Deep Reinforcement Learning and Control

Off policy RL

Fall 2021, CMU 10-703

Ruosong Wang

Extrapolation Error

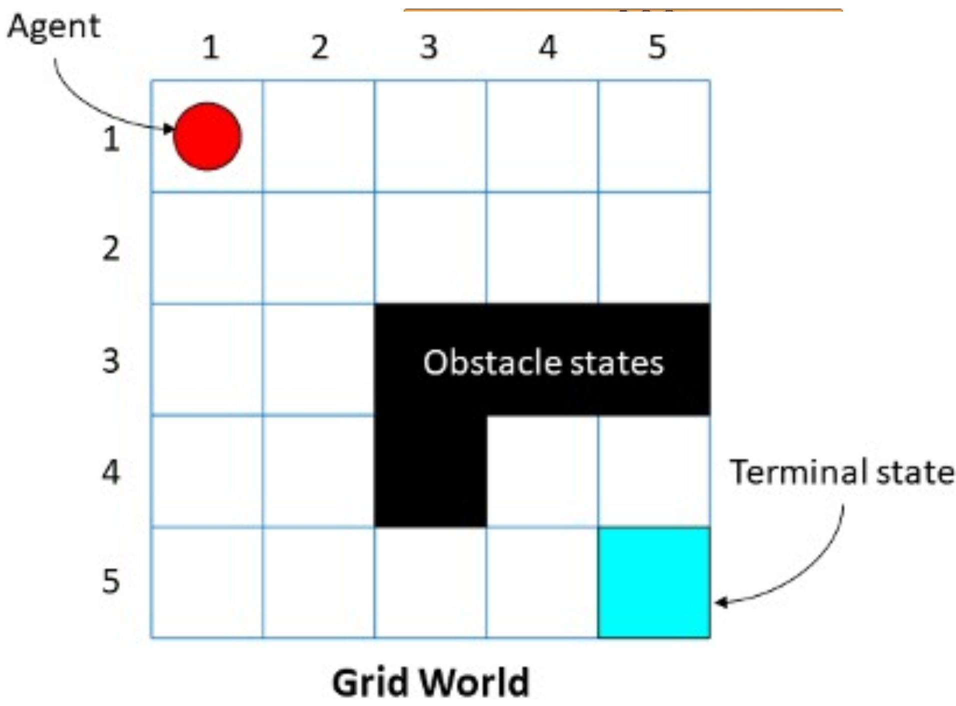
- $Q(s,a) = r(s,a) + \gamma Q(s',a')$

- If (s',a') is not in the dataset, then estimate for $Q(s,a)$ could be bad

- Could function approximation help here?

- I.e., can we use the dataset + supervised learning to predict $Q(s',a')$?

Offline Policy Evaluation



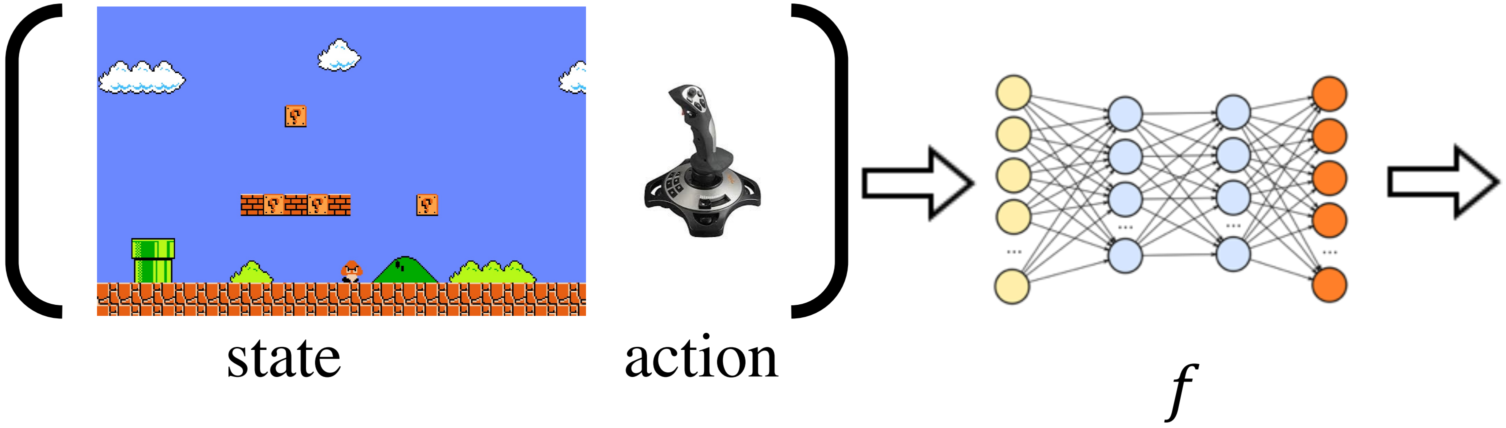
- Given a dataset $D = \{(s_i, a_i, r_i, s_i')\}$
- A target policy π
- Goal: estimate the value of the policy

