Please refer readme.txt before running the code.

Answer for Q1:

1. Initialize

$V(s) \in 0$, for all $s \in S$

$N(s) \in 0$, for all $s \in S$

Loop forever :

Generate episode following π: $S_0\ A_0\ R_1\ S_1\ A_1 \cdots R_T$

$G \leftarrow 0$

Loop $t = T-1, T-2 \cdots 0$:

$G \leftarrow \gamma \cdot G + R_{t+1}$

Unless S_t appears in $S_0, S_1 \cdots S_{t-1}$:

$N(S_t) = N(S_t) + 1$

$V(S_t) = V(S_t) + 1/N(S_t)\,(G - V(S_t))$

1. b. Initialize : $Q(s,a) = 0$

$N(s,a) = 0$, for all $s \in S$; $a \in A(s)$

$\pi(s) \in A(s)$ arbitrarily

Loop forever :

Choose $S_0 \in S$, $A_0 \in A(S_0)$ randomly

Generate an episode from S_0, A_0, ~~following~~ following π

$S_0, A_0, R_1 \cdots S_{T-1}, A_{T-1}, R_T$.

$G \leftarrow 0$

Loop for each step, $t = T-1, T-2 \cdots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

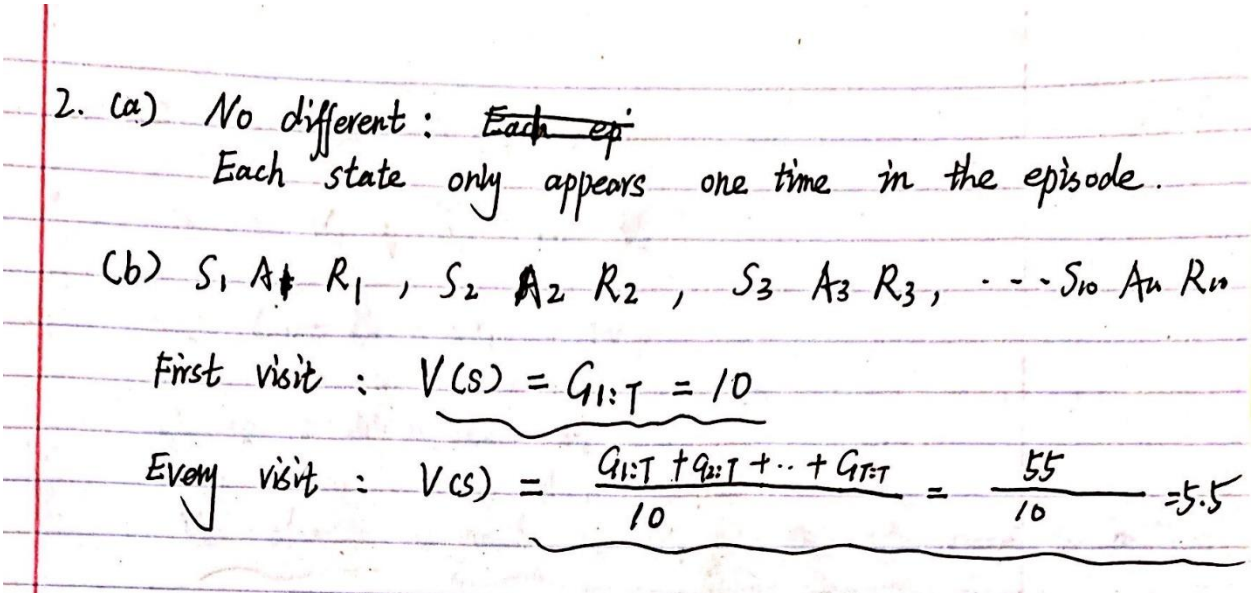
Unless the pair S_t, A_t appears in $S_0, A_0 \cdots A_{t-1}$:

$N(S_t, a_t) \leftarrow N(S_t, a_t) + 1$

$Q(S_t, a_t) = Q(S_t, a_t) + 1/N(S_t, a_t) \cdot (G - Q(S_t, a_t))$

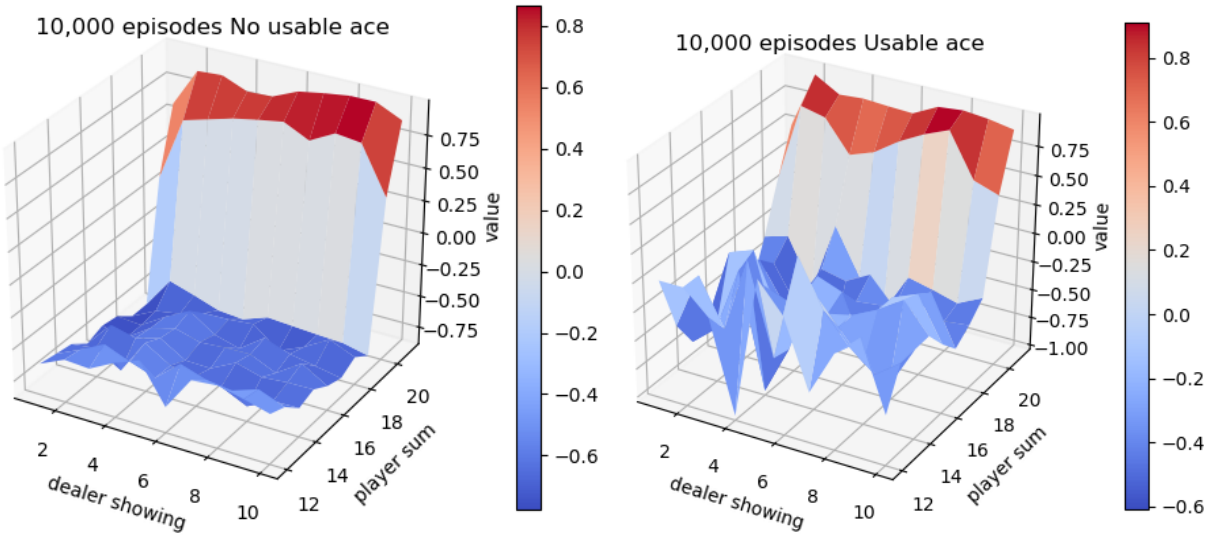
$\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$

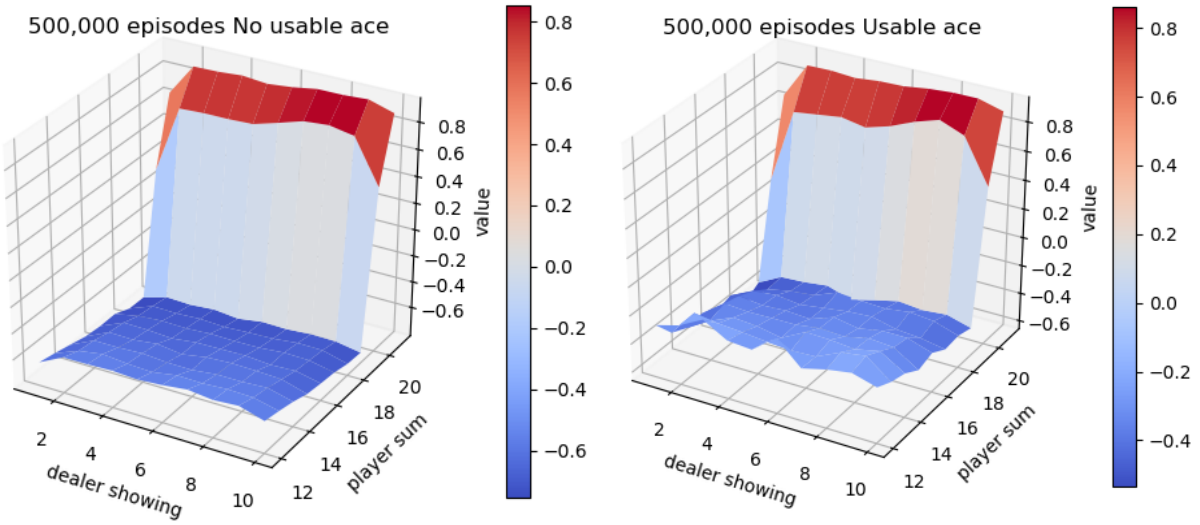
Answer for Q2:



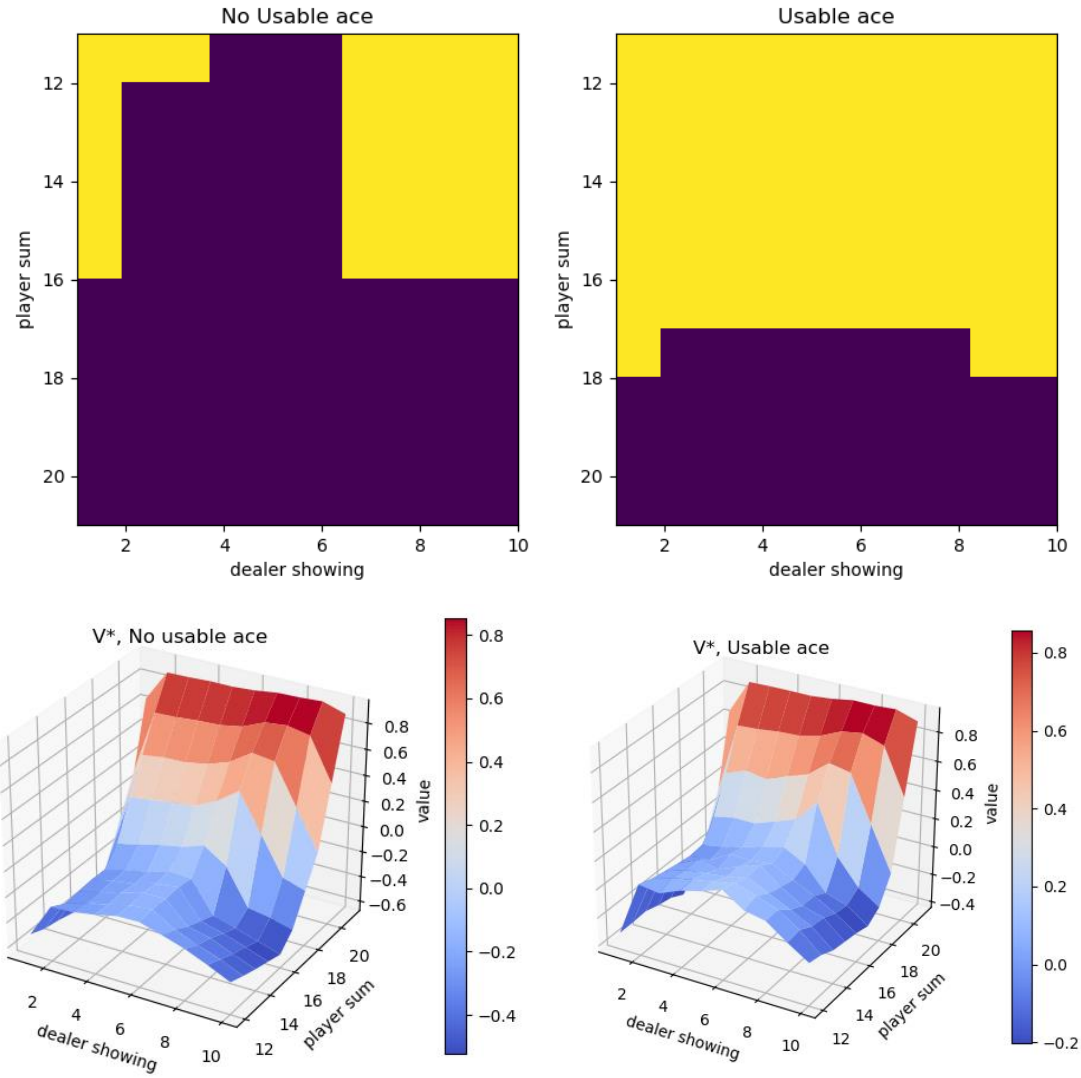
Answer for Q3:

a) run python q3_a.py to generate the data (the default episode number is 10,000)
run python q3_a_plot.py to plot the figures below.



