Sequential Learning

Home Assignment

This homework should be uploaded by **Friday, March 12, 2021** on the website

`http://pierre.gaillard.me/teaching/mva.php`

The password to upload is `mva2021`. The penalty scale is minus two points (on the final grade over 20 points) for every day of delay. The homework can be done alone or in groups of two students. The code can be done in any langage (`python`, `R`, `matlab`,...) and should not be returned but the results and the figures must be included into the pdf report.

All questions require a proper mathematical justification or derivation (unless otherwise stated), but most questions can be answered concisely in just a few lines. No question should require lengthy or tedious derivations or calculations.

Part 1. Rock Paper Scissors

We consider the sequential version of a repeated two-player zero-sum games between a player and an adversary.

Let $L \in [-1,1]^{M\times N}$ be a loss matrix.
At each round $t = 1, \dots, T$
- The player choose a distribution $p_t \in \Delta_M := \{p \in [0,1]^M, \sum_{i=1}^M p_i = 1\}$
- The adversary chooses a distribution $q_t \in \Delta_N$
- The actions of both players are sampled $i_t \sim p_t$ and $j_t \sim q_t$
- The player incurs the loss $L(i_t, j_t)$ and the adversary the loss $-L(i_t, j_t)$.

Setting 1: Setting of a sequential two-player zero sum game

1. Recall M, N and a loss matrix $L \in [-1,1]^{M\times N}$ that corresponds to the game "Rock paper scissors"[1].

Full information feedback We assume that both players know the matrix L in advance and can compute $L(i,j)$ for any (i,j).

2. *Implementation of* ***EWA****.*
   (a) In order to implement the exponential weight algorithm, you need a way to sample from the exponential weight distribution. Implement the function `rand_weighted` that takes as input a probability vector $p \in \Delta_M$ and uses a single call to `rand()` to return $X \in [M]$ with $P(X = i) = p_i$.

[1] This is a common game where two players choose one of 3 options: (Rock, Paper, Scissors). The winner is decided according to the following: Rock crushes scissors, Paper covers Rock, Scissors cuts paper

(b) Define a function `EWA_update` that takes as input a vector $p_t \in \Delta_M$ and a loss vector $\ell_t \in [-1,1]^M$ and return the updated vector $p_{t+1} \in \Delta_M$ defined for all $i \in [M]$ by

$$p_{t+1}(i) = \frac{p_t(i)\exp(-\eta \ell_t(i))}{\sum_{j=1}^M p_t(j)\exp(-\eta \ell_t(j))}\,.$$

3. *Simulation against a fixed adverary.* Consider the game "Rock paper scissors" and assume that the adversary chooses $q_t = (1/2, 1/4, 1/4)$ and samples $j_t \sim q_t$ for all rounds $t \geq 1$.
   (a) What is the loss $\ell_t(i)$ incurred by the player if he chooses action i at time t? Simulate an instance of the game for $t = 1, \dots, T = 100$ for $\eta = 1$.
   (b) Plot the evolution of the weight vectors $p_1, p_2, \dots, p_T$. What seems to be the best strategy against this adversary?
   (c) Plot the average loss $\bar{\ell}_t = \frac{1}{t}\sum_{s=1}^t \ell(i_s, j_s)$ as a function of t.
   (d) Plot the cumulative regret.
   (e) To see if the algorithm is stable, repeat the simulation $n = 10$ times and plot the average loss $(\bar{\ell}_t)_{t\geq 1}$ obtained in average, in maximum and in minimum over the n simulations.
   (f) Repeat one simulation for different values of learning rates $\eta \in \{0.01, 0.05, 0.1, 0.5, 1\}$ and plot the final regret as a function of η. What are the best η in practice and in theory.

4. *Simulation against an adaptive adversary.* Repeat the simulation of question 3) when the adversary is also playing EWA with learning parameters $\eta = 0.05$.
   (a) Plot $\frac{1}{t}\sum_{s=1}^t \ell(i_s, j_s)$ as a function of t.

