Advance Neuroscience HW6

Ali Ghavampour - 97102293

Question 1& 2 - Methods

First, reward matrix is initialized. A reward (platform) is placed at [x,y] = [10,10] and a punishment is placed at [x,y] = [8,5]. Then the agent lerans how to find the platform by using the Q-learning algorithm. To build up the Q matrix, we have 15 states and 4 actions (2 or 3 at some states). So we initialize a 225 matrix with 0 values. In each row (state) of Q matrix, columns 1, 2, 3 and 4 are actions right, bottom, left and top respectively. For each row (state), the value of forbidden actions is set as 'NaN'.

The algorithm consists of three stages:
1) Choosing action (which is done using 2 different methods)
2) Performing the action
3) Update

In stage 2, simply the chosen action is performed (we dicuss stage 1 later). Also, in stage 3, the Q matrix is updated using the TD rule which is quite straight forward. It's stage 1 that needs a little bit of attention. Stage 1 has been implemented using two different methods. The first on is a deterministic method. In this method, agent looks at the current stage in Q matrix and chooses the action that has the most significant expected reward. e.g. if a row consists of the values 0.1, 0.002, 0.5 and 'NaN' the agent chooses third action which is Left action. Also, if there are actions with same values, the action is choosed randomly between them.

Second method is a stochastic method. In this method, agent chooses the action based on the output probability of softmax function which is brought in the bottom:

$$P(a|s) = \frac{e^{Q[s,a]/T}}{\sum_{a_j} e^{Q[s,a_j]/T}}$$

Note that the stages are repeated every time point of each trial. Specially the update stage which is performed by each step the agent takes.

Figure 1 is the learning trials of method 1. We can see that after some trials, the path is converged and we can see that here agent is greedy and after it finds the right path, it doesn't want to change its course. The initial point is set to be a exact number in this plot.

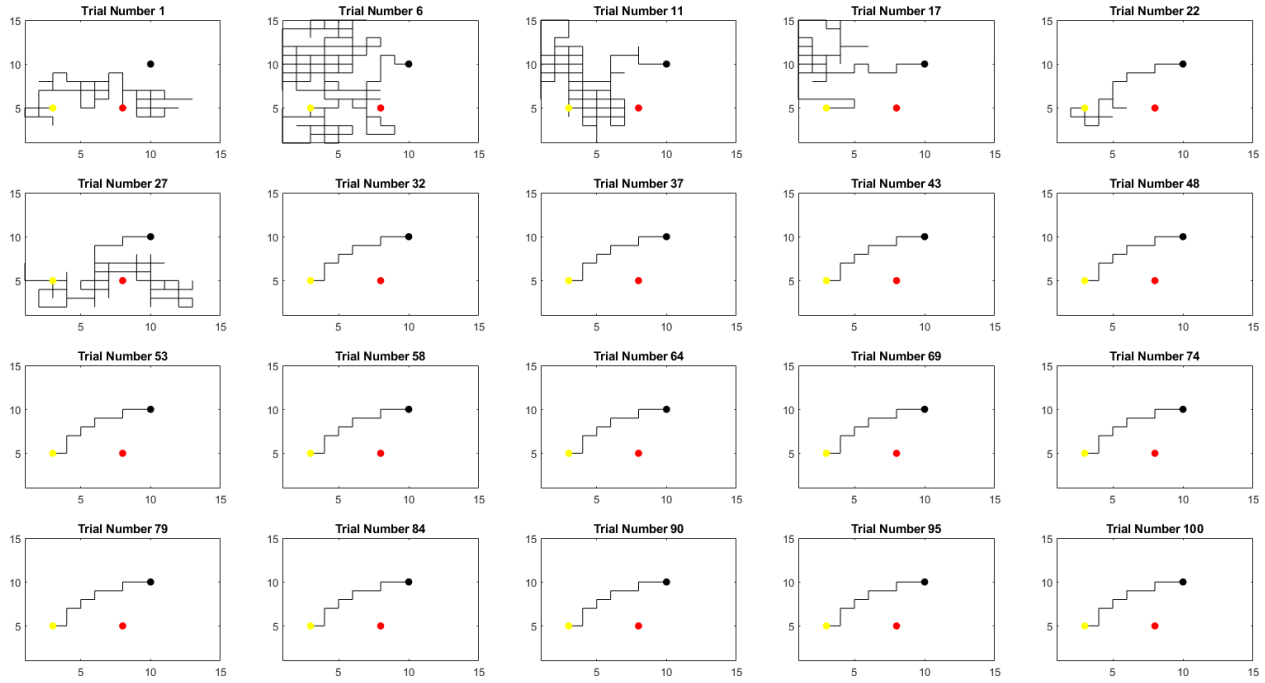

Figure 1: $\eta = 0.2$ and $\gamma = 0.5$, method 1, red point: cat - black point: platform - yellow point: fixed initial point

We can see that in first trials, agent moves randomly until it reaches either reward or the punishment. Then, the path converges and the agent always choose the converged path. Figure **??** is the contour plot of learned values. The maximum state action value is chosen as each state's vlaue.

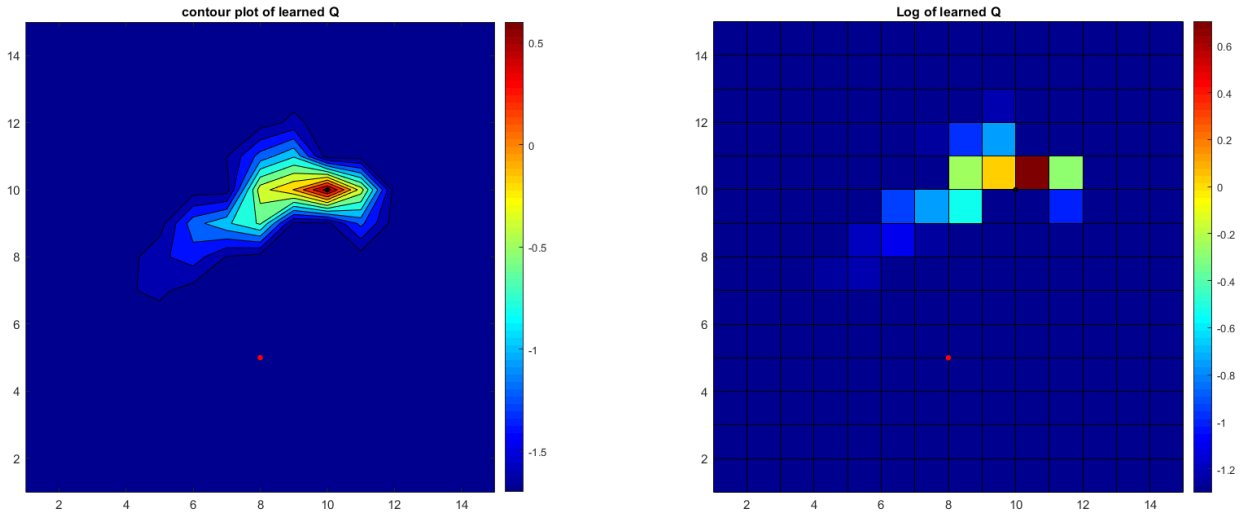

Figure 2: $\eta = 0.2$ and $\gamma = 0.5$, method 1, red point: cat - black point: platform - fixed initial point

Figure 3 is the gradient plot. It shows which way the agent choses in each state.