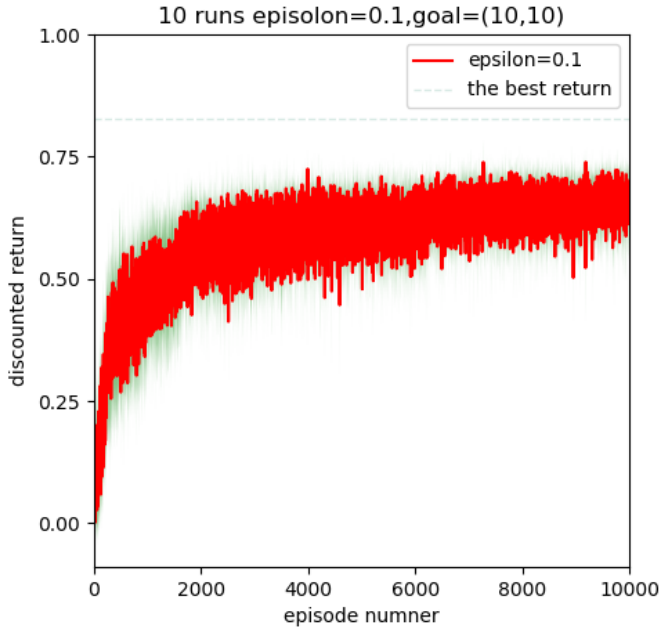


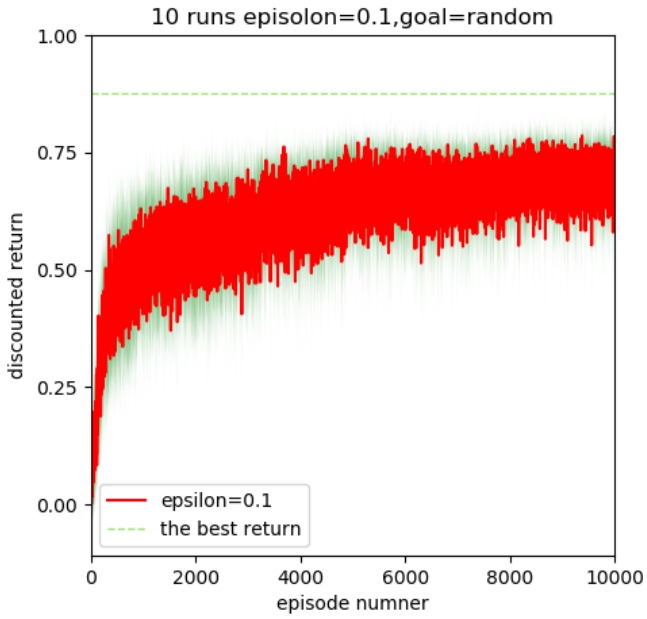
b) python q3_b.py to generate the data; python q3_b_plot is used to plot the figure below



Answer for Q4:

a) python q4_a.py is used to generate the data, python q4_b is used to plot the figure below:



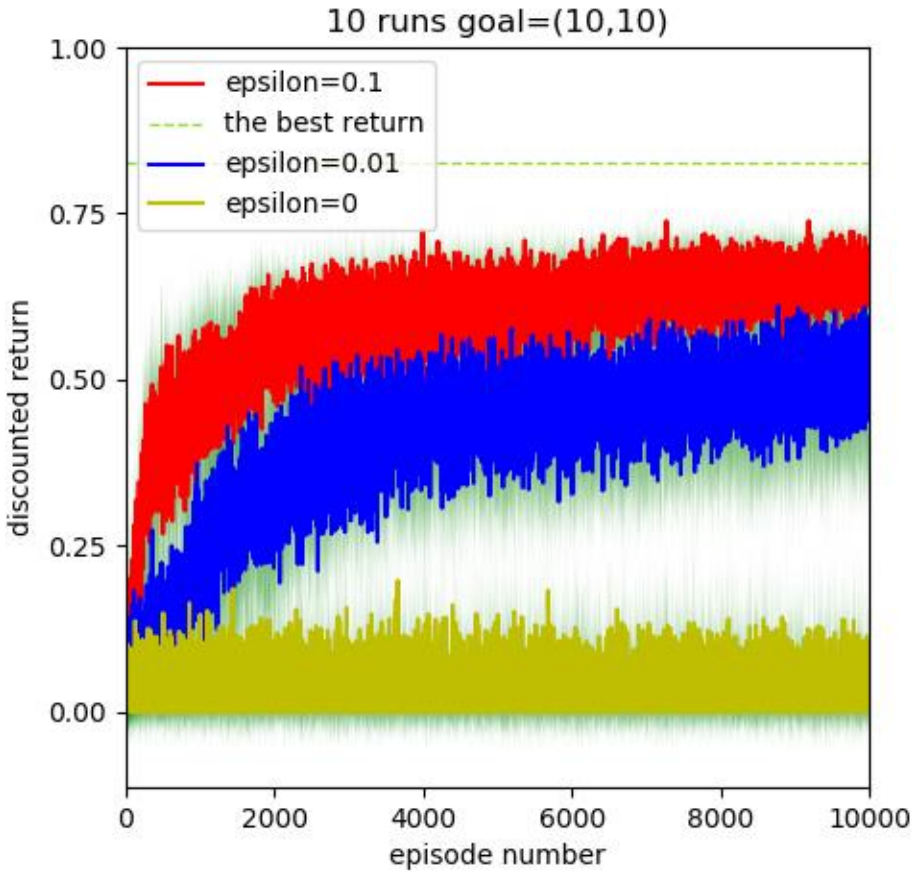


Details: the figures above are ten runs with episode number of 10,000, one for goal position (10,10), one for random goal position. The initial policy is random. The green fade area are the confidence bands.

- With first-visit MC, the agent learns to reach the goal.

b)

q4_a.py with the specific parameters is used to generate the data; q4_b_plot.py is used to plot the figure below:

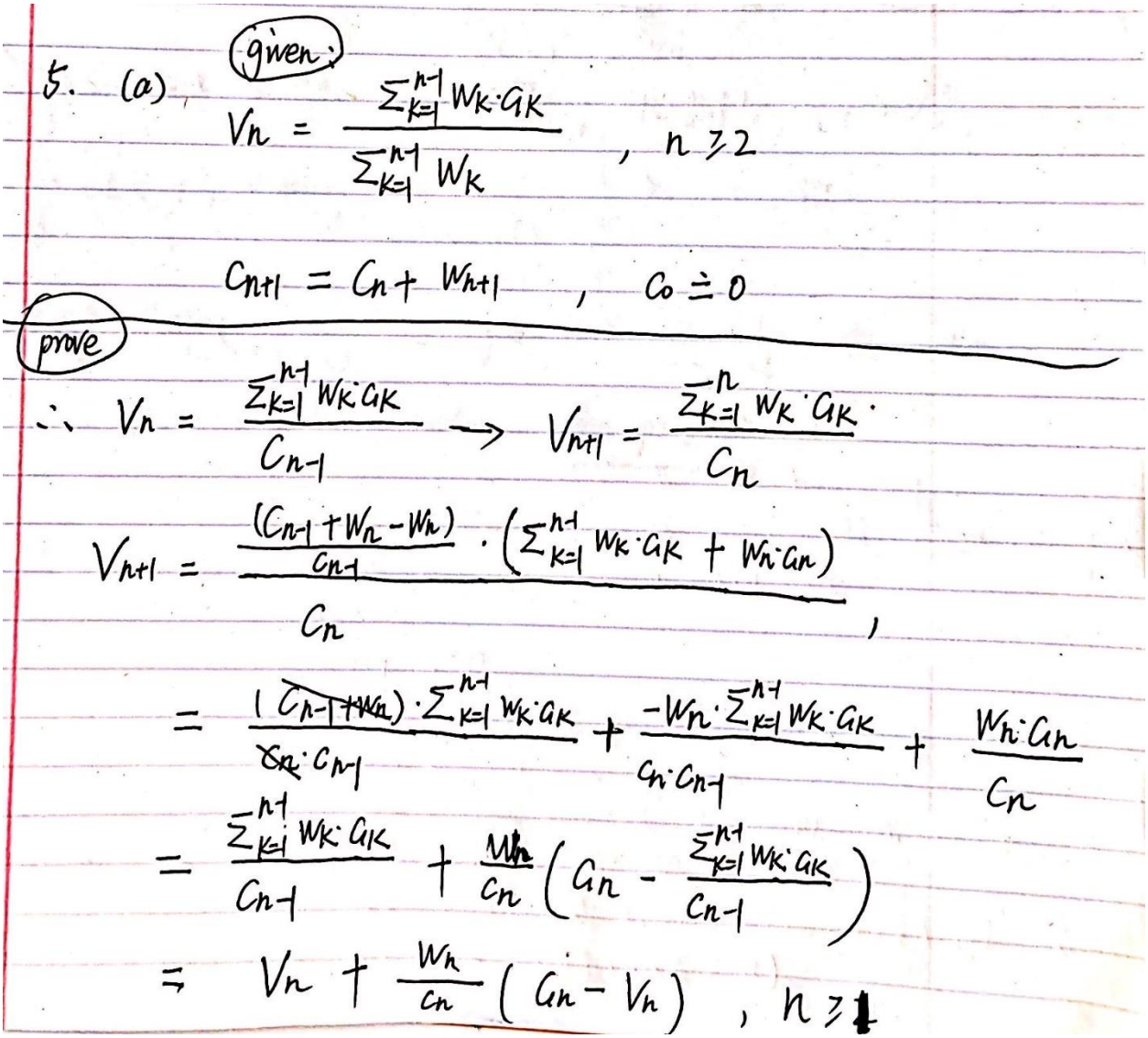


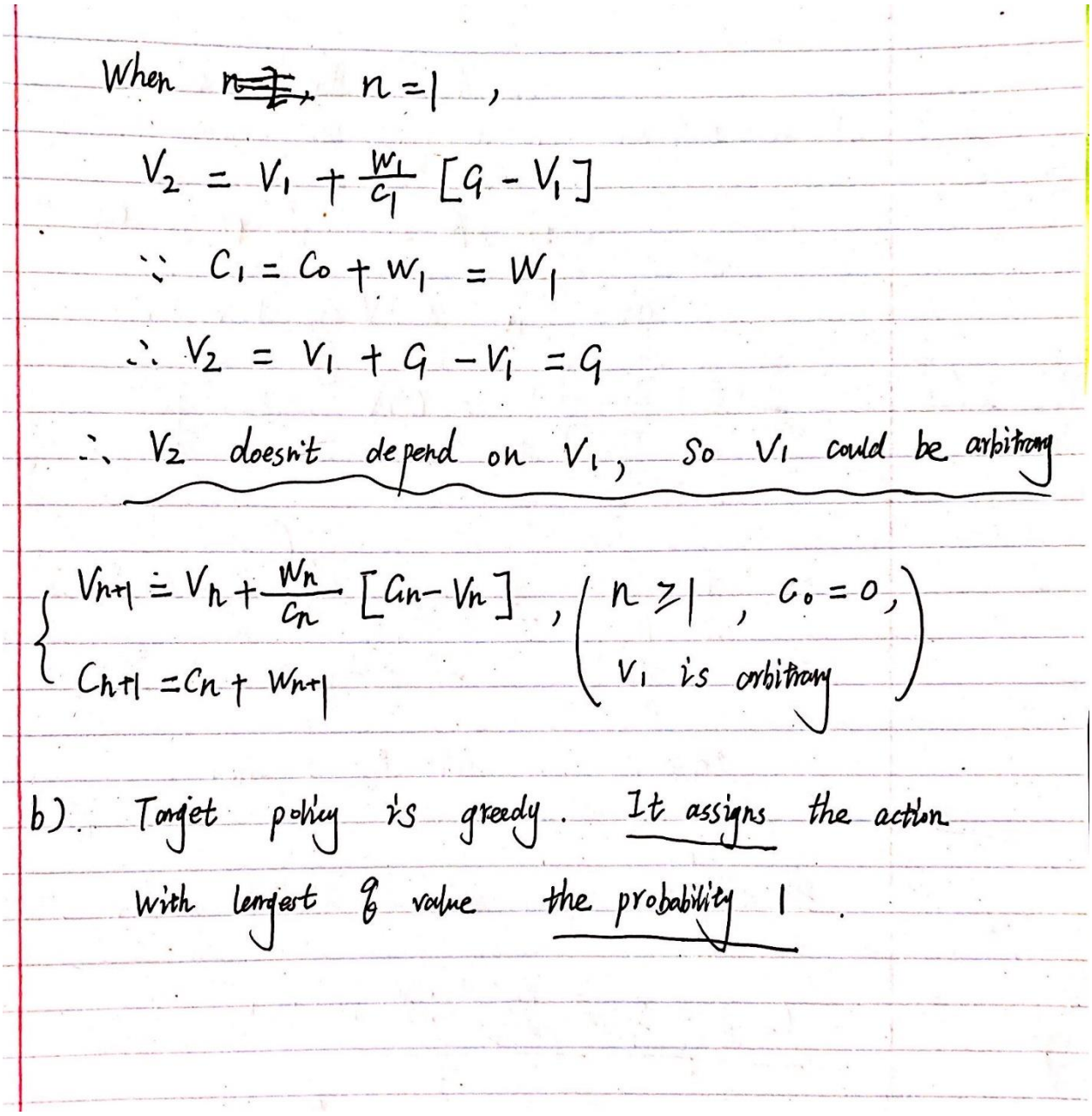
The confidence bands are in shade green colors

c)

when epsilon=0, there is no explore and the agent learns nothing. But in MCES, the random start state and start action is used to explore and substitute the epsilon greedy method.

