CCE IISc -Reinforcement Learning Assignment 1

Akashleena Sarkar

Feb 23, 2020

Aim:

For a 2000 randomly generated 20 arm bandit problem design an experiment to compare and analyze their performance using difference action selection method and draw a conclusion for each of the following by conducting the following experiments:

1) Comparison of performance between Greedy and ε-greedy action value methods for different values of ε (ε=0, 0.05, 0.1 and 0.5)
2) Comparison between the processes mentioned in (i) whose Q_0 is assumed as 0 with that of processes with an optimistic initial value(Q_0=5)
3) Performance analysis between reward variance of 1[$R_t \sim \mathcal{N}(q_*(A_t),1)$] and reward variance of 10 [$R_t \sim \mathcal{N}(q_*(A_t), 10)$]
4)Performance analysis for different values of the degree of exploration parameter [c] in the Upper Confidence- Bound (UCB) Action Selection Method [c= 1.0, 2.0, 5.0]
5)Performance Analysis and Comparison between ε -greedy action selection method and Upper Confidence- Bound (UCB) Action Selection Method

Introduction:

To estimate the value of action we define Q_t as the average of all the rewards obtained so far by choosing action a at time t. We will use incremental implementation to implement the action selection method for greedy and epsilon greedy based bandit algorithms as it has optimal space complexity.

Pseudo Code for Incremental Implementation of bandit algorithm

Initialize, for $a = 1$ to k:
$Q(a) \leftarrow 0$
$N(a) \leftarrow 0$

Repeat forever:
$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$
$R \leftarrow bandit(A)$
$N(A) \leftarrow N(A) + 1$
$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}[R - Q(A)]$

Method:

1. For each of the 2000 bandit problem the action values, $q_*(a)$, a=1,2,3,..20 were selected according to normal Gaussian distribution $\mathcal{N}(0, 1)$ distribution and the actual reward R_t was selected from normal distribution $\mathcal{N}(q_*(A_t),1)$. For any learning method, we can measure its performance and behavior as it improves with experience over 1000 time steps when applied to one of the bandit problems. This makes up 1 run and we repeat this for 2000 independent runs.

