Reinforcement Learning

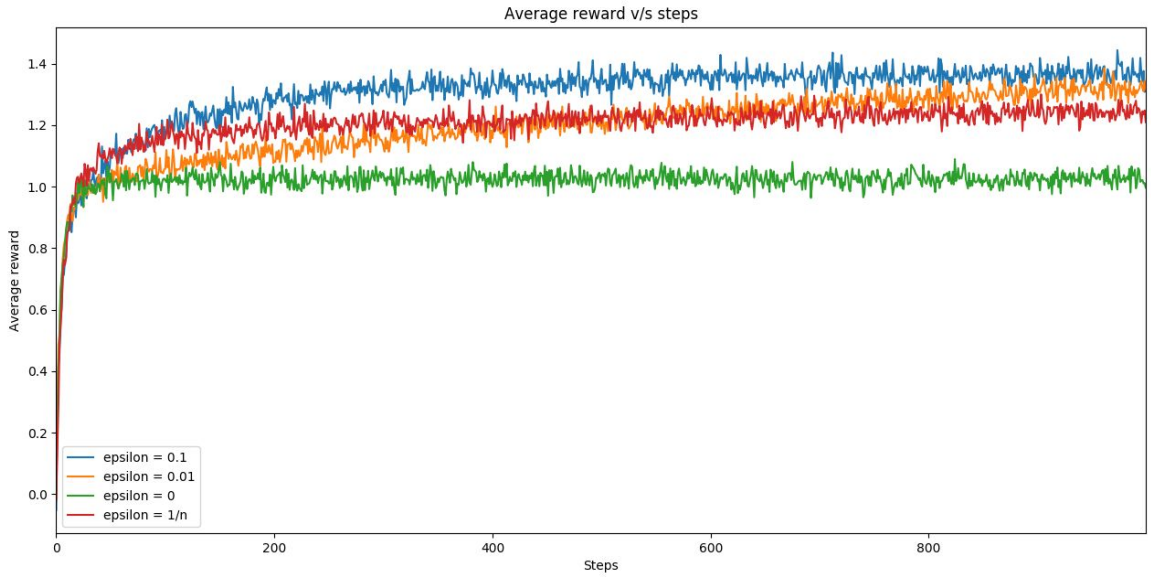
Assignment 1

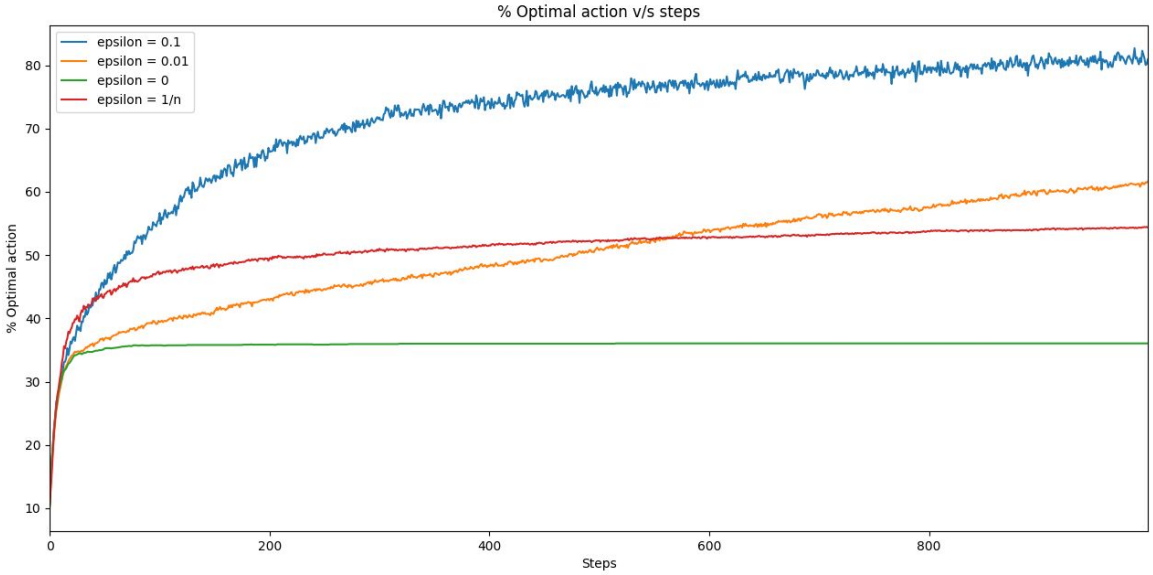
Setu Gupta (2018190)

Q1.)

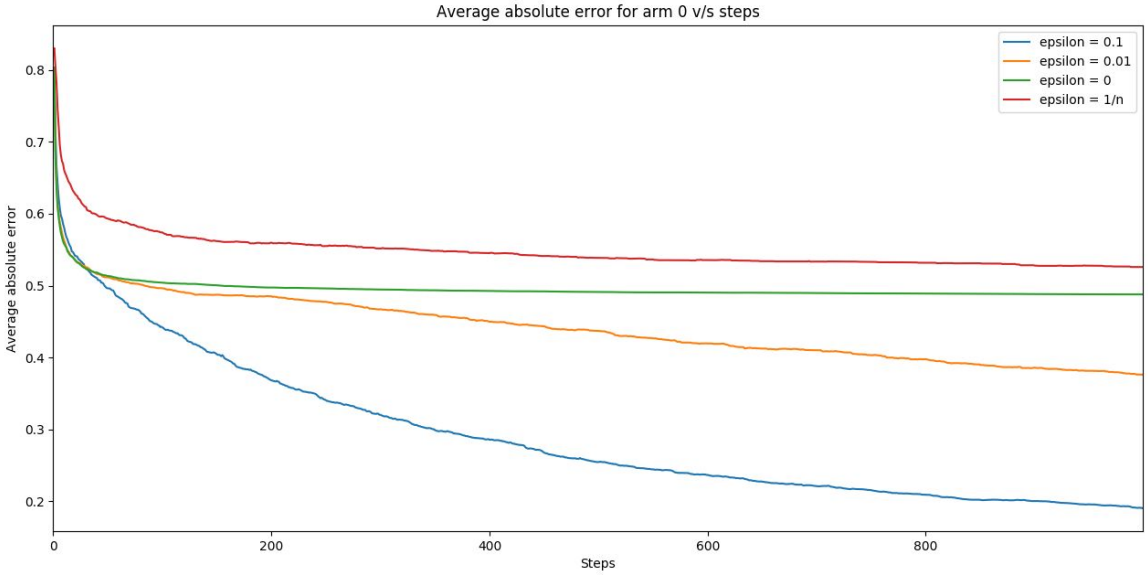
epsilon = 1/n is a sequence of epsilon which satisfies Eq 2.7 of SB. Here n is the number of current timestep.

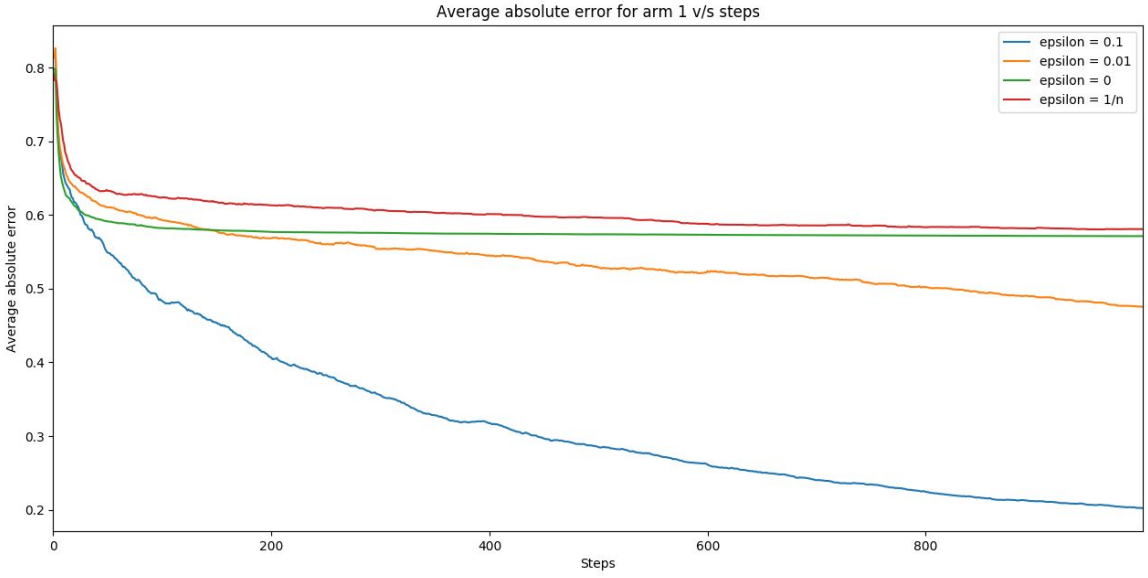
Mean of 10 arms were picked from normal distribution with mean = 0, variance = 1. Variance of each arm was 1. Training was done using epsilon-greedy. Experiment was averaged over 2000 iterations each running for 1000 steps.

