Large Scale Social Network : Models and Algorithms Project #3

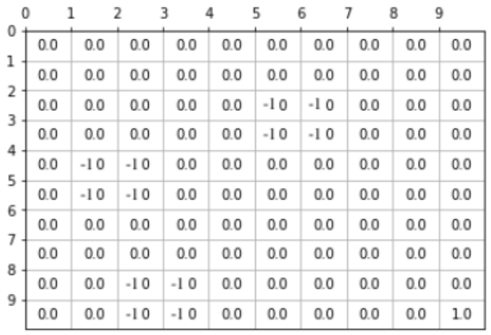
Due on May 24th, 2020 at 9:00pm

Professor Roychowdhury, Vwani

Wang, Yin, He

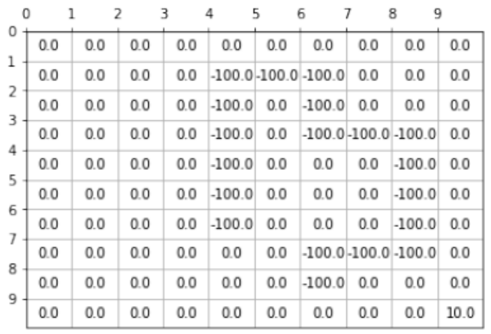
Question 1

In this project, we are going to explore some concepts in *Reinforcement Learning* (RL) and *Inverse Reinforcement Learning*. Specifically, in the first part of the project, we will explore the optimal policy of an agent navigating in a 2-D environment. Detailed discussion on RL components can be found in the project handout, thus in this report, we will just briefly describe them. Before going into the RL algorithms, in this question we are going to plot the heat maps of the two Reward functions we are going to use in this project. They are defined as Figure 1. Specifically, the value of reward function 1 is somewhat unclear to distinguish, we will treat them as -1 or $+1$. All the following results are derived based on this assumption, which might cause somewhat difference to the standard result.



	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	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	-1.0	-1.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	-1.0	-1.0	0.0	0.0	0.0
4	0.0	-1.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.0	-1.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	0.0	0.0	-1.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0
9	0.0	0.0	-1.0	-1.0	0.0	0.0	0.0	0.0	0.0	1.0

(a) Reward function 1

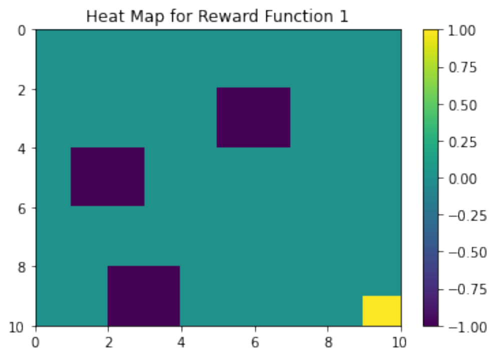


	0	1	2	3	4	5	6	7	8	9
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	-100.0	-100.0	-100.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	-100.0	0.0	-100.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	-100.0	0.0	-100.0	-100.0	-100.0	0.0
4	0.0	0.0	0.0	0.0	-100.0	0.0	0.0	0.0	-100.0	0.0
5	0.0	0.0	0.0	0.0	-100.0	0.0	0.0	0.0	-100.0	0.0
6	0.0	0.0	0.0	0.0	-100.0	0.0	0.0	0.0	-100.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	-100.0	-100.0	-100.0	0.0
8	0.0	0.0	0.0	0.0	0.0	0.0	-100.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	10.0

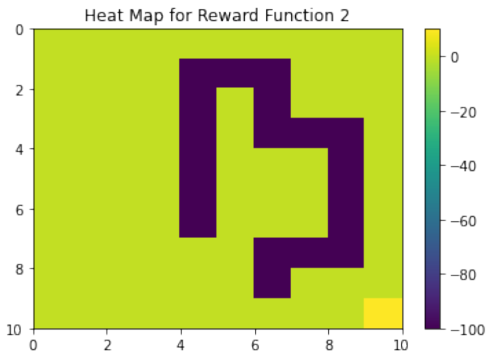
(b) Reward function 2

Figure 1: Reward functions for the 2-D square grid

And the resulting heatmaps are shown as Figure 2:



(a) Heat map for reward function 1



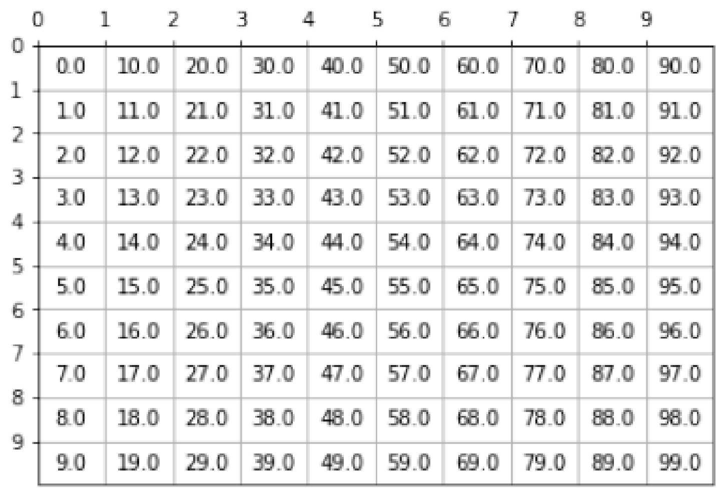
(b) Heat map for reward function 2

Figure 2: Corresponding heat maps for two reward functions

Question 2

In this question, we will start to build the actual RL algorithm. Specifically, we will use the Value iteration algorithm for this project. First, the environment is modeled by a Markov Decision Process (MDP), and the state space is defined as a 2-D square grid with 100 states, as shown in the Figure 3:

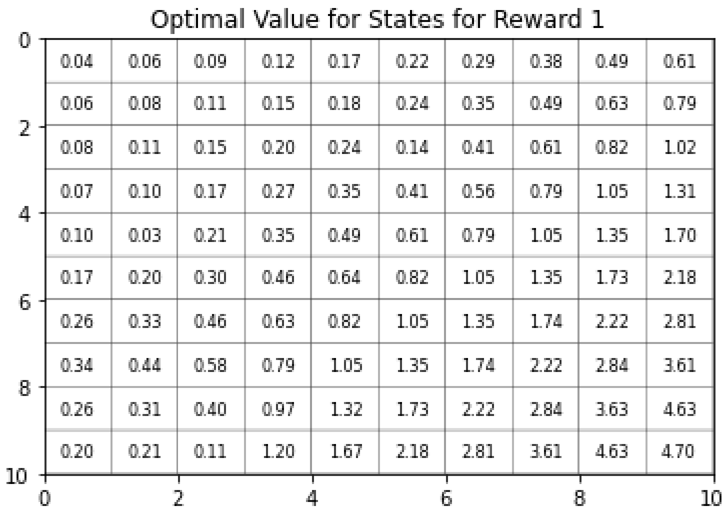
Question 2 continued on next page. . .



