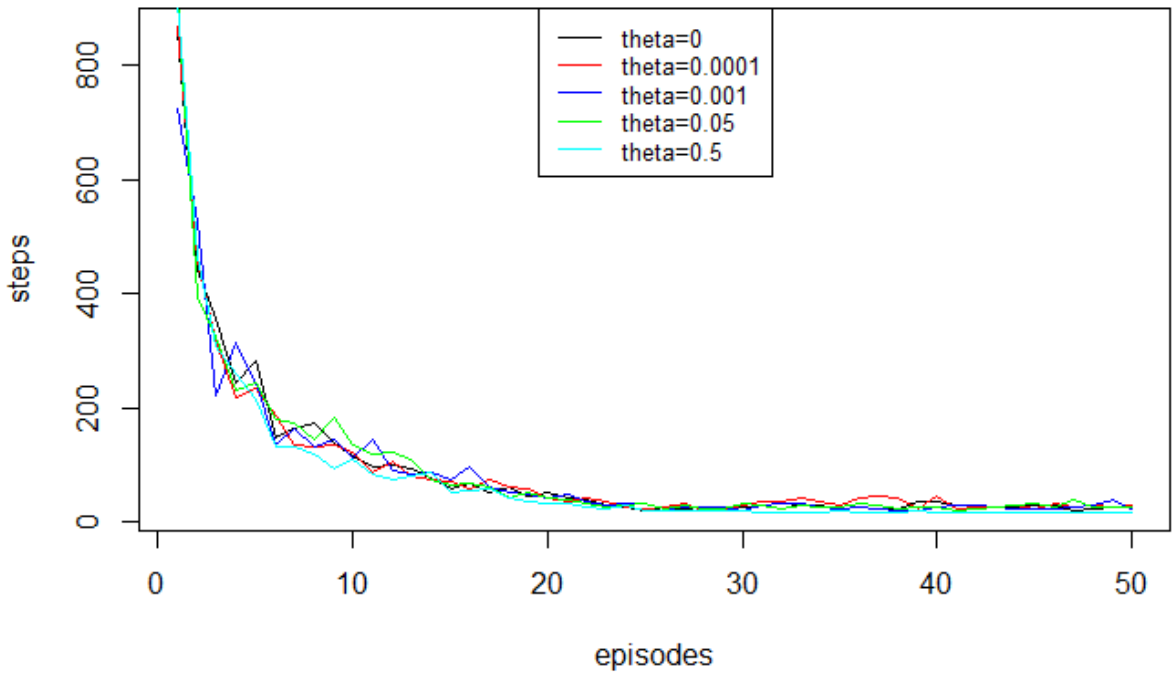
Assignment 4

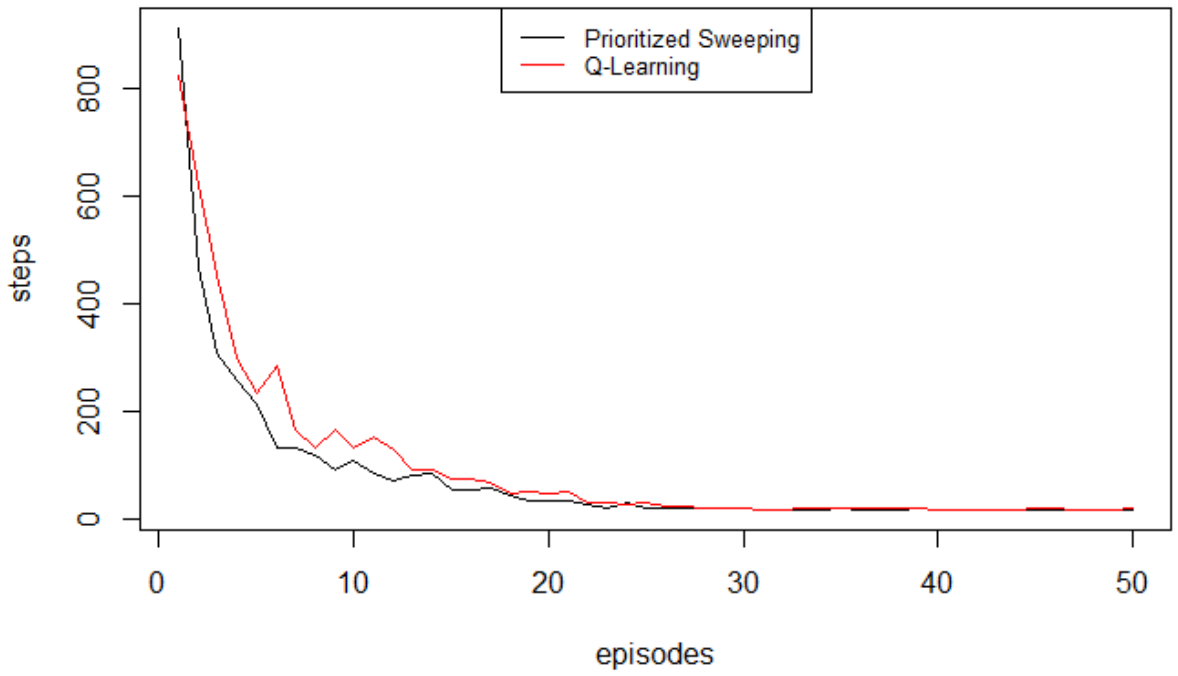
1) The book explains planning as a computational process that takes a model as input and produces or improves a policy for interacting with the modeled environment. A model of the environment is anything that an agent can use to predict how the environment will respond to its actions. Given a state and an action, a model produces a prediction of the resultant next state.

