NEURAL NETWORK PROJECT II
Q-Learning for World Grid Navigation

SUBMITTED BY

NAME: LIN KANG

STUDENT NUMBER: A0191818W

EMAIL ADDRESS: e0344070@u.nus.edu

1. Requirement of SVM project

Task 1: Write a MATLAB (M-file) program to implement the Q-learning algorithm, using the reward function as given in task1.mat and with the ϵ-greedy exploration algorithm by setting ϵ_k, α_k and as specified in Table I.

The file task1.mat is included in the zip file that also contains this document. It can be directly loaded into MATLAB and contains the matrix variable reward (dimension: 100x4), in which each column corresponds to an action and each row to a state. For example, the reward for taking action a = 3 at state s = 1 to enter state s = 2 is given by the (1, 3) entry of reward, i.e., $\rho(1, 3, 2)$ = reward (1, 3). Note that rewards can be negative.

In this task, ϵ_k and α_k are set to the same value. You are required to run your program 10 times (i.e., 10 runs) for each set of parameter values and record the number of times the goal state is reached. (Note: It is possible that some of the runs may not yield an optimal policy that results in the robot reaching the goal state; these runs are not to be counted.) The maximum number of trials in each run is set to 3000. The average program execution time of the "goal-reaching" runs is to be calculated and entered into the table (as indicated by "?"). The final output of your program should be an optimal policy (if the goal state is reached in any of the 10 runs), and the reward associated with this optimal policy. The optimal policy is to be expressed as a column vector containing the state transitions, and also illustrated (in your report) on a 10 x 10 grid with arrows marking the trajectory of the robot moving from state s = 1 to state s = 100.

Task 2: Write a MATLAB (M-file) program to implement Q-learning using your own values of the relevant parameters. Assume that the grid size is 10 x 10 and implement your program in a MATLAB M-file. This M-file will be used to find the optimal policy using a reward function not provided to the students, as part of the assessment scheme discussed in Section V.

2. Results of Q-Learning for world grid navigation

ϵ_k, α_k	No. of goal-reached runs		Execution time (sec.)	
	$\gamma = 0.5$	$\gamma = 0.9$	$\gamma = 0.5$	$\gamma = 0.9$
$\frac{1}{k}$	Failed	Failed	Failed	Failed
$\frac{100}{100+k}$	Failed	180.60	Failed	2.53
$\frac{1+log(k)}{k}$	Failed	Failed	Failed	Failed
$\frac{1+5log(k)}{k}$	Failed	427.90	Failed	8.26

Table 1 Results of Q-learning

With the converge condition and ensure the difference between this Q-function and the previous Q-function is less than 1e-3.

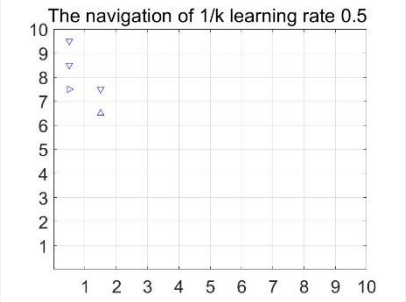
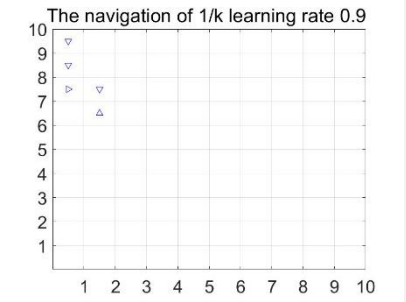
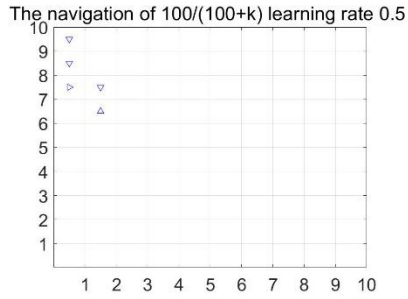
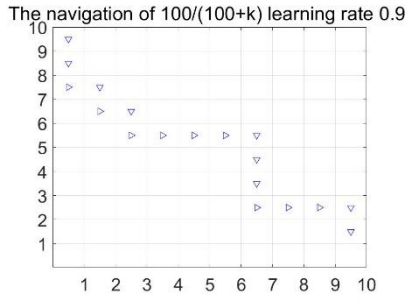
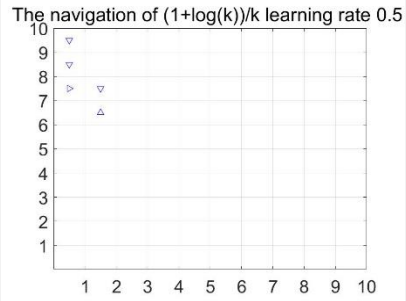
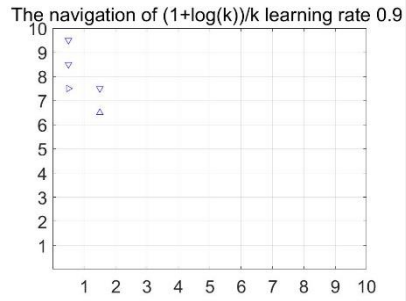
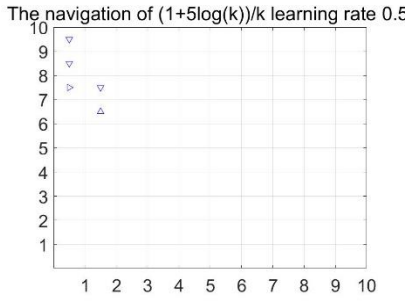
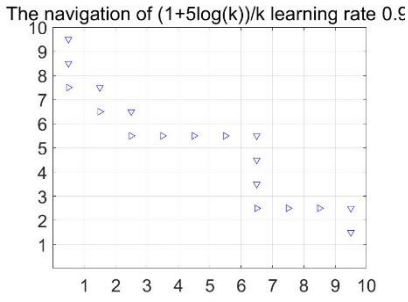
ϵ_k, α_k	$\gamma = 0.5$	$\gamma = 0.9$
$\frac{1}{k}$		
$\frac{100}{100+k}$		
$\frac{1+log(k)}{k}$		
$\frac{1+5log(k)}{k}$		

