The University of Vermont

CS253A QR: Reinforcement Learning: Assignment №2

Ayat Ospanov

September 11, 2018

Contents

1 2.1

If $\varepsilon = 0.5$, each second step is the greedy step. This means the probability of choosing the greedy action is at least 0.5. Further, as we do random step with the probability of 0.5 and select greedy action in this step with the probability of $\frac{1}{2}$ (because we have two actions and one of them is the greedy action), the probability of randomly selecting the greedy action is 0.25. Therefore, the overall probability of selecting the greedy action is 0.75.

We can generalize this task to the case of n options/actions and any ε. The answer is $(1 - \varepsilon) + \frac{\varepsilon}{n}$.

2 2.2

Table 1: Work of a bandit algorithm

t	1	2	3	4	5
A_i	1	2	2	2	3
R_i	-1	1	-2	2	0
$Q_i(1)$	0	-1	-1	-1	-1
$Q_i(2)$	0	0	1	-0.5	0.(3)
$Q_i(3)$	0	0	0	0	0
$Q_i(4)$	0	0	0	0	0
Choice	Random	Random	Greedy	ε case	Random

On the table 1 the work of a bandit algorithm is provided by time step. Each arrow shows the Q-value of a chosen action. As on greedy step an algorithm choose the $\arg\max_a Q_t(a)$, the only case of choosing the argmax is step 3. On the step 4 the algorithm chose the value of -0.5

which is not the argmax. It means that at this step the ε case has occured. On the other steps (1, 2, 5) as we have more that one maximum value of Q, the alogrithm chose random argmax. On these steps ε case could possibly have occurred.

3 2.3

In the long run, $\varepsilon = 0.01$ will act better as 99.1% of the time (see Section. 1) it choose the correct actions, while in the case of $\varepsilon = 0.1$ the rate of correct actions is 0.91 or 91%.

4 2.4

$$\begin{aligned} Q_{n+1} &= (1-\alpha_n)Q_n + \alpha_n R_n = \\ &= (1-\alpha_n)[(1-\alpha_{n-1})Q_{n-1} + \alpha_{n-1}R_{n-1}] + \alpha_n R_n = \\ &= (1-\alpha_n)(1-\alpha_{n-1})Q_{n-1} + \alpha_{n-1}(1-\alpha_n)R_{n-1} + \alpha_n R_n = \\ &= \prod_{i=1}^{n}(1-\alpha_i)Q_1 + \sum_{i=1}^{n}\Big(\alpha_i \prod_{j=i+1}^{n}(1-\alpha_j)R_i\Big) \end{aligned}$$

5 2.5

Orange lines – constant learning rate (step-size)
Blue lines – step-size = $\frac{1}{n}$

