ECE 232E Project 3

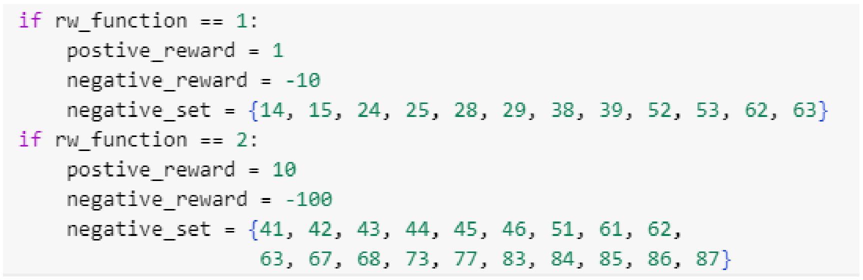
Reinforcement learning and Inverse Reinforcement learning

Chenchen Kuai	206074833
Hao Wang	405629183
Yuning Yang	705930008

1. Reinforcement learning (RL)

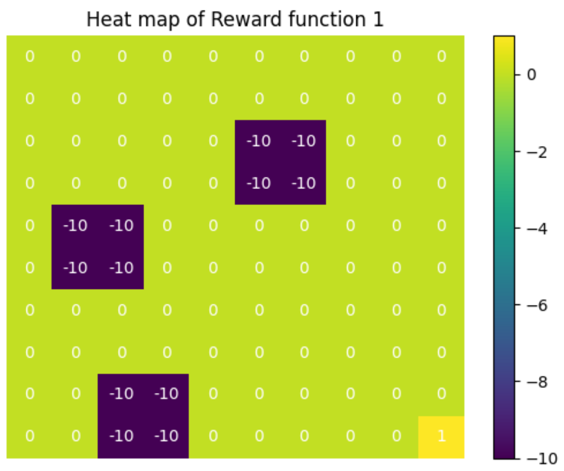
Question 1: (10 points) For visualization purpose, generate heat maps of Reward function 1 and Reward function 2. For the heat maps, make sure you display the coloring scale. You will have 2 plots for this question

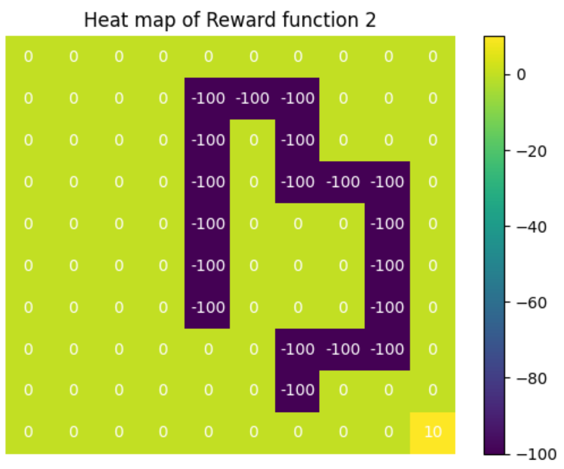
Answer: For this question, we added a parameter to the reward function in the given MDP class to pass different reward functions. The positive reward always appears at state 99 and we use a negative set to store the location of the negative rewards.



```
if rw_function == 1:
    postive_reward = 1
    negative_reward = -10
    negative_set = {14, 15, 24, 25, 28, 29, 38, 39, 52, 53, 62, 63}
if rw_function == 2:
    postive_reward = 10
    negative_reward = -100
    negative_set = {41, 42, 43, 44, 45, 46, 51, 61, 62,
                    63, 67, 68, 73, 77, 83, 84, 85, 86, 87}
```

The plotting function is already in the helper code so the heat map is as follows:





Intuitively, we can know that the whole map only have one state of positive reward, it is like the "destination" and the negative rewards are like the "traps" that need to be avoided. Each state should find itself a way to the bottom right states "destination" and avoid the negative reward "traps". When the agent get to the bottom right states, it should try to stay at that state.

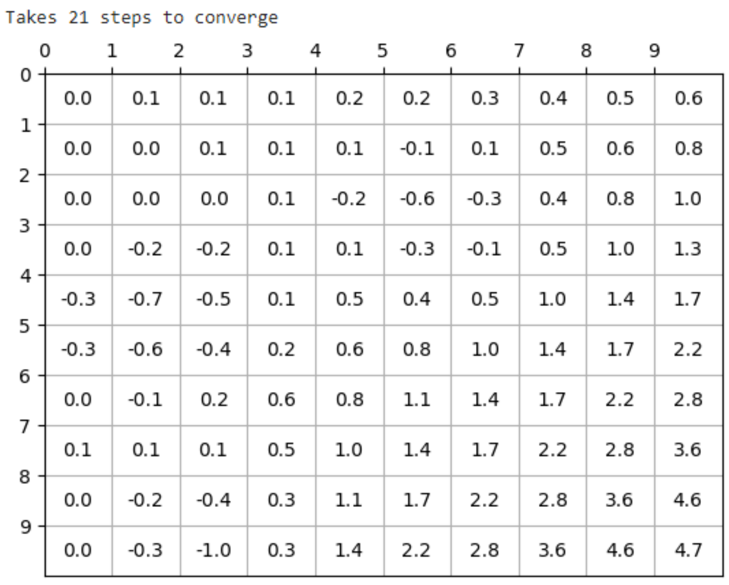
Question 2: (40 points) Create the environment of the agent using the information provided in section 2. To be specific, create the MDP by setting up the state-space, action set, transition probabilities, discount factor, and reward function. For creating the environment, use the following set of parameters:

- *Number of states = 100 (state space is a 10 by 10 square grid as displayed in figure 1)*
- *Number of actions = 4 (set of possible actions is displayed in figure 2)*
- *w = 0.1*
- *Discount factor = 0.8*
- *Reward function 1*

After you have created the environment, then write an optimal state-value function that takes as input the environment of the agent and outputs the optimal value of each state in the grid. For the optimal state-value function, you have to implement the Initialization (lines 2-4) and Estimation (lines 5-13) steps of the Value Iteration algorithm. For the estimation step, use = 0.01. For visualization purpose, you should generate a figure similar to that of figure 1 but with the number of state replaced by the optimal value of that state. In this part of question, you should have 1 plot.

Let's assume that your value iteration algorithm converges in N steps. Plot snapshots of state values in 5 different steps linearly distributed from 1 to N. Report N and your step numbers. What observations do you have from the plots?

Answer: To calculate the transition probabilities, we follow exactly the instructions in the project description and the complete the helper code. We also implement the iteration algorithm as declared in the project description to make the state value converge. At last, the stable state value matrix is as follows:

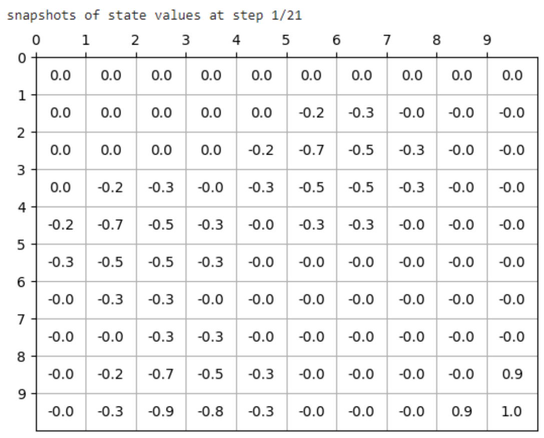


Takes 21 steps to converge

	0	1	2	3	4	5	6	7	8	9
0	0.0	0.1	0.1	0.1	0.2	0.2	0.3	0.4	0.5	0.6
1	0.0	0.0	0.1	0.1	0.1	-0.1	0.1	0.5	0.6	0.8
2	0.0	0.0	0.0	0.1	-0.2	-0.6	-0.3	0.4	0.8	1.0
3	0.0	-0.2	-0.2	0.1	0.1	-0.3	-0.1	0.5	1.0	1.3
4	-0.3	-0.7	-0.5	0.1	0.5	0.4	0.5	1.0	1.4	1.7
5	-0.3	-0.6	-0.4	0.2	0.6	0.8	1.0	1.4	1.7	2.2
6	0.0	-0.1	0.2	0.6	0.8	1.1	1.4	1.7	2.2	2.8
7	0.1	0.1	0.1	0.5	1.0	1.4	1.7	2.2	2.8	3.6
8	0.0	-0.2	-0.4	0.3	1.1	1.7	2.2	2.8	3.6	4.6
9	0.0	-0.3	-1.0	0.3	1.4	2.2	2.8	3.6	4.6	4.7

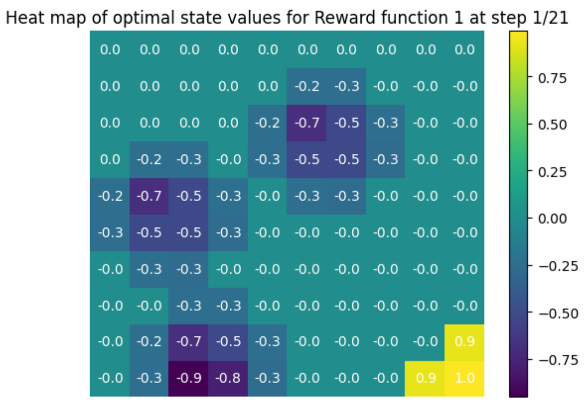
It takes 21 steps to converge so we choose snapshot after the calculation of step 1, 6, 11, 16, 21 to print the value. (snapshot after step 21 is the final result and the step number starts from 1) We also provide the heatmap for those snapshots for better looking.

Step 1/21

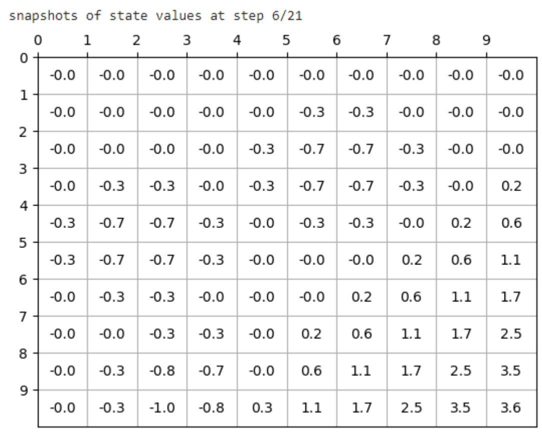


snapshots of state values at step 1/21

	0	1	2	3	4	5	6	7	8	9
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	-0.2	-0.3	-0.0	-0.0	-0.0
2	0.0	0.0	0.0	0.0	-0.2	-0.7	-0.5	-0.3	-0.0	-0.0
3	0.0	-0.2	-0.3	-0.0	-0.3	-0.5	-0.5	-0.3	-0.0	-0.0
4	-0.2	-0.7	-0.5	-0.3	-0.0	-0.3	-0.3	-0.0	-0.0	-0.0
5	-0.3	-0.5	-0.5	-0.3	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
6	-0.0	-0.3	-0.3	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
7	-0.0	-0.0	-0.3	-0.3	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
8	-0.0	-0.2	-0.7	-0.5	-0.3	-0.0	-0.0	-0.0	-0.0	0.9
9	-0.0	-0.3	-0.9	-0.8	-0.3	-0.0	-0.0	-0.0	0.9	1.0

