Offline Reinforcement Learning

CS 224R

Course reminders

Project

- Survey due today
- CA mentors assigned soon after
- Proposal due next Weds

graded fairly lightly- it's for your benefit!

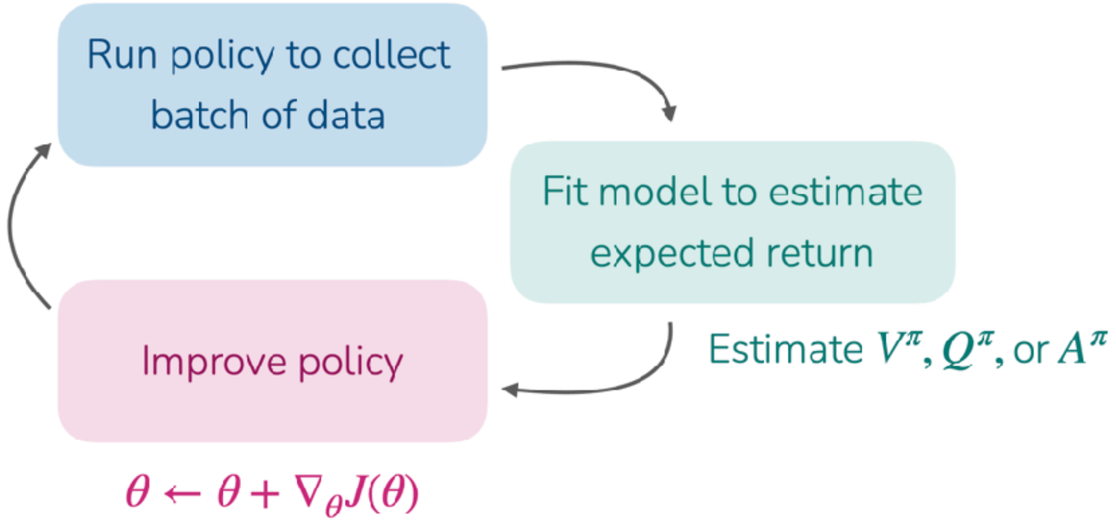
Homework

- Homework 2 due Friday (start early!)

Recap: Online RL Algorithm Summary (model-free algorithms)

	Vanilla PG (w/o importance weights)	PPO-like methods	Off-policy actor-critic (e.g. SAC)	Q-learning
What data used?	On-policy	Technically off-policy, often called on-policy	Off-policy, with replay buffer	Off-policy, with replay buffer
How to make it off-policy?	n/a	Importance weights	Fit Q with TD, sample a from π	Fit Q* with TD
Fit value func?	No	V^π	Q^π	$Q^\star$
How to estimate goodness?	$\sum r_t - b$	V^π: MC, TD, or n-step returns A^π: $r+\gamma V(s')-V(s)$ or GAE	TD or n-step returns	TD or n-step returns
Policy update	$(\nabla \log \pi)\left(\sum r_t - b\right)$	$(\nabla \log \pi)\hat{A}^\pi$	$(\nabla \log \pi)\hat{Q}^\pi$	$\max_{\mathbf{a}} \hat{Q}^\star(\mathbf{s}, \mathbf{a})$

Recap: Fitting Value Functions



Online RL with actor-critic

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) A^{\pi_\theta}(\mathbf{s}_{i,t}, \mathbf{a}_{i,t})$$

Estimate what is good and bad, then do more of the good stuff.

1. Estimate V^π with Monte Carlo

$$\min_\phi \sum_{\mathbf{s}_t \sim \mathcal{D}} \| \hat{V}_\phi^{\pi_\theta}(\mathbf{s}_t) - \sum_{t'=t}^{T} \gamma^{t'-t} r(\mathbf{s}_t, \mathbf{a}_t) \|^2$$

2. Estimate V^π with bootstrapped / TD updates

$$\min_\phi \sum_{(\mathbf{s},\mathbf{a},\mathbf{s}') \sim \mathcal{D}} \| \hat{V}_\phi^{\pi_\theta}(\mathbf{s}) - \left(r(\mathbf{s}, \mathbf{a}) + \gamma \hat{V}_\phi^{\pi_\theta}(\mathbf{s}') \right) \|^2$$

