CS 5180

Reinforcement Learning

Exercise 1
Sep 20, 2024
Anuj Patel 002874710

Northeastern University

Question 1:-

$A_1 = 1, R_1 = -1$
$A_2 = 2, R_2 = 1$
$A_3 = 2, R_3 = -2$
$A_4 = 2, A_4 = 2$
$A_5 = 3, R_5 = 0$

k= 4 (1, 2, 3, 4) actions

⇒ The update rule for the action-value estimate Q(t) is $Q_{t+1}(a) = Q_{(t)}(a) + \frac{1}{N(a)} (R_t - Q_t(a))$

-) consider all initial ~~stat~~ estimates $Q_1(1) = Q_1(2) = Q_1(3) = Q_1(4) = 0$

-) Step 1 :- $A_1 = 1, R_1 = -1.$

$Q_2(1) = 1 + \frac{1}{0} (-1-0)$

$Q_2(1) = -1$ [possibly ε:- because no actions has a distinct advantage

-) Step 2: $A_2 = 2, R_2 = 1$

$Q_3(2) = 0 + \frac{1}{1} (1-0) = 1$ [definitely]

-) Step 3:- $A_3 = 2, R_3 = -2$

$Q_4(2) = 1 + \frac{1}{2} (-2-1)$

$Q_4(2) = -0.5$ [possibly]

Step 4:- $A_4 = 2, R_4 = 2$

$$Q_5(2) = -0.5 + \frac{1}{3}(2 + 0.5)$$

$= 0.33$ [possibly]

step 5:- $A_5 = 3$ $R_5 = 0$

$$Q_6(3) = 0 + \frac{1}{1}(0 - 0) = 0$$ [definitely]

-> So overall value of Q is

$Q_6(1) = -1$

$Q_6(2) = 0.33$

$Q_6(3) = 0$

$Q_6(4) = 0$

-> definitely ε :- At step 2 (A2) and step 5 (A3)

-> Possibly occur :- At step 2, 3, 4

Que 2: The update rule for action-value estimates is

$$Q_{n+1} = Q_n + \alpha_n (R_n - Q_n)$$

→ here I will define a genral formula for the action value estimate Q_{n+1} using varying step size parameters

→ Recursive expansion

$$Q_{n+1} = Q_n (1-\alpha_n) + \alpha_n R_n$$

→ substitute Q_n with recursive

$$Q_n = Q_{n-1} + \alpha_{n-1} (R_{n-1} - Q_{n-1})$$

→ update of Q_{n+1}

$$Q_{n+1} = \alpha_n R_n + (1-\alpha_n)(\alpha_{n-1} R_{n-1} + (1-\alpha_{n-1}) Q_{n-1})$$

$$Q_{n+1} = \alpha_n R_n + (1-\alpha_n)\alpha_{n-1} R_{n-1} + (1-\alpha_n)(1-\alpha_{n-1}) Q_{n-1}$$

Previous Initial

→ weighted sum of all rewards and Q_1

$$Q_{n+1} = \alpha_n R_n + (1-\alpha_n)\alpha_{n-1} R_{n-1} + (1-\alpha_n)(1-\alpha_{n-1}) \alpha_{n-2} R_{n-2} + \dots + \prod_{i=2}^{n} (1-\alpha_i) \alpha_1 R_1$$

→ Mathematical induction

$$Q_{n+1} = \prod_{i=1}^{n} (1-\alpha_i) Q_1 + \sum_{i=1}^{n} \alpha_i \prod_{j=i+1}^{n} (1-\alpha_j) R_i$$

final Ans

Same as above formula

$$Q_{n+1} = \prod_{i=1}^{n} (1-\alpha_i) Q_1 + \sum_{i=1}^{n} \alpha_i \prod_{j=i+1}^{n} (1-\alpha_j) R_i$$

Q_1 reduce as more reward get

recent reward contribute significantly since fewer subsequent stepsize product affect their weight. old rewards has less contribution.

