Problem Set 01

Ahana Deb

February 11, 2023

The code used in this can be found here- https://github.com/ahanadeb/RL_assignments

1 Problem 1: Policy Evaluation

Plotting the values obtained from following π_{lazy} and $\pi_{aggresive}$ and the difference between them as shown in Fig. 1. The states are represented as continuous rows from 0 to 99.

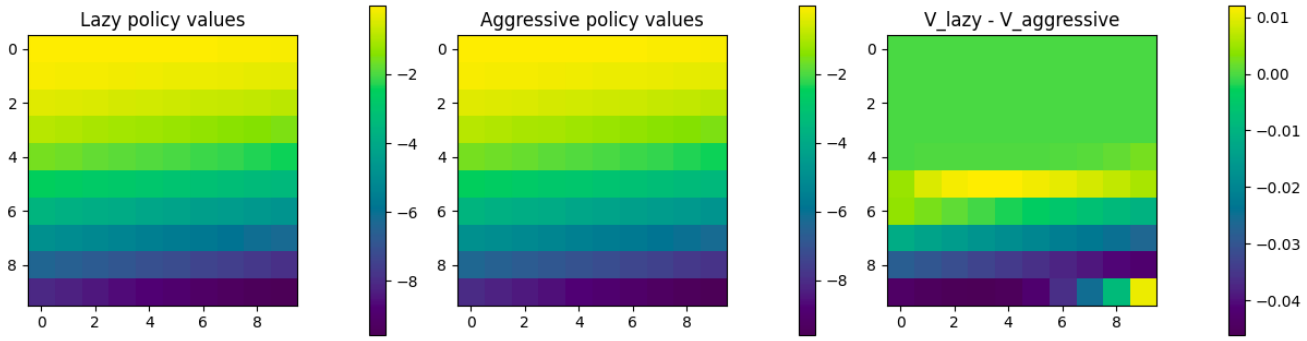

Figure 1: Policy evaluation.

The difference is even more clear when we plot it as a line in Fig. 2. For the given problem, cost of the high service rate action, i.e., $c(a_{high}) = 0.01$. So plotting the difference between the two sets of values obtained, in the left graph, we can there's no difference for the initial state, since both the lazy and aggressive policies take a_{high} for this part, but as the length of the queue increases after that, after state 60, following the lazy policy of a low service rate action is an objectively bad one. If we increase the cost of the high service rate action 10 times, i.e., $c(a_{high}) = 0.1$ in the right graph, the cost of this action becomes so great, taking the high rate action is worse than the lazy action even when the queue length approaches its maximum value. In the last state however, taking a low service rate is better because of the *trunc* function which truncates any increase to the last state, so a_{low} would be better since it has a lower cost.

At state 50, for $c(a_{high}) = 0.01$, we get the values $V_{\pi_{lazy}} = -2.396$ and $V_{\pi_{aggressive}} = -2.399$ and we can say that at this state π_{lazy} policy is slightly better.

At state 80, we get the values $V_{\pi_{lazy}} = -6.231$ and $V_{\pi_{aggressive}} = -6.204$ so $\pi_{aggressive}$ policy is clearly better here, as we can also observe from the left graph in Fig. 2.

2 Problem 2: Value Iteration and Policy Iteration

Looking at Fig. 3 obtained from value iteration, we can observe a significant difference in the values occur between the $10th$ and $20th$ iterations, but not much occurs afterwards. Now looking at Fig. 4 of policy iteration, practically no change can be observed across the iterations, which implies that it is possibly converging very very fast, 55 iterations for value iterations and only 3 iterations for policy iteration (we take an arbitrary small value $\epsilon = 0.1$, the total absolute change in the value function to declare convergence). However the total time taken to run 100 iterations of value iteration and policy iteration are 0.149 seconds and 3.98 seconds resp, and the time taken for convergence .08 seconds and .12 seconds resp.. So even though it appears that policy iteration converges must faster because it takes less iterations, value iteration takes lesser time. This is also illustrated in Fig. 5. 5 equally spaced states are chosen from the original 100 states, and their values are plotted across the iterations. The blue lines is for the values obtained through value iteration, whereas the red is for policy iteration for the same states.

