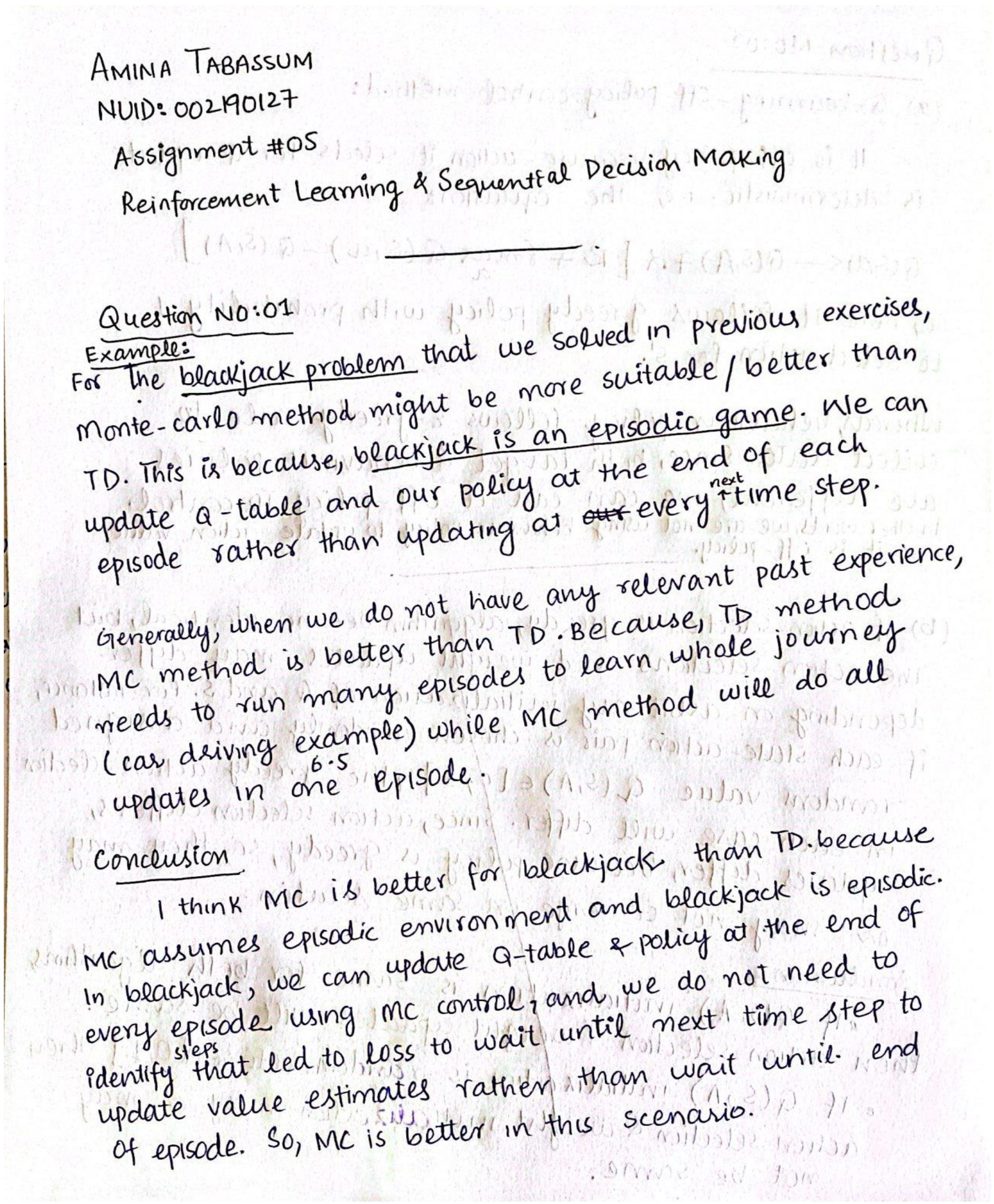
Amina Tabassum

NUID:002190127

Exercise 05



AMINA TABASSUM
NUID: 002190127
Assignment #05
Reinforcement Learning & Sequential Decision Making

Question NO:01

Example:
For the blackjack problem that we solved in previous exercises, Monte-carlo method might be more suitable / better than TD. This is because, blackjack is an episodic game. We can update Q-table and our policy at the end of each episode rather than updating at ~~our~~ every next time step.