Q-3(a) Sample average estimate

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}}$$

- This is unbiased.
- This directly calculates the average of all rewards associated with taking action a, each rewards contribute equally. According to law of large numbers, when a goes to infinity, the average of observed rewards converge to the expected reward q^* of the action.
- That is why, expectation of $E[Q_n]$ of the estimate equal to q^*, make it unbiased.

→ Exponential Recency weighted average

$$Q_{n+1} = Q_n + \alpha (R_n - Q_n)$$

- This is biased.
- α is constant between 0 and 1, it gives lower importance to older rewards. The weight of each reward to estimate of Q diminishes with time. Recent rewards have high importance
- Also this is not allow all rewards to equally contribute to the estimate.

→ α remains fixed, the estimate converges to a value that is not true expected reward q^*. This does not give accurate long term average, when reward probabilities are consistent.

Q.3 (b)

The Initial value of zero creates a lasting impact on the computed average because each update is a weighted average of the previous estimate and new reward, influenced by the initial setting of $Q_1 = 0$. The update are more reliant on the recent rewards due to the factor α and $Q_1 = 0$ diminishes over time. It still affects how the estimate converges to the true value.

→ Also Q value depends on initial estimate, the weight applied to ~~Past~~ ~~or~~ latest ~~&~~ later rewards are not enough to completely neglect the ~~e~~ effect of initial value. This will gives the biased estimate eventually

Q.3

(c) $Q_{n+1} = Q_n + \alpha (R_n - Q_n)$

however It is genrally biased due to its dependency on initial values and α. These are some condition where it is unbiased

(1) Varying α.

It is important to change α with each iteration. The selection of α helps effectively turn the estimate into a sample average, as each rewards contribibutes equally over time to change. Thus α_n must decay in a such way that it sums to infinity but square of α_n sums to finite value.

$\sum_{n=1}^{\infty} \alpha_n = \infty$ and $\sum_{n=1}^{\infty} \alpha_n^2 < \infty$. This allows the estimate to converge in Probability to the true expected value q^*

(2) Initial Q_1.

Q_1 does not inherently bias the estimate if α_n is adjusted as described. so if each reward influence become properly weight impact of Q_1's impact diminishes and becomes negligible as n large. Q_1 can be non-zero but it must not be systemically set in a way that misrepresents expected outcomes

.3

(d) $Q_{n+1} = Q_n + \alpha_n (R_n - Q_n)$

condition we will take is $\alpha_n = 1/n$, the check how Q_n envolves as $n \to \infty$

$$Q_{n+1} = Q_n + \frac{1}{n}(R_n - Q_n)$$

$$Q_{n+1} = \frac{n-1}{n} Q_n + \frac{1}{n} R_n$$

→ Iterative expansion

Q_1 to Q_n $= Q_1 + \frac{1}{1}(R_1 - Q_1)$ $= Q_2$, $Q_3 = \frac{2}{3} Q_2 + \frac{1}{3} R_2$

$$Q_n = 1/n \sum_{i=1}^{n} R_i$$

Shows that $n \to \infty$, Q_n approaches the sample average all rewards.

→ By law of large numbers as $n \to \infty$, the average of $1/n \sum_{i=1}^{n} R_i$ converges to the expected reward.

$q^*: \lim_{n \to \infty} Q_n = \lim_{n \to \infty} (1/n \sum_{i=1}^{n} R_i) = q^*$

This shows that Q_n is asymptotically unbiased.

(e) -)

1) constant learning rate:
The estimated value is always high responsive to recent rewards, which prevents convergence in time to a correct average.

2) discounting of older rewards:
older rewards rapidly become irrelevant, which puts too much weight in the estimate on most recent data.

3) Initial estimate:
Q_1 has the strong influence on the subsequent estimates and kind of initial inaccuracy will persist across all the estimation.