- **Value Iteration**
- For $t = 1, 2, \ldots$
  - $\hat{Q}_t(s_i, a_i) = r_i + \gamma \hat{V}_{t-1}(s_i')$
  - $\hat{V}_t(s) = \hat{Q}_t(s, \pi(s))$

Tabular setting:
If every state-action pair has $\mathrm{poly}(1/(1-\gamma), 1/\varepsilon)$ samples, then estimated value is accurate up to an error of ε

How to deal with larger (or even continuous) state space?

RL with Function Approximation



Value-based Learning

$Q(s,a) \approx f(s,a)$

- Function approximation $f \in \mathcal{F}$
- $\mathcal{F}$: function class with bounded complexity.
- Linear functions, kernels, neural networks, etc

This talk: the linear setting

Feature extractor $\phi : S \times A \to \mathbb{R}^d$

$\mathcal{F}$ = linear functions with respect to ϕ

$Q^\pi(s,a) = \phi(s,a)^\top \theta^*$

One-Step Offline RL ($\gamma = 0$)

- Given a dataset $D = \{(s_i, a_i, r_i)\}$
- We know that $Q(s,a) = r(s,a) + \gamma Q(s', a') = r(s,a)$
- And $Q(s,a)$ is linear, i.e., $Q(s,a) = \phi(s,a)^\top \theta^*$ for some unknown θ^*

- Can we use the given dataset to learn Q-values for other state-action pairs?
- Linear regression (with distribution shift)

- Feature Matrix: $\Phi \in \mathbb{R}^{N \times d}$ with $\phi(s_i, a_i)$ as rows
- Least squares predictor: $\hat{\theta} = (\Phi^\top \Phi)^{-1} \Phi r$

One-Step Offline RL ($\gamma = 0$)

- Feature Covariance Matrix
  - $\Sigma = \mathbb{E}_{(s,a)\sim\mu}[\phi(s,a)\phi(s,a)^\top]$
- Suppose
  - **Coverage**: $\sigma_{\min}(\Sigma) \geq \lambda_{\min}$

- Lemma: When $|D| \geq \mathrm{poly}(d, 1/\varepsilon, 1/\lambda_{\min})$, then least squares works
- For any (s,a), $\left|Q(s,a) - \hat{\theta}^\top \phi(s,a)\right| \leq \varepsilon$

How to deal with large state space + long planning horizon?

Fitted-Q Iteration (FQI)

- Value Iteration + Linear Regression

- For $t = 1, 2, \dots$
  - For each data (s_i, a_i), $\hat{Q}_t(s_i, a_i) = r_i + \gamma \hat{V}_{t-1}(s_i')$
  - Run linear regression on $\{\phi(s_i, a_i), \hat{Q}_t(s_i, a_i)\}$ to learn $\theta_t \in \mathbb{R}^d$
  - $\hat{V}_t(s) = \phi\big(s, \pi(s)\big)^\top \theta_t$

- Simple and widely used
- When does it work?

Characterizing FQI

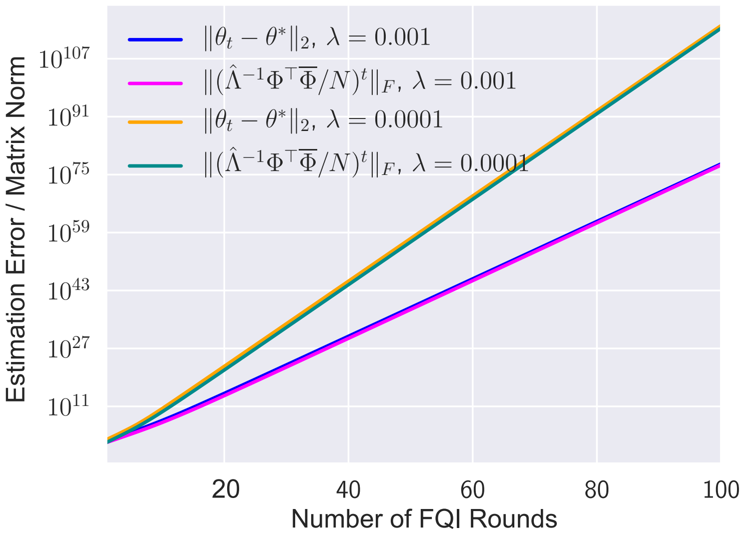
- Notations:
  - Feature Matrix: $\Phi \in \mathbb{R}^{N \times d}$ with $\phi(s_i, a_i)$ as rows
  - Empirical Feature Covariance Matrix: $\Sigma = \Phi^\top \Phi$
  - "Next" Feature Matrix: $\overline{\Phi} \in \mathbb{R}^{N \times d}$ with $\phi(s_i{}', \pi(s_i'))$ as rows
- Lemma

