4180/5180: Reinforcement Learning and Sequential Decision Making (Fall 2024) Christopher Amato
Northeastern University Due Sep 20, 2024

Exercise 1: Multi-armed Bandits

Please remember the following policies:

- Exercise due at **11:59 PM EST Sep 20, 2024**.
- Submissions should be made electronically on Canvas. Please ensure that your solutions for both the written and programming parts are present. You can upload multiple files in a single submission, or you can zip them into a single file. You can make as many submissions as you wish, but only the latest one will be considered.
- For **Written** questions, solutions may be handwritten or typeset. If you write your answers by hand and submit images/scans of them, please please ensure legibility and order them correctly in a single PDF file.
- The PDF file should also include the figures and answers from the **Plot** questions.
- For both **Plot** and **Code** questions, submit your source code in Jupyter Notebook (.ipynb file) along with reasonable comments of your implementation. Please make sure the code runs correctly.
- You are welcome to discuss these problems with other students in the class, but you must understand and write up the solution and code yourself. Also, you *must* list the names of all those (if any) with whom you discussed your answers at the top of your PDF solutions page.
- Each exercise may be handed in up to two days late (24-hour period), penalized by 10% per day late. Submissions later than two days will not be accepted.
- Contact the teaching staff if there are medical or other extenuating circumstances that we should be aware of.
- **Notations: RL2e is short for the reinforcement learning book 2nd edition. x.x means the Exercise x.x in the book.**

1. **1 point.** (RL2e 2.2) *Exploration vs. exploitation.*
   Written: 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 $Q_1(a) = 0$, for all a. Suppose the initial sequence of actions and rewards is $A_1 = 1, R_1 = -1, A_2 = 2, R_2 = 1, A_3 = 2, R_3 = -2, A_4 = 2, R_4 = 2, A_5 = 3, R_5 = 0$. On some of these time steps the ε 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?

2. **1 point.** (RL2e 2.4) *Varying step-size weights.*
   Written: If the step-size parameters, α_n, are not constant, then the estimate Q_n is a weighted average of previously received rewards with a weighting different from that given by Equation 2.6. What is the weighting on each prior reward for the general case, analogous to Equation 2.6, in terms of the sequence of step-size parameters?

3. **2 points.** *Bias in Q-value estimates.*
   This question is required for 5180 and extra credit for 4180

   Written: Recall that $Q_n \triangleq \frac{R_1+\ldots+R_{n-1}}{n-1}$ is an estimate of the true expected reward q_* of an arbitrary arm a. We say that an estimate is *biased* if the expected value of the estimate does not match the true value, i.e., $\mathbb{E}[Q_n] \neq q_*$ (otherwise, it is *unbiased*).

   (a) Consider the *sample-average* estimate in Equation 2.1. Is it biased or unbiased? Explain briefly.

   For the remainder of the question, consider the *exponential recency-weighted average* estimate in Equation 2.5. Assume that $0 < \alpha < 1$ (i.e., it is strictly less than 1).

   (b) If $Q_1 = 0$, is Q_n for $n > 1$ biased? Explain briefly.

(c) Derive conditions for when Q_n will be unbiased (Q_1 can be non-zero).

(d) Show that Q_n is *asymptotically unbiased*, i.e., it is an unbiased estimator as $n \to \infty$.

(e) Why should we expect that the *exponential recency-weighted average* will be biased in general?

4. **1 point.** (RL2e 2.9) *Gradient Bandit*
Written: 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.

In the next few questions, you will be implementing the 10-armed testbed described in Section 2.3, reproducing some textbook figures based on this testbed, and performing further experimentation on bandit algorithms. Some general tips:

- Read all the questions first before implementing your experimental pipeline. If you design the pipeline well, implementing experiments and plots after Q5 will be require fairly small changes.
- You are encouraged to use good software engineering practices (related to previous point), but it is not required as long as your code is readable.
- Some of the experiments will take a while. During development, you should use smaller numbers of steps and trials to make iteration much faster. Here are some possible settings:
  - Tiny: 100 steps, 20 trials – for fast iteration, should take < 1 second
  - Small: 100 steps, 200 trials – for development and debugging purposes, should take < 10 seconds
  - Medium: 1000 steps, 2000 trials – the setting used in the textbook, useful for verifying correctness of implementation and seeing initial trends, should take < 5 minutes
  - Large: 10^4 steps, 2000 trials – the final setting in this assignment, should take ~ 1 hour, consider running these overnight or while doing something else
- The above times were based on our implementation running on a single CPU thread of a Lenovo ThinkPad T480s (circa 2018) laptop. You can even consider multiprocessing, but that should not be necessary. You may consider adjusting the settings above of steps/trials in order to achieve fast iteration.
- If you are having difficulty running the required number of steps/trials even after trying to ensure your implementation is efficient, you may use the "Medium" setting above (1000 steps, 2000 trials), and indicate that you did this in your submission.