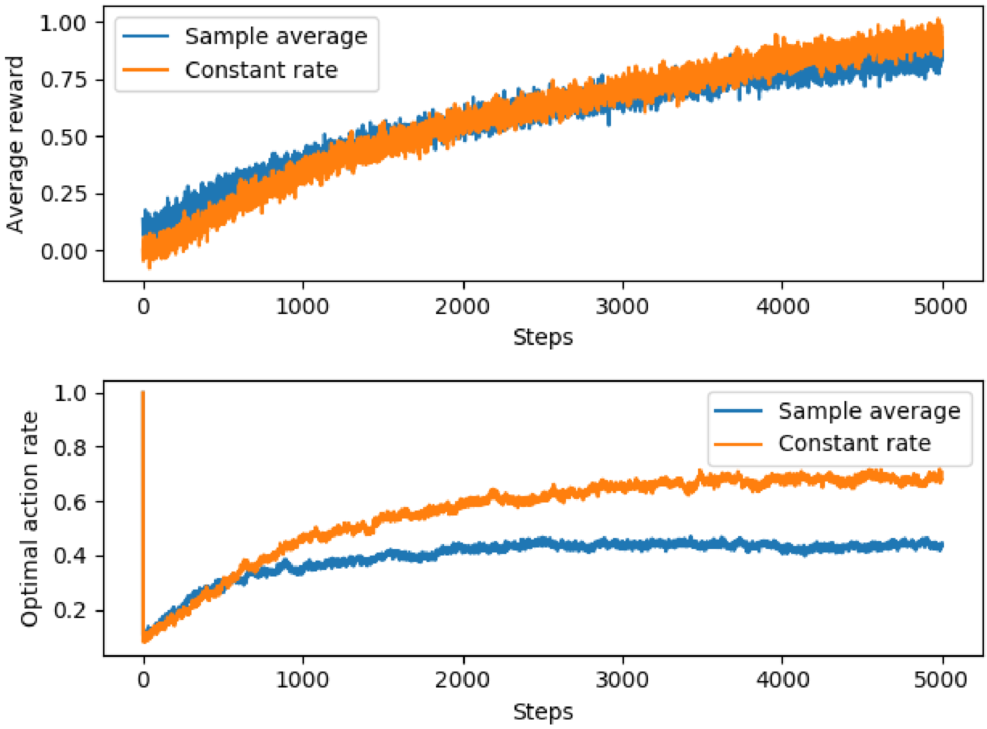

Figure 1: Initial values for $q_*(a)$ are equal

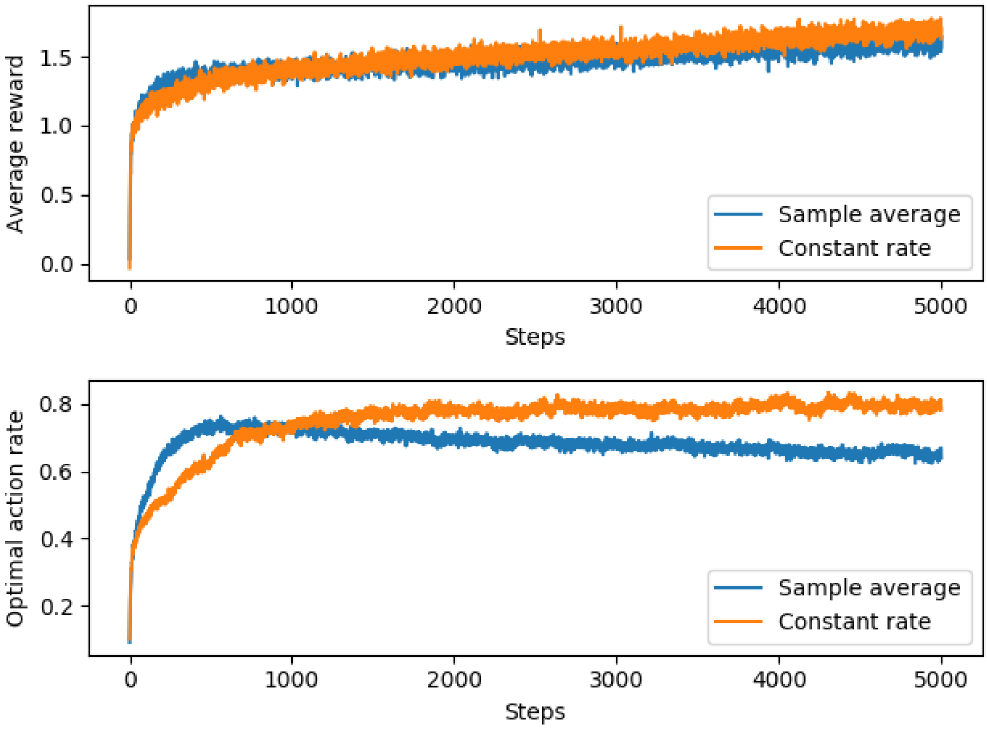

Figure 2: Initial values for $q_*(a)$ are randomly sampled from normal distribution

When initial values for $q_*(a)$ are equal, the convergence is slow. Also in this case the rate of optimal solutions is 1 in the beginning because all values are equal and therefore optimal. When initial values for $q_*(a)$ are random, convergence is faster. In both cases algorithm with a constant step-size converges faster. Moreover, in the case of random $q_*(a)$ sample average method decreases in rate of optimal actions, because it doesn't adapt to the new rewards. Thus, we were convinced that constant step-size is better for nonstationary problems.