CS 4180/5180: Reinforcement Learning and Sequential Decision Making (Fall 2020)– Lawson Wong

Name: [Virender Singh]
Collaborators: [Sunny Shukla]

To run the codes please use BlackJack.py, Q4.py and the Q6.py included in the src folder. All the plots can also be seen in the plt folder.

Question 1. *(a) Modify the algorithm for first-visit MC policy evaluation (Section 5.1) to use the incremental implementation for sample averages described in Section 2.4.*
(b) The pseudocode for Monte Carlo ES is inefficient because, for each state-action pair, it maintains a list of all returns and repeatedly calculates their mean. It would be more efficient to use techniques similar to those explained in Section 2.4 to maintain just the mean and a count (for each state–action pair) and update them incrementally. Describe how the pseudocode would be altered to achieve this.

Response:
(a)

$$\begin{aligned} V_n(S_t) &= \frac{1}{n}\Sigma_{i=1}^{n} G_i(t) \\ &= \frac{1}{n}[G_n(t) + \Sigma_{i=1}^{n-1} G_i(t)] \\ &= \frac{1}{n}[G_n(t) + \frac{n-1}{n-1}\Sigma_{i=1}^{n-1} G_i(t)] \\ &= \frac{1}{n}[G_n(t) + (n-1)V_{n-1}(S_t)] \\ &= \frac{1}{n}[nV_{n-1}(S_t)] + \frac{1}{n}[G_n(t) - V_{n-1}(S_t)] \\ &= V_{n-1}(S_t) + \frac{1}{n}[G_n(t) - V_{n-1}(S_t)] \end{aligned}$$

Now instead of taking average of returns at each step for calculating value for each state, we can use the above modification to update the value function.

(b)

$$
\begin{aligned}
Q_n(S_t, A_t) &= \frac{1}{n}\Sigma_{i=1}^{n} G_i(S_t, A_t) \\
&= \frac{1}{n}[G_n(S_t, A_t) + \Sigma_{i=1}^{n-1} G_i(S_t, A_t)] \\
&= \frac{1}{n}[G_n(S_t, A_t) + \frac{n-1}{n-1}\Sigma_{i=1}^{n-1} G_i(S_t, A_t)] \\
&= \frac{1}{n}[G_n(S_t, A_t) + (n-1)Q_{n-1}(S_t, A_t)] \\
&= \frac{1}{n}[nQ_{n-1}(S_t, A_t)] + \frac{1}{n}[G_n(S_t, A_t) - Q_{n-1}(S_t, A_t)] \\
&= Q_{n-1}(S_t, A_t) + \frac{1}{n}[G_n(S_t, A_t) - Q_{n-1}(S_t, A_t)]
\end{aligned}
$$

Now instead of taking average of returns at each step for calculating q-value for each state, action pair, we can use the above modification to update the q-value function.

Question 2. *(a) Suppose every-visit MC was used instead of first-visit MC on the blackjack task. Would you expect the results to be very different? Why or why not?*
(b) Consider an MDP with a single nonterminal state and a single action that transitions back to the nonterminal state with probability p and transitions to the terminal state with probability 1 p. Let the reward be +1 on all transitions, and let = 1. Suppose you observe one episode that lasts 10 steps, with a return of 10. What are the first-visit and every-visit estimators of the value of the nonterminal state?
(c) [Extra credit.] Read and understand example 5.5 first. The results with Example 5.5 and shown in Figure 5.4 used a first-visit MC method. Suppose that instead an every-visit MC method was used on the same problem. Would the variance of the estimator still be infinite? Why or why not?
Code/plot: Implement Example 5.5 and reproduce Figure 5.4 to verify your answer.

Response:

(a) In the blackjack task, the state comprises of a 3-tuple of: the players current sum, the dealer's one showing card (1-10 where 1 is ace), and whether or not the player holds a usable ace (0 or 1). Therefore, as long as the action is hit, i.e. player picks additional cards and thus the player's current sum increases at each step and when the player sticks i.e. picks no further card, then the dealer picks and the game concludes. Therefore, no state is repeated and hence we can say that in the blackjack the first visit MC would be same as every-visit MC as every state is visited only once.

(b)

For first visit value estimation :

$V_s = 1 + 1 + ... + 1/1 = 10$

For second visit value estimation :

$V_s = 1(\text{last visit}) + 2(\text{second last visit}) + 3 + ... + 10(\text{first visit})/10 = 5.5$

(c)

Yes the variance would still be infinite because the method remains still the ordinary weighted importance sampling. Let's see it mathematically, variance is given by

$$variance = E_b\left[\left(\Pi_{t=0}^{T-1}\frac{\pi(A_t|S_t)}{b(A_t|S_t)}G_0\right)^2\right] \tag{1}$$

For the case of every visit, we can rewrite it as:

$$variance = E_b\left[\left(\frac{1}{T-1}\Sigma_{k=1}^{T-1}\Pi_{t=0}^{k}\frac{\pi(A_t|S_t)}{b(A_t|S_t)}G_0\right)^2\right] \tag{2}$$

$$= 0.5 \times 0.1 \times 2^2 + \tag{3}$$

$$\frac{1}{2}[0.5 \times 0.9 \times 0.5 \times 0.1 \times 2^{2.2} + 0.5 \times 0.1 \times 2^2] + \tag{4}$$

$$\frac{1}{3}[0.5 \times 0.9 \times 0.5 \times 0.9 \times 0.5 \times 0.1 \times 2^{2.3} + \tag{5}$$

$$0.5 \times 0.9 \times 0.5 \times 0.1 \times 2^{2.2} + 0.5 \times 0.1 \times 2^2] \tag{6}$$

$$= 0.1\Sigma_{k=1}^{\infty}\frac{1}{k}\Sigma_{l=0}^{k-1}0.9^l \times 2^l \times 2 \tag{7}$$

$$= 0.2\Sigma_{k=1}^{\infty}\frac{1}{k}\Sigma_{l=0}^{k-1}1.8^l = \infty \tag{8}$$

Question 3.