4) Asymptotic behaviour:
With the constant alpha the estimate converges not to the true expected value but to a bias towards recent rewards

Que 4

Softmax distribution for given action a among k actions

$$\pi(a) = \frac{e^{H(a)}}{\sum_{b=1}^{k} e^{H(b)}}$$

-) Two actions:-

The probability of choose action a using Softmax

$$\pi(a) = \frac{e^{H(a)}}{e^{H(a)} + e^{H(b)}}$$

for b

$$\pi(b) = \frac{e^{H(b)}}{e^{H(a)} + e^{H(b)}}$$

-) Logistic function

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

-) connecting softmax to logistic distribution

$\sigma(x) = \frac{1}{1+e}$ $\quad x = H(a) - H(b)$

$$\pi(a) = \frac{e^{H(a)}}{e^{H(a)} + e^{H(b)}} = \frac{e^{H(a)-H(b)}}{1+e^{H(a)-H(b)}} = \frac{e^{x}}{1+e^{x}}$$

This is same as sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}} = \frac{e^{x}}{1+e^{x}}$$

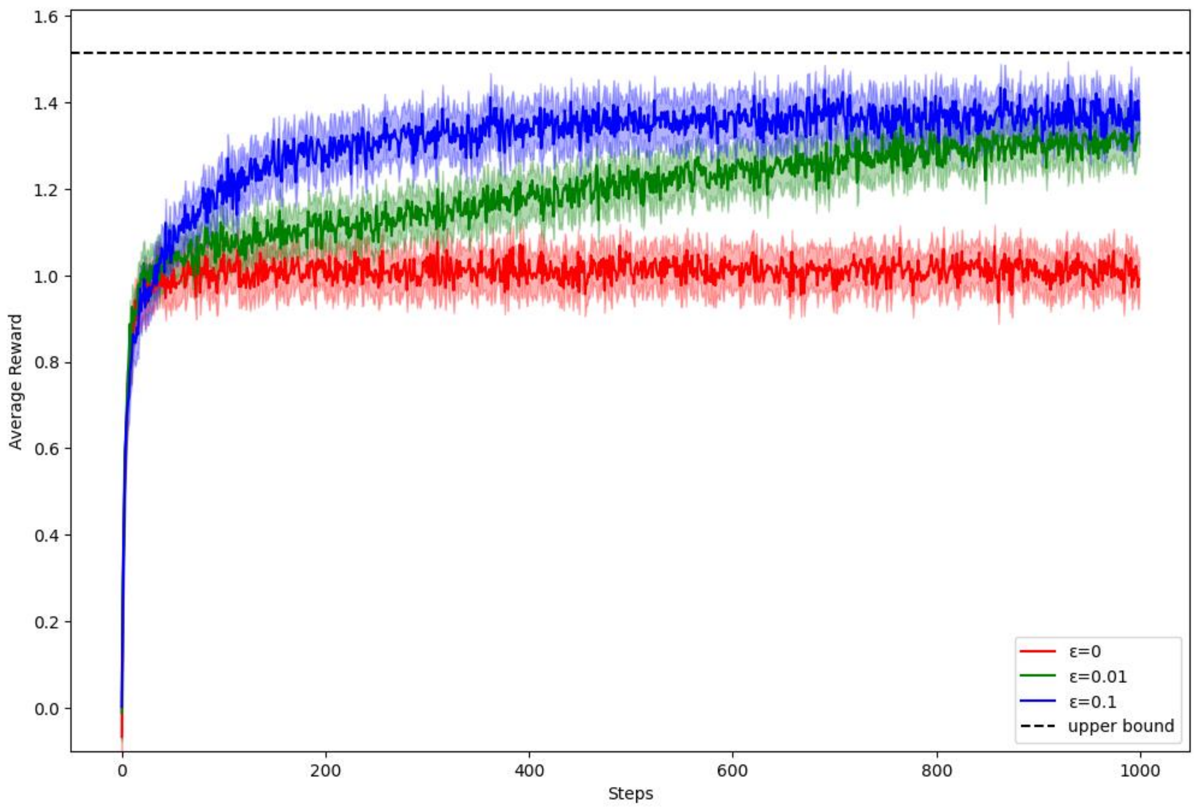
Thus, in case of two actions the softmax function for choosing a over action b simplifies to logistic function. The Input of sigmoid is difference in prefrences between the two actions.

