Learning Dynamics: Assignment 3
Multi-Armed Bandits

Hakim Boulahya

Université Libre de Bruxelles
hboulahy@ulb.ac.be - 000391737

December 18, 2017

Contents

1 N-Armed Bandit

Remark About the notation, here the first arm/action starts at #0.

1.1 Exercice 1

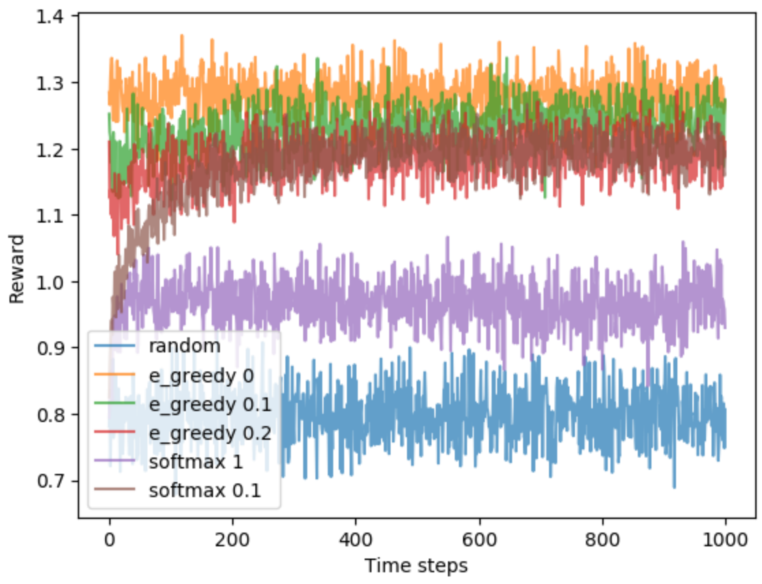
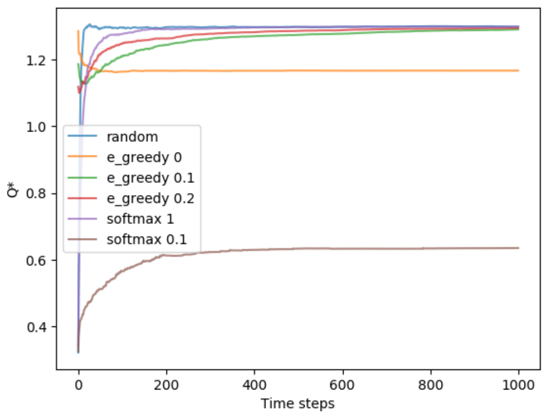
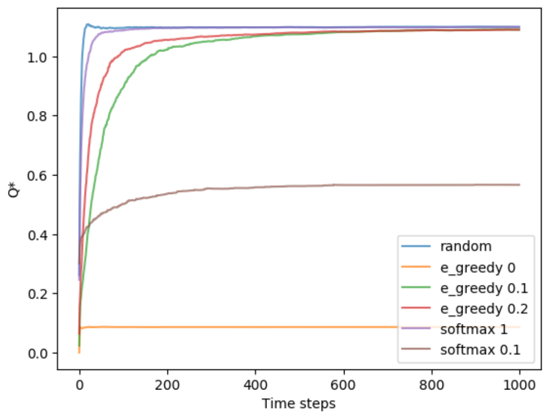


Figure 1: Average rewards for all algorithms

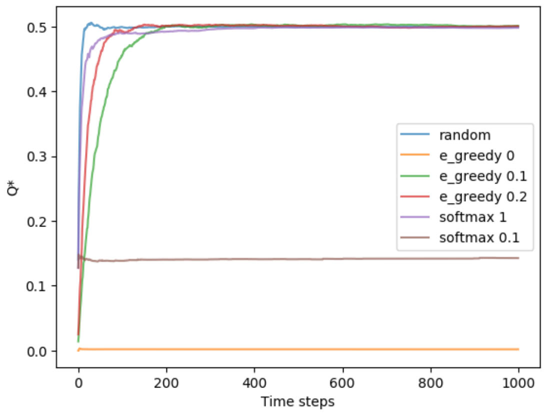
We can see that only exploring *i.e.* doesn't perform well, and the reward stay low. The boltzmann distribution of softmax with a temparature of 1 also has a poor performance compared to the other methods. Using softmax with a temparature of 0.1 gives better results. Using e-greedy with small value, i.e. will likely choose the action with the best optimum, tends to better results. egreedy with 0 seems to arrives *directly* to the best arm. We can see on the graph that when ϵ is bigger it usually takes more time to get to the best arm. Softmax with a temperature of 0.1 does tends to the best arm, but takes more time.



