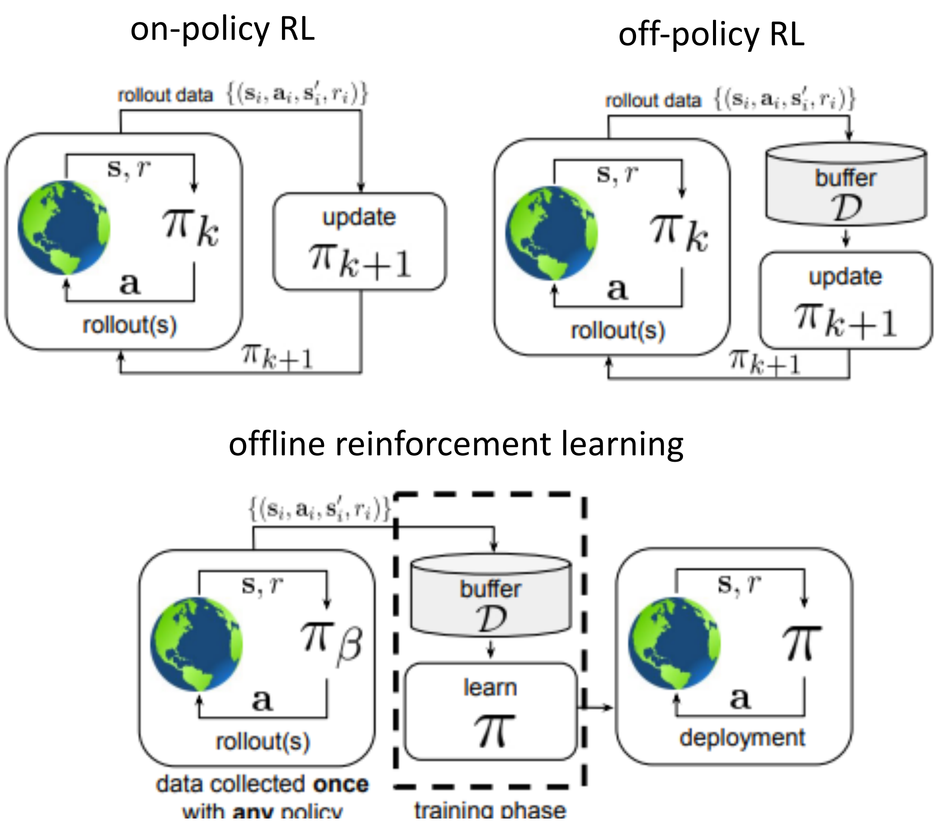
# Batch / Offline Reinforcement Learning

With thanks to Emma Brunskill, Scott Fujimoto, Pieter Abeell, George Tucker, Sergey Levine, Bilal Piot, Yuxin Chen, Yuejie Chi

COMP579 Lecture 20, 2025

# On-policy vs off-policy vs offline RL




Formally:

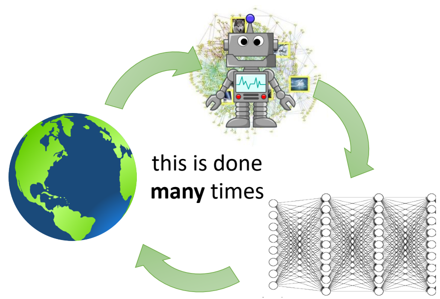
$$\mathcal{D} = \{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)\}$$

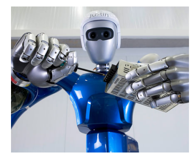
$$\mathbf{s} \sim d^{\pi_\beta}(\mathbf{s})$$
$$\mathbf{a} \sim \pi_\beta(\mathbf{a}|\mathbf{s})$$ generally **not** known
$$\mathbf{s}' \sim p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$$
$$r \leftarrow r(\mathbf{s}, \mathbf{a})$$

RL objective: $\max_\pi \sum_{t=0}^{T} E_{\mathbf{s}_t \sim d^\pi(\mathbf{s}), \mathbf{a}_t \sim \pi(\mathbf{a}|\mathbf{s})}[\gamma^t r(\mathbf{s}_t, \mathbf{a}_t)]$

# Why is this important?

- Collecting new data may be expensive / infeasible
- We may have access to existing/historical data instead




