Reinforcement Learning HW2

Hadi Askari

20th April 2023

1 Implementation

The code takes approximately 5 seconds to run and implements the Iterative Policy Evaluation function. The code calls the *policyevaluation* function that has one input of the probability of heads. The function then initializes V using the possible states. We chose threshold to be 0.0001. For each of the states, we first save the old value. Then we select one of 3 possible policies:

1. **Aggressive**: Here we bet our entire cash at hand. The policy is *action = state.*
2. **Conservative**: Here we only bet 1 dollar each time. The policy is *action = 1*
3. **Random**: Here we bet a random number from the uniform distribution of 1 to the cash at hand. The policy is a distribution from *1:s*

After we select the action from the policy we solve the following equation:

$$V(s) \leftarrow \sum_{a} \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma V(s')]$$

Then we update our value function and compute the delta between the old and new values. This continues until the delta falls below our threshold.

[Note: The rewards here are the bets one or lost according to the Professor's lecture. Alternatively, they could have been only +1 for winning the game and 0 for everything else. That would result in different final value functions]

2 Final Value Functions

We get the following Value Functions:

Heads Prob = 0.9

Aggressive Policy 0 0.8000 1.6000 2.4000 3.2000 4.0000 4.8000 5.6000 6.4000 7.2000

Conservative Policy 0 0.8000 0.8800 0.8880 0.8888 0.8889 0.8889 0.8889 0.8889 0.8889

Random Policy 0 0.8000 1.2400 1.6680 2.0927 2.5160 2.9386 3.3608 3.7827 4.2044

Heads Prob = 0.1

Aggressive Policy 0 -0.8000 -1.6000 -2.4000 -3.2000 -4.0000 -4.8000 -5.6000 -6.4000 -7.2000

Conservative Policy 0 -0.8000 -1.5200 -2.1680 -2.7512 -3.2761 -3.7485 -4.1736 -4.5563 -4.9006

Random Policy 0 -0.8000 -1.5600 -2.3080 -3.0503 -3.7893 -4.5261 -5.2615 -5.9957 -6.7291

It is evident that the bias of the coin is having an effect since 0.9 has positive values (with aggressive being the best performing and conservative the least) and 0.1 having negative values (with conservative performing relatively the best and aggressive performing the worst).