# Assignment 6 Reinforcement Learning

### Xuehan Jiang 4578791

**Exercise 1**

Actually, it is a question to prove that the infinite series *Rt* is bounded.From the condition, 0 ≤ γ < 1 and −10 ≤ *r t+h+1*≤ 10, we know that *Rt* is a series whose values are between positive and negative.

If γ = 0, , which is bounded.

If γ ≠ 0 , let , so we can calculate:

=

Because 0 < γ < 1, is bounded. What’s more, we have the theory that if every element of a series is multiplied by a nonzero constant, the astringency of the series won’t change. Therefore, *Rt* is bounded too.

**Exercise 2**

Iteration 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| state  action | 1 | 2 | 3 | 4 | 5 | 6 |
| Left | 0 | 1 | 0 | 0 | 0 | 0 |
| Right | 0 | 0 | 0 | 0 | 5 | 0 |

Iteration 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| state  action | 1 | 2 | 3 | 4 | 5 | 6 |
| Left | 0 | 1 | 0.5 | 0 | 0 | 0 |
| Right | 0 | 0 | 0 | 2.5 | 5 | 0 |

Iteration 3

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| state  action | 1 | 2 | 3 | 4 | 5 | 6 |
| Left | 0 | 1 | 0.5 | 0.25 | 1.25 | 0 |
| Right | 0 | 0.25 | 1.25 | 2.5 | 5 | 0 |

Iteration 4

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| state  action | 1 | 2 | 3 | 4 | 5 | 6 |
| Left | 0 | 1 | 0.5 | 0.625 | 1.25 | 0 |
| Right | 0 | 0.625 | 1.25 | 2.5 | 5 | 0 |

**Exercise 3**

Q\* for γ = 0

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| state  action | 1 | 2 | 3 | 4 | 5 | 6 |
| Left | 0 | 1 | 0 | 0 | 0 | 0 |
| Right | 0 | 0 | 0 | 0 | 5 | 0 |

Q\* for γ = 0.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| state  action | 1 | 2 | 3 | 4 | 5 | 6 |
| Left | 0 | 1 | 0.1 | 0.01 | 0.05 | 0 |
| Right | 0 | 0.01 | 0.05 | 0.5 | 5 | 0 |

Q\* for γ = 0.9

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| state  action | 1 | 2 | 3 | 4 | 5 | 6 |
| Left | 0 | 1 | 3.2805 | 3.6450 | 4.05 | 0 |
| Right | 0 | 3.6450 | 4.05 | 4.5 | 5 | 0 |

Q\* for γ = 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| state  action | 1 | 2 | 3 | 4 | 5 | 6 |
| Left | 0 | 1 | 5 | 5 | 5 | 0 |
| Right | 0 | 5 | 5 | 5 | 5 | 0 |

Obviously, γ influences the effect of . When γ becomes larger, the influence of becomes bigger. Otherwise, the influence of reward(s) is bigger. In other words, when γ is small, the learning algorithm take more immediate interests (reward) into accounts. So the policy of states close to the state is to go to the left, while the policy of states close to the state is to go to the right. On the contrary, when γ is large, the learning algorithm take more future interest () into account. The bigger award (state 6) has the bigger influence so that all the states’ policy is to go to the right, even for the furthest state (state 2).