   It is possible to show that if both players play according to a regret minimizing strategy the cumulative loss of the player converges to the value of the game

   $$V = \min_{p\in\Delta_M} \max_{q\in\Delta_q} p^\top L q\,.$$

   (b) Define $\bar{p}_t = \frac{1}{t}\sum_{s=1}^t p_s$. Plot in log log scale $\|\bar{p}_t - (1/3,1/3,1/3)\|_2$ as a function of $t = 1, \dots, 10\,000$.

   It is possible to show that $(\bar{p}_t, \bar{q}_t)_{t\geq 1}$ converges almost surely to a Nash equilibrium of the game. This means that if $p \times q$ is a Nash equilibrium, none of the players should change is strategy if the other player does not change hers.

Bandit feedback Now, we assume that the players do not know the game in advance but only observe the performance $L(i_t, j_t)$ (that we assume here to be in $[0,1]$) of the actions played at time t. They need to learn the game and adapt to the adversary as one goes along.

5. *Implementation of `EXP3`.* Since both players are symmetric, we focus on the first player.
   (a) Implement the function `estimated_loss` that takes as input the action $i_t \in [M]$ played at round $t \geq 1$ and the loss $L(i_t, j_t)$ suffered by the player and return the vector of estimated loss $\widehat{\ell}_t \in \mathbb{R}_+^M$ used by `EXP3`.
   (b) Implement the function `EXP3_update` that takes as input a vector $p_t \in \Delta_M$, the action $i_t \in [M]$ played by the player and the loss $L(i_t, j_t)$ and return the updated weight vector $p_{t+1} \in \Delta_M$.

6. Repeat Questions 3.a) to 3.f) with `EXP3` instead of `EWA`.

7. Repeat Question 4.a) and 4.b) with `EXP3` instead of `EWA`.

Optional extentions The following questions are optional.

8. Repeat Question 4.a) when the adversary is playing a `UCB` algorithm. Who wins between `UCB` and `EXP3`?

9. In the lecture 3, we see that EXP3 has a sublinear expected regret. Yet, as shown by question 6.e), it is extremely unstable with a large variance. Implement `EXP3.IX` (see Chapter 12 of [3]) a modification of `EXP3` that controls the regret in expectation and simultaneously keeps it stable. Repeat question 3.e) with `EXP3.IX`

10. Try different games (not necessarily zero-sum games). In particular, how these algorithms behave for the prisoner's dilemna (see wikipedia)? The prisoner's dilemna is a two-player games that shows why two completely rational individuals might not cooperate, even if it appears that it is in their best interests to do so. The losses matrices are:
$$L^{(player)} = \begin{pmatrix} 1 & 3 \\ 0 & 2 \end{pmatrix} \quad \text{and} \quad L^{(adversary)} = \begin{pmatrix} 1 & 0 \\ 3 & 2 \end{pmatrix}.$$

Part 2. Bernoulli Bandits

We consider a stochastic bandit setting in which the arm rewards have Bernoulli distributions. A random variable X is said to have Bernoulli distribution with parameter p, which we denote by $\mathcal{B}(p)$, if it takes value 0 with probability $1-p$ and value 1 with probability p. The set $\{1,\dots,K\}$ is denoted by $[K]$.

Each arm $k \in [K]$ has a reward distribution $\mathcal{B}(p_k)$.
At each round $t = 1,\dots,T$
- The player chooses an arm $k_t \in [K]$,
- The player observes a reward $X_t^{k_t} \sim \mathcal{B}(p_{k_t})$, independent of all other rewards.

Setting 2: Bernoulli bandit

Notations:
- In this part, the term "regret" refers to the quantity $R_T = \max_{k\in[K]} Tp_k - \sum_{t=1}^T p_{k_t}$.
- N_t^k denotes the number of pulls of arm k *before* time t, i.e. $N_t^k = \sum_{s=1}^{t-1} \mathbb{I}\{k_s = k\}$.
- $\hat{\mu}_t^k$ denotes the empirical mean of arm k: $\hat{\mu}_t^k = \frac{1}{N_t^k}\sum_{s=1}^{t-1} X_s^{k_s}\mathbb{I}\{k_s = k\}$.

