Imperial College London

Derivation of the Bellman Equation for v_π

2021.10.12

Abstract

This note demonstrates the derivation of the Bellman equation in some detail.

We consider an **agent** (e.g. a robot) that operates in a certain **environment**. At each **time step** $t \in \{0, 1, 2, 3, \ldots\}$, the environment is in a particular **state** $S_0, S_1, S_2, S_3, \ldots \in \mathcal{S}$. The actions of the agent influence the state of the environment. Assuming the initial state of the environment is $S_0 \in \mathcal{S}$:

- At time $t = 0$, the agent takes action $A_0 \in \mathcal{A}(S_0)$[1]. The environment responds with a reward $R_1 \in \mathbb{R}$ and a (possibly) new state $S_1 \in \mathcal{S}$.
- At time $t = 1$, the agent takes action $A_1 \in \mathcal{A}(S_1)$. The environment responds with a reward $R_2 \in \mathbb{R}$ and a (possibly) new state $S_2 \in \mathcal{S}$.
- At time $t = 2$, the agent takes action $A_2 \in \mathcal{A}(S_2)$. The environment responds with a reward $R_3 \in \mathbb{R}$ and a (possibly) new state $S_3 \in \mathcal{S}$.
- ...and so on...

We say "possibly" new state because it may be possible that, for example, $S_2 = S_1$ (or not).

The agent follows a specific **policy** π. The policy is a probability distribution over actions a given that the environment is in a particular state s: $\pi(a \mid s)$. It may be **deterministic**:

- when in state $s \in \mathcal{S}$, go to the left;
- when in state $s' \in \mathcal{S}$, go to the right;
- when in state $s'' \in \mathcal{S}$, go to the left...

or it may be **stochastic**:

- when in state $s \in \mathcal{S}$, go to the left with probability $\frac{1}{4}$, to the right with probability $\frac{3}{4}$;
- when in state $s' \in \mathcal{S}$, go forward with probability $\frac{1}{2}$, backward with probability $\frac{1}{2}$;
- when in state s'' go to the left with probability 1...

[1] The set of possible actions depends on the environment's state. For example, the robot may be able to move forward, backward, to the left, or to the right. However, if the robot is facing a wall (a particular state), the robot can only move backward, to the left, to the right, but not forward.

To make our mathematical life easier, we assume that the state has the **Markov property**: for all states $s' \in \mathcal{S}$ and all rewards $r \in \mathbb{R}$

$$p(R_{t+1} = r, S_{t+1} = s' \mid S_t = s) = p(R_{t+1} = r, S_{t+1} = s' \mid S_1, \ldots, S_{t-1}, S_t = s)$$

for all possible histories $S_1, \ldots, S_{t-1}$. In other words, the state captures all the relevant information from the history. Once the state is known, the history may be thrown away.

As the state is assumed to have the Markov property, we use the **Markov decision process** model. The dynamics of the environment is specified by the **dynamics function** $p(r, s' \mid s, a)$, which is the joint probability of a reward r and next state s', given a current state s and action a.[2]

The reward R_t is somewhat myopic: it describes the benefit on a single time step after taking the action A_{t-1}. We need something that captures all the future rewards. This "something" is the return

$$G_t := R_{t+1} + R_{t+2} + R_{t+3} + \ldots = \sum_{k=0}^{\infty} R_{t+k+1}. \tag{1}$$

This may be a finite or infinite sum. In case it is infinite, it may not converge. So we use the same mathematical trick that we use in finance[3]: **discounting**. Instead of (1), we use the **discounted return**

$$G_t := R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}.$$

where the **discount rate** parameter $\gamma \in [0, 1]$. The greater the γ the more we care about the future rewards versus the immediate rewards.

We can describe our **goal** as the maximisation of the expected discounted return given that we are in a particular state now:

$$\text{maximise } \mathbb{E}\left[G_t \mid S_t = s\right].$$

The value of $\mathbb{E}\left[G_t \mid S_t = s\right]$ depends on the policy π that we follow, so we use the notation

$$\mathbb{E}_\pi\left[G_t \mid S_t = s\right]$$

to highlight this. In fact, the value of a state s under the policy π is given by what is referred to as the **state-value function**

$$v_\pi(s) := \mathbb{E}_\pi\left[G_t \mid S_t = s\right].$$

The state-value function evaluates both the state *and* the policy. A different policy π' under which all states have state-values at least as high as under π and for some states strictly greater is a better policy than π. Computing v_π for all states amounts to **policy evaluation**. And, once we have evaluated a policy, we may hope to **improve** on it.

Notice that the expected return satisfies the following relationship:

$$\begin{aligned} G_t &= R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \ldots \\ &= R_{t+1} + \gamma\left(R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \ldots\right) \\ &= R_{t+1} + \gamma G_{t+1}, \end{aligned}$$

so we can write

$$v_\pi(s) := \mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s\right].$$

[2] We are lucky to have the dynamics function. Much of reinforcement learning deals with the situation when this function is not known.

