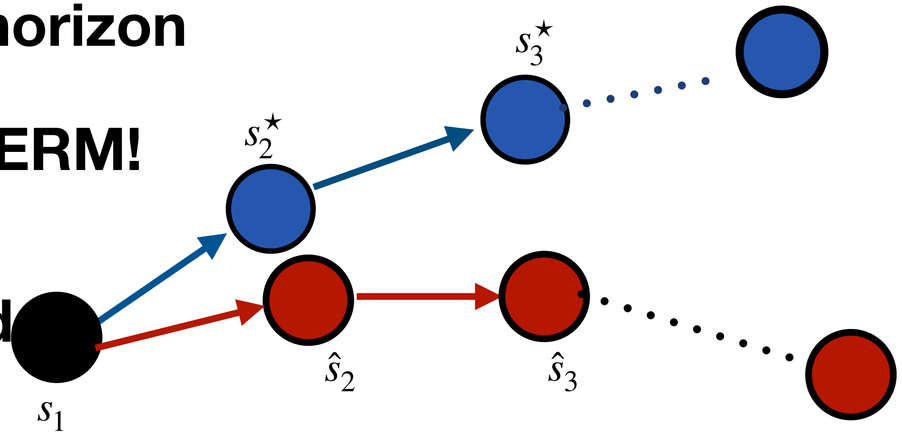
1. Error compounds exponentially in problem horizon

2. The result applies to all algorithms, not just ERM!

(as long as policies satisfy (4)….)

3. The result allows the learner to know reward

(rules out offline RL algorithms, e.g. Kumar et al ’20)



Continuous-action imitation is hard

Theorem (**S**PJ '25)**:** Fix any scaling law $\epsilon_n = n^{-\alpha}$. There exists a set of $(\pi, f) \in \mathcal{I}$ satisfying the **niceness** assumptions of the past slide such that

ERM on the square loss has $L_{\mathrm{bc}}(\hat{\pi}) \lesssim n^{-\alpha}$

Any algorithm satisfying (4) suffers $J(\pi^\star) - J(\hat{\pi}) \gtrsim \min\{2^H \cdot n^{-\alpha}\}$

Continuous-action imitation is hard

Theorem (**S**PJ '25)**:** Fix any scaling law $\epsilon_n = n^{-\alpha}$. There exists a set of $(\pi, f) \in \mathcal{I}$ satisfying the **niceness** assumptions of the past slide such that

ERM on the square loss has $L_{\mathrm{bc}}(\hat{\pi}) \lesssim n^{-\alpha}$

Any algorithm satisfying (4) suffers $J(\pi^\star) - J(\hat{\pi}) \gtrsim \min\{2^H \cdot n^{-\alpha}\}$

Compare to the 0-1 loss!

$$J(\pi^\star) - J(\hat{\pi}) \leq \mathrm{poly}(H) \cdot L_{\mathrm{bc}}(\hat{\pi})$$

(or better via *Foster, Block, Misra 2024*)

Continuous-action imitation is hard

Theorem (**S**PJ '25)**:** Fix any scaling law $\epsilon_n = n^{-\alpha}$. There exists a set of $(\pi, f) \in \mathcal{I}$ satisfying the **niceness** assumptions of the past slide such that

ERM on the square loss has $L_{\mathrm{bc}}(\hat{\pi}) \lesssim n^{-\alpha}$

Any algorithm satisfying (4) suffers $J(\pi^\star) - J(\hat{\pi}) \gtrsim \min\{2^H \cdot n^{-\alpha}\}$

Continuous-action imitation is hard

Theorem (**S**PJ ’25)**:** Fix any scaling law $\epsilon_n = n^{-\alpha}$. There exists a set of $(\pi, f) \in \mathcal{I}$ satisfying the **niceness** assumptions of the past slide such that

ERM on the square loss has $L_{\mathrm{bc}}(\hat{\pi}) \lesssim n^{-\alpha}$

Any algorithm satisfying (4) suffers $J(\pi^\star) - J(\hat{\pi}) \gtrsim \min\{2^H \cdot n^{-\alpha}\}$

A Quantitative Moravec’s Paradox!

Exponential Separation Between Continuous 🤖 and Discrete 📚

Continuous-action imitation is hard

Theorem (**S**PJ '25)**:** Fix any scaling law $\epsilon_n = n^{-\alpha}$. There exists a set of $(\pi, f) \in \mathcal{I}$ satisfying the **niceness** assumptions of the past slide such that

ERM on the square loss has $L_{\mathrm{bc}}(\hat{\pi}) \lesssim n^{-\alpha}$

Any algorithm satisfying (4) suffers $J(\pi^\star) - J(\hat{\pi}) \gtrsim \min\{2^H \cdot n^{-\alpha}\}$

But why?

Imitation Learning in Continuous Action Spaces

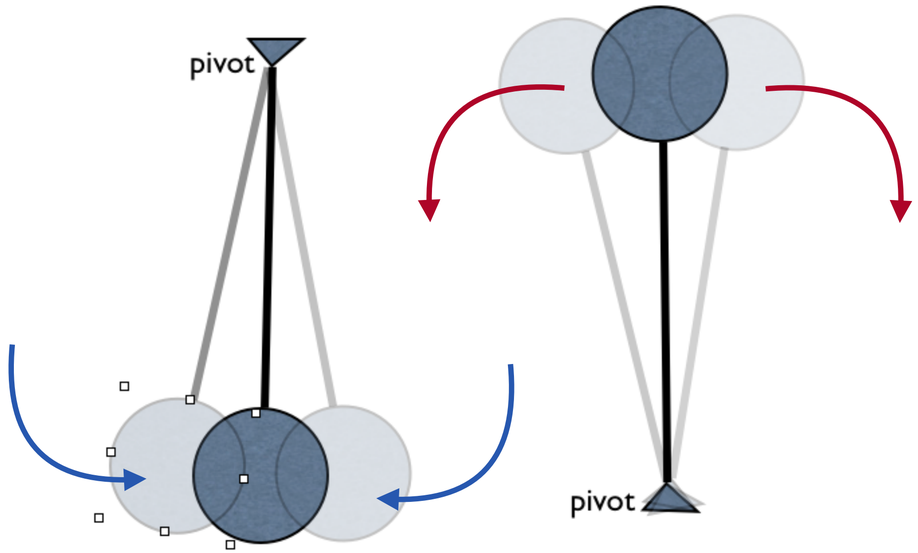
4. Control Theoretic Stability: Why Imitation is Hard

Control Theoretic Stability 🤸

Control Theoretic Stability 🤸

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Control Theoretic Stability 🤸



Pendulum

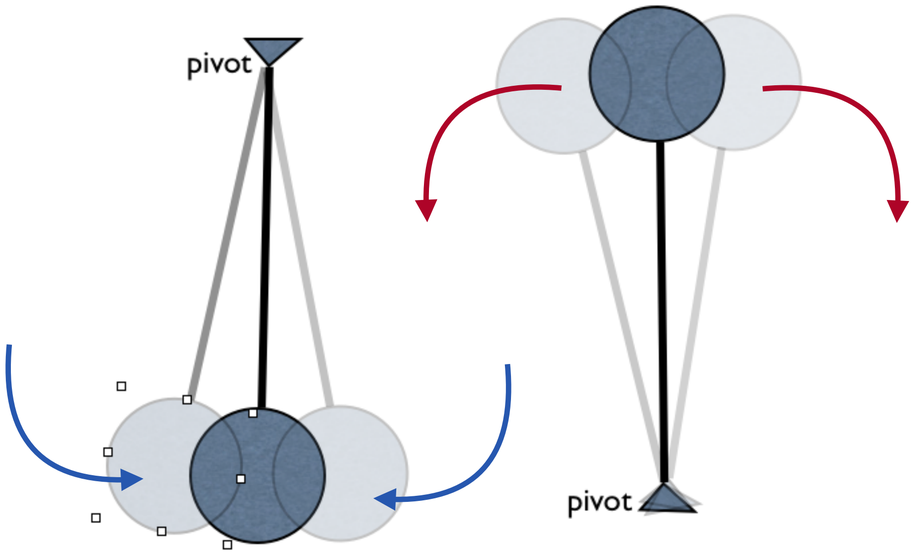
stable

Inverted Pendulum

unstable

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Control Theoretic Stability



Pendulum

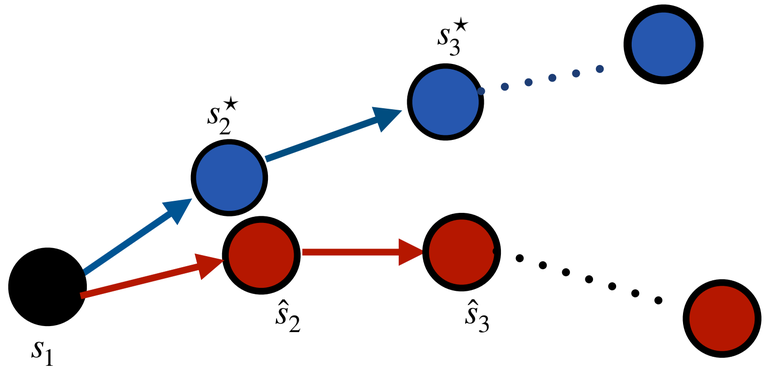
stable

Inverted Pendulum

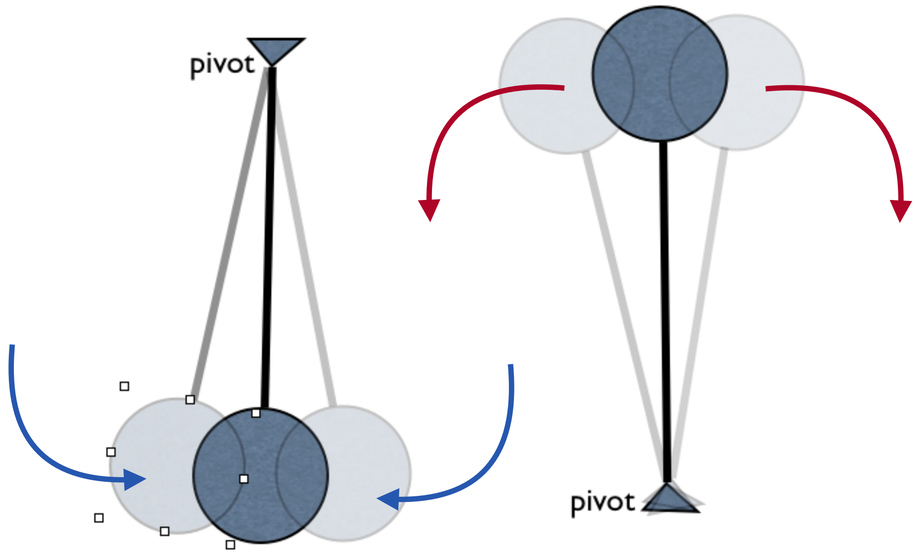
unstable

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

naturally related to compounding error.



Control Theoretic Stability 🤸



Pendulum

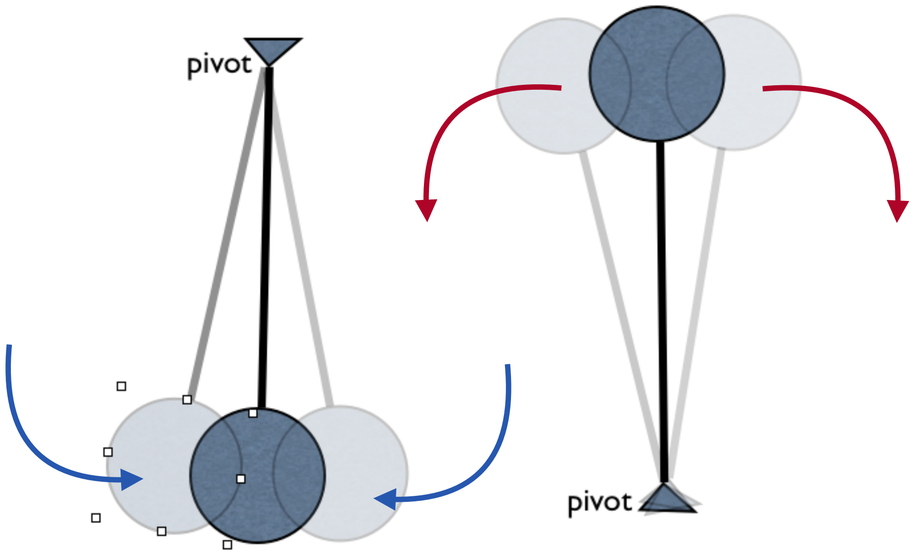
stable

Inverted Pendulum

unstable

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Control Theoretic Stability



Pendulum

stable

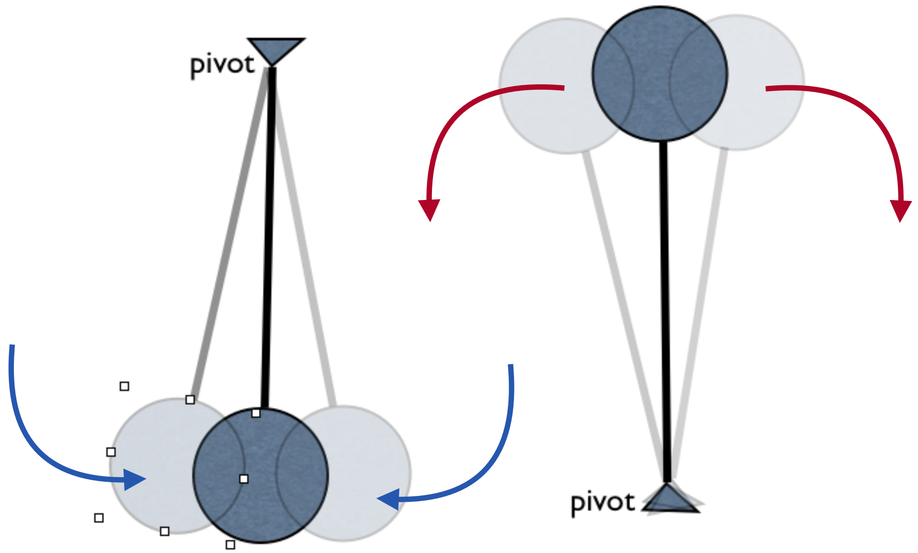
Inverted Pendulum

unstable

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Incremental Stability, Agrachev ‘06

Control Theoretic Stability



Pendulum
stable

Inverted Pendulum
unstable

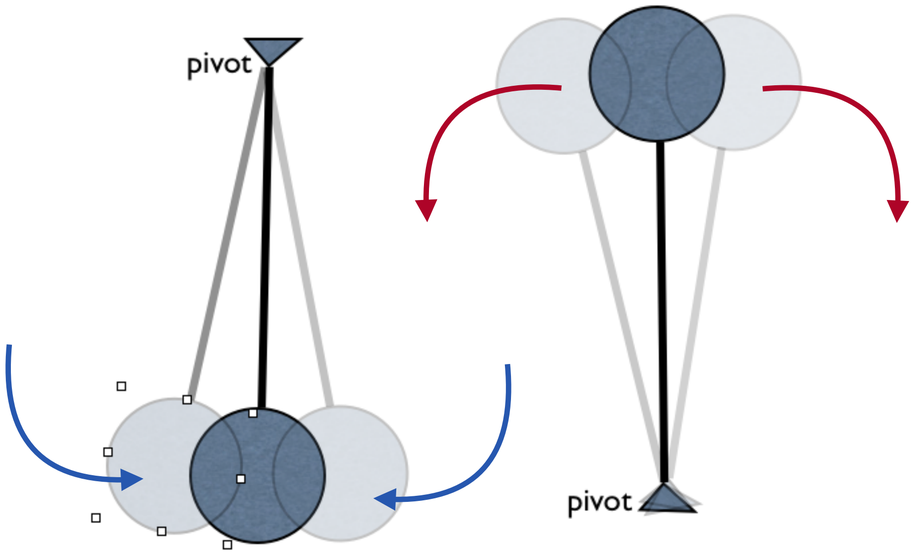
Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Incremental Stability, Agrachev '06

Definition: Dynamics $s_{t+1} = f(s_t, a_t)$ are (C, ρ) **-stable** if, for same initial condition $s_1 = s'_1$

$$\|s'_{t+1} - s'_{t+1}\| \leq C \sum_{s \leq t} \rho^{t-s} \|a_s - a'_s\|$$

Control Theoretic Stability



Pendulum
stable

Inverted Pendulum
unstable

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

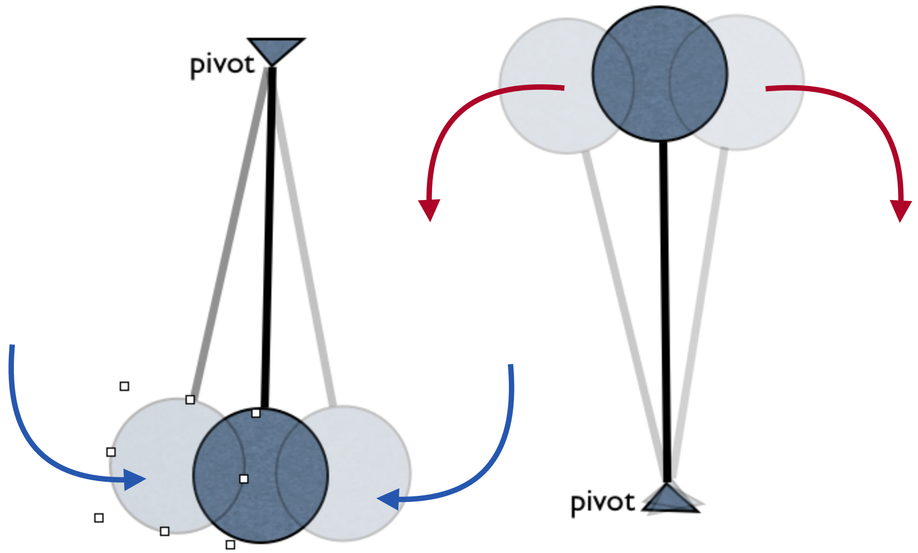
Incremental Stability, Agrachev '06

Definition: Dynamics $s_{t+1} = f(s_t, a_t)$ are (C, ρ) **-stable** if, for same initial condition $s_1 = s'_1$

$$\|s'_{t+1} - s'_{t+1}\| \leq C \sum_{s \leq t} \rho^{t-s} \|a_s - a'_s\|$$

$\rho \in (0,1)$ = **exponentially quick forgetting of mistakes**

Control Theoretic Stability



Pendulum
stable

Inverted Pendulum
unstable

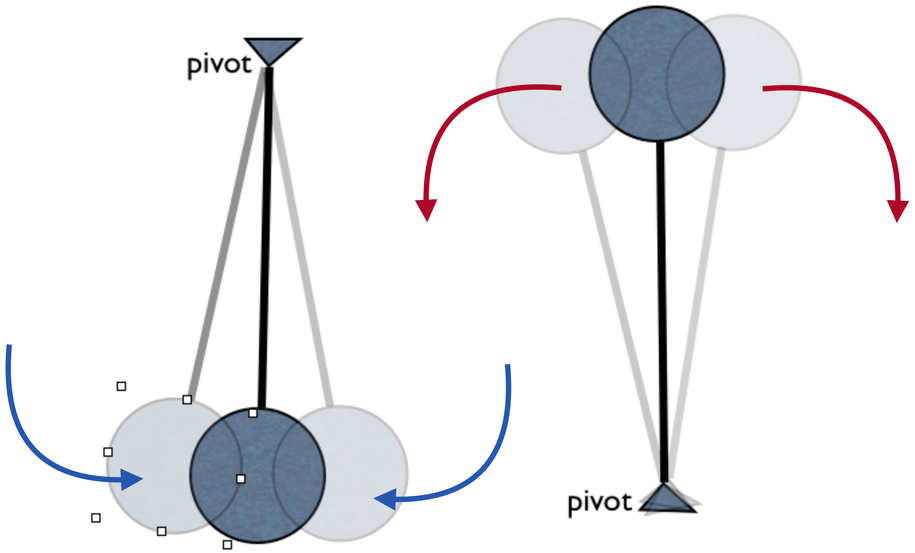
Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Incremental Stability, Agrachev ‘06

Definition: Dynamics $s_{t+1} = f(s_t, a_t)$ are (C, ρ) **-stable** if, for same initial condition $s_1 = s'_1$

$$\|s'_{t+1} - s'_{t+1}\| \leq C \sum_{s \leq t} \rho^{t-s} \|a_s - a'_s\|$$

Control Theoretic Stability



Pendulum
stable

Inverted Pendulum
unstable

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

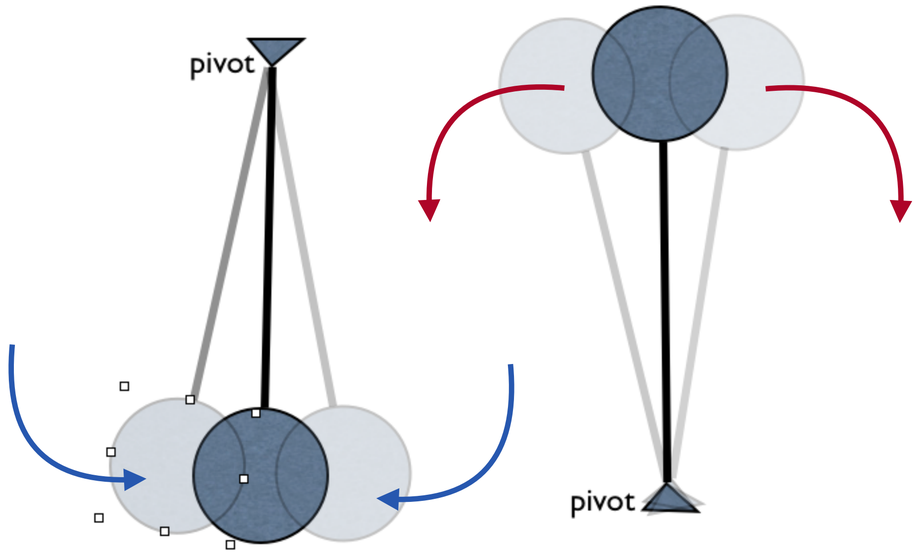
Incremental Stability, Agrachev ‘06

Definition: Dynamics $s_{t+1} = f(s_t, a_t)$ are (C, ρ) **-stable** if, for same initial condition $s_1 = s'_1$

$$\|s'_{t+1} - s'_{t+1}\| \leq C \sum_{s \leq t} \rho^{t-s} \|a_s - a'_s\|$$

Naively, stability should prevent compounding error

Control Theoretic Stability



Pendulum
stable

Inverted Pendulum
unstable

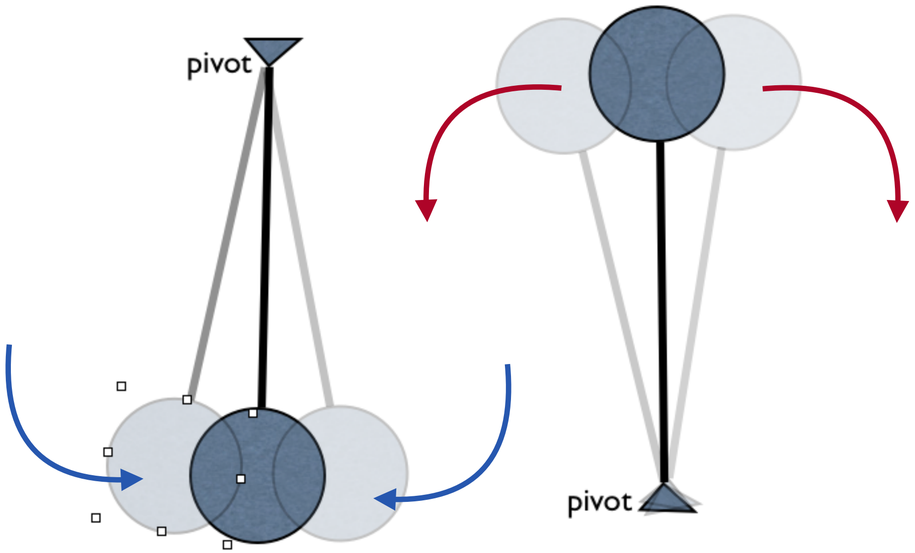
Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Incremental Stability, Agrachev ‘06

Definition: Dynamics $s_{t+1} = f(s_t, a_t)$ are (C, ρ) **-stable** if, for same initial condition $s_1 = s'_1$

$$\|s'_{t+1} - s'_{t+1}\| \leq C \sum_{s \leq t} \rho^{t-s} \|a_s - a'_s\|$$

Control Theoretic Stability



Pendulum
stable

Inverted Pendulum
unstable

Definition (Informal)**: A dynamical system** is said to be **stable** if it has limited sensitivity to perturbations of input.

Incremental Stability, Agrachev '06

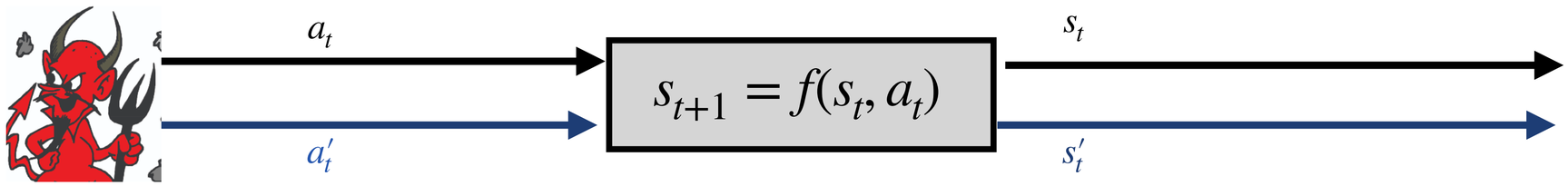
Definition: Dynamics $s_{t+1} = f(s_t, a_t)$ are (C, ρ) **-stable** if, for same initial condition $s_1 = s'_1$

$$\|s'_{t+1} - s'_{t+1}\| \leq C \sum_{s \leq t} \rho^{t-s} \|a_s - a'_s\|$$

Why does this not prevent exponential compounding error?

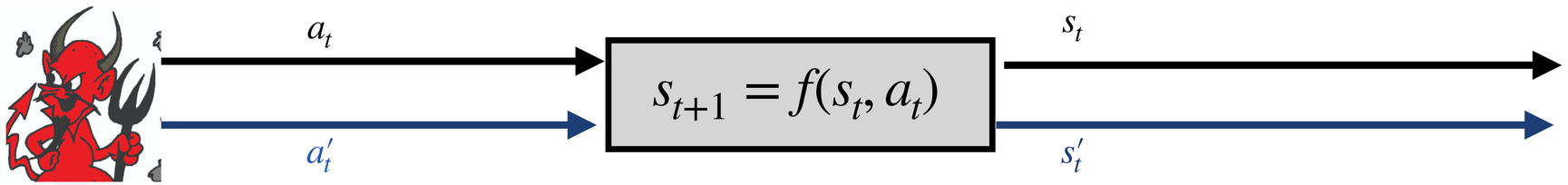
Scenarios of (Incremental) **Stability**

Scenarios of (Incremental) **Stability**

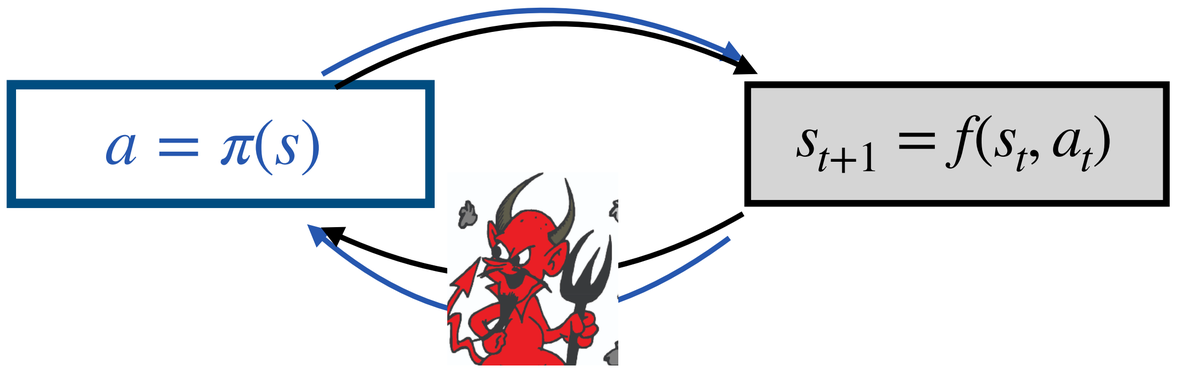


open-loop stable

Scenarios of (Incremental) **Stability**

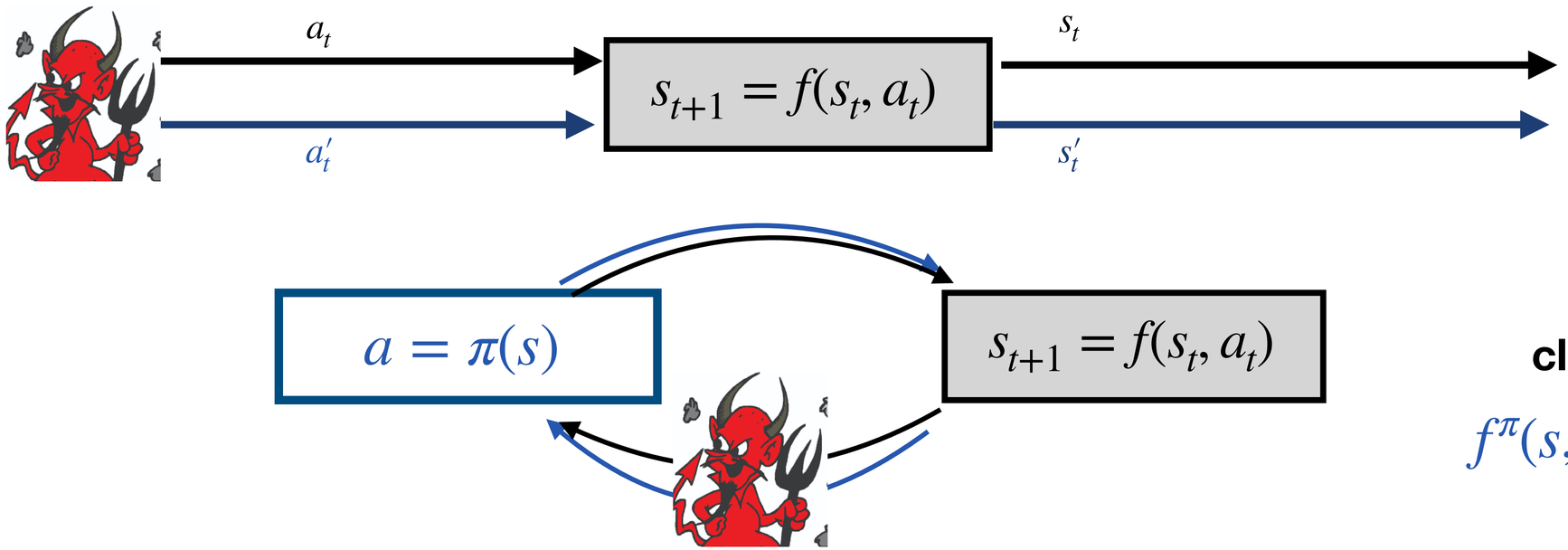


open-loop stable



closed-loop stable

Scenarios of (Incremental) **Stability**

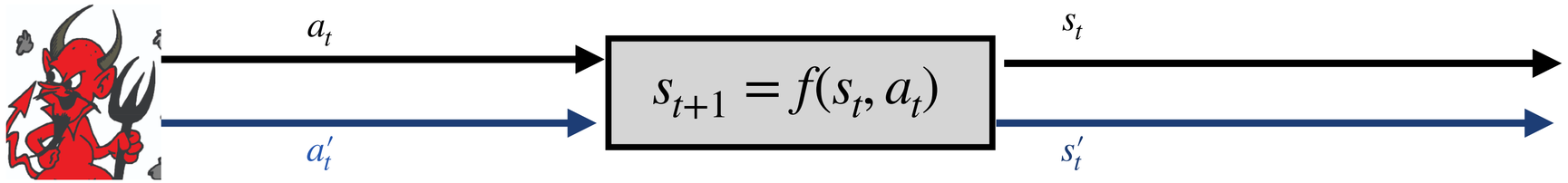


open-loop stable

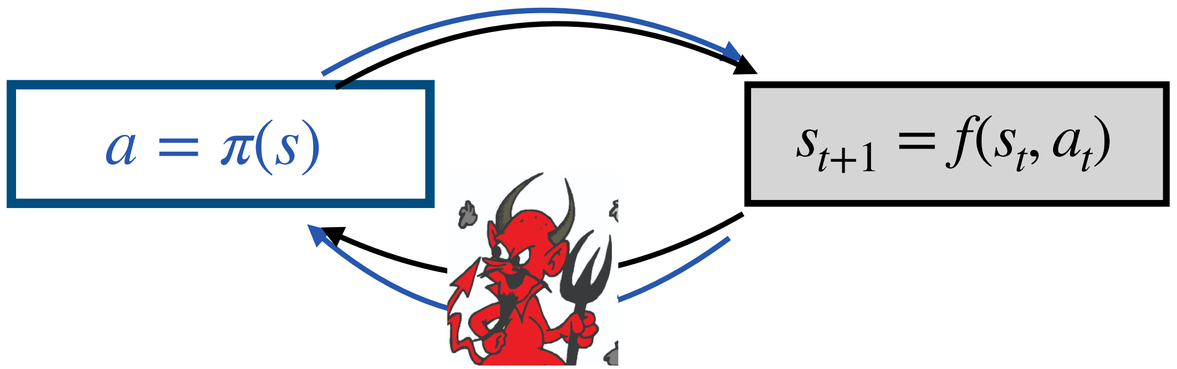
closed-loop stable

$$f^{\pi}(s, \Delta a) = f(s, \pi(s) + \Delta a)$$

Scenarios of (Incremental) **Stability**

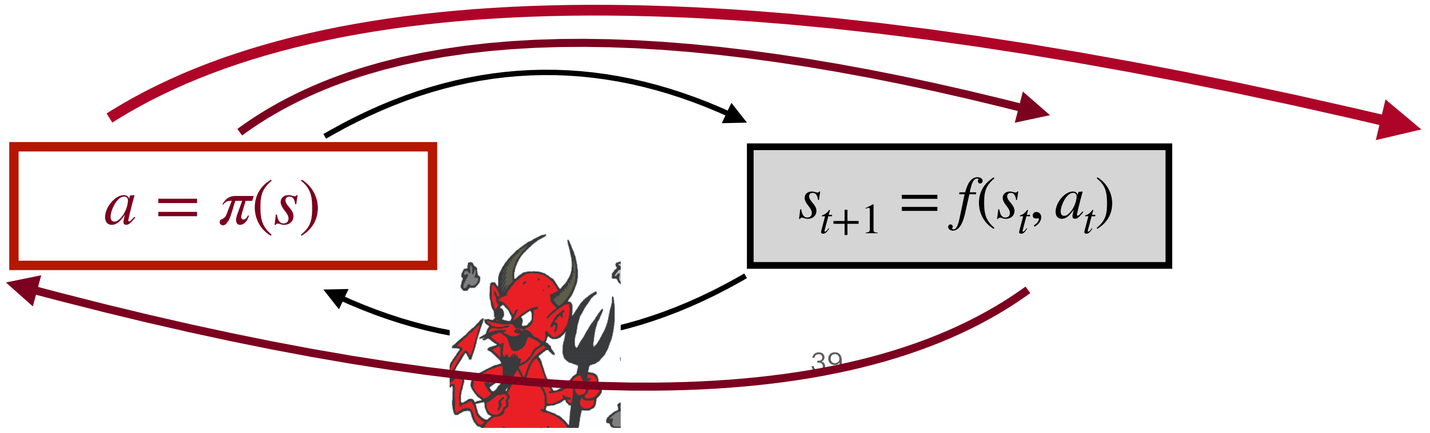


open-loop stable



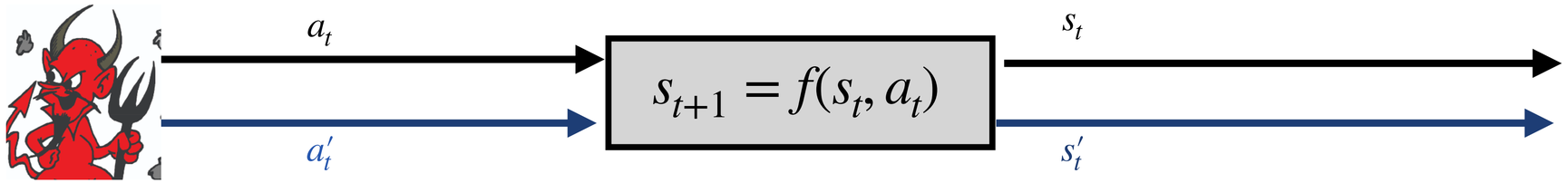
closed-loop stable

$$f^{\pi}(s, \Delta a) = f(s, \pi(s) + \Delta a)$$

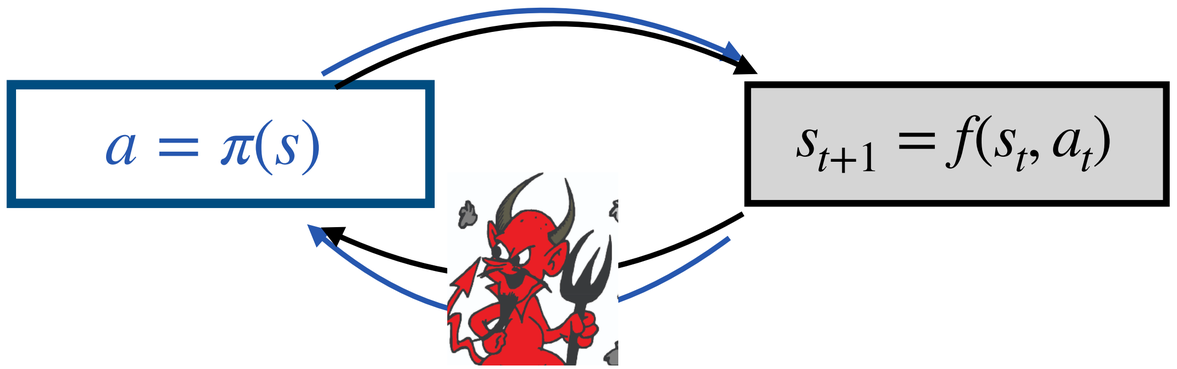


closed-loop unstable

Scenarios of (Incremental) Stability

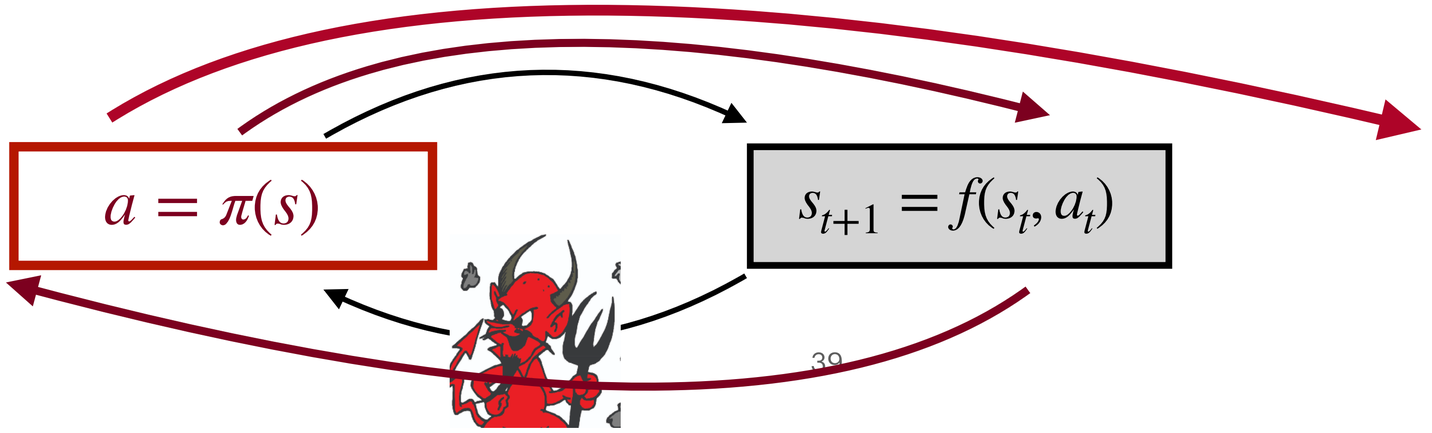


open-loop stable



closed-loop stable

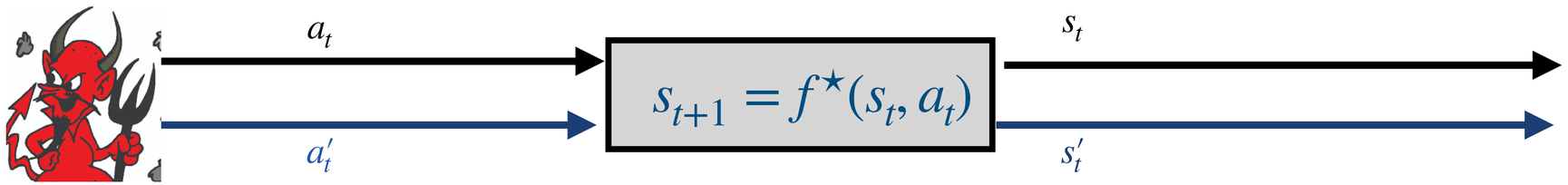
$$f^{\pi}(s, \Delta a) = f(s, \pi(s) + \Delta a)$$



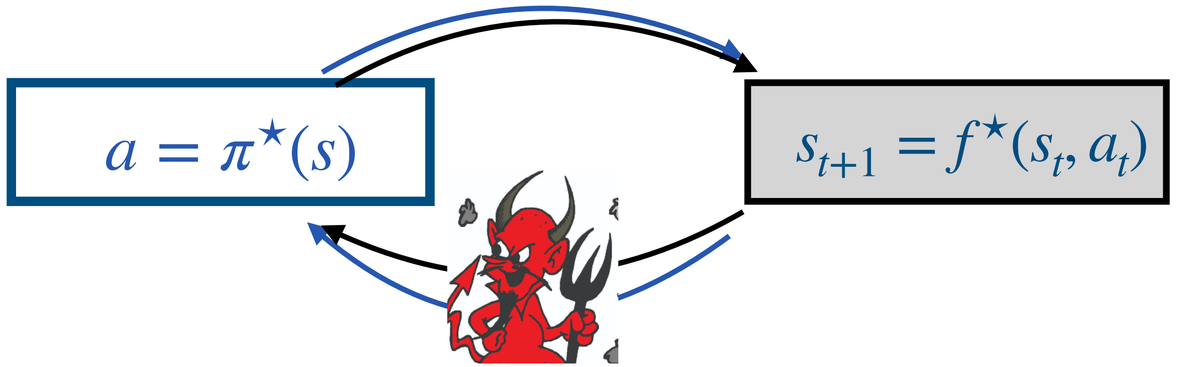
closed-loop unstable

open-loop unstable … see later

Scenarios of (Incremental) **Stability**



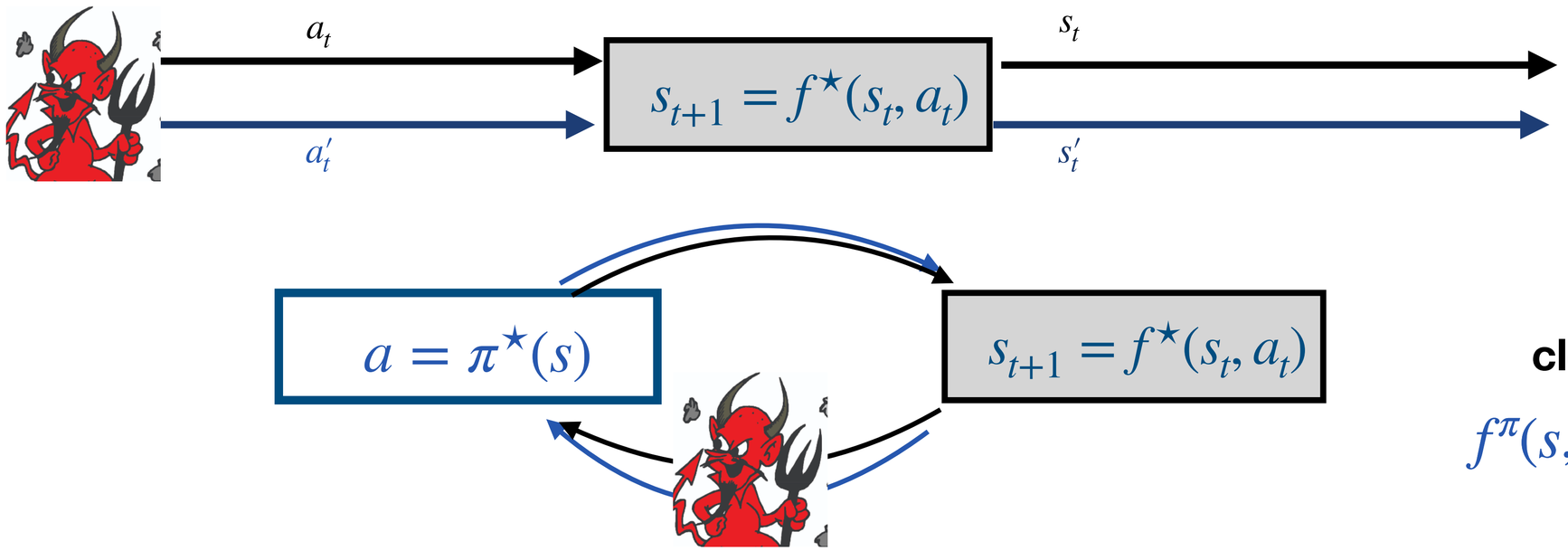
open-loop stable



closed-loop stable

$$f^\pi(s, \Delta a) = f(s, \pi(s) + \Delta a)$$

Scenarios of (Incremental) **Stability**



open-loop stable

closed-loop stable

$$f^\pi(s, \Delta a) = f(s, \pi(s) + \Delta a)$$

(2) Learner **promised** $(\pi^\star, f^\star) \in \mathcal{I}$ is **open & closed-loop stable***

Key Insight

Key Insight

Just because $(\pi^\star, f^\star) \in \mathcal{J}$ is open & closed-loop stable….

Key Insight

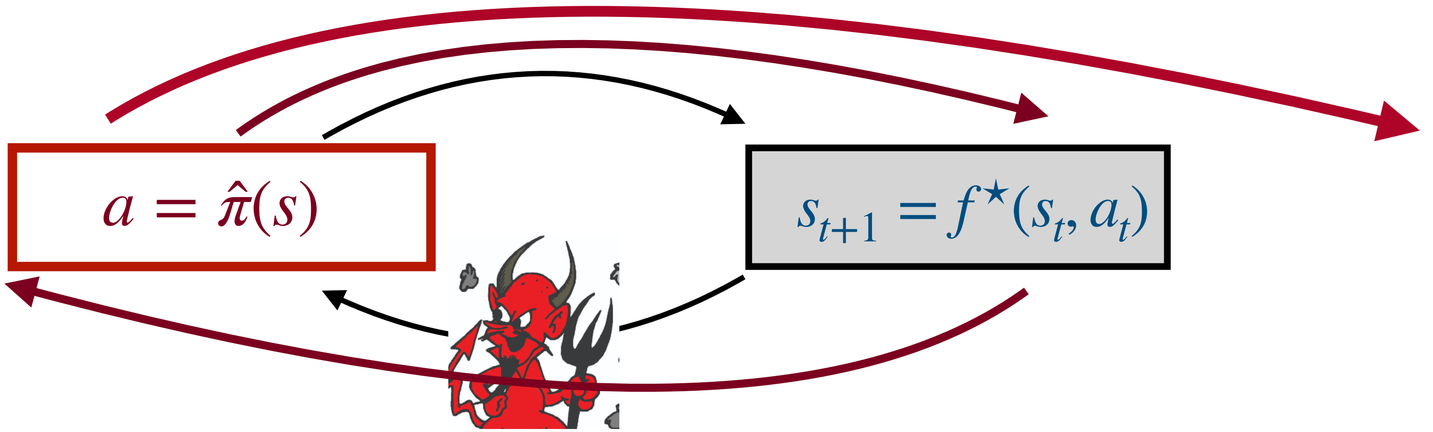
Just because $(\pi^\star, f^\star) \in \mathcal{J}$ is open & closed-loop stable….

Does not mean that $(\hat{\pi}, f^\star)$ is closed loop stable.

Key Insight

Just because $(\pi^\star, f^\star) \in \mathcal{J}$ is open & closed-loop stable….

Does not mean that $(\hat{\pi}, f^\star)$ is closed loop stable.

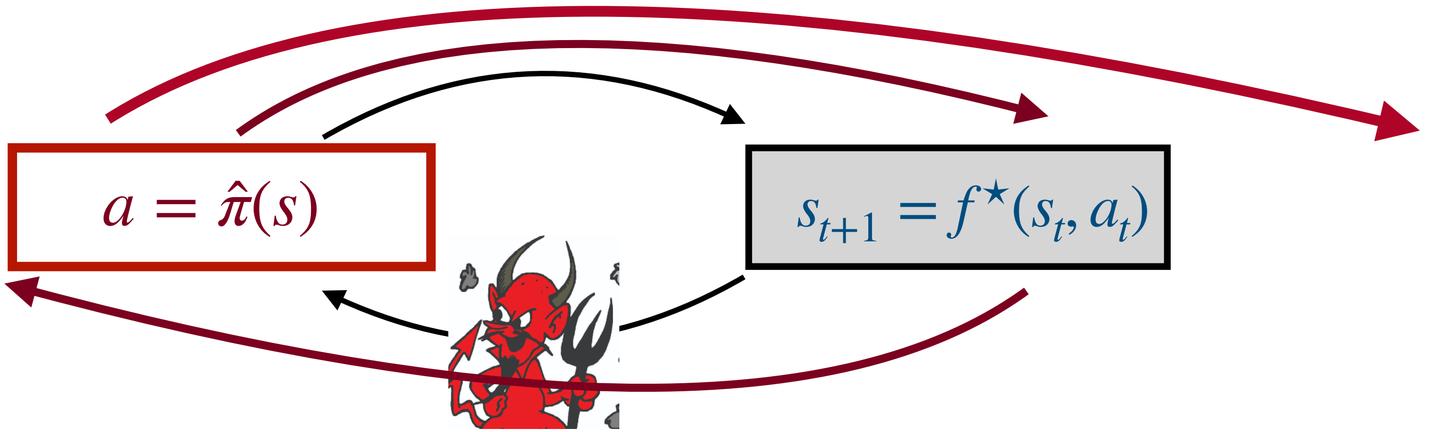


closed-loop unstable

Key Insight

Just because $(\pi^\star, f^\star) \in \mathcal{J}$ is open & closed-loop stable….