3. Estimate Q^π with bootstrapped / TD updates

$$\min_\phi \sum_{(\mathbf{s},\mathbf{a},\mathbf{s}') \sim \mathcal{D}} \| \hat{Q}_\phi^{\pi_\theta}(\mathbf{s}, \mathbf{a}) - \left(r(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{a}' \sim \pi_\theta(\cdot|\mathbf{s}')}[\hat{Q}_\phi^{\pi_\theta}(\mathbf{s}', \mathbf{a}')] \right) \|^2$$

Recap: Full Off-Policy Actor-Critic Method

online actor-critic algorithm:

1. take action $\mathbf{a} \sim \pi_\theta(\mathbf{a}|\mathbf{s})$, get $(\mathbf{s}, \mathbf{a}, \mathbf{s}', r)$, store in $\mathcal{R}$
2. sample a batch $\{\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}'_i\}$ from buffer $\mathcal{R}$
3. update $\hat{Q}^\pi_\phi$ using targets $y_i = r_i + \gamma \hat{Q}^\pi_\phi(\mathbf{s}'_i, \mathbf{a}'_i)$ where $\mathbf{a}'_i \sim \pi_\theta(\cdot|\mathbf{s}'_i)$
4. $\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_i \nabla_\theta \log \pi_\theta(\mathbf{a}^\pi_i|\mathbf{s}_i) \hat{Q}^\pi(\mathbf{s}_i, \mathbf{a}^\pi_i)$ where $\mathbf{a}^\pi_i \sim \pi_\theta(\mathbf{a}|\mathbf{s}_i)$
5. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

The plan for today

Offline RL

1. Why offline RL?
2. Data stitching example
3. Core method ideas
    1. Constraining policy to actions in the data
    2. Estimating better value functions without OOD queries

Part of homework 3!

Key learning goals:

- the key challenges arising in offline reinforcement learning
- two core techniques for offline RL (& why they work!)
- how offline RL can improve over imitation learning

Why offline RL?

Online RL process (on-policy or off-policy)

- Collect data
- Update policy on latest data or data so far

Offline RL process

- Given static dataset
- Train policy on provided dataset

Why, or when, might offline RL be more useful?

- leverage datasets collected by people, existing systems
- online policy collection may be risky, unsafe
- reuse previously collected data rather than recollecting
  (e.g. previous experiments, projects, robots, institutions)

Note: A blend of offline then online RL is also possible!

- offline pretraining -> online fine-tuning
- iterated offline RL training or “batch online RL”

Why offline RL?

Offline RL process

- Given static dataset
- Train policy on provided dataset

More formally:

Offline dataset $\mathcal{D} : \{(\mathbf{s}, \mathbf{a}, \mathbf{s}', r)\}$ sampled from some *unknown policy* π_β

"behavior policy"

(**Note**: π_β may be a mixture of policies)

i.e. data sampled from $p_{\pi_\beta}(\tau) = p(\mathbf{s}_1) \prod_{t=1}^{T} \pi_\beta(\mathbf{a}_t|\mathbf{s}_t) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$

Objective: $\max_\theta \mathbb{E}_{p_{\pi_\theta}(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right]$ <- expectation under *the learned policy* π_θ

distribution shift

Why offline RL?

Offline RL process

- Given static dataset
- Train policy on provided dataset

Where does the data come from?