	0	1	2	3	4	5	6	7	8	9
0	0.0	10.0	20.0	30.0	40.0	50.0	60.0	70.0	80.0	90.0
1	1.0	11.0	21.0	31.0	41.0	51.0	61.0	71.0	81.0	91.0
2	2.0	12.0	22.0	32.0	42.0	52.0	62.0	72.0	82.0	92.0
3	3.0	13.0	23.0	33.0	43.0	53.0	63.0	73.0	83.0	93.0
4	4.0	14.0	24.0	34.0	44.0	54.0	64.0	74.0	84.0	94.0
5	5.0	15.0	25.0	35.0	45.0	55.0	65.0	75.0	85.0	95.0
6	6.0	16.0	26.0	36.0	46.0	56.0	66.0	76.0	86.0	96.0
7	7.0	17.0	27.0	37.0	47.0	57.0	67.0	77.0	87.0	97.0
8	8.0	18.0	28.0	38.0	48.0	58.0	68.0	78.0	88.0	98.0
9	9.0	19.0	29.0	39.0	49.0	59.0	69.0	79.0	89.0	99.0

Figure 3: 2-D square grid with states numbering

And the agent can take four actions, which are move right, move left, move up and move down. The definition of the transition probabilities is defined in detail in the handout, thus we will not describe them in this report. One thing to be noticed is the definition of w, where we define that each action corresponds to a movement in the intended direction with probability 1?w, but has a probability of w of moving in a random direction instead due to wind. In this question, we set ? to be 0.1, that means the agent will go in the intended direction with high probability and move to a random direction with low probability. Next, we will create the environment and 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. Specifically, we will implement the Initialization part and Estimation parts of the Value Iteration algorithm, which can be found in the project handout. Note that we set the discount factor to be 0.8, which is the γ in the Bellman Equation and the threshold to be 0.01 for the iteration. And in this question, we are using Reward Function 1. The output is shown as Figure 4.


Optimal Value for States for Reward 1

0.04	0.06	0.09	0.12	0.17	0.22	0.29	0.38	0.49	0.61
0.06	0.08	0.11	0.15	0.18	0.24	0.35	0.49	0.63	0.79
0.08	0.11	0.15	0.20	0.24	0.14	0.41	0.61	0.82	1.02
0.07	0.10	0.17	0.27	0.35	0.41	0.56	0.79	1.05	1.31
0.10	0.03	0.21	0.35	0.49	0.61	0.79	1.05	1.35	1.70
0.17	0.20	0.30	0.46	0.64	0.82	1.05	1.35	1.73	2.18
0.26	0.33	0.46	0.63	0.82	1.05	1.35	1.74	2.22	2.81
0.34	0.44	0.58	0.79	1.05	1.35	1.74	2.22	2.84	3.61
0.26	0.31	0.40	0.97	1.32	1.73	2.22	2.84	3.63	4.63
0.20	0.21	0.11	1.20	1.67	2.18	2.81	3.61	4.63	4.70

Figure 4: Optimal state value functions based on reward function 1

Our algorithm converges in 22 steps, in order to have 5 snapshots of state values linearly distributed from 1 to 22, we choose to take snapshots at step 4, 8, 12, 16, and 20. The results are shown together as Figure 5. From the plots, we first can see that all the state values are increasing and finally converge at some point. And also, we can see that states near the highest reward have relatively larger values compare to the states that are far away from the highest reward, which clearly make sense. Moreover, we can see that area that are far away from the highest reward do not get their values updated immediately, i.e. the further away from the highest reward the later their values will get updated.

Question 3

Now that we have the optimal state values across the 2-D grid, we are going to plot the heatmap for them. The resulting plot is shown as Figure 6.

Question 4

From the above heat map, considering the concept of the expected discounted cumulative reward for the agent in a given state, we can see that states in the bottom right have the largest values, because the highest reward is in that region. The value decreases when the number of steps to each that state increases. For example, the neighbors of the highest reward have the highest values among all the states, because it only takes one step to reach the highest reward, while states in the top left have the lowest values because it takes so many steps to reach the highest reward. Another thing to be noticed is that there are three regions outside the top left corner also have the lowest value, that is because they have the lowest rewards according to the Reward Function 1, and the neighbors of these three regions will of course have low values, but notice that some of them are close to the highest reward, thus they will have slightly higher values compare to the neighbors that are further from the highest reward, which clearly make sense.

Question 5

In this question, we are going to complete the algorithm by finishing the Computation part, i.e. we will compute the optimal policy of the agent. Detailed discussion on how to do this can be found in the project handout, thus we will not discuss it in this part. The resulting plot can refer to Figure 7.

Question 6

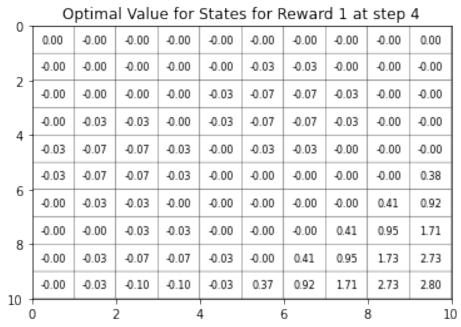
Now we will use the Reward 2 instead of Reward 1, and first generate the optimal state values. The resulting plot is shown as Figure 8.

Question 7

Now that we have the optimal state values across the 2-D grid, we are going to plot the heat map for them. The resulting plot is shown as Figure 9.

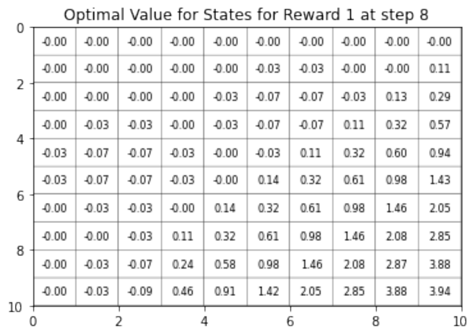
Following the same logic as in Question 4, we see that the highest value is in the bottom right corner and the lowest values are in the middle right with a 'snake' pattern, which is consistent with the values in Reward Function 2. Notice that in the 'snake' pattern, although the rewards are -100 in that region, none of the states actually have value of -100, this is because the agent can escape the -100 reward in one step, thus when combining we will not have a state with value of -100. There is one block with the darkest color, i.e. the lowest value. This because when the agent is in this state, there are three neighbors with value of -100