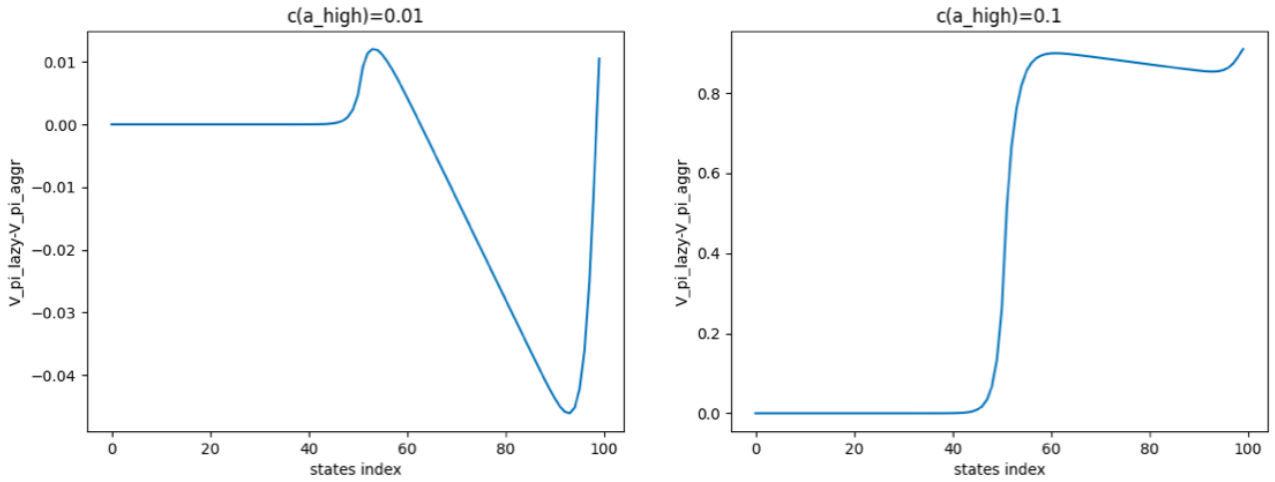

Figure 2: Policy evaluation.

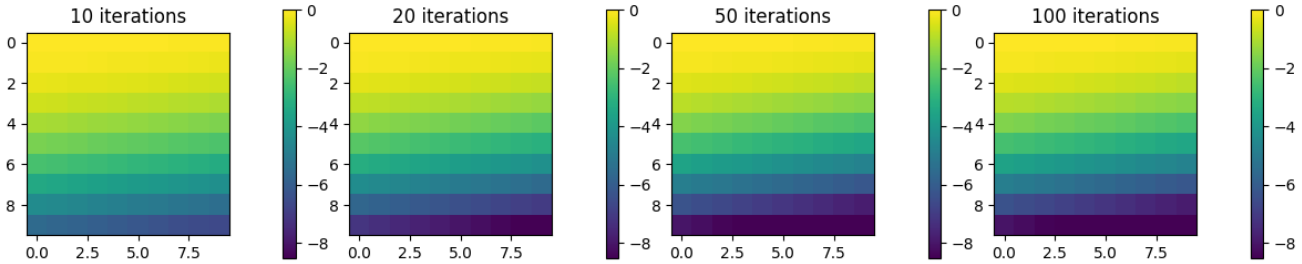

Figure 3: Value Iteration (N=100)

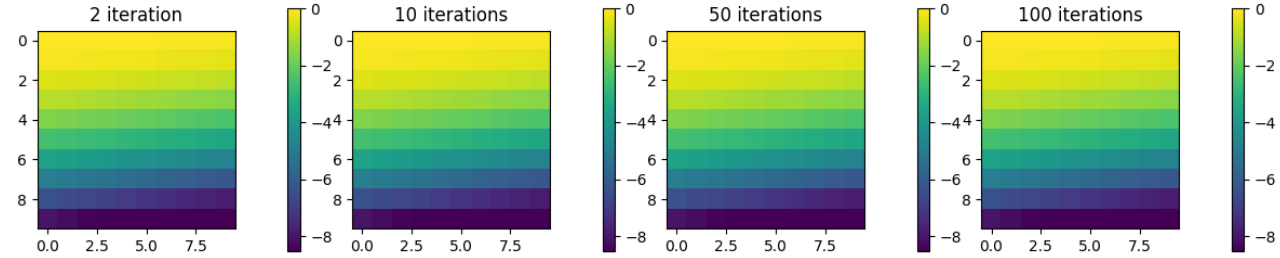

Figure 4: Policy Iteration (N=100)

As we can see, the values in policy iteration converges within the first three iterations, while the value iteration takes more than 50 iterations to achieve stability.

In Fig. 6, the values obtained from the two policies evaluated before are compared with the optimal values. We can see that the optimal values are better than both these cases, since the difference never becomes negative.