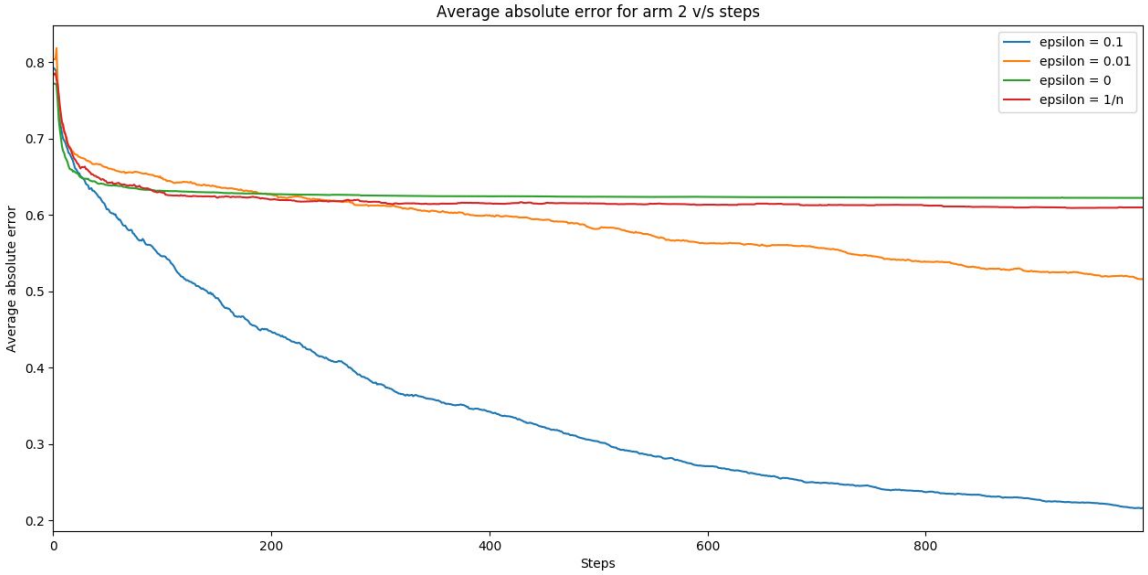


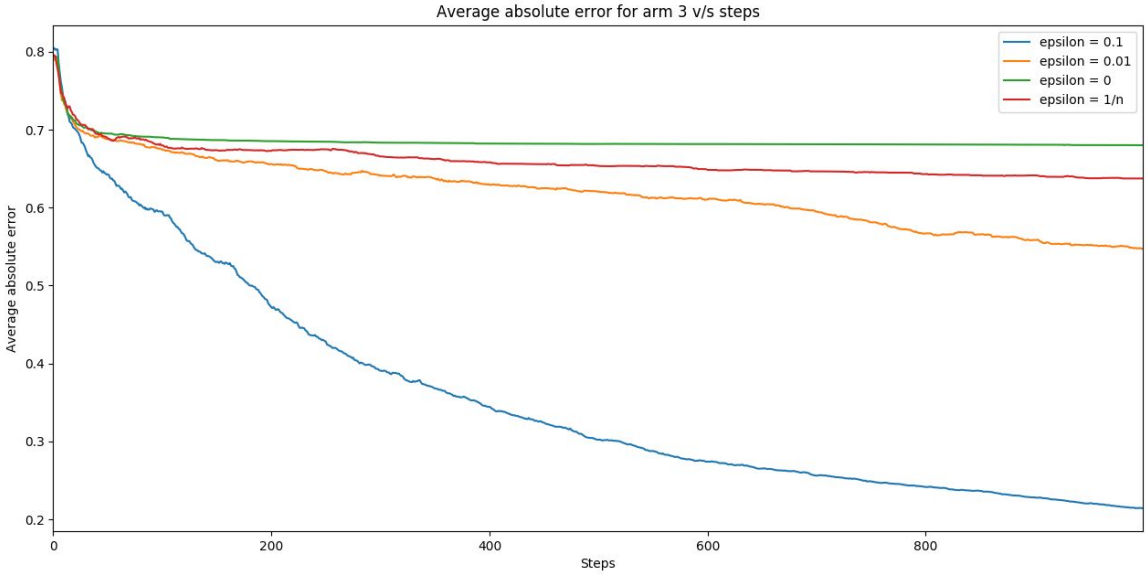

% Optimal action v/s steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
% Optimal action
Steps
10
20
30
40
50
60
70
80
0
200
400
600
800

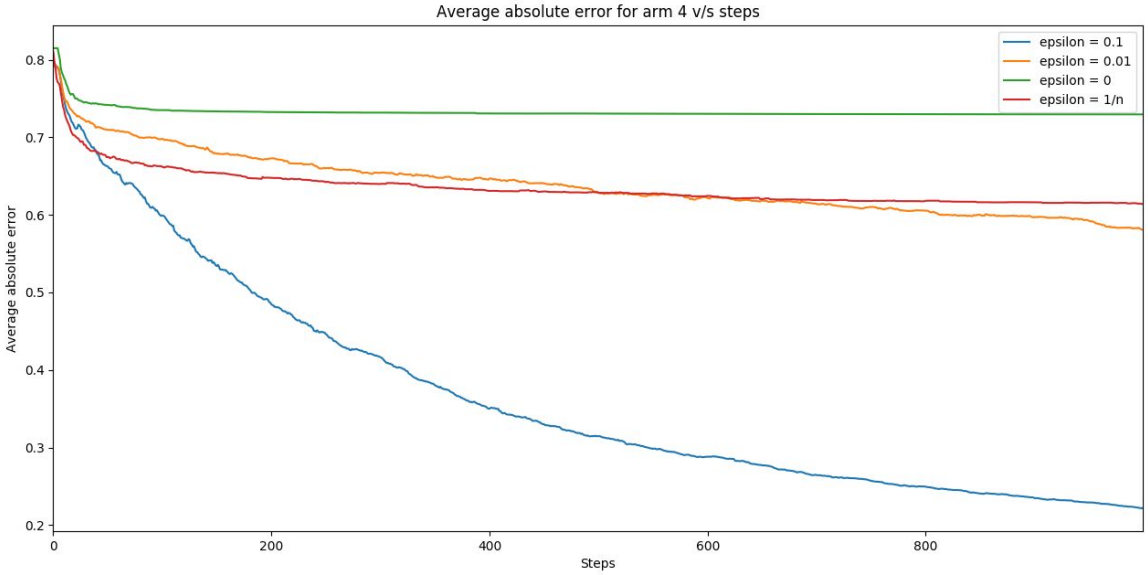
The following plots are for each arm numbered from 0 to 9 inclusive. They show the average absolute error of each arm across 2000 experiments each running for 1000 steps (same as before).

