

Lecture 8: Imitation Learning and RLHF

Emma Brunskill

CS234 Reinforcement Learning.

Select all that are true

- (a) Maximizing a lower bound on the performance gap between a new policy and old policy can ensure monotonic convergence T
- (b) Behavior cloning requires knowing the dynamics model F
- (c) DAGGER uses demonstrations from experts but no further interactions F
- (d) Not sure

Select all that are true

- a Maximizing a lower bound on the performance gap between a new policy and old policy can ensure monotonic convergence
- b Behavior cloning requires knowing the dynamics model
- c DAGGER uses demonstrations from experts but no further interactions
- d Not sure

How Can RL Enable This?

PM You

Please write me a program to demonstrate how RLHF works. Please be brief in your explanations and only say 1-2 sentences before you show me the code.

ChatGPT

Reinforcement Learning from Human Feedback (RLHF) trains a model to perform tasks based on human-derived feedback. Here's a simplified Python program that demonstrates RLHF using a scenario where an AI chooses responses in a conversation.

```
python                                Copy code

import numpy as np

# Define possible actions (responses) the AI can use
actions = ["I'm doing well, thanks!", "I'm not sure how I feel.", "Let's talk about RLHF"]
q_table = np.zeros(len(actions))  # Initialize Q-values for each action to zero

def human_feedback(action):
    feedback = {"I'm doing well, thanks!": 10, "I'm not sure how I feel.": -5, "Let's talk about RLHF": 0}
    return feedback[action]

# Hyperparameters
learning_rate = 0.1
discount_factor = 0.6

# Training loop
for episode in range(100):
    action_index = np.random.choice(len(actions))  # Randomly choose an action
```

- Last time: Policy search continued and Imitation Learning
- This time: Imitation Learning and RLHF
- Next time: Author of Direct Preference Optimization (best paper runner up at top ML conference) guest lecture

- Imitation Learning
 - Max entropy inverse RL
- Human feedback
 - RLHF

- Expert provides a set of **demonstration trajectories**: sequences of states and actions
- Imitation learning is useful when it is easier for the expert to demonstrate the desired behavior rather than:
 - Specifying a reward that would generate such behavior,
 - Specifying the desired policy directly

Problem Setup

- Input:
 - State space, action space
 - Transition model $P(s' | s, a)$ (sometimes)
 - No reward function R
 - Set of one or more expert's demonstrations $(s_0, a_0, s_1, a_1, \dots)$
(actions drawn from expert's policy π^*)
- Behavioral Cloning:
 - Can we directly learn the expert's policy using supervised learning?
- Inverse RL:
 - Can we recover R ?
- Apprenticeship learning via Inverse RL:
 - Can we use R to generate a good policy?



DAGGER

Recall: Feature Matching

$\pi \rightarrow \gamma \rightarrow (s, a)$ distrib

- Want to find a reward function such that the expert policy outperforms other policies.
- For a policy π to be guaranteed to perform as well as the expert policy π^* , sufficient if its discounted summed feature expectations match the expert's policy [Abbeel & Ng, 2004].
- More precisely, if

$$\|\mu(\pi) - \mu(\pi^*)\|_1 \leq \epsilon$$

then for all w with $\|w\|_\infty \leq 1$ (uses Holder's inequality):

$$|w^T \mu(\pi) - w^T \mu(\pi^*)| \leq \epsilon$$

- where here μ is used to represent the features experienced under π

Recall: Ambiguity

- There is an infinite number of reward functions with the same optimal policy.
- There are infinitely many stochastic policies that can match feature counts
- Which one should be chosen?

- Many different approaches
- Two of the key papers are:
 - Maximum Entropy Inverse Reinforcement Learning (Ziebart et al. AAAI 2008)
 - Generative adversarial imitation learning (Ho and Ermon, NeurIPS 2016)

Max Entropy Inverse RL. Ziebart et al. 2008. Note: Much of this presentation follows the slides from Katerina Fragkiadaki's Deep Reinforcement Learning and Control Lecture on Maximum Entropy Inverse RL.

$$\sum_s p(s) = 1 \quad \text{prob dist'n}$$

- Recall that entropy of a distribution $p(s)$ is $-\sum_{s'} p(s = s') \log p(s = s')$
- Principle of max entropy: The probability distribution which best represents the current state of knowledge is the one with the largest entropy, given the constraints of precisely stated prior data.
- Intuitively: consider all probability distributions consistent with observed data, and select the probability distribution with the maximum entropy.

- Recall that entropy of a distribution $p(s)$ is $-\sum_{s'} p(s = s') \log p(s = s')$
- Principle of max entropy: The probability distribution which best represents the current state of knowledge is the one with the largest entropy, given the constraints of precisely stated prior data.
- Intuitively: consider all probability distributions consistent with observed data, and select the probability distribution with the maximum entropy.
- In the linear reward case, this is equivalent to specifying the weights w that yield a policy with the max entropy constrained to matching the feature expectations:

$$\max_P - \sum_{\tau} P(\tau) \log P(\tau) \text{ s.t. } \sum_{\tau} P(\tau) \mu(\tau) = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \mu(\tau_i) \quad \sum_{\tau} P(\tau) = 1$$

experts (1)

- where $\mu(\tau)$ are the features for trajectory τ and \mathcal{D} is the observed expert data

- In the linear reward case, this is equivalent to specifying the weights w that yield a policy with the max entropy constrained to matching the feature expectations:

$$\max_P - \sum_{\tau} P(\tau) \log P(\tau) \text{ s.t. } \sum_{\tau} P(\tau) \mu(\tau) = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \mu(\tau_i) \quad \sum_{\tau} P(\tau) = 1 \quad (2)$$

- where $\mu(\tau)$ are the features for trajectory τ and \mathcal{D} is the observed expert data
- In general, would like to find a policy π that induces a distribution over trajectories $p(\tau)$ which has the same expected reward as the expert's demonstrations $\hat{P}(\tau)$ given a reward function r_ϕ

$$\max_{p(\tau)} - \sum_{\tau} p(\tau) \log p(\tau) \quad \text{s.t.} \quad \sum_{\tau} p(\tau) r_\phi(\tau) = \sum_{\tau} \hat{P}(\tau) r_\phi(\tau) \quad \sum_{\tau} p(\tau) = 1$$

expert (3)

Matching Rewards to Learning Policies

- In the linear reward case, this is equivalent to specifying the weights w that yield a policy with the max entropy constrained to matching the feature expectations:

$$\max_P - \sum_{\tau} P(\tau) \log P(\tau) \text{ s.t. } \sum_{\tau} P(\tau) \mu(\tau) = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \mu(\tau_i) \quad \sum_{\tau} P(\tau) = 1 \quad (4)$$

- where $\mu(\tau)$ are the features for trajectory τ and \mathcal{D} is the observed expert data
- In general, would like to find a policy π that induces a distribution over trajectories $p(\tau)$ which has the same expected reward as the expert's demonstrations $\hat{P}(\tau)$ given a reward function r_ϕ

$$\max_{p(\tau)} - \sum_{\tau} p(\tau) \log p(\tau) \quad \text{s.t.} \quad \sum_{\tau} p(\tau) r_\phi(\tau) = \sum_{\tau} \hat{P}(\tau) r_\phi(\tau) \quad \sum_{\tau} p(\tau) = 1 \quad (5)$$

- To do so, will alternate between computing a reward function, using that reward function to learn an optimal policy, and then updating the trajectory / state frequencies needed to update the reward function
- Note: in original maximum entropy inverse RL paper, assumed dynamics / reward model is known

From Maximum Entropy to Probability over Trajectories

- given constrained obj. if we know the costs, what would be the form of the distib over τ ?

$$\mathcal{L}(\rho, \lambda) = \sum_{\tau} p(\tau) \log p(\tau) + \lambda_1 (\underbrace{\sum_{\tau} \hat{p}(\tau) r_{\phi}(\tau)}_{-\sum_{\tau} p(\tau) r_{\phi}(\tau)}) + \lambda_0 (\sum_{\tau} p(\tau) - 1)$$

$$\frac{d\mathcal{L}}{dp(\tau)} = \log p(\tau) + p(\tau) \cdot \frac{1}{p(\tau)} - \lambda_1 r_{\phi}(\tau) + \lambda_0 = 0$$

$$\log p(\tau) = -1 + \lambda_1 r_{\phi}(\tau) - \lambda_0$$

$$p(\tau) = e^{-1 + \lambda_1 r_{\phi}(\tau) - \lambda_0}$$

$$p(\tau) \propto e^{r_{\phi}(\tau)}$$

Maximizing Entropy Over $\tau \equiv$ Maximize Likelihood of Observed Data
Under Max Entropy (Exponential Family) Distribution (Jaynes 1957)

$$p(\gamma_i | \phi) = \frac{1}{Z(\phi)} e^{r_\phi(\gamma_i)}$$

$$Z(\phi) = \sum_{\gamma} e^{r_\phi(\gamma)}$$

$$p(\gamma_i | \phi) = \frac{1}{Z(\phi)} e^{\sum_{s_i \in \gamma_i} r_\phi(s_i)}$$

we don't know what r_ϕ is constraint
this observation (prob over γ that max 1
entropy looks like a normalized expn)
means we can now estimate / learn r_ϕ
by max prob of our observed data

Maximize (over r_ϕ) Likelihood of Observed Data Under Max Entropy

$$\max_{\phi} \log \prod_{Y^* \in D} p(Y^*)$$

$$\max_{\phi} \sum_{Y^* \in D} \log \left(\frac{e^{r_\phi(Y^*)}}{\sum e^{r_\phi(Y)}} \right)$$

$$= \max_{\phi} \sum_{Y^* \in D} r_\phi(Y^*) - \underbrace{\log \sum_Y e^{r_\phi(Y)}}_{J(\phi)}$$

$$J(\phi) = \max_{\phi} \sum_{Y^* \in D} r_\phi(Y^*) - |D| (\log \sum_Y e^{r_\phi(Y)})$$

$$\nabla_{\phi} J(\phi) = \sum_{Y^* \in D} \frac{d r_\phi(Y^*)}{d \phi} - |D| \underbrace{\frac{1}{\sum_Y e^{r_\phi(Y)}}}_{\text{Note} \Rightarrow p(Y|\phi)} \sum_Y e^{r_\phi(Y)} \frac{d r_\phi(Y)}{d \phi}$$

$$= \sum_{Y^* \in D} \frac{d r_\phi(Y^*)}{d \phi} - |D| \sum_Y p(Y|\phi) \frac{d r_\phi(Y)}{d \phi}$$

From Trajectories to States

$$p(\gamma) = p(s_1) \prod_{t=1}^T p(a_t | s_t) p(s_{t+1} | s_t, a_t)$$

$$p(\gamma) \propto e^{-r_p(\gamma)} = e^{-\sum_{s \in \gamma} r_\phi(s)}$$

$$\nabla_\theta J(\theta) = \sum_{s \in \gamma^* \in D} \frac{d r_\phi(s)}{d \theta} - |D| \sum_s p(s | \theta, \gamma) \frac{d r_\phi(s)}{d \theta}$$

State Densities

if tabular & dynamics known so π is given

$$\mu_t(s) = \rho(s_t)$$

for $t=1\dots T$ (time index t)

$$\mu_{t+1}(s) = \sum_a \sum_{s'} \mu_t(s') \rho(a|s') \rho(s|s; a)$$

$$\rho(s_t|\phi, \gamma) = \sum_t \mu_t(s)$$

$$\nabla_\phi J(\phi) = \sum_{s_t \in \text{ep}} \frac{\partial r_\phi(s)}{\partial \phi} - 1/D \sum_s \rho(s|\phi, T) \frac{\partial r_\phi(s)}{\partial \phi}$$

if $r_\phi(s) = \phi f_s$ (linear)

$$\frac{\partial r_\phi(s)}{\partial \phi} = f_s$$

Note: Assuming known dynamics model and linear rewards

- ① Input: expert demonstrations \mathcal{D}
- ② Initialize r_ϕ
- ③ Compute optimal $\pi(a|s)$ given r_ϕ e.g. with value iteration
- ④ Compute state visitation frequencies $p(s|\phi, T)$ (linear)
- ⑤ Compute gradient on reward model

$$\nabla J(\phi) = \frac{1}{M} \sum_{t_d \in \mathcal{D}} f_{t_d} - \sum_s p(s|\phi, T) f_s \quad (6)$$

- ⑥ Update ϕ with one gradient step
- ⑦ Go to step 3

Note: Assuming known dynamics model and linear rewards

- ① Input: expert demonstrations \mathcal{D}
- ② Initialize r_ϕ
- ③ Compute optimal $\pi(a|s)$ given r_ϕ e.g. with value iteration)
- ④ Compute state visitation frequencies $p(s(\phi, T))$
- ⑤ Compute gradient on reward model

$$\nabla J(\phi) = \frac{1}{M} \sum_{\gamma_d \in \mathcal{D}} f_\zeta - \sum_s p(\zeta | \phi, \gamma) f_s \quad (7)$$

- ⑥ Update ϕ with one gradient step
 - ⑦ Go to step 3
- What steps in the above algorithm rely on knowing the dynamics model? (select all)
 - (1) Computing the optimal policy
 - (2) Computing the state visitation frequencies
 - (3) Computing the gradient
 - (4) No steps required it
 - (5) Not sure

Check Your Understanding: L8N2 Maximum entropy inverse RL Solutions

Note: Assuming known dynamics model and linear rewards

- ① Input: expert demonstrations \mathcal{D}
- ② Initialize r_ϕ
- ③ Compute optimal $\pi(a|s)$ given r_ϕ e.g. with value iteration
- ④ Compute state visitation frequencies $p(s|\phi, T)$
- ⑤ Compute gradient on reward model

$$\nabla J(\phi) = \quad (8)$$

- ⑥ Update ϕ with one gradient step
- ⑦ Go to step 3
 - What steps in the above algorithm rely on knowing the dynamics model? (select all)
 - (1) Computing the optimal policy
 - (2) Computing the state visitation frequencies
 - (3) Computing the gradient
 - (4) No steps required it
 - (5) Not sure

1 and 2

- Max entropy approach has been hugely influential
- Initial formulation (Ziebart et al) using linear rewards and assumed dynamics model is known
 - Check your understanding: was this needed in behavioral cloning?
- Finn et al. 2016 (Guided cost learning: Deep inverse optimal control via policy optimization) showed how to use general reward/cost functions and removed the need to know the dynamics model

- Imitation learning can greatly reduce the amount of data need to learn a good policy
- Challenges remain and one exciting area is combining inverse RL / learning from demonstration and online reinforcement learning
- For a look into some of the theory between imitation learning and RL, see Sun, Venkatraman, Gordon, Boots, Bagnell (ICML 2017)

- Define behavior cloning and how it differs from reinforcement learning
- Understand principle of maximum entropy, the resulting distribution over trajectories, and how this can be used to learn a reward function and fit a policy

Batch / Offline RL Policy Evaluation & Optimization

Emma Brunskill
CS234
Spring 2024

Class Outline

- Last time: Learning from Past Human Preferences, RLHF and DPO
- **Today: Learning from Past Decisions and Actions, Offline RL**
- Next time: Fast / Data efficient RL (and bandits, relevant to HW3)