Generally, when we do not have any relevant past experience, MC method is better than TD. Because, TD method needs to run many episodes to learn whole journey (car driving example) while MC method will do all updates in one 6.5 episode.

Conclusion

I think MC is better for blackjack than TD. because MC assumes episodic environment and blackjack is episodic. In blackjack, we can update Q-table & policy at the end of every episode using MC control, and we do not need to identify steps that led to loss to wait until next time step to update value estimates rather than wait until end of episode. So, MC is better in this scenario.

Question NO:02

(a) Q-learning - Off policy control method:

It is off-policy because action it selects for Q-update is deterministic. i.e, the equation:

$$Q(S,A) \leftarrow Q(S,A) + \alpha \left[R + \gamma \max_a Q(S',a) - Q(S,A) \right]$$

so, here it follows greedy policy with probability 1 to select action for S'.

Whereas, behaviour policy follows ε-greedy rule to collect data. Since, both target & behaviour policies are different, we can call it Off-policy TD-control.
In other words, we are not using behaviour policy to update action value, so, it is off-policy.

Question NO:02 (b)

If action-selection is greedy, still these algorithms will be different. Because, in case of SARSA, we select next action A' according to greedy policy, then update next Q-value i.e,

$$Q_{t+1}(S,A) = Q_t(S,A) + \alpha \left[R + \gamma Q_t(S'A') - Q_t(S,A) \right)$$

where
Q_t = state-action value function at time t.

Whereas, in case of Q-learning, we first update our state-action value function:

$$Q_{t+1}(S,A) = Q_t(S,A) + \alpha \left[R + \gamma Q_t(S',A') - Q_t(S,A) \right)$$

and select next action according to greedy policy.

So, difference is in order of update & action selection. Hence, two algorithms will still be different.

Question NO: 03

(a) It appears from plot that:

TD methods:

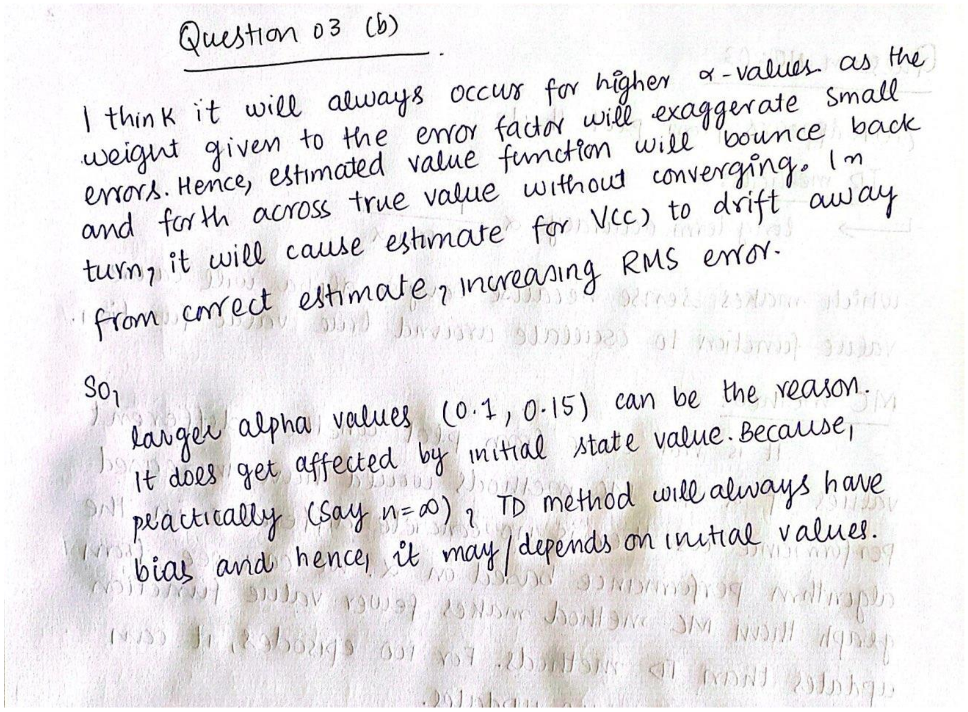
$\hookrightarrow$ long term accuracy $\propto \frac{1}{\text{chosen } \alpha}$.

which makes sense because larger alpha will cause value function to oscillate around true value function.

