CSE 564: Reinforcement Learning **(Due: 13 /09/20)**

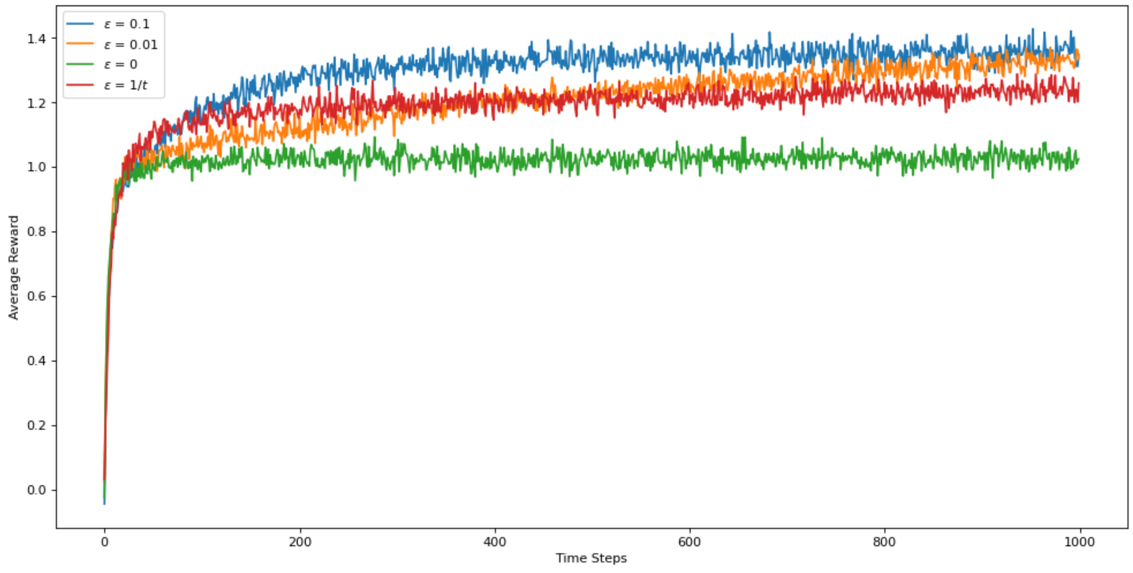
Home-Work #1

Instructor: Dr.Sanjit Kaul *Name:* Aditya Rastogi, *Roll No:* 2018273

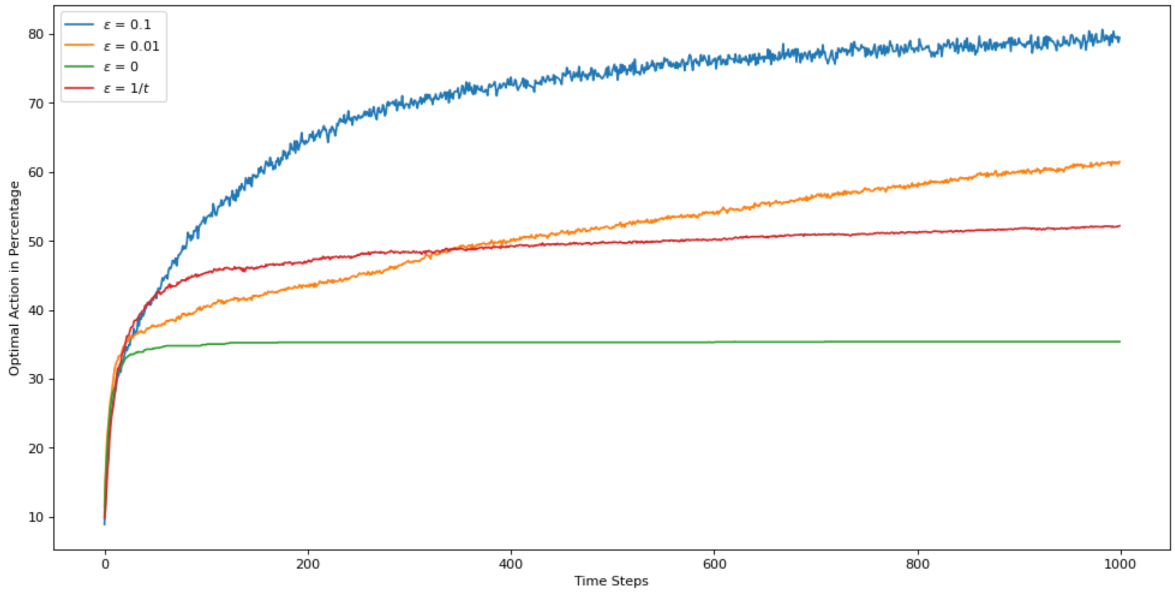
Problem 1

I prepared a 10 arm test-bed with variance = 1, $\epsilon = 0, 0.1, 0.01$ and also a time dependant $\epsilon(t) = 1/t$. The plots are as follows:

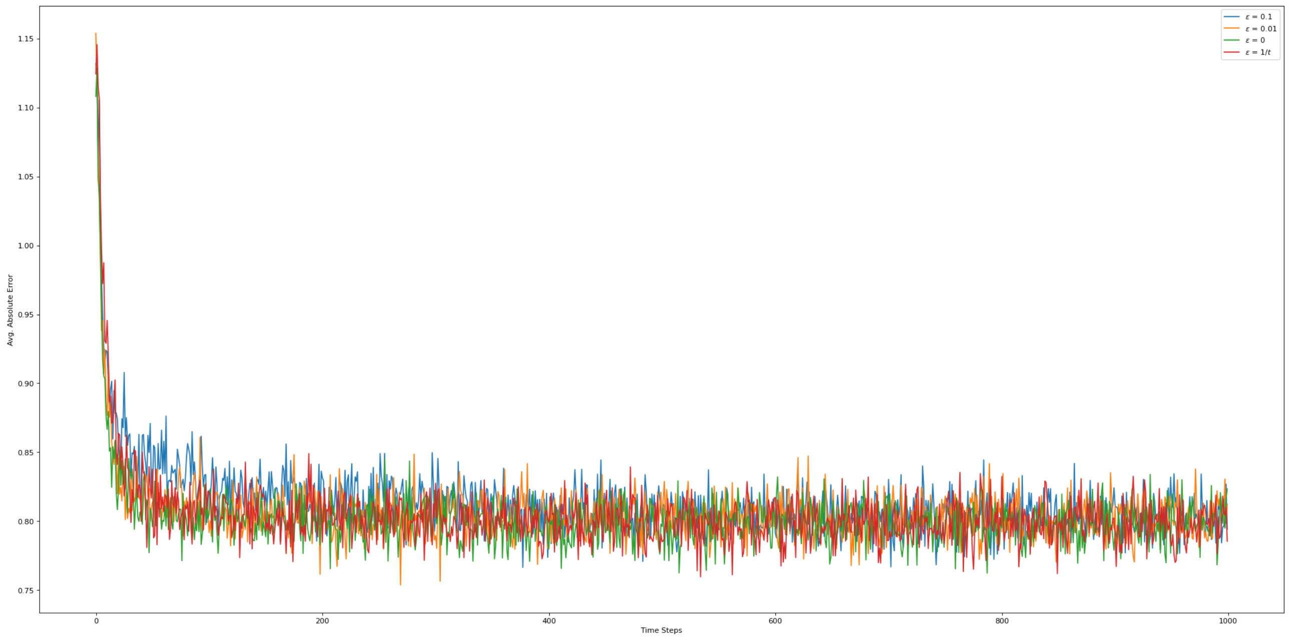
Average Reward



Optimal Action in (%)



Average Absolute Error



For each of the graphs, I simulated 2000 independent runs of 1000 time-steps each using the basic ϵ - greedy method, with varying ϵ over [0.1, 0.01, $1/t$] and also a greedy option and plot them on various parameters such as **Average Reward**, **Optimal Action** and **Average Absolute Error**.

Each arm was picked with mean = 0 and variance = 1. A unit variance test-bed has a considerable lower amount of difference among rewards and the graph was quite a less noisier.

