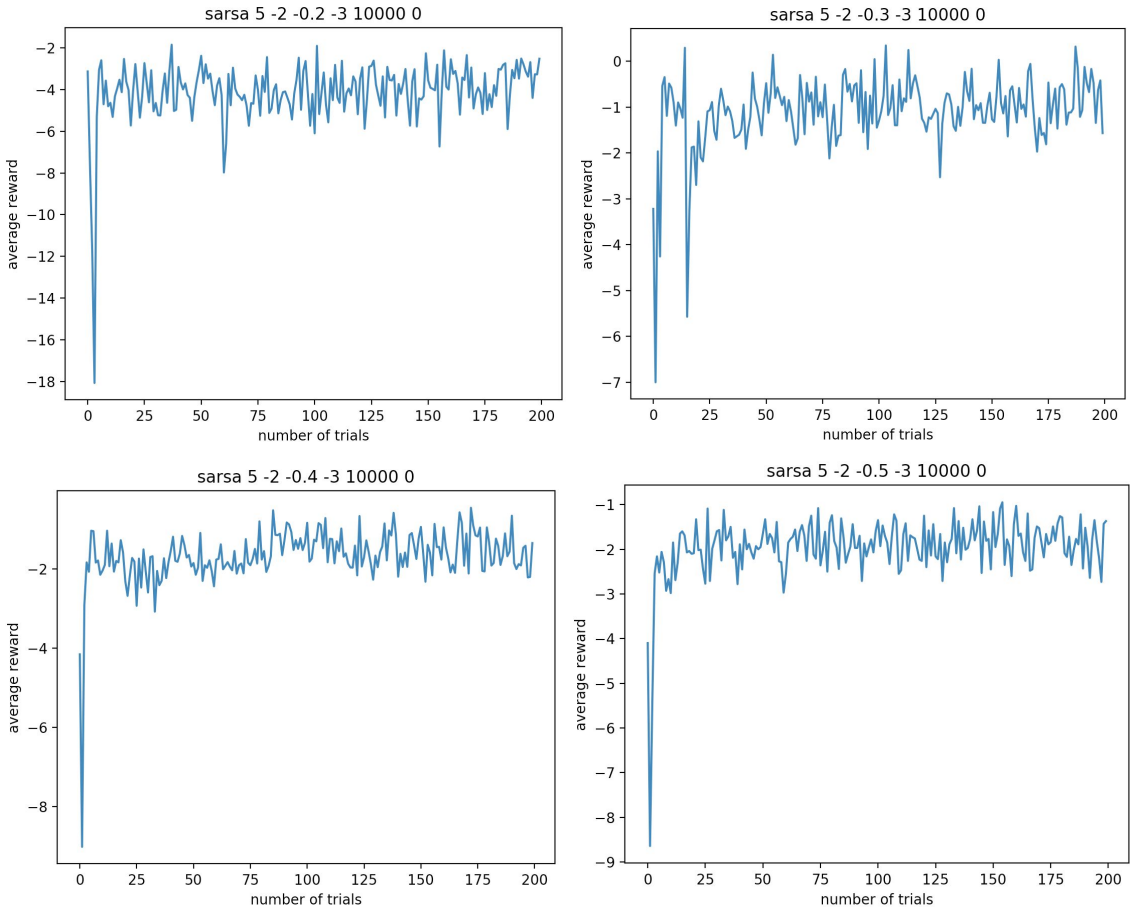
CS534 Spring 18
Assignment 4: reinforcement learning
Mingquan Liu, Daniel Wivagg, Mengdi Li, Ying Fang