# Problem formulation

**A historical dataset** $\mathcal{D} = \left\{(s^{(i)}, a^{(i)}, s'^{(i)})\right\}$**:** $N$ independent copies of

$$s \sim \rho^{\mathsf{b}}, \qquad a \sim \pi^{\mathsf{b}}(\cdot \,|\, s), \qquad s' \sim P(\cdot \,|\, s, a)$$

for some state distribution $\rho^{\mathsf{b}}$ and behavior policy $\pi^{\mathsf{b}}$

**Goal:** given some test distribution $\rho$ and accuracy level $\varepsilon$, find an $\varepsilon$-optimal policy $\widehat{\pi}$ based on $\mathcal{D}$ obeying

$$V^{\star}(\rho) - V^{\widehat{\pi}}(\rho) = \mathop{\mathbb{E}}_{s\sim\rho}\big[V^{\star}(s)\big] - \mathop{\mathbb{E}}_{s\sim\rho}\big[V^{\widehat{\pi}}(s)\big] \le \varepsilon$$

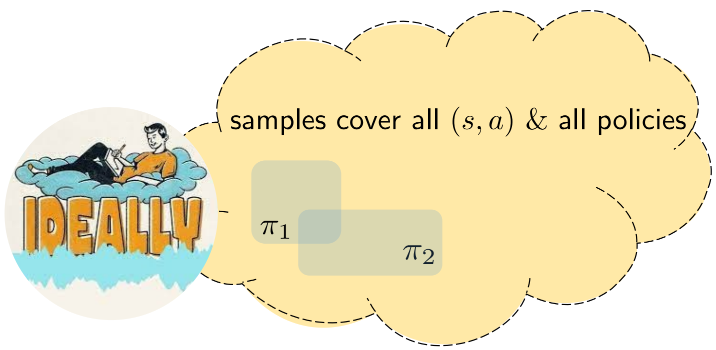
— *in a sample-efficient manner*

# Challenges of offline / batch RL (1)

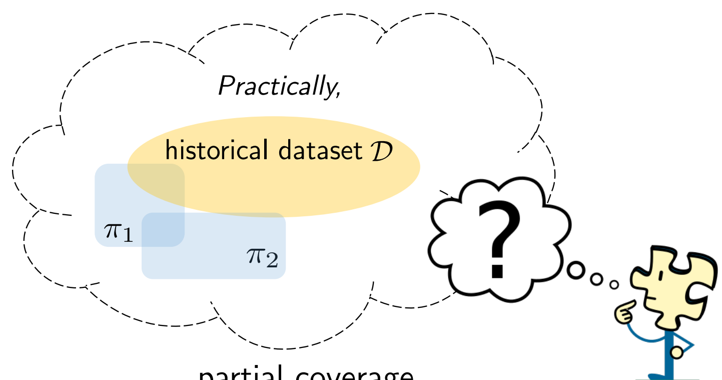
- **Distribution shift**:

$$\text{distribution}(\mathcal{D}) \neq \text{target distribution under } \pi^{\star}$$

- **Partial coverage of state-action space**:



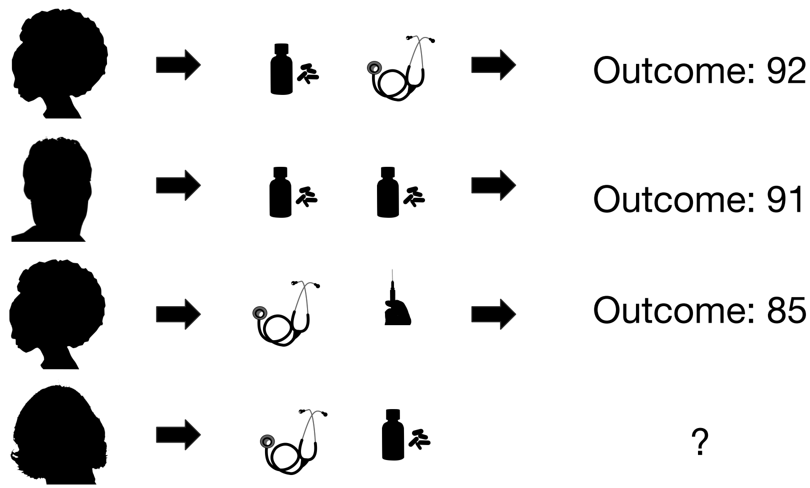

uniform coverage over entire space
(sufficiently explored)




partial coverage
(inadequately explored)

# Challenges of offline / batch RL (2)

- Data is *censored*: we only observe outcomes for decisions made (and need to generalize from them)




- Need for *counterfactual inference*: what would happen if one would take a different action?
- Often we do not observe rewards, just states and actions!