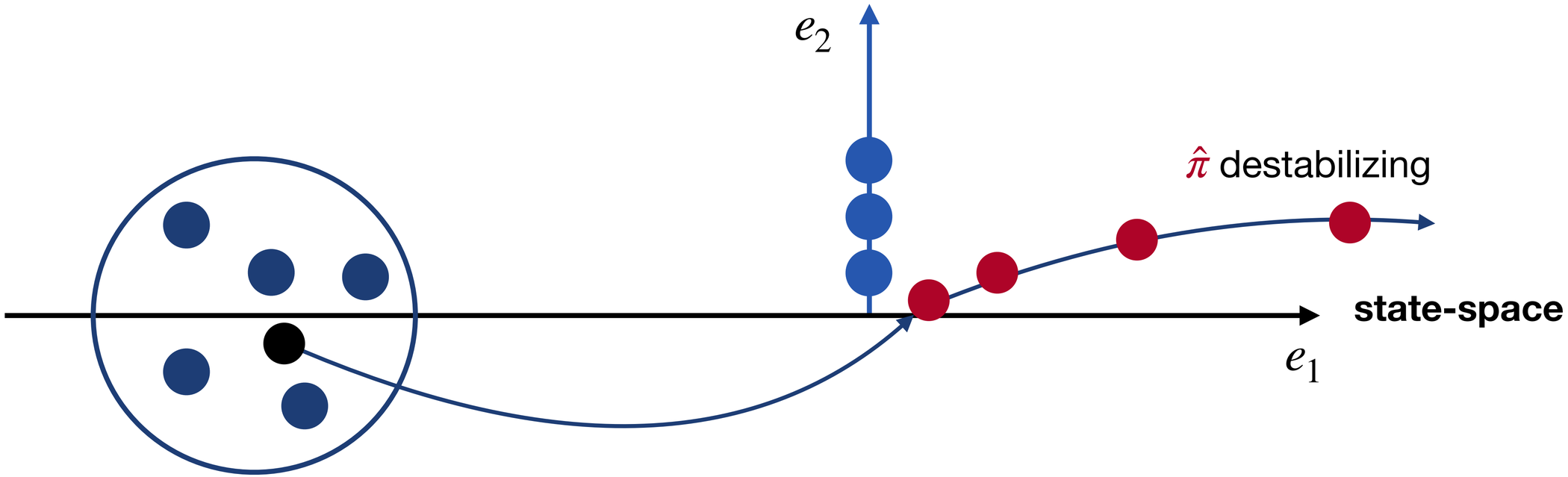
Does not mean that $(\hat{\pi}, f^\star)$ is closed loop stable.



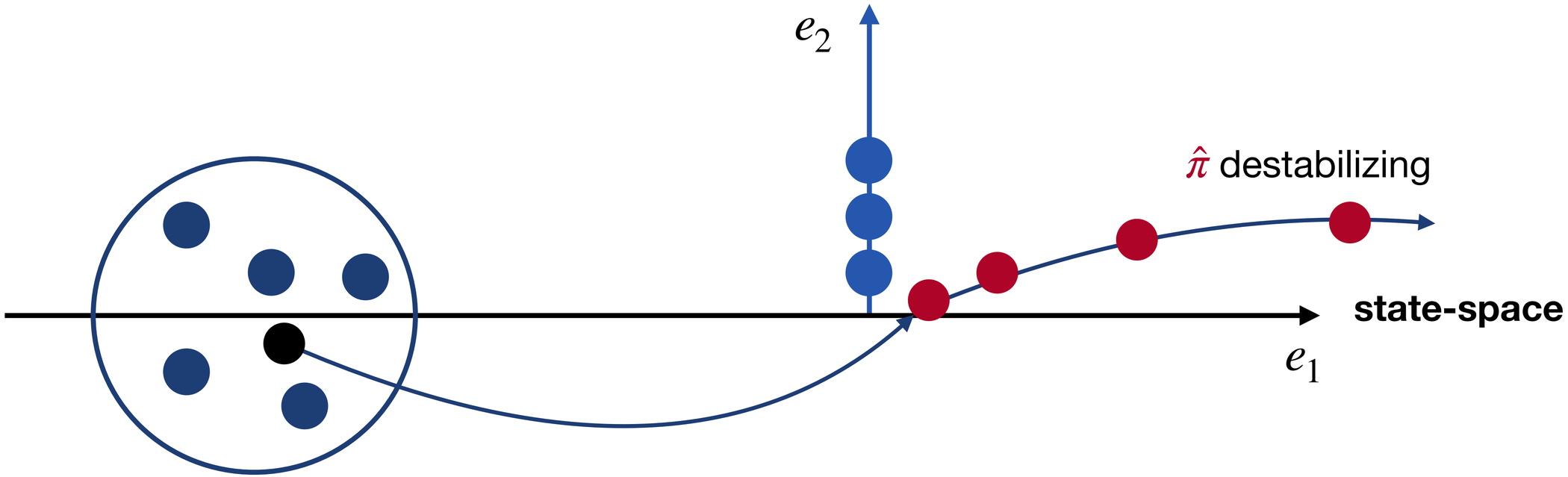
closed-loop unstable

Because $f^\star$ is unknown, it may be impossible to ensure $(\hat{\pi}, f^\star)$ is closed-loop stable

Proof Intuition



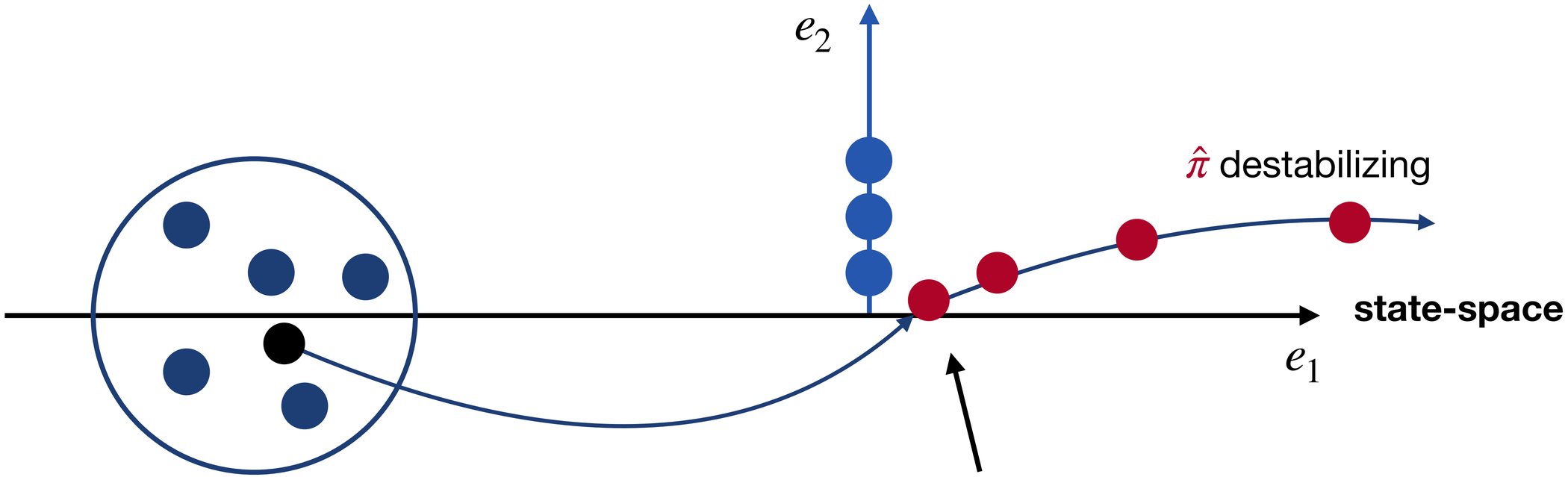
Proof Intuition



Initial Mistake

(theory of non-parametric statistics)

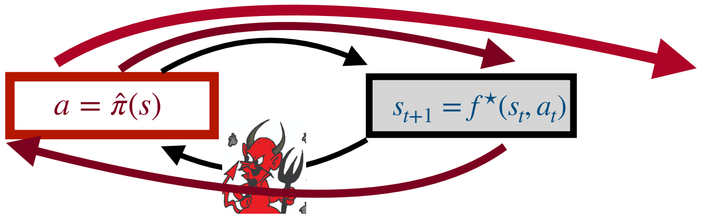
Proof Intuition



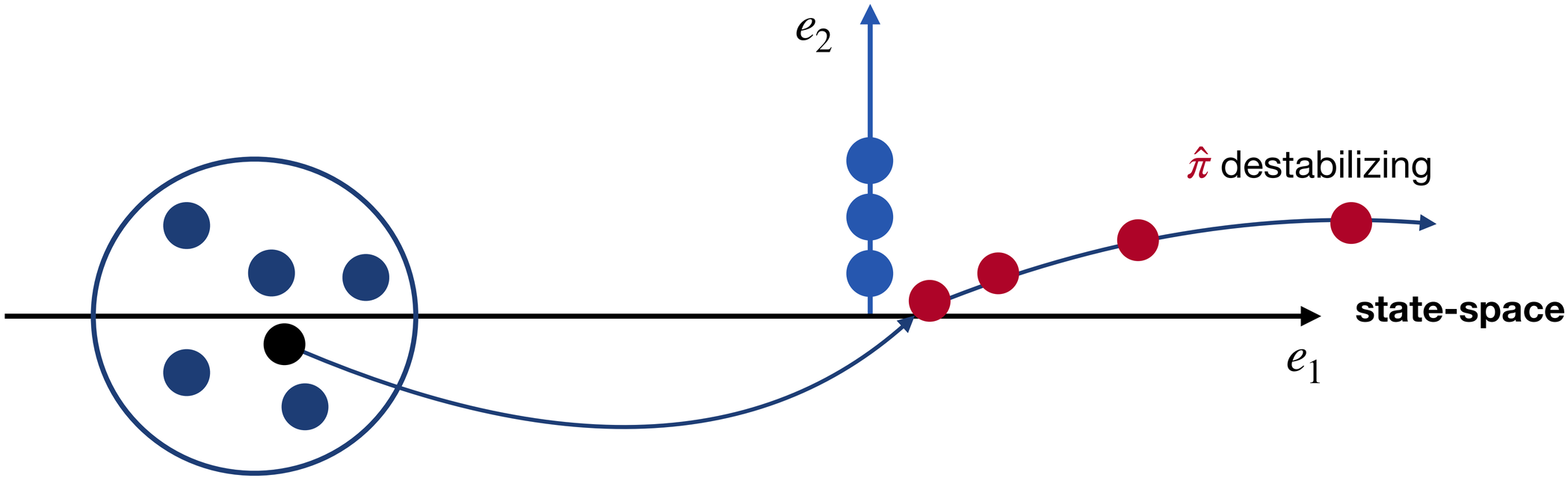
Initial Mistake

(theory of non-parametric statistics)

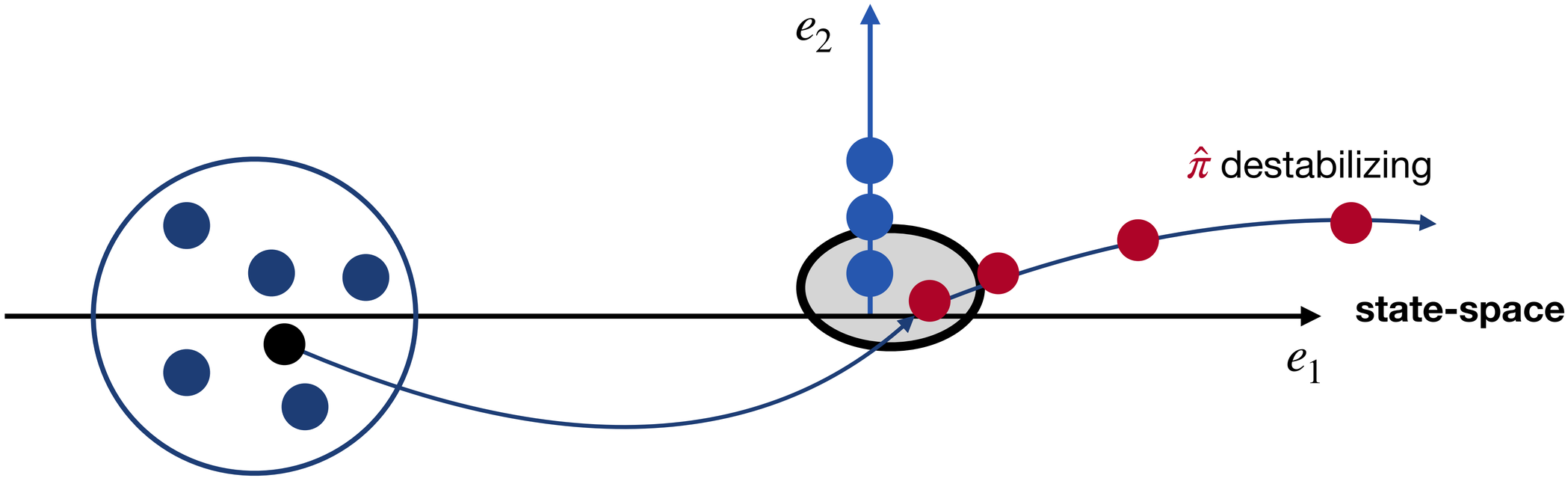
Compounded due to closed-loop instability

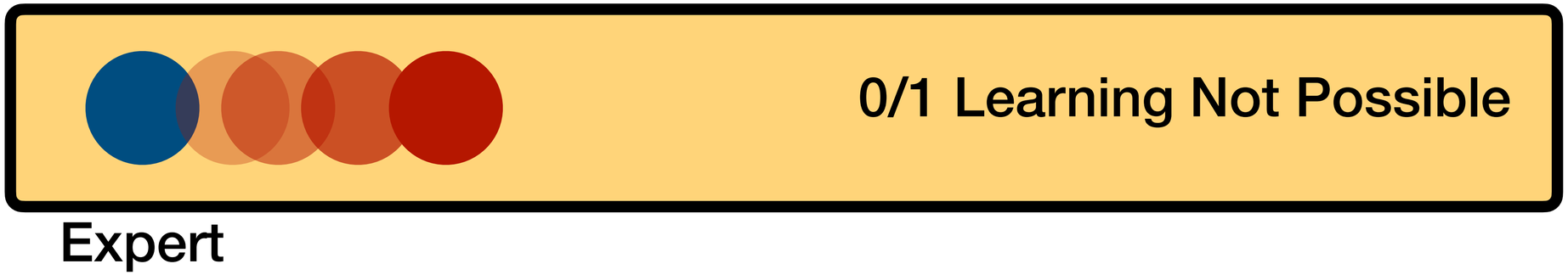


Crucial: Distance-Wise Error in Continuous Actions!



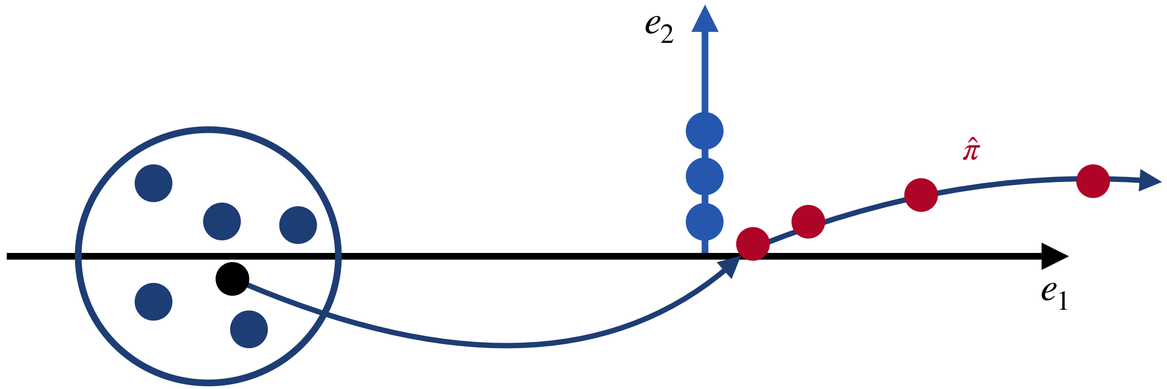
Crucial: Distance-Wise Error in Continuous Actions!





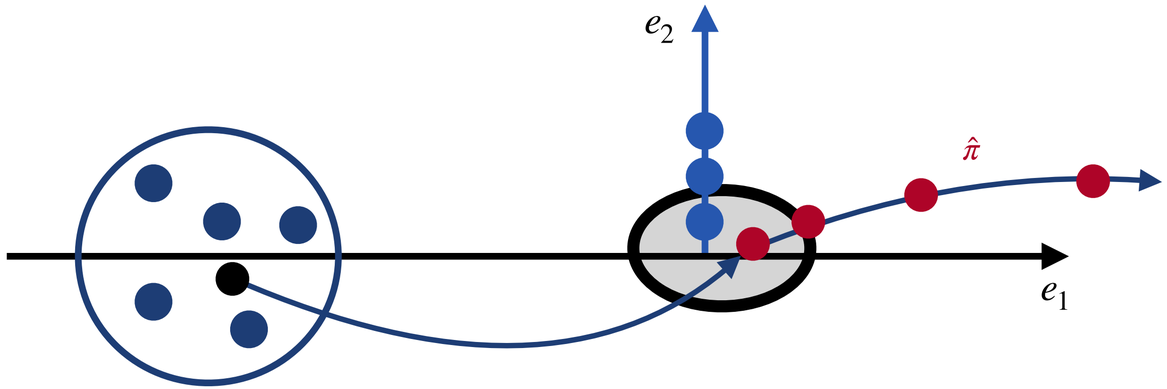
Summary: The Physical World 🤖 v.s. The Discrete World 📚

Summary: The Physical World 🤖 v.s. The Discrete World 📚

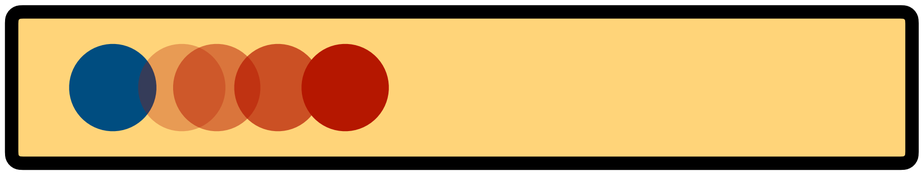


Learned policies cannot both follow the expert and stabilize unknown dynamics

Summary: The Physical World 🤖 v.s. The Discrete World 📚

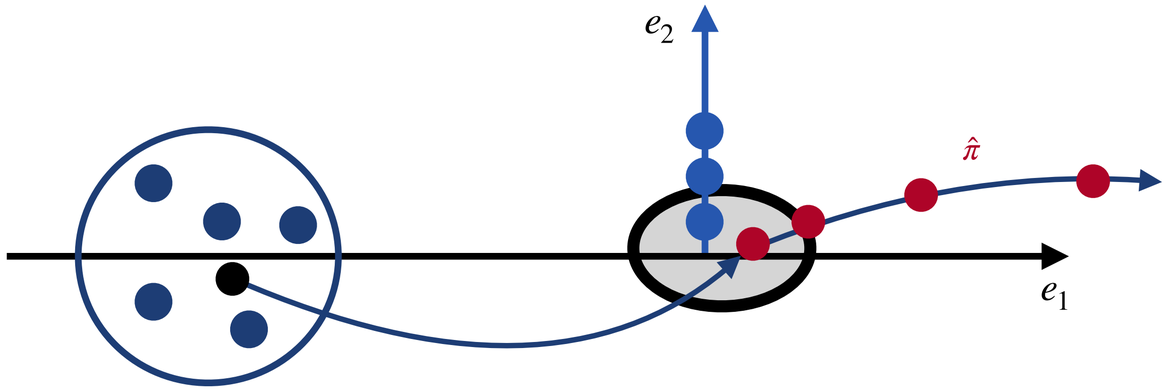


Learned policies cannot both follow the expert and stabilize unknown dynamics

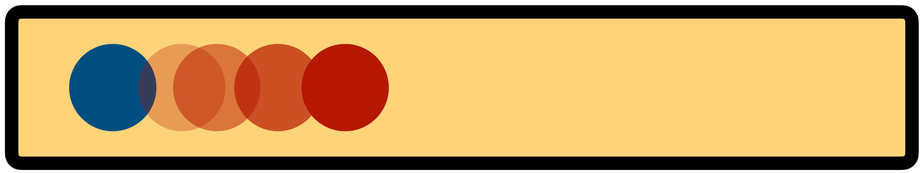


Inexactness of Learning Seeds
Compounding Error

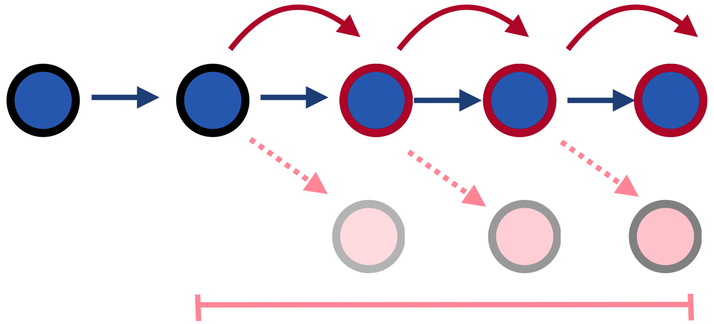
Summary: The Physical World 🤖 v.s. The Discrete World 📚



Learned policies cannot both follow the expert and stabilize unknown dynamics



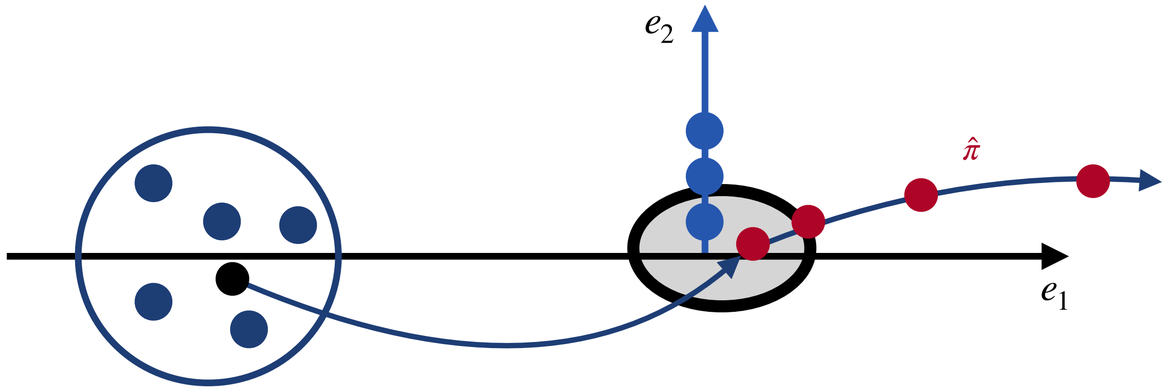
Inexactness of Learning Seeds
Compounding Error



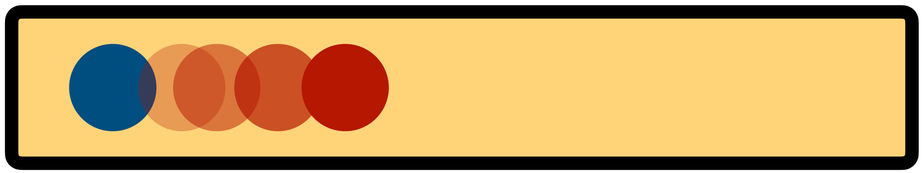
Compare to the 0-1 loss!

$$J(\pi^\star) - J(\hat{\pi}) \leq \text{poly}(H) \cdot L_{\text{bc}}(\hat{\pi})$$

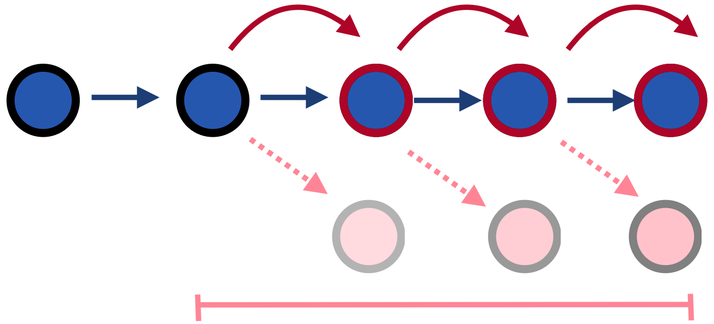
Summary: The Physical World 🤖 v.s. The Discrete World 📚



Learned policies cannot both follow the expert and stabilize unknown dynamics



Inexactness of Learning Seeds
Compounding Error

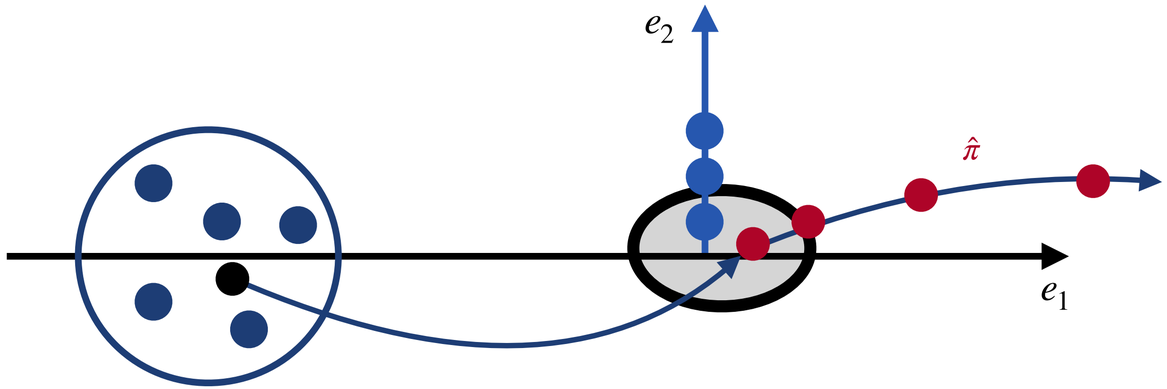


Compare to the 0-1 loss!

$$J(\pi^\star) - J(\hat{\pi}) \leq \mathrm{poly}(H) \cdot L_{\mathrm{bc}}(\hat{\pi})$$

Theory Note: L_{IPM} **(**Swami '21**) is scaled by Lipschitz Constant of Q-functions =** $\exp(\Omega(H))$**.**

Summary: The Physical World 🤖 v.s. The Discrete World 📚



Learned policies cannot both follow the expert and stabilize unknown dynamics

Is there a loophole to this Quantitative Moravec's Paradox?

Imitation Learning in Continuous Action Spaces

5. Intervention: Action Chunking and Exploratory Data

Motivating Action Chunking

Theorem (**S**PJ '25)**:** Consider the following setting

(1) There learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) There learner is **promised** that each $(\pi, f) \in \mathscr{I}$ is **smooth & stable***

(3) There learner is **promised** that $(\pi^\star, f^\star) \in \mathscr{I}$ (realizability)

Motivating Action Chunking

Theorem (**S**PJ ’25)**:** Consider the following setting

(1) There learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) There learner is **promised** that each $(\pi, f) \in \mathscr{I}$ is **smooth & stable***

(3) There learner is **promised** that $(\pi^\star, f^\star) \in \mathscr{I}$ (realizability)

(4) The learner must return **smooth, Markovian, deterministic*** $\hat{\pi}$

(generalizations permitted)

Motivating Action Chunking

Theorem (**S**PJ ’25)**:** Consider the following setting

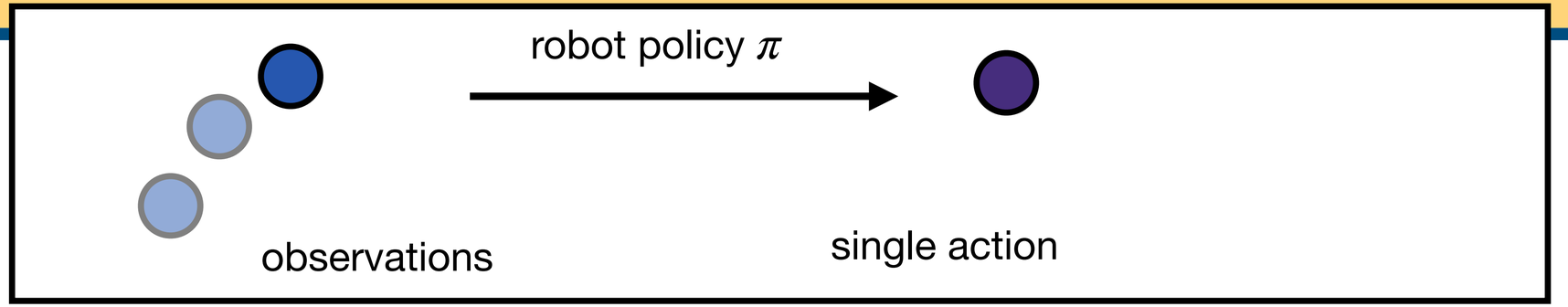
(1) There learner is given a set of pairs $(\pi, f) \in \mathscr{I}$ of possible expert/dynamics

(2) There learner is **promised** that each $(\pi, f) \in \mathscr{I}$ is **smooth & stable***

(3) There learner is **promised** that $(\pi^\star, f^\star) \in \mathscr{I}$ (realizability)

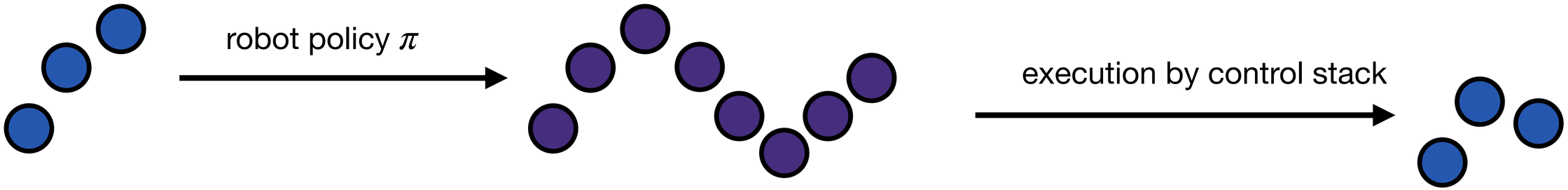
(4) The learner must return **smooth, Markovian, deterministic*** $\hat{\pi}$

(generalizations permitted)



Action Chunking

(Zhao et al ’23, Chi et. al ’23)

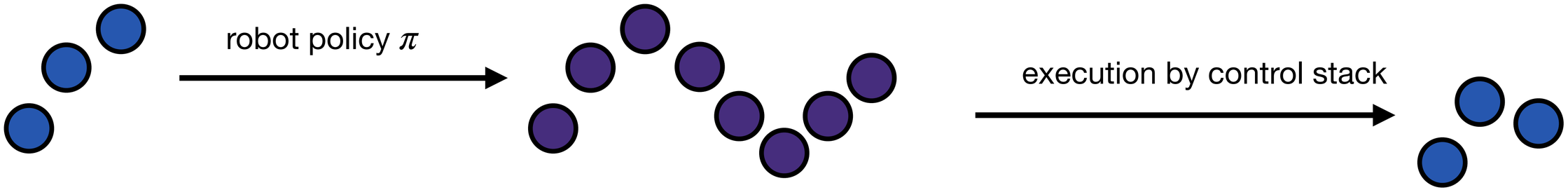


high-dimensional observations $o_{t-h:t}$

sequence, or ‘**chunk**’ of actions $a_{t:t-h+1}$

Action Chunking

(Zhao et al ’23, Chi et. al ’23)



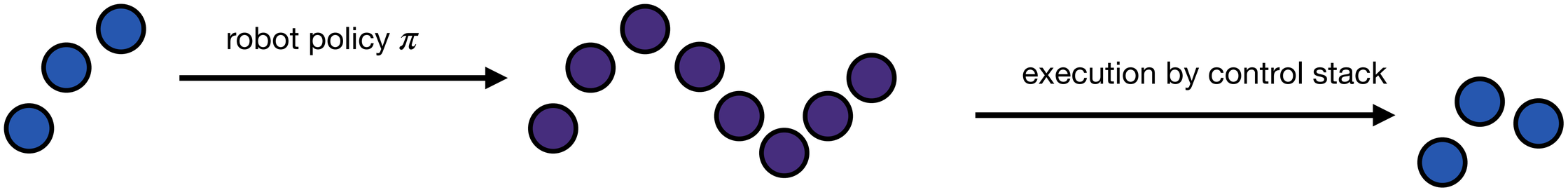
high-dimensional observations $o_{t-h:t}$

sequence, or ‘**chunk**’ of actions $a_{t:t-h+1}$

Action Chunking is a Foundational Innovation

Action Chunking

(Zhao et al ’23, Chi et. al ’23)

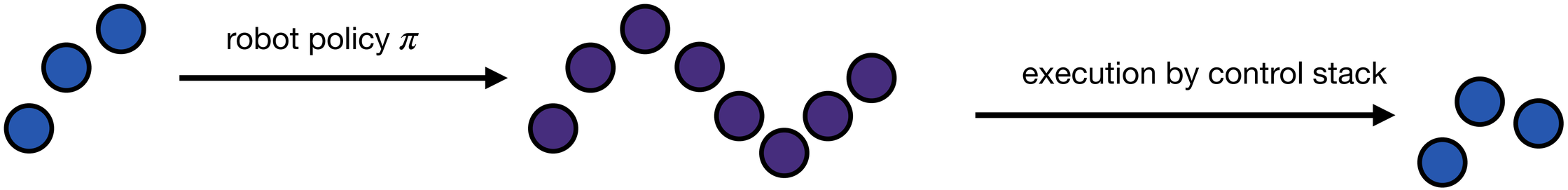


high-dimensional observations $o_{t-h:t}$

sequence, or ‘**chunk**’ of actions $a_{t:t-h+1}$

Action Chunking

(Zhao et al ’23, Chi et. al ’23)



high-dimensional observations $o_{t-h:t}$

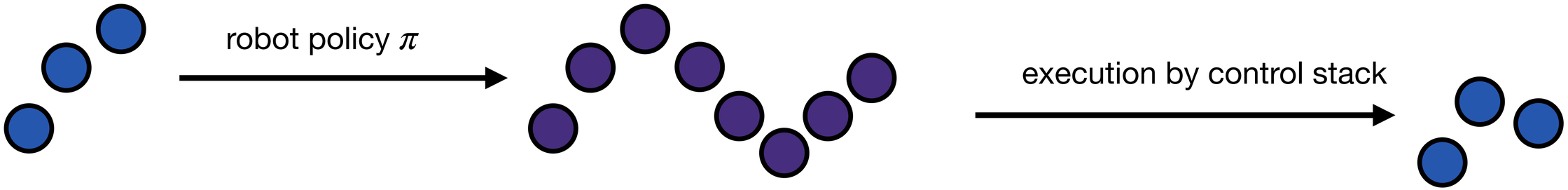
sequence, or ‘**chunk**’ of actions $a_{t:t-h+1}$

actions correlated across time

(Non-Markov!)

Action Chunking

(Zhao et al ’23, Chi et. al ’23)



high-dimensional observations $o_{t-h:t}$

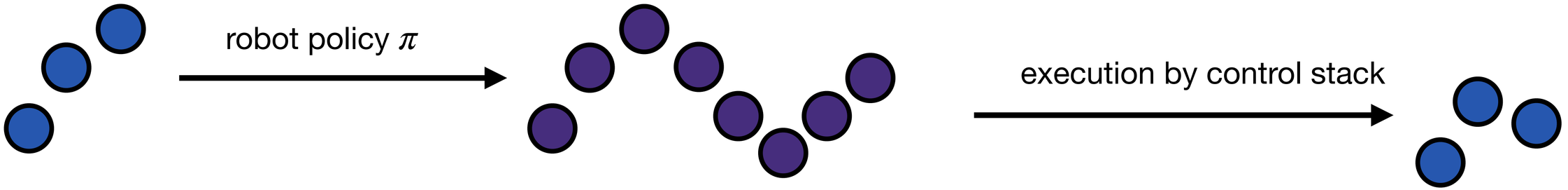
sequence, or ‘**chunk**’ of actions $a_{t:t-h+1}$

actions correlated across time

(Non-Markov!)

Action Chunking

(Zhao et al ’23, Chi et. al ’23)



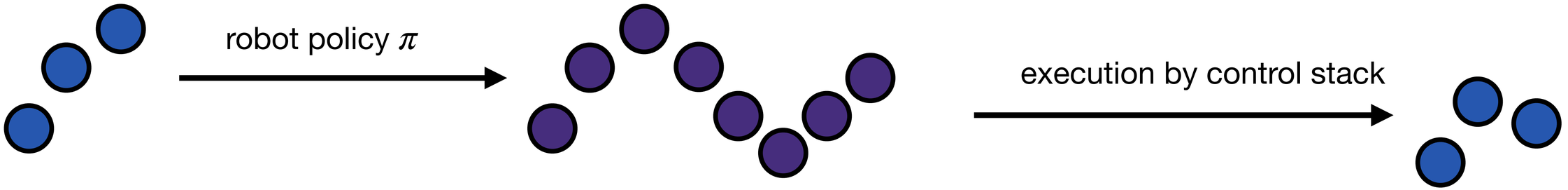
high-dimensional observations $o_{t-h:t}$

sequence, or ‘**chunk**’ of actions $a_{t:t-h+1}$

Define $L_{\text{bc,chunk}}(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=1}^{H}\left(\sum_{i=0}^{k-1}\|\pi(s,i) - a_{t+i}\|^2\right)\right]$

Action Chunking

(Zhao et al ’23, Chi et. al ’23)



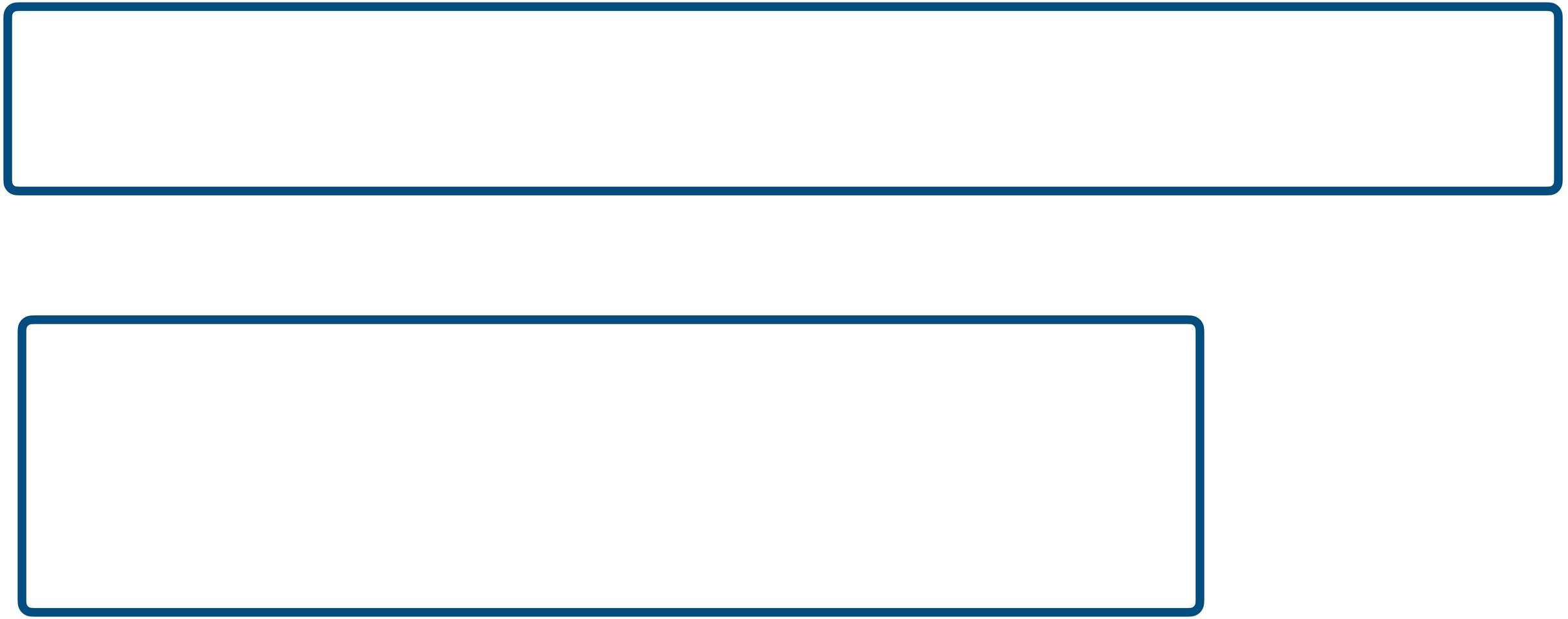
high-dimensional observations $o_{t-h:t}$

sequence, or ‘**chunk**’ of actions $a_{t:t-h+1}$

Define $L_{\text{bc,chunk}}(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=1}^{H}\left(\sum_{i=0}^{k-1}\|\pi(s,i) - a_{t+i}\|^2\right)\right]$

step index in chunk

Action Chunking



Action Chunking

Theorem (ZFPM**S** ’25)**:** Suppose that all $(\pi, f) \in \mathscr{I}$ are **open/closed-loop stable**. Then, for all chunk lengths $k \geq k_\star$

Action Chunking

Theorem (ZFPM**S** ’25)**:** Suppose that all $(\pi, f) \in \mathscr{I}$ are **open/closed-loop stable**. Then, for all chunk lengths $k \geq k_\star$

$$J(\pi^\star) - J(\hat{\pi}) \leq C_\star \, L_{\text{bc,chunk}}(\hat{\pi})$$

Action Chunking

Theorem (ZFPM**S** ’25)**:** Suppose that all $(\pi, f) \in \mathscr{I}$ are **open/closed-loop stable**. Then, for all chunk lengths $k \geq k_\star$

$$J(\pi^\star) - J(\hat{\pi}) \leq C_\star \, L_{\text{bc,chunk}}(\hat{\pi})$$

1. Action Chunking Avoids Exponential Compounding Error

Action Chunking

Theorem (ZFPM**S** ’25)**:** Suppose that all $(\pi, f) \in \mathscr{I}$ are **open/closed-loop stable**. Then, for all chunk lengths $k \geq k_\star$

$$J(\pi^\star) - J(\hat{\pi}) \leq C_\star \, L_{\mathrm{bc,chunk}}(\hat{\pi})$$

1. Action Chunking Avoids Exponential Compounding Error

2. $k_\star, C_\star$ **polynomial in relevant factors / horizon-independent**

Action Chunking

Theorem (ZFPM**S** ’25)**:** Suppose that all $(\pi, f) \in \mathcal{I}$ are **open/closed-loop stable**. Then, for all chunk lengths $k \geq k_\star$

$$J(\pi^\star) - J(\hat{\pi}) \leq C_\star \, L_{\text{bc,chunk}}(\hat{\pi})$$

1. **Action Chunking Avoids Exponential Compounding Error**

2. $k_\star, C_\star$ **polynomial in relevant factors / horizon-independent**

3. **Effect saturates at critical** $k_\star$

Action Chunking

Theorem (ZFPM**S** ’25)**:** Suppose that all $(\pi, f) \in \mathcal{I}$ are open/closed-loop stable. Then, for all chunk lengths $k \geq k_\star$

$$J(\pi^\star) - J(\hat{\pi}) \leq C_\star \, L_{\text{bc,chunk}}(\hat{\pi})$$

1. Action Chunking Avoids Exp. Compounding Error

2. $k_\star, C_\star$ **polynomial in relevant factors / horizon-independent**

3. Effect saturates at critical $k_\star$

Action Chunking

Theorem (ZFPM**S** ’25)**:** Suppose that all $(\pi, f) \in \mathcal{I}$ a
for all chunk lengths $k \geq k_\star$

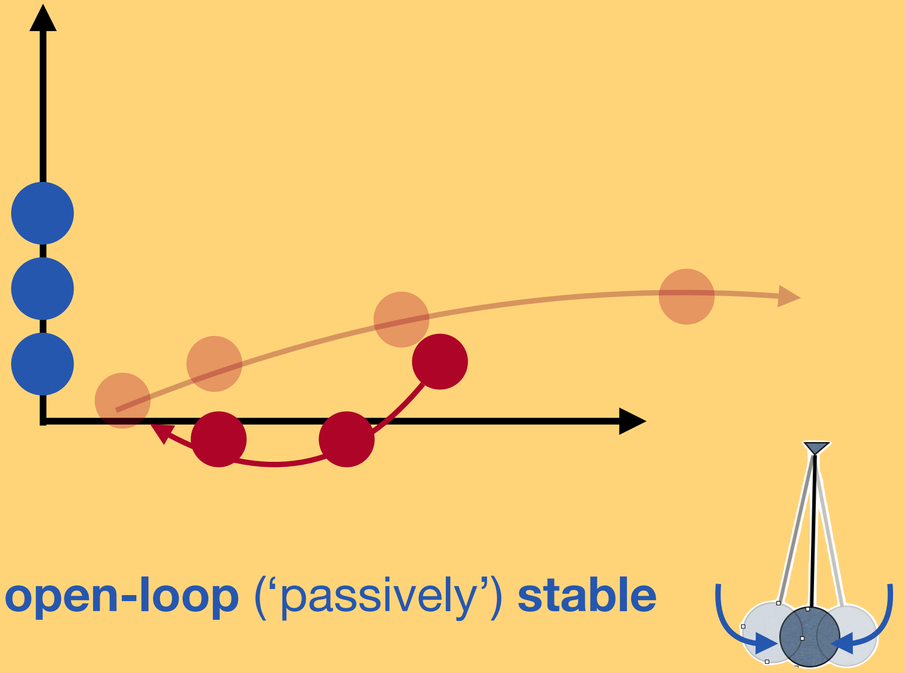
$$J(\pi^\star) - J(\hat{\pi}) \leq C_\star \, L_{\mathrm{bc,chunk}}(\hat{\pi})$$

