

Project 3: Reinforcement learning and Inverse Reinforcement learning

1. Introduction

In this project, we used python to learn the optimal policy of an agent navigating in a 2-D environment. We implemented the Value iteration algorithm to learn the optimal policy. Also, we explored the application of IRL (Inverse reinforcement learning) in the context of apprenticeship learning.

2. Reinforcement Learning (RL)

Question 1

In this question, we generate heat maps of Reward function1 and Reward function 2. For the heat maps, we display the coloring scale.

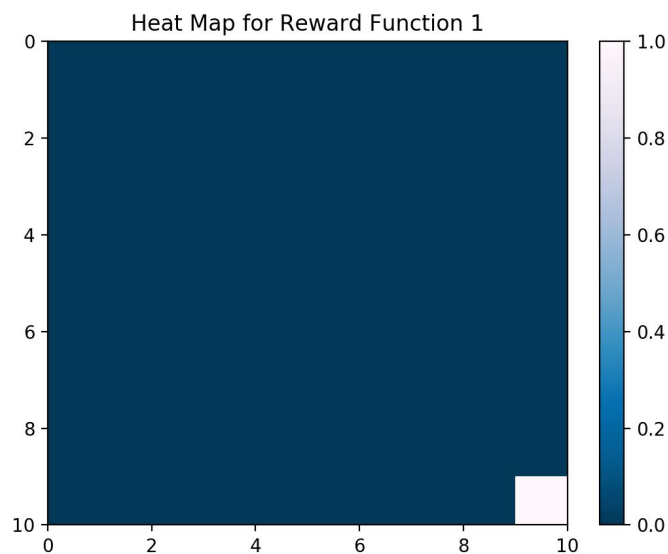


Figure 1. Heatmap for reward function 1

We can see that heatmap for reward function 1 is symmetric according to the diagonal line.

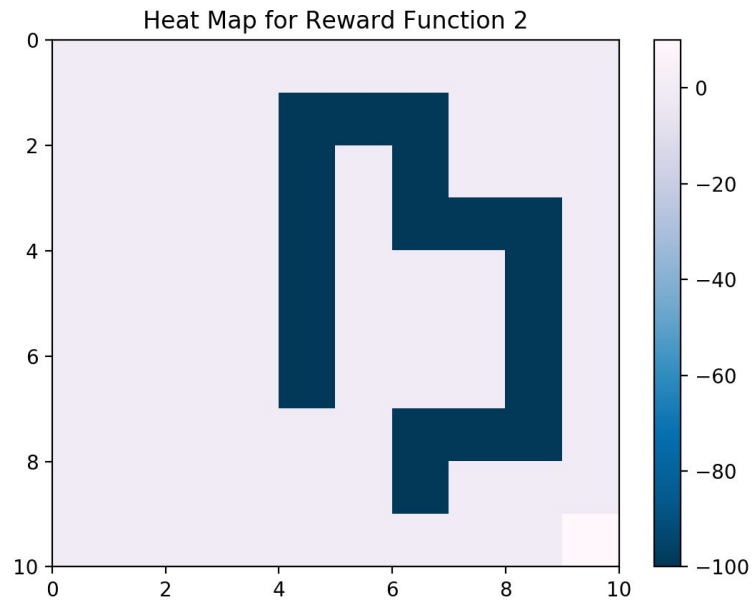


Figure 2. Heatmap for reward function 2

3. Optimal Policy Learning Using RL Algorithms

Question 2

We create the agent with given parameter, number of states = 100, number of actions = 4, $w = 0.1$, discount factor $\gamma = 0.8$, reward function = reward function 1, and threshold = 0.01. Then we 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, we implemented the Initialization and Estimation steps of the Value Iteration algorithm. For the estimation step, use $\epsilon = 0.01$. For visualization purpose, we generated the graph shown in Figure 3.

0.044	0.065	0.091	0.125	0.168	0.223	0.292	0.38	0.491	0.61
0.065	0.088	0.122	0.165	0.219	0.289	0.378	0.491	0.633	0.788
0.091	0.122	0.165	0.219	0.289	0.378	0.491	0.636	0.818	1.019
0.125	0.165	0.219	0.289	0.378	0.491	0.636	0.82	1.052	1.315
0.168	0.219	0.289	0.378	0.491	0.636	0.82	1.054	1.352	1.695
0.223	0.289	0.378	0.491	0.636	0.82	1.054	1.353	1.733	2.182
0.292	0.378	0.491	0.636	0.82	1.054	1.354	1.735	2.22	2.807
0.38	0.491	0.636	0.82	1.054	1.353	1.735	2.22	2.839	3.608
0.491	0.633	0.818	1.052	1.352	1.733	2.22	2.839	3.629	4.635
0.61	0.788	1.019	1.315	1.695	2.182	2.807	3.608	4.635	4.702

Figure 3. Optimal values of states using reward function 1

Question 3

We generated a heat map of the optimal state values across the 2-D grid for result of question 2. The heat map is shown in Figure 4.

