6.853 Algorithmic Game Theory and Data Science February 15, 2017

Lecture 3

Lecturer: Vasilis Syrgkanis *Scribe: Vasilis Syrgkanis*

Last time we examined two player zero-sum games, where two players are deciding on taking one among several available actions. The payoff/loss of each player was a known function of the actions of the two players, and in particular given by a payoff or loss matrix. We saw the notion of a Nash equilibrium and its proof of existence for zero-sum games via von-Neumman's minimax theorem.

In this lecture we will address two concerns of equilibrium theory: first how do player's converge to an equilibrium in a game, second what happens if they do not know the payoff function or for more general games, the payoff of their opponent. We will address these two concerns by looking at the theory of learning in sequential decision making scenarios against adversaries, in particular online learning theory. In this lecture we will analyze a stylized example of an online learning theory scenario were a decision maker is optimizing over two possible actions every day. In the next lecture we will look at a general online learning setting and see how online learning applies to game theoretic scenarios and in particular to zero-sum games.

1 Online Learning Example

Consider the following scenario: you just moved to Boston in the 90's and you found a place to stay in Cambridge, but your work is in downtown Boston. Every day you are trying to decide whether to take the Longfellow or the Harvard bridge to cross to the other side during rush hour. Your ignorance of Boston's traffic and the absence of any high tech gadgets live you with only one option: learning by doing. You want to devise an algorithm that decides which bridge to take every day based on the knowledge that you acquire after every day, such that you manage to avoid the traffic jams. Moreover, given the uncertainty of Boston's traffic situation you don't want to make minimal assumptions on how traffic jams are created in Boston. Your goal: at the end of the year, looking back at the traffic on the bridges, you don't want to regret not having picked one of the two bridges all the time.

We will formulate a stylized version of this problem that captures the essence of the learning task that you are faced with. We consider a setting where a learner needs to decide among two actions $\{H, L\}$ on each of T days. At each day t he picks an action $i_t \in \{H, L\}$ based on what he has observed so far. After picking the action the player incurs a loss $\ell_t^{i_t} \in [0, 1]$ and observes the loss that both actions would have incurred had he chosen them, i.e. he observes the vector $\ell_t = (\ell_t^1, \ell_t^2)$. The goal of the learner is at the end of the T iterations, no matter what the sequences of loss vectors occurred, to have incurred an average loss that is closed to the loss of any fixed action in hindsight: i.e. for all $\ell_1, \ldots, \ell_T$

$$\frac{1}{T}\sum_{t=1}^{T} \ell_t^{i_t} \leq \min_{i \in \{H,L\}} \frac{1}{T}\sum_{t=1}^{T} \ell_t^i + \epsilon(T) \tag{1}$$

for some error term $\epsilon(T)$ that goes to zero as $T \to \infty$.

More formally, we will define the regret of the algorithm as the difference of the cumulative loss of the algorithm vs the cumulative loss of the best fixed action in hindsight, in the worst-case over loss sequences:

$$\textsc{Regret}(T) = \sup_{\ell_1,\ldots,\ell_T} \left(\sum_{t=1}^{T} \ell_t^{i_t} - \min_{i \in \{H,L\}} \sum_{t=1}^{T} \ell_t^i \right) \tag{2}$$

and we will say that the algorithm is a no-regret algorithm if it's regret grows sub-linearly with T, i.e., $\textsc{Regret}(T) = o(T)$.

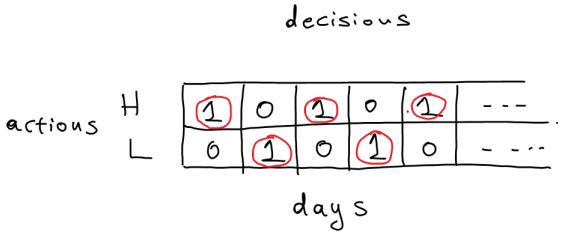


Figure 1: An adversarial loss sequence for Follow-the-Leader

A first attempt. One first thought of how you would attempt to play in this setting is to always play the action that historically performed best, i.e. at time-step t play action (breaking ties in favor of action H):

$$i_t = \operatorname*{argmin}_{i \in \{H,L\}} \sum_{\tau=1}^{t-1} \ell_\tau^i \tag{3}$$

This algorithm is typically referred to in the literature as the Follow-the-Leader (FTL) algorithm, as at every iteration it plays the historical "leader" of the two actions.