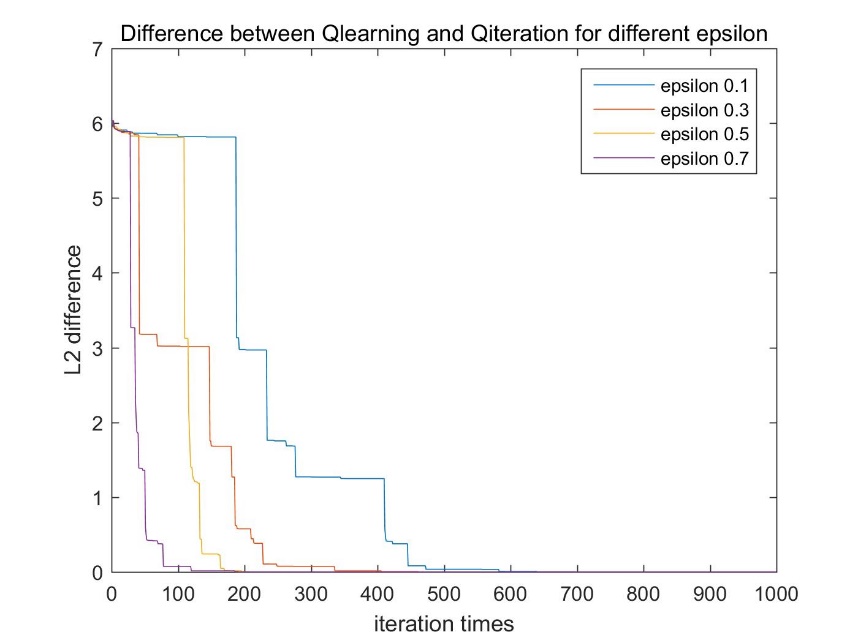
When γ = 1,

= = 1

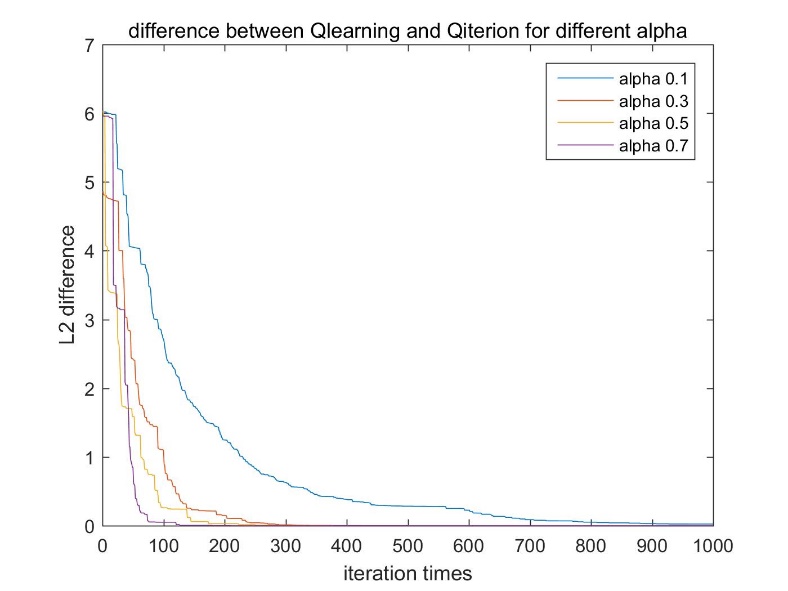
So can be bounded or not. Therefore, we can not make sure whether *Rt* is bounded or not.

In the table above (Q\* for γ = 1), it is easy to find that if t is large enough, all the St+1 is 6, no matter what is the initial state. It means that if h is large, the *r t+h+1*is 5. So *Rt* will become infinite as a sum of infinite 5.

