EE 290 Theory of Multi-armed Bandits and Reinforcement Learning Lecture 20 - 04/01/2021

Lecture 20: Value Iteration and Q-learning

Lecturer: Jiantao Jiao *Scribe: Chinmay Maheshwari, Sandy Tanwisuth*

1 Recap from last lecture

Recall that last time we started the discussion on the offline learning. We constructed the *plug-in* estimators of transition and reward functions. The main result stated that if the difference between the actual rewards and transitions from their plug-in estimators is small then the difference between the value function of the actual system and the "plug-in" system for any policy π is small. We gave upper bounds on the difference of value functions.

2 Introduction

In this lecture, we analyze the offline version of a popular RL algorithm *Q-learning.* We analyze the performance of this algorithm on one of the simplest RL formulations, namely *fixed-horizon time varying transition and reward* setting. Further discussion and references, see [1].

3 Setting

We denote the state space by $\mathcal{S}$ while the action space is denoted by $\mathcal{A}$. The time horizon for the problem is denoted by H. For any time $h \in [H]$ the probability of transitioning to state $s' \in \mathcal{S}$ on taking action $a \in \mathcal{A}$ from state $s \in \mathcal{S}$ is denoted by $\mathbb{P}_h(s'|s,a)$. Naturally, for all $(s,a,h) \in \mathcal{S} \times \mathcal{A} \times [H]$ we have $\sum_{s' \in \mathcal{S}} \mathbb{P}_h(s'|s,a) = 1$. To tackle the fixed horizon time varying transitions and reward problem we work with an augmented state space, namely $\mathcal{S}_{\mathsf{aug}} \coloneqq \{(s,h) : s \in \mathcal{S}, h \in [H]\}$. This ensures that the state space across time horizon are disjoint. Note that at any time $h \in [H]$ the states only lie in the set $\mathcal{S}_h \coloneqq \{(s,h) : s \in \mathcal{S}\}$. Thus for any $h, h' \in [H]$ such that $h \neq h'$, $\mathcal{S}_h \bigcap \mathcal{S}_{h'} = \varnothing$.

Note that we can equivalently consider the time-varying transitions on the state space $\mathcal{S}$ as *time-invariant* transition on the augmented state space $\mathcal{S}_{\mathsf{aug}}$. In particular, let $\mathbb{P}$ denote the time invariant transition on $\mathcal{S}_{\mathsf{aug}}$ which is defined as

$$\mathbb{P}((s',h')|(s,h),a) \coloneqq \mathbb{P}_h(s'|s,a)\mathbb{I}(h' = h+1),$$

where $\mathbb{I}(h' = h+1)$ is an indicator random variable which is 0 everywhere except on the event that $h' = h+1$ on which it takes value 1. This is the formulation that should be kept at the back of our mind for the remaining lecture.

4 Algorithms for fixed horizon, time-variant transition & reward

Before proceeding to Q-learning, we see how does plug in based approach discussed last time look like when applied to fixed-horizon time varying transition and reward setting. In addition, we also work with the uniform coverage assumption as in the last lecture.

4.1 Version of Value iteration For the ease of presentation, we assume that the rewards are known and deterministic. This is because the plug in estimator for transition takes a lot of sample to converge close enough to the true transition as compared to the empirical average of rewards. We denote the reward obtained at time $h \in [H]$ in state $s \in \mathcal{S}$ on taking action $a \in \mathcal{A}$ by $r_h(s,a) \in [0, R_{\max}]$ where $R_{\max}$ is the upper limit on the reward.

For all $(s,h,a) \in \mathcal{S} \times [H] \times \mathcal{A}$ we construct the plug in estimator for transition function by sampling N iid states $\{s'_1, s'_2, \ldots, s'_N\}$ from the transition function $\mathbb{P}_h(\cdot|s,a)$. The estimator is denoted by

$$\hat{\mathbb{P}}_h(s'|s,a) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(s' = s'_i)$$

Using the above plug in estimator and the reward function we compute the value function at all times using finite horizon dynamic programming principle [2, Chapter 4]. That is, for all $(s,h,a) \in \mathcal{S} \times [H] \times \mathcal{A}$ we have

$$\hat{Q}_h(s,a) = r_h(s,a) + \hat{\mathbb{P}}_h \hat{V}_{h+1}(s,a)$$

where for every $s \in \mathcal{S}$

$$\hat{V}_h(s) = \max_{a \in \mathcal{A}} \hat{Q}_h(s,a).$$

We initialize the value function such that $\hat{Q}_{H+1}(s,a) = 0$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$.