However, it is easy to see that such an algorithm cannot perform well in the worst-case over loss sequences. In particular, consider the loss sequence of the form depicted in Figure 1. In this sequence, one each day a different action is the optimal. The algorithm keeps playing the optimal action in hindsight but is always one step behind. Thus it always incurs a loss of 1 on each day, leading to a cumulative loss of T. However, by picking a fixed action he incurs a loss of $T/2$. Thus the difference of the algorithm's average loss and the average loss of the best fixed action is a constant $1/2$ that does not go to zero over time.

There are two caveats with the FTL algorithm:

1. Its choice of an action every day is deterministic given the past.
2. Its choice of action is very unstable from one day to the other.

In fact it is easy to see that any algorithm which given the history of losses, decides deterministically what to play in the next iteration, cannot have sub-linear regret.

Lemma 1. *Any deterministic learning algorithm has to have linear worst-case regret.*

Proof. Consider any such algorithm and consider the loss sequence where on every day the loss of the action picked by the algorithm is 1, while the loss of the alternative action is 0. Then in the end the algorithm incurs a loss of T, while the best of the two actions has a loss of at most $T/2$. Hence, the regret of the algorithm is at least $T/2$. □

This example shows that if we restrict to deterministic algorithms, then the task we set out to achieve is doomed to fail. However, randomization still leaves a glimmer of hope and as we will see in the next section there exist good randomized algorithms that satisfy the no-regret condition.

Moreover, we will see that randomized algorithms that fix the stability issue of the FTL algorithm are no-regret algorithms. In fact, an interesting observation is that even without randomization, if we restrict to sequences of losses $\ell_1, \ldots, \ell_T$, such that the best action in hindsight does not change too often (i.e. the number of time-steps where the best action in hindsight changes from time-step t to time-step $t+1$ is at most $o(T)$), then the FTL algorithm will have $o(T)$ regret. However, we will see that randomization allows us to achieve stability even without any assumption on the sequence of losses.

2 Randomized Algorithms and Expected Regret

Suppose now the learner picks every day action H with some probability p_t and action L with probability $1 - p_t$. Then the algorithm incurs an expected loss of:

$$f(p_t; \ell_t) = p_t \ell_t^H + (1 - p_t)\ell_t^L \tag{4}$$

We will now be interested in achieving an expected loss that is not much more than the loss of the best action in hindsight. However, crucially, we will take a worst case over sequences of losses that do not depend on the actual realization of the action of the algorithm (even though they can depend on the probability p_t).

More formally, we will measure the performance of the algorithm with respect to its expected regret:

$$\textsc{Expected-Regret}(T) = \sup_{\ell_1,\ldots,\ell_t} \left(\sum_{t=1}^{T} f(p_t;\ell_t) - \min_{p\in[0,1]} \sum_{t=1}^{T} f(p;\ell_t) \right) \tag{5}$$

As we argued in the previous section, the regret of the FTL algorithm was large not only because it was deterministic, but also because it was *unstable*, i.e. it's decision was very sensitive to new observations and the number of times that $i_t \neq i_{t+1}$, can be very large for some input sequences.

3 Stability and Regret

We begin our quest of good no-regret algorithms, by an endeavor that is doomed to fail, but which will shed light at what we need to fix. We will first show that the expected regret of the FTL algorithm is bounded above by a stability quantity $\sum_{t=1}^{T} 1\{i_t \neq i_{t+1}\}$. To make it closer to the randomized algorithms that we will see towards the end of the section, we will first re-define FTL in the probability notation, i.e.:

$$\text{Follow-The-Leader (FTL): } p_t = \operatorname*{argmin}_{p\in[0,1]} \sum_{\tau=1}^{t-1} f(p;\ell_\tau) \tag{6}$$

Moreover, for notational convenience let:

$$F_t(p) = \sum_{\tau=1}^{t} f(p;\ell_\tau), \tag{7}$$

be the cumulative loss of always playing action H with probability p up until and including time-step t. Thus the FTL algorithm plays $p_t = \operatorname{argmin}_{p\in[0,1]} F_{t-1}(p)$. Observe, that because each function $f(p;\ell_\tau)$ is linear in p, hence $F_{t-1}(p)$ is also linear in p, the latter minimization problem will have a solution that is either 0 or 1, yielding back the deterministic FTL algorithm from the previous section. But for now let's stick with the seemingly "randomized" definition of FTL. Under this notation will show our stability vs regret lemma.