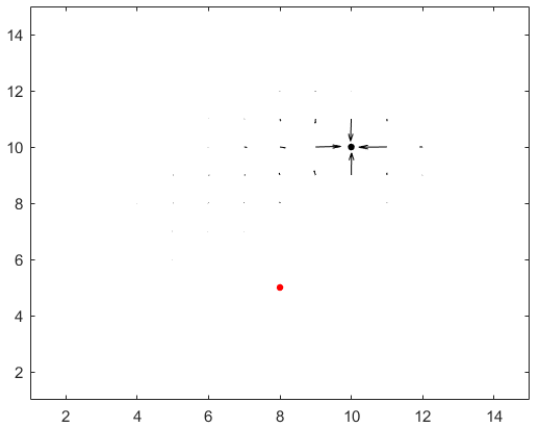

Figure 3: $\eta = 0.2$ and $\gamma = 0.5$, method 1, red point: cat - black point: platform - fixed initial point

Now we use the same method but this time, the initial point is changed randomly. Figure 4, 5 and 6.

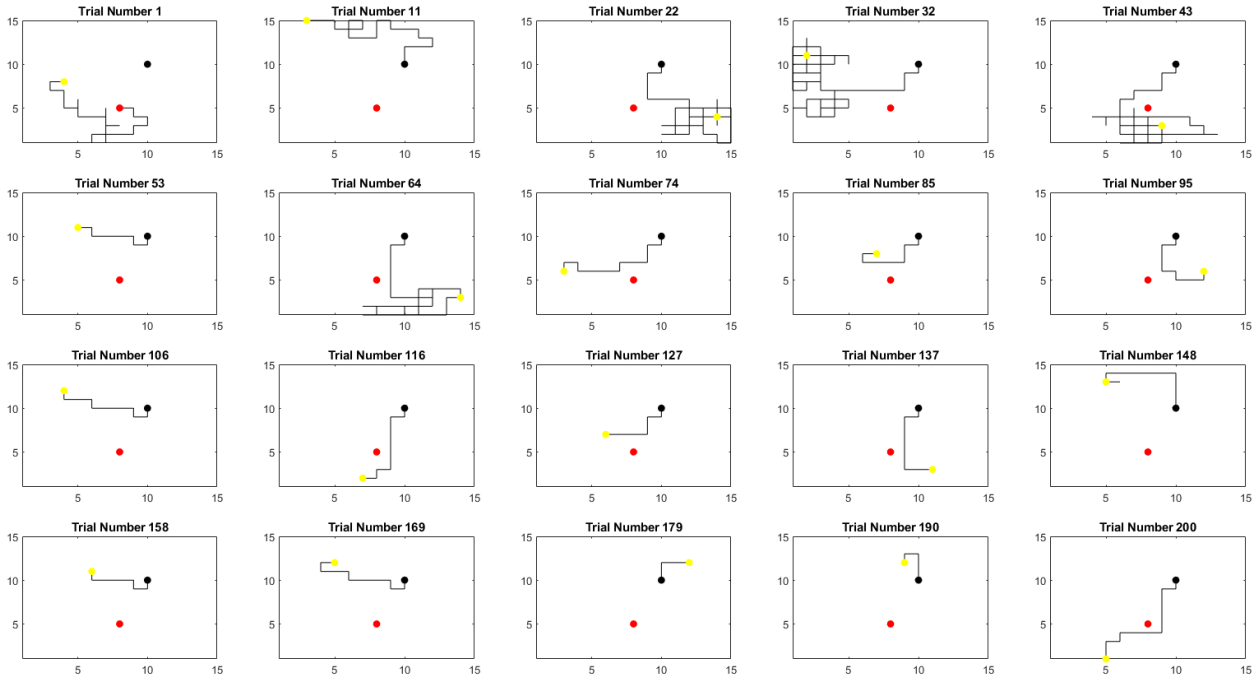

Figure 4: $\eta = 0.2$ and $\gamma = 0.5$, method 1, red point: cat - black point: platform - yellow point: random initial point

We can see that after learning, no matter where the agent is, it can find the path to the platfrom.

In figure 5, we can see that the agent has descovered more places and we have a wider knwoledge about the map.

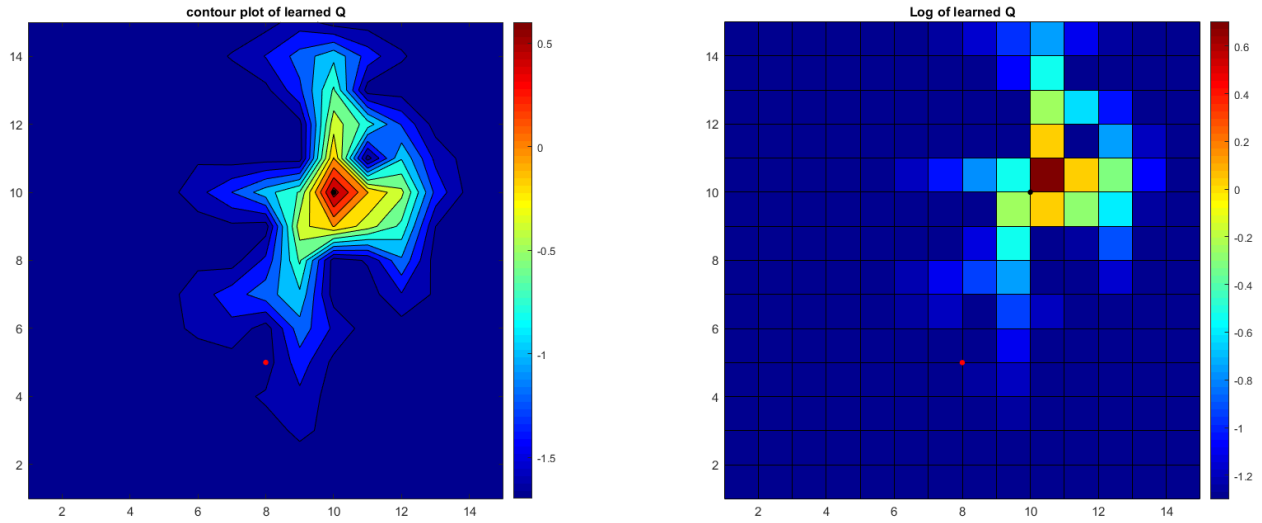

Figure 5: $\eta = 0.2$ and $\gamma = 0.5$, method 1, red point: cat - black point: platform - random initial point

Figure 6 is similar to figure 3.