Response:

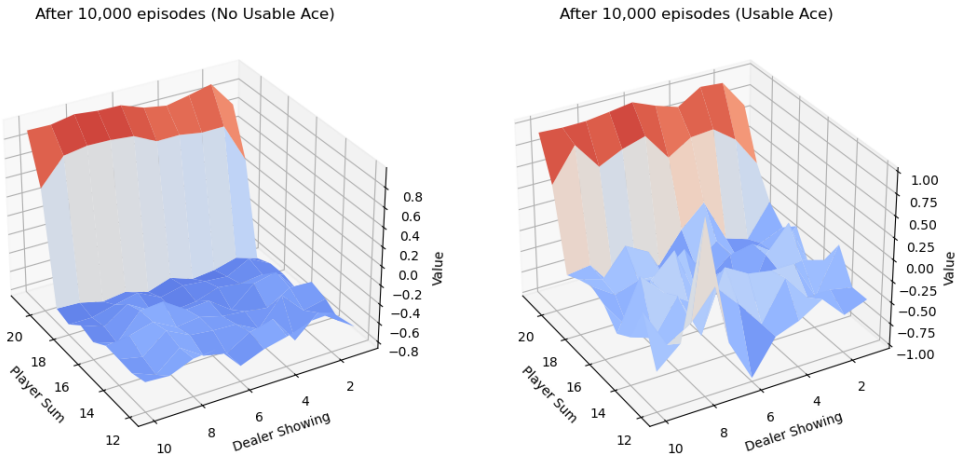

Figure 1: Q3(a) : Value function for MC policy evaluation for 20 or 21 policy after 10000 episodes

(a) 1 shows the approximate state value functions for the ”sticks only on 20 or 21” policy for both usable and non usable aces for 10000 episodes.
2 shows the approximate state value functions for the ”sticks only on 20 or 21” policy for both usable and non usable aces for 500000 episodes.
(b) 4 shows the approximate state value functions for the Monte-Carlo control with exploring starts for both usable and non usable aces for 500000 episodes.
3 shows the optimal policy and state-value function for blackjack, found by Monte Carlo ES.

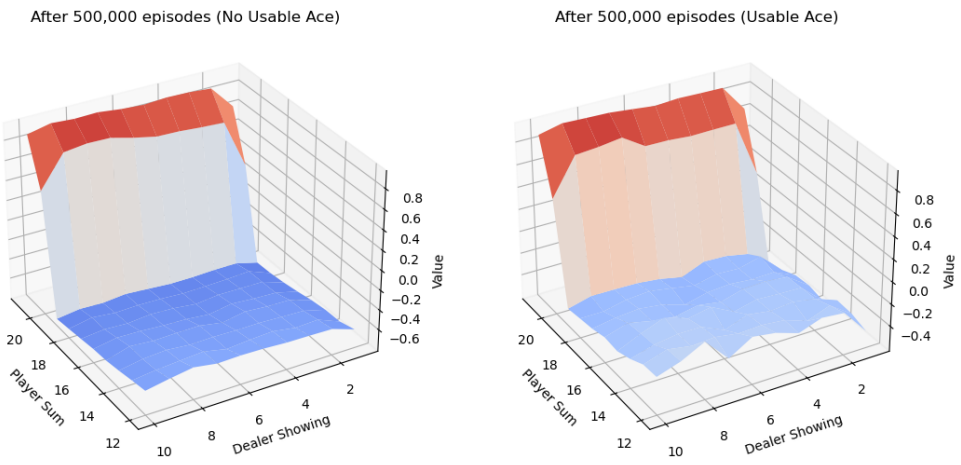

Figure 2: Q3(b) : value function for MC policy evaluation for 20 or 21 policy after 500000 episodes

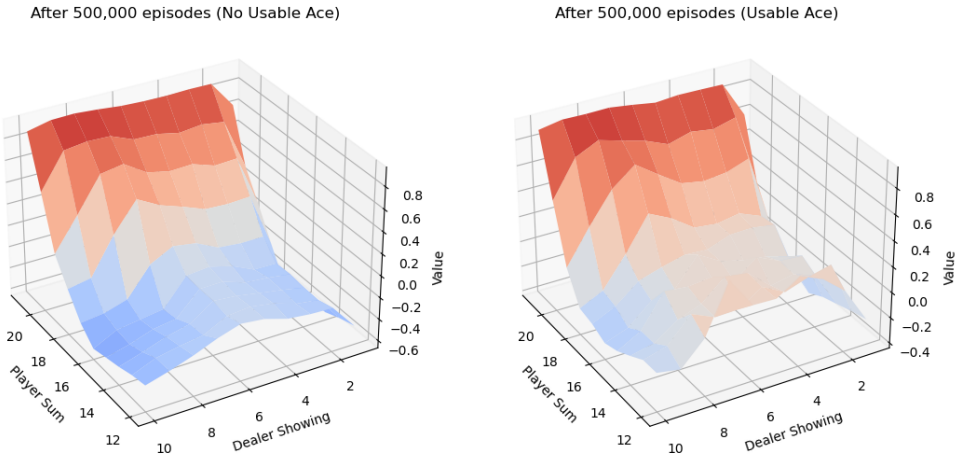

Figure 3: Q3(a) : Value function for Monte Carlo Control with exploring starts after 500000 episodes

Question 4.