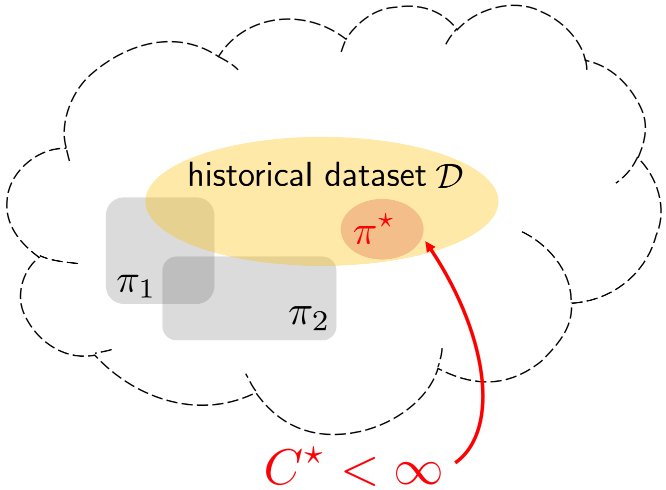
# Dataset quality assessment

**Single-policy concentrability coefficient**

$$C^\star := \max_{s,a} \frac{d^{\pi^\star}(s,a)}{d^{\pi^\mathsf{b}}(s,a)} = \left\| \frac{\textit{occupancy density of } \pi^\star}{\textit{occupancy density of } \pi^\mathsf{b}} \right\|_\infty \geq 1$$

*where* $d^\pi(s,a) = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \mathbb{P}\big((s^t, a^t) = (s,a) \,|\, \pi\big)$

- captures distributional shift
- allows for partial coverage

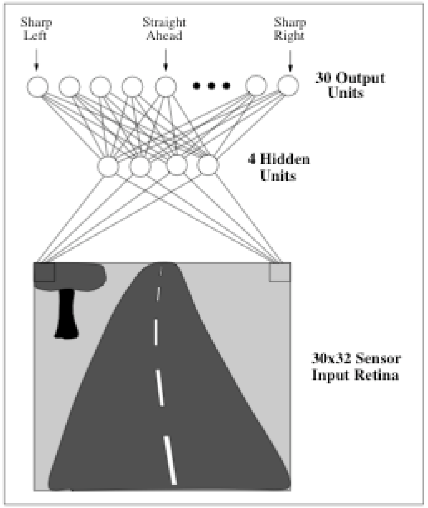


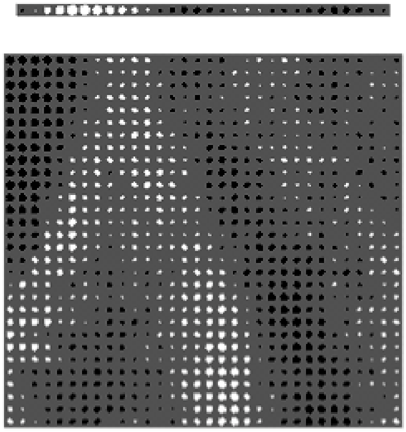
# Classes of algorithms

1. Behavior cloning (no rewards required)
2. Learn a model, use it for model-based RL (LSTD, LSPI)
3. Pessimistic algorithms (require rewards)
4. Inverse RL (learn reward function from data, use it for RL agent)