MC method:

It is not clear from plot whether different values of α for MC methods would have increased performance as ther is no concrete difference in the algorithm performance based on α. We can see from graph than MC method makes fewer value function updates than TD methods. For 100 episodes, it can make maximum 100 updates.

For TD methods, episode length is 6. So, there are updates at every timestep. Hence, it makes 600 value function updates in 100 episodes. No value of α can overcome reduced sample rate. So, short answer is. No, for long term accuracy, smaller value of α is more likely to converge to optimal value. For MC, $\alpha = 0.01$ and for TD: $\alpha = 0.05$, so, both methods are likely to converge. Chosing smaller alpha will not make huge difference. Nor does it will affect ~~of~~ ~~of~~ which algorithm will perform better.


Question 03 (b)

I think it will always occur for higher α-values as the weight given to the error factor will exaggerate small errors. Hence, estimated value function will bounce back and forth across true value without converging. In turn, it will cause estimate for V(c) to drift away from correct estimate, increasing RMS error.

So,
larger alpha values (0.1, 0.15) can be the reason.
It does get affected by initial state value. Because, practically (say $n=\infty$), TD method will always have bias and hence, it may/depends on initial values.

Question No: 04

The resulting plots for sarsa,q-learning and expected sarsa are shown in figure below.

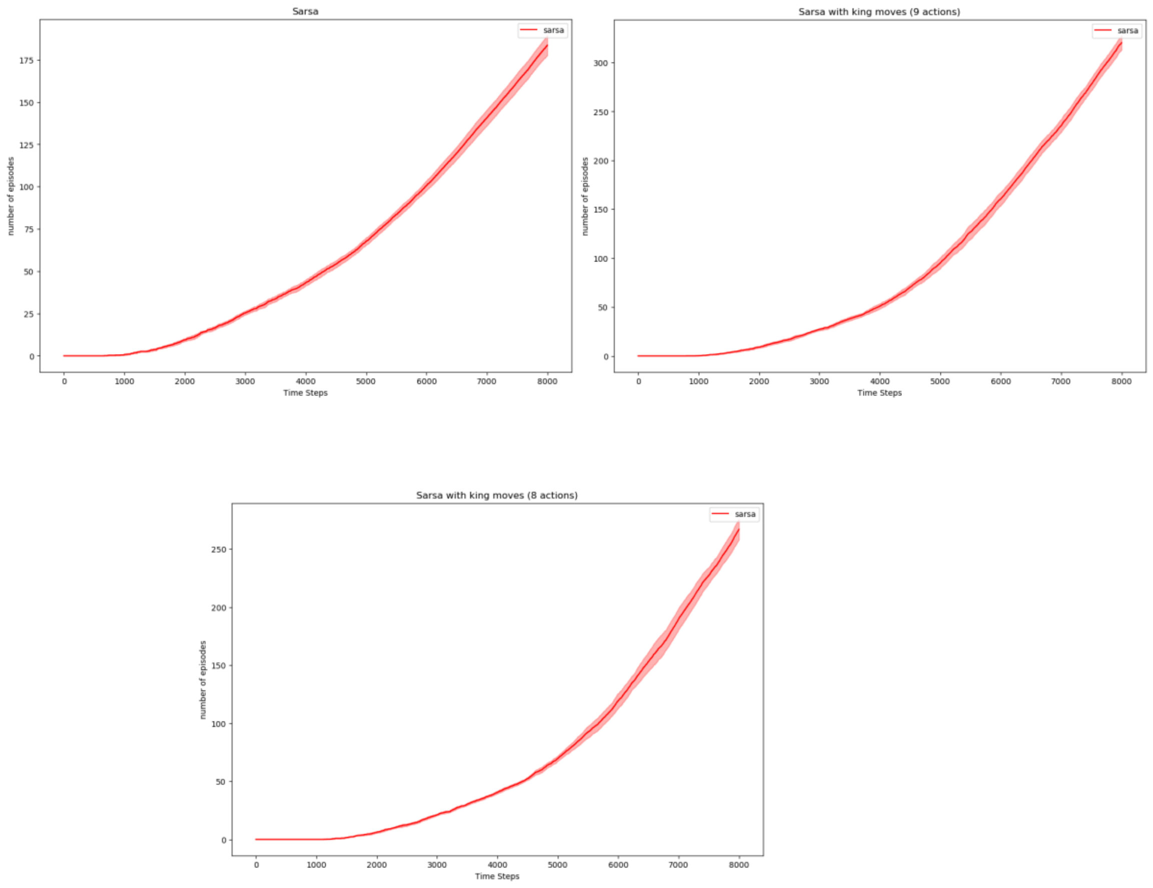


Figure 1: Plots for Sarsa without king moves, with king moves (considering no move action and without no move action) from left to right