   Link to Code

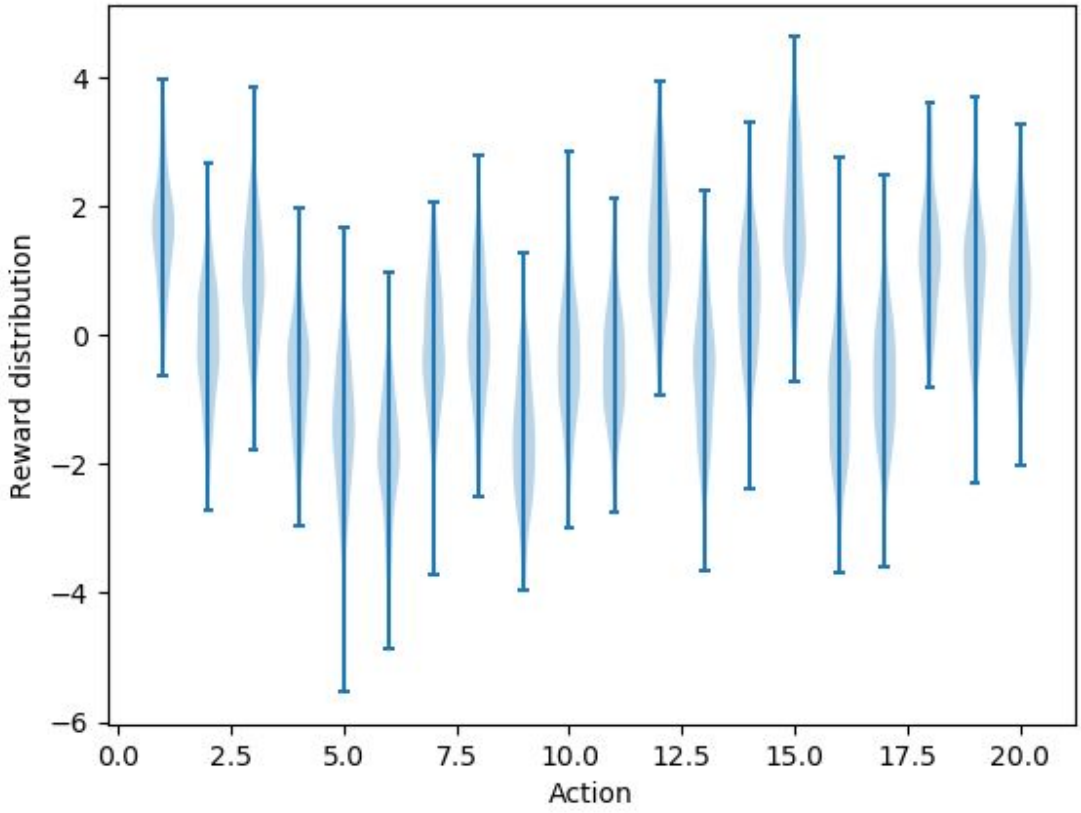


Fig 1: Reward distribution for 20-armed testbed with true value $q_*(a)=\mathcal{N}(0, 1)$ and actual rewards $R_t=\mathcal{N}(q_*(A_t),1)$

2. Using sample averages action-value estimates, we obtain the plot of Average reward and optimal action vs time steps (1000 in this case) for 2000 independent runs each with a different bandit problem. The initial estimate Q_0 is taken to be 0.

1.1 Hypothesis for the Plots:

The expected reward will increase with experience as time steps increase. In the longer run the ε-greedy should perform better than the greedy method because the greedy method will exploit current knowledge to maximize rewards, it won't sample actions which were not tried previously.

The ε-greedy method will be greedy most of the time (probability : 1-ε) but also explore random action (probability : ε), so when $\lim_{t\to\infty} Q_t(a) \to q_*(a)$ so the chances of recognizing the optimal action is more in the ε-greedy case.

1.2 Performance Analysis between greedy and ε-greedy action value

Link: ***(Image of plot)***

1.3 Observations and Analysis:

The graph shows incremental discovery of optimal action across 2000 20-armed bandit problem

1. For ε=0, the greedy method initially improved faster than others but then did not rise and achieved a reward per step of 1 compared with best possible reward of 1.6. if we check the %optimal action graph we see the greedy method found the optimal action only 15% of the task. For the rest of the time, it never explored other actions and kept performing suboptimal tasks and hence the poor performance.

2. For ε=0.05, check the %optimal action graph, we see the graph improved slowly and found the optimal action after 500 time steps, but after that it kept on exploiting the same action 99% of the time. ε=0.05 means it only explores the new action 50 times out of 1000 time steps. The chance that the optimal action is not explored is pretty high and even if the optimal action was explored, its reward is most of the time lower than the ε=0.10 case. Hence, this plot ε=0.05 will take a long time to converge.

3. For ε=0.10, This explored more and found the optimal action earlier but did not select it for more than 90% of the time. It was more into exploitation and less into exploration and since it already found the optimal action quite early it kept exploiting it and so, the average reward is maximum, therefore justified.

4. For ε=0.5, for this high value of ε. it has low asymptotic performance of almost 40% where 40% is pure exploitation and an added 10% due to the random exploration and remaining 50% for exploration. In this case, it forms the best arm fastest but does not exploit it rigorously; rather it explores more and its average reward per step is 0.9 which is lower compared to the rewards in case of ε=0.1 or 0.05. However, since it explores more it will converge faster than the rest of the cases of ε-greedy.

1.4 Conclusion from the above plots:

Higher the rate of exploration, earlier the algorithm discovers the best action and also lower its asymptotic performance.

2. Experiment for Part (2) of the AIM :

Comparison between the processes mentioned in (i) whose Q_0 is assumed as 0 with that of processes with an optimistic initial value(Q_0=5)

We repeat the same process for the same list of values of ε=0, 0.05, 0.1 and 0.5 but with initial optimistic value of Q_0=5

Code

2.1 Hypothesis for the Plots: <check the plots here for Q0=5>

Giving an initial estimate value and using this bias as a tool to encourage exploration and speed up convergence.