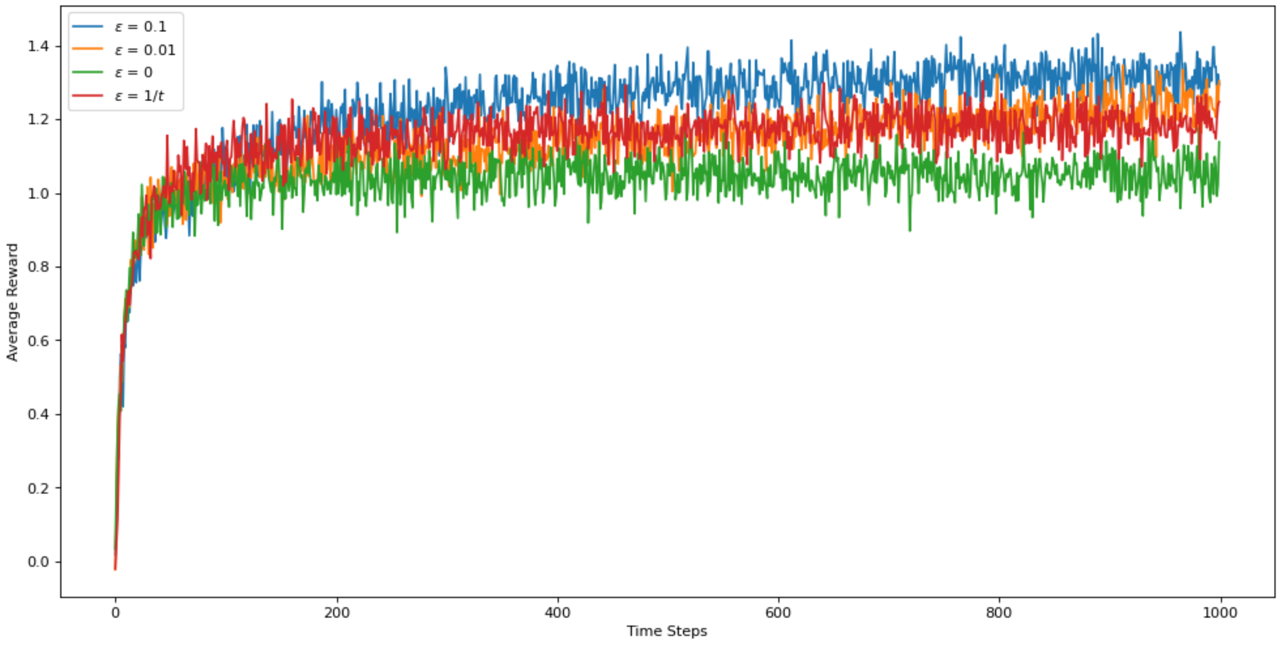
From the graph as we are dealing in a *Practical Sense* (only a 1000 Time Steps) $\epsilon = 0.1$ tends to perform the best. This is so because of a higher initial exploration rate, $\epsilon = 0.1$, but clearly the rate of increase for $\epsilon = 0.01$ is better, i.e. say maybe a greater $t = 10000$, $\epsilon = 0.01$ might perform better.

In the long run tho, $\epsilon = 1/t$ would perform the best as every ϵ - greedy would have fairly good estimates of the truth, $\epsilon = 1/t$ would pick the greedy almost all of the times, therefore have the maximum average reward.