Lemma 2 (Stability bounds regret)**.** *The expected regret of the FTL algorithm for any sequence $\ell_1,\ldots,\ell_T$ is at most:*

$$\underbrace{\sum_{t=1}^{T} f(p_t;\ell_t) - \min_{p\in[0,1]} \sum_{t=1}^{T} f(p;\ell_t)}_{\textit{expected regret}} \leq \underbrace{\sum_{t=1}^{T} |p_t - p_{t+1}|}_{\textit{stability}} \tag{8}$$

Proof. We will show this property in a sequence of two lemmas. In the first one we simply observe that the regret of the FTL algorithm is very close to the regret of an almost identical algorithm which is tweaked in the following minimal way: suppose that on each day t we knew the loss vector that was going to arise at that day and picked the probability p that minimizes the cumulative loss up to *and including time-step t*, i.e.:

$$\text{Be-The-Leader (BTL): } p_t^* = \operatorname*{argmin}_{p\in[0,1]} \sum_{\tau=1}^{t} f(p;\ell_\tau) = \operatorname*{argmin}_{p\in[0,1]} F_t(p) \tag{9}$$

Observe that this is an infeasible algorithm as we do not know the loss vector ahead of time. However, it is a useful hypothetical algorithm for analyzing the FTL algorithm.

Then as we shall show in the first lemma, the regret of the FTL algorithm is at most the regret of the BTL algorithm plus the stability term. This should be easy to see, since $p_t^* = p_{t+1}$. Subsequently, we will show the more crucial and subtle argument that the BTL algorithm has non-positive regret. Combining these two observations will give the lemma.

Lemma 2.1 (BTL vs FTL)**.** *The regret of the FTL algorithm is at most the regret of the BTL algorithm plus* $\sum_{t=1}^{T} |p_t - p_{t+1}|$.

Proof. We will show that the loss of the FTL algorithm is at most the loss of the BTL algorithm plus the stability term. The latter will conclude the proof.

$$\begin{aligned}
\sum_{t=1}^{T} f(p_t;\ell_t) &= \sum_{t=1}^{T} f(p_t^*;\ell_t) + \sum_{t=1}^{T} \left(f(p_t;\ell_t) - f(p_t^*;\ell_t)\right) \\
&= \sum_{t=1}^{T} f(p_t^*;\ell_t) + \sum_{t=1}^{T} (p_t - p_t^*)\cdot\left(\ell_t^H - \ell_t^L\right) && \text{(by the definition of } f(p;\ell_t) \text{ in Equation (4))} \\
&\le \sum_{t=1}^{T} f(p_t^*;\ell_t) + \sum_{t=1}^{T} |p_t - p_t^*| && \text{(since losses } \ell_t^i \in [0,1]) \\
&\le \sum_{t=1}^{T} f(p_t^*;\ell_t) + \sum_{t=1}^{T} |p_t - p_{t+1}| && \text{(by the definition of FTL and BTL, } p_t^* = p_{t+1})
\end{aligned}$$

□

Lemma 2.2 (BTL has no regret)**.** *The regret of the BTL algorithm is non-positive.*

Proof. To show this we need to argue that the cumulative loss of BTL is at most the cumulative loss of the best fixed action in hindsight. We first argue this in words and then more formally: Suppose that up till time step t the cumulative loss of BTL was at most the loss of the best action in hindsight up until and including time-step t. Thus it is at most the loss of any action in hindsight up until and including time-step t. Then in the next iteration the algorithm will play the best action in hindsigh up-until and including time-step $t+1$. Hence, the loss of the algorithm can increase by at most the loss incurred by that action at time-step $t+1$. Together with the fact that until that point it had loss at most the loss of that action, yields the lemma.