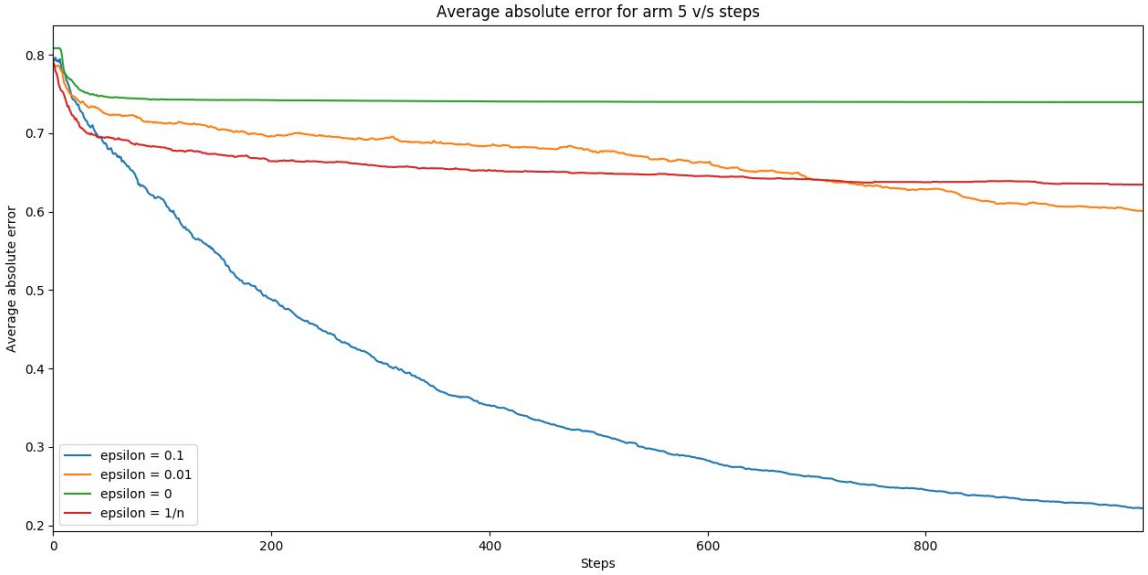


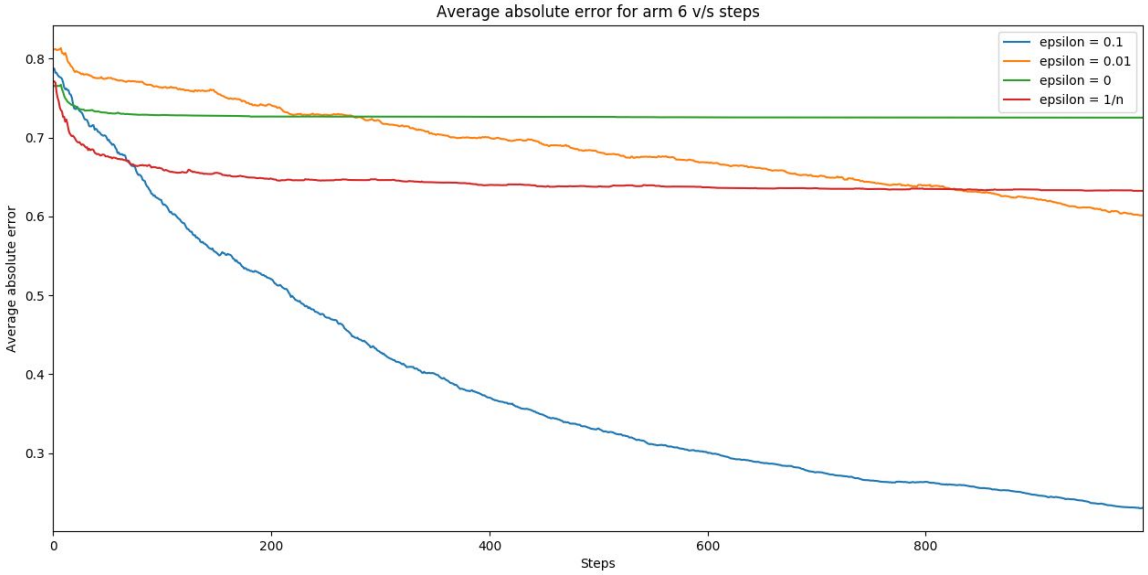


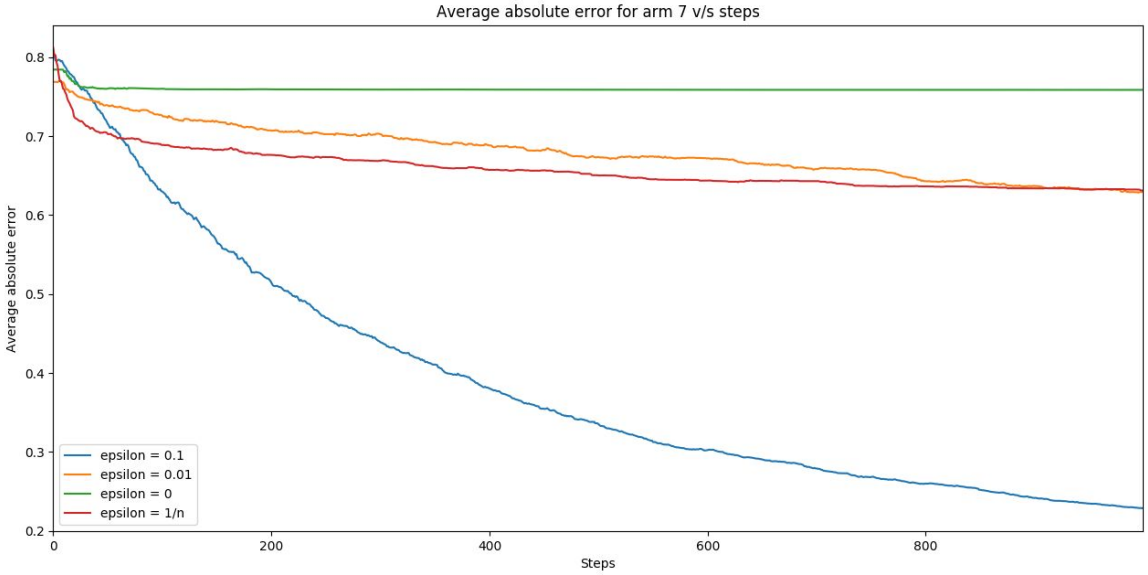

Average absolute error for arm 2 v/s steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
Average absolute error
Steps
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
200
400
600
800

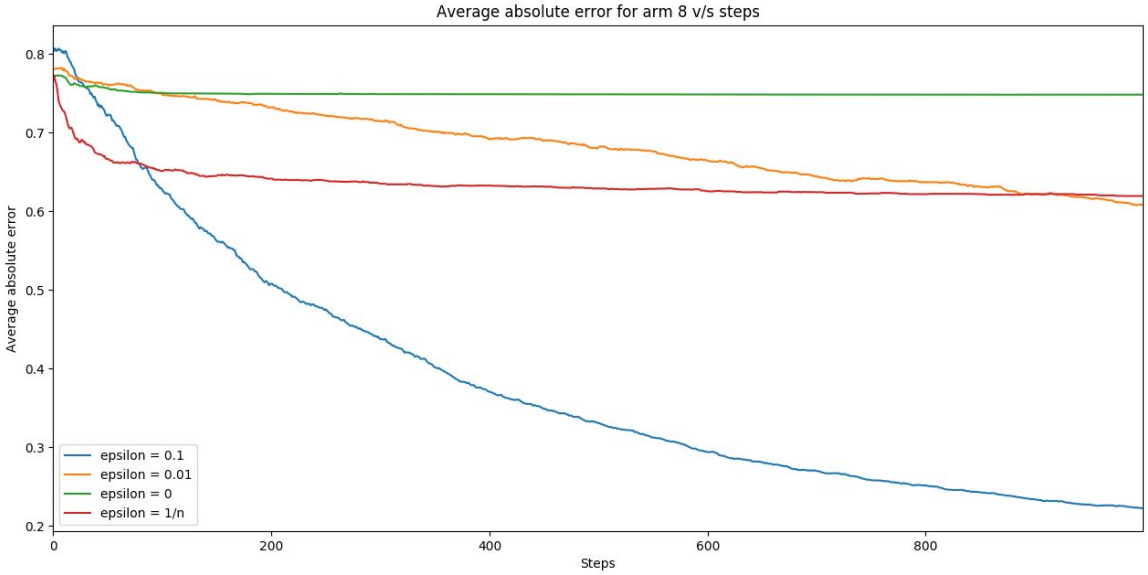

Average absolute error for arm 3 v/s steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
Average absolute error
Steps
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
200
400
600
800

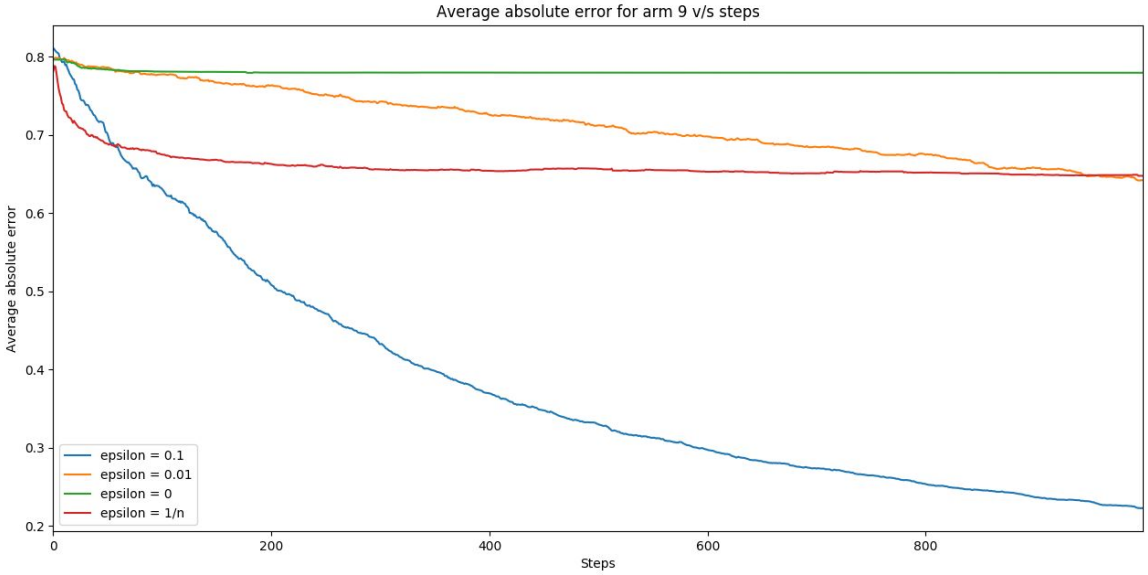

Average absolute error for arm 4 v/s steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
Average absolute error
Steps
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
200
400
600
800