If $N = \infty$ then $\hat{\mathbb{P}}_h = \mathbb{P}_h$ and if we initialize $\hat{Q}_{H+1}(s,a) = 0$ for all s,a then we can recover the exact Q-function as the iterations of the above algorithm then become exactly the dynamic programming recursion for the actual system. The performance guarantees for the above algorithm can be found in [3].

4.2 Q-learning

We now start with the Q-learning algorithm. The main highlight of this method is that you do not need store the plug-in estimator $\hat{\mathbb{P}}_h$. Therefore there is no compulsion to learn the true transition matrix we only aim to obtain the best policy. Thus there will be definitely less computational storage as compared to the plug in approach. The idea of Q learning is to sample from distribution and then use gradient type update. The main question is – does this converge? Before answering that we first mention the Q-learning algorithm

In this algorithm, at every iteration we maintain an estimate of the Q-function for all time steps. Moreover, at every iteration of the algorithm we draw a sample from $\mathbb{P}_h(\cdot|s,a)$ for all $(s,h,a) \in \mathcal{S} \times [H] \times \mathcal{A}$ and use that to update the Q-functions. The update law in the iteration t of the algorithm is the following:

$$Q_h^{(t)}(s,a) = (1-\alpha_t)Q_h^{(t-1)}(s,a) + \alpha_t(r_h(s,a) + V_{h+1}^{(t-1)}(s'_t)), \qquad \forall (s,h,a) \in \mathcal{S} \times [H] \times \mathcal{A} \tag{1}$$

where α_t is the step size and $s'_t \sim \mathbb{P}_h(\cdot|s,a)$. We now present the pseudo code below

Algorithm 1:

```
Input: $\{\alpha_t\}, N, Q_h^{(0)}(s,a) = 0$ for all $(s,h,a) \in \mathcal{S} \times [H+1] \times \mathcal{A}$;
for $t = 1, 2, \ldots, N$ do
    for $h = H, H-1, \ldots, 1$ do
        for $(s,a) \in \mathcal{S} \times \mathcal{A}$ do
            sample state $s'_t \sim \mathbb{P}_h(\cdot|s,a)$ ;
            Update $Q_h^{(t)}(s,a) = (1-\alpha_t)Q_h^{(t-1)}(s,a) + \alpha_t(r_h(s,a) + V_{h+1}^{(t-1)}(s'_t))$;
            Compute $V_h^{(t)}(s,a) = \max_{a \in \mathcal{A}} Q_h^{(t)}(s,a)$;
        end
    end
end
```

The above form of Q-learning was recursive. We now focus on non-recursive form of Q-learning. Note that the recursive formula can be equivalently written as

$$Q_h^{(t)}(s,a) = \sum_{i=1}^{t} \gamma_t^{(i)}(r_h(s,a) + V_{h+1}^{(i-1)}(s'_i))$$

where

$$\gamma_t^{(i)} := \alpha_i \prod_{j=i+1}^{t} (1-\alpha_j)$$

One can verify that $\sum_{i=1}^{t} \gamma_t^{(i)} = 1$(refer [4, page 7]). Using this observation we can rewrite the non-recursive form of update as

$$Q_h^{(t)}(s,a) = r_h(s,a) + \sum_{i=1}^{t} \gamma_t^{(i)} V_{h+1}^{(i-1)}(s_i').$$

One more interesting observation of this iteration is that if we choose $\alpha_t = \frac{1}{t}$ then $\gamma_t^{(i)} = \frac{1}{t}$ which is independent of i. This means that the contribution in final Q-function of each value function is same. We can control how much contribution each value function in the past iterate has to make to current Q-function by choosing $\gamma_t^{(i)}$ (actually $\alpha_t^{(i)}$) appropriately. So large values of $\gamma_t^{(i)}$ for smaller i means we are giving more weight to initial iterates while if $\gamma_t^{(i)}$ is large for large i then this means we are giving more weight to more recent iterates. Ideally, we want to strike a balance. So the above observation hints that a good learning rate should be around $\frac{1}{t}$. Indeed, in our analysis we work with similar step size.

