COMP 138 RL: Assignment 1

Cullen McCaleb

February 14, 2025

1 Goal

This experiment attempts to demonstrate the difficulties that sample-average methods face in nonstationary problems, which will be done by comparing sample-average methods with methods that use constant step-size parameters on a ten-armed bandit problem.

2 Background

2.1 Sample-Average Methods

In reinforcement learning, action value methods are used to estimate the values of actions, and these estimations are used to optimize action selection in order to maximize a reward. The sample-average method is a type of action value method that simply averages the actual reward received when a given action was previously chosen. If $Q_t(a)$ represents the estimated value on the t time step, $N_t(a)$ represents the number of times an action a has been previously chosen, which have yielded reward $R_1, R_2, ..., R_{N_t(a)}$, then the sample-average method can be represented by

$$Q_t(a) = \frac{R_1, R_2, ..., R_{N_t(a)}}{N_t(a)}$$

2.2 Constant Step-Size Methods

While sample-averages are extremely intuitive, these methods face a big problem: the memory and computational requirements only increase over time. A solution to this problem is to use incremental update formulas that efficiently update averages with each new reward. This formula would let Q_k represent the estimate for a certain action's kth reward, R_k represent the actual reward, and look like this:

$$Q_{k+1} = Q_k + \frac{1}{k}[R_k - Q_k]$$

2.3 Greedy Action Selection

Once the value of each action is determined, the agent must have a strategy to select an action. In this experiment, greedy and ϵ-greedy action selection will be used. In a fully greedy strategy, the agent chooses the action with the highest expected reward. In an ϵ-greedy strategy, the agent picks the action with the highest expected reward most of the time, but picks a random action with the small probability of ϵ.

2.4 Nonstationary Problems

The sample-average method is an intuitive example of an action value method which works well in stationary problems, but most problems in the real world are nonstationary, meaning the underlying dynamics in a problem's environment change over time. In this experiment, a nonstationary environment will be simulated by changing the true reward values for each action with a random walk (sigma walk), or standard deviation of 0.01.

2.5 Average Reward and Optimal Action

To measure each strategy's effectiveness, this experiment will keep track of both the average reward after each method selects an action, and the percentage of when this action selection is the optimal action (action with the highest true value).

3 Experimental Design

This experiment will use the a python program to simulate a nonstationary 10-armed bandit problem with a random walk of $\sigma = 0.01$. All the true action values will be initialized at 0. To solve the bandit problem, both sample-average methods and constant step-size methods will be used for value estimation. The step-size parameter will be $\alpha = 0.1$ throughout. Only greedy action selections will be used: fully greedy, 0.1-greedy, and 0.01-greedy. Each combination of action-value and action-selection methods will be run on the bandit problem for a total of 3,000 runs, with 10,000 time steps during each run, and the results for each run will be averaged to plot average reward and % optimal action versus the number of time steps. The sample-average method will be compared with the constant step-size method using these plots.

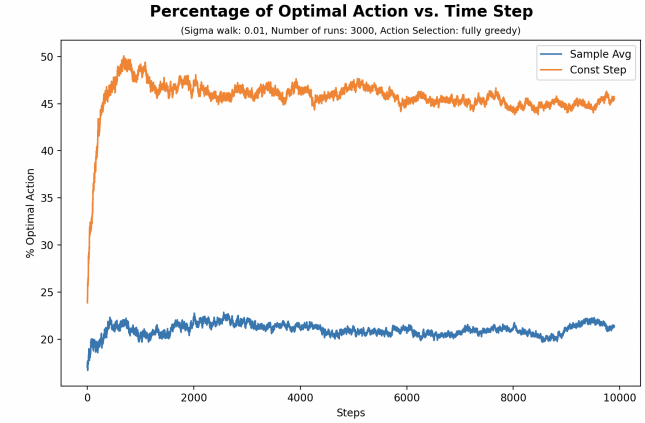
4 Additional Question

To better understand the exploration vs. exploitation dynamics in this experiment, an additional question I hope to answer is: how often does the agent switch actions while using each method? This will ideally help me compare the effectiveness of both methods, and allow for a more detailed explanation for why thy behave differently, if they do. I will answer this question by creating a % switch action versus number of time steps graph, using the python program mentioned above.