1. Action Chunking Avoids Exp. Compounding Error

2. $k_\star, C_\star$ **polynomial in relevant factors / horizon-indep**

3. Effect saturates at critical $k_\star$

Proof: By rare policy update leverage **open-loop stability of dynamics.**



open-loop (‘passively’) **stable**

Action Chunking

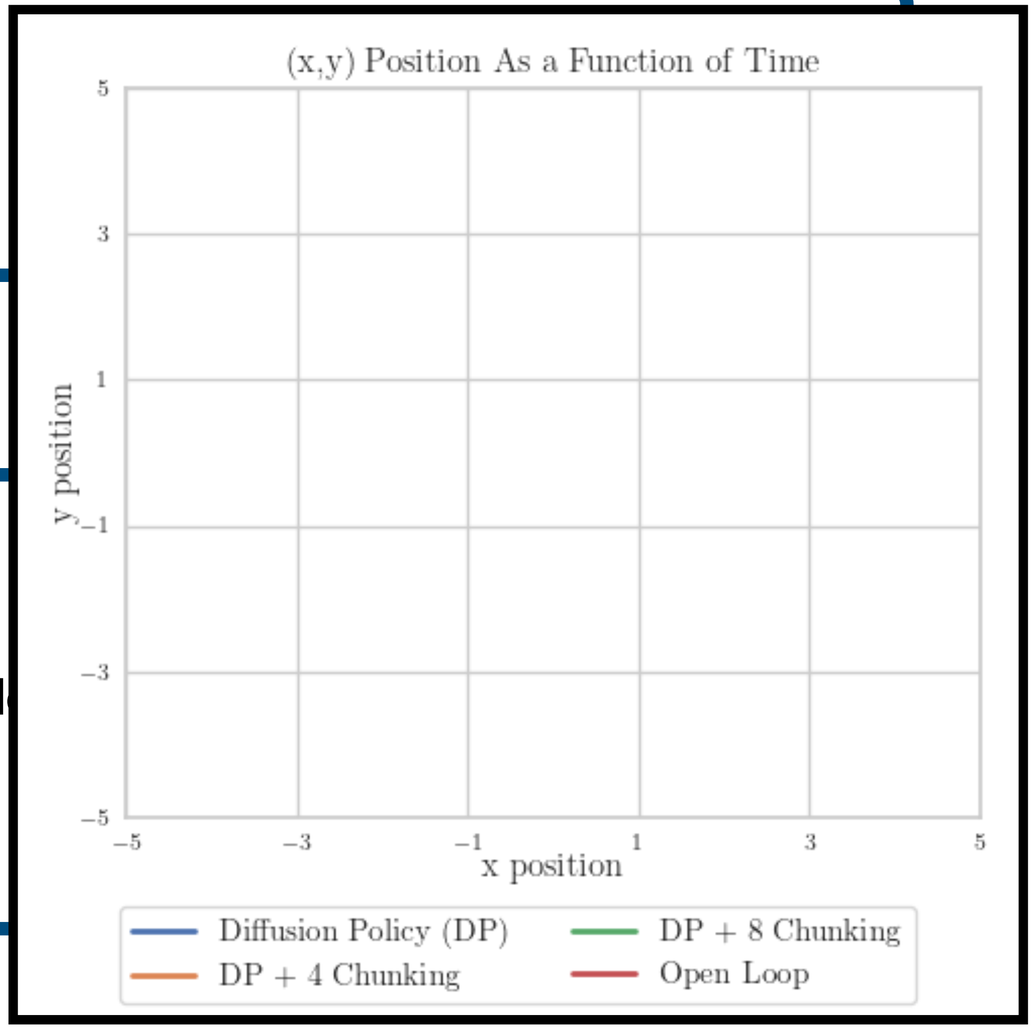
Theorem (ZFPM**S** ’25)**:** Suppose that all $(\pi, f) \in \mathcal{I}$ are
for all chunk lengths $k \geq k_\star$

$$J(\pi^\star) - J(\hat{\pi}) \leq C_\star \, L_{\text{bc,chunk}}(\hat{\pi})$$

1. Action Chunking Avoids Exp. Compounding Error

2. $k_\star, C_\star$ **polynomial in relevant factors / horizon-independ**

3. Effect saturates at critical $k_\star$



Action Chunking

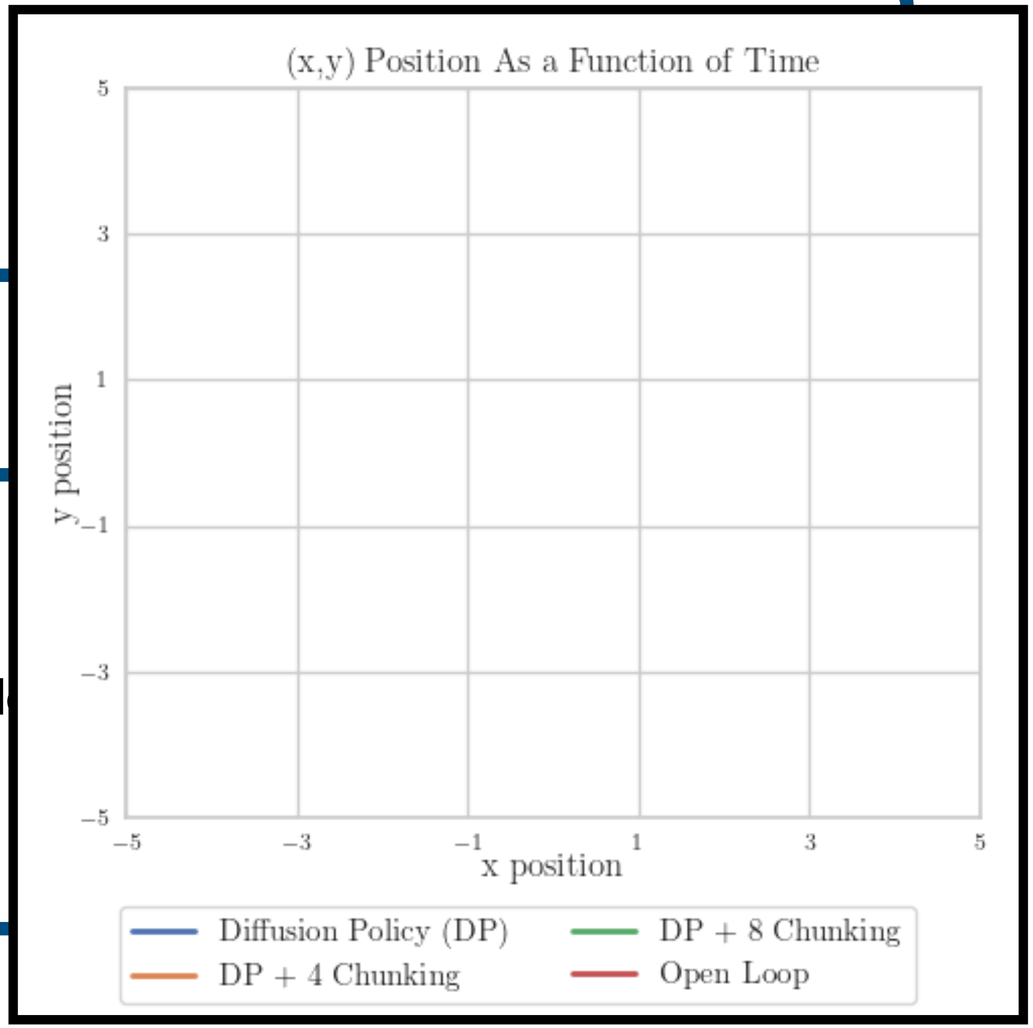
Theorem (ZFPM**S** ’25)**:** Suppose that all $(\pi, f) \in \mathcal{I}$ are
for all chunk lengths $k \geq k_\star$

$$J(\pi^\star) - J(\hat{\pi}) \leq C_\star \, L_{\text{bc,chunk}}(\hat{\pi})$$

1. Action Chunking Avoids Exp. Compounding Error

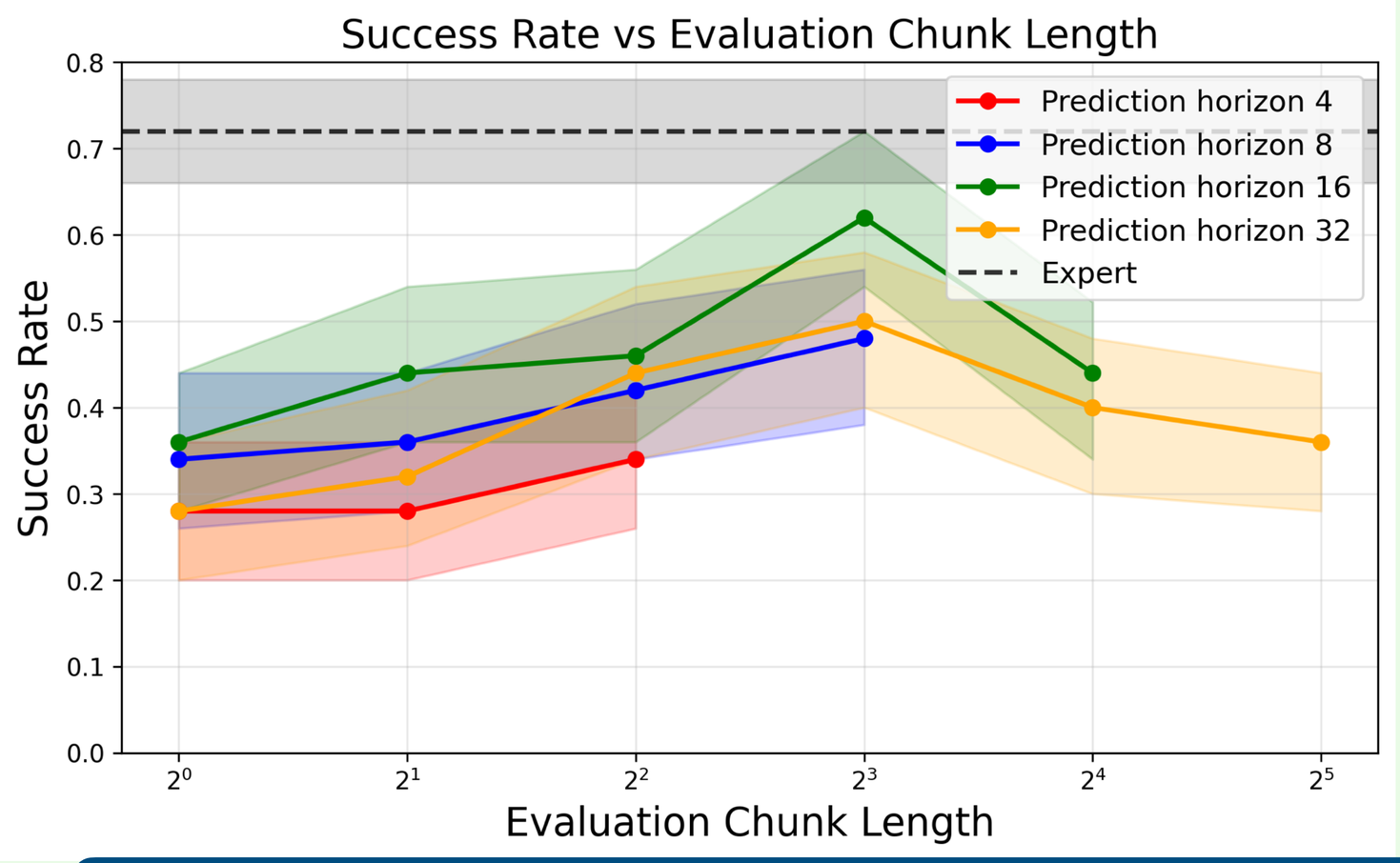
2. $k_\star, C_\star$ **polynomial in relevant factors / horizon-independ**

3. Effect saturates at critical $k_\star$



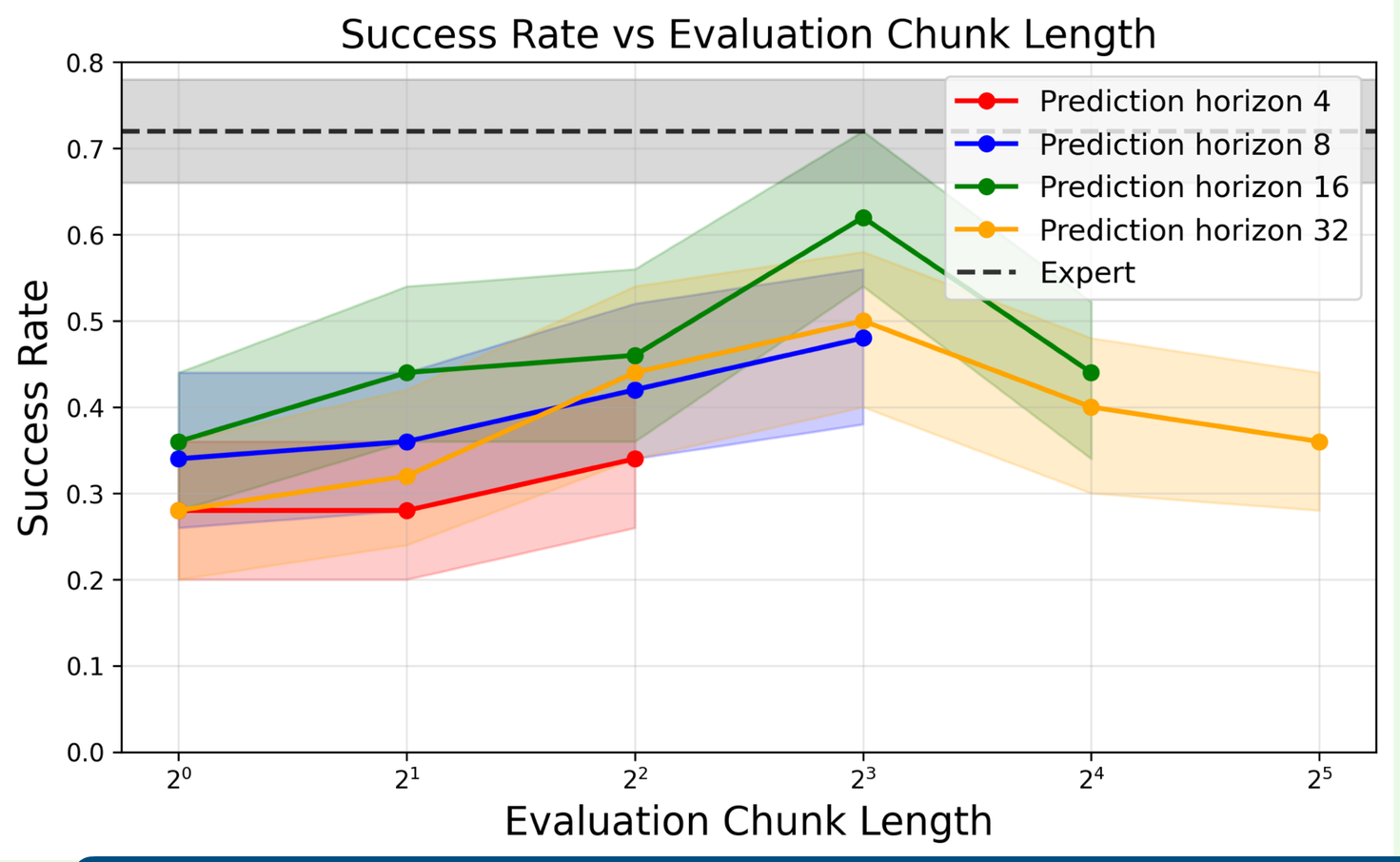
Testing the Stability Hypothesis

Testing the Stability Hypothesis



We train a state-based regression policy from deterministic expert

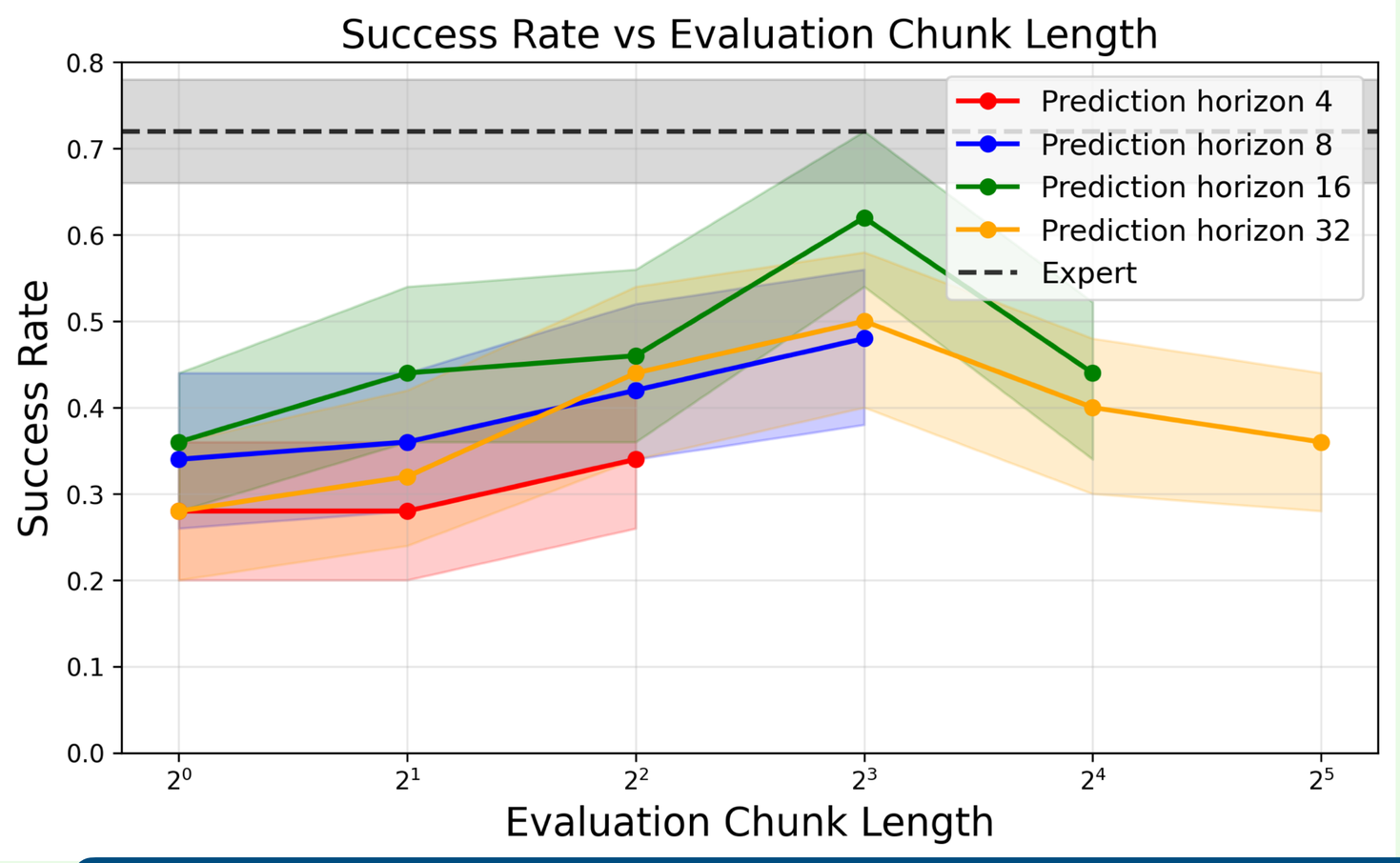
Testing the Stability Hypothesis



Theory predicts length of evaluated chunk as determining compounding error

We train a state-based regression policy from deterministic expert

Testing the Stability Hypothesis



Theory predicts length of evaluated chunk as determining compounding error

We train a state-based regression policy from deterministic expert

Other hypotheses

1. representation learning
2. partial observation
3. receding horizon control

No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

No Free Lunch: Open-Loop Stability

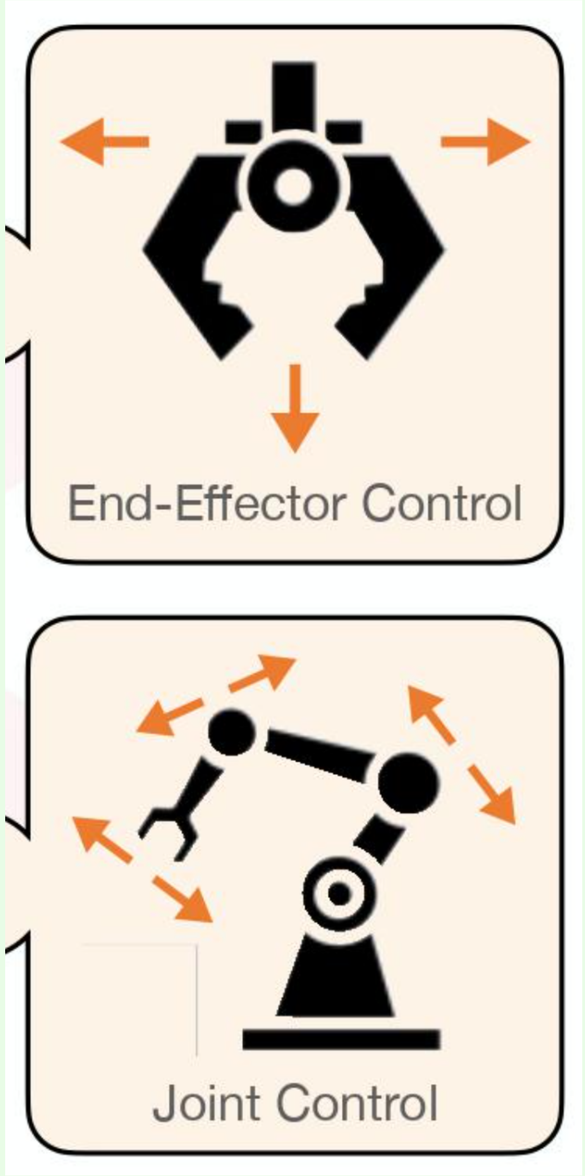
Core Assumption: Dynamics are Open-Loop Stable

Robotics uses End-Effector Control to *Deliberately* Put us in this Regime

No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

Robotics uses End-Effector Control to *Deliberately* Put us in this Regime



No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

End effector control puts us in this regime!

1. position commands (e.g. 20 Hz) **tracked** by low-level controller, e.g. PD torque

No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

End effector control puts us in this regime!

1. position commands (e.g. 20 Hz) **tracked** by low-level controller, e.g. PD torque

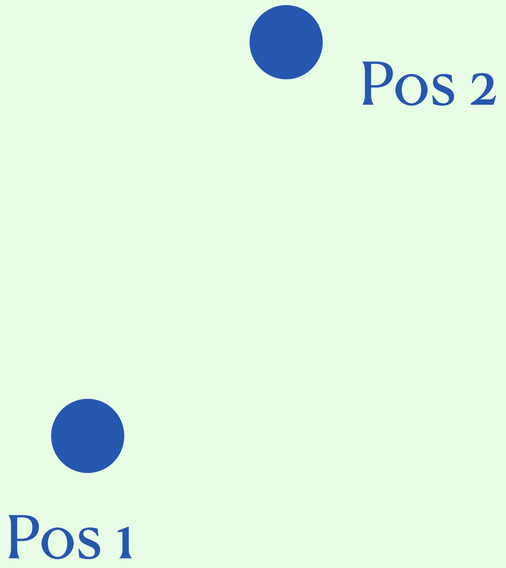
Pos 1

No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

End effector control puts us in this regime!

1. position commands (e.g. 20 Hz) **tracked** by low-level controller, e.g. PD torque

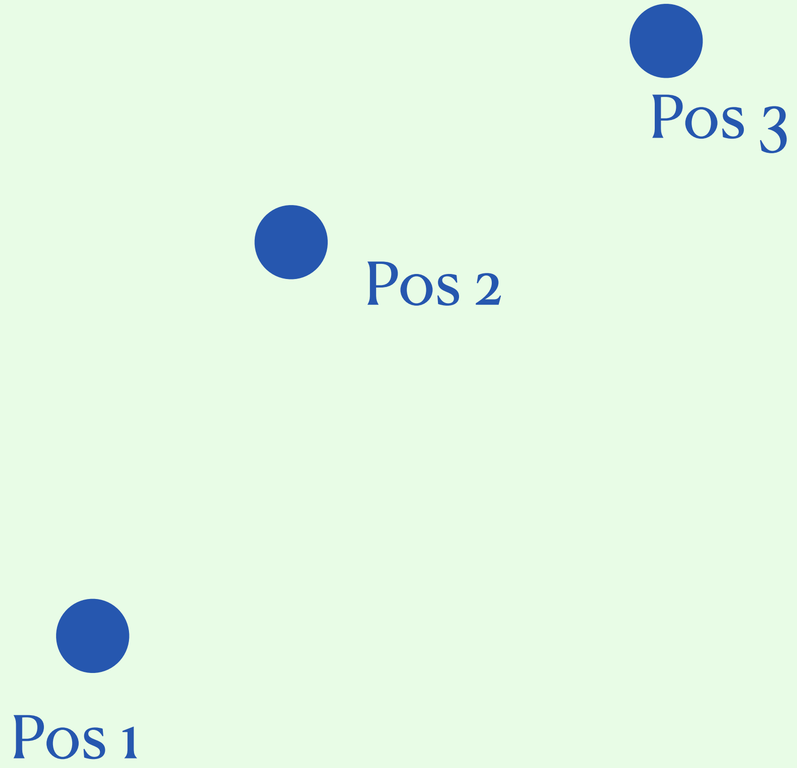


No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

End effector control puts us in this regime!

1. position commands (e.g. 20 Hz) **tracked** by low-level controller, e.g. PD torque

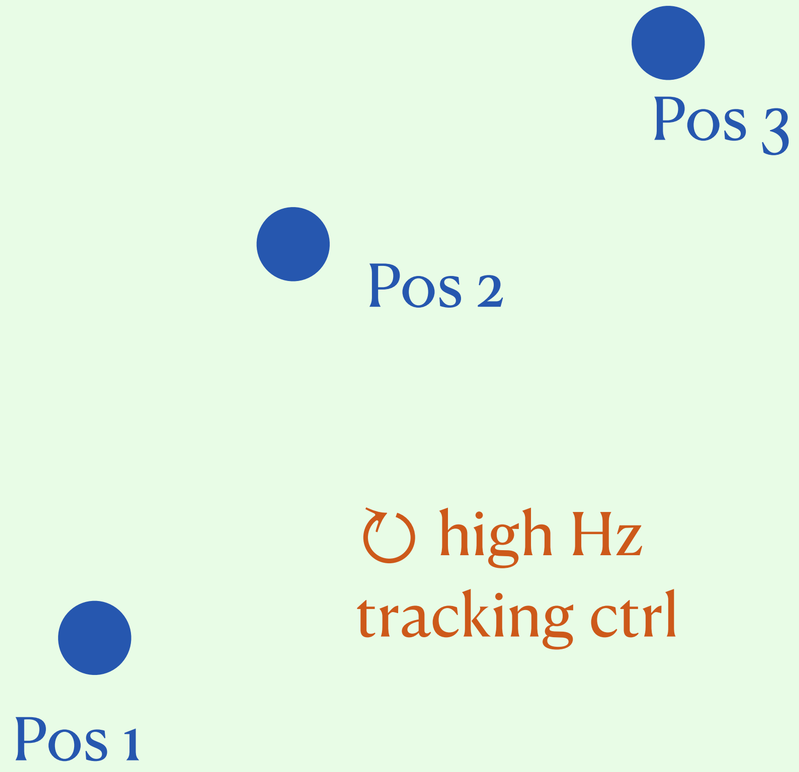


No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

End effector control puts us in this regime!

1. position commands (e.g. 20 Hz) **tracked** by low-level controller, e.g. PD torque

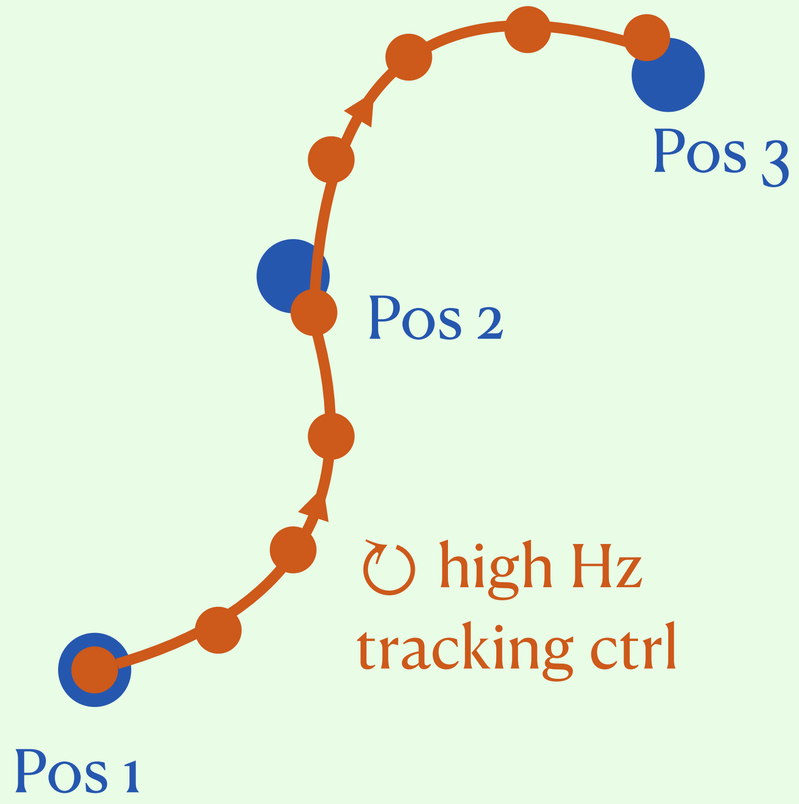


No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

End effector control puts us in this regime!

1. position commands (e.g. 20 Hz) **tracked** by low-level controller, e.g. PD torque

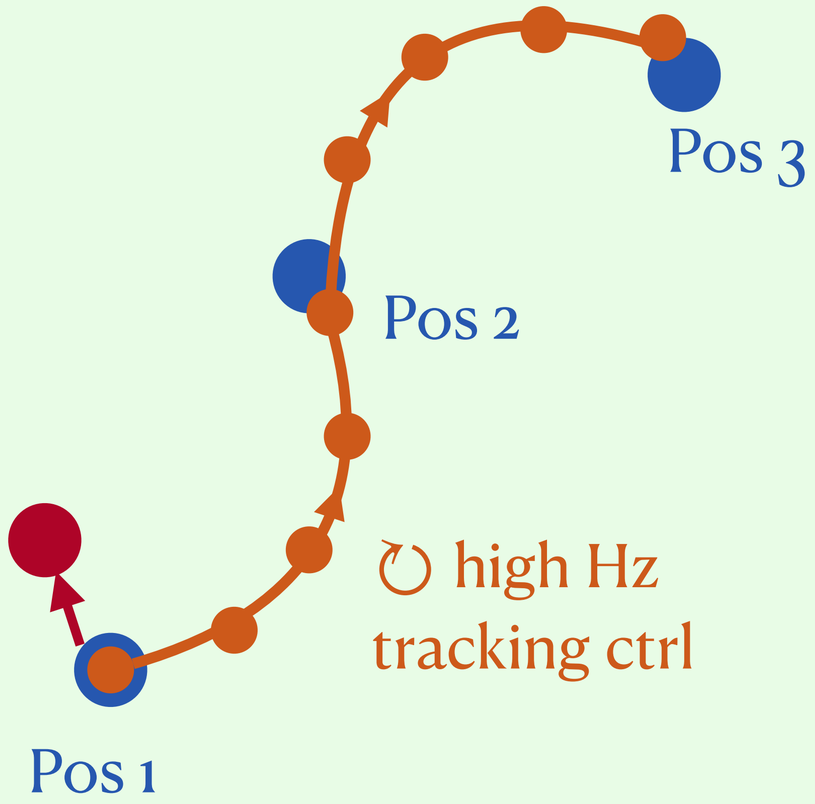


No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

End effector control puts us in this regime!

1. position commands (e.g. 20 Hz) **tracked** by low-level controller, e.g. PD torque

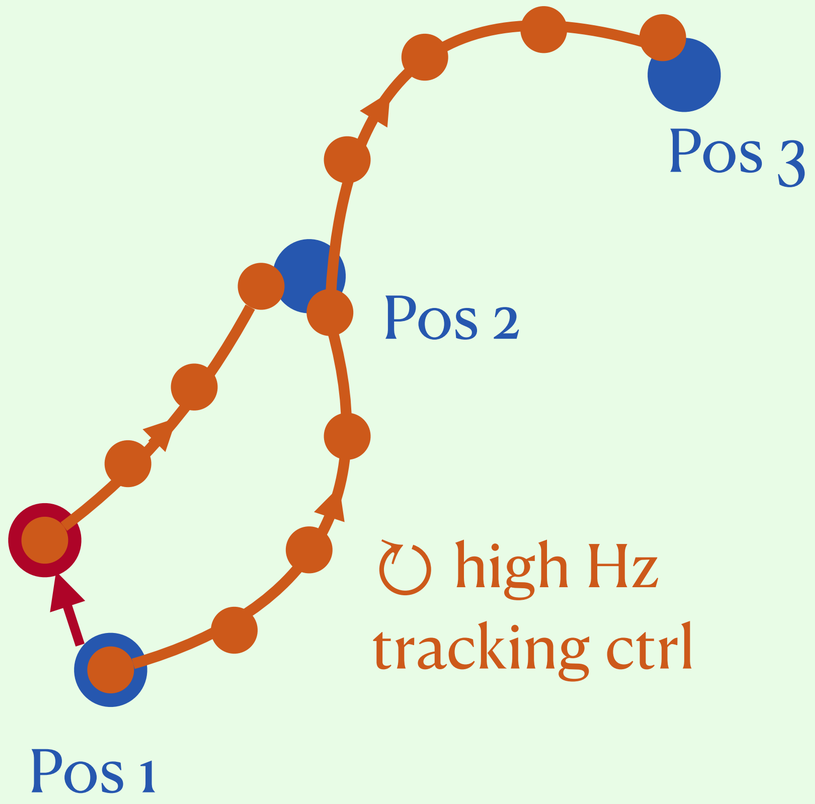


No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

End effector control puts us in this regime!

1. position commands (e.g. 20 Hz) **tracked** by low-level controller, e.g. PD torque



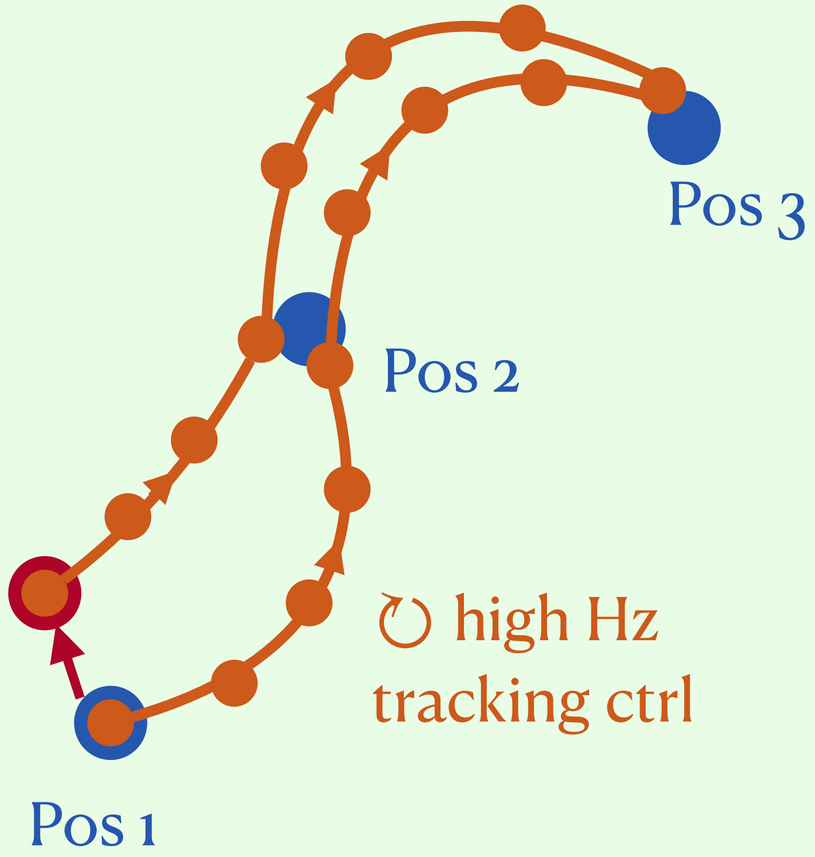
No Free Lunch: Open-Loop Stability

Core Assumption: Dynamics are Open-Loop Stable

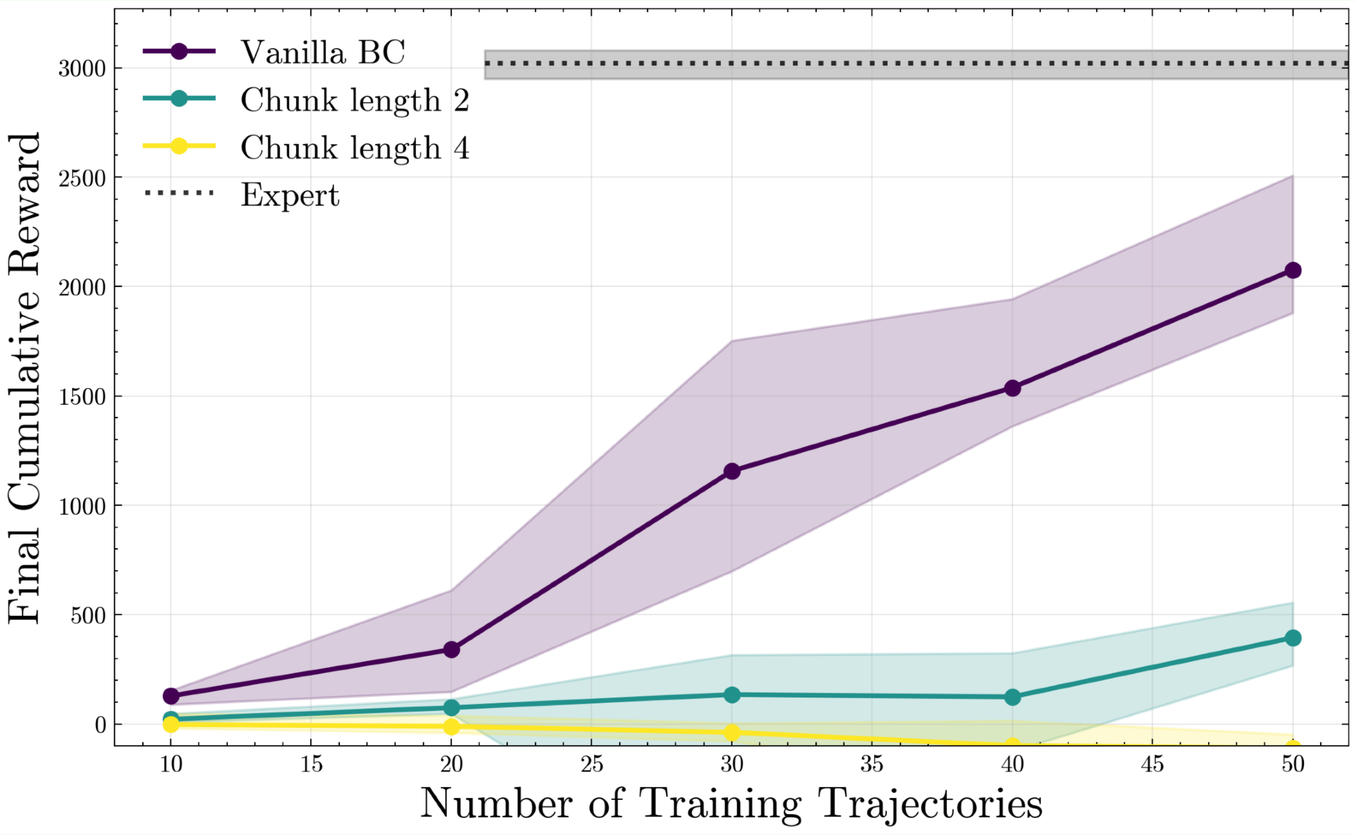
End effector control puts us in this regime!

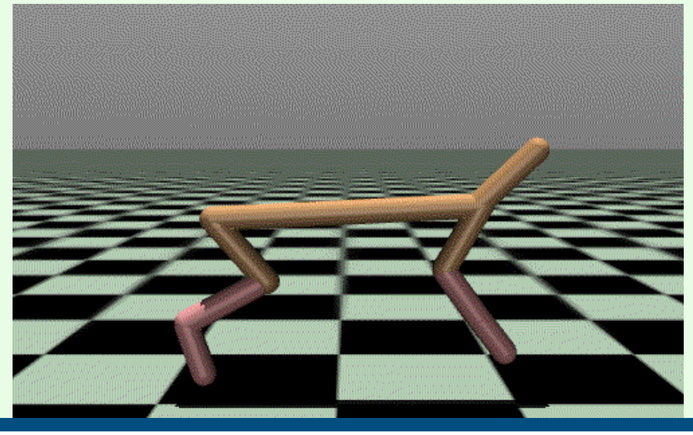
1. position commands (e.g. 20 Hz) **tracked** by low-level controller, e.g. PD torque

2. "Open-loop" system mapping end-effector to state at policy freq (20 Hz) is **stable**



Weaknesses of Action Chunking



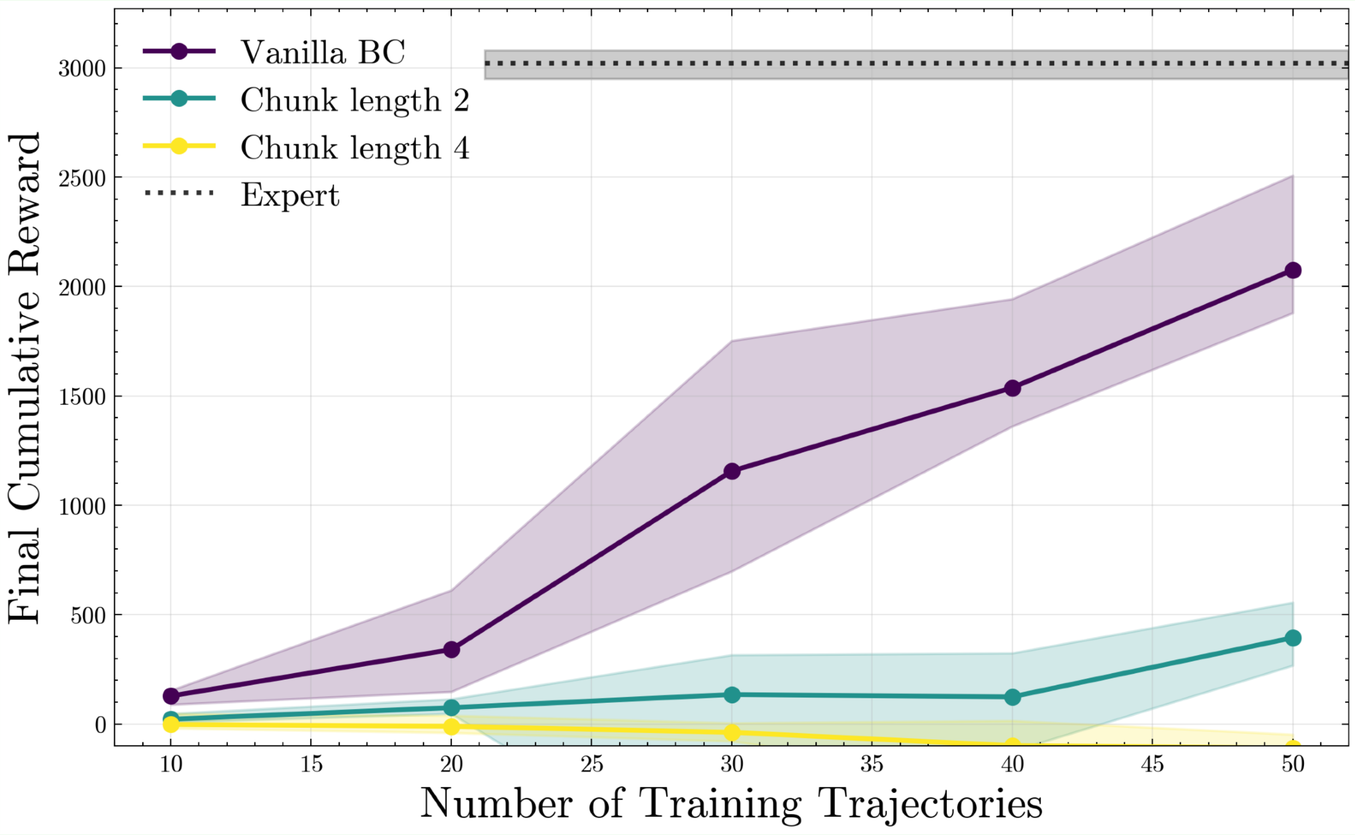


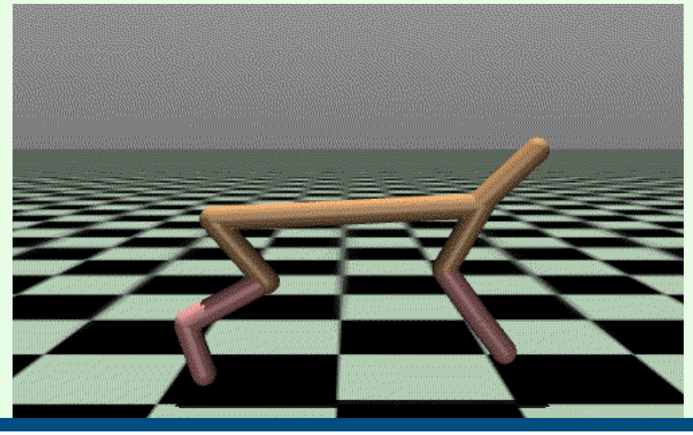
1. limited reactivity

2. poor performance if system unstable

3. can require large network

Weaknesses of Action Chunking





1. limited reactivity

2. poor performance if system unstable

3. can require large network

Summary: Action Chunking

Summary: Action Chunking

1. In continuous action space, exponential compounding error can occur

Summary: Action Chunking

1. In continuous action space, exponential compounding error can occur *(quantitative Moravec)*

Summary: Action Chunking

1. In continuous action space, exponential compounding error can occur *(quantitative Moravec)*

2. Action chunking surmounts this with open-loop stability

Summary: Action Chunking

1. In continuous action space, exponential compounding error can occur *(quantitative Moravec)*

2. Action chunking surmounts this with open-loop stability *(e.g. end-effector control)*

Summary: Action Chunking

1. In continuous action space, exponential compounding error can occur *(quantitative Moravec)*

2. Action chunking surmounts this with open-loop stability *(e.g. end-effector control)*

3. Action chunking fails without open-loop stability

Summary: Action Chunking

1. In continuous action space, exponential compounding error can occur *(quantitative Moravec)*

2. Action chunking surmounts this with open-loop stability *(e.g. end-effector control)*

3. Action chunking fails without open-loop stability *(no free lunch)*

Can we do better?

Can we do better?

Do we need handcrafted stabilizing controllers? What about without stability?

Can we do better?

Do we need handcrafted stabilizing controllers? What about without stability?

Slippage means even if we control the hand, we can’t guarantee stable control of the object.

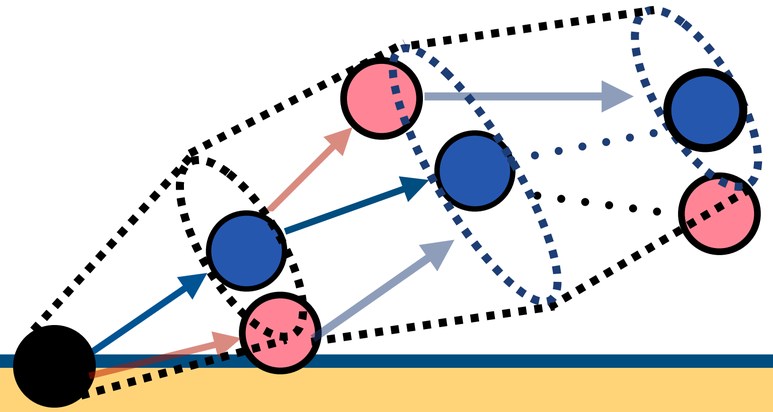
Exploratory Data Collection

Exploratory Data Collection

Theorem (ZPPMS '25): Define exploratory expert $\tilde{\pi}^\star = \pi^\star + \sigma z$

$$L_{\mathrm{bc},\sigma}(\pi) = \mathbb{E}^{\tilde{\pi}^\star}\left[\sum_{t=1}^{H} \|\pi(s_t) - \pi^\star(s_t)\|^2\right]$$

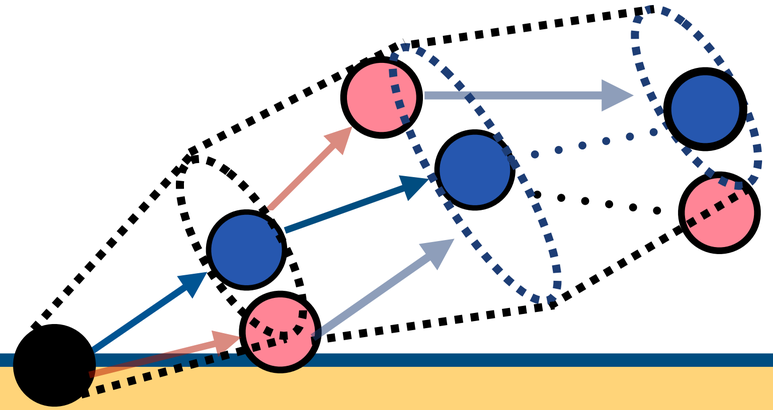
Exploratory Data Collection



Theorem (ZPPMS '25): Define exploratory expert $\tilde{\pi}^\star = \pi^\star + \sigma z$

$$L_{\mathrm{bc},\sigma}(\pi) = \mathbb{E}^{\tilde{\pi}^\star}\left[\sum_{t=1}^{H} \|\pi(s_t) - \pi^\star(s_t)\|^2\right]$$

Exploratory Data Collection



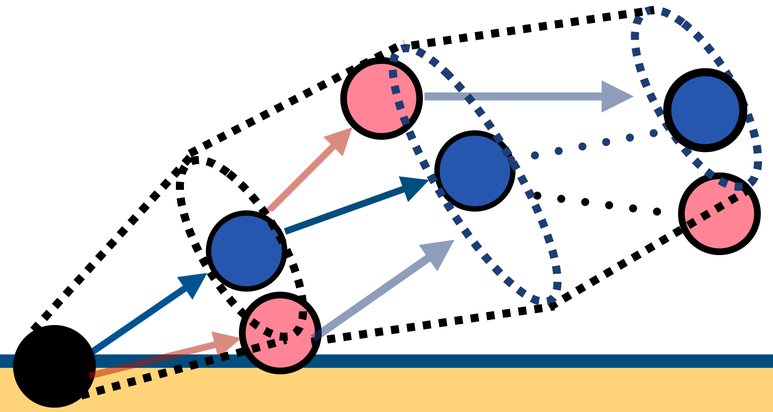
Theorem (ZPPMS '25): Define exploratory expert $\tilde{\pi}^\star = \pi^\star + \sigma z$

$$L_{\mathrm{bc},\sigma}(\pi) = \mathbb{E}^{\tilde{\pi}^\star}\left[\sum_{t=1}^{H} \|\pi(s_t) - \pi^\star(s_t)\|^2\right]$$

$$J(\pi^\star) - J(\hat{\pi}) \lesssim L_{\mathrm{bc}}(\hat{\pi}) + L_{\mathrm{bc},\sigma}(\hat{\pi})$$

No compounding error!

Exploratory Data Collection



Theorem (ZPPMS ‘25): Define exploratory expert $\tilde{\pi}^\star = \pi^\star + \sigma z$

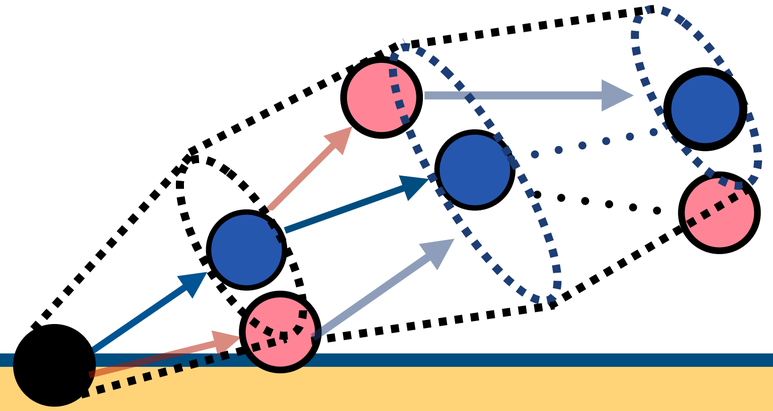
$$L_{\mathrm{bc},\sigma}(\pi) = \mathbb{E}^{\tilde{\pi}^\star}\left[\sum_{t=1}^{H} \|\pi(s_t) - \pi^\star(s_t)\|^2\right]$$

$$J(\pi^\star) - J(\hat{\pi}) \lesssim L_{\mathrm{bc}}(\hat{\pi}) + L_{\mathrm{bc},\sigma}(\hat{\pi})$$

No compounding error!