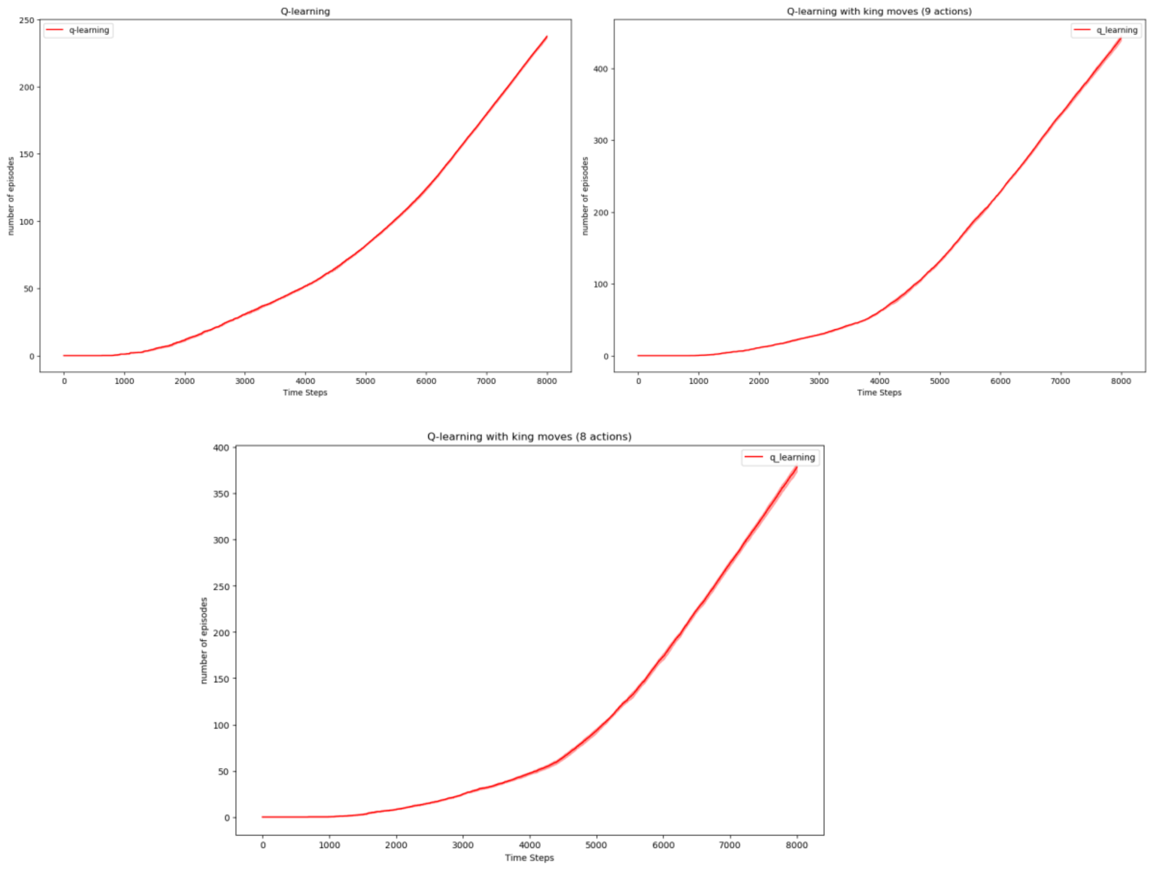


Figure 2: Plots for Q-learning without king moves, with king moves (considering no move action and without no move action) from left to right

