Lecture 6: Value Functions

Admin:

-

1 Story

Polynesians discovered almost every inhabitable piece of land in the Pacific, including ones thousands of miles away. Potentially even South America [2], potentially explaining how sweet potatoes (native to South America) ended up in Polynesian [4]. How did they find islands, which account for a tiny fraction of the land area of the Pacific? Birds (which may fly dozens of miles from land), certain types of clouds that only form over land, using clouds as a mirror, coral reefs, changes in wave point to wave reflections, which are detectable far beyond the point where you loose (visible) sight of land [1].

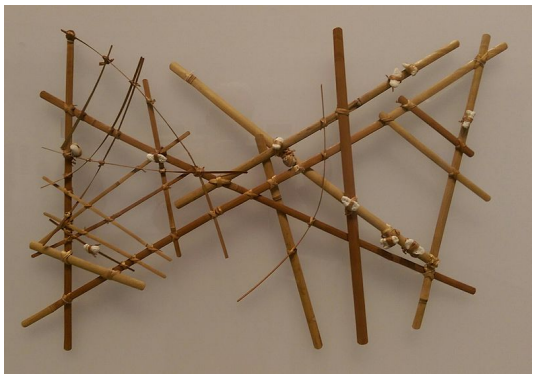


Figure 1: A navigational chart from the Marshall Islands, on display at the Berkeley Art Museum and Pacific Film Archive

2 Review

- Bandits
- RL
- Imitation learning

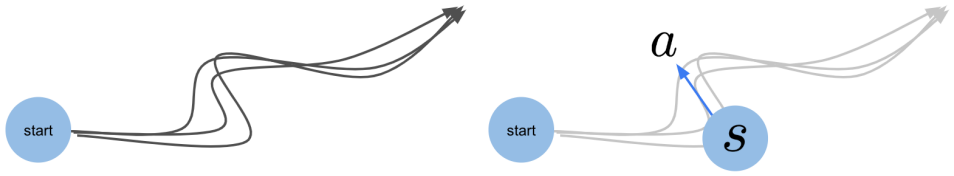
3 Where are we going?

Next two lectures are very closely related, and could really be taught in either order. The big question is "how can we get reward maximizing policies, going beyond just IL?" This is going to require two ingredients:

- Notion of value functions. Namely, a way of predicting the future returns.
- Policy gradient. A way of updating the policy so that it maximizes returns, rather than just mimicking the past data.

3.1 Outline for Today

1. What is the value function?
2. MC estimation
3. Some preliminary ways to use value functions
4. Value functions and dynamic programming
5. Outlook



(a) Estimating the RL objective using samples. (b) Estimating the Q-value

Figure 2: *(Left)* We can estimate the RL objective by adding up the returns on rollouts. *(Right)* Estimating the value of an arbitrary state and action is challenging to do directly using samples. It would require reset access (typically not allowed) or an approximate model of the environment (can be hard to build).

4 Value Functions

Definition: A state-action-conditioned version of the RL objective:

$$\max_{\pi} \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right] \tag{1}$$

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a\right] \tag{2}$$

$$V^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s\right] \tag{3}$$

$$V^{\pi}(s) = \mathbb{E}_{\pi(a|s)}[Q^{\pi}(s,a)]. \tag{4}$$

- names: value, value-function, Q-function Q-values.
- Note the dependence on π

5 Monte Carlo Estimation of Value Functions

The value function looks like the RL objective. We could estimating the RL objective by just sampling. Why can't we do the same for estimating the value function?

- This would require resetting to arbitrary states and actions.
- This could still work if you had a model, but would be noisy and/or computationally expensive.

What about the 1-sample estimate?

- Only well defined on states+actions you've seen before.

Instead, we're going to do something with the following properties:

- Can evaluate on arbitrary states and actions. The hope is generalization. An image classifier that's seen enough examples of cats should be able to accurately detect new cats, unseen in the training data. Similarly, the hope is that after training the value function on a sufficiently broadly set of states and actions, it'll continue to make reliable predictions on new states and actions.
- Less noisy than Monte Carlo estimates.
- Doesn't require a model. It'll be precisely for this reason that methods based on value functions are known as "model-free" methods.

Instead, we'll have to learn something. Just a supervised learning problem:

$$\{(s, a, R = \sum_{t} \gamma^t r_t)\}. \tag{5}$$

Learn via regression:

$$\min_Q \frac{1}{|\mathcal{D}|} \sum_{(s,a,R)} (Q(s,a) - R)^2 \tag{6}$$

$$\min_Q \frac{1}{2} \mathbb{E}_{(s,a,R)} \left[(Q(s,a) - R)^2 \right]. \tag{7}$$

Consistency of the MSE estimator:

$$\frac{d}{dQ(s,a)} = Q(s,a) - \mathbb{E}_{p(R|s,a)}[R] = 0 \tag{8}$$

$$\implies Q(s,a) = \mathbb{E}[R]. \tag{9}$$

- Which value function is this estimating?
- Question (hint to future week): How do you estimate the value function of a different policy?