The optimal policy obtained from this shows that the best course of action would be to take the lower service rate action a_{low} from states 0 to state 62, and follow the higher service rate action at every state from state 63 till the queue is full, but takes the low service rate action at the final state. This is very close to the aggressive policy we evaluated, and can be inferred from the bottom right graph in Fig. 6 as well. The weird behaviour at the end state is probably because of this 'bouncing back' of the *trunc* function, that the probability of staying in the last state is equal to the probability of both increasing and decreasing the queue size + the probability of only increasing which brings it back to the last state, since it cannot increase further. In such a case, we don't actually

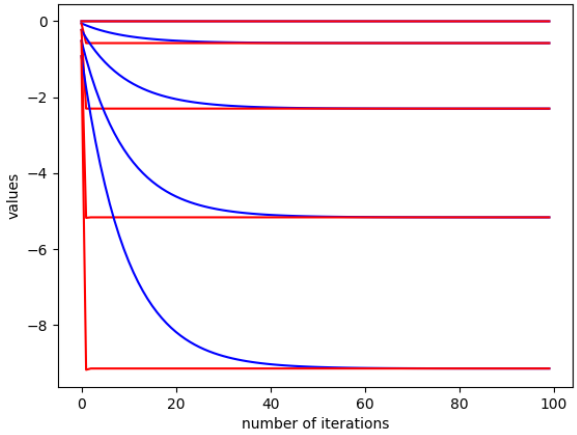

Figure 5: Convergence: Value Iteration vs Policy Iteration

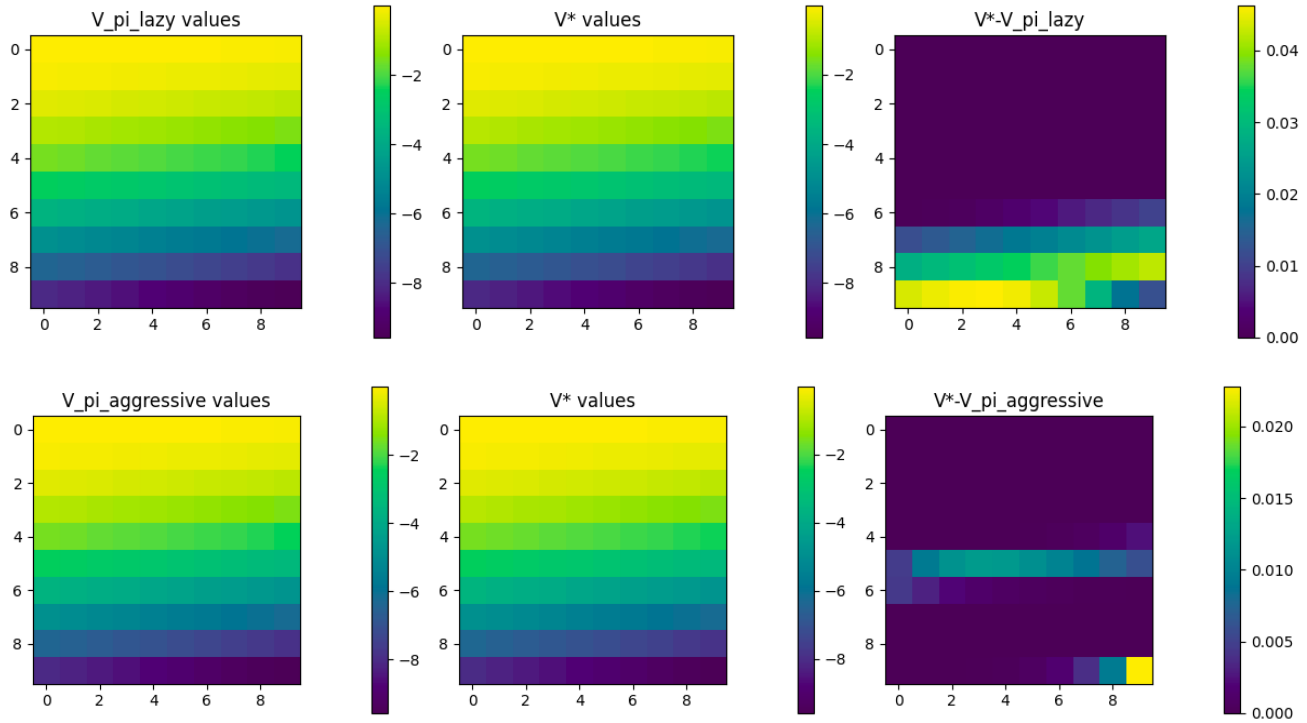

Figure 6: (a) The top 3 plots show the comparison between the values obtained from the π_{lazy} policy and the optimal values. (b) The bottom 3 plots show the comparison between the values obtained from the $\pi_{aggressive}$ policy and the optimal values.

need to have a high service rate action at the last state, the low service rate would be less costly.