DART (Laskey et al. ’17), but non-adaptive

Exploratory Data Collection



Theorem (ZPPMS ‘25): Define exploratory expert $\tilde{\pi}^\star = \pi^\star + \sigma z$

$$L_{\mathrm{bc},\sigma}(\pi) = \mathbb{E}^{\tilde{\pi}^\star}\left[\sum_{t=1}^{H} \|\pi(s_t) - \pi^\star(s_t)\|^2\right]$$

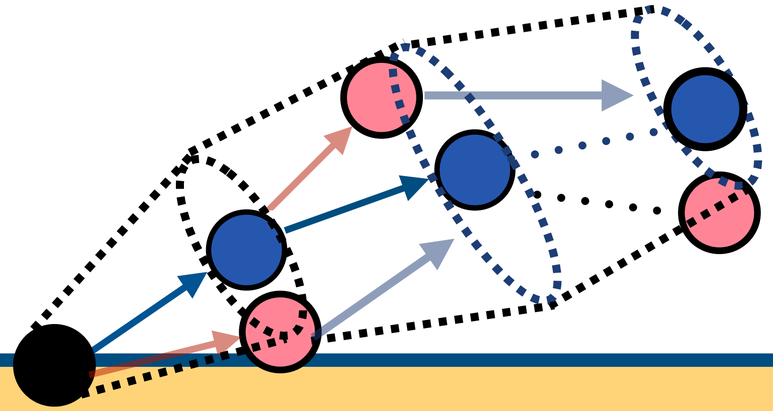
$$J(\pi^\star) - J(\hat{\pi}) \lesssim L_{\mathrm{bc}}(\hat{\pi}) + L_{\mathrm{bc},\sigma}(\hat{\pi})$$

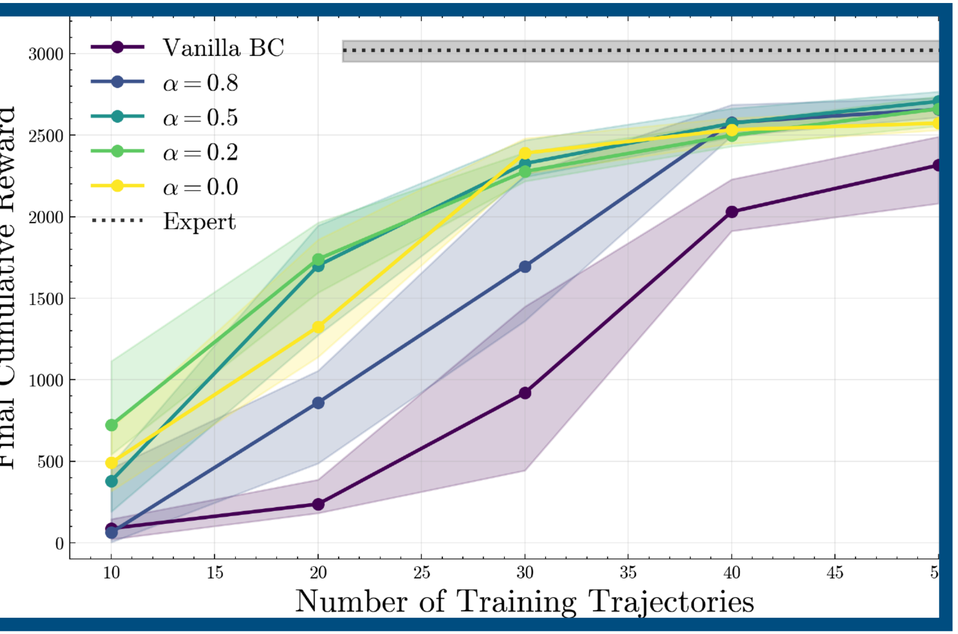
No compounding error!

DART (Laskey et al. ’17), but non-adaptive

Builds off analysis from **TaSIL** (Pfrommer et al. 2022)

Exploratory Data Collection





Effective Without Open-Loop Stability

Theorem (ZPPMS '25): Define exploratory expert $\tilde{\pi}^\star = \pi^\star + \sigma z$

$$L_{\mathrm{bc},\sigma}(\pi) = \mathbb{E}^{\tilde{\pi}^\star}\left[\sum_{t=1}^{H} \|\pi(s_t) - \pi^\star(s_t)\|^2\right]$$

$$J(\pi^\star) - J(\hat{\pi}) \lesssim L_{\mathrm{bc}}(\hat{\pi}) + L_{\mathrm{bc},\sigma}(\hat{\pi})$$

No compounding error!

DART (Laskey et al. '17), but non-adaptive

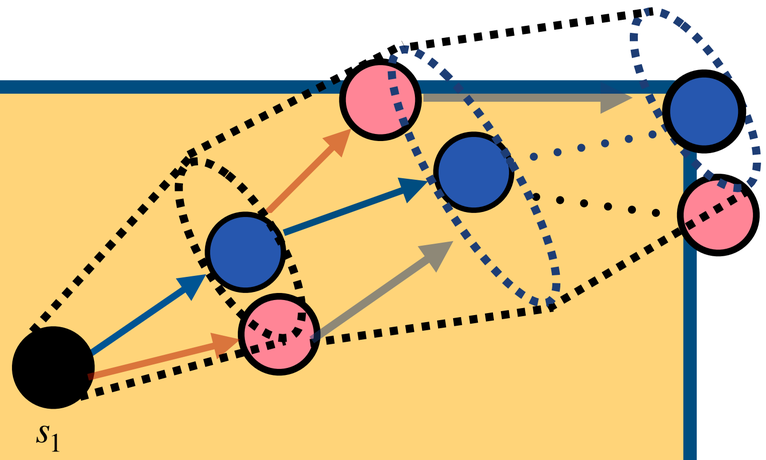
Builds off analysis from **TaSIL** (Pfrommer et al. 2022)

Optimal Data Collection is Still Open

Optimal Data Collection is Still Open

1. Noise Injection: Great for local corrections, can be too **unstructured dexterous tasks**

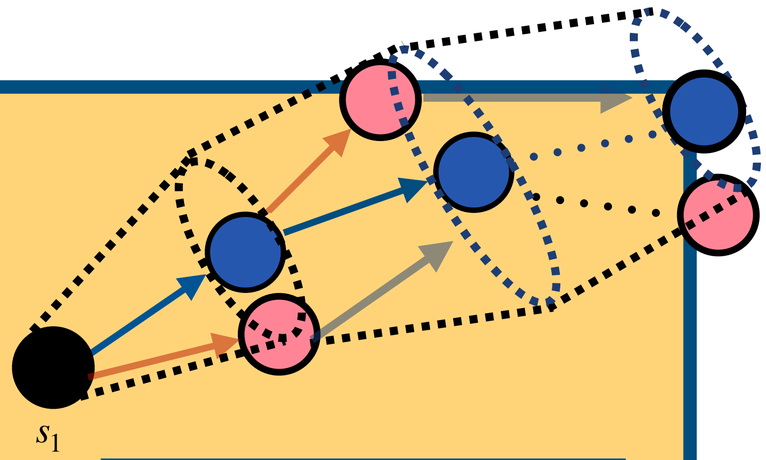
DART Laskey et. ’17, ZPPMS ‘25



Optimal Data Collection is Still Open

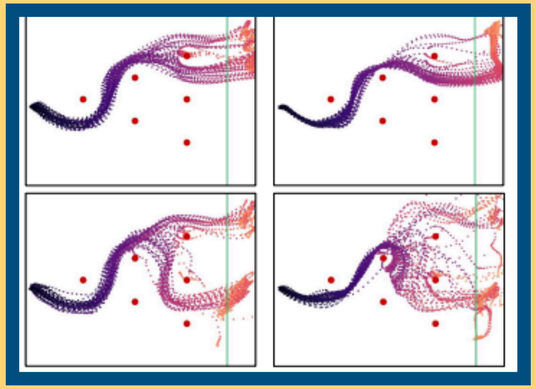
1. Noise Injection: Great for local corrections, can be too **unstructured dexterous tasks**

DART Laskey et. ’17, ZPPMS ‘25



2. On Manifold Exploration Excited Exploration Direction in More Structured Way, but very local

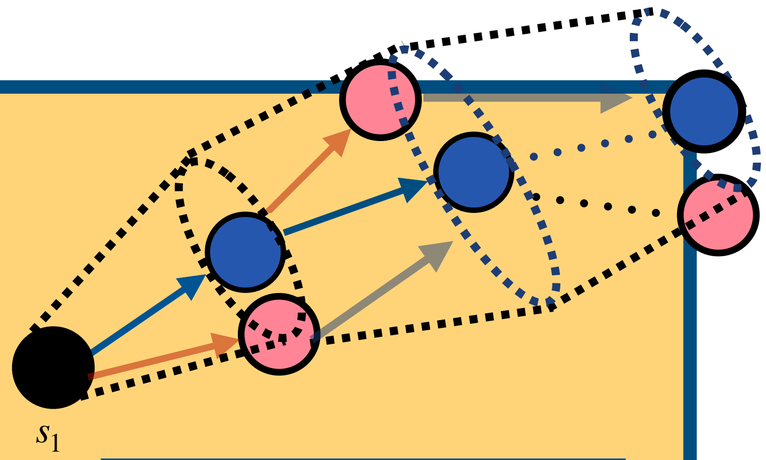
DPPO Ren…S ’24 , DSRL Wagenmaker '24



Optimal Data Collection is Still Open

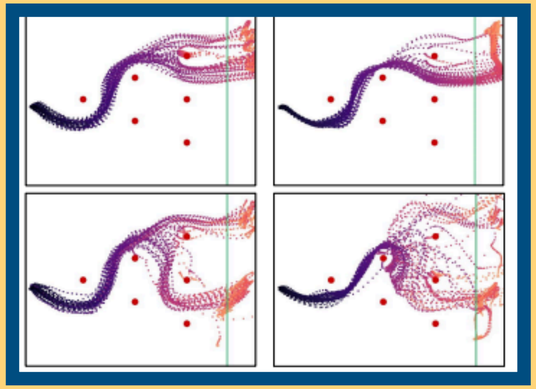
1. Noise Injection: Great for local corrections, can be too **unstructured dexterous tasks**

DART Laskey et. ’17, ZPPMS ‘25



2. On Manifold Exploration Excited Exploration Direction in More Structured Way, but very local

DPPO Ren…S ’24 , DSRL Wagenmaker '24



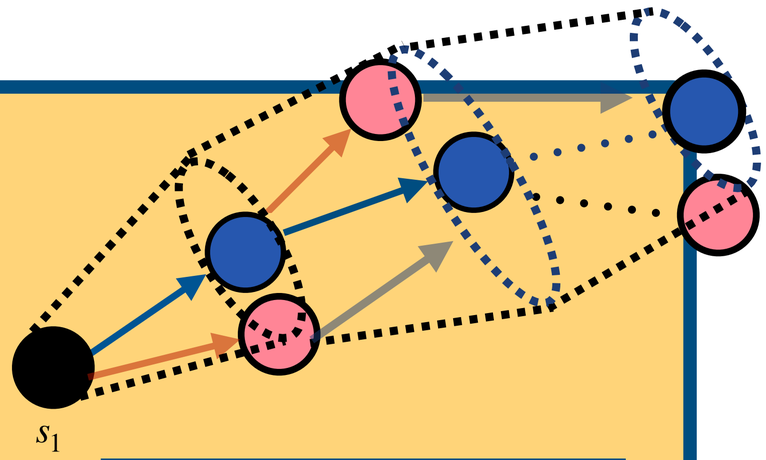
3. ‘Semantic’ Data Collection: Learning re-try/corrections

RAC: Hu…Kumar ’25, PI 0.6

Optimal Data Collection is Still Open

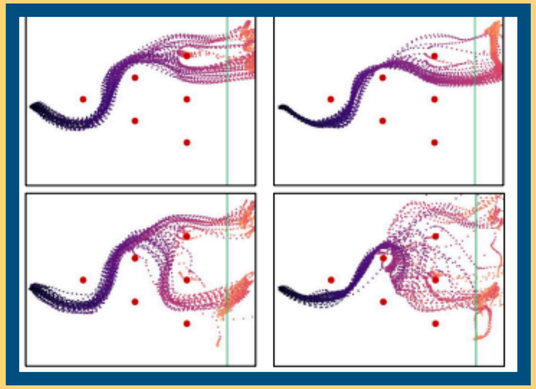
1. Noise Injection: Great for local corrections, can be too **unstructured dexterous tasks**

DART Laskey et. ’17, ZPPMS ‘25



2. On Manifold Exploration Excited Exploration Direction in More Structured Way, but very local

DPPO Ren…S ’24 , DSRL Wagenmaker '24



3. ‘Semantic’ Data Collection: Learning re-try/corrections

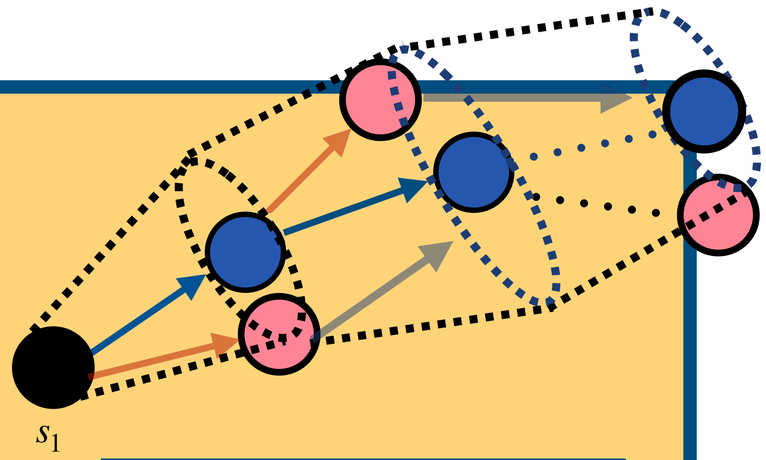
RAC: Hu…Kumar ’25, PI 0.6

4. Relabeling Tricks: Relabel failures with corrections

Optimal Data Collection is Still Open

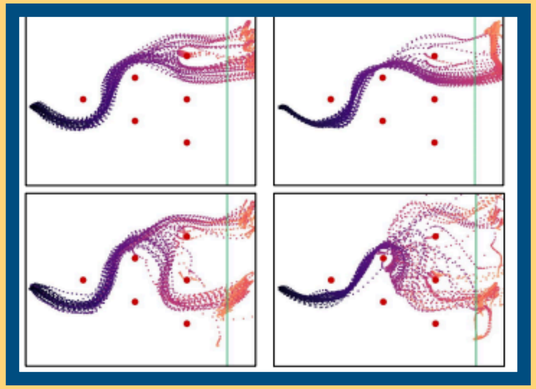
1. Noise Injection: Great for local corrections, can be too **unstructured dexterous tasks**

DART Laskey et. ’17, ZPPMS ‘25



2. On Manifold Exploration Excited Exploration Direction in More Structured Way, but very local

DPPO Ren…S ’24 , DSRL Wagenmaker '24



3. ‘Semantic’ Data Collection: Learning re-try/corrections

RAC: Hu…Kumar ’25, PI 0.6

4. Relabeling Tricks: Relabel failures with corrections

‘**Grasping with Chopsticks’:** Ke et al. 2020

Summary: Imitation Learning in Continuous Action Space

Summary: Imitation Learning in Continuous Action Space

1. Learning in continuous action spaces can suffer from exponential-in-horizon compounding error!

Summary: Imitation Learning in Continuous Action Space

1. Learning in continuous action spaces can suffer from exponential-in-horizon compounding error! *(much worse than discrete action spaces)*

Summary: Imitation Learning in Continuous Action Space

1. Learning in continuous action spaces can suffer from exponential-in-horizon compounding error! *(much worse than discrete action spaces)*

2. Action Chunking avoids this mistake, *if* we can make the dynamics open-loop stable

Summary: Imitation Learning in Continuous Action Space

1. Learning in continuous action spaces can suffer from exponential-in-horizon compounding error! *(much worse than discrete action spaces)*

2. Action Chunking avoids this mistake, *if* we can make the dynamics open-loop stable

3. Otherwise, we need more intentional data collection to avoid compounding errors.

Coda

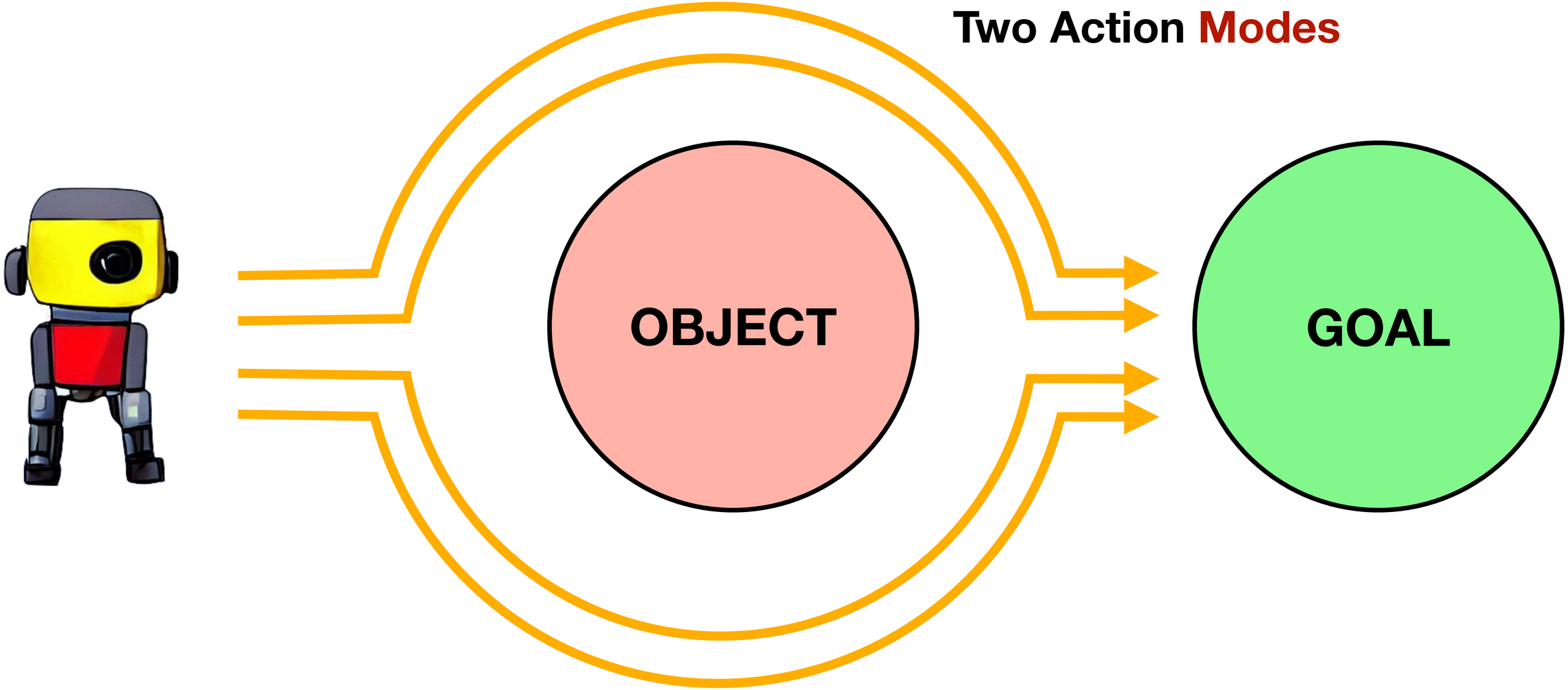
Generative Control Policies

Generative Control Policies

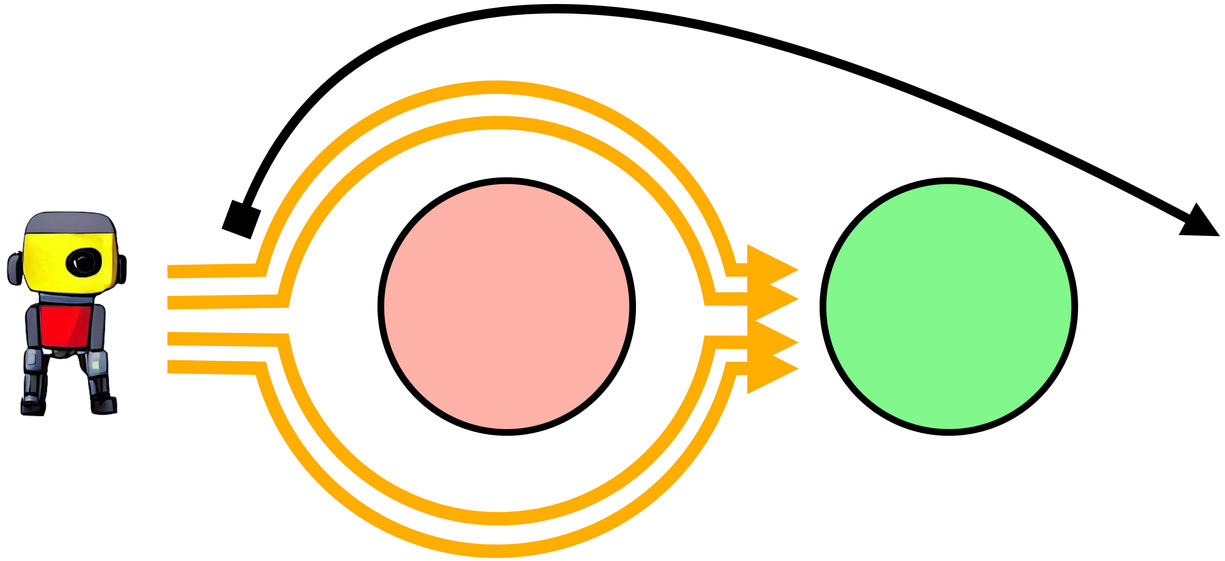
Previously: Deterministic Experts

Generative Control Policies

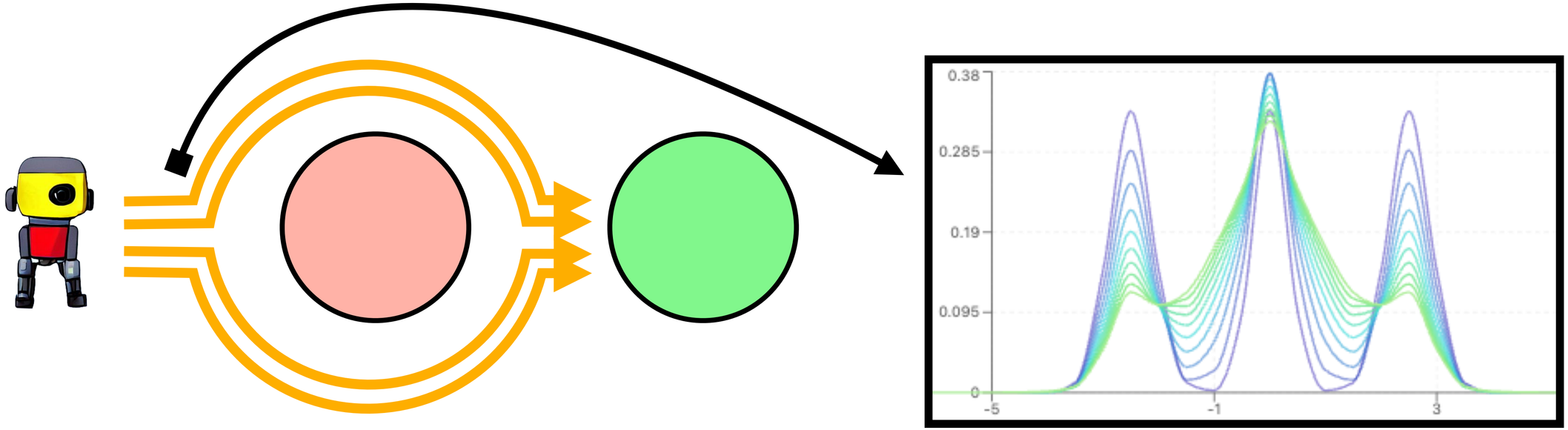
Previously: Deterministic Experts



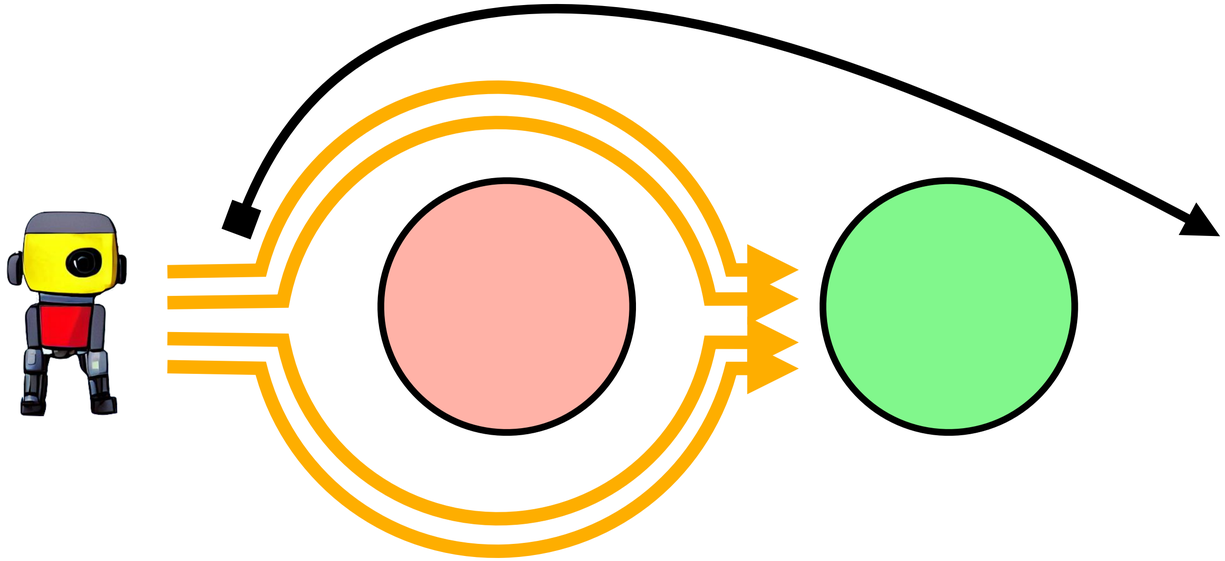
Generative Control Policies

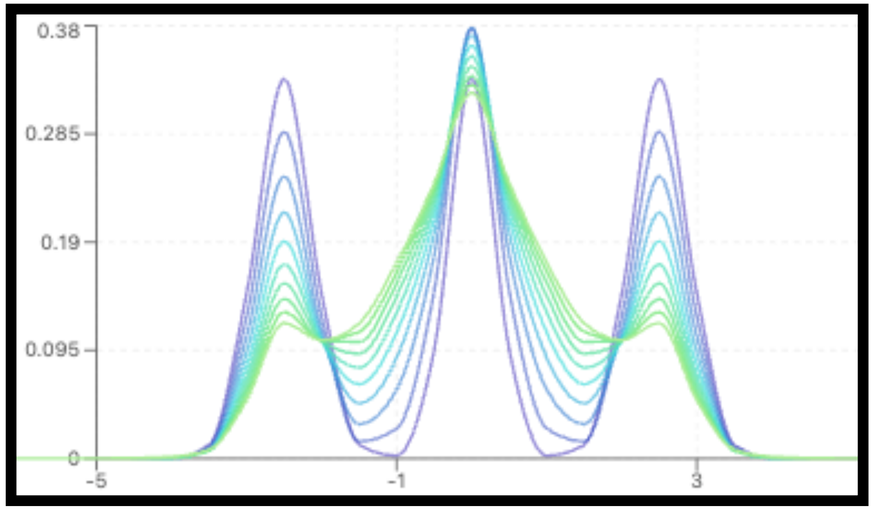


Generative Control Policies



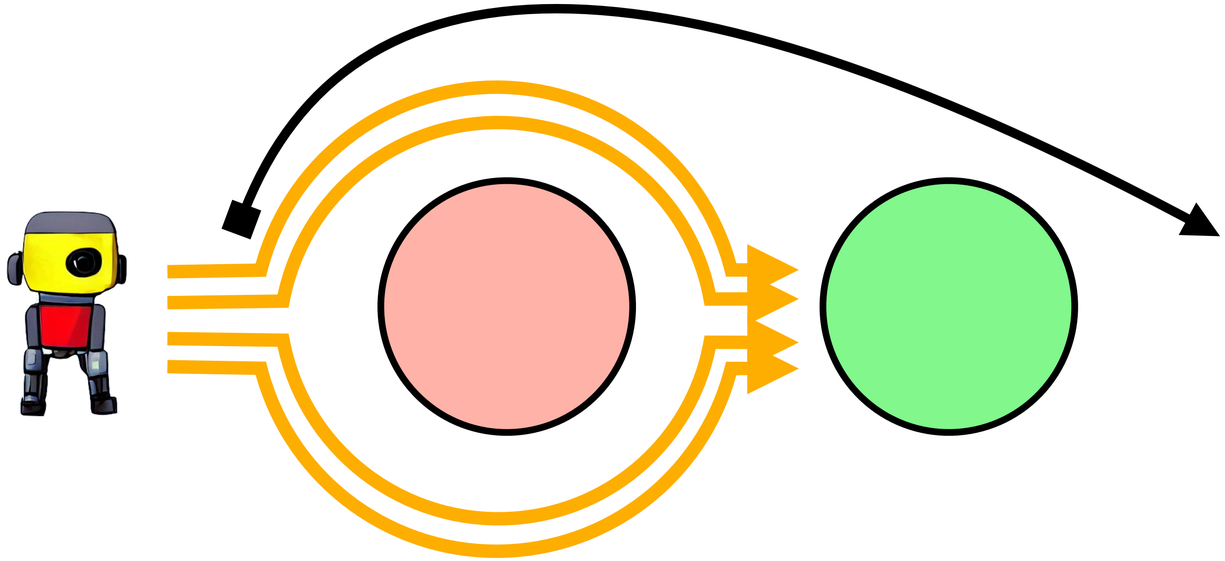
Generative Control Policies

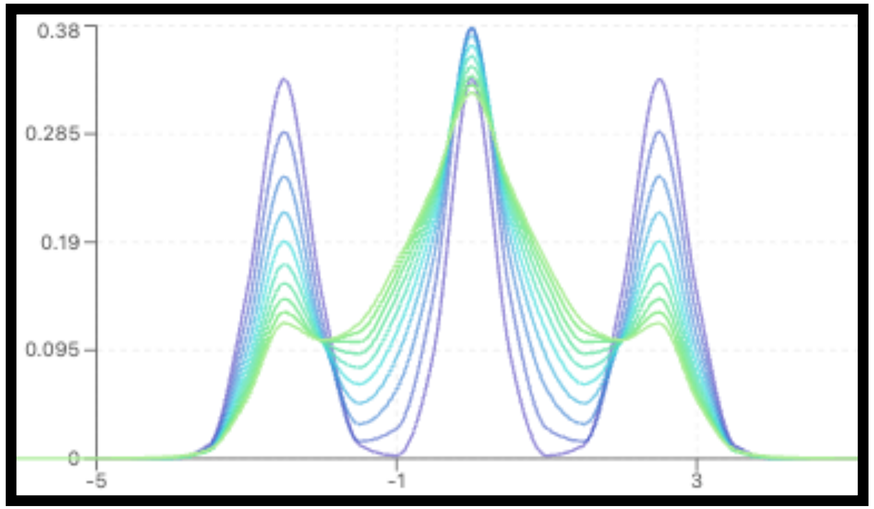




Train policy for predict **multimodal distributions of** $p(a \mid o)$

Generative Control Policies

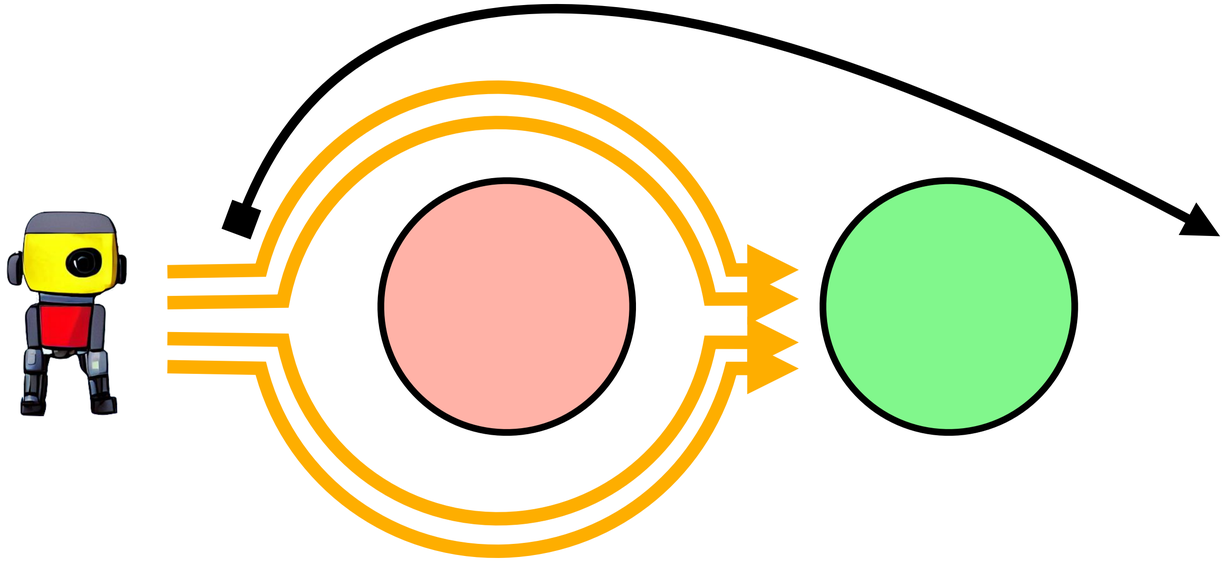




Train policy for predict **multimodal distributions of** $p(a \mid o)$

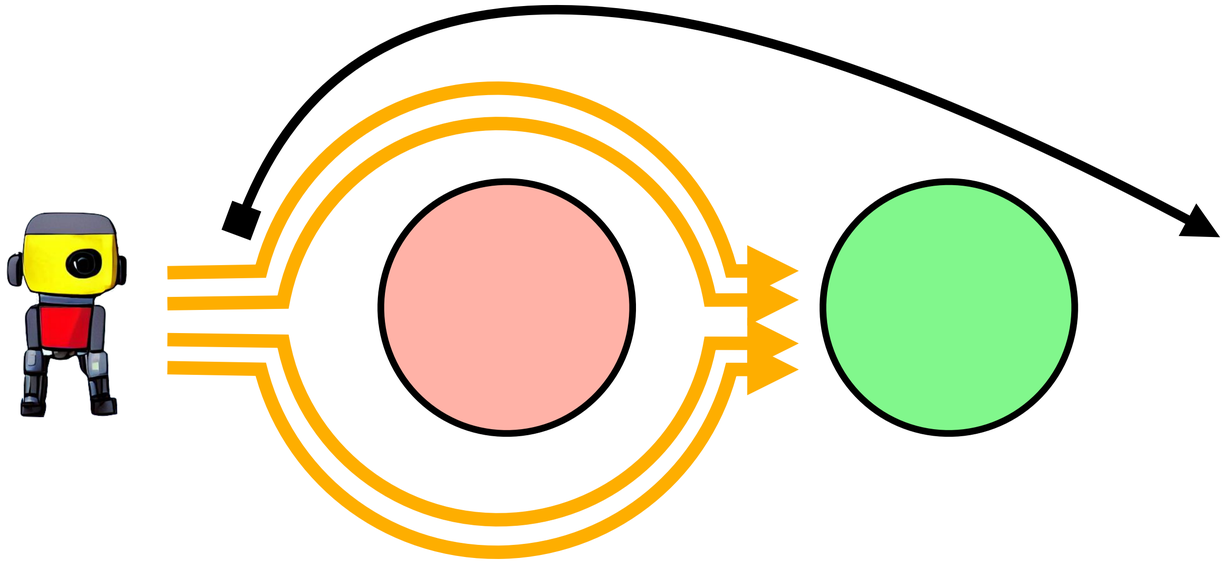
(Diffusion Policy, Chi et al. ’23)

Generative Control Policies (GCPs)



Train policy for predict
multimodal distributions
of $p(a \mid o)$

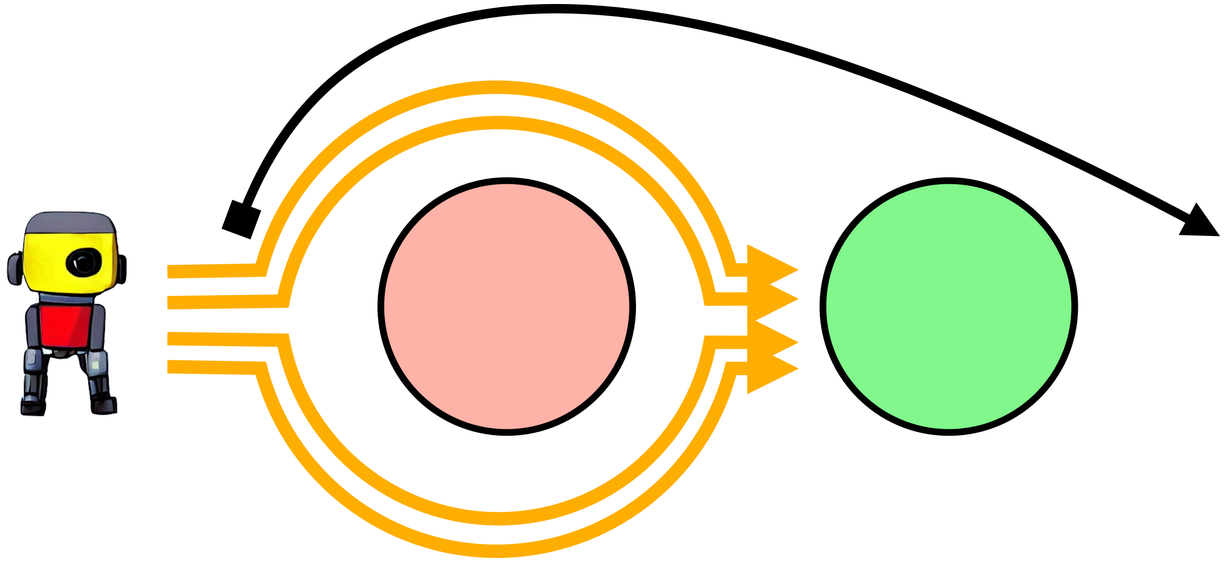
Generative Control Policies (GCPs)



Train policy for predict **multimodal distributions of** $p(a \mid o)$

1. GCPs are the basis for **the majority of robot VLA foundation models**

Generative Control Policies (GCPs)



Train policy for predict **multimodal distributions of** $p(a \mid o)$

1. GCPs are the basis for **the majority of robot VLA foundation models**

2. Theoretically, GCPs can imitate **arbitrary stochastic expert demonstrations!**

(Block, Jadbabie, Pfrommer, Simchowitz, Tedrake '23)

But do we need GCPs?

Train policy for predict
multimodal distributions
of $p(a \mid o)$

(e.g. ***denoising*** *diffusion models, DDPMs)*

But do we need GCPs?

Train policy for predict **multimodal distributions of** $p(a \mid o)$

(e.g. ***denoising*** *diffusion models, DDPMs)*

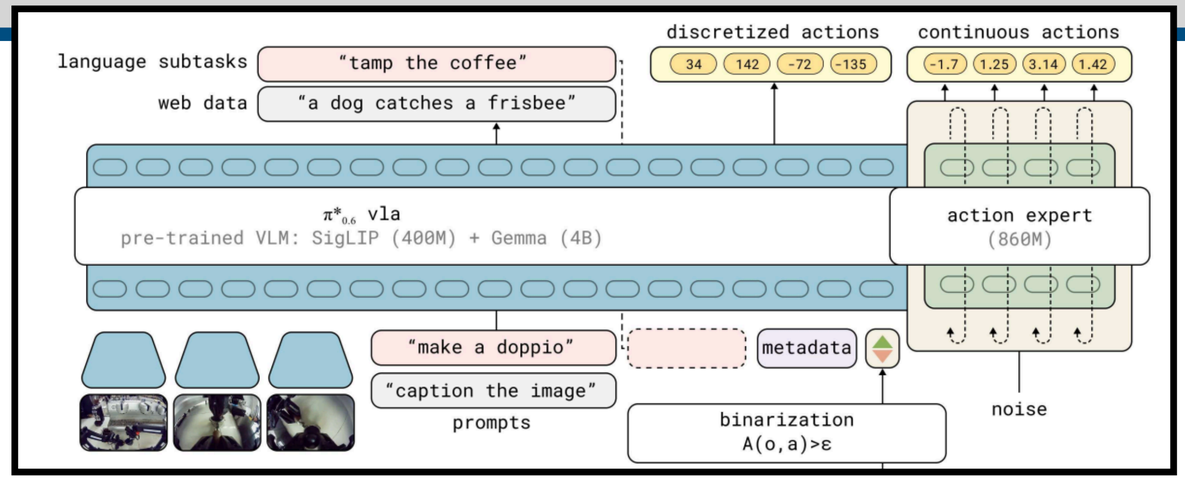
Again, to stress: generative control policies are the technology claimed to responsible for much of the recent **revolution in robot capabilities!**

But do we need GCPs?

Train policy for predict **multimodal distributions of** $p(a \mid o)$

(e.g. ***denoising*** *diffusion models, DDPMs)*

Again, to stress: generative control policies are the technology claimed to responsible for much of the recent **revolution in robot capabilities!**



$\pi^{\star}_{0,6}$ **model**

But do we need GCPs?

Train policy for predict
multimodal distributions
of $p(a \mid o)$

(e.g. ***denoising*** *diffusion models, DDPMs)*

But do we need GCPs?

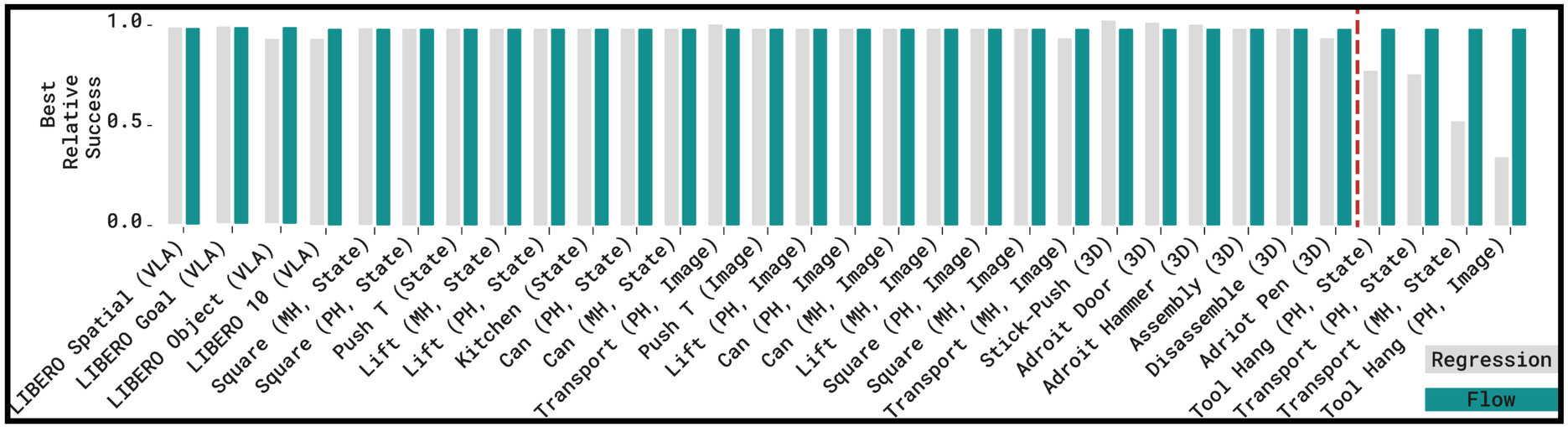
Train policy for predict **multimodal distributions of** $p(a \mid o)$

(e.g. ***denoising*** *diffusion models, DDPMs)*

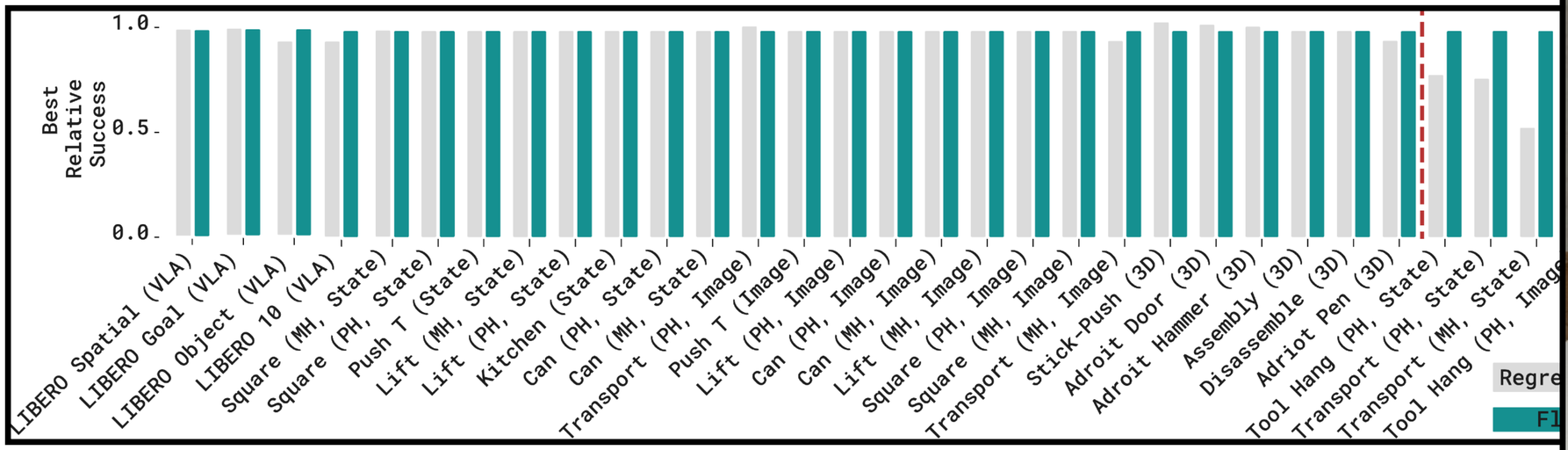
But perhaps, their claimed benefits are … **much ado about noising?**

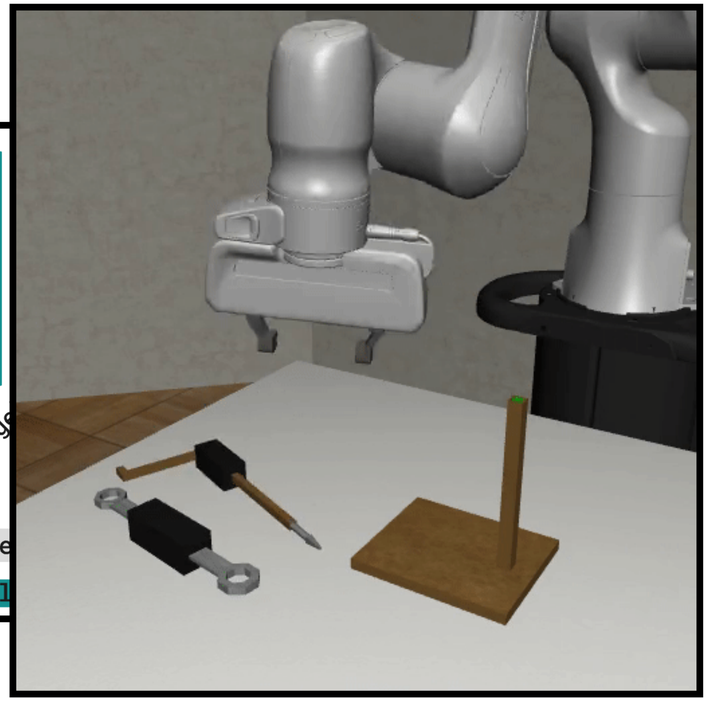
But do we need GCPs?

But do we need GCPs?



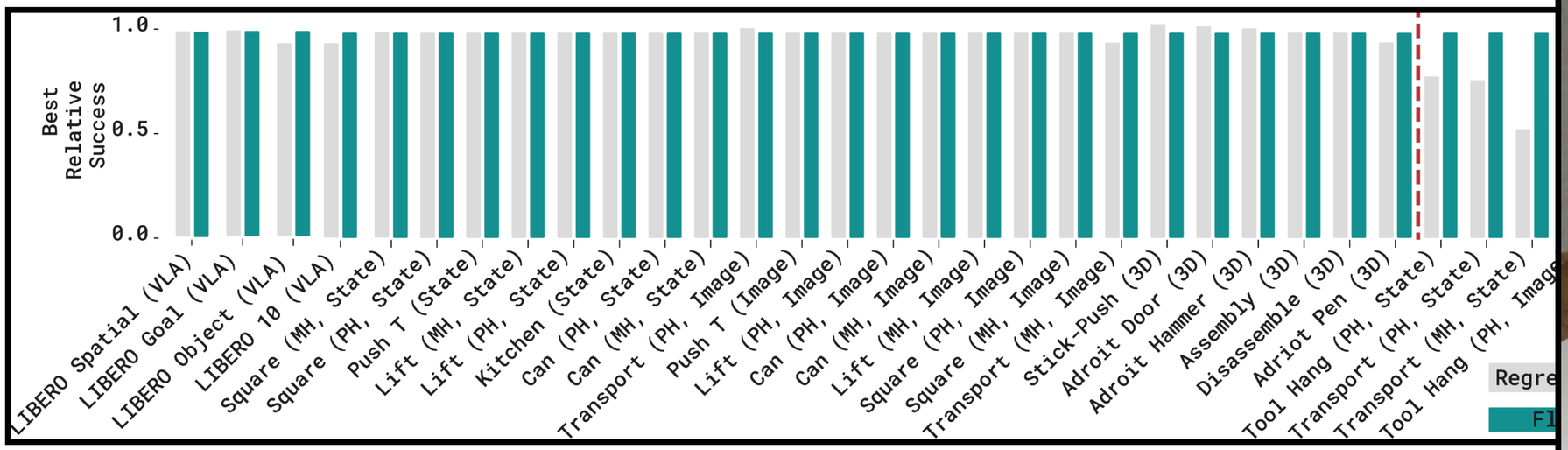
But do we need GCPs?

