Chapter 2 Exercises

Exercise 2.1 In "ϵ-greedy action selection, for the case of two actions and ϵ = 0.5, what is the probability that the greedy action is selected?

With a probability of 0.5, the greedy action will be chosen. With the probability of 0.5, there is still a 50% chance that the greedy action will be chose (since there are two actions). Thus the final probability of being greedy is 0.5 + (0.5)(0.5) = 0.75.

Exercise 2.2: Bandit example Consider a k-armed bandit problem with k = 4 actions, denoted 1, 2, 3, and 4. Consider applying to this problem a bandit algorithm using "ϵ-greedy action selection, sample-average action-value estimates, and initial estimates of Q1(a) = 0, for all a. Suppose the initial sequence of actions and rewards is A1 = 1, R1 =1, A2 = 2, R2 = 1, A3 = 2, R3 = 2, A4 = 2, R4 = 2, A5 = 3, R5 = 0. On some of these time steps the " case may have occurred, causing an action to be selected at random. On which time steps did this definitely occur? On which time steps could this possibly have occurred?

At t =1: Q(1) = 1, Q(2) = 0, Q(3) = 0, Q(4) = 0 // could be random
At t = 2: Q(1) = 1, Q(2) = 1, Q(3) = 0, Q(4) = 0 // random since only action 1 has positive value
At t = 3: Q(1) = 1, Q(2) = 1.5, Q(3) = 0, Q(4) = 0 // could be random
At t = 4: Q(1) = 1, Q(2) = 1.67, Q(3) = 0, Q(4) = 0 // could be random
At t = 5: Q(1) = 1, Q(2) = 1.67, Q(3) = 0, Q(4) = 0 // random since Q(3) = 0 before t = 5

Exercise 2.3 In the comparison shown in Figure 2.2, which method will perform best in the long run in terms of cumulative reward and probability of selecting the best action? How much better will it be? Express your answer quantitatively.

ϵ-greedy will perform the best, and it appears to be the smaller epsilon (0.01) will, in the long run, determine the best action in the long run and will continue to have better cumulative reward since, once it knows the optimal values, will select the best action with a higher probability than the algorithm with the higher epsilon. If epsilon is 0.1, that means that, in the long run, the algorithm will select the optimal action with 90% probability, while with 0.01 epsilon, it will select the optimal action with 99% probability.

Exercise 2.4 If the step-size parameters, α_n, are not constant, then the estimate Qn is a weighted average of previously received rewards with a weighting different from that given by (2.6). What is the weighting on each prior reward for the general case, analogous to (2.6), in terms of the sequence of step-size parameters?

$Q_{n+1} = Q_n + \alpha_n(R_n - Q_n)$. We can rewrite this formula as $Q_{n+1} = (1-\alpha_n)Q_n + \alpha_n R_n$.
Expanding this recursive formula, we get an equation that looks like this:
$Q_{n+1} = (1-\alpha_n)[(1-\alpha_{n-1})Q_{n-1} + \alpha_{n-1}R_{n-1}] + \alpha_n R_n = (1-\alpha_n)(1-\alpha_{n-1})Q_{n-1} + (1-\alpha_n)\alpha_{n-1}R_{n-1}$
and so on and so forth which can be rewritten as
$Q_{n+1} = \Sigma_{i=1}^{n}[\alpha_i \Pi_{j=i+1}^{n}(1-\alpha_j)]R_i + [\Pi_{i=1}^{n}(1-\alpha_i)Q_1$ which gives the generalized weight for each R_i.

Exercise 2.5 (programming) Design and conduct an experiment to demonstrate the difficulties that sample-average methods have for nonstationary problems. Use a modified version of the 10-armed testbed in which all the q⇤(a) start out equal and then take independent random walks (say by adding a normally distributed increment with mean 0 and standard deviation 0.01 to all the q⇤(a) on each step). Prepare plots like Figure 2.2 for an action-value method using sample averages, incrementally computed, and another action-value method using a constant step-size parameter, ↵ = 0.1. Use " = 0.1 and longer runs, say of 10,000 steps.

Exercise 2.6: Mysterious Spikes The results shown in Figure 2.3 should be quite reliable because they are averages over 2000 individual, randomly chosen 10-armed bandit tasks. Why, then, are there oscillations and spikes in the early part of the curve for the optimistic method? In other words, what might make this method perform particularly better or worse, on average, on particular early steps?

The initial timesteps are essentially random, since every time an arm is tried, the value of that state decreases, so even though $\epsilon = 0$, the policy will try each arm at least once, causing this random oscillations and spikes in the early part of the curve. As time goes on, the initial value of the state will be discounted and overshadowed by experience, and it will converge to the optimal values for each of the arms, and the greedy actions will allow it to get the best average return.

Exercise 2.7: Unbiased Constant-Step-Size Trick

This exercise was solved by hand. If you'd like an explanation, please reach out!

Exercise 2.8: UCB Spikes In Figure 2.4 the UCB algorithm shows a distinct spike in performance on the 11th step. Why is this? Note that for your answer to be fully satisfactory it must explain both why the reward increases on the 11th step and why it decreases on the subsequent steps. Hint: If c = 1, then the spike is less prominent.

For the first 10 steps, UCB samples each of the arms once. On the 11th step, each arm has been chosen exactly once, so the term after $Q_t(a)$ is the same across all arms and thus the one that gave the largest reward is again chosen. This is better than the randomly chosen arms, so there is a spike upwards. It decreases on the subsequent steps since, as UCB chooses the optimal arm, the other arms will increase in vaue since the term under the sqrt will quickly decrease for the optimal arm while it increases for non-optimal arms. The c=2 magnifies this scaling and the decrease becomes even larger since it's quicker to select the suboptimal arms as their values quickly become larger than the optimal.

Exercise 2.9 Show that in the case of two actions, the soft-max distribution is the same as that given by the logistic, or sigmoid, function often used in statistics and artificial neural networks.

This exercise was solved by hand. If you'd like an explanation, please reach out!

Exercise 2.10 Suppose you face a 2-armed bandit task whose true action values change randomly from time step to time step. Specifically, suppose that, for any time step, the true values of actions 1 and 2 are respectively 10 and 20 with probability 0.5 (case A), and 90 and 80 with probability 0.5 (case B). If you are not able to tell which case you face at any step, what is the best expected reward you can achieve and how should you behave to achieve it? Now suppose that on each step you are told whether you are facing case A or case B (although you still don’t know the true action values). This is an associative search task. What is the best expected reward you can achieve in this task, and how should you behave to achieve it?

The expected reward in the first case is $[(1)] = 0\ \ 10 + 0\ \ 0 = 0$ and $[()] = 0\ \ 0 + 0\ \ 0 = 0$ so the best expected reward is 50 and a policy that picks any action can have the expected reward of 50. In the associative case, the best expected reward is picking action 2 in case A and action 1 in case B which results in $[(1)] = 0\ \ 0 + 0\ \ 0 =$.

Exercise 2.11 (programming) Make a figure analogous to Figure 2.6 for the nonstationary case outlined in Exercise 2.5. Include the constant-step-size "-greedy algorithm with α = 0.1. Use runs of 200,000 steps and, as a performance measure for each algorithm and parameter setting, use the average reward over the last 100,000 steps.