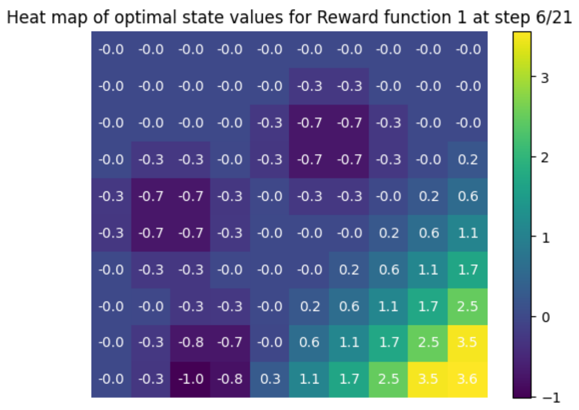


Step 6/21

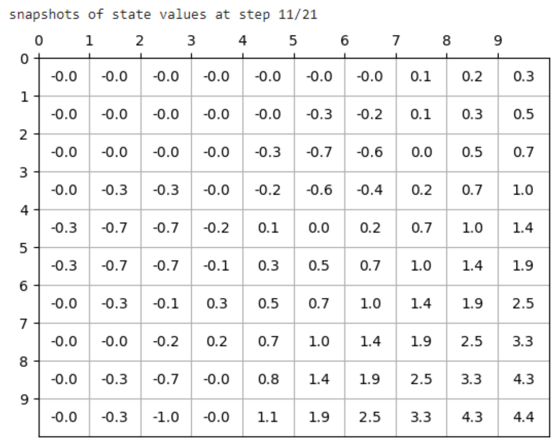


snapshots of state values at step 6/21

	0	1	2	3	4	5	6	7	8	9
0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
1	-0.0	-0.0	-0.0	-0.0	-0.0	-0.3	-0.3	-0.0	-0.0	-0.0
2	-0.0	-0.0	-0.0	-0.0	-0.3	-0.7	-0.7	-0.3	-0.0	-0.0
3	-0.0	-0.3	-0.3	-0.0	-0.3	-0.7	-0.7	-0.3	-0.0	0.2
4	-0.3	-0.7	-0.7	-0.3	-0.0	-0.3	-0.3	-0.0	0.2	0.6
5	-0.3	-0.7	-0.7	-0.3	-0.0	-0.0	-0.0	0.2	0.6	1.1
6	-0.0	-0.3	-0.3	-0.0	-0.0	-0.0	0.2	0.6	1.1	1.7
7	-0.0	-0.0	-0.3	-0.3	-0.0	0.2	0.6	1.1	1.7	2.5
8	-0.0	-0.3	-0.8	-0.7	-0.0	0.6	1.1	1.7	2.5	3.5
9	-0.0	-0.3	-1.0	-0.8	0.3	1.1	1.7	2.5	3.5	3.6

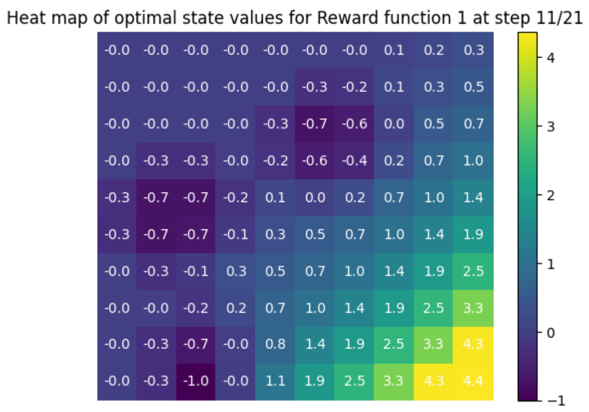


Step 11/21

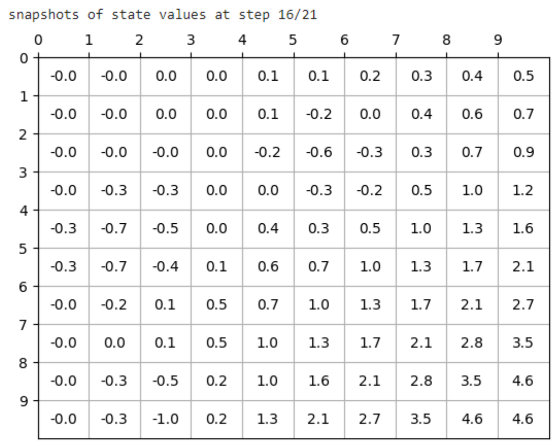


snapshots of state values at step 11/21

	0	1	2	3	4	5	6	7	8	9
0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	0.1	0.2	0.3
1	-0.0	-0.0	-0.0	-0.0	-0.0	-0.3	-0.2	0.1	0.3	0.5
2	-0.0	-0.0	-0.0	-0.0	-0.3	-0.7	-0.6	0.0	0.5	0.7
3	-0.0	-0.3	-0.3	-0.0	-0.2	-0.6	-0.4	0.2	0.7	1.0
4	-0.3	-0.7	-0.7	-0.2	0.1	0.0	0.2	0.7	1.0	1.4
5	-0.3	-0.7	-0.7	-0.1	0.3	0.5	0.7	1.0	1.4	1.9
6	-0.0	-0.3	-0.1	0.3	0.5	0.7	1.0	1.4	1.9	2.5
7	-0.0	-0.0	-0.2	0.2	0.7	1.0	1.4	1.9	2.5	3.3
8	-0.0	-0.3	-0.7	-0.0	0.8	1.4	1.9	2.5	3.3	4.3
9	-0.0	-0.3	-1.0	-0.0	1.1	1.9	2.5	3.3	4.3	4.4

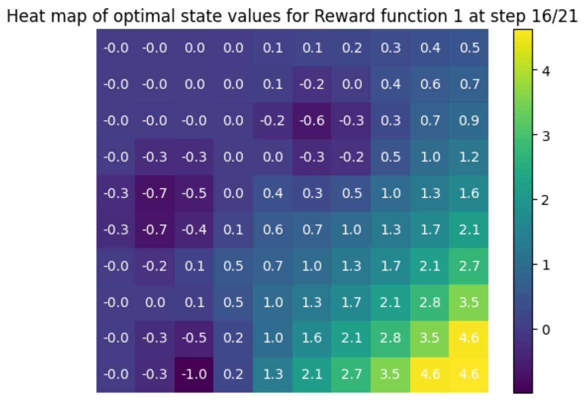


Step 16/21

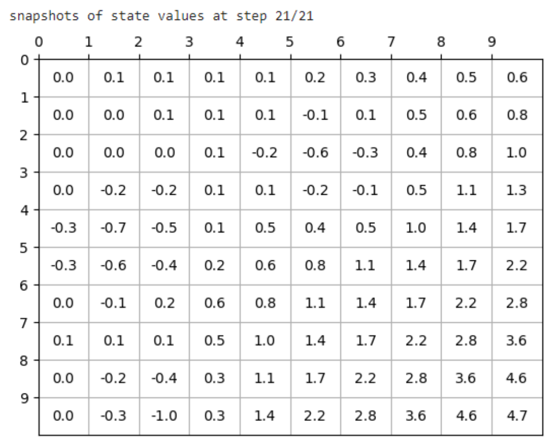


snapshots of state values at step 16/21

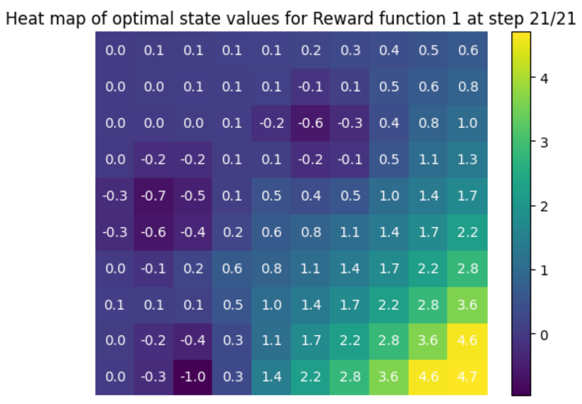
	0	1	2	3	4	5	6	7	8	9
0	-0.0	-0.0	0.0	0.0	0.1	0.1	0.2	0.3	0.4	0.5
1	-0.0	-0.0	0.0	0.0	0.1	-0.2	0.0	0.4	0.6	0.7
2	-0.0	-0.0	-0.0	0.0	-0.2	-0.6	-0.3	0.3	0.7	0.9
3	-0.0	-0.3	-0.3	0.0	0.0	-0.3	-0.2	0.5	1.0	1.2
4	-0.3	-0.7	-0.5	0.0	0.4	0.3	0.5	1.0	1.3	1.6
5	-0.3	-0.7	-0.4	0.1	0.6	0.7	1.0	1.3	1.7	2.1
6	-0.0	-0.2	0.1	0.5	0.7	1.0	1.3	1.7	2.1	2.7
7	-0.0	0.0	0.1	0.5	1.0	1.3	1.7	2.1	2.8	3.5
8	-0.0	-0.3	-0.5	0.2	1.0	1.6	2.1	2.8	3.5	4.6
9	-0.0	-0.3	-1.0	0.2	1.3	2.1	2.7	3.5	4.6	4.6



Step 21/21


snapshots of state values at step 21/21

	0	1	2	3	4	5	6	7	8	9
0	0.0	0.1	0.1	0.1	0.1	0.2	0.3	0.4	0.5	0.6
1	0.0	0.0	0.1	0.1	0.1	-0.1	0.1	0.5	0.6	0.8
2	0.0	0.0	0.0	0.1	-0.2	-0.6	-0.3	0.4	0.8	1.0
3	0.0	-0.2	-0.2	0.1	0.1	-0.2	-0.1	0.5	1.1	1.3
4	-0.3	-0.7	-0.5	0.1	0.5	0.4	0.5	1.0	1.4	1.7
5	-0.3	-0.6	-0.4	0.2	0.6	0.8	1.1	1.4	1.7	2.2
6	0.0	-0.1	0.2	0.6	0.8	1.1	1.4	1.7	2.2	2.8
7	0.1	0.1	0.1	0.5	1.0	1.4	1.7	2.2	2.8	3.6
8	0.0	-0.2	-0.4	0.3	1.1	1.7	2.2	2.8	3.6	4.6
9	0.0	-0.3	-1.0	0.3	1.4	2.2	2.8	3.6	4.6	4.7

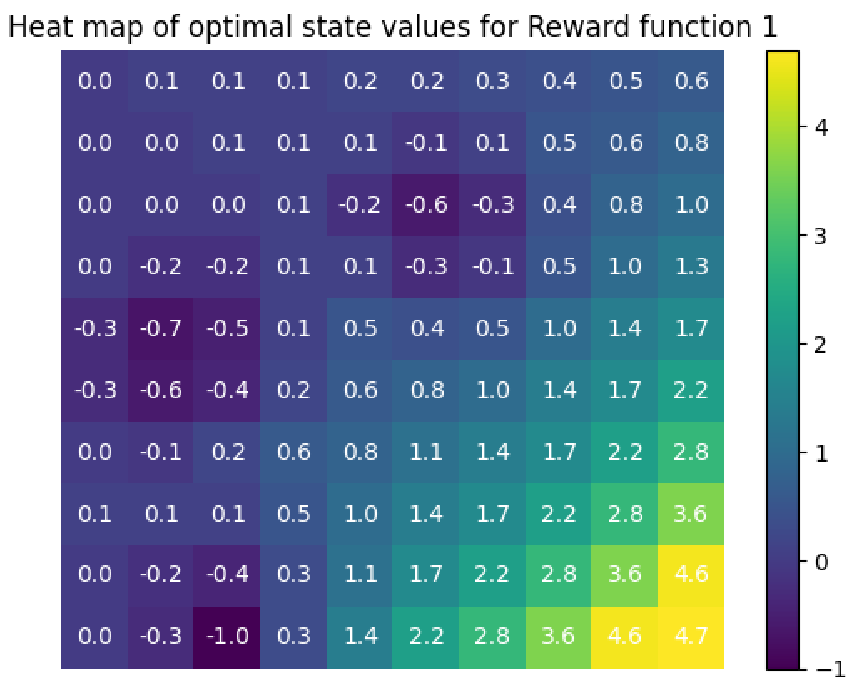


We can have the following observations.

1. From the beginning, there are many zeros, because there are many state that are far from traps and destination so their neighbours are zero. In the beginning these states have no sense of a very far trap or destination but only a sense of their neighbour, so their value is 0. But after steps of iteration, those zero states can sense farther information so that their values are changed.
2. The whole map also becomes more and more "smooth" because the rewards influence farther and farther so that the value variation becomes smooth between neighbors.
3. The state value near the "traps" is always lower than that near the "destination". This is obvious because the traps have negative reward and destination have a positive reward.
4. The state value at the destination is growing. The value in step 1 is only 1 but finally it is 4.7. That is because if the agent stops at the destination point, it can still take a next step to go right or down, in this case, it probably stays at the destination point and the value got increased. If the wind is 0, the final state value should be 1/(1-Discount factor)

Question 3: (5 points) Generate a heat map of the optimal state values across the 2-D grid. For generating the heat map, you can use the same function provided in the hint earlier (see the hint after question 1).

Answer: The map is as follows, same as the map we gave in Q2.