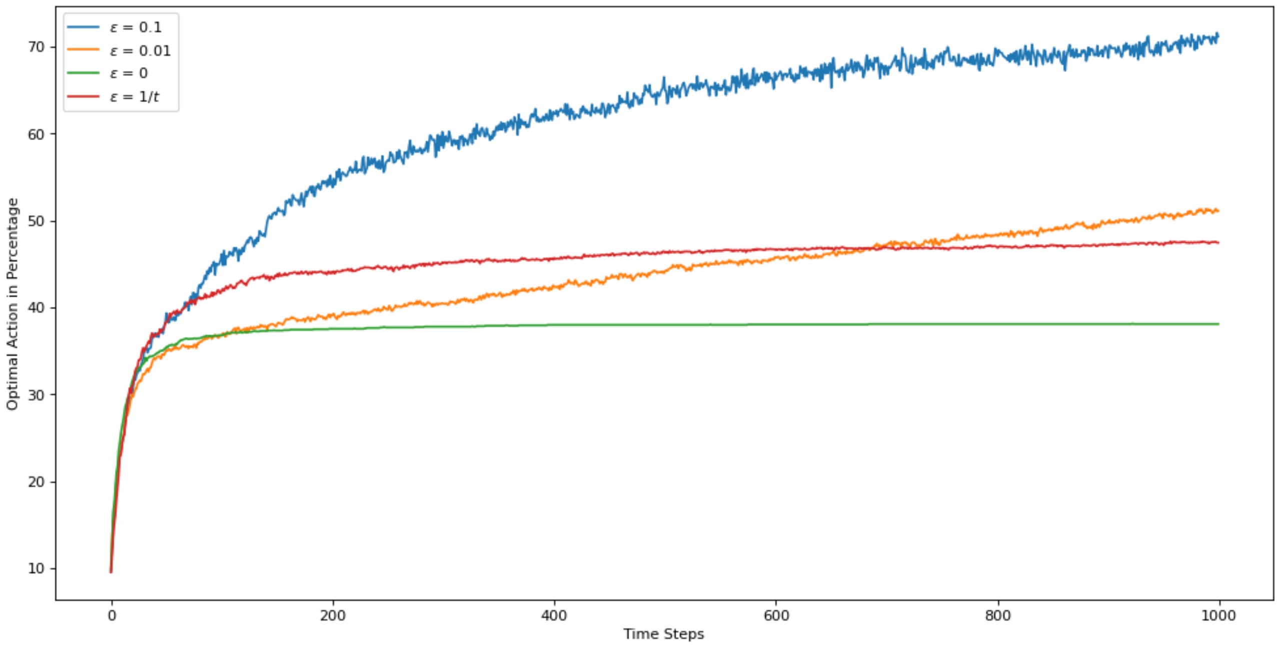
Problem 2

This time around the variance = 4.

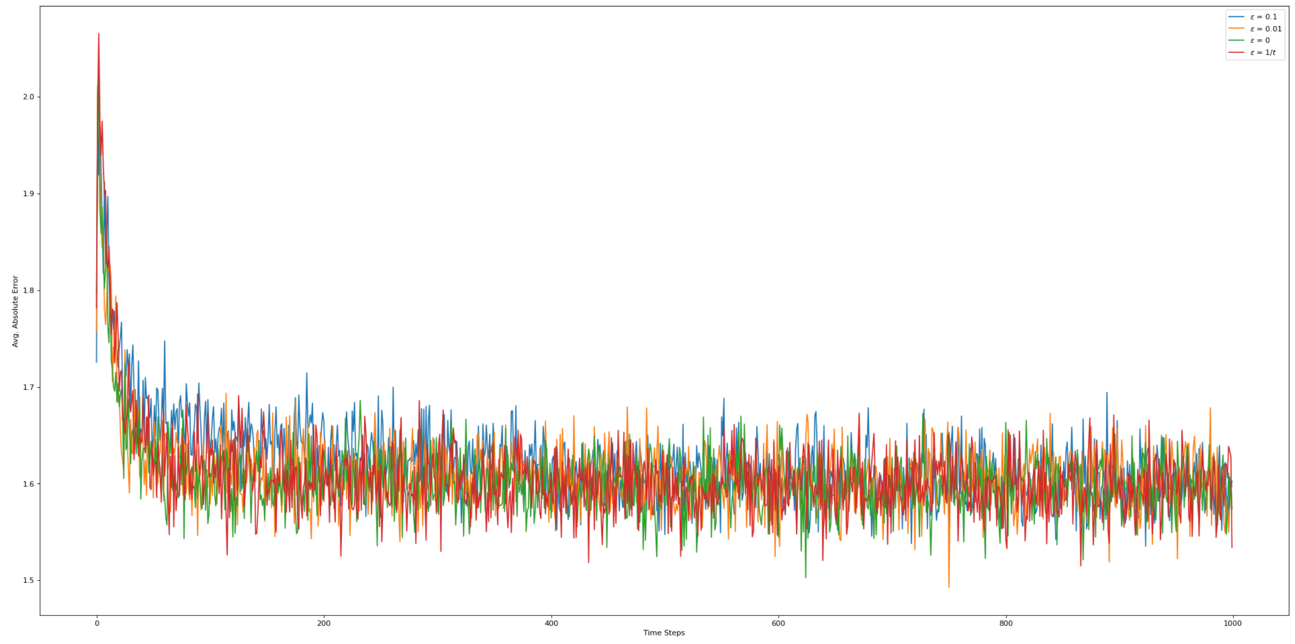
Average Reward



Optimal Action in (%)



Average Absolute Error



For each of the graphs, I simulated 2000 independent runs of 1000 time-steps each using the basic ϵ - greedy method, with varying ϵ over [$0.1,\ 0.01,\ 1/t$] and also a greedy option and plot them on various parameters such as **Average Reward**, **Optimal Action** and **Average Absolute Error**.

Each arm was picked with mean $= 0$ and variance $= 4$. A variance $= 4$, results in quite noisier estimates as compared to a unit variance test bed.

