CS5180 Ex5

Sudhendra Kambhamettu
NUID: 002786797

Q1.

Part a)

We know that,

$$C_n = \sum_{k=1}^{n} W_k$$

$$V_{n+1} = \frac{\sum_{k=1}^{n} W_k G_k}{C_n}$$

Re-arranging the terms

$$V_{n+1} \times C_n = \sum_{k=1}^{n} W_k G_k$$

This can also be written as

$$= W_n G_n \sum_{k=1}^{n-1} W_k G_k$$

From the previous equation, we can modify this equation as

$$= W_n G_n + V_n \sum_{k=1}^{n-1} W_k$$

$$= W_n G_n + V_n C_{n-1}$$

Again, modifying

$$= W_n G_n + V_n (C_n - W_n)$$

$$\vdots$$

$$\therefore \boldsymbol{V_{n+1} = V_n + \frac{W_n}{C_n}[G_n - V_n]}$$

Part b)

A_t is only allowed to change W if $A_t = \pi(S_t)$. And since we know that our policy is a deterministic policy, a greedy one, we are only observing trajectories where $\pi(A_t|S_t) = 1$ during the update of W. Hence, we can say that the numerator is 1 & is correct.

Q2.

Part a)

The hint provided suggests imagining a situation where you have extensive experience driving home from work but then move to a new building and parking lot, although you still enter the highway at the same point. Now, as you start learning predictions for the new building, the question hints that TD updates are likely to be much better, at least initially, than Monte-Carlo updates.

A concise answer to this question would point out that TD methods can update their estimates based not just on final outcomes but on every step of experience. This characteristic allows them to learn more quickly in environments where certain aspects remain consistent even as others change (like the highway entrance in the example). In the specific scenario provided, if you have a lot of experience driving home and then your starting point changes (new building and parking lot), TD methods can quickly adjust the estimated times from the new starting point by incorporating the known values (from previous experiences) as soon as you hit the familiar part of the journey (entering the highway). This is because TD methods update estimates partly based on other learned estimates (bootstrapping), allowing for quicker adaptation when only part of the environment has changed, as opposed to waiting for the outcome of the entire episode as Monte-Carlo methods do.

Part b)

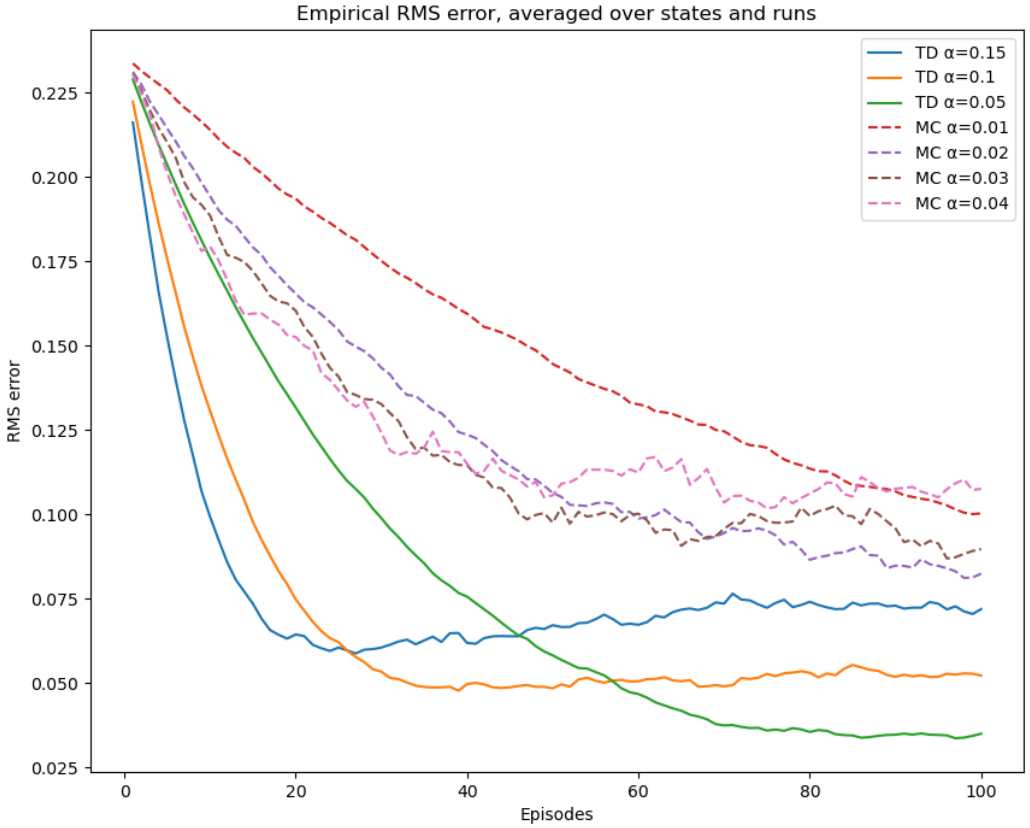
The Monte-Carlo approach can be advantageous in environments where the model is non-stationary or when dealing with very long episodes. In non-stationary environments, the outcomes can change over time, making the immediate updates provided by TD methods potentially less reliable because they are based on assumptions that the environment's dynamics remain consistent. Monte-Carlo methods, which wait until the end of an episode to update value estimates, are not affected by changes that occur during the episode. This makes them potentially more robust in such contexts because they base updates on actual outcomes rather than predictions of future states.