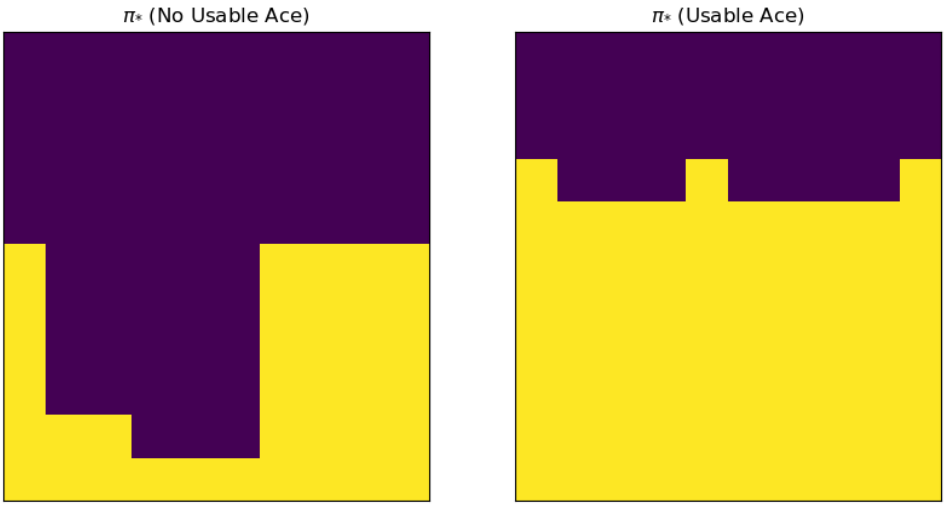

Figure 4: Q3(a) : Optimal policy obtained for blackjack by Monte Carlo Exploration starts

Response:

(a) & (b) Plots for part a and b are shown in figures 5, 6, 7 and 8. To run the code please run Q4.py.

(c) The $\epsilon = 0$ demonstrates the importance of doing exploring starts in Monte-Carlo ES. In all the above plots we can see that without exploration i.e. $\epsilon = 0$, the average returns do not improve. Thus the agent does not learn if it follows greedy policy blindly, however the curves with $\epsilon = 0.1$ and $\epsilon = 0.01$ are constatntly improving average returns and thus learning the optimal policy. This shows the importance of the exploration in the Monte Carlo exploration starts method.

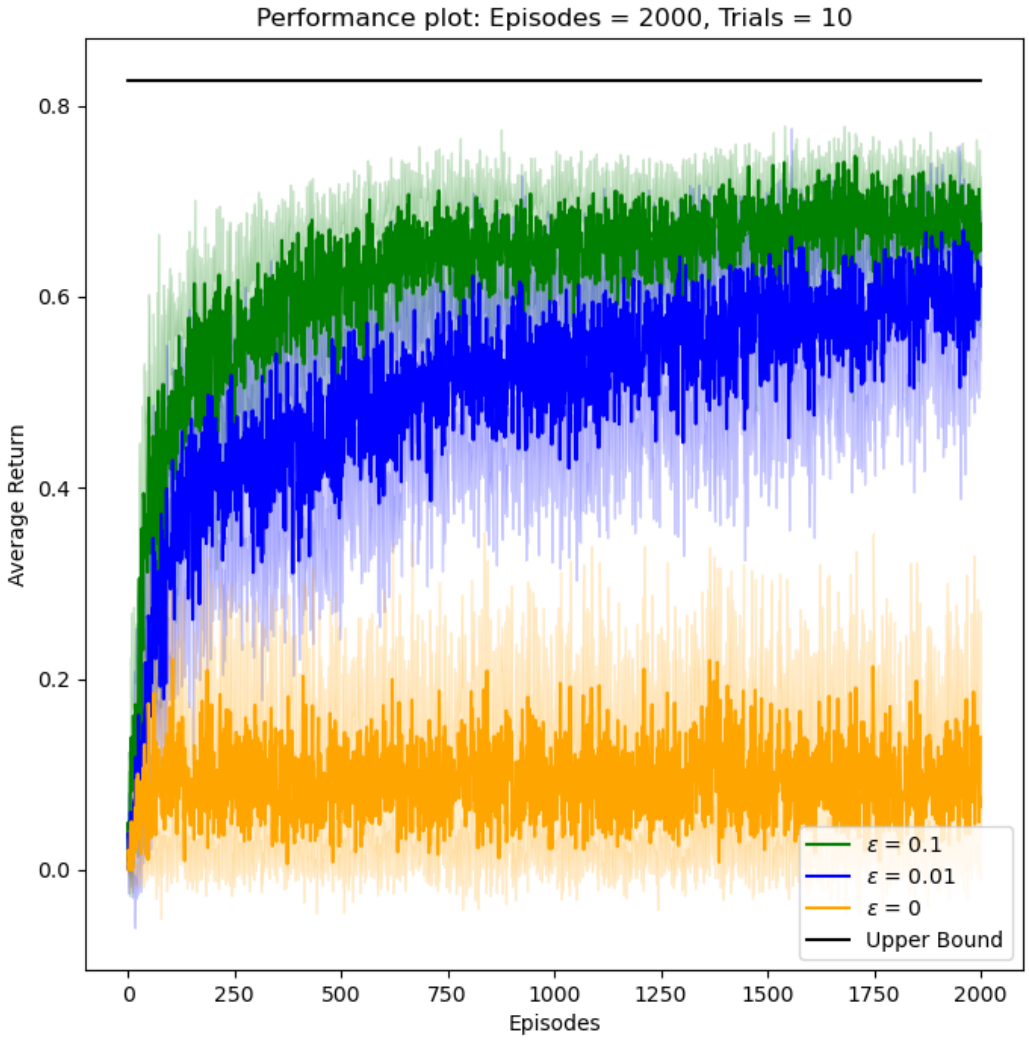

Figure 5: Q4(a) : Average return for different epsilon values in the four rooms environment for goal state (10,10) and 2000 episodes

Question 5. *(a) Derive the weighted-average update rule (Equation 5.8) from (Equation 5.7). Follow the pattern of the derivation of the unweighted rule (Equation 2.3).*
(b) In the boxed algorithm for off-policy MC control, you may have been expecting the W update to have involved the importance-sampling ratio $\frac{\pi(A_t|S_t)}{b(A_t|S_t}$*, but instead it involves* $\frac{1}{b(A_t|S_t)}$*. Why is this correct?*