Average absolute error for arm 5 v/s steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
Average absolute error
Steps
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
200
400
600
800

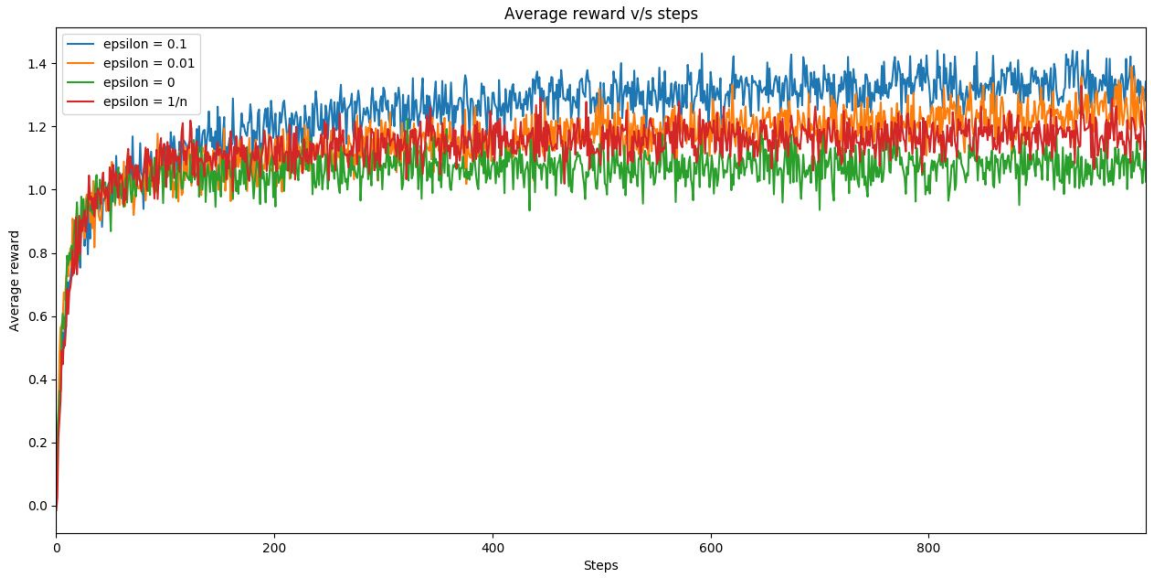

Average absolute error for arm 6 v/s steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
Average absolute error
Steps
0.8
0.7
0.6
0.5
0.4
0.3
0
200
400
600
800

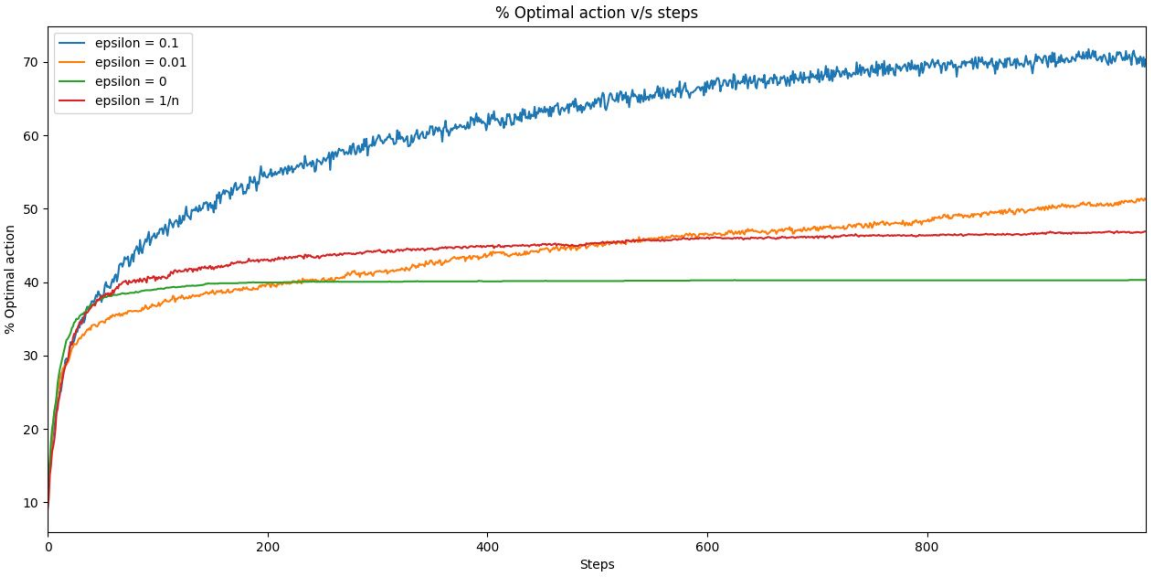

Average absolute error for arm 7 v/s steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
Average absolute error
Steps
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
200
400
600
800


Average absolute error for arm 8 v/s steps
Average absolute error
Steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
200
400
600
800

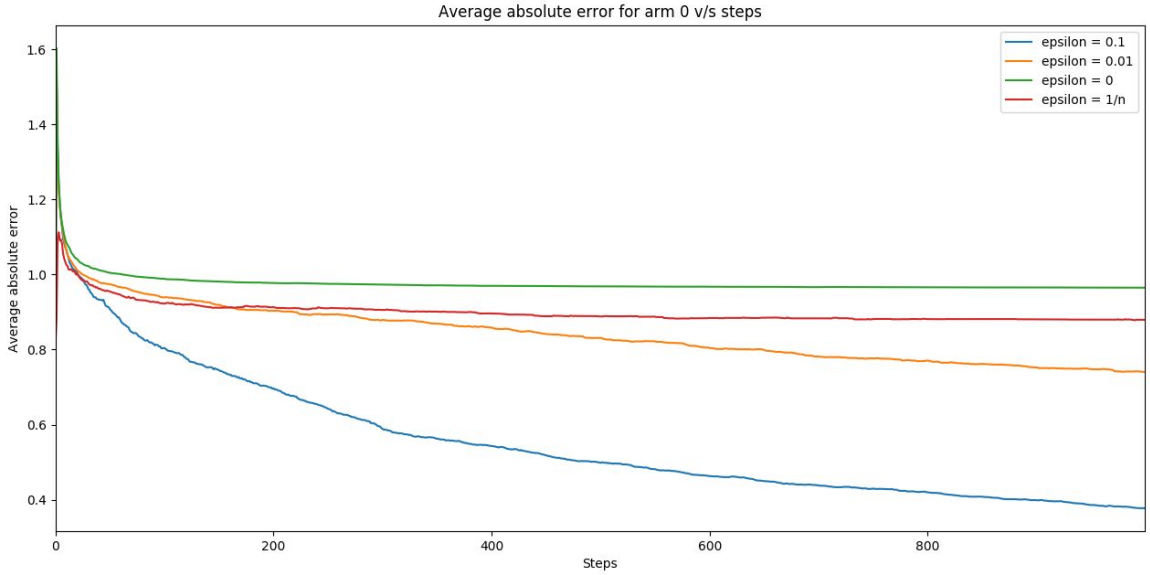

Average absolute error for arm 9 v/s steps
Average absolute error
Steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
200
400
600
800