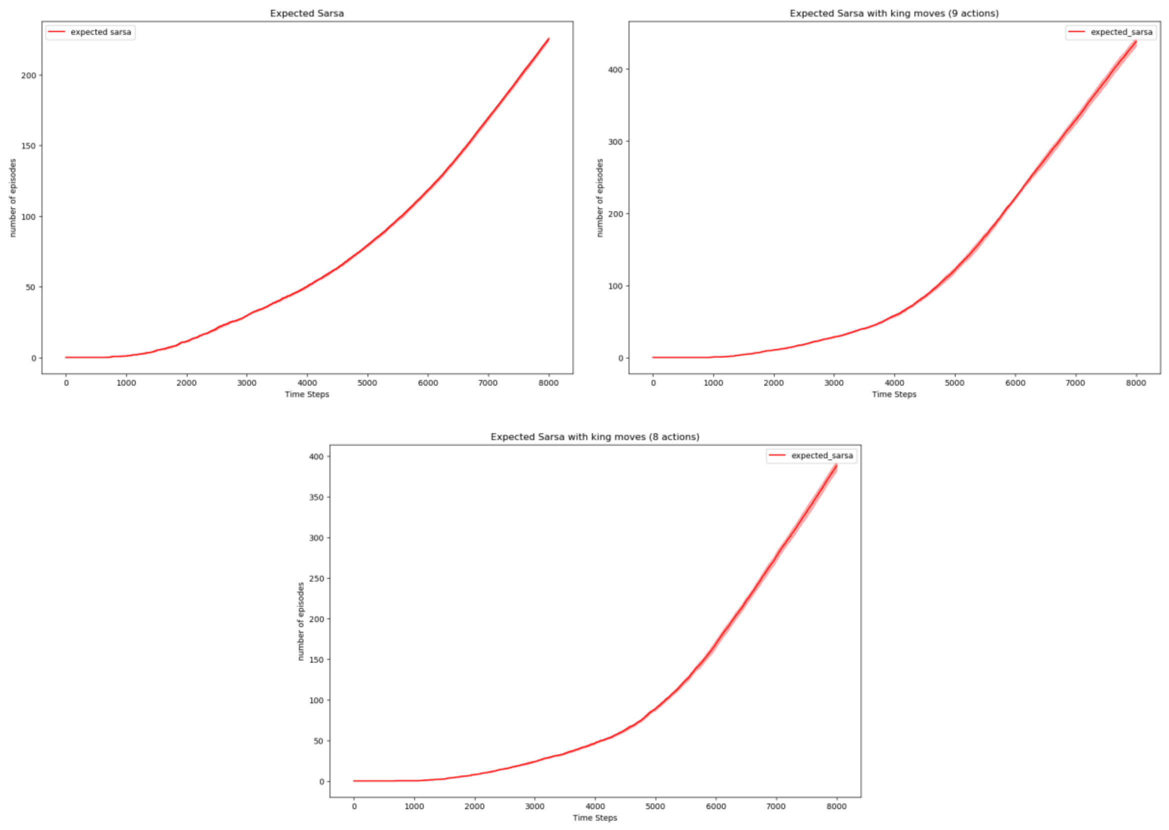
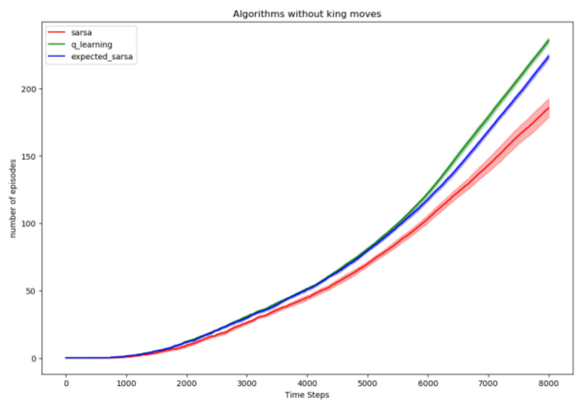
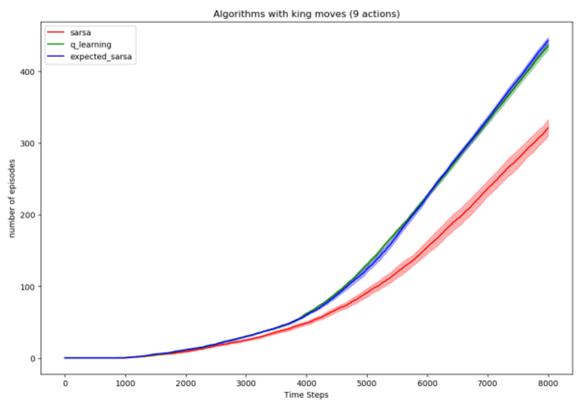