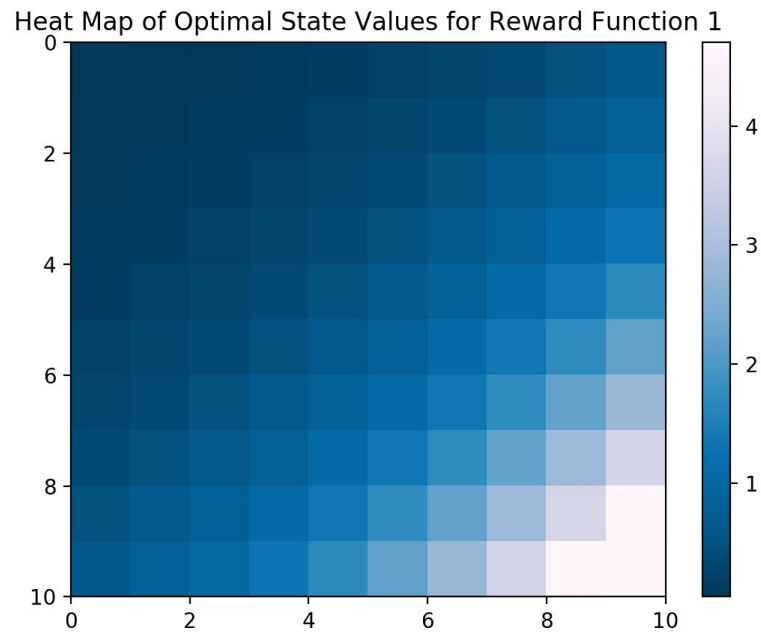


Figure 4. Heat Map of Question 3

Question 4

The distribution of the optimal state values across the 2-D grid indicates that when the state is away from the reward state (number of state is 99), then its state value will be smaller. This distribution looks like a Gaussian distribution with the center at the bottom-right corner. This is because each state value is influenced by its surrounding four neighbors and itself and the discount factor, which is 0.8. The discount factor causes the decay from one state to its neighbor states. Therefore, if a state is farther to the reward, then it will take more steps for the agent to get to the state, and then more decay happens. The distribution follows the rule that the values on the top-left is always smaller than those on the bottom-right.

Question 5

We implemented the computation step of the value iteration algorithm to compute the optimal policy of the agent navigating the 2-D space. We generated a figure with the optimal action at that state. The figure is shown in Figure 5.

↓	→	→	→	→	→	→	→	↓	↓
↓	→	→	→	→	→	↓	↓	↓	↓
↓	↓	↓	→	→	↓	↓	↓	↓	↓
↓	↓	↓	→	↓	↓	↓	↓	↓	↓
↓	↓	↓	→	↓	↓	↓	↓	↓	↓
↓	↓	→	→	→	→	↓	↓	↓	↓
↓	→	→	→	→	→	→	↓	↓	↓
↓	→	→	→	→	→	→	→	↓	↓
→	→	→	→	→	→	→	→	→	↓
→	→	→	→	→	→	→	→	→	→

Figure 5 Optimal Action at Each State

We find that this optimal policy of the agent does match our intuition. The agent goes to reward state (bottom-right) as quickly as possible by going down or right, which is the optimal policy. According to figure 5, we can find that upper state will go down and left one will go right. The last state has the highest optimal value, and other states will try to reach it and they will try to go to the neighbors with higher optimal values.

Based on what we have discussed above, we conclude that It is possible for the agent to compute the optimal action to take at each state by observing the optimal values of its neighboring states. The agent can only move to its neighbors and it will try to reach the state with higher values and avoid those with lower values.

Question 6

Now we chose to use reward function 2 and remain others the same with what we were given in question 2. Figure 6 shows the result.

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

Figure 6. Optimal values of states using reward function 2

Question 7

We generated a heat map of the optimal state values (found in question 6) across the 2-D grid shown in figure 7.

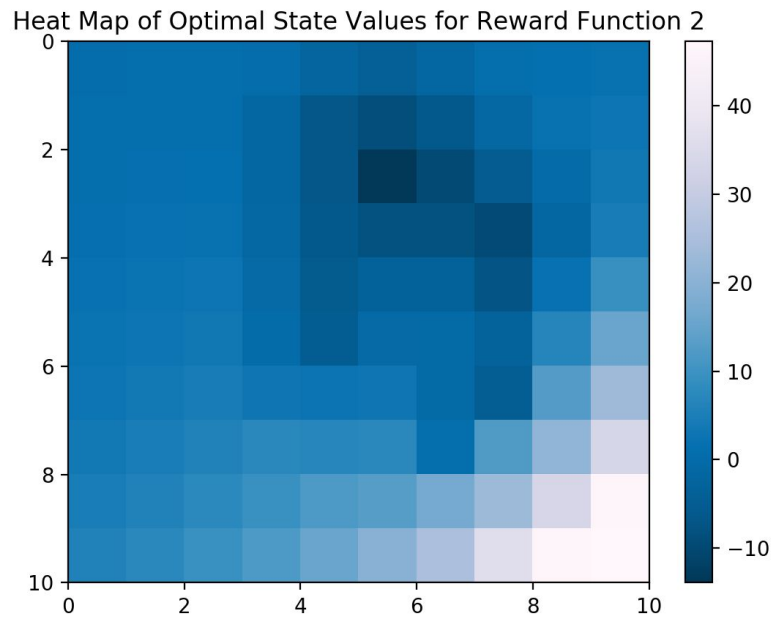


Figure 7 Heat Map of Question 6

Question 8

The distribution is somewhat similar to what we have got in question 4 as the reward function has high positive value on state 99. Also, the decay exists and the value also decrease as the distance increases at the bottom-right part. However, there are some states with negative values forming like a unclosed circle. The states with negative neighbors are highly influenced, so they tend to have lower optimal values and they are mostly negative. Therefore, the minimum value happens at state 52.

Question 9

Like what we did in question 5, we implemented the computation step of the value iteration algorithm to compute the optimal policy of the agent navigating the 2-D space. We generated a figure with the optimal action at that state. The figure is shown in Figure 8.