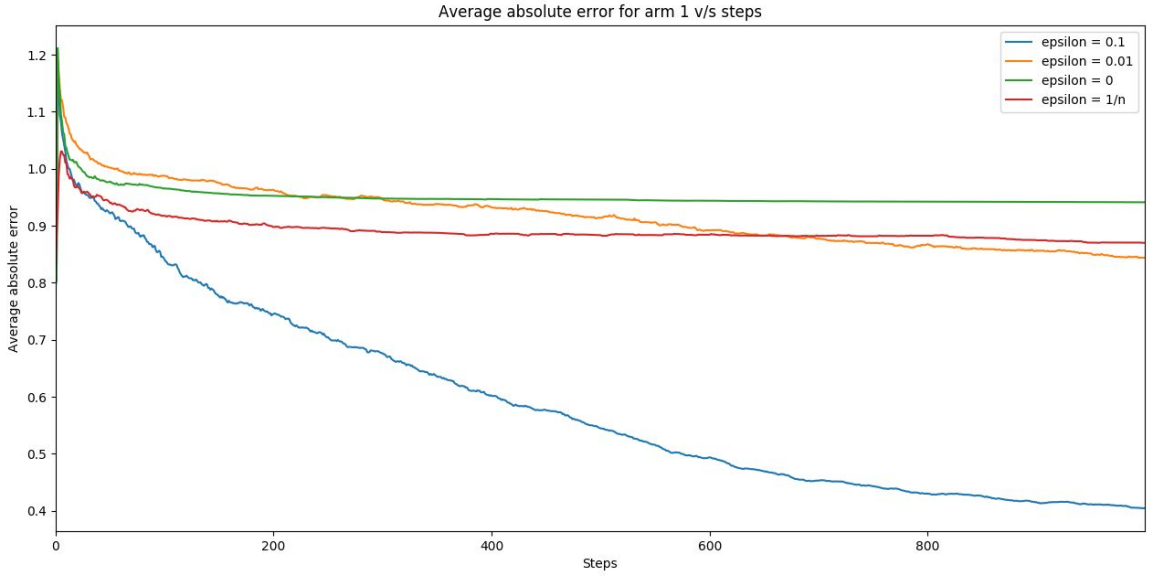
Q2.) epsilon = 1/n is a sequence of epsilon which satisfies Eq 2.7 of SB. Here n is the number of current timestep.
Mean of 10 arms were picked from normal distribution with mean = 0, variance = 1. Variance of each arm was 4. Training was done using epsilon-greedy. Experiment was averaged over 2000 iterations each running for 1000 steps.

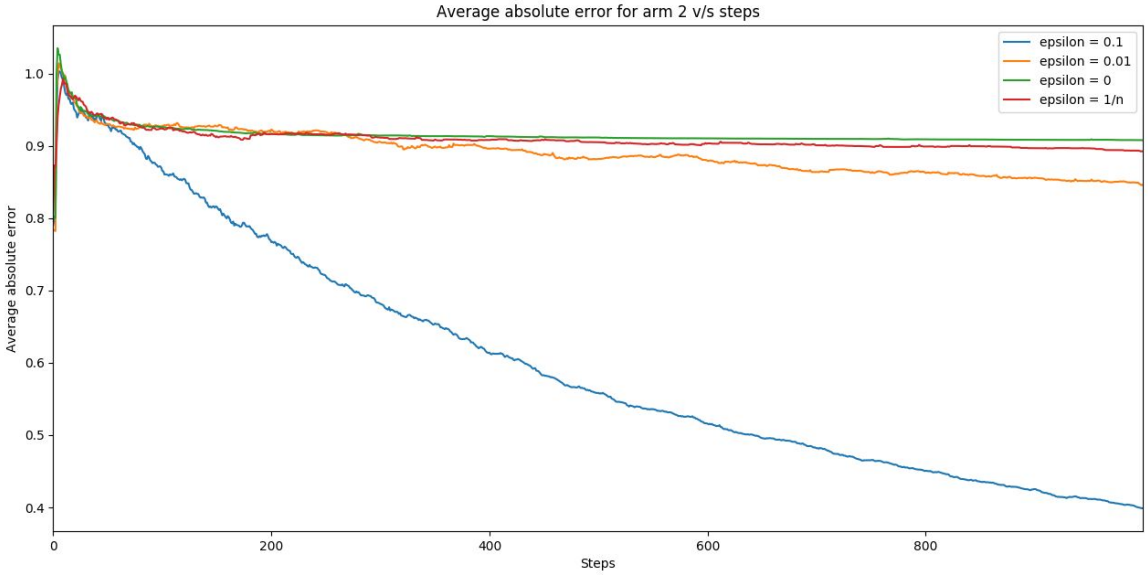


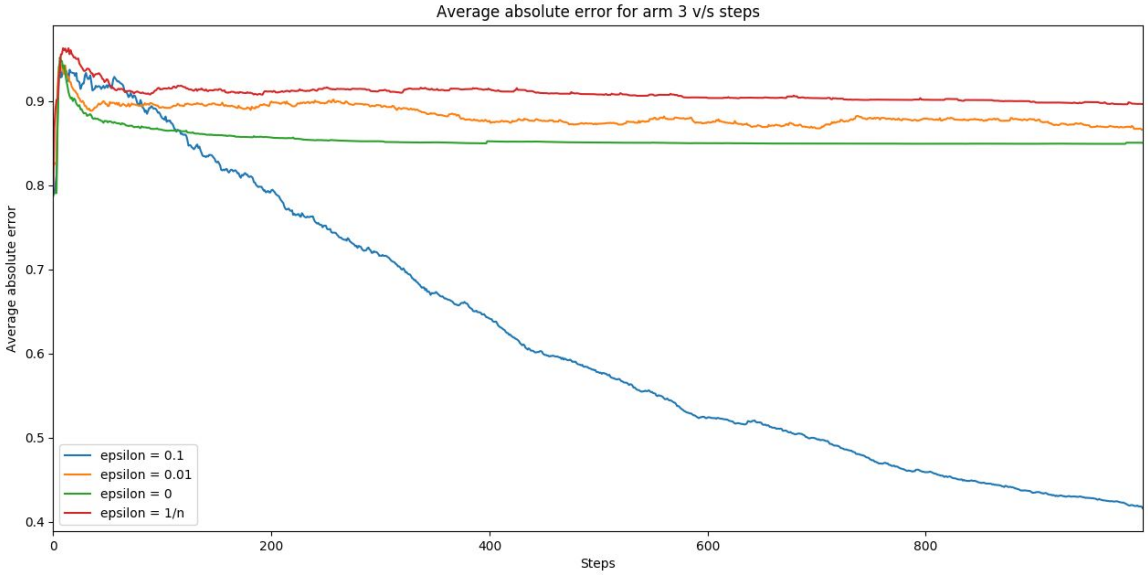


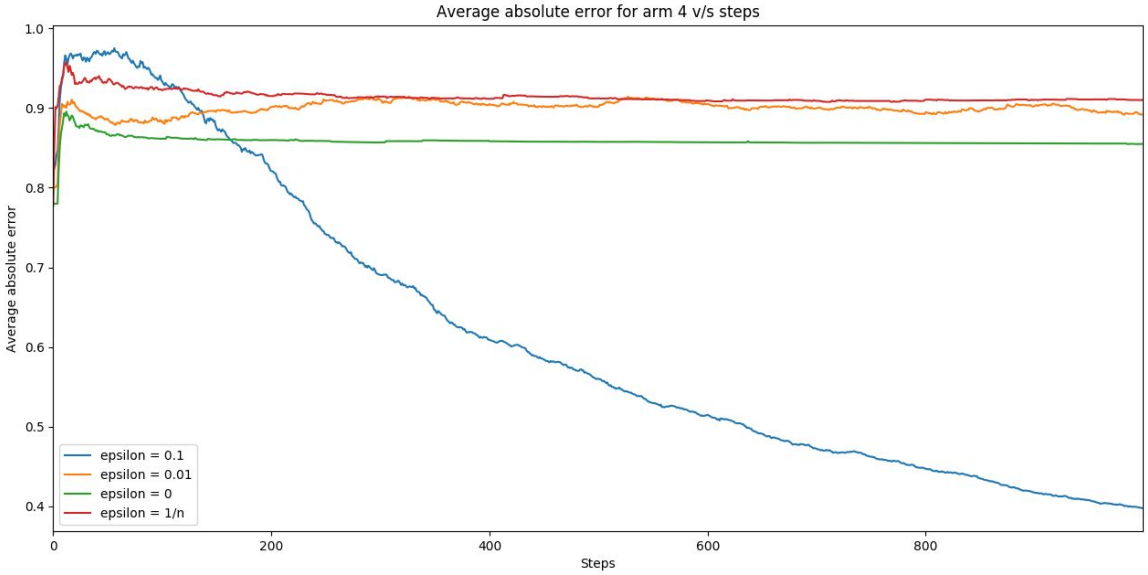
The following plots are for each arm numbered from 0 to 9 inclusive. They show the average absolute error of each arm across 2000 experiments each running for 1000 steps (same as before).