Question 7 continued on next page…


Optimal Value for States for Reward 1 at step 4

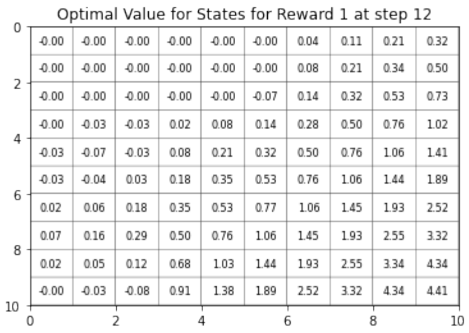
0.00	-0.00	-0.00	-0.00	-0.00	-0.00	-0.00	-0.00	-0.00	0.00
-0.00	-0.00	-0.00	-0.00	-0.00	-0.03	-0.03	-0.00	-0.00	-0.00
-0.00	-0.00	-0.00	-0.00	-0.03	-0.07	-0.07	-0.03	-0.00	-0.00
-0.00	-0.03	-0.03	-0.00	-0.03	-0.07	-0.07	-0.03	-0.00	-0.00
-0.03	-0.07	-0.07	-0.03	-0.00	-0.03	-0.03	-0.00	-0.00	-0.00
-0.03	-0.07	-0.07	-0.03	-0.00	-0.00	-0.00	-0.00	-0.00	0.38
-0.00	-0.03	-0.03	-0.00	-0.00	-0.00	-0.00	-0.00	0.41	0.92
-0.00	-0.00	-0.03	-0.03	-0.00	-0.00	-0.00	0.41	0.95	1.71
-0.00	-0.03	-0.07	-0.07	-0.03	-0.00	0.41	0.95	1.73	2.73
-0.00	-0.03	-0.10	-0.10	-0.03	0.37	0.92	1.71	2.73	2.80

(a) step 4


Optimal Value for States for Reward 1 at step 8

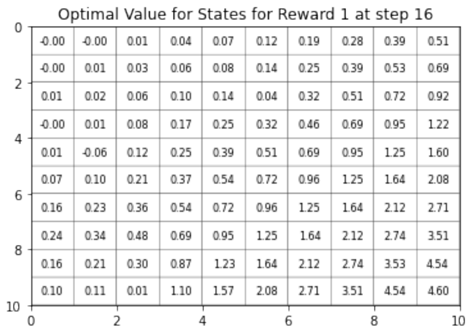
-0.00	-0.00	-0.00	-0.00	-0.00	-0.00	-0.00	-0.00	-0.00	-0.00
-0.00	-0.00	-0.00	-0.00	-0.00	-0.03	-0.03	-0.00	-0.00	0.11
-0.00	-0.00	-0.00	-0.00	-0.03	-0.07	-0.07	-0.03	0.13	0.29
-0.00	-0.03	-0.03	-0.00	-0.03	-0.07	-0.07	0.11	0.32	0.57
-0.03	-0.07	-0.07	-0.03	-0.00	-0.03	0.11	0.32	0.60	0.94
-0.03	-0.07	-0.07	-0.03	-0.00	0.14	0.32	0.61	0.98	1.43
-0.00	-0.03	-0.03	-0.00	0.14	0.32	0.61	0.98	1.46	2.05
-0.00	-0.00	-0.03	0.11	0.32	0.61	0.98	1.46	2.08	2.85
-0.00	-0.03	-0.07	0.24	0.58	0.98	1.46	2.08	2.87	3.88
-0.00	-0.03	-0.09	0.46	0.91	1.42	2.05	2.85	3.88	3.94

(b) step 8


Optimal Value for States for Reward 1 at step 12

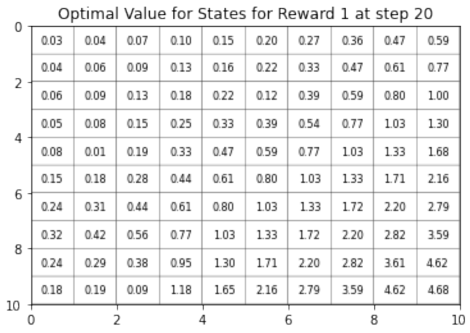
-0.00	-0.00	-0.00	-0.00	-0.00	-0.00	0.04	0.11	0.21	0.32
-0.00	-0.00	-0.00	-0.00	-0.00	-0.00	0.08	0.21	0.34	0.50
-0.00	-0.00	-0.00	-0.00	-0.00	-0.07	0.14	0.32	0.53	0.73
-0.00	-0.03	-0.03	0.02	0.08	0.14	0.28	0.50	0.76	1.02
-0.03	-0.07	-0.03	0.08	0.21	0.32	0.50	0.76	1.06	1.41
-0.03	-0.04	0.03	0.18	0.35	0.53	0.76	1.06	1.44	1.89
0.02	0.06	0.18	0.35	0.53	0.77	1.06	1.45	1.93	2.52
0.07	0.16	0.29	0.50	0.76	1.06	1.45	1.93	2.55	3.32
0.02	0.05	0.12	0.68	1.03	1.44	1.93	2.55	3.34	4.34
-0.00	-0.03	-0.08	0.91	1.38	1.89	2.52	3.32	4.34	4.41

(c) step 12


Optimal Value for States for Reward 1 at step 16

-0.00	-0.00	0.01	0.04	0.07	0.12	0.19	0.28	0.39	0.51
-0.00	0.01	0.03	0.06	0.08	0.14	0.25	0.39	0.53	0.69
0.01	0.02	0.06	0.10	0.14	0.04	0.32	0.51	0.72	0.92
-0.00	0.01	0.08	0.17	0.25	0.32	0.46	0.69	0.95	1.22
0.01	-0.06	0.12	0.25	0.39	0.51	0.69	0.95	1.25	1.60
0.07	0.10	0.21	0.37	0.54	0.72	0.96	1.25	1.64	2.08
0.16	0.23	0.36	0.54	0.72	0.96	1.25	1.64	2.12	2.71
0.24	0.34	0.48	0.69	0.95	1.25	1.64	2.12	2.74	3.51
0.16	0.21	0.30	0.87	1.23	1.64	2.12	2.74	3.53	4.54
0.10	0.11	0.01	1.10	1.57	2.08	2.71	3.51	4.54	4.60

(d) step 16


Optimal Value for States for Reward 1 at step 20

0.03	0.04	0.07	0.10	0.15	0.20	0.27	0.36	0.47	0.59
0.04	0.06	0.09	0.13	0.16	0.22	0.33	0.47	0.61	0.77
0.06	0.09	0.13	0.18	0.22	0.12	0.39	0.59	0.80	1.00
0.05	0.08	0.15	0.25	0.33	0.39	0.54	0.77	1.03	1.30
0.08	0.01	0.19	0.33	0.47	0.59	0.77	1.03	1.33	1.68
0.15	0.18	0.28	0.44	0.61	0.80	1.03	1.33	1.71	2.16
0.24	0.31	0.44	0.61	0.80	1.03	1.33	1.72	2.20	2.79
0.32	0.42	0.56	0.77	1.03	1.33	1.72	2.20	2.82	3.59
0.24	0.29	0.38	0.95	1.30	1.71	2.20	2.82	3.61	4.62
0.18	0.19	0.09	1.18	1.65	2.16	2.79	3.59	4.62	4.68