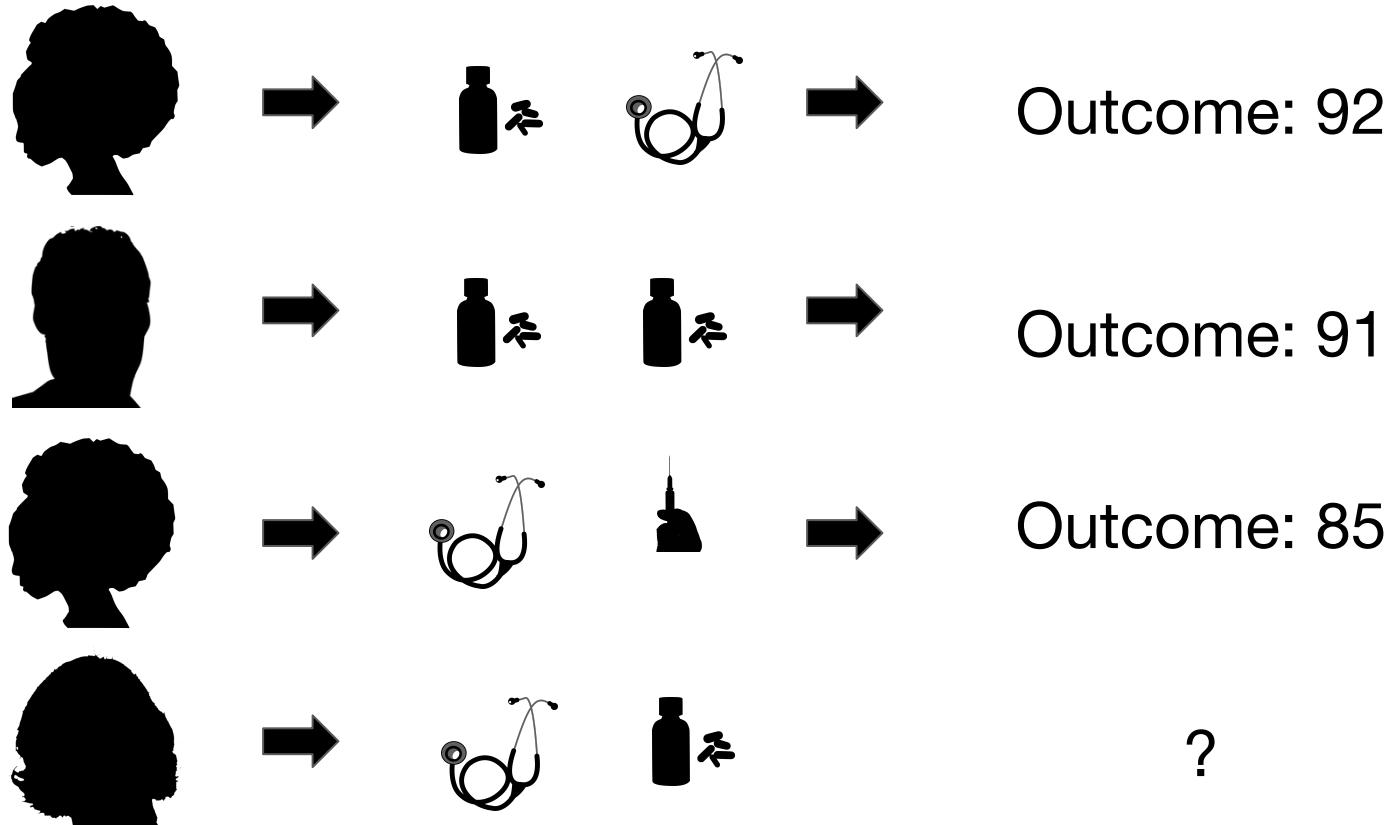
Learning from the Past

- Learning from Past Human Demonstrations: Imitation Learning
- Learning from Past Human Preferences: RLHF and DPO
- **Learning from Past Decisions and Actions: Offline RL**

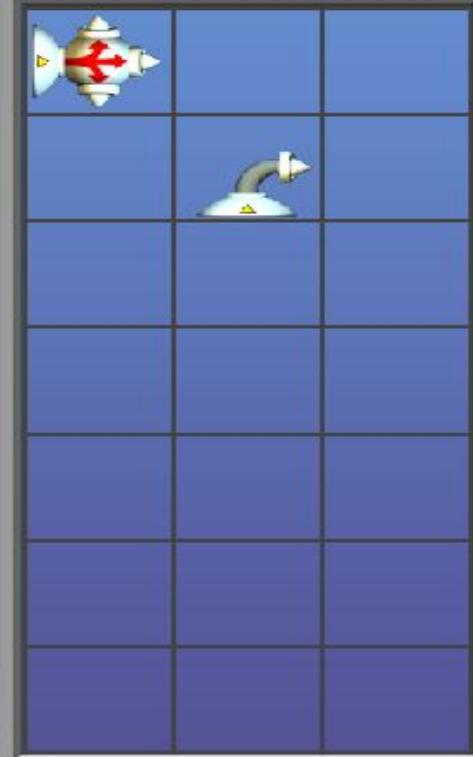
Outline for Today

1. Introduction and Setting
2. Offline batch policy evaluation
 - a. Using models
 - b. Using model free methods
 - c. Use importance sampling
3. Offline policy learning / optimization

Can We Do Better than Imitation Learning?



Level 1:8
Fork

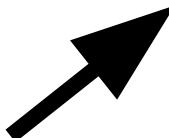


MENU

OPTIONS

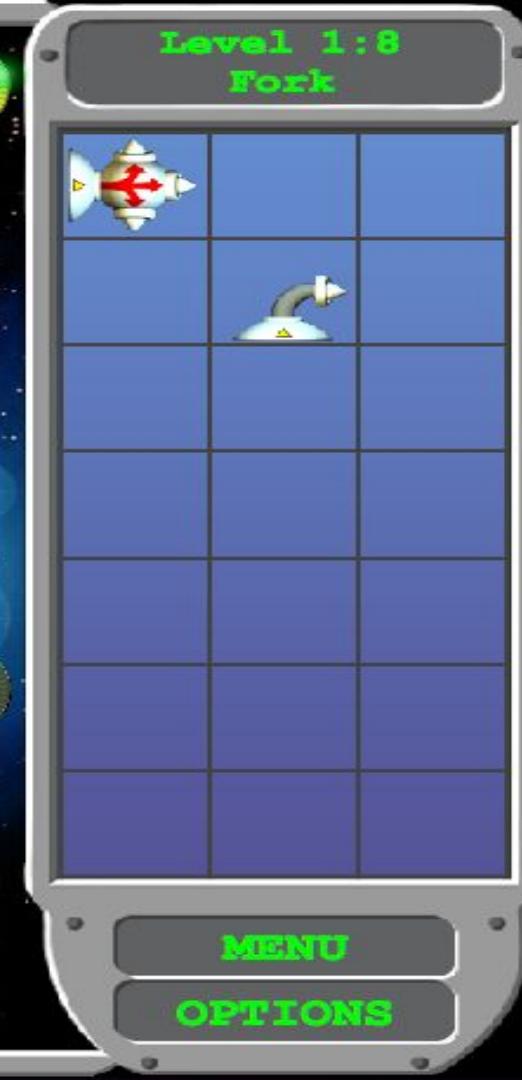
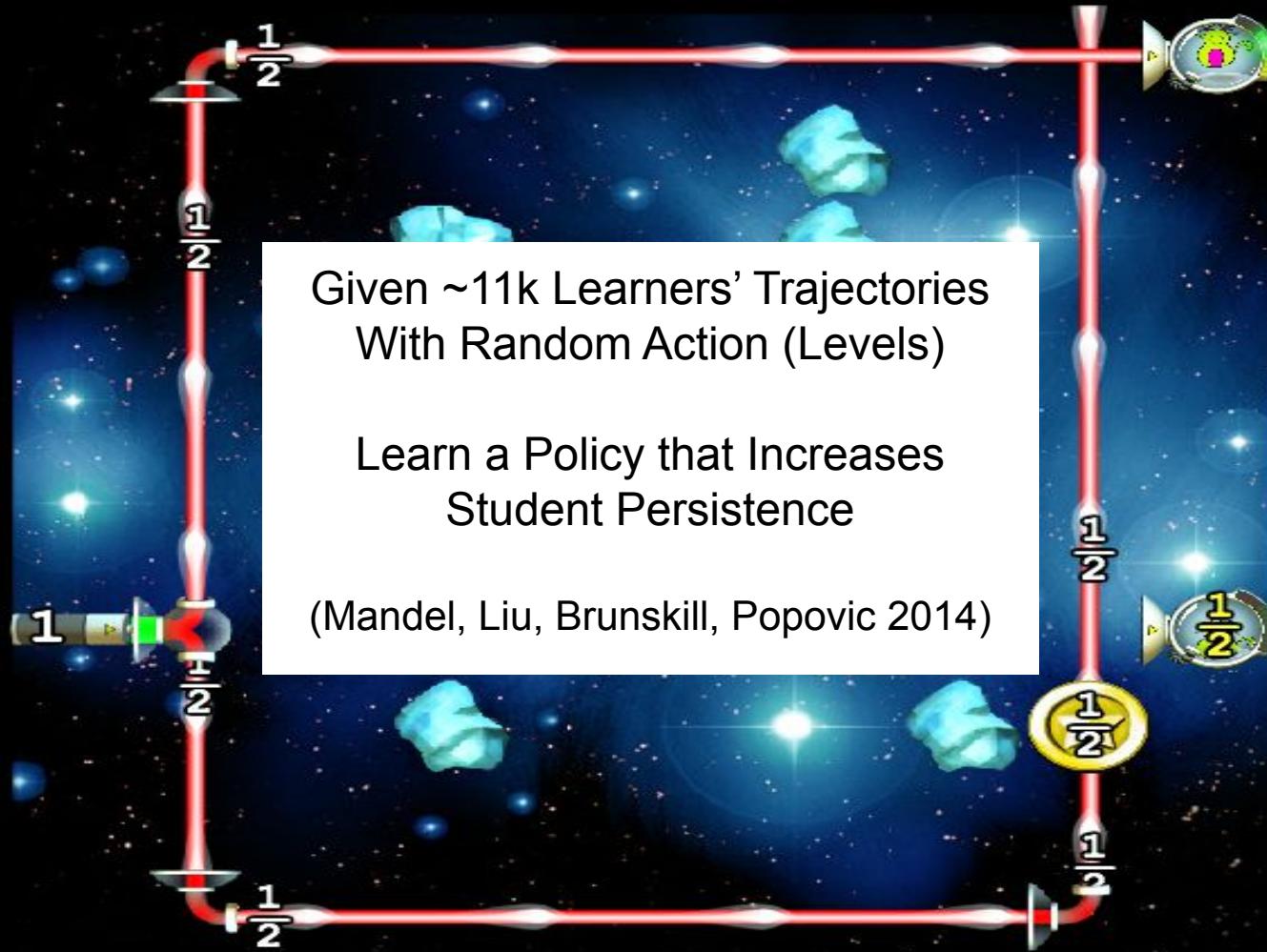