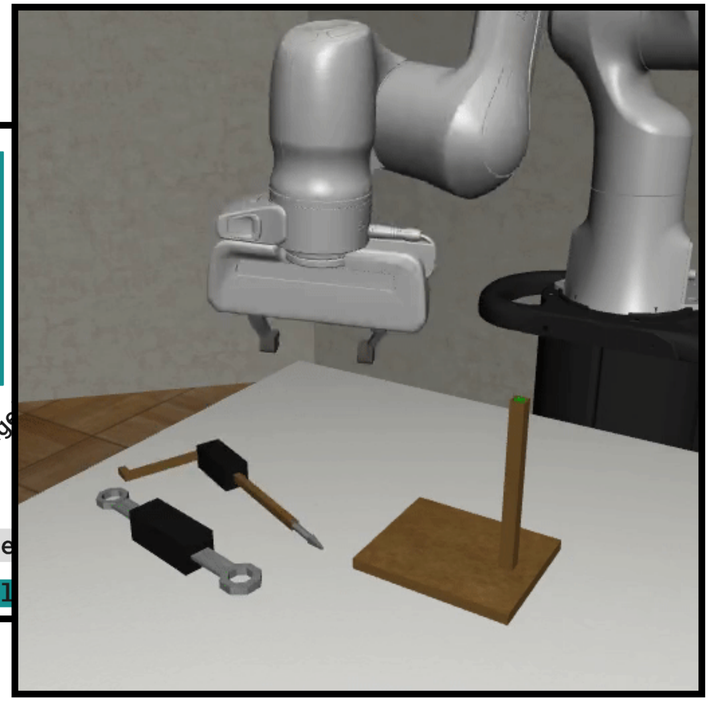




High-Precision, Not Multi-Modal Tasks

But do we need GCPs?





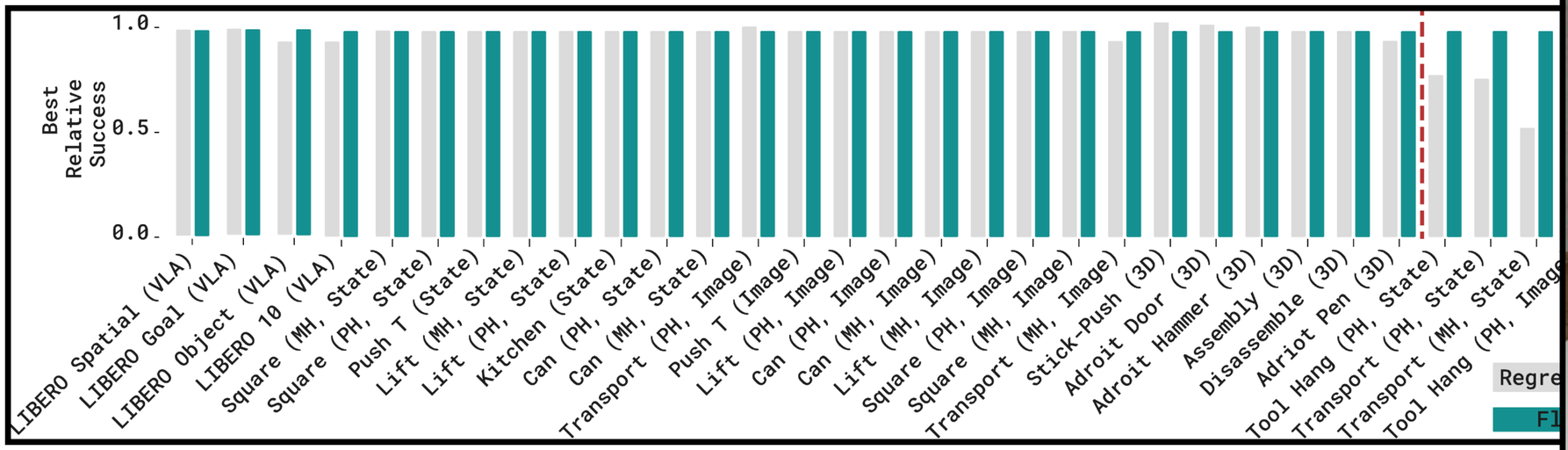
High-Precision, Not Multi-Modal Tasks

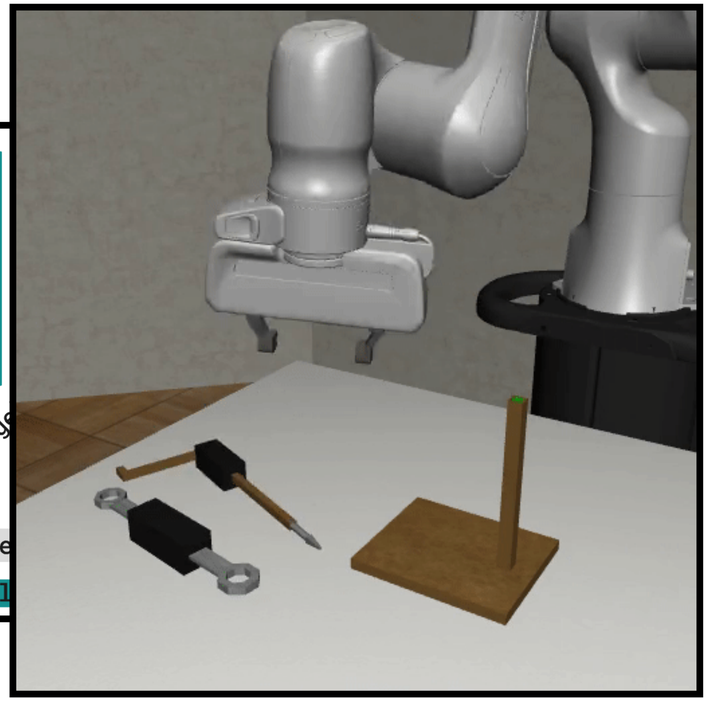
Much Ado About Noising: Dispelling the Myths of Generative Robotic Control

Chaoyi Pan[a,$] Giri Anantharaman[a] Nai-Chieh Huang[a] Claire Jin[a] Daniel Pfrommer[b]
Chenyang Yuan[c] Frank Permenter[c] Guannan Qu[a,†] Nicholas Boffi[a,†] Guanya Shi[a,†]
Max Simchowitz[a,†]

[$]Project lead. [†]Equal advising.

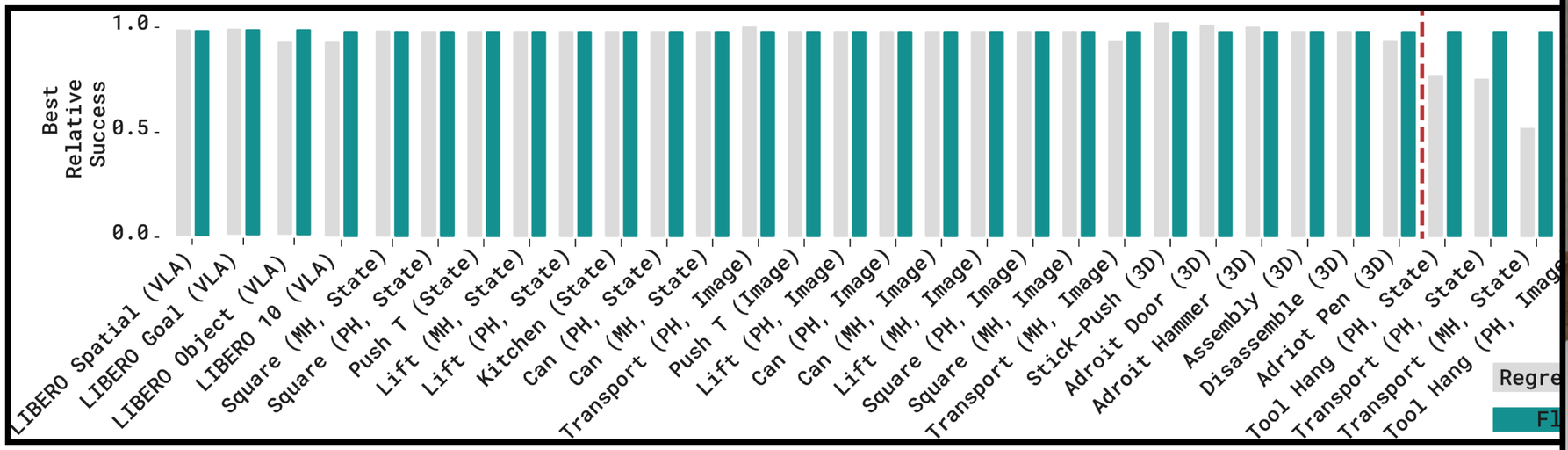
But do we need GCPs?

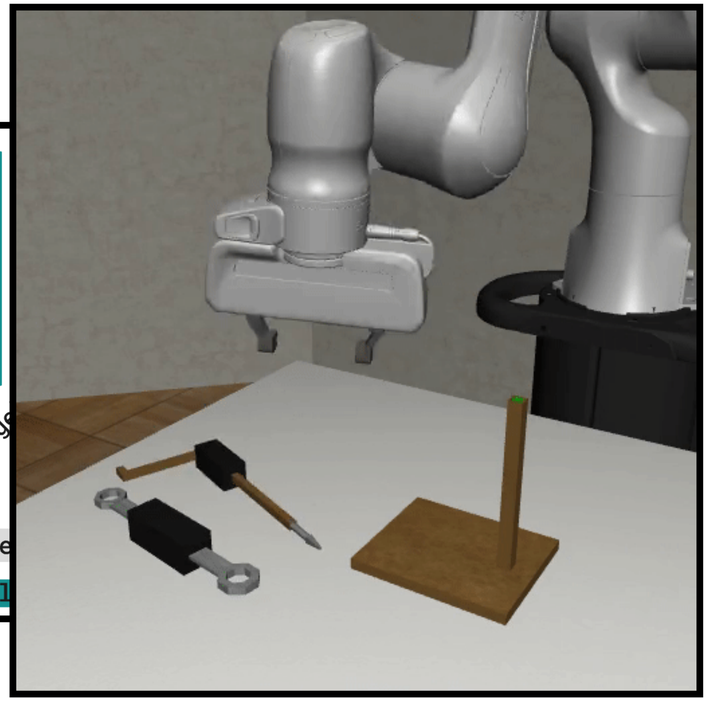




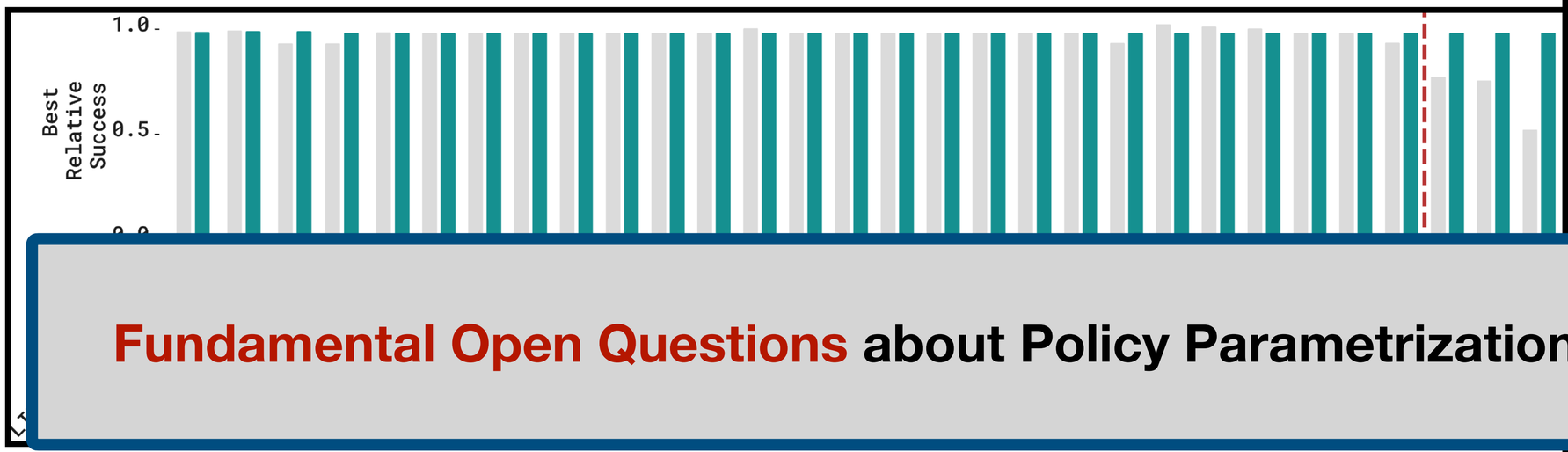
Along the way, two-step ‘minimal’ iterative policy (MIP) that recovers the benefits of flow on these high precision tasks

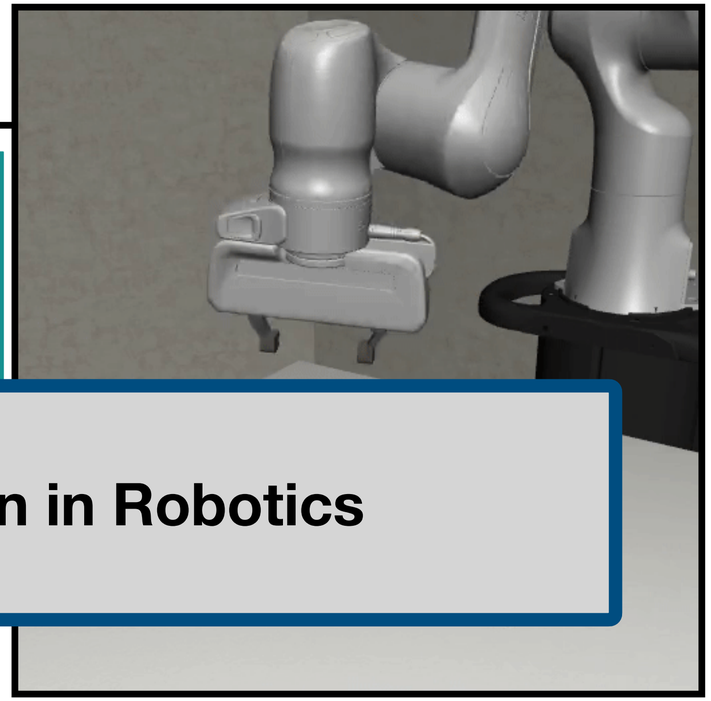
But do we need GCPs?





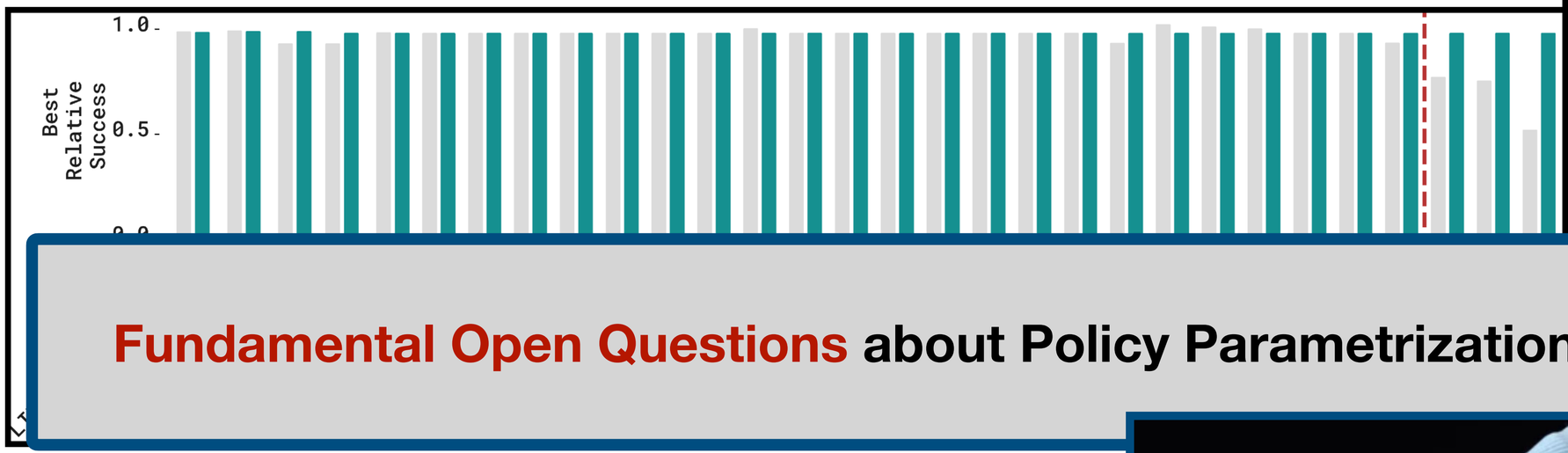
But do we need GCPs?

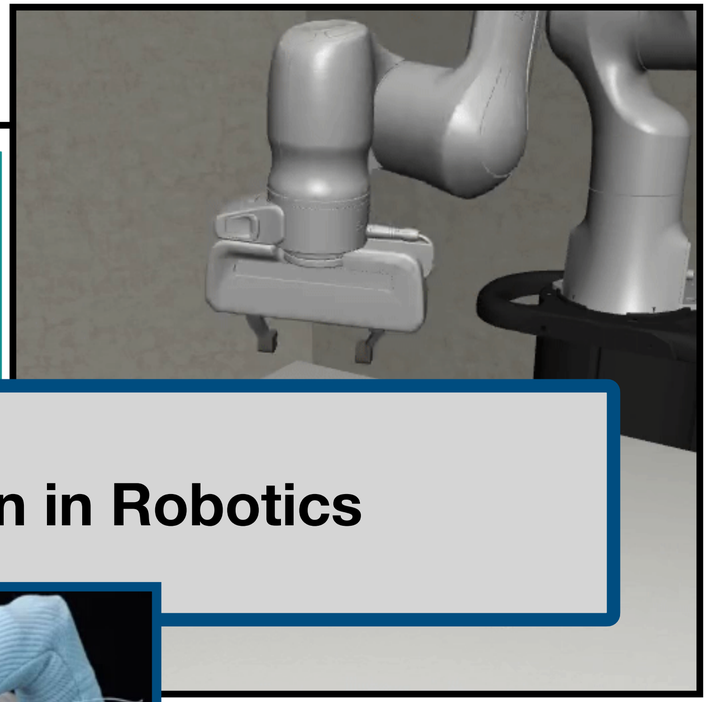




Fundamental Open Questions about Policy Parametrization in Robotics

But do we need GCPs?





Fundamental Open Questions about Policy Parametrization in Robotics

Outline

I. Background and Introduction.

II. Classical Imitation Learning.

III. Modern IL in discrete settings.

IV. Modern IL in continuous settings.

V. Opening the black box of optimization.

Outline

I. Background and Introduction.

II. Classical Imitation Learning.

III. Modern IL in discrete settings.

IV. Modern IL in continuous settings.

V. Opening the black box of optimization.

Previously…

Statistical limits of IL are fairly well understood.

Previously…

Statistical limits of IL are fairly well understood.

How do we instantiate the algorithms?

Previously…

Statistical limits of IL are fairly well understood.

How do we instantiate the algorithms?

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$.

Previously…

Statistical limits of IL are fairly well understood.

How do we instantiate the algorithms?

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\pi} \in \operatorname{argmin}_{\pi \in \Pi} L_{\mathsf{BC}}(\pi)$.

SGD

Previously…

Statistical limits of IL are fairly well understood.

How do we instantiate the algorithms?

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta L_{\mathrm{BC}}(\theta_t)$$

SGD

Previously…

Statistical limits of IL are fairly well understood.

How do we instantiate the algorithms?

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta L_{\mathrm{BC}}(\theta_t)$$ **SGD**

Adam, AdamW, etc.

Previously…

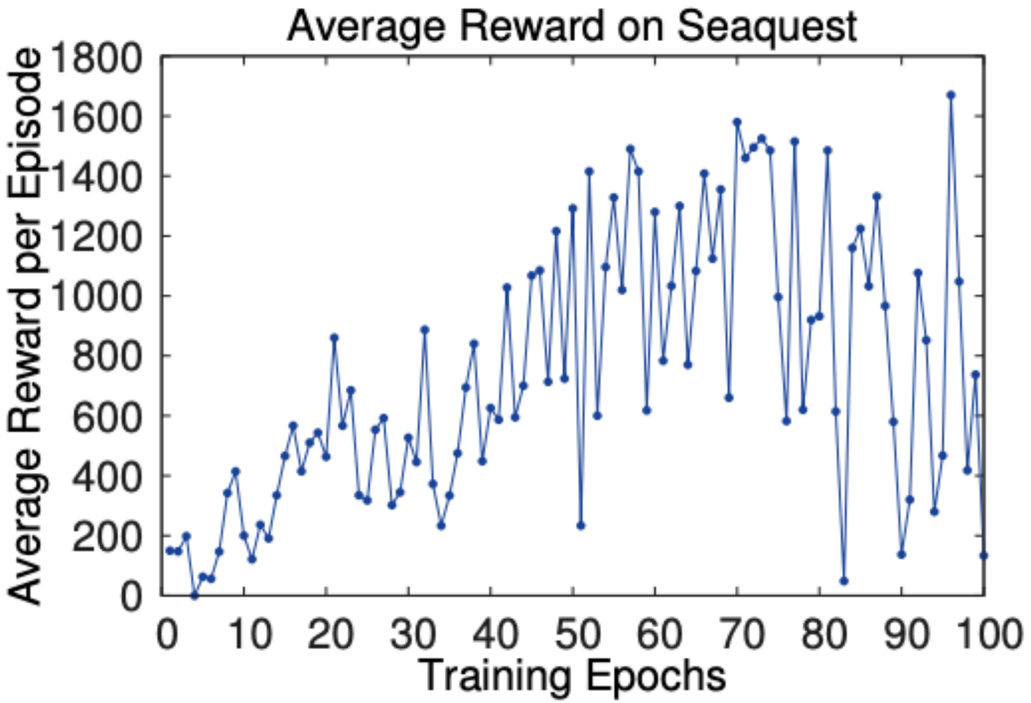
Statistical limits of IL are fairly well understood.

How do we instantiate the algorithms?

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta L_{\mathrm{BC}}(\theta_t)$$ **SGD**

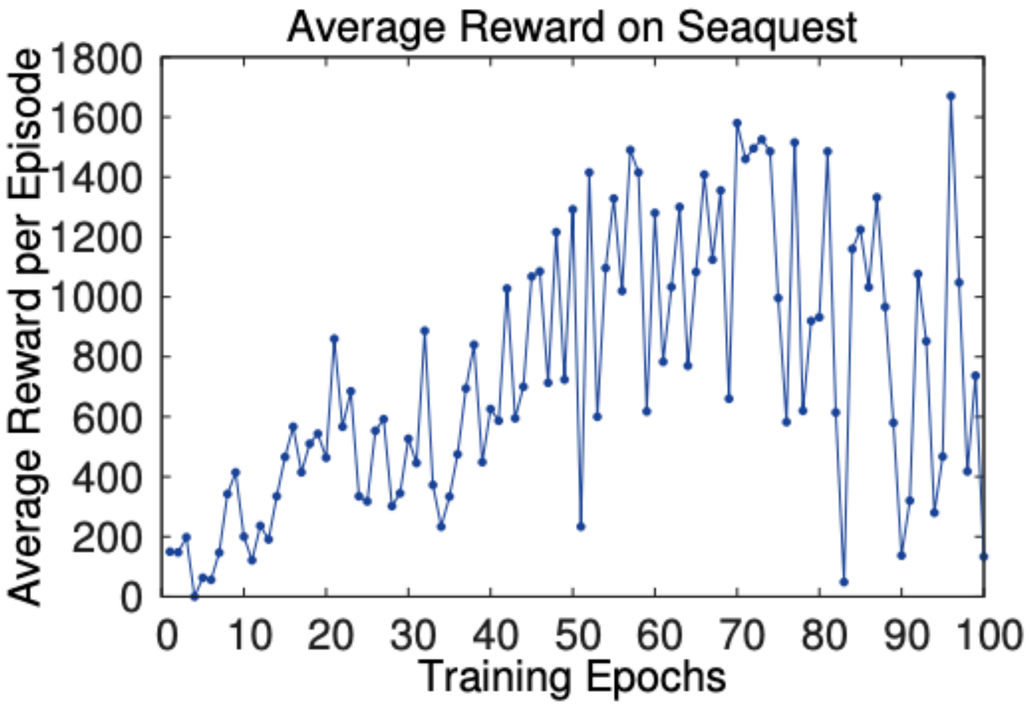
Adam, AdamW, etc. (Also for RL)

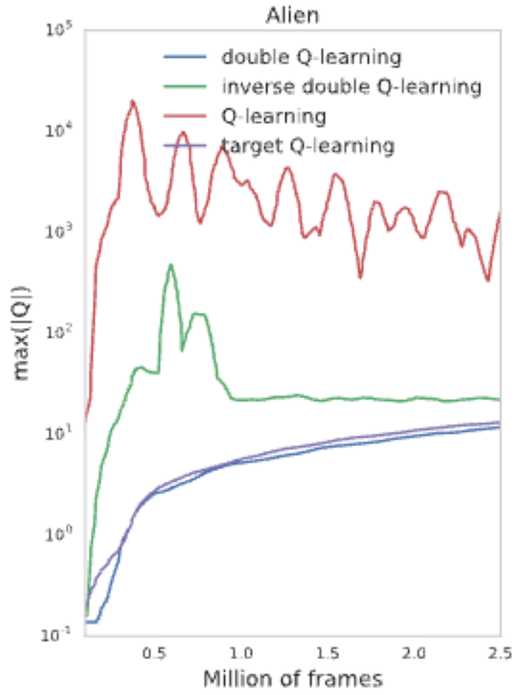
Reinforcement Learning is unstable…

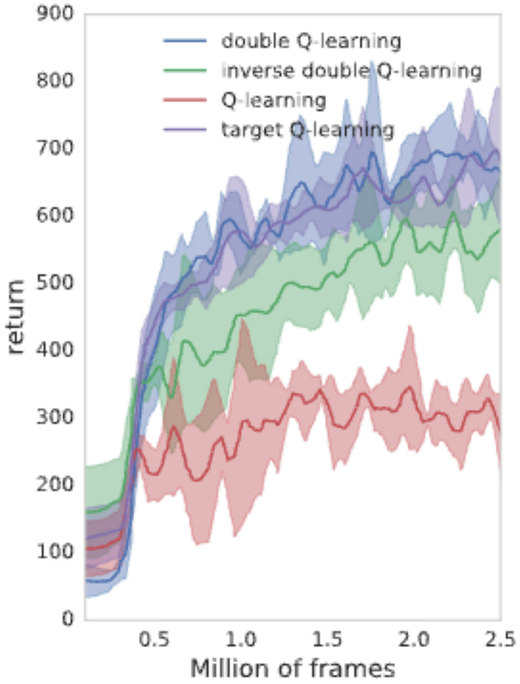


Playing Atari with Deep Reinforcement Learning, Mnih et al (2013); *Deep Reinforcement Learning and the Deadly Triad*, van Hasselt et al (2018)

Reinforcement Learning is unstable…

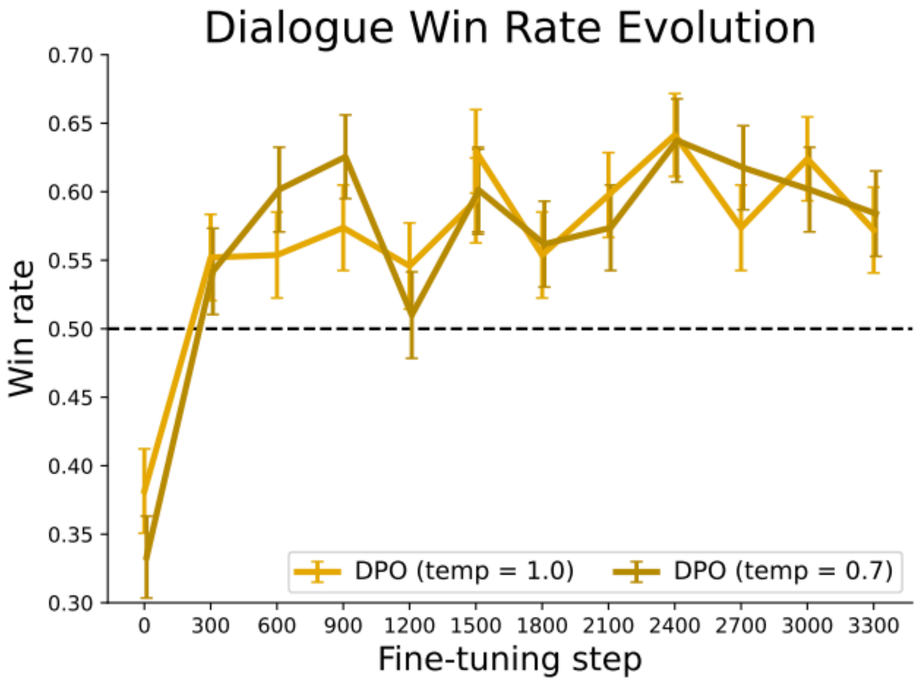






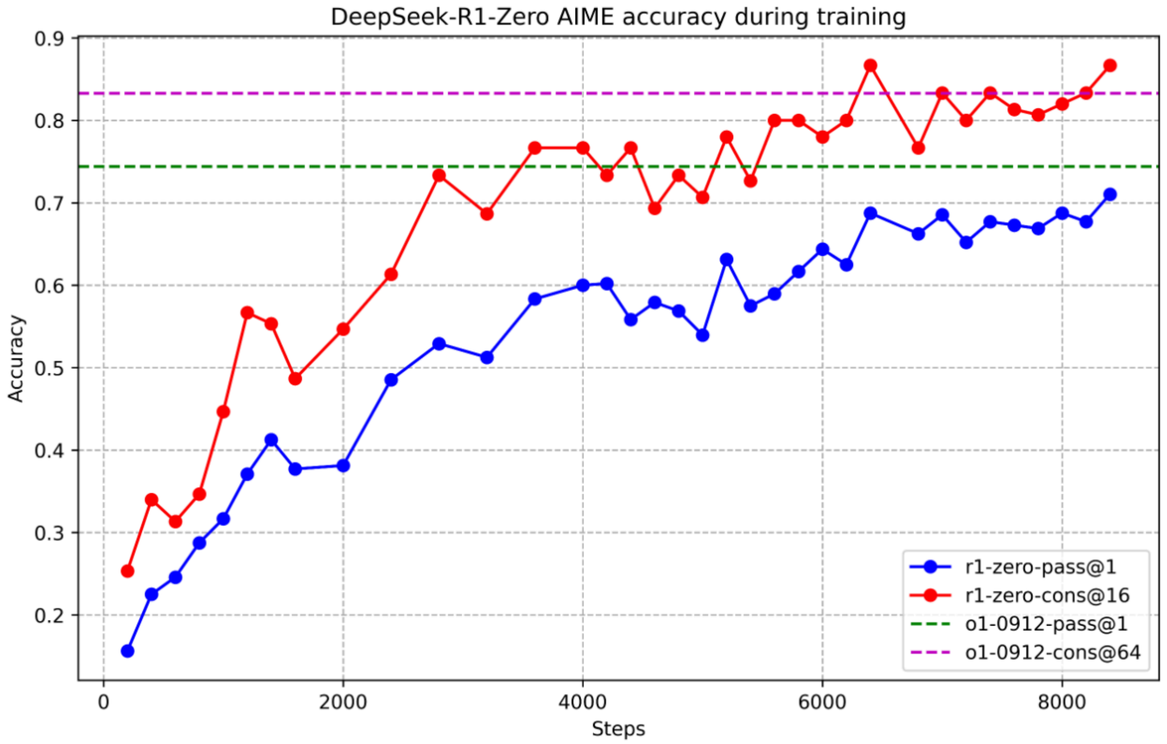
Playing Atari with Deep Reinforcement Learning, Mnih et al (2013); *Deep Reinforcement Learning and the Deadly Triad*, van Hasselt et al (2018)

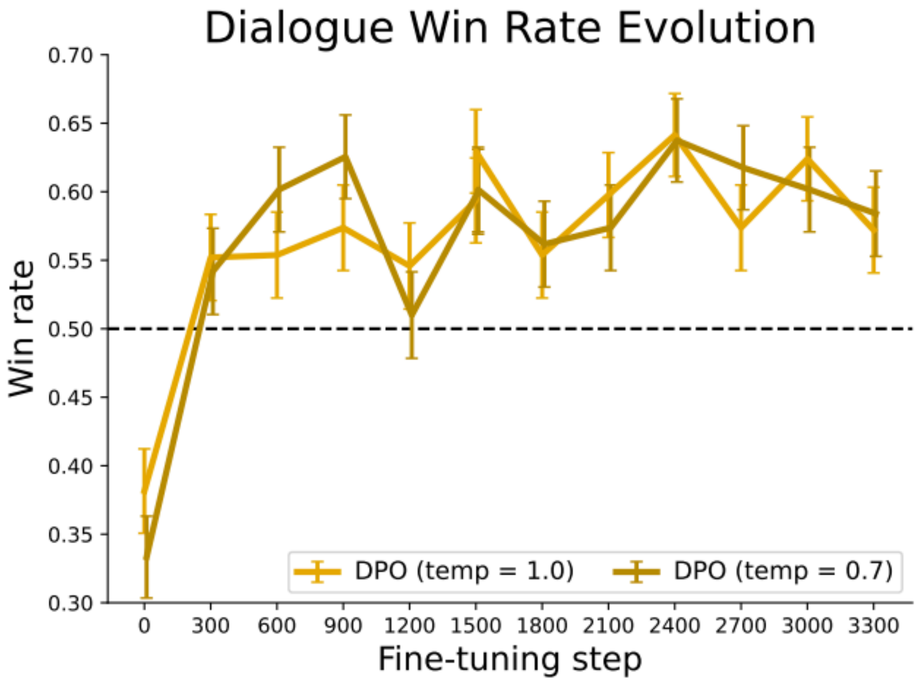
…Even in LMs…



DeepSeek-R1, DeepSeek-AI (2025); *Direct Preference Optimization: Your Language Model is Secretly a Reward Model*, Rafailov et al (2023)

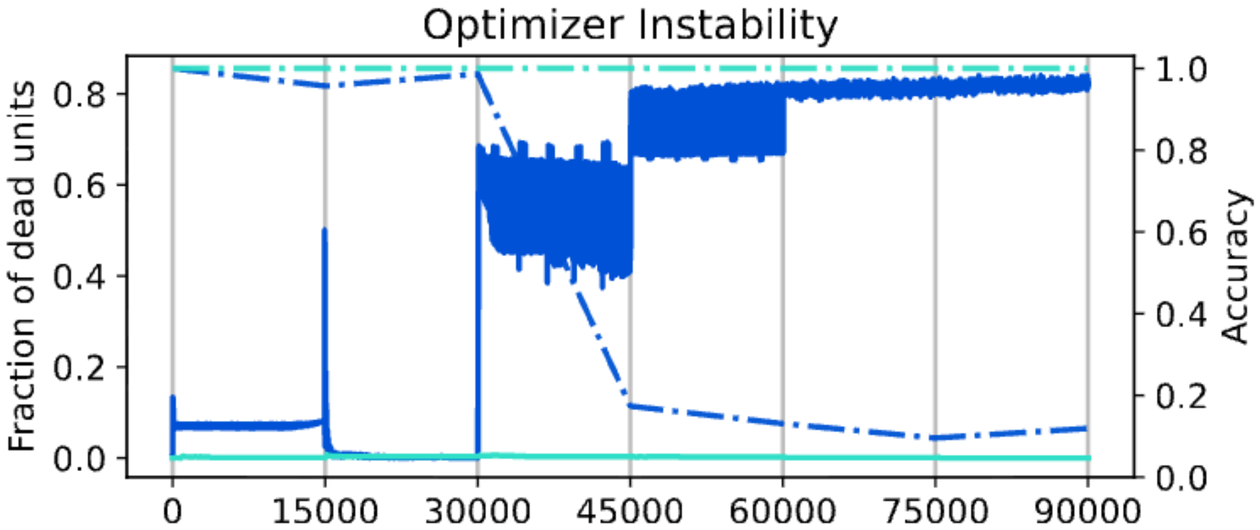
…Even in LMs…

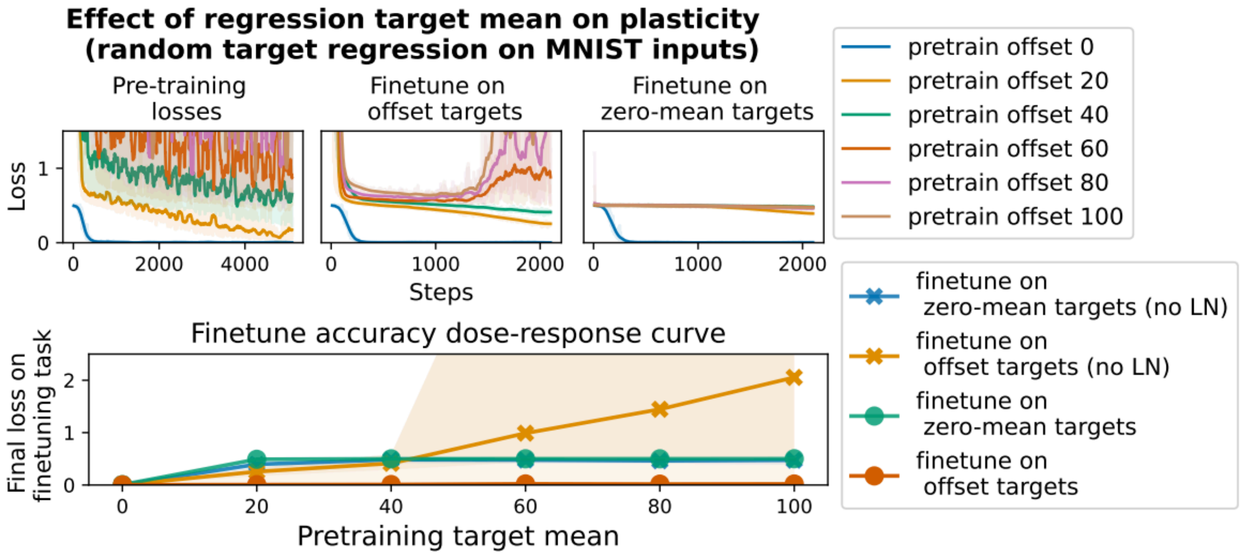




DeepSeek-R1, DeepSeek-AI (2025); *Direct Preference Optimization: Your Language Model is Secretly a Reward Model*, Rafailov et al (2023)

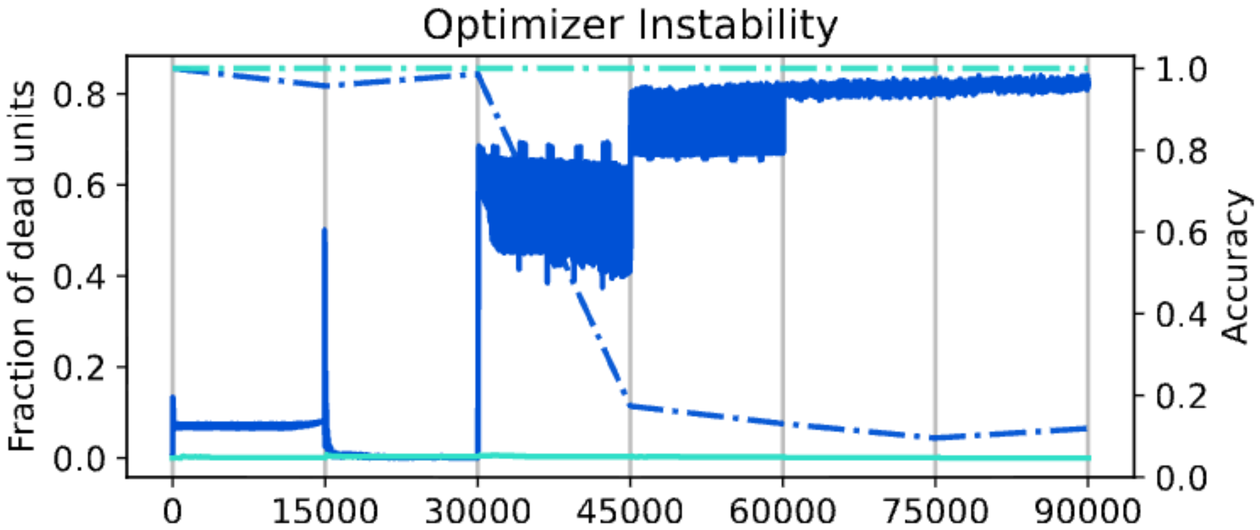
…Due to nonstationarity

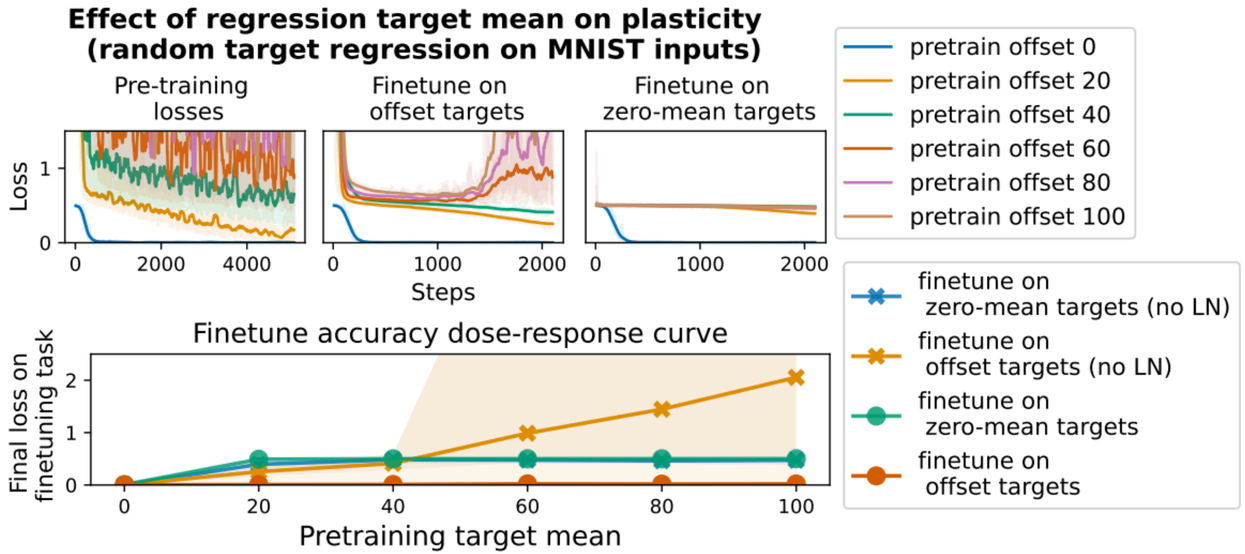




Understanding Plasticity in Neural Networks, Lyle et al (2023); *Disentangling the Causes of Plasticity Loss in Neural Networks*, Lyle et al (2024)

…Due to nonstationarity

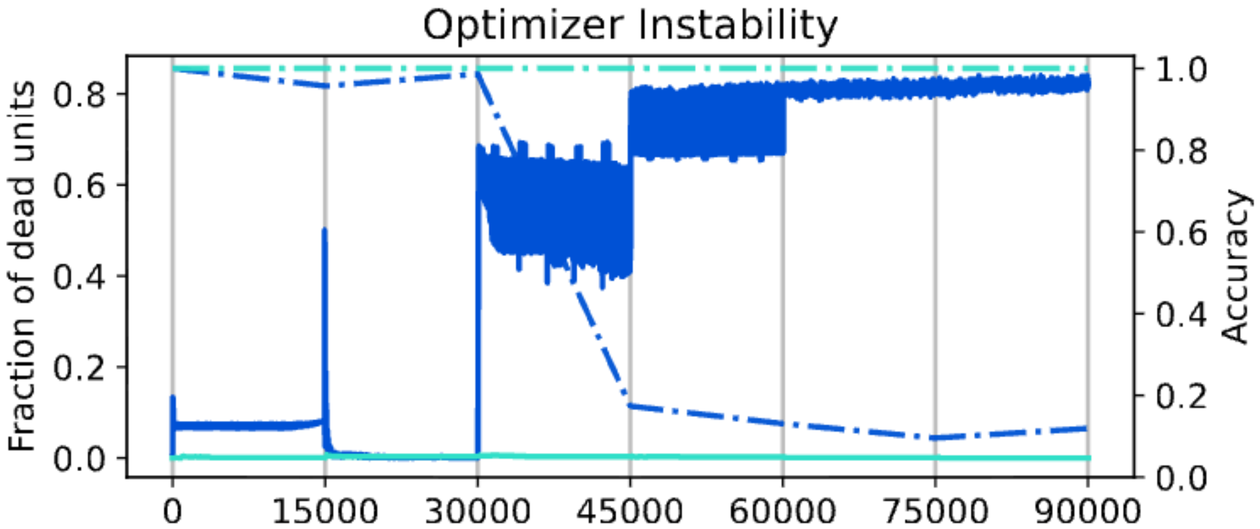


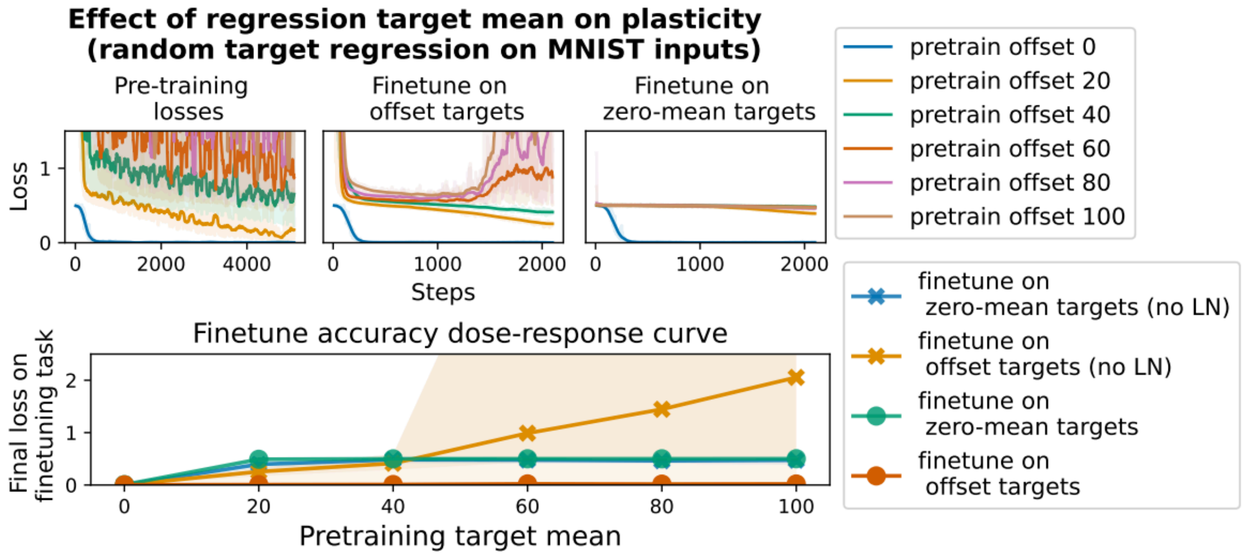


Behavior cloning is stationary.

Understanding Plasticity in Neural Networks, Lyle et al (2023); *Disentangling the Causes of Plasticity Loss in Neural Networks*, Lyle et al (2024)

…Due to nonstationarity



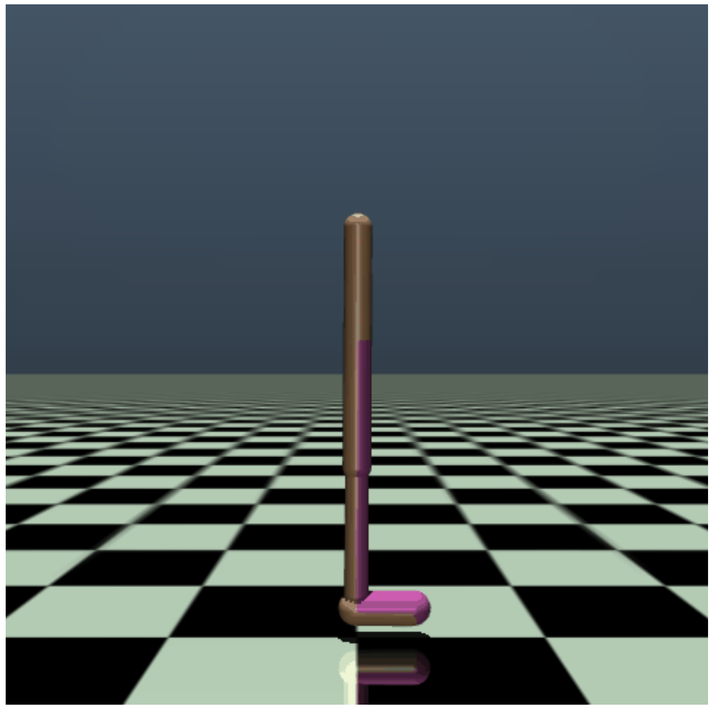


Behavior cloning is stationary.

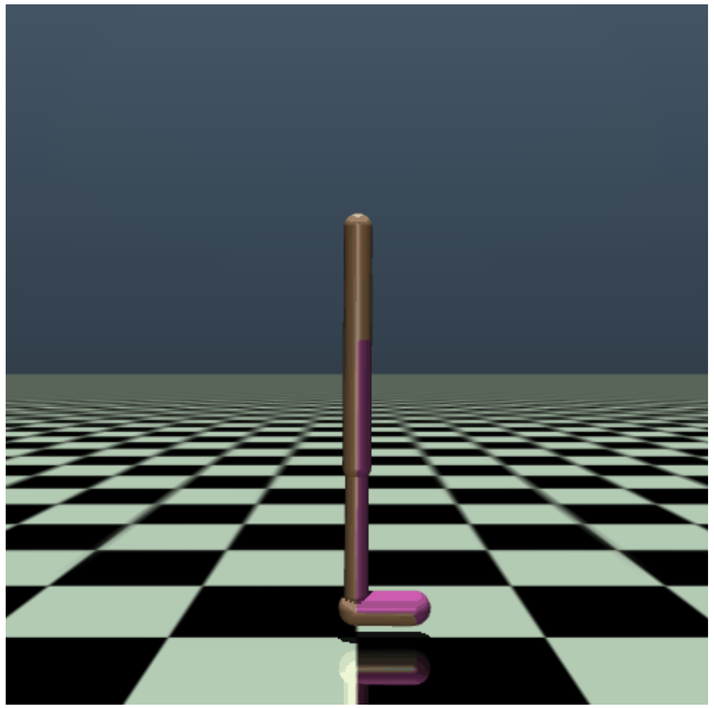
IL should be stable, right?