5.1 Parametrization

- Take action + state as input, produce single scalar value
- With discrete actions, take state as input and produce list of values, one per action

6 How do you use a value function?

This is mostly a preview of things to come, but I want to hint at this to motivate why we care about the value function.

Approach 1. Reward weighted regression [3]. Just behavioral cloning, but weight the samples by the estimated value.

$$\max_\pi \mathbb{E}_{(s,a)}[Q(s,a) \log \pi(a \mid s)]. \tag{10}$$

- This is very similar to many of the ideas discussed in class last time. In Monday's lecture we'll see that there is still some subtle issue with this objective, but it nonetheless works quite well in practice.
- Q: When would you expect this to do better than BC? When would you expect it to do worse? (bias/variance)

Approach 2. Act greedily:

$$\pi(a \mid s) = \begin{cases} 1 & \text{if } a = \arg\max_{a'} Q(s,a') \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

In future lectures, we'll see that this is guaranteed to improve the policy.

Consideration: Recall that $Q^\pi(s,a)$ depends on the policy. Which policy should we use above?

7 Value Functions and Dynamic Programming

Can we do better than regressing to the empirical returns? To start, we note that the value function obeys some recursive identities:

$$Q^\pi(s,a) = r(s,a) + \gamma \mathbb{E}_{p(s'|s,a)\pi(a'|s')}[Q^\pi(s',a')] \tag{12}$$

$$Q^\pi(s,a) = r(s,a) + \gamma \mathbb{E}_{p(s'|s,a)}[V^\pi(s')] \tag{13}$$

$$V^\pi(s) = \mathbb{E}_{\pi(a|s)}[r(s,a) + \gamma \mathbb{E}_{p(s'|s,a)}[V^\pi(s')]]. \tag{14}$$

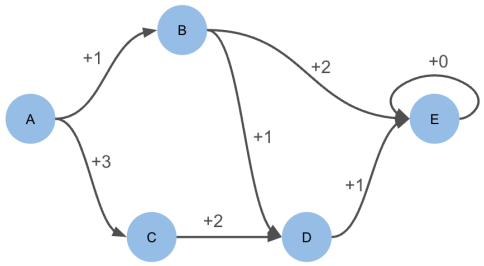


Figure 3: Computing value functions on a graph.

So, it seems like there's some sort of recursive relationship. This should remind you of dynamic programming, and things like Dijkstra's algorithm. Let's start there, seeing how we could use something like Dijkstra's algorithm to estimate these values. We'll use the graph shown in Fig. 3.

No actions. To start, let's assume that there are no actions, so we're just looking at the transitions $p(s' \mid s)$. When there are multiple paths, let's assume that we choose randomly.

$$
\begin{aligned}
V(E) &= 0 && (15)\\
V(D) &= 1 + \gamma V(E) = 1 && (16)\\
V(C) &= 2 + \gamma V(D) = 2 + \gamma && (17)\\
V(B) &= \tfrac{1}{2}(2 + \gamma V(E)) + \tfrac{1}{2}(1 + \gamma V(D)) && (18)\\
&= \tfrac{1}{2}(2 + \gamma 0) + \tfrac{1}{2}(1 + \gamma 1) && (19)\\
&= \tfrac{3}{2} + \tfrac{1}{2}\gamma && (20)\\
V(A) &= \tfrac{1}{2}(1 + \gamma V(B)) + \tfrac{1}{2}(3 + \gamma V(C)) && (21)\\
&= \tfrac{1}{2}(1 + \gamma(\tfrac{3}{2} + \tfrac{1}{2}\gamma)) + \tfrac{1}{2}(3 + \gamma(2 + \gamma)) && (22)\\
&= \tfrac{1}{2} + \tfrac{3}{2} + (\tfrac{3}{4} + 1)\gamma + (\tfrac{1}{4} + \tfrac{1}{2})\gamma^2 && (23)\\
&= 2 + \tfrac{7}{4}\gamma + \tfrac{3}{4}\gamma^2. && (24)
\end{aligned}
$$