$$\theta_T - \theta^* = \gamma^T L^T \left(\theta_0 - \theta^*\right) \text{ where } L = \Sigma^{-1} \Phi^\top \overline{\Phi}$$

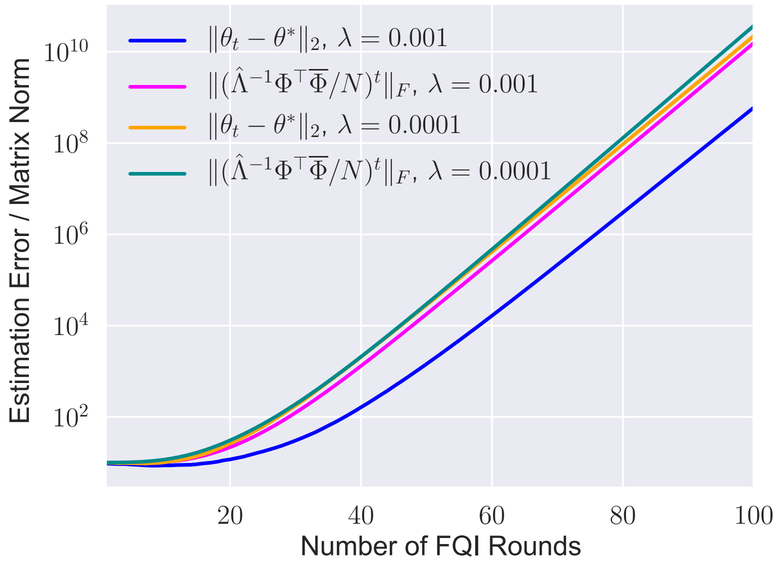
- Non-expansive L => error goes to 0 by taking T large
- Expansive L => geometric error amplification
- Low distribution shift => non-expansive L

Simulation Results

- $N = 100$ or $N = 200, d = 100, \gamma = 0.99$
- $\theta^*, \phi(s,a), \phi\big(s', \pi(s')\big) \sim N(0, I)$
- $r_i = \Big(\phi(s_i, a_i) - \gamma\phi\big(s_i', \pi(s_i')\big)\Big)^\top \theta^*$ to ensure linearity



$N = 100$



$N = 200$

Is geometric error amplification inherent in Offline RL?

Hardness Result

- Geometric error amplification is inherent
- Coverage assumption: feature covariance matrix is well-conditioned

Theorem [**W.**, Foster, Kakade'20]

Suppose coverage + linear Q^{π}. There is an MDP such that for any policy π, any algorithm requires an exponential number of samples to approximately evaluate π.

How serious is the hardness result in practice?

Experimental Methodology

- Step 1: Run online RL methods (DQN, TD3) to find a target policy π and a good representation
  - Target policy: final policy output by DQN / TD3
  - Feature mapping: output of the last hidden layer of the learned value function networks.
- Step 2: Collect
  - $D^\star$: 1 million
  - We combine L rom lower performing policies
- Step 3: Run off