Theorem 1. *Choose $\alpha_t = \frac{H+1}{H+t} > \frac{1}{t}$. Let the iterates of the algorithm be denoted by $\{Q_h^{(t)}\}_{h\in[H],t\in[N]}$ and let the actual Q-function be denoted by $\{Q_h^{(*)}\}_{h\in[H]}$. Then with probability of $1-\delta$, we have*

$$\left\| \frac{1}{t}\sum_{i=1}^{t} Q_h^{(i)} - Q_h^{(*)} \right\|_\infty \le \sqrt{\frac{H^5(\log(\frac{HSA}{\delta}))}{t}}.$$

We introduce the following algebraic results from [4, Lemma 4.1] (without proof) to prove Theorem 1.

Lemma 1. *Let $\alpha_t = \frac{H+1}{H+t}$ then $\gamma_t^{(i)}$ satisfies the following properties:*

$$\sum_{i=1}^{t} \left(\gamma_t^{(i)}\right)^2 \le \frac{2H}{t} \quad (1)$$

$$\sum_{t=i}^{\infty} \gamma_t^{(i)} \le 1 + \frac{1}{H} \quad (2)$$

The first property (1) shows that the overall the ℓ_2 norm is not too large. This is an important property especially when we use Hoeffding's inequality later on in the proof. The second term (2) implies that $\gamma_t^{(i)}$ has a fast decay rate.

Proof of Theorem 1. Essentially, we are trying to analyze the difference between $Q_h^{(j)}(s,a)$ and $Q_h^{(*)}(s,a)$.

$$Q_h^{(j)}(s,a) - Q_h^{(*)}(s,a) = \sum_{i=1}^{j} \gamma_j^{(i)} \left[V_{h+1}^{(i-1)}(s_i') - \left(\mathbb{P}_h V_{h+1}^{(*)} \right)(s,a) \right]$$

We introduced a natural intermediate term $V_{h+1}^{(*)}(s_i')$ to aid the comparison. As a result, the summation terms will be as follows:

$$Q_h^{(j)}(s,a) - Q_h^{(*)}(s,a) = \sum_{i=1}^{j} \gamma_j^{(i)} \left(V_{h+1}^{(i-1)}(s_i') - V_{h+1}^{(*)}(s_i') \right) + \underbrace{\sum_{i=1}^{j} \gamma_j^{(i)} \left(V_{h+1}^{(*)}(s_i') - \left(\mathbb{P} V_{h+1}^{(*)} \right)(s,a) \right)}_{:=\Delta_j}.$$

Notice that the second summation term is the weighted average of i.i.d. random variables with bounded support. Thus, using Hoeffding inequality the second summation term defined as Δ_j is upper bounded by $\sqrt{\sum_{i=1}^{j} (\gamma_j^{(i)})^2 H^2 (\log(\frac{HSA}{\delta}))}$, for each (s,a) with probability of at least $1 - \frac{\delta}{HSA}$, which is less than $\sqrt{\frac{H^3}{j}}$ ignoring some log terms. Note that by union bounds with probability of at least $1-\delta$ for all h,

$$\left\| Q_h^{(j)} - Q_h^{(*)} \right\|_\infty \le \sum_{i=1}^{j} \gamma_j^{(i)} \left\| V_{h+1}^{(i-1)} - V_{h+1}^{(*)} \right\|_\infty + \Delta_j.$$

Again, Δ_j is upper bounded by $\sqrt{\frac{H^3}{j}}$ ignoring the log terms. Moreover $\|V_{h+1}^{(i-1)} - V_{h+1}^{(*)}\|_\infty \le \|Q_{h+1}^{(i-1)} - Q_{h+1}^{(*)}\|_\infty$, the proof of which is given in Lemma 2 at the end of this lecture. By summing both sides over j from 1 to t, we have

$$\begin{aligned}
\sum_{j=1}^{t} \left\| Q_h^{(j)} - Q_h^{(*)} \right\|_\infty &\le \sum_{j=1}^{t} \sum_{i=1}^{j} \gamma_j^{(i)} \left\| Q_{h+1}^{(j-1)} - Q_{h+1}^{(*)} \right\|_\infty + \beta_t \\
&\le \sum_{i=1}^{t} \left(\sum_{j=i}^{t} \gamma_j^{(t)} \right) \left\| Q_{h+1}^{(i-1)} - Q_{h+1}^{(*)} \right\|_\infty + \beta_t.
\end{aligned}$$