(e) step 20

Figure 5: state value functions during iterations

Question 7 continued on next page…

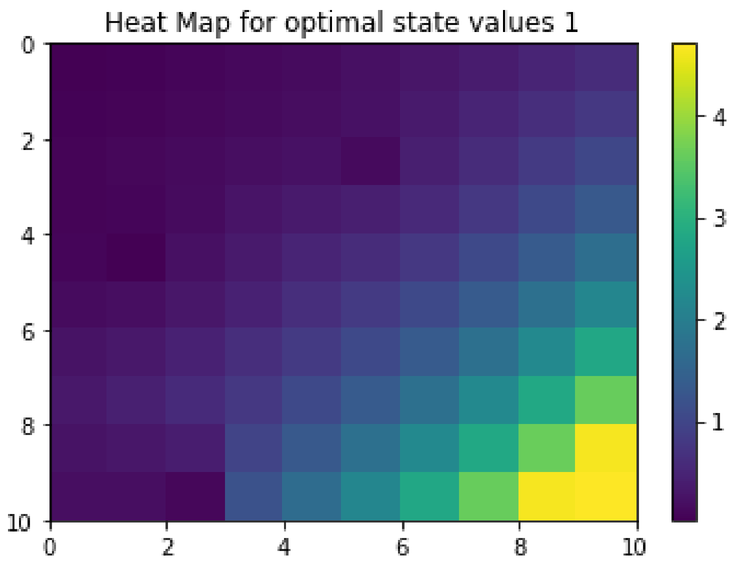


Figure 6: Heat map for optimal state value function based on reward 1

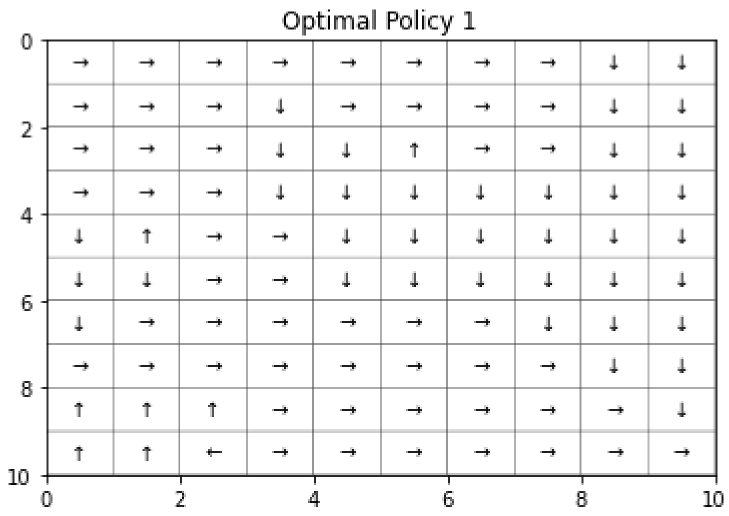
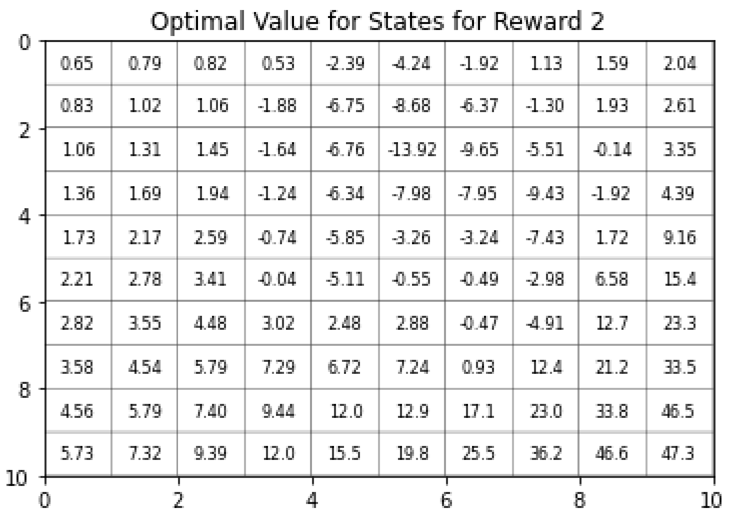


Figure 7: Optimal policy based on state value function derived in Question 2

Question 7 continued on next page. . .


Optimal Value for States for Reward 2
0
2
4
6
8
10

0.65	0.79	0.82	0.53	-2.39	-4.24	-1.92	1.13	1.59	2.04
0.83	1.02	1.06	-1.88	-6.75	-8.68	-6.37	-1.30	1.93	2.61
1.06	1.31	1.45	-1.64	-6.76	-13.92	-9.65	-5.51	-0.14	3.35
1.36	1.69	1.94	-1.24	-6.34	-7.98	-7.95	-9.43	-1.92	4.39
1.73	2.17	2.59	-0.74	-5.85	-3.26	-3.24	-7.43	1.72	9.16
2.21	2.78	3.41	-0.04	-5.11	-0.55	-0.49	-2.98	6.58	15.4
2.82	3.55	4.48	3.02	2.48	2.88	-0.47	-4.91	12.7	23.3
3.58	4.54	5.79	7.29	6.72	7.24	0.93	12.4	21.2	33.5
4.56	5.79	7.40	9.44	12.0	12.9	17.1	23.0	33.8	46.5
5.73	7.32	9.39	12.0	15.5	19.8	25.5	36.2	46.6	47.3

Figure 8: Optimal state value functions based on reward function 2

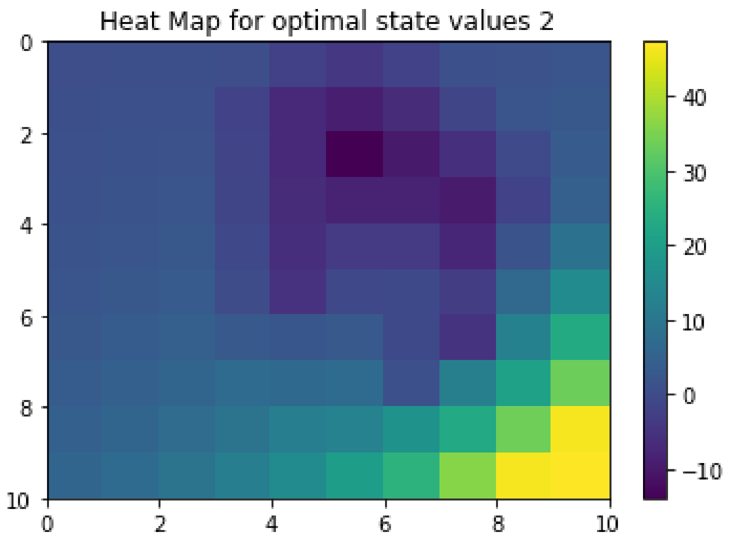


Figure 9: Heat map for optimal state value function based on reward 2

Question 7 continued on next page. . .

and only one neighbor with value of 0, thus of this state, there is a large probability that the agent will go to a -100 state, thus having the lowest value, which clearly make sense.