Table 2 Grid navigation of optimal policy

3. Discussion about the result on table 1 and table 2

3.1 Effect of discount rate

From the table 1, it shows that all the trails with learning rate as 0.5 failed. And the result of navigation shows that the calculation of Q function has some problems in step4 and step5 which makes the greedy algorithm trapped at this step.

Since for low value discount rate, the calculation of Q function will take less concentration on the reward from the next step, which misguides the direction and makes the navigation failed.

The result shows that the learning rate 0.9 is more suitable than the learning rate 0.5 in this navigation.

3.2 Choose of suitable ϵ_k, α_k

From the figure 1 shown below, ϵ_k, α_k will decrease quickly at the first 20-time step when use the function 1/k and (1+log(k)/k. On the other hand, for function 100/(100+k) and function (1+log5(k)/k, the ϵ_k, α_k will decrease slightly at first.

The result shows that the trail with function 1/k and (1+log(k)/k are failed to get the right navigation. It shows that for this navigation, we need to control the decreasing speed of ϵ_k, α_k. Too fast decreasing will affect the result of navigation. The slow rate decreasing is recommended for this project.

Since for the given rewards map, there exist a lot of misguiding that some direction could get a higher reward but lead to a wrong destination that could not get the final big reward (desired destination). To avoid these traps, we need more exploration at the previous steps which means that higher ϵ_k, α_k provides us more chances to explore. So the slow decreasing speed is a more suitable choice.

