CMPUT 366 A1

Yiyang Wang

TOTAL POINTS

105 / 115

QUESTION 1

Q1 50 pts

1.1 Part 1 15 / 15

✓ **- 0 pts** Correct

1.2 Part 2 5 / 5

✓ **- 0 pts** Correct

1.3 Part 3 5 / 5

✓ **- 0 pts** Correct

1.4 Part 4 10 / 10

✓ **- 0 pts** Correct

1.5 Part 5 14 / 15

✓ **- 1 pts** Explanation on when step size is poor choice partly lacking

You gave situations where certain constant step sizes work well, but the sample average might also work well. In what situations will the sample average perform poorly?

QUESTION 2

2 Q2 10 / 10

✓ **- 0 pts** Correct

QUESTION 3

Q3 (Programming) 40 pts

3.1 Part 1 10 / 10

✓ **- 0 pts** Correct

3.2 Part 2 30 / 30

✓ **- 0 pts** Correct

Please use rlg.rl_env_message() to get information from the environment.

QUESTION 4

4 Bonus Q1 (Programming) 0 / 5

✓ **- 5 pts** No answer found or completely incorrect answer.

QUESTION 5

5 Bonus Q2 4 / 5

✓ **- 0 pts** Correct

- 1 Point adjustment

j is not from 1 to n.

QUESTION 6

6 Bonus Q3 2 / 5

✓ **- 1 pts** unclear analysis of why spikes occur

✓ **- 1 pts** did not note that action selection follows a round robin strategy due to constant step-size

✓ **- 1 pts** analysis of spikes in the context of particular example is not clear or complete

QUESTION 7

7 Late Penalty 0 / 0

✓ **- 0 pts** Correct

Last name: WANG First name: YI YANG SID#: 1503849

Collaborators:

CMPUT 366Assignment 1: Step sizes & Bandits

Due: Tuesday Sept 18 by gradescope

Policy: Can be discussed in groups (acknowledge collaborators) but must be written up individually

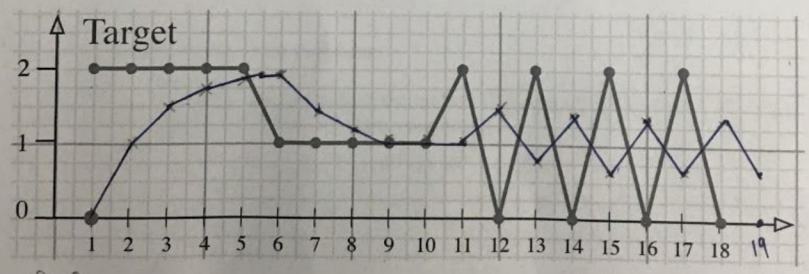
There are a total of 100 points on this assignment, plus 15 points available as extra credit!

Question 1 [50 points] Step-sizes. Plotting recency-weighted averages.

Equation 2.5 (from the SB textbook, 2nd edition) is a key update rule we will use throughout the course. This exercise will give you a better hands-on feel for how it works. This question has **five** parts.

Do all the plots in this question by hand. To make it easier for you, I'll include some graphing area and a start on the first plot here, so you should just be able to print these pages out and draw on them.

Part 1. [15 pts.]
Suppose the target is 2.0 for five steps, then 1 for five steps, and then alternates between 2.0 and 0 for 8 more steps, as shown by the grey line in the graph below. Suppose the initial estimate is 0, and that the step-size (in the equation) is 0.5. Your job is to apply Equation 2.5 iteratively to determine the estimates for time steps 1-19. Plot them on the graph below, using a blue pen, connecting the estimate points by a blue line. The first estimate Q_1 is already marked below:



