Lecture 10: Fundamental limits for linear function approximation

Akshay Krishnamurthy
akshay@cs.umass.edu

April 5, 2022

In the last lecture, we saw two methods for exploration with linear function approximation. The first one, LSVI-UCB operates in the linear/low-rank MDP model and achieves a regret bound of $\text{poly}(H)\sqrt{d^3T}$ over T episodes. This algorithm is based on "local optimism" and is similar to the UCB-VI algorithm for tabular MDPs. The second one doesn't really have a name, and is based on "global optimism." This is more like the confidence ball version of LinUCB and we'll see that this algorithm can be generalized in subsequent lectures. Last time we sketched how this method achieves $\text{poly}(H)\sqrt{d^2T}$ regret under the weaker assumption of "linear Bellman completeness."

While these algorithms represent statistically tractable approaches, the assumptions (linear MDP, or bellman completeness) are quite strong, so we should ask if we can get away with weaker assumptions. In this direction, perhaps the most natural question is to ask what can be done with just linear realizability of $Q^\star$. This is reasonable to ask because linear realizability for $Q^\star$ is sufficient in bandits (as we saw with LinUCB). So we are asking if we can get away with similar function approximation assumptions, while dealing with credit assignment.

The short answer is that not much can be done with just linear realizability, which has been the topic of a number of "lower bounds" papers in the last couple of years. The only two positive results I know are:

1. Finite horizon, deterministic dynamics (and start state), and deterministic reward.
2. Finite horizon, deterministic dynamics (and start state), stochastic reward, with a "gap" between the Q-values of the optimal and suboptimal actions.

The upper bounds are fairly simple and rely on *exactly* identifying $Q^\star$ or $\pi^\star$ on some states and actions. Today we will sketch one such algorithm. In terms of lower bounds there are many (this list may not be exhaustive):

1. Offline RL, discounted setting, coverage in the second moment sense
2. Offline RL, episodic setting, coverage in the second moment sense
3. Offline RL, coverage in the density ratio sense.
4. Online RL, exponentially many actions, stochastic dynamics, gaps
5. Online RL, polynomially many actions.

The key mechanisms are instability of bellman backup operator and the ability to hide a vector in an exponentially large set. We will discuss some of these mechanisms today as well.

1 Upper bounds

In this section, we assume that $Q^\star_h(s,a) = \langle \theta^\star_h, \phi(s,a)\rangle$. The upper bound for working with linear realizability in deterministic systems is based on a simple depth first search algorithm. The idea is to maintain a subspace $B_h \subset \mathbb{R}^d$ for each layer h on which we know $Q^\star_h$ *exactly*. More precisely, we maintain a set of linear equations for $Q^\star_h$. We represent B_h by a set of linearly independent vectors with regression targets $\{(\psi_1, y_1), \ldots, (\psi_{d'}, y_{d'}\}$ where $d' = \dim(B_h) \le d$ and y_i is the $Q^\star_h$ function for feature ψ_i, $y_i = \langle \theta^\star_h, \psi_i\rangle$. Implicitly we then let $\hat{\theta}_h$ be any vector that satisfies the linear equations $\left\langle \hat{\theta}_h, \psi_i \right\rangle = y_i$. Since y_i is the $Q^\star_h$ value for feature ψ_i, due to linear realizability,

Algorithm 1.1 DFS algorithm for deterministic dynamics/rewards and linear $Q^\star$.

Global state: $B_{0:H-1}$ initialized to empty sets (implicitly $\hat{\theta}_h$ is any vector satisfying linear equations in B_h).
Time step h, and an action sequence $a_{0:h-1}$ to reach some state s_h.
If $h = H$, return $V_H^\star(s_H) = 0$
while there exists $a_h \in \mathcal{A}$: $\phi(s_h, a) \notin \text{span}(B_h)$ **do**
  Let s_{h+1} be the state reached by executing $a_{0:h}$, recursively learn $V_{h+1}^\star(s_{h+1})$.
  Update $B_h \leftarrow B_h \cup \{(\phi(s_h, a_h), r(s_h, a_h) + V_{h+1}^\star(s_{h+1}))\}$.
end while
return $V_h^\star(s_h) = \max_a \left\langle \hat{\theta}_h, \phi(s_h, a_h) \right\rangle$