0.0	0.1	0.1	0.1	0.2	0.2	0.3	0.4	0.5	0.6
0.0	0.0	0.1	0.1	0.1	-0.1	0.1	0.5	0.6	0.8
0.0	0.0	0.0	0.1	-0.2	-0.6	-0.3	0.4	0.8	1.0
0.0	-0.2	-0.2	0.1	0.1	-0.3	-0.1	0.5	1.0	1.3
-0.3	-0.7	-0.5	0.1	0.5	0.4	0.5	1.0	1.4	1.7
-0.3	-0.6	-0.4	0.2	0.6	0.8	1.0	1.4	1.7	2.2
0.0	-0.1	0.2	0.6	0.8	1.1	1.4	1.7	2.2	2.8
0.1	0.1	0.1	0.5	1.0	1.4	1.7	2.2	2.8	3.6
0.0	-0.2	-0.4	0.3	1.1	1.7	2.2	2.8	3.6	4.6
0.0	-0.3	-1.0	0.3	1.4	2.2	2.8	3.6	4.6	4.7

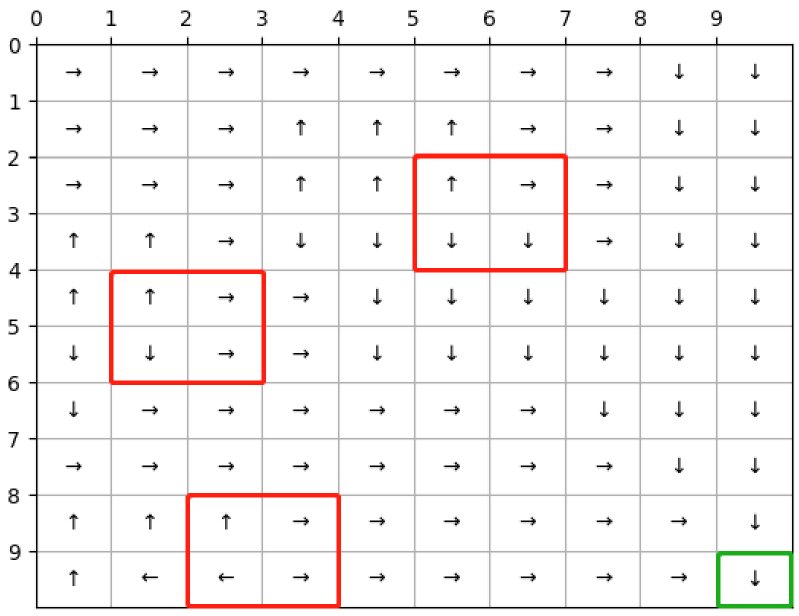
Question 4: (15 points) Explain the distribution of the optimal state values across the 2-D grid. (Hint: Use the figure generated in question 3 to explain)

Answer:

1. We can see that the heat map has a "high score area" that is at the buttom right corner. This is because that near the buttom right corner it is easily to get to the destination point and get a positive reward. So the nearer it is to destination, the higher the value. We can also see that there are three dim areas, which is around the negative rewards area. Because these states has a higher probability to step on the negative reward area ("traps"), although the best policy suggest the agent move away from the traps, there is still wind to blow it to the trap.
2. We can see the value of the destination state is 4.7. It is given in Q2 and if there is no wind, the value of the state 99 (destination state) should be 1/(1-Discount factor). Here gives the analysis: if there is no wind, the best policy must be going right or going down so that the agent has a probability of 1 to remains state 99 and
   v[99] = 1 * (1 + Discount factor * v[99])
   So v[99] = 1/(1-Discount factor)
   In this case, Discount factor is 0.8 so v[99] should be 5 without wind. But with wind of 0.1 which is not huge, the final v[99] is 4.7

Question 5: (20 points) Implement the computation step of the value iteration algorithm (lines 14-17) to compute the optimal policy of the agent navigating the 2-D state-space. For visualization purpose, you should generate a figure similar to that of figure 1 but with the number of state replaced by the optimal action at that state. The optimal actions should be displayed using arrows. Does the optimal policy of the agent match your intuition? Please provide a brief explanation. Is it possible for the agent to compute the optimal action to take at each state by observing the optimal values of it's neighboring states? In this question, you should have 1 plot.

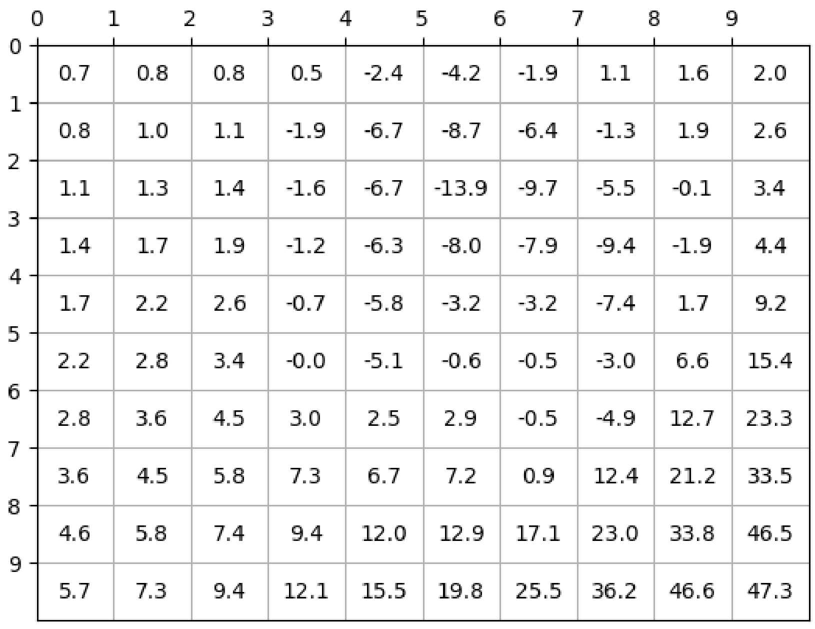
Answer: The following is the policy plot. In the plot, we use red box to indicate the negative rewards and the green box to indicate the positive reward. We can see that in consist with the intuition we said in Q1: *Each state should find itself a way to the bottom right states "destination" and avoid the negative reward "traps"*. We can see that none of the states make a decision to go into the red box, which is directly into traps and all the states find its own way to the button right "destination" state.



	0	1	2	3	4	5	6	7	8	9
0	→	→	→	→	→	→	→	→	↓	↓
1	→	→	→	↑	↑	↑	→	→	↓	↓
2	→	→	→	↑	↑	↑	→	→	↓	↓
3	↑	↑	→	↓	↓	↓	↓	→	↓	↓
4	↑	↑	→	→	↓	↓	↓	↓	↓	↓
5	↓	↓	→	→	↓	↓	↓	↓	↓	↓
6	↓	→	→	→	→	→	→	↓	↓	↓
7	→	→	→	→	→	→	→	→	↓	↓
8	↑	↑	↑	→	→	→	→	→	→	↓
9	↑	←	←	→	→	→	→	→	→	↓

Question 6: (10 points) Modify the environment of the agent by replacing Reward function 1 with Reward function 2. Use the optimal state-value function implemented in question 2 to compute the optimal value of each state in the grid. For visualization purpose, you should generate a figure similar to that of figure 1 but with the number of state replaced by the optimal value of that state. In this question, you should have 1 plot.

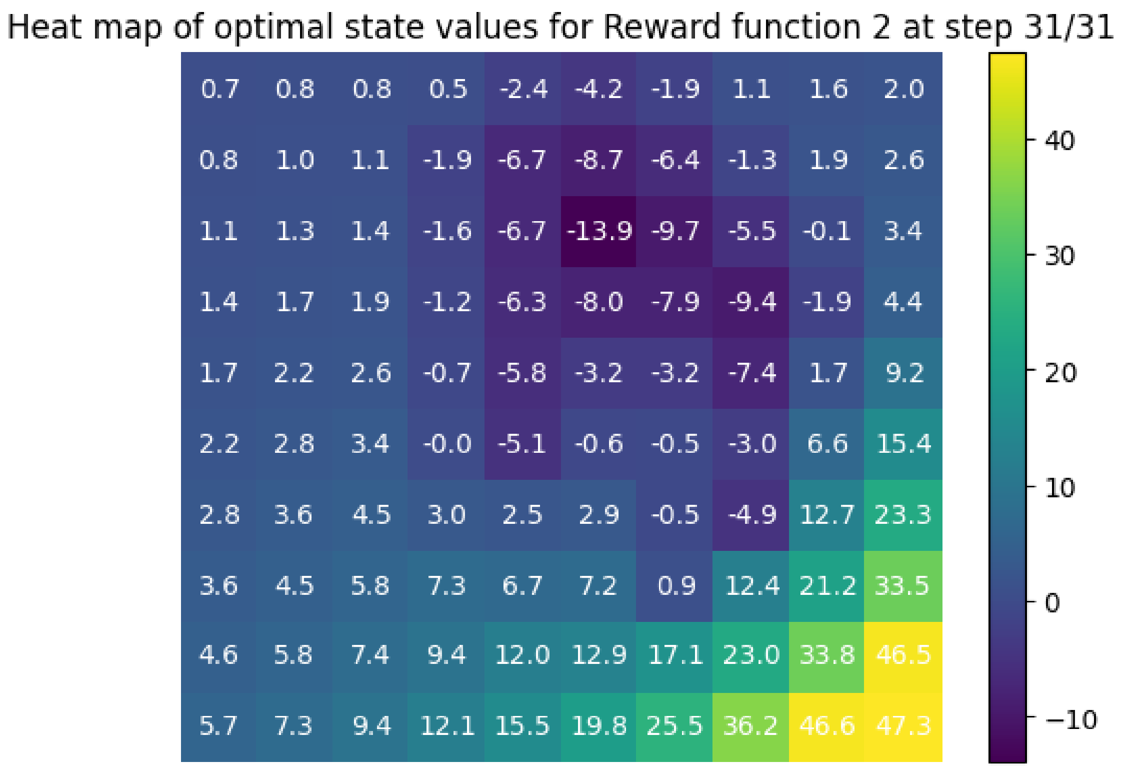
Answer: The following is the optimal value of each state with reward function 2.



	0	1	2	3	4	5	6	7	8	9
0	0.7	0.8	0.8	0.5	-2.4	-4.2	-1.9	1.1	1.6	2.0
1	0.8	1.0	1.1	-1.9	-6.7	-8.7	-6.4	-1.3	1.9	2.6
2	1.1	1.3	1.4	-1.6	-6.7	-13.9	-9.7	-5.5	-0.1	3.4
3	1.4	1.7	1.9	-1.2	-6.3	-8.0	-7.9	-9.4	-1.9	4.4
4	1.7	2.2	2.6	-0.7	-5.8	-3.2	-3.2	-7.4	1.7	9.2
5	2.2	2.8	3.4	-0.0	-5.1	-0.6	-0.5	-3.0	6.6	15.4
6	2.8	3.6	4.5	3.0	2.5	2.9	-0.5	-4.9	12.7	23.3
7	3.6	4.5	5.8	7.3	6.7	7.2	0.9	12.4	21.2	33.5
8	4.6	5.8	7.4	9.4	12.0	12.9	17.1	23.0	33.8	46.5
9	5.7	7.3	9.4	12.1	15.5	19.8	25.5	36.2	46.6	47.3

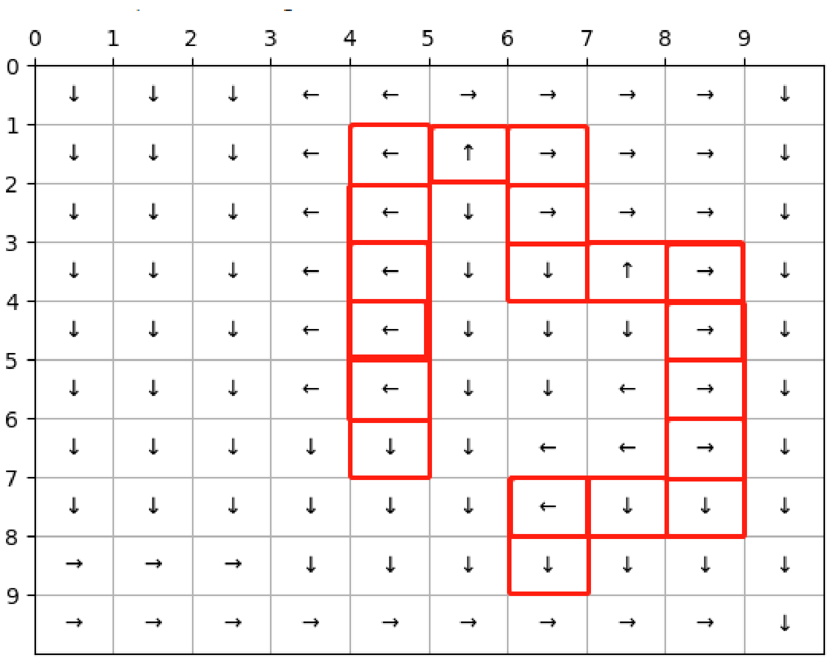
Question 7: (20 points) Generate a heat map of the optimal state values (found in question 6) across the 2-D grid. For generating the heat map, you can use the same function provided in the hint earlier. Explain the distribution of the optimal state values across the 2-D grid. (Hint: Use the figure generated in this question to explain)

Answer: The following is the heat map of the optimal state values. It takes 31 steps to converge. We can see that there is a lowest score at state 52 with value of -13.9. We can see in the reward function that this state is kind of "trapped": the upper, left, right bound of it are all traps. and it takes a long detour to get to the destination state. It is highly possible that the agent got blowed to the traps and receive a negative reward. And similar to the heatmap of reward function 1, the buttom right corner has a highest value and there is a smooth change around that state, the value of that point is 47.3. It is similar to Q2 that if there is no wind, that value should be 50.