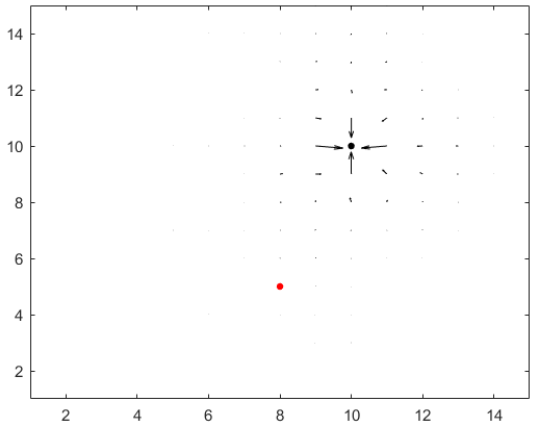

Figure 6: $\eta = 0.2$ and $\gamma = 0.5$, method 1, red point: cat - black point: platform - random initial point

Now using method 2 (softmax), we repeat the random starting point paradigm. This time we can see more curiosity in agent. After the values are learned, it does'nt always select the same path. Sometimes it wanders around so that maybe it cant find a better place to be. But finally it goes to the platform.

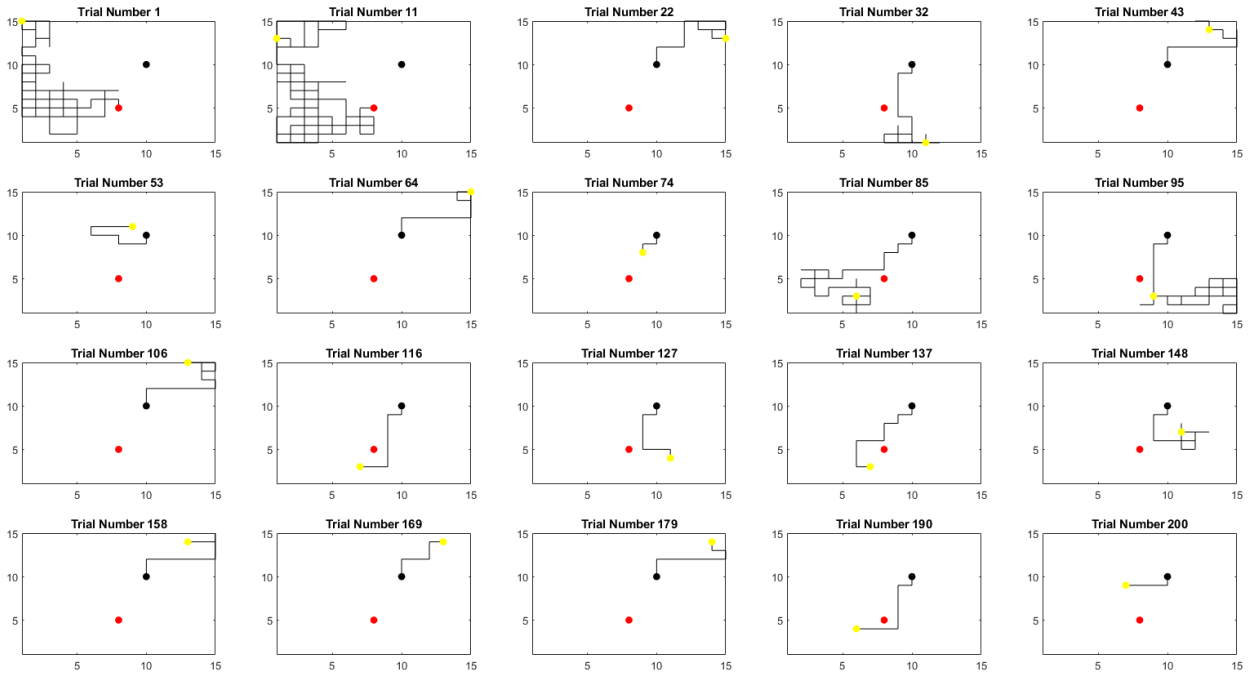

Figure 7: $\eta = 0.9$ and $\gamma = 0.5$, method 2, red point: cat - black point: platform - yellow point: random initial point

We can see that the agent has discovered more areas.

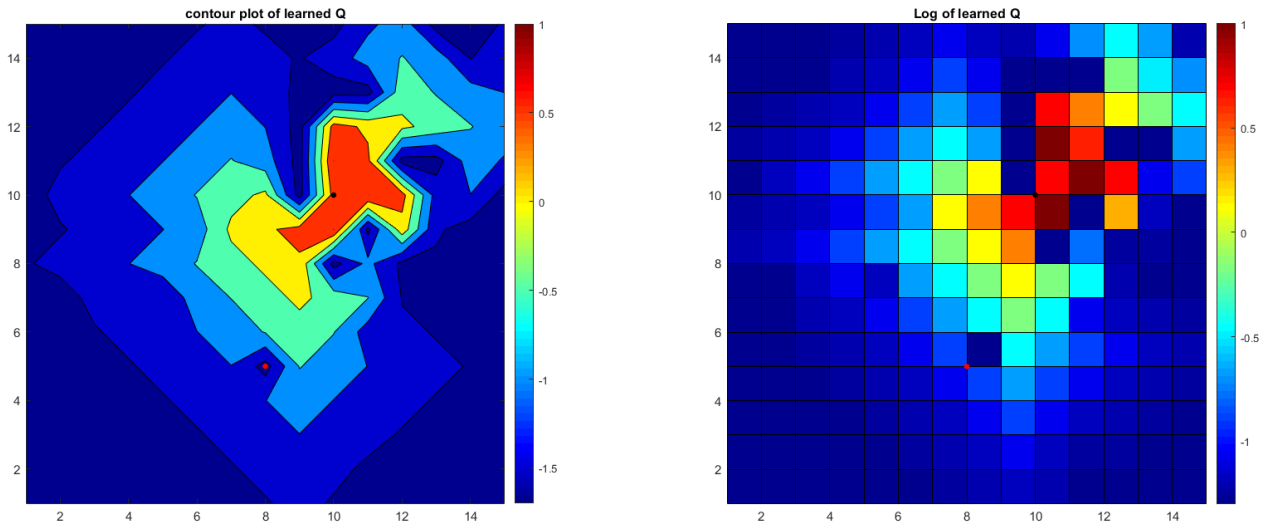

Figure 8: $\eta = 0.9$ and $\gamma = 0.5$, method 2, red point: cat - black point: platform - random initial point