↓	↓	↓	←	←	→	→	→	→	↓
↓	↓	↓	←	←	↑	→	→	→	↓
↓	↓	↓	←	←	↓	→	→	→	↓
↓	↓	↓	←	←	↓	↓	↑	→	↓
↓	↓	↓	←	←	↓	↓	↓	→	↓
↓	↓	↓	←	←	↓	↓	←	→	↓
↓	↓	↓	↓	↓	↓	←	←	→	↓
↓	↓	↓	↓	↓	↓	←	↓	↓	↓
→	→	→	↓	↓	↓	↓	↓	↓	↓
→	→	→	→	→	→	→	→	→	→

Figure 8 Optimal Action at Each State for Reward Function 2

We find that this optimal policy of the agent does match our intuition. Overall, the upper one goes down and the left one goes to the right as the bottom-right corner has the largest value, except when dealing with negative value. As for the negative rewards states, they are pointing out of negative area trying to avoid the negative parts, which is consistent with our intuition. In conclusion, the overall mode is interrupted by these points within the negative area. And those in the negative area will avoid strong negative values to reach the bottom-right corner.

4. Inverse Reinforcement learning (IRL)

4.1 IRL algorithm

Question 10

Using block matrices, we can get the equivalent linear program

$$\begin{aligned} & \underset{x}{\text{maximize}} && c^T x \\ & \text{subject to} && Dx \leq 0, \forall a \in A \setminus a_1 \end{aligned}$$

Expressing c , x , D in terms of R , P_a , P_{a_1} , t_i , u , I , and R_{\max}

$$c = \begin{bmatrix} \mathbf{1}_{|\mathcal{S}| \times 1} \\ -\lambda \\ \mathbf{0}_{|\mathcal{S}| \times 1} \end{bmatrix}, \quad x = \begin{bmatrix} t \\ u \\ R \end{bmatrix}, \quad D = \begin{bmatrix} I_{|\mathcal{S}| \times |\mathcal{S}|} & \mathbf{0} & (P_a - P_{a_1})(I - \gamma P_{a_1})^{-1} \\ \mathbf{0} & \mathbf{0} & (P_a - P_{a_1})(I - \gamma P_{a_1})^{-1} \\ \mathbf{0} & -I_{|\mathcal{S}| \times |\mathcal{S}|} & I_{|\mathcal{S}| \times |\mathcal{S}|} \\ \mathbf{0} & -I_{|\mathcal{S}| \times |\mathcal{S}|} & -I_{|\mathcal{S}| \times |\mathcal{S}|} \\ \mathbf{0} & \mathbf{0} & I_{|\mathcal{S}| \times |\mathcal{S}|} \\ \mathbf{0} & \mathbf{0} & -I_{|\mathcal{S}| \times |\mathcal{S}|} \end{bmatrix}$$

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

4.2 Performance measure

Question 11

We use optimal policy of the agent found in question 5 to fill in the $O_{E(s)}$ values. Sweeping λ from 0 to 5 for 500 evenly spaced values, with Extracted Reward Function 1 to fill out $O_{A(s)}$ values, we compute the accuracy and have the plot:

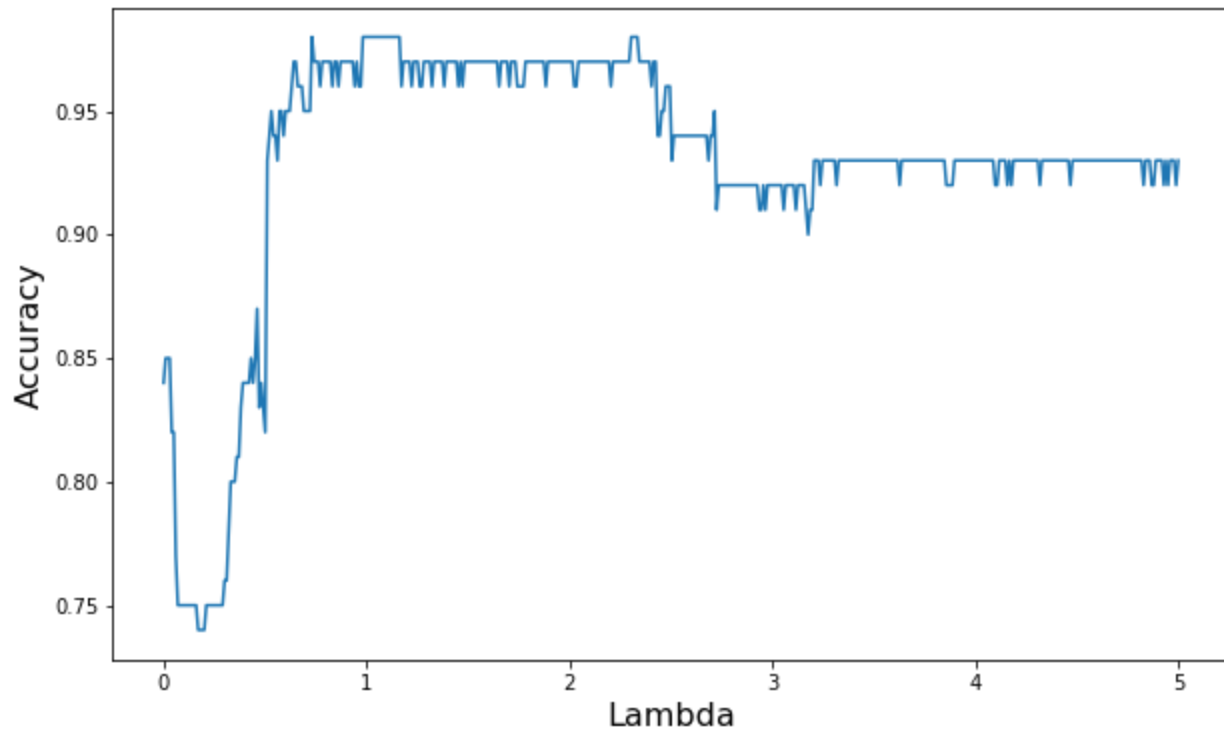


Figure 9 Penalty factor λ vs accuracy with Extracted Reward Function 1

Question 12

The results are shown in command shell:

```
Max Lambda: 2.334669
Max Accuracy: 0.980000
Accuracy: 0.980000
```