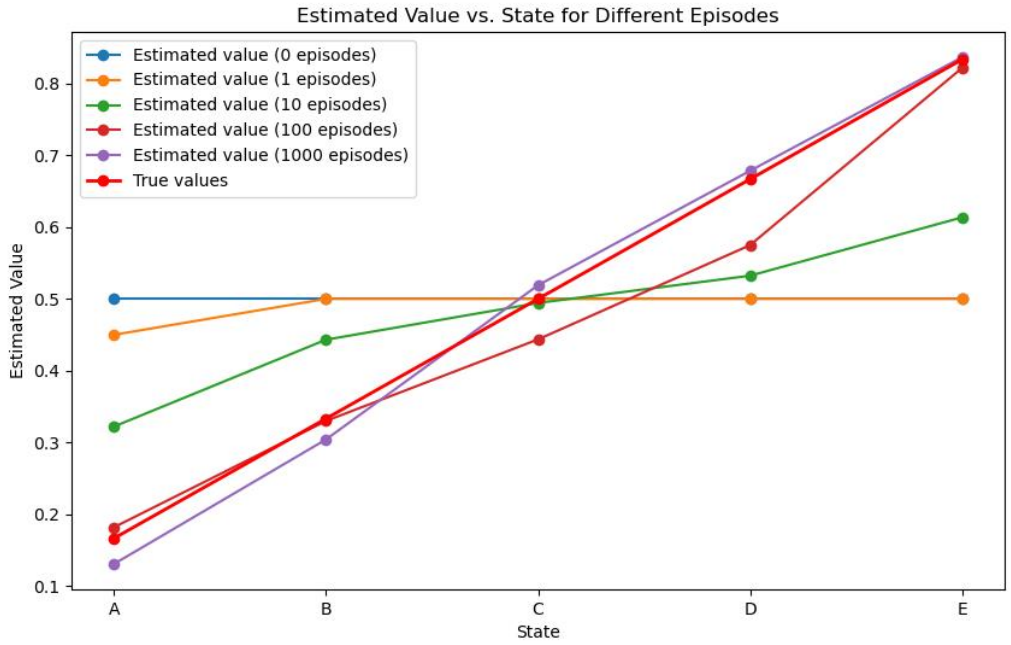
Furthermore, in tasks with very long episodes or in continuous tasks without clear terminal states, the delay in feedback inherent to Monte-Carlo methods is not as significant a drawback. For example, in strategic games or simulations where outcomes are only determined after many moves, the full-episode feedback provided by Monte-Carlo methods can offer a clearer picture of the effectiveness of early actions than TD methods, which might prematurely adjust value estimates based on incomplete information.

In summary, while TD methods are generally more efficient and faster to converge in many scenarios due to their bootstrapping approach, Monte-Carlo methods hold an edge in environments where the accuracy of full-episode outcomes outweighs the benefits of immediate, incremental updates.

Q3.

Plots


Empirical RMS error, averaged over states and runs
TD α=0.15
TD α=0.1
TD α=0.05
MC α=0.01
MC α=0.02
MC α=0.03
MC α=0.04
RMS error
Episodes
0.225
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0
20
40
60
80
100


Estimated Value vs. State for Different Episodes
Estimated value (0 episodes)
Estimated value (1 episodes)
Estimated value (10 episodes)
Estimated value (100 episodes)
Estimated value (1000 episodes)
True values
Estimated Value
State
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
A
B
C
D
E

Written

Part a)

The first graph indicates that only the value of state A was updated after the first episode, suggesting that the episode ended before any other states were visited. The V(A) is changed by a difference of 0.05, meaning 0.45 after the first episode as observed empirically & as shown in the graph.

Part b)

The observation from the graph is that the TD method's long-term accuracy decreases with higher alpha values due to potential oscillations around the true value function. For the MC method, different alpha values don't show a clear difference in performance on the plot, suggesting that the method's main limitation might be the fewer number of updates it can make compared to the TD method. The TD method benefits from a greater number of value function updates within the same number of episodes because it updates more frequently within each episode.