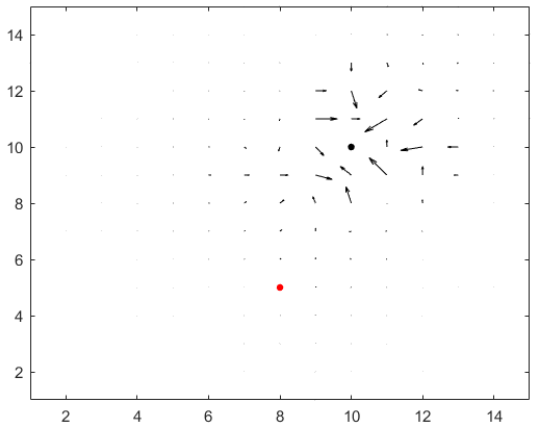

Figure 9: $\eta = 0.2$ and $\gamma = 0.5$, method 2, red point: cat - black point: platform - random initial point

Question 3 - Effect of learning rate and gamma

Learning rate in a machine learning model describes how much the agent is willing to learn about the environment by receiving a single observation. The learning rate can't be too small or too large. It must be something in the middle. In some models, the learning rate changes during the time and it results in a faster learning if it is appropriate for that exact problem. Figure 10 and 11 show the effect of learning rate on learning.

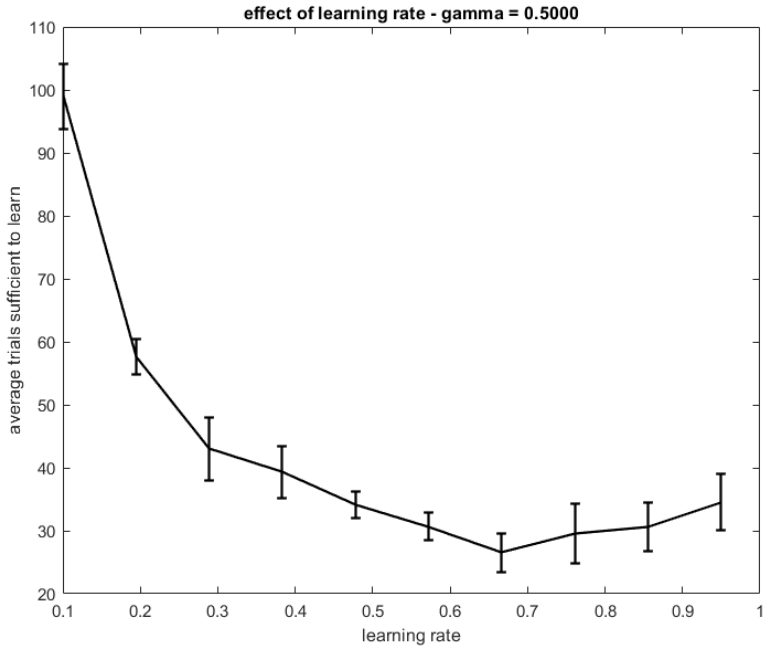

Figure 10: The bars on each point are the sem of the values.

We can see that as the learning rate gets larger, the sufficient trials tend to fall. Although, in the end of the plot we can see a rise again which is totaly expected. In order to see if the rise will continue we have figure 11.

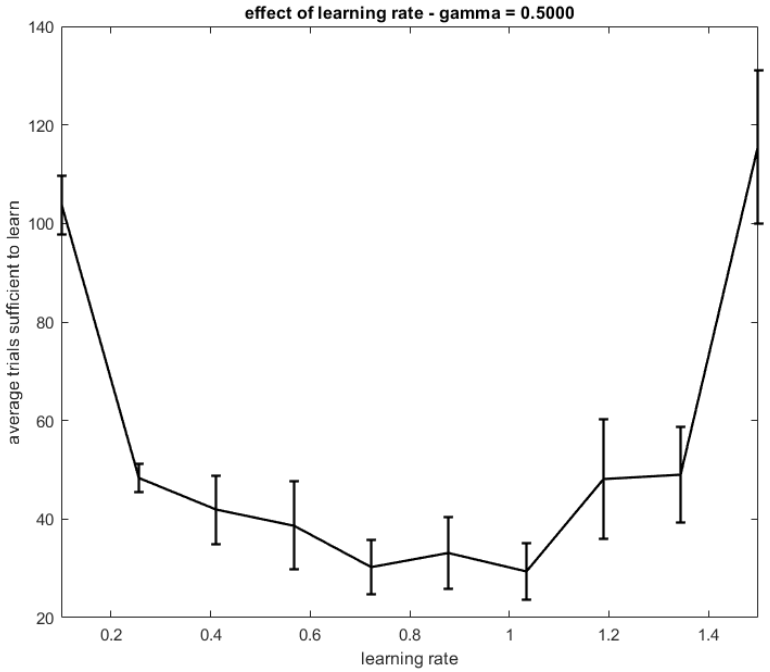

Figure 11: The bars on each point are the sem of the values.

As we expected, there are omptimum fixed learning values in the middle of interval rather than in the begining or the end. Figure 12 shows the effect of gamma in the learning. If we think about it, in such a problem that there is only one reward and one punishment in the map, more back propagation of error would make the learning faster.

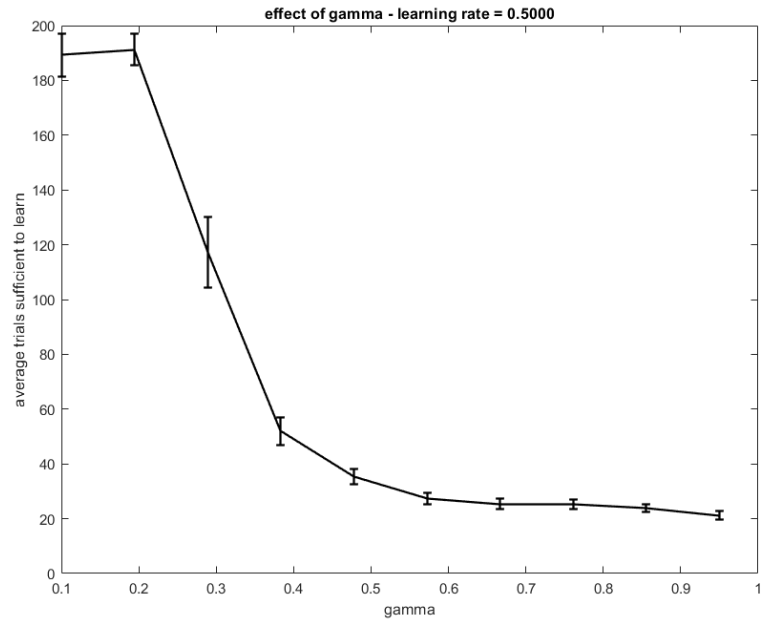

Figure 12: The bars on each point are the sem of the values.