More formally: First observe that the best loss in hindsight up until and including time-step t is exactly p_t^*. Suppose by induction that:

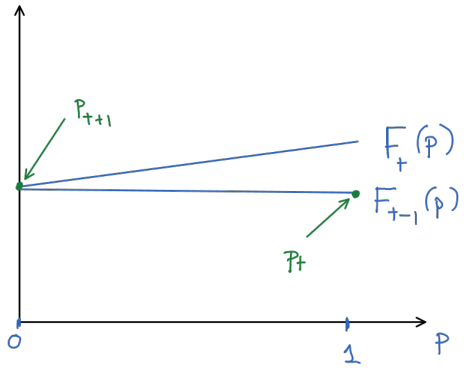
$$\underbrace{\sum_{\tau=1}^{t} f(p_\tau^*;\ell_\tau)}_{\text{loss of BTL until time } t} \le \underbrace{\min_{p\in[0,1]} F_t(p)}_{\text{cumulative loss of best fixed action until time } t} = \underbrace{F_t(p_t^*)}_{\text{by definition of } p_t^*} \tag{10}$$

Then we conclude the induction step that:

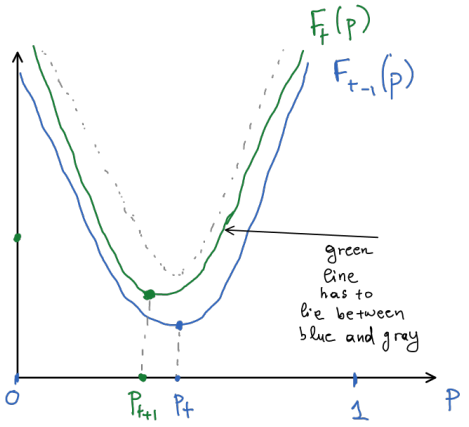
$$\begin{aligned}
\sum_{\tau=1}^{t+1} f(p_\tau^*;\ell_\tau) &= f(p_{t+1}^*;\ell_{t+1}) + \sum_{\tau=1}^{t} f(p_\tau^*;\ell_\tau) \\
&\le f(p_{t+1}^*;\ell_{t+1}) + \min_{p\in[0,1]} F_t(p) && \text{(by induction hypothesis)} \\
&\le f(p_{t+1}^*;\ell_{t+1}) + F_t(p_{t+1}^*) && \text{(since minimum is only better)} \\
&= F_{t+1}(p_{t+1}^*) && \text{(by definition of } F_t(p))
\end{aligned}$$

which concludes the induction step. Finally, the base case of $t = 1$ is satisfied by the definition of p_1^*. □

Combining these two lemmas, yields Lemma 2. □



(a) Two linear functions that are close to each other can have very far minima.



(b) For convex functions, closeness of the functions implies closeness of their minima.

Figure 2

4 Convexity and Stability

Lemma 2 shows that if we could somehow argue that the stability quantity is small, then we would be done. However, we have no control over the stability of FTL in the worst-case over all possible loss sequences as the example in the previous section showed.

The weird thing that leads to instability of FTL is that even though the two functions $F_{t-1}(p)$ and $F_t(p)$, are very close to each other (as they only differ by $f_t(p) \in [0,1]$) their minima (which are the points p_t and p_{t+1}) can differ by a lot! The reason for this pathology, is that the functions $F_t(\cdot)$ are linear functions, and as Figure 2a shows, we can create two linear functions that are very close to each other, but such that their minima are very far apart.

Instead we would have liked to be able to reason that, since $F_t(\cdot)$ is not that different from $F_{t-1}(\cdot)$, then their minima have to be close to each other. Such a property is not unheard off, and in fact each easy to see that such a property holds if the functions $F_t(\cdot)$ where not linear but rather *strictly convex*. As you can see in Figure 2b if two functions $f(\cdot)$ and $g(\cdot)$ are very convex and at the same time they are close to each other, then their minima have to be close![1] More formally:

Lemma 3 (Closeness of minima of convex functions)**.** *Consider two convex functions $f : [0,1] \to \mathbb{R}$ and $g : [0,1] \to \mathbb{R}$, such that $f''(p)$ and $g''(p) \geq \frac{1}{\eta}$ for all p, and such that $h(p) = g(p) - f(p)$ is an L-Lipschitz function, i.e. $|h(p) - h(p')| \leq L \cdot |p - p'|$. Then if $p_f = \operatorname{argmin}_{p\in[0,1]} f(p)$ and $p_g = \operatorname{argmin}_{p\in[0,1]} g(p)$, it must hold that: $|p_f - p_g| \leq \eta \cdot L$.*

Proof. The proof is given pictorially in Figure 3. Below we present the complete proof.

