Note on Q functions and V functions in Reinforcement Learning

(from Chapter 3 of *Sutton and Barto's* book)

written by D. Gueorguiev, Nov 26, 2023

Table of Contents

Notation and Definitions

Distribution of the Dynamics of the MDP: defined through the following 4 arguments function:
$p(s', r|s, a) \doteq \Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\}$
which is the probability to get from state s to state s' with action a and with reward r.

Distribution of the state-transition probabilities: defined through the following 3 arguments function:
$p(s'|s, a) \doteq \Pr\{S_t = s'|S_{t-1} = s, A_{t-1} = a\} = \sum_{r \in \mathcal{R}} p(s', r|s, a)$

Markov Decision Process (abbrev *MDP*): a 4-tuple $(\mathcal{S}, \mathcal{A}, p, \gamma)$ with

- $\mathcal{S}$ is a set of states (finite or infinite, discrete, or continuous)
- $\mathcal{A}$ is a set of actions (finite or infinite, discrete, or continuous)
- $p(s', r|s, a) \doteq \Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\}$ is the probability to get from state s to state s' with action a and with reward r. This is the function describing the dynamics of the MDP.
- $\gamma \in [0,1]$ is the discount factor which determines to what extent the focus is on the most recent rewards. with $\gamma = 1$ there is no focus on the most recent rewards only.

Note: There is another equivalent definition of Markov process which uses the *state-transition probabilities distribution* represented by the three-argument function $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0,1]$ and *reward function* $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. With this definition the Markov Decision Process is defined as a 5-tuple $(\mathcal{S}, \mathcal{A}, T, r, \gamma)$, where:

- $\mathcal{S}$ is a set of states (finite or infinite, discrete, or continuous)
- $\mathcal{A}$ is a set of actions (finite or infinite, discrete, or continuous)

- $p(s'|s,a) \doteq \Pr\{S_t = s'|S_{t-1} = s, A_{t-1} = a\}$ is the probability to get from state s to state s' with action a.
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defines a *reward function*
- $\gamma \in [0,1]$ is the discount factor

Note 2: A more detailed definition of MDP involves specifying the initial state distribution $d_0(s_0)$ and augments either of the MDP definitions as:

Markov Decision Process with specified *initial state* is a 6-tuple $(\mathcal{S}, \mathcal{A}, T, r, d_0, \gamma)$, where:
- $\mathcal{S}$ is a set of states (finite or infinite, discrete, or continuous)
- $\mathcal{A}$ is a set of actions (finite or infinite, discrete, or continuous)
- $p(s'|s,a) \doteq \Pr\{S_t = s'|S_{t-1} = s, A_{t-1} = a\}$ is the probability to get from state s to state s' with action a.
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defines a *reward function*
- $d_0(s_0)$ defines the initial state distribution
- $\gamma \in [0,1]$ is the discount factor

Learning Policy (or just *Policy*): function $\pi : \mathcal{S} \to \mathcal{A}$ which represents mapping from states to probabilities of selecting each possible action.
If the agent is following policy π at time t, then $\pi(a|s)$ is the probability that $A_t = a$ if $S_t = s$. Note that $\pi(a|s)$ is an ordinary function which defines a probability distribution over $a \in \mathcal{A}(s)$ for each $s \in \mathcal{S}$.

We would like to modify the policy π with training or experience.

State-Value and State-Action Functions

Let us assume that the current state is S_t, and actions are selected according to a stochastic policy π. Then we would like to derive an expression for the expectation of R_{t+1} in terms of π and $p(s', r|s, a)$.
Recall, the function $p(s', r|s, a)$ defines the dynamics of the MDP and is given as:

$$p(s', r|s, a) \doteq \Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\} \text{ for all } s', s \in \mathcal{S}, r \in \mathcal{R}, a \in \mathcal{A}(s) \qquad (1)$$

Then we can write:

$$\mathbb{E}_\pi[R_{t+1}|S_t = s] = \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a)\,[r] \qquad (2)$$

Here r denotes the reward of going from state s to state s' taking action a is given by MDP's R function: $r = R(s, s', a)$.
State-Value Function for Policy π (or simply *Value function;* aka *V function*): the value function of a state s under a policy π, denoted with $v_\pi(s)$, is the expected return when starting in s and following π thereafter. For MDPs, we can define v_π formally by

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi[\textstyle\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|\, S_t = s] \text{ for all } s \in \mathcal{S} \qquad (3)$$

where $\mathbb{E}_\pi[\cdot]$ denotes the expected value of a random variable given that the agent follows policy π, and t is any time step. Note that the value of the terminal state, if any, is always zero.