From the graph as we are dealing in a *Practical Sense* (only a 1000 Time Steps) $\epsilon = 0.1$ tends to perform the best. This is so because of a higher initial exploration rate, $\epsilon = 0.1$, but clearly the rate of increase for $\epsilon = 0.01$ is better, i.e. say maybe a greater $t = 10000$, $\epsilon = 0.01$ might perform better.

In the long run tho, $\epsilon = 1/t$ would perform the best as every ϵ - greedy would have fairly good estimates of the truth, $\epsilon = 1/t$ would pick the greedy almost all of the times, therefore have the maximum average reward.

Problem 3 (Exercise 2.3)

In the Hand-Written PDF.

Problem 4

In the Hand-Written PDF.

RL - HW - 1

Sol: At time $t = \infty$ ie in the long run all epsilons except $\varepsilon = 0$ ie a greedy policy will all have fairly good estimates of the ~~reth~~ ie $Q*(a)$.

The value of the cumulative ~~rewards~~ average rewards would then depend on simply the number of times the greedy action is picked.

That is

$\varepsilon = 0.01$

Using the formula $(1-\varepsilon) + \frac{\varepsilon}{|A|}$

$\Rightarrow$ 0.991 ie 99.1% of the times this policy picks the greedy (optimal arm).

$\varepsilon = 0.1$

Similarly as above

$\Rightarrow$ 0.91 ie 91% of the times picks the greedy (optimal arm)

$\underline{\varepsilon = 1/t}$

At $t \to \infty$ we $\underline{\varepsilon = 0}$. Now this means in theory At $t = \infty$ $\varepsilon = 1/t$ will always pick the optimal (100%) Arm. But practically it is not necessary for it to perform best in say $(t = 1000)$ time-steps

i.e. $\varepsilon = 1/t$ in the long run performs the best.

~~[illegible]~~

Ques 4

Part 1

(a)

While taking the Sample mean to estimate 'Q_n', even if Q_1 defines how the first arm is picked, but for Q_2, and subsequent Q_n(s) the choice is independent of Q_1 and depends solely on the reward achieved from that arm.

Part 1 (a)

We know that the sample mean (Q_n)

$$Q_n = \frac{R_1 + R_2 + \dots R_{n-1}}{n-1}$$

where 'n' represents the number of times arm has been selected.

$$\Rightarrow Q_n = \frac{1}{n-1}\sum_{i=1}^{n-1} R_i \quad \text{— (i)}$$

$$= \frac{1}{n-1}\left(R_{n-1} + \sum_{i=1}^{n-2} R_i\right) \quad \text{(separating } R_{n-1})$$

$$= \frac{1}{n-1}\left(R_{n-1} + \frac{(n-2)}{(n-2)}\sum_{i=1}^{n-2} R_i\right) \quad \text{(Using eqn (i) i.e. } \left[Q_n = \frac{1}{n-1}\sum_{i=1}^{n-1} R_i\right])$$

$$= \frac{1}{n-1}\left(R_{n-1} + (n-2) Q_{n-1}\right)$$

$$= \frac{1}{n-1}\left(R_{n-1} + (n-1) Q_{n-1} - Q_{n-1}\right)$$

$$\boxed{Q_n = Q_{n-1} + \frac{1}{n-1}\left(R_{n-1} - Q_{n-1}\right)}$$

∴ Hence no dependance on Q_1.

On the other hand using a constant step-size 'α' $\rightarrow$

(b) We know that for a testbed with constant step-size 'α' we have, n is the no. of times ~~[illegible]~~ picked,

$$Q_{n+1} = Q_n + \alpha\,[R_n - Q_n]$$

$\Rightarrow Q_{n+1} = Q_n + \alpha\,[R_n - Q_n]$ $\longrightarrow$ (ii)

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

Again using (ii) to expand Q_n.

$\Rightarrow Q_{n+1} = \alpha R_n + (1-\alpha)(\alpha R_{n-1} + (1-\alpha) Q_{n-1}]$.

$\Rightarrow$ We can now expand the above equation till n reaches 1. i.e.

$\Rightarrow$ ~~[illegible]~~ $= (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha (1-\alpha)^{n-i} R_i$.

$$\Rightarrow \boxed{Q_{n+1} = (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha (1-\alpha)^{n-i} R_i} \quad -(iii).$$

In this case Q_{n+1} clearly depends on Q_1.