Consider a second order Taylor expansion of f around it's minimum. Then by the mean value theorem, for all $p \in [0,1]$ and for some $\bar{p} \in [0,1]$:

$$\begin{aligned} A = f(p) - f(p_f) = \; & f'(p_f) \cdot (p - p_f) + \frac{f''(\bar{p})}{2} \cdot (p - p_f)^2 \\ \geq \; & \frac{f''(\bar{p})}{2} \cdot (p - p_f)^2 && \text{(since } p_f \text{ is minimizer, } f'(p_f) \cdot (p - p_f) \geq 0) \\ \geq \; & \frac{1}{2\eta} \cdot (p - p_f)^2 && \text{(since } f''(p) \geq \tfrac{1}{\eta}) \end{aligned}$$

Similarly, we can show that $B = g(p) - g(p_g) \geq \frac{1}{2\eta} \cdot (p - p_g)^2$.

[1] In fact you don't really need that the functions are close to each other, rather just that their difference $f(p) - g(p)$ does not vary a lot with p.

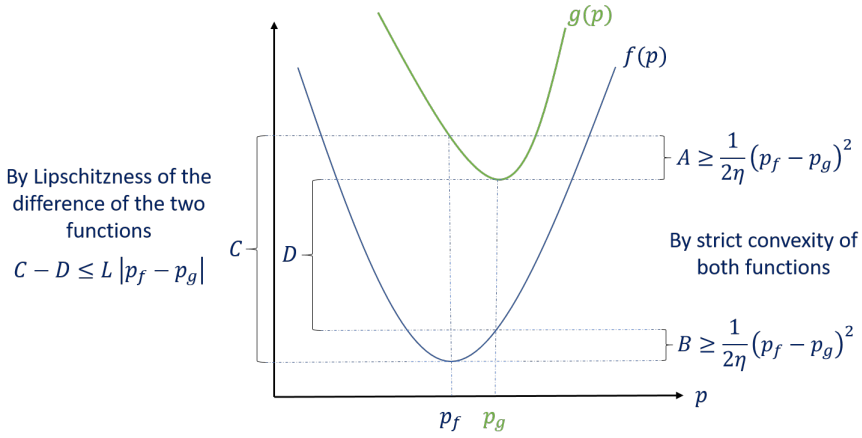


Figure 3: The proof of Lemma 3 follows immediately by noting that $C - D = A + B$ in the above figure, together with the fact that $C - D \le L|p_f - p_g|$ by Lipschitzness of the difference of the two functions and $A + B \ge \frac{1}{\eta}(p_f - p_g)^2$ by the strict convexity of the two functions.

Finally, we can conclude by invoking Lipschitzness of the difference function:

$$\begin{aligned}
L \cdot |p_f - p_g| \ge\ & h(p_f) - h(p_g) && \text{(by lipschitzness of } h\text{)} \\
=\ & \underbrace{g(p_f) - f(p_f)}_{C} - \underbrace{(g(p_g) - f(p_g))}_{D} && \text{(by definition of } h\text{)} \\
=\ & \underbrace{g(p_f) - g(p_g)}_{A} + \underbrace{f(p_g) - f(p_f)}_{B} && \text{(by re-arranging)} \\
\ge\ & \frac{1}{2\eta}(p_f - p_g)^2 + \frac{1}{2\eta}(p_g - p_f)^2 && \text{(by strict convexity of } f(\cdot) \text{ and } g(\cdot)\text{)} \\
=\ & \frac{1}{\eta}(p_f - p_g)^2
\end{aligned}$$

Dividing the above inequality by $|p_f - p_g|$ yields the Lemma. □

5 Follow the Regularized Leader: Convexity through Regularization

So if the cumulative loss functions $F_t(\cdot)$ happened to be strictly convex, i.e. had second derivatives bounded from below by $\frac{1}{\eta}$, then we would be able to upper bound the stability quantity $\sum_{t=1}^{T} |p_t - p_{t+1}|$ by $\eta \cdot T$, since the difference between any two cumulative functions $F_t(p) - F_{t-1}(p) = f_t(p)$ is 1-Lipschitz (from the fact that losses are bounded in $[0, 1]$). If η was a very small number, i.e. inversely proportional to some power of T, then we would get sub-linear regret.