5. **2 points.** *Reproducing Figure 2.2. (RL2e page 29)*
Code: Implement the ε-greedy algorithm with incremental updates. Note that in the graph: "All the methods formed their action-value estimates using the **sample-average technique (with an initial estimate of** 0**)**. See equation 2.1"
Plot: Reproduce both plots shown in Figure 2.2, with the following modifications:
   - We provide you with the scaffolding code to run experiments and to plot figures.
   - Use 10^3 steps with 2000 independent runs (same as in text) for the final report. Make sure that all methods are evaluated on the same set of 2000 10-armed bandit problems. To test your implementation, you can start with small runs (e.g., 100) and steps (e.g., 100).
   - In each subplot, you have to show three curves corresponding to $\epsilon = 0, 0.01, 0.1$, respectively.
   - We provide the code for plotting. In particular, we add an extra constant upper bound line corresponding to the best possible *average* performance in the running trials, based on the known true expected rewards $q_*(a)$. That is, the line should correspond to $\max_a q_*(a)$, averaged over all trials. For each curve, we also plot confidence bands corresponding to ($1.96\times$ standard error) of the rewards. The standard error of the mean is defined as the standard deviation divided by $\sqrt{n}$: $\frac{\sigma}{\sqrt{n}}$. This corresponds to a $\sim 95\%$ confidence interval around the average performance. In other words, our uncertainty in the average performance decreases as the number of trials increases. See the following for an example of plotting a confidence band in `matplotlib.pyplot`: `https://matplotlib.org/3.3.1/gallery/lines_bars_and_markers/fill_between_demo.html#example-confidence-bands`
   - Specifically, the plotting function is named *plot_curves* in Jupyter Notebook. Please read the function carefully and make sure the input arguments are valid.

- For the `Average reward` figure, the x axis is the steps and the y axis is the averaged reward for each step over all runs. For example, at step t, we run N trails and receives N rewards $[r_1, r_2, ..., r_N]$. Then, the average reward at step t is computed as $\frac{\sum_{i=1}^{N} r_i}{N}$.
- For the `Optimal action` figure, the x axis is the steps and the y axis is the percentage of selecting the optimal action at each step. For example, at step t, we run N trails and select N actions $[A_1, A_2, ..., A_N]$. We can compute the percentage of the optimal action at step t as $\frac{\sum_{i=1}^{N} \mathbb{1}_{A_i=a_i}}{N}$, where A_i is the selected action at step t for the i-th trail and a_i is the optimal action for the i-th trail.

Written: What are the average rewards that the algorithm converges to using different ϵ values? Why does the algorithm converge to different average rewards for different ϵ? Explain briefly.

6. **2 points.** *Reproducing and supplementing Figures 2.3 (RL2e page 34) and 2.4. (RL2e page 36)*
Code: Implement the ε-greedy algorithm with optimistic initial values, and the bandit algorithm with UCB action selection. **Plot:** Reproduce the plots shown in both Figures 2.3 and 2.4, with the following modifications:

   - We provide you with the scaffolding code to run experiments and to plot figures.
   - Use 10^3 steps with 2000 independent runs (same as in text) for the final report. Make sure that all methods are evaluated on the same set of 2000 10-armed bandit problems. To test your implementation, you can start with smaller runs (e.g., 100) and steps (e.g., 100).
   - Figure 2.3 should show % optimal action and Figure 2.4 should show the average reward. Both plots should each contain 5 curves using the following parameters:
     - ε-greedy $(Q_1 = 0, \varepsilon = 0)$
     - ε-greedy $(Q_1 = 5, \varepsilon = 0)$
     - ε-greedy $(Q_1 = 0, \varepsilon = 0.1)$
     - ε-greedy $(Q_1 = 5, \varepsilon = 0.1)$
     - UCB $(c = 2)$
   - Use the provided plotting function (*plot_curves*), plot the confidence bands corresponding to ($1.96\times$ standard error) of the rewards and show the constant upper bound line corresponding to the best possible *average* performance in your trials, based on the known true expected rewards $q_*(a)$. That is, the line should correspond to $\max_a q_*(a)$, averaged over all trials.

   Written: We have seen the spike for optimistic initialization in class (Figure 2.3, Figure 2.4 in RL2e). Observe that UCB also produce spikes in the very beginning in both the two reproduced figures. Explain in your own words why the spikes appear (both the sharp increase and sharp decrease). Analyze and use your experimental data as further empirical evidence to back your reasoning.

7. **[Extra credit.]** (RL2e 2.5) *Investigating nonstationary environments.*
Code: 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 to 0 and then take independent random walks (by adding a normally distributed increment with mean 0 and standard deviation 0.01 to all the $q_*(a)$ on each step). You can use the template provided or code yourself. **Plot:** Prepare plots like Figure 2.2 (with average reward upper bound and confidence bands as in Q5) for an action-value method using sample averages, incrementally computed, and another action-value method using a constant step-size parameter, $\alpha = 0.1$. Use $\varepsilon = 0.1$ and longer runs of 10^4 steps.