we can impute $Q_h^\star$ on any other vector ϕ, in the subspace. In particular, there exists $\alpha_1, \ldots, \alpha_{d'} \in \mathbb{R}$ such that $\phi = \sum_{i=1}^{d'} \alpha_i \psi_i$. On the other hand we should not try to "extrapolate" to ϕ outside of the subspace.

We can find this basis using a recursive depth first search procedure, displayed in Algorithm 1.1. The algorithm tries to compute the $V^\star$ function for a single state s. To do so it makes sure that all features $\phi(s, a)$ are in the subspace for the current level and if they are it simply outputs the inputed value. If some features are not in the subspace it recursively asks for the value function of the subsequent states and adds these features in.

The correctness of the algorithm under deterministic dynamics and rewards is straightforward and we can see that the number of episodes required is $O(dH)$. This is because at each level we will make at most d recursive calls. However, note that it uses determinism of both dynamics and rewards in a critical way. With stochastic rewards we may not be able to guarantee that our regression targets $r(s_h, a_h) + V_{h+1}^\star(s_{h+1})$ are exactly equal to $Q_h^\star(s_h, a_h)$, since we will have some statistical noise. Stochastic dynamics are more problematic, since we may not be able to return to any given state s_h which is important in the "backtracking" part of the recursion.

Stochastic rewards can be handled under a "gap" assumption, via a simple modification of this approach. The gap assumption is that $Q_h^\star(s, \pi_h^\star(s)) \geq \max_{a \neq \pi_h^\star(s)} Q_h^\star(s, a) + \Delta$ for some $\Delta > 0$. Here, although we cannot learn $V^\star$ exactly, we can learn $\pi^\star$ exactly. So rather than do "dynamic programming" backups, we can instead track the optimal actions and use the roll-outs to get unbiased estimates for $Q_h^\star$. Essentially, to get $Q_h^\star(s_h, a_h)$ we first ensure that we know the optimal actions from (s_h, a_h) to the end and then we collect $O(1/\Delta^2)$ samples by rolling into (s_h, a_h) and rolling out with this optimal policy.

2 Lower bounds in offline RL

I find it easier to understand the lower bound mechanisms in offline RL. Since we have not discussed this problem before, let us go over the setup. In (discounted) offline RL, there is an unknown MDP $M = (\mathcal{S}, \mathcal{A}, P, R, \mu, \gamma)$, but we will not get to interact with it. Instead we will receive a dataset $D = \{(s_i, a_i, r_i, s_i')\}_{i=1}^n$ sampled via the process: $(s_i, a_i) \sim \rho, r_i \sim R(s_i, a_i), s_i' \sim P(s_i, a_i)$ where ρ is some *data collection distribution*. Using this dataset we would like to output a policy $\hat{\pi}$ that is near-optimal for the MDP. An even simpler problem is offline policy evaluation where we have a fixed target policy π and we simply want to estimate Q^π, (say we measure the error by $|\hat{Q}^\pi(s^\star, a^\star) - Q^\pi(s^\star, a^\star)|$ where $s^\star, a^\star$ are known as well).

(You will study some closely related problems in the homework.)

The first observation when thinking about these problems is that you will need ρ to "cover the states" in some sense. With linear realizability we can draw inspiration for a suitable coverage condition from the $H = 1$ case or linear contextual bandits. Here the offline dataset is just $\{(s_i, a_i, r_i)\}_{i=1}^n$ (since there is no subsequent state) and we assume that $\mathbb{E}[r|s, a] = \langle \phi(s, a), \theta^\star \rangle$ for some known features with $\|\phi(s, a)\|_2 \leq 1$. This is the analog of linear realizability of $Q^\star$. Given this dataset, maybe the only thing we can really do is linear regression onto the rewards. If we define $\Sigma_\rho = \mathbb{E}_{s,a \sim \rho}[\phi(s, a)\phi(s, a)^\top]$ then linear regression ensures that our estimate $\hat{\theta}$ satisfies $\|\hat{\theta} - \theta^\star\|_{\Sigma_\rho}^2 \leq O(d/n)$. Unfortunately, this may not tell us anything about $Q^\star(s^\star, a^\star)$ if Σ_ρ is rank-degenerate and $\phi(s^\star, a^\star)$ is orthogonal to its column space. Indeed, essentially the best bound we can get is

