STATS 710 – Seq. Dec. Making with mHealth Applications Sep 29, 2016

Lecture 8: The non-stochastic MAB problem

Instructors: Susan Murphy and Ambuj Tewari *Scribe: Young Jung*

1 The non-stochastic MAB problem

1.1 Introduction

- Original paper: Auer et al. [ACBFS02]
- This lecture mainly focuses on chapter 3 of Bubeck and Cesa-Bianchi [BCB$^+$12]

Until now, we have assumed that reward is given as an i.i.d. sequence for each action. However, we will throw out this assumption, and nature (an adversary) chooses the reward matrix in an arbitrary fashion. In other words, if our MAB problem consists of K arms and T time steps, nature decides the reward matrix $(r_{a,t})_{a\in\mathcal{A},t\in[T]}$, which a player cannot observe. At time t, if the learner picks $A_t = a$, then it observes the reward $r_{a,t}$. Note that the learner only observes just one element of t^{th} column. If the entire column is revealed, we call this problem as a full information problem.

We usually assume the reward matrix is bounded, and specifically throughout the lecture, we will assume every element is in $[0, 1]$. In the previous MAB we have studied, there was randomness in distribution and potentially randomness in the algorithm. Now, there is no randomness in distribution, and therefore the only source of randomness is the algorithm. Thus if the leaner is deterministic, there is no randomness.

We can define regret as a difference from the best row-sum:

$$\mathcal{R}_0(\mathcal{L}, (r_{a,t})_{a\in\mathcal{A},t\in[T]}) = \max_{a\in\mathcal{A}} \sum_t r_{a,t} - \sum_t r_{A_t,t}$$

It will be shown shortly that a deterministic algorithm cannot attain sublinear regret and a random algorithm can attain $O(\sqrt{TK\log K})$ regret. In full information case, one can have minimax regret of $O(\sqrt{T\log K})$. Thus we suffer a bigger regret of order $\sqrt{K}$ by not looking at whole column.

One can also define the regret using different best cases. For example, instead of taking maximum row-sum, we can pick maximum value from each column:

$$\mathcal{R}_1(\mathcal{L}, (r_{a,t})_{a\in\mathcal{A},t\in[T]}) = \max_{a_1,\cdots,a_T} \sum_t r_{a_t,t} - \sum_t r_{A_t,t}$$

This is also a legitimate way to define regret, but this might not that interesting in a sense that there is no learning algorithm that can attain sublinear regret. We can calibrate this definition by introducing new notion of *switch*. That is to say, we are allowed to switch the action at most certain number of times. If we are not allowed to switch, then we get $\mathcal{R}_0$, and if there is no restriction, then we get $\mathcal{R}_1$. If we constrain the number of switches far less than T, then it is possible to have sublinear regret. For simplicity, however, we will use $\mathcal{R} = \mathcal{R}_0$ in this lecture.

1.2 Failure of Deterministic Strategy

The following lemma states that in non-stochastic setting, a deterministic algorithm cannot achieve meaningful regret.

Lemma 1. *If $\mathcal{L}$ is deterministic, i.e., $A_t = a_t$ is a fixed function of $\mathcal{H}_t := (a_1, r_{a_1,1}, \cdots, a_{t-1}, r_{a_{t-1},t-1})$, then there exists a matrix $(r_{a,t})_{a\in\mathcal{A},t\in[T]}$ such that $\mathcal{R}(\mathcal{L}, (r_{a,t})_{a\in\mathcal{A},t\in[T]}) \geq T/2$.*

Proof. Consider binary case when $K = 2$.

If $a_1 = 1$, set $r_{1,1} = 0, r_{2,1} = 1$. Otherwise, set $r_{1,1} = 1, r_{2,1} = 0$.

This will determine a_2 and again set the reward for a_2 equal to 0 and the other reward to be 1.