[3] Where it is sometimes known as the **time value of money**.

Recall the **law of total probability**: if $\{Y_n \mid n = 1, 2, 3, \ldots\}$ is a finite or countably infinite partition of a sample space, $Y_1, Y_2, Y_3, \ldots$ being events, then for any event X of the same probability space

$$\mathbb{P}[X] = \sum_n \mathbb{P}[X \cap Y_n] = \sum_n \mathbb{P}[X \mid Y_n]\,\mathbb{P}[Y_n].$$

There is a similar result, known as the **law of total expectation** (**law of iterated expectations (LIE)**, **tower rule**, **Adam's law**, the **smoothing theorem**)[4]:

$$\mathbb{E}[X] = \sum_n \mathbb{E}[X \mid Y_n]\,\mathbb{P}[Y_n].$$

Picking as Y_n all the possible actions that are taken under the policy π, we obtain

$$v_\pi(s) := \sum_a \mathbb{P}[A_t = a \mid S_t = s]\,\mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a].$$

Since the probabilities $\mathbb{P}[A_t = a \mid S_t = s]$ are given by our policy, which we have denoted $\pi(a \mid s)$, with this change of notation this becomes

$$v_\pi(s) := \sum_a \pi(a \mid s)\mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a].$$

Let us apply the law of total expectation again, this time conditioning on the pair (s', r) where s' is the state that the environment puts us into and r the reward after the action a:

$$v_\pi(s) := \sum_a \pi(a \mid s) \sum_{s'} \sum_r \mathbb{P}\left[S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a\right] \times$$
$$\mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a, S_{t+1} = s', R_{t+1} = r\right].$$

But $\mathbb{P}[S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a]$ is the dynamics function $p(s', r \mid s, a)$, and using this notation the above becomes

$$v_\pi(s) := \sum_a \pi(a \mid s) \sum_{s'} \sum_r p(s', r \mid s, a) \times$$
$$\mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a, S_{t+1} = s', R_{t+1} = r\right].$$

Using the linearity of expectations,

$$v_\pi(s) := \sum_a \pi(a \mid s) \sum_{s'} \sum_r p(s', r \mid s, a) \times$$
$$\left(\mathbb{E}_\pi\left[R_{t+1} \mid S_t = s, A_t = a, S_{t+1} = s', R_{t+1} = r\right] + \gamma \mathbb{E}_\pi\left[G_{t+1} \mid S_t = s, A_t = a, S_{t+1} = s', R_{t+1} = r\right]\right).$$

The first expectation is deterministic, since we are conditioning on $R_{t+1} = r$, and so

$$v_\pi(s) := \sum_a \pi(a \mid s) \sum_{s'} \sum_r p(s', r \mid s, a) \times$$
$$\left(r + \gamma \mathbb{E}_\pi\left[G_{t+1} \mid S_t = s, A_t = a, S_{t+1} = s', R_{t+1} = r\right]\right).$$

By the Markov assumption, G_{t+1} is independent of S_t, A_t, R_{t+1}. It depends only on S_{t+1}, therefore

$$v_\pi(s) := \sum_a \pi(a \mid s) \sum_{s'} \sum_r p(s', r \mid s, a)\left(r + \gamma \mathbb{E}_\pi\left[G_{t+1} \mid S_{t+1} = s'\right]\right).$$

[4] See `https://en.wikipedia.org/wiki/Law_of_total_expectation`

We recognise that

$$\mathbb{E}_\pi \left[G_{t+1} \mid S_{t+1} = s' \right] = v_\pi(s'),$$

so our equation becomes

$$v_\pi(s) := \sum_a \pi(a \mid s) \sum_{s'} \sum_r p(s', r \mid s, a) \left(r + \gamma v_\pi(s') \right).$$

This is the **Bellman equation for v_π** [1].

Markov decision processes and the Bellman equation are fundamental ideas in reinforcement learning, on which the rest of the theory (including modern developments in deep reinforcement learning) are built.

The Bellman equation occurs in several different forms. In the continuous setting in optimal control this equation is usually referred to as the Hamilton–Jacobi–Bellman (HJB) equation. The connection to the Hamilton–Jacobi equation from classical physics was first drawn by Rudolf Kálmán [3]. In discrete-time problems, the equation is usually referred to as the Bellman equation.

See [2] for details.

References

[1] Richard Bellman. An introduction to the theory of dynamic programming. techreport, RAND Corp., 1953.

[2] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 4 edition, 2017.

[3] Rudolf Emil Kálmán. *Mathematical Optimization Techniques*, chapter The Theory of Optimal Control and the Calculus of Variations, pages 309–331. University of California Press, 1963.