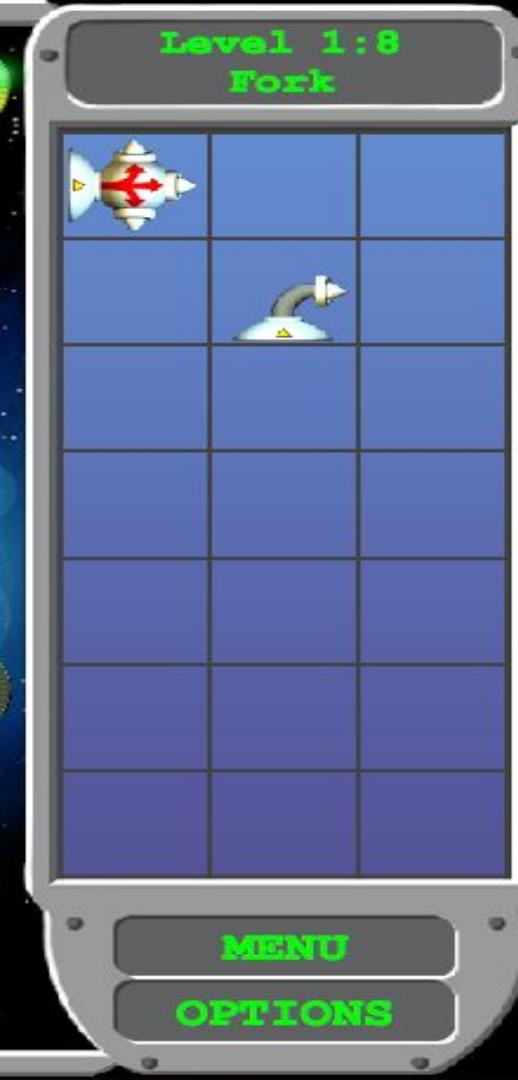
Took > 30s



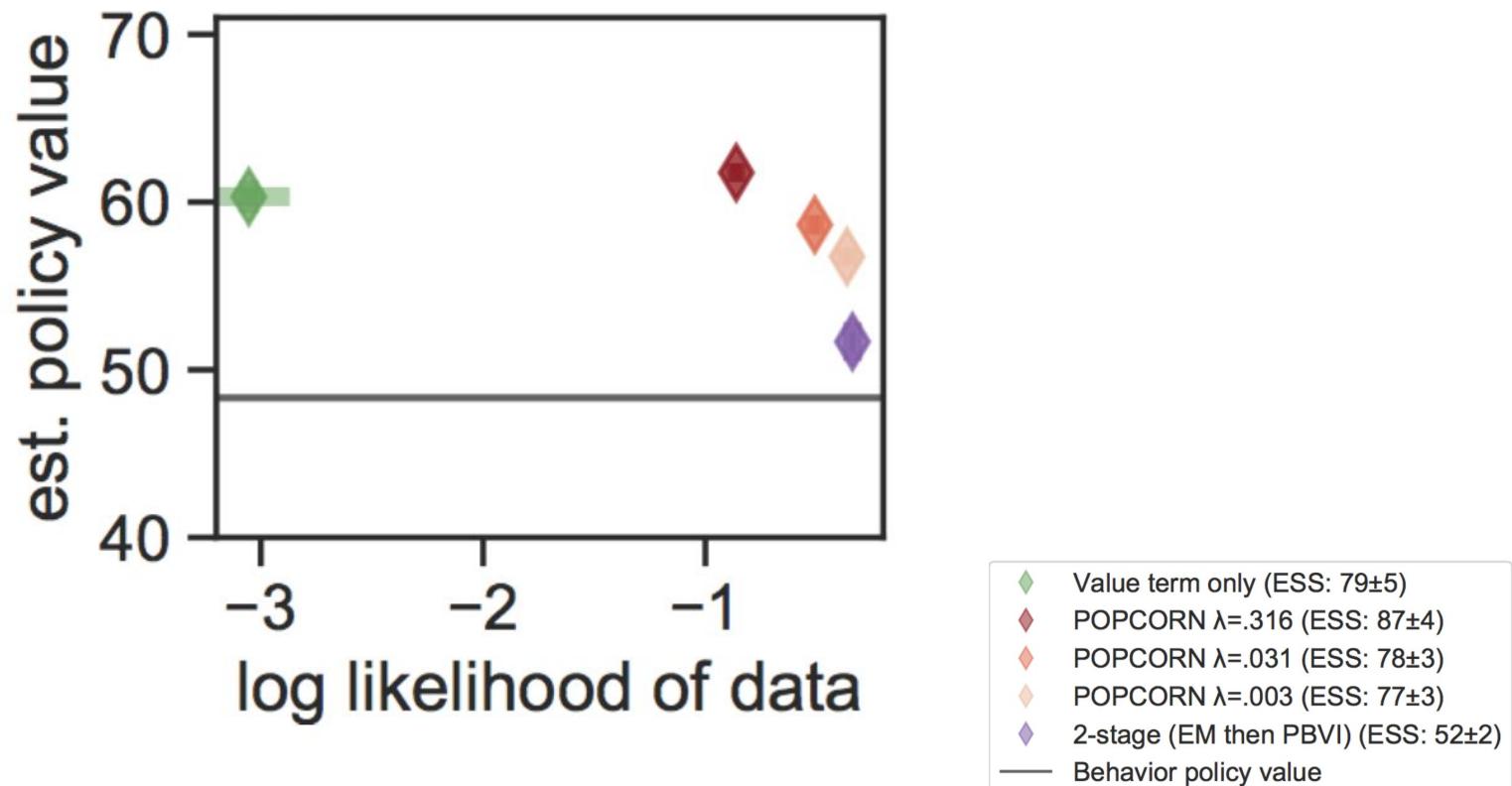
Took <= 30s



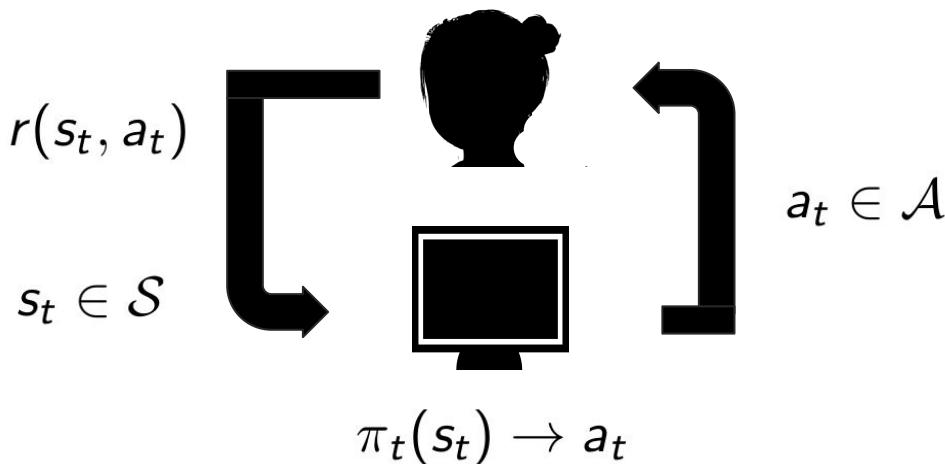




Encouraging Work on Observational Health Data (MIMIC) Hypotension

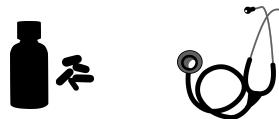


New Topic: Counterfactual / Batch RL



\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$

Patient group 1 →



Outcome: 92

Patient group 2 →



Outcome: 91

Patient group 1 →



Outcome: 92

Patient group 2 →



Outcome: 91



?

“What If?” Reasoning Given Past Data

Patient group 1 →   → Outcome: 92

Patient group 2 →   → Outcome: 91



What information would you want to know in order to decide, given the above evidence, how best to treat new patient?

Data Is Censored in that Only Observe Outcomes for Decisions Made

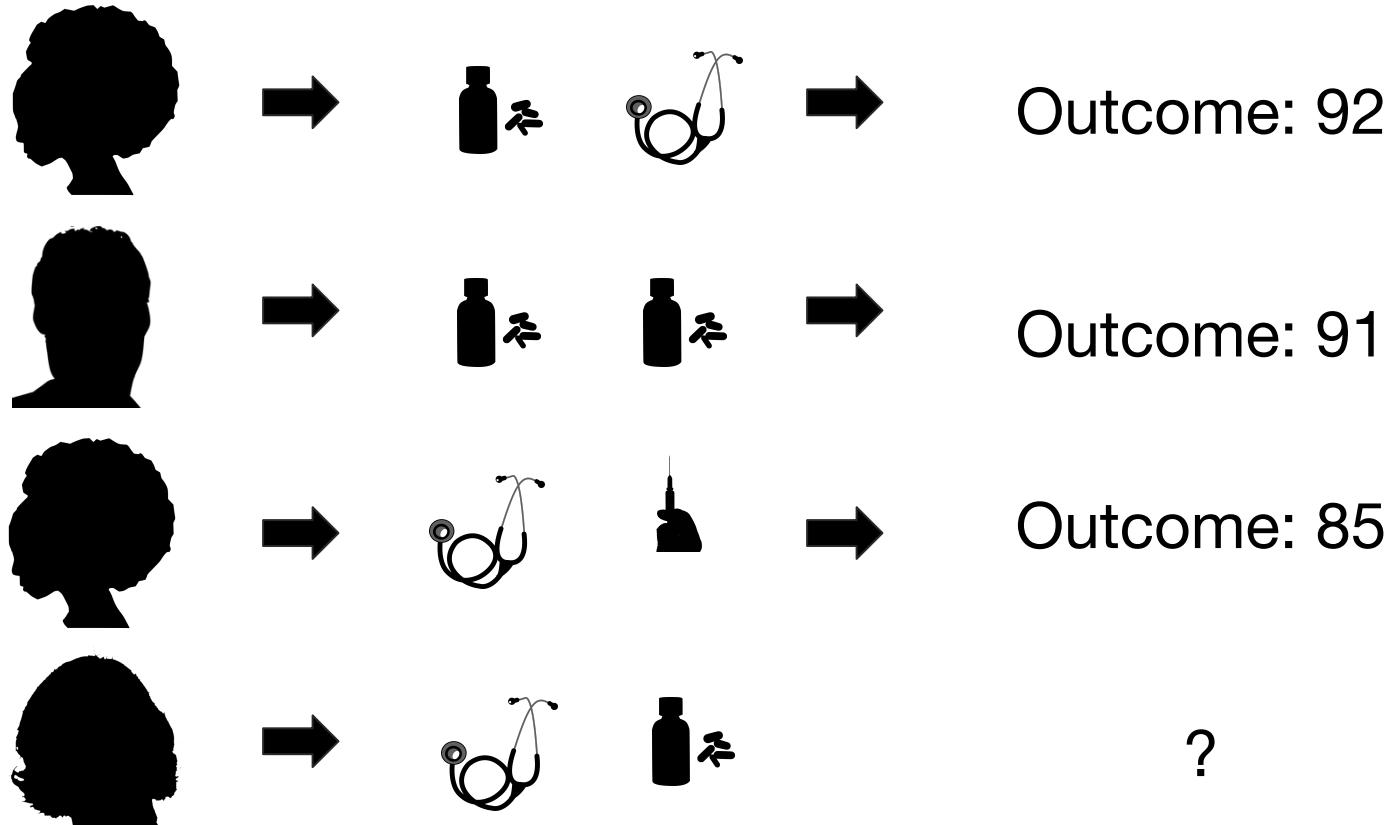
Patient group 1 →   → Outcome: 92

Patient group 2 →   → Outcome: 91

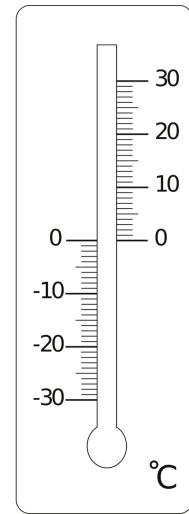


?

Need for Generalization



Potential Applications



Off Policy Reinforcement Learning

Watkins 1989

Watkins and Dayan 1992

Precup et al. 2000

Lagoudakis and Parr 2002

Murphy 2005

Sutton, Szepesvari and Maei 2009

Shortreed, Laber, Lizotte, Stroup, Pineau, & Murphy 2011

Degirs, White, and Sutton 2012

Mnih et al. 2015

Mahmood et al. 2014

Jiang & Li 2016

Hallak, Tamar and Mannor 2015

Munos, Stepleton, Harutyunyan and Bellemare 2016

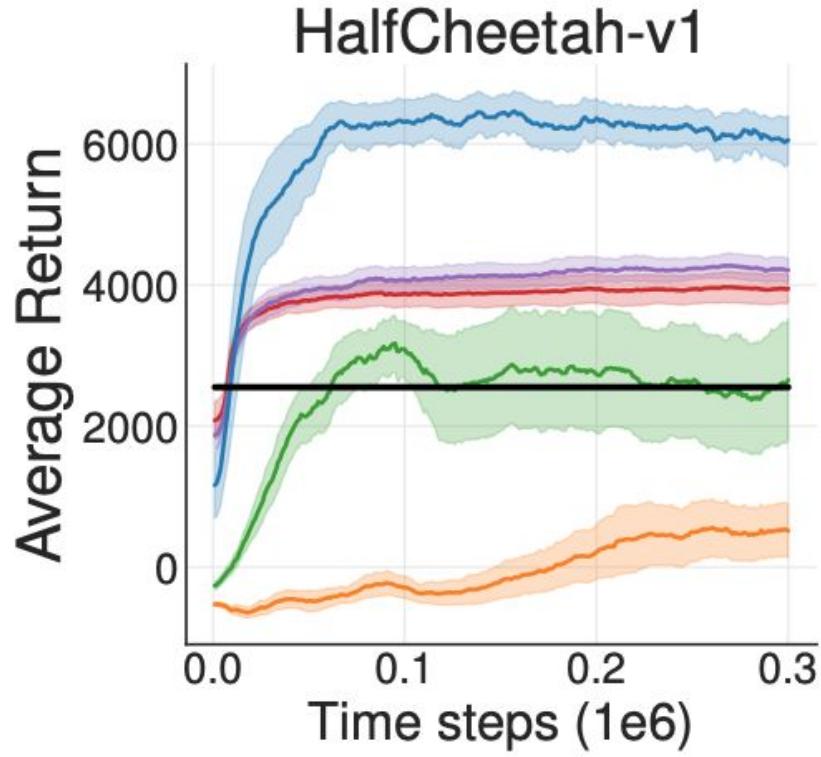
Sutton, Mahmood and White 2016

Du, Chen, Li, Ziao, and Zhou 2016 ...

Why Can't We Just Use Q-Learning?

- Q-learning is an off policy RL algorithm
 - Can be used with data different than the state--action pairs would visit under the optimal Q state action values
- But deadly triad of bootstrapping, function approximation and off policy, and can fail

Important in Practice



BCQ figure from Fujimoto,
Meger, Precup ICML 2019

BCQ

DDPG

DQN

BC

VAE-BC

Behavioral
22

Outline for Today

1. Introduction and Setting
2. Offline batch policy evaluation
 - a. Using models
 - b. Using model free methods
 - c. Use importance sampling
3. Offline policy learning / optimization

Batch Policy Evaluation: Estimate Performance of a Specific Decision Policy

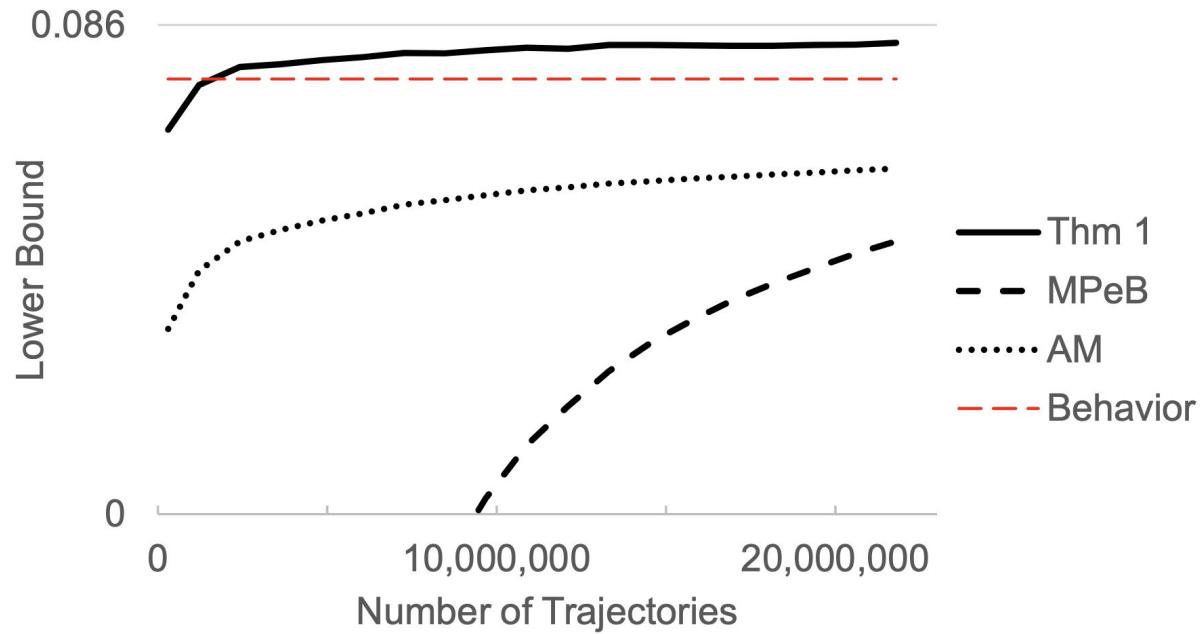
$$\underbrace{\int_{s \in S_0} \hat{V}^\pi(s, \mathcal{D}) ds}_{\text{Policy Evaluation}}$$

batch policy



\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$
 π : Policy mapping $s \rightarrow a$
 S_0 : Set of initial states
 $\hat{V}^\pi(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

Sample Efficient Methods Matter Policy Evaluation

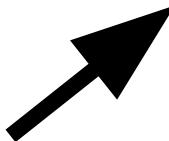


Outline for Today

1. Introduction and Setting
2. Offline batch policy evaluation
 - a. **Using models**
 - b. Using model free methods
 - c. Use importance sampling
3. Offline policy learning / optimization