5 Results

Figure 1: Average reward vs. time for fully greedy action selection

Figure 2: % optimal action for fully greedy action selection



Figures 1 and 2 compare the sample-average method, shown in blue, with the constant step-size method, shown in orange, while using a fully greedy action selection method. In figure 1, the step-size method clearly outperforms sample-average, as the orange line has a steeper slope, peaking at an average reward of about 1.2 after 10,000 time steps. In figure 2, the step-size method also performs better, operating at a 45-50% optimal action, while the sample-average method stays only at around 20%. So clearly, with a fully greedy action selection, the step-size method performs much better.

Figure 3: Average reward vs. time for ϵ-greedy action selection, $\epsilon = 0.01$

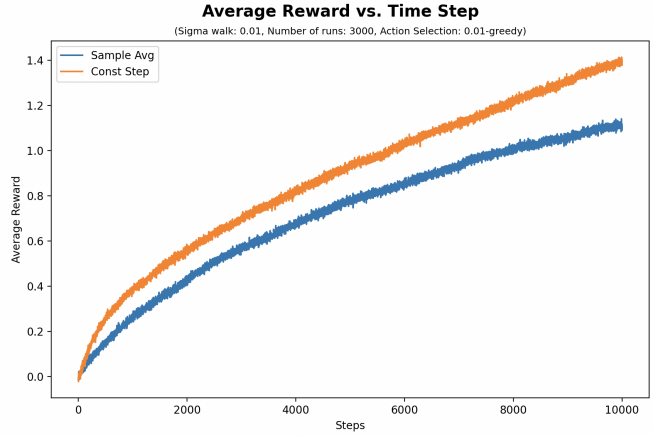
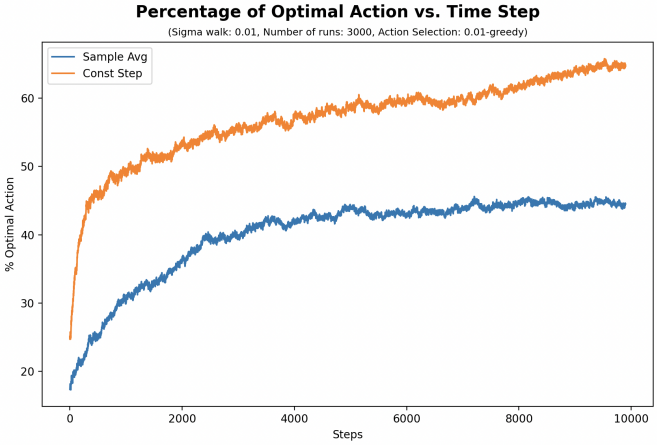


Figure 4: % Optimal action for ϵ-greedy action selection, $\epsilon = 0.01$



Figures 3 and 4 compare the sample-average method, shown in blue, with the constant step-size method, shown in orange, this time using a ϵ-greedy action selection method with $\epsilon = 0.01$. In figure 3, the step-size method outperforms sample-average yet again, but not by as large a margin as with the fully greedy action selection. The step-size method peaks at an average reward of about 1.4 after 10,000 time steps, while the sample-average method only reaches about 1.0. Notably, the gap increases as more steps are reached. In figure 4, the step-size method performs much better, operating at a very high 70% optimal action, while the sample-average method stays only at around 40%. So with a 0.01-greedy action selection, the step-size method performs much better again.