Question 4

In this part first we will see what reward (one +10 and the other +1) the agent will choose for different parameter values. First, we take the effect of gamma into account. By taking $\eta = 0.5$, we change gamma and calculate the percentage that higher result is chosen. Also, the initial point is placed nearer to the lower reward. Basically, one may think that the agent must always go for the higher reward. But, if the first random steps go toward the lower reward (especially that it's closer to the agent) and the agent is greedy, it can't predict that there is a higher reward in the map. Using the sofmax policy and different parameters, we will see how the agent will act in this paradigm.

Figure 13 is the example that we discussed. Note that starting point is fixed and is closer to the lower reward (not higher reward). The y axis is the percentage of choosing higher reward by the agent. Higher percentage means that we have a smarter agent. We can see that when gamma is around 0.5 we have a peak in percentage and around it, the percentage falls.

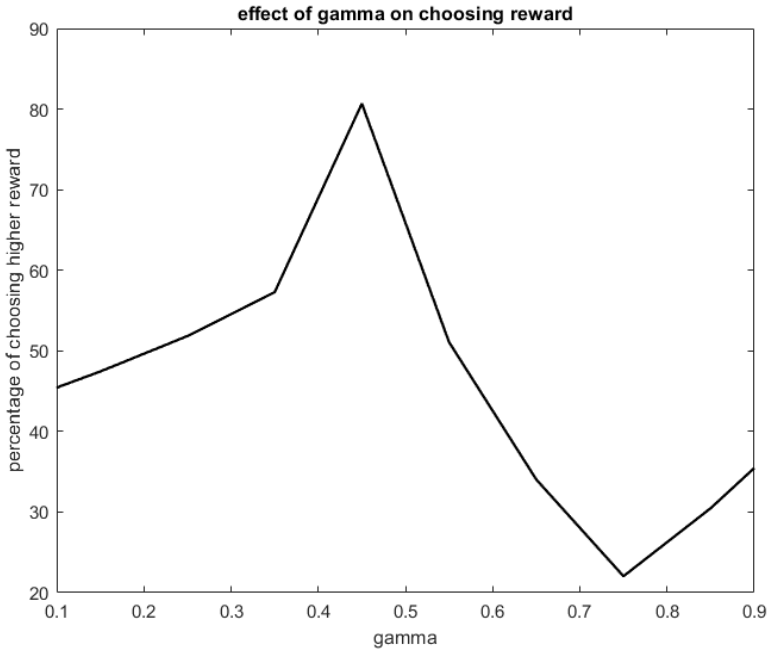

Figure 13: Percentage of choosing the higher reward - fixed starting point closer to lower reward.

Also, by choosing a fixed starting point which is closer to the higher reward we get figure 14. As expected we have a larger overal probability of choosing the higher reward but we still can see the behavior that we had in figure 13.

The reason that gamma has such an effect could be because of the definition of gamma. Gamma is the factor that propagates the value of current state to the last state action. The larger gamma is, the more of that value propagates to the last state action. So what happens if gamma is to large or too small? In question 1 and 2 where we had one poisitve and one negative reward, more propagation makes things more clear. If you have more propagation you will know what paths you must not go and what you must go. But in the case that there are two positive rewards, larger gamma could confuse the agent because based on what random first steps agent takes, the high propagation could lead to lower reward for the rest of trials.

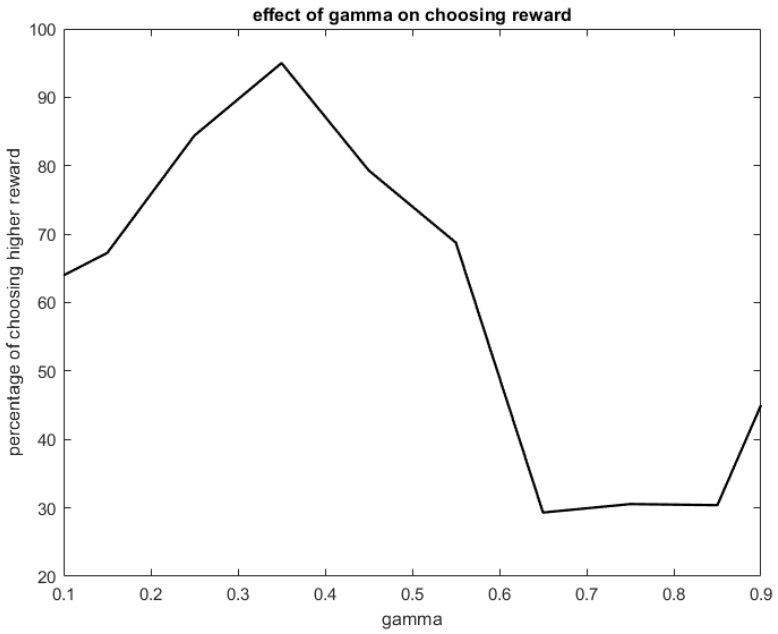

Figure 14: Percentage of choosing the higher reward - fixed starting point closer to higher reward.

Figure 15 is the effect of learning rate on the percentage. As expected it does not have any effect on the percentage and the important factor here is only gamma.

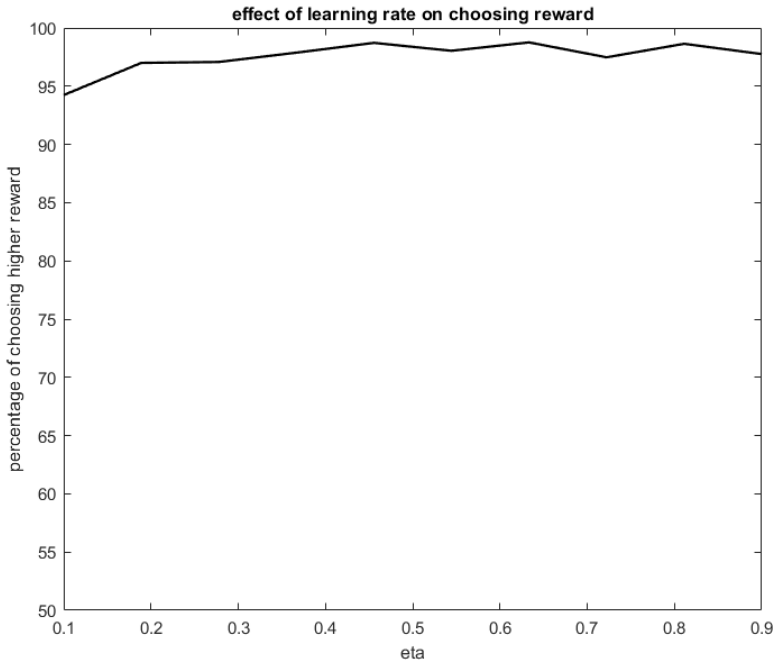

Figure 15: Percentage of choosing the higher reward - fixed starting point closer to higher reward.