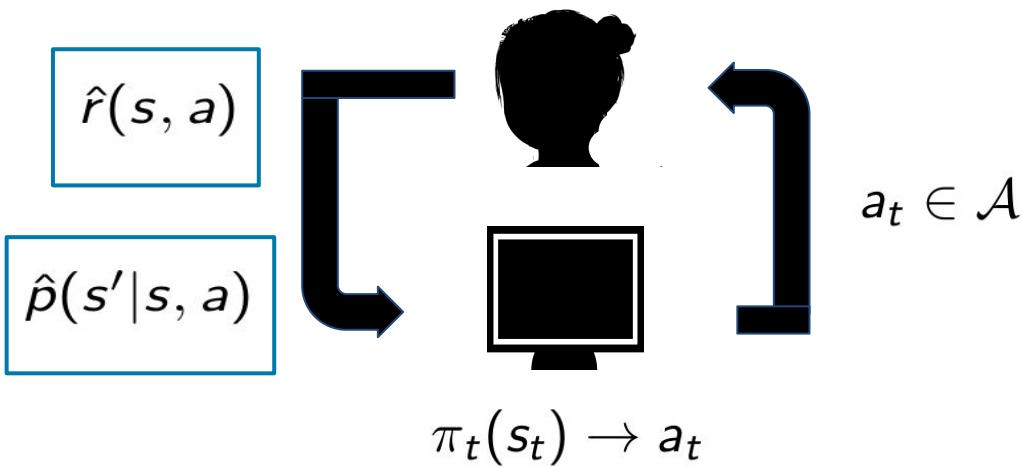
Took > 30s



Took <= 30s

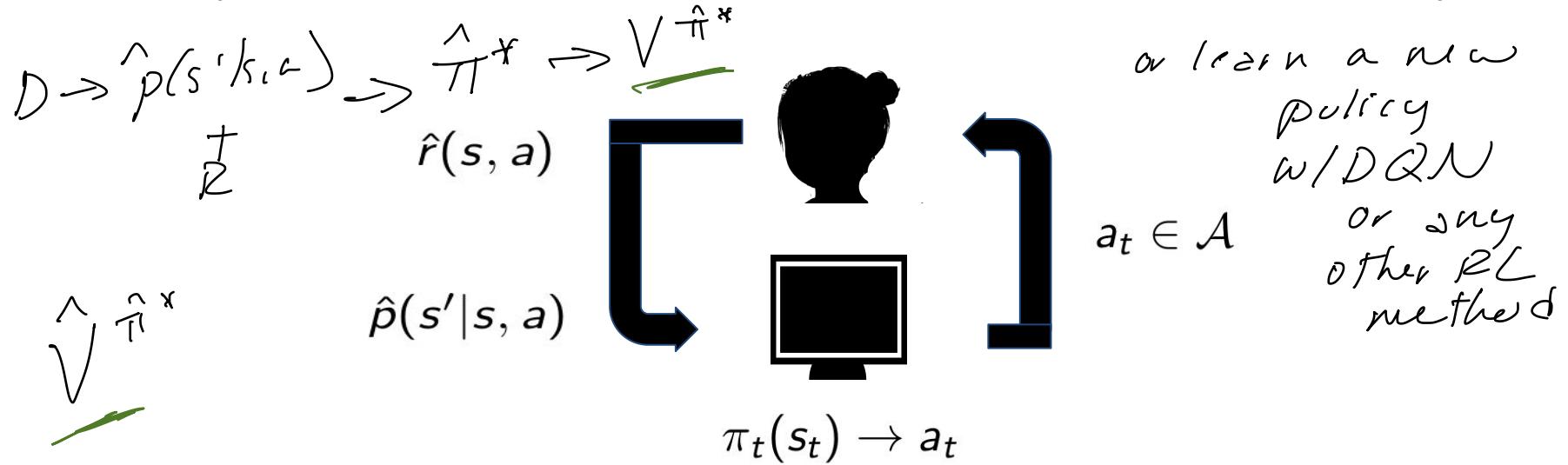


Learn Dynamics and Reward Models from Data



\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$
 π : Policy mapping $s \rightarrow a$
 S_0 : Set of initial states
 $\hat{V}^\pi(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

Learn Dynamics and Reward Models from Data, Evaluate Policy



$$V^\pi \approx (I - \gamma \hat{P}^\pi)^{-1} \hat{R}^\pi$$

$$P^\pi(s'|s) = p(s'|s, \pi(s))$$

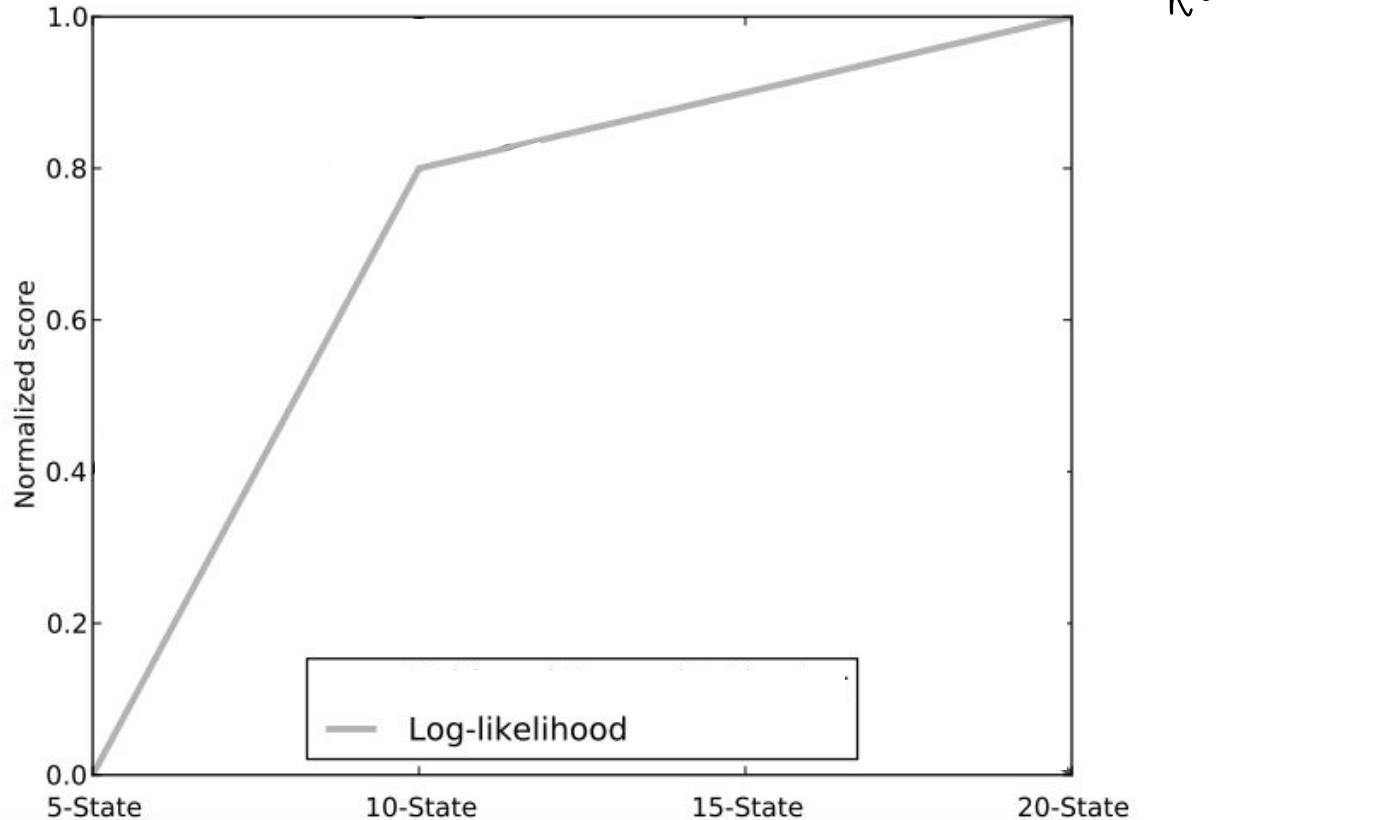
\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$

π : Policy mapping $s \rightarrow a$

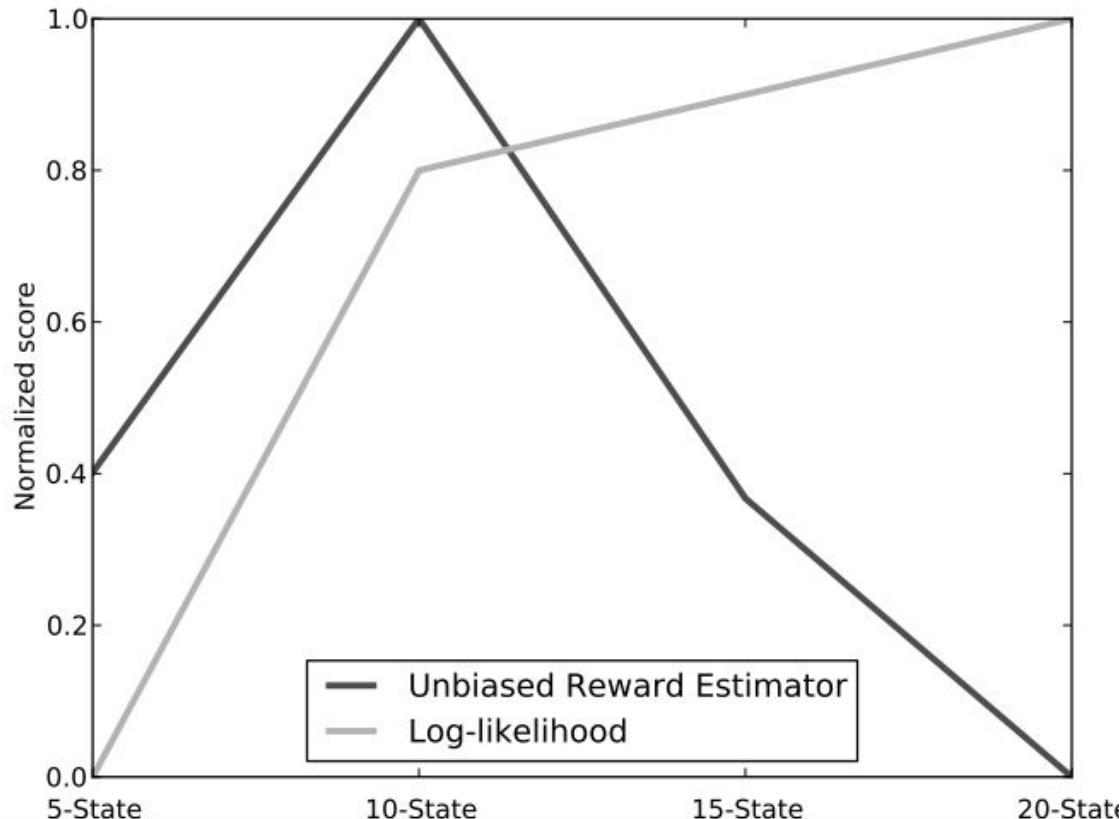
S_0 : Set of initial states

$\hat{V}^\pi(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

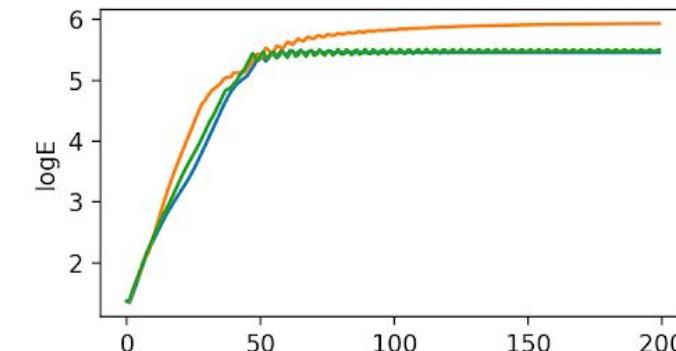
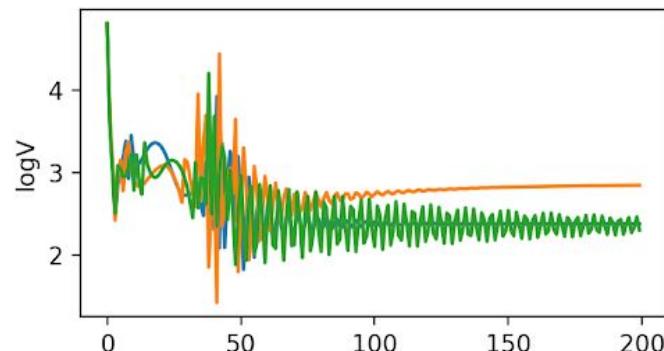
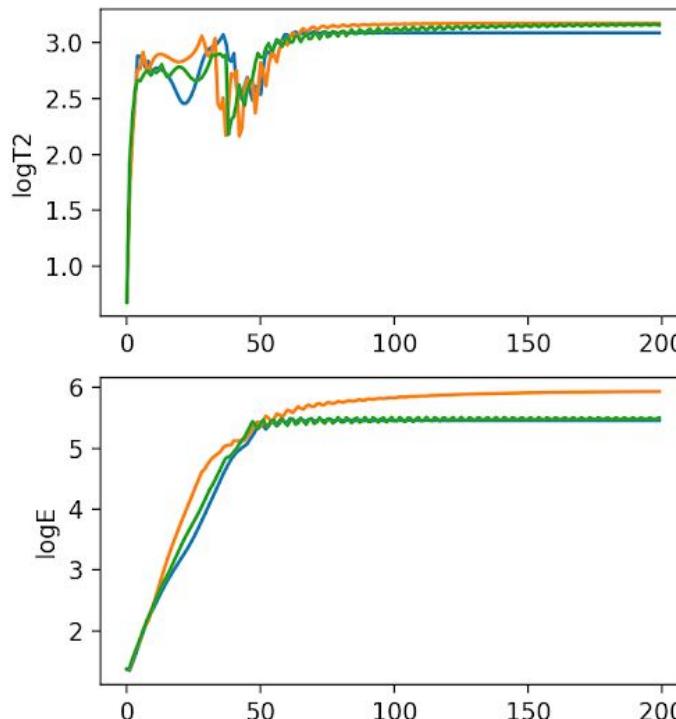
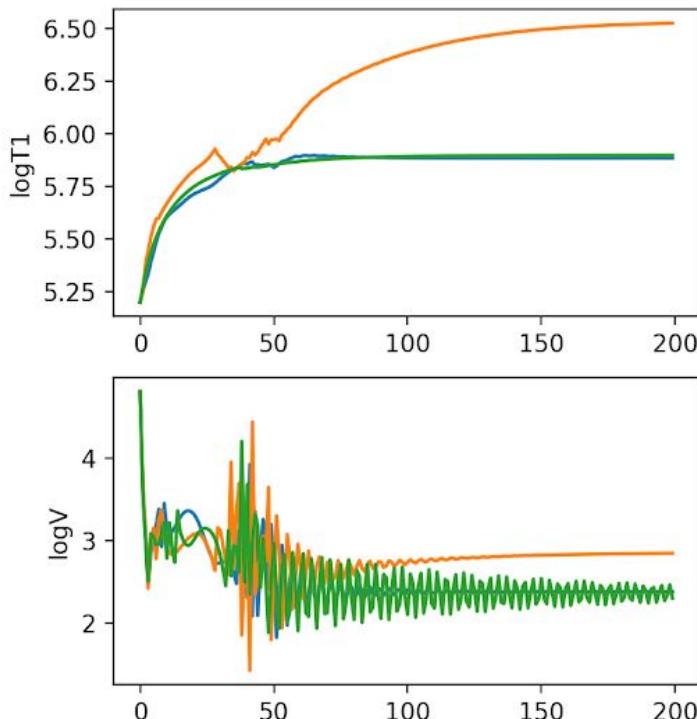
Better Dynamics/Reward Models for Existing Data (Improve likelihood)



Better Dynamics/Reward Models for Existing Data, May **Not** Lead to Better Policies for Future Use → Bias due to Model Misspecification



Models Fit for Off Policy Evaluation Can Result in Better Estimates When Trained Under a **Different Loss Function**



— RepBM — MLE Model — Ground truth

Outline for Today

1. Introduction and Setting
2. Offline batch policy evaluation
 - a. Using models
 - b. **Using model free methods**
 - c. Use importance sampling
3. Offline policy learning / optimization

Model Free Value Function Approximation: Fitted Q Evaluation

FQ | Iteration

$$\mathcal{D} = (s_i, a_i, r_i, s_{i+1}) \quad \forall i$$

$$\tilde{Q}^\pi(s_i, a_i) = r_i + \gamma V_\theta^\pi(s_{i+1}) \quad \text{target for}$$

$$\arg \min_{\theta} \sum_i (Q_\theta^\pi(s_i, a_i) - \tilde{Q}^\pi(s_i, a_i))^2$$

Ernesto?

- Fitted Q evaluation, LSTD, ...