Response:

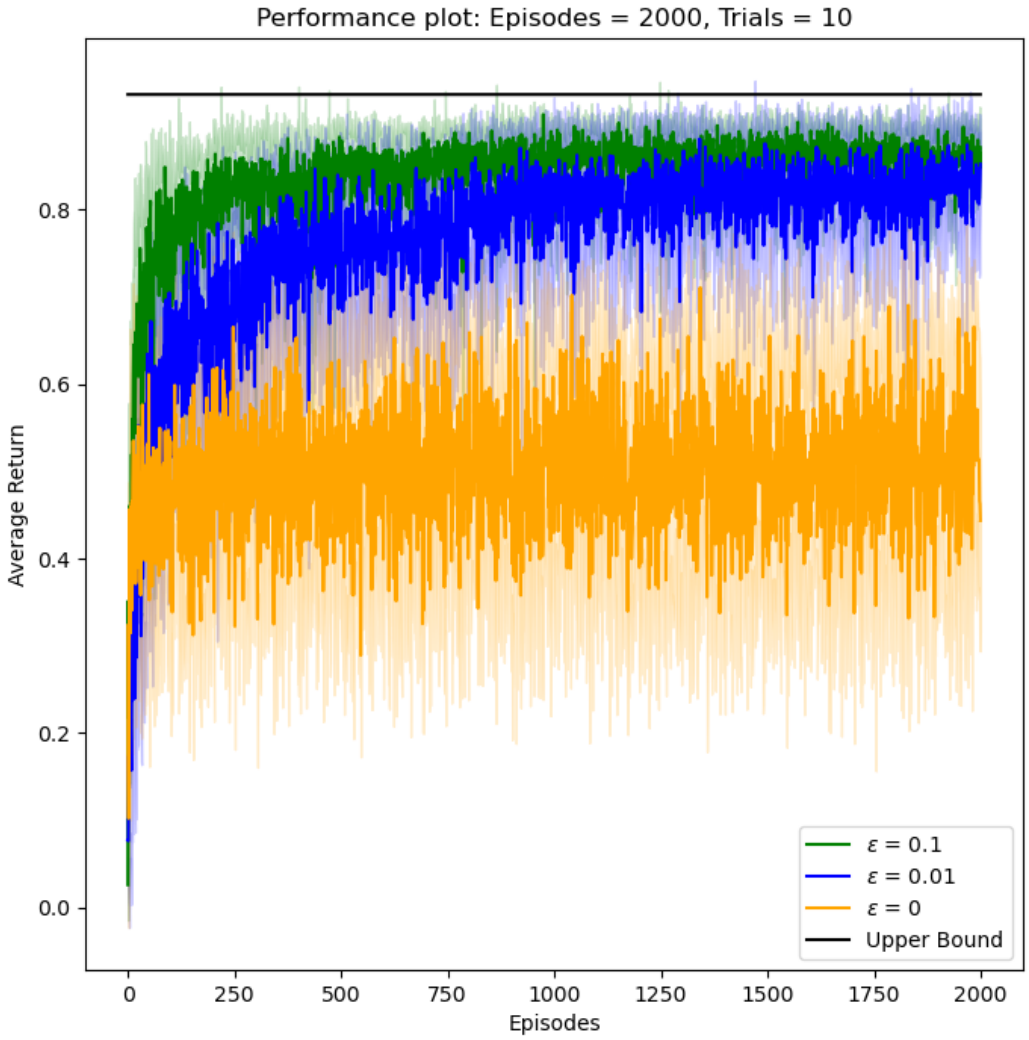

Figure 6: Q4(b) : Average return for different epsilon values in the four rooms environment for goal state (6,0) and 2000 episodes

(a) We know from equation 5.7 that,

$$V_{n+1} = \frac{\Sigma_{k=1}^{n} W_k G_k}{\Sigma_{k=1}^{n} W_k} \tag{9}$$

$$= \frac{W_n G_n + \Sigma_{k=1}^{n-1} W_k G_k}{\Sigma_{k=1}^{n} W_k} \tag{10}$$

$$= \frac{W_n G_n + \Sigma_{k=1}^{n-1} W_k G_k}{\Sigma_{k=1}^{n} W_k} \times \frac{\Sigma_{k=1}^{n-1} W_k}{\Sigma_{k=1}^{n-1} W_k} \tag{11}$$

$$\overline{\gamma} \frac{W_n G_n + \Sigma_{k=1}^{n-1} W_k G_k}{\Sigma_{k=1}^{n-1} W_k} \times \frac{\Sigma_{k=1}^{n-1} W_k}{\Sigma_{k=1}^{n} W_k} \tag{12}$$

$$= \left(\frac{W_n G_n}{\Sigma_{k=1}^{n-1} W_k} + V_n \right) \times \frac{\Sigma_{k=1}^{n-1} W_k}{\Sigma_{k=1}^{n} W_k} \tag{13}$$