Question 8

Now, we compute the optimal policy based on second reward function and the corresponding result is shown as Figure 10. Following the same logic as in Question 5, the arrows match our intuition. We can see that

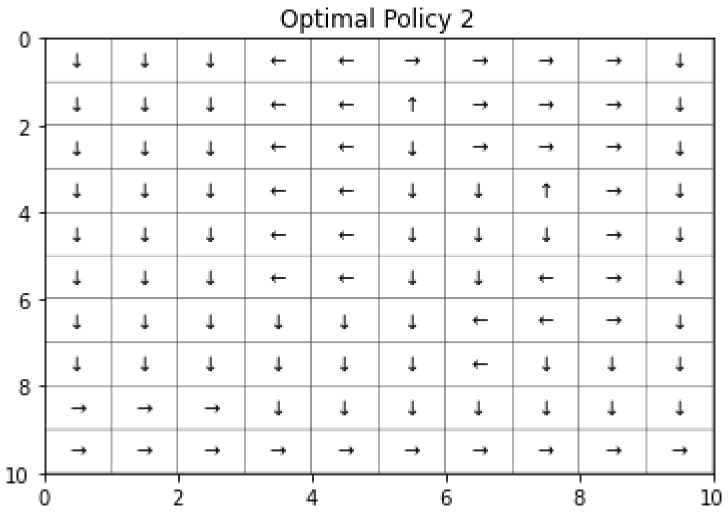


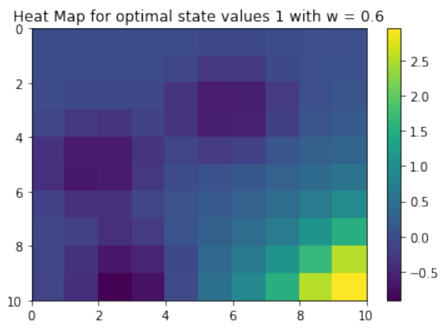
Figure 10: Optimal policy of states of the 2-D grid

the arrows are leading the agent to go to the bottom right, because bottom right has the highest value, and inside the 'snake' pattern, the arrows are trying to escape from this region as quickly as possible, which clearly make sense.

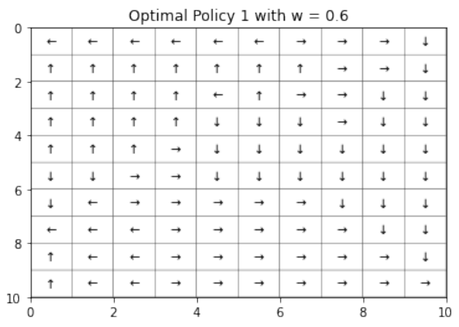
Question 9

In this part, we will change w to be 0.6 and find the optimal policy map similar to the previous questions using Reward 1 and Reward 2. For completeness and for the aid of comparison, we also plot the heatmaps for the optimal state values. The resulting plots are shown below. Specifically, Figure 11 shows the heat map and optimal policy based on reward function 1 with $w = 0.6$; Figure 12 shows the heat map and optimal policy based on reward function 2 with $w = 0.6$.

First, we must be aware of the fact that when $w = 0.6$, the agent will go in the intended direction with a probability of 0.4 and move to a random direction with a probability of 0.6. And we can see from both optimal policies, that the agent now tends to get stuck in the local optimal area. For example, when using Reward 2, we see that now in the top and bottom left corners, the optimal state values are relatively high compare to the 'snake' pattern, but are still not as high as the bottom right corner, however, the agent when following the arrows will get stuck in these areas. And we can see that only in the areas that are close to the bottom right corner will the arrows lead the agent to the true, global optimal. Thus, we think that $w = 0.6$ is not a good choice compare to $w = 0.1$.

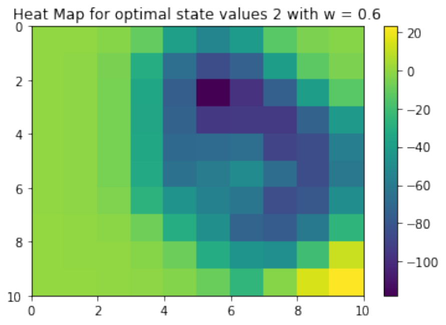


(a) Heat map for optimal values 1 with $w = 0.6$

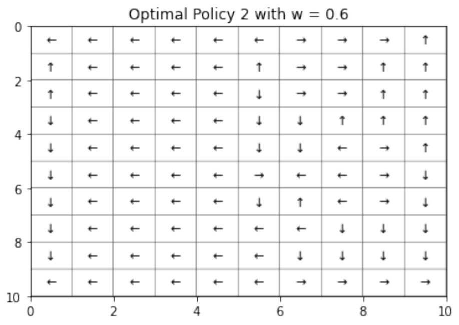


(b) Optimal Policy for Reward 1 with $w = 0.6$

Figure 11: Heat map and optimal policy for reward function 1 based on $w = 0.6$



(a) Heat map for optimal values 2 with $w = 0.6$



(b) Optimal Policy for Reward 2 with $w = 0.6$

Figure 12: Heat map and optimal policy for reward function 1 based on $w = 0.6$

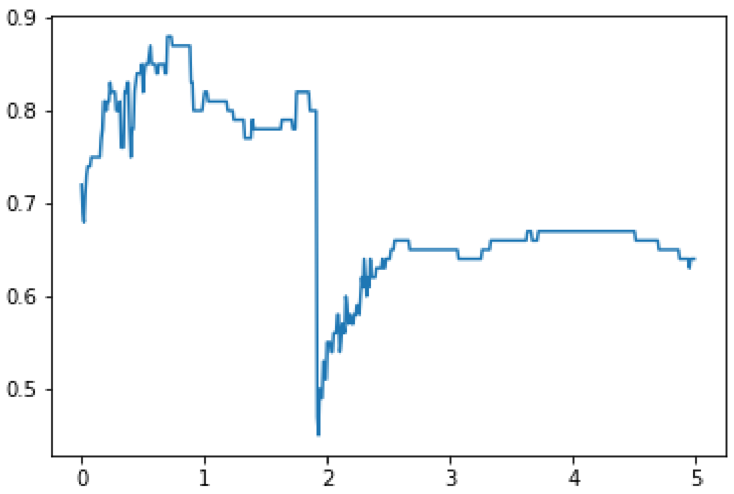


Figure 13: The trend of IRL algorithm accuracy against penalty coffcient λ

Question 10

To solve the optimization problem given by linear programming, we can reformulate the equation as follows:

$$\underset{\mathbf{x}}{\text{maximize}}\, \mathbf{c}^T\mathbf{x} \tag{10.1}$$

$$\text{subject to } \mathbf{D}\mathbf{x} \preceq \mathbf{b}, \forall a \in \mathcal{A}\backslash a_1 \tag{10.2}$$

where $\mathbf{c} = \begin{bmatrix} \mathbf{1}_{|\mathcal{S}|\times 1} \\ -\lambda\mathbf{1}_{|\mathcal{S}|\times 1} \\ \mathbf{0}_{|\mathcal{S}|\times 1} \end{bmatrix}$, $\mathbf{x} = \begin{bmatrix} \mathbf{t} \\ \mathbf{u} \\ \mathbf{R} \end{bmatrix}$,$\mathbf{D} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & -(\mathbf{P}_{a_1} - \mathbf{P}_a)(\mathbf{I} - \gamma P_{a_1})^{-1} \\ \mathbf{0} & \mathbf{0} & -(\mathbf{P}_{a_1} - \mathbf{P}_a)(\mathbf{I} - \gamma P_{a_1})^{-1} \\ \mathbf{0} & -\mathbf{I} & -\mathbf{I} \\ \mathbf{0} & -\mathbf{I} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & -\mathbf{I} \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} \mathbf{0}_{|\mathcal{S}|\times 1} \\ \mathbf{0}_{|\mathcal{S}|\times 1} \\ \mathbf{0}_{|\mathcal{S}|\times 1} \\ \mathbf{0}_{|\mathcal{S}|\times 1} \\ R_{max}\cdot\mathbf{1}_{|\mathcal{S}|\times 1} \\ R_{max}\cdot\mathbf{1}_{|\mathcal{S}|\times 1} \end{bmatrix}$

Question 11

According to the result of Question 10, we can solve this optimization problem by linear programming given $\mathbf{c}, \mathbf{D}$ and $\mathbf{b}$. In order to satisfy the inequality for all $a \in \mathcal{A}\backslash a_1$, $\mathbf{D}$ should include all the matrix generated by possible non-optimal actions. Hence, the size of the matrix, $\mathbf{D}$ is set as $(2k+2)|\mathcal{S}| \times 3|\mathcal{S}|$, where k is the cardinality of the set of possible actions, $\mathcal{A}$. The shape of other elements in this linear programming can be derived correspondingly. In order to simplify the following questions, we decide to define two classes,*Maze* and *IRL*. The details of these classes can be found in our handed code, and will not be discussed here. After solving the optimization problem, we can get the optimal objective vector, $\mathbf{x}$, whose last 100 elements are our recovered reward. Repeating the optimization problem with different λ, we can finally get the accuracy of IRL algorithm against λ as Figure 13.

Question 12

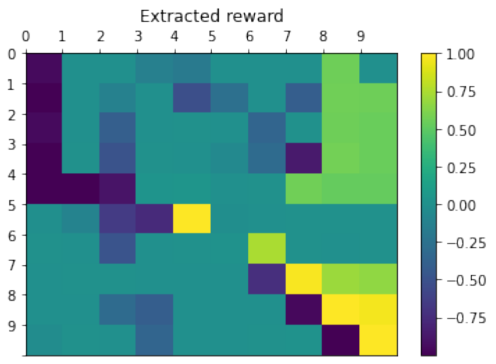
To get a evenly distributed 500 numbers between 0 and 5, we are going to use the function *np.linspace* to partition the space. Moreover, we decide to ignore the endpoint so as to get a 0.01 interval between every

Question 12 continued on next page...

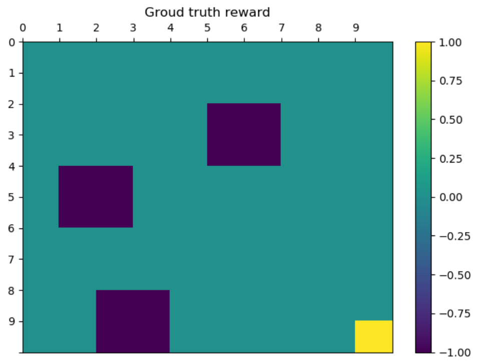
two consecutive number. According to the Figure 13, we can find that the optimal value of λ to achieve the maximum accuracy is 0.70. The following three questions are all based on this result.

Question 13

Now that we derive the optimal penalty coefficient, λ_{max}, we can extracted the recovered reward function from the vector $\mathbf{x}$ and reshape it to a 10×10 grid shown as Figure 14a while the heat map of original reward is shown as Figure 14b.



(a) Recovered Reward



(b) Ground truth Reward

Figure 14: Heat map for ground truth and recovered reward function

Question 14

Using the same method, value-iteration algorithm, we can easily derive the optimal state value function. To make the result more straight-forward, the state value function is shown through heat map as Figure 15.

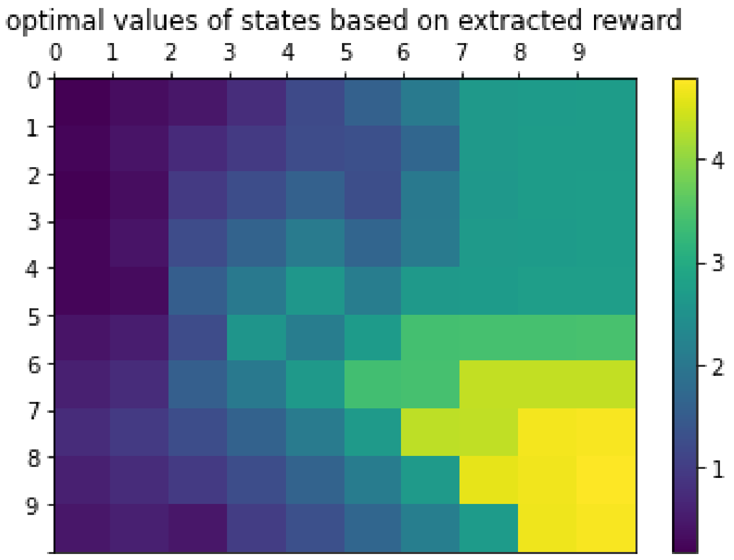


Figure 15: The heat map of optimal state value function based on the recovered reward function 1

Question 14 continued on next page. . .