- human collected data
- data from a hand-designed system / controller
- data from previous RL run(s)
- a mixture of sources

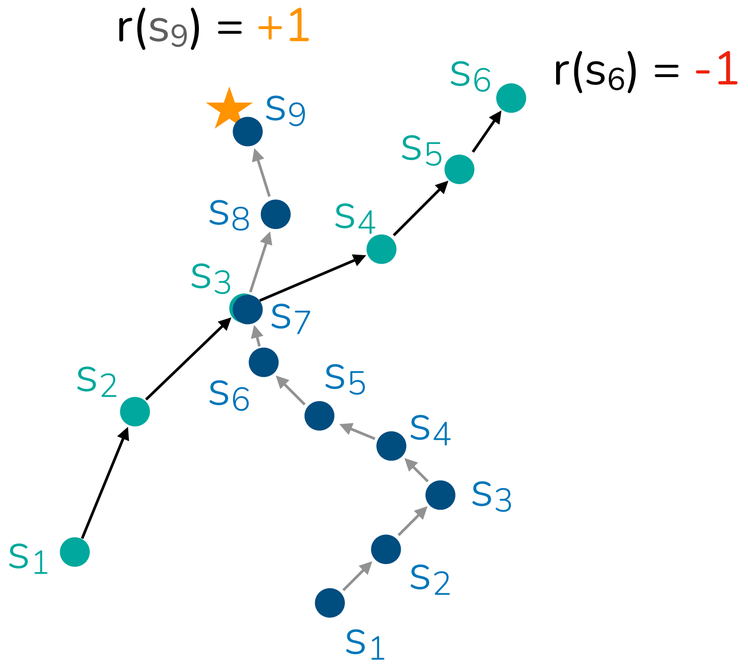
The plan for today

Offline RL

1. Why offline RL?
2. **Data stitching example**
3. Core method ideas
    1. Constraining policy to actions in the data
    2. Estimating better value functions without OOD queries

Why offline RL versus imitation learning?

Offline data may not be optimal!



s_1 -> s_3 is good behavior

s_7 -> s_9 is good behavior

Can we learn a policy that goes from s_1 to s_9?

Questions:

1. What will a policy learned with vanilla imitation learning do?
2. What should a policy learned with RL be able to do? Does it depend on whether you use TD or MC for value function learning?

Recall: Imitation methods can't outperform the policy that collected the data.

—> Offline RL can leverage reward information to outperform behavior policy.

—> Good offline RL methods can ***stitch*** together good behaviors.

The plan for today

Offline RL

1. Why offline RL?
2. Data stitching example
3. **Methods**
    1. Filtered imitation baseline
    2. Weighted imitation to the actions in the data
    3. More than one-step improvement

Can we just use off-policy algorithms?

Recall: Off-policy actor & critic updates (e.g. SAC):

Q-function update: $\min_{\phi} \sum_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}} \|\hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s},\mathbf{a}) - \left(r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{a}'\sim\pi_\theta(\cdot|\mathbf{s}')}[\hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s}',\mathbf{a}')]\right)\|^2$

Subsequent policy update: $\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_i \nabla_\theta \log \pi_\theta(\mathbf{a}_i^\pi|\mathbf{s}_i)\hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s}_i,\mathbf{a}_i^\pi)$ where $\mathbf{a}_i^\pi \sim \pi_\theta(\mathbf{a}|\mathbf{s}_i)$

What happens if you optimize this using a static dataset?

(e.g. say data collected by a mediocre policy)

Can we just use off-policy algorithms?

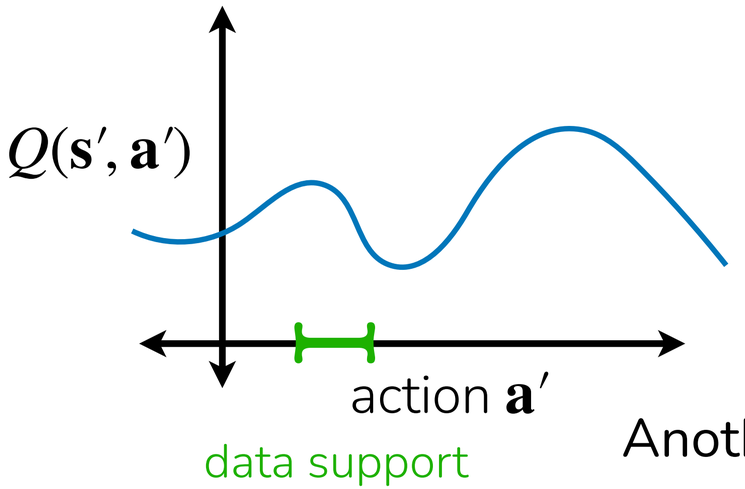
Recall: Off-policy critic objective $\min_{\phi} \sum_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}} \| \hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s},\mathbf{a}) - \left(r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{a}'\sim\pi_\theta(\cdot|\mathbf{s}')}[\hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s}',\mathbf{a}')] \right) \|^2$

What happens if you optimize this using a static dataset?