Que 5)
What are the average rewards that the algorithm converges to using different ϵ values? Why does the algorithm converge to different average rewards for different ϵ?

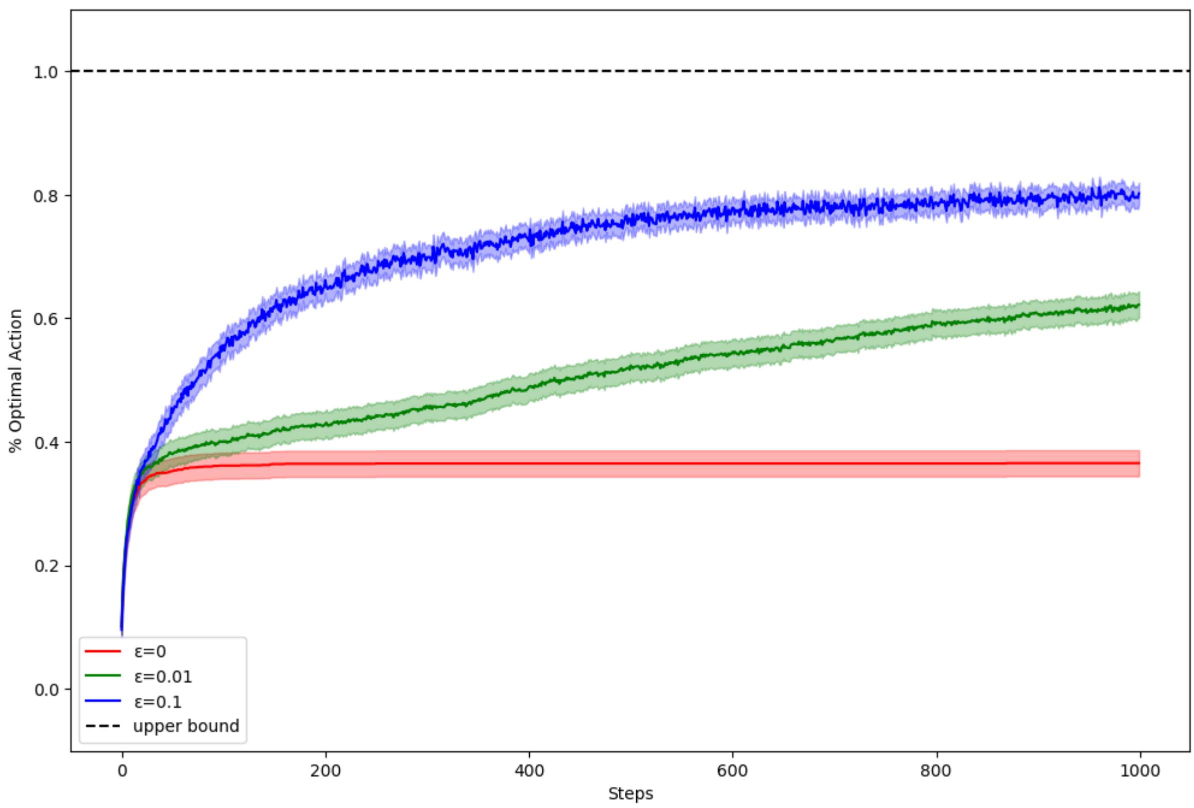
```python
# After running the experiments for ε = 0.0, ε = 0.01, and ε = 0.1
average_reward_0 = np.mean(rewards_e0)
average_reward_01 = np.mean(rewards_e01)
average_reward_1 = np.mean(rewards_e1)

# Print out the average rewards
print("Average Reward for ε = 0.0: ", average_reward_0)
print("Average Reward for ε = 0.01: ", average_reward_01)
print("Average Reward for ε = 0.1: ", average_reward_1)
```

✓ 0.0s

```
Average Reward for ε = 0.0:  1.0037727285698652
Average Reward for ε = 0.01:  1.1920367395199638
Average Reward for ε = 0.1:  1.3081713161618416
```



Average Reward
Steps
ε=0
ε=0.01
ε=0.1
upper bound



% Optimal Action
Steps
ε=0
ε=0.01
ε=0.1
upper bound

What are the average rewards that the algorithm converges to using different ϵ values?