We can also see that if 'α' is <u>smaller</u> then the term <u>$(1-\alpha)$</u> is bigger which in turn keeps the coefficient of Q_1 larger as compared to a higher 'α'.

i.e. <u>Q_1 has</u> a higher dependance for <u>smaller</u> 'α'

Part 2

We can update our constant step-size 'α'. with another value ω_n s.t.

$$\boxed{\omega_n = \omega_{n-1} + \alpha(1 - \omega_{n-1})}.$$

ie $\boxed{\alpha_{new} = \alpha / \omega_n}$

$$\Rightarrow Q_{n+1} = (1-\alpha)^n Q_1 + \sum_{i=1}^{n} R_i \alpha (1-\alpha)^{n-i}.$$

α_{new} for $Q_1 = \alpha / \omega_1$.

Now using the formulas for ω_1 we have $\boxed{\omega_1 = \alpha}$

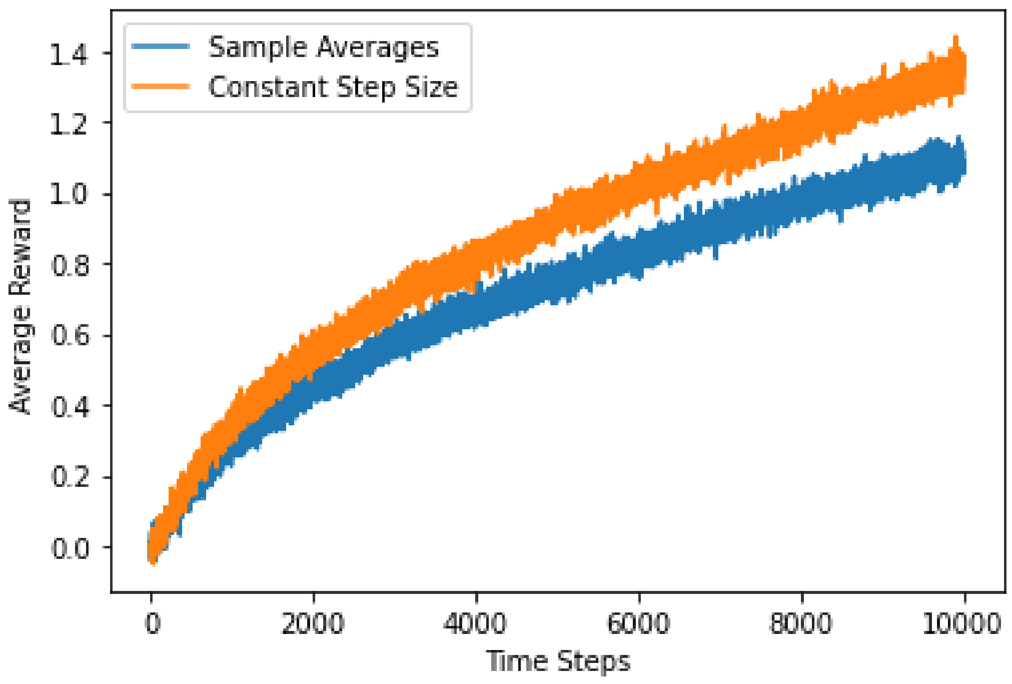
$$\Rightarrow Q_{n+1} = \cancel{(0) Q_1} + \sum_{i=1}^{n} R_i \left(\frac{\alpha}{\omega_i}\right)\left(1 - \frac{\alpha}{\omega_i}\right)^{n-i}$$

i.e. independent of $\underline{Q_i}$.

