Online and Reinforcement Learning

2023-2024

Home Assignment 5

Yevgeny Seldin **Sadegh Talebi**
Department of Computer Science
University of Copenhagen

The deadline for this assignment is **13 March 2024, 21:00**. You must submit your *individual* solution electronically via the Absalon home page.

A solution consists of:

- A PDF file with detailed answers to the questions, which may include graphs and tables if needed. Do *not* include your full source code in the PDF file, only selected lines if you are asked to do so.
- A .zip file with all your solution source code with comments about the major steps involved in each question (see below). Source code must be submitted in the original file format, not as PDF. The programming language of the course is Python.

Important Remarks:

- IMPORTANT: Do NOT zip the PDF file, since zipped files cannot be opened in *SpeedGrader*. Zipped PDF submissions will not be graded.
- Your PDF report should be self-sufficient. I.e., it should be possible to grade it without opening the .zip file. We do not guarantee opening the .zip file when grading.
- Your code should be structured such that there is one main file (or one main file per question) that we can run to reproduce all the results presented in your report. This main file can, if you like, call other files with functions, classes, etc.
- Handwritten solutions will not be accepted.

1 Improved Parametrization of UCB1 (Optional, but recommended. 0 points)

In this question we improve the UCB1 algorithm presented in the lecture notes.

Show that if we define the upper confidence bounds in UCB1 by

$$\hat{\mu}_{t-1}(a) + \sqrt{\frac{\ln t}{N_{t-1}(a)}}$$

then its pseudo-regret satisfies

$$\bar{R}_T \leq 4 \sum_{a:\Delta(a)>0} \frac{\ln T}{\Delta(a)} + (2\ln(T) + 3)\sum_a \Delta(a).$$

Hint: The T-th harmonic number, $\sum_{t=1}^{T} \frac{1}{t}$, satisfies $\sum_{t=1}^{T} \frac{1}{t} \leq \ln(T) + 1$.

Comment: The UCB1 algorithm in the lecture notes takes confidence intervals $\sqrt{\frac{3\ln t}{2N_{t-1}(a)}} = \sqrt{\frac{\ln t^3}{2N_{t-1}(a)}}$, corresponding to confidence parameter $\delta = \frac{1}{t^3}$. The modified UCB1 algorithm in this question takes confidence intervals $\sqrt{\frac{2\ln t}{2N_{t-1}(a)}} = \sqrt{\frac{\ln t^2}{2N_{t-1}(a)}}$, corresponding to confidence parameter $\delta = \frac{1}{t^2}$. The original algorithm and analysis by Auer et al. (2002) uses confidence intervals $\sqrt{\frac{4\ln t}{2N_{t-1}(a)}} = \sqrt{\frac{\ln t^4}{2N_{t-1}(a)}}$, corresponding to confidence parameter $\delta = \frac{1}{t^4}$, due to one unnecessary union bound. The reason we can move from $\delta = \frac{1}{t^3}$ to $\delta = \frac{1}{\delta^2}$ is not due to elimination of additional union bounds (we still need two of them), but due to a compromise on the last term in the regret bound.

2 Introduction of New Products (25 points)

Imagine that we have an established product on the market, which sells with probability 0.5. We have received a new product, which sells with an unknown probability μ. Assume that at every sales round we can offer only one product, so we have to choose between offering the established or the new product. Propose a strategy for maximizing the number of sales and analyze its pseudo-regret. Write your answer in terms of the gap $\Delta = 0.5 - \mu$. (The regret bound for $\Delta > 0$ is different from the regret bound for $\Delta < 0$, i.e., you should obtain two separate answers for positive and negative gap.)

Hint: The solution is *not* an application of an existing algorithm. You should design a *new* algorithm tailored for the problem. The new algorithm will not be very different from something we have studied, but you have to make a small adaptation; this is the whole point of the question.