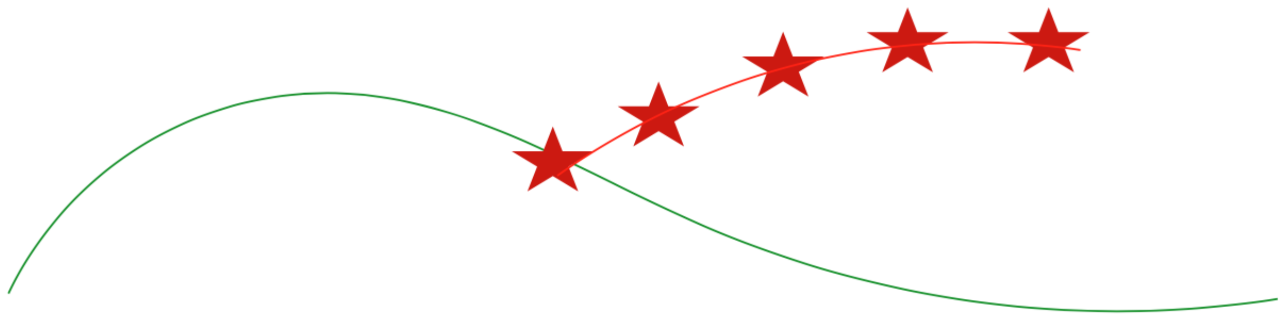
# Part 1: Behavior cloning

- Take dataset $\mathcal{D}$, learn a policy from states to actions
- Often uses a rich policy class (neural net)



# Problem: compounding errors



- Error at time $t$ with probability $\epsilon$
- Approximate intuition: $\mathbb{E}[\text{Total errors}]$
  $\leq \epsilon(T + (T-1) + (T-2) \ldots + 1) \propto \epsilon T^2$

# One solution: dataset aggregation

Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in $\Pi$.
**for** $i = 1$ **to** $N$ **do**
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
  Sample $T$-step trajectories using $\pi_i$.
  Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by $\pi_i$ and actions given by expert.
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
  Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
**end for**
**Return** best $\hat{\pi}_i$ on validation.

- Idea: Get more labels of the expert action along the path taken by the policy computed by behavior cloning
- Obtains a stationary deterministic policy with good performance under its induced state distribution

# Least-squares regression

- Given value function approximation $\hat{v}(s, \mathbf{w}) \approx v_\pi(s)$
- And *experience* $\mathcal{D}$ consisting of $\langle state, value \rangle$ pairs

$$\mathcal{D} = \left\{ \langle s_1, v_1^\pi \rangle, \langle s_2, v_2^\pi \rangle, ..., \langle s_T, v_T^\pi \rangle \right\}$$

- Which parameters $\mathbf{w}$ give the *best fitting* value fn $\hat{v}(s, \mathbf{w})$?
- Least squares algorithms find parameter vector $\mathbf{w}$ minimising sum-squared error between $\hat{v}(s_t, \mathbf{w})$ and target values $v_t^\pi$,

$$\begin{aligned} LS(\mathbf{w}) &= \sum_{t=1}^{T} (v_t^\pi - \hat{v}(s_t, \mathbf{w}))^2 \\ &= \mathbb{E}_\mathcal{D} \left[ (v^\pi - \hat{v}(s, \mathbf{w}))^2 \right] \end{aligned}$$

# Part 2: Least-squares regression

- At minimum of $LS(\mathbf{w})$, the expected update must be zero

$$\mathbb{E}_{\mathcal{D}}\left[\Delta\mathbf{w}\right] = 0$$

$$\alpha \sum_{t=1}^{T} \mathbf{x}(s_t)(v_t^\pi - \mathbf{x}(s_t)^\top \mathbf{w}) = 0$$

$$\sum_{t=1}^{T} \mathbf{x}(s_t) v_t^\pi = \sum_{t=1}^{T} \mathbf{x}(s_t)\mathbf{x}(s_t)^\top \mathbf{w}$$

$$\mathbf{w} = \left(\sum_{t=1}^{T} \mathbf{x}(s_t)\mathbf{x}(s_t)^\top\right)^{-1} \sum_{t=1}^{T} \mathbf{x}(s_t) v_t^\pi$$

- For $N$ features, direct solution time is $O(N^3)$
- Incremental solution time is $O(N^2)$ using Shermann-Morrison

# Model-based solution: Least-squares algorithms

- We do not know true values $v_t^\pi$
- In practice, our "training data" must use noisy or biased samples of $v_t^\pi$

**LSMC** Least Squares Monte-Carlo uses return
$v_t^\pi \approx G_t$

**LSTD** Least Squares Temporal-Difference uses TD target
$v_t^\pi \approx R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w})$