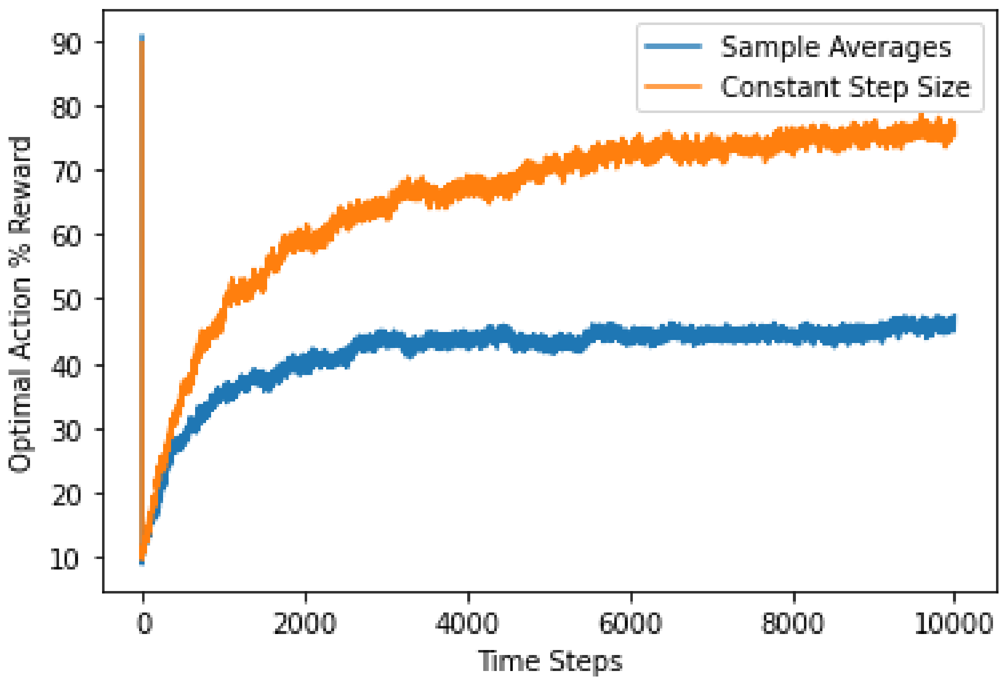
$\times \quad \times \quad \times$

Problem 5

Average Reward



Optimal Action

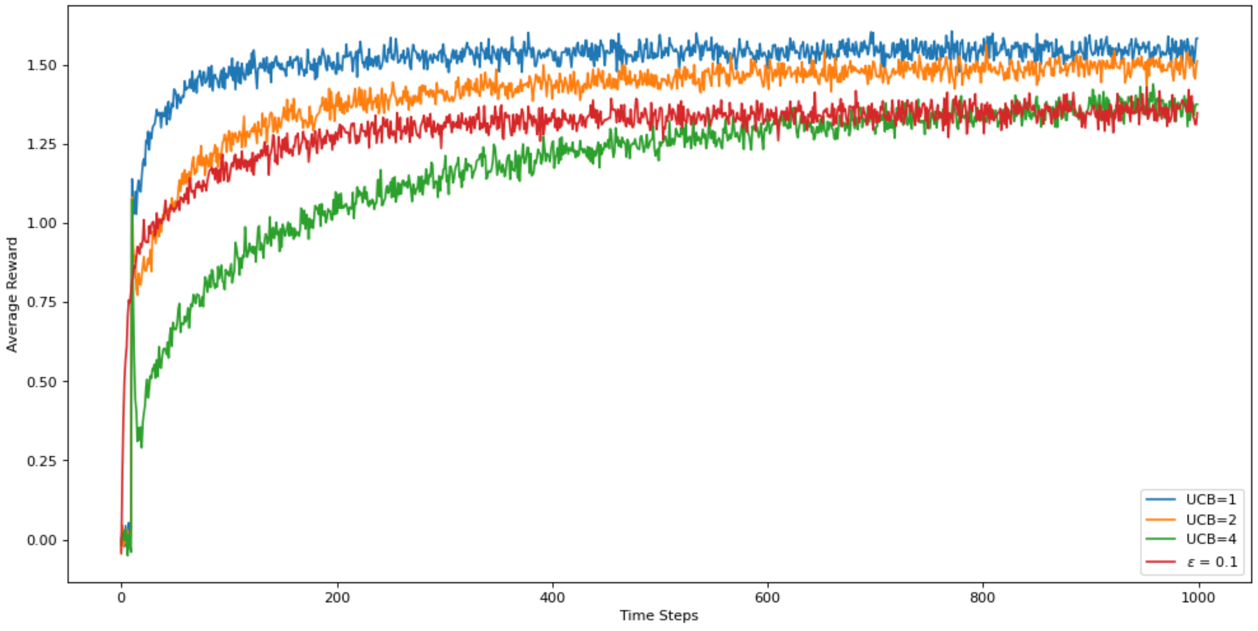


In a Non-Stationary bandit problem, Constant Step Size tend to perform generally quite a lot better as they

never fully converge and tend to readjust themselves according to their most recent updates, a quality desirable in a Non - Stationary setting.
Here I ran 200 simulations over 10000 time-steps to better understand the performance difference of both the policies, *Sample Averages* and *Constant Step Size*.