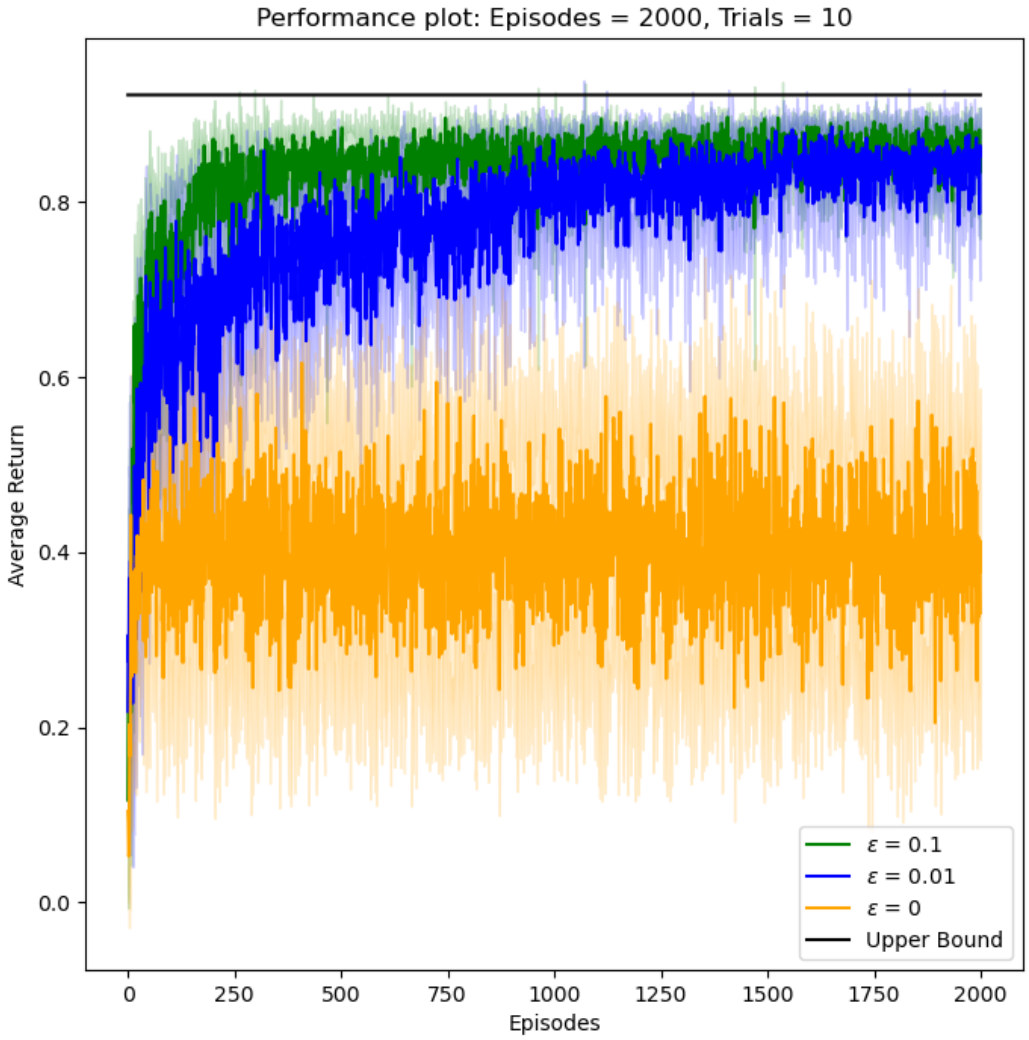

Figure 7: Q4(c) Average return for different epsilon values in the four rooms environment for goal state (7,2) and 2000 episodes

(b)

In the boxed algorithm, if $A_t \neq \pi(S_t)$, then we exit inner loop and next episode starts. But to reach the weight update case, A_t has to be equal to the $\pi(S_t)$. Therefore, for the deterministic π, $\pi(A_t|S_t) = 1$ and hence

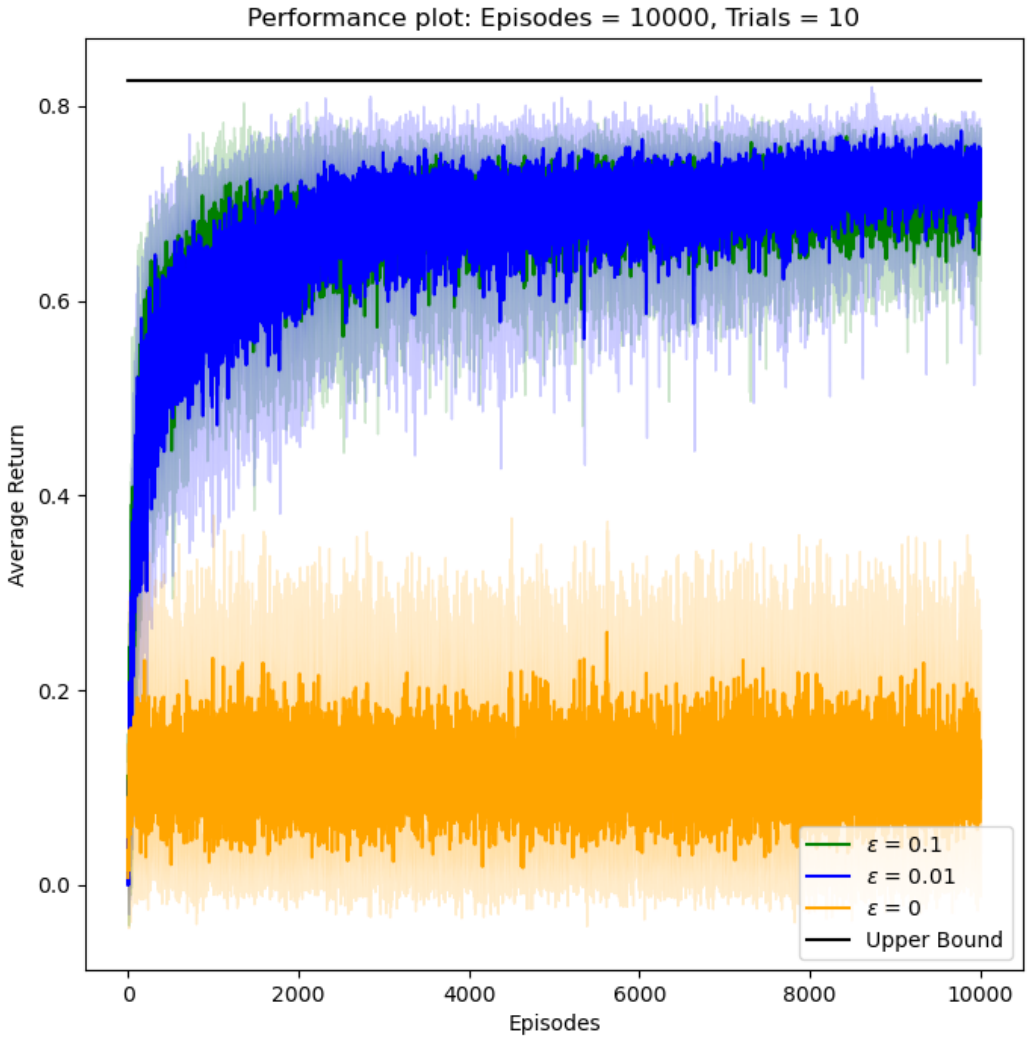

Figure 8: Q4(d) : Average return for different epsilon values in the four rooms environment for goal state (10, 10) and 10000 episodes

$$\frac{\pi(A_t|S_t)}{b(A_t|S_t)} = \frac{1}{b(A_t|S_t)} \tag{24}$$

Therefore, this relation is correct.