But our cumulative functions are not strictly convex! The solution to this problem is the obvious one: let's make our cumulative functions strictly convex by adding an arbitrary strictly convex function $R(p)$ to them...

In particular, consider a strictly convex function $R(p)$, satisfying $R''(p) \ge 1$ for all $p \in [0, 1]$. There are many functions like that, such as $R(p) = \frac{1}{2}p^2$ or $R(p) = p\log(p) + (1-p)\log(1-p)$ or $R(p) = \log(p) + \log(1-p)$ etc. Then we can alter our cumulative functions to make them strictly convex by modifying them as: $\tilde{F}_t(p) = F_t(p) + \frac{1}{\eta}R(p)$. Then $\tilde{F}_t''(p) \ge \frac{1}{\eta}$, which means that if every day we played according to the minimum of this modified function, then the probabilities p_t and p_{t+1} will be at most η far apart from each other.

We will call this strictly convex function $R(p)$ a *regularizer*, as it "regularizes" our hindsight minimization problem to not "overfit" to the history and hence be unstable from one iteration to the other. We will subsequently call the the algorithm that every day plays according to the p that minimizes the hindsight *regularized cumulative loss*, the *Follow-the-Regularized-Leader*:

$$\text{Follow-the-Regularized-Leader (FTRL): } \tilde{p}_t = \operatorname*{argmin}_{p\in[0,1]} F_{t-1}(p) + \frac{1}{\eta}R(p) = \operatorname*{argmin}_{p\in[0,1]} \tilde{F}_{t-1}(p) \tag{11}$$

Adding the regularization solved the instability problem of the FTL algorithm. However, it introduced some error in our hindsight minimization. In particular, the loss of our algorithm is now not as close to the loss of the BTL algorithm, since $\tilde{p}_{t+1} \neq p_t^*$. However, they are not that far apart. In fact they are at most $2\max_{p\in[0,1]}|R(p)|/\eta$ far apart. The reason is that FTRL is closed to a regularized version of BTL, which in turn has regret at most the above quantity. The reason why the regularized version of BTL has small regret, is that we can view the regularizer as an extra fake loss, that happened at time-step 0. The regularized version of BTL is essentially, BTL on this augmented loss sequence. Hence, we can invoke our Lemma 2.2 on this augmented sequence, to claim that the regularized version of BTL has no regret on this augmented sequence. Combining all these arguments leads to the following bound on the regret of FTRL:

Theorem 1. *The expected regret of FTRL with a regularizer $R(p)$, satisfying $R''(p) \geq 1$ and a parameter η, is upper bounded:*

$$\textsc{Expected-Regret}(T) \leq \frac{2\max_{p\in[0,1]}|R(p)|}{\eta} + \eta \cdot T \tag{12}$$

Proof. We will show this again in two lemmas, which are the "regularized" analogues of Lemma 2.1 and Lemma 2.2. Similar to that proof, let's consider a slight modification of the FTRL algorithm which includes the next iteration's loss vector in the optimization:

$$\text{Be-the-Regularized-Leader (BTRL): } \tilde{p}_t^* = \operatorname*{argmin}_{p\in[0,1]} \tilde{F}_t(p) \tag{13}$$

Lemma 1.1 (BTRL vs FTRL)**.** *The regret of FTRL is at most the regret of BTRL plus $\sum_{t=1}^{T}|\tilde{p}_t - \tilde{p}_{t+1}|$. The latter is upper bounded by $\eta \cdot T$.*

Proof. The first part of the theorem follows along the same lines as the proof of Lemma 2.1, by observing that $\tilde{p}_{t+1} = \tilde{p}_t^*$.

