Exploring Q-Learning for Pendulum Swing-Up Contro

Venkata Amith Palacherla

January 16, 2023

Abstract

The project's objective is to learn a policy for an inverted pendulum model to make it make a swing-up motion. Q learning is used to learn the policy, which is model free. Policies were learned for two controls (-4,0,4) and (-5,0,5).

1 Part 1

1.1 Basics

State of the pendulum is given by -

$$x = \begin{bmatrix} \theta \\ \omega \end{bmatrix}$$

where θ is the angle pendulum makes with the vertical and ω is it's angular velocity.

The pendulum is controlled by producing torque at its pivot point.
States θ and ω are discreatized between $(0, 2\pi)$ and $(-6, 6)$.

The discounted cost function is defined as -

$$\sum_{i=0}^{\infty} \alpha^i g(x_i, u_i)$$

Where,

$$g(x_i, u_i) = (\theta - \pi)^2 + 0.01 \cdot \dot{\theta}_i^2 + 0.0001 \cdot u_i^2$$

Cost penalizes when the pendulum is away from the inverted position.

1.2 Q learning with table

Q-table defines the quality of the state and action pair. I will is compromised on quality value for all possible state-action pairs. This specific problem has 2500 possible states after discretization and 3 possible controls. Therefore the shape of the Q table will be **2500x3**. Q value for each state-action pair is given by -

$$Q(x_t, u_t) = g(x_t, u_t) + \alpha \min_u Q(x_{t+1}, u)$$

The Q-table is updated during training based on the observed rewards and the Q-values for the next states.

1.3 Optimal Policy and Value function

In Q -learning, the Optimal policy and value function, given a Q-table and state(x_t) are -

$$J^*(x_t) = \min_u Q(x_t, u)$$

$$\mu^*(x) = arg \min_u Q(x_t, u)$$

A function is defined in the python notebook to do the same.

1.4 ϵ-greedy policy

Q-learning algorithms often include an exploration rate(ϵ), which determines the probability that the agent will choose a random action rather than the action with the highest estimated reward. This allows the agent to explore the action space and gather more information about the environment, which can improve the accuracy of the Q-table and the performance of the learned policy.

$$u_t = \begin{cases} arg \min Q(x_t, u), & \textit{with probability } 1 - \epsilon \\ \textit{random action} & \textit{with probability } \epsilon \end{cases} \quad (1)$$

1.5 Q learning

During training, the agent continually updates the Q-table based on its experiences in the environment. At each time step, the agent selects an action according to an action selection policy, such as a greedy policy or an epsilon-greedy policy. It then receives a reward and transitions to a new state, and the process is repeated. Over time, the agent's estimates of the rewards for different actions in different states will converge towards the true values, and the agent will learn an optimal policy that maximizes its long-term reward.
The delta error is calculated by -

$$\delta_t = g(x_t, u_t) +_u Q(x_{t+1}, u) - Q(x_t, u_t)$$

2 Part 2

The following section discusses the results of Q learning with possible controls $[-4, 0, 4]$.
With $[-4, 0, 4]$ as possible controls, Q-learning algorithms is taking 5000 episodes to train. Although it is being trained for 10,000 episodes, the learning plateaus after 5000 episodes. The following can be

Algorithm 1 Q-learning

```
1: Set $\gamma \in [0,1]$, $\epsilon$ and $\alpha$.
2: Initialize $Q(x,u)$ for all states and controls .
3: for episode in episodes do
4:     Initialize state $x_0$
5:     for step in episode do
6:         Choose a control using $\epsilon$-greedy policy from $Q$
7:         calculate $x_{t+1}$
8:         Compute $g(x_t, \mu(x_t))$
9:         Compute delta error using $\alpha$
10:        Update $Q(x_t, u_t)$
11:    end for
12: end for
```

seen from the cost vs episode plot below.

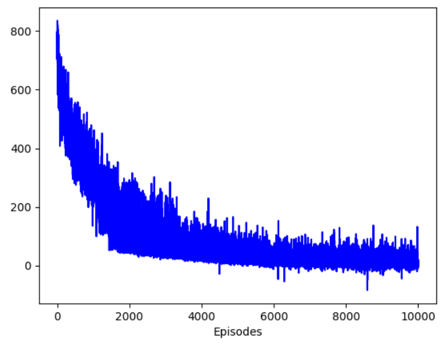

Figure 1: Plot of cost vs episode

It took two back and forth swings to go into inverted position. This can be seen from the evolution plot of theta and omega below. It took almost 4 seconds to reach the inverted position.