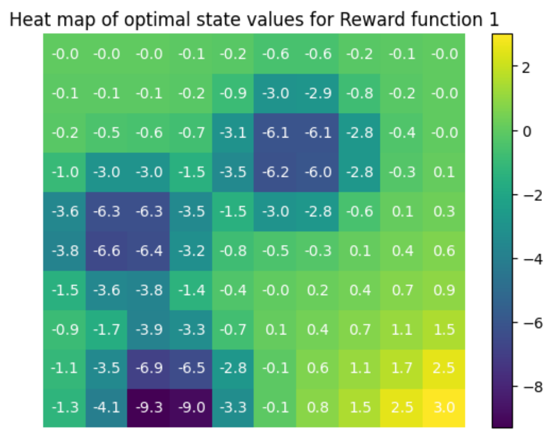
Question 8: (20 points) Implement the computation step of the value iteration algorithm (lines 14-17) to compute the optimal policy of the agent navigating the 2-D state-space. For visualization purpose, you should generate a figure similar to that of figure 1 but with the number of state replaced by the optimal action at that state. The optimal actions should be displayed using arrows. Does the optimal policy of the agent match your intuition? Please provide a brief explanation. In this question, you should have 1 plot.

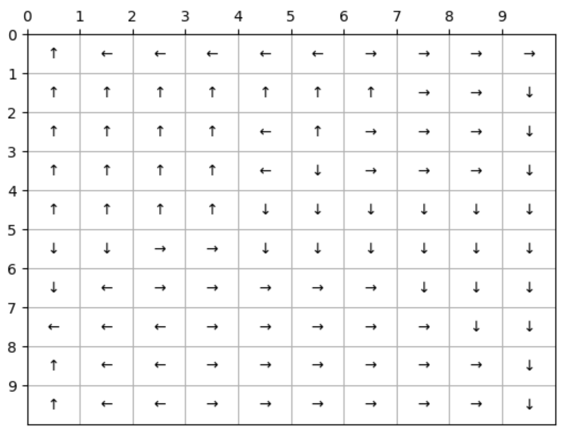
Answer: The following is the optimal actions. We also highlighted the negative rewards (traps) using red boxes. We can see that the actions are in consist of our intuition that all the states tried to find its own way to the destination and none of the state take action to hit the trap.



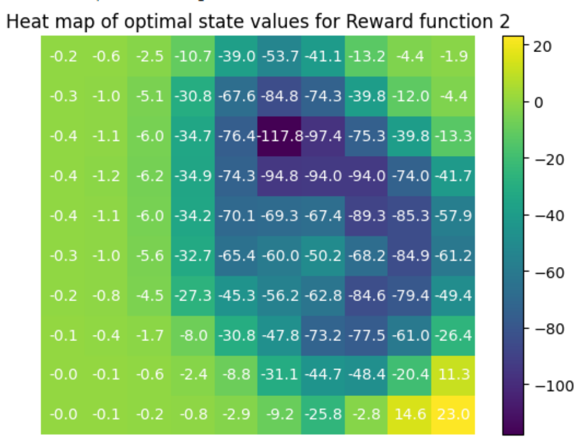
Question 9:(20 points) Change the hyper parameter w to 0.6 and find the optimal policy map similar to previous question for reward functions. Explain the differences you observe. What do you think about value of new w compared to previous value? Choose the w that you think give rise to better optimal policy and use that w for the next stages of the project.

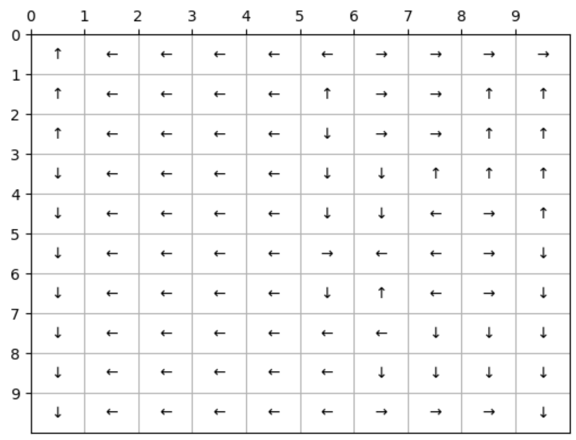
Answer: with w changed to 0.6, it means that the agent has a higher probability to hit the trap, because the optimal action will never take it to the trap, it can only stand on the trap because of the wind, The following is the heatmap of the optimal state value of reward function 1 and the optimal action.





The following is the optimal state value of reward function 2 and the optimal action.





Observing the plots, we can find there exist local optimal. For example, in the state 27 of the reward function 1, the agent choose to go to left and stays at a edge and don’t go to the destination. In this case, although it will never receive the positive reward, it can stay on the edge and always get 0, avoiding a high probability to be blowed to the trap and get a negative reward. The same thing happens in the reward function 2, where the state 91 choose to go to up and stay forever.

I think a better policy will be when w is 0.1. since in this case, we can the agent will try to get to the buttom right and receive the positive reward forever, but not try to avoid a temporary negative reward and stays at a zero reward point.

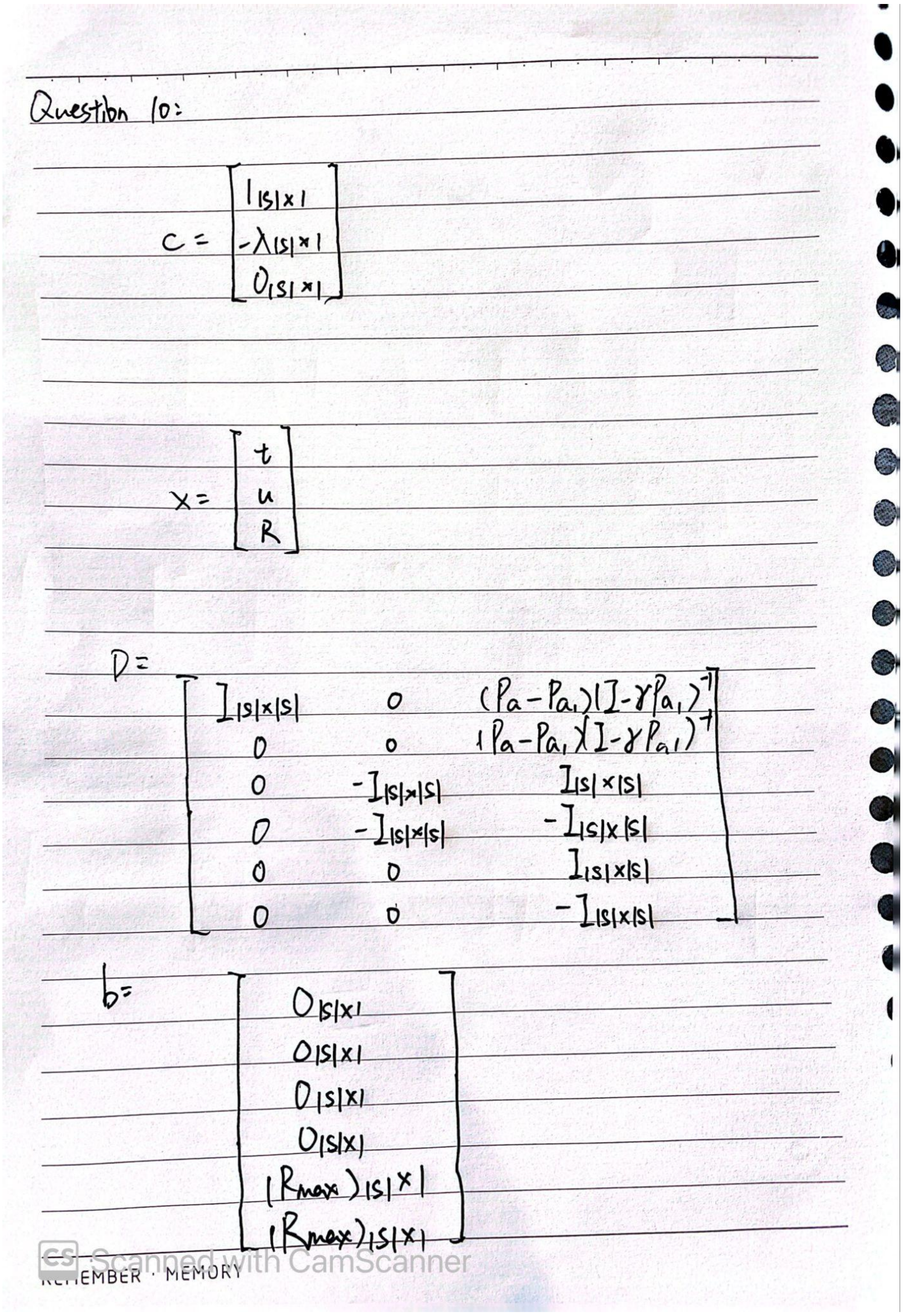
A little bit follow up:

Actually, the trap is like the short term incentive and the destination is the long term incentive. (For a state at upper left, the traps is nearer to it and the destination is farther from it.) So we find it very interesting that if we increase the discount factor and pay more attention to the long term reward, the policy will change and in that case all the states still try to get to the buttom right state.

We increase the discount factor to 0.99 and find that for reward function 1, there will be no more local optimal. (But this factor is still not enough for reward function 2.) We further increase the factor to 0.999 and there is no local optimal for reward function 2. You can find the plots in our notebook.