Understanding Plasticity in Neural Networks, Lyle et al (2023); *Disentangling the Causes of Plasticity Loss in Neural Networks*, Lyle et al (2024)

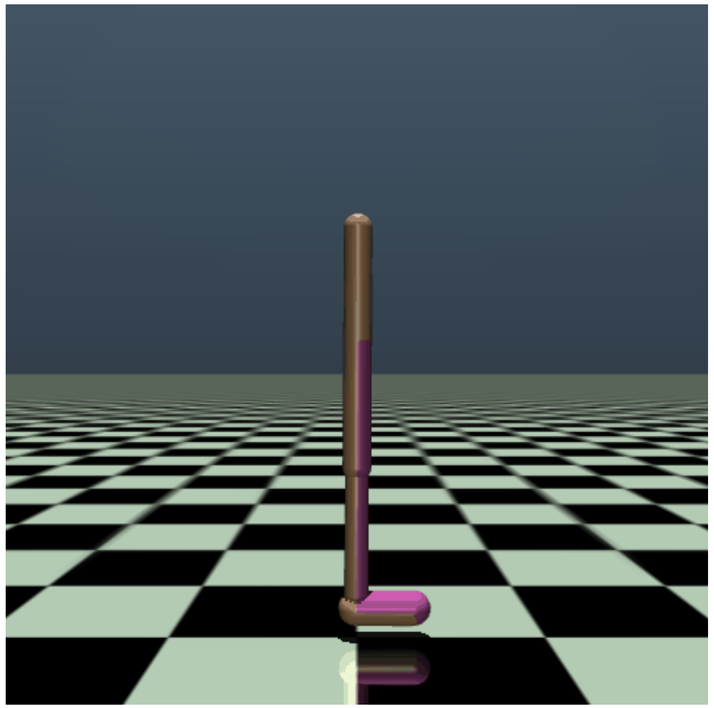
Walker2d (MuJoCo)

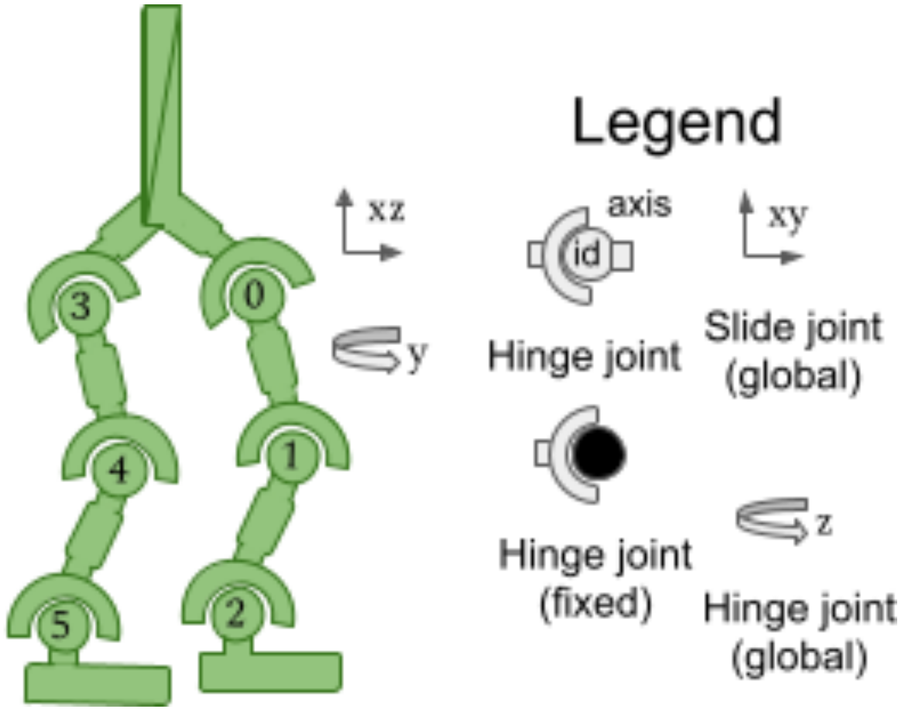


Walker2d (MuJoCo)



Walker2d (MuJoCo)





A simple experiment

1. Train $\pi^\star$ using RL.

A simple experiment

1. Train $\pi^\star$ using RL.

2. Roll out $\pi^\star$ to generate trajectories $\tau^{(1)}, \ldots, \tau^{(n)}$.

A simple experiment

1. Train $\pi^\star$ using RL.

2. Roll out $\pi^\star$ to generate trajectories $\tau^{(1)}, \ldots, \tau^{(n)}$.

3. Let $L_{\text{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \left\| a_h^{(i)} - \pi(s_h^{(i)}) \right\|^2$. Train so $L_{\text{BC}}(\hat{\pi}) \ll 1$.

A simple experiment

1. Train $\pi^\star$ using RL.

2. Roll out $\pi^\star$ to generate trajectories $\tau^{(1)}, \ldots, \tau^{(n)}$.

3. Let $L_{\text{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \left\| a_h^{(i)} - \pi(s_h^{(i)}) \right\|^2$. Train so $L_{\text{BC}}(\hat{\pi}) \ll 1$.

4. Roll out $\hat{\pi}$ to evaluate $J(\hat{\pi})$.

A simple experiment

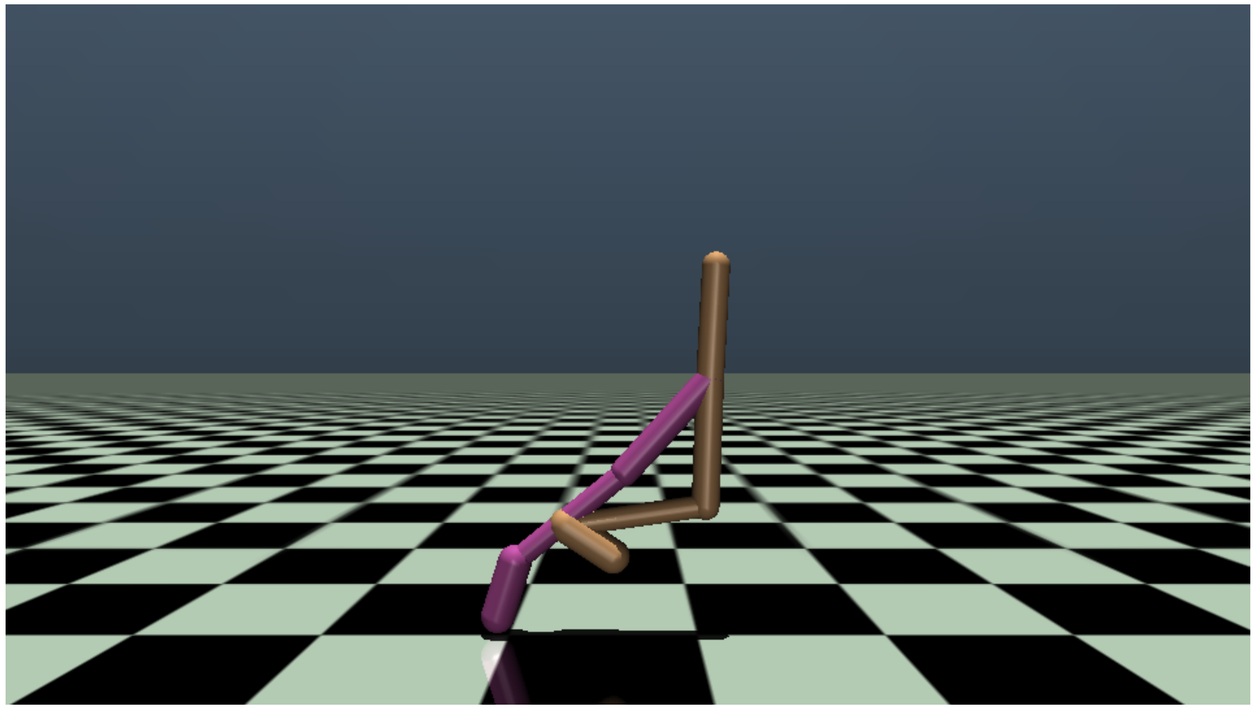
1. Train $\pi^\star$ using RL.

2. Roll out $\pi^\star$ to generate trajectories $\tau^{(1)}, \ldots, \tau^{(n)}$.

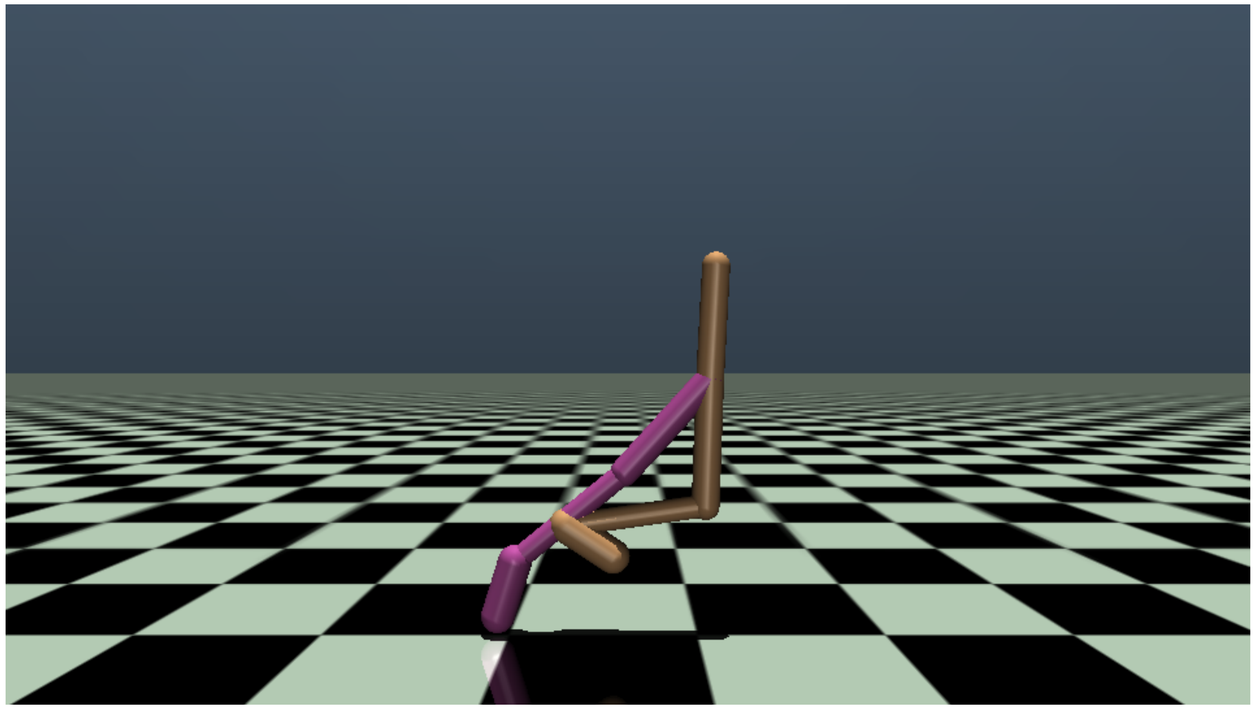
3. Let $L_{\text{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \left\| a_h^{(i)} - \pi(s_h^{(i)}) \right\|^2$. Train so $L_{\text{BC}}(\hat{\pi}) \ll 1$.

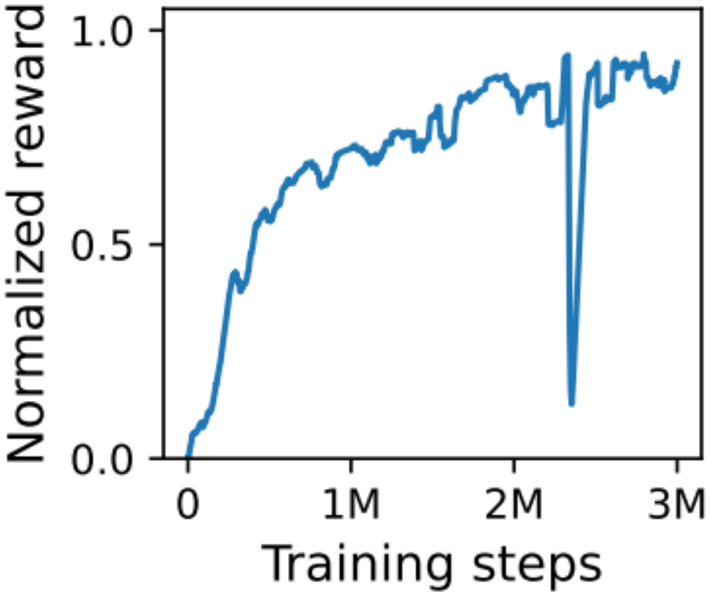
4. Roll out $\hat{\pi}$ to evaluate $J(\hat{\pi})$.

A simple experiment: RL

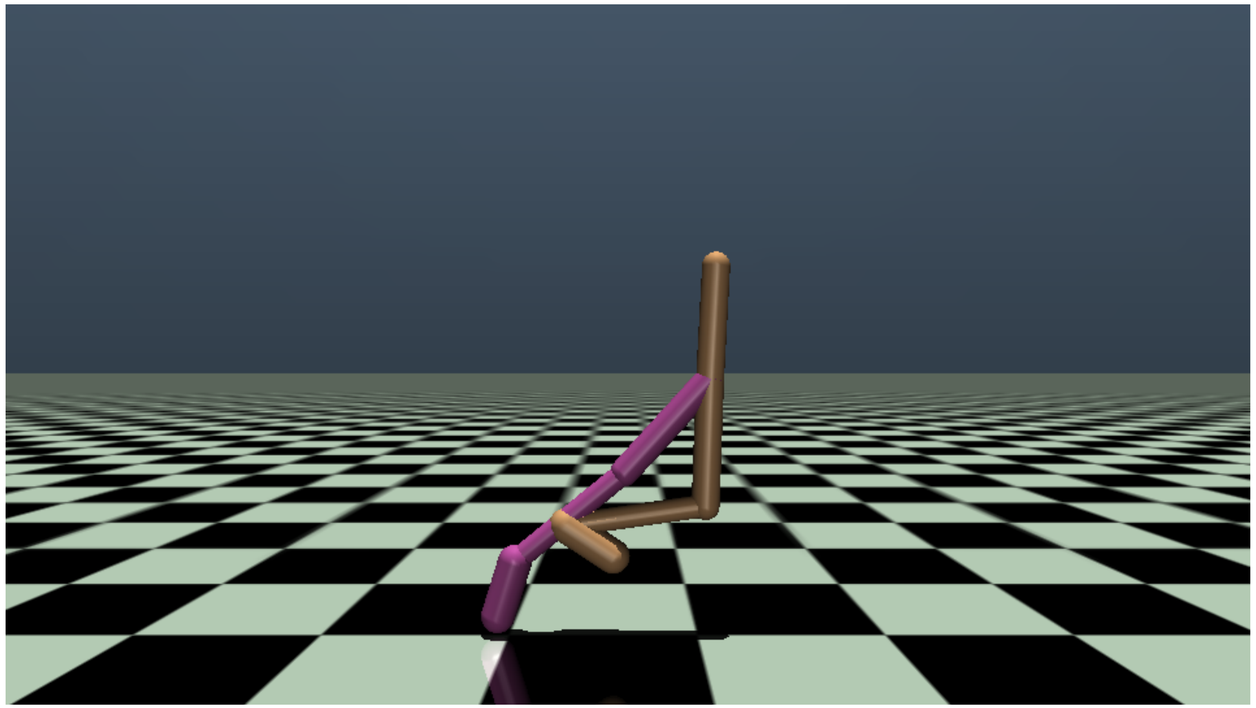


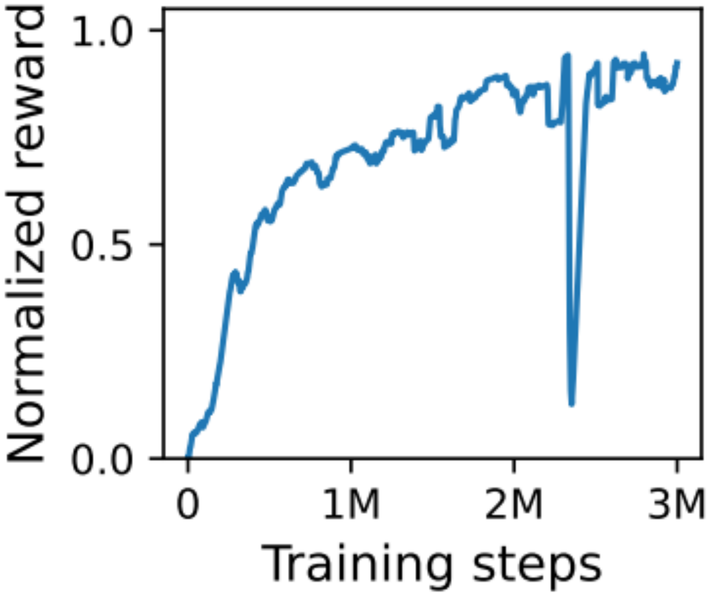
A simple experiment: RL

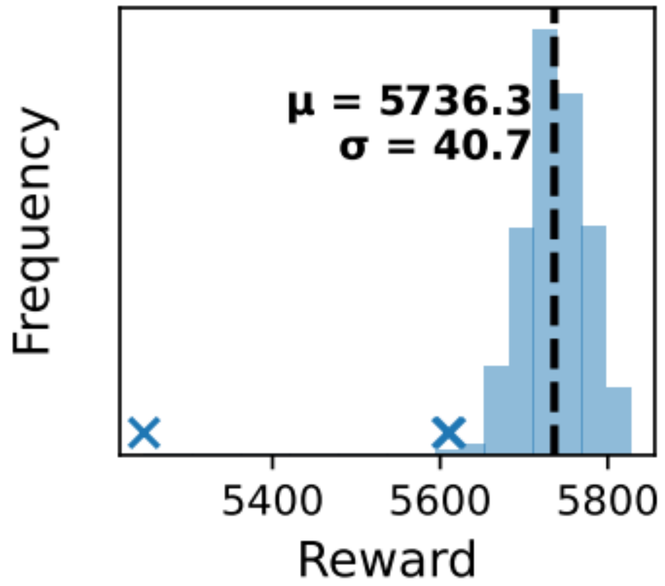




A simple experiment: RL







A simple experiment

1. Train $\pi^\star$ using RL.

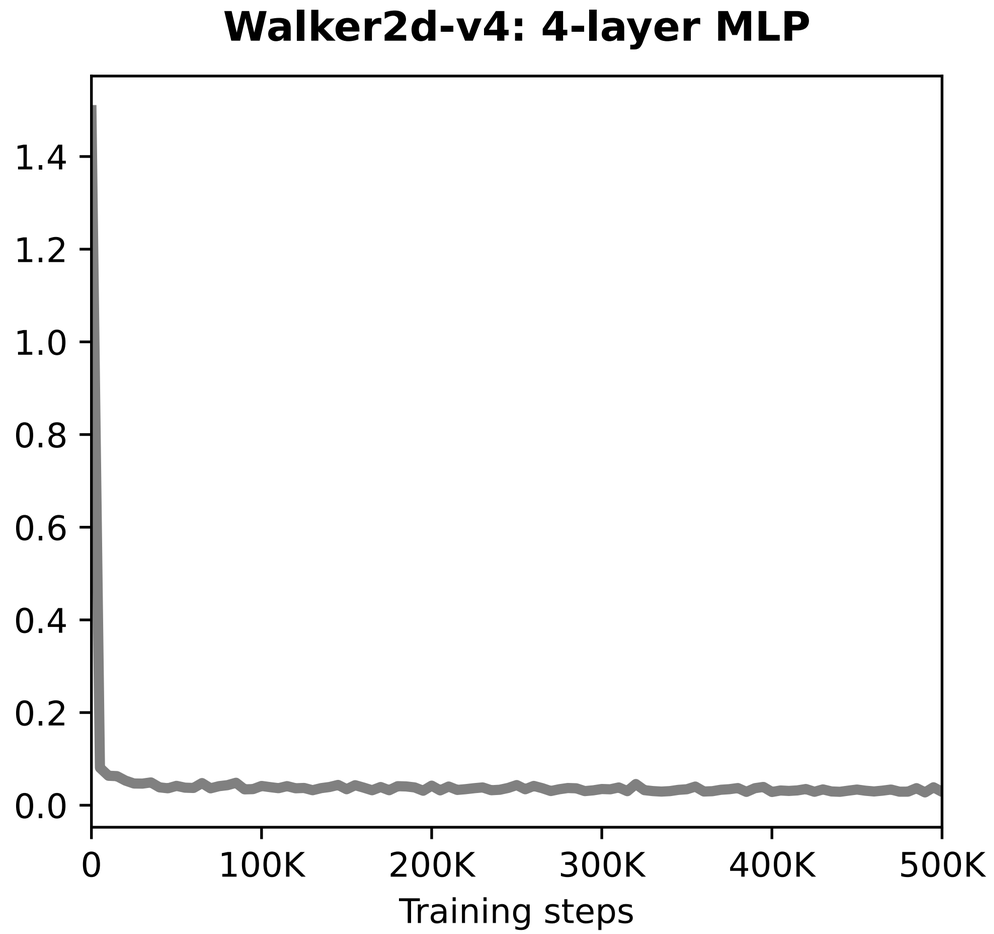
2. Roll out $\pi^\star$ to generate trajectories $\tau^{(1)}, \ldots, \tau^{(n)}$.

3. Let $L_{\text{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \left\| a_h^{(i)} - \pi(s_h^{(i)}) \right\|^2$. Train so $L_{\text{BC}}(\hat{\pi}) \ll 1$.

4. Roll out $\hat{\pi}$ to evaluate $J(\hat{\pi})$.

A simple experiment: IL

$$L_{\mathrm{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \left\| a_h^{(i)} - \pi(s_h^{(i)}) \right\|^2$$



A simple experiment

1. Train $\pi^\star$ using RL.

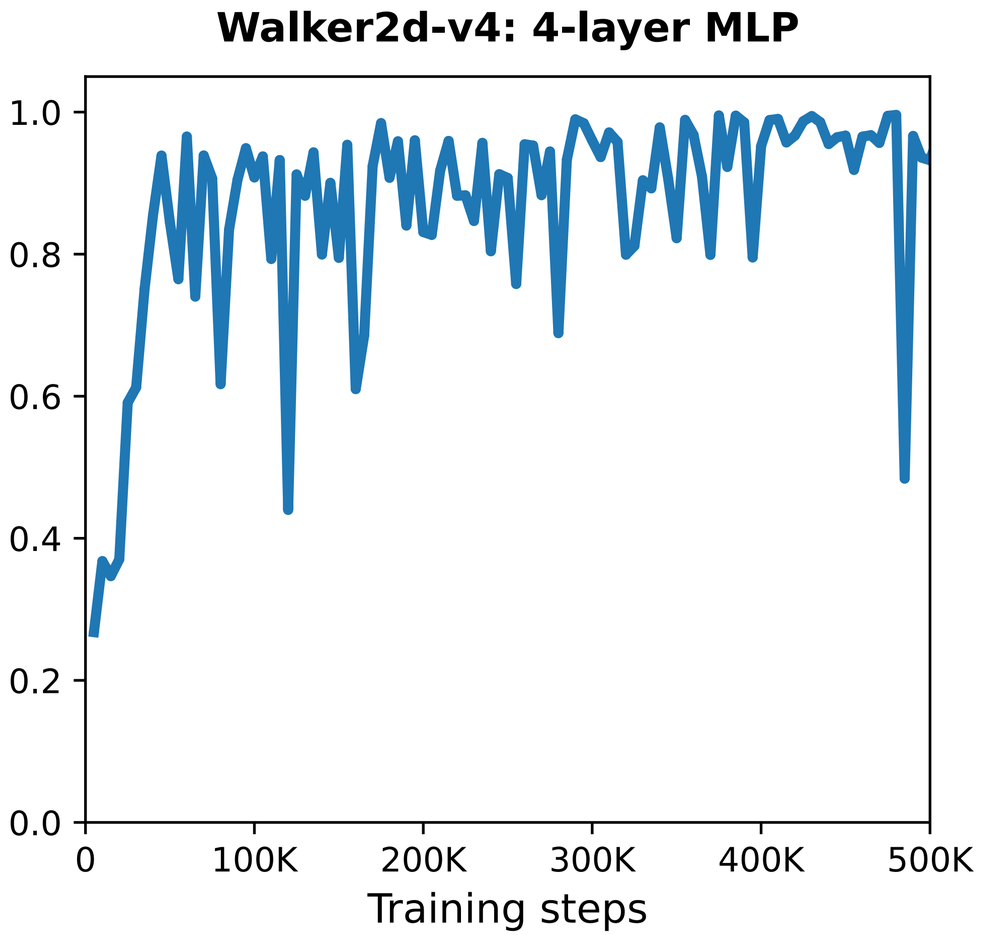
2. Roll out $\pi^\star$ to generate trajectories $\tau^{(1)}, \ldots, \tau^{(n)}$.

3. Let $L_{\text{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \left\| a_h^{(i)} - \pi(s_h^{(i)}) \right\|^2$. Train so $L_{\text{BC}}(\hat{\pi}) \ll 1$.

4. Roll out $\hat{\pi}$ to evaluate $J(\hat{\pi})$.

A simple experiment: Rolling out

$$J(\pi) = \mathbb{E}^{\pi}\left[\sum_{h=1}^{H} r(s_h, a_h)\right]$$

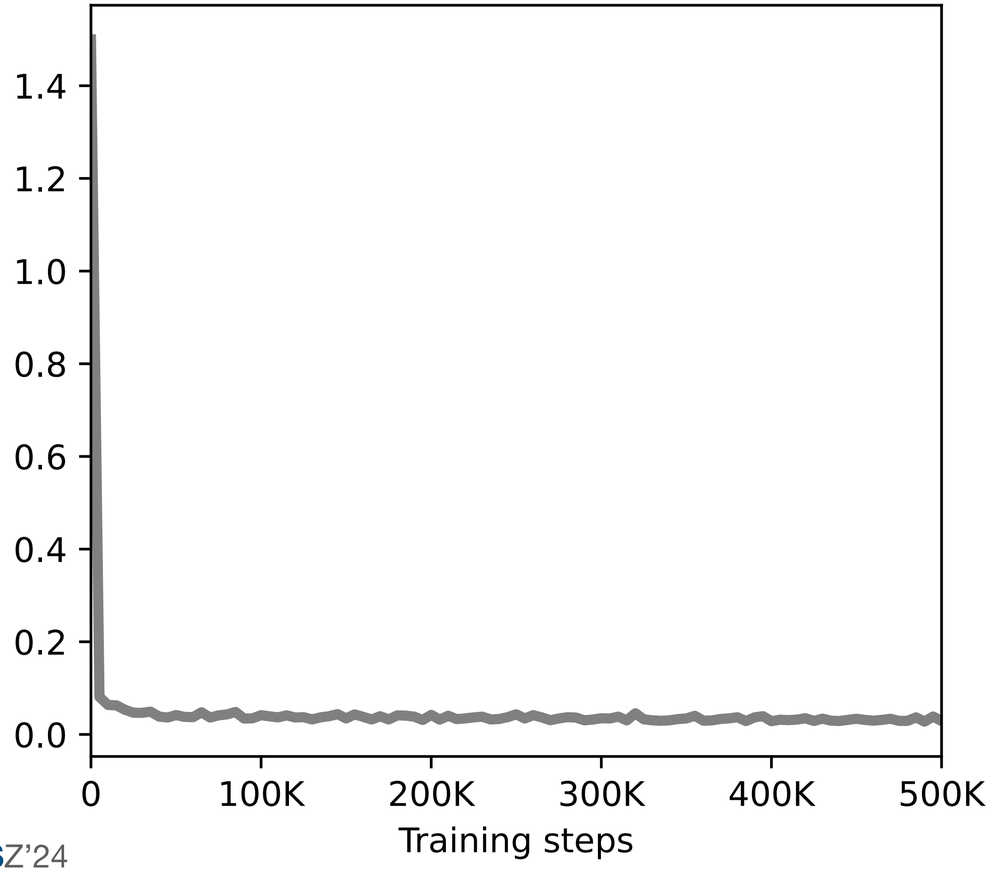


A simple experiment?

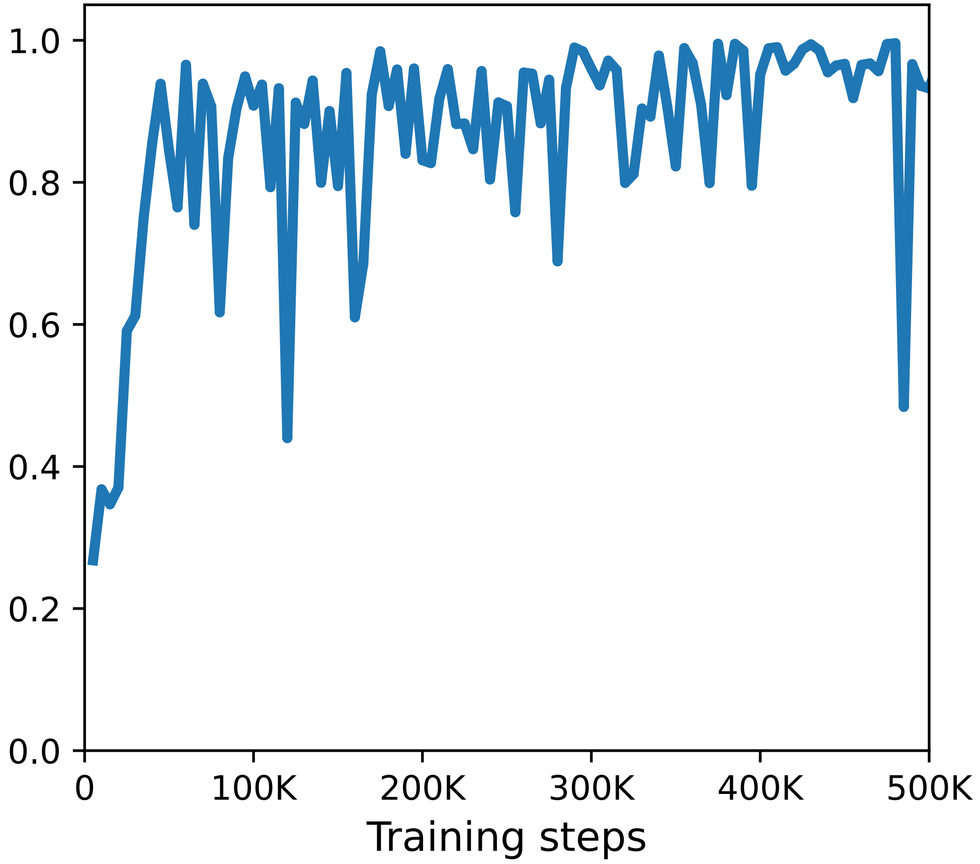
$$L_{\text{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \left\| a_h^{(i)} - \pi(s_h^{(i)}) \right\|^2$$

$$J(\pi) = \mathbb{E}^{\pi} \left[\sum_{h=1}^{H} r(s_h, a_h) \right]$$

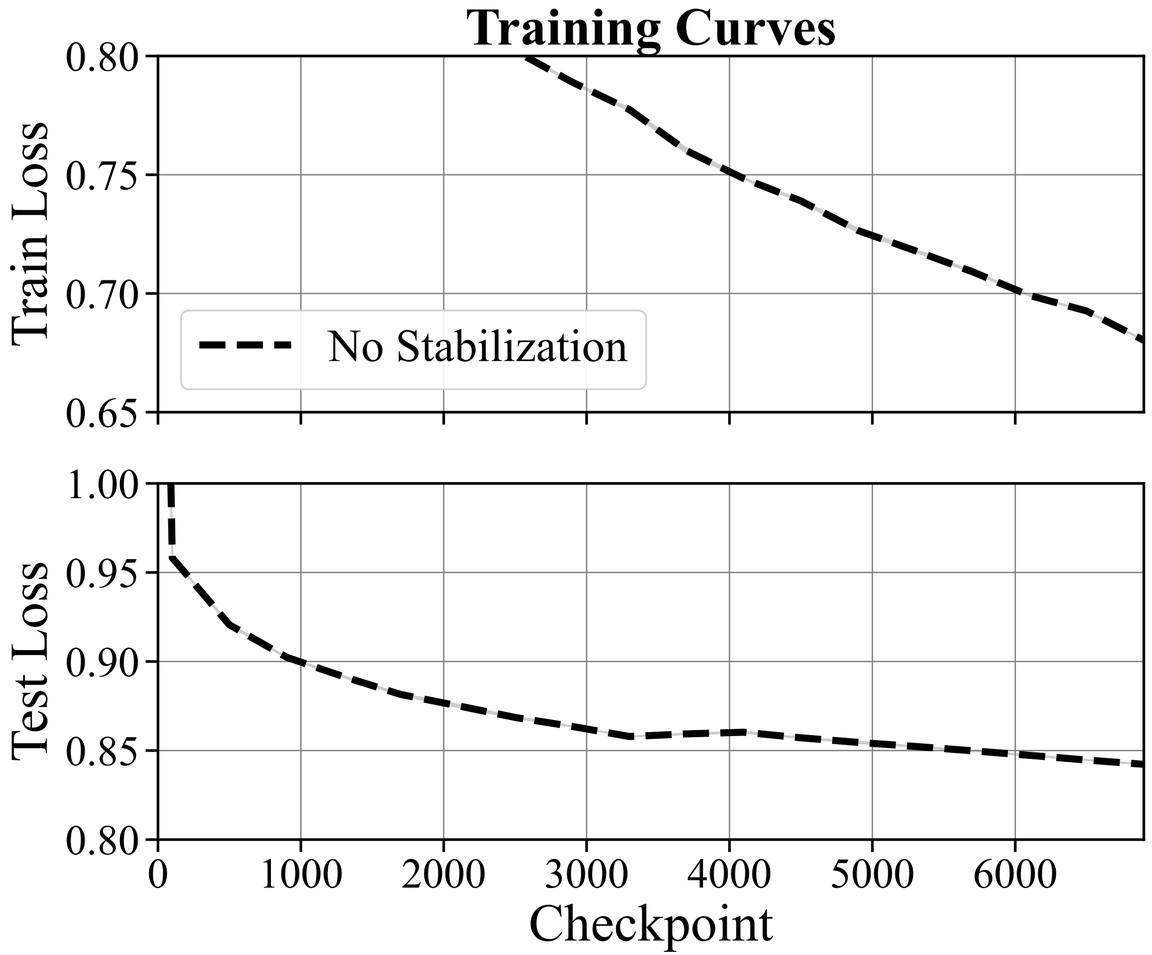
Walker2d-v4: 4-layer MLP



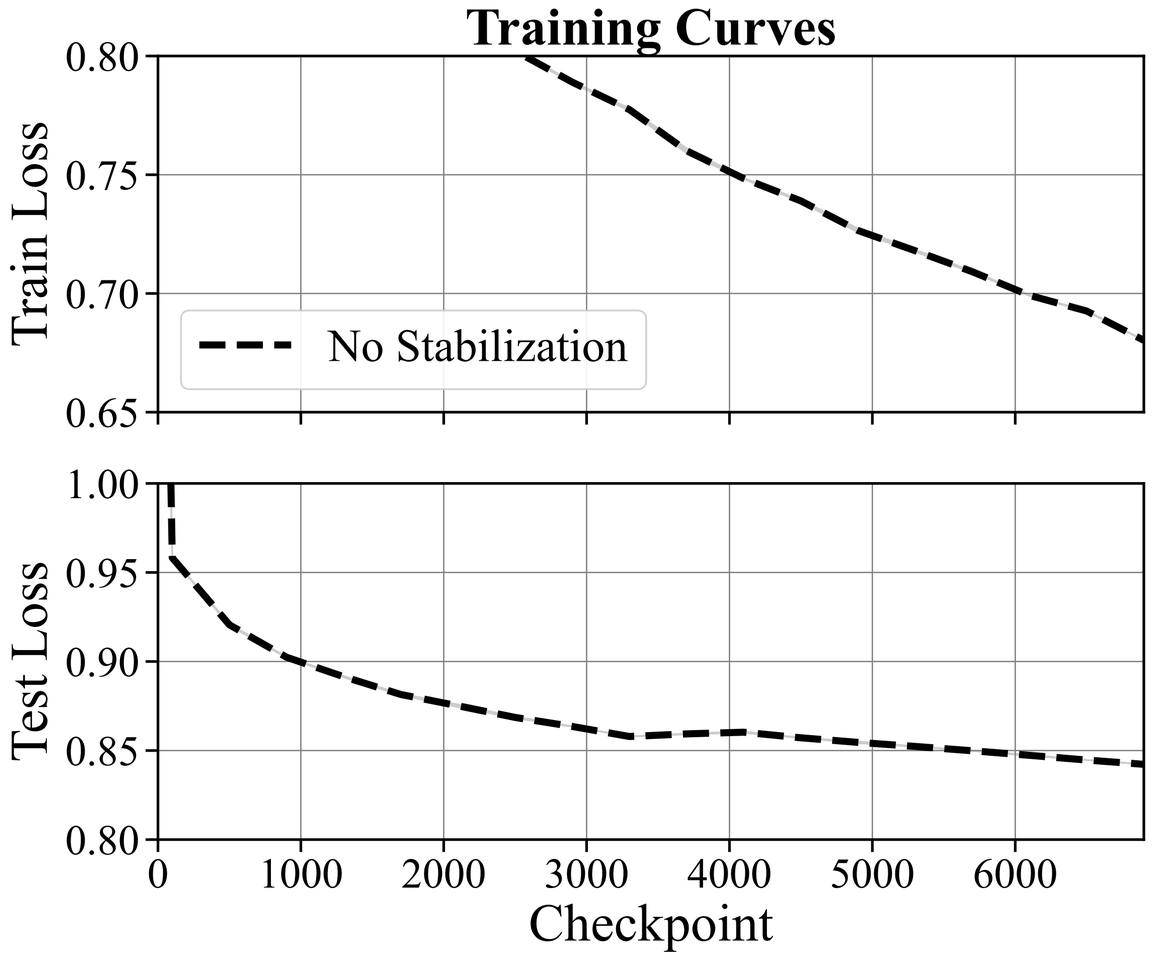
Walker2d-v4: 4-layer MLP

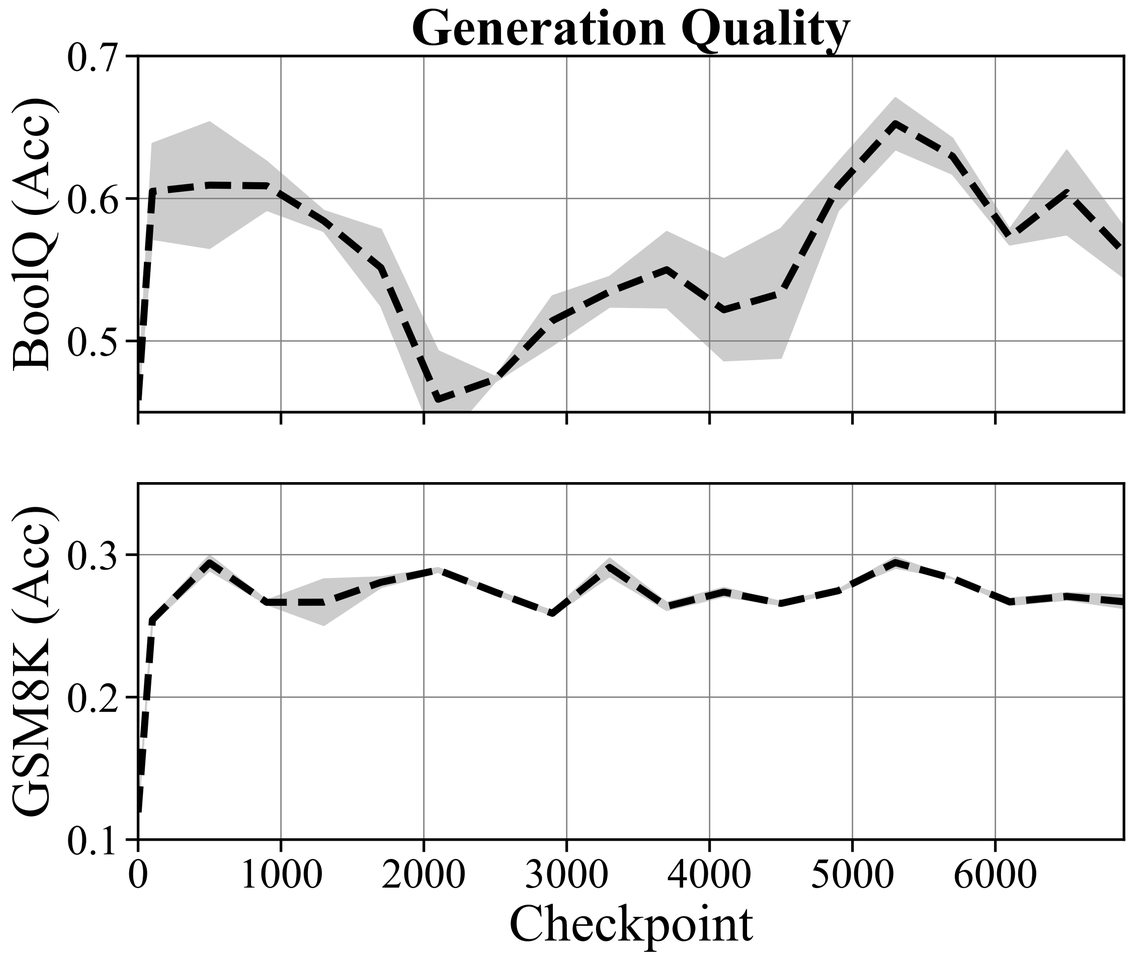


Does instability arise in LMs?



Does instability arise in LMs?





1. What causes these **wild oscillations** across different **problem settings**?

1. What causes these **wild oscillations** across different **problem settings**?

2. How can we **stabilize** training effectively and efficiently?

1. What causes these **wild oscillations** across different **problem settings**?

2. How can we **stabilize** training effectively and efficiently?

Optimization matters

Parameterize the policy class $\Pi = \{\pi_\theta | \theta \in \Theta \subset \mathbb{R}^d\}$.

Optimization matters

Parameterize the policy class $\Pi = \{\pi_\theta | \theta \in \Theta \subset \mathbb{R}^d\}$.

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\theta} \in \operatorname{argmin}_{\theta \in \Theta} L_{\mathsf{BC}}(\pi_\theta)$.

Optimization matters

Parameterize the policy class $\Pi = \{\pi_\theta | \theta \in \Theta \subset \mathbb{R}^d\}$.

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\theta} \in \operatorname{argmin}_{\theta \in \Theta} L_{\mathsf{BC}}(\pi_\theta)$.

Approximate minimization found through SGD.

Optimization matters

Parameterize the policy class $\Pi = \{\pi_\theta | \theta \in \Theta \subset \mathbb{R}^d\}$.

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\theta} \in \operatorname{argmin}_{\theta \in \Theta} L_{\mathsf{BC}}(\pi_\theta)$.

Approximate minimization found through SGD.

Gradient Variance Amplification (GVA):

Stochasticity in estimates of $\nabla_\theta L_{\mathsf{BC}}(\pi_\theta)$ has a pronounced effect on $J(\pi_\theta)$ but not $L_{\mathsf{BC}}(\pi_\theta)$.

Optimization matters

Parameterize the policy class $\Pi = \{\pi_\theta | \theta \in \Theta \subset \mathbb{R}^d\}$.

Behavior Cloning (BC) [P'89;RB'10]

Return $\hat{\theta} \in \operatorname{argmin}_{\theta \in \Theta} L_{\mathsf{BC}}(\pi_\theta)$.

Approximate minimization found through SGD.

Gradient Variance Amplification (GVA):

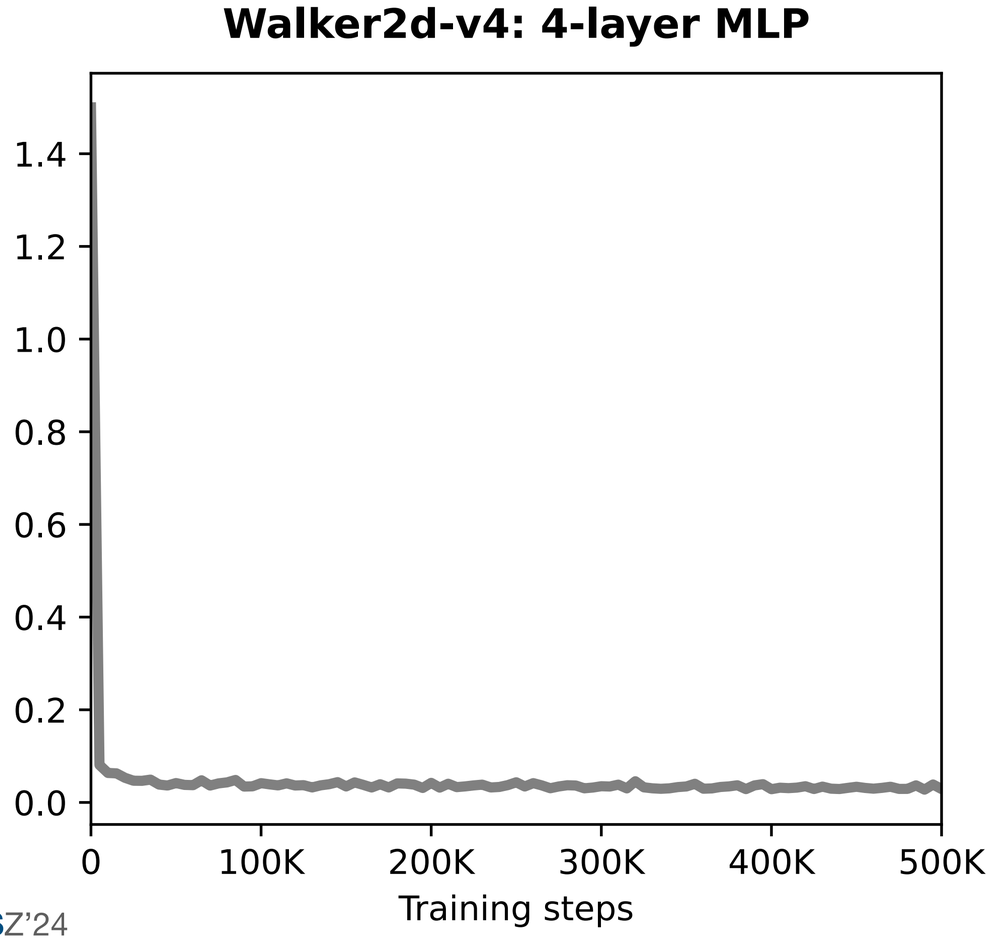
Stochasticity in estimates of $\nabla_\theta L_{\mathsf{BC}}(\pi_\theta)$ has a pronounced effect on $J(\pi_\theta)$ but not $L_{\mathsf{BC}}(\pi_\theta)$.