Figure 10 The value of max λ for which accuracy is maximum

Therefore, the value of $\lambda_{\max}^{(1)}$ is 2.334 and the corresponding maximum accuracy is 0.98

Question 13

Heat maps of the ground truth reward and the extracted reward are shown below:

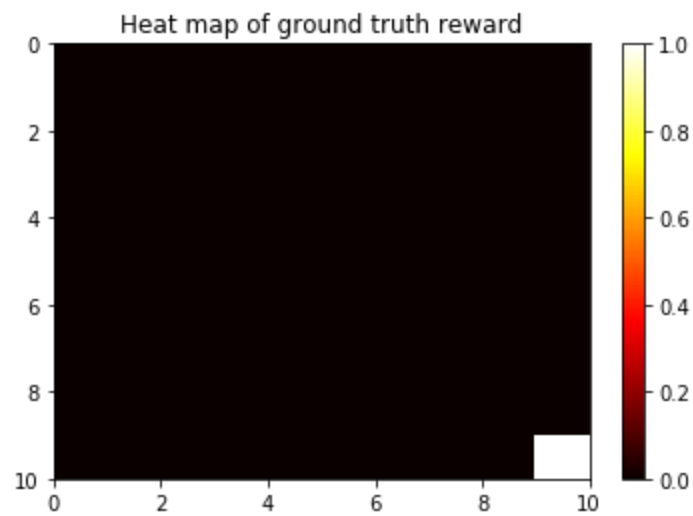


Figure 11 Heat map of the ground truth reward

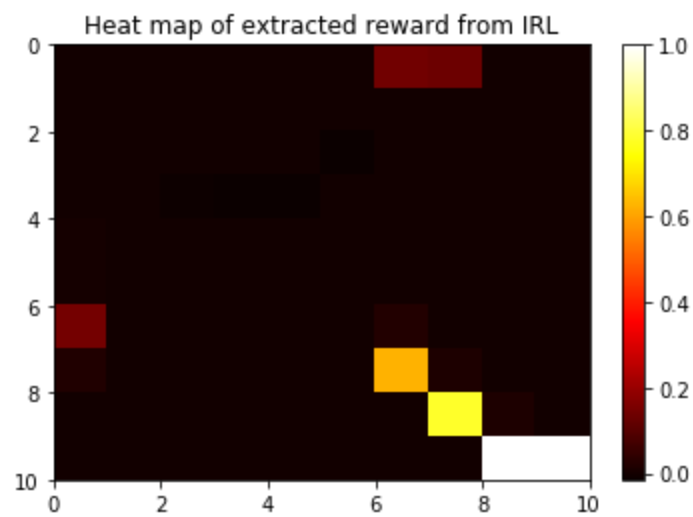


Figure 12 Heat map of the extracted reward

Question 14

We use the optimal state-value function to compute the optimal values and generate the heat map of the optimal state values across the 2-D grid:

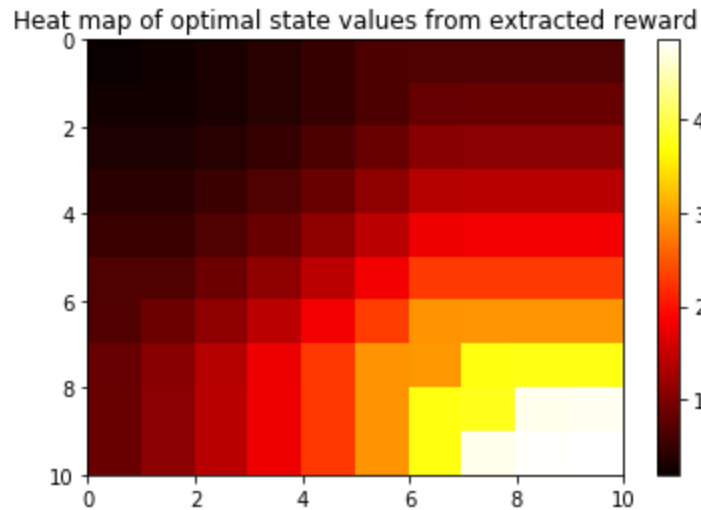


Figure 13 Heat map of the optimal state values from extracted reward function

Question 15

The heat maps look similar in Question 3 and Question 14. The trends in both heat maps are increasing from top left to the bottom right. Moreover, the state values are determined by the distance to the bottom right, and the extracted reward function is similar to the ground truth Reward Function 1, which are also similar to the optimal state values.

There exist differences in the two heat maps. We can clearly observe that the optimal state values for Reward Function 1 are more like a diagonal line from top left to bottom right, while the state values seem changed for extracted reward function. The explanation is that only one state, with highest value in Reward Function 1, has the equal effect to other states. For the extracted reward function, there are several rewards in states around the bottom right, all of which have effects on the surrounding states. Therefore, the optimal state values are not as distinctive as those in Reward Function 1 from state to state.

Question 16

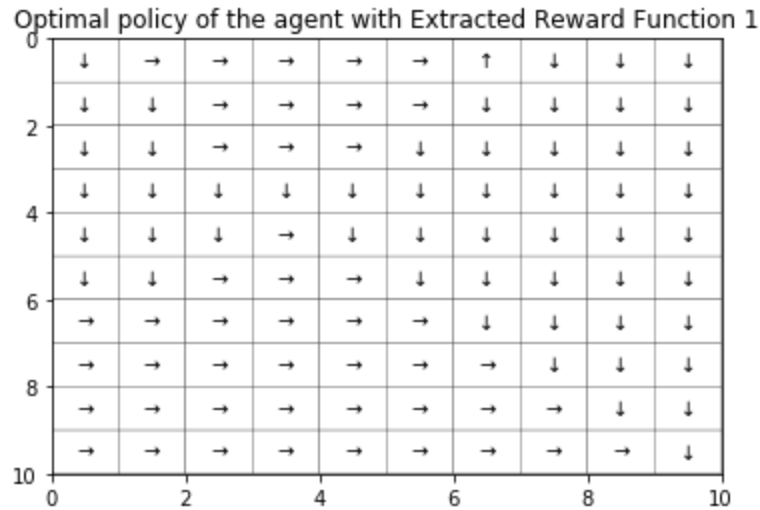


Figure 14 Optimal policy of the agent with Extracted Reward Function 1

Question 17

Table 1. Optimal policy in Question 5 and 16

Optimal policy of the agent with ground truth Reward Function 1 (Question 5)	Optimal policy of the agent with Extracted Reward Function 1 (Question 16)
	<p>Optimal policy of the agent with Extracted Reward Function 1</p>

