Luke Davidson

CS 5180

Ex5

1.) A scenario opposite that of the one described at the end of exercise 6.2 may initially be more suitable for a MC approach than TD. The situation described is, “Suppose you have lots of experience driving home from work. Then you move to a new building and a new parking lot (but you still enter the highway at the same place). Now you are starting to learn predictions for the new building.” What if we consider the opposite, where you still have a lot of previous experience driving home from work, but instead of changing jobs (change starting state), you move homes to a new home in the same neighborhood (change end state). After a lot of experience driving home, it is assumed that a MC will converge to a pretty good estimate. After the house move, only the last state estimate in your commute will be changed. Since MC waits until the episode is complete, works backwards from the end state and only makes use of a true sample of what Q(S, A) should be, one could imagine that at least initially, the changes to the end state by moving houses to a different house in the neighborhood may be slightly more accurate than the changes of estimates by TD. Although, as the number of times the agent drives to the new house (number of episodes) increases, TD will converge to a more accurate estimation than that of a MC approach.

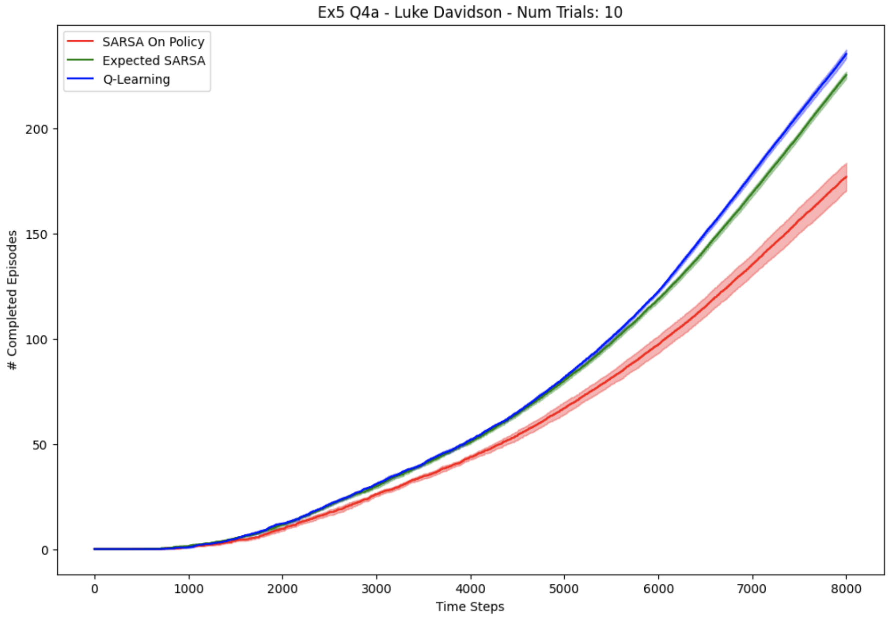
2.) a.) Q-learning is considered an off-policy control method because it is completely independent of the policy being evaluated. It aims to estimate q*, the optimal state-action value function, which is independent of any policy pi.

b.) Even if action selection is greedy, Q-learning will still not be exactly the same as SARSA. At first it seems like it will be since Q-learning uses the max of Q(S’, a) which will be the same for a greedy selection of a, although the difference is where in the algorithm the actions to take are selected. For SARSA, A’ is selected just before the incremental update of Q, and for Q-learning, the selection happens after the update of Q. This will lead to slightly different convergences between the two due to slightly different Q functions, although the convergences will be much similar in a greedy policy than an e-greedy policy.