**Exercise 4**



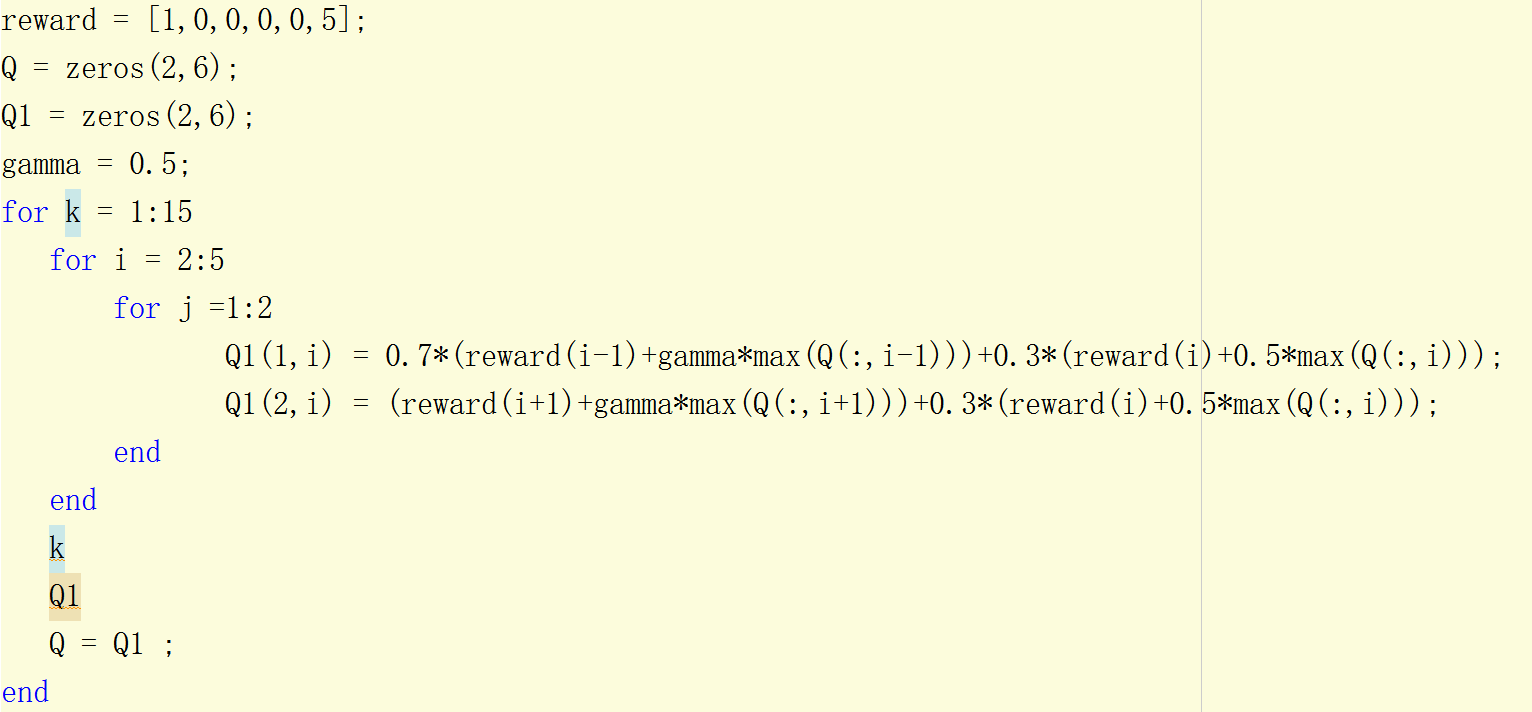
First, I fixed the value of α (α = 0.8) and change the value of ε. From the figure above, we can find that the number of times of iteration goes down as ε increase. ε-greedy policy is used to decision which action to pick. With probability ε we pick a random action (exploration), and with probability 1-ε we pick the greedy action (exploitation). Large ε makes the model more flexible and faster too explore the unknown. In this exercise, large ε makes the model faster to converge. On the contrary, small ε can keep the model more stable.



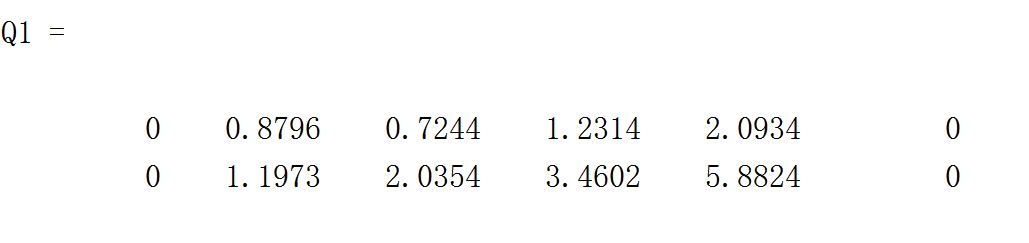
Then, I fixed the value of ε (ε = 0.5) and change the value of α. As shown in the figure above, larger the α is, faster the algorithm converges. α is the learning rate, which means that larger α makes the model keep less effect of the previous step. In other words, large α can make the model explore faster, which is consistent with the figure. Similar to ε, small α makes the model more stable.