Comparing two figures in the table above, we can see that they have similarities on showing the increasing reward from the bottom right to the top left. Furthermore, the optimal state values with the extracted reward function are close to those with ground truth reward function, thus the optimal policy is very similar.

However, several states on boundaries are different in these two figures. The reason may be caused by the extracted reward function. When the state values decrease to those states with gradual decreasing rewards, the values around them will get lower than the state values, because the boundary states would choose for a higher reward for themselves by moving out. After all, there exist fluctuations in the extracted reward function.

Question 18

We use optimal policy to of the agent found in question 5 to fill in the $O_{E(s)}$ values. Sweeping λ from 0 to 5 for 500 evenly spaced values, with Extracted Reward Function 2 to fill out $O_{A(s)}$ values, we compute the accuracy and have the plot:

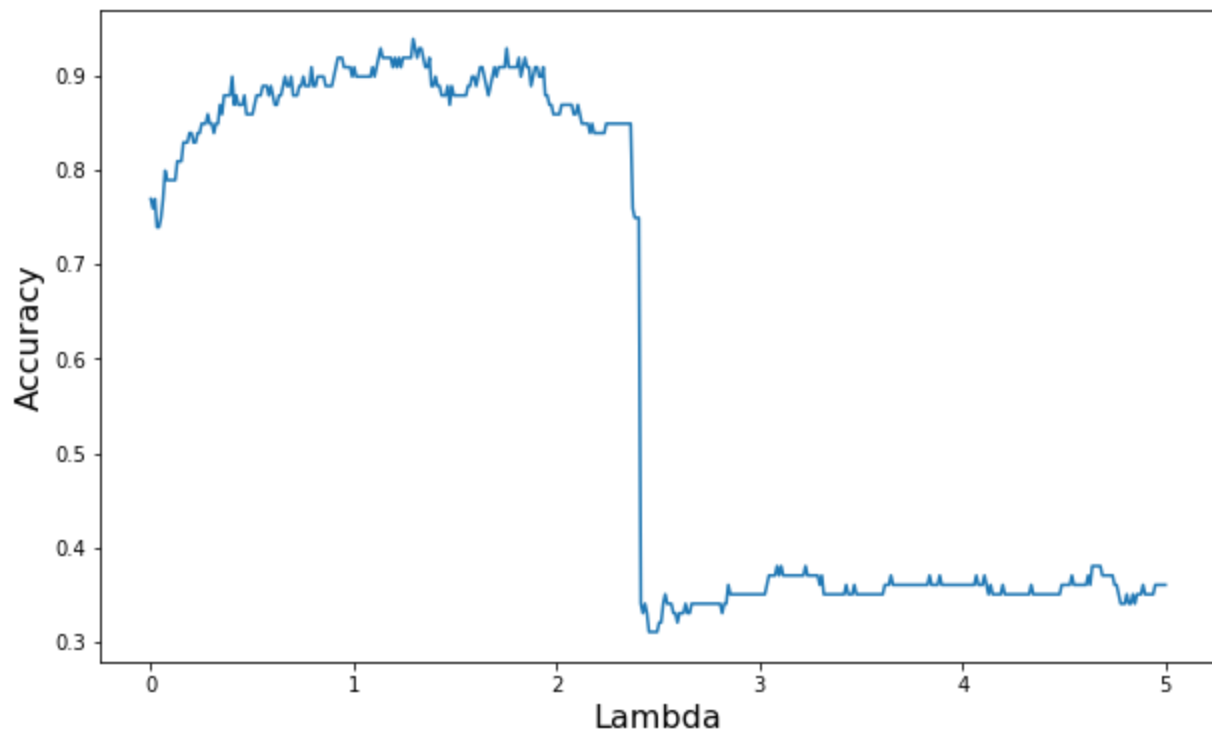


Figure 15 Penalty factor λ vs accuracy with Extracted Reward Function 1

Question 19

The results are shown in command shell:

```
Max Lambda: 1.292585
Max Accuracy: 0.940000
Accuracy: 0.940000
```

Figure 16 The value of max λ for which accuracy is maximum

Therefore, the value of $\lambda_{\max}^{(2)}$ is 1.2926 and the corresponding maximum accuracy is 0.94

Question 20

Heat maps of the ground truth reward and the extracted reward are shown below:

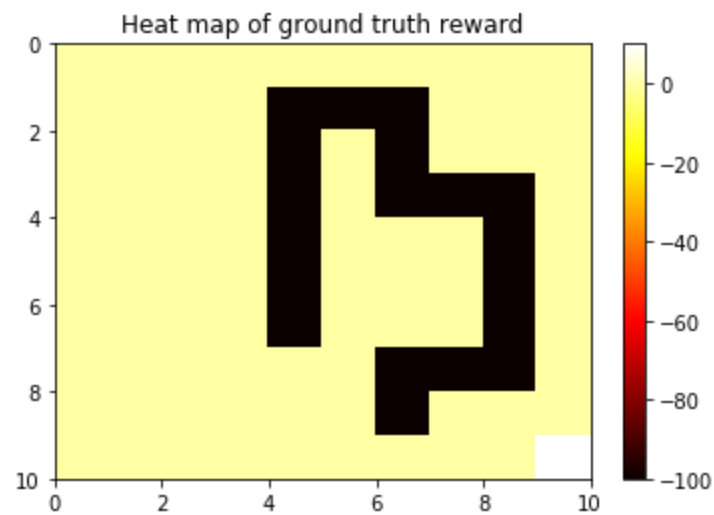


Figure 17 Heat map of the ground truth reward

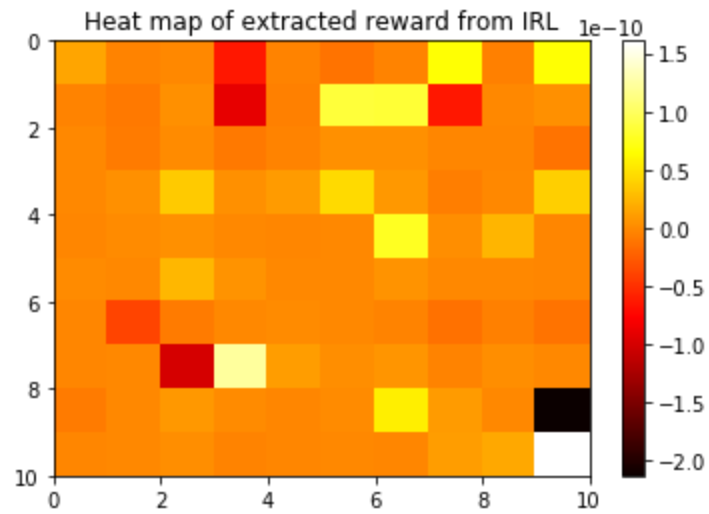


Figure 18 Heat map of the extracted reward

Question 21

We use the optimal state-value function to compute the optimal values and generate the heat map of the optimal state values across the 2-D grid:

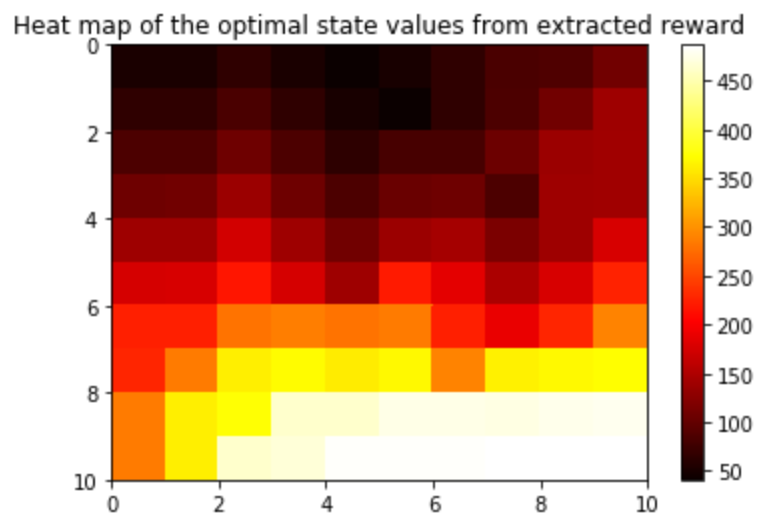


Figure 19 Heat map of the optimal state values from extracted reward function

Question 22

Generally speaking, the optimal state values in two figures are increasing from the top left to the bottom right to the reward for both Reward Function 2 and Extracted Reward Function 2, and we can observe states with larger state values.

One obvious difference is that the color scaling is much more different between these two figures. In the heat map of the optimal state values for ground truth, the range of state values are from -15 to 45, while for the heat map of the optimal state values for extracted reward, that range is from 40 to 490. Such difference is caused by the higher upper bound of the Reward Function 2 and Extracted Reward Function 2. In addition, another observed difference is that the state values decay from the bottom right to the top left in the heat map of the optimal state values for ground truth, while the state values decay from the bottom to the top in the other heat map for extracted reward. The reasonable explanation is that for the extracted reward heat map, the states in the last few rows have higher rewards than those in the rows above the last few ones. Consequently, the trend is decaying from bottom to top for the extracted reward heat map.

Question 23

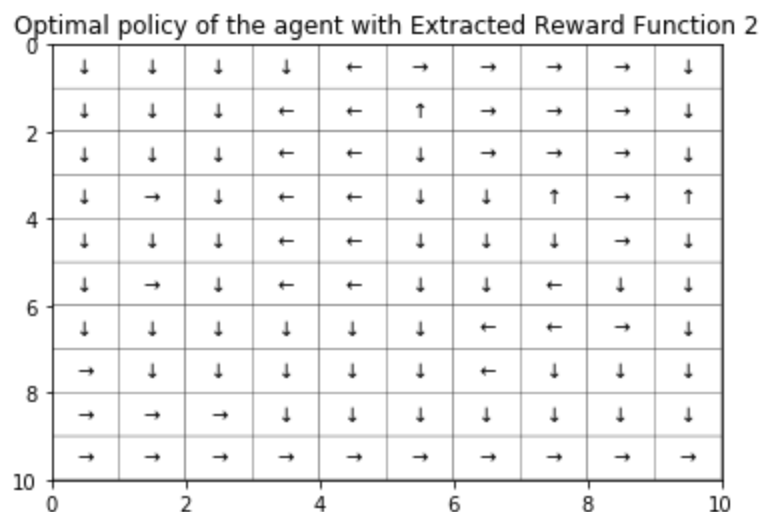
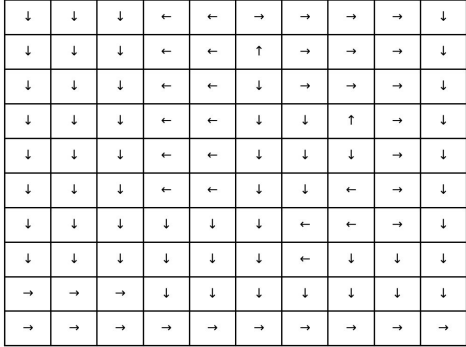