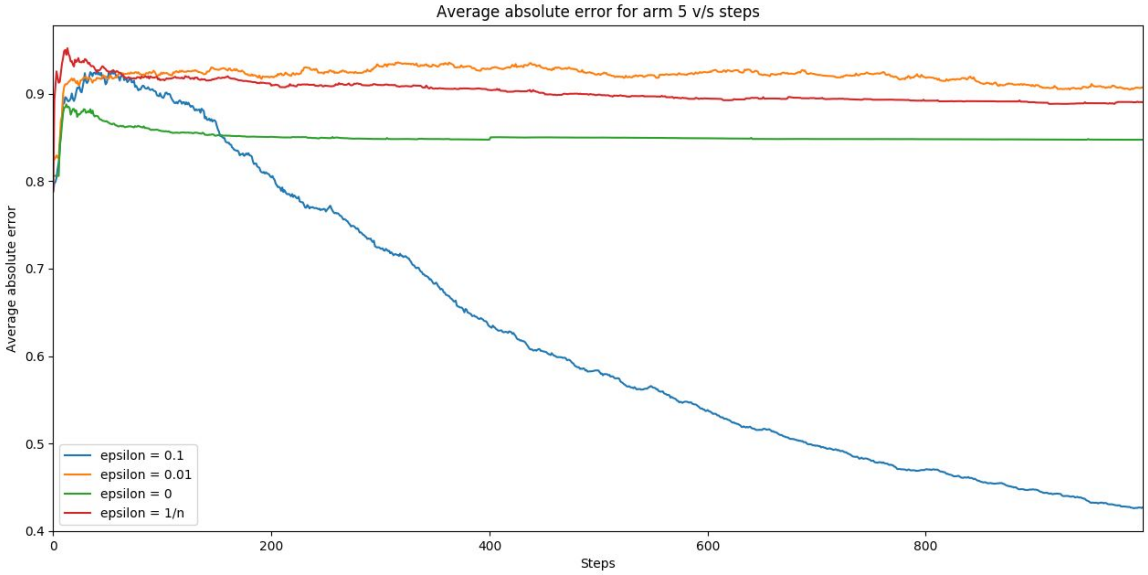


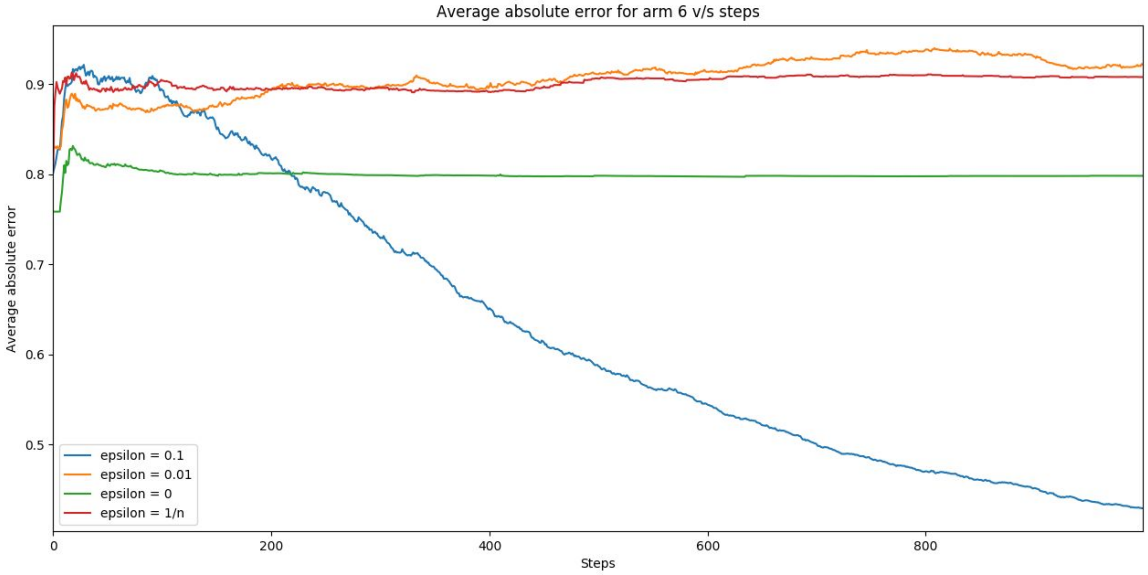


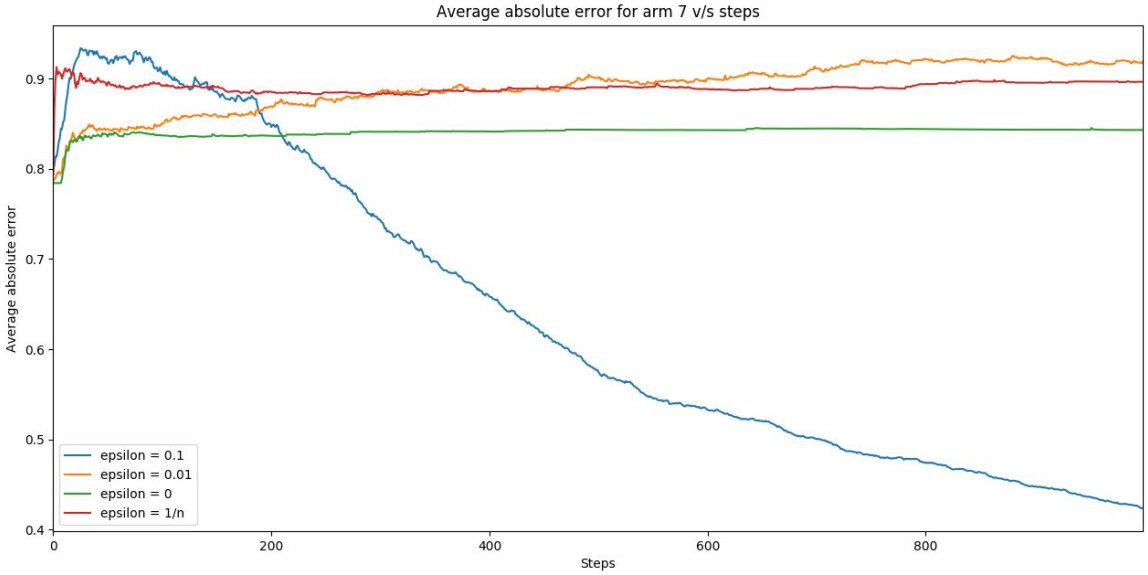

Average absolute error for arm 2 v/s steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
Average absolute error
Steps
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
200
400
600
800

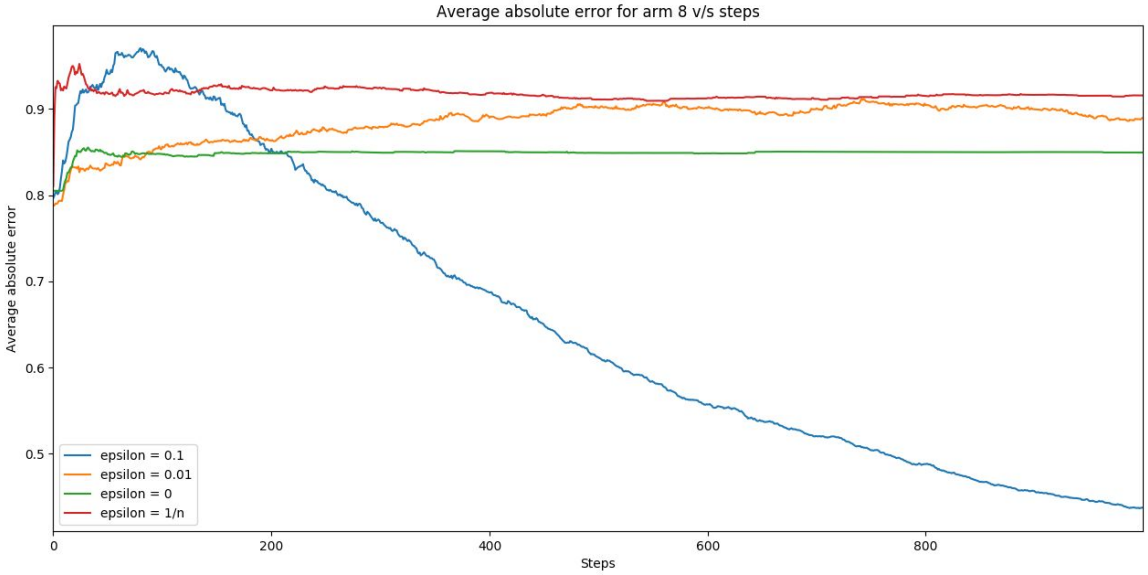

Average absolute error for arm 3 v/s steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
Average absolute error
Steps
0.9
0.8
0.7
0.6
0.5
0.4
0
200
400
600
800