Answer for Q5:





Answer for Q6:

a)

q6_a.py is used to generate the data and saved the trajectory with probability of action taken in the file named 'episode.p'. It also saves the final policy and the q values.

b)

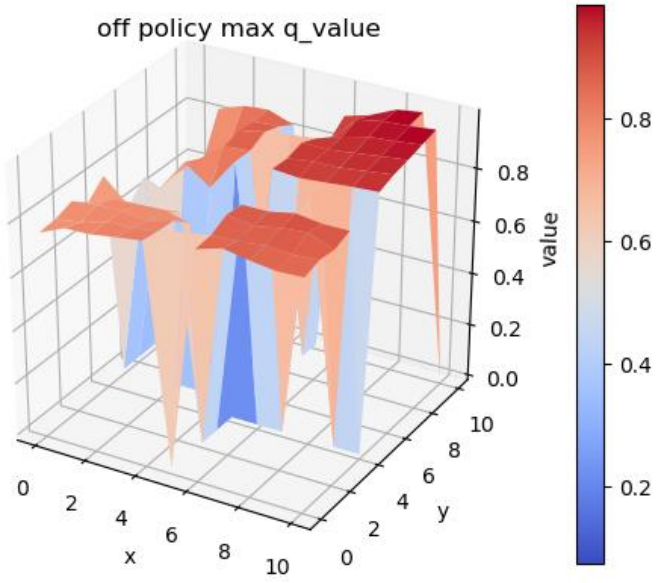
q6_b.py is used to compute the greedy policy respect to the q value stored in part (a). The greedy policy is shown below:

down	right	left	left	right	wall	right	up	right	right	right
right	down	up	down	right	wall	left	down	right	right	up
left	right	up	right	right	right	down	right	right	right	up
left	right	down	up	right	wall	right	right	right	up	up
right	right	down	right	right	wall	up	up	right	up	up
wall	left	wall	wall	wall	wall	right	right	up	up	left
right	left	up	down	up	wall	wall	wall	up	wall	wall
right	right	down	right	down	wall	right	right	up	left	left
up	up	right	down	right	wall	right	right	up	up	up
right	right	down	right	right	right	right	up	up	up	up
up	left	right	right	up	wall	up	right	up	up	up

- The greedy policy is better than the policy generated in the part (a) but it has no exploration and it doesn't converge to the optimal policy.
- The policy is suboptimal or gives some bad actions in some cells like (4,10), (3,10)

c)

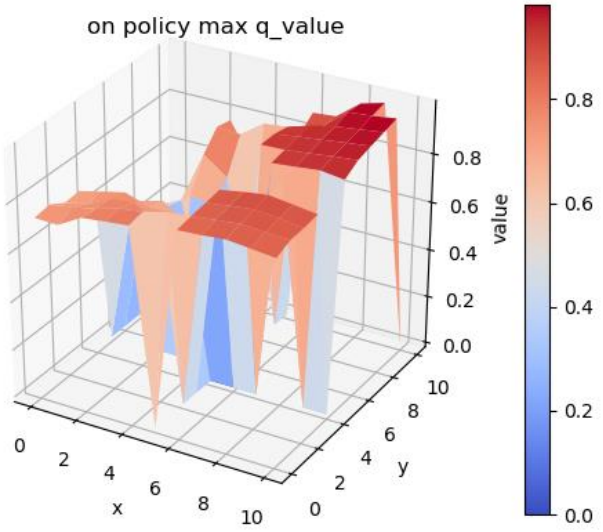
q6_c.py is use to do the off-policy prediction, instead of plot the q values for each action pair. I plot the max q value for each state as below:



Where the initial position is in (0,0) and the goal position is in (10,10). The value of the wall is set to zero.

d)

the code q6_d.py is used to calculate the q value for the greedy policy and q6_d_plot.py is used to plot the max q value for each cell.



e)

- Off-policy prediction value seems coarser (some area lower, some area higher) than the on-policy prediction due to the variance
- Off-policy method seems to converge to the on-policy prediction
- Due to the explore of the behavior policy, the target policy could learn the value of uncommon states. (the top left room)