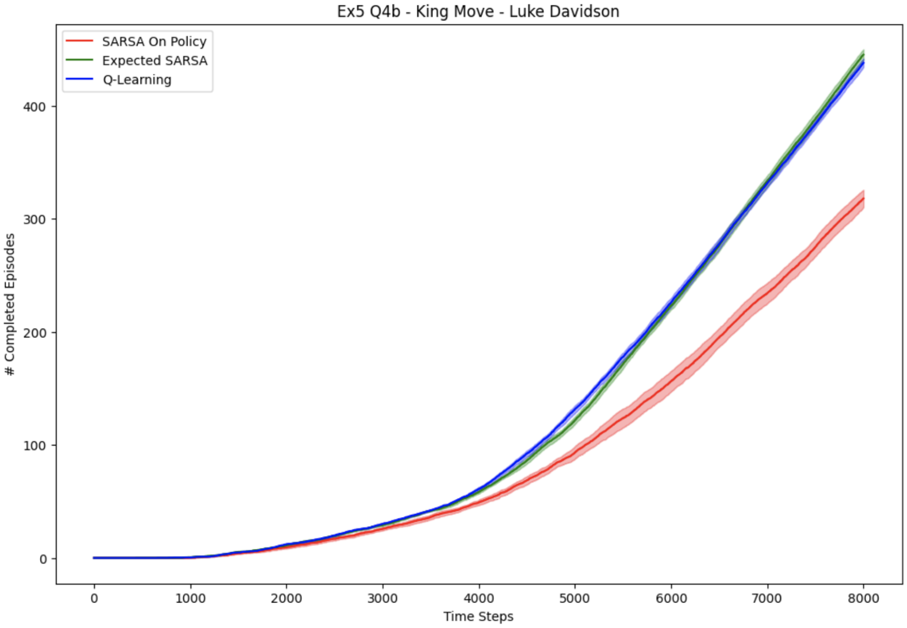
3.) a.) I do not believe the value of alpha will change the general conclusion of which algorithm is better, mainly due to the variability in MC. For TD, it is pretty clear that as alpha increases, the TD estimation learns faster yet yields a higher error in the long run. However for MC, a pattern for varying alphas is not as clear. Initially it seems that an increase in alpha leads to better performance (faster learning, lower error), although from alpha = 0.3 to alpha = 0.4 we can see that the error does in fact rise and the estimation has a much higher variability. Therefore, I don't think we can conclude that there is a simple answer to whether there is a better algorithm given varying alphas, as the variability will continue to increase as alpha increases. For alpha = 0.5, it appears that TD will perform significantly better than MC.

b.) In looking at the Q function update equations, alpha will essentially amplify the error in estimates. For a higher alpha, the error will be more amplified, causing the increase in variability we see in the graph for alpha going from 0.1 to 0.4. This will in turn lead to convergence issues with high alpha values, disallowing correct convergence to optimal values of V/Q. The decrease then increase behavior we see in the figure likely represents the point at which the alpha value was high enough, and therefore reached a high enough level of variability, to disrupt the natural loss of RMS error.

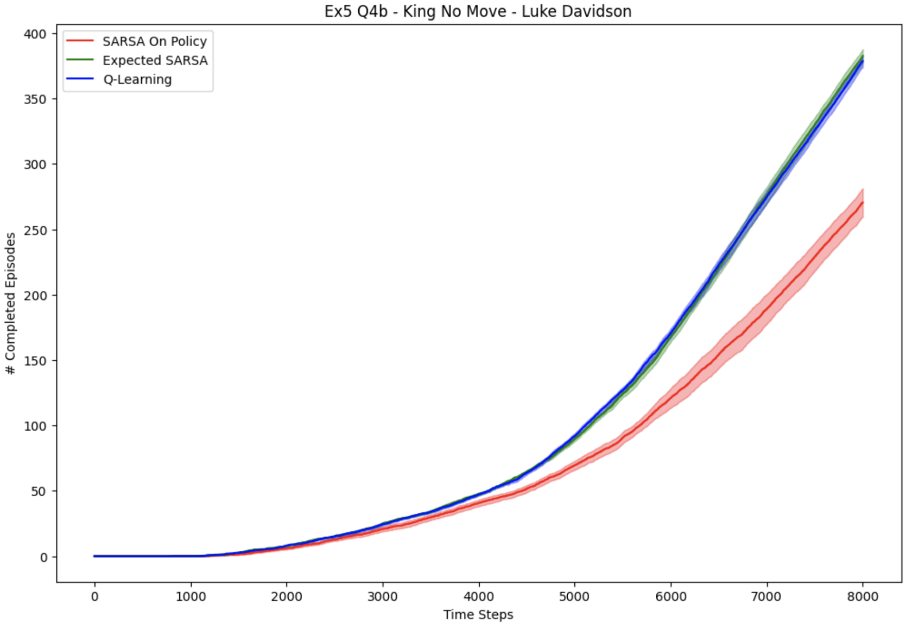
4.) a.) SARSA, Expected SARSA, and Q-learning plots reproduced over 10 trials. Specific parameters are specified in the code.



b.) SARSA, Expected SARSA, and Q-learning plots reproduced over 10 trials for the King enabled, no stay environment. Specific parameters are specified in the code.