\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$
π : Policy mapping $s \rightarrow a$
S_0 : Set of initial states
$\hat{V}^\pi(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

Algorithm 3 Fitted Q Evaluation: FQE(π, c)

Input: Dataset $D = \{x_i, a_i, x'_i, c_i\}_{i=1}^n \sim \pi_D$. Function class F .

Policy π to be evaluated

- 1: Initialize $Q_0 \in F$ randomly
- 2: **for** $k = 1, 2, \dots, K$ **do**
- 3: Compute target $y_i = c_i + \gamma Q_{k-1}(x'_i, \pi(x'_i)) \quad \forall i$
- 4: Build training set $\tilde{D}_k = \{(x_i, a_i), y_i\}_{i=1}^n$
- 5: Solve a supervised learning problem:
$$Q_k = \arg \min_{f \in F} \frac{1}{n} \sum_{i=1}^n (f(x_i, a_i) - y_i)^2$$

- 6: **end for**

Output: $\hat{C}^\pi(x) = Q_K(x, \pi(x)) \quad \forall x$ only for π fixed

Let's assume
we use a DNN
for F .

What is
different vs
DQN?

no $m \geq x$

Example Fitted Q Evaluation Guarantees

$$d_F^\pi = \sup_{g \in F} \inf_{f \in F} \|f - B^\pi g\|_\pi$$

Theorem 4.2 (Generalization error of FQE). Under Assumption 1, for $\epsilon > 0$ & $\delta \in (0, 1)$, after K iterations of Fitted Q Evaluation (Algorithm 3), for $n = O\left(\frac{\bar{C}^4}{\epsilon^2} \left(\log \frac{K}{\delta} + \dim_F \log \frac{\bar{C}^2}{\epsilon^2} + \log \dim_F \right)\right)$, we have with probability $1 - \delta$:

$$\left| \int_{s_0 \in \rho} \hat{V}^\pi(s_0) - V^\pi(s_0) \right| \leq \frac{\gamma^5}{(1 - \gamma)^{1.5}} \left(\sqrt{\beta_u} (2d_F^\pi + \epsilon) + \frac{2\gamma^{K/2}\bar{C}}{(1 - \gamma)^{.5}} \right)$$

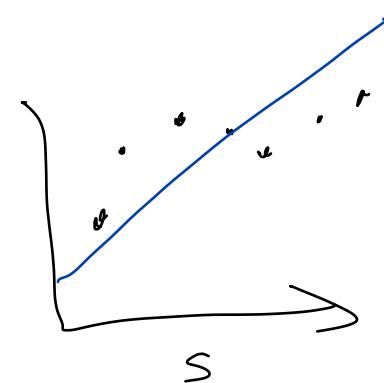
Concentration
Coefficients
Shift distribution
"unbiased"
much better
new desired target
accuracy

\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$
 π : Policy mapping $s \rightarrow a$
 S_0 : Set of initial states
 $\hat{V}^\pi(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

Model Free Policy Evaluation

- Challenge: still relies on Markov assumption
- Challenge: still relies on models being well specified or have no computable guarantees if there is misspecification

$$d_F^\pi = \sup_{g \in F} \inf_{f \in F} \|f - B^\pi g\|_\pi$$



Outline for Today

1. Introduction and Setting
2. Offline batch policy evaluation
 - a. Using models
 - b. Using model free methods
 - c. **Use importance sampling**
3. Offline policy learning / optimization

Off Policy Evaluation With Minimal Assumptions

- Would like a method that doesn't rely on models being correct or Markov assumption
- Monte Carlo methods did this for online policy evaluation
- We would like to do something similar
- Challenge: data distribution mismatch

Importance Sampling*

$x = \text{state}$, $r(x) = \text{reward of state}$
 $p(x) = \text{prob of reaching } x \text{ under a policy}$
 but no data from $p(x)$

$$\begin{aligned} \mathbb{E}_p[r] &= \sum_x p(x)r(x) \\ &= \sum_x \frac{q(x)}{q(x)} p(x)r(x) \quad q(x) \text{ is a different policy} \\ &= \sum_x q(x) \left[\frac{p(x)}{q(x)} r(x) \right] \\ &\approx \frac{1}{N} \sum_{i=1}^N \frac{p(x_i)}{q(x_i)} r(x_i) \quad \text{unbiased estimate} \\ &\quad x \sim q(x) \end{aligned}$$

*Former CS234 student said this was his favorite idea of the class!

Importance Sampling: Can Compute Expected Value Under An Alternate Distribution!

$$\begin{aligned}\mathbb{E}_p[r] &= \sum_x p(x)r(x) \\ &= \sum_x \frac{p(x)q(x)}{q(x)}r(x) \\ &\approx \frac{1}{N} \sum_{i=1, x \sim q}^N \frac{p(x_i)}{q(x_i)}r(x_i)\end{aligned}$$

Importance Sampling is an Unbiased Estimator of True Expectation Under Desired Distribution If

$$\begin{aligned}\mathbb{E}_p[r] &= \sum_x p(x)r(x) \\ &= \sum_x \frac{p(x)q(x)}{q(x)}r(x) \\ &\approx \frac{1}{N} \sum_{i=1, x \sim q}^N \frac{p(x_i)}{q(x_i)}r(x_i)\end{aligned}$$

* consider healthcare
where may want to
consider patients
getting different
treatments

x_i are ✓

- The sampling distribution $q(x) > 0$ for all x s.t. $p(x) > 0$ (Coverage / overlap)
- No hidden confounding ✗

Check Your Understanding: Importance Sampling

$\sum p^h$

We can use importance sampling to do batch bandit policy evaluation. Consider we have a dataset for ~~p₁₁₁~~ from 3 actions. Consider that

- Action 1 is a Bernoulli var where with probability 0.02 r= 100 else r = 0
- Action 2 is a Bernoulli var where with probability 0.55 r=2 else r = 0
- Action 3 is a Bernoulli var where with probability 0.5 r=1, else r = 0

Select all that are true.

behavior policy

- Data is sampled from π_1 where with probability 0.8 it pulls action 3 else it pulls action 2. The policy we wish to evaluate, π_2 , pulls action 2 with probability 0.5 else it pulls action 1. π_2 has higher true reward than π_1 .
- We cannot use π_1 to get an unbiased estimate of the average reward π_2 using importance sampling.
- If rewards can be positive or negative, we can still get a lower bound on π_2 using data from π_1 using importance sampling
- Not Sure

Check Your Understanding: Importance Sampling

We can use importance sampling to do batch bandit policy evaluation. Consider we have a dataset for pulls from 3 actions. Consider that

- Action 1 is a Bernoulli var where with probability 0.02 r= 100 else r = 0 $E[r(a_1)] = 2 + 0 = 2$
- Action 2 is a Bernoulli var where with probability 0.55 r=2 else r = 0 $E[r(a_2)] = 5.5 \cdot 2 = 11$
- Action 3 is a Bernoulli var where with probability 0.5 r=1, else r = 0 $E[r(a_3)] = 5$

Select all that are true.

$$\pi_1, 0.8r(a_3) + 0.2r(a_2)$$

rwins, 0.8 \cdot 5 + 0.2 \cdot 11 \approx 12 \text{ish}

- Data is sampled from π_1 where with probability 0.8 it pulls action 3 else it pulls action 2. The policy we wish to evaluate, π_2 , pulls action 2 with probability 0.5 else it pulls action 1. π_2 has higher true reward than π_1 . $\pi_2, 0.5r(a_2) + 0.5r(a_1) = 0.5 \cdot 11 + 0.5 \cdot 2 \approx 1.5$ true
- We cannot use π_1 to get an unbiased estimate of the average reward π_2 using importance sampling.
- If rewards can be positive or negative, we can still get a lower bound on π_2 using data from π_1 using importance sampling \rightarrow if rewards ≥ 0 you can do this
- Not Sure

inverse propensity weighting

Importance Sampling for RL Policy Evaluation

$$\begin{aligned}
 V^\pi(s) &= \sum_{\tau} \underbrace{p(\tau | \pi, s)}_{\text{prob of traj}} R(\tau) \\
 &= \sum_{\gamma} p(\gamma | \pi, s) \frac{p(\gamma | \pi_b, s)}{p(\gamma | \pi_b, s)} R(\gamma) \\
 &= \sum_{\gamma} p(\gamma | \pi_b, s) \left[\frac{p(\gamma | \pi, s)}{p(\gamma | \pi_b, s)} R(\gamma) \right]
 \end{aligned}$$

\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$

π : Policy mapping $s \rightarrow a$

S_0 : Set of initial states

$\hat{V}^\pi(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

Importance Sampling for RL Policy Evaluation

$$\begin{aligned} V^\pi(s) &= \sum_{\tau} p(\tau|\pi, s) R(\tau) \\ &= \sum_{\tau} p(\tau|\pi_b, s) \frac{p(\tau|\pi, s)}{p(\tau|\pi_b, s)} R_\tau \\ &\approx \frac{1}{N} \sum_{i=1, \tau_i \sim \pi_b}^N \frac{p(\tau_i|\pi, s)}{p(\tau_i|\pi_b, s)} R_{\tau_i} \end{aligned}$$

$$p(\tau_i|\pi, s) = \frac{\prod_{t=1}^T p(s_{t+1}|s_t, a_t)}{\pi(a_t|s_t)}$$

=

\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$
 π : Policy mapping $s \rightarrow a$
 S_0 : Set of initial states
 $\hat{V}^\pi(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

Importance Sampling for RL Policy Evaluation: Don't Need to Know Dynamics Model!

$$\begin{aligned}
 V^\pi(s) &= \sum_{\tau} p(\tau|\pi, s) R(\tau) \\
 &= \sum_{\tau} p(\tau|\pi_b, s) \frac{p(\tau|\pi, s)}{p(\tau|\pi_b, s)} R_\tau \\
 &\approx \sum_{i=1, \tau_i \sim \pi_b}^N \frac{p(\tau_i|\pi, s)}{p(\tau_i|\pi_b, s)} R_{\tau_i} \\
 &= \sum_{i=1, \tau_i \sim \pi_b}^N R_{\tau_i} \prod_{t=1}^{H_i} \frac{p(s_{i,t+1}|s_{it}, a_{it}) p(a_{it}|\pi, s_{it})}{p(s_{i,t+1}|s_{it}, a_{it}) p(a_{it}|\pi_b, s_{it})} \\
 &= \sum_{i=1, \tau_i \sim \pi_b}^N R_{\tau_i} \prod_{t=1}^{H_i} \frac{p(a_{it}|\pi, s_{it})}{p(a_{it}|\pi_b, s_{it})}
 \end{aligned}$$

\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$
 π : Policy mapping $s \rightarrow a$
 S_0 : Set of initial states
 $\hat{V}^\pi(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

- First used for RL by Precup, Sutton & Singh 2000. Recent work includes: Thomas, Theocharous, Ghavamzadeh 2015; Thomas and Brunskill 2016; Guo, Thomas, Brunskill 2017; Hanna, Niekum, Stone 2019

Importance Sampling

- Does not rely on Markov assumption
- Requires minimal assumptions
- Provides unbiased estimator
- Similar to Monte Carlo estimator but corrects for distribution mismatch

Optional Check Your Understanding: Importance Sampling 2

Select all that you'd guess might be true about importance sampling

- It requires the behavior policy to visit all the state--action pairs that would be visited under the evaluation policy in order to get an unbiased estimator
- It is likely to be high variance
- Not Sure

Per Decision Importance Sampling (PDIS)

- Leverage temporal structure of the domain (**similar to policy gradient**)

$$IS(D) = \frac{1}{n} \sum_{i=1}^n \left(\prod_{t=1}^L \frac{\pi_e(a_t | s_t)}{\pi_b(a_t | s_t)} \right) \left(\sum_{t=1}^L \gamma^t R_t^i \right)$$

$$PSID(D) = \sum_{t=1}^L \gamma^t \frac{1}{n} \sum_{i=1}^n \left(\prod_{\tau=1}^t \frac{\pi_e(a_\tau | s_\tau)}{\pi_b(a_\tau | s_\tau)} \right) R_t^i$$

Importance Sampling Variance

- Importance sampling, like Monte Carlo estimation, is generally high variance
- Importance sampling is particularly high variance for estimating the return of a policy in a sequential decision process

$$= \sum_{i=1, \tau_i \sim \pi_b}^N R_{\tau_i} \prod_{t=1}^{H_i} \frac{p(a_{it}|\pi, s_{it})}{p(a_{it}|\pi_b, s_{it})}$$

- Variance can generally scale exponentially with the horizon
 - a. Concentration inequalities like Hoeffding scale with the largest range of the variable
 - b. The largest range of the variable depends on the product of importance weights
 - c. **Optional Check your understanding: for a H step horizon with a maximum reward in a single trajectory of 1, and if $p(a|s, \pi_b) = .1$ and $p(a|s, \pi) = 1$ for each time step, what is the maximum importance-weighted return for a single trajectory?**

$$R_{\tau_i} \prod_{t=1}^{H_i} \frac{p(a_{it}|\pi, s_{it})}{p(a_{it}|\pi_b, s_{it})}$$

Extensions

- Leveraging Markov structure to break curse of horizon.
 - Marginalized importance sampling (state-action distribution)
 - Dai, Nachum, Chow, Li (dualdice, coindice) 2019/2020
 - Liu, Li, Tang, Zhou Neurips 2018
- Doubly robust estimation (Jiang and Li 2016; Thomas and Brunskill 2016)
- Blended estimators (Thomas and Brunskill 2016)