Question 6.

Response:

(a) The probabilities of the actions taken under the behavior policy are stored and can be found in the Q6.py main function. (b) Yes the policy makes sense as it shows best action to reach the top right corner target state (10,10). We can see the figure in 9 and it is avoiding all the blocked states. We can see the q values of the greedy target policy in the figure 10

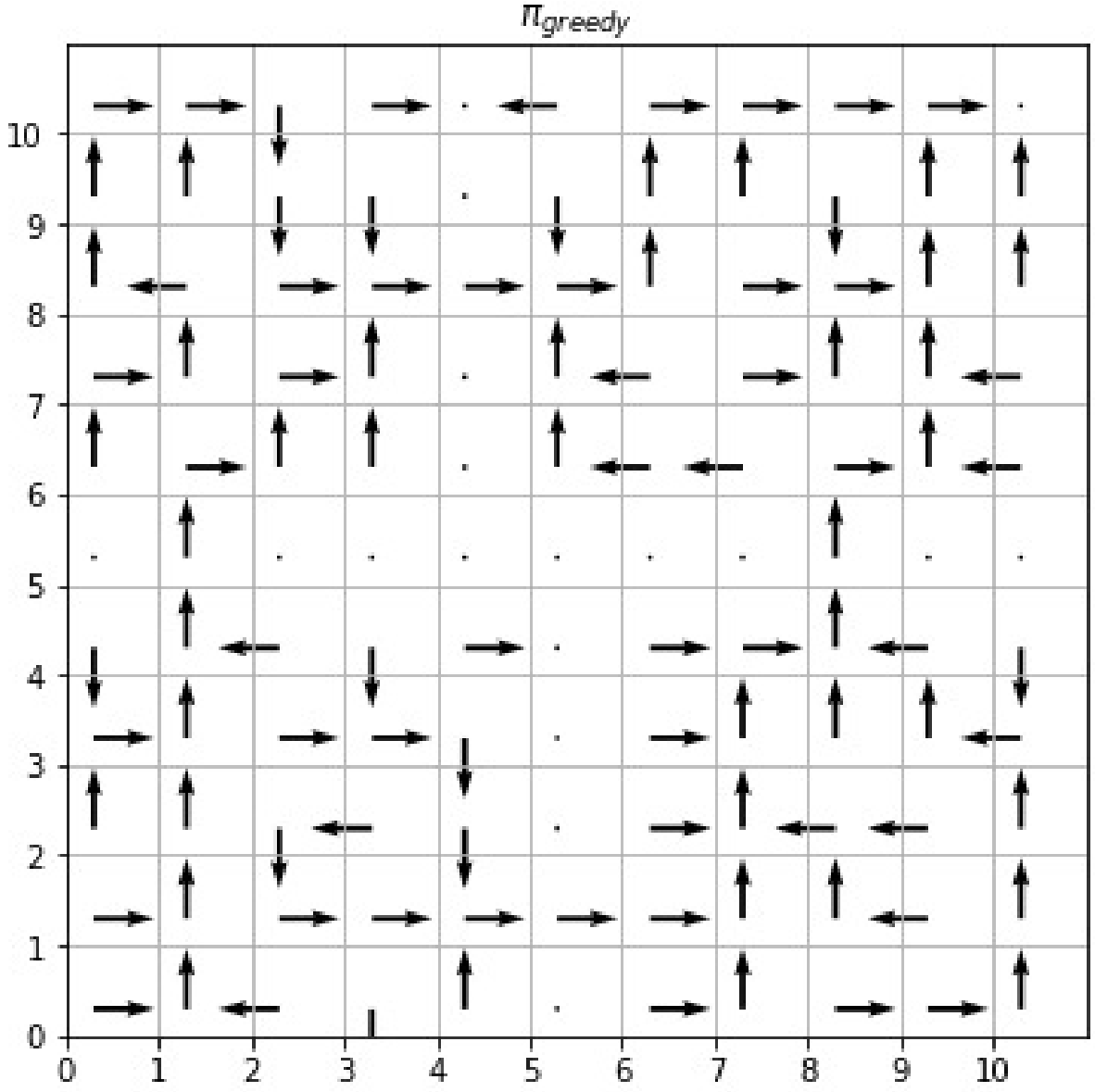

Figure 9: Q6(a) : greedy policy obtained with respect to estimated Q-values.