Figure 3: Plots for Expected Sarsa without king moves, with king moves (considering no move action and without no move action) from left to right





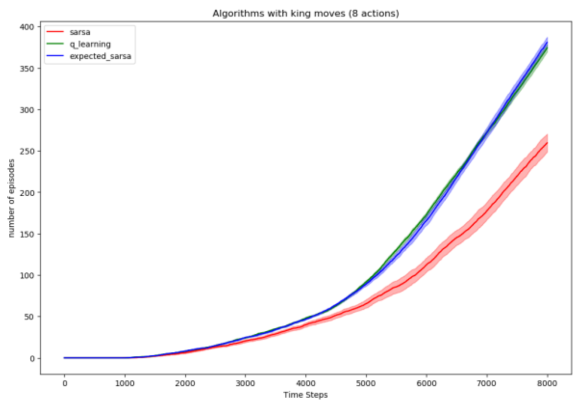


Figure 4: Combined plots without king moves, with king moves (considering no move action and without no move action) from left to right

The above figures demonstrate results of applying Sarsa,Q-learning and expected Sarsa algorithms on windy grid world environment with e=0.1 and alpha=0.5. There are four standard actions left,right,up and down. But the king's moves introduce additional diagonal actions. The increasing slope of plots demonstrates that goal was reached more quickly over time. So, based on this criterion, Q-learning performed better than the other two algorithms. Moreover, Q-learning has the highest number of episodes in the 8000 steps timeframe. So, generally for these three algorithms, the trend is:

Q-learning>> Expected Sarsa >> Sarsa

Switching action space to king's moves, it can be observed from above plots that performance improved for all algorithms. The number of episodes they explore in 8000-time steps increases and slope of all the plots has also improved. Statistically, performance improvement measures for: Sarsa is approximately *20%, expected sarsa approximately 50% and Q-learning approximately 85%.*

Before changing, the optimal policy was comprised of 17 steps. Switching to king's moves without no move step reduces the number of steps for optimal policy to be 7. Hence, the number of episodes in 8000 timesteps increases gradually and hence the goal was reached more quickly. So, consequently, the slope increased. Comparing these three algorithms, the pattern is:

Q-learning >> Expected Sarsa >> Sarsa

According to my observation and opinion, including 9th action (no move action) will not further improve performance because staying at same location will result in longer episodes or the best it can move is one or two steps by wind and reach goal state, but agent can always move left, right or up to make it even closer to goal state.

Question No: 05

a)