Question 10: (10 points) Express c , x , D , b in terms of R , P a , P a 1 , t i , u , λ and R max

Answer:



Question 10:

$$c = \begin{bmatrix} 1_{|S|\times 1} \\ -\lambda_{|S|\times 1} \\ 0_{|S|\times 1} \end{bmatrix}$$

$$x = \begin{bmatrix} t \\ u \\ R \end{bmatrix}$$

$D =$

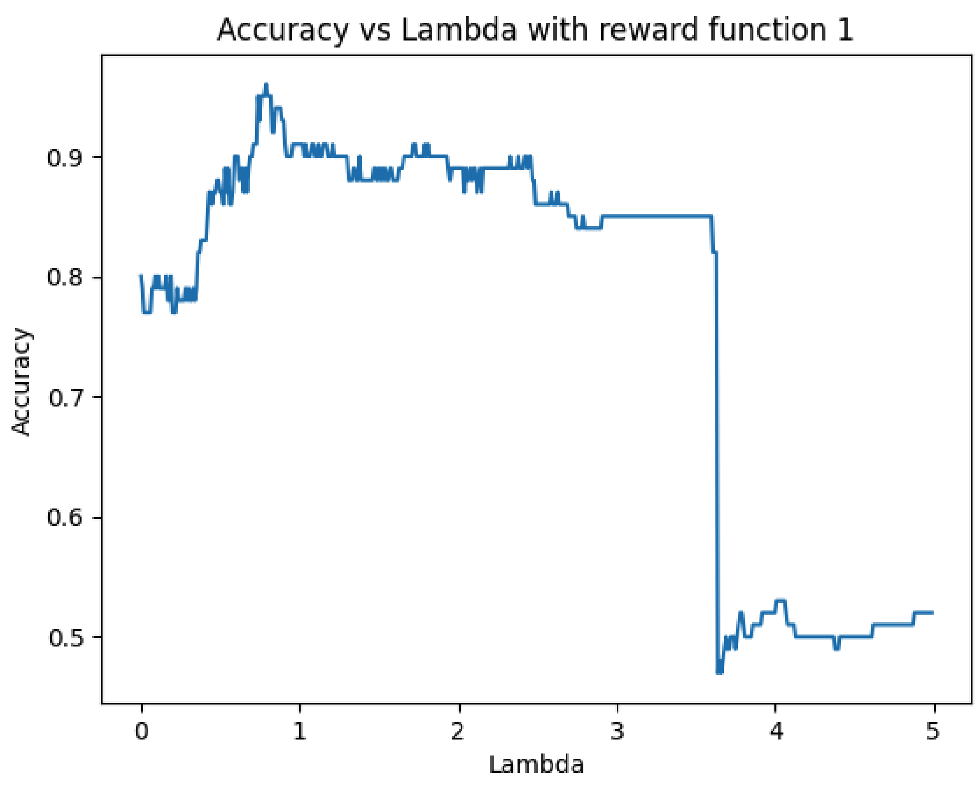
$$\begin{bmatrix} I_{|S|\times|S|} & 0 & (P_a - P_{a_1})(I - \gamma P_{a_1})^{-1} \\ 0 & 0 & (P_a - P_{a_1})(I - \gamma P_{a_1})^{-1} \\ 0 & -I_{|S|\times|S|} & I_{|S|\times|S|} \\ 0 & -I_{|S|\times|S|} & -I_{|S|\times|S|} \\ 0 & 0 & I_{|S|\times|S|} \\ 0 & 0 & -I_{|S|\times|S|} \end{bmatrix}$$

$$b = \begin{bmatrix} 0_{|S|\times 1} \\ 0_{|S|\times 1} \\ 0_{|S|\times 1} \\ 0_{|S|\times 1} \\ (R_{max})_{|S|\times 1} \\ (R_{max})_{|S|\times 1} \end{bmatrix}$$

Question 11: (30 points) Sweep λ from 0 to 5 to get 500 evenly spaced values for λ . For each value of λ compute O A (s) by following the process described above. For this problem, use the optimal policy of the agent found in question 5 to fill in the O E (s) values. Then use *equation 3 to compute the accuracy of the IRL algorithm for this value of λ . You need to repeat the above process for all 500 values of λ to get 500 data points. Plot λ (x -axis) against Accuracy (y -axis). In this question, you should have 1 plot.*

Answer:

```
100%                                              500/500 [02:23<00:00, 10.04it/s]
```



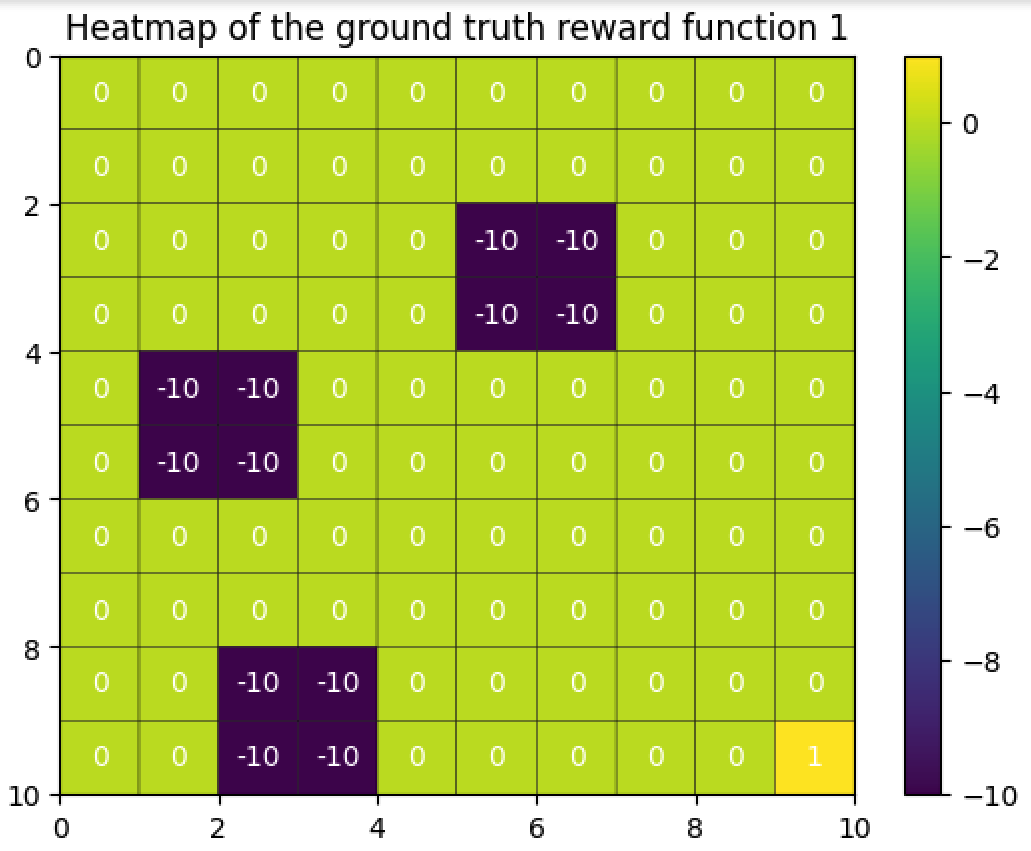
Question 12: (5 points) Use the plot in question 11 to compute the value of λ for which accuracy is maximum. For future reference we will denote this value as λ (1) max . Please report λ (1) max

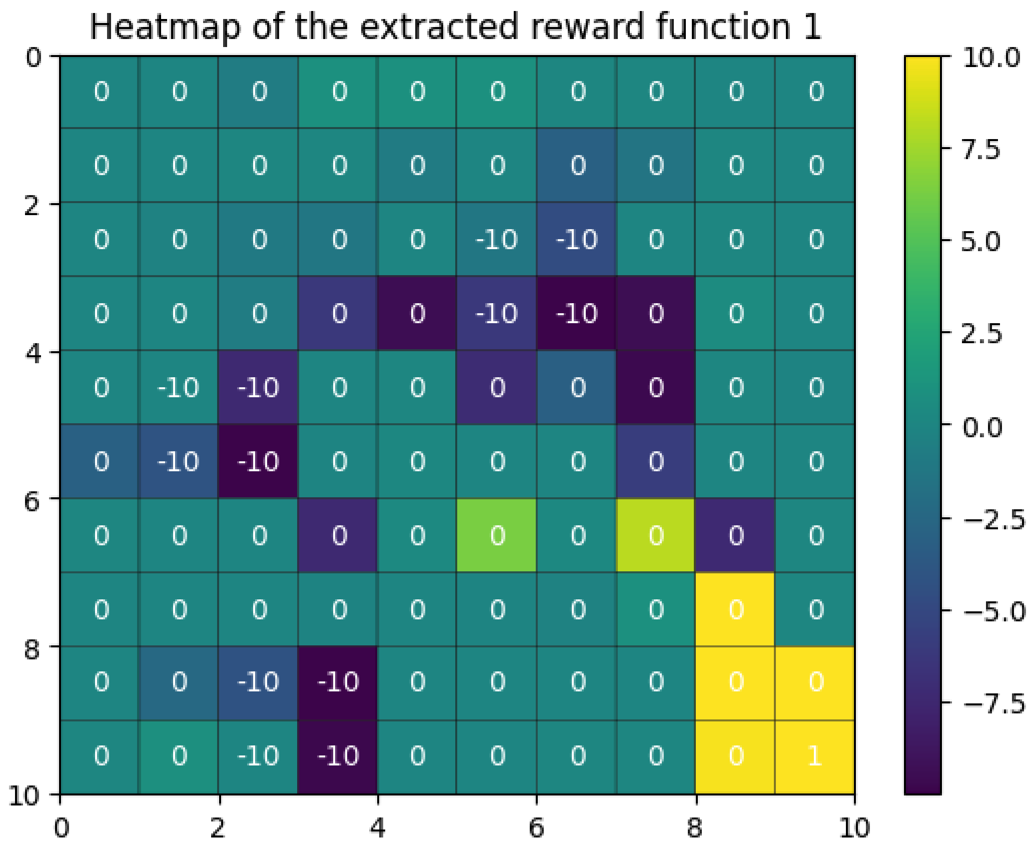
Answer:

The accuracy reached the highest (96%) as *λ (1) max* equals 0.79.

Question 13: (15 points) For λ (1) max , generate heat maps of the ground truth reward and the extracted reward. Please note that the ground truth reward is the Reward function 1 and the extracted reward is computed by solving the linear program given by equation 2 with the λ parameter set to λ (1) max . In this question, you should have 2 plots.

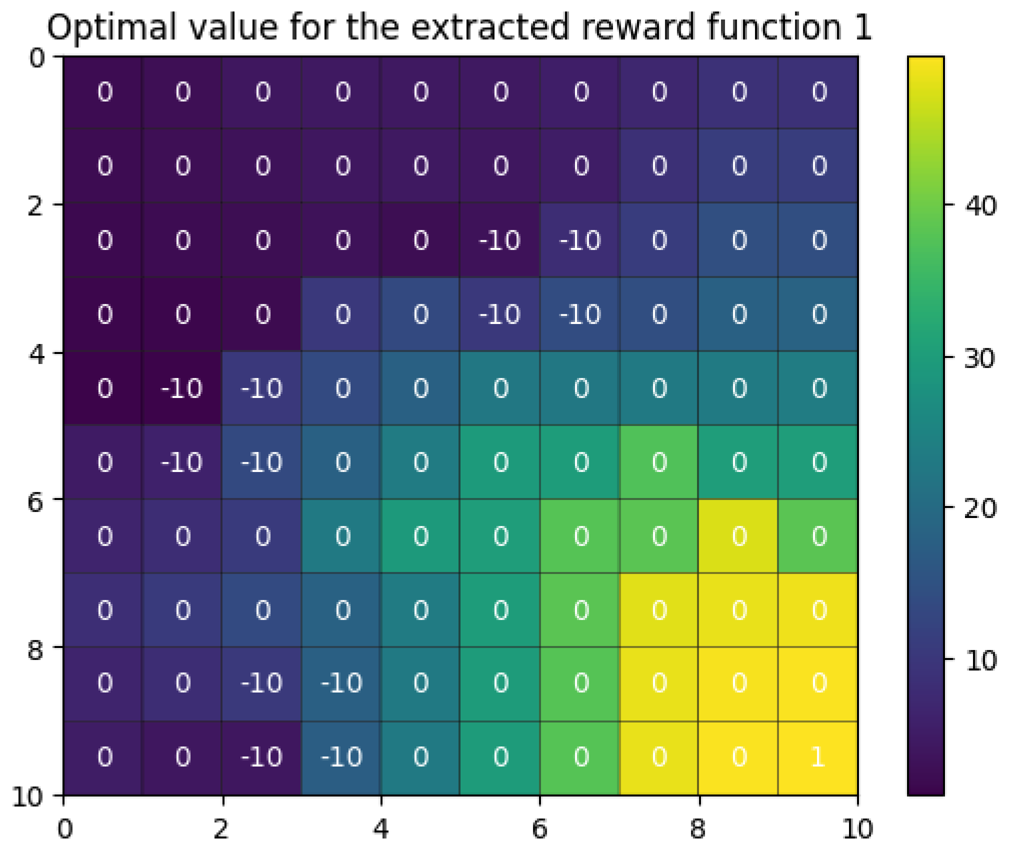
Answer:


Heatmap of the ground truth reward function 1
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 -10 -10 0 0 0
0 0 0 0 0 -10 -10 0 0 0
0 -10 -10 0 0 0 0 0 0 0
0 -10 -10 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 -10 -10 0 0 0 0 0 0
0 0 -10 -10 0 0 0 0 0 1
0
−2
−4
−6
−8
−10


Heatmap of the extracted reward function 1
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 -10 -10 0 0 0
0 0 0 0 0 -10 -10 0 0 0
0 -10 -10 0 0 0 0 0 0 0
0 -10 -10 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 -10 -10 0 0 0 0 0 0
0 0 -10 -10 0 0 0 0 0 1
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5

Question 14: (10 points) Use the extracted reward function computed in question 13, to compute the optimal values of the states in the 2-D grid. For computing the optimal values you need to use the optimal state-value function that you wrote in question 2. For visualization purpose, generate a heat map of the optimal state values across the 2-D grid (similar to the figure generated in question 3). In this question, you should have 1 plot.