$$|\hat{Q}^\star(s^\star, a^\star) - Q^\star(s^\star, a^\star)| = |\left\langle \phi(s^\star, a^\star), \hat{\theta} - \theta^\star \right\rangle| \leq \|\phi(s^\star, a^\star)\|_{\Sigma_\rho^{-1}} \cdot \|\hat{\theta} - \theta^\star\|_{\Sigma_\rho},$$

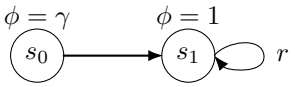

Figure 1: The construction of Amortila-Jiang-Xie.

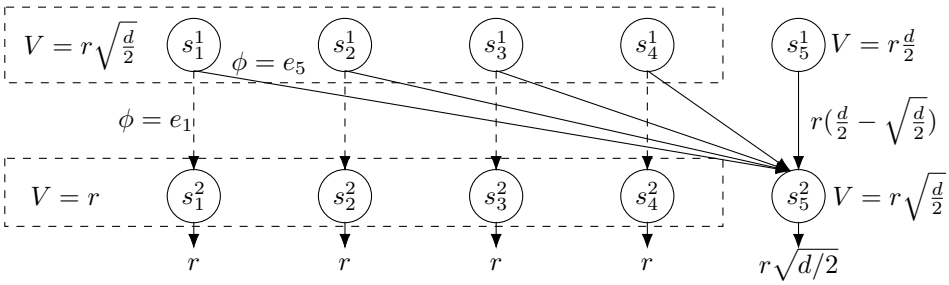

Figure 2: The construction of Wang-Foster-Kakade.

but the "distribution shift coefficient" $\|\phi(s^\star, a^\star)\|_{\Sigma_\rho^{-1}}$ may not be small. So, one coverage assumption is simply that this coefficient is bounded. A stronger assumption is that $\lambda_{\min}(\Sigma_\rho) \geq 1/d$ (which is as large as it can be due to feature normalization), which implies that this coefficient is bounded.

Definition 1. *We say that an offline RL problem has coverage in the second moment sense if $\lambda_{\min}(\Sigma_\rho) \geq c/d$, where d is the feature dimension and c is some constant.*

Let us now show how value functions may not even be identifiable for the discounted offline RL problems even if second moment coverage is satisfied. This can be seen by a very simple problem with two states and one feature, visualized in Figure 1. There, when the problem has discount factor $\gamma \in [1/2, 1)$ then it is easy to see that linear realizability holds, as we can take $\theta^\star = r/(1-\gamma)$. Additionally, we can have ρ supported only s_0 and satisfy second moment coverage. But then r will never be observed, which shows that $\theta^\star$ is not identifiable.

In episodic offline RL, it may be more natural to assume we have a distribution ρ_h for each level $h \in [H]$ and ask that second moment coverage holds for all layers. While this precise phenomena can no longer occur, something very similar is possible. The idea is to make the features of a layer be roughly $\{e_1, \ldots, e_d, \mathbf{1}/\sqrt{d}\}$ and have the value function be $r_h\mathbf{1}$. Then the value function on the state with feature $\mathbf{1}$ is a factor of $\sqrt{d}$ larger than the "standard basis" states. We can use this to create a geometric amplification that results in a $\Omega(d^H)$ type lower bound.