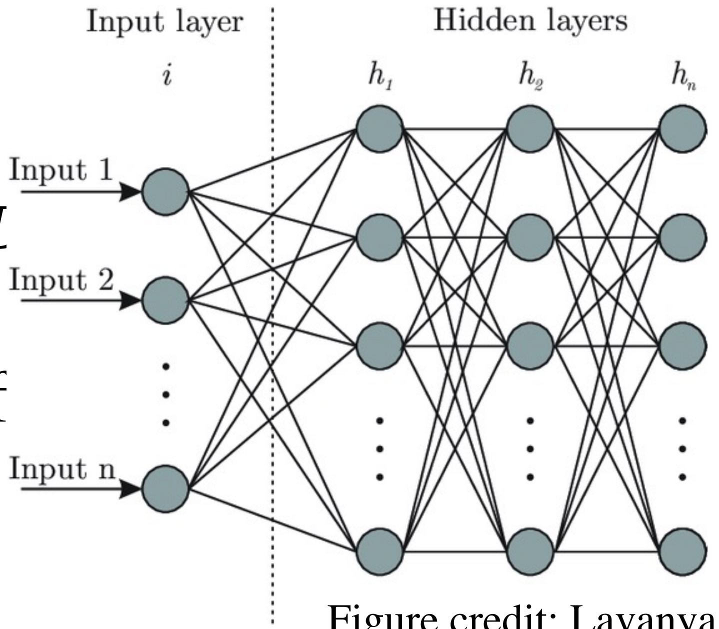
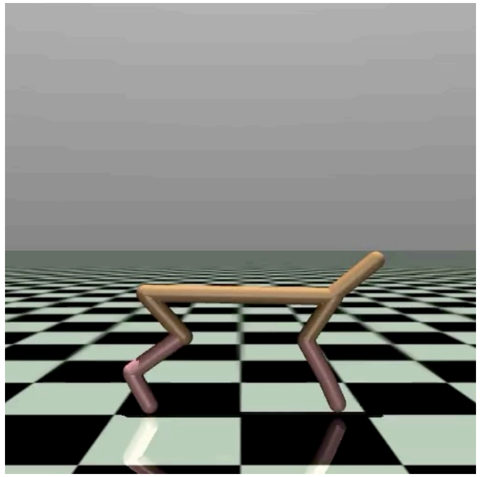
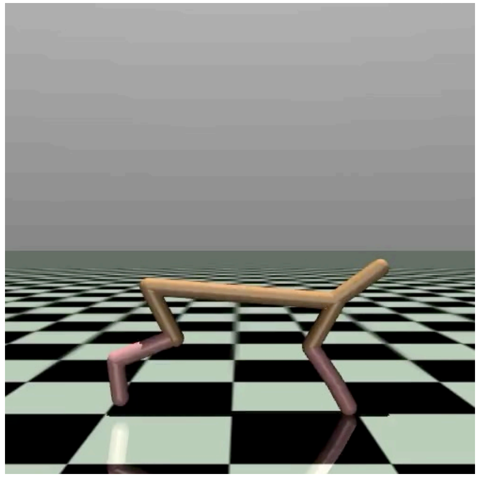