The learner will try and explore other actions after getting disappointed with the actions it is receiving as the rewards will be less than the starting estimates. The system will not perform better than epsilon greedy in the beginning because it will be doing a lot of exploration even if it will be selecting greedy actions all the time.

2.2 Performance Analysis between Optimistic ε-greedy and Realistic ε-greedy action value <Image of plot for comparison>

As expected, in the initial steps most agents across all steps will go through all actions after getting disappointed with the actions they receive. In the next step, depending on the rewards they receive, they have a pretty high chance (around 40% as evident from the plot) of selecting the optimal action after trying out all actions once, as on average the optimal action was supposed to give more reward.

The spike in the graph is because a large proportion of the agents selects the optimal action at the same timestep. But once they have selected it, most agents will have been disappointed even more and switch back to selecting other actions hence, we don't have other interference or spikes like in this case.

2.3 Conclusion from the above plots

Initially the optimistic method performs worse because of exploration but eventually it performs better because the exploration decreases with time. This also speeds up convergence. This method is effective for boosting exploration in stationary problems.

3. Experiment for Part (3) of the AIM :

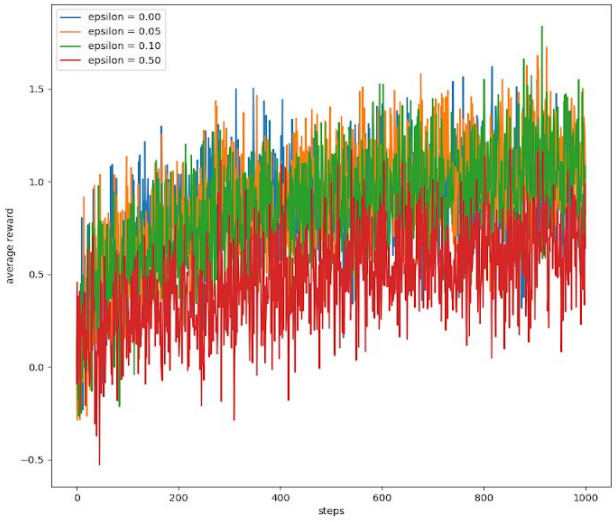
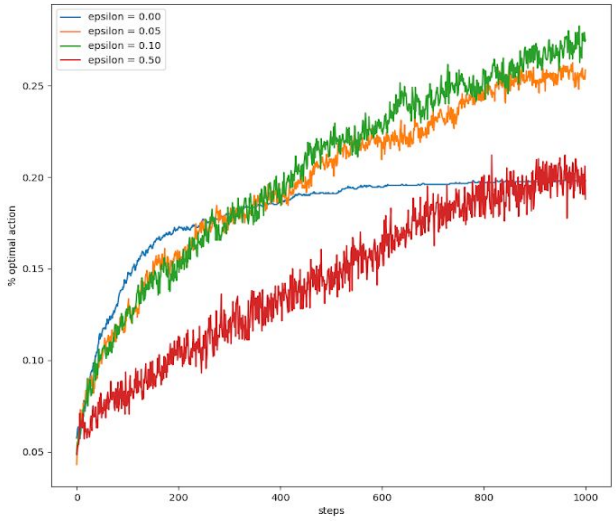
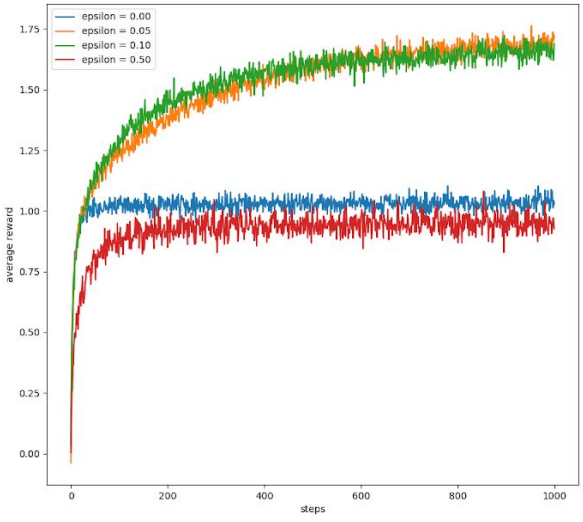
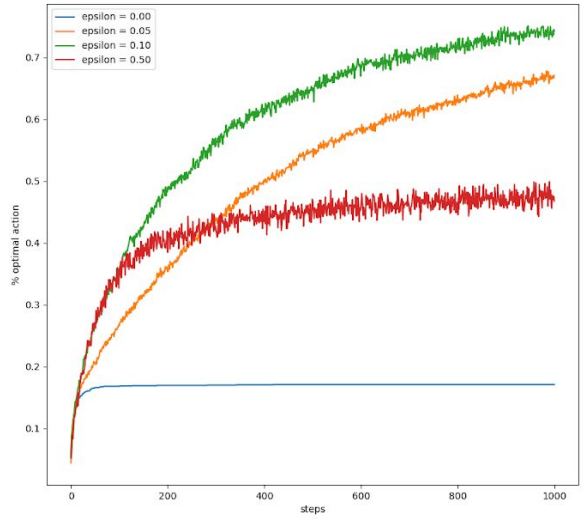
The process in Method 1 was repeated with Qo=0 but with a reward variance of 10. Performance analysis between reward variance of 1[$R_t \sim \mathcal{N}(q_*(A_t),1)$] and reward variance of 10 [$R_t \sim \mathcal{N}(q_*(A_t), 10)$]
We repeat the same process for the same list of values of ε=0, 0.05, 0.1 and 0.5 but with a reward variance of 10. Code