In another scenario, when changing the place of targets (higher reward = [5,12] and lower reward = [13,8]) and placing starting point at [7,10], we get figure 16. The agent is closer to the higher reward, so as expected, most of the time it goes to the higher target because it's closer to it. Also, as gamma increases this percentage becomes larger.

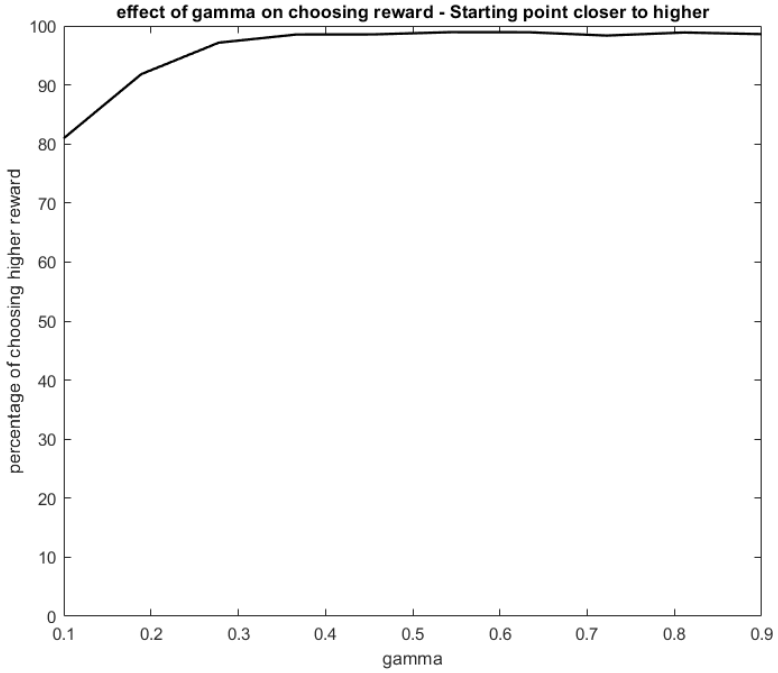

Figure 16: Percentage of choosing the higher reward - fixed starting point closer to higher reward.

Also, by swapping the place of rewards, (higher reward = [13,8] and lower reward = [5,12]) again the agent reaches the closer target but this time it's the lower reward (figure 17) and again we have the same effect of gamma that we had in figure 16.

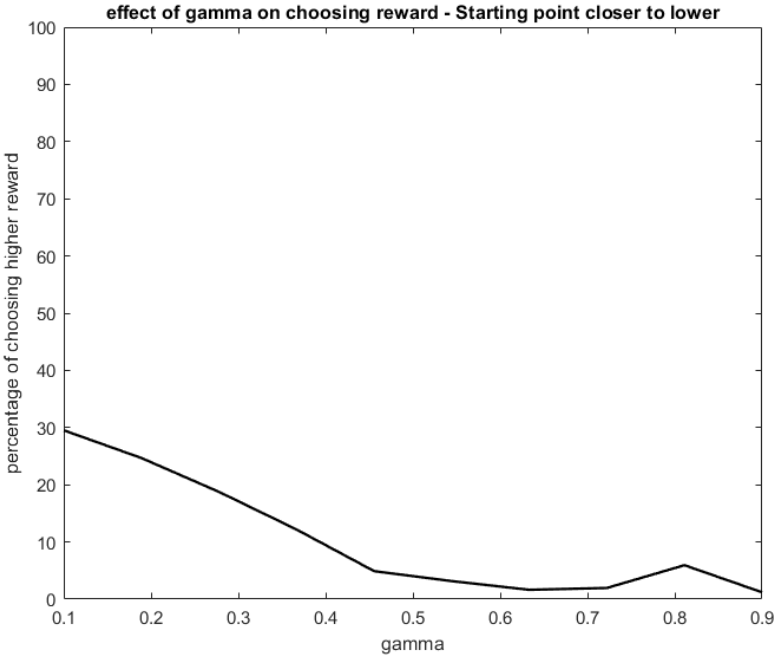

Figure 17: Percentage of choosing the higher reward - fixed starting point closer to higher reward.

Question 5 - TD(λ) Rule

In this section we add TD(λ) to the model. In update stage of each step, not only the value of last step will be updated, the value of last 4 steps are updated with respect to discount factor λ and dt. By propagating error so far back, the agent could understand more about the map each step it takes. So, The learning would probably become faster and stronger. Faster means that there are less trials needed to find the best path. Stronger means that every step it takes, it is more confident about the outcome. (Remember the stochasticity of the process by using softmax) Figure 18, 19 and 20 show the learning using lambda.

