CS 138: Homework 1 – Multi-Armed Bandits

Lawrence Qiu

September 25, 2025

1 Introduction

Chapter 2 of the textbook introduces the multi-armed bandit problem, in which an agent is tasked with maximizing the reward over a certain number of steps by choosing one of several possible actions. Each action has an intrinsic value $q_\star(a)$ which is sampled from a normal distribution at the beginning of the experiment. On each step, the agent's reward is sampled from a normal distribution centered at the intrinsic value of the action chosen by the agent. It is necessary therefore for the agent to explore by picking each action to estimate the intrinsic value of it, before exploiting by picking the highest value one.

One strategy to implement the multi-armed bandit is the ϵ-greedy strategy, in which on each step the agent has an ϵ probability of picking a random action and a $1 - \epsilon$ probability of choosing the maximum valued one based on its estimates. There are multiple ways to calculate this estimate. One method is to calculate the mean of past rewards from that action, which has a simple incremental implementation:

$$Q_{n+1} = Q_n + \tfrac{1}{n}(R_n - Q_n),$$

where Q_n is the value estimate at the nth sample of the action. Another method is to calculate the exponential moving average (EMA) of the action rewards, favoring recent samples over older ones. This has the simple incremental implementation:

$$Q_{n+1} = Q_n + \alpha(R_n - Q_n).$$

2 Part 1: Implementing the Exercise

Exercise 2.5 of the textbook asks us to implement the multi-armed bandit problem for a nonstationary version of the multi-armed bandit problem, in which the rewards change over time. Specifically, instead of initializing intrinsic values $q_\star(a) \sim \mathcal{N}(0, 1)$ at the beginning, the values are initialized to zero and vary on each time step according to a normal distribution:

$$q_\star(a) \leftarrow q_\star(a) + \Delta q, \quad \Delta q \sim \mathcal{N}(0, 0.01).$$

I first implemented the basic stationary version of the problem in Python. For the following experiments, I used # arms = 10, # steps = 10,000, and # experiments = 2000. The original $q_\star(a)$ values are sampled from $\mathcal{N}(0, 1)$ and the value estimates are calculated using the mean. Figure 1 shows the results of the simulation, which matches the results shown in the textbook.

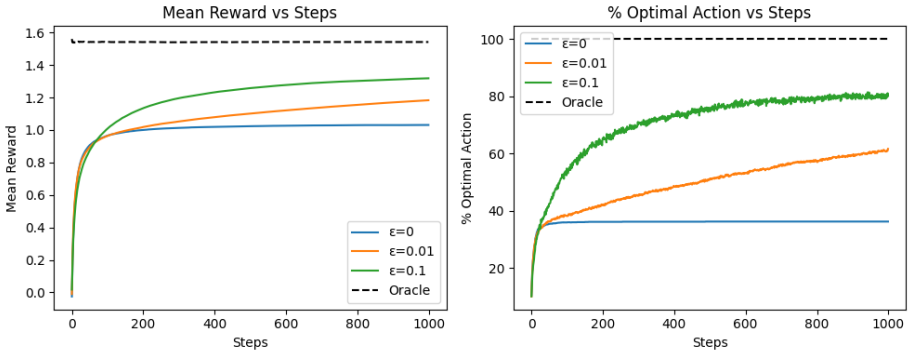

Figure 1: Mean reward and % optimal action vs. step of the reimplementation of the stationary multi-armed bandit problem. ϵ values of 0, 0.1, and 0.01 are shown, with 0.1 performing the best, 0.01 lagging behind, and 0 performing the worst, ranging from final mean reward of 1.4 to 1.0. Additionally, the optimal policy is plotted for comparison. These results match the textbook.

Next, I implemented the nonstationary version of the problem, in which $q_\star(a)$ values are initialized to 0 and sampled from $\mathcal{N}(0, 0.01)$. Here, ϵ is fixed to 0.1 and both the mean value estimation strategy and the EMA value estimation with α values of 0.03, 0.1, and 0.3 are tested.