Action-Value Function for Policy π (aka *Q function*):
We define the value of taking action a in state s under a policy π, denoted $q_\pi(s, a)$, as expected return starting from s, taking the action a, and thereafter following policy π:

$$q_\pi(s, a) \doteq \mathbb{E}_\pi[G_t|S_t = s, A_t = a] = \mathbb{E}_\pi[\textstyle\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|\, S_t = s, A_t = a] \text{ for all } s \in \mathcal{S} \text{ and } a \in \mathcal{A}(s) \quad (4)$$

Let us express v_π in terms of q_π and π. Given a state s, the state value function $v_\pi(s)$, given with (3), is equal to the expected cumulative return from that state given a distribution of actions π. The action value function q_π is the expectation of the return given state s, and taking action a as a starting point, and following policy π thereafter. Therefore, given a state s, the action-value function q_π is the weighted sum of the action-values over all relevant actions weighted by the policy weight:

$$v_\pi(s) = \sum_a \pi(a|s) q_\pi(s,a) \quad (5)$$

Given a state s and an action a let us express the action-value function q_π in terms of the state value function v_π and the function defining the MDP dynamics $p(s', r|s, a)$. Recall, given a state s and an action a, the action value function q_π is given by the mathematical expectation of the discounted future rewards i.e. return G_t. The return G_t is the discounted sequence of rewards after the time step t and it can be written as:

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} = R_{t+1} + \gamma G_{t+1} \quad (6)$$

It is important to recognize that

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a] = \mathbb{E}_\pi[R_{t+1}|S_t = s, A_t = a] + \gamma \mathbb{E}_\pi[G_{t+1}|S_t = s, A_t = a]. \quad (7)$$

The first term on the right-hand side of (7) can be expressed as:

$$\mathbb{E}_\pi[R_{t+1}|S_t = s, A_t = a] = \sum_{s'} \sum_r p(s', r|s, a)\,[r]. \quad (8)$$

As before, r denotes the reward of going from state s to state s' taking action a is given by MDP's R function: $r = R(s, s', a)$.

The expectation in the second term on the right-hand side of (8) can be expressed as:

$$\mathbb{E}_\pi[G_{t+1}|S_t = s, A_t = a] = \sum_{s'} \sum_r p(s', r|s, a) v_\pi(s'). \quad (9)$$

This is the expectation of the return starting at the next time step $t + 1$ following the policy π given the current state s and the action a, chosen according to π.
Substituting (8) and (9) into (7) gives us:

$$q_\pi(s,a) = \sum_{s'} \sum_r p(s', r|s, a)\,[r + \gamma v_\pi(s')]. \quad (10)$$

Thus, the action-value function q_π given state s and action a following policy π is expressed as the sum of the next reward and discounted state-value weighted by probability distribution over the possible next states and next rewards from the given action a and state s.

Estimation Methods for State-Value and State-Action Functions

The value functions v_π and q_π can be estimated from experience. For example, if an agent follows policy π and maintains an average, for each state encountered, of the actual returns that have followed that state, then the average will converge to the state value $v_\pi(s)$, as the number of times that state is encountered approaches infinity. If separate averages are kept for each action taken in each state, then these averages will similarly converge to the action values, $q_\pi(s,a)$. We call estimation methods of this kind *Monte Carlo methods* because they involve averaging over many random samples of actual returns. Of course, in case there is a large number of states then it would not be feasible to manage separate averages for each state. Instead the agent would have to maintain v_π and q_π as parametrized functions (with fewer parameters than states) and adjust the parameters to

better match the observed returns. This approach can produce accurate estimates, although much depends on the nature of the parametrized function approximator.

