CS 4/5789 2 March 2022 Prof Sarah Dean

Lecture 11: Approximate & Conservative Policy Iteration

Setting: MDP $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, P, r, \gamma\}$

unknown!

Last lecture, we considered Model-based RL. In MBRL, we learn $\hat{P}$ from data, and then use it to design $\hat{\pi}$. Now, we consider Approximate Dynamic Programming methods: we will learn the Value and/or Q function from data instead.

Meta Algorithm: ADP

For $i = 1, 2, \dots$

1) $\hat{Q}^i = \text{SAMPLEANDEVAL}(\hat{\pi}^i)$
2) $\hat{\pi}^{i+1} = \text{IMPROVE}(\hat{Q}^i)$

1) Supervision via Rollouts

Today, we focus on a method for approximating Q^π via rollout-based supervision. Recall that

$$Q^\pi(s,a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,\middle|\, \begin{matrix} s_0, a_0 = s, a \\ P, \pi \end{matrix}\right]$$

Alg: Infinite Rollout(s, a, π)

$s_0 = s$, $a_0 = a$

for $t = 0, 1, \dots$

take action a_t, observe $r_t = r(s_t, a_t)$, $s_{t+1} \sim P(s_t, a_t)$

update $a_{t+1} = \pi(s_{t+1})$

return $y = \sum_{t=0}^{\infty} \gamma^t r_t$

We have $\mathbb{E}[y] = Q^{\pi}(s,a)$. useful as a label for supervised learning! But takes infinite time ...

Another try:

Alg: Rollout with breaks (s, a, π)

$s_0 = s$, $a_0 = a$

for $t = 0, 1, \ldots$

  take action a_t & observe $r_t = r(s_t, a_t)$, $s_{t+1} \sim P(s_t, a_t)$

  With probability $1-\gamma$:

    Break and return $y = \sum_{k=0}^{t} r_k$

  update $a_{t+1} = \pi(s_{t+1})$

Now what is $\mathbb{E}[y]$?

probability of returning	r_0	is	$1-\gamma$
	$r_0 + r_1$	is	$\gamma(1-\gamma)$
	$r_0 + r_1 + r_2$	is	$\gamma^2(1-\gamma)$
	$\sum_0^t r_k$	is	$\gamma^t(1-\gamma)$

So
$$\begin{aligned}
\mathbb{E}[y] &= (1-\gamma) r_0 + \gamma(1-\gamma)(r_0 + r_1) + \gamma^2(1-\gamma)(r_0 + r_1 + r_2) + \ldots \\
&= r_0 (1-\gamma) \sum_{t=0}^{\infty} \gamma^t + r_1 (1-\gamma) \gamma \sum_{t=0}^{\infty} \gamma^t + \ldots \\
&= r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \\
&= \sum_{t=0}^{\infty} \gamma^t r_t
\end{aligned}$$

Also a useful label for supervised learning!

Dataset: $\{(\underbrace{s_i, a_i}_{\text{features}}, \underbrace{y_i}_{\text{label} \approx Q^{\pi}(s_i, a_i)})\}$

But how should we choose (s_i, a_i) to sample from?

Recall: prediction error guarantee for supervised learning with $x \sim D_x$, $y = f_*(x) + w$

$$\mathbb{E}_{x \sim D_x}\left[(f_*(x) - \hat{f}(x))\right] \leq \varepsilon \quad (\text{usually } \mathcal{O}(1/\sqrt{N}))$$

What distribution do we want to estimate Q^π over?

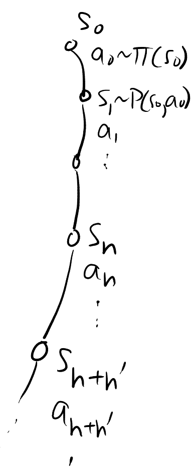
The discounted state-action distribution $d^\pi_{\mu_0}$!
We will sample $(s,a) \sim d^\pi_{\mu_0}$ using a similar idea.

Algorithm: Sample (π):

sample $a_0, s_0 \sim \mu_0$
for $t = 0, 1, \ldots$
  Take action a_t, observe $s_{t+1} \sim P(s_t, a_t)$
  with probability $1-\gamma$:
    break and return a_t, s_t
  update $a_{t+1} = \pi(s_{t+1})$

Notice that both algorithms use rollouts under policy π.
We call this "on policy" - using data collected with π to estimate Q^π.