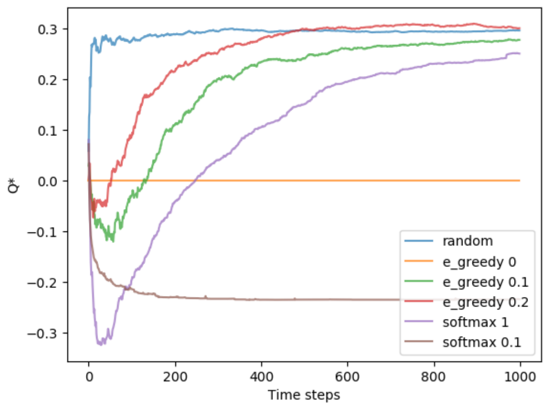
(a) $Q_{a_0}(t)$



(b) $Q_{a_1}(t)$



(c) $Q_{a_2}(t)$



(d) $Q_{a_3}(t)$

Figure 2: Plot per arm showing the $Q^*_{a_i}$ of that action along with the actual Q_{a_i} a i estimate over time with $\mu = (1.3, 1.1, 0.5, 0.3)$, $\sigma = (0.9, 0.6, 0.4, 2.0)$

2 Q-values estimation

Figure 2 shows the estimation of the Q-values for all algorithms.

Random

Since random methods is full exploration, we can see that for all Q-values the estimation of the $Q^*_{a_i}$ are accurate.

2.1 ϵ-greedy

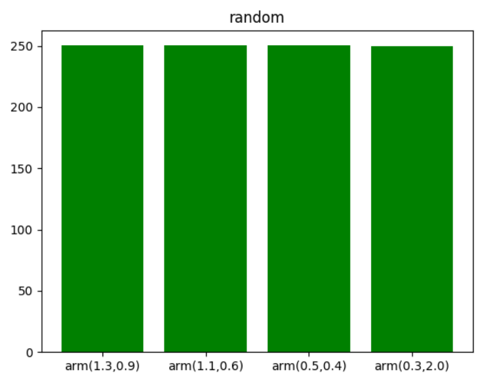
$\epsilon = 0$ Using a *only* greedy exploration, we can see that only the best actions have Q-values different from 0. Actually the best action #0 tends to be estimated fast enough. For action #1, the Q-values is bigger than 0 because at the beginning of the learning, the algorithm is still looking for the greedy action, therefore action #1 could be the best one, until the algorithm find that #action 0 is better. The two other actions, are never explored since the it is a full greedy method.

$\epsilon = \{0.1, 0.2\}$ The estimation is more accurate when $\epsilon > 0$ *i.e.* when the action are also selected randomly which allows exploration. We can see that by using a non-null ϵ, the estimation of the Q-values overtime tends to be accurate. An intersting observation is that the estimation of the Q-values is faster (the accuration rate) when ϵ is larger. This is logical, since more is large, more the action are selected randomly.

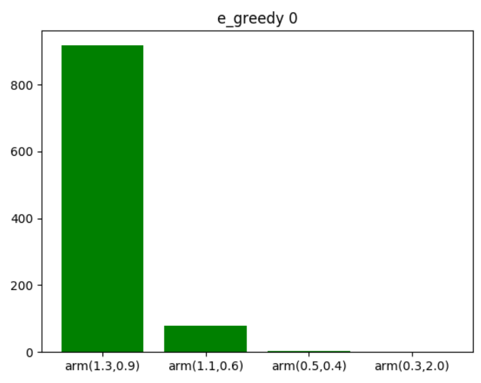
Softmax

$\tau = 1$ Using $\tau = 1$ (a large enough temperature) the Q-values estimation follows the same pattern as for any exploration methods like random or $\epsilon = 0$. We can see that except for the last action the curves are close to the random ones, which confirm the exploration nature of a large temperature. The fact that the curves of the last action #3 are more biased is because the standard deviation is larger that for the other actions, which make the Q-values estimation harder to make.