When ε = 0.0, the average reward is 1.0038. In this scenario, an algorithm starts out with only exploitation of the best-known action and no exploration at all. This method of action selection could yield a consequence in the long run, when if there was an error in the initial action value estimations or the best action was not selected.

With ε = 0.01, the average reward reaches about 1.1920. Here, the algorithm primarily exploits but allows for a small amount of exploration (1% of the time). This slight exploration can help rectify any early misjudgements regarding action values.

When ε = 0.1, the average reward comes close to 1.3082. "This new value of ε is interpreted as a 10% probability of exploration, which is markedly increased compared to the previous ones. This value is a level of exploration which is so high that the algorithm tries as much as possible to sample each action several times before drawing a more accurate estimation of the actual action. This way, the algorithm will be able and by following this way it will always come the highest average reward.

Why does the algorithm converge to different average rewards for different ϵ?

Greater Exploration (Higher ε): Indeed, raising the chance of downloading the best action can cause much more inconsistency in rewards to the point that success rate may reduce due to the choice of some unwise actions temporarily. However, in the long term, it provides better coverage of the action space, leading to more accurate estimates of the action values.

Lower ε: Gets you faster as the action becomes the best one when comparing the estimates of the first numerals, but it can be the most dangerous in case, the estimated values are wrong since it may tend to inaccuracy without enough data for its correction.

Que 6)

Explanation of the Spike

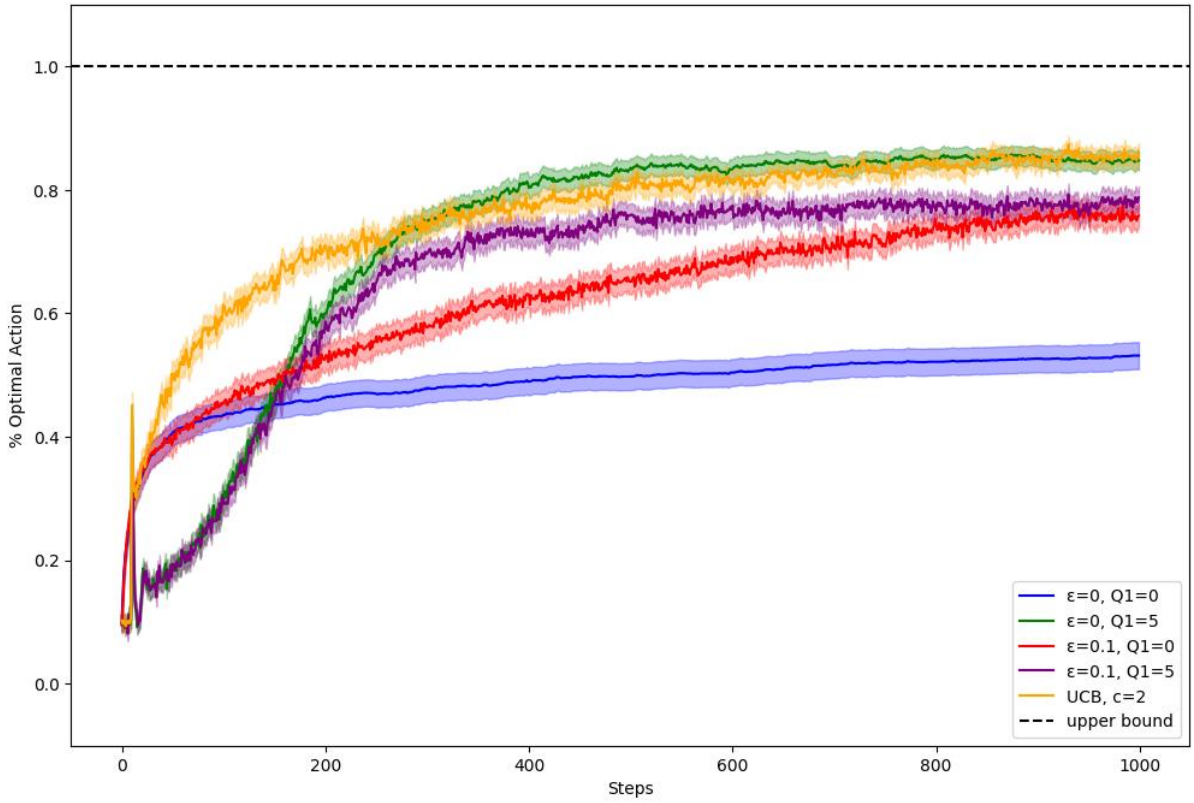
--- Optimistic Initialisation: ε-greedy with Q1 = 5, ε = 0 and ε = 0.1:
Sharply Increasing: With optimistic initial values, for example, Q1=5, all actions are initially overestimated. While the algorithm operates, it will try to immediately make as many explorations of all the actions as possible in order to validate these optimistic assumptions.
Sharp Decline: When the actual rewards begin to arrive and are typically smaller than optimistic starting values, the estimated values, or Q-values, are decreased. The effect of this can be that the per cent of optimal actions chosen sees an obvious decline as the algorithm corrects its estimates to reflect more realistic values.
UCB (Upper Confidence Bound):
Sharp Rise: To the action-value estimates, UCB adds a confidence interval that is high for actions not taken so far or taken only a few times. This will early in the learning process give for all actions high UCB values and therefore will force strong exploration.