The construction with just two layers is visualized in Figure 2. In the construction we take $d = 8$, and set the features of the states $s_{1:4}$ to be the standard basis elements (say $e_1, \ldots, e_4$ correspond to the dashed actions and $e_{5:8}$ to the solid actions). Then we set the features for state s_5 to be $\phi(s_5, a) = \sum_{j=5}^{8} e_j/\sqrt{\frac{d}{2}}$. By playing with the rewards we can make it so that $V(s_5)$ is a factor of $\sqrt{d/2}$ larger than the value of the other states. Stacking this construction we can make it so the value function is roughly $r(d/2)^{H/2}$ at the first time step.

The issue is that the offline data distribution will not cover the special state s_5, and instead will be uniform on the standard basis elements. This is problematic since, although there is a large immediate reward after s_5, we will never see it. Instead we have to rely on the rewards at the last layer to learn r, but we can make r exponentially small (and noisy) while keeping the value function of constant magnitude.

Mechanism: error amplification. The key mechanism here is error amplification under bellman backups. Indeed, suppose we knew $\hat{r}$ satisfying $|r - \hat{r}| \leq \epsilon$. Then at the last layer we know the Q function with error $\epsilon\sqrt{d/2}$, and in particular we would set $\hat{V}(s_5^2) = \hat{r}\sqrt{d/2}$. To estimate the Q function at the second-to-last layer the "population" regression solution would yield that $\hat{\theta}_1[1], \ldots, \hat{\theta}_1[4] = \hat{r}$ and $\hat{\theta}_1[5], \ldots, \hat{\theta}_1[8] = V(s_5^2) = \hat{r}\sqrt{d/2}$. But note that the error on s_5^1 for this estimate is now $\epsilon(d/2)$, which is multiplicatively larger than at the previous layer!

Remark 2. *Note that the error amplification cannot happen if $\phi(s_5)$ is in the* convex hull *of the other features, rather than simply in the* span *of the other features. Indeed this observation can be turned into an upper bound analysis if the data collection distribution is supported on the vertices of the convex hull containing all the features, but this analysis pays for the number of vertices, which could be exponentially large.*

For an algorithmic perspective on this issue, we can consider value iteration with function approximation in the discounted policy evaluation setting. Let us review the algorithm. To simplify the notation, we can drop the actions entirely, or you can think of replacing $\phi(s, \pi(s)) = \phi(s)$. We start with $\theta^{(0)} = 0$ and we repeat the iteration:

$$\theta^{(t)} \leftarrow \operatorname*{argmin}_{\theta} \sum_{i=1}^{n} \left(\langle \phi(s_i), \theta \rangle - r_i - \gamma \left\langle \phi(s_i'), \theta^{(t-1)} \right\rangle \right)^2 \tag{1}$$

The population regression problem can be written as $\theta^{(t)} = \Sigma_{\text{cov}}^{-1} \theta_r + \gamma \Sigma_{\text{cov}}^{-1} \Sigma_{\text{cr}} \theta^{(t-1)}$ where $\theta_r = \mathbb{E}_\rho[\phi(s, \pi(s)) \cdot r]$, $\Sigma_{\text{cov}} = \Sigma_\rho$ and $\Sigma_{\text{cr}} = \mathbb{E}_{(s,a,s') \sim \rho}[\phi(s, \pi(s))\phi(s', \pi(s'))^\top]$ is the "cross covariance." Unrolling this relationship, we see that if we do T rounds of the population problem we obtain

$$\theta^{(T)} = \Sigma_{\text{cov}}^{-1} \theta_r + \gamma \Sigma_{\text{cov}}^{-1} \Sigma_{\text{cr}} \theta^{(T-1)} = \ldots = \sum_{k=0}^{T} (\gamma \Sigma_{\text{cov}}^{-1} \Sigma_{\text{cr}})^k \Sigma_{\text{cov}}^{-1} \theta_r$$

From here we can obtain a better understanding of what's happening with error amplification: If the eigenvalues of $\gamma \Sigma_{\text{cov}}^{-1} \Sigma_{\text{cr}}$ are greater than 1 we may exponentially amplify any errors we have in estimating θ_r. And this is indeed the issue with the construction in Figure 2, where $\Sigma_{\text{cov}} = I_d/d$ and Σ_{cr} has an eigenvalue with a very large magnitude so that $\lambda_{\max}(\Sigma_{\text{cov}}^{-1} \Sigma_{\text{cr}}) = \Theta(\sqrt{d})$. (A very similar unrolling occurs in the finite horizon setting, without the γ factor.) So this matrix dictates whether error amplification will occur or not for value iteration type methods.