Furthermore, the process of constructing s, a, y can be approximated by a single trace $\{(s_t, \pi(s_t), r_t)\}_t$ by appropriately re-indexing & weighting



1) sample $h \propto \gamma^h \rightarrow s_h, a_h$,
2) sample $h' \propto \gamma^{h'} \rightarrow y = \sum_{t=h}^{h+h'} r_t$

Methods that construct labels from long rollouts are called MCMC (Markov chain Monte Carlo)

2) Approximate Policy Iteration

Putting together the pecies, our rollout-based regression approach is defined as

Alg: ROLLOUTEVAL (π):

for $i=1,\dots,N$:

  $s_i, a_i = \text{SAMPLE}(\pi)$

  $y_i = \text{ROLLOUTWITHBREAKS}(s_i, a_i, \pi)$

$\hat{Q}^{\pi} = \underset{Q \in \mathcal{Q}}{\operatorname{argmin}} \sum_{i=1}^{N} (Q(s_i,a_i) - y_i)^2$ ← Empirical risk minimization with squared loss

some function class – e.g. neural networks

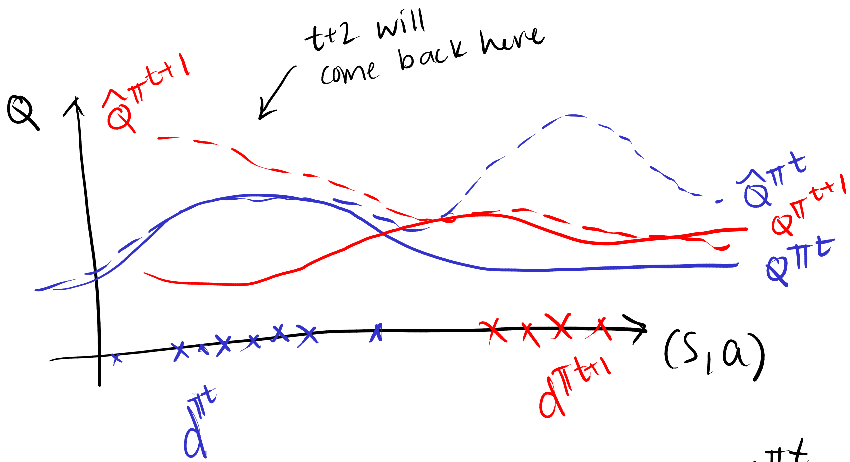
Alg: APPROXIMATE POLICY ITERATION

for $t=0,1,\dots$

  $\hat{Q}^{\pi_t} = \text{ROLLOUTAPPROX}(\pi_t)$ ← regression-based

  $\pi_{t+1}(s) = \underset{a}{\operatorname{argmax}}\ \hat{Q}^{\pi_t}(s,a)$ ← Policy improvement same as PI

Recall: For Policy Iteration, we proved monotonic improvement, i.e. that $V^{\pi_{t+1}}(s) \geq V^{\pi_t}(s)\ \forall s$. Is the same true for Approx policy iteration?



oscillation!

Our estimates are only good on d^{π_t} which might be very different from $d^{\pi_{t+1}}$!

3) Performance Difference Lemma

Goal: Understand V^π vs. $V^{\pi'}$ in terms of the difference between π vs. π'.

Lemma (Performance Difference):

$$V^\pi(s_0) - V^{\pi'}(s_0) = \frac{1}{1-\gamma} \mathbb{E}_{s \sim d^\pi_{s_0}} \Big[\underbrace{\mathbb{E}_{a \sim \pi(s)} \big[Q^{\pi'}(s,a) \big] - V^{\pi'}(s)}_{A^{\pi'}(s,a)} \Big]$$

For $r(s,a) \in [0,1]$,

$$|V^\pi(s_0) - V^{\pi'}(s)| \leq \frac{1}{(1-\gamma)^2} \mathbb{E}_{s \sim d^\pi_{s_0}} \Big[\sum_{a \in \mathcal{A}} \underbrace{|\pi(a|s) - \pi'(a|s)|}_{\|\pi(\cdot|s) - \pi'(\cdot|s)\|_1} \Big]$$

The first expression inspires us to define

Def (Advantage) $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$

The "advantage" of taking action a at state s rather than following π.