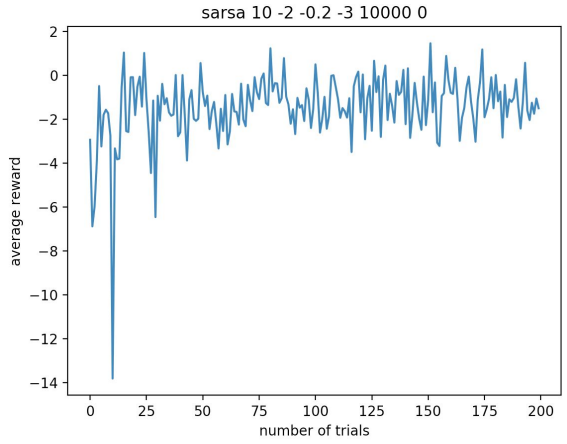
1. We trained our network with epsilon = 0.0 and alpha = 0.5 (stepsize) holding the trials number state as 10000, using different parameters of goal reward, pit penalty, move cost and give up penalty. We plotted the results with number of trials as the x-axis and average reward of 50 trials as the y-axis.

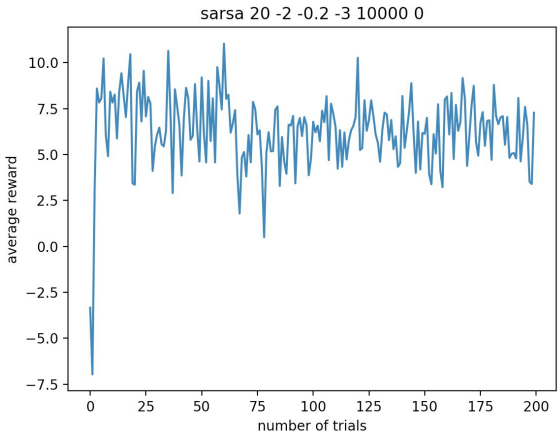
   1.1. Holding goal reward, pit penalty and give up penalty state, only change move cost:



When we set the step cost as -0.1, we found that it would stuck in a trial — it goes around the states greedily according to the SARSA q function, but the function can not take the agent towards to the goal or pit, even choose to give up. We think this is because the move cost is too small to update the q value for each move. On the other hand, the average obtained reward asymptotes well with the step cost of -0.2, -0.3, -0.4 and -0.5. Also, we found that although the reward asymptotes performed good, the map of recommended actions we got does not give a better path.

   1.2. Holding move cost, pit penalty and give up penalty state, only change move cost:

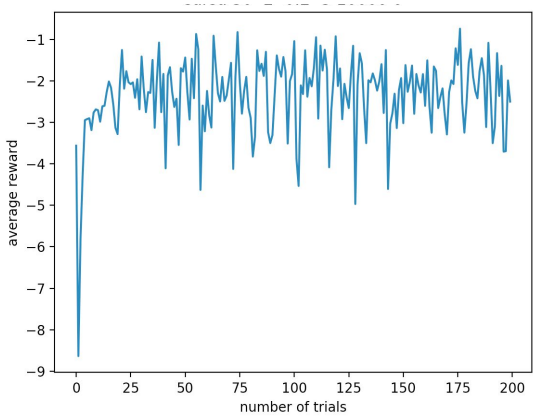




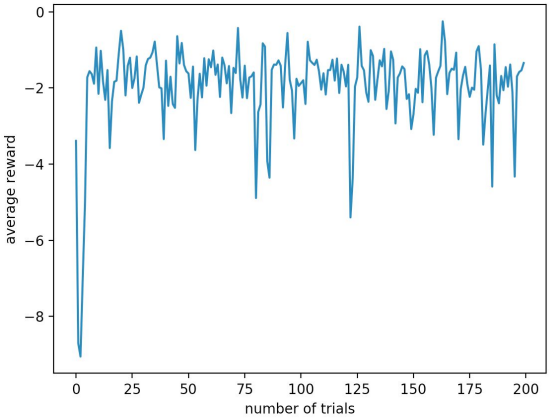
Changing the goal reward larger would not help with getting reward asymptotes a lot.