A bit of context: why Bernoulli bandits matter. Many applications have binary outcomes, in which the reward then follows a Bernoulli distribution. A prominent example is online advertising, in which a seller shows advertisements to visitors of a website, and a usual goal is to maximize the probability that the visitor clicks on the ad. In its most basic form, this is exactly the bandit interaction described above: the seller (player) chooses an ad (arm) which is displayed to the visitor, and then the seller observes whether there is a click or not (reward). More elaborate models of that interaction take into account prior information that the seller has about the visitor, turning it into a *contextual* bandit, or get rid of the independence assumption, etc.

1. *Follow the leader.* All experiments in this question will be done for $K = 2$, $p = (0.5, 0.6)$.
   (a) Prove that the expected regret of the Follow-the-leader algorithm (`FTL`) verifies $\mathbb{E}R_T \geq \alpha T$, for some $\alpha > 0$. Recall that `FTL` pulls at each time the arm with highest empirical mean.
   (b) Implement `FTL`.

(c) For time $T = 100$, plot a histogram of the regret R_T of `FTL` over 1000 repetitions of the experiment. Explain the figure.

(d) Plot the mean regret of `FTL` over 1000 repetitions, as a function of $t \in \{1, \dots, 1000\}$. Is FTL a good algorithm for stochastic bandits?

2. *UCB.* A random variable is said to be σ^2-sub-Gaussian if for all $\lambda \in \mathbb{R}$, $\mathbb{E}[e^{\lambda(X-\mathbb{E}[X])}] \leq e^{\frac{1}{2}\sigma^2\lambda^2}$. The $\texttt{UCB}(\sigma^2)$ algorithm pulls arm $k_t = \arg\min_{k\in[K]} \hat{\mu}_t^k + \sqrt{\frac{2\sigma^2 \log(t)}{N_t^k}}$. It is designed to have low regret on σ^2-sub-Gaussian random variables.

(a) Compute the cumulant generating function, defined for $\lambda \in \mathbb{R}$ by $\phi_X(\lambda) = \log \mathbb{E}[e^{\lambda(X-\mathbb{E}[X])}]$, for a Bernoulli random variable with parameter p.

(b) Prove that if a random variable X (not necessarily Bernoulli) verifies $\phi_X''(\lambda) \leq \sigma^2$ for all $\lambda \in \mathbb{R}$, then the random variable is σ^2-sub-Gaussian. Remark: this is not an equivalence (you are not required to prove this).

(c) Using question 2.b, find σ^2 such that a random variable with distribution $\mathcal{B}(p)$ is σ^2-sub-Gaussian.

(d) Prove that a random variable X supported on $[0, 1]$ with mean $p \in [0, 1]$ verifies $\phi_X(\lambda) \leq \phi_Y(\lambda)$ for all $\lambda \in \mathbb{R}$, where Y has a $\mathcal{B}(p)$ distribution. Hint: prove that for all $x \in [0, 1]$, for all $\lambda \in \mathbb{R}$, $e^{\lambda x} \leq 1 - x + xe^{\lambda}$.

(e) Prove that all random variables supported on $[0, 1]$ are sub-Gaussian.

(f) Implement the $\texttt{UCB}(\sigma^2)$ algorithm.

(g) Plot the mean regret of $\texttt{UCB}(1/4)$ as a function of time up to $T = 1000$ for $K = 2$, $p = (0.5, 0.6)$, over 1000 repetitions. Compare with the result of question 1.d.

(h) For $K = 2$, $p = (0.6, 0.5)$, $T = 1000$, plot the mean regret of $\texttt{UCB}(\sigma^2)$ over 1000 repetitions as a function of σ^2, for $\sigma^2 \in \{0, 1/32, 1/16, 1/4, 1\}$. Do it again for $p = (0.85, 0.95)$ and compare the results: does the optimal parameter change? How does it compare to the theoretic parameter?

The results of the question 2.c on Bernoulli distributions can be improved: it is possible to prove that a random variable with distribution $\mathcal{B}(p)$ is σ^2-sub-Gaussian with parameter $\sigma^2(p) = 0$ if $p \in \{0, 1\}$, $\sigma^2(p) = 1/4$ if $p = 1/2$ and $\sigma^2(p) = \frac{1}{2}\frac{p-(1-p)}{\log p - \log(1-p)}$ for $p \in (0, 1) \setminus \{1/4\}$.