(e.g. say data collected by a mediocre policy)

What happens when evaluating Q on actions $\mathbf{a}'$ not in the dataset?

Randomly init. Q-function for state $\mathbf{s}'$



- Q-function will be unreliable on OOD actions
- policy will seek out actions where Q-function is over-optimistic
- After policy update, Q-values will become substantially overestimated.

Another perspective: learned policy deviates too much from behavior policy.

How to mitigate overestimation in offline RL?

This is the **core goal** of offline RL methods!

The plan for today

Offline RL

1. Why offline RL?
2. Data stitching example
3. Core method ideas
    1. **Constraining policy to actions in the data**
    2. Estimating better value functions without OOD queries

A simple way to leverage rewards in imitation

If we have reward labels: imitate only the good trajectories?

Filtered behavior cloning:

1. Rank trajectories by return $r(\tau) = \sum_{(\mathbf{s}_t, \mathbf{a}_t) \in \tau} r(\mathbf{s}_t, \mathbf{a}_t)$
2. Filter dataset to include top k% of data $\tilde{D} : \{\tau \,|\, r(\tau) > \eta\}$
3. Imitate filtered dataset: $\max_\theta \sum_{(\mathbf{s},\mathbf{a}) \in \tilde{D}} \log \pi_\theta(\mathbf{a} \,|\, \mathbf{s})$

A very primitive approach to using reward information.

Therefore, a **good baseline** to test against!

Better way to do weighted imitation learning?

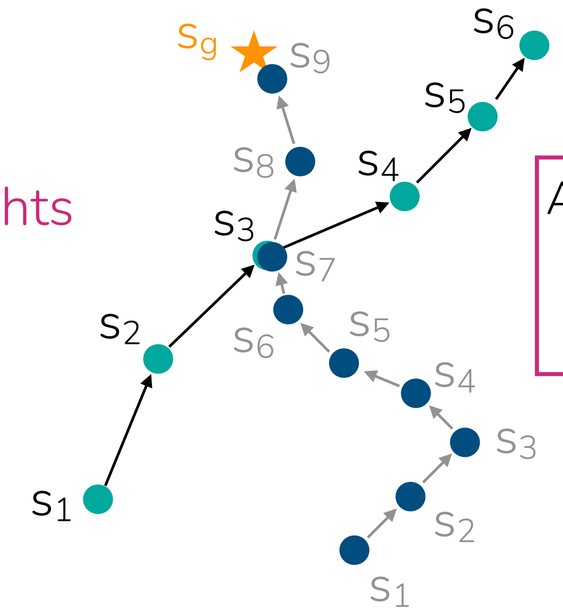
Could we weight each transition depending on **how good the action is**?

How do you measure how good an action is? Recall: advantage function A

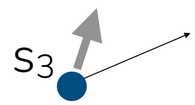
$A^\pi(\mathbf{s}_t, \mathbf{a}_t) = Q^\pi(\mathbf{s}_t, \mathbf{a}_t) - V^\pi(\mathbf{s}_t)$: how much better $\mathbf{a}_t$ is

$$\theta \leftarrow \arg\max_\theta E_{\mathbf{s},\mathbf{a}\sim D}\left[\log \pi_\theta(\mathbf{a}\,|\,\mathbf{s})\exp(A(\mathbf{s},\mathbf{a}))\right]$$

standard imitation learning with advantage weights



Adv. weights for s_3



Aside: Can show that advantage-weighted objective approximates KL-constrained objective.

$$\pi_{new} = \arg\max_\pi E_{\mathbf{a}\sim\pi(\cdot|\mathbf{s})} Q(\mathbf{s},\mathbf{a}) \text{ s.t. } D_{KL}(\pi\|\pi_\beta) < \epsilon$$

See Peters et al. (REPS), Rawlik et al. ("psi-learning")

Better way to do weighted imitation learning?