**Exercise 5**

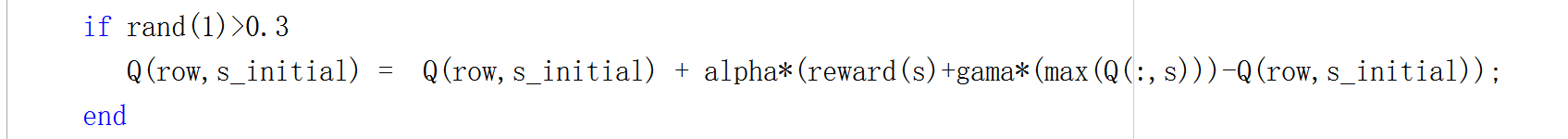
For the Q-iteration, I implemented the new value equation as below:



At the 12th time of iteration, the Q value converged. The optimal value function Q∗ is shown below



For the Q-learning, we can keep the Q-learning parameters. However, after using the ε-greedy policy to decide the action, another judge should be implemented as shown below. The model have 30% probability to do nothing and 70% probability to update the Q(s,a).



This will slow down the convergence of Q-learning significantly. To test, I run 500 times Q-learning and “partially broken” Q-learning and calculate their mean time of convergence respectively. The parameters for these 2 model are the same (α=0.8, ε=0.7). The mean time of convergence for Q-learning is 112.1020 while that for “partially broken” Q-learning is 163.4680.

**Exercise 6**

Pseudo-code:

**Define function Qvalue:**

%%calculate the normalized Q value of continuous state given one action(using 6 radial basis functions (Gaussian distributions))%%

Q(s,a) = ( is the radial function)

3 Inputs: weight (1\*6 vector), state, variance

1 output: Q value of the state, given one direction

**Main functions:**

Initialize and give the parameters.

For each iteration

Using ε-greedy policy to decide action and s’(s’= s+a+N (0,0.01)).

Using s’ to decide the reward.

for all basis function parameter

update the weight\_left (1\*6 vector):

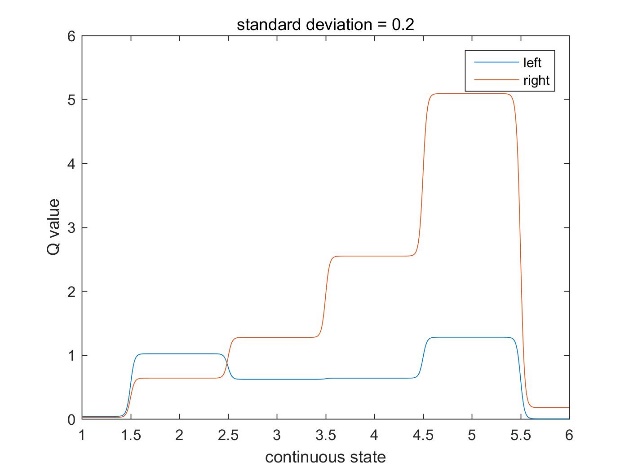
end

judge whether s’ reachs the terminal state

Until convergence

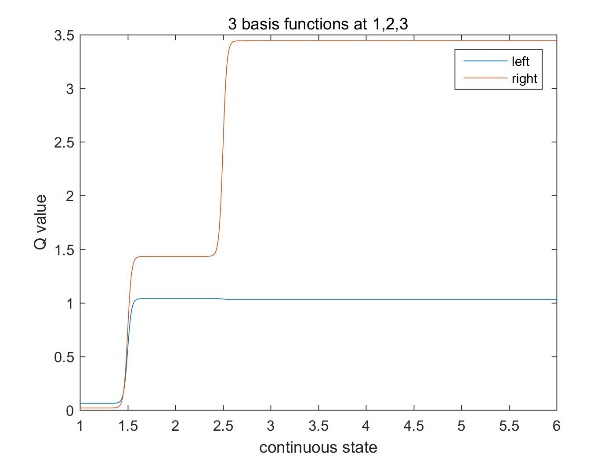
I tried different number of basis function with same Q-learning parameters.

Figure below shows the curve of 6 basis functions at {1,2,3,4,5,6}. Compare to the result of discrete Q-learning, we find that the approximation of continuous state is accurate in this situation.