3. On the same figure, plot the variance of $\mathcal{B}(p)$ and the sub-Gaussian constant $\sigma^2(p)$ described above as a function of $p \in [0, 1]$.

4. **(optional)** Prove that a σ^2-sub-Gaussian random variable has variance bounded by σ^2.

Adaptation to the variance. The algorithm $\texttt{UCB}(\sigma^2)$ uses only the empirical mean of the arms to choose the next arm, except for a parameter σ^2 which has to be chosen such that all arms are σ^2-sub-Gaussian. In particular, all variance information about the distributions is lost. Intuitively an arm with lower variance should require fewer samples in order to know its mean with enough precision.

5. UCB-V. For bounded rewards belonging to $[0, b]$, the algorithm $\texttt{UCB-V}(b, \xi, c)$ (V for variance) computes the empirical variance of the arms, $\hat{v}_t^k = \frac{1}{N_t^k}\sum_{s=1}^{t-1} \mathbb{I}\{k_s = k\}(X_s^{k_s} - \hat{\mu}_t^k)^2$ and pulls the arm $k_t = \arg\max_{k\in[K]} \hat{\mu}_t^k + \sqrt{\frac{2\hat{v}_t^k \xi \log t}{N_t^k}} + \frac{3bc\xi}{N_t^k}$. For theoretical regret bounds to hold, ξ should be taken slightly larger than 1 and c larger than a function of ξ, which increases as $\xi \to 1$. **All experiments in this question will be done for $b = 1$, $\xi = 1.2$ and $c = 1$.**

(a) Prove that $N_t^k \hat{v}_t^k = \sum_{s=1}^{t-1} \mathbb{I}\{k_s = k\}(X_s^{k_s})^2 - \frac{1}{N_t^k}(\sum_{s=1}^{t-1} \mathbb{I}\{k_s = k\} X_s^{k_s})^2$.

(b) Prove that $N_{t+1}^{k_t} \hat{v}_{t+1}^{k_t} = N_t^{k_t} \hat{v}_t^{k_t} + (X_t^{k_t} - \hat{\mu}_t^{k_t})(X_t^{k_t} - \hat{\mu}_{t+1}^{k_t})$. What is the practical advantage of that formulation?

(c) Implement `UCB-V`.

(d) On the same figure, plot the mean regret of `UCB-V` and `UCB`$(1/4)$ as a function of time up to $T = 1000$ for $K = 2$, $p = (0.5, 0.6)$, over 1000 repetitions.

(e) Same question for $p = (0.1, 0.2)$ and $p = (0, 0.1)$. Compare to the results of 5.d. When does `UCB-V` improve over `UCB`?

Algorithms for parametric distributions. `UCB` uses only an estimate of the mean, while `UCB-V` uses estimates of the mean and variance. However, Bernoulli distributions have many properties beyond their mean and variance, and these properties are not used by `UCB-V`. We can design algorithms that perform better by using fully the knowledge that the distribution of the arms are Bernoulli $\mathcal{B}(p)$, with the only unknown being the parameter p. The algorithm `kl-UCB` is designed precisely to take advantage of the knowledge that distributions belong to a so-called one-parameter exponential family, and that algorithm can use fully the Bernoulli assumption. See [2, 1].

6. **(optional)** Implement the `kl-UCB` algorithm for Bernoulli bandits (see [2, 1]), and compare with `UCB` and `UCB-V` on various Bernoulli bandit problems, for examples the settings of questions 5.d and 5.e.

References

[1] Olivier Cappé, Aurélien Garivier, Odalric-Ambrym Maillard, Rémi Munos, Gilles Stoltz, et al. Kullback–leibler upper confidence bounds for optimal sequential allocation. *Annals of Statistics*, 41(3):1516–1541, 2013.

[2] Aurélien Garivier and Olivier Cappé. The kl-ucb algorithm for bounded stochastic bandits and beyond. In *Proceedings of the 24th annual conference on learning theory*, pages 359–376. JMLR Workshop and Conference Proceedings, 2011.

[3] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.