Part 1 - Multi-Armed Bandits

Runar, Lars and Årne

1. How would this modified problem fit in the MAB or in the MDP formalism? What challenges arise in the current setting?

   This modified problem would fit into the MAB formalism as there still exist only one state, and we are trying to infer the expectation value of different actions. However, here the expectation value of the different medicines will change with time! Therefore, we are trying to compute a "moving target". This is a challenge, as an assumption of the MAB problem is that their expected value stays constant. Formalising using the MDP formalism we would get $\langle \mathcal{S} = \mathbb{I}, \mathcal{A} = \{A, B, C\}, \mathcal{R} = \{\mathcal{R}_A(t), \mathcal{R}_B(t), \mathcal{R}_C(t)\}, \mathcal{T} = \mathbb{I}, \gamma \rangle$.

2. Assume your aim is still to minimize regret or maximize reward. How would you change or adapt a MAB algorithm to deal with this scenario in which you do not know when changes in effectiveness happen? Explain and justify your algorithmic choices. Describe assumptions and/or dependence of your solution on the parameters of the problem.

   If we assume that the medicines stay independent and uniformly distributed, and that only the parameters of each distribution (medicine) changes over time, we could keep track of every observed reward and compute a running average over a window of observations. Here we are also assuming that changes in the variance of the distribution don't have that big of an impact in sampled expectation value.

   By keeping a window (therefore forgetting observed values over time), the computed expectation value will stay more accurate given changes over time. However, we still need to figure out how the window should be stored/when we should forget earlier values.

   The simplest solution would be to store a sliding window containing a fixed size of samples, where the mean of the window is the expected value. Here, the challenge would be to balance sliding window size such that it would solve the problem well. If the size of the sliding window is big, we would ensure that our sampled running average is accurate towards the expectation value, however a bigger size would also lose accuracy when it comes to expectation value changing over time.

3. Run the algorithmic solution proposed above alongside (at least) one of the standard MAB algorithm discussed in the course. Compare and comment on the result.

   We run the algorithm proposed above as a UCB, with different hyperparameters $\tau \in \{10, 25, 50, 100, 250\}$ for sliding window size, and compare it with the standard UCB (hypothetically $\tau = \infty$). The results are gathered from running 2500 episodes and averaging reward over 25 experiments.

   From the plot (figure 1) we can see that the sliding window UCB algorithm significantly outperforms the standard version for every sliding window size τ. The best performing model is the sliding window UCB with either $\tau = 50$ or $\tau = 100$ (their performance is very similar), and the worst performing model is the standard UCB.

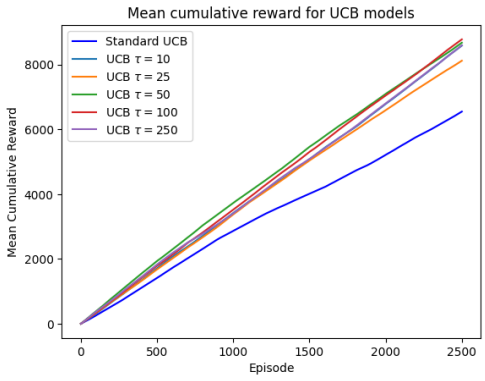

Figure 1: Comparison of different UCB on non-stationary problem