Reinforcement Learning and Optimal Control for Robotics ROB-GY 6323 Exercise Series 4

December 9, 2024

JITHIN GEORGE
Univ ID: N10719458
Net ID: jg7688

Question 1 Implement the Q-learning algorithm.

Answer We have been given the algorithm as,

Algorithm 1 Q-Learning Algorithm

for each episode **do**
  Initialize the episode: $x_0 = [0, 0]$
  for each step of the episode **do**
    Select u_n using an ϵ-greedy policy
    Compute the next state: x_{n+1}
    Compute the target: $y_n = g(x_n, u_n) + \alpha \min_a Q(x_{n+1}, a)$
    Perform one SGD step on the neural network parameters to minimize:

$$(Q(x, u) - y_n)^2$$

  end for
end for

Implementing this in code, we have the loop as shown below. The code can be found in the attached jupyter notebook as well.

```
for prgs_bar in tqdm(range(max)):

    x_init_np = np.zeros([1, 2])
    x_init = torch.tensor(x_init_np, dtype=torch.float32)

    for i in range(epis_len):

        x = q_function(x_init.unsqueeze(0))

        if torch.rand(1) > epsilon:
            index = torch.argmin(x).item()
            u = possible_controls[index]
            q_xu = x.squeeze()[index]
        else:
            index = torch.randint(0, 3, (1,)).item()
            u = possible_controls[index]
            q_xu = x.squeeze()[index]

        with torch.no_grad():
            state_next = torch.tensor(
                pendulum.step(x_init.numpy().flatten(), u),
                dtype=torch.float32
            )
            y_n = torch.tensor(
                cost(x=x_init.numpy().flatten(), u=u) +
                alpha * torch.min(q_function.forward
                                  (state_next.unsqueeze(0))).item(),
                dtype=torch.float32
            )

        loss = loss_fn(q_xu, y_n)

        optimizer.zero_grad()  # Reset gradients
        loss.backward()        # Backpropagation
        optimizer.step()       # Update parameters

        x_init = state_next
```

Question 3 Plot the cost per episode (to visualize learning)

Answer The plot is shown below -

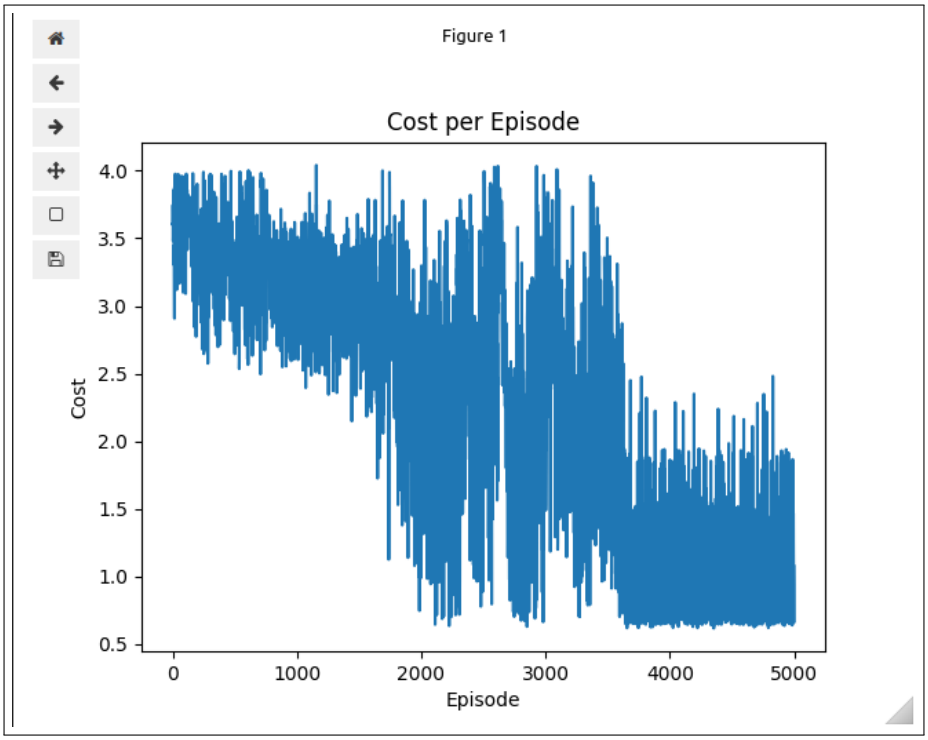

Figure 1: Cost Per Episode

Question 4 Plot the learned value function (in 2D as a function of pendulum position and velocity) as well as the policy.