Repeat this until we complete the entire matrix. Then by construction,

$$\sum_t r_{a_t,t} = 0$$

Meanwhile, since each column has exactly one entry of value 1, the sum of all elements equals to T. Then by pigeon-hole principle,

$$\max(\sum_t r_{1,t}, \sum_t r_{2,t}) \geq T/2$$

This completes the proof for the binary case. For the general case, construct the matrix in a similar fashion so that our action gets 0 reward while the others have 1. In this way, we can prove the regret is greater than $\frac{K-1}{K}T$. □

1.3 Randomized Algorithm

In this section, we will introduce a randomized algorithm called EXP3 (**exp**onential weights for **exp**loration and **exp**loitation). From now on, we will assume that we suffer losses $l_{a,t}$, instead of getting rewards. i.e., $l_{a,t} = 1 - r_{a,t}$, and $\mathcal{R}(\mathcal{L}, (l_{a,t})_{a\in\mathcal{A},t\in[T]}) = \sum_t l_{A_t,t} - \min_{a\in\mathcal{A}} \sum_t l_{a,t}$.

Algorithm 1 EXP3

Tuning parameter $\eta > 0$
Choose p_1 to be uniform over $\mathcal{A}$
for t=1,2, $\cdots$,T **do**
  Draw A_t according to p_t
  Estimated loss $\tilde{l}_{a,t} := \frac{l_{a,t}}{p_{a,t}} \mathbb{1}(a = A_t)$
  Update cumulative estimated losses: $\tilde{L}_{a,t} := \tilde{L}_{a,t-1} + \tilde{l}_{a,t}$
  Update $p_{a,t+1} \propto \exp(-\eta \tilde{L}_{a,t})$
  Or, equivalently, $p_{a,t+1} = \exp(-\eta \tilde{L}_{a,t}) / \sum_{a^*\in\mathcal{A}} \exp(-\eta \tilde{L}_{a^*,t})$
end for

The algorithm updates the distribution among actions after suffering losses, and decides the next action randomly from this distribution.

1.4 Sublinear Regret of EXP3

The following theorem proves the sublinear regret of EXP3 algorithm.

Theorem 2. *EXP3 has an expected regret of* $\sqrt{2TK\log K}$ *when* $\eta = \sqrt{\frac{2\log K}{TK}}$.

Proof. This proof consists of four steps, and this lecture will introduce the first step and the key idea. The rest of the proof will be completed in the next lecture.

STEP1 There are four useful observations:

1. Given $\mathcal{H}_t$, $p_{a,t}$ and $l_{a,t}$ are not random. Only $\mathbb{1}(a = A_t)$ is random.

$$\begin{aligned}\mathbb{E}\left[\tilde{l}_{a,t}|\mathcal{H}_t\right] &= \mathbb{E}\left[\frac{l_{a,t}}{p_{a,t}}\mathbb{1}(a = A_t)|\mathcal{H}_t\right] \\ &= \sum_{a'\in\mathcal{A}} \frac{l_{a,t}}{p_{a,t}}\mathbb{1}(a = a')p_{a',t} \\ &= \frac{l_{a,t}}{p_{a,t}}p_{a,t} \\ &= l_{a,t}\end{aligned}$$

2. Let $p_t := (p_{1,t}, \cdots, p_{K,t})^T, l_t := (l_{1,t}, \cdots, l_{K,t})^T, \tilde{l}_t := (\tilde{l}_{1,t}, \cdots, \tilde{l}_{K,t})^T$.
   Then we have: $p_t^T\tilde{l}_t = \sum_a p_{a,t}\tilde{l}_{a,t} = l_{A_t,t}$

3. Consider any function is applied component-wisely.
   $p_t^T\tilde{l}_t^2 = \sum_a p_{a,t}\tilde{l}_{a,t}^2 = l_{A_t,t}^2/p_{A_t,t}$

4. $\mathbb{E}\left[\frac{1}{p_{A_t,t}}|\mathcal{H}_t\right] = \sum_a \frac{1}{p_{a,t}}p_{a,t} = K$

Now consider a regret against some fixed action $b \in \mathcal{A}$ (later we will take maximum over b to get the real regret).

$$\sum_t l_{A_t,t} - \sum_t l_{b,t} = \sum_t p_t^T\tilde{l}_t - \sum_t \mathbb{E}\left[\tilde{l}_{b,t}|\mathcal{H}_t\right] \quad (\because \text{observation 1 and 2})$$

KEY IDEA

Every function is calculated component-wisely.

$$p_t^T\tilde{l}_t = \frac{1}{\eta}[\log p_t^T \exp(-\eta\tilde{l}_t) + \eta p_t^T\tilde{l}_t] - \frac{1}{\eta}\log p_t^T \exp(-\eta\tilde{l}_t)$$

□

References

[ACBFS02] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

[BCB+12] Sébastien Bubeck, Nicolò Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.