3.1 Hypothesis for the Plots with different variance:

With noisier rewards it takes more time to find the optimal action so ε-greedy should perform better than greedy. If the reward variance is zero, then the greedy method would know the true value of each action after trying it once. In this case the greedy method would actually perform best because it would soon find the optimal action and then never explore.

3.2 Performance Analysis between ε-greedy with reward variance 10 and ε-greedy with reward variance 1 action value

<Link : Average Reward Optimal Action Plots>

Plot a: with reward variance =10	**Plot b: with reward variance=1**
 	 

3.3 Analysis:

1. The spikes in the graph is because in all the test cases there is a lot of exploration due to reward variance of 10. All the test cases reach the average reward at a particular timestep and then deviate to explore other actions, hence the plot looks crowded.

2. Let's compare the plots of reward variance 10 (call it plot a) with that of reward variance 1(call it plot b).
   The greedy method (in plot a)because of large variance received noisier rewards and took around 200 timesteps to find the optimal action as compared to some 30 timesteps in case of plot b. In the rest part of the graph it kept exploiting the same actions in plot b but we see there is some exploration still going on in plot a as conveyed by the non linear increase in the value of the %optimal action plot. Consequently, the average reward per timestep has reached about 1.75 for plot a as compared to just 1.0 for plot b. Hence, we can say plot a has performed better.

3. The ε-greedy performs better than greedy as expected, it is taking more time to find the optimal action in case of ε=0.05 or ε=0.1 in plot a than in plot b. The average reward per timestep for ε=0.1 is 1.75 in plot a and 1.6 in plot b. So, plot a has performed better in this case too.

3.4 Conclusion from the above plots

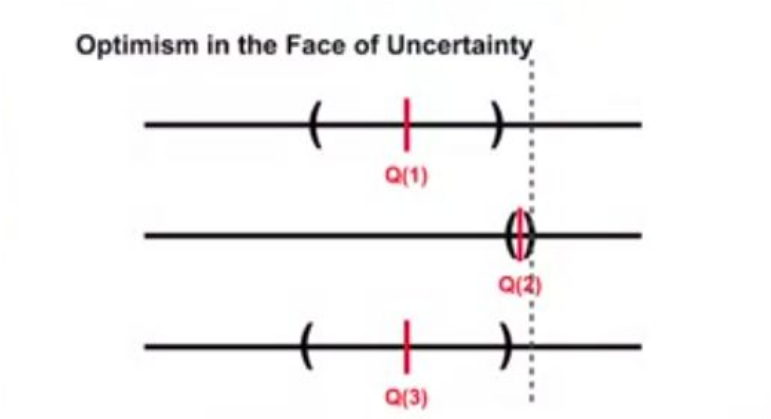
With the goal of exploration in mind, having a variance of 10 definitely improves the performance with a higher reward per time step both in the greedy and ε-greedy case.