2. Define “better”: fewer trials to get reward convergence?

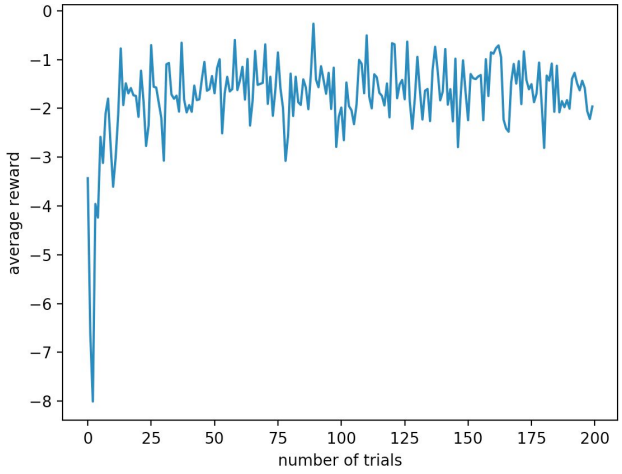
2.1 When we set stepsize as 0.5 and input as *sarsa 5 -2 -0.3 -3 10000 0.1,* We plotted the results with number of trials as the x-axis and average reward of 50 trials as the y-axis. The range of fluctuation is between -1 to -4.



2.2 When we set stepsize as 0.2 and input as *sarsa 5 -2 -0.3 -3 10000 0.1,* We plotted the results with number of trials as the x-axis and average reward of 50 trials as the y-axis. The range of fluctuation is between -1 to -4. But the frequency of great fluctuation of this one is less the former one.

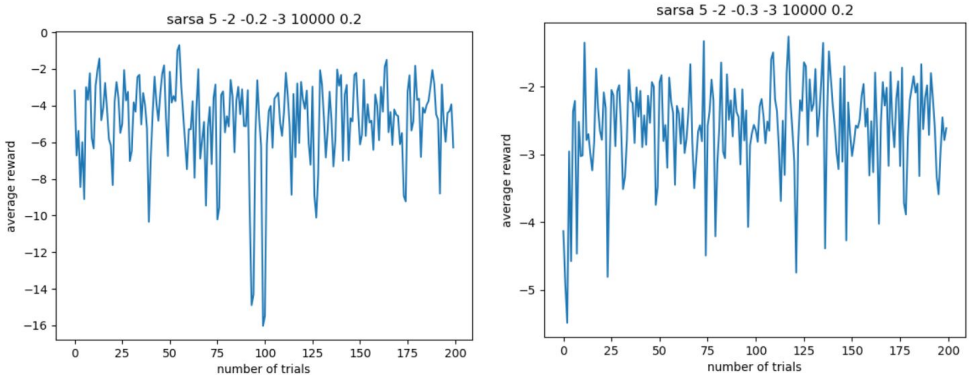


2.3 When we set stepsize as 0.1 and input as *sarsa 5 -2 -0.3 -3 10000 0.1,* We plotted the results with number of trials as the x-axis and average reward of 50 trials as the y-axis. The range of fluctuation is between -1 to -3, which is the smallest one among the three plots.