Question 15

The heat map of optimal state value function based on original reward is shown above as Figure 6, which indeed is very similar to Figure 15. First of all, the maximum value function and the minimum value function are obtained at the same place; Secondly, the trend that the closer to the right-lower corner, the larger the value function is can be observed from both figures. However, there still be some difference between these two figures. Notice that in the ground truth optimal value function, there exist some locla minimum, which are located at where the reward function is -1. Nevertheless, in the recovered optimal value function, this phenomenon is hard to observe. In other words, the recovered optimal function is more smooth than the original one, which totally makes sense. Moreover, since the recovered reward is smoothed, there exist more positive reward than the ground truth, making the whole optimal value function is brighter, namely larger than the original one.

Question 16

Using same algorithm mentioned in the Question 5, we can derive the optimal policy based on the recovered reward function. The resulting image is shown as Figure 16.

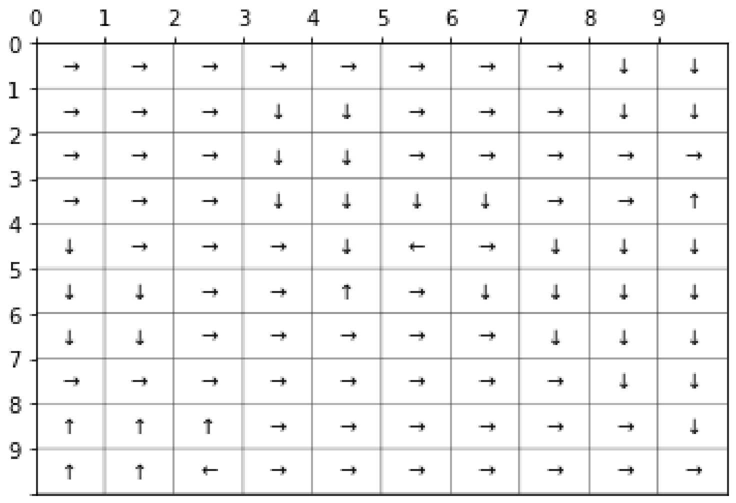


Figure 16: The optimal policy for 2D-grid with 100 states based on recovered reward function 1.

Question 17

Notice that the accuracy with optimal penalty coefficient λ_{max} is relatively high, achieving nearly 0.90. In other words, there are about 10 differences between the ground truth optimal policy between recovered professional policy. Inspecting these difference, it is easy to figure out they all happen at the place where the shape reward shift is smoothed. Taking state $[3, 9]$ as example, it is taught to move up to achieve optimal value, which is reasonable for recovered reward function since the value of reward is positive at it's future state. However, the ground truth reward have positive reward merely at the right-lower corner. That is to say, states have to follow the direction making them closer to the corner to get optimal value function. In all, the smoothed reward function, which makes the reward of some states positive, misleads some states to deviate the direction getting close to the right-lower corner, resulting in discrepancies.

Question 18

The only change we need to do here is to change the reward function to reward 2. And the corresponding figure is shown as Figure 17.

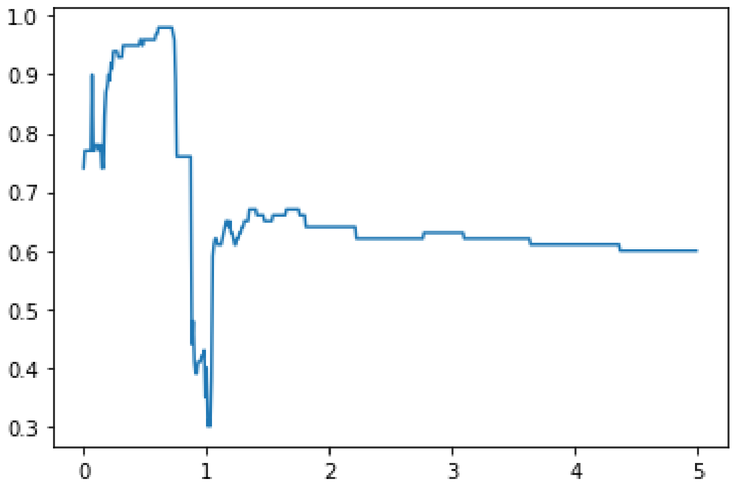


Figure 17: The trend of IRL algorithm accuracy against penalty coffcient λ adopting optimal policy derived by Reward 2.

Question 19

From the Figure, we can find the λ that maximize accuracy shown as follows:

$$\lambda_{max}^{(2)} = 0.61; Accuracy_{max} = 0.98 \tag{19.1}$$

Question 20

We plot the figure of ground truth reward and extracted reward as Figure 18.

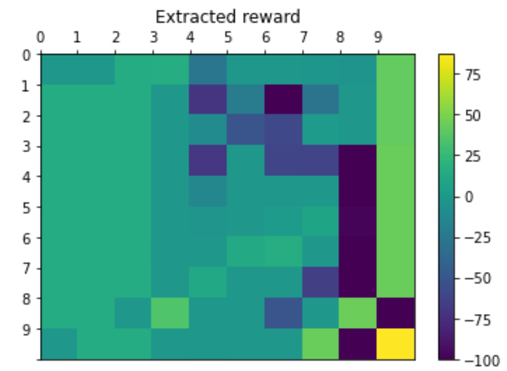
Question 21

We plot the figure of heat map of the optimal value of states as Figure 19.

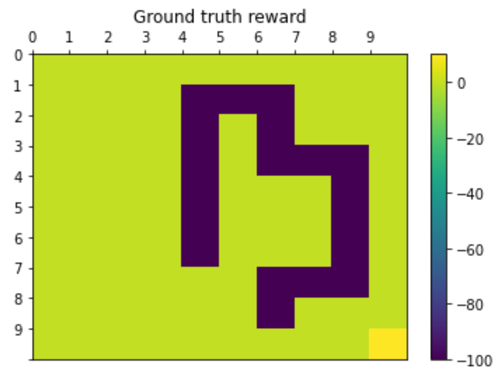
Question 22

By comparing the theoretical and actual best state value heat maps, we can find that most of them are similar: they have a higher reward value in the lower right corner (maze exit) and a lower value in the upper left corner (maze entrance) , And has the lowest value in the penalty area. This is easy to understand because the only positive reward is located in the lower right corner, while the upper left corner is far from the reward.

But we also found that there is a difference between the two. Specifically, the value range is about -100 to 480, which is much larger than that in question 7. In addition, the boundary of the penalty area is not quite obvious, which may come from the extracted rewards.



(a) Recovered Reward



(b) Ground truth Reward

Figure 18: Heat map for ground truth and recovered reward function 2.