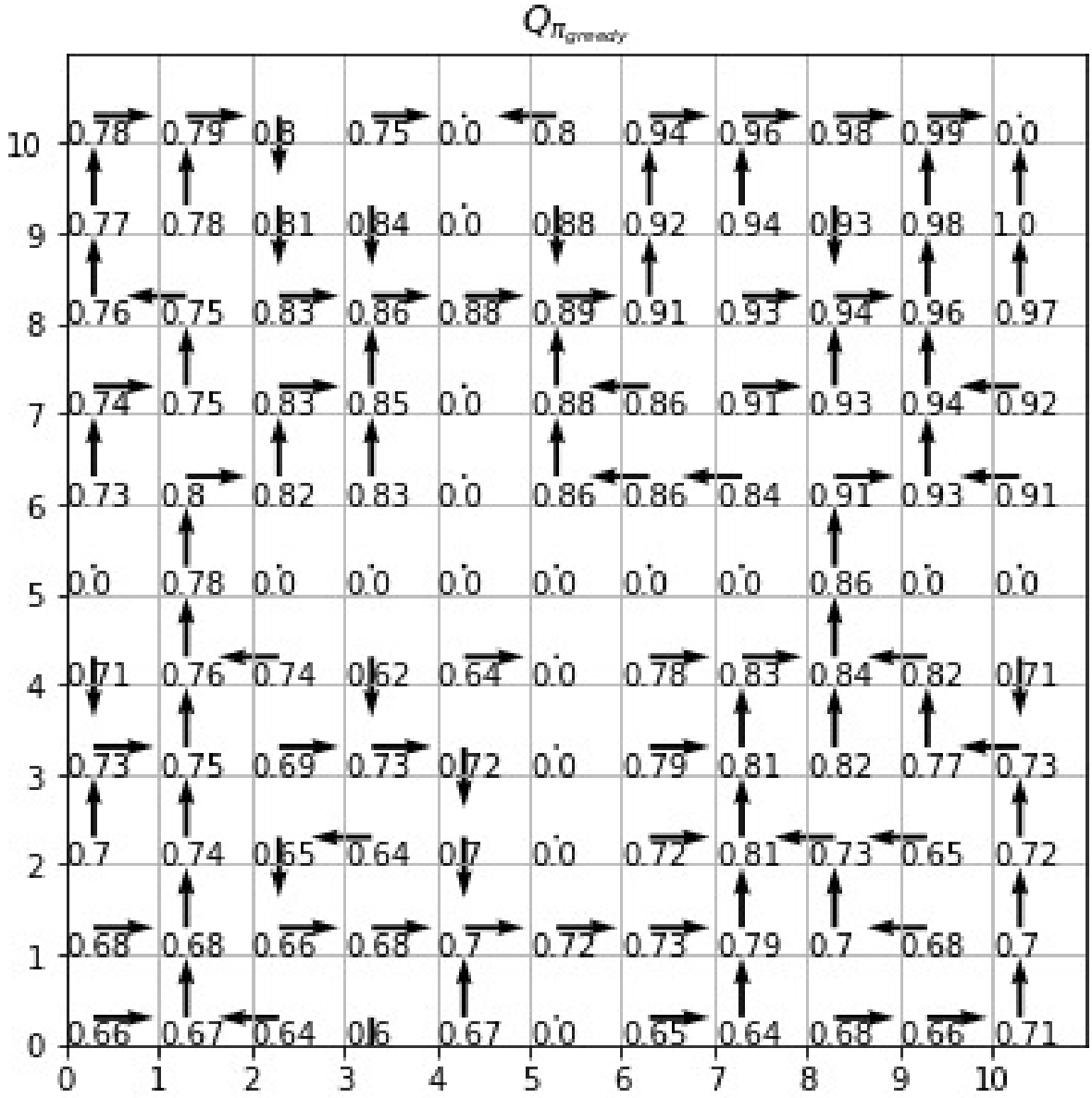

Figure 10: Q6(b) : Q values of the target greedy policy obtained using the ϵ soft policy

(c) The plot is shown in figure 11 (d) The plot is shown in figure 12 (e) From the Off Policy and the On Policy plots shown in figure 11 and figure 12, we can see that the Off Policy outperforms on policy. In Off policy MC prediction we are using ϵ greedy and that's why it explores more and thus outperforms the On Policy. The policy learned for the Off-policy monte carlo prediction is the greedy policy w.r.t Q-Value and it is not used during policy evaluation. For On-Policy monte carlo prediction, the policy used to evaluate Q-Values is same as the actual policy learned which is the greedy policy. This will not be optimal due to lack of exploration.