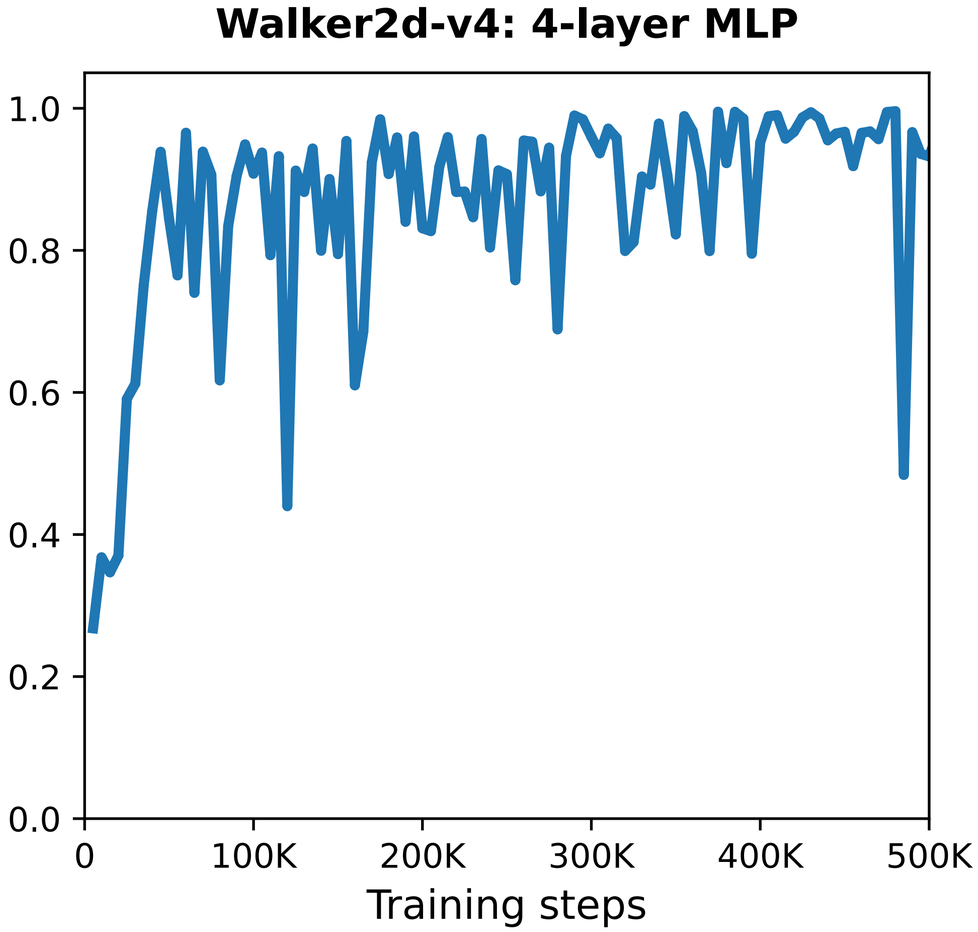
Expected reward of rolled out trajectory

Gradient Variance Amplification

$$L_{\mathrm{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \left\| a_h^{(i)} - \pi(s_h^{(i)}) \right\|^2$$

$$J(\pi) = \mathbb{E}^{\pi} \left[\sum_{h=1}^{H} r(s_h, a_h) \right]$$

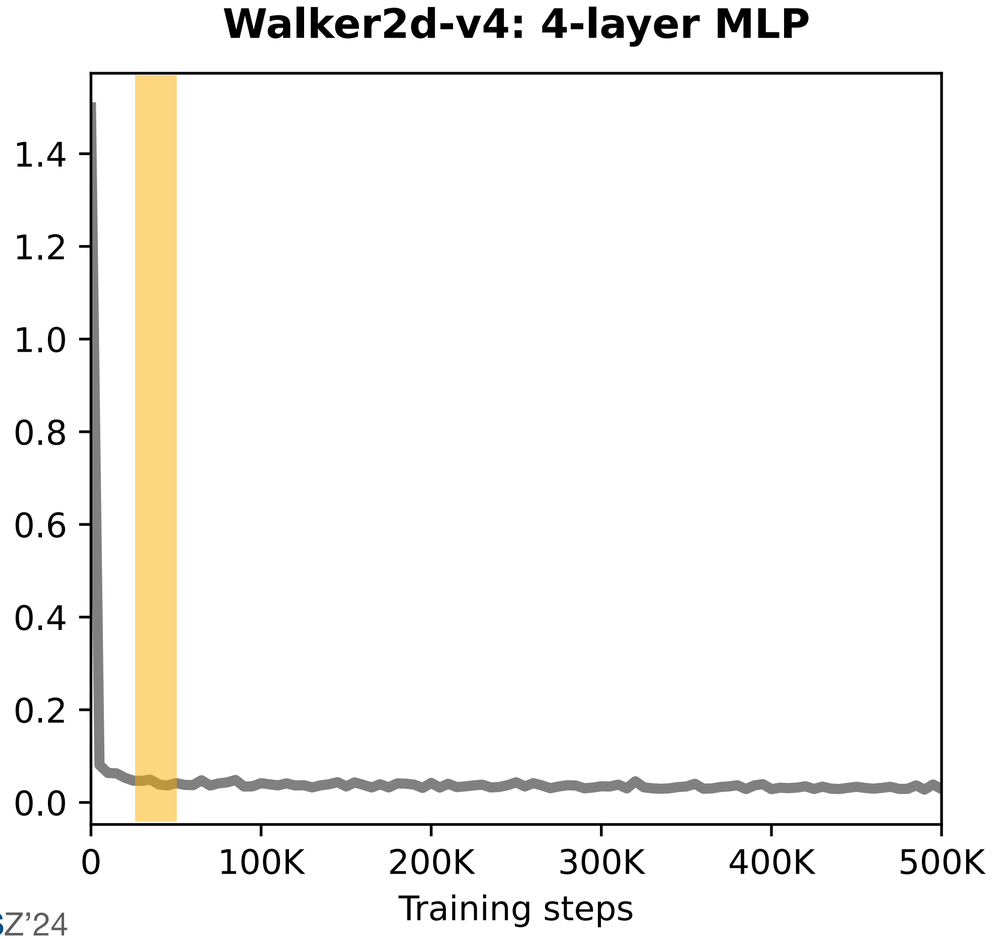


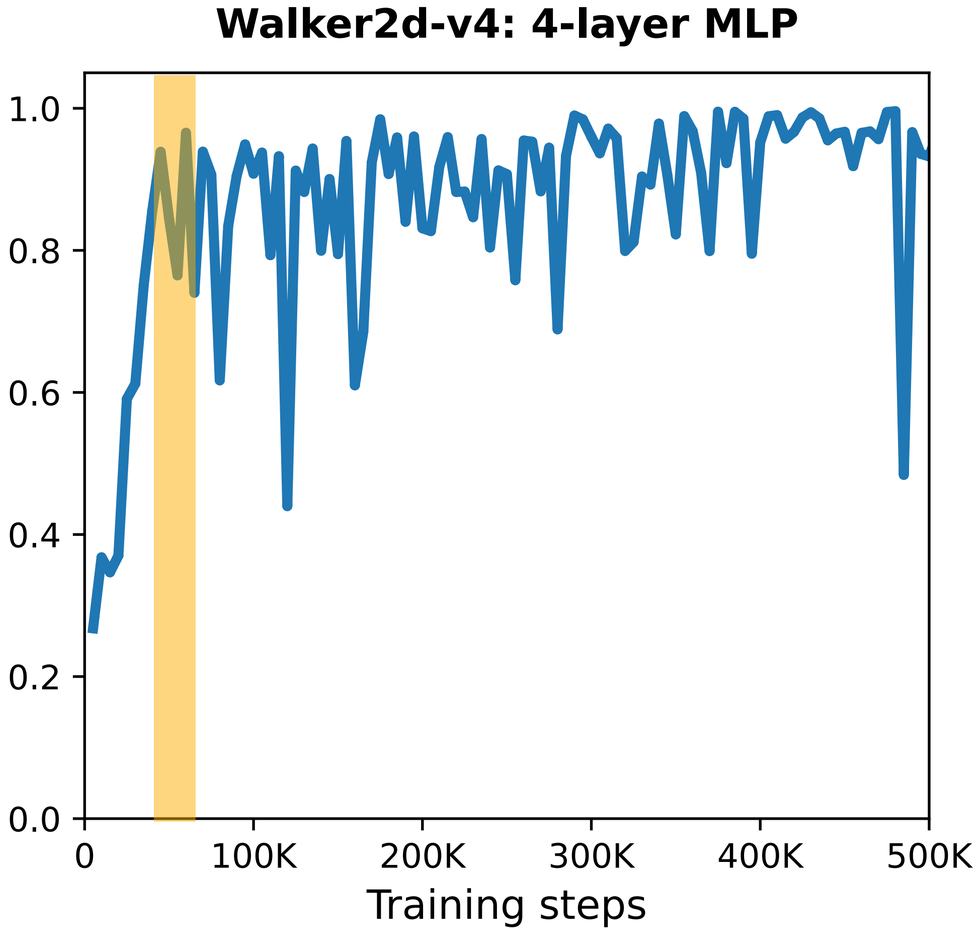


Gradient Variance Amplification

$$L_{\mathrm{BC}}(\pi) = \sum_{i=1}^{n} \sum_{h=1}^{H} \left\| a_h^{(i)} - \pi(s_h^{(i)}) \right\|^2$$

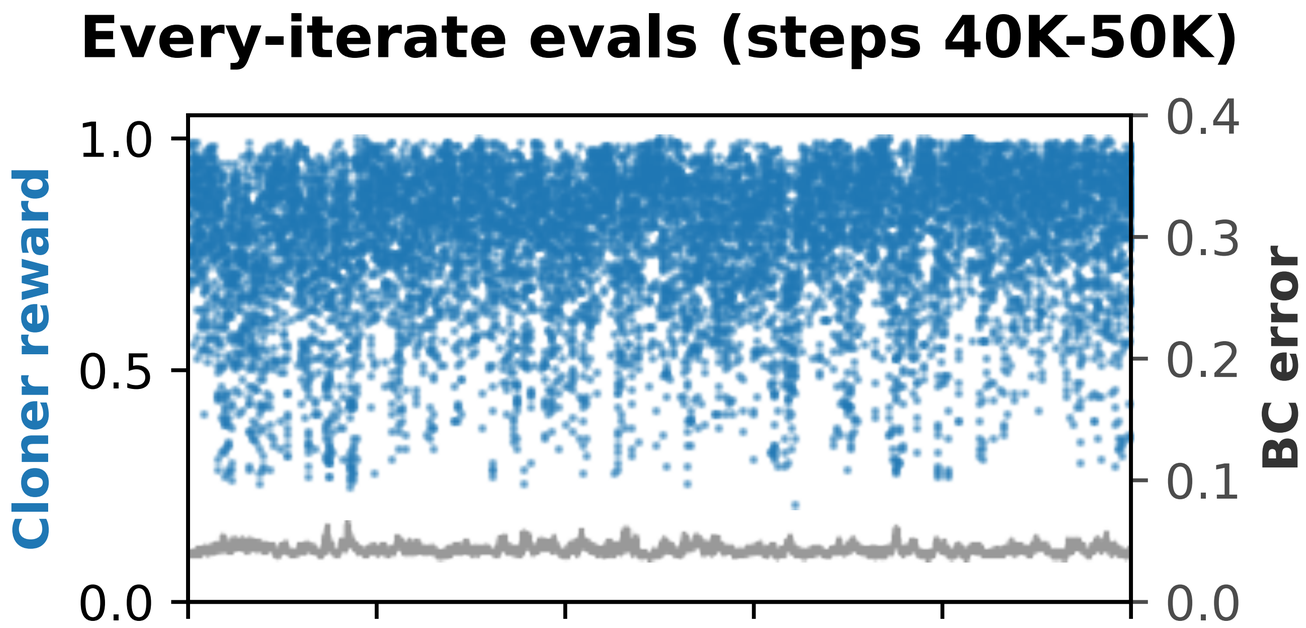
$$J(\pi) = \mathbb{E}^{\pi} \left[\sum_{h=1}^{H} r(s_h, a_h) \right]$$

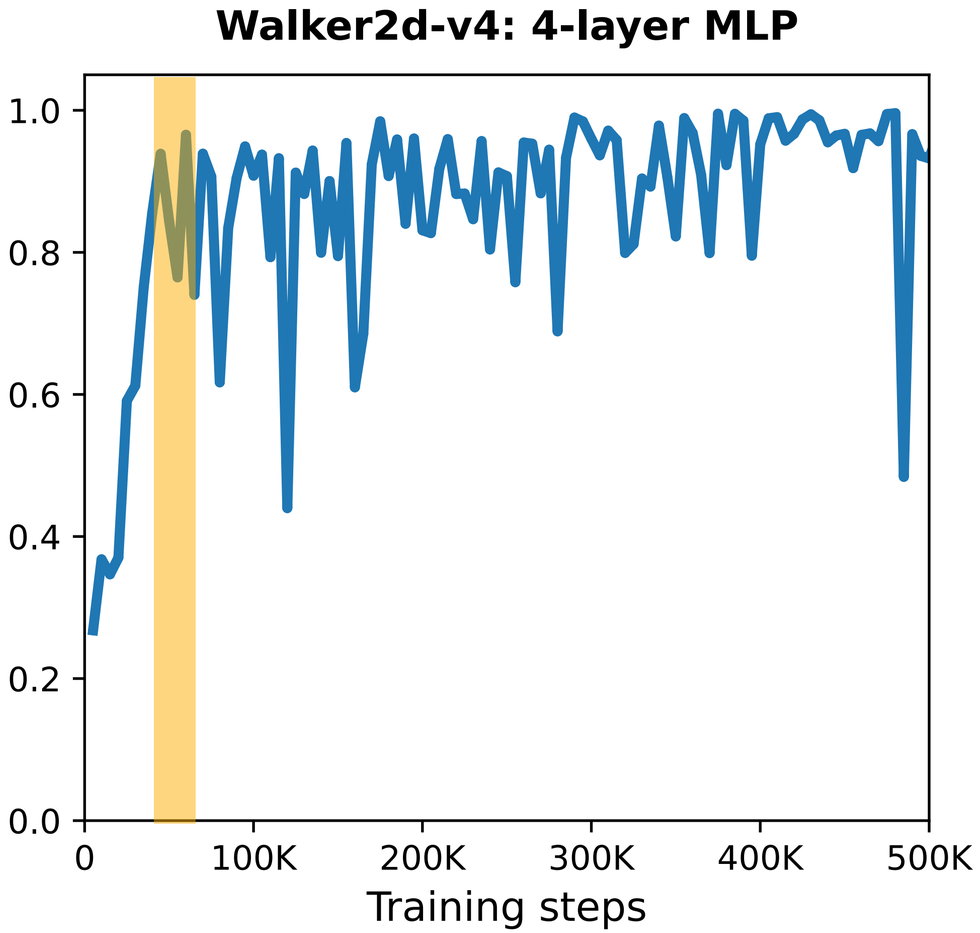




Gradient Variance Amplification

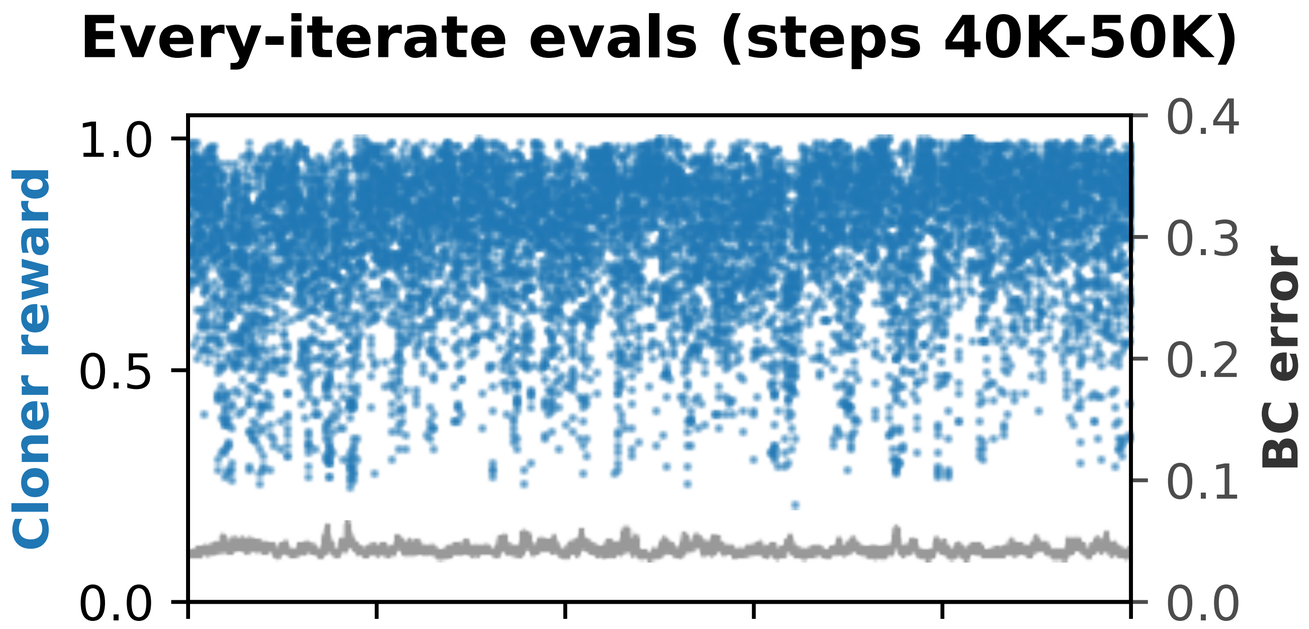
$$J(\pi) = \mathbb{E}^{\pi}\left[\sum_{h=1}^{H} r(s_h, a_h)\right]$$

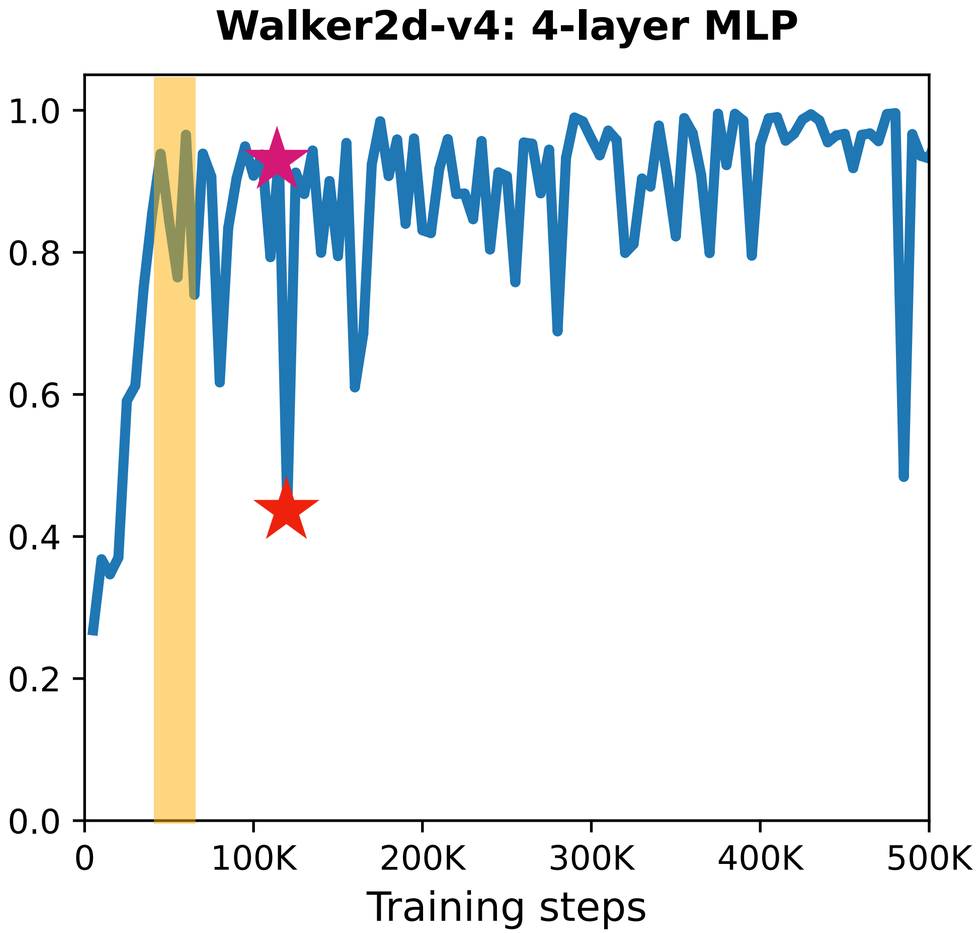




Gradient Variance Amplification

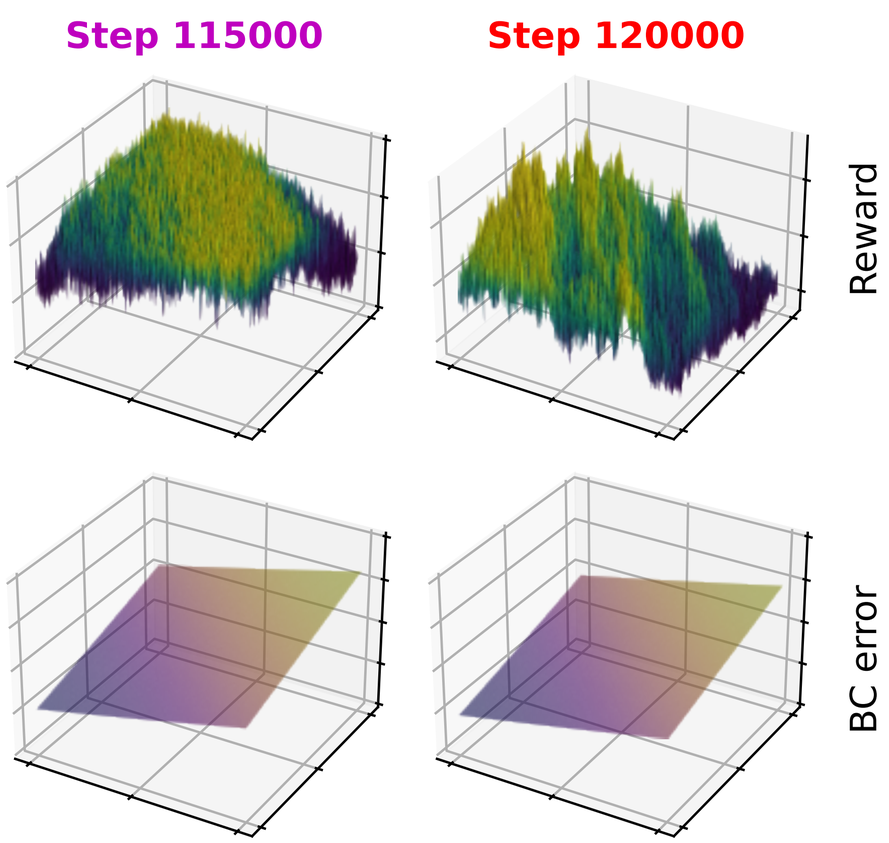
$$J(\pi) = \mathbb{E}^{\pi}\left[\sum_{h=1}^{H} r(s_h, a_h)\right]$$

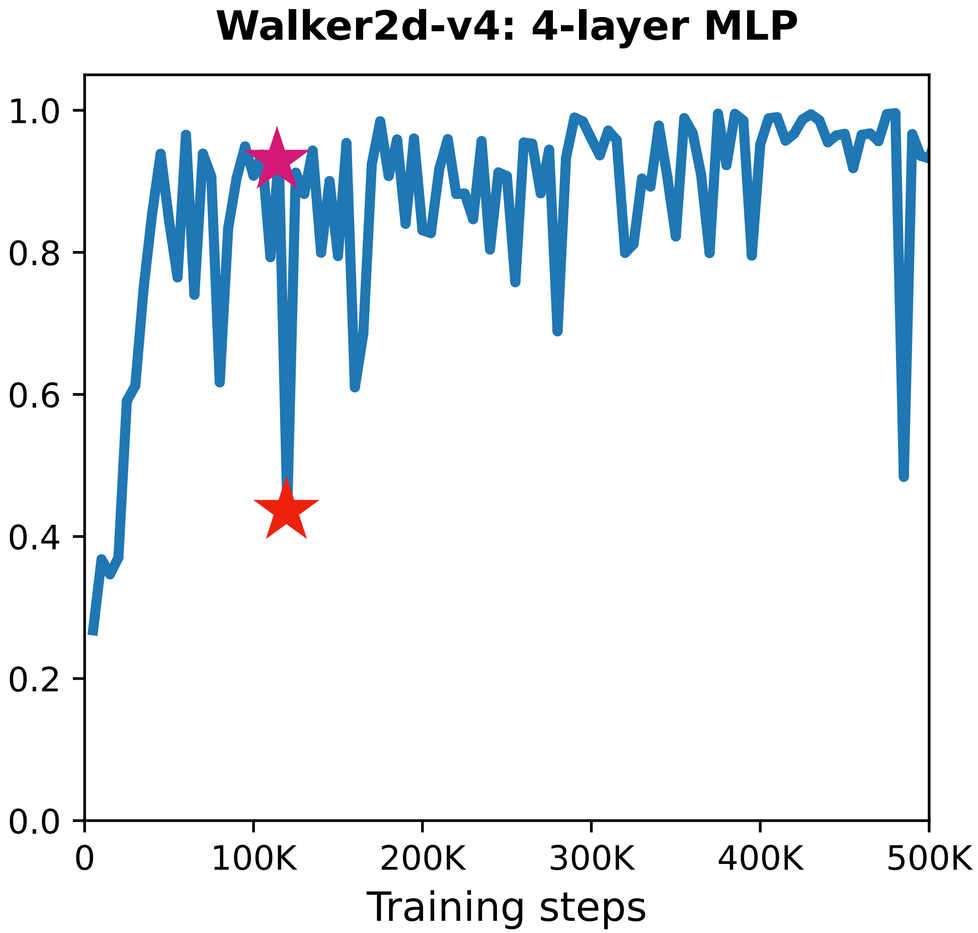




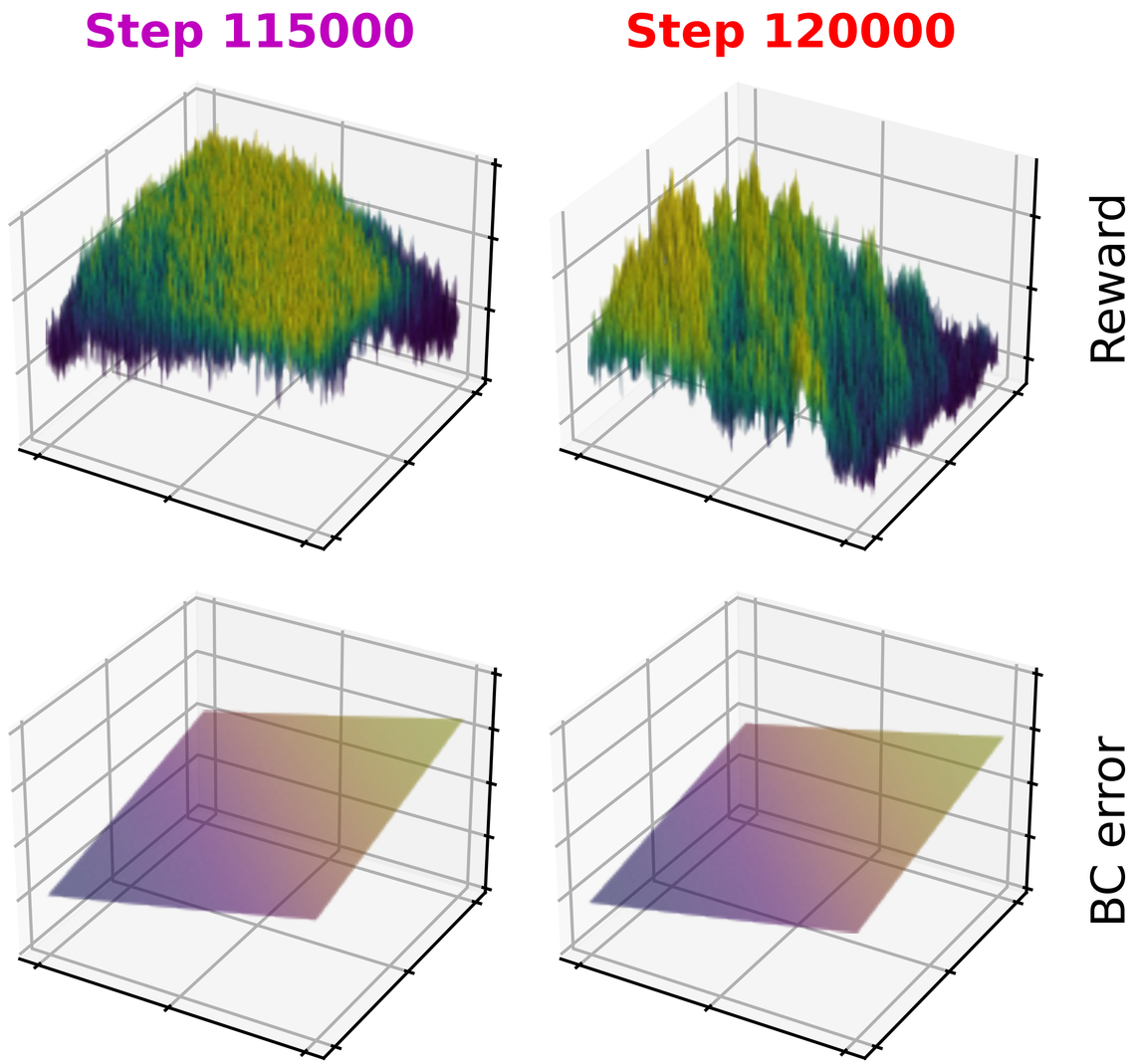
Gradient Variance Amplification

$$J(\pi) = \mathbb{E}^{\pi}\left[\sum_{h=1}^{H} r(s_h, a_h)\right]$$

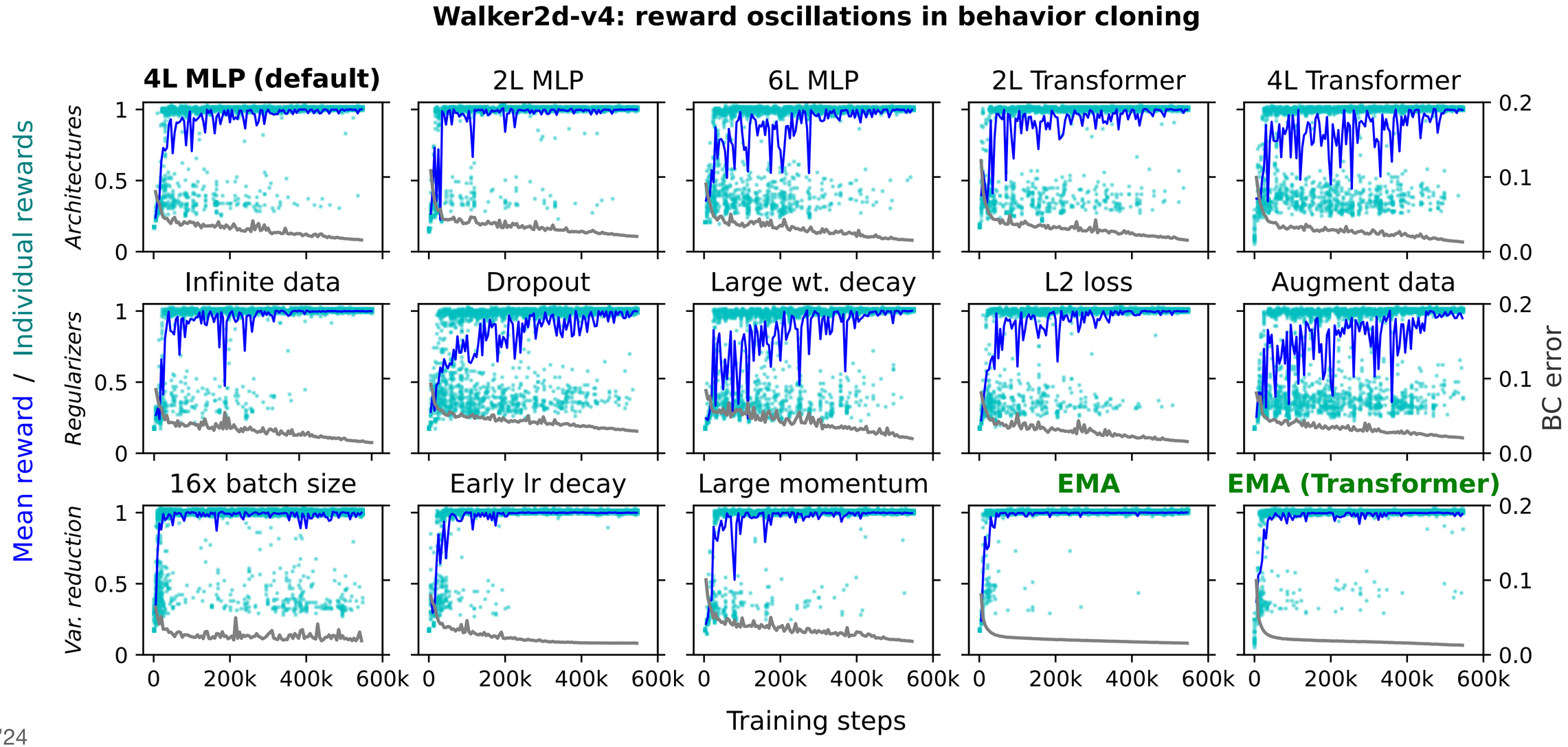




Gradient Variance Amplification



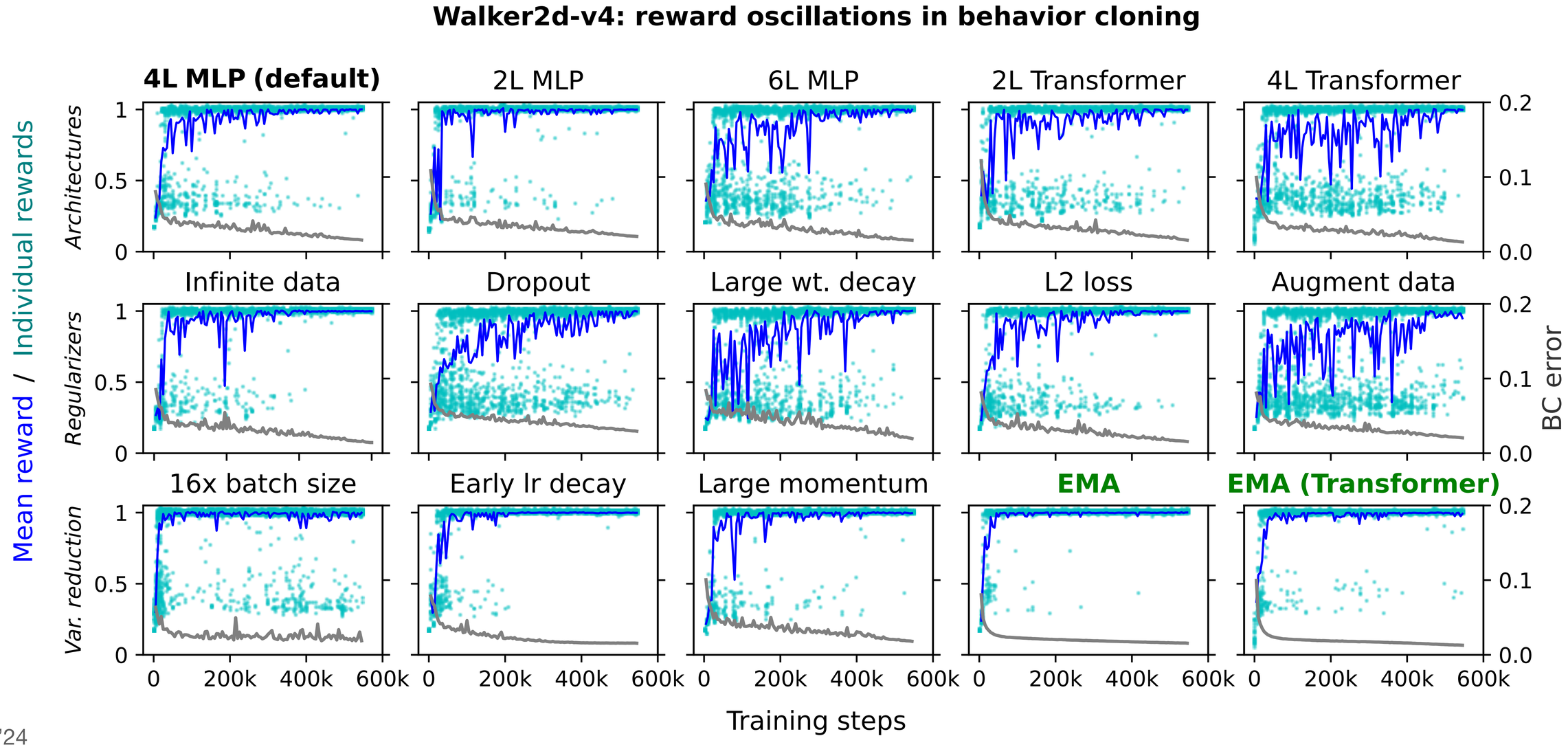
Mitigating GVA



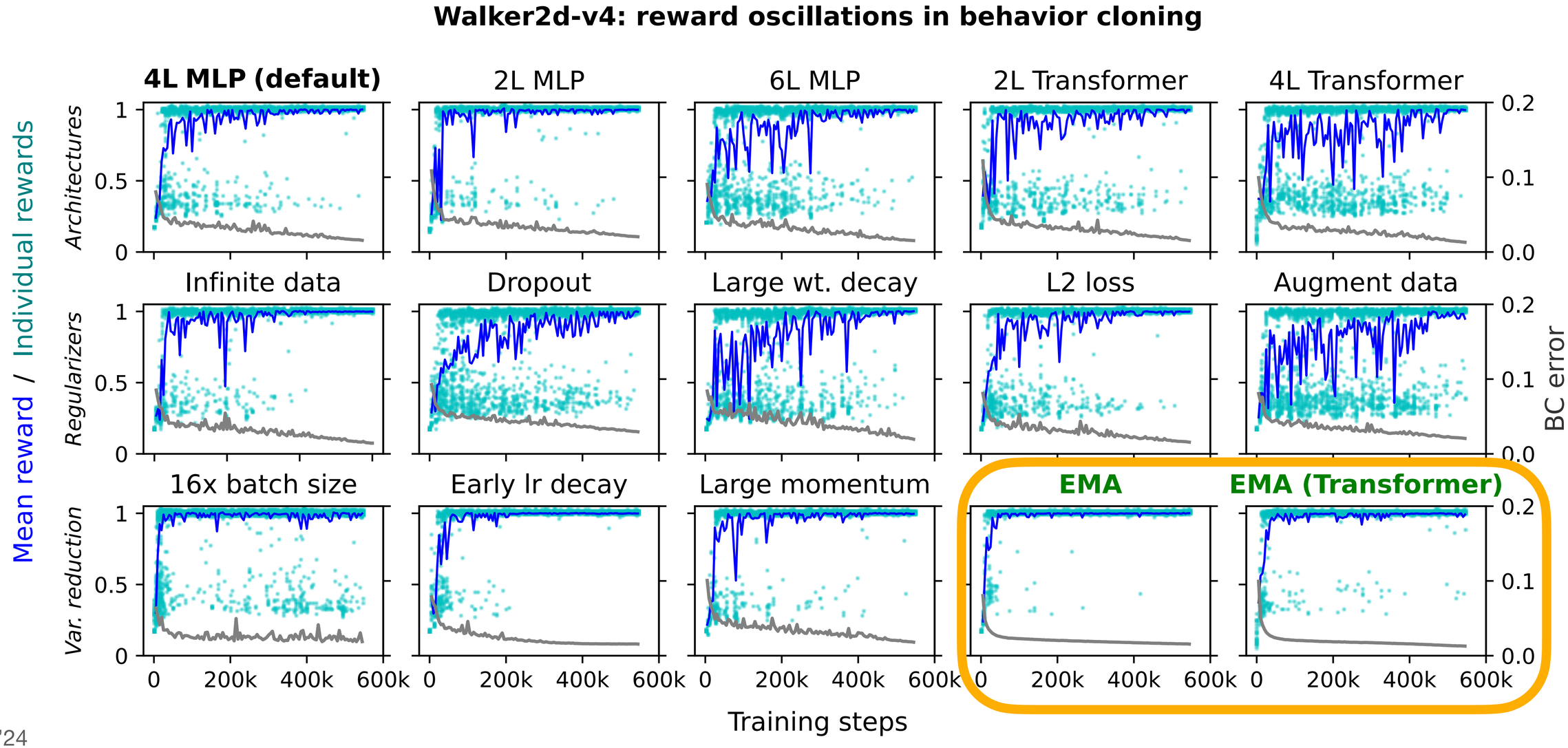
1. What causes these **wild oscillations** across different **problem settings**?

2. How can we **stabilize** training effectively and efficiently?

Mitigating GVA



Mitigating GVA



EFFICIENT ESTIMATORS FROM A SLOWLY CONVERGENT ROBBINS–MONRO PROCESS

by

David Ruppert

ACCELERATION OF STOCHASTIC APPROXIMATION BY AVERAGING*

B. T. POLYAK† AND A. B. JUDITSKY‡

Averaging Weights Leads to Wider Optima and Better Generalization

---

Pavel Izmailov[*1] **Dmitrii Podoprikhin**[*2,3] **Timur Garipov**[*4,5] **Dmitry Vetrov**[2,3] **Andrew Gordon Wilson**[1]
[1]Cornell University, [2]Higher School of Economics, [3]Samsung-HSE Laboratory,
[4]Samsung AI Center in Moscow, [5]Lomonosov Moscow State University

Exponential Moving Average

Given:

Training trajectory $\left(\theta_t\right)_{0 \leq t \leq T}$.

Exponential Moving Average

Given:

Training trajectory $\left(\theta_t\right)_{0 \leq t \leq T}$.

EMA:

$$\theta_{\mathrm{EMA},T} = \sum_{t=1}^{T} \alpha_t \cdot \theta_t$$

Exponential Moving Average

Given:

Training trajectory $\left(\theta_t\right)_{0 \le t \le T}$.

EMA:

$$\theta_{\mathrm{EMA},T} = \sum_{t=1}^{T} \alpha_t \cdot \theta_t$$

Does not affect training trajectory.

Exponential Moving Average

Given:

Training trajectory $\left(\theta_t\right)_{0 \le t \le T}$.

EMA:

$$\theta_{\mathrm{EMA},T} = \sum_{t=1}^{T} \alpha_t \cdot \theta_t$$

Does not affect training trajectory.

Memory-efficient: $\theta_{\mathrm{EMA},t+1} = (1 - \beta_t) \cdot \theta_{\mathrm{EMA},t} + \beta_t \cdot \theta_{t+1}$.

Exponential Moving Average

Given:

Training trajectory $\left(\theta_t\right)_{0 \leq t \leq T}$.

EMA:

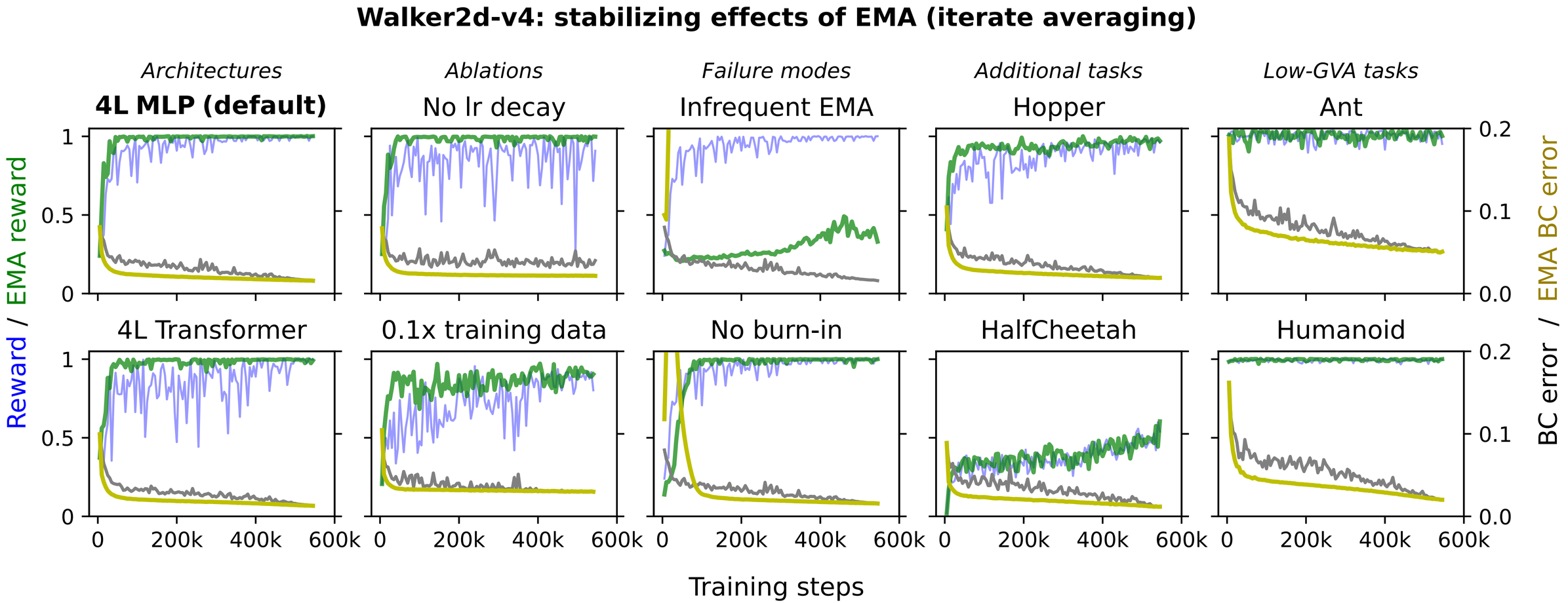
$$\theta_{\mathrm{EMA},T} = \sum_{t=1}^{T} \alpha_t \cdot \theta_t$$

Does not affect training trajectory.

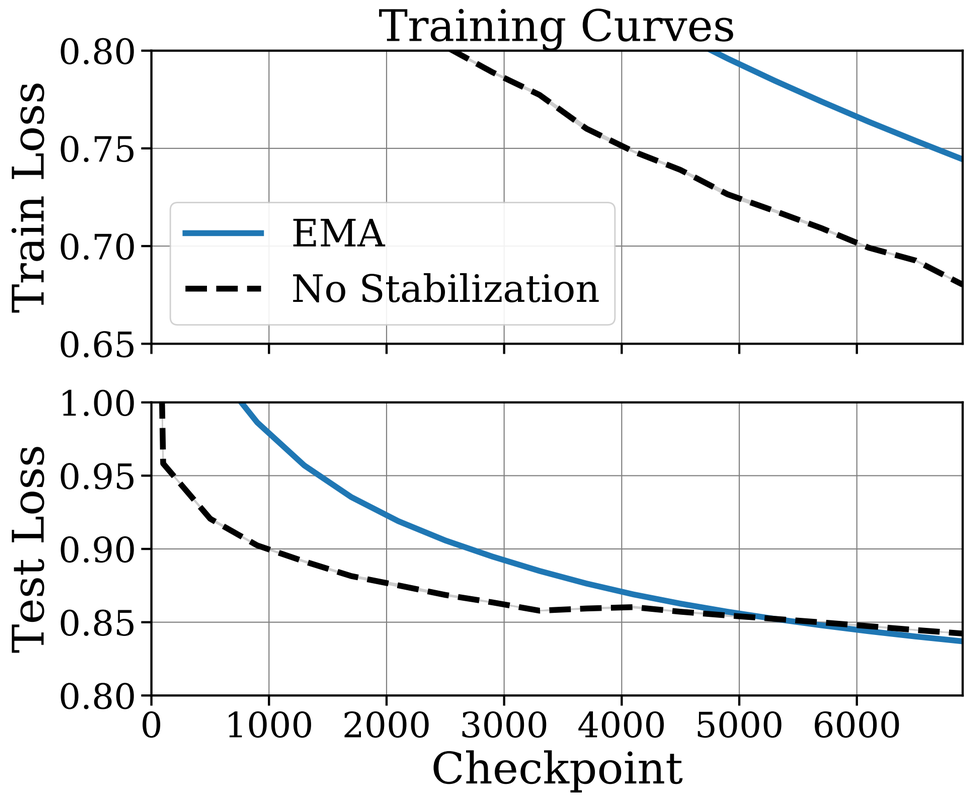
Memory-efficient: $\theta_{\mathrm{EMA},t+1} = (1 - \beta_t) \cdot \theta_{\mathrm{EMA},t} + \beta_t \cdot \theta_{t+1}$.

EMA is ubiquitous throughout RL, IL, diffusion models, deep learning, etc.