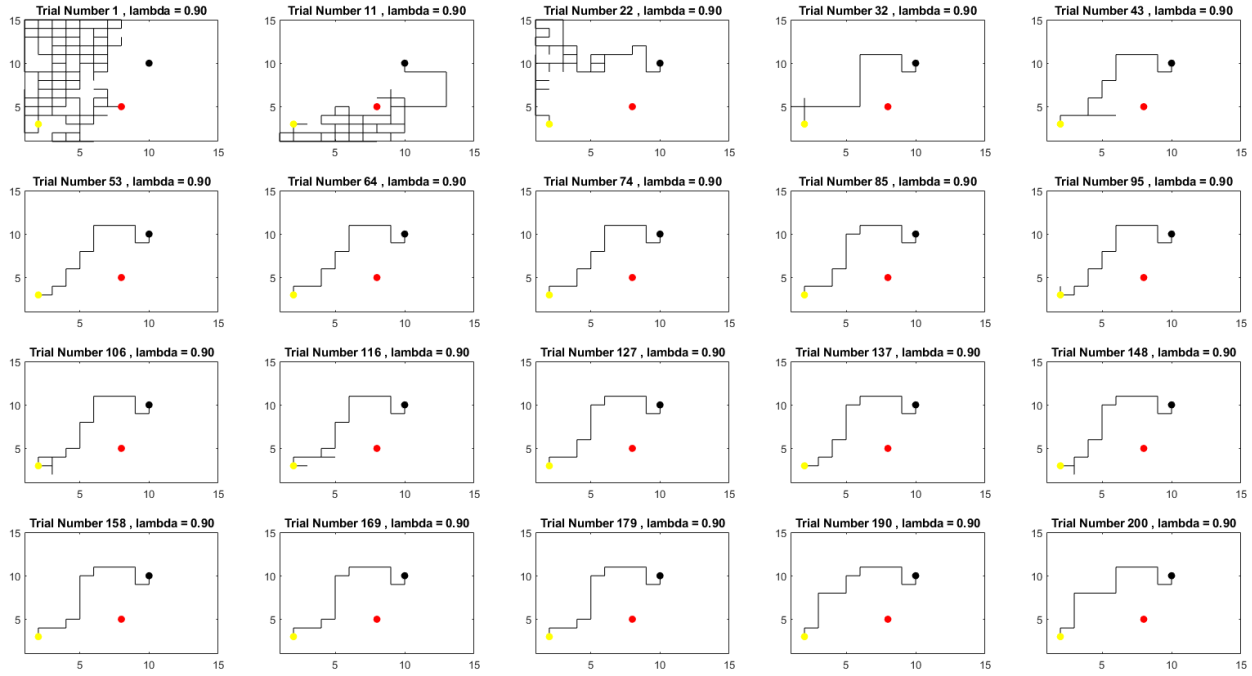

Figure 18: Agent path during learning - gamma = 0.8 , $\eta = 0.5$, $\lambda = 0.9$

We can already see the increased confidence in the last steps of figure 18. Also in it is apparent from figure 19, there is more knwoledge about the map.

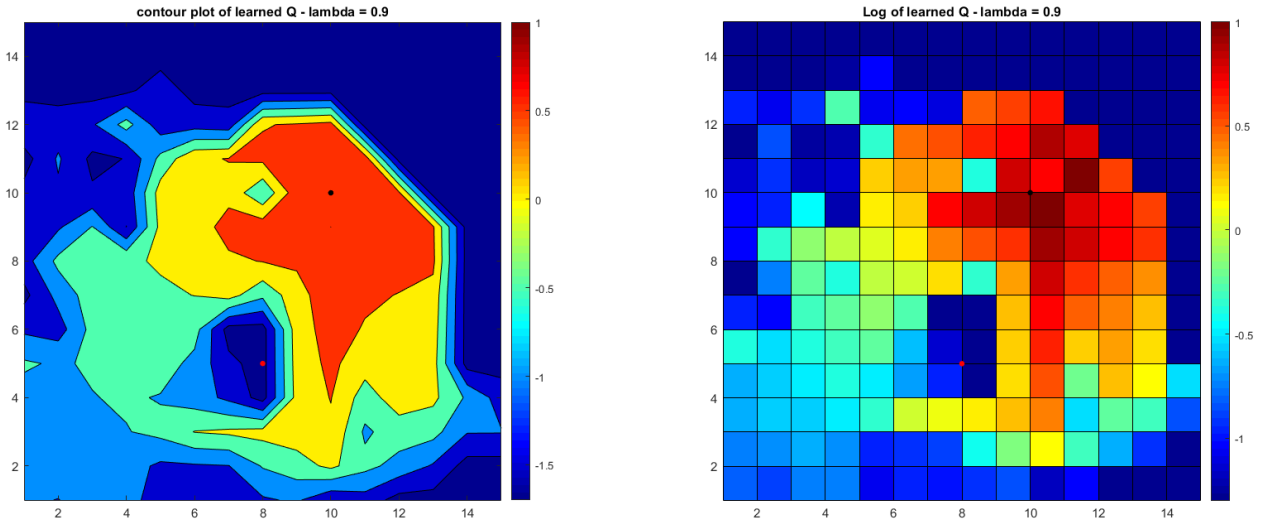

Figure 19: Learned values - gamma = 0.8 , $\eta = 0.5$, $\lambda = 0.9$

The last three figures was for a fixed starting point. The next three figures are the same

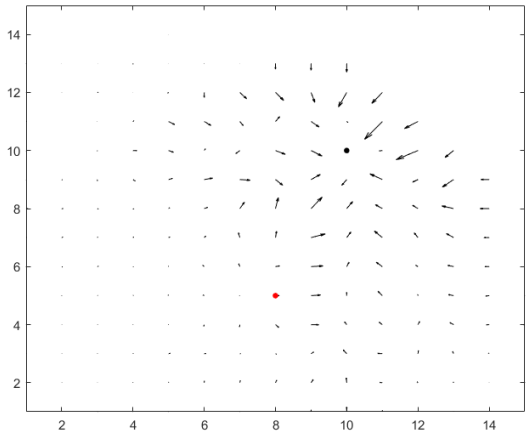

Figure 20: Gradient of learned values - gamma = 0.8 , $\eta = 0.5$, $\lambda = 0.9$

but for random starting point.