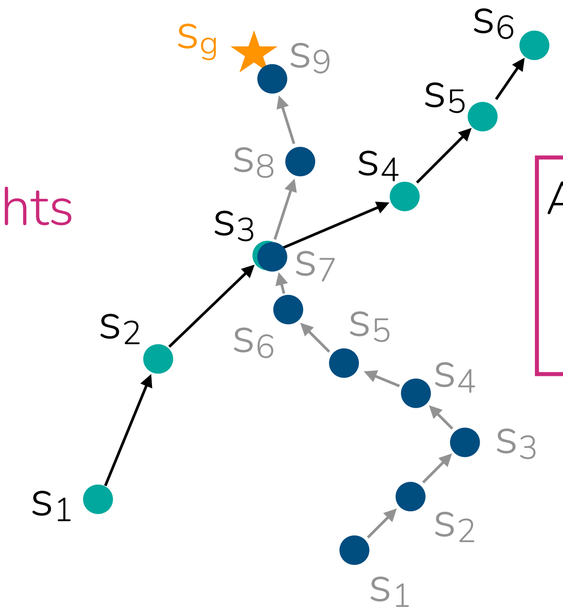
Could we weight each transition depending on **how good the action is**?

How do you measure how good an action is? Recall: advantage function A

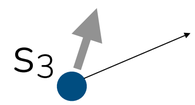
$A^\pi(\mathbf{s}_t, \mathbf{a}_t) = Q^\pi(\mathbf{s}_t, \mathbf{a}_t) - V^\pi(\mathbf{s}_t)$: how much better $\mathbf{a}_t$ is

$$\theta \leftarrow \arg\max_\theta E_{\mathbf{s},\mathbf{a}\sim D}\left[\log \pi_\theta(\mathbf{a}\,|\,\mathbf{s})\exp(A(\mathbf{s},\mathbf{a}))\right]$$

standard imitation learning with advantage weights



Adv. weights for s_3



Advantage of which policy? We'll use A^{π_β} for now.

Key question: How to estimate the advantage function?

Advantage-weighted regression

Could we weight each transition depending on **how good the action is**?

$$\theta \leftarrow \arg\max_{\theta} E_{\mathbf{s},\mathbf{a}\sim D}\left[\log \pi_\theta(\mathbf{a}\,|\,\mathbf{s})\exp(A(\mathbf{s},\mathbf{a}))\right]$$

standard imitation learning with advantage weights

Key question: How to estimate the advantage function?

First simple approach

Estimate $V^{\pi_\beta}(s)$ with Monte Carlo: $\min_{\phi} \sum_{\mathbf{s}_t\sim\mathcal{D}} \|\hat{V}_\phi^{\pi_\beta}(\mathbf{s}_t) - \sum_{t'=t}^{T} r(\mathbf{s}_t,\mathbf{a}_t)\|^2$

Approximate $\hat{A}^{\pi_\beta}(\mathbf{s}_t,\mathbf{a}_t) = \sum_{t'=t}^{T} r(\mathbf{s}_t,\mathbf{a}_t) - \hat{V}_\phi^{\pi_\beta}(\mathbf{s}_t)$

empirical return

Question: What do you learn for deterministic policy π_β?

Peng, Kumar, Zhang, Levine. Advantage-Weighted Regression. ‘19

Advantage-weighted regression

Full AWR algorithm

1. Fit value function: $\min_{\phi} \sum_{\mathbf{s}_t \sim \mathcal{D}} \| \hat{V}_{\phi}^{\pi_\beta}(\mathbf{s}_t) - \sum_{t'=t}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \|^2$

2. Train policy: $\max_{\theta} \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \mathcal{D}} \left[\log \pi_\theta(\mathbf{a}_t | \mathbf{s}_t) \exp\left(\frac{1}{\alpha} \left(\sum_{t'=t}^{T} r(\mathbf{s}_t, \mathbf{a}_t) - \hat{V}_{\phi}^{\pi_\beta}(\mathbf{s}_t) \right) \right) \right]$

hyperparameter

+ Simple

+ Avoids querying or training on any OOD actions!