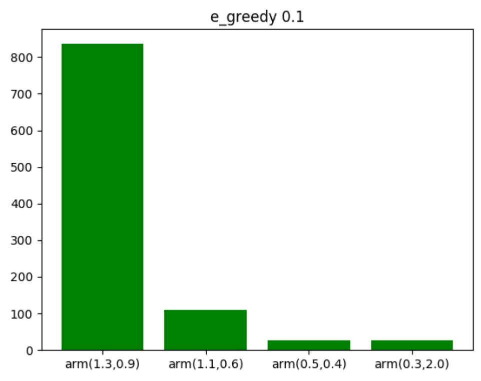
$\tau = 0.1$ Using a smaller temperature for the softmax selection method, we have a more greedy action selection. This action selection method is the worst one that estimate Q-values (excluding the full exploiting method 0-greedy). We can see that for the best arms #0 and #1, the estimations are larger than for there worst arms, but it is *far* from the actual Q^*. This can be explained by the nature of the softmax selection method with a small τ: it sticks to the best arm found and doesn't explore much to estimate the correct Q-values overtime.



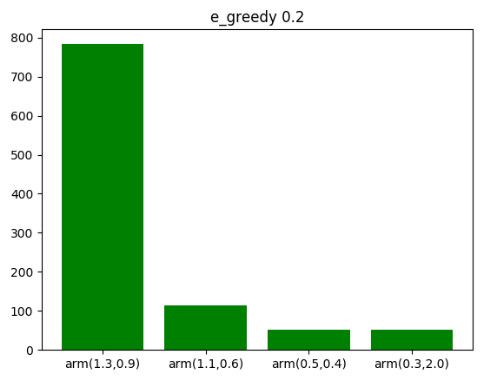
(a)



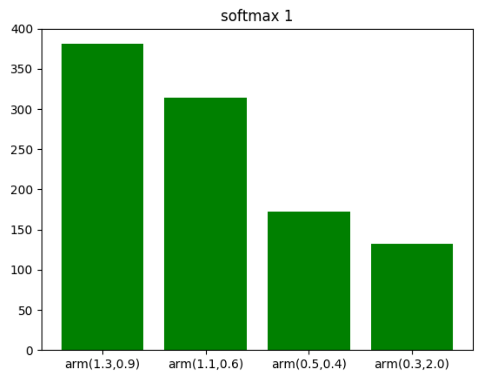
(b)



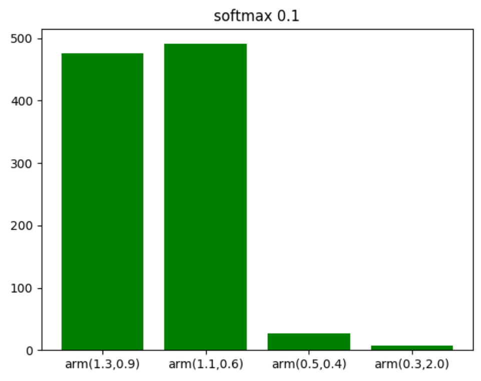
(c)



(d)



(e)



(f)

Figure 3: Histograms showing the number of times each action is selected per selection strategy with $\mu = (1.3, 1.1, 0.5, 0.3)$, $\sigma = (0.9, 0.6, 0.4, 2.0)$

3 Histograms discussion

Figure 3 shows the histograms for all algorithms.

Random

The action selections, shown in Figure 3a, for the random selection method is equidistributed. Since all actions have the same probablity it is logical that the selection is equidistributed when there $t \to \infty$.

ϵ-greedy

Figures 3b 3c 3d shows the histograms of action selections for ϵ-greedy selection methods.

$\epsilon = 0$ With a null ϵ, the action selection will be greedy *i.e.*, and since we have a pretty small standard deviation, the best action #0 is always chosen when found by the algorithm. This means that there is no exploration, the method chooses the best arm directly. The fact that the other actions are chosen, is that at the beginning it is necessary to explore a little bit to find the best action.

$\epsilon = \{0.1, 0.2\}$ It follows the same pattern that when ϵ is null, except that here it is possible for the methods to explore more *i.e.* choosing other actions than the best one. This is why the bar of other actions are higher when ϵ is bigger. If $\epsilon = 1$ the bars will be as shown for the random methods in Figure 3a.