Answer:



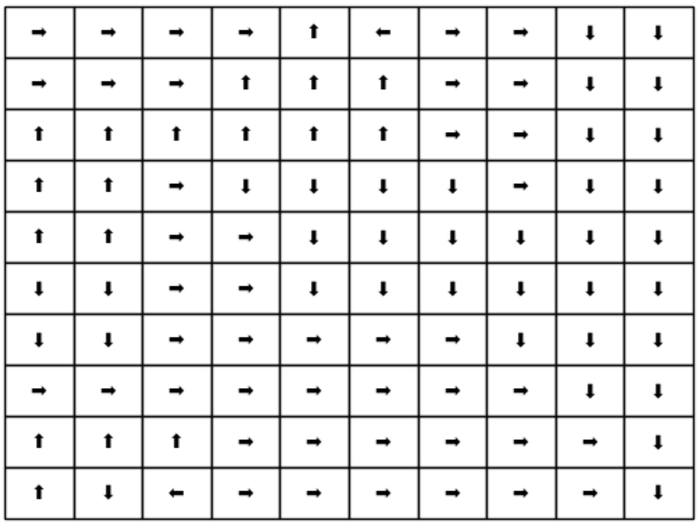
Question 15: (10 points) Compare the heat maps of Question 3 and Question 14 and provide a brief explanation on their similarities and differences.

Answer:

In terms of similarities, both cases exhibit the highest optimal value at state 99, and the optimal values progressively decrease from the bottom right of the map. However, there are notable differences between the two heatmaps. Firstly, they have different scales, indicating variations in the magnitudes of the optimal values. Additionally, the heatmap of Question 3 bears a closer resemblance to its corresponding reward function.The regions in the heatmap in Question 3 are much more well-defined and homogeneous compared to Question 14. This can be attributed to the fact that the ground truth reward function displays clearer clusters, whereas the extracted reward function contains more noise. Consequently, the optimal values derived from the extracted reward function may deviate to some extent from the underlying reward function, resulting in dissimilarities.

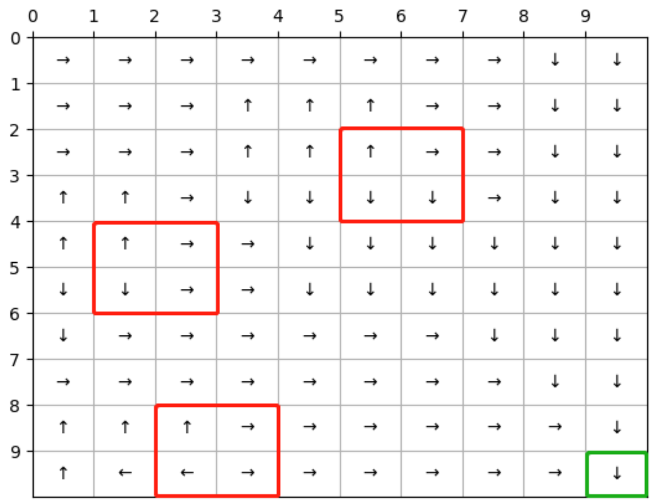
Question 16: (10 points) Use the extracted reward function found in question 13 to compute the optimal policy of the agent. For computing the optimal policy of the agent you need to use the function that you wrote in question 5. For visualization purpose, you should generate a figure similar to that of figure 1 but with the number of state replaced by the optimal action at that state. The actions should be displayed using arrows. In this question, you should have 1 plot.

Answer:



Question 17: (10 points) Compare the figures of Question 5 and Question 16 and provide a brief expla- nation on their similarities and differences.

Answer:

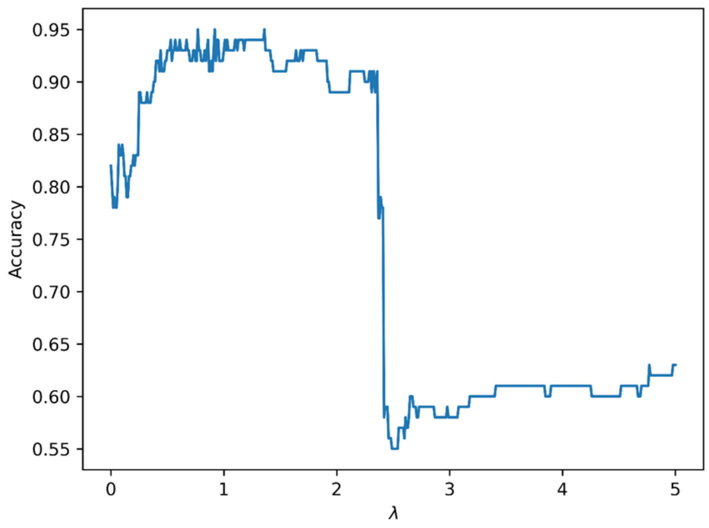


Question 5 Figure

I also showed the question5 figure. We know that with an accuracy of 0.96, the two sets of data are highly similar, sharing nearly identical actions. Majority of the actions in both maps involve moving down and right. This is because state 99, which is the most optimal state, is located in the lower right corner in both cases. For difference, Some of the actions in Question 16 have the potential to result in the agent being blown off the grid, which means it will no longer be within the boundaries of the grid. However, this is not the case for the agent in Question 5. The reason for this difference is that in Question 16, there is a higher probability for the agent to choose actions that might lead it away from the optimal state, which is state 99. On the other hand, in Question 5, the agent's actions are more likely to guide it towards the optimal state rather than away from it. Therefore, the agent in Question 5 has a better chance of reaching the desired state compared to the agent in Question 16. And For state 16, despite the differences, it still leads to convergence towards state 99. However, for states 19, 40, and 50, the disparities result in the emergence of two additional local optimal points. The divergence in optimal actions for these states can be attributed to the extracted reward function, which assigns a higher reward to states 19 and 40 compared to their neighboring states. Consequently, the most favorable action at these points deviates from the original path towards state 99.

Question 18: Sweep λ from 0 to 5 to get 500 evenly spaced values for λ. For each value of λ compute OA(s) by following the process described above. For this problem, use the optimal policy of the agent found in question 8 to fill in the OE(s) values. Then use equation 3 to compute the accuracy of the IRL algorithm for this value of λ. You need to repeat the above process for all 500 values of λ to get 500 data points. Plot λ (x-axis) against Accuracy (y-axis). In this question, you should have 1 plot.

Answer:

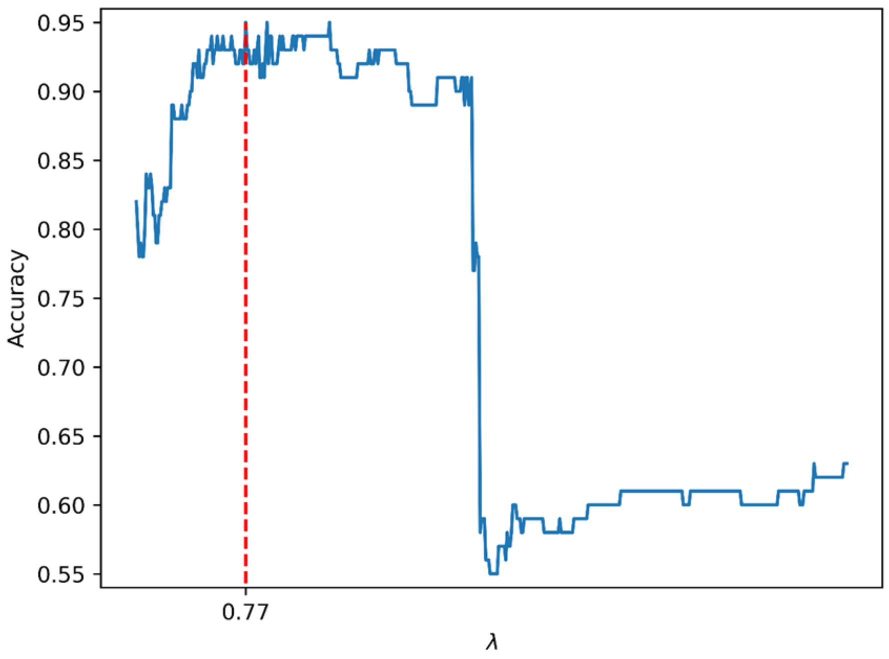


Lambda against Accuracy with Extracted Reward Functions

As shown in this graph, when lambda is greater than 2.4, there is a huge drop in the performance of the accuracy. Also, when lambda is smaller than 0.7, the accuracy gradually increases as the lambda increases.

Question 19: (5 points) Use the plot in question 18 to compute the value of λ for which accuracy is maximum. For future reference we will denote this value as λmax. Please report λmax.

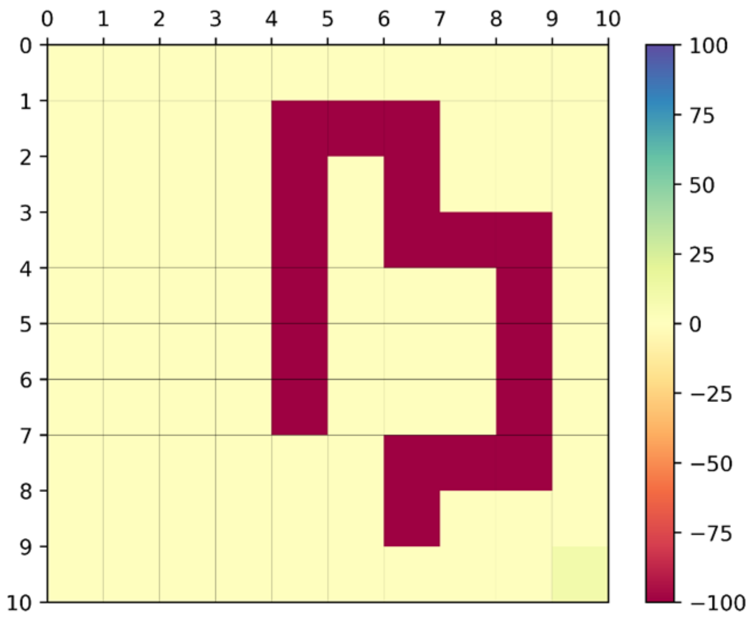
Answer:



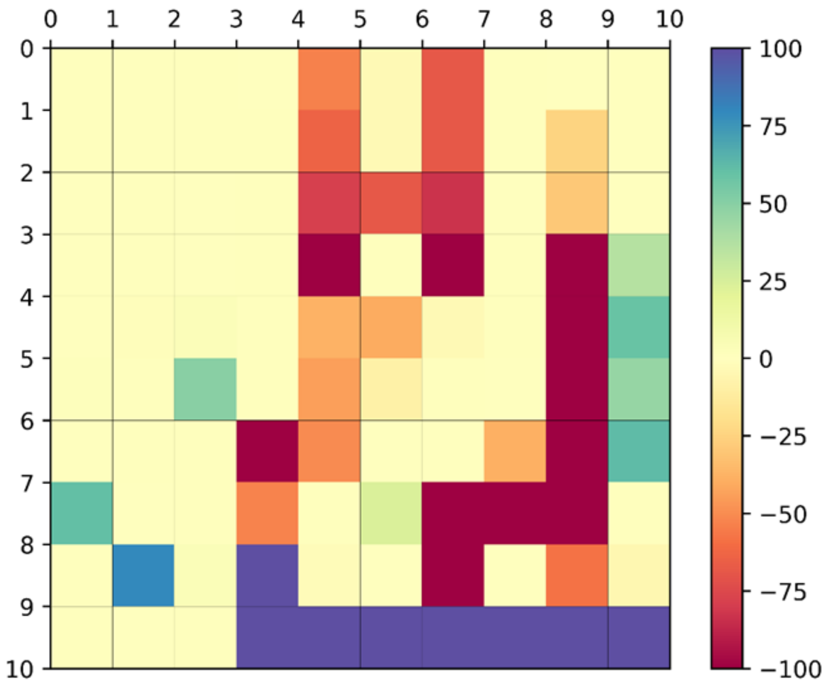
The accuracy reached the highest (95%) as lambda equals 0.77.