- Monte Carlo estimation is noisy

- $\hat{A}^{\pi_\beta}$ is for weaker policy than $\hat{A}^{\pi_\theta}$

Peng, Kumar, Zhang, Levine. Advantage-Weighted Regression. ‘19

Can we get a better advantage estimate?

Estimate advantages using TD updates?

Estimate Q-function: $\min_\phi \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D}}\left[\left(\hat{Q}^{\pi_\theta}_\phi(\mathbf{s},\mathbf{a}) - \left(r + \gamma \mathbb{E}_{\mathbf{a}'\sim\pi(\cdot|\mathbf{s})}[\hat{Q}^{\pi_\theta}_\phi(\mathbf{s}',\mathbf{a}')]\right)\right)^2\right]$

alternative: $\mathbf{a}' \sim D$

Update policy with AWR: $\theta \leftarrow \arg\max_\theta E_{\mathbf{s},\mathbf{a}\sim D}\left[\log \pi_\theta(\mathbf{a}|\mathbf{s}) \exp(\hat{A}^{\pi_\theta}(\mathbf{s},\mathbf{a}))\right]$

where $\hat{A}^{\pi_\theta}(\mathbf{s},\mathbf{a}) = \hat{Q}^{\pi_\theta}_\phi(\mathbf{s},\mathbf{a}) - \mathbb{E}_{\bar{\mathbf{a}}\sim\pi_\theta(\cdot|\mathbf{s})}[\hat{Q}^{\pi_\theta}_\phi(\mathbf{s},\bar{\mathbf{a}})]$

+ Gives Q-function estimate for current policy π_θ, instead of π_β

- Queries Q on OOD actions

+ Still only trains policy on actions in the dataset

Key question: how can we estimate advantage for a policy better than π_β *without querying* Q *on OOD actions?*

"advantage-weighted actor-critic"

You will implement AWAC in homework 3!

The plan for today

Offline RL

1. Why offline RL?
2. Data stitching example
3. Core method ideas
    1. Constraining policy to actions in the data
    2. **Estimating better value functions without OOD queries**

Can we do better?

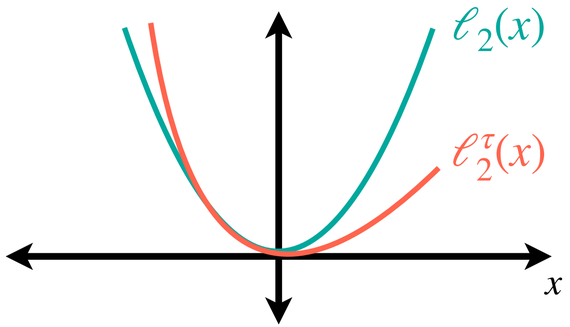
Want to estimate advantages using TD updates, without querying Q on OOD actions.

