Q-LEARNING

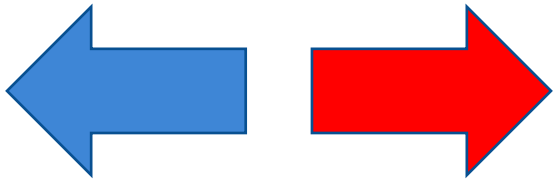
Q

A SIMPLE GAME

States

Rewards 0 0 0 0 10

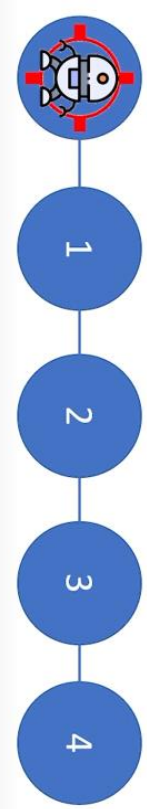
Actions



Q - TABLE

Actions

States



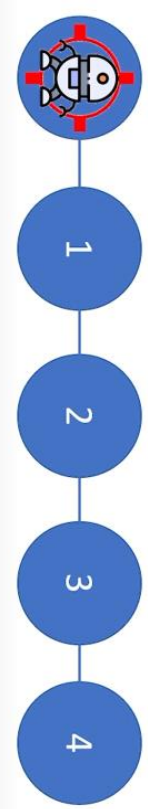
5.9	6.7
6.7	7.3
7.3	8.1
8.1	9
10	10

Q - TABLE

Q(state, action) = future reward from playing strategy from that position

Actions

States



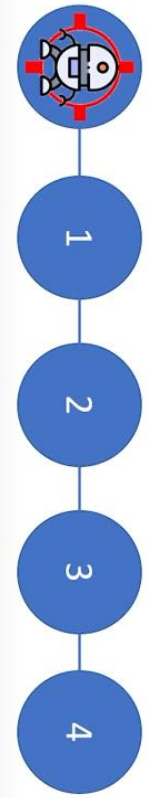
5.9	6.7
6.7	7.3
7.3	8.1
8.1	9
10	10

CHOOSING ACTION WITH Q-TABLE

From a state (row in the Q-table), choose the action with the highest Q-value

Actions

States



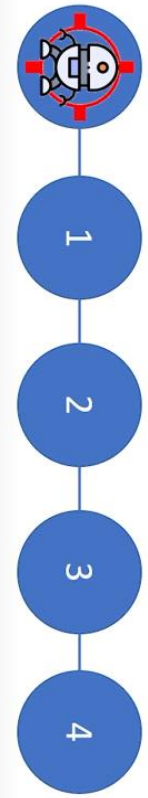
←	→
5.9	6.7
6.7	7.3
7.3	8.1
8.1	9
10	10

CHOOSING ACTION WITH Q-TABLE

From a state (row in the Q-table), choose the action with the highest Q-value

Actions

States



←	→
5.9	6.7
6.7	7.3
7.3	8.1
8.1	9
10	10

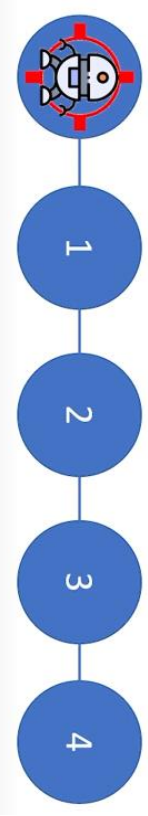
CHOOSING ACTION WITH Q-TABLE

From a state (row in the Q-table), choose the action with the highest Q-value

Actions

States

Action =

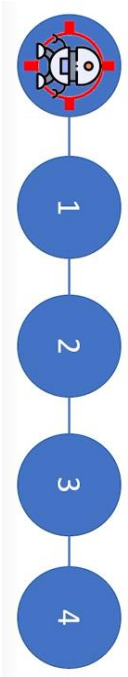


5.9	6.7
6.7	7.3
7.3	8.1
8.1	9
10	10

ONE STEP LOOK-AHEAD Q-VALUE

- Assume we are in state 3 and choose action "go right"

$$Q(3, \text{go right}) = 9$$



States

Actions

←	→
5.9	6.7
6.7	7.3
7.3	8.1
8.1	9
10	10

ONE STEP LOOK-AHEAD Q-VALUE

- Assume we are in state 3 and choose action "go right"

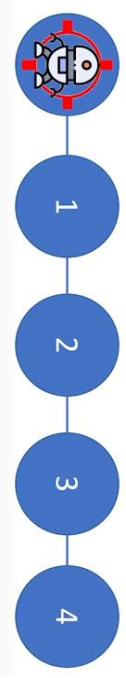
$$Q(3, \text{go right}) = 9$$

- We can get Q value by looking one step ahead

$$Q(3, \text{go right}) = 0 + \gamma \max_a Q(4, a)$$

Actions

States



←	→
5.9	6.7
6.7	7.3
7.3	8.1
8.1	9
10	10

ONE STEP LOOK-AHEAD Q-VALUE

- Assume we are in state 3 and choose action "go right"

$$Q(3, \text{go right}) = 9$$

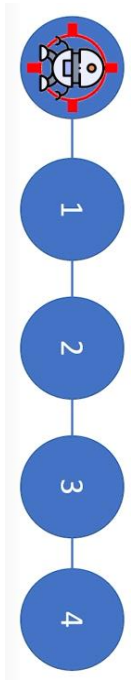
- We can get Q value by looking one step ahead

$$Q(3, \text{go right}) = 0 + \gamma \max_a Q(4, a)$$

Immediate reward

Discount factor

Future Reward (if playing best action)