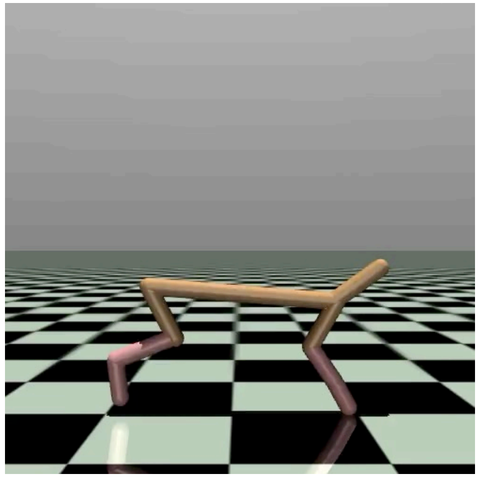
Figure credit: Lavanya Shukla

Target Policy + Random Policy

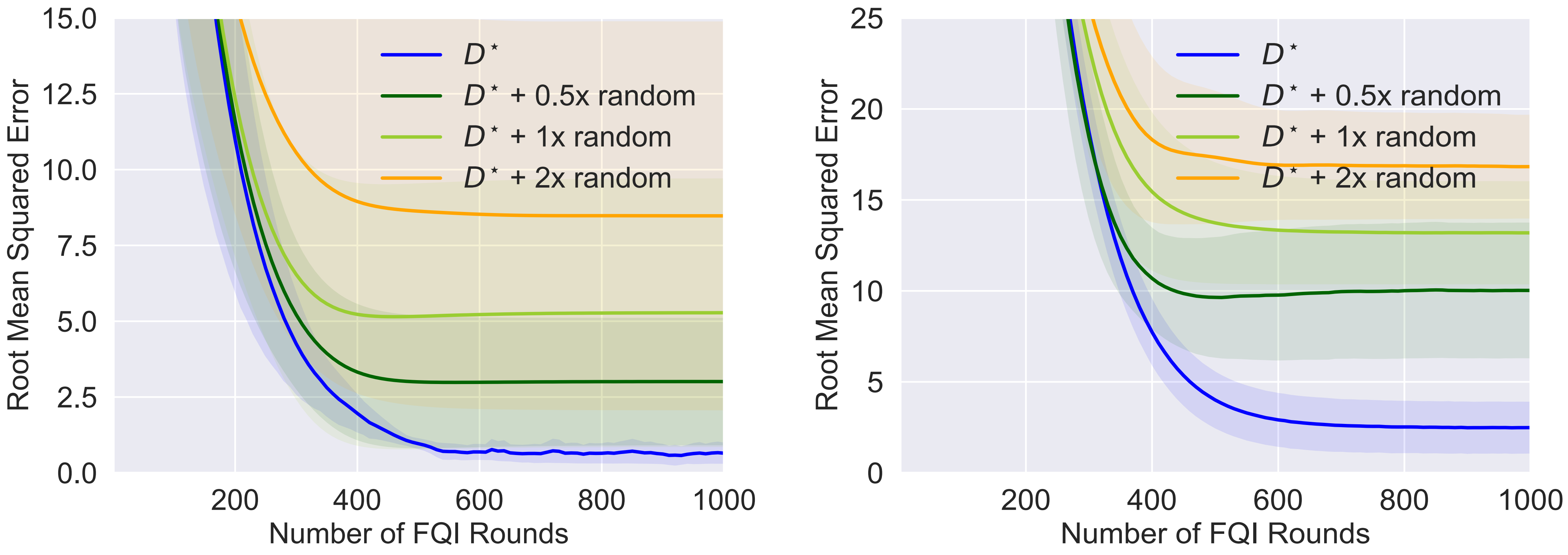
What happens if we use {  +  } +

Random policy

Target policy π

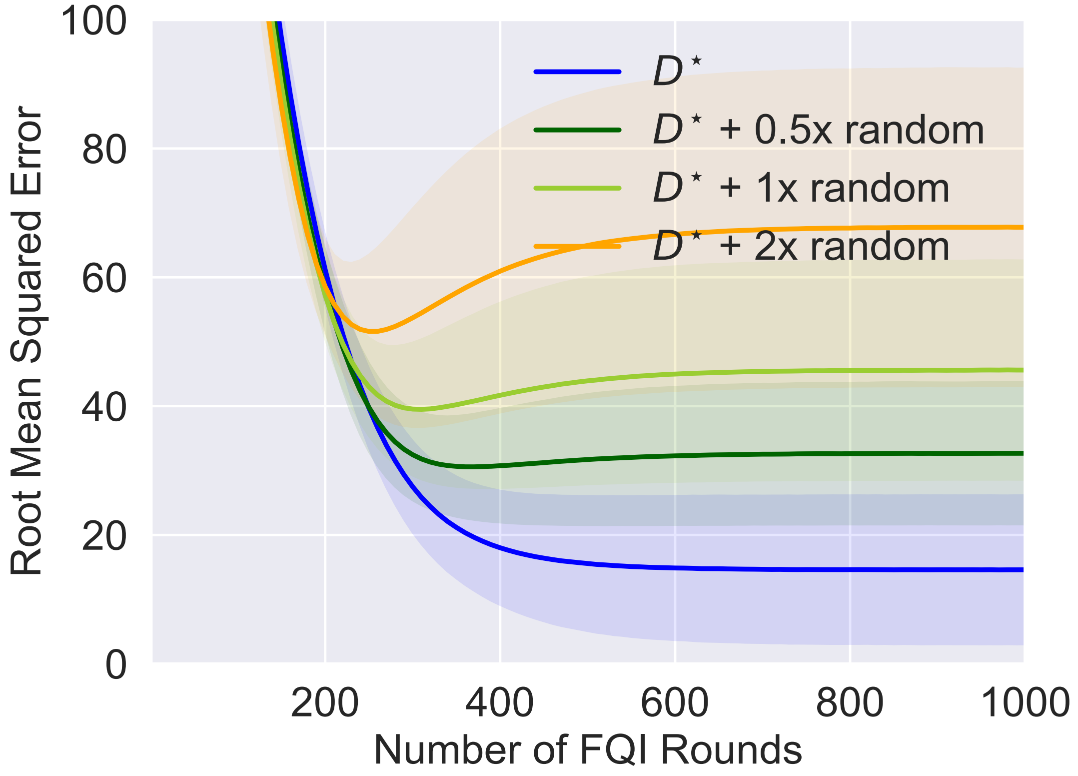
neural representation + offline RL to evaluate  ?

