Reinforcement Learning HW2

By Madhur Tandon
(2016053)

Explanation for Q2

The value function equation is of type $P = x + Qy$ where P and Q are states.
Thus, $P - Qy = x$
There are 25 states in total (since a grid of 5 x 5)

Now, there will be 25 such equations (one for each P) in 25 variables (each Q).

Thus, A is a 25 x 25 matrix with the coefficient -y in Pth row and Qth column
But, the coefficient for Pth row and Pth column is (1-y) since P = Q in that case.

B is a matrix of 25x1 values of different x

Solving AX=B using numpy after forming the above matrices gives us the solution as a 25x1 vector which when reshaped to 5x5 gives us the evaluated value function at the given policy.

Explanation for Q4

The equation is now of the form $P = \max(x + Qy)$
Since there are 4 possible actions from each state, thus the max is to be taken over 4 such (x+Qy)'s

Now, $P = \max(x + Qy)$ can be written as $P >= x + Qy$
Thus, each P has 4 such inequalities since 4 possible actions from each P

$P >= x + Qy$
Thus, $-P <= -x - Qy$
or $-P + Qy <= -x$

We form A as a 100 x 25 matrix since 100 such inequalities each in 25 variables and B as a 100x1 matrix.

This gives us the form AX<=B which can be solved using Scipy's linprog.

Explanation for Q6

Directly Implemented from book

Explanation for Q7

Each state is a 2 tuple (x, y) where x is the number of cars in location 1 and y is the number of cars in location 2.
Thus, there are 21 x 21 possible states since 0 cars can also be there

An action is defined as the movement of car from location 1 to location 2.
Thus, possible actions are -5,-4,-3,-2,-1,0,1,2,3,4,5

Now, once we have moved the cars aka taken the action, we need to calculate the probabilities of going into the next state -- which also requires us to calculate all possible next states after the action has been taken. This can be done as follows:

Suppose there are 5 cars remaining at location 1 after the action has been taken, then I can request 0 to 5 cars from location 1 (both inclusive).

Let's say I request i cars from location 1, (let i = 2), then we will have 3 cars remaining at location 1

Then, the number of cars I can return to location 1 is given by 0 to 20 + i - 5 (both inclusive)
Or from 0 to 20 + 2 - 5 => from 0 to 17 (both inclusive) [since only 3 were left after 2 were requested]

Thus, in general, If location 1 has p cars and location 2 has q cars after the action has been taken, then

I can request 0 to p cars from location 1, (let's say i were actually requested)
I can return 0 to 20 + i - p to location 1 (let the actual number of cars returned be j)
Similarly, I can request 0 to q cars from location 2 (let's say k were actually requested)
And I can return 0 to 20 + k - q cars to location 2 (let's say l were actually returned)

Thus, the probability of i request from location 1
j returns to location 1
k requests from location 2
l returns to location 2

Can be calculated via the 4 poisson distribution given to us.

The end states are given by all the possible combinations of (p - i +j, q - k + l) with the probability of transitioning into this state given by multiplying the above 4 probabilities together.

This allows us to use the Bellman's equation (since we have now recognized all the possible next states the system can go to after taking an action a and being in the state s)