Part c)

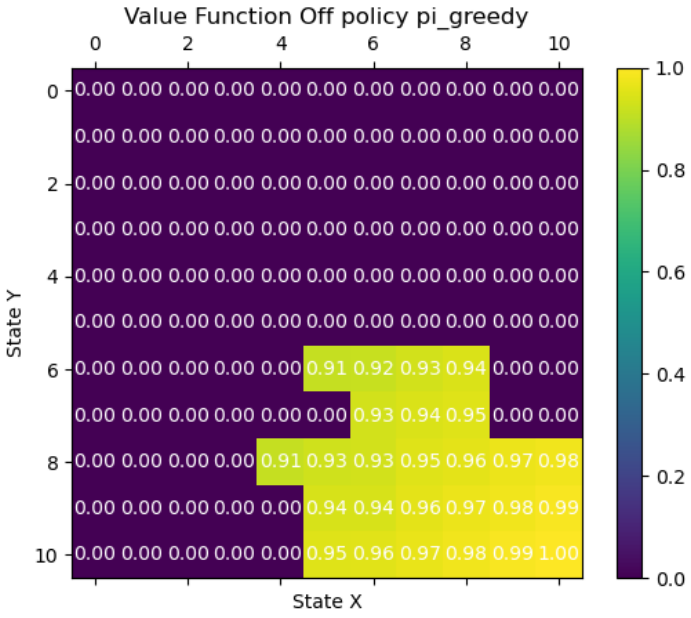
The observation of a somewhat "U" shape in the RMS error for the TD method, especially at higher alpha values, suggests that with a large step-size parameter, the weight given to the TD error might be exaggerating small errors. This can cause the estimated value function to oscillate and potentially diverge, leading to an increase in RMS error. A high learning rate causes the estimates to bounce around the true value without converging, thus drifting away from the correct estimate and increasing the RMS error.

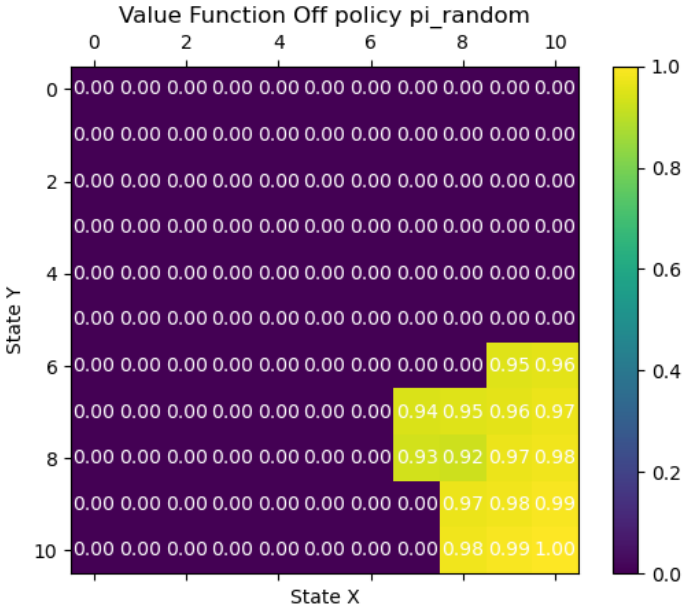
This behavior is indeed expected with high alpha values, as they tend to amplify the updates from each individual sample, leading to instability in learning. It is a reflection of the trade-off between learning speed and stability: while a higher alpha can accelerate learning initially, it can also cause the value estimates to overshoot and fail to settle at the true values.

Q4.