As an aside, if we return to the construction in Figure 1 $\gamma \Sigma_{\text{cov}}^{-1} \Sigma_{\text{cr}} = 1$ but $\theta_r = 0$, so the value iteration solution is just 0. This is wrong! The issue here is that the true value function is not uniquely defined by observable quantities under this offline data distribution.

3 Interlude: Double sampling

Since we have been discussing value-iteration type methods, it is also a good point to introduce another issue with these methods, known as the *double sampling* issue. I have not seen this explicitly appear in any lower bound, but it is a well-documented failure mode for dynamic programming methods in particular.

Consider a discounted MDP and let $\mathcal{T}$ denote the Bellman optimality operator $(\mathcal{T} f)(s, a) := \mathbb{E}[r + \gamma \max f(s', a') \mid s, a]$. Since $Q^\star$ is the solution to the fixed point problem $Q = \mathcal{T} Q$, we could try to estimate $Q^\star$ by solving

$$\hat{Q} \leftarrow \operatorname*{argmin}_{f \in \mathcal{F}} \sum_{i=1}^{n} \left(f(s_i, a_i) - r_i - \gamma \max_{a'} f(s_i', a_i') \right)^2,$$

either with an offline dataset or with data we have collected while exploring. Note that the optimization variable f appears in two places in the objective, so this may not be computationally tractable but it is reasonable to ask if this method has favorable statistical properties.

For a given function f the population objective decomposes into two terms, the bias squared and variance, as:

$$\mathbb{E}\left[\left(f(s, a) - r - \gamma \max_{a'} f(s', a') \right)^2 \right] = \underbrace{\mathbb{E}\left[(f(s, a) - (\mathcal{T} f)(s, a))^2 \right]}_{=:\text{bias}(f)} + \underbrace{\mathbb{E}_{s,a}\left[\text{Var}\left(r + \gamma \max_{a'} f(s', a') \mid s, a \right) \right]}_{=:\text{var}(f)}$$

We know that $Q^\star$ is the only function that has $\text{bias}(Q^\star) = 0$. The issue is that the variance term depends on the function f being considered, and there may be some other function f that actually has a lower population objective, due to having $\text{var}(f) \ll \text{var}(Q^\star)$. (It is easy to construct examples.) This means that $Q^\star$ may not actually be the minimizer of the squared Bellman error objective!

There are essentially three ways to mitigate this. The first is to assume access to a sampler for the MDP which allows you to obtain many samples from the same (s, a), regress onto the "average" next-step value, and therefore substantially shrink the variance term. The second is to assume deterministic dynamics which implies that the variance term does not depend on the function f being considered. The final way is to assume Bellman completeness, which means that $\mathcal{T} f \in \mathcal{F}$, so we can subtract off the variance term by considering the objective

$$\hat{Q} \leftarrow \operatorname*{argmin}_{f \in \mathcal{F}} \max_{g \in \mathcal{F}} \sum_{i=1}^{n} \left(f(s_i, a_i) - r_i - \gamma \max_{a'} f(s_i', a_i') \right)^2 - \sum_{i=1}^{n} \left(g(s_i, a_i) - r_i - \gamma \max_{a'} f(s_i', a_i') \right)^2,$$

In fact, this is essentially what we did in the global optimism algorithm in the previous lecture.

4 Lower bounds in Online RL

The offline RL lower bounds highlight some challenges with error amplification, but they don't imply lower bounds for online exploration. Indeed the construction in Figure 1 is trivial in the online setting since we will visit s_1 and learn the reward. Something similar arises in the construction in Figure 2. Nevertheless, online exploration with only linear realizability is also statistically intractable.