Results

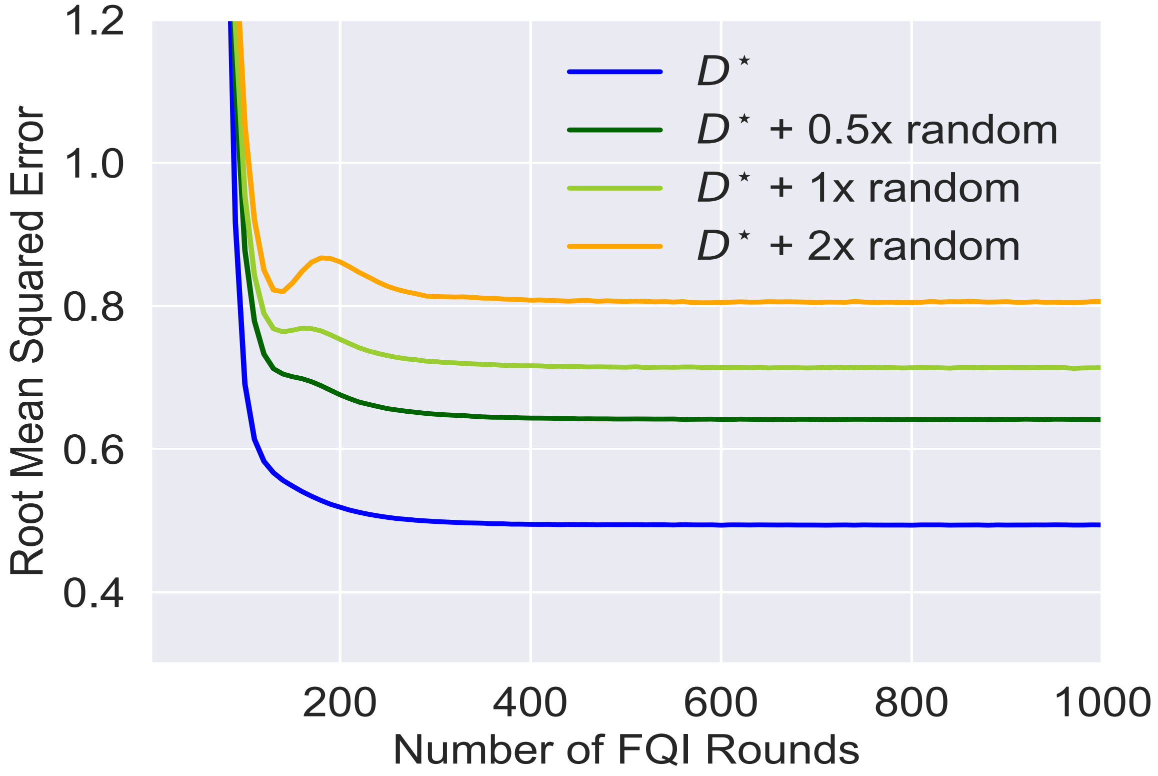


CartPole-v0 Hopper-v2

Results



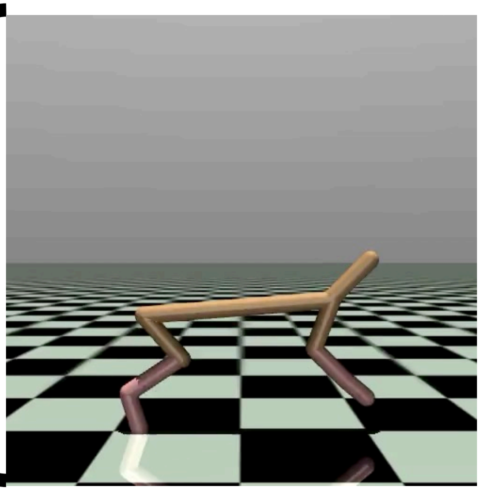
Walker2d-v2



MountainCar-v0

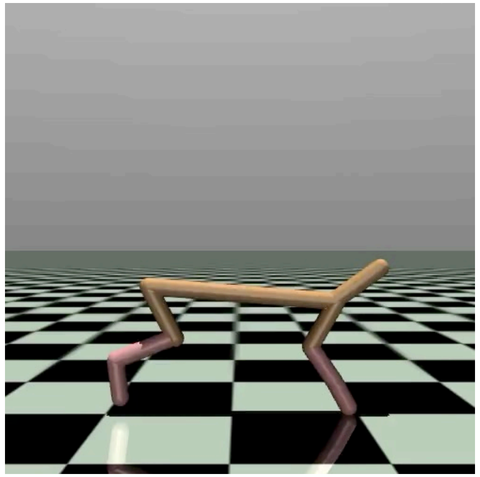
Target Policy + Lower Performing Policy

What happens if we use



Lower performing policy

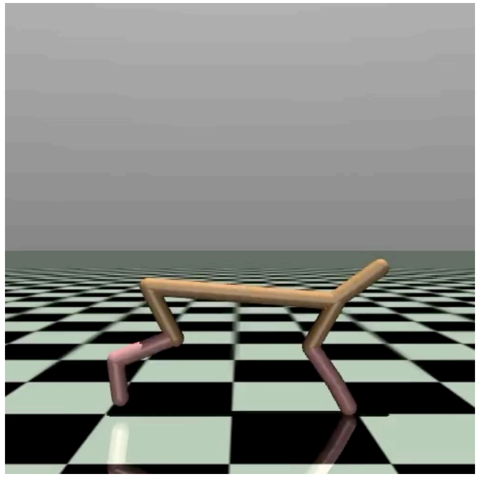