Figure 20 Optimal policy of the agent with Extracted Reward Function 2

Question 24

Table 2. Optimal policy in Question 9 and 23

Optimal policy of the agent with ground truth Reward Function 2 (Question 9)	Optimal policy of the agent with Extracted Reward Function 2 (Question 23)
	

Like the figures shown above, there are several states with different policy for the extracted reward. The optimal policy for extracted reward heads to the states with larger reward. In the first three columns, the general trends still go to the end and they are all outside the negative penalty regions. For the other different states, they may be caused by the threshold of RL algorithm, where there exist fluctuations in the extracted reward.

Question 25

In Table 2, we can clearly find several discrepancies in the optimal policy for the extracted reward, and they may be caused by the threshold of RL algorithm. To remove the influence of fluctuations in the extracted reward and fix the discrepancies, the extracted reward ought to be more precise to converge.

Optimal state values of extracted reward										
0	52.5	52.8	65.5	51.7	41.1	51.4	65.9	84.5	87.1	109.6
	66.7	67.1	84.0	65.9	51.4	41.2	66.1	84.7	109.7	139.2
2	85.2	85.6	107.7	84.8	65.2	81.2	82.4	107.2	138.6	141.2
	108.7	109.3	137.3	108.1	85.6	105.1	107.6	84.7	139.3	141.4
4	138.7	139.5	175.1	138.7	109.2	137.9	143.9	116.3	140.0	178.2
	176.6	177.6	218.6	176.8	140.1	220.9	186.5	147.4	178.5	226.9
6	224.3	225.4	281.0	286.8	280.2	286.0	224.0	189.5	227.4	290.8
	227.6	286.1	361.5	370.4	361.1	369.3	291.1	364.5	369.5	372.8
8	285.8	361.5	375.2	463.3	463.6	475.0	474.8	474.3	476.6	479.3
	285.8	363.0	463.2	468.0	483.8	484.4	484.3	486.5	486.7	486.7
10										
	0	2	4	6	8	10				

Figure 21 Optimal state values of extracted reward

Above is the optimal state values of extracted reward, which shows that the probabilities are close among four directions. We can try to make the optimal policy be the same by reducing the value of threshold ϵ . As a result, we decrease ϵ from 0.01 to 0.0000001, and we get the optimal policy with decreased ϵ .

Table 3. Optimal policy in Question 9 and 23 (with new $\epsilon = 0.0000001$)

Optimal policy of the agent with ground truth Reward Function 2 (Question 9)					Optimal policy of the agent with Extracted Reward Function 2 (Question 23)				
Optimal policy from ground truth reward					Optimal policy from extracted reward				
0	↓	↓	↓	←	←	→	→	→	↓
	↓	↓	↓	←	←	↑	→	→	↓
2	↓	↓	↓	←	←	↓	→	→	↓
	↓	↓	↓	←	←	↓	↓	↑	↓
4	↓	↓	↓	←	←	↓	↓	↓	↓
	↓	↓	↓	←	←	↓	↓	←	↓
6	↓	↓	↓	↓	↓	↓	←	←	↓
	↓	↓	↓	↓	↓	↓	↓	↓	↓
8	→	→	→	↓	↓	↓	↓	↓	↓
	→	→	→	→	→	→	→	→	↓
10									
	0	2	4	6	8	10			

As expected, the optimal state values are all the same for both ground truth Reward Function 2 and Extracted Reward Function 2. The maximum accuracy reaches 100%.

Then, we define a random reward function to generate its heat map, optimal state value and optimal policy.