Softmax

Figures 3e 3f shows the histograms of action selections for softmax selection methods.

When temperature $\tau = 1$, the action selections are more randomize *i.e.* the exploration is more noticable. An interesting observation in Figure 3f is that when $\tau = 0.1$, *i.e.* the action selection is more greedy, softmax algorithm tends to hesitate to choose the best actions. The number of times that the 2 best actions, action #0 et #1, are chosen is more or less equal.

3.1 Exercice 2

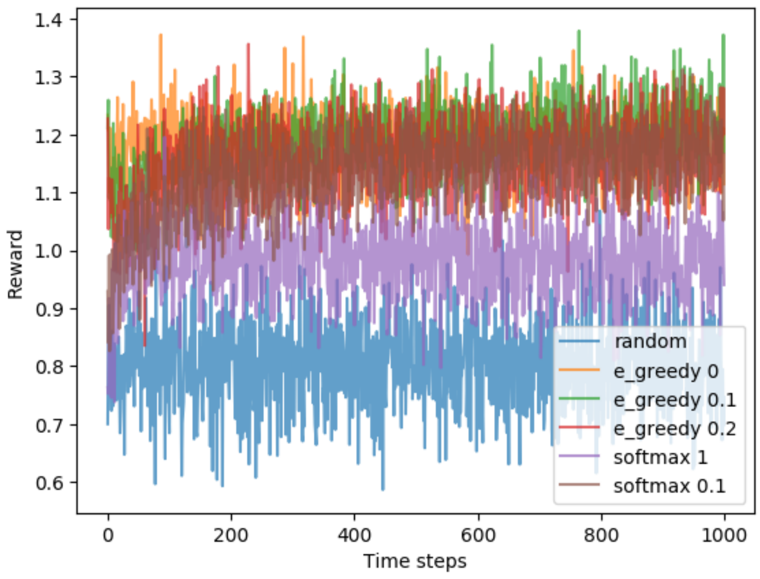
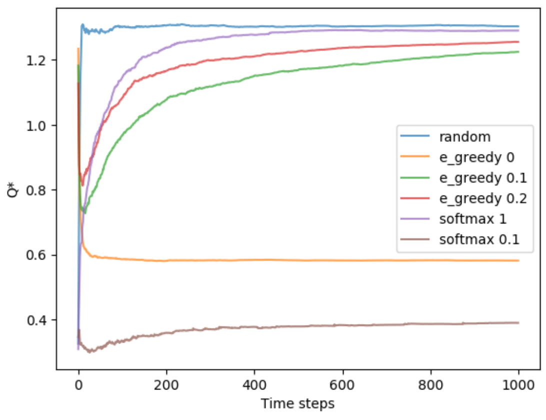


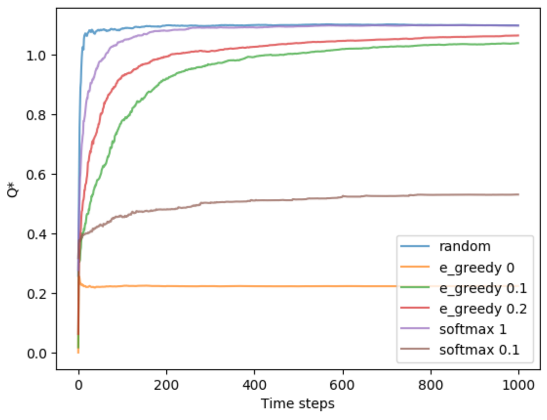
Figure 4: Average rewards for all algorithms

The results order of the algorithm that perform best doesn't really changes. By doubling the standard deviation, we only allowed the algorithms to provide a more random rewards *i.e.* a bigger range. We can see that the dynamics of the algorithms don't change.

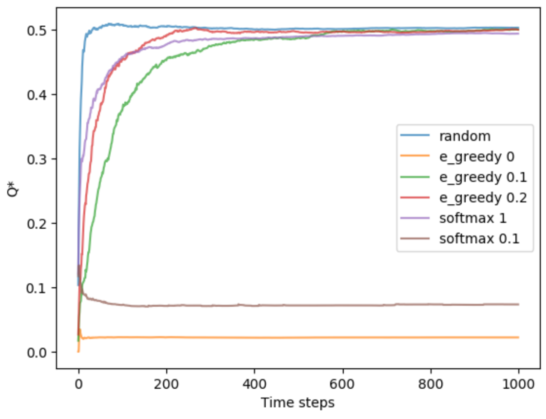
It is harder to learn because of the standard deviation that provides ar more randomize outcomes, therefore harder to find the best reward.



