# Examples using Policy and Value Functions

(from *Sutton and Barto*)

Compiled by D. Gueorguiev 1/21/2024

## Gridworld

The cells of the grid correspond to the states of the environment. At each cell, four actions are possible: ***north***, ***south***, ***east***, and ***west*** which deterministically cause the agent to move one cell in the respective direction on the grid. Actions that would take the agent off the grid leave its position unchanged, but also result in a reward of .

Other actions result in a reward of 0, except those that move the agent out of the special states and . From state , all four actions yield a reward of and take the agent to . From state , all actions yield a reward of and take the agent to .

A black background with a black square

Description automatically generated with medium confidence

Figure 1: Gridworld Example

Suppose the agent selects all four actions with equal probability in all states. Figure 2 below shows the value

Figure 2: State-value function for Gridworld

## Appendix

### Notation and Definitions from Sutton and Barto’s RL book

**Distribution of the Dynamics of the MDP**: defined through the following 4 arguments function:

which is the probability to get from state to state with action and with reward .

**Distribution of the state-transition probabilities**: defined through the following 3 arguments function:

**Markov Decision Process** (abbrev *MDP*): a 5-tuple with

* is a set of states (finite or infinite, discrete, or continuous)
* is a set of actions (finite or infinite, discrete, or continuous)
* is the function which describes the MDP dynamics i.e. probability to get from state to state with action and with reward .
* defines a *reward function*
* is the discount factor which determines to what extent the focus is on the most recent rewards. with 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 . With this definition the Markov Decision Process is defined as a 5-tuple , where:

* is a set of states (finite or infinite, discrete, or continuous)
* is a set of actions (finite or infinite, discrete, or continuous)
* is the distribution of the *state-transition probabilities* i.e. the probability to get from state to state with action .
* defines a *reward function*
* is the discount factor

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

Markov Decision Process with specified *initial state* is a 6-tuple , where:

* is a set of states (finite or infinite, discrete, or continuous)
* is a set of actions (finite or infinite, discrete, or continuous)
* is the probability to get from state to state with action .
* defines a *reward function*
* defines the initial state distribution
* is the discount factor

**Learning Policy** (or just *Policy*): function which represents mapping from states to probabilities of selecting each possible action.

If the agent is following policy at time , then is the probability that if . Note that is an ordinary function which defines a probability distribution over for each .

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

### State-Value and State-Action Functions

Let us assume that the current state is , and actions are selected according to a stochastic policy . Then we would like to derive an expression for the expectation of in terms of and .

Recall, the function defines the dynamics of the MDP and is given as:

for all (1)

Then we can write:

(2)

Here denotes the reward of going from state to state taking action is given by MDP’s function: .

**State-Value Function for Policy** (or simply *Value function;* aka *function*): the value function of a state under a policy , denoted with , is the expected return when starting in and following thereafter. For MDPs, we can define formally by

for all (3)

where denotes the expected value of a random variable given that the agent follows policy , and is any time step. Note that the value of the terminal state, if any, is always zero.

**Action-Value Function for Policy** (aka *function*):

We define the value of taking action in state under a policy , denoted , as expected return starting from , taking the action , and thereafter following policy :

for all and (4)

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

(5)

Given a state and an action let us express the action-value function in terms of the state value function and the function defining the MDP dynamics . Recall, given a state and an action , the action value function is given by the mathematical expectation of the discounted future rewards i.e. return . The return is the discounted sequence of rewards after the time step and it can be written as:

(6)

It is important to recognize that

. (7)

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

. (8)

As before, denotes the reward of going from state to state taking action is given by MDP’s function: .

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

. (9)

This is the expectation of the return starting at the next time step following the policy given the current state and the action , chosen according to .

Substituting (8) and (9) into (7) gives us:

. (10)

Thus, the action-value function given state s and action following policy 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 and state .

## Bibliography

[Reinforcement Learning, Richard S. Sutton, Andrew G. Barto, second edition, 2020](https://github.com/dimitarpg13/reinforcement_learning_and_game_theory/blob/main/books/ReinforcementLearningSuttonSecondEdition2020.pdf)