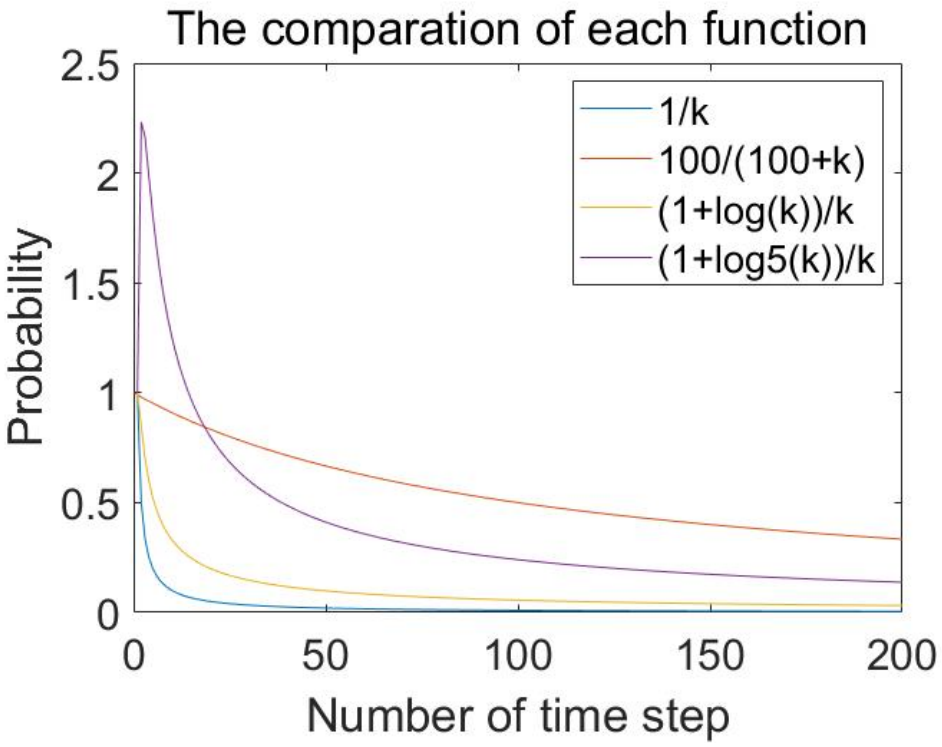


Figure 1 The comparation of each function

4. Task 2 discussion about the parameter

First, we need to determine the best choice of decreasing function for ϵ_k, α_k. I choose some functions as shown below with the number of goal-reached runs and the execution time in average. (with discount rate 0.9) The result is shown below table 2. I choose the function based on the condition provided and improve the function to get a slower decreasing at previous time steps. To get the final result with minimum iterations and time, choose 2000/(2000+k).

ϵ_k, α_k	**No. of goal-reached runs**	**Execution time (sec.)**
	$\gamma = 0.9$	$\gamma = 0.9$
$\frac{100}{100+k}$	115.90	1.67
$\frac{300}{300+k}$	58.70	0.37
$\frac{500}{500+k}$	45.60	0.19
$\frac{1000}{1000+k}$	39.40	0.20
$\frac{2000}{2000+k}$	37.30	0.21
$\frac{5000}{5000+k}$	41.30	0.33
$\frac{1+5log(k)}{k}$	260.40	4.82

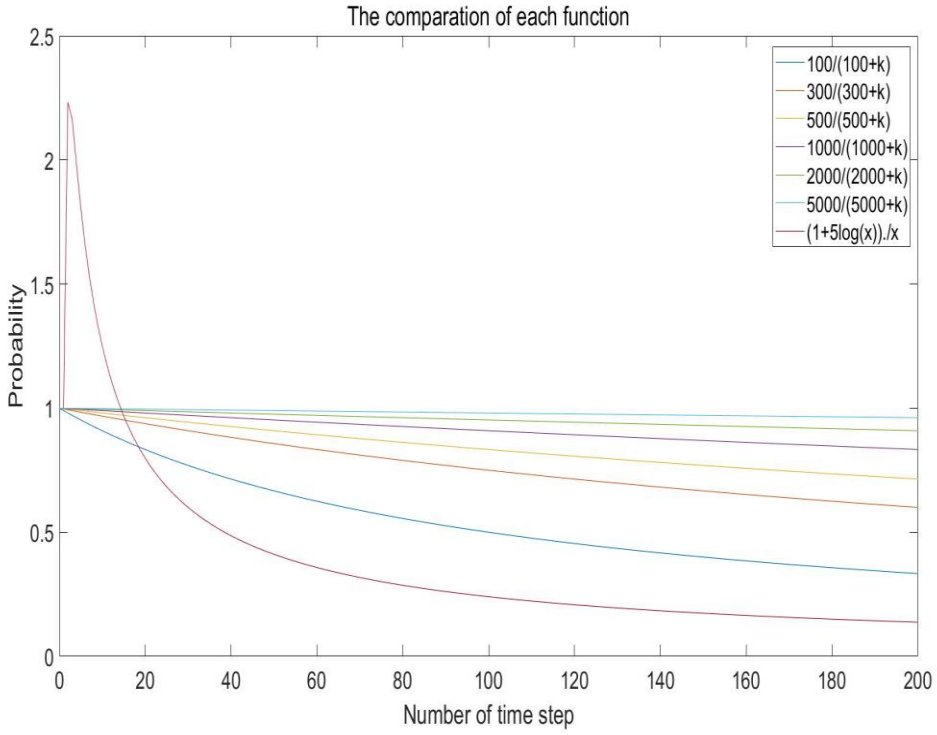


Figure 2 The comparation for task 2

From this figure 2, previous 6 functions will converge in less than 100 iterations, which means that for this navigation, more exploration and better the navigation. But too much more exploration is also a problem with cost more time to explore. So need to control the proportion of the exploration and exploitation to get the best choice.

Second, the determination of discount rate is also important. The result is shown below. The result is different every time. For the discount rate between 0.86-0.94, the number of goal-reached runs and the execution time are similar to each other. So for this project, I choose the discount rate as 0.90.

Discount rate	$\frac{2000}{2000+k}$	
	No. of goal-reached runs	**Execution time (sec.)**
$\gamma = 0.5$	Failed	Failed
$\gamma = 0.6$	Failed	Failed
$\gamma = 0.7$	30.3	0.24
$\gamma = 0.8$	36.8	0.22
$\gamma = 0.86$	35.1	0.20
$\gamma = 0.88$	37.1	0.18
$\gamma = 0.9$	34.4	0.18
$\gamma = 0.92$	38.1	0.29
$\gamma = 0.94$	36.2	0.24
$\gamma = 0.96$	Failed	Failed
$\gamma = 0.98$	Failed	Failed

Table 3 The comparation of different discount rate