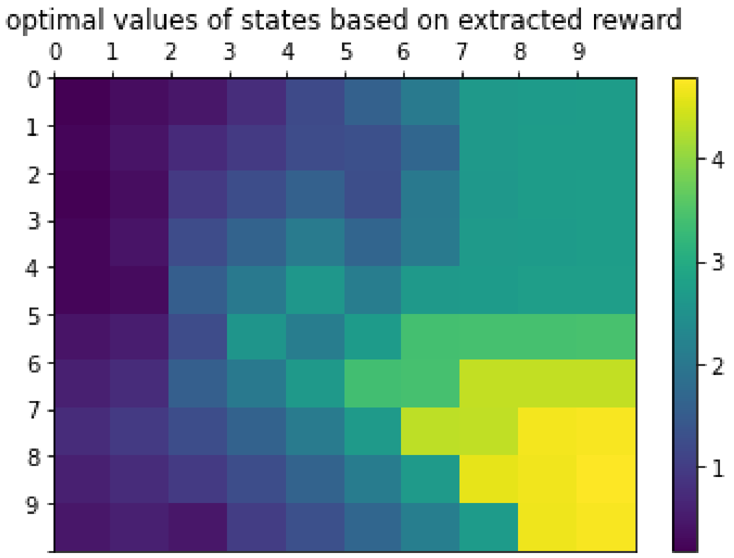


Figure 19: The heat map of optimal state value function based on the recovered reward function 2

Question 23

The optimal policy corresponding to the optimal value function can refer to Figure 20.

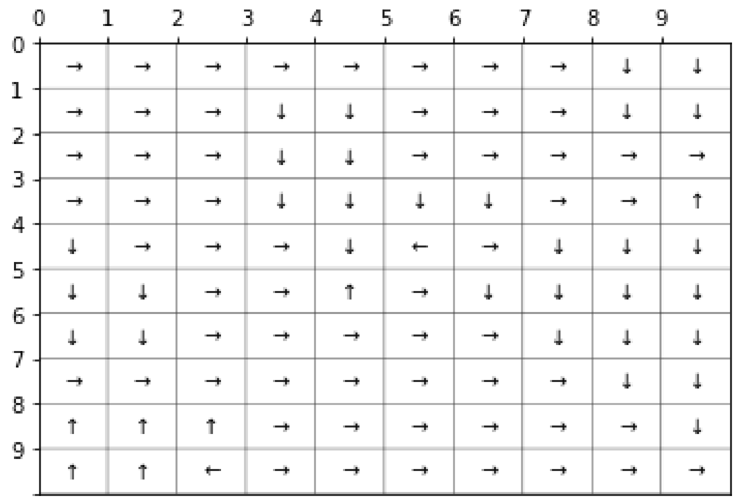
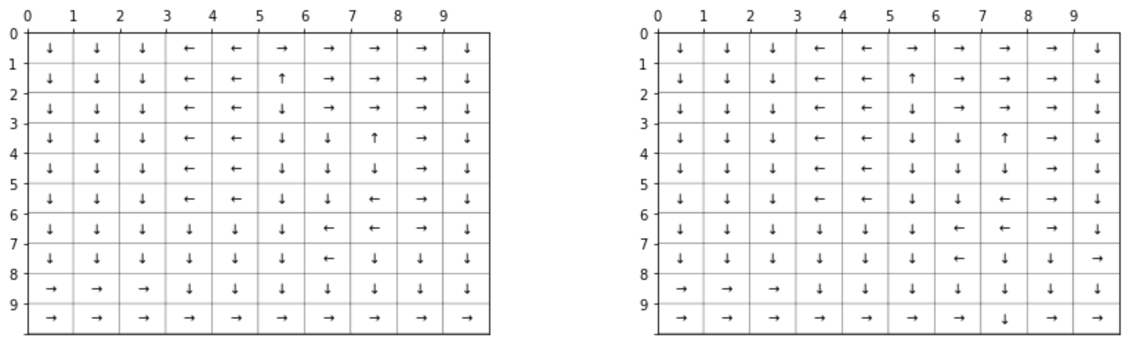


Figure 20: The optimal policy for 2D-grid with 100 states based on recovered reward function 2.

Question 24

To evaluate the performance of IRL method, we align the original optimal policy and the one derived according to the recovered reward together as Figure 21. There are two types of errors. The first error is



(a) Original optimal policy (b) Optimal policy derived according to recovered reward

Figure 21: Heat map for ground truth and recovered reward function 2.

located at (9,7), and the second is at (7,9). These errors come from the extracted reward matrix, which has a negative reward beside the exit of the maze. We will discuss in Question 25.

Question 25

From question 24, we can find two main errors. Inspecting these two errors, we can find that all these discrepancies result from the twisted recovered reward. In the original reward, the negative reward exists only at the 'snake' shape. However, for recovered reward, in order to maximize the the sum of the differences

Question 25 continued on next page. . .

between the quality of the optimal action and the quality of the next best action, some reward is set as negative number. The only two discrepancies both results from the intuition to avoid these negative rewards. According to the problem set, it is said that we can slightly modify the value-iteration function to eliminate one of the discrepancy. One direct consideration is to modify the parameter ϵ so as to make the iteration result is closer to the true optimal value function. Checking the intermediate variables generated through the algorithm carefully, we find out that when $\epsilon = 0.01$, the default value mentioned in Question 2, it is not enough for the algorithm to converge to the true optimal value function. That is to say, the optimal value function derived will somewhat change if ϵ decreases and the optimal policy will then change to a more reasonable action correspondingly. Moreover, we plot the trend of accuracy against the λ with smaller ϵ as Figure 22. From this figure, we can conclude that our modification do fix one of the discrepancy and achieve 0.99 accuracy.

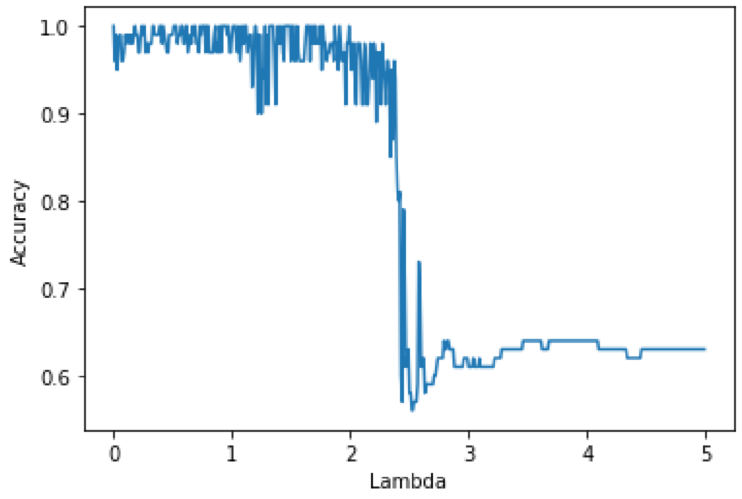


Figure 22: The trend of IRL algorithm accuracy against penalty coffcient λ with smaller ϵ adopted.