Outline for Today

1. Introduction and Setting
2. Offline batch policy evaluation
 - a. Using models
 - b. Using model free methods
 - c. Use importance sampling
3. Offline policy learning / optimization

$$\arg \max_{\pi} \int_{s \in S_0} \hat{V}^{\pi}(s, \mathcal{D}) ds$$

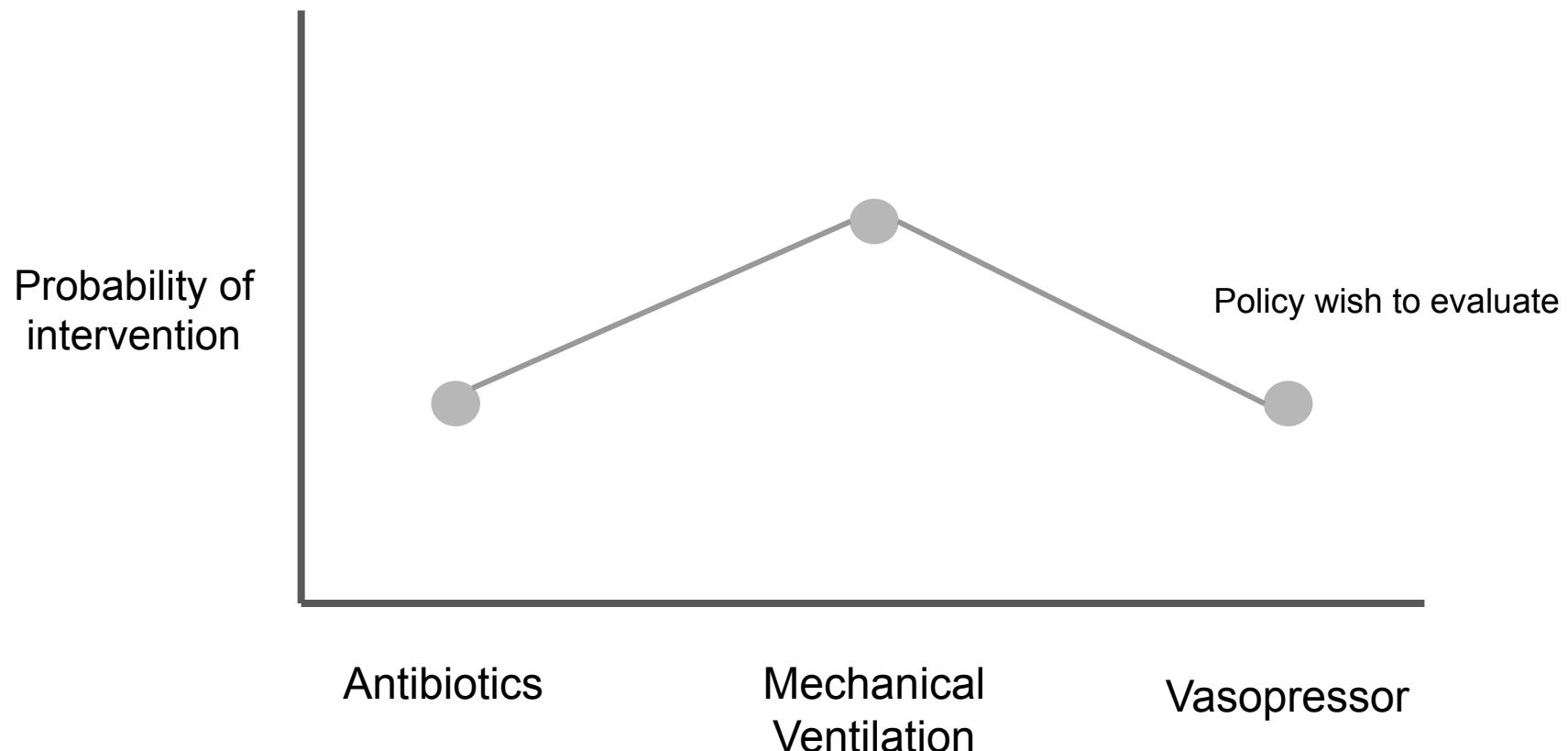
\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$

π : Policy mapping $s \rightarrow a$

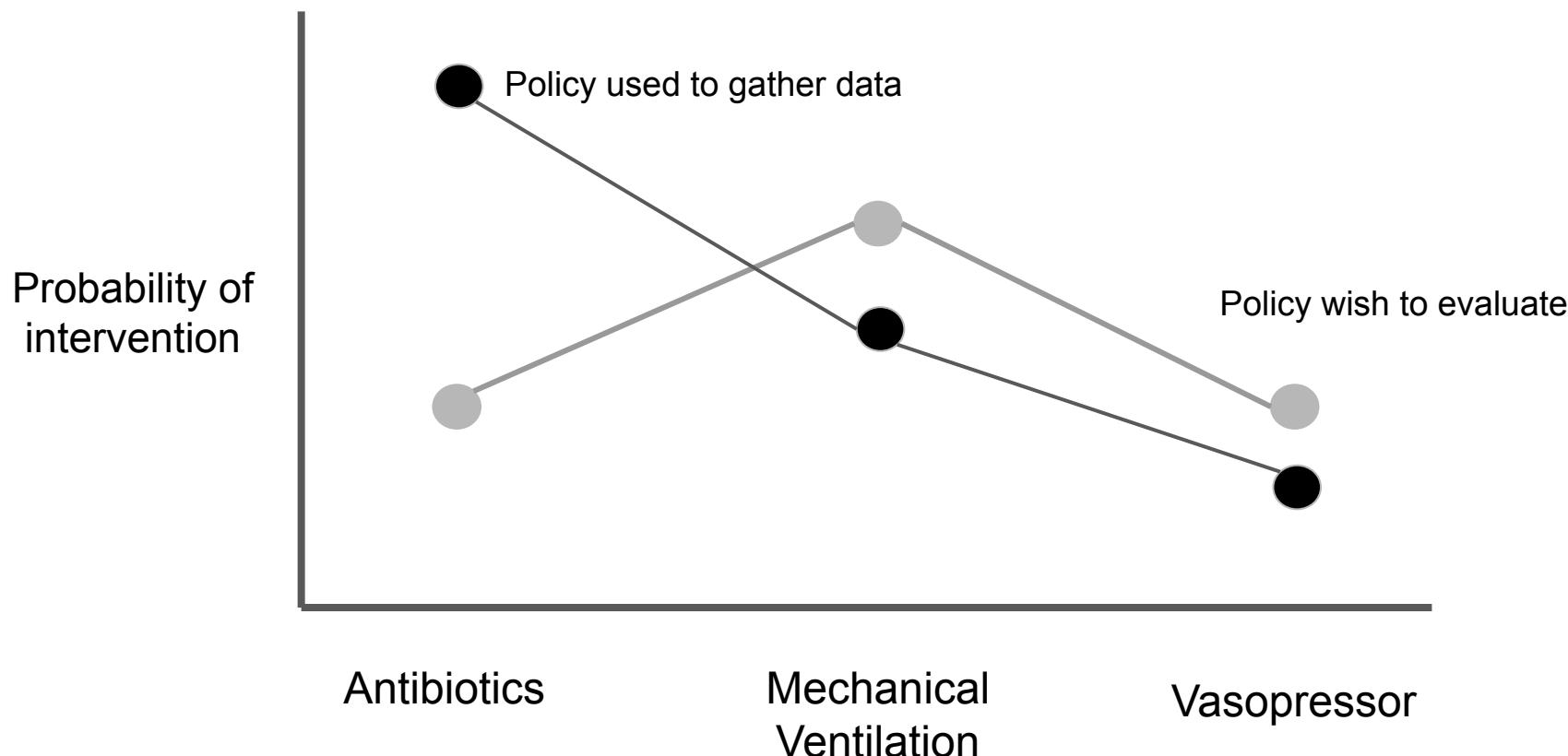
S_0 : Set of initial states

$\hat{V}^{\pi}(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

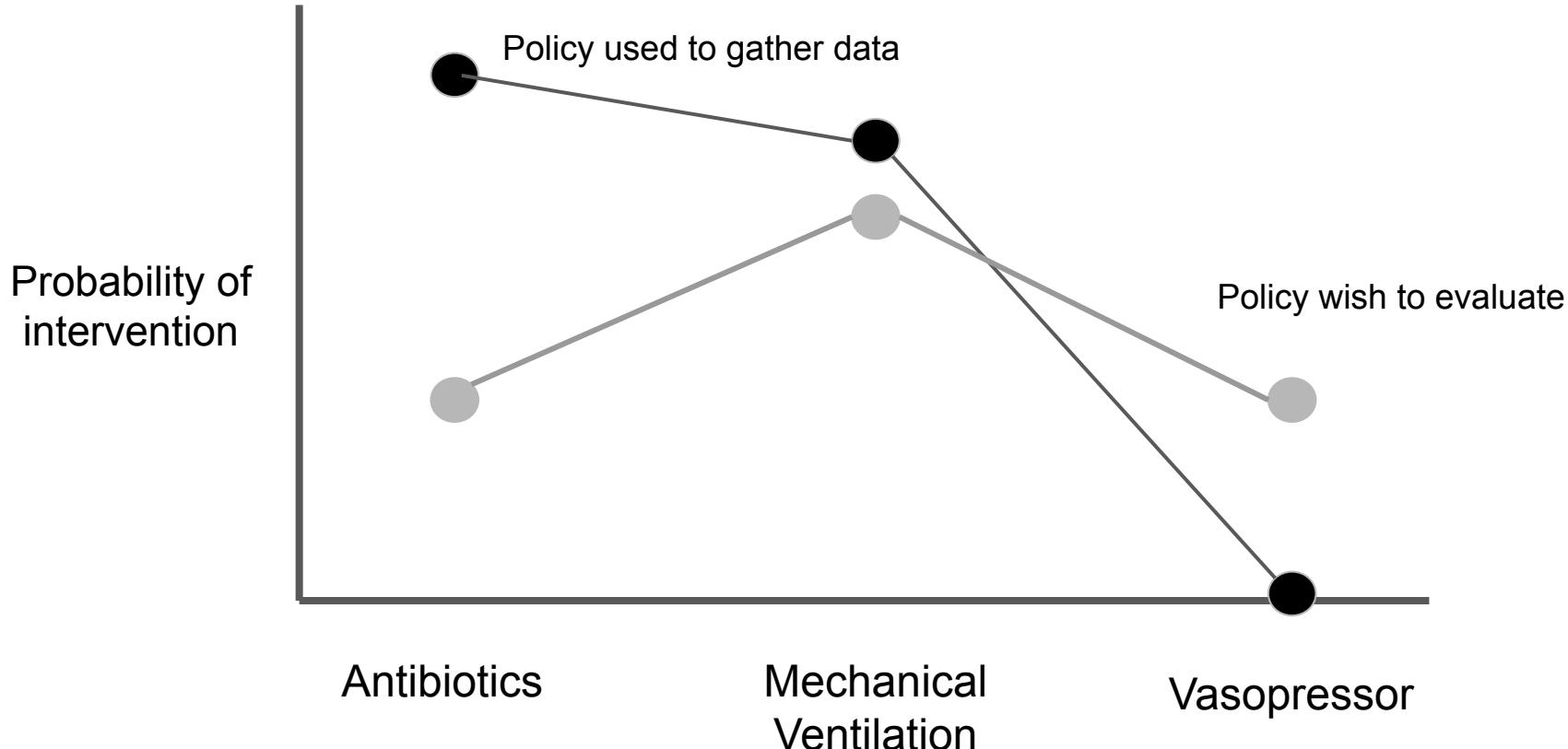
Challenges in Offline Policy Optimization



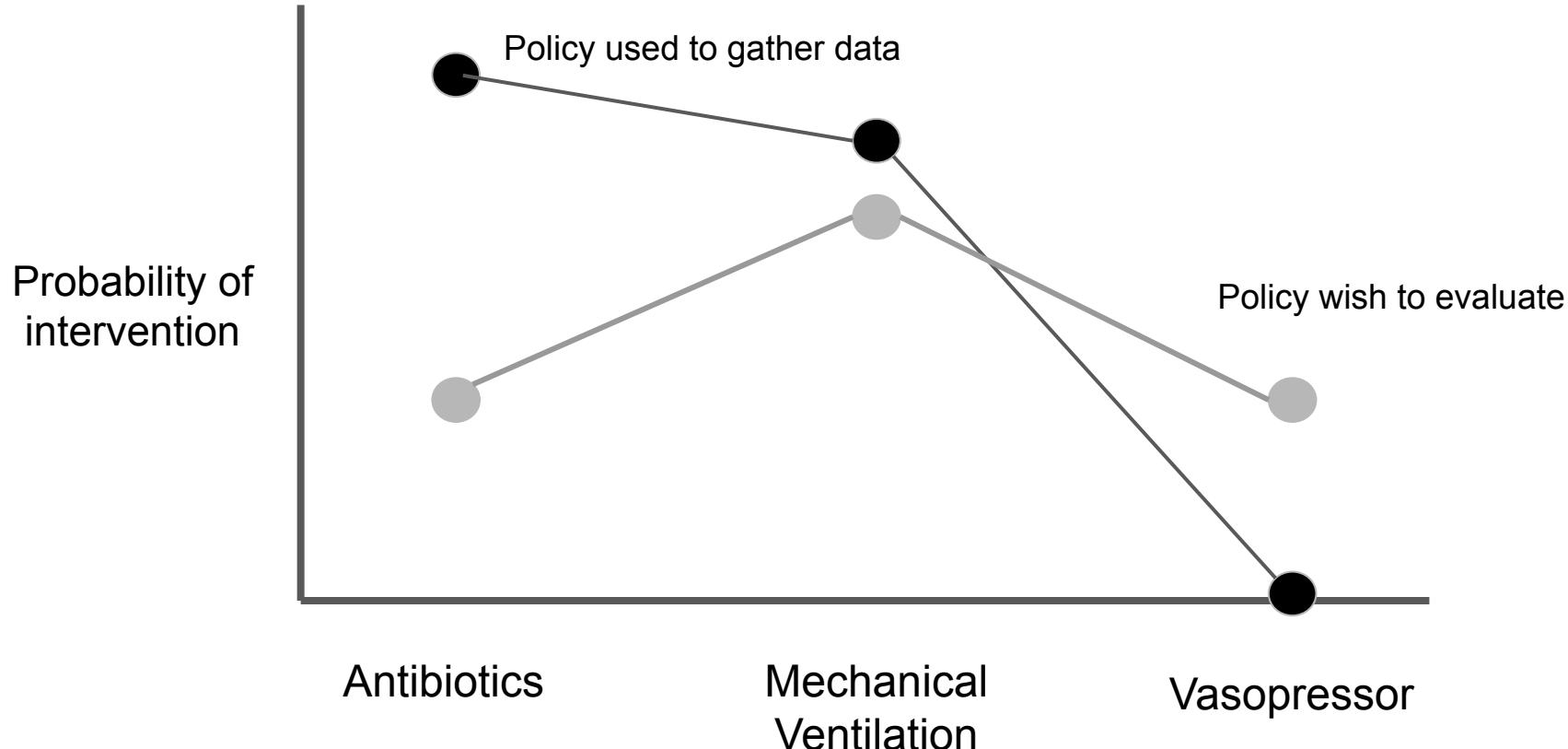
Overlap Requirement: Data Must Support Policy Wish to Evaluate



No Overlap for Vasopressor \Rightarrow Can't Do Off Policy Estimation for Desired Policy



Seen Data Distribution Shift Challenge Before. PPO. DPO. RLHF...