Sharp Decrease: As steps are taken and rewards begin to stabilize the estimates, the confidence bounds shrink and reduce the exploratory pulls of suboptimal arms. The more explored an arm, the quicker its exploration bonus decreases

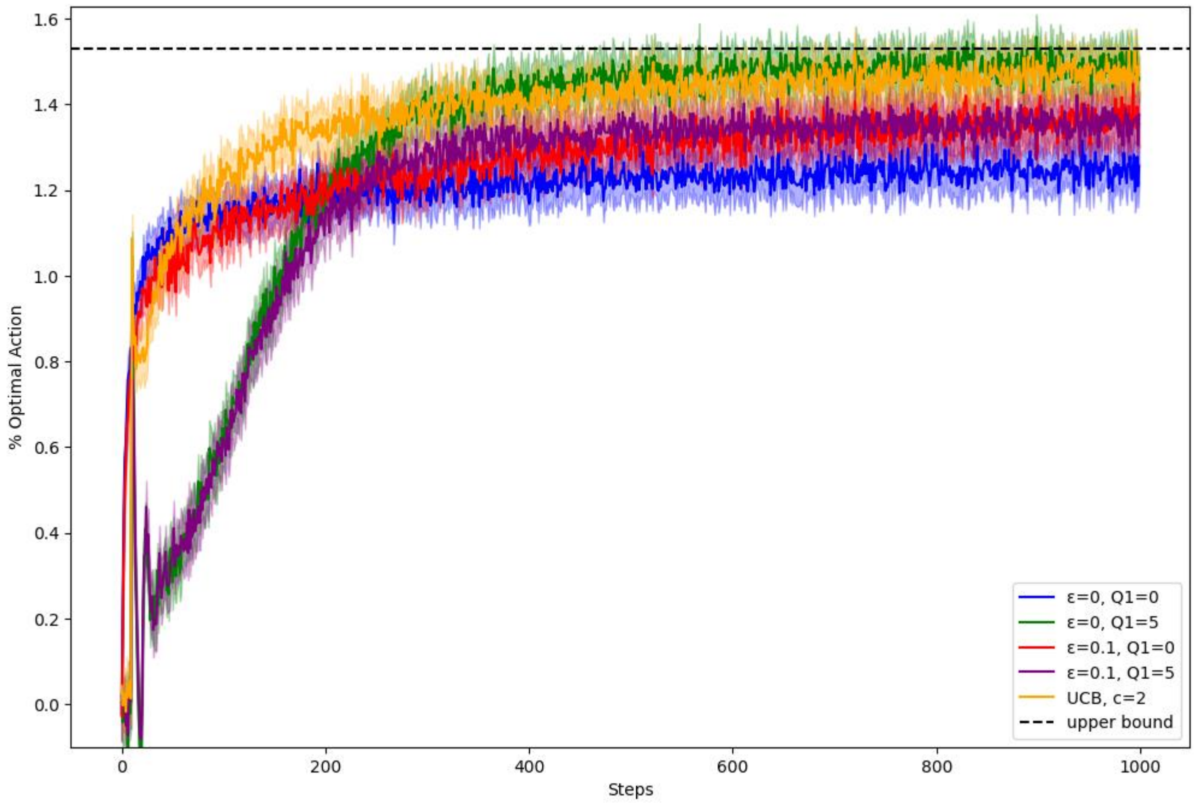
--- **Empirical Evidence from Experimental Data**