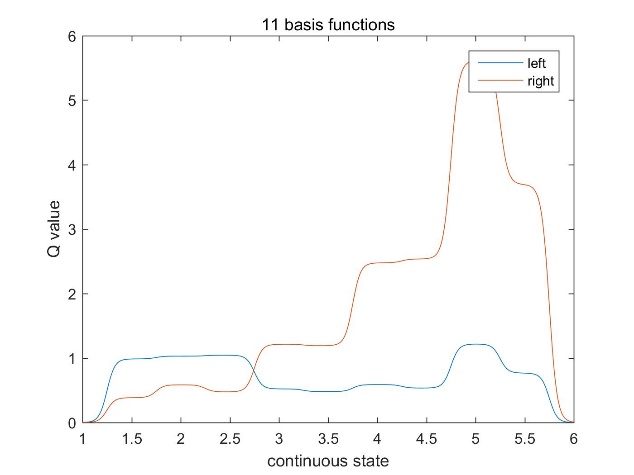
6 basis functions

Figure below shows the curve of 3 basis functions at {1,2,3}. It is obviously that only the states in (1,3) can get a not bad approximation.



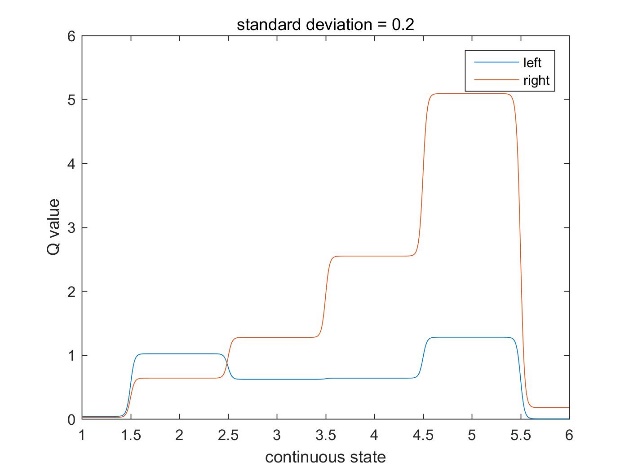
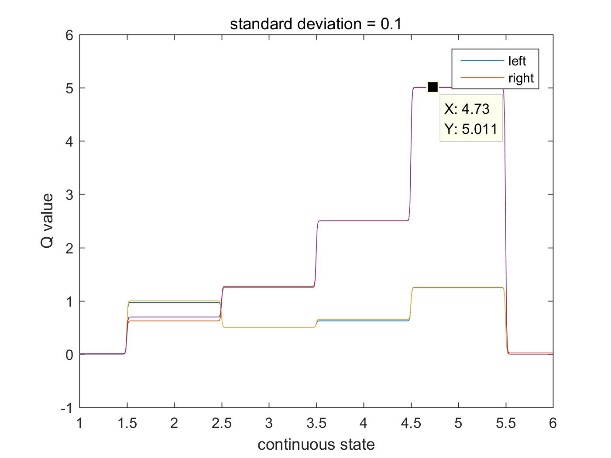
3 basis functions

As shown in figure below, I tried 11 basis functions at {1, 1.5 , 2 , 2.5 , 3 , 3.5 , 4 , 4.5 , 5 , 5.5 ,6}. We can observe the distortion of the Q on the continuous state. It is a kind of overfitting.

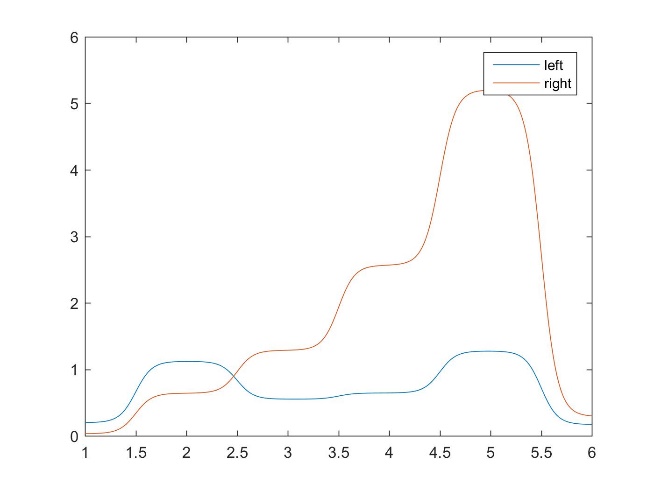


11 basis functions

Then, I tried different width (standard deviation) of basis function. As shown below, bigger the width, smoother the curve is. Large width can give us a smooth and more accurate approximation.



width = 0.1 width = 0.2



width = 0.4