Bellman's Equations for State-Value and State-Action Functions

Bellman's equation for state values v

The value functions satisfy recursive relationships this property of value functions will prove quite useful. For any policy π and for any state s , the following consistency condition holds between the value of s and the value of its successor states. Starting with (6) applied to the definition of $v_\pi(s)$:

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s] \quad (11)$$

Using (5) the last equation becomes:

$$\begin{aligned} v_\pi(s) &= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) \left[r + \gamma \mathbb{E}_\pi[G_{t+1}|S_{t+1} = s']\right] \\ &= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) \left[r + \gamma v_\pi(s')\right] \quad \text{for all} \quad s \in \mathcal{S} \end{aligned} \quad (12)$$

where it is implicit that the actions, a, are taken from the set $\mathcal{A}(s)$, that the next states, s', are taken from the set $\mathcal{S}$, and that the rewards, r, are taken from the set $\mathcal{R}$. Here the reward of going from state s to state s' taking action a is given by MDP's R function: $r = R(s, s', a)$. Note that the right-hand side of (12) is interpreted as an expected value obtained as a sum over the values of the triplet a, s', and r. For each triplet (a, s', r) the quantity $r + \gamma v_\pi(s')$ is weighed by its probability, $\pi(a|s)p(s', r|s, a)$.

Eq. (12) is known as the *Bellman equation* for v_π. It expresses a relationship between the value of a state and the values of its successor states.

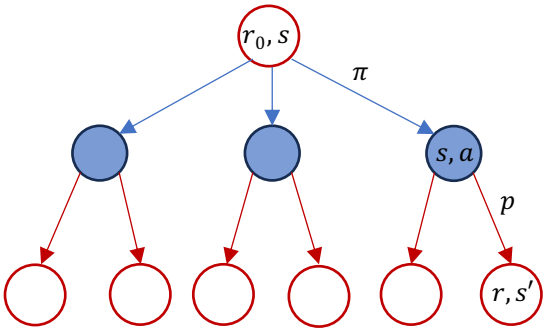

Figure 1: Backup diagram for v_π

This relationship is expressed by the *Backup diagram* shown on Figure 1. Each open circle, which will be denoted as *reward-state node* so forth, colored in red represents a state and the reward, which is associated with this state For instance, the root node shown on Figure 1 has associated reward r_0 and state s. Each solid circle, colored in blue represents a state-action pair and will be denoted as *state-action node* so forth. The specific state on the rightmost state-action node is shown as (s, a). Each directed blue edge connects state node with state-action node and represents application of the policy π to the root reward-state node (r_0, s). Each directed red edge emanating from a state-action node ends in a possible reward-state node corresponding to specific probable pair of reward r and new state s'. Thus, each directed red edge represents the application of the function p of the MDP dynamics. The Bellman equation (12) averages over all of the possibilities weighing each possibility represented by a path from the root of the Backup diagram on Figure 1 to a leaf by its probability of occurring. It states that the value of the start state must equal the discounted value of the expected next state plus the reward expected along

the way. The value function v_π is the unique solution to its Bellman equation. Various methods exist to compute exactly, approximate, or learn the value function.

Bellman's equation for state-action values q

Let us derive a similar recursive relation with respect to the state-action value function. That is, we will find out what is the relation between the action value $q_\pi(s,a)$ and that for the possible successors to the state-action pair - $q_\pi(s',a')$. The derivation follows from the Backup diagram shown on Figure 2 below.

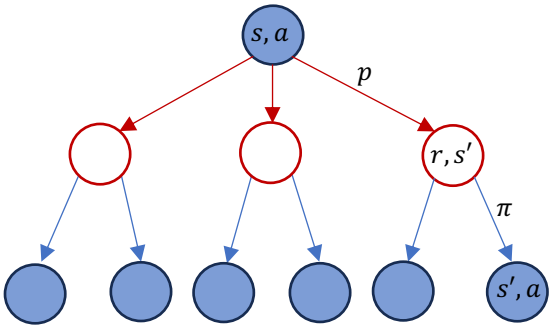

Figure 2: Backup diagram for q_π

From (7) we can write:
$q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a] = \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s, A_t = a]$ (13)

The expectation of the reward on the right-hand side can be rewritten as:

$\mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s, A_t = a] = \sum_{s',r} p(s',r|s,a)[r + \gamma q_\pi(s',a')]$ (14)

Here using (7) again we denote with $q_\pi(s',a')$ the expression for $\mathbb{E}_\pi[G_{t+1}|S_t = s, A_t = a]$.