The setting of ε=0, Q1=5 has a very pronounced spike since it starts with high estimates and only exploits-meaning the system realizes and adjusts rather quickly since true values don't live up to initial optimism.
This is appropriate because the spike of the UCB method is a result of its mechanism of balancing exploration and exploitation considering the uncertainty in the estimates, which is inherently higher at the beginning when few or no samples have been drawn from each distribution.

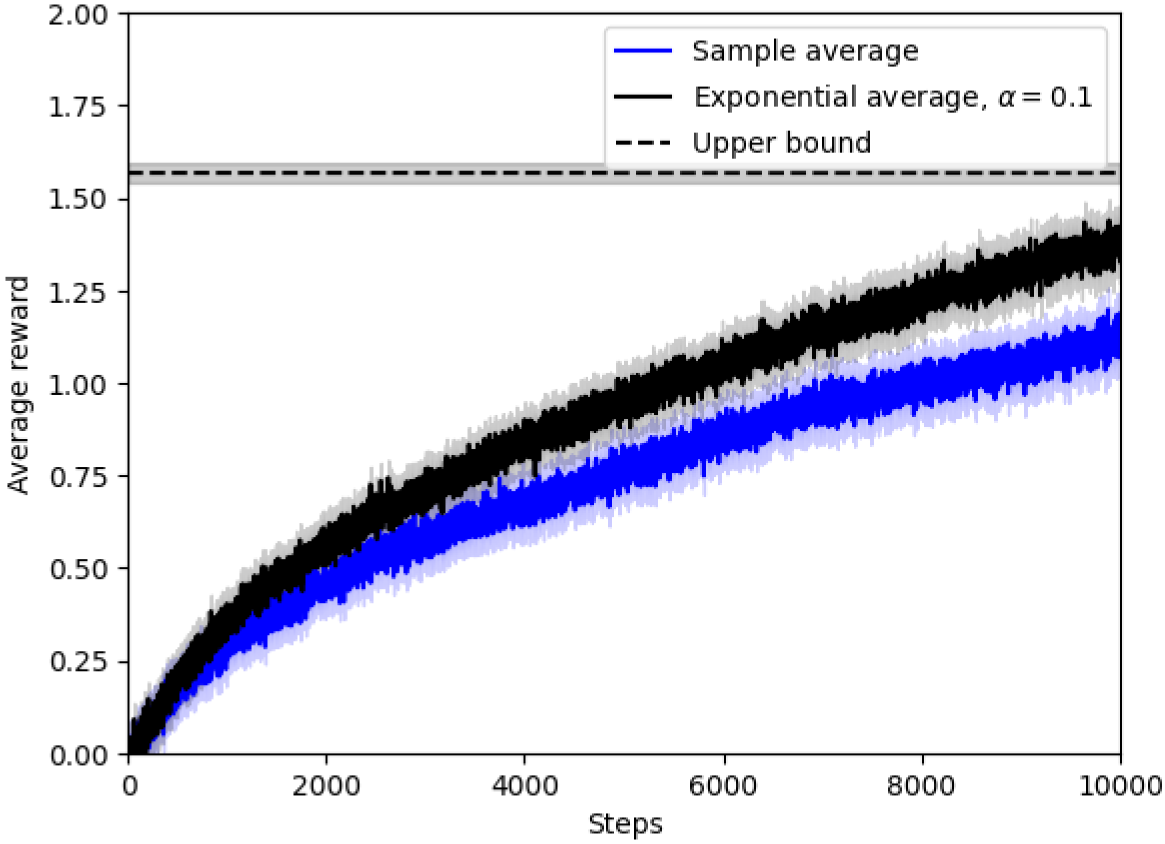
Reproduce Figure 2.3 using exponential average