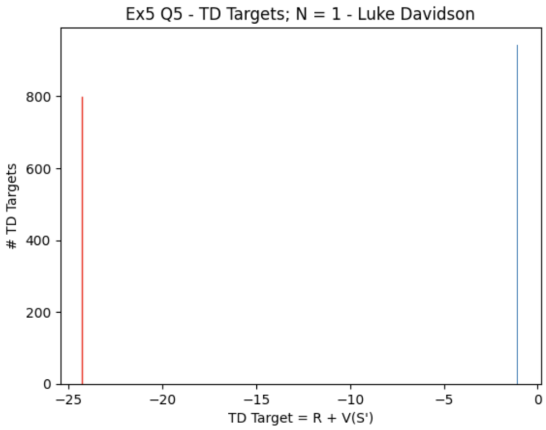
SARSA, Expected SARSA, and Q-learning plots reproduced over 10 trials for the King enabled and “stay” action enabled environment. Specific parameters are specified in the code.

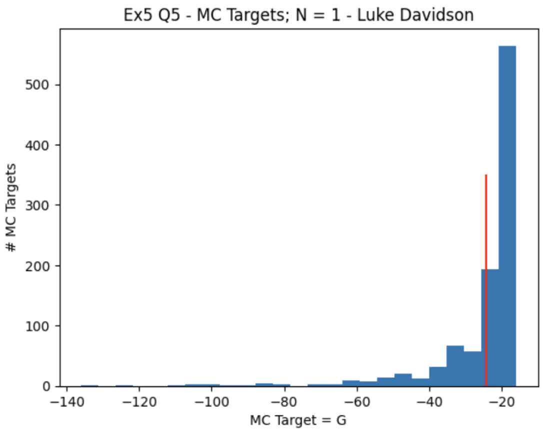


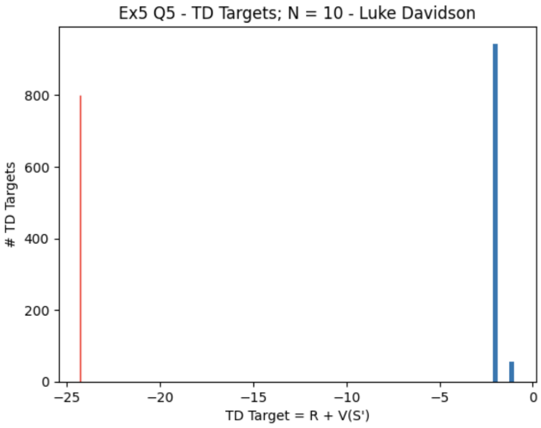
It is evident that higher levels of success can be reached with king movement enabled, with no stay action compared to no king movement because the agent can essentially perform 2 actions in the timestep of 1, causing much quicker episodes once a good policy is learned.

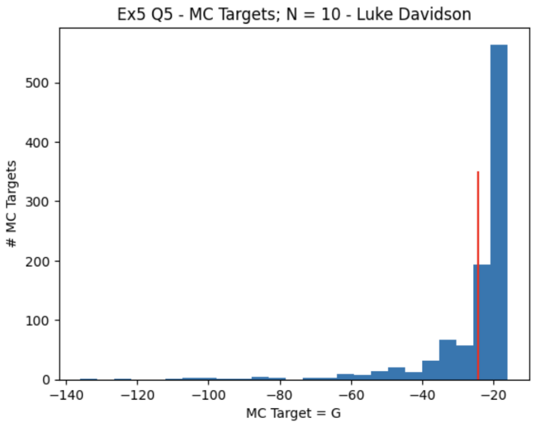
Lower levels of success were seen when the “stay” action was inputted into the environment compared to when it wasn’t in the environment. This is likely because a no movement still receives a reward of -1 and doesn’t allow us to make Q function updates of high importance. It is essentially a wasted move.