$\alpha = 1/2$

$Q_1 = 0$

$\alpha = 0.5$

$Q_2 = Q_1 + 0.5[R_1 - Q_1] = 1$

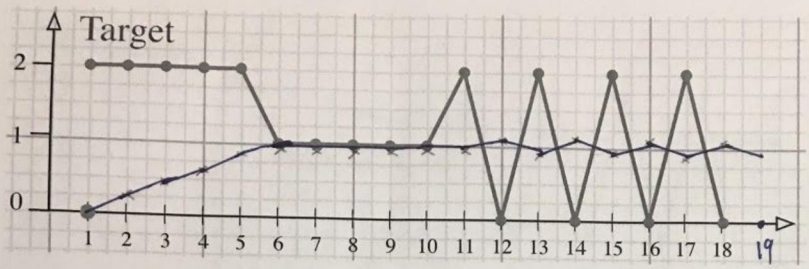
$Q_3 = Q_2 + 0.5[R_2 - Q_2] = 1.5$

$Q_4 = Q_3 + 0.5[R_3 - Q_3] = 1.75$

$Q_5 = Q_4 + 0.5[R_4 - Q_4] = 1.875$

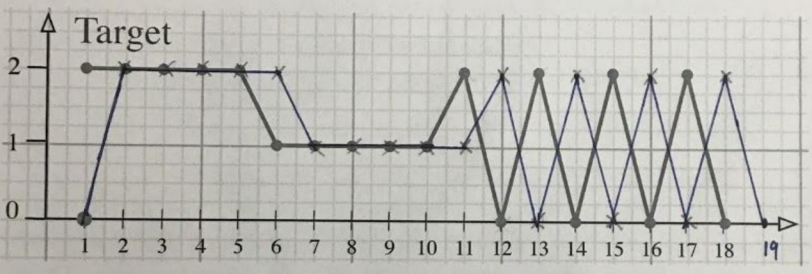
$Q_6 =$

Part 2. [5 pts] Repeat the graphing/plotting portion of Part 1, this time with a step size of 1/8.



$\alpha = 1/8$

Part 3. [5 pts.] Repeat with a step size of 1.0.



$\alpha = 1$

Part 4. [10 pts.] Best step-size questions.

Which of these step sizes would produce estimates of smaller absolute error if the target continued alternating for a long time? Please explain your answer.

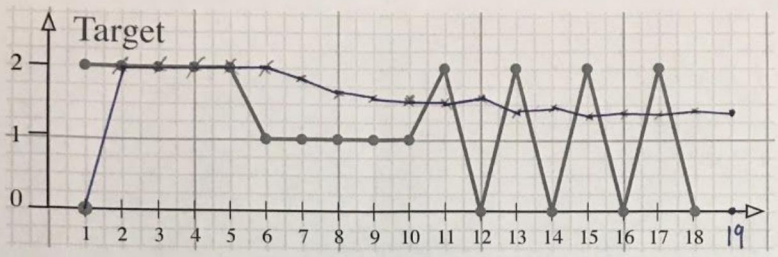
$\alpha = 1/8$ Based on graphs, when $\alpha = 1/8$, the line is much more stable between time-step 6 and 18. It means the fluctuation of target doesn't affect estimate values much. And from equation $Q_{n+1} = Q_n + \alpha[R_n - Q_n]$, for non-stationary target, we want R_n weight less to reduce abs error, so we use small $\alpha = \frac{1}{8}$.

Which of these step sizes would produce estimates of smaller absolute error if the target remained constant for a long time? Please explain your answer.

greedy

$\alpha = 1$. The target is stable, we want very last reward, R_n, weight more to estimate better. From time-step 7 to 10, we can observe that the estimate value follows the trend of target, so it will produce less error.

Part 5. [15 pts.] Repeat with a step size of $1/(t\text{-}1)$ for $t \geq 2$. (i.e., the first step size you will use is 1, the second is 1/2, the third is 1/3, etc.).



$\alpha = 1/(t-1)$

graph of

Based on all of these graphs, why is the 1/(t-1) step size appealing?

Compared with above 3 graphs, $\alpha = 1/(t-1)$ generates less error for a long time. For stable target, it follows the trend, producing small error; for alternating target, the estimate is stable, producing less error. $Q_{n+1} = Q_n + \frac{1}{t-1}(R_n - Q_n) = \frac{R_n + tQ_n}{t-1}$, close to sample average.

Why is the 1/(t-1) step size not always the right choice?

We can observe that when target changes. from time-step 5, it is following the trend, but changes slowly..

For stationary target, $\alpha = 1$ performs better.

For alternating target, $\alpha = \frac{1}{8}$ is better. So constant α sometime is better choice

Question 2 [10 points] Bandit Example. Consider a multi-arm bandit problem with k = 5 actions, denoted 1, 2, 3, 4, and 5. 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 $A_1 = 2$, $R_1 = -2$, $A_2 = 1$, $R_2 = 5$, $A_3 = 3$, $R_3 = 3$, $A_4 = 1$, $R_4 = 4$, $A_5 = 4$, $R_5 = 3$, $A_6 = 2$, $R_6 = -1$. 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?

Possible Possible def random possible def def Conclusion:

$Q_1(a)$	k \ $Q_t(a)$	A_1	A_2	A_3	A_4	A_5	A_6	A_1, A_2, A_4 possibly ~~have~~
0	k= 1	0	▲ 5	5	▲ 4.5	4.5	4.5	
0	k= 2	▲ -2	-2	-2	-2	-2	▲ -1.5	A_3 A_6 A_5 definitely
0	k= 3	0	0	▲ 1	1	1	1	
0	k= 4	0	0	0	0	▲ 3	3	
0	k= 5	0	0	0	0	0	0	

- all a $Q_1(a) = 0$, action 1 choose k_2 — may greedy max (0) — A_1 possible
  - may random
- action 2 choose k_1 — greedy max (0) — A_2 possible
  - random
- action 3 choose k_3 — not greedy max (5) — A_3 random
  - random
- A4 choose k_1 — greedy (5)
  - random — A_4 possible