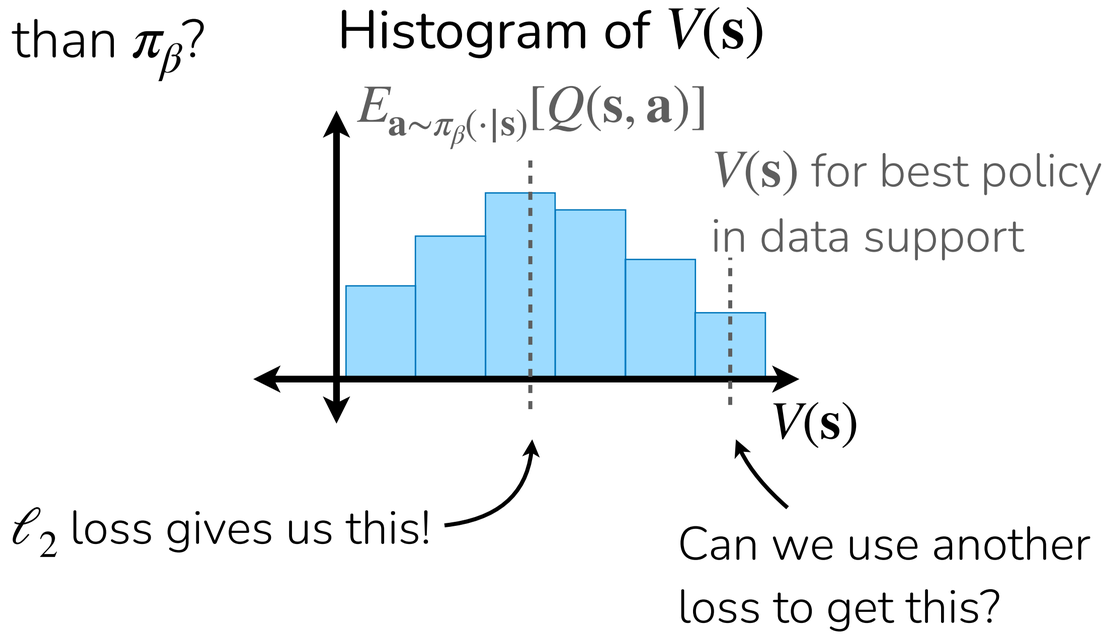
TD update for Q^{π_β}: $\hat{Q}^{\pi_\beta} \leftarrow \arg\min_Q E_{(\mathbf{s},\mathbf{a},\mathbf{s}',\mathbf{a}')\sim D}\left[\left(Q(\mathbf{s},\mathbf{a}) - \left(r + \gamma Q(\mathbf{s}',\mathbf{a}')\right)\right)^2\right]$

a sample of $V^{\pi_\beta}(\mathbf{s}')$

Can we estimate Q for a policy that is better than π_β?