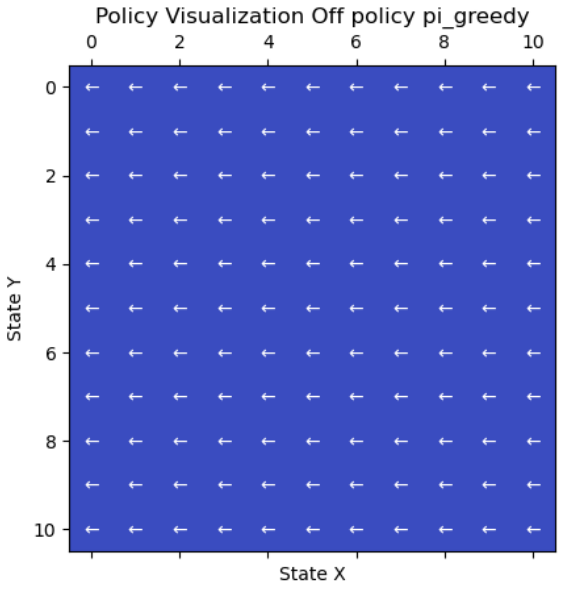
Off Policy Plots

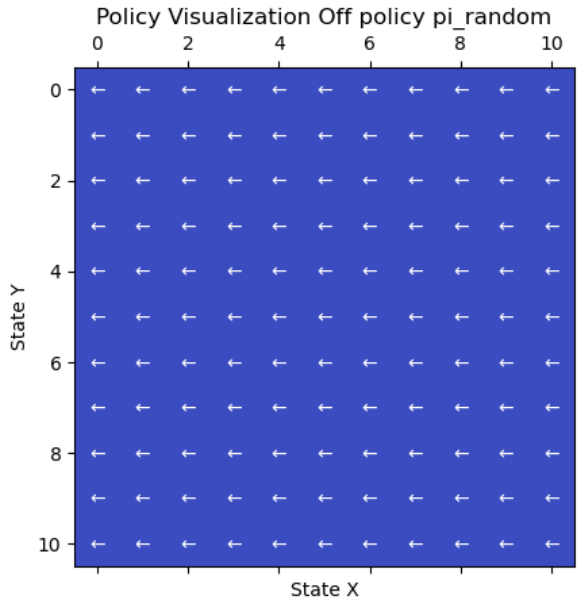
Value functions





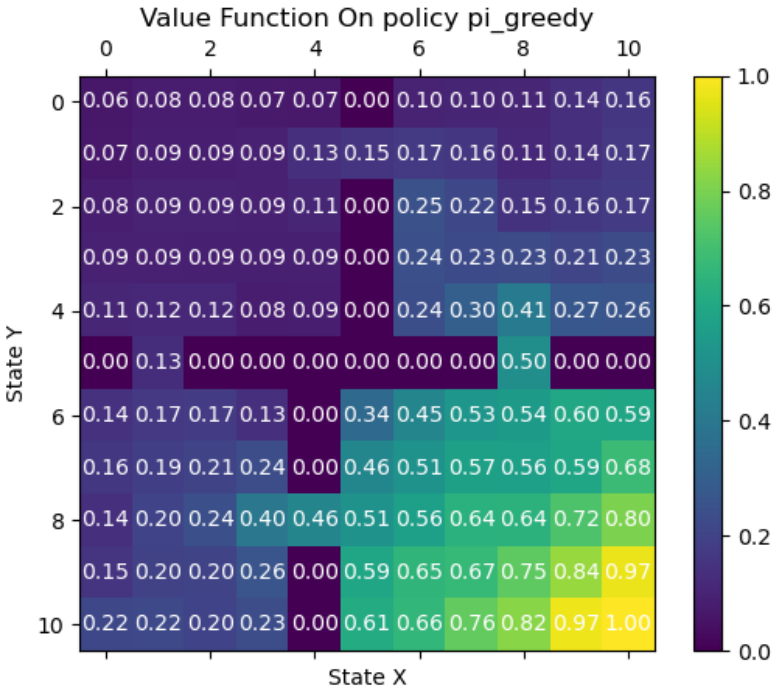
Policy


Policy Visualization Off policy pi_greedy
0
2
4
6
8
10
State Y
State X

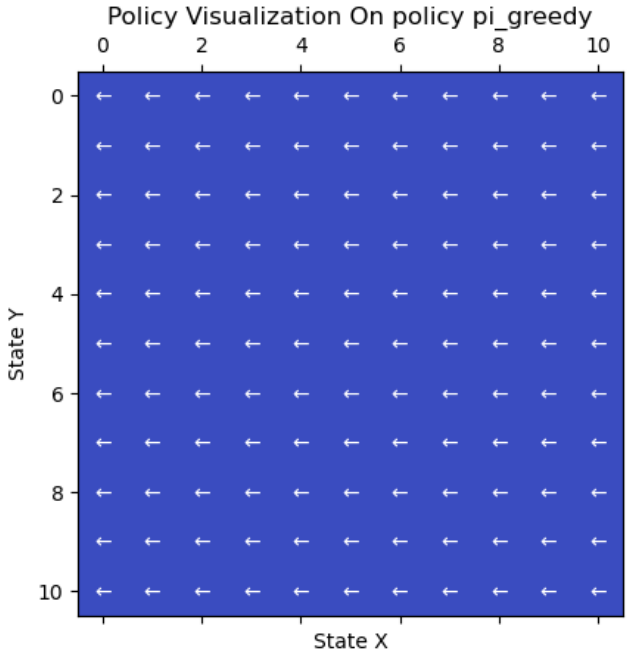

Policy Visualization Off policy pi_random
0
2
4
6
8
10
State Y
State X

On-policy Plots

Value function



Policy



Q5.

The form is filled and submitted