Notice that $A^\pi(s, \pi(s)) = 0$.

Also notice $\arg\max_a A^\pi(s,a) = \arg\max_a Q^\pi(s,a)$

Proof of PDL:

$$V^\pi(s_0) - V^{\pi'}(s_0) = V^\pi(s_0) - \mathbb{E}_{a_0\sim\pi(s_0)}\Big[r(s_0,a_0) + \gamma \mathbb{E}_{s_1\sim P(s_0,a_0)}[V^{\pi'}(s_1)]\Big] + \mathbb{E}_{a_0\sim\pi(s_0)}\Big[r(s_0,a_0) + \gamma \mathbb{E}_{s_1\sim P(s_0,a_0)}[V^{\pi'}(s_1)]\Big] - V^{\pi'}(s_0)$$

$$= \gamma \mathbb{E}_{\substack{a_0\sim\pi(s_0)\\ s_1\sim P(s_0,a_0)}}\left[V^\pi(s_1) - V^{\pi'}(s_1)\right] + \mathbb{E}_{a_0\sim\pi(s_0)}\left[Q(a_0,s_0) - V^{\pi'}(s_0)\right]$$

The first statement in the lemma follows by iteration (similar to Simulation Lemma)

$$\mathbb{E}_{a\sim\pi(s)}\left[Q^{\pi'}(s,a) - V^{\pi'}(s)\right] = \mathbb{E}_{a\sim\pi(s)}\left[Q^{\pi'}(s,a)\right] - \mathbb{E}_{a\sim\pi'(s)}\left[Q^{\pi'}(s,a)\right]$$

$$= \sum_{a\in\mathcal{A}} \left(\pi(a|s) - \pi'(a|s)\right) Q^{\pi'}(s,a)$$

Therefore,

$$|V^\pi(s_0) - V^{\pi'}(s_0)| \le \frac{1}{1-\gamma} \mathbb{E}_{s\sim d^\pi_{s_0}}\left[\sum_{a\in\mathcal{A}} |\pi(a|s) - \pi'(a|s)|\, Q^{\pi'}(s,a)\right]$$

The second statement follows by noting $0 \le Q^{\pi'}(s,a) \le \frac{1}{1-\gamma}$ $\square$

We can use the PDL to prove monotonic improvement of policy iteration (HW2).

$$V^{\pi^{t+1}}(s_0) - V^{\pi^t}(s_0) = \frac{1}{1-\gamma} \mathbb{E}_{s\sim d^{\pi^{t+1}}_{s_0}}\left[A^{\pi^t}(s, \pi^{t+1}(s))\right]$$

4) Conservative Policy Iteration

The trouble with Approx Policy Iteration is the potential difference between $d^{\pi^t}_{\mu_0}$ & $d^{\pi^{t+1}}_{\mu_0}$. In CPI, we control this by only incrementally updating the policy.

Alg Conservative Policy Iteration:

for $t = 0, 1, \ldots$

$$\hat{Q}^{\pi_t} = \text{ROLLOUT EVAL}(\pi_t)$$

$$\pi'(s) = \underset{a}{\operatorname{argmax}}\ \hat{Q}^{\pi_t}(s,a)$$

$$\pi^{t+1}(\cdot|s) = (1-\alpha)\pi^t(\cdot|s) + \alpha\pi'(\cdot|s)$$

stochastic policies

incremental update controlled by stepsize $\alpha \in [0,1]$

Another way to veiw the incremental update:

$$\pi^{t+1}(\cdot|s) = \pi^t(\cdot|s) + \alpha\big(\underbrace{\pi'(\cdot|s) - \pi^t(\cdot|s)}_{\text{"error"}}\big)$$

CPI has proveable properties

1) $d^{\pi^{t+1}}_{\mu_0}$ and $d^{\pi^t}_{\mu_0}$ are close
2) Expected improvement:

$$\mathbb{E}_{s\sim\mu_0}\left[V^{\pi^{t+1}}(s) - V^{\pi^t}(s)\right] \geq 0$$