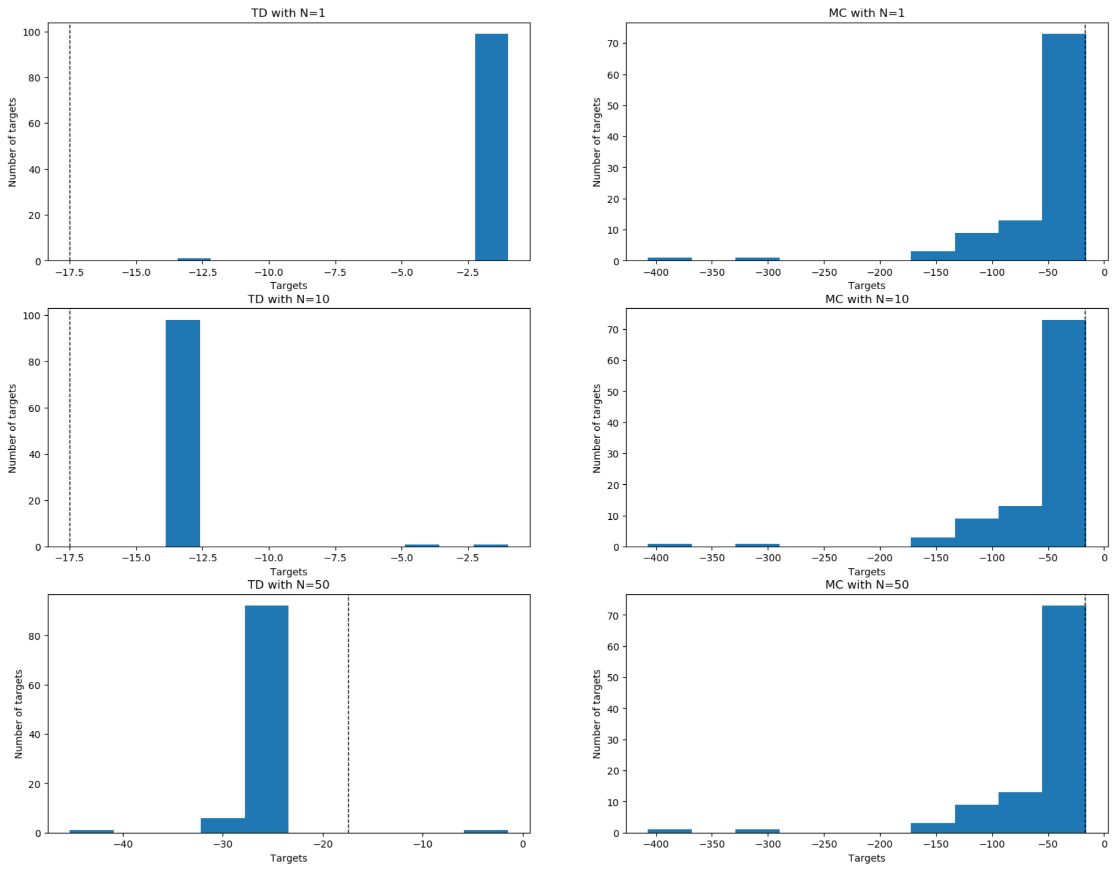


Figure5: Histograms to observe bias-variance tradeoff of MC and TD

b)

Since Monte Carlo has low bias and high variance, it provides low bias and large variance during training phase. Whereas TD (0) provides large bias and low variance. So, MC provides accurate value estimate whereas TD(0) provides stable value estimate during training. As the number of training episodes is increased, the bias for TD (0) decreases. Monte Carlo does not get affected by training set because it does not use bootstrapping. So, increasing N value has helped to reduce TD (0) bias and converge to the true value asymptotically.

c)

Considering the controls scenario, I used SARSA and MC for initial online action-value function estimation, then update policy and further use this updated policy to generate next episodes. My initial opinion was that in this scenario, the results will not get affected despite differences in internal representation of environment. It can be observed from the following figure that

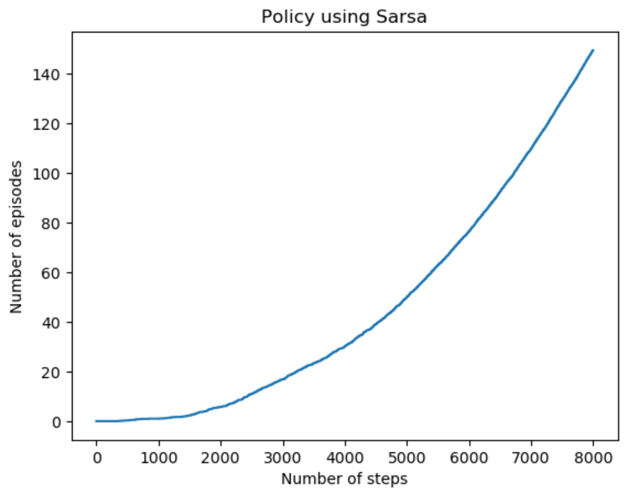


Figure6: Online Policy plot

Initially the policy explores more as compared to off-line policy. Once it learns about the environment, it ultimately converges to same optimal path. I used 8 actions (king's movement case) . Moreover, it can be observed from the following figure that in TD case, as we increase number of evaluations, the bias increases as compared to the offline policy case were bias decreases with increase of number of evaluations. This is because, here we don't have fixed policy. We are estimating both policy and value function to optimal policy and optimal value function. Since, in this case, agent is exploring hence, there are chances that bias increases with increase in explorations steps. Variance is low in TD case. Coming towards the MC case, there is slight variation in variance as compared to bias.