Thus we get the Bellman's equation with respect q_π:

$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma q_\pi(s',a')]$ (15)

Expressing the current state values v in terms of the next action values q

It is instructive to compare Eq (5) which we derived earlier with Eq (12) and Eq (15).
Eq (5) deserves its own Backup diagram shown on Figure 3:

$v_\pi(s) = \sum_a \pi(a|s) q_\pi(s,a)$ (5)

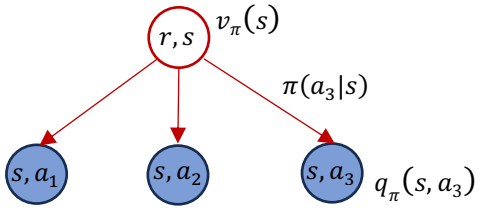

Figure 3: Backup diagram for relation between $v_\pi(s)$ and $q_\pi(s,a)$

Eq (5) tells us how the value of a state depends on the values of the actions possible in that state and on how likely each action is to be taken under the current policy. The state value which corresponds to the state-value node at the root is obviously $v_\pi(s)$ and the action values which corresponds to its children are $q_\pi(s, a_i), i = 1..3$. The probability with which each action a_i is taken is given by the policy i.e. $\pi(a_i|s), i = 1..3$.

Expressing the current action values q in terms of the next state values v

The value of an action, $q_\pi(s, a)$, depends on the expected next reward and the expected sum of the remaining rewards. Expressed as a Backup diagram we arrive at Figure 4 shown below.

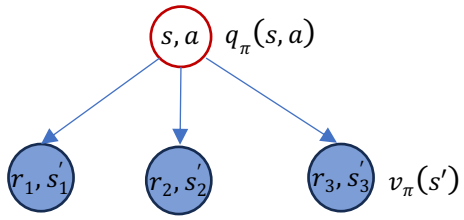

Figure 4: Backup diagram expressing the dependence of the current action value on the expected next reward-state values.

Formally expressed this relation becomes:

From Eq (7) we have
$q_\pi(s, a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a] = \mathbb{E}_\pi[R_{t+1}|S_t = s, A_t = a] + \gamma\mathbb{E}_\pi[G_{t+1}|S_t = s, A_t = a]$

Clearly, $\mathbb{E}_\pi[R_{t+1}|S_t = s, A_t = a] = \sum_{s',r} p(s', r|s, a) \cdot r$ where $r = R(s, s', a)$
Eq. (9) states that the expectation in the second term of (7) can be written as:

$\mathbb{E}_\pi[G_{t+1}|S_t = s, A_t = a] = \sum_{s',r} p(s', r|s, a) \cdot v_\pi(s')$

Combining the last two results we obtain:

$q_\pi(s, a) = \mathbb{E}[R_{t+1} + \gamma v_\pi(s')|S_t = s, A_t = a] = \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')]$ (16)

Optimal Policies and Optimal Value Functions

We want to find a policy π which maximizes the reward over long enough run that is, maximizes the return. For finite MDPs we define an optimal policy in the following way. Value functions define a partial ordering over policies. A policy π is defined to be better than or equal to a policy π' if its expected return G is greater than or equal to that of π' for all states. We write:

$\pi \geq \pi' \Leftrightarrow v_\pi(s) \geq v_{\pi'}(s)\ \forall\ s \in \mathcal{S}$ (17)

For finite MDPs it can be shown that there is always one policy that is better or equal to all other policies. This is an *optimal policy*. We denote with π_* any one of the optimal policies which have the same *state-value function*, denoted with v_*, and defined as:

$v_*(s) \doteq \max_\pi v_\pi(s)\ \forall\ s \in \mathcal{S}$ (18)

Optimal policies also share the same *optimal action-value function*, denoted with q_* and defined similarly:

$$q_*(s,a) \doteq \max_{\pi} q_\pi(s,a) \;\; \forall\, s \in \mathcal{S} \;\wedge\; \forall\, a \in \mathcal{A}(s) \qquad (19)$$

From Eq (16) it follows that for the state-action pair (s,a), the function $q_*(s,a)$ gives the expected return for taking action a in state s and thereafter following the optimal policy. So we can rewrite (16) in terms of the optimal policy as:

$$q_*(s,a) = \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a] \qquad (20)$$