Idea: Use an asymmetric loss function



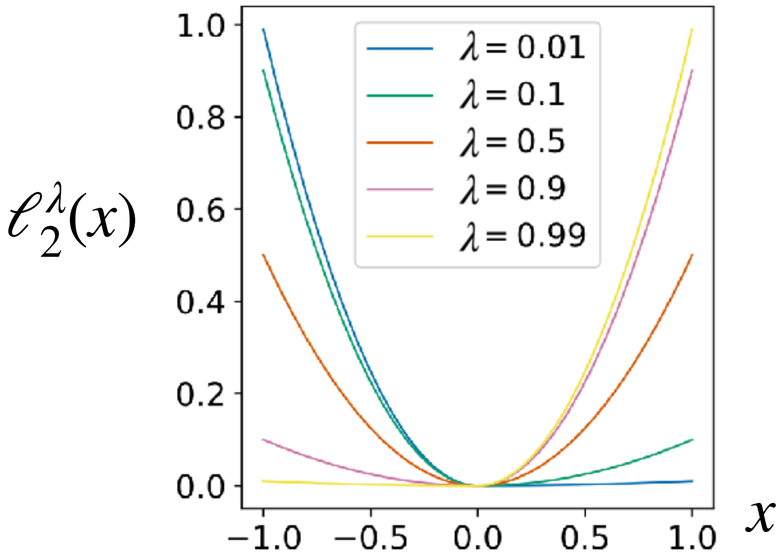


Aside: Expectile regression

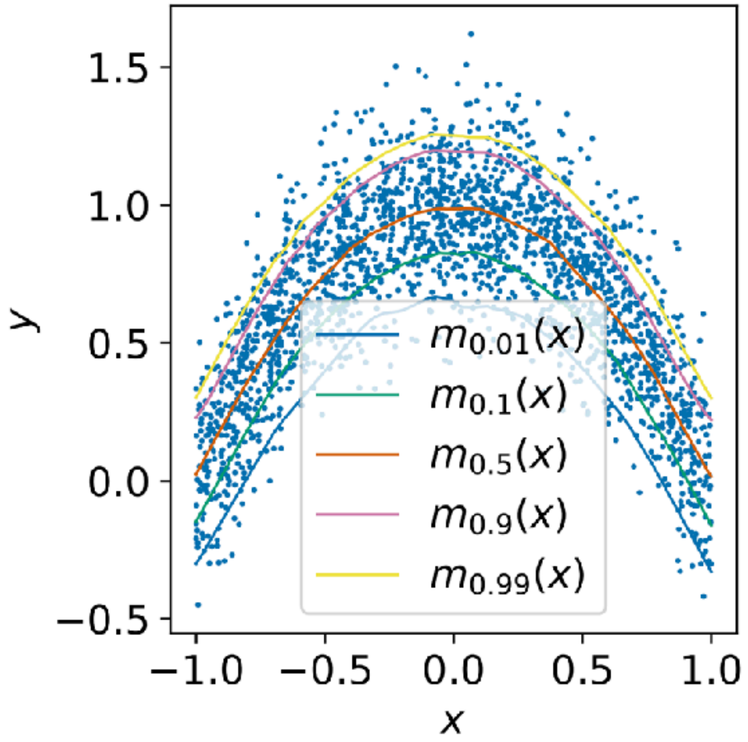
Instead of getting the mean of a random variable, can we get a higher or lower expectile?

Expectile regression loss:

$$\ell_2^\lambda(x) = \begin{cases} (1-\lambda)x^2 & \text{if } x < 0 \\ \lambda x^2 & \text{otherwise} \end{cases}$$



Example with a 2D random variable



Kostrikov, Nair, Levine. Implicit Q-Learning. ICLR ‘22

Can we do better?

Want to estimate advantages using TD updates, without querying Q on OOD actions.

Full algorithm

Fit V with expectile loss: $\hat{V}(\mathbf{s}) \leftarrow \arg\min_V E_{(\mathbf{s},\mathbf{a})\sim D}\left[\ell_2^\lambda\left(V(\mathbf{s}) - \hat{Q}(\mathbf{s},\mathbf{a})\right)\right]$ using small $\lambda < 0.5$

Update Q with typical MSE loss: $\hat{Q}(\mathbf{s},\mathbf{a}) \leftarrow \arg\min_Q E_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim D}\left[\left(Q(\mathbf{s},\mathbf{a}) - \left(r + \gamma\hat{V}(\mathbf{s}')\right)\right)^2\right]$

Extract policy with AWR: $\hat{\pi} \leftarrow \arg\max_\pi E_{\mathbf{s},\mathbf{a}\sim D}\left[\log\pi(\mathbf{a}\,|\,\mathbf{s})\exp\left(\frac{1}{\alpha}\left(\hat{Q}(\mathbf{s},\mathbf{a}) - \hat{V}(\mathbf{s})\right)\right)\right]$

+ Never need to query OOD actions!

+ Policy (still) only trained on actions in data.

+ Decoupling actor & critic training —> computationally fast

policy improvement is implicit

-> implicit Q-learning (IQL)

You will implement IQL in homework 3!

Kostrikov, Nair, Levine. Implicit Q-Learning. ICLR ‘22

The plan for today

Offline RL

1. Why offline RL?
2. Data stitching example
3. Core method ideas
    1. Constraining policy to actions in the data
    2. Estimating better value functions without OOD queries

Part of homework 3!

Key learning goals:

- the key challenges arising in offline reinforcement learning
- two core techniques for offline RL (& why they work!)
- how offline RL can improve over imitation learning

Summary

Why offline RL? Online data is expensive. *Reusing offline data* is good!

Key challenge: Overestimating Q-values because of shift between π_β and π_θ

Techniques:

1. filtered or weighted imitation learning is a simple baseline
2. **implicitly constrain** the policy to π_β by only supervising on actions in data
3. use an asymmetric loss to estimate value for policies better than π_β

Trajectory stitching allows offline RL methods to improve over imitation.

Next time

Friday lecture: How do we get rewards??

Course reminders

Project

- Survey due today
- CA mentors assigned soon after
- Proposal due next Weds

graded fairly lightly- it's for your benefit!

Homework

- Homework 2 due Friday (start early!)