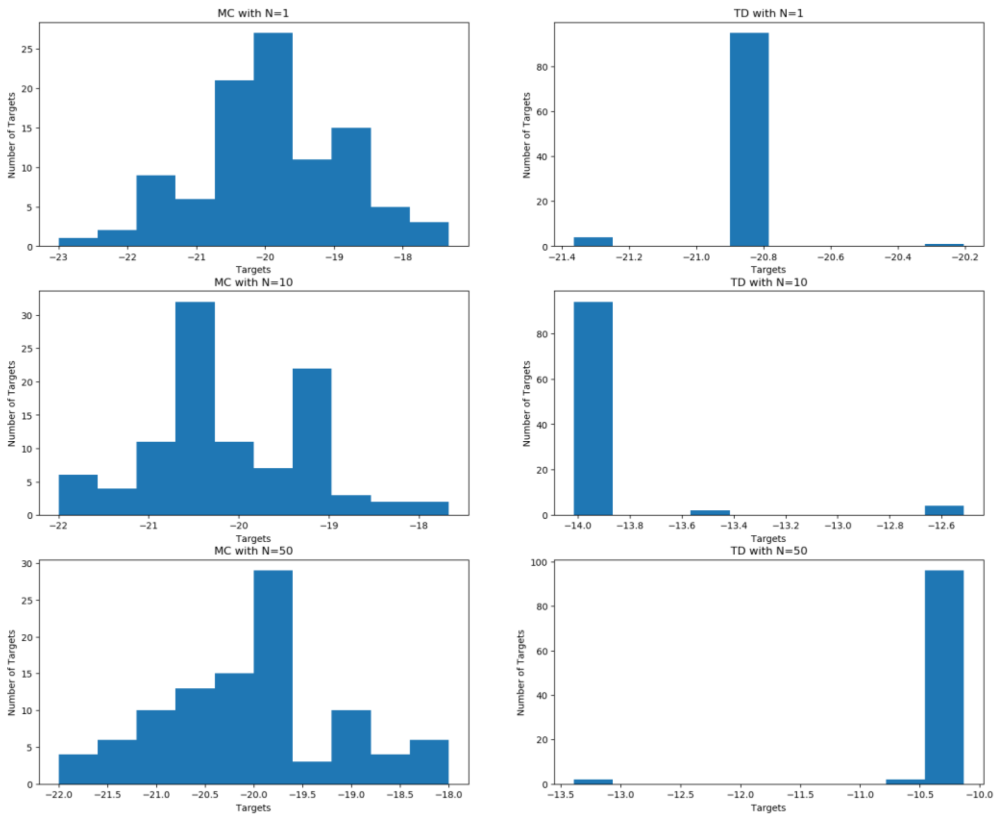


Figure6: Histograms to observe bias-variance tradeoff of MC and TD

In order to observe the convergence, I plotted heat map and again it can be seen from heat map as well that despite differences in environments, both prediction and control eventually converge to find same optimal path as shown below:

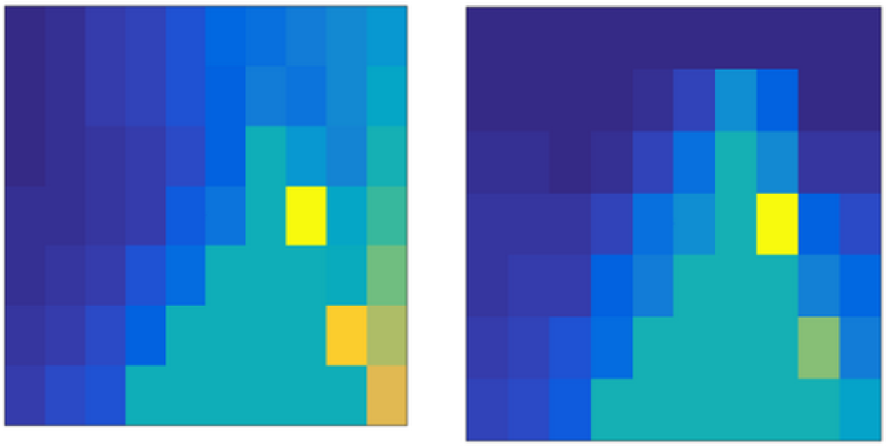


Figure7: Prediction (Offline policy) Controls (Online policy) heatmap