**LSTD($\lambda$)** Least Squares TD($\lambda$) uses $\lambda$-return
$v_t^\pi \approx G_t^\lambda$

- In each case solve directly for fixed point of MC / TD / TD($\lambda$)

# LSMC and LSTD

LSMC

$$0 = \sum_{t=1}^{T} \alpha(G_t - \hat{v}(S_t, \mathbf{w}))\mathbf{x}(S_t)$$

$$\mathbf{w} = \left(\sum_{t=1}^{T} \mathbf{x}(S_t)\mathbf{x}(S_t)^\top\right)^{-1} \sum_{t=1}^{T} \mathbf{x}(S_t) G_t$$

LSTD

$$0 = \sum_{t=1}^{T} \alpha(R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}) - \hat{v}(S_t, \mathbf{w}))\mathbf{x}(S_t)$$

$$\mathbf{w} = \left(\sum_{t=1}^{T} \mathbf{x}(S_t)(\mathbf{x}(S_t) - \gamma\mathbf{x}(S_{t+1}))^\top\right)^{-1} \sum_{t=1}^{T} \mathbf{x}(S_t) R_{t+1}$$

# LSMC and LSTD

LSMC

$$0 = \sum_{t=1}^{T} \alpha(G_t - \hat{v}(S_t, \mathbf{w}))\mathbf{x}(S_t)$$

$$\mathbf{w} = \left(\sum_{t=1}^{T} \mathbf{x}(S_t)\mathbf{x}(S_t)^\top\right)^{-1} \sum_{t=1}^{T} \mathbf{x}(S_t) G_t$$

LSTD

$$0 = \sum_{t=1}^{T} \alpha(R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}) - \hat{v}(S_t, \mathbf{w}))\mathbf{x}(S_t)$$

$$\mathbf{w} = \left(\sum_{t=1}^{T} \mathbf{x}(S_t)(\mathbf{x}(S_t) - \gamma\mathbf{x}(S_{t+1}))^\top\right)^{-1} \sum_{t=1}^{T} \mathbf{x}(S_t) R_{t+1}$$

# Theoretical properties: Policy evaluation

| On/Off-Policy | Algorithm | Table Lookup | Linear | Non-Linear |
|---|---|---|---|---|
| On-Policy | MC | ✓ | ✓ | ✓ |
| | LSMC | ✓ | ✓ | - |
| | TD | ✓ | ✓ | ✗ |
| | LSTD | ✓ | ✓ | - |
| Off-Policy | MC | ✓ | ✓ | ✓ |
| | LSMC | ✓ | ✓ | - |
| | TD | ✓ | ✗ | ✗ |
| | LSTD | ✓ | ✓ | - |

# Theoretical properties: Control

| Algorithm | Table Lookup | Linear | Non-Linear |
|---|---|---|---|
| Monte-Carlo Control | ✓ | (✓) | ✗ |
| Sarsa | ✓ | (✓) | ✗ |
| Q-learning | ✓ | ✗ | ✗ |
| LSPI | ✓ | (✓) | - |

(✓) $=$ chatters around near-optimal value function

# Part 3: Pessimism in the face of uncertainty

- *Conservative* approach
- Assume that states or state-action pairs not visited are bad
- Use a penalty to avoid the new policy visiting them

# Value iteration with lower confidence bounds

**Pessimism in the face of uncertainty:** penalize value estimate of those $(s,a)$ pairs that were poorly visited [Jin et al., 2021, Rashidinejad et al., 2021]

**Algorithm:** value iteration w/ lower confidence bounds

- compute empirical estimate $\widehat{P}$ of $P$
- initialize $\widehat{Q} = 0$, and repeat

$$\widehat{Q}(s,a) \;\leftarrow\; \max\Big\{ r(s,a) + \gamma \big\langle \widehat{P}(\cdot \,|\, s,a), \widehat{V} \big\rangle - \underbrace{b(s,a;\widehat{V})}_{\text{Bernstein-style confidence bound}}\,,\; 0 \Big\}$$