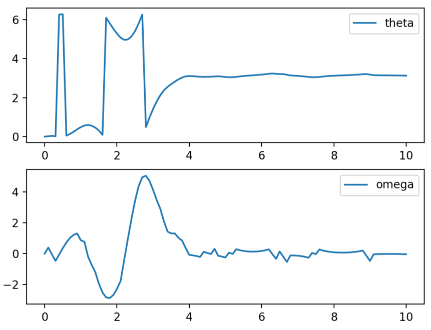

Figure 2: Plot of state vs time

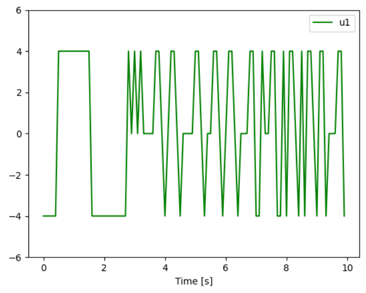

Figure 3: Plot of control vs time

The value function for different states is shown below-

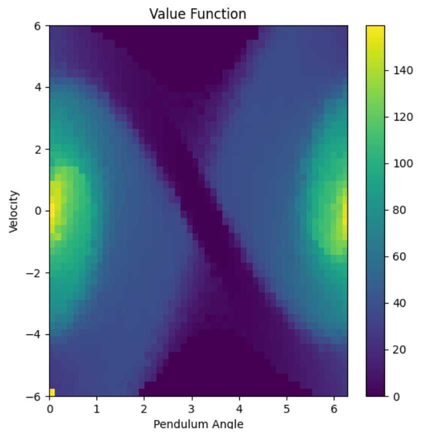

Figure 4: Plot of state vs value

The policy for different states shown below. Yellow corresponds to the control value 4 and blue corresponds to -4.It can be seen that for majority of states with velocity more than 0 control value is 4.

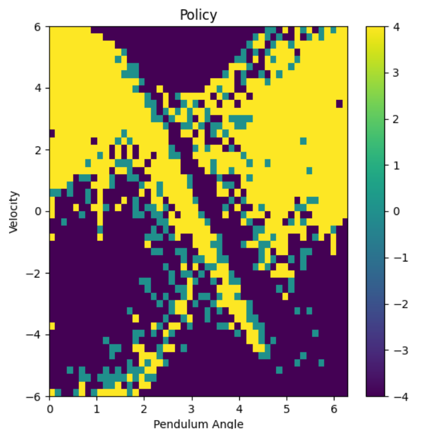

Figure 5: Plot of state vs Policy

3 Part 3

The following section discusses the results of Q learning with possible controls $[-5, 0, 5]$.
With $[-5, 0, 5]$ as possible controls, Q-learning algorithms took 5000 episodes to train. Although it was

trained for 10,000 episodes, the learning plateaus after 5000 episodes. The following can be seen from the cost vs episode plot below.

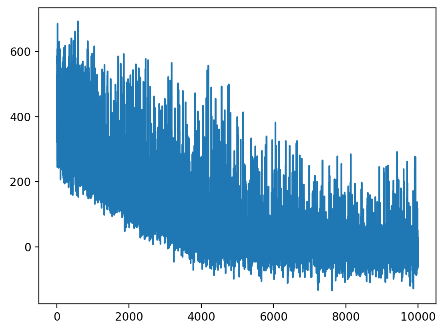

Figure 6: Plot of cost vs episode

It took one back and forth swings to go into inverted position. This can be seen from the evolution plot of theta and omega below. The pendulum goes into inverted position in 2 seconds. When compared [-4,0,4] controls it took 2 swings less and was 2 seconds quicker to go into inverted position.

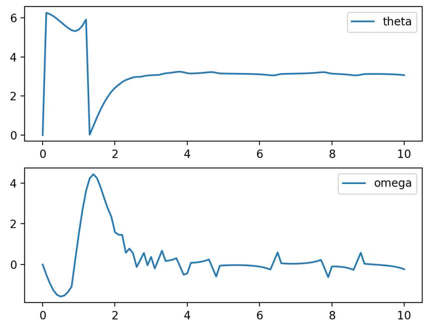

Figure 7: Plot of state vs time

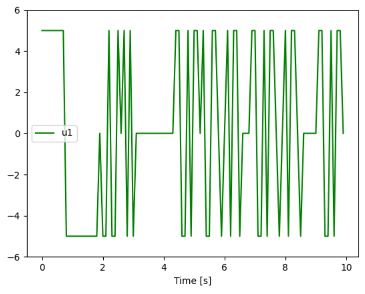

Figure 8: Plot of control vs time

The value function for different states is shown below-

The policy for different states shown below. Yellow corresponds to the control value 5 and blue corresponds to -5.It can be seen that for majority of states with velocity more than 0 control value is 5. The value and policy function for controls [-4,0,4] and [-5,0,5] were same. There wasn't any major different in both the plots.