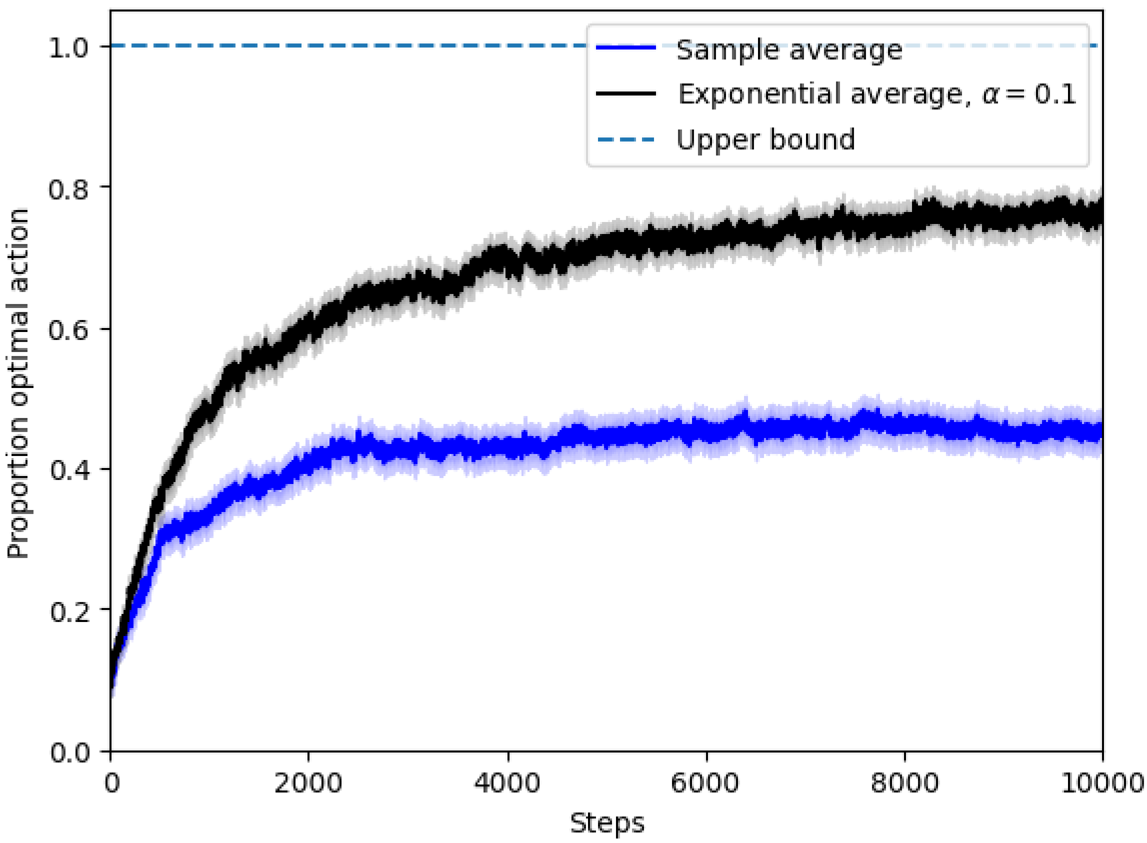


Reproduce Figure 2.4 using sample average (equation 2.1)



Que 7)


Sample average
Exponential average, $\alpha = 0.1$
Upper bound
Average reward
Steps
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
2000
4000
6000
8000
10000


Sample average
Exponential average, $\alpha = 0.1$
Upper bound
Proportion optimal action
Steps
1.0
0.8
0.6
0.4
0.2
0.0
0
2000
4000
6000
8000
10000