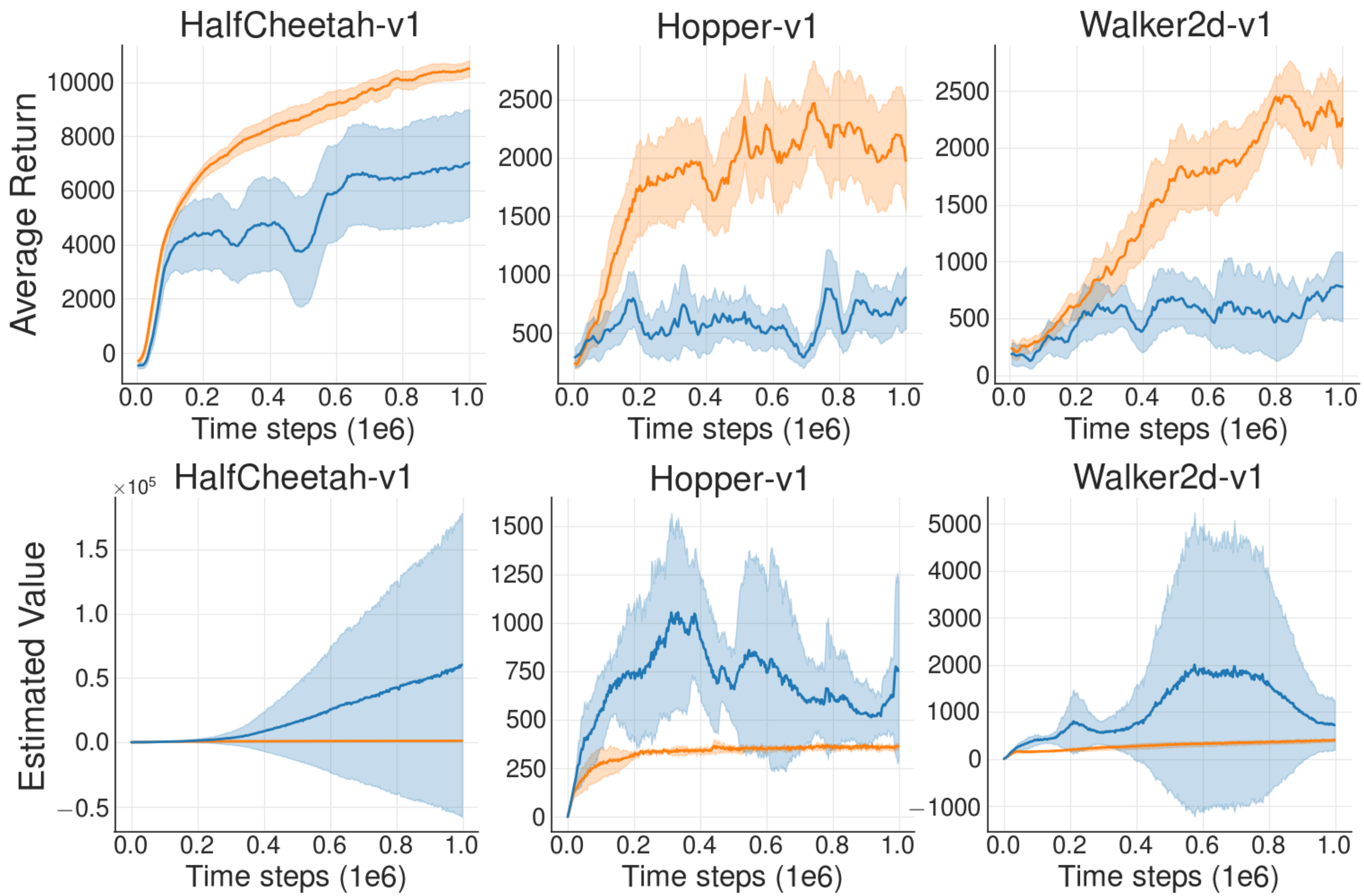
for all $(s,a)$, where $\widehat{V}(s) = \max_a \widehat{Q}(s,a)$

Q-learning version exists as well

# Alternative approach: Batch-Constrained RL

- Do NOT try to optimize value function/policy everywhere, if you only have a limited batch
- Instead, only look at policies $\pi$ such that the batch contains pairs $(s, a)$ that $\pi$ visits

# Motivation: Two versions of DDPG



# Motivation: Two versions of DDPG

- Orange agent interacts with the environment in a standard RL loop: collect data, put it in replay buffer, train, repeat
- Blue agent is trained with data collected by the orange agent concurrently
- Even though the data and algorithm are the same, being off-policy throws the learning off!!!