Bellman's Optimality Equations

Bellman's optimality equation for the optimal state-value function

Because v_* is the value function for a policy, it must satisfy the self-consistency condition given by the Bellman equation for state values Eq. (12). Because it is the optimal value function, however, v_*'s consistency condition can be written in a special form without reference to any specific policy. The result will be the Bellman equation for v_* or the *Bellman optimality equation*. Intuitively, the Bellman optimality equation expressed the fact that the value of a state under an optimal policy must equal the expected return for the best action from that state:

$$\begin{aligned} v_*(s) &= \max_{a \in \mathcal{A}(s)} q_{\pi_*}(s,a) \\ &= \max_a \mathbb{E}_{\pi_*}[G_t | S_t = s, A_t = a] \\ &= \max_a \mathbb{E}_{\pi_*}[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a] \qquad \text{by Eq. (7)} \\ &= \max_a \mathbb{E}_{\pi_*}[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a] \qquad (21) \\ &= \max_a \textstyle\sum_{s',r} p(s', r | s, a)[r + \gamma v_*(s')] \qquad (22) \end{aligned}$$

Eq. (21) and Eq (22) are two forms of the Bellman optimality equation for v_*.

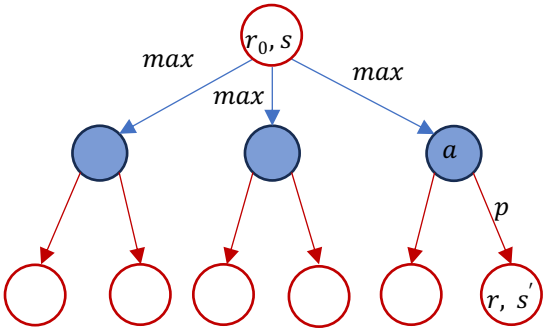

Figure 5: Backup diagram for optimal state value function v_*

Bellman's optimality equation for the optimal state-value function

The Bellman optimality equation for q_* is

$$\begin{aligned} q_*(s,a) &= \mathbb{E}\left[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \,|\, S_t = s, A_t = a\right] \\ &= \textstyle\sum_{s',r} p(s', r | s, a)\left[r + \gamma \max_{a'} q_*(s', a')\right] \qquad (23) \end{aligned}$$

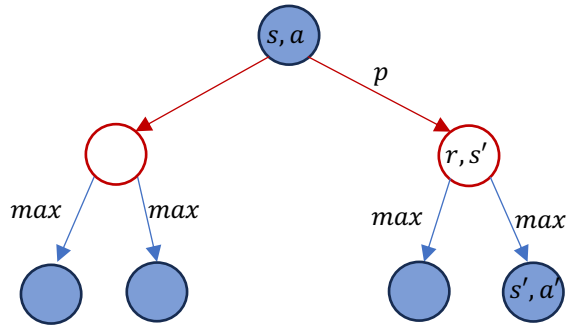

Figure 6: Backup diagram for optimal state value function q_*

The Bellman optimality equation is actually a system of equations, one for each state, so if there are n states, then there would be n equations with n unknowns. If the dynamics p of the environment is known, then in principle one can solve this system of equations for v_* as well as the set of equations for q_*.

Determining the Optimal Policy from the Bellman Optimality Equations

Once we have determined v_* it is straightforward to the determine an optimal policy π_*. For each state s, there will be one or more actions at which the maximum is obtained in the Bellman optimality equation. Any policy that assigns nonzero probability only to these actions is an optimal policy. One can devise an algorithm based on one-step-ahead search. With the found optimal value function, v_*, then the actions which appear best after one-step-ahead search will be optimal actions . In other words, any policy that is greedy with respect to the optimal evaluation function v_* is an optimal policy. Thus, for any optimal policy the actions can be selected based only on the short-term consequences. The found greedy policy is actually optimal in a non-local sense – this is true because the equations for v_* already account for the reward consequences of all possible future behaviors – that is sequential choices of actions.

Having q_* makes choosing optimal actions even easier. With q_* we do not need to perform one-step-ahead search: for any state s, we need to find any action that maximizes $q_*(s, a)$. The action-value function $\boldsymbol{q_*}$ effectively caches the results of all one-step-ahead searches and it provides a **non-local optimal return** as a value that is locally and immediately available for each state-action pair! Hence, at the cost of representing a function of state-action pairs, instead of just states, the optimal action-value function allows optimal actions to be selected without having to know anything about possible successor states and their values , that is, without having to know anything about the environmental dynamics p! This in turn may represent significant computational advantage compared to using v_* for the construction of optimal policy.