Offline Policy Optimization Up to ~ 2020

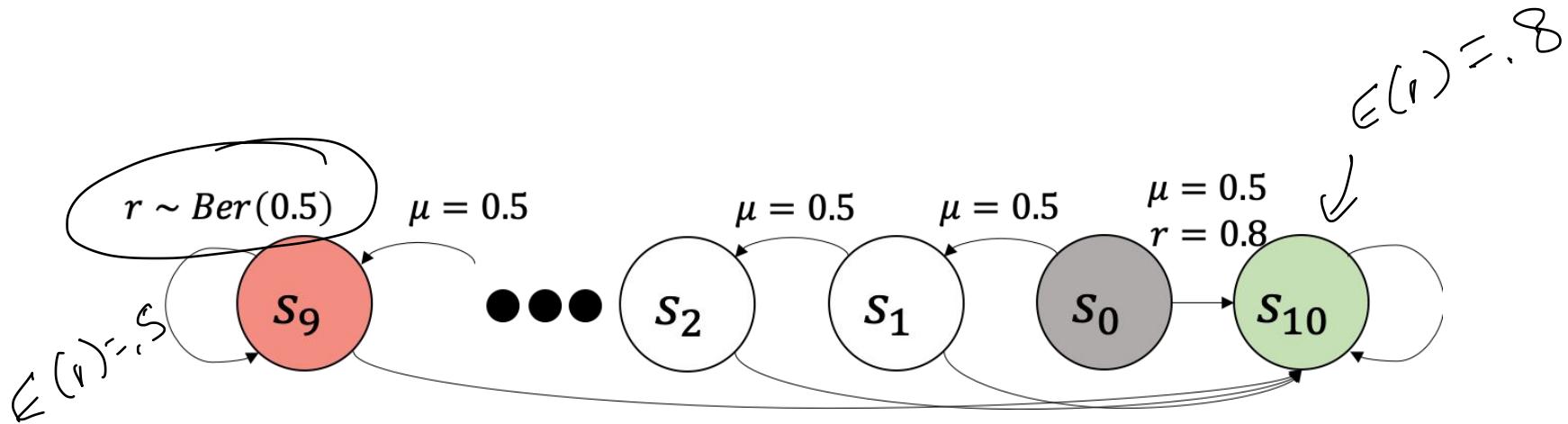
- Algorithms often assume overlap
 - Off policy estimation: for policy of interest
 - Off policy optimization: for all policies including optimal one (“concentrability” assumption in batch RL)
- Unlikely to be true in many settings
- Many real datasets don’t include complete random exploration
- Assuming overlap when it’s not there can be a problem:
 - We can end up with a policy with estimated high performance, but actually does poorly when deployed

Doing the Best with What We've Got: Off Policy Optimization Without Full Data Coverage

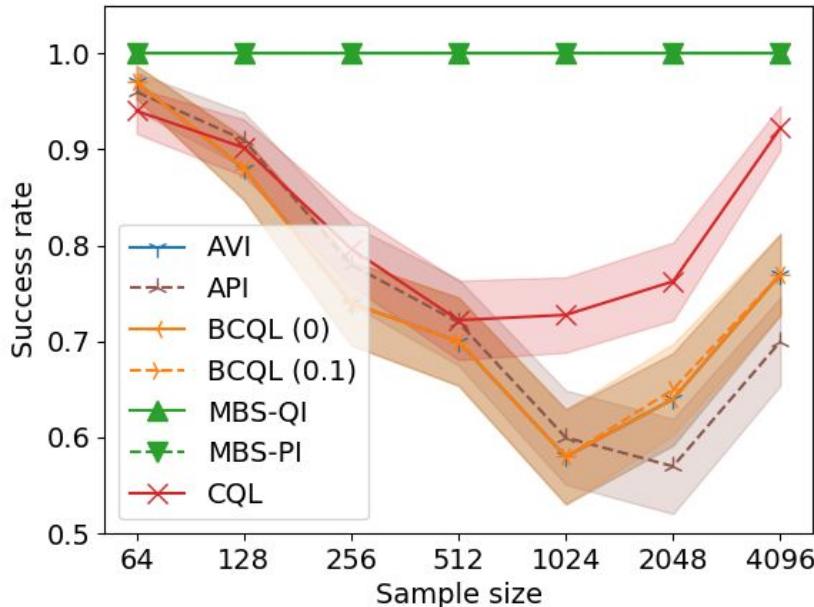
- Restrict off policy optimization to those with overlap in data
 - We've seen related ideas before: KL constraint or PPO clipping
- Computationally tractable algorithm
- Simple idea: assume **pessimistic outcomes** for areas of state--action space with insufficient overlap/support

Common challenge that's attracted growing interest before 2020 but...

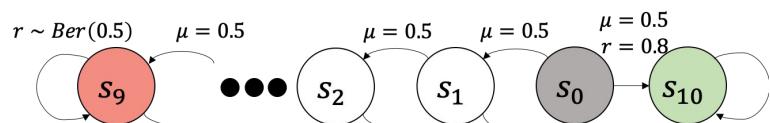
Illustrative Examples



Recent Conservative Batch Reinforcement Learning Are Insufficient



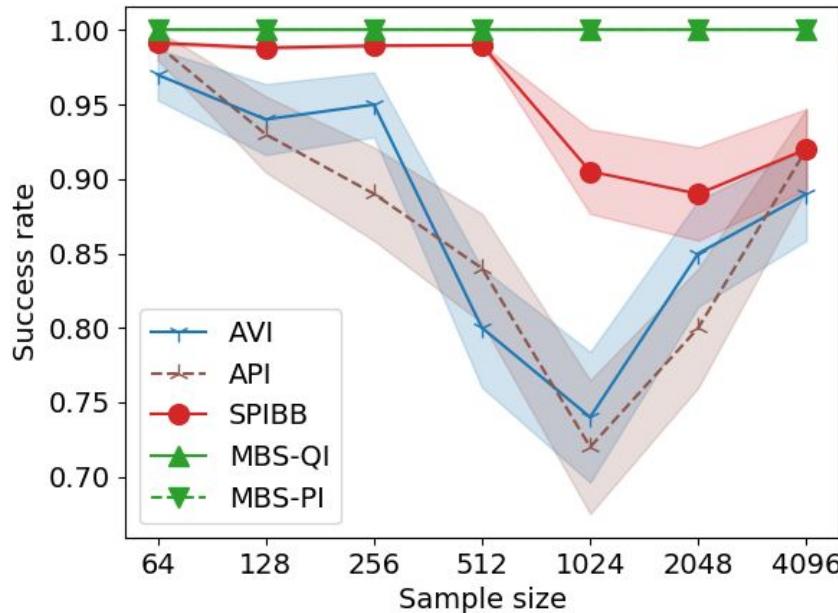
Success rate: #(getting the optimal policy)/#(trials)



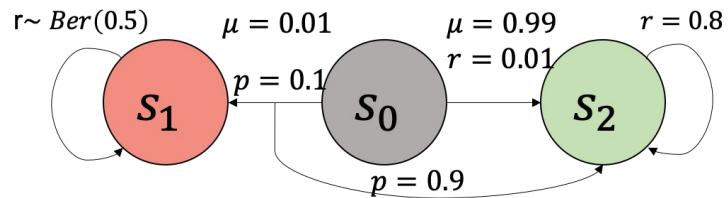
Reasons why baselines fail:

- Many baselines focus on penalty/constraints that are based on $\text{dist}(\pi(a|s), \pi_b(a|s))$.
- In this example a sequence of large action conditional probabilities leads to a rare state.
- Due to finite samples, estimates of the reward of this rare state can be overestimated.

Recent Conservative Batch Reinforcement Learning Are Insufficient



Success rate: #(getting the optimal policy)/#(trials)



Reasons why baselines fail:

- SPIBB adds conservatism based on estimates of π_b & V of π_b .
- In this example, the actions which are rare under π_b also have a stochastic transition and reward, thus the π_b 's V is overestimated.

Idea: Use pessimistic value for state-action space with insufficient data

- Filtration function:

$$\zeta(s, a; \hat{\mu}, b) = 1(\hat{\mu}(s, a) > b)$$

Idea: Use pessimistic value for state-action space with insufficient data

- Filtration function:

$$\zeta(s, a; \hat{\mu}, b) = 1(\hat{\mu}(s, a) > b)$$

b can account for statistical uncertainty due to finite samples

Idea: Use pessimistic value for state-action space with insufficient data

- Filtration function:

$$\zeta(s, a; \hat{\mu}, b) = 1(\hat{\mu}(s, a) > b)$$

- Bellman operator and Bellman evaluation operator:

$$\mathcal{T}f(s, a) = r(s, a) + \gamma \mathbb{E}_{s'} \left[\max_{a'} \zeta(s', a') f(s', a') \right]$$

Idea: Use pessimistic value for state-action space with insufficient data

- Filtration function:

$$\zeta(s, a; \hat{\mu}, b) = 1(\hat{\mu}(s, a) > b)$$

- Bellman operator and Bellman evaluation operator:

$$Tf(s, a) = r(s, a) + \gamma \mathbb{E}_{s'} \left[\max_{a'} \zeta(s', a') f(s', a') \right]$$

$\Rightarrow = 0$ for (s', a') with insufficient data.

We assume $r(s, a) \geq 0$

Therefore pessimistic estimate for such tuples

Idea: Use pessimistic value for state-action space with insufficient data

- Filtration function:

$$\zeta(s, a; \hat{\mu}, b) = 1(\hat{\mu}(s, a) > b)$$

- Bellman operator and Bellman evaluation operator:

$$\mathcal{T}f(s, a) = r(s, a) + \gamma \mathbb{E}_{s'} \left[\max_{a'} \zeta(s', a') f(s', a') \right]$$

$$\mathcal{T}^\pi f(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P, a' \sim \pi} [\zeta(s', a') f(s', a')]$$

Marginalized Behavior Supported (MBI) Policy Optimization

- Filtration function:

$$\zeta(s, a; \hat{\mu}, b) = 1(\hat{\mu}(s, a) > b)$$

- Bellman operator and Bellman evaluation operator:

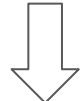
$$\mathcal{T}f(s, a) = r(s, a) + \gamma \mathbb{E}_{s'} \left[\max_{a'} \zeta(s', a') f(s', a') \right]$$

$$\mathcal{T}^\pi f(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P, a' \sim \pi} [\zeta(s', a') f(s', a')]$$

Majority of Past Model-Free Batch RL Theory for Function Approximation Setting

Assume for any $\nu(s,a)$ distribution possible
under some policy in this MDP

$$\forall (s, a) \in \mathcal{S} \times \mathcal{A}, \frac{\nu(s, a)}{\mu(s, a)} \leq C.$$

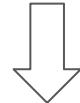


$$V^* - V^{\pi_A} \leq \epsilon$$

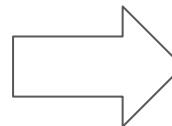
Best in Well Supported Policy Class*

Assume for any $\nu(s,a)$ distribution possible under some policy in this MDP

$$\forall (s, a) \in \mathcal{S} \times \mathcal{A}, \frac{\nu(s, a)}{\mu(s, a)} \leq C.$$



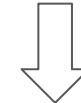
$$V^* - V^{\pi_A} \leq \epsilon$$



Define

$$\Pi_{all} : \pi \text{ s.t.}$$

$$\mathbb{E}_{s, a \sim \eta^\pi} [\mathbb{1} (\zeta(s, a) = 0)] \leq \epsilon_\zeta$$



$$\max_{\pi' \in \Pi_{all}} V^{\pi'} - V^{\pi_A} \leq \epsilon$$

*Note: Policy set Π_{all} is not constructed, but implicitly our algorithm only considers elements in it

Assumption 1 (Bounded densities). *For any non-stationary policy π and $h \geq 0$, $\eta_h^\pi(s, a) \leq U$.*

Assumption 2 (Density estimation error). *With probability at least $1 - \delta$, $\|\hat{\mu} - \mu\|_{TV} \leq \epsilon_\mu$.*

Assumption 3 (Completeness under $\tilde{\mathcal{T}}^\pi$). $\forall \pi \in \Pi$, $\max_{f \in \mathcal{F}} \min_{g \in \mathcal{F}} \|g - \tilde{\mathcal{T}}^\pi f\|_{2,\mu}^2 \leq \epsilon_{\mathcal{F}}$.

Assumption 4 (Π Completeness). $\forall f \in \mathcal{F}$, $\min_{\pi \in \Pi} \|\mathbb{E}_\pi [\zeta \circ f(s, a)] - \max_a \zeta \circ f(s, a)\|_{1,\mu} \leq \epsilon_\Pi$.

$$\boxed{\begin{aligned}\eta_h^\pi(s) &:= \Pr[s_h = s | \pi], \\ \eta_h^\pi(s, a) &= \eta_h^\pi(s) \pi(a | s)\end{aligned}}$$

$$\zeta(s, a; \hat{\mu}, b) = \mathbb{1}(\hat{\mu}(s, a) \geq b)$$

Theoretical Result

We bound the error w.r.t. the best policy in the following policy set:
{all policies such that $\Pr(\zeta(s, a) = 0 | \pi) \leq \epsilon_\zeta$ }

Error bounds¹:

- PI:

$$O\left(\frac{V_{\max}}{(1-\gamma)^3 b} \sqrt{\frac{\ln(|\mathcal{F}| |\Pi| / \delta)}{n}}\right) + \frac{V_{\max} \epsilon_\zeta}{1-\gamma}$$

- VI²:

$$O\left(\frac{V_{\max}}{(1-\gamma)^2 b} \sqrt{\frac{\ln(|\mathcal{F}| / \delta)}{n}}\right) + \frac{V_{\max} \epsilon_\zeta}{1-\gamma}$$

1: We omit some constant terms that is same as standard ADP analysis with function approximation.

2: For VI results there is another important constant term, see our paper for detailed result and discussion.

$$\zeta(s, a; \hat{\mu}, b) = \mathbb{1}(\hat{\mu}(s, a) \geq b)$$

Theoretical Result

We bound the error w.r.t. the best policy in the following policy set:

$$\{\text{all policies such that } \Pr(\zeta(s, a) = 0 | \pi) \leq \epsilon_\zeta\}$$

Error bounds¹:

- PI:

$$O\left(\frac{V_{\max}}{(1-\gamma)^3 b} \sqrt{\frac{\ln(|\mathcal{F}| |\Pi| / \delta)}{n}}\right) + \frac{V_{\max} \epsilon_\zeta}{1-\gamma}$$

- VI²:

$$O\left(\frac{V_{\max}}{(1-\gamma)^2 b} \sqrt{\frac{\ln(|\mathcal{F}| / \delta)}{n}}\right) + \frac{V_{\max} \epsilon_\zeta}{1-\gamma}$$

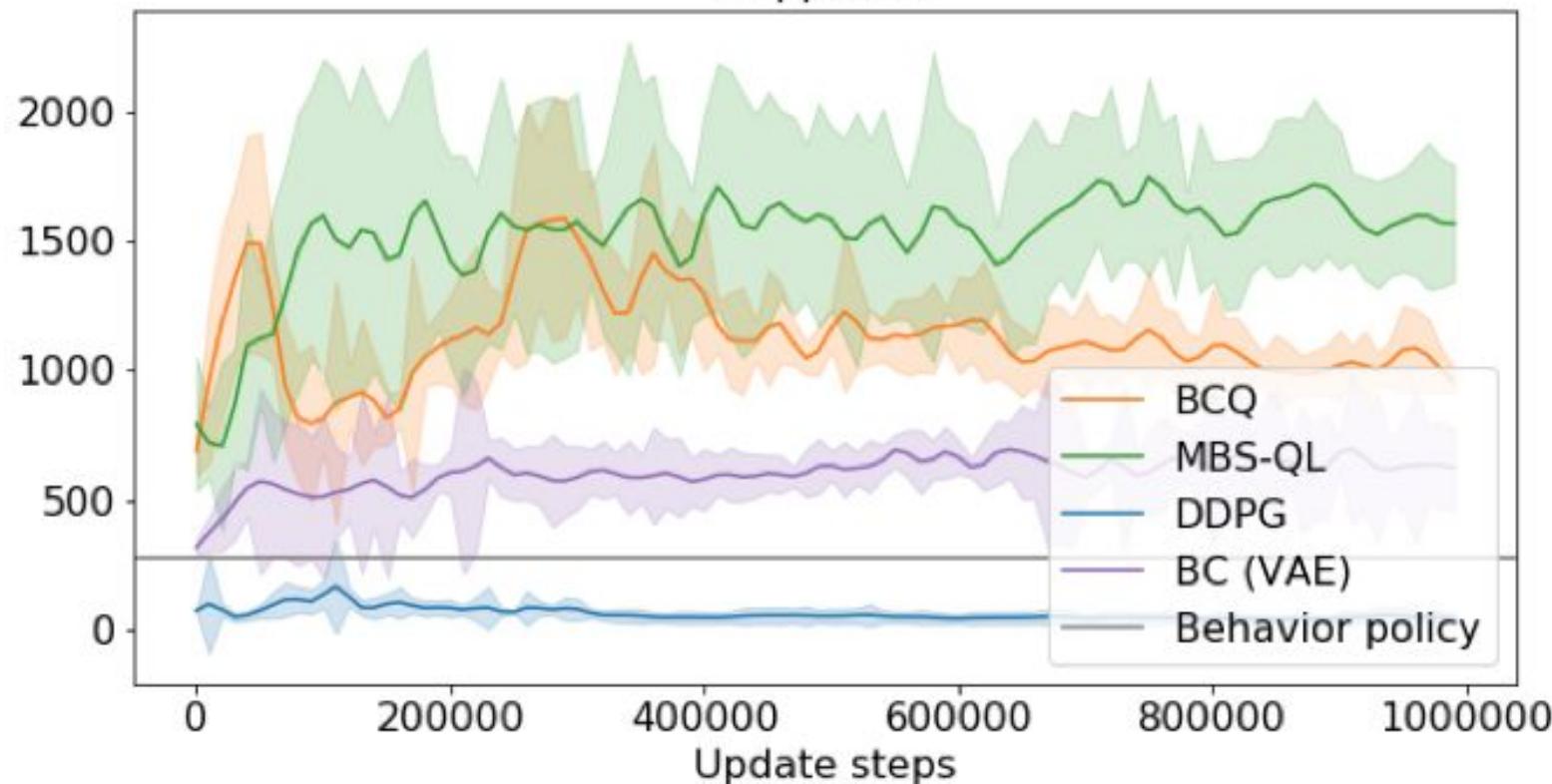
Note: Results are for function approximation, finite sample setting

1: We omit some constant terms that is same as standard ADP analysis with function approximation.