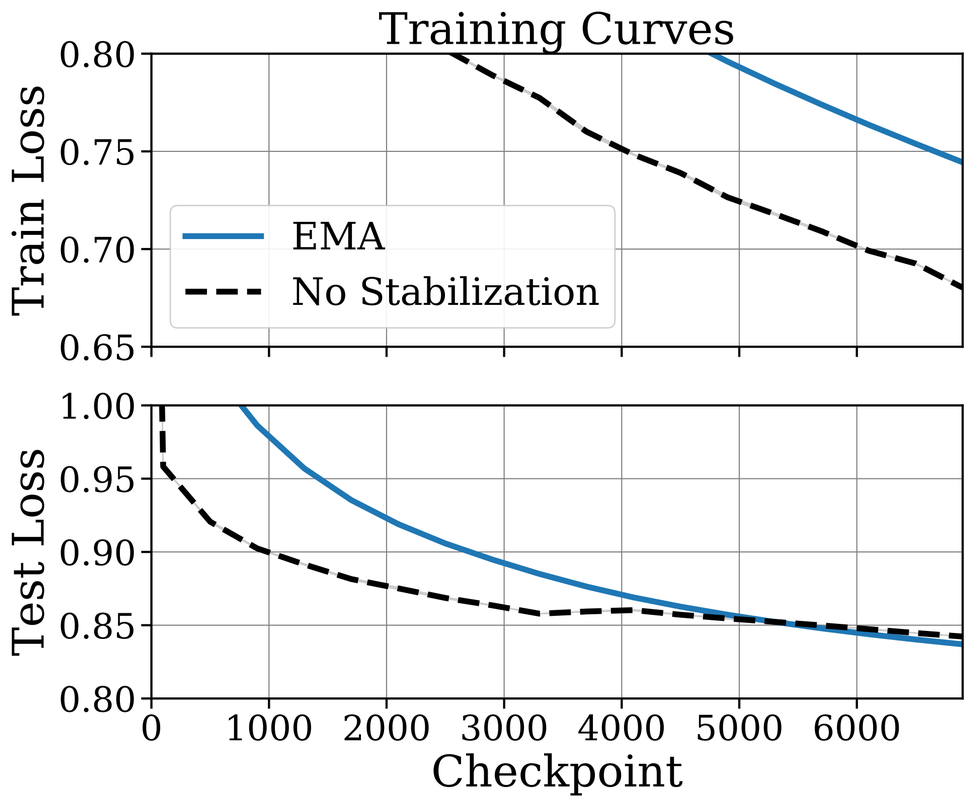
Mitigating GVA with EMA

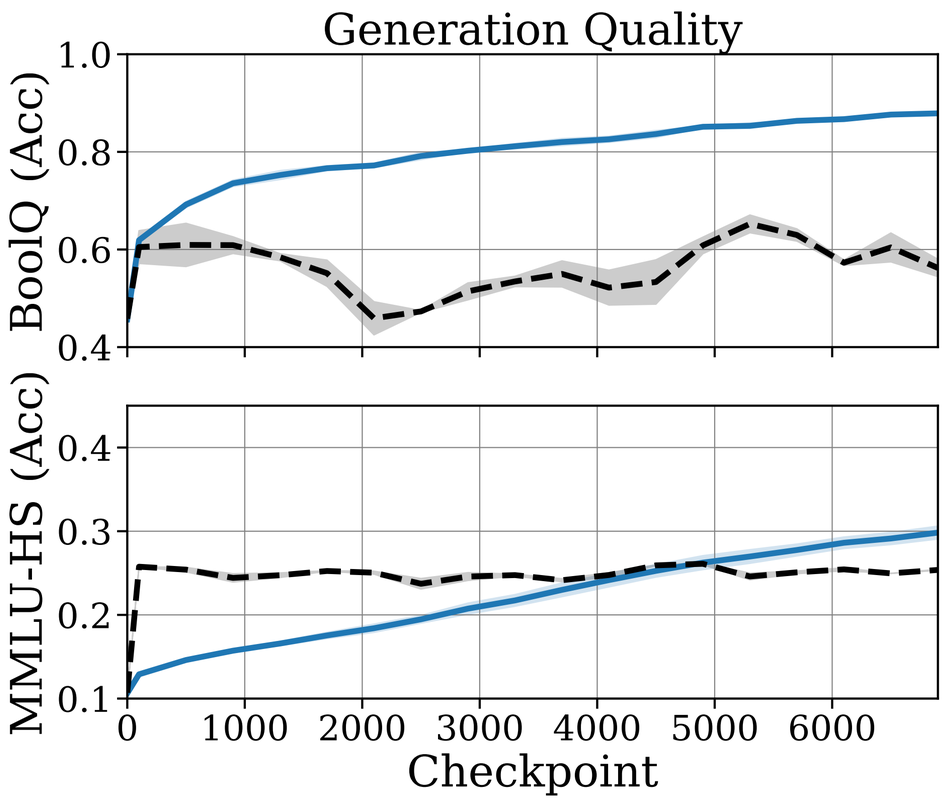


Lag due to EMA

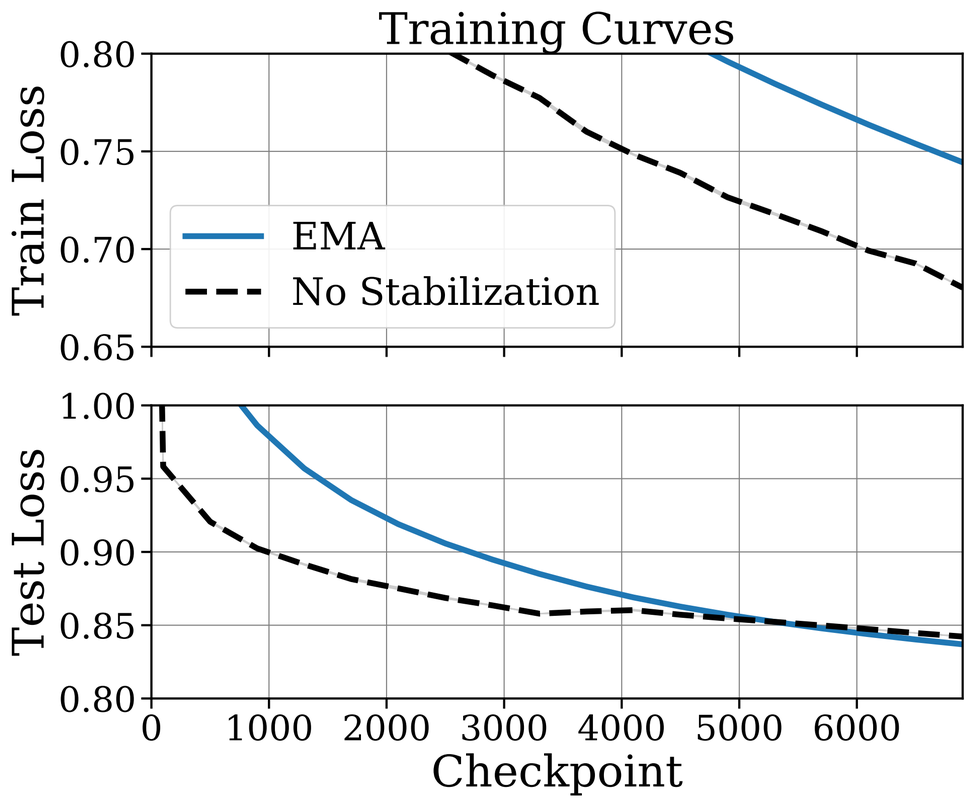


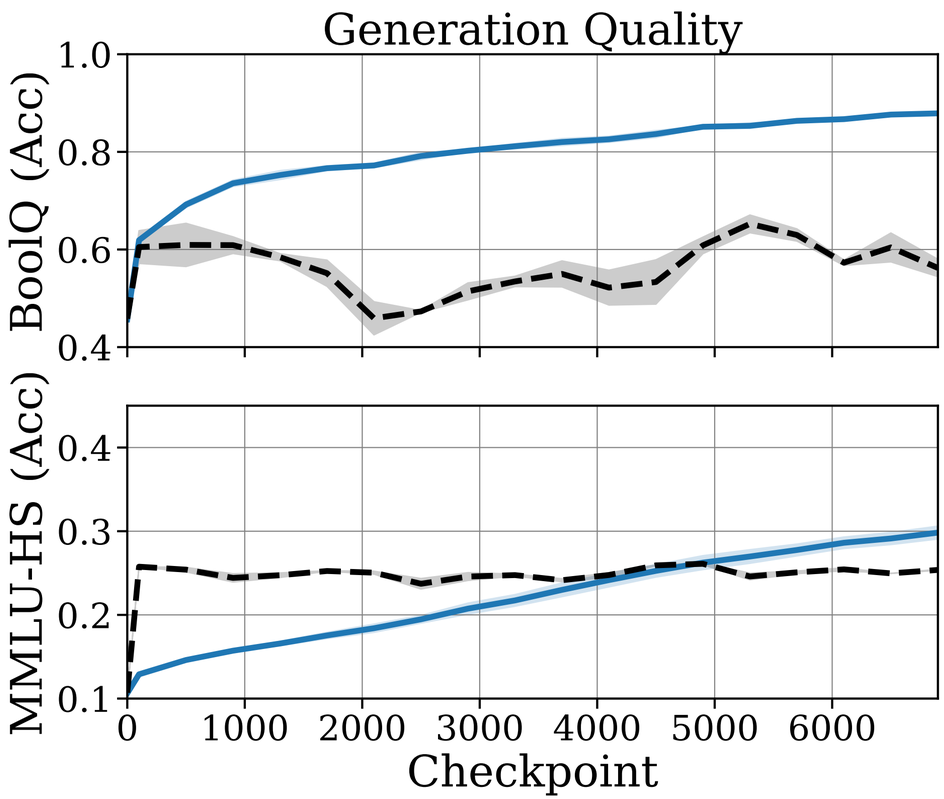
Lag due to EMA





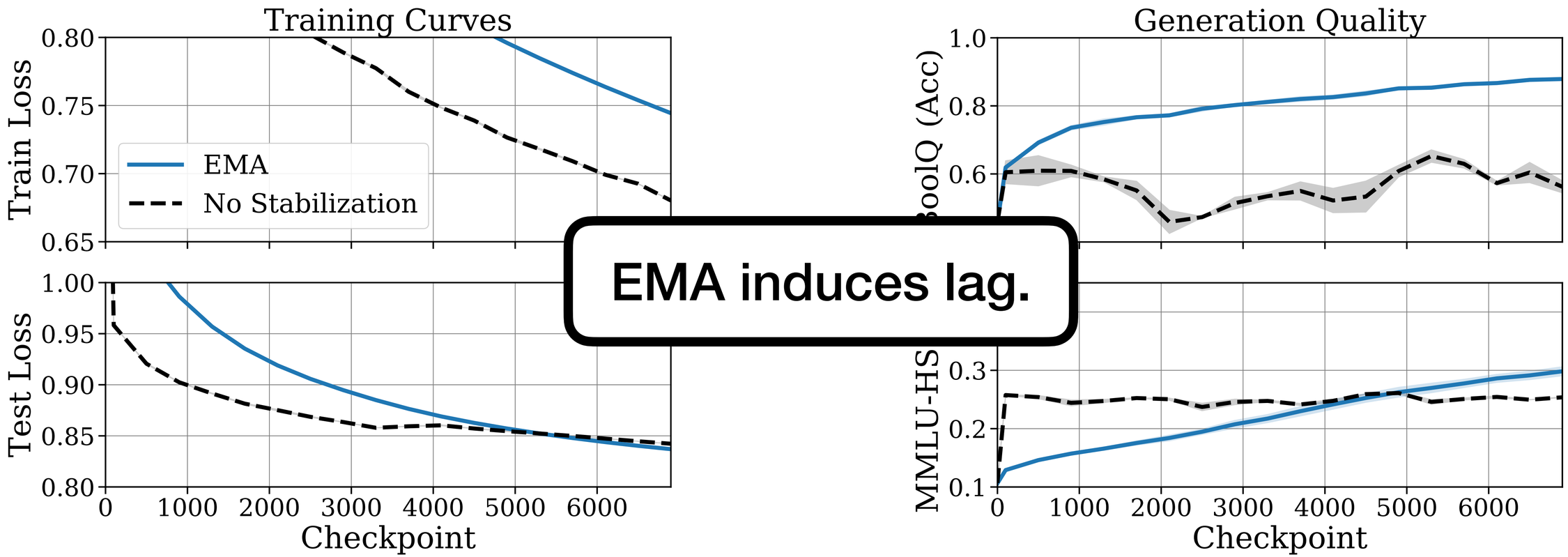
Lag due to EMA





Does EMA suffice?

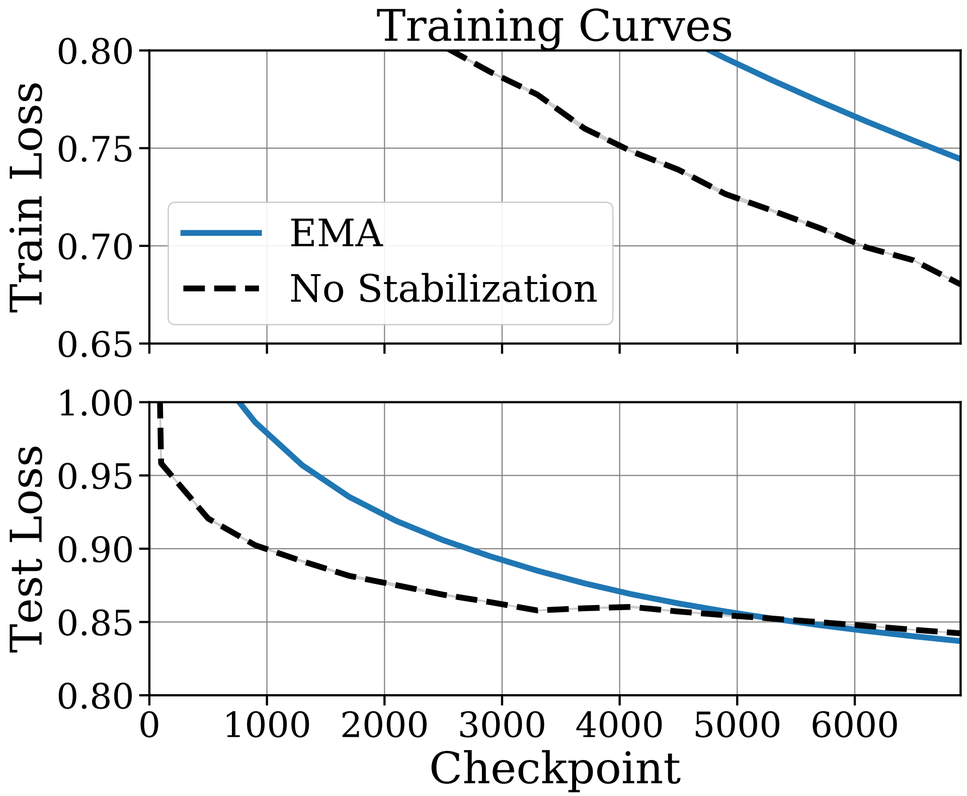
Lag due to EMA



EMA induces lag.

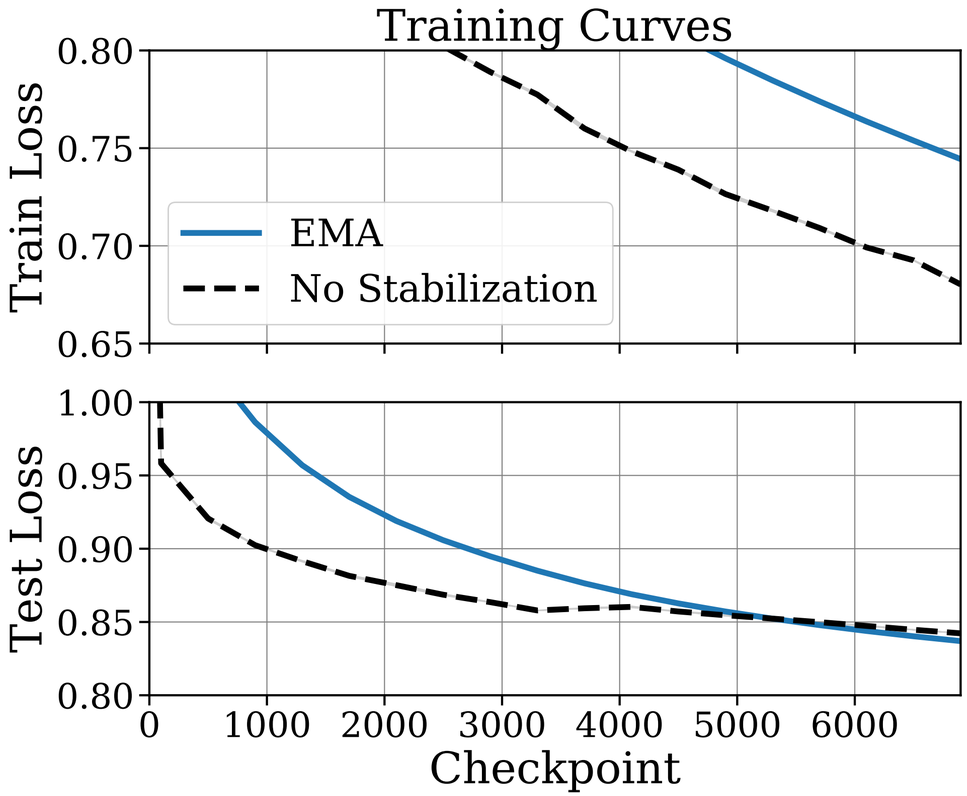
Does EMA suffice?

Bias-corrected EMA (BEMA)



Bias-corrected EMA (BEMA)

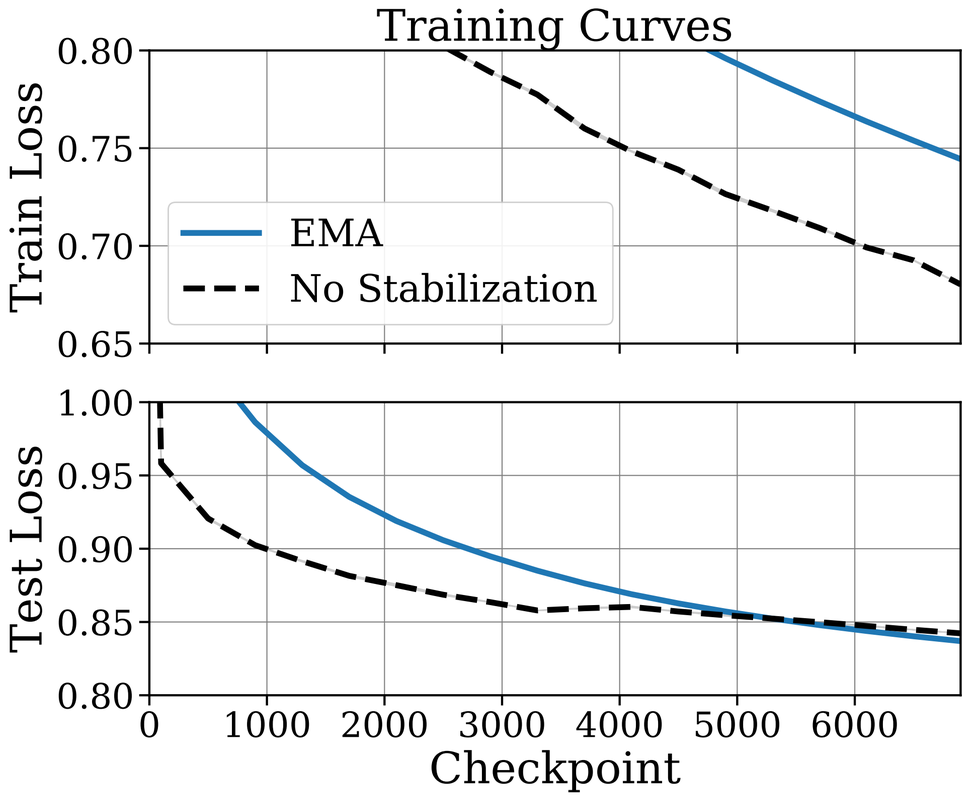
EMA reduces **variance** of stochastic optimization.



Bias-corrected EMA (BEMA)

EMA reduces **variance** of stochastic optimization.

Cost is increase in bias from stale iterates.

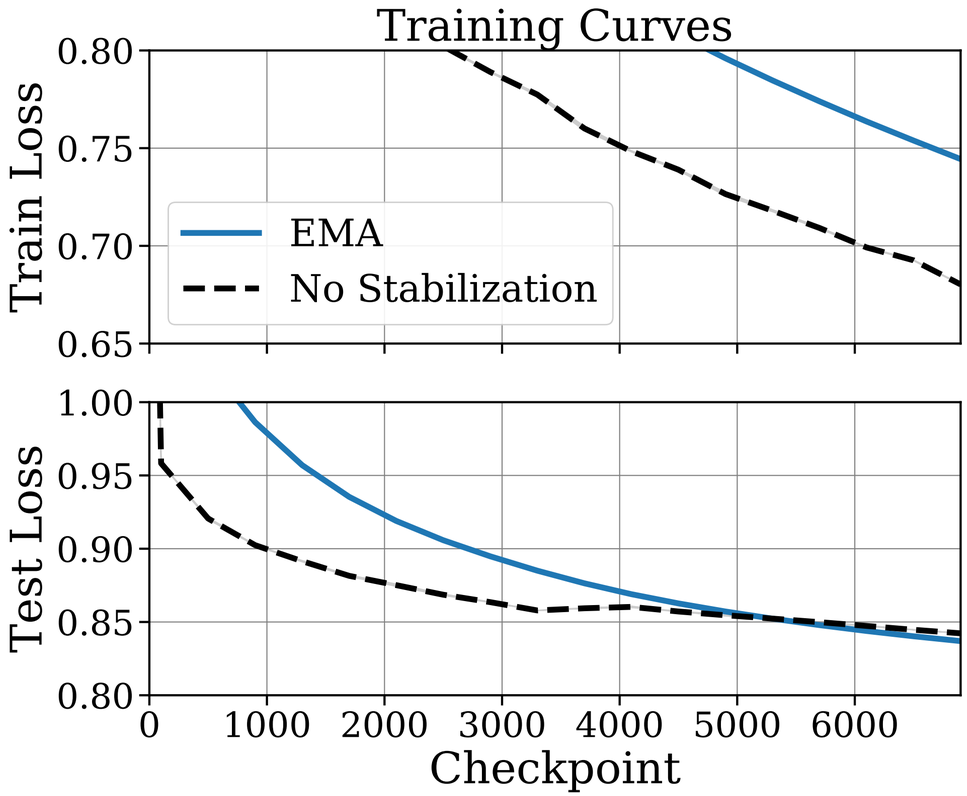


Bias-corrected EMA (BEMA)

EMA reduces **variance** of stochastic optimization.

Cost is increase in **bias** from stale iterates.

Simple, quadratic model reveals this tradeoff.

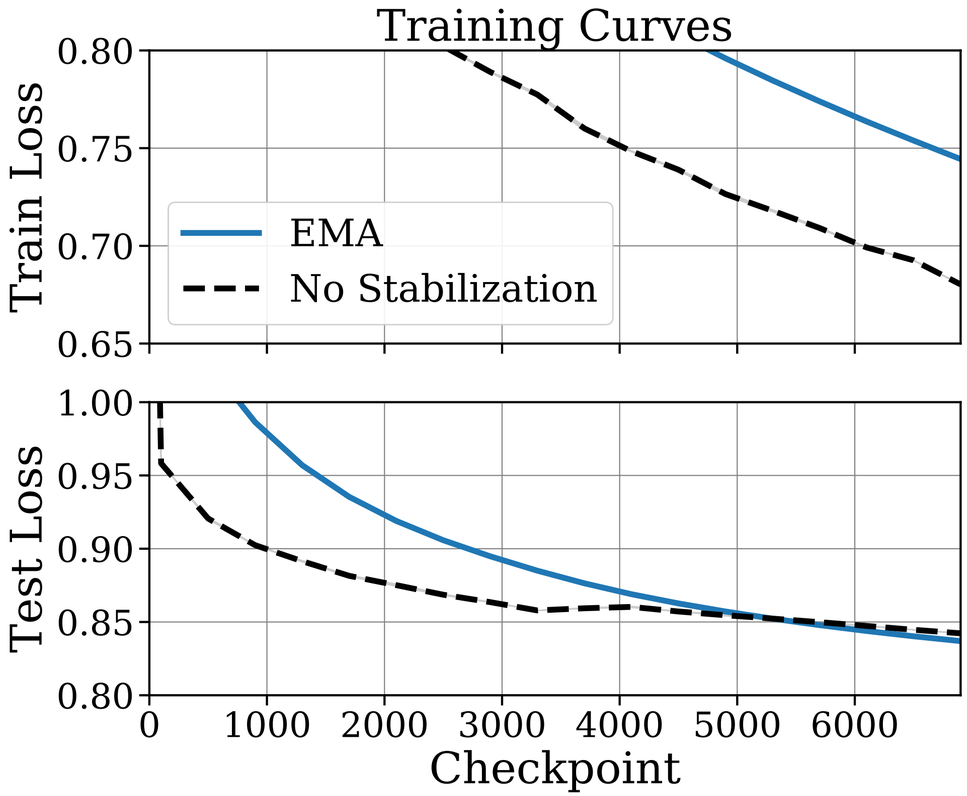


Bias-corrected EMA (BEMA)

EMA reduces **variance** of stochastic optimization.

Cost is increase in **bias** from stale iterates.

Simple, quadratic model reveals this tradeoff.



Bias can be removed in quadratic model!

Formalizing stabilization

Assume $f : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ is **quadratic** with minimum at 0.

Formalizing stabilization

Assume $f : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ is **quadratic** with minimum at 0.

Suppose $d\theta_t = \nabla f(\theta_t - \mu^\star)\ dt + \sqrt{\eta} \cdot \Sigma\ dB_t$

Formalizing stabilization

Assume $f : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ is **quadratic** with minimum at 0.

Suppose $d\theta_t = \nabla f(\theta_t - \mu^\star)\ dt + \sqrt{\eta} \cdot \Sigma\ dB_t$

Minimizer

Formalizing stabilization

Assume $f : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ is **quadratic** with minimum at 0.

Suppose $d\theta_t = \nabla f(\theta_t - \mu^\star)\ dt + \sqrt{\eta} \cdot \Sigma\ dB_t$

Minimizer **Learning rate**

Formalizing stabilization

Assume $f : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ is quadratic with minimum at 0.

Gradient covariance

Suppose $d\theta_t = \nabla f(\theta_t - \mu^\star)\ dt + \sqrt{\eta} \cdot \Sigma\ dB_t$

Minimizer Learning rate

Formalizing stabilization

Assume $f : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ is quadratic with minimum at 0.

Suppose $d\theta_t = \nabla f(\theta_t - \mu^\star)\, dt + \sqrt{\eta} \cdot \Sigma\, dB_t$

Gradient covariance

Brownian motion

Minimizer

Learning rate

Formalizing stabilization

Assume $f : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ is **quadratic** with minimum at 0.

Suppose $d\theta_t = \nabla f(\theta_t - \mu^\star)\, dt + \sqrt{\eta} \cdot \Sigma\, dB_t$

Gradient covariance

Brownian motion

Minimizer

Learning rate

Stabilization

What is the best estimator $\hat{\mu}$ of $\mu^\star$ given $\left(\theta_t\right)_{0 \leq t \leq T}$?

Stabilizing optimization with EMA

Assumption:

$$d\theta_t = \mathbf{A}(\mu^\star - \theta_t)\ dt + \sqrt{\eta} \cdot \sigma\ dB_t$$

Stabilizing optimization with EMA

Assumption:

$$d\theta_t = \mathbf{A}(\mu^\star - \theta_t)\ dt + \sqrt{\eta} \cdot \sigma\ dB_t$$

EMA:

$$\hat{\mu}_T^{\mathrm{EMA}} = \frac{1}{T}\int_0^T \theta_t\ dt.$$

Stabilizing optimization with EMA

Assumption:

$$d\theta_t = \mathbf{A}(\mu^\star - \theta_t)\, dt + \sqrt{\eta} \cdot \sigma\, dB_t$$

EMA:

$$\hat{\mu}_T^{\mathrm{EMA}} = \frac{1}{T}\int_0^T \theta_t\, dt.$$

Theorem [BZ'25]**:** It holds that

$$\mathbb{E}\left[\left\| \hat{\mu}_T^{\mathrm{EMA}} - \mu^\star \right\|^2 \right] \le \frac{\eta\sigma^2 \cdot \mathrm{Tr}\left(\mathbf{A}^{-2}\right)}{T} + \frac{\left\| \mathbf{A}^{-1} \right\|_{\mathrm{op}}^2 \cdot \|\mu^\star - \theta_0\|^2}{T^2}.$$

Stabilizing optimization with EMA

Assumption:

$$d\theta_t = \mathbf{A}(\mu^\star - \theta_t)\ dt + \sqrt{\eta} \cdot \sigma\ dB_t$$

EMA:

$$\hat{\mu}_T^{\mathrm{EMA}} = \frac{1}{T}\int_0^T \theta_t\ dt.$$

Theorem [BZ'25]**:** It holds that

$$\mathbb{E}\left[\ \left\|\ \hat{\mu}_T^{\mathrm{EMA}} - \mu^\star\ \right\|^2\right] \le \frac{\eta\sigma^2 \cdot \mathrm{Tr}\left(\mathbf{A}^{-2}\right)}{T} + \frac{\left\|\ \mathbf{A}^{-1}\ \right\|_{\mathrm{op}}^2 \cdot \|\mu^\star - \theta_0\|^2}{T^2}.$$

Stabilizing optimization with EMA

Assumption:

$$d\theta_t = \mathbf{A}(\mu^\star - \theta_t)\, dt + \sqrt{\eta} \cdot \sigma\, dB_t$$

EMA:

$$\hat{\mu}_T^{\mathrm{EMA}} = \frac{1}{T}\int_0^T \theta_t\, dt.$$

Theorem [BZ'25]**:** It holds that

$$\mathbb{E}\left[\left\| \hat{\mu}_T^{\mathrm{EMA}} - \mu^\star \right\|^2\right] \le \frac{\eta\sigma^2 \cdot \mathrm{Tr}\left(\mathbf{A}^{-2}\right)}{T} + \frac{\left\| \mathbf{A}^{-1} \right\|_{\mathrm{op}}^2 \cdot \|\mu^\star - \theta_0\|^2}{T^2}.$$

Bias-Corrected EMA (BEMA)

Proposition [BZ'25]**:** The MLE of $\mu^\star$ given $(\theta_t)_{0\le t\le T}$ is

$$\hat{\mu}_T^{\mathrm{BEMA}} = \frac{\mathbf{A}^{-1}}{T}(\theta_T - \theta_0) + \frac{1}{T}\int_0^T \theta_t \, dt.$$

Bias-Corrected EMA (BEMA)

Proposition [BZ'25]**:** The MLE of $\mu^\star$ given $(\theta_t)_{0\le t\le T}$ is

$$\hat{\mu}_T^{\mathrm{BEMA}} = \frac{\mathbf{A}^{-1}}{T}(\theta_T - \theta_0) + \frac{1}{T}\int_0^T \theta_t \, dt.$$

Theorem [BZ'25]**:** It holds that

$$\mathbb{E}\left[\left\| \hat{\mu}_T^{\mathrm{BEMA}} - \mu^\star \right\|^2 \right] \le \frac{\eta\sigma^2 \cdot \mathrm{Tr}\left(\mathbf{A}^{-2}\right)}{T}.$$

Making BEMA practical

Algorithm 1: Bias corrected Exponential Moving Average (BEMA)

Input: Trajectory $\{\theta_t | t \in [T]\}$, EMA power κ, bias power η, multiplier γ, lag ρ, burn in time τ, frequency ϕ.

Set $\widehat{\mu} \leftarrow \theta_0$ and $\widehat{\mu}^{\mathsf{EMA}} \leftarrow \theta_0$.

for $t = 1, \ldots, T$ **do**

  if $t \leq \tau$ **then**

    Update $\widehat{\mu} \leftarrow \theta_t$, $\theta_0 \leftarrow \theta_t$, and $\widehat{\mu}^{\mathsf{EMA}} \leftarrow \theta_t$. (%% No bias correction before τ)

  else if $(t - \tau) \mod \phi \neq 0$ **then**

    Continue (%% Only update every ϕ steps)

  else

    Define $\alpha_t \leftarrow (\rho + \gamma t)^{-\eta}$ and $\beta_t \leftarrow (\rho + \gamma t)^{-\kappa}$ (%% Define weights for EMA and bias correction.)

    Update $\widehat{\mu}^{\mathsf{EMA}} \leftarrow (1 - \beta_t) \cdot \widehat{\mu}^{\mathsf{EMA}} + \beta_t \cdot \theta_t$ (%% Update EMA.)

    Update $\widehat{\mu} \leftarrow \alpha_t (\theta_t - \theta_0) + \widehat{\mu}^{\mathsf{EMA}}$

Return $\widehat{\mu}$.

Making BEMA practical

Algorithm 1: Bias corrected Exponential Moving Average (BEMA)

Input: Trajectory $\{\theta_t | t \in [T]\}$, EMA power κ, bias power η, multiplier γ, lag ρ, burn in time τ, frequency ϕ.

Set $\widehat{\mu} \leftarrow \theta_0$ and $\widehat{\mu}^{\mathsf{EMA}} \leftarrow \theta_0$.

for $t = 1, \ldots, T$ **do**

  if $t \leq \tau$ **then**

    Update $\widehat{\mu} \leftarrow \theta_t$, $\theta_0 \leftarrow \theta_t$, and $\widehat{\mu}^{\mathsf{EMA}} \leftarrow \theta_t$. (%% No bias correction before τ)

  else if $(t - \tau) \mod \phi \neq 0$ **then**

    Continue (%% Only update every ϕ steps)

  else

    Define $\alpha_t \leftarrow (\rho + \gamma t)^{-\eta}$ and $\beta_t \leftarrow (\rho + \gamma t)^{-\kappa}$ (%% Define weights for EMA and bias correction.)

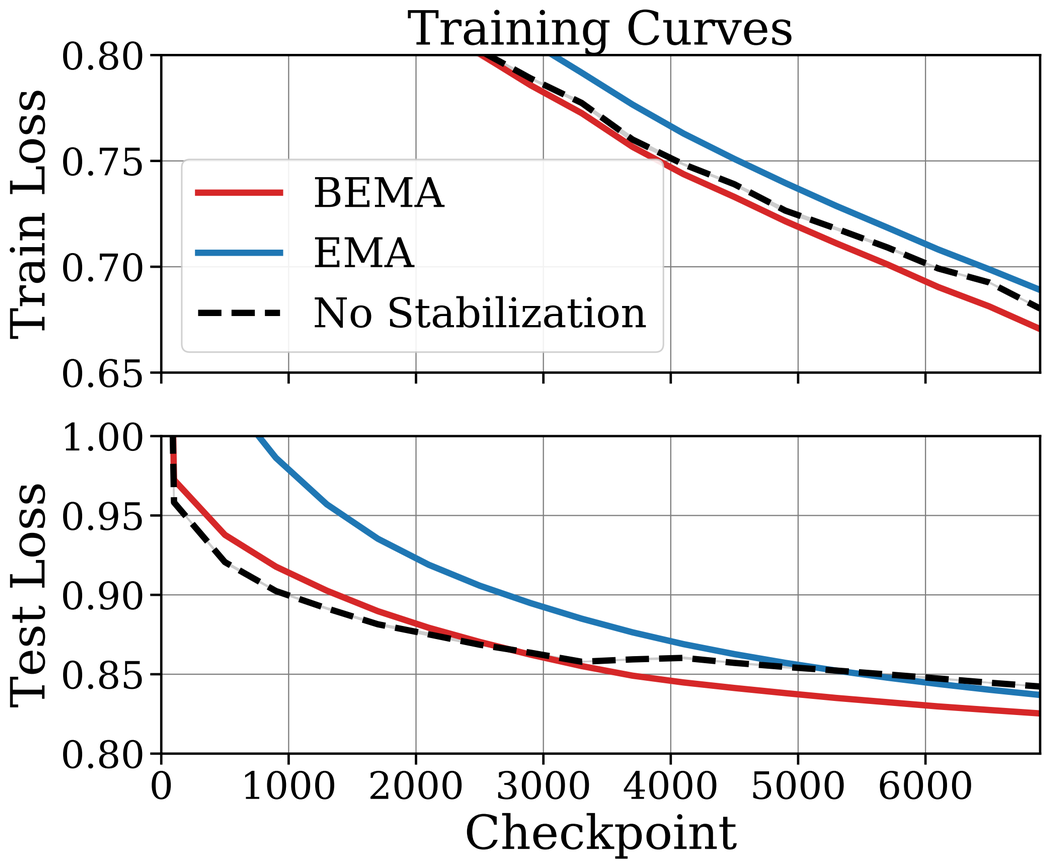
    Update $\widehat{\mu}^{\mathsf{EMA}} \leftarrow (1 - \beta_t) \cdot \widehat{\mu}^{\mathsf{EMA}} + \beta_t \cdot \theta_t$ (%% Update EMA.)

    Update $\widehat{\mu} \leftarrow \alpha_t (\theta_t - \theta_0) + \widehat{\mu}^{\mathsf{EMA}}$

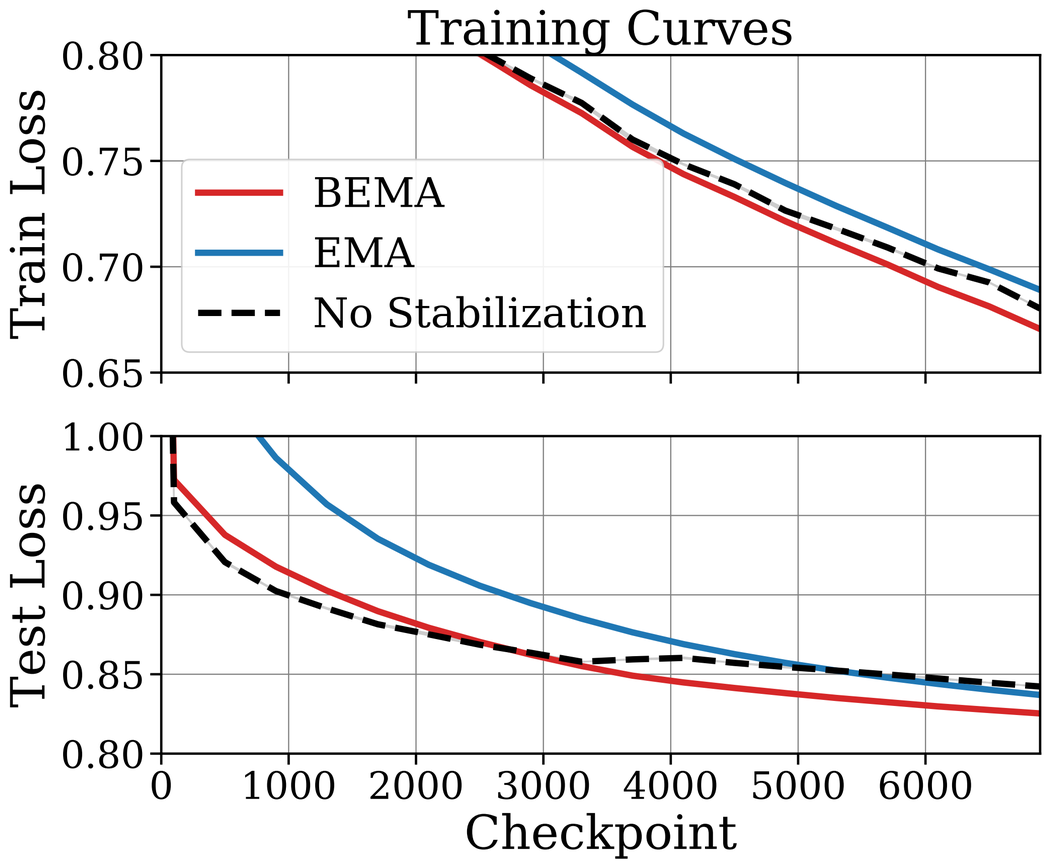
Return $\widehat{\mu}$.

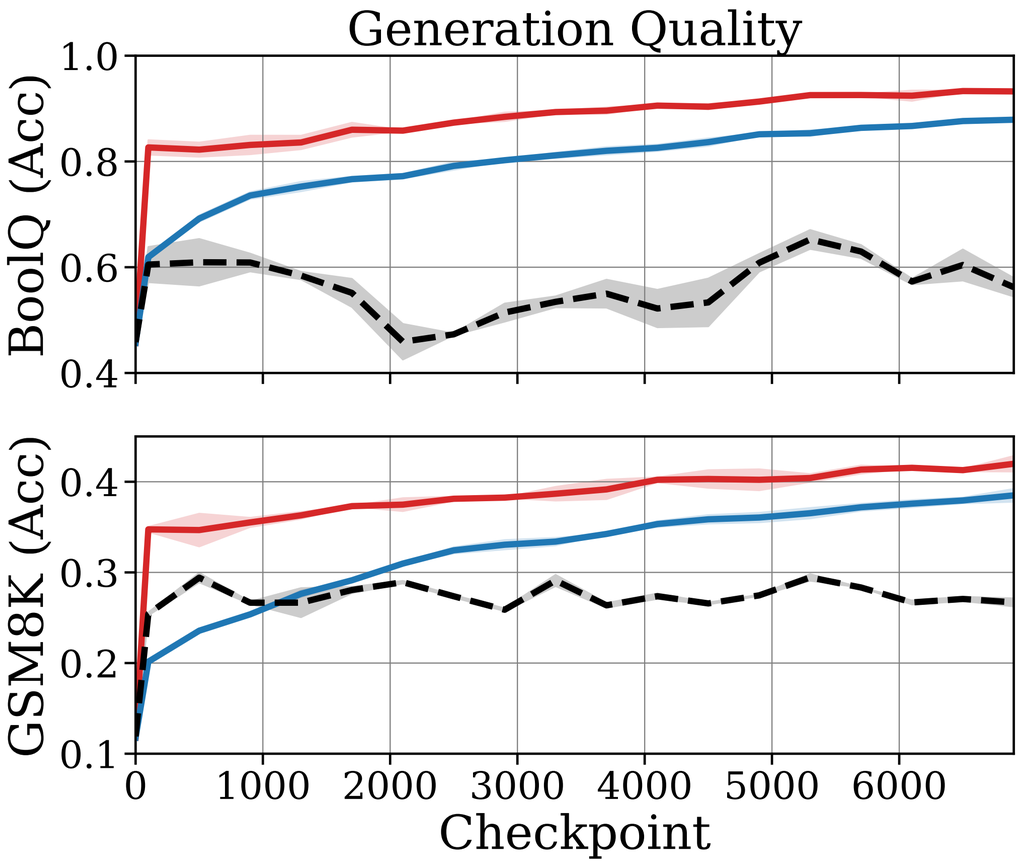
Does it work?

Does BEMA work?

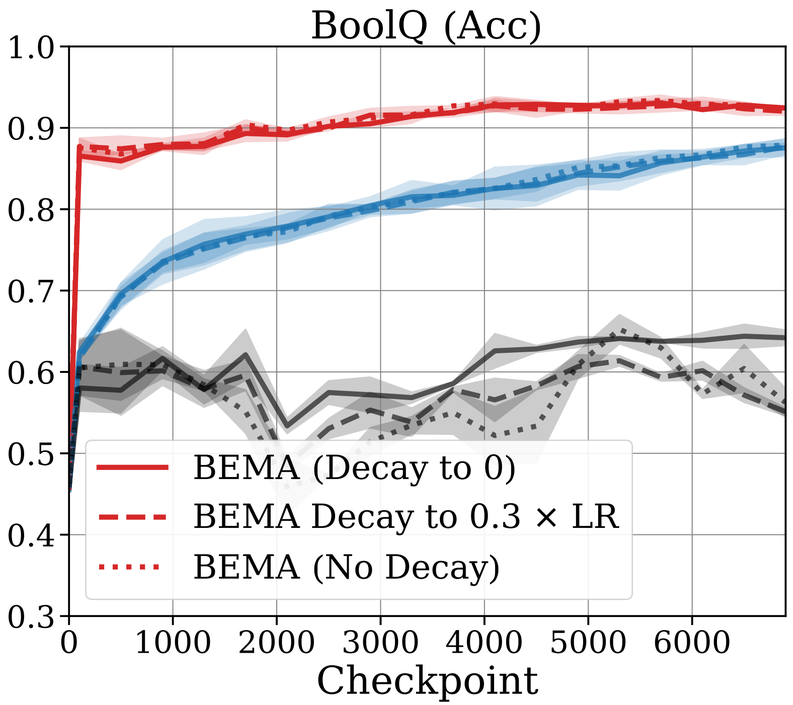


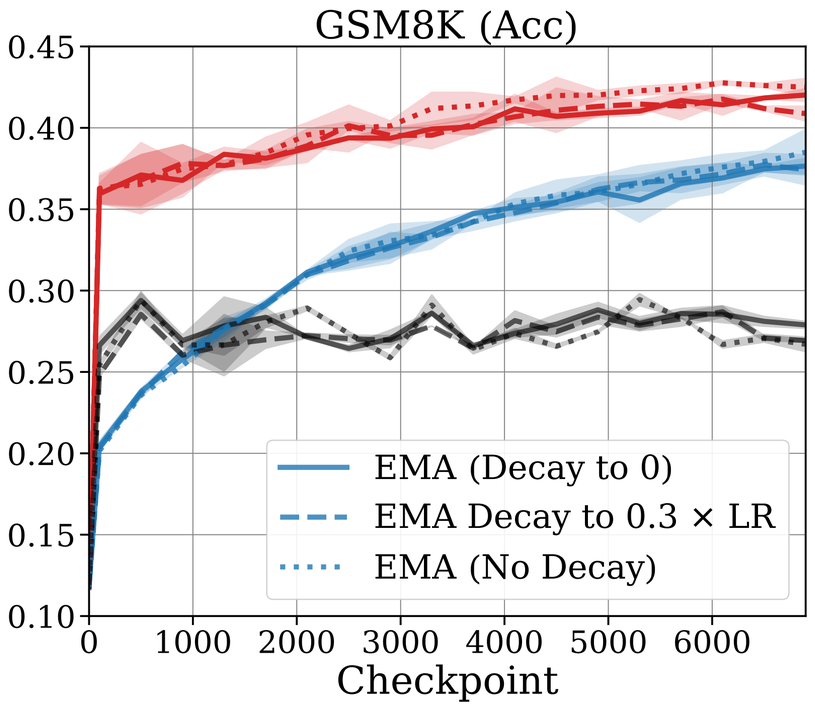
Does BEMA work?

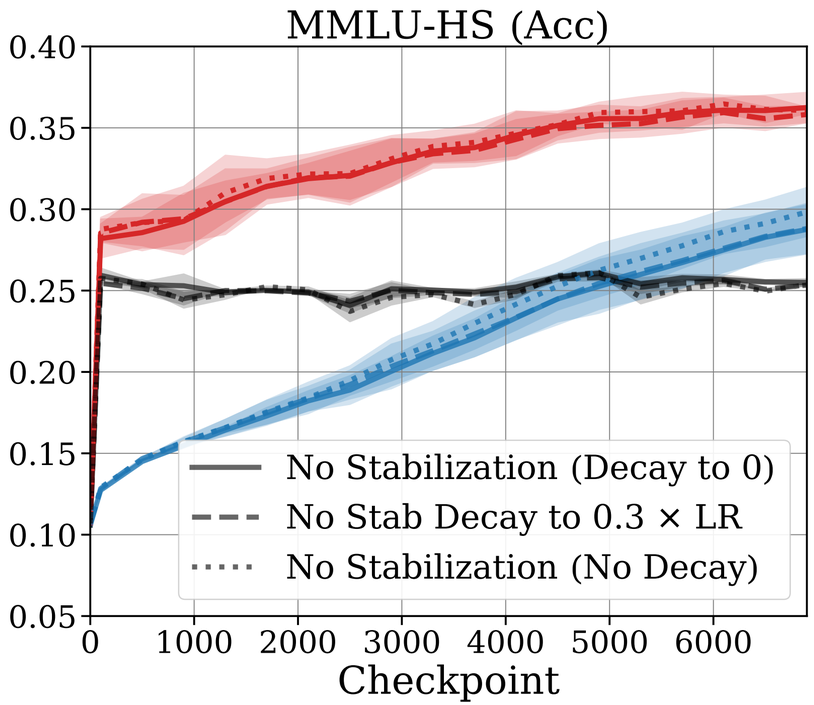




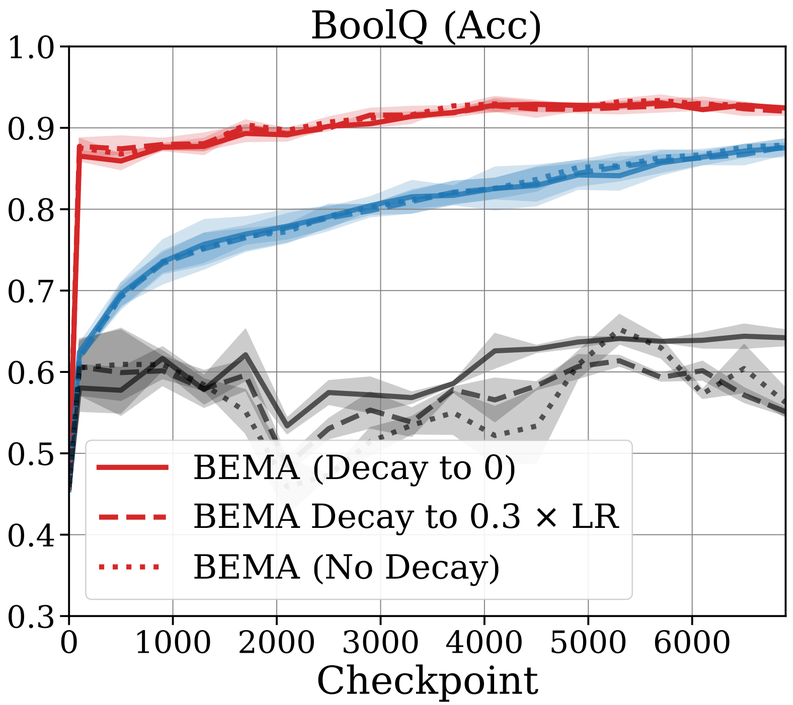
What about LR decay?

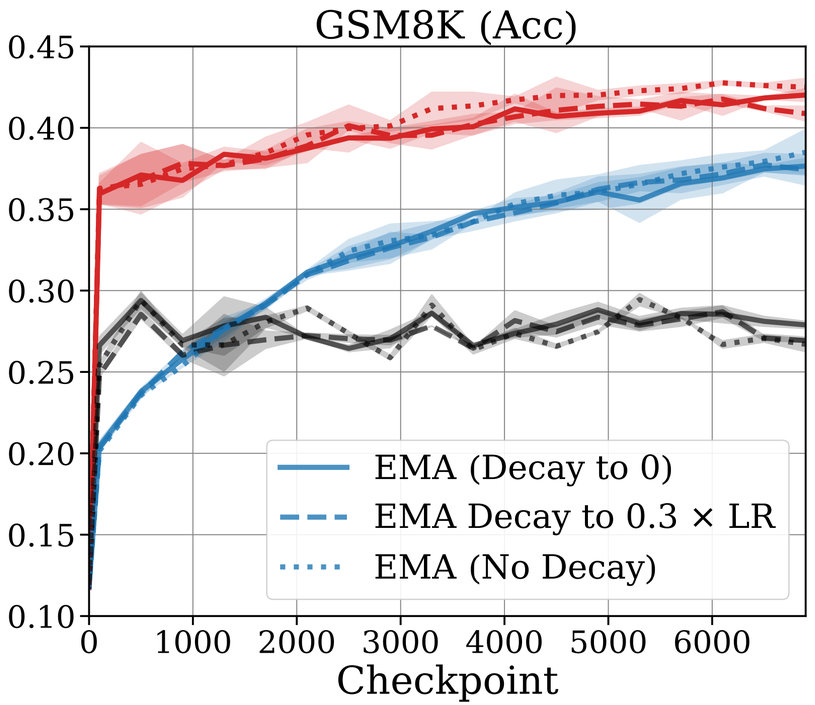


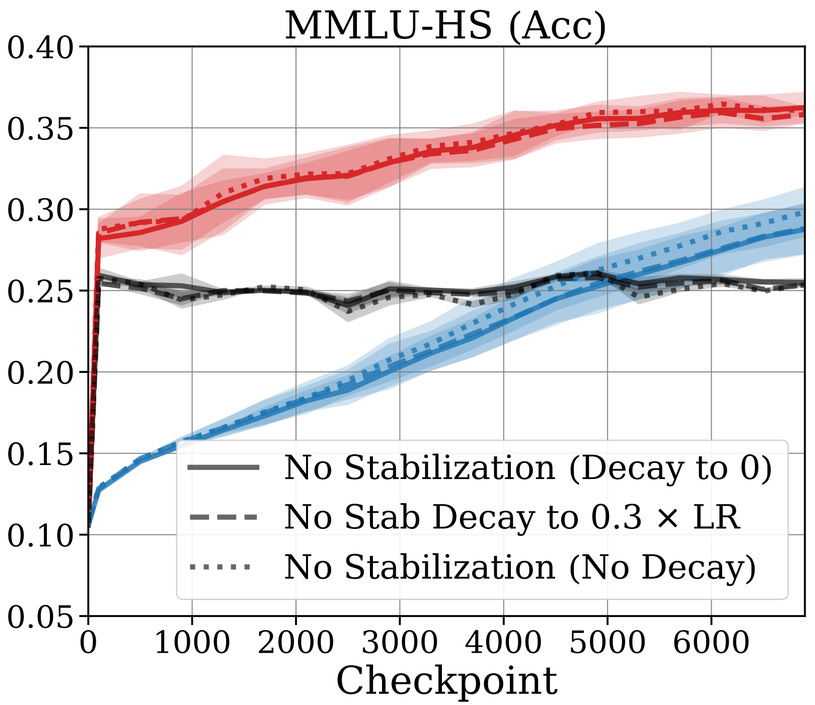




What about LR decay?







BEMA works across many training regimes and is robust to hyper parameters.

1. Why have we converged on such a **unified paradigm** to address different **problem settings**?

1. Why have we converged on such a **unified paradigm** to address different **problem settings**?

2. When does **Imitation Learning** succeed and how is it different from **supervised learning**?

1. Why have we converged on such a **unified paradigm** to address different **problem settings**?

2. When does **Imitation Learning** succeed and how is it different from **supervised learning**?

3. How can we leverage **theoretical principles** to (empirically) **improve** Imitation Learning?

1. What are general conditions under which active data collection helps IL?

2. What algorithmic interventions improve IL for downstream tasks?

3. What additional optimization challenges exist in IL beyond supervised learning?

Thank you!