4. Experiment for Part (4) of the AIM :

For the same 20 armed testbed,(1) we use UCB for the action selection and compare performance for different degrees of exploration.[c= 1.0, 2.0, 5.0]

4.1 Hypothesis for the Plots: Link:<Reward distribution plot>

The UCB is a more intelligent way of picking up an action where we are interested in the upper bound and pick up that action which has the highest upper bound reward.



In this figure we will choose Q(2) having a higher upper bound even though exploration is less.

$$A_t = \text{argmax}_a \left(Q_t(a) + c \sqrt{\frac{\ln(t)}{N_t(a)}} \right)$$

Exploit

Explore

t = timesteps

$N_t(a)$ = no. of times action (a) is taken

Since the value of c denotes the degree of exploration, greater the value of c more the uncertainty in estimation and farther it is from the optimal value, so average reward should decrease with an increase of c.

4.2 Performance Analysis between for different values of the degree of exploration parameter [c] <Link for Plot>

Yipee! My hypothesis was correct -with an increase in value of c the average reward will decrease. But another thing to note in the plot was the longer spike on the 20th step.This is because for the first 19 steps we pick random actions not chosen yet ($N_t(a)=0$) which will be considered optimal. For the first 20 steps all actions would have yielded the same highest reward of $c\sqrt{\frac{ln\, t}{Nt(a)}}$ with the only discriminatory factor of Qt(a). This sync of agents causes the spike.

After the 20th step they will not be in sync the reward will decrease until more certainty about the system is gathered.

4.3 Conclusion

With an increase in value of c, more the exploration, more the uncertainty in estimation and farther it is from the optimal value, hence average reward decreases.

5. Experiment for Part (5) of the AIM :

<LINK for the CODE>

For the same 20 armed testbed, we will analyze and
1)Compare the performance of UCB vs ε-greedy action selection method. <Link of reward distribution plot>
2)Compare UCB, ε-greedy and ε-greedy with optimistic initialization (with a step factor of 0.1)

5.1 Hypothesis about the performance between UCB and ε-greedy

The average reward for UCB should be more than ε-greedy because the ε-greedy chooses actions randomly whereas UCB selects those actions among the non greedy actions who have the potential of being optimal by considering both the uncertainty and the proximity of the estimate to being maximal.

Comparing UCB with different values of c(1,2,5)and ε-greedy with different values of ε(=0, 0.05, 0.1 and 0.5), I think the UCB with the least value of c will give the best performance in terms of average reward per time step.

5.2 Performance Analysis between greedy and ε-greedy<Link for average reward plot for UCB and Egreedy>

My hypothesis was correct that UCB performs better than ε-greedy because of the biasing of selecting the optimal actions which have a greater upper bound. Also, as expected the best performance was given by UCB with c value =1, this is because the degree of exploration was less, and hence lesser the deviation from the optimal action

5.3 Comparison between UCB, ε-greedy and ε-greedy with optimistic initialization (with a step factor of 0.1) <Link for the Parameter Study Plot>

Since the three methods have different parameters, we can summarize a complete learning curve by its average value over 1000 time steps. This value is proportional to the area under the curve.

1. All the curves are somewhat inverted U-shaped i.e all the algorithm perform best at an intermediate value of the parameter neither too small nor too large.
2. All the curves are not too sensitive to its parameter value, performing well over a range of parameter values.

3. Overall the ε-greedy with optimistic initialization performed best in this case. But this does not guarantee it will work the best in all other real world cases (specially in non-stationary bandit problems

References

1. Sutton, R.S. & Barto, A.G., 2018. *Reinforcement learning: An introduction*, MIT press.
2. Prof. Shalabh Bhatnagar Reinforcement Learning- 2021 CCE-IISc Bangalore Class Lecture