2 Multi-armed Bandits

Reinforcement learning involves evaluative feedback rather than instructive feedback. We get told whether our actions are good ones or not, rather than what the single best action to take is. This is a key distinction between reinforcement learning and supervised learning.

2.1 A k-armed Bandit Problem

In the k-armed bandit problem there are k possible actions, each of which yields a numerical reward drawn from a stationary probability distribution for that action. We want to maximise the expected total reward, taking an action at each *time step*. Some notation:

- Index timesteps by t
- Action A_t
- Corresponding reward R_t
- *Value* of action a is $q_*(a) = \mathbb{E}[R_t|A_t = a]$
- Estimate of value of action a at t is denoted $Q_t(a)$

We therefore want to choose $\{a_1, \ldots, a_T\}$ to maximise $\sum_{t=1}^{T} q_*(a_t)$.

At each timestep, the actions with the highest estimated reward are called the *greedy* actions. If we take this action, we say that we are *exploiting* our understanding of the values of actions. The other actions are known as *non-greedy* actions, sometimes we might want to take one of these to improve our estimate of their value. This is called *exploration*. The balance between exploration and exploitation is a key concept in reinforcement learning.

2.2 Action-value Methods

We may like to form estimates of the values of possible actions and then choose actions according to these estimates. Methods such as this are known as *action-value methods*. There are, of course, many ways of generating the estimates $Q_t(a)$.

An ε-greedy method is one in which with probability ε we take a random draw from all of the actions (choosing each action with equal probability), providing some exploration.

2.5 Tracking a Non-stationary Problem

If we decide to implement the sample average method, then at each iteration that we choose the given action we update our estimate by

$$Q_{n+1} = Q_n + \frac{1}{n}[R_n - Q_n] \tag{1}$$

Note that this has the (soon to be familiar) form

$$\text{NewEstimate} \leftarrow \text{OldEstimate} + \text{StepSize} \times [\text{Target} - \text{OldEstimate}]. \tag{2}$$

If the problem was non-stationary, we might like to use a time weighted exponential average for our estimates (*exponential recency-weighted average*). This corresponds to a constant step-size $\alpha \in (0, 1]$ (you can check).

$$Q_{n+1} = Q_n + \alpha[R_n - Q_n]. \tag{3}$$

We might like to vary the step-size parameter. Write $\alpha_n(a)$ for the step-size after the n^{th} reward from action a. Of course, not all choices of $\alpha_n(a)$ will give convergent estimates of the values of a. To converge with probability 1 we must have

$$\sum_n \alpha_n(a) = \infty \qquad \text{and} \qquad \sum_n \alpha_n(a)^2 < \infty. \tag{4}$$

Meaning that the coefficients must be large enough to recover from initial fluctuations, but not so large that they don't converge in the long run. Although these conditions are used in theoretical work, they are seldom used in empirical work or applications. (Most reinforcement learning problems have non-stationary rewards, in which case convergence is undesirable.)

2.6 Optimistic Initial Values

The exponential recency weighted method is biased by the initial value one gives. If we like, we may set initial value estimates artificially high to encourage exploration in the short run – this is called *optimistic initial values*. This is a useful trick for stationary problems, but does not apply so well to non-stationary problems as the added exploration is only temporary.

2.7 Upper-Confidence Bound Action Selection

We might like to discriminate between potential explorative actions. Note that ε-greedy does not do this. We define the *upper-confidence bound* action at t as follows

$$A_t \doteq \operatorname*{argmax}_a \left[Q_t(a) + c\sqrt{\frac{\ln(t)}{N_t(a)}} \right] \tag{5}$$

where $Q_t(a)$ is the value estimate for the action a at time t, $c > 0$ is a parameter that controls the degree of exploration and $N_t(a)$ is the number of times that a has been selected by time t. If $N_t(a) = 0$ then we consider a a maximal action.

This approach favours actions with a higher estimated rewards but also favours actions with uncertain estimates (more precisely, actions that have been chosen few times).

2.8 Gradient Bandit Algorithms

Suppose that we choose actions probabilistically based on a preference for each action, $H_t(a)$. Let the action at t be denoted by A_t. We then define the probability of choosing action a via the softmax

$$\pi_t(a) \doteq \mathbb{P}(A_t = a) = \frac{e^{H_t(a)}}{\sum_i e^{H_t(i)}}. \tag{6}$$

We then iteratively perform updates according to

$$H_{t+1}(a) = H_t(a) + (R_t - \bar{R}_t)(\mathbb{1}_{A_t=a} - \pi_t(a)), \tag{7}$$

where $\bar{R}_t$ is the mean of previous rewards. The box in the notes shows that this is an instance of stochastic gradient ascent since the expected value of the update is equal to the update when doing gradient ascent on the (total) expected reward.