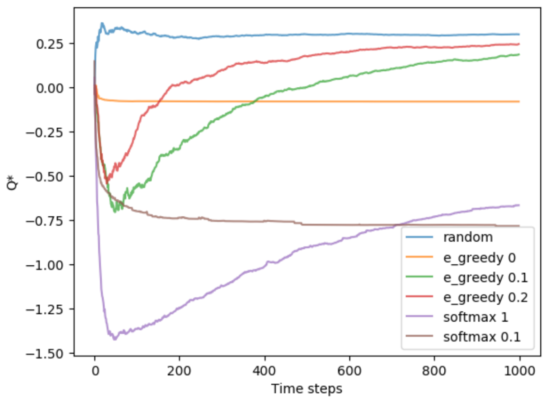
(a) $Q_{a_0}(t)$



(b) $Q_{a_1}(t)$

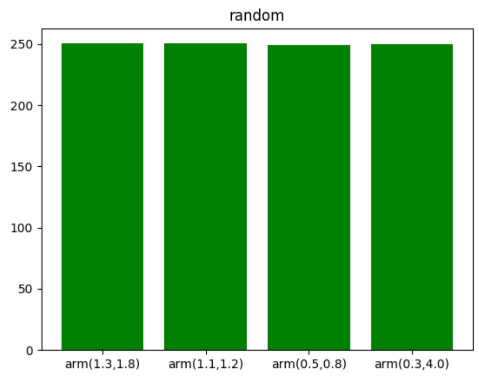


(c) $Q_{a_2}(t)$

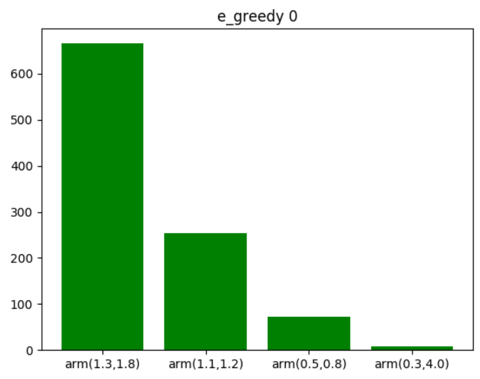


(d) $Q_{a_3}(t)$

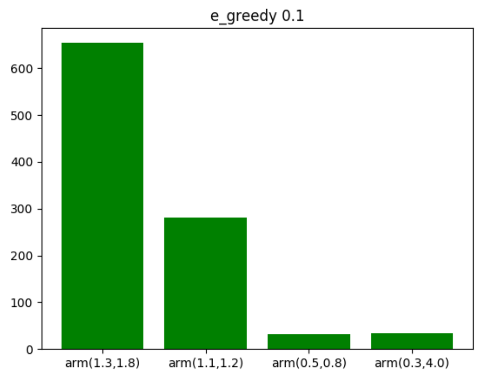
Figure 5: Plot per arm showing the $Q^*_{a_i}$ of that action along with the actual Q_{a_i} a i estimate over time with μ = (1.3, 1.1, 0.5, 0.3), σ = (1.8, 1.2, 0.8, 4.0)



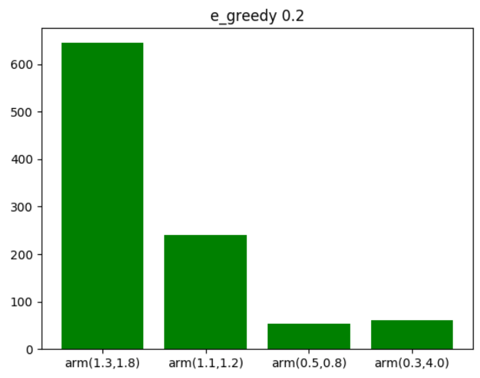
(a)