Answer The plots are shown below -

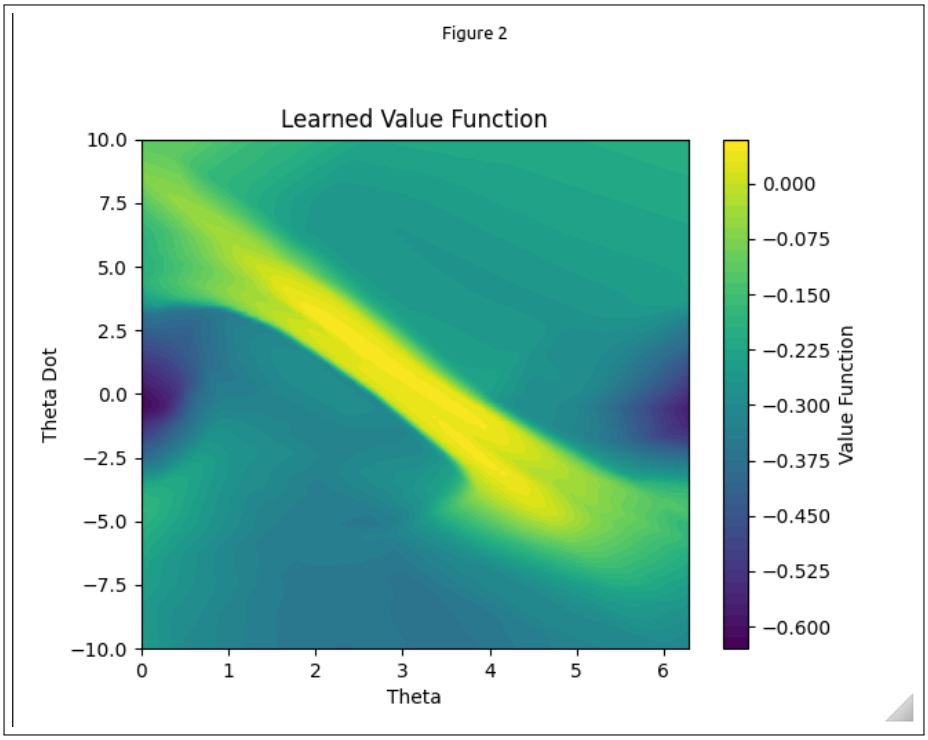

Figure 2: Value Function Plot

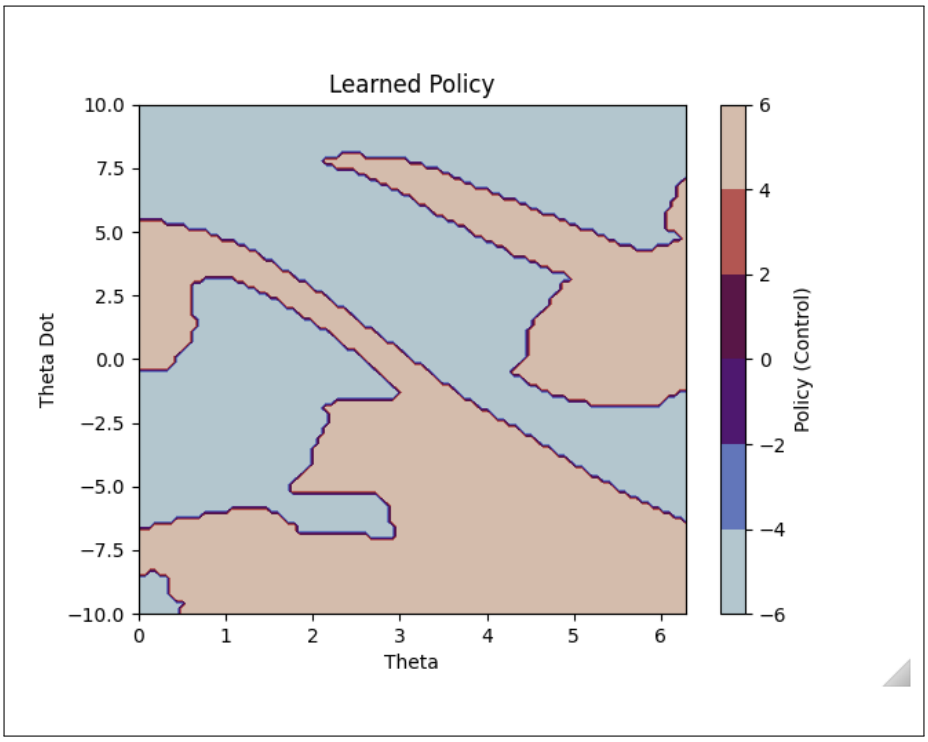

Figure 3: Learned Policy