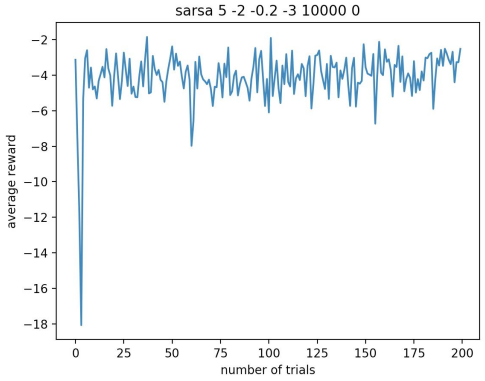


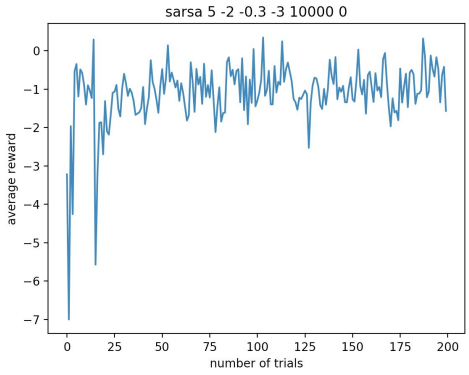
3. Comparing to the graph with the graph from question one which has the epsilon of 0 and step size of 0.5. For mine graphs below, they have the epsilon of 0 and step size of 0.5. The performance of the epsilon of 0.2 has more variation comparing to epsilon of 0 when the number of trials gets larger, but it has a better performance at the beginning, which was shown by less fluctuation at the beginning. Since our step penalty is kind of big, our reward values tend to be negative.

Graph for Epsilon of 0.2:



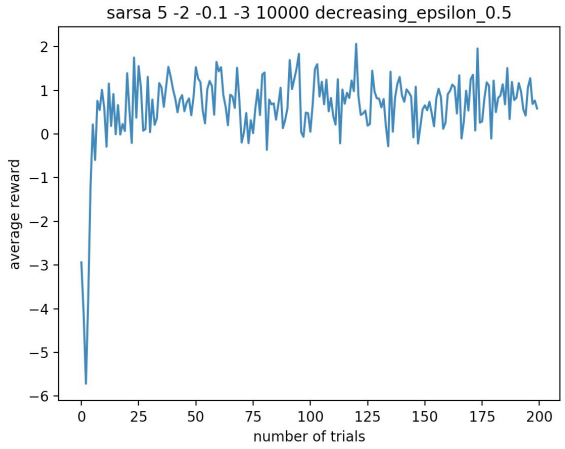
Graph for Epsilon of 0:

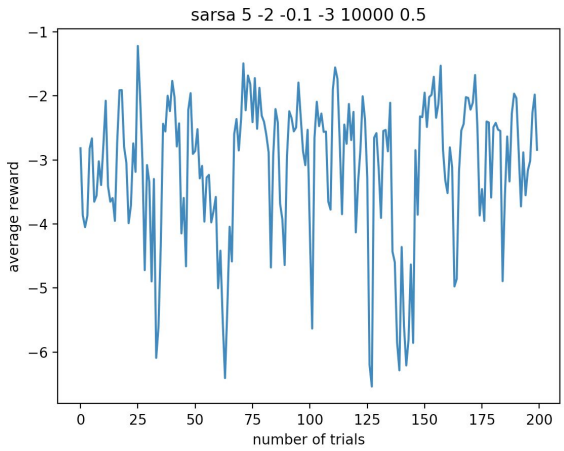


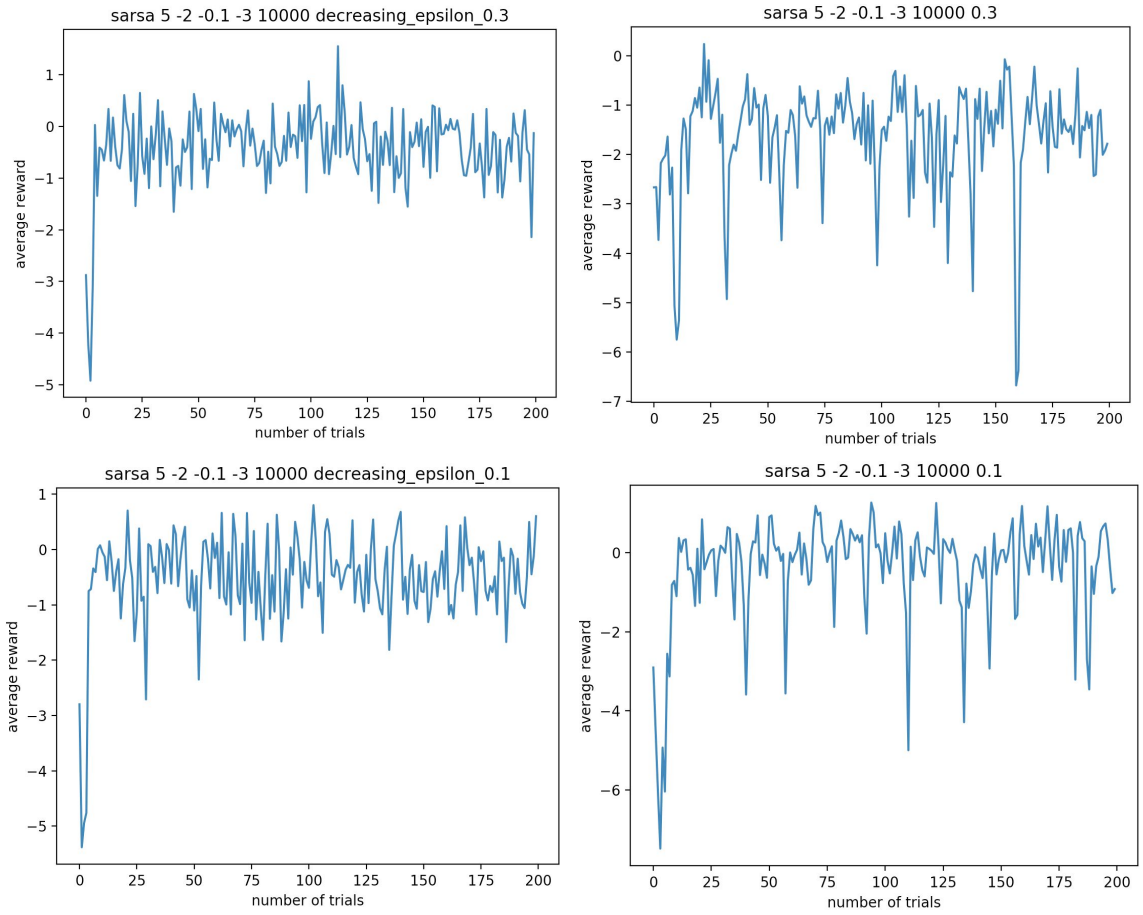


As for conclusion, we think the better result is depend on how we evaluate them.
On one side, if we consider the convergence in the late trials, the epsilon with 0 definitely has a better convergence in reward value.
On the other side, if we consider the range of the rewards, the graphs of epsilon of 0.2 has a smaller range in rewards.

4. We came up an exploration scheme that set a larger epsilon value at the beginning of the training, as the number of trails growing, make the epsilon value decrease gradually. So, it will explore much brave at first, then, do less exploration gradually during the training process. We plotted the results of training with this exploration scheme and compare with the results of training with epsilon value as a constant, as shown below.







The larger epsilon value will help with finding a good path quickly. As shown in graphs, when we set the epsilon decreasing over time during the process, the agent will do less exploration in the latter of the process, thus, the average rewards will become more stable in the latter trails in training.

5. Answer are for each sub question

5.1. When we set the input as *sarsa 5 -2 -0.1 -3 10000 0.1* and set gamma= 0.9 and stepsize = 0.1, we get a policy where never go into a pit or give up.

> | > | > | v | < | < |

^ | ^ | ^ | v | < | < |

^ | ^ | p | p | v | ^ | ^ |

^ | p | g | < | < | p | ^ |

^ | < | p | p | p | > | ^ |

^ | < | < | > | > | > | ^ |

```
G | > | v | v | v | G | G |
v | > | v | v | < | < | v |
> | v | p | p | < | < | v |
> | p | g | < | < | p | < |
^ | > | p | p | p | < | < |
^ | ^ | > | ^ | ^ | < | ^ |
```

5.2. when we set the input as *sarsa 5 -2 -1 -0.5 10000 0.1* and set gamma = 0.9 and stepsize = 0.4, we get a policy where states' best action have give up, goal or pit.

Extra Credit: We initialize our q value with the rewards of that action when the first time we do the update Q, because they are initially empty or 0 when it starts.