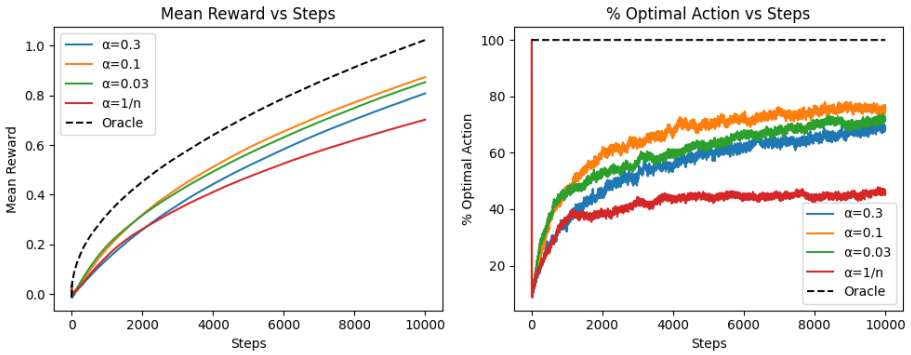

Figure 2: Mean reward and % optimal action vs. step for nonstationary multi-armed bandit problem. EMA value estimation with $\alpha = 0.1$ performs the best, with $\alpha = 0.03$ and $\alpha = 0.3$ lagging behind. Mean value estimation performs the worst, with % optimal action asymptotically limited at 40%.

Figure 2 shows a comparison of these strategies. All of the EMA value estimation methods outperform the mean value estimation method, which is expected as the mean method gives equal weights to all past samples, whereas only recent samples are likely to be relevant as the action values drift. The asymptotic % optimal action performance indicates that the method is unable to adapt to changing optimal actions.

The best EMA strategy is $\alpha = 0.1$, with both $\alpha = 0.3$ and $\alpha = 0.03$ performing worse. This is expected—too high α leads to too few value estimates and high variance whereas too low α puts too much weight on old samples. I performed a full grid hyperparameter search for $10^{-3} < \alpha < 10^{0}$ and $10^{-4} < \epsilon < 10^{0}$, which is shown in Figure 3.

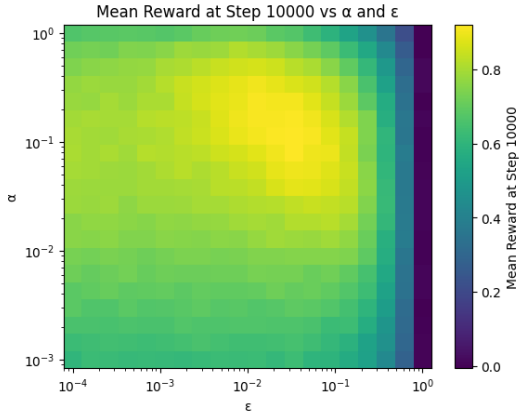

Figure 3: Final mean reward over ϵ and α. Good performance occurs over multiple magnitudes of both hyperparameters, indicating that the strategy is robust. The highest reward occurs at $\alpha = 0.11$ and $\epsilon = 0.034$, with mean reward of 0.92 compared to the optimal policy value of 1.02.

3 Part 2: Comparison with Upper-Confidence-Bound Selection

Another action selection strategy is upper-confidence-bound (UCB) selection. In this method, the action with the highest value is always selected, but this value is biased with an uncertainty factor based on both the current time and the number of samples for the particular action:

$$A_t = \arg\max_a \left[Q_t(a) + c \cdot \sqrt{\frac{\ln t}{N_t(a)}} \right].$$

The coefficient c dictates the weight of the factor—high c promotes more exploration whereas low c reduces to the greedy strategy. Figure 4 shows the performance of this strategy with different values of c, using the mean action value estimation.

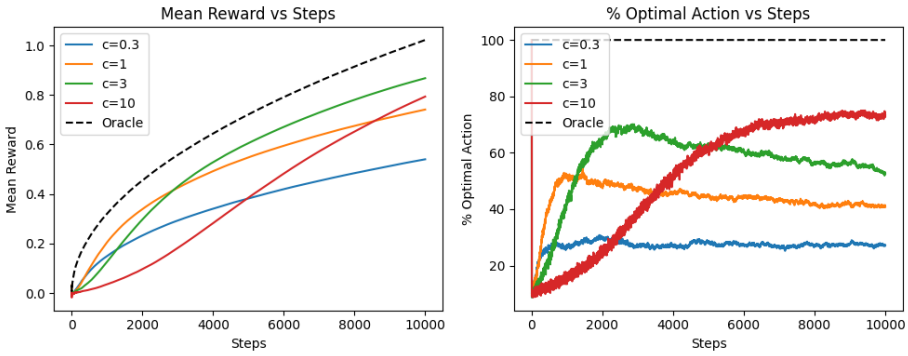

Figure 4: Mean reward and % optimal action vs. step for UCB selection in the nonstationary multi-armed bandit problem. From best to worst: $c = 3$, $c = 10$, $c = 1$, $c = 0.3$. The $c = 10$ case is an outlier as it underperforms early but catches up later.