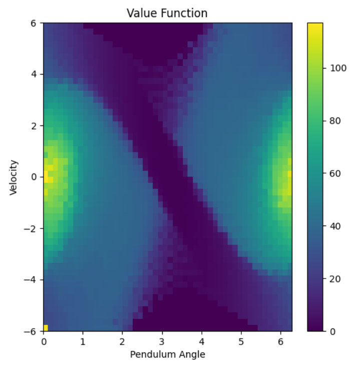

Figure 9: Plot of state vs value

The policy depends on velocity more than θ. This can be seen from both state vs policy plots and control,velocity vs time plots.
Even after the pendulum reaches the inverted position, control values are not zero rather they are extreme values of -4 and 4. This is because of fluctuations in velocity of pendulum. For small positive fluctuation of velocity extreme positive value is applied on the system and opposite for small negative fluctuation.

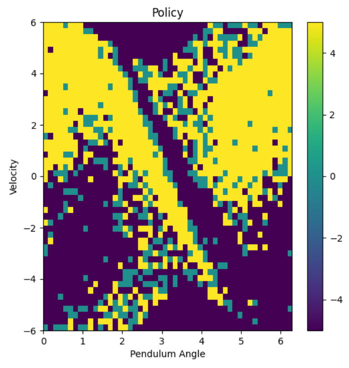

Figure 10: Plot of state vs policy

4 Part 4

In this section the effect of parameters in Q-learning will be discussed. Two important parameters are -
The learning rate(Step size):This determines the extent to which new information will overwrite old information in the Q-table. A higher learning rate means that new information will be given more weight, while a lower learning rate means that old information will be more persistent.

