Q3 Initially, it seems like $\varepsilon = 0.1$ performs the best as can be seen in the plots generated.

But in the long run, $\varepsilon = 0.01$ should outperform it.

This is true because as steps $\to \infty$, both methods will have the expected rewards converge to the true expected reward value.

After this stage, $\varepsilon = 0.01$ will explore less and exploit 10 times more than $\varepsilon = 0.1$.

This is the reason why $\varepsilon = 0.01$ will surpass $\varepsilon = 0.1$ in the total rewards after a large no. of steps.

A similar explanation follows for why $\varepsilon = 1/t$ would outperform $\varepsilon = 0.01$.

This is because exploration would eventually tend to 0 after a large no. of steps. And the sum rewards would be more due to more exploitation.

$\therefore$ $\varepsilon = 1/t > \varepsilon = 0.01 > \varepsilon = 0.1 > \varepsilon = 0$ (greedy)

The expected reward value of a random guess = 0

For $\varepsilon = 0.1$, 90% chance of having a 1.5 reward value, and 10% chance of having a random guess = 0

$\therefore$ Expected reward $= 1.5 \times 0.9 = 1.35$

Similarly for $\varepsilon = 0.01$, 99% chance of having a 1.5 reward value, and 1% chance of having a random guess = 0

$\therefore$ Expected reward $= 1.5 \times 0.99 = 1.485$

Q4 To show that sample mean is not influenced by the initial choice of $Q_1(a)$ $\forall$ a.

The sample mean estimate $Q_t(a)$ after t steps is given by :-

$$Q_t(a) = \frac{1}{N_t(a)} \sum_{i=1}^{t} R_i(a)$$

$N_t(a)$: No. of times action a has been selected upto time t

$R_i(a)$: is the reward received on the i^{th} selection of action a

$\therefore$ $Q_t(a)$ does not depend on the initial choice of $Q_1(a)$ as soon as we pick the arm a even once. It's the avg. of the actual rewards received when picking action a.

To show that $Q_t(a)$ depends on $Q_1(a)$ when using a constant step size α.

Update rule for $Q_t(a)$:-

$$Q_{t+1}(a) = Q_t(a) + \alpha\left(R_t(a) - Q_t(a)\right)$$

$$Q_2(a) = Q_1(a) + \alpha\left(R_1(a) - Q_1(a)\right)$$
$$Q_3(a) = Q_2(a) + \alpha\left(R_2(a) - Q_2(a)\right)$$

Subs. $Q_2(a)$ into $Q_3(a)$:-

$$Q_3(a) = Q_1(a) + \alpha(R_1(a) - Q_1(a)) + \alpha\left(R_2(a) - (Q_1(a) + \alpha(R_1(a) - Q_1(a)))\right)$$

Generalising for any t :-

$$Q_t(a) = Q_1(a)(1-\alpha)^{t-1} + \alpha \sum_{i=1}^{t-1} (1-\alpha)^{t-1-i} R_i(a)$$

This eqⁿ shows that $Q_t(a)$ is indeed a function of the initial estimate $Q_1(a)$. The dependence on $Q_1(a)$ diminishes as t increases but is influenced by the constant α.

Smaller α :-

When α is small, $(1-\alpha)^{t-1}$ decays more slowly, meaning the influence of $Q_1(a)$ on $Q_t(a)$ persists longer.

Larger α :-

For larger α, $(1-\alpha)^{t-1}$ decays faster, reducing the dependence on $Q_1(a)$ more quickly.

Method with constant step size but no dependence on $Q_1(a)$.

Initialize $Q_1(a) = 0$

Using this, you can use a constant step size α and eliminate the dependence on $Q_1(a)$.

Another way :- (update rule)

$$Q_{t+1}(a) = \alpha \sum_{i=1}^{t} (1-\alpha)^{t-1} R_i(a)$$

↳ no impact of $Q_1(a)$.

Q6 Exercise 2.8

UCB action selection rule:-

$$a_t = \underset{a}{\text{argmax}} \left[Q_t(a) + c \sqrt{\frac{\ln t}{N_t(a)}} \right]$$

During the first 10 steps, each of the 10 arms is selected once due to the structure of UCB.

$$\frac{\ln t}{N_t(a)} \to \infty \quad \text{when } N_t(a) = 0$$

At the 11th step, every action has been tried exactly once

$$\therefore N_t(a) = 1 \text{ and } t = 11 \text{ for all } a$$

Now, the algorithm selects the action with the highest summation of $Q_t(a)$ and $c\sqrt{\frac{\ln 11}{1}}$

This action would be the one ~~that~~ that had a higher reward in it's first trial.

∴ The avg. reward at the 11th step increases significantly resulting in a spike.

Date:

After the 11th step,

the algorithm begins to exploit the actions with higher $Q_t(a)$ values and lower uncertainty as the exploration term decreases as $N(a)$ increases.

~~Effect of c.~~

But, because UCB is still exploring less-frequently tried actions, some suboptimal actions might be chosen resulting in lower average reward at those time steps.

Effect of c:

c controls the trade off between exploration and exploitation.

If $c=1$, the algorithm explores more conservatively. This means the spike is less prominent because the exploration term isn't as dominant.

$c=4$

With a higher c, the exploration term is larger, making the initial spike more prominent.