## Why? Extrapolation error

$$Q(s,a) \leftarrow r + \gamma Q(s',a')$$

GIVEN GENERATED

$$Q(s,a) \leftarrow r + \gamma Q(s',a')$$

$$(s',a') \notin Dataset \rightarrow Q(s',a') = \textbf{bad}$$

$$\rightarrow Q(s,a) = \textbf{bad}$$

## Batch-constrained RL

1. $a \sim \pi(s)$ such that $(s, a) \in Dataset$.
2. $a \sim \pi(s)$ such that $\left(s', \pi(s')\right) \in Dataset$.
3. $a \sim \pi(s)$ such that $Q(s, a)$ is maxed.

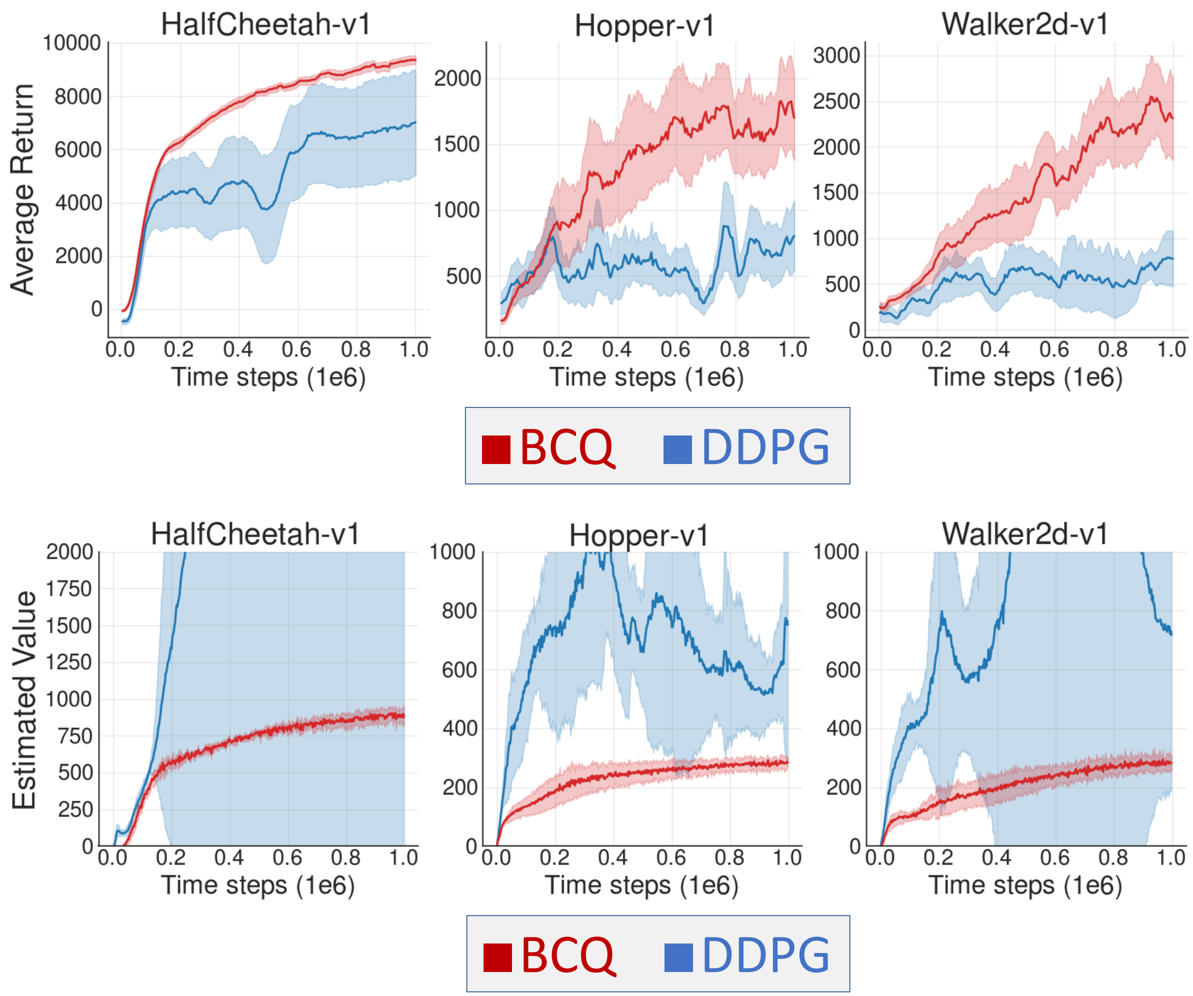
# Batch-constrained Q-learning

First imitate dataset via generative model:
$G(a|s) \approx P_{Dataset}(a|s)$.