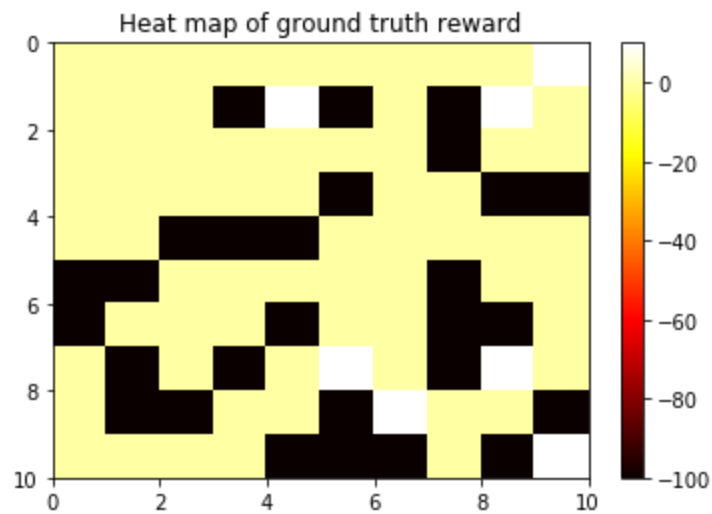


Figure 21 Heat map of the ground truth reward

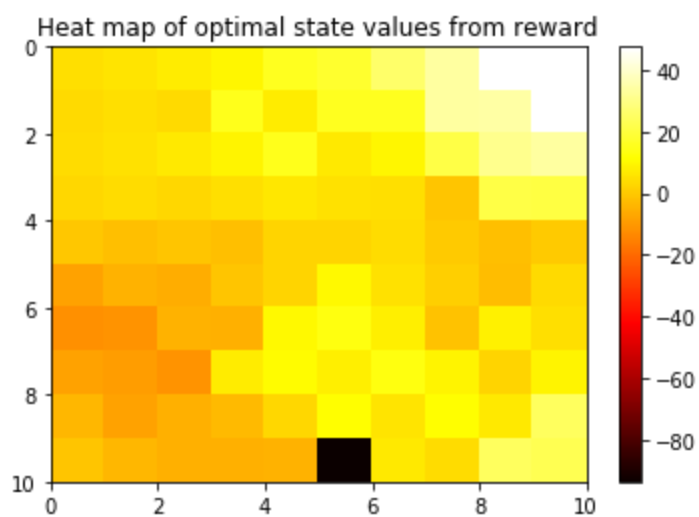


Figure 22 Heat map of the optimal state values from reward function

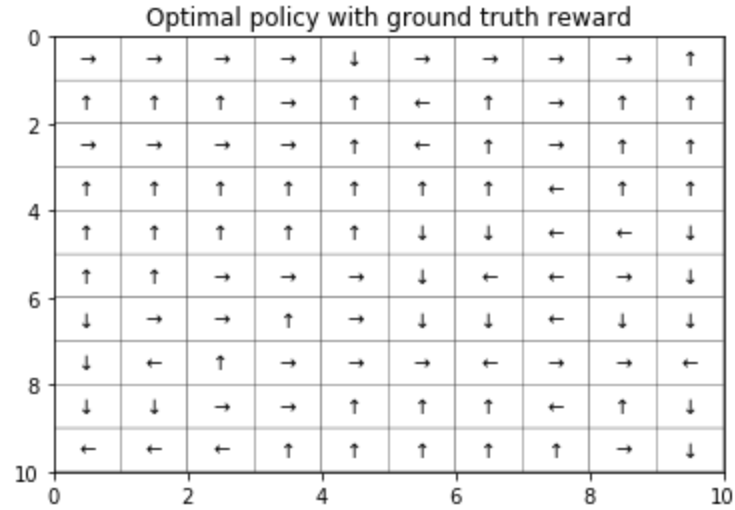
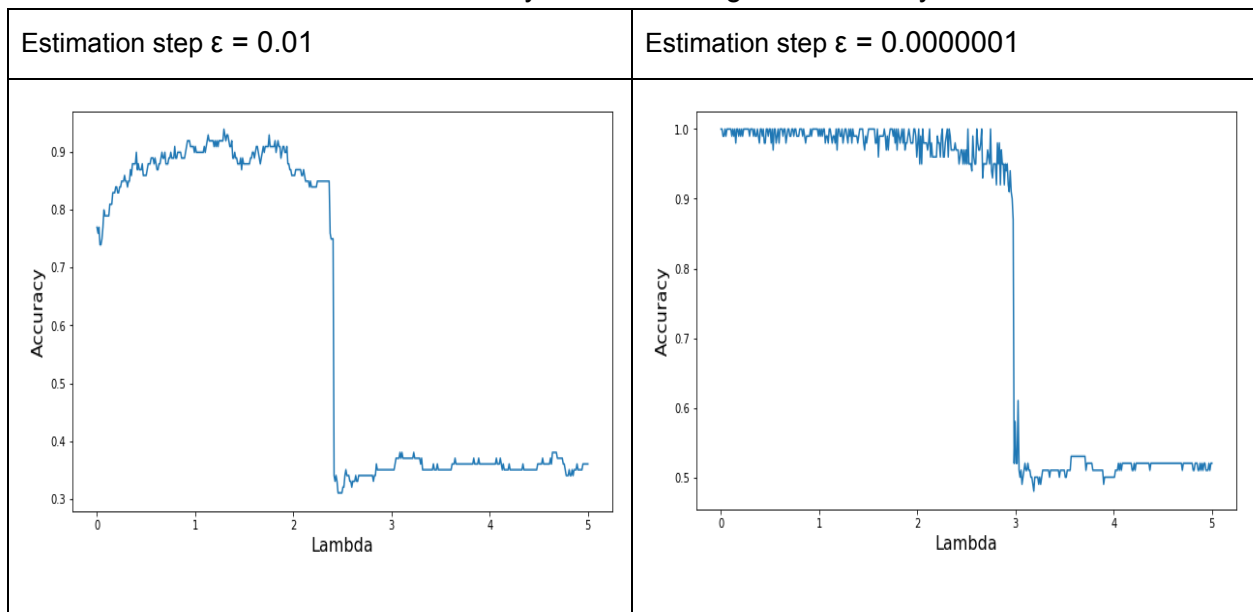


Figure 23 Optimal policy of the agent with reward Function

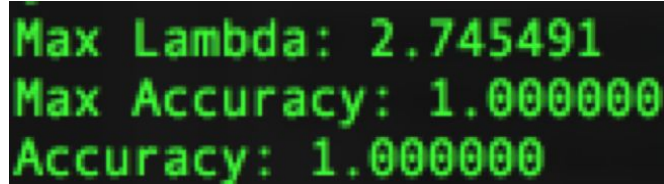
Additionally, we also compare the value of accuracy vs the penalty factor with different value of ϵ , sweeping λ from 0 to 5 for 500 evenly spaced values

Table 3. Penalty coefficient λ against Accuracy



For estimation step $\epsilon = 0.01$, we have $\lambda = 1.2926$ with the corresponding accuracy is 0.94 shown in Figure 16.

For estimation step $\epsilon = 0.0000001$, the results are:



```
Max Lambda: 2.745491
Max Accuracy: 1.000000
Accuracy: 1.000000
```

Figure 24 The value of max λ for which accuracy is maximum

As a result, we have $\lambda = 2.7455$ with the corresponding accuracy becomes 100% shown in Figure 24.

Therefore, inverse reinforcement learning (IRL) algorithm can be the same with the optimal policy learning using reinforcement learning (RL) algorithm by a slight modification with precisely enough small value of estimation step in the value iteration algorithm so that such discrepancies can be fixed.