Put attention that if $\mu > 0.5$ (the new product is better than the old one) then $\Delta < 0$, and if $\mu < 0.5$ (the new product is worse than the old one) then $\Delta > 0$. If you do things correctly, for $\Delta < 0$ the regret of your algorithm should be bounded by a constant that is independent of the number of game rounds T, and for $\Delta > 0$ it should grow logarithmically with T. The algorithm should not know whether Δ is positive or negative, but the analysis of the two cases can be done separately.

3 Empirical comparison of FTL and Hedge (25 points)

Assume that you have to predict a binary sequence $X_1, X_2, \ldots$ and that you know that X_i-s are i.i.d. Bernoulli random variables with an unknown bias μ. (You know that X_i-s are i.i.d., but you do not know the value of μ.) At every round you can predict "0" or "1" (i.e., you have two actions - "predict 0" or "predict 1") and your loss is the zero-one loss depending on whether your prediction matches the outcome. The regret is computed with respect to the performance of the best out of the two possible actions.

1. Write a simulation that compares numerically the performance of Follow The Leader (FTL) algorithm with performance of the Hedge algorithm that we presented in class (with $\eta = \sqrt{\frac{2\ln K}{T}}$) and performance of the reparametrized Hedge algorithm from Question 7 (with $\eta = \sqrt{\frac{8\ln K}{T}}$). The Hedge algorithm should operate with the aforementioned two actions. To make things more interesting we will add an "anytime" version of Hedge to the comparison. "Anytime" algorithm is an algorithm that does not depend on the time horizon. Let t be a running time index ($t = 1, \ldots, T$). Anytime Hedge corresponding to the simple analysis uses $\eta_t = \sqrt{\frac{\ln K}{t}}$ and anytime Hedge corresponding to the tighter analysis in Question 7 uses $\eta_t = 2\sqrt{\frac{\ln K}{t}}$ (the learning rate η_t of anytime Hedge changes with time and does not depend on the time horizon). Some instructions for the simulation:
   - Take time horizon $T = 2000$. (In general, the time horizon should be large in comparison to $\frac{1}{(\mu-\frac{1}{2})^2}$.)
   - Test several values of μ. We suggest $\mu = \frac{1}{2} - \frac{1}{4}$, $\mu = \frac{1}{2} - \frac{1}{8}$, $\mu = \frac{1}{2} - \frac{1}{16}$.

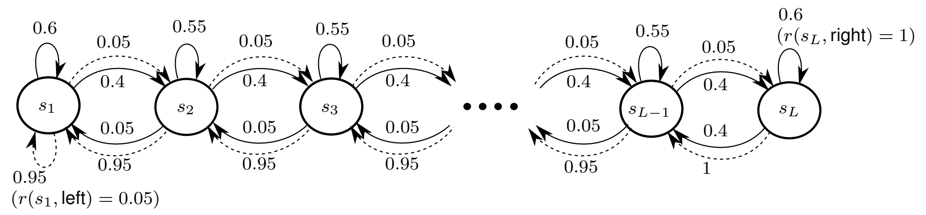

Figure 1: The Ergodic RiverSwim MDP

- Plot the empirical pseudo regret defined by $\hat{R}_t = \sum_{s=1}^{t} \Delta(A_s)$ of the five algorithms with respect to the best out of "0" and "1" actions as a function of t for $1 \leq t \leq T$ and for the different values of μ (make a separate plot for each μ). Make 10 runs of each algorithm and report the average empirical pseudo regret over the 10 runs and the average empirical pseudo regret + one standard deviation over the 10 runs. Do not forget to add a legend to your plot.

2. Which values of μ lead to higher regret? How the relative performance of the algorithms evolves with time and does it depend on μ?

3. Design an adversarial (non-i.i.d.) sequence, which makes the FTL algorithm perform poorly. Explain the design of your adversarial sequence and report a plot with a simulation, where you compare the performance of FTL with the different versions of Hedge. As before, make 10 repetitions of the experiment and report the average regret (in this case you should use regret and not pseudo regret) and the average + one standard deviation. Comment on your observations.

4 Q-Learning vs. UCB Q-Learning (5 points)

Consider Q-Learning and UCB Q-Learning. Briefly compare the two algorithms in terms of settings and objectives.