Notice that we introduce $\beta_t := \sum_{j=1}^{t} \Delta_j$ which is upper bounded by $\sqrt{H^3 t}$. From Lemma 1, we learned that $\sum_{j=i}^{t} \gamma_j^{(t)}$ is upper bounded by $1 + \frac{1}{H}$. Then, we have

$$\sum_{j=1}^{t} \left\| Q_h^{(j)} - Q_h^{(*)} \right\|_\infty \le \left(1 + \frac{1}{H}\right) \sum_{i=0}^{t-1} \left\| Q_{h+1}^{(i)} - Q_{h+1}^{(*)} \right\|_\infty + \beta_t.$$

When we recursively call H times for all $h \in H$, we can propagate up to 1. First, let $h = H$. Thus, the final bound is

$$\begin{aligned}
\sum_{i=1}^{t} \left\| Q_h^{(i)} - Q_h^{(*)} \right\|_\infty &\lesssim C \left(1 + \left(1 + \frac{1}{H}\right) + \left(1 + \frac{1}{H}\right)^2 + \ldots + \left(1 + \frac{1}{H}\right)^H \right) \left(\sqrt{H^3 t} \right) \\
&\lesssim C \sqrt{H^5 t}
\end{aligned}$$

where C is a contant term. As a result, we have

$$\left\|\frac{1}{t}\sum_{i=1}^{t} Q_h^{(i)} - Q_h^{(*)}\right\|_\infty \leq \frac{1}{t}\sum_{i=1}^{t}\left\|Q_h^{(i)} - Q_h^{(*)}\right\|_\infty \lesssim \sqrt{\frac{H^5}{t}},$$

where the first inequality is due to Jensen's inequality. This completes the proof. □

A Appendix

We present one of the results from analysis which is used in the proof of Theorem 1.

Lemma 2. *Let Q, Q' be two Q-functions such that for all (s,a) we have $Q(s,a), Q'(s,a) \geq 0$ and let V, V' be the corresponding value functions such that for any $s \in \mathcal{S}$, $V(s) = \max_{a\in\mathcal{A}} Q(s,a)$ and $V'(s) = \max_{a\in\mathcal{A}} Q'(s,a)$. Then*

$$\|V - V'\|_\infty \leq \|Q - Q'\|_\infty$$

Proof Note that for any s, a,

$$\begin{aligned} &Q(s,a) = Q(s,a) - Q'(s,a) + Q'(s,a) \\ &\Rightarrow Q(s,a) \leq |Q(s,a) - Q'(s,a)| + Q'(s,a) \end{aligned}$$

where we have used that fact that Q, Q' are always non-negative by definition of reward structure. This means that

$$\max_a Q(s,a) \leq \max_a(|Q(s,a) - Q'(s,a)| + Q'(s,a)) \leq \max_a(|Q(s,a) - Q'(s,a)|) + \max_a Q'(s,a)$$

Thus we obtain the identity

$$\max_a Q(s,a) - \max_a Q'(s,a) \leq \max_a(|Q(s,a) - Q'(s,a)|)$$

We can as well swap the order of Q and Q' thus we have

$$|\max_a Q(s,a) - \max_a Q'(s,a)| \leq \max_a(|Q(s,a) - Q'(s,a)|)$$

Now note that

$$\begin{aligned} \|V - V'\|_\infty &= \max_s |V(s) - V'(s)| \\ &= \max_s |\max_a Q(s,a) - \max_a Q'(s,a)| \\ &\leq \max_s \max_a |Q(s,a) - Q'(s,a)| \\ &= \|Q - Q'\|_\infty \end{aligned}$$

□

References

[1] C. Jin, B. Wang, Y. Yan, and Z. Li, “ELE524: Foundations of Reinforcement Learning – Lecture 6,” *Princeton University*, Spring 2020.

[2] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming.* John Wiley & Sons, 2014.

[3] C. Jin, B. Wang, Y. Yan, and Z. Li, “ELE524: Foundations of Reinforcement Learning – Lecture 5,” *Princeton University*, Spring 2020.

[4] C. Jin, Z. Allen-Zhu, S. Bubeck, and M. I. Jordan, “Is Q-Learning Provably Efficient?” *Advances in Neural Information Processing Systems*, vol. 31, 2018.