Interestingly, all but the $c = 10$ run show % optimal action peaking early in the run and then decreasing. This is likely because, unlike in the ϵ-greedy strategy, exploration steps are not evenly spaced throughout the run, but rather concentrated in the beginning (since the numerator of the exploration term scales with $\ln t$ rather than linearly with t). As the optimal action drifts later, the agent is unable to adapt, so the % optimal action drops.

These tests were done with a mean action value estimation strategy. I also performed the experiment with c fixed to 3 and varying α. Figure 5 shows that EMA value estimation is able to partially compensate for the uneven exploration, with higher α reducing the dip in action accuracy.

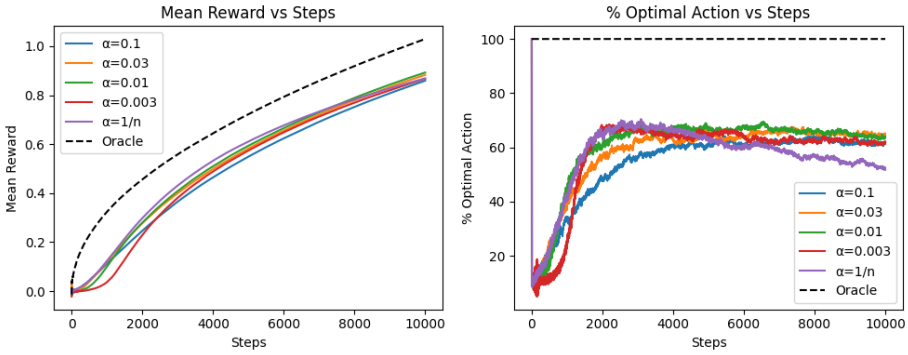

Figure 5: Mean reward and % optimal action vs. step for UCB selection in nonstationary bandit problem. α values of 0.003, 0.01, 0.03, and 0.1 are shown. $\alpha = 0.01$ performs the best and 0.1 the worst, though final mean rewards are all ≈ 0.8. Larger α reduces the dip in % optimal action.

Lastly, I also performed a full grid hyperparameter search for $10^{-4} < \alpha < 10^{0}$ and $10^{-3} < c < 10^{1}$, shown in Figure 6.

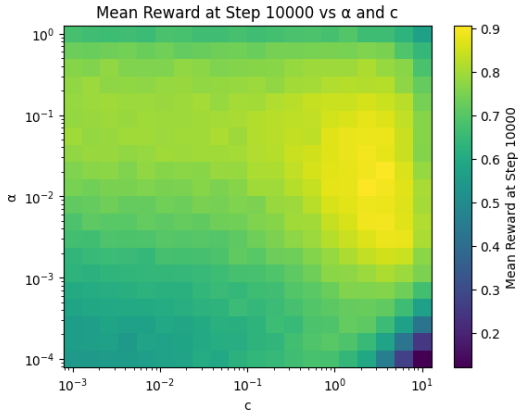

Figure 6: Final mean reward over α and c. Good performance occurs over multiple magnitudes of both hyperparameters. The highest reward occurs at $\alpha = 0.013$ and $c = 0.23$, with mean reward of 0.91 compared to the optimal policy value of 1.02.

4 Summary and Conclusion

I performed several simulations of the nonstationary multi-armed bandit problem, in which the action values vary throughout the experiment. Several strategies were tested, including ϵ-greedy with mean value estimation, ϵ-greedy with EMA value estimation, UCB selection with mean value estimation, and UCB selection with EMA value estimation. In general, EMA value estimation outperforms mean value estimation as it weighs recent observations over past ones that may be out of date. The optimal hyperparameters for both ϵ-greedy and UCB were found, and both perform close to the maximum, although ϵ-greedy was slightly closer. This is likely because ϵ-greedy distributes exploration steps uniformly throughout the experiment while UCB concentrates them toward the beginning.