With actions. Now, let's assume that actions allow the agent to choose between the outgoing edges. In this setting, let's figure out what the state-action value function looks like:

$$
\begin{aligned}
Q(E, \varnothing) &= 0 && (25)\\
Q(D, \varnothing) &= 1 + \gamma V(E) = 1 && (26)\\
Q(B, \text{right}) &= 2 + \gamma V(E) = 2 && (27)\\
Q(B, \text{down}) &= 1 + \gamma V(D) = 1 + \gamma && \text{(Note that discount means we prefer "right.")}\\
Q(A, \text{right}) &= 1 + \gamma V(B) && \text{(But this requires knowing actions for B!)}\\
&= 1 + \gamma Q(B, \text{right}) = 1 + 2\gamma. && (28)
\end{aligned}
$$

7.1 Circle MDP.

We consider one more example. In the example above, it was clear where to start estimating the values, with $V(E)$. As shown in Fig. 4, in some MDPs this isn't clear. In this setting, we can write down the identities from above:

$$
\begin{aligned}
V(A) &= 1 + \gamma V(B) && (29)\\
V(B) &= 1 + \gamma V(C) && (30)\\
V(C) &= 1 + \gamma V(A). && (31)
\end{aligned}
$$

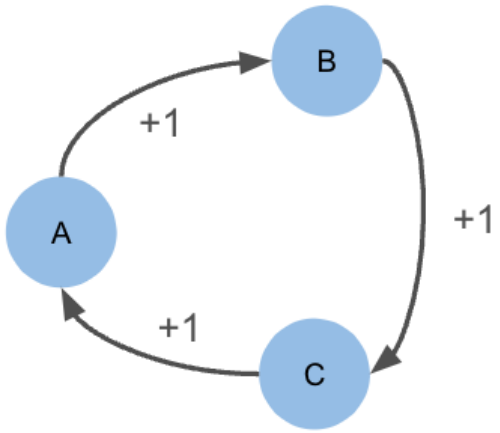


Figure 4: An example MDP that can be solved with linear algebra.

Overloading notation to use $V = (V(A) \quad V(B) \quad V(C))$, we can write this as a system of linear equations:

$$V = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \gamma \underbrace{\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}}_{\triangleq P} V, \tag{32}$$

where we've introduced the matrix P to denote the transitions. We can solve this system of equations using linear algebra:

$$V = (I - \gamma P)^{-1} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{1-\gamma} \\ \frac{1}{1-\gamma} \\ \frac{1}{1-\gamma} \end{pmatrix}. \tag{33}$$

This example highlights the connections between value functions, linear algebra, and graphs.

7.2 Considerations.

- How do you decide the order in which to update the nodes?
- How do you do this when you don't have the graph? How to extend these ideas of dynamic programming to continuous or high-dim settings? This is nice on the graph we presented, but think about what the graph for Tetris or Go looks like.
- Moving forward, we will use a function to represent these Q-values and value functions, just like we did above.
- In the same way that we used the value function at one state to figure out the value function at a different state, we will use the *function* evaluated at one state to figure out the function evaluated at a different state.

8 Outlook

- Much of today's lecture was focused on *data*; on learning a (value) function via regression to data.
- At the end, we saw that there's a close connection with graphs and dynamic programming. Indeed, prior work has referred to the RL problem as *approximate dynamic programming* or (my favorite) *neurodynamic programming*.
- In the end, we see that these are two sides of the same coin.

  - Learning from data, using ideas from dynamic programming to get lower variance estimators.
  - Coping with graphs as they become bigger and bigger. Dynamic programming would require a lookup table that scales with the number of nodes. Instead, we replace that lookup table with a neural network.
- It's completely OK if some of this material is new or seems strange. We'll dive into it in more detail in the coming weeks.
- Next Tuesday, we'll turn back to policies for a moment to see how to can build our first RL algorithm that is actually guaranteed to converge!

References

[1] Holmes, L. D. (1995). Island migrations (2): Birds and sea currents aided canoe navigators. *Pacific Islands Monthly*.

[2] Ioannidis, A. G., Blanco-Portillo, J., Sandoval, K., Hagelberg, E., Miquel-Poblete, J. F., Moreno-Mayar, J. V., Rodríguez-Rodríguez, J. E., Quinto-Cortés, C. D., Auckland, K., Parks, T., et al. (2020). Native american gene flow into polynesia predating easter island settlement. *Nature*, 583(7817):572–577.

[3] Peters, J. and Schaal, S. (2007). Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th international conference on Machine learning*, pages 745–750.

[4] Switek, B. (2013). Dna shows how the sweet potato crossed the sea. *Nature. com. http://www. nature. com/news/dna-shows-how-the-sweet-potatocrossed-the-sea-1.12257 (accessed December, 2013)*.