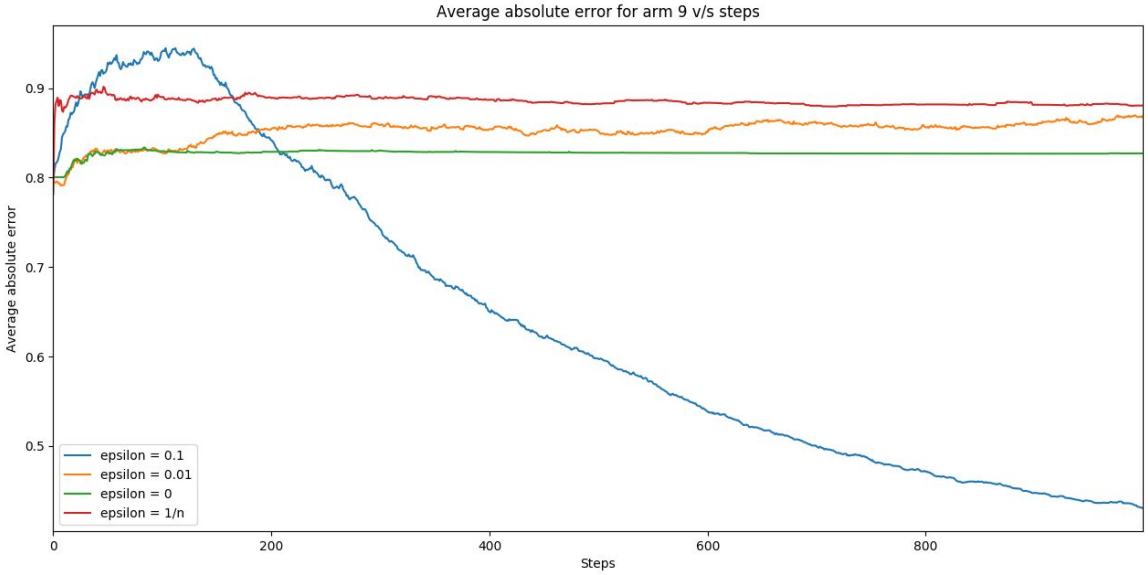

Average absolute error for arm 4 v/s steps
Average absolute error
Steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
200
400
600
800


Average absolute error for arm 5 v/s steps
Average absolute error
Steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
0.9
0.8
0.7
0.6
0.5
0.4
0
200
400
600
800

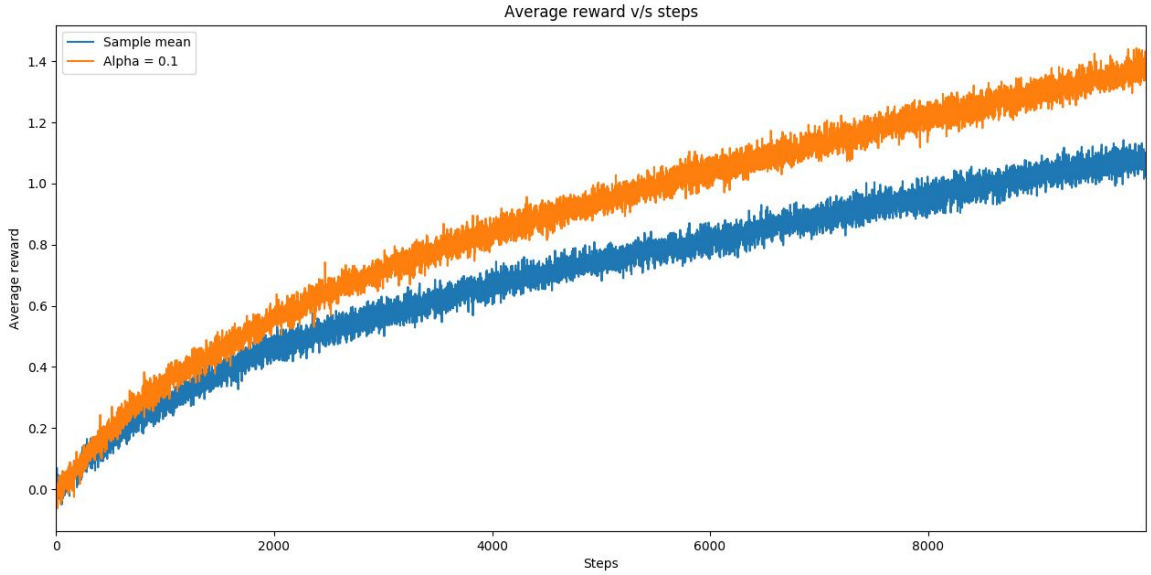

Average absolute error for arm 6 v/s steps
Average absolute error
Steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
0.5
0.6
0.7
0.8
0.9
0
200
400
600
800

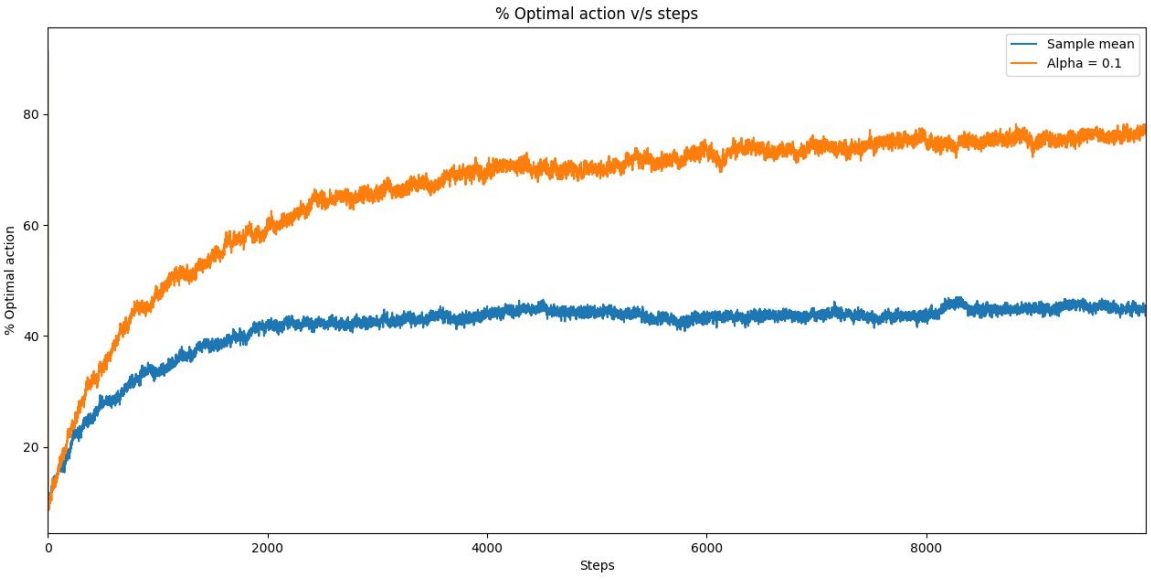

Average absolute error for arm 7 v/s steps
Average absolute error
Steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
0.4
0.5
0.6
0.7
0.8
0.9
0
200
400
600
800


Average absolute error for arm 8 v/s steps
Average absolute error
Steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
0.5
0.6
0.7
0.8
0.9
0
200
400
600
800

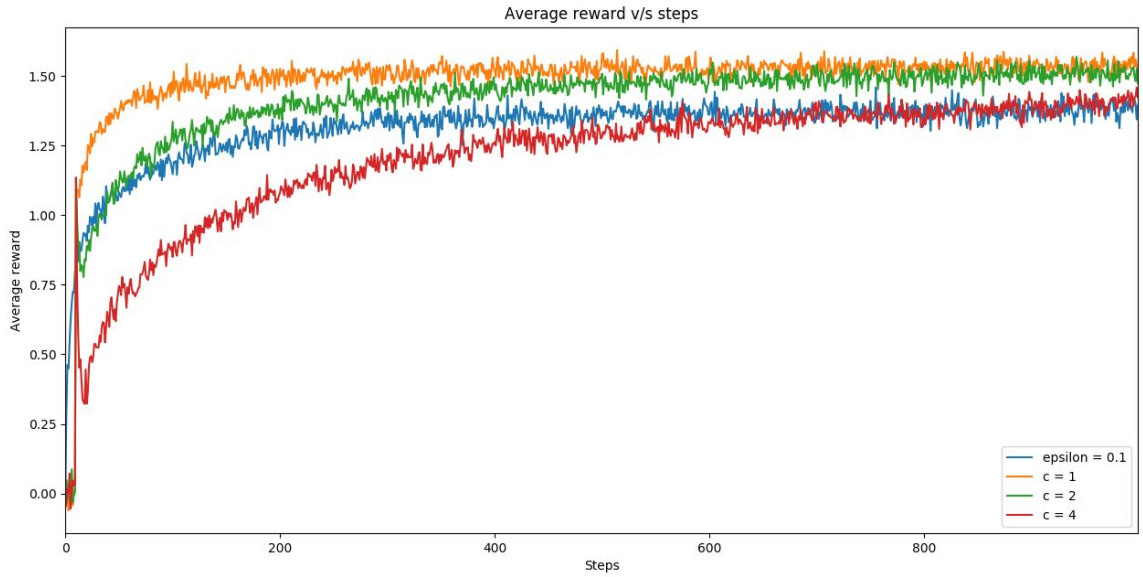

Average absolute error for arm 9 v/s steps
Average absolute error
Steps
epsilon = 0.1
epsilon = 0.01
epsilon = 0
epsilon = 1/n
0.5
0.6
0.7
0.8
0.9
0
200
400
600
800