5 Empirical Evaluation of MBIE (25 points)

In this exercise, we empirically examine the performance of the `MBIE` algorithm (Strehl and Littman, 2008). Let us consider the specific version of the algorithm presented in the lecture slides.

We are interested in examining it in a 5-state *Ergodic RiverSwim* (Figure 1) with $\gamma = 0.92$. We set $\varepsilon = 0.1$ and $\delta = 0.05$, and let the starting state be sampled from a distribution μ defined as:

$$\mu(s_1) = \mu(s_5) = 0, \qquad \mu(s_2) = \frac{2}{5}, \qquad \mu(s_3) = \frac{2}{5}, \qquad \mu(s_4) = \frac{1}{5}.$$

A Python implementation of this variant is provided in `HA5_Q5.py`, but you need to modify the code so that the starting state is sampled from μ.

We set the time horizon to $T = 3 \times 10^4$. Let $n(t)$ denote the cumulative number of ε-bad time steps as a function of t, namely,

$$n(t) = \sum_{t'=1}^{t} \mathbb{I}\{V^{\pi_{t'}}(s_{t'}) < V^\star(s_{t'}) - \varepsilon\}.$$

(i) Plot $n(t)$, averaged over 30 independent runs, versus time t, and report the corresponding 95% confidence intervals. (Recall that for a set of i.i.d. random variables $\mathbf{X} = \{X_1, \ldots, X_n\}$, we define the 95% confidence interval by

$$\left[\texttt{mean}(\mathbf{X}) - 1.96\frac{\texttt{std}(\mathbf{X})}{\sqrt{n}}, \texttt{mean}(\mathbf{X}) + 1.96\frac{\texttt{std}(\mathbf{X})}{\sqrt{n}}\right],$$

where `mean` and `std` denote the mean and the standard deviation, respectively.)

MDP \ Class	Ergodic	Communicating	Weakly-Communicating
RiverSwim			
RiverSwim-2			
4-room grid-world			

Table 1: MDP Classes

(ii) We would like to assess the quality of the output (i.e., learned) policy π_t at time t, in terms of $V^{\pi_t}(s_t)$. Plot $V^{\pi_t}(s_t)$, the (true) value of the learned policy at s_t as a function of time t, averaged over 30 independent runs. Also, report the corresponding 95% confidence intervals. Also plot $V^\star(s_t)$ versus time.

(Note: Computing the value function at each t could render computationally heavy if done naively. One fix is to maintain a look-up table as there are a few relevant policies only. Another way is to report V^{π_t} every, e.g., 100 *steps.)*

(iii) We now repeat Part (ii) but for V^{π_t} evaluated at a state sampled from a *test distribution* μ_{test}. We consider a uniform test distribution (i.e., $\mu_{\mathsf{test}}(s) = 1/S$ for all $s \in$). Plot $\mathbb{E}_{s\sim\mu_{\mathsf{test}}}[V^{\pi_t}(s)] = \frac{1}{S}\sum_{s\in\mathcal{S}} V^{\pi_t}(s)$ as a function of time t, averaged over 30 independent runs. Also, report the corresponding 95% confidence intervals. In your plot, also plot the horizontal line $\mathbb{E}_{s\sim\mu_{\mathsf{test}}}[V^\star(s)]$.

6 MDP Classes (20 points)

For each of the following MDPs, determine whether or not it belongs to the class of ergodic MDPs, communicating MDPs, or weakly-communicating MDPs. Then fill out Table 1 with `True` or `False`. (Note that all entries must be filled out.)

In each case, argue why membership to a certain class holds. To rule out membership, you must provide a concrete counter example; e.g., if M is not ergodic, you must provide a policy that violates the definition of ergodic MDP.

(i) RiverSwim (Figure 2).

(ii) A modified RiverSwim (let's call it RiverSwim-2), which is defined similarly to RiverSwim except that it has an additional state $s_{\texttt{extra}}$ with two actions a_1 and a_2: Under action a_1, the agent is moved to s_1 (left-hand bank) deterministically; under action a_2, the next-state is s_1 (w.p. 0.5) and s_2 (w.p. 0.5). The rest of transitions are the same as RiverSwim.