Figure 5: Average reward vs. time for ϵ-greedy action selection, $\epsilon = 0.1$

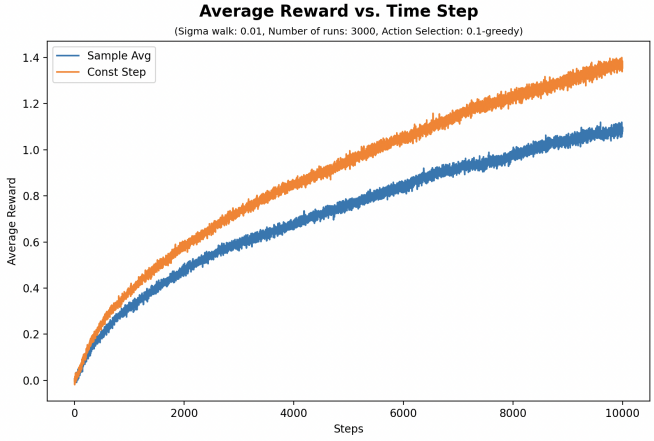
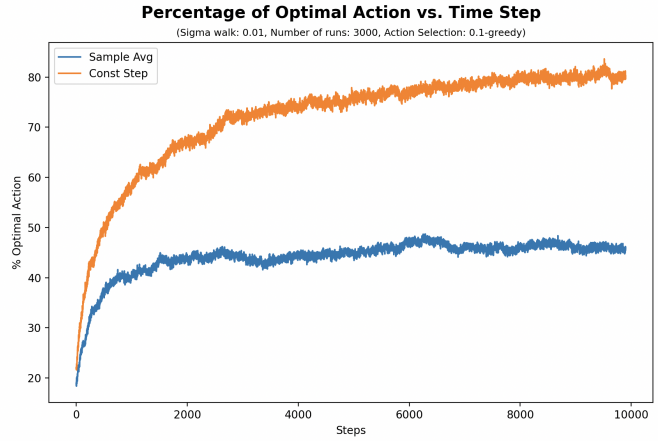


Figure 6: % Optimal action for ϵ-greedy action selection, $\epsilon = 0.1$



Figures 5 and 6 compare the sample-average method, shown in blue, with the constant step-size method, shown in orange, this time using a more exploratory ϵ-greedy action selection method with a $\epsilon = 0.1$. Figure 5 shows the step-size method outperforming sample-average once more, but by even less of a margin that the 0.01-greedy action selection portrayed. In figure 5, the step-size method peaks at an average reward of about 1.4 after 10,000 time steps again, while the sample-average method only reaches about 1.1. The gap still increases as more steps are reached, but not by as much as figures 1 and 3 did. In figure 6, the step-size method still performs much better, operating at about 70% optimal action, while the sample-average method stays only at around 45%. This is less of a gap than in figures 2 and 4, though. So with a 0.01-greedy action selection, the step-size method performs much better again.