   In Dyna we learn model from experience and use planning to construct value functions, which in turn can be used to generate policy. This falls under the definition of planning methods mentioned above, i.e. use model to produce or improve policy.

2) I used different values of theta to check which seems to perform better. The plot shows the number of steps in each of the 50 episodes for different values of theta. There isn't much difference between the plots. But theta = 0.5 seem to perform the best.



This plot shows the comparison of DynaQ with Q learning. Prioritized Sweeping seems to converge faster as expected. But still it's not converging as fast as shown in the book.

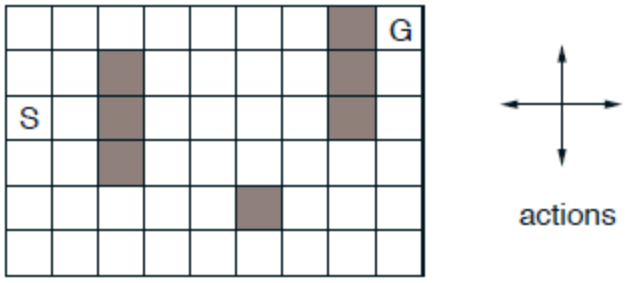


Question

In this experiment we try to identify the effect of alpha in DynaQ with prioritized sweeping applied to a grid world problem as described in Example 8.1 of Reinforcement Learning: An Introduction 2nd edition. In the book, Figure 2.6 describes what we might expect for behavior of alpha. To put simple, we can expect the performance increases as alpha increases and after certain point it decreases. It is like a bell curve.

Problem

The figure below taken from the book (Figure 8.3), represents the grid world we will be working on. The agent can move in 4 directions up, down, left, right. State transitions are deterministic, up action takes the agent to the square on the top, however if the agent was to try and go out of bounds or the grey blocks the agent remains in the current square.