(iii) 4-room grid-world, which was introduced in earlier assignments. We present it below for completeness. This MDP is shown in Figure 3. The agent has perform 4 actions (when away from walls): Going up, left, down, or right. However, the floor is slippery and brings stochasticity to the next-state. Specifically, under each of the aforementioned four actions, she moves in the chosen direction (with probability 0.7), stays in the same state (with probability 0.1), or goes in each of the two perpendicular directions (each with probability 0.1) —this environment is sometimes referred to as the *frozen lake* MDP. *Walls act as reflectors, i.e., they cause moving back to the* ***current state****.* Once the agent reaches the rewarding state (in yellow), she is *teleported* to the initial state irrespective of the action, that is, she will find herself in the upper-left corner at the beginning of the following slot.

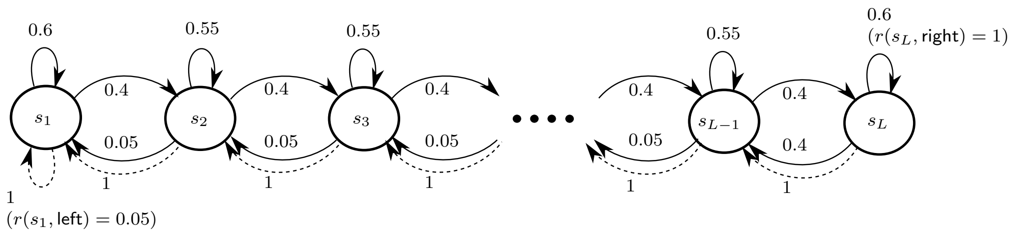

Figure 2: The RiverSwim MDP

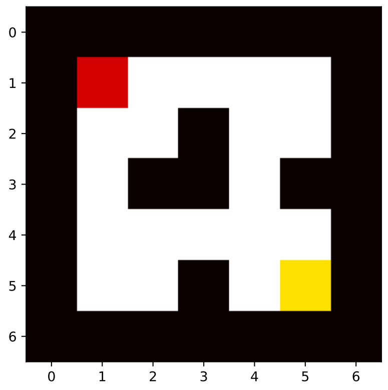

Figure 3: The 4-Room Grid-World MDP

7 [Optional, not for submission] A tighter analysis of the Hedge algorithm

1. Apply Hoeffding's lemma (Lemma 2.6 in Yevgeny's lecture notes) in order to derive a better parametrization and a tighter bound for the expected regret of the Hedge algorithm. [Do not confuse Hoeffding's lemma with Hoeffding's inequality!] Guidance:

   (a) Traverse the analysis of the Hedge algorithm that we did in class. There will be a place where you will have to bound expectation of an exponent of a function ($\sum_a p_t(a)e^{-\eta X_{t,a}}$). Instead of going the way we did, apply Hoeffding's lemma.

   (b) Find the value of η that minimizes the new bound. (You should get $\eta = \sqrt{\frac{8 \ln K}{T}}$ - please, prove this formally.)

   (c) At the end you should obtain $\mathbb{E}\left[R_T\right] \leq \sqrt{\frac{1}{2} T \ln K}$. (I.e., you will get an improvement by a factor of 2 compared to what we did in class.)

   Remark: Note that the regret upper bound matches the lower bound up to the constants. This is an extremely rare case.

2. Explain why the same approach cannot be used to tighten the regret bound for the EXP3 algorithm.

8 [Optional, not for submission] Regularization by relative entropy and the Gibbs distribution