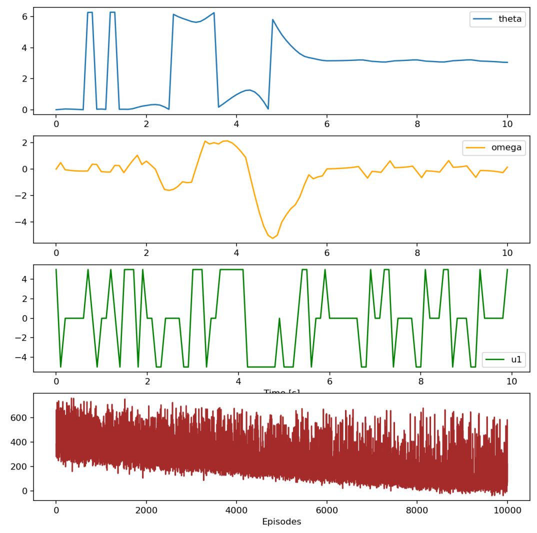

Figure 11: Plot of state,control and learning plots with eta 0.2

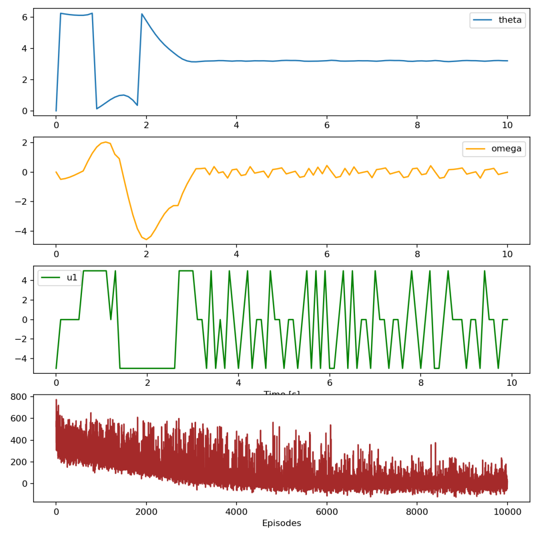

Figure 12: Plot of state,control and learning plots with eta 0.5

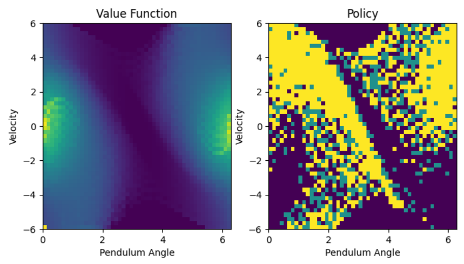

Figure 13: Value function and policy plots with step size 0.1

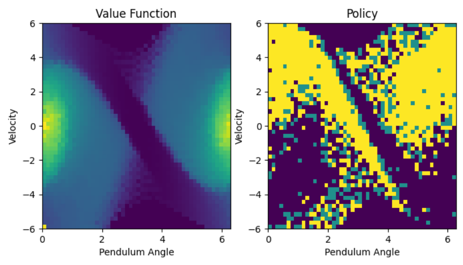

Figure 14: Value function and policy plots with step size 0.3

With increasing step size the learning rate becomes quicker. This can be seen from the learning plot. In the first plot step size was set to 0.1 and it took almost 8000 episodes to decay while with 0.3 step size it only took 4000 episodes to decay. As the step size increases more weight is given to new information.

The exploration rate(eta): This determines the probability that the agent will choose a random action rather than the action with the highest estimated reward. A higher exploration rate means that the agent is more likely to explore new actions, while a lower exploration rate means that the agent is more likely to exploit its current knowledge.

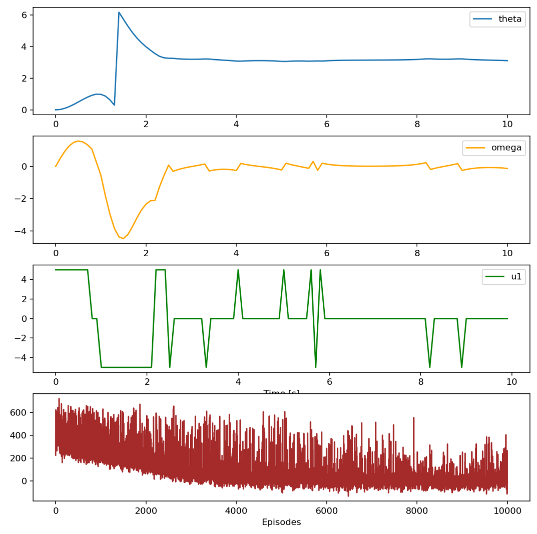

Figure 15: Plot of state,control and learning plots with eta 0.2

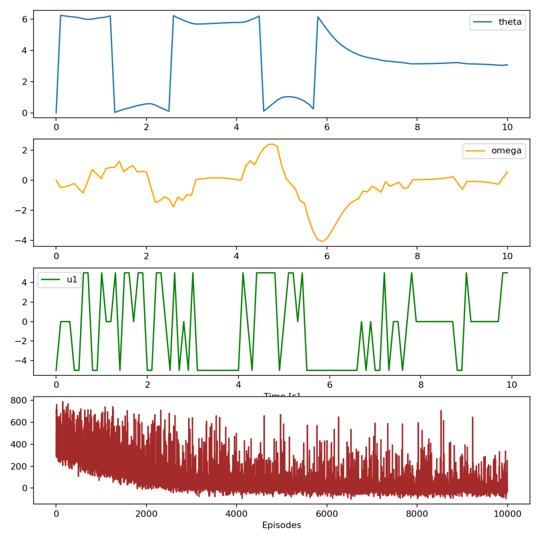

Figure 16: Plot of state,control and learning plots with eta 0.5

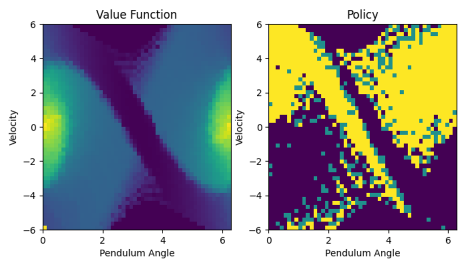

Figure 17: Value function and policy plots with eta 0.2

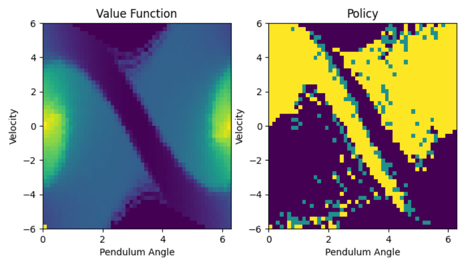

Figure 18: Value function and policy plots with eta 0.5

With increasing eta it was seen that learning decayed quickly. With eta value as 0.5 it took 2000 episodes to decay where as with 0.2 it took close to 4000 episodes. Step size was set to 0.3 for both the eta values . This is expected because with higher eta value there will be chance of more exploration and therefore less episodes are required to train. After training with higher eta value it took more swings to reach inverted position.