2: For VI results there is another important constant term, see our paper for detailed result and discussion.

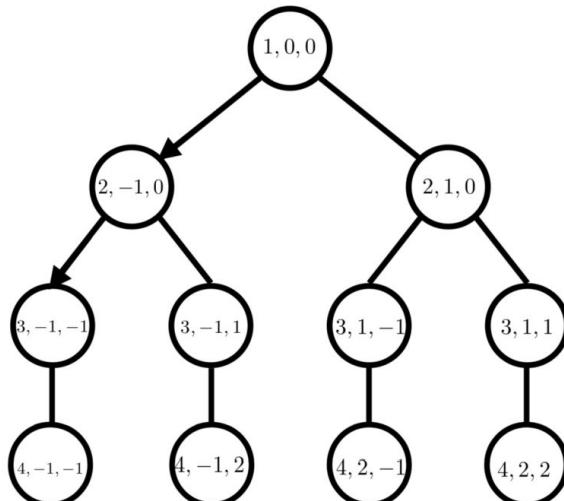
Can Do Get Substantially Better Solutions, With Same Data

Hopper-v3



This Was Model Free. Might Models Be Even Better?

- Model based approaches can be provably more efficient than model free value function for *online* evaluation or control



Sun, Jiang, Krishnamurthy,
Agarwal, Langford COLT 2019

$$x_{t+1} = A_\star x_t + B_\star u_t + w_t ,$$

$$V^K(x) := \lim_{T \rightarrow \infty} \mathbb{E} \left[\sum_{t=0}^{T-1} (x_t^\top Q x_t + u_t^\top R u_t - \lambda_K) \mid x_0 = x \right]$$

Tu & Recht COLT 2019

Concurrent Work Conservative Model-Based Offline RL

- Yu, Thomas, Yu, Ermon, Zou, Levine, Finn & Ma (NeurIPS 2020)
- Kidambi, Rajeswaran, Netrapalli & Joachims (NeurIPS 2020)
- **Learn a model and penalize model uncertainty** during planning
- Empirically very promising on D4RL tasks
- Their work has more limited theoretical analysis

\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$

π : Policy mapping $s \rightarrow a$

S_0 : Set of initial states

$\hat{V}^\pi(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

Concurrent Work Conservative Offline RL

- Yu, Thomas, Yu, Ermon, Zou, Levine, Finn & Ma (NeurIPS 2020)
- Kidambi, Rajeswaran, Netrapalli & Joachims (NeurIPS 2020)
- **Learn a model and penalize model uncertainty** during planning
- Empirically very promising on D4RL tasks
- Their work has more limited theoretical analysis
- **Conservative Q Learning (CQL) (Kumar et al.) continues to be popular**

\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$

π : Policy mapping $s \rightarrow a$

S_0 : Set of initial states

$\hat{V}^\pi(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

Early Comparison with Concurrent Work

	MBS-BCQ	MBS-BEAR	BCQ	BEAR	MOPO	CQL
Hopper-medium	75.9	32.3	54.5	52.1	26.5	58.0
HalfCheetah-medium	38.4	39.7	40.7	41.7	40.2	44.4
Walker2d-medium	64.4	75.4	53.1	59.1	14.0	79.2

Comparison with Concurrent Work

	MBS-BCQ	MBS-BEAR	BCQ	BEAR	MOPO	CQL
Hopper-medium	75.9	32.3	54.5	52.1	26.5	58.0
HalfCheetah-medium	38.4	39.7	40.7	41.7	40.2	44.4
Walker2d-medium	64.4	75.4	53.1	59.1	14.0	79.2

- Pessimistic approaches do quite well, different methods win in different areas
- MBS has stronger theory results

Pessimistic Offline Policy Learning

- Restrict off policy optimization to those with overlap in data
- Simple idea: assume pessimistic outcomes for areas of state--action space with insufficient overlap/support
 - In model
 - In Q function

*PPD constrained
update ⊆*

Outline for Today

1. Introduction and Setting
2. Offline batch policy evaluation
 - a. Using models
 - b. Using model free methods
 - c. Use importance sampling
3. Offline policy learning / optimization

$$\arg \max_{\pi} \int_{s \in S_0} \hat{V}^{\pi}(s, \mathcal{D}) ds$$

\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$

π : Policy mapping $s \rightarrow a$

S_0 : Set of initial states

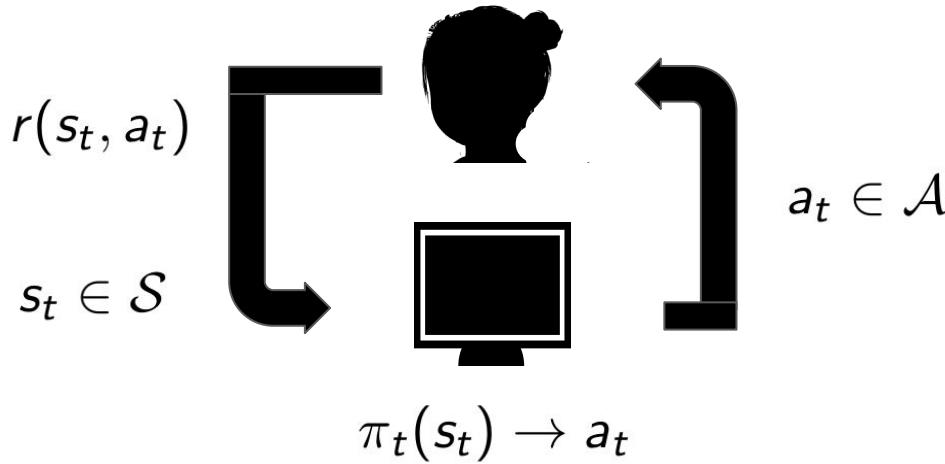
$\hat{V}^{\pi}(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

Optimizing while Ensuring Solution Won't, in the Future, Exhibit Undesirable Behavior

$$\begin{aligned} & \arg \max_{a \in \mathcal{A}} f(a) \\ \text{s.t. } & \text{s.t. } \forall i \in \{1, \dots, n\}, \Pr(g_i(a(D)) \leq 0) \geq 1 - \delta_i \end{aligned}$$

Constraints

Offline RL with Constraints on Future Performance of Policy



\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$

An Algorithm for Offline RL with Safety Constraints

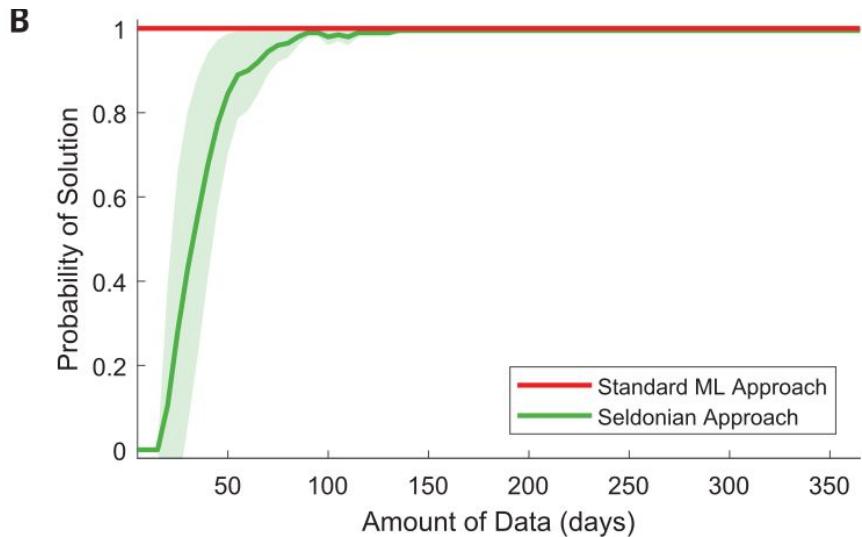
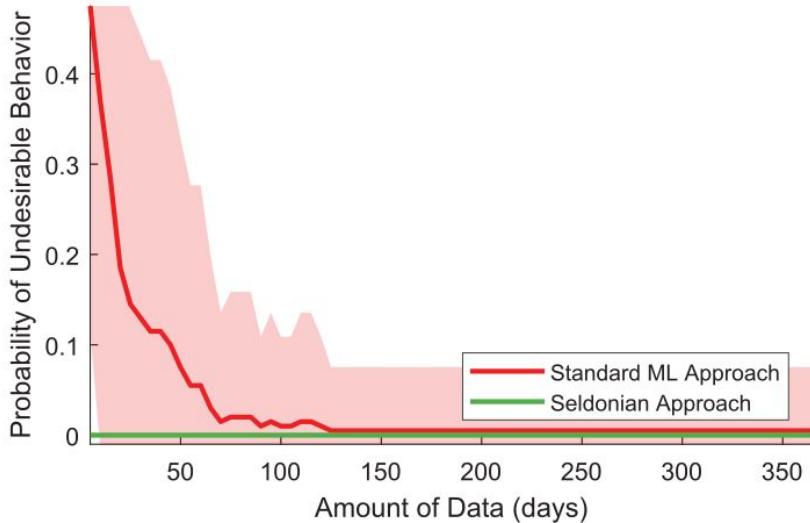
- Take in desired behavior constraints g and confidence level & data
- Given a finite set of decision policies, for each policy i
 - Compute generalization bound for each constraint
 - If passes all with desired confidence*, $\text{Safe}(i) = \text{true}$
- Estimate performance f of all policies that are safe
- Return best policy that is safe, or no solution if safe set is empty

Diabetes Insulin Management

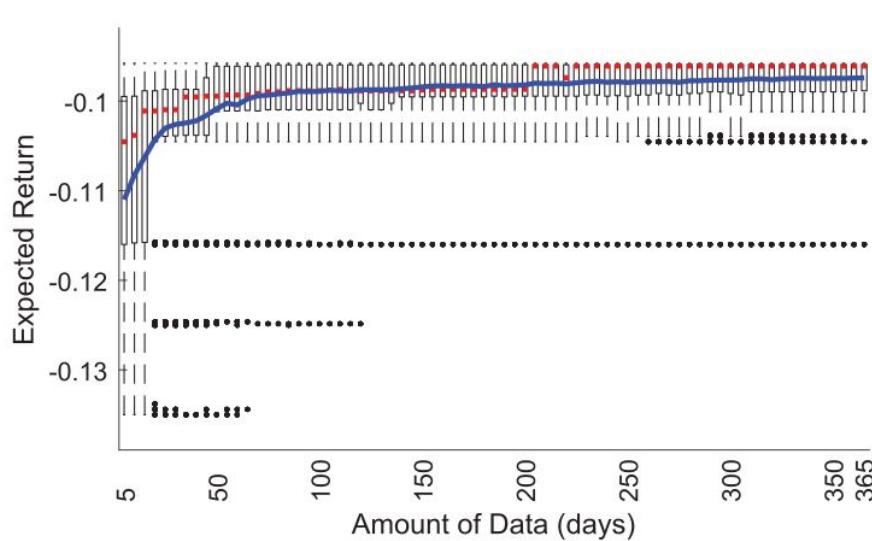


- Blood glucose control
- Action: insulin dosage
- Search over policies
- Constraint:
hypoglycemia
- Very accurate simulator:
**approved by FDA to
replace early stage
animal trials**

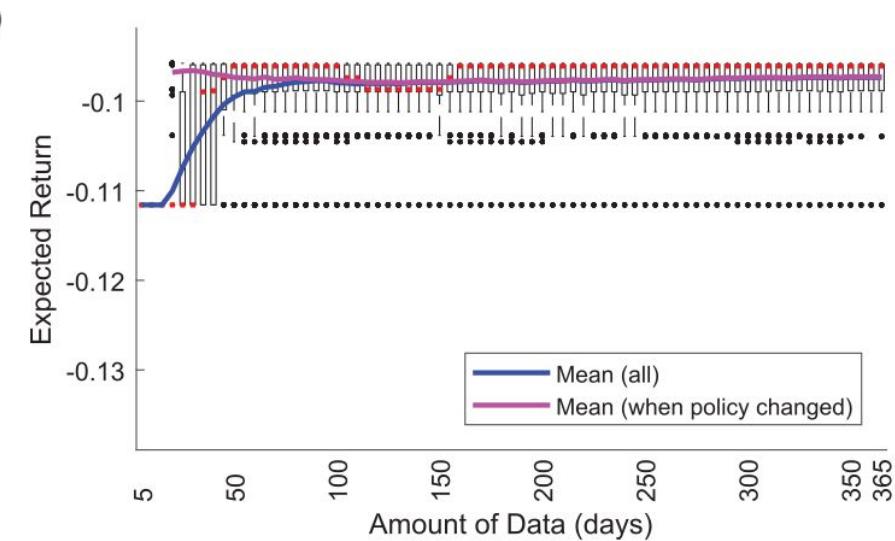
Personalized Insulin Dosage: Safe Batch Policy Improvement



Personalized Insulin Dosage: Quickly Can Have Confidence in Safe Better Policy



Standard RL



Our Safe Batch RL

Outline for Today

1. Introduction and Setting
2. Offline batch policy evaluation
 - a. Using models
 - b. Using model free methods
 - c. Use importance sampling
3. Offline policy learning / optimization

$$\arg \max_{\pi} \int_{s \in S_0} \hat{V}^{\pi}(s, \mathcal{D}) ds$$

\mathcal{D} : Dataset of n traj.s τ , $\tau \sim \pi_b$

π : Policy mapping $s \rightarrow a$

S_0 : Set of initial states

$\hat{V}^{\pi}(s, \mathcal{D})$: Estimate $V(s)$ w/dataset \mathcal{D}

What You Should Know/ Be Able to Do

- Define and apply importance sampling for off policy policy evaluation
- Describe limitations of model and model free off policy evaluation
- Define some limitations of IS (variance)
- Explain when and why offline RL may outperform imitation learning
- Describe the idea of pessimism under uncertainty and why it is useful
- Provide application examples where offline RL and offline policy evaluation would be useful