Question 20: For λmax, generate heat maps of the ground truth reward and the extracted reward. Please note that the ground truth reward is the Reward function 2 and the extracted reward is computed by solving the linear program given by equation 2 with the λ parameter set to λmax. In this question, you should have 2 plots.

Answer:



Heat map of ground truth rewards

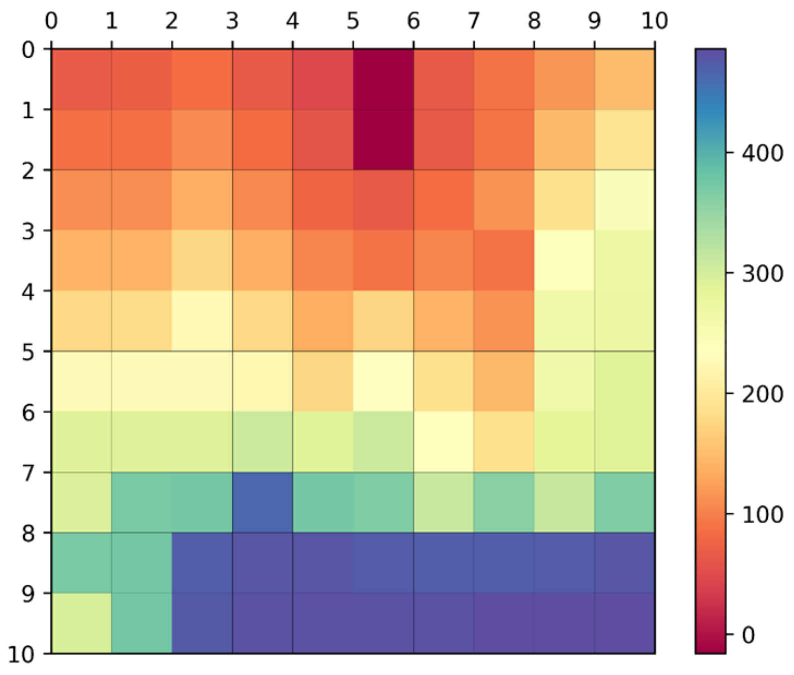


Heat map of extracted rewards

From the figures, we observe that the extracted reward function 2 is different from the ground truth Reward Function 2. The extracted rewards keep most of the shapes and values of the negative rewards of ground truth, while the upper and left shape has not been extracted well. Also, the positive rewards extracted has a much higher value of up to 100. This is caused by the LP algorithm that cares about the highest absolute value. Thus, the points that are close to the bonus most have a high positive bonus, which is quite different from the ground truth data.

Question 21: Use the extracted reward function computed in question 20, to compute the optimal values of the states in the 2-D grid. For computing the optimal values, you need to use the optimal state-value function that you wrote in question 2. For visualization purposes, generate a heat map of the optimal state values across the 2-D grid (similar to the figure generated in question 7). In this question, you should have 1 plot.

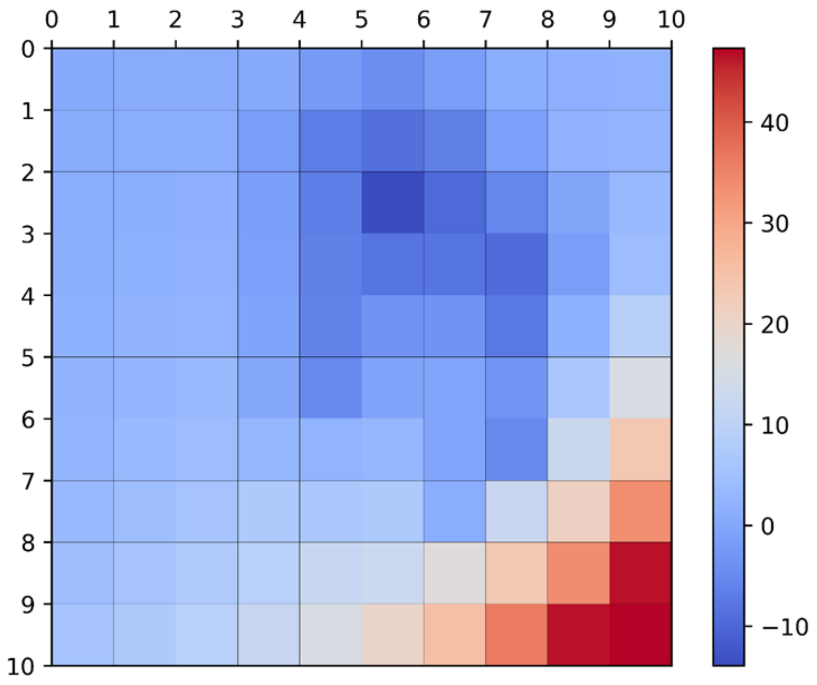
Answer:



Heat Map of Optimal State Values with Extracted Reward Function

Question 22: Compare the heat maps of Question 7 and Question 21 and provide a brief explanation on their similarities and differences.

Answer:



Heat Maps of Optimal State Values of reward function 2 (Ground truth)

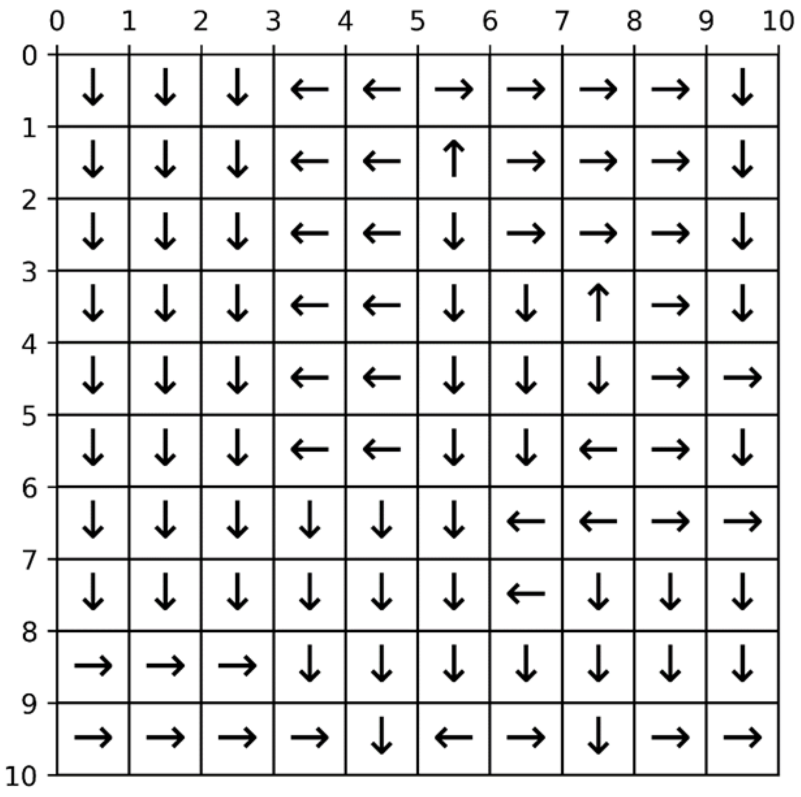
The above figures exhibit both similarities and differences.

Similarities: In both figures, the best state values decrease from the bottom right, specifically from state 99. Both maps display a ladder-like structure, owing to the circular arrangement in the center of the two reward functions.

Differences: Firstly, there is a significant difference in the color scales between the figures. The ground truth figure presents state values ranging from -15 to 45, whereas, in the extracted figure, these values range from -10 to 500. This discrepancy is due to the reward function in the extracted version having a higher upper limit. Secondly, the value distribution also varies between the two. In the ground truth figure, values reduce from the bottom-right to the top-left, with a distinct ladder-like pattern in the upper middle. Contrarily, in the extracted figure, there's a general tendency for the values to reduce from bottom to top, with the lowest values appearing at states 50 and 51. This variation can be attributed to the reward function in the extracted map, where the last row states have considerably higher rewards compared to the other states, thus the declining trend is from bottom to top.

Question 23: Use the extracted reward function found in question 20 to compute the optimal policy of the agent. For computing the optimal policy of the agent you need to use the function that you wrote in question 9. For visualization purposes, you should generate a figure similar to that of figure 1 but with the number of states replaced by the optimal action at that state. The actions should be displayed using arrows. In this question, you should have 1 plot.

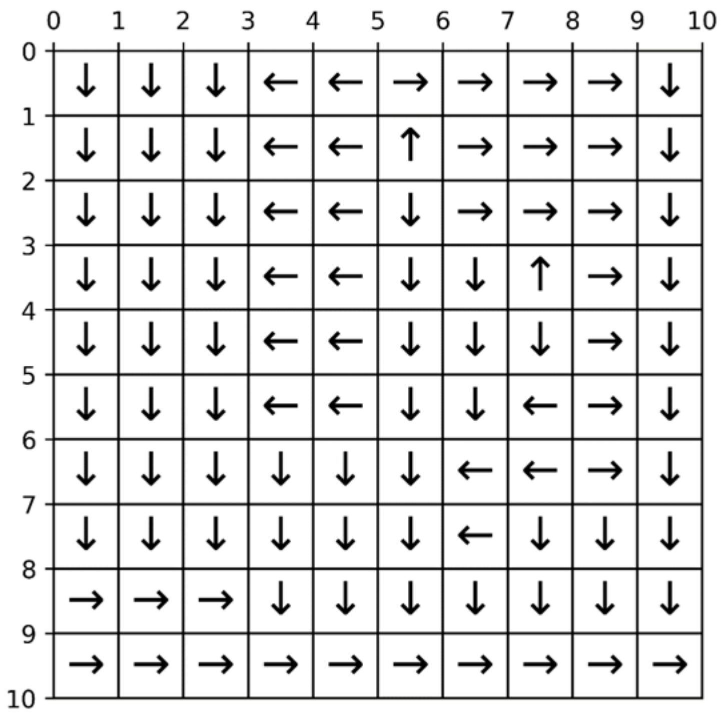
Answer:



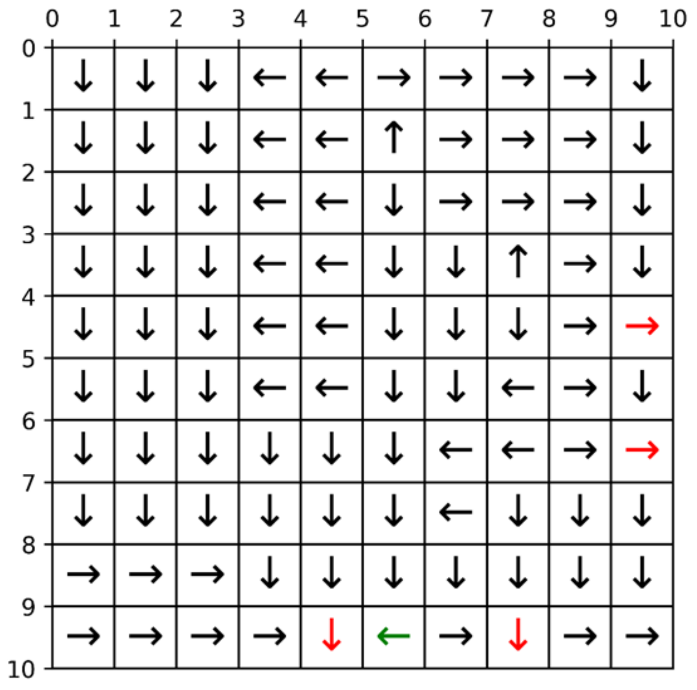
Question 24: (10 points) Compare the figures of Question 9 and Question 23 and provide a brief explanation on their similarities and differences.