+



Target policy π

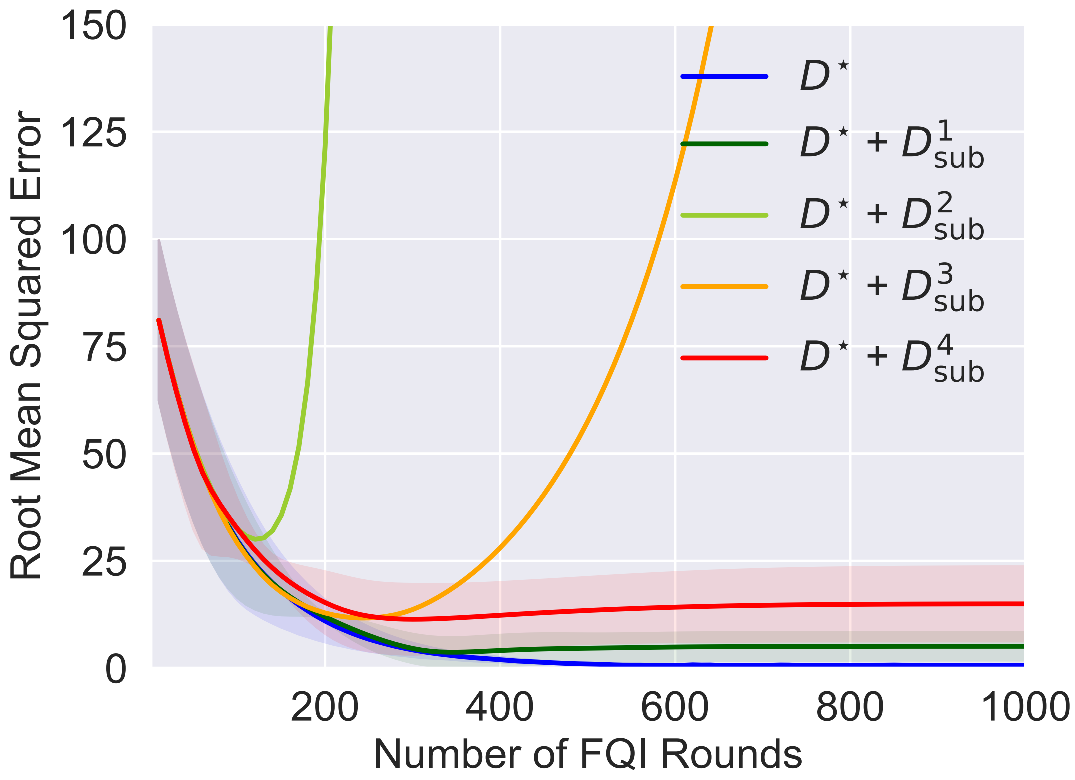
+

neural representation + offline RL to evaluate

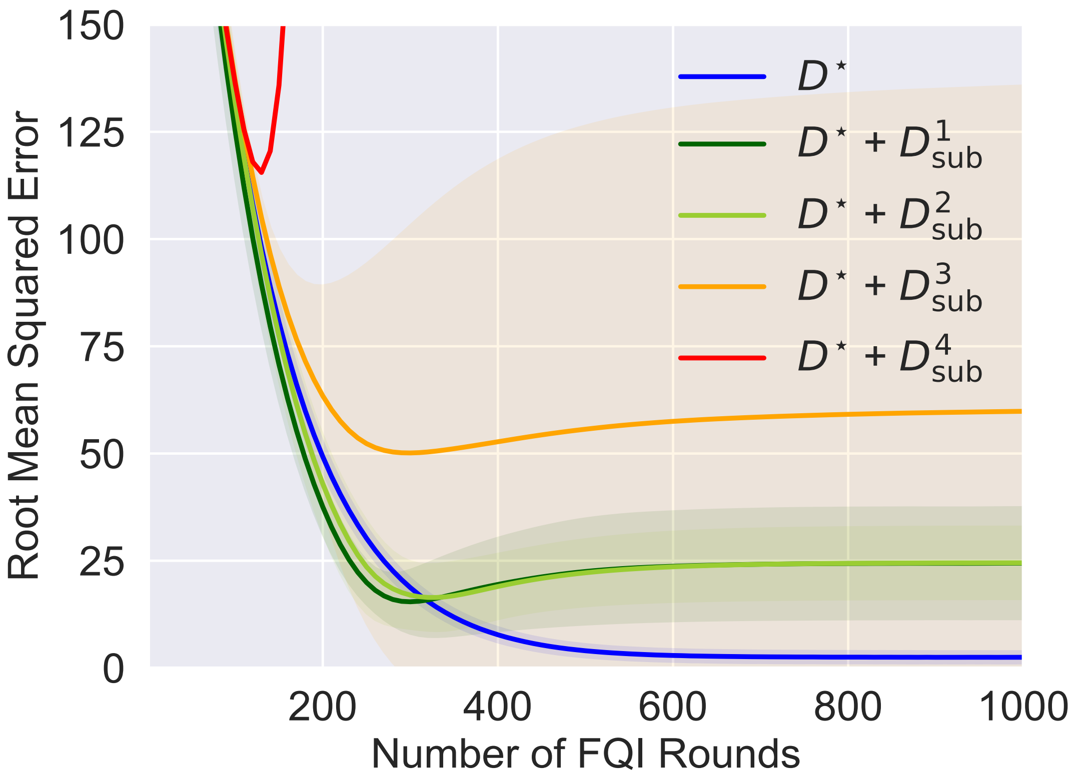


?

Results



CartPole-v0



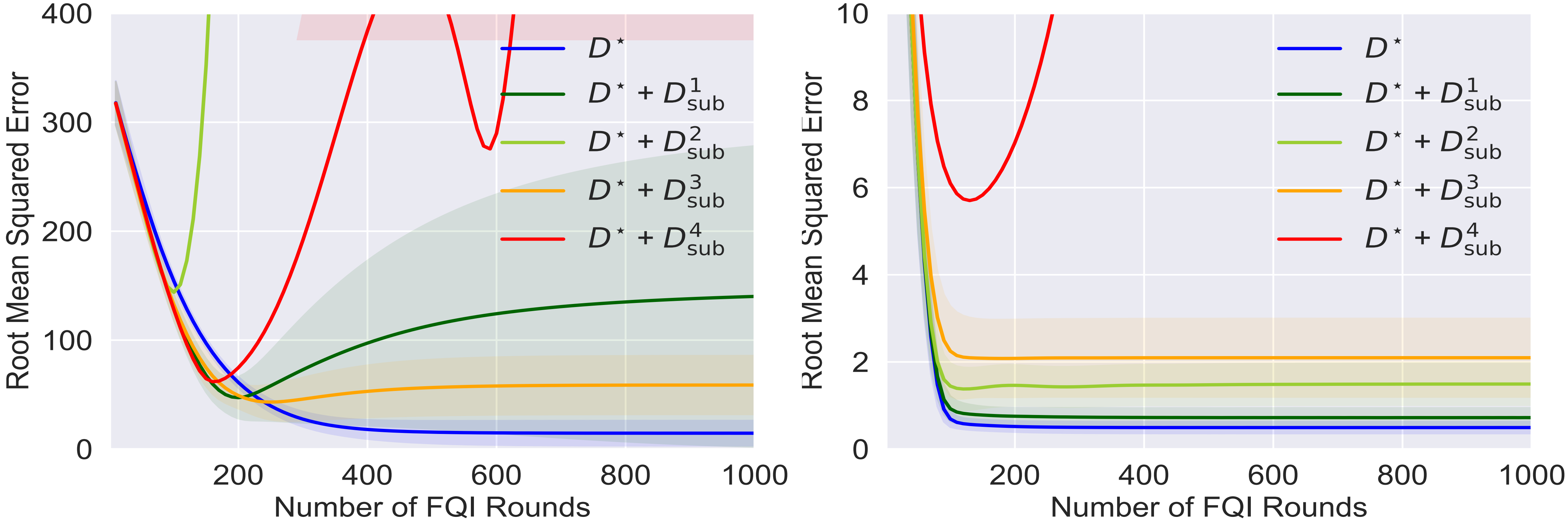
Hopper-v2

$V^{\pi_1} > V^{\pi_2} > V^{\pi_3} > V^{\pi_4}$

$D^\star$: induced by target policy with 1 million samples

D^i_{sub} : induced by π_i with 1 million samples

Results



Walker2d-v2

Mountain-v0

Observations

- Adding more data (from random trajectories / lower performing policies) into the dataset generally hurts the performance

- Geometric error amplification does occur