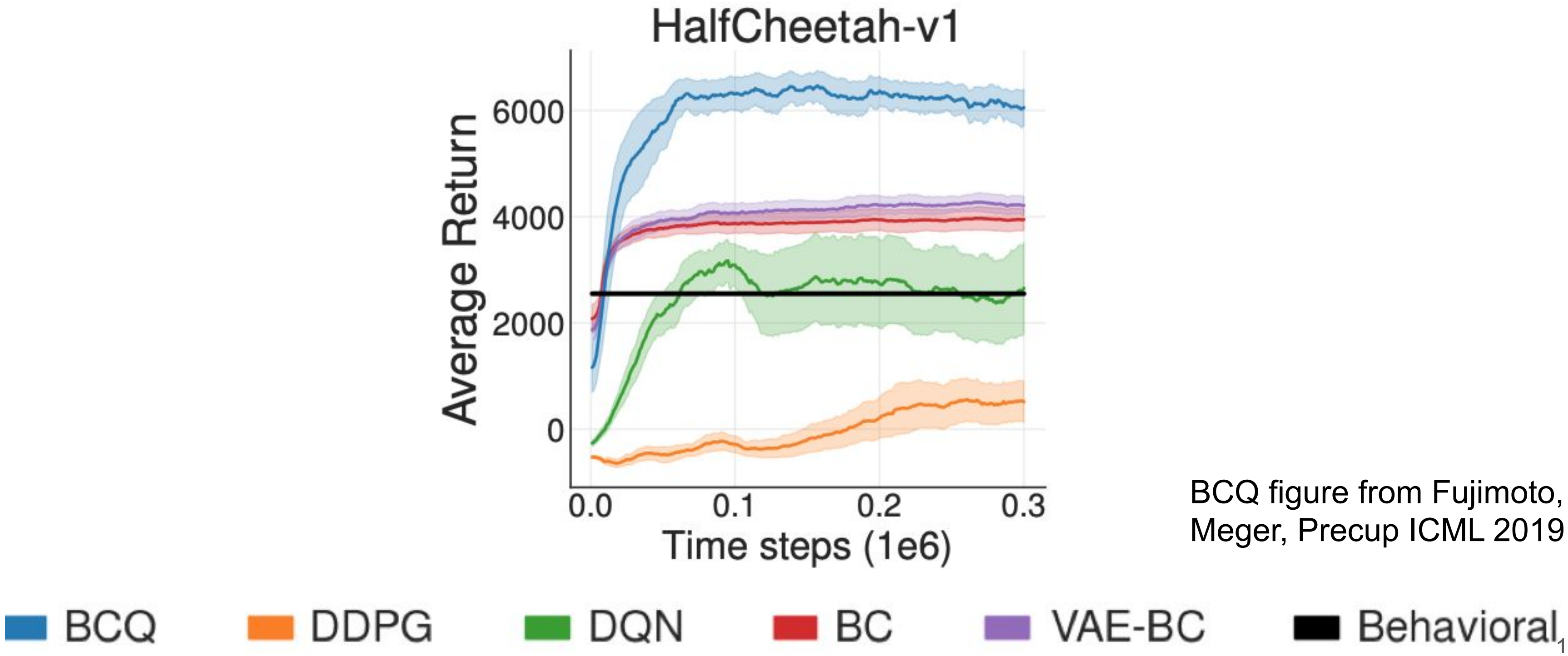
$\pi(s) = \text{argmax}_{a_i} Q\ (s, a_i)$, where $a_i \sim G$
(I.e. select the best action that is likely under the dataset)

(+ some additional deep RL magic)

# Revisiting previous example



# BCQ comparison




BCQ figure from Fujimoto, Meger, Precup ICML 2019