A_5 choose k_4
- not greedy max (4.5)
- random

A_6 choose k_2
- not greedy max (4.5)
- random

$$
\begin{aligned}
Q_{n+1} &= Q_n + \alpha_n [R_n - Q_n] \\
&= \alpha_n R_n + (1-\alpha_n) Q_n \\
&= \alpha_n R_n + (1-\alpha_n)[\alpha_{n-1} R_{n-1} + (1-\alpha_{n-1}) Q_{n-1}] \\
&= \alpha_n R_n + (1-\alpha_n)\alpha_{n-1} R_{n-1} + (1-\alpha_n)(1-\alpha_{n-1}) Q_{n-1} \\
&= \alpha_n R_n + (1-\alpha_n)\alpha_{n-1} R_{n-1} + (1-\alpha_n)(1-\alpha_{n-1})[\alpha_{n-2} R_{n-2} + (1-\alpha_{n-2}) Q_{n-2}] \\
&= [\alpha_n R_n + (1-\alpha_n)\alpha_{n-1} R_{n-1} + (1-\alpha_n)(1-\alpha_{n-1})\alpha_{n-2} R_{n-2}] + [(1-\alpha_n)(1-\alpha_{n-1})(1-\alpha_{n-2})] Q_{n-2} \\
&= \alpha_n R_n + (1-\alpha_n)\alpha_{n-1} R_{n-1} + (1-\alpha_n)(1-\alpha_{n-1})\alpha_{n-2} R_{n-2} \\
&\quad \cdots + (1-\alpha_n)(1-\alpha_{n-1})(1-\alpha_{n-2})(1-\alpha_{n-3})(1-\alpha_{n-4}) \cdots \alpha_1 R_1 \\
&= \sum_{i=1}^{n} \alpha_i \prod_{j=1}^{n} (1-\alpha_j) R_i + \prod_{i=1}^{n} (1-\alpha_i) Q_1
\end{aligned}
$$

$$
\prod_{i=1}^{n} (1-\alpha_i) Q_1
$$

Question 3. Bandit task Programming. [40 pts.]

This programing exercise will give you hands-on feel for how bandit problems are implemented, and how incremental learning algorithms select actions based on observed rewards. In addition, this exercise will be your first experience with RL-glue, the interface we will use for all programing questions in this course.

Recreate the learning curves for the optimistic bandit agent, and the epsilon-greedy agent in Figure 2.3 of Sutton and Barto. This requires you to implement **three** main components:

10 numbers.

1) A RL-Glue Environment program implementing the 10-armed bandit problem
2) A RL-Glue Agent program implementing an epsilon-greedy bandit learning algorithm. Use the incremental update rule (Equation 2.5), with two different parameter settings:
   - alpha = 0.1, epsilon = 0, and $Q_1 = 5$
   - alpha = 0.1, epsilon = 0.1, and $Q_1 = 0$
3) A RL-Glue Experiment program implementing the experiment to generate the data for your plot. Compute the % Optimal action per time-step, averaged over 2000 runs

All code must be written in **Python3** to be compatible with the RL-Glue interface provided to the class. It is not acceptable to implement your own interface.

$$Q_{n+1} = (1-\alpha_1)^n Q_1 + \sum_{i=1}^{n} \alpha_i (1-\alpha_i)^{n-i} R_i$$

$$=$$

Please submit:
1) your plot [10 pts.]
2) all your code (including any graphing code used to generate your plot) [30 pts.]

Bonus Programing Question. [5 pts.]
Implement the UCB agent described in chapter two and evaluate it on the bandit environment from Question 3. Can you get the UCB agent to outperform the epsilon-greedy agent? Feel free to modify the parameters of the epsilon-greedy agent (alpha, epsilon, and the initial Q estimates) in order to better understand the relative strengths of both algorithms. Describe how we would go about determining and reporting on which agent is better for this task.

$$Q_{n+1} = \prod_{i=1}^{n} (1-\alpha_i) Q_1 + \sum_{i=1}^{n} \alpha_i \prod_{j=i+1}^{n} (1-\alpha_j) R_i$$

Bonus Question. [5 points extra credit]
Exercise 2.4 from Sutton and Barto (*Reward weighting for general step sizes*)

Bonus Question. [5 pts.]
Exercise 2.6 from Sutton and Barto (*Mysterious Spikes*. Use your implementation from Question 3 to better understand what is happening in Figure 2.3)

at the beginning

According to the plot, the method is so optimal which means it is highly possible to select the optimal action. And the agents will go through all the ten actions once initially. We can observe an obvious spike on step 11, as an optimal action, After selecting an obvious optimal action, it may select at step 12 as well. We can also observe that another spike on step 19 or step 21, shows optimistic. On average, it shows less fluctuation.

Optimistic ($\alpha = 0, Q_1 = 5$)