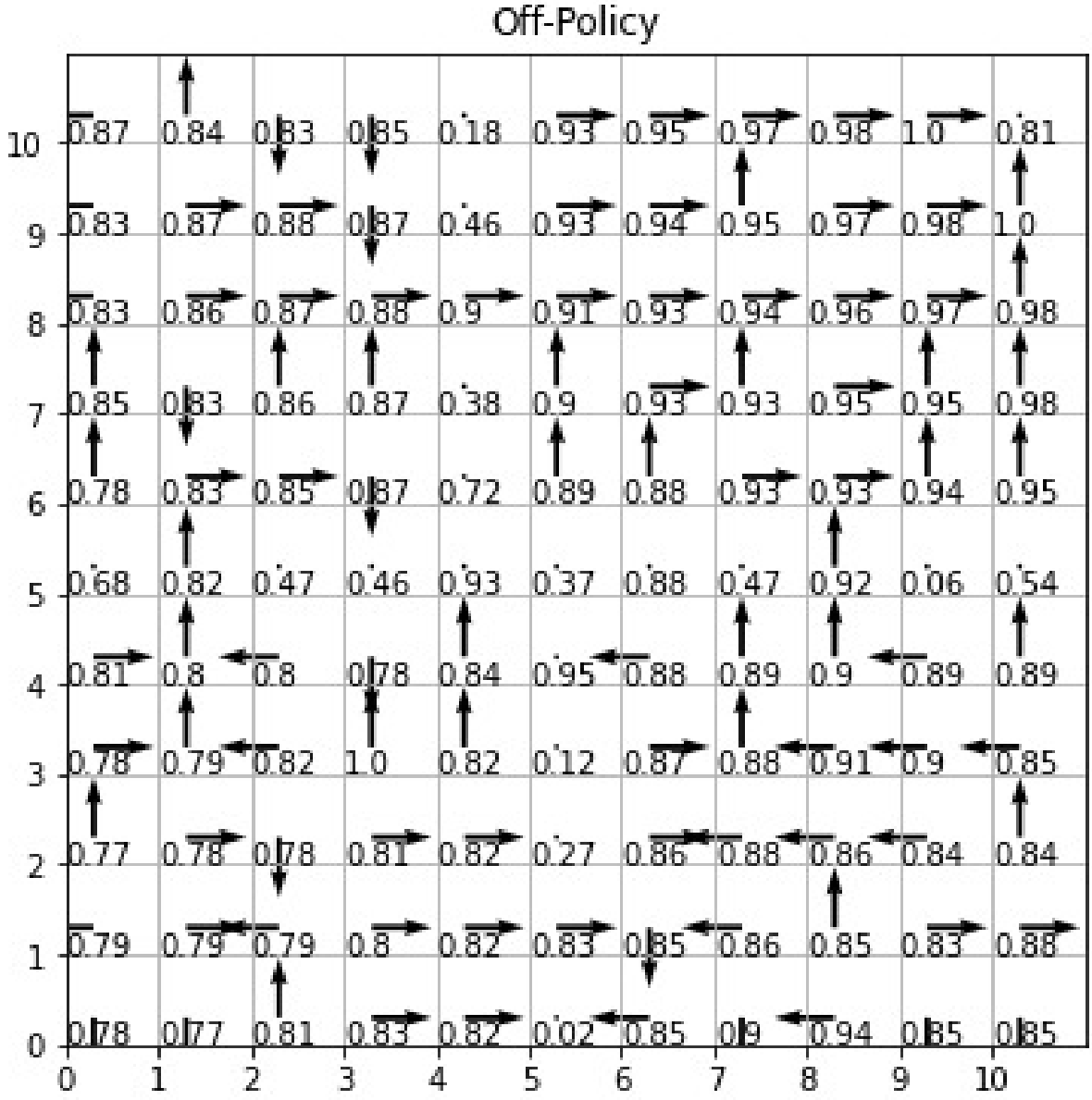

Figure 11: Q6(c) : Q values of the target greedy policy using Off Policy Monte Carlo Prediction

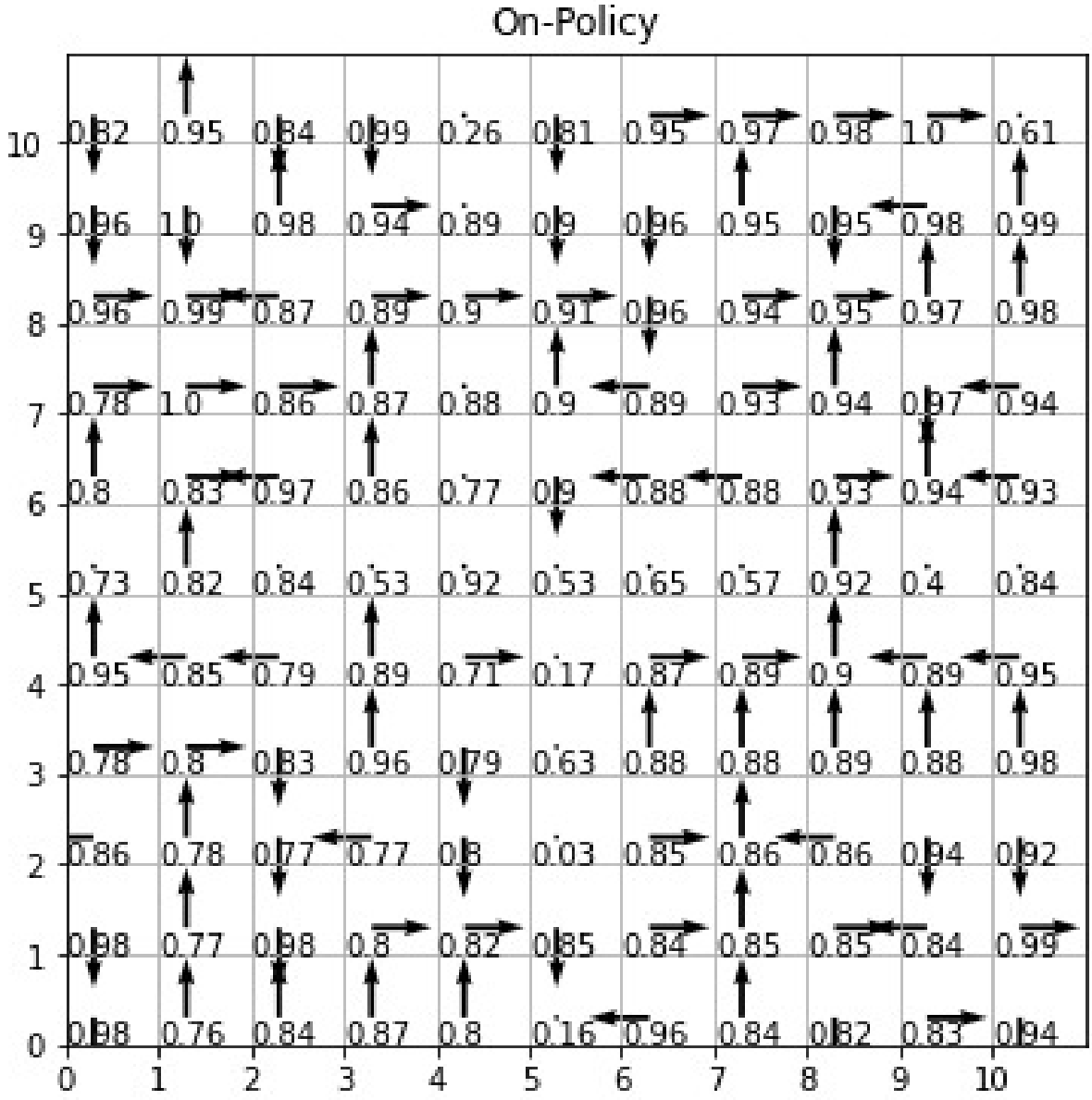

Figure 12: Q6(d) : Q values of the target greedy policy using On Policy Monte Carlo Prediction