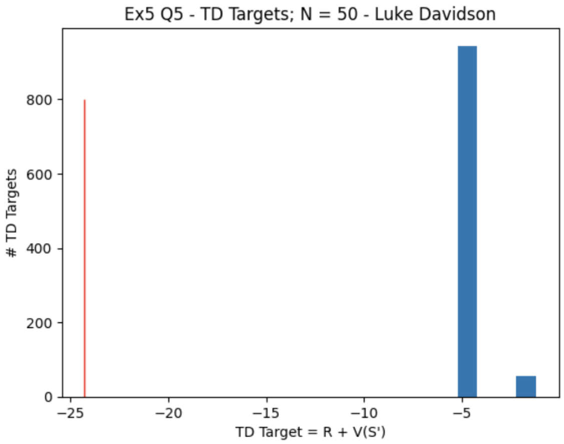
5.) a.) The resulting histogram plots for TD and MC for episodes N = {1, 10, 50} are shown below for a set of 1000 test episodes, with V* shown in red.

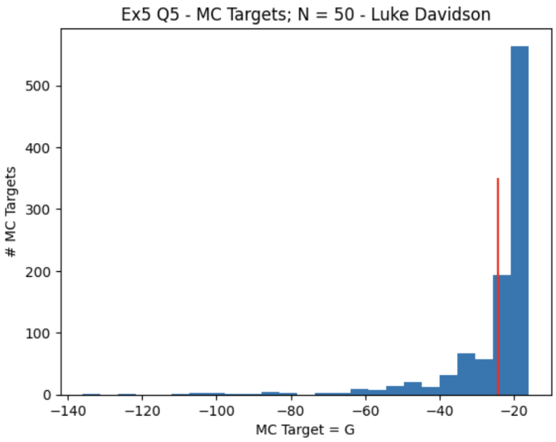












5.) b.) As one can see, the above plots represent the bias in Temporal Difference control versus the variance in Monte Carlo control. Starting with the Temporal Difference plots, it is evident that even though I used 8 bins to bin the 1000 targets, they only fell into two of those eight bins for all three state-value functions for N = {1, 10, 50}. It is also clear that around 85% of the 1,000 samples fell into one bin. This bin is located at around -1.1, -2.5, and -5 for the three N values. This represents the high bias towards the initialization value of V, in this case 0, with very little variance (all values nearly the same so they all fall into the same bin). It is also worth noting that as N increases, specifically as N increases to infinity, this bias will diminish and the target values will approach the true value of V*(S). This is seen with -1.1, -2.5, and -5 approaching the true value of about -24 as N increases from 1 to 10 to 50.

On the contrary, the MC plots show high variance with little bias towards the initialization values of V(S). All three plots are actually the same because the targets for MC only depend on the returns of each episode, not on the different state-value functions found with the increasing episodes. It is very clear that the 1,000 target values fell into many different bins compared to that of TD, representing the high variance of the results. The bins, however, are surrounding the true value of V*(S), representing the low bias towards the initialization of V(S), also 0 in this case. Unlike TD with diminishing bias as training increases, this high variance and low bias will remain the same.

c.) The results for using control during the training phases will depend on whether we start with an arbitrary policy or with the almost optimal policy like we did in the previous parts of this problem. If we start with an arbitrary policy, we can anticipate that the results will be different for a small number of training episodes (ex. 1, 10). This is because the arbitrary policy will take many more timesteps to reach the goal, resulting in an increase in probability that the same state is visited multiple times. This extra visitation will cause a difference in the state-value function estimation compared to that of an almost optimal policy, thus leading to different target values. As the number of training episodes increases, say to infinity, the results will eventually become the same once the same V*(S) is reached. If we were to begin with an almost optimal policy like we did in this problem, we would expect the results to be similar, even for a low number of training episodes. Since V(S) is almost optimal to begin with, the increases in accuracy to V*(S) due to control and policy iteration will likely be relatively negligible compared to the already almost optimal policy.

We will also assume the same for MC, although mainly due to the length of episodes for an arbitrary policy versus an almost optimal policy rather than the difference in the V(S) estimate. With an arbitrary policy, the amount of steps in an episode will greatly increase compared to the amount of steps for an almost optimal policy, especially for low trial episode runs. This will lead to very different return values, G, for the first state, which directly affects the computation of target values for MC.