In this question we will show that regularization by relative entropy leads to solutions in a form of the Gibbs distribution. Let's assume that we have a finite hypothesis class $\mathcal{H}$ of size m and we want to minimize

$$\mathcal{F}(\rho) = \alpha\mathbb{E}_\rho\left[\hat{L}(h,S)\right] + \mathrm{KL}(\rho\|\pi) = \alpha\sum_{h=1}^m \rho(h)\hat{L}(h,S) + \sum_{h=1}^m \rho(h)\ln\frac{\rho(h)}{\pi(h)}$$

with respect to the distribution ρ. This objective is closely related to the objective of PAC-Bayes-λ inequality when λ is fixed and this sort of minimization problem appears in many other places in machine learning. Let's slightly simplify and formalize the problem. Let $\rho = (\rho_1, \dots, \rho_m)$ be the posterior distribution, $\pi = (\pi_1, \dots, \pi_m)$ the prior distribution, and $L = (L_1, \dots, L_m)$ the vector of losses. You should solve

$$\begin{aligned} \min_{\rho_1,\dots,\rho_m} \quad & \alpha\sum_{h=1}^m \rho_h L_h + \sum_{h=1}^m \rho_h \ln\frac{\rho_h}{\pi_h} \\ s.t. \quad & \sum_{h=1}^m \rho_h = 1 \\ & \forall h : \rho_h \geq 0 \end{aligned} \tag{1}$$

and show that the solution is $\rho_h = \frac{\pi_h e^{-\alpha L_h}}{\sum_{h'=1}^m \pi_{h'} e^{-\alpha L_{h'}}}$. Distribution of this form is known as the Gibbs distribution.

Guidelines:

1. Take a shortcut. Instead of solving minimization problem (1), solve the following minimization problem

$$\begin{aligned} \min_{\rho_1,\dots,\rho_m} \quad & \alpha\sum_{h=1}^m \rho_h L_h + \sum_{h=1}^m \rho_h \ln\frac{\rho_h}{\pi_h} \\ s.t. \quad & \sum_{h=1}^m \rho_h = 1, \end{aligned} \tag{2}$$

   i.e., drop the last constraint in (1).

2. Use the method of Lagrange multipliers to show that the solution of the above problem has a form of $\rho_h = \pi_h e^{-\alpha L_h + \texttt{something}}$, where `something` is something involving the Lagrange multiplier.

3. Show that $\rho_h \geq 0$ for all h. (This is trivial. But it gives us that the solutions of (1) and (2) are identical.)

4. Finally, $e^{\texttt{something}}$ should be such that the constraint $\sum_{h=1}^m \rho_h = 1$ is satisfied. So you can easily get the solution. You even do not have to compute the Lagrange multiplier explicitly.

9 [Optional, not for submission] Decoupling exploration and exploitation in i.i.d. multiarmed bandits

Assume an i.i.d. multiarmed bandit game, where the observations are not coupled to the actions. Specifically, we assume that at each round of the game the player is allowed to observe the reward of a single arm, but it does not have to be the same arm that was played at that round (and if it is not the same arm, the player does not observe its own reward, but instead observes the reward of the alternative option).

Derive a playing strategy and a regret bound for this game. (You should solve the problem for a general K and you should get that the regret does not grow with time.)

Remark: note that in this setting the exploration is "free", because we do not have to play suboptimal actions in order to test their quality. And if we contrast this with the standard multiarmed bandit setting *we observe that the regret stops growing with time instead of growing logarithmically with time. Actually, the result that you should get is much closer to the regret bound for FTL with full information than to the regret bound for multiarmed bandits. Thus, it is not the fact that we have just a single observation that makes i.i.d. multiarmed bandits harder than full information games, but the fact that this single observation is linked to the action. In adversarial multiarmed bandits the effect of decoupling is more involved (Avner et al., 2012, Seldin et al., 2014, Rouyer and Seldin, 2020).*

References

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47, 2002.

Orly Avner, Shie Mannor, and Ohad Shamir. Decoupling exploration and exploitation in multi-armed bandits. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2012.

Chloé Rouyer and Yevgeny Seldin. Tsallis-INF for decoupled exploration and exploitation in multi-armed bandits. In *Proceedings of the Conference on Learning Theory (COLT)*, 2020.

Yevgeny Seldin, Peter L. Bartlett, Koby Crammer, and Yasin Abbasi-Yadkori. Prediction with limited advice and multiarmed bandits with paid observations. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2014.

Alexander L Strehl and Michael L Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.