Answer:

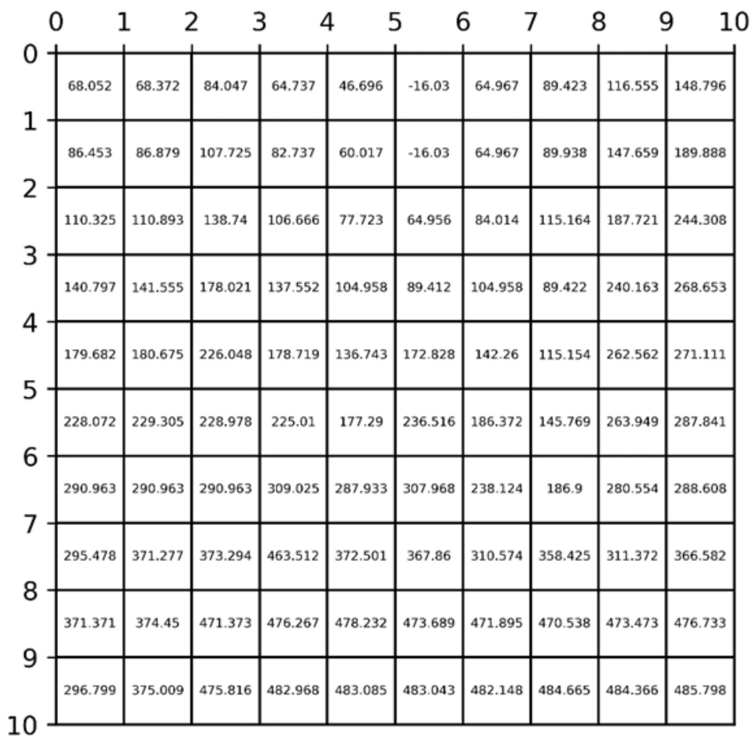
The optimal policy in reinforcement learning (ground truth):



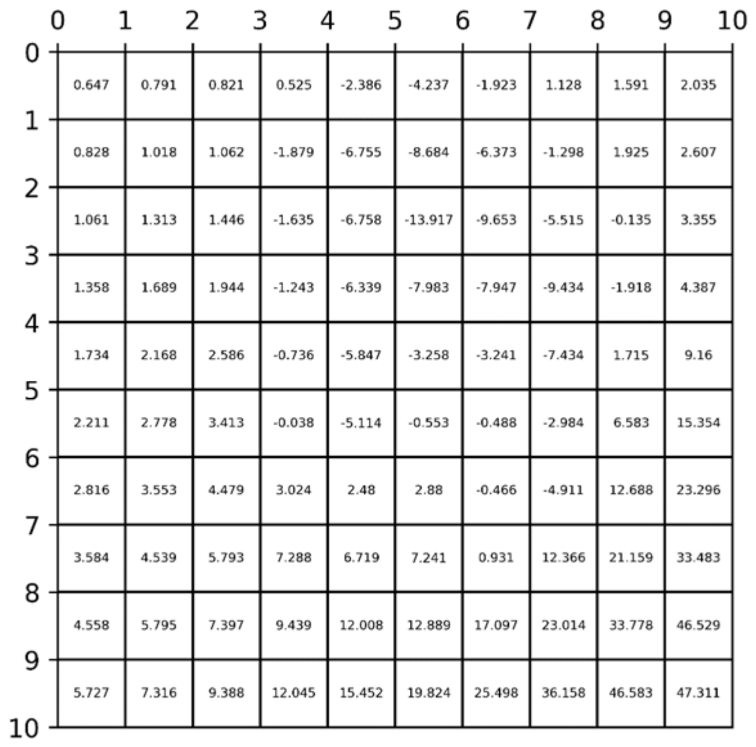
The optimal policy produced by inverse reinforcement learning:



The diagrams above highlight cells with double arrows to indicate deviations in optimal policies compared to the ground truth. The accuracy of the optimal policies between these two figures is around 95%, signifying a substantial overlap. However, there are two primary zones of divergence, designated by red and blue circles. The reasons behind these differences will be elucidated in the subsequent question. The optimal state value is provided for analyze purposes.



68.052	68.372	84.047	64.737	46.696	-16.03	64.967	89.423	116.555	148.796
86.453	86.879	107.725	82.737	60.017	-16.03	64.967	89.938	147.659	189.888
110.325	110.893	138.74	106.666	77.723	64.956	84.014	115.164	187.721	244.308
140.797	141.555	178.021	137.552	104.958	89.412	104.958	89.422	240.163	268.653
179.682	180.675	226.048	178.719	136.743	172.828	142.26	115.154	262.562	271.111
228.072	229.305	228.978	225.01	177.29	236.516	186.372	145.769	263.949	287.841
290.963	290.963	290.963	309.025	287.933	307.968	238.124	186.9	280.554	288.608
295.478	371.277	373.294	463.512	372.501	367.86	310.574	358.425	311.372	366.582
371.371	374.45	471.373	476.267	478.232	473.689	471.895	470.538	473.473	476.733
296.799	375.009	475.816	482.968	483.085	483.043	482.148	484.665	484.366	485.798



0.647	0.791	0.821	0.525	-2.386	-4.237	-1.923	1.128	1.591	2.035
0.828	1.018	1.062	-1.879	-6.755	-8.684	-6.373	-1.298	1.925	2.607
1.061	1.313	1.446	-1.635	-6.758	-13.917	-9.653	-5.515	-0.135	3.355
1.358	1.689	1.944	-1.243	-6.339	-7.983	-7.947	-9.434	-1.918	4.387
1.734	2.168	2.586	-0.736	-5.847	-3.258	-3.241	-7.434	1.715	9.16
2.211	2.778	3.413	-0.038	-5.114	-0.553	-0.488	-2.984	6.583	15.354
2.816	3.553	4.479	3.024	2.48	2.88	-0.466	-4.911	12.688	23.296
3.584	4.539	5.793	7.288	6.719	7.241	0.931	12.366	21.159	33.483
4.558	5.795	7.397	9.439	12.008	12.889	17.097	23.014	33.778	46.529
5.727	7.316	9.388	12.045	15.452	19.824	25.498	36.158	46.583	47.311

Q25

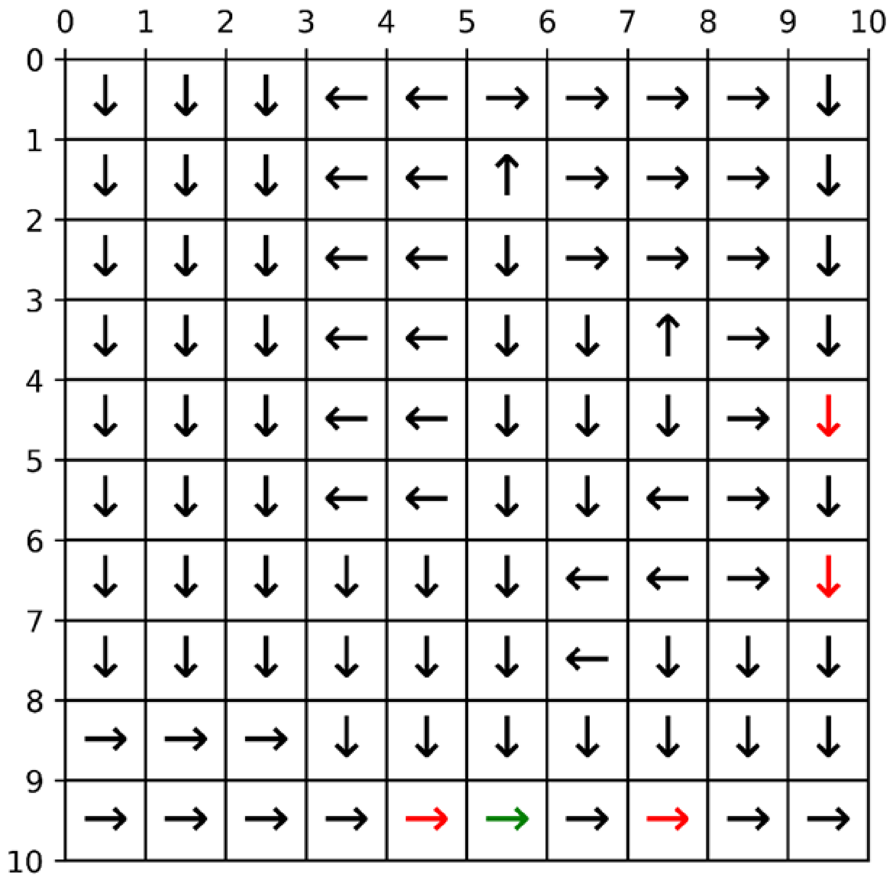
Major discrepancies:

1. For state 59, the optimal policy goes against the direction of reinforcement learning. We found that in the optimal state value graph, the values of 49, 59, 69, and 79 are quite close.

This indicates that the possibilities of choosing four directions are close and the policy is not stable. We can reduce the value of the threshold ϵ, to ensure the optimal state value is stable enough.

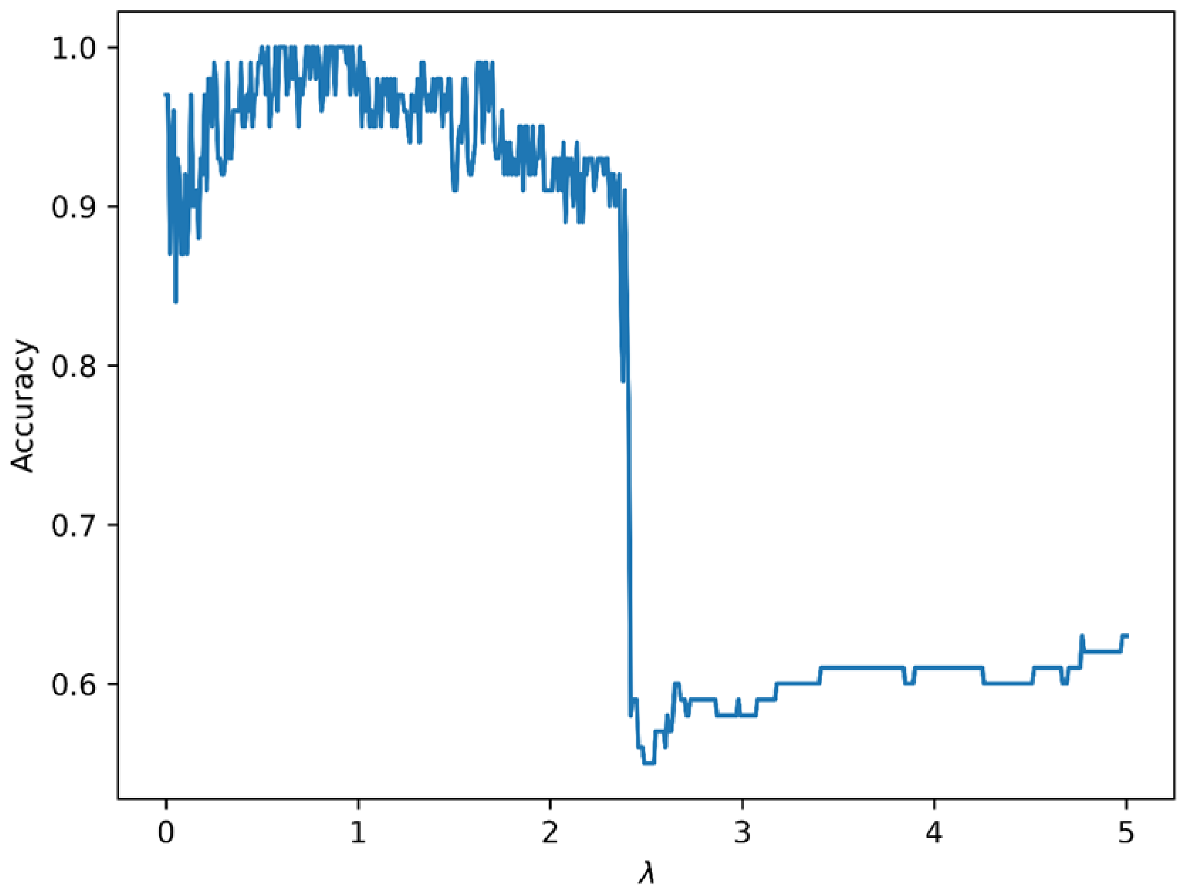
2. On the right and the below edge, there were two states where the optimal policy is pointing outside the box. This is caused by the optimal state value being highest than its neighbors, and it prefers to keep its old state. This could be due to the LP optimal function could be oversimplified when considering some small but positive rewards.

To fix the first discrepancy, the threshold ϵ is set smaller to 0.00001.



As shown in this graph, the policy has been modified to be the same as the policy in reinforcement learning.

The maximum accuracy of the modified version has become 100%.



Moreover, the solution to the convergence problem that some states tend to point outside is hard to solve. We leave it as future scope.