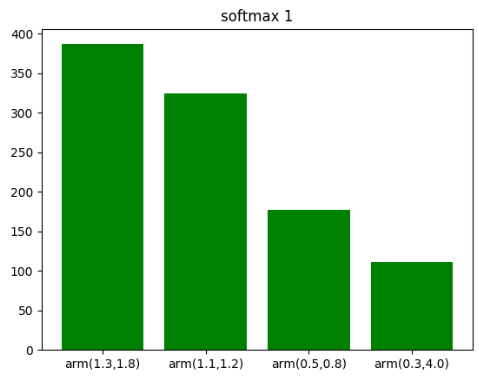
(b)



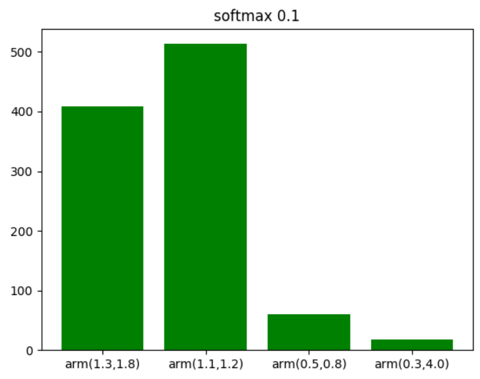
(c)



(d)



(e)



(f)

Figure 6: Histograms showing the number of times each action is selected per selection strategy with μ = (1.3, 1.1, 0.5, 0.3), σ = (1.8, 1.2, 0.8, 4.0)

The standard deviation is bigger, therefore the algorithms takes more time to find the best arm.

3.2 Exercice 3

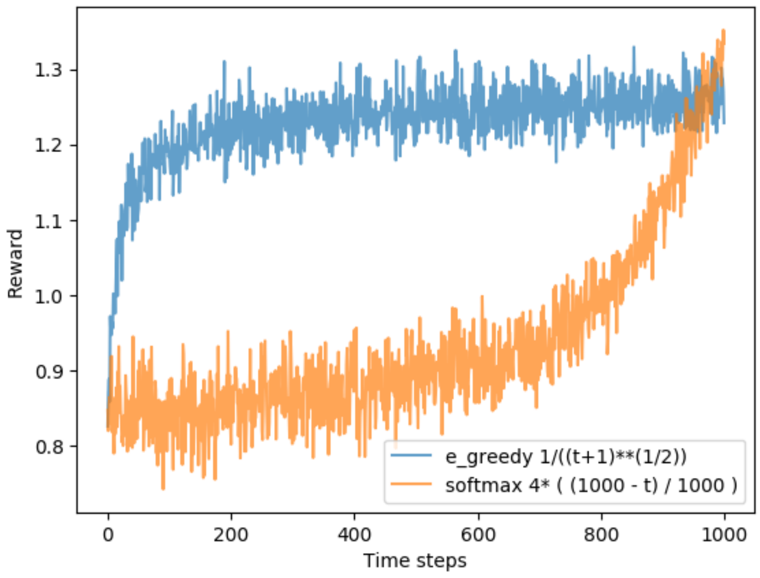
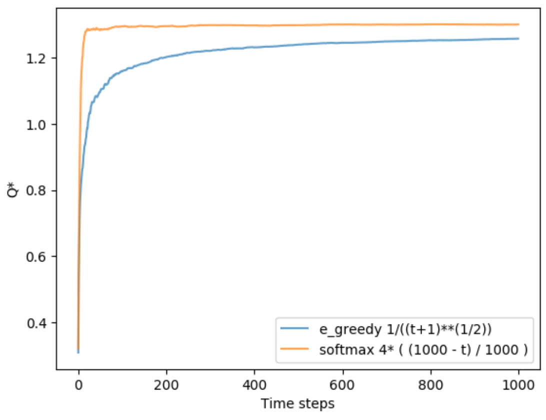


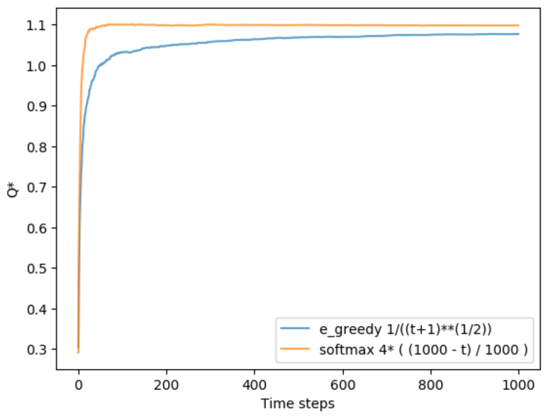
Figure 7: Average rewards for all algorithms

Dynamic egreedy doesn't seems to perform better that the statics one, the resulting rewards tends to be identical. Observing the graph shows us that it takes more time to stabilize to the best arm.