States

Actions

5.9	6.7
6.7	7.3
7.3	8.1
8.1	9
10	10

Q-LEARNING

- Q-learning has us update the Q-value as the weighted average of the current value and the one-step look ahead value

$$Q(3, \text{go right}) = (1 - \alpha)Q(3, \text{go right}) + \alpha(0 + \gamma \max_a Q(4, a))$$

Learning rate

EPSILON GREEDY Q-LEARNING

- We choose a probability ε between 0 and 1

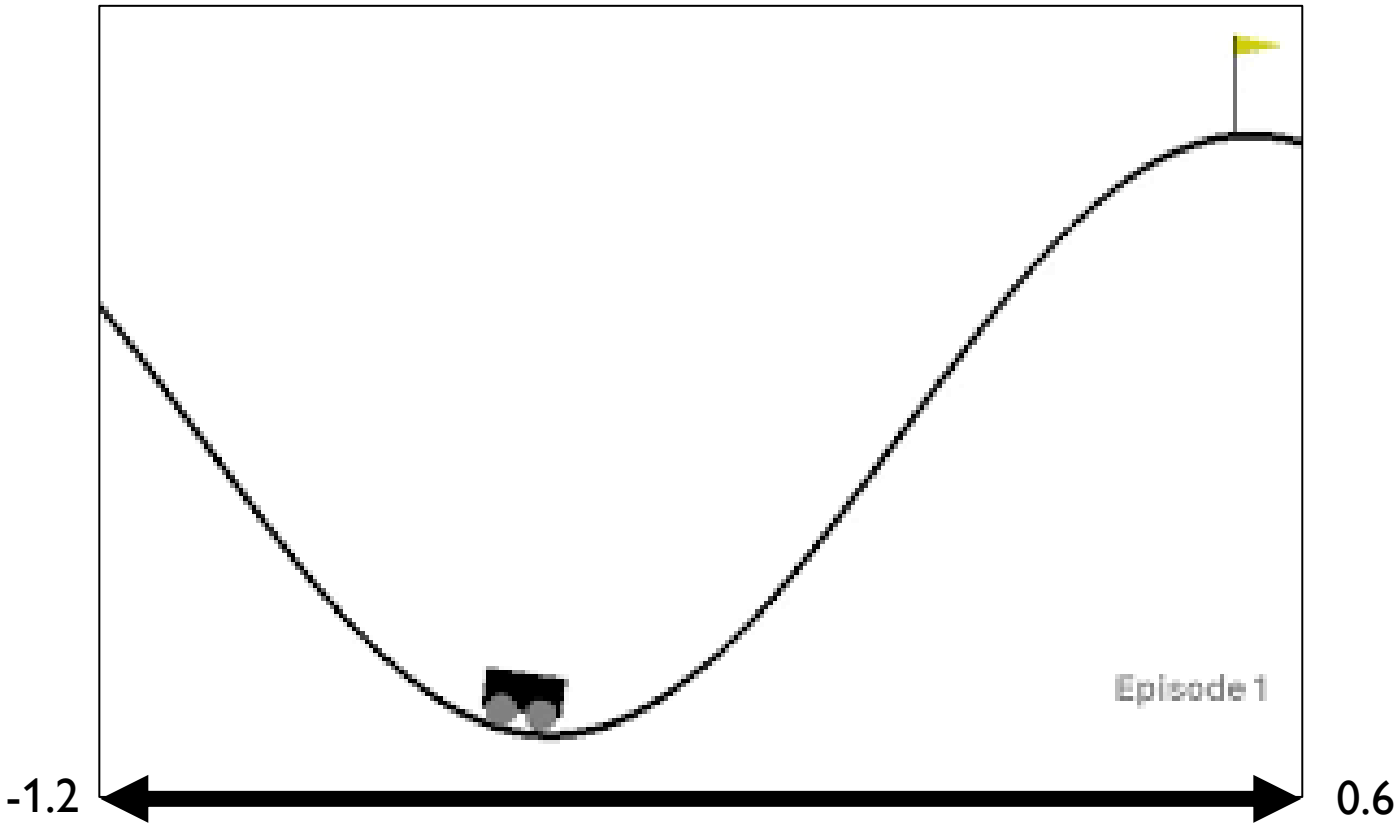
- In each step of the simulation,
  - with probability ε pick a random action
  - with probability 1- ε pick the best action from the current Q-table

- Decrease ε after each episode is complete (don't want to be doing random actions forever)

Q-LEARNING FACTS

- Q-learning is best when we have **discrete** states and actions
- If states or actions are continuous, we have to make them discrete
  - Ex) state = (0.41, 0.94) -> (4, 9)
- Q-learning can avoid getting "stuck" when there are no reward updates
  - Policy gradient got stuck a lot

MOUNTAIN CAR



$-1.2 \leq x \leq 0.6$

$-0.7 \leq \text{velocity} \leq 0.7$