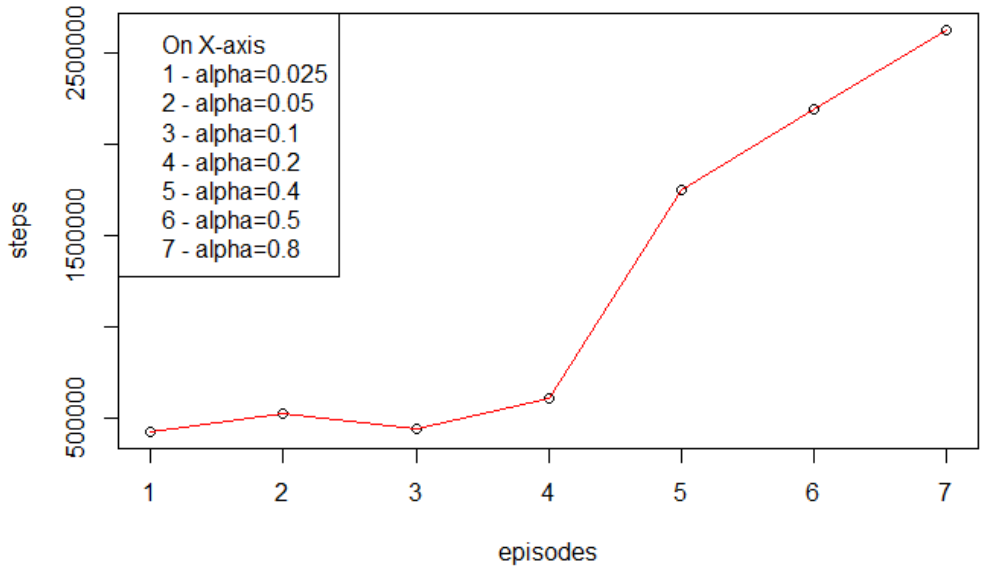


Algorithm

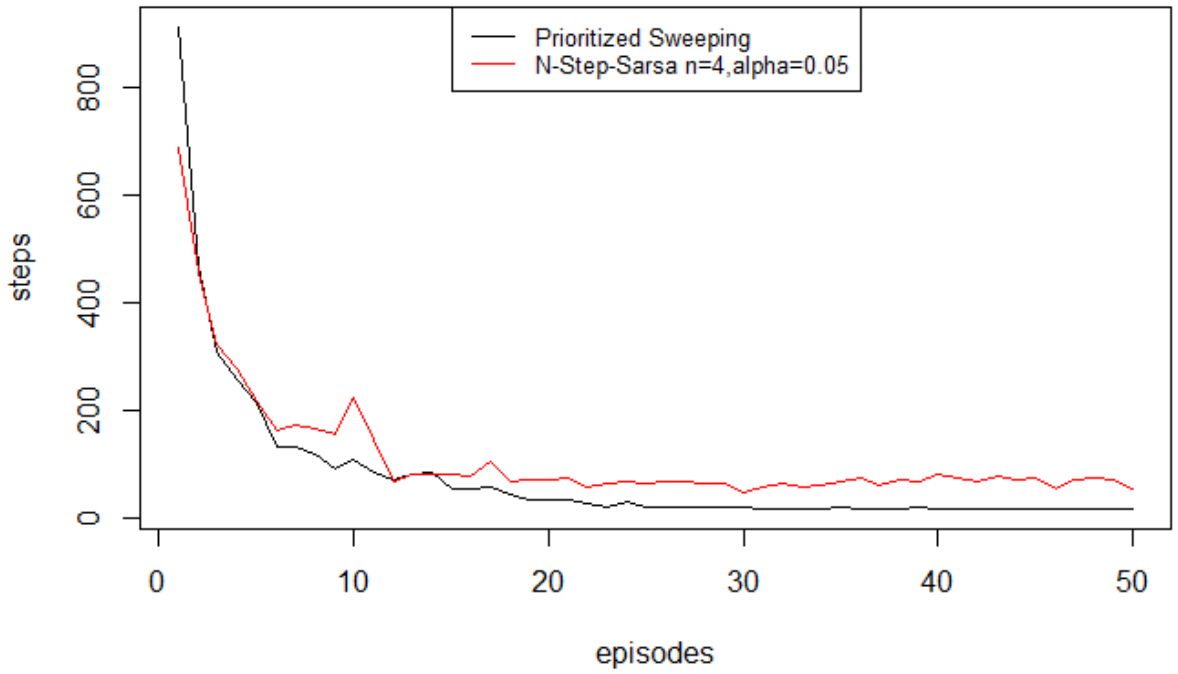
We use DynaQ with prioritized sweeping. DynaQ includes planning, acting, model learning, and direct reinforcement learning. We construct model from the experience, and then use this model in calculating action values.

Results and Conclusions

The plot shows the total steps to goal summed over 50 episodes (averaged over 30 runs) on the y-axis, verses value on the x-axis. From the plot we can see that as we increase value of alpha the number of steps taken to reach the goal increases, this is because due to high alpha the action values oscillate and takes more steps to converge. The values of alphas 0.025, 0.05 and 0.1 seem to have good performance. From the figure 2.6 in book we can expect performance to decrease as alpha becomes very small, because it takes a lot of steps as the step size is very small. But for the alphas we used in this experiment the performance has not reduced yet. Perhaps using even lower values of alpha would show reduced performance.



This plot shows comparison the N step to DynaQ with prioritized sweeping. For the n step sarsa, n=4 and alpha = 0.05 worked best of n = {1, 2, 4, 8, 12} and alpha = {0.01, 0.05, 0.2, 0.4, 0.5}.



I think it won't be possible to make n step sarsa outperform DynaQ with prioritized sweeping. This is because in N step methods we can go back up to n steps and update all action values. It usually updates action values from the end of the episode to the front, since we have reward = 1 for only reaching terminal state. But in case of prioritized sweeping we would be updating action values of important states and actions according to a priority.

But if the step size of N step methods is significantly larger than the planning steps in Dyna, I think the N step method might perform better.