The constructions are much more involved, but one basic mechanism is the ability to pack exponentially many near-orthogonal vectors into Euclidean space. This is formalized in the following lemma, which can be proved by the probabilistic method.

Lemma 3 (Packing lemma)**.** *Fix $d \in \mathbb{N}$ and $\alpha > 0$. If $m \leq \exp(\alpha^2 d/8)$ then there exists m unit vectors $\{v_i\}_{i=1}^m \subset \mathbb{R}^d$ such that $\forall i \neq j : |\langle v_i, v_j \rangle| \leq \alpha$.*

Of course, this packing construction must be used in tandem with some "credit assignment" issue to establish a lower bound, since we can solve any linear stochastic bandit problem. However, when coupled with credit assignment effects, this can lead to an exponential sample complexity lower bound. Let us sketch a construction.

Consider $m + 1$ states and m actions, where $m \asymp \exp(d\alpha^2)$ as in the packing lemma. There is one special "terminal state," which we call f and there is a special action called $a^\star \in [m]$. We set up the transitions so that

$$P(f \mid s = i, a = a^\star) = P(f \mid s = i, a = i) = 1, \qquad P(\cdot \mid s = i, a = j) = \begin{cases} i & \text{w.p. } \langle v_i, v_j \rangle + 2\alpha \\ f & \text{w.p.} 1 - \langle v_i, v_j \rangle - 2\alpha \end{cases}$$

In words, we can immediately terminate from any state $s = i$ if we guess the special action $a^\star$ or if we take the action with the same name as the state. Otherwise we have some large chance of terminating but we continue to some other state at the next level. This is referred to as the "leaking complete graph." We set the rewards so that

$$r(i, a^\star) = \langle v_i, v_{a^\star} \rangle + 2\alpha, \quad r(i, i) = \tfrac{3\alpha}{4}, \quad r(i, j) = -2\alpha \left(\langle v_i, v_j \rangle + 2\alpha \right), \quad V^\star(f) = 0$$

Based on these rewards, and since $|\langle v_i, v_j \rangle| \leq \alpha$, intuitively the optimal thing to do is take $a^\star$, immediately pick up at least α reward, and transit to the terminal state. But this requires guessing $a^\star$ which seems hard to do. Note that, unlike in the bandit setting, choosing the wrong actions here gives you essentially no information since neither the immediate rewards $r(i, j)$ nor the transitions have anything to do with $a^\star$.

The tricky part is to make sure that linear realizability is satisfied. For this we mostly have to worry about $Q^\star(i, j)$ but we can use that $V^\star(j)$ is quite large and in particular depends on $a^\star$. The main computation is

$$\begin{aligned} V^\star(j) &= \langle v_j, v_{a^\star} \rangle + 2\alpha \\ Q^\star(i, j) &= -2\alpha(\langle v_i, v_j \rangle + 2\alpha) + (\langle v_i, v_j \rangle + 2\alpha) V^\star(j) = (\langle v_i, v_j \rangle + 2\alpha) \langle v_j, v_{a^\star} \rangle \end{aligned}$$

Using this, we set $\phi(i, j) = (\langle v_i, v_j \rangle + 2\alpha) v_j$ (and appropriately set $\phi(f, \cdot)$ and $\phi(i, i)$) to satisfy linear realizability. The coefficient vector is essentially $v_{a^\star}$. Crucially, we've made it so that linear realizability holds despite the fact that the immediate rewards reveal nothing about $v_{a^\star}$.

Finally, the lower bound arises by stacking this construction for H layers. The last layer results in a stochastic linear bandit problem, which is problematic, but observe that the probability of getting to a non-terminal state at the last layer is very small, as it is upper bounded by $< (1 - \alpha)^H$. This means that we learn essentially nothing about $a^\star$ unless we guess it exactly correctly or make it to the last layer and results in a $\min(\exp(d), c^H)$ lower bound.