Q5.) The following plots are for non stationary problems as described in Ex 2.5 of SB

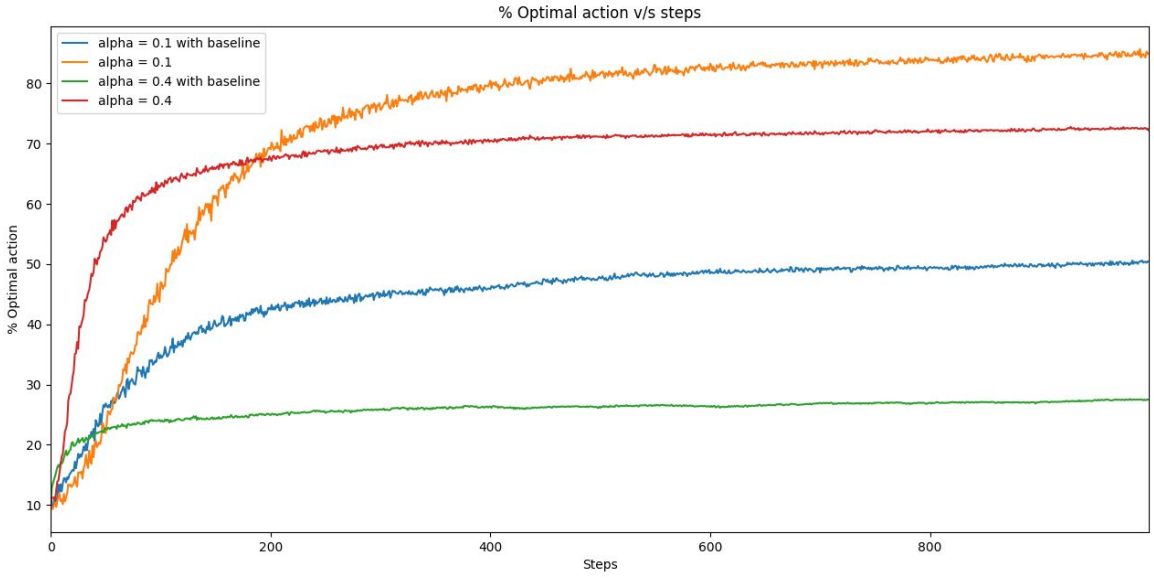




Q6.) Average performance of UCB action selection on the 10-armed testbed. The results are for c = 1,2,4 and epsilon = 0.1 for epsilon greedy.



Q7.)



Theory questions follow…

7/9/20 Reinforcement Learning ECE564

Sem-5 HW 1

Monsoon 2020 Seta Gupta (2018190

Q-3 In the long run the best performing strategy is the one which has

$P(E) = P(\text{Choosing the optimal arm as } step \to \infty)$

highest. This strategy will give highest cumulative reward in long run.

a) $\varepsilon = 0$ (Purely greedy)

This can possibly give $P(E)$ but there is a high possibility that we get stuck at some sub optimal choice of arm.

b) ε = constant.

$$P(E) = 1 - \varepsilon + \frac{\varepsilon}{|A|} = 1 + \varepsilon\left(\frac{1}{|A|} - 1\right)$$

Here $\frac{1}{|A|} - 1$ is constant $= \frac{1}{10} - 1 = 0.1 - 1 = -0.9$

$$\therefore P(E) = 1 - 0.9\varepsilon$$

$\therefore$ lower ε performs better $\therefore 0.01$

$$P(E \mid \varepsilon = 0.01) = 1 - (0.01 \times 0.9)$$
$$= 1 - 0.009$$
$$= 0.991$$

(c.) $\epsilon = 1/n$

$$P(\epsilon) = \lim_{n \to \infty} 1 - 0.9\left(\frac{1}{n}\right) = 1$$

$\therefore$ This strategy always picks the best arm as $n \to \infty$.

$\therefore$ The best performing strategy among 4 options is $\epsilon = 1/n$.

Q-4.) $Q_{n+1} = Q_n + \beta(R_n - Q_n)$

where β is step size

a.) Sample mean i.e. $\beta = 1/n$.

$\therefore Q_{n+1} = Q_n + \frac{1}{n}[R_n - Q_n]$ + a (omitting writing $Q_{n+1}(a)$ for clarity)

Consider Q_2 i.e. $n = 1$

$$Q_2 = Q_1 + \frac{1}{1}[R_1 - Q_1]$$
$$= Q_1 + R_1 - Q_1$$
$$= R_1$$

$\therefore$ After the action is chosen at least once, the dependence on Q_1 goes away.

b.) (i) Constant step-size i.e. $\beta = \alpha$

$$Q_{n+1} = Q_n + \alpha[R_n - Q_n]$$
$$= \alpha R_n + (1-\alpha) Q_n$$
$$= \alpha R_n + (1-\alpha)[\alpha R_{n-1} + (1-\alpha) Q_{n-1}]$$
$$= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 Q_{n-1}$$
$$= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 \alpha R_{n-2} \cdots$$
$$\cdots + (1-\alpha)^{n-1} \alpha R_1 + (1-\alpha)^n Q_1$$

$$= \alpha_n \sum_{i=0}^{n-1} (1-\alpha)^i \, \alpha_{n-i} R_{n-i} + (1-\alpha)^n Q_1$$

$\therefore$ Dependent on Q_1

(ii) Notice that $0 < \alpha \leq 1$

$\therefore$ for smaller α, $(1-\alpha)^n$ is larger i.e. larger dependence on Q_1.

(iii) I propose two way of dealing w/ dependence on Q_1

MI $\alpha = 1$

$\Rightarrow Q_{n+1} = Q_n + 1 \cdot [R_n - Q_n]$

$= Q_n + R_n - Q_n = R_n$

Drawback: No regard to history

MII Instead of choosing values of $Q_1(a)$ $\forall a$ manually, pull each arm one by one to obtain

$R_0(a)$ $\forall a$

Set $Q_1(a) = R_0(a)$ $\forall a$

This way we don't need to decide the values.

Drawback: Does not follow constant α strategy for first $|A|$ actions.

Q-6) $A_t = \underset{a}{\operatorname{argmax}} \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right]$

Note that initially $Q_t(a) = \lambda$ (constant) $\forall\ a$

Also $N_t(a) = 0\ \forall\ a$

$\therefore$ For the first step the value

$$\$(a) \triangleq \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right] \to \infty$$

$\therefore$ First action is chosen at random. Suppose we choose action i s.t. $1 \le i \le |A|$

$\Rightarrow \$_2(i)$ becomes finite as $N_2(i) = 1$

$\therefore$ we will not choose i this time.

By the same logic we can say that for first 10 steps we choose every action once in an arbitrary order.

Now lets look @ 11^{th} step.

$$c\sqrt{\frac{\ln(t)}{N_t(a)}} = c\sqrt{\frac{\ln(11)}{1}} \quad \cdots \quad \forall a$$

$\therefore$ This choice will purely be influenced by $Q_{t+1}(a)\ \forall\ a.$

Now across 2000 runs, the probability of $Q_{11}(a^*)$ being higher for truly optimal action a^* is higher

$\therefore$ @ 11^{th} step since $P\left[Q_{11}(a^*) > Q_{11}(a) \quad \forall\, a \in A, a \neq a^*\right]$

is high, we pick a^* which explains the spike.

@ 12^{th} step $S_{12}(a^*) = \left[Q_{12}(a^*) + c\sqrt{\frac{\ln(12)}{2}}\right]$

Note that the factor $c\sqrt{\frac{\ln(12)}{N_t(a)}}$ $\forall a$ is lowest for a^* as $N_t(a^*) = 2$ whereas $N_t(a)$ $\forall a \neq a^*$ is 1, $\therefore$ it is $1/\sqrt{2}$ times less for a^*.

Now this low value may be overpowered by $Q_{12}(a^*)$ but if c is less, this is unlikely. Hence we get a dip.

This is also the reason that the spike is more pronounced for higher c as higher the c, more will $c\sqrt{\frac{\ln(t)}{N(t)}}$ over power $Q_t(a)$.

Also note that it is most pronounced @ 11^{th} step as

at later steps, values of $S(a)$ & $A(a)$ start to converge.