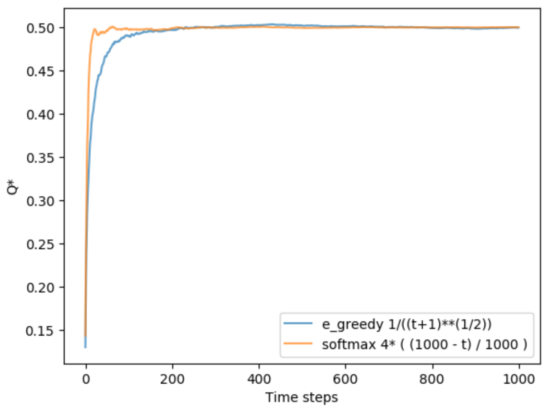
Dynamic softmax seems to be better, if you consider to long term reward. It takes to the 1000 epoch for the algorithm to find the best arm in opposition to the softmax using a a temperature of 0.1, where it is close to the best arm at around epoch 200, but not as close as the dynamic softmax, where it is closer to the best arm, when it is stabilized.



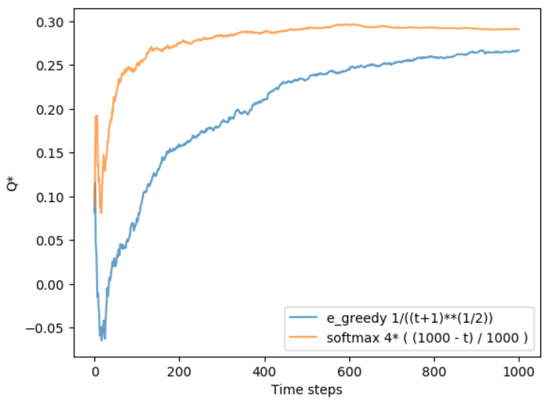
(a) $Q_{a_0}(t)$



(b) $Q_{a_1}(t)$

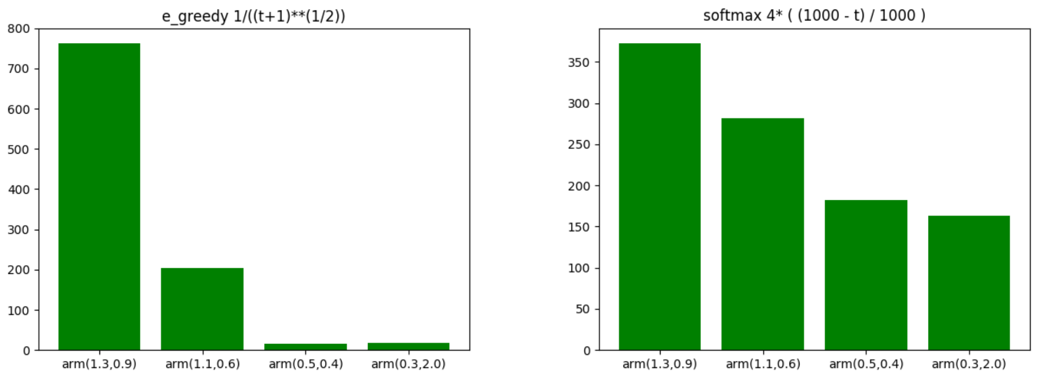


(c) $Q_{a_2}(t)$



(d) $Q_{a_3}(t)$

Figure 8: Plot per arm showing the $Q^*_{a_i}$ of that action along with the actual Q_{a_i} a i estimate over time



(a) (b)

Figure 9: Histograms showing the number of times each action is selected per selection strategy

4 Climbing game

When using the formula, it is important to have a big tau, because exp(Q/tau) will give errors if Q is too big. (See how EV(a) is calculated)

I also remarked that when using EV with $\max_{r}ewardsormax_{q}whenmin_{t}au = 0.001the11isgood,butwhen0.1or1,not.$