Figure 7: % Action switch vs. time for fully greedy action selection

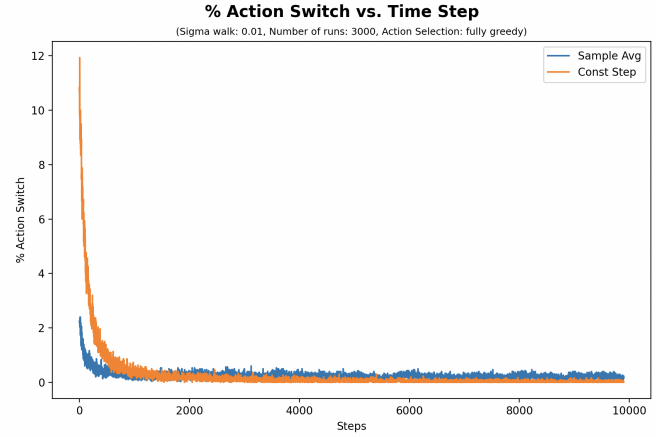


Figure 8: % Action switch for ϵ-greedy action selection, $\epsilon = 0.01$

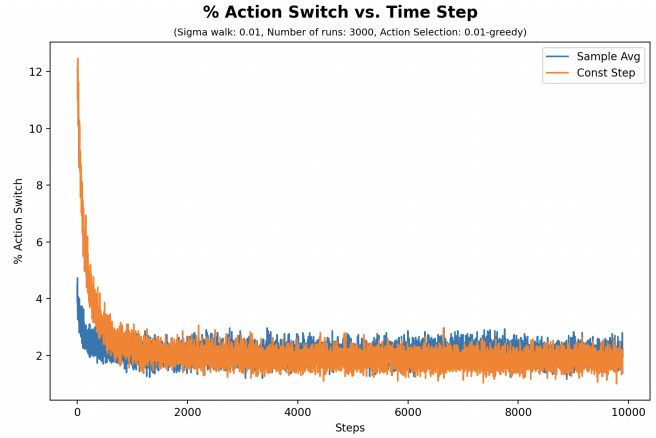


Figure 9: % Action switch vs. time for ϵ-greedy action selection, $\epsilon = 0.1$

Figures 7, 8, and 9 show the percent of actions that are not the same as the previous action - or when an action switches. This means either the agent has chosen to explore according to its epsilon value, or the agent has decided that there is a different action with the highest expected reward. Interestingly, the plots above show practically no difference between sample-average and step-size methods for this metric. Additionally, the plots show an increase in variability, or noise, as epsilon increases.

6 Analysis

6.1 Overall Analysis

These six figures above show us that the constant step-size method is better suited for a non-stationary environment than the sample-average method, as it performs better in terms of both average reward and percent optimal action, regardless of the epsilon chosen for action selection.

6.2 Average Reward Plots

The most notable finding was that the gap between each method for the average reward plots decreased as the epsilon value increased. In other words, the margin of victory for the constant step-size method was not as much with more exploration. This means that the sample-average method easily gets stuck at a sub-optimal action, and receives no help from a positive non-zero epsilon value to force exploration. This makes sense because sample-average methods weigh all past rewards equally so it has trouble adapting, while constant step-size methods will adapt much quicker with fully greedy action selection. Once ϵ-greedy action selection is implemented, sample-average methods receive help exploring, and the gap between each method should decrease.

6.3 Optimal Action Plots

On the other hand, the gap between the methods in the % optimal action plot actually increases. This makes sense because with a higher exploration rate, the agent using sample-average will choose actions less optimally. For the constant step-size method, a higher exploration means the agent can better detect when the optimal action changes, so the percent optimal action will therefore be higher.

6.4 Action Switch Plots

It may have been expected that constant step-size methods would result in more action switches, but surprisingly this was not the case, as both methods exhibited similar action switch rates. This suggests that both methods switch actions a similar number of times. Yet as shown above the constant step-size method still achieves a much higher percentage of optimal actions. Together, these findings indicate that the constant step-size method learns better — not by switching more frequently, but by adapting more effectively to changes in the optimal action. Conversely, the sample-average method struggles to adapt to changes.

7 Conclusion

This experiment compared sample-average and constant step-size methods in a nonstationary 10-armed bandit environment to evaluate their effectiveness during nonstatinoary problems. The results consistently showed that the constant step-size method outperformed the sample-average method, achieving higher average rewards and a greater percentage of optimal actions across all tested epsilon values.

Notably, the gap in average reward between methods shrank as epsilon increased, and the gap in % optimal action widened, meaning that higher exploration helps the step-size method recognize optimal actions more effectively while causing more instability for the sample-average method. Additionally, both methods exhibited similar action switch rates, suggesting that learning quality, not just action-switch frequency, determines adaptability.

In conclusion, the findings in this experiment show that sample-average methods struggle in non-stationary environments because they assign equal weight to all past experiences, making them slow to adapt to the changing optimal actions.