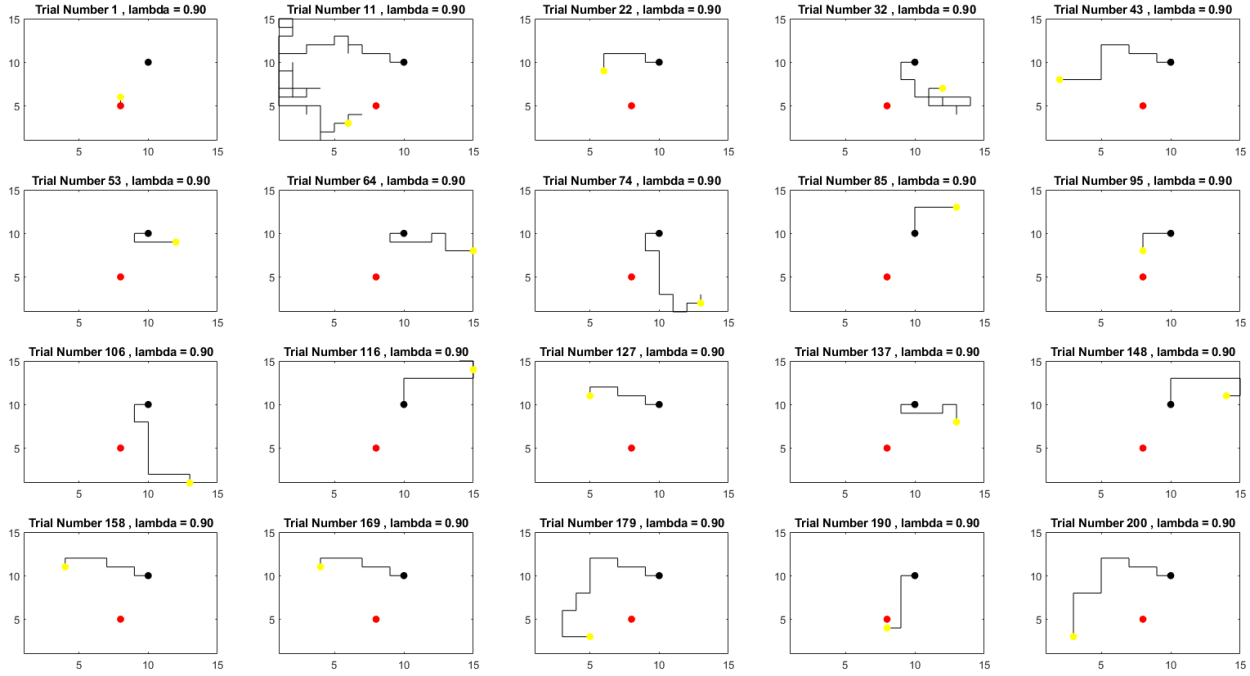

Figure 21: Agent path during learning - gamma = 0.8 , $\eta = 0.5$, $\lambda = 0.9$, random starting point

The explained confidence can also be seen here. Propagating error to the behind steps is like taking more trials to learn much more about the map.

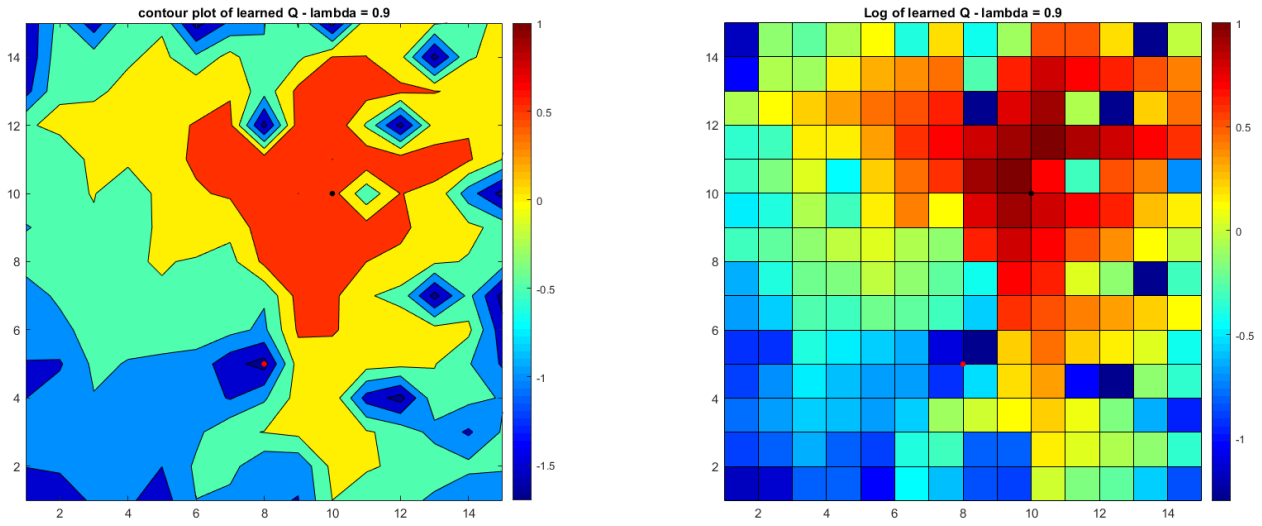

Figure 22: Learned values - gamma = 0.8 , $\eta = 0.5$, $\lambda = 0.9$, random starting point

Because of the random starting point, even more areas are known.

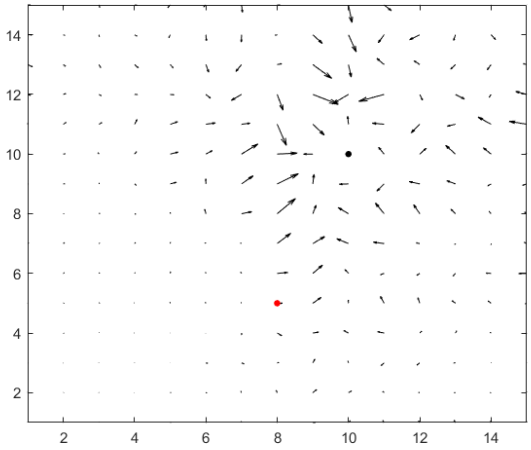

Figure 23: Gradient of learned values - gamma = 0.8 , $\eta = 0.5$, $\lambda = 0.9$, random starting point

Now in order to see the effect of λ on learning, we run the model for different λ values. (Figure 24) We can see that as λ increases, learning becomes faster.

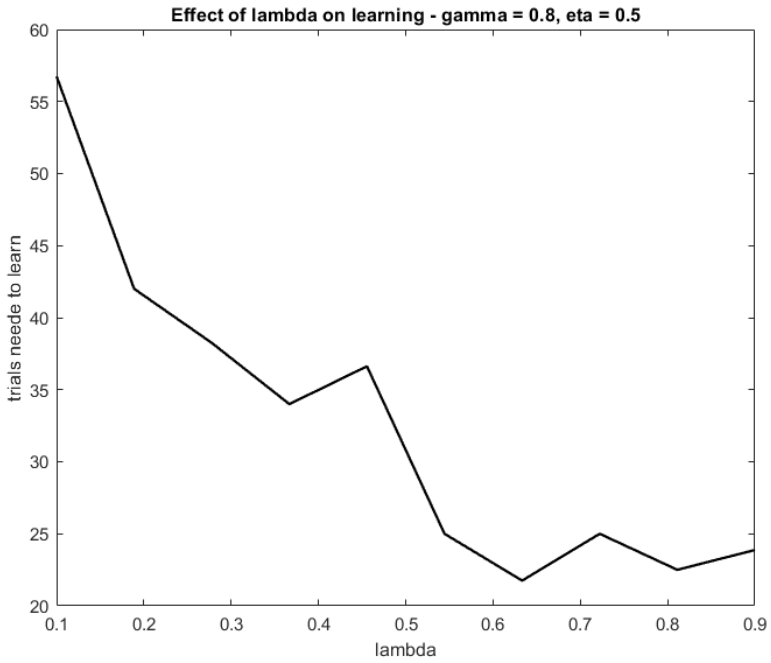

Figure 24: effect of λ on learning speed. y axis is the trials needed for the agent to confidently learn the path