Problem 6

Average Reward



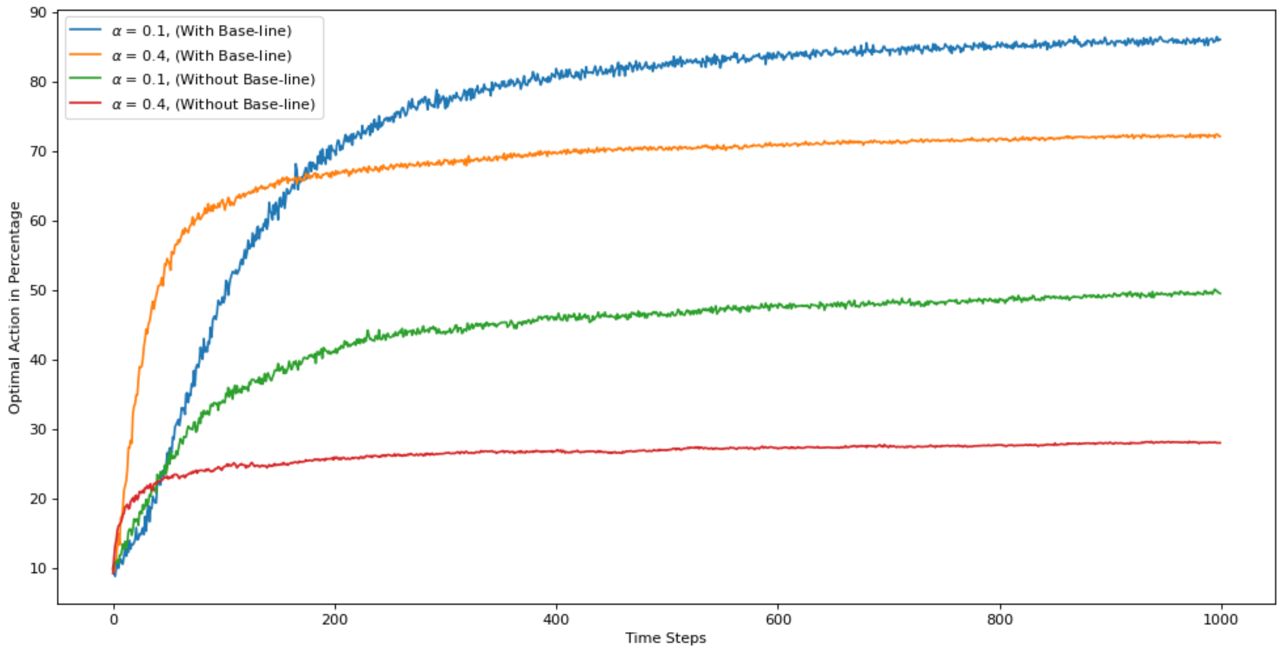
The Upper Confidence Bound algorithm initially favours exploration over exploitation and in the later stages gives more priority to exploitation. UCB forces the selection of arms that have not been selected even once in the first k time steps for a k armed bandit system. As a result in our case of a 10 arm bandit system, the first 10 selections (although in a random order) are always the 10 different arms of the 10 armed bandit. As a result we can see for the first 10 time steps the average reward is considerably low and at times even drops below 0.

But at the 11^{th} time-step as each arm has been picked exactly once, which mathematical terms means that the value $c\sqrt{\frac{\log t}{N_t(a)}}$ is same for every arm a, hence the system now picks the arm which at that stage is optimal i.e. on the basis of $Q_t(a)$ only. As this happens for every one of the 2000 runs, the average reward at the 11^{th} time step is quite considerably better than the previous 10 time steps and thus a **significant spike** is observed.

After the 11^{th} time step the reward would again go down due to the uncertainty increased by the now unequal Confidence Bound Term $c\sqrt{\frac{\log t}{N_t(a)}}$. The average reward value finally rises when the algorithm has a lot more certainty about the optimal arms.

Problem 7

Optimal Action



The simulation was carried over for 2000 runs and 1000 time-steps. For the plots :

- The mean for the test-bed was +4 and the variance was 1
- $\overline{R_t}$ was calculated using incremental updates, i.e. $\overline{R}_{t-1}$
- $H_t(a)$ was calculated using the gradient update rules (from Book)
- $\pi_t(a)$ was calculated using the soft-max distribution

The Optimal Action performs quite better for $\alpha = 0.1$, with baseline as compared to the similar α without the base line i.e. $\overline{R_t} = 0$.

***