The second part follows by observing that $\tilde{p}_t$ and $\tilde{p}_{t+1}$ are minimizers of $\tilde{F}_{t-1}(\cdot)$ and $\tilde{F}_t(\cdot)$ respectively. The latter are strictly convex functions with second derivatives at least $\frac{1}{\eta}$. Moreover, $\tilde{F}_{t+1}(p) - \tilde{F}_t(p) = f_t(p) = p \cdot \ell_t^H + (1-p)\cdot(1-\ell_t^L)$, which is a 1-Lipschitz function when $\ell_t^i \in [0,1]$. Thus invoking Lemma 3, we get that $|\tilde{p}_t - \tilde{p}_{t+1}| \leq \eta$. □

Lemma 1.2 (BTRL has small regret)**.** *The regret of BTRL is upper bounded by $\frac{2\max_{p\in[0,1]}|R(p)|}{\eta}$.*

Proof. Observe that in the proof of Lemma 2.2 we did not really use any property of the form of our functions $f_t(\cdot)$. Hence, even if our loss functions at each iteration are arbitrary convex functions, that lemma holds. Thus we can think our regularization term as an extra fake loss at time-step 0, i.e.: $f_0(p) = \frac{1}{\eta}R(p)$. Observe that BTRL is equivalent to BTL on this augmented sequence of losses. By invoking Lemma 2.2 we have that BTRL has no regret on this augmented sequence, i.e.:

$$\sum_{t=0}^{T} f_t(\tilde{p}_t^*) \leq \min_{p\in[0,1]} \sum_{t=0}^{T} f_t(p) \leq \max_{p\in[0,1]} f_0(p) + \min_{p\in[0,1]} \sum_{t=1}^{T} f_t(p)$$

where $\tilde{p}_0^* = \operatorname{argmin}_p \frac{1}{\eta}R(p)$. By re-arranging we get:

$$\sum_{t=1}^{T} f_t(\tilde{p}_t^*) - \min_{p\in[0,1]} \sum_{t=1}^{T} f_t(p) \leq \max_{p\in[0,1]} \frac{1}{\eta}R(p) - \min_{p} \frac{1}{\eta}R(p) \leq \max_{p\in[0,1]} \frac{2}{\eta}|R(p)|$$

□

Combining the two lemmas yields the theorem. □

An easy corollary of the latter theorem is that if we pick $\eta = \sqrt{\frac{2 \max_{p\in[0,1]} |R(p)|}{T}}$, then we get regret that grows sub-linearly! Hence, we get a no-regret algorithm.

Corollary 1. *The expected regret of FTRL with* $\eta = \sqrt{\frac{\max_{p\in[0,1]} |R(p)|}{T}}$ *is at most:*

$$\textsc{Expected-Regret}(T) \leq 2\sqrt{2 \max_{p\in[0,1]} |R(p)| \cdot T} \tag{14}$$

Let's now look at one of the many regularizers that satisfy our conditions. Since we are looking at optimizing over probability distributions, a natural regularizer to consider is the negative entropy function:

$$R(p) = p \cdot \log(p) + (1-p) \cdot \log(1-p) \tag{15}$$

The negative entropy function also has a nice intuition as to why it should lead to good stability: in essence the FTRL algorithm with the negative entropy regularizer, is trying to play the best action in hindsight, but also is trying to play with a distribution over actions that has high entropy (i.e. it is not very deterministic). This takes care of the problem of the FTL algorithm that was very deterministic and could be made to incur high regret for some loss sequences. Moreover, since the negative entropy function is a strictly convex, it leads to an algorithm that makes only small modifications towards the action that performs best in hindsight, rather than always jumping around to the best action. For example: if the two actions have almost identical performance, then they will be played with almost equal probability.

In fact for this type of a regularizer we can find in closed form the optimal solution to the problem that FTRL is solving:

$$\begin{aligned}
\tilde{p}_t = & \operatorname*{argmin}_{p\in[0,1]} \; F_{t-1}(p) + \frac{1}{\eta}\left(p\log(p) + (1-p)\log(1-p)\right) \\
= & \operatorname*{argmin}_{p\in[0,1]} \; \sum_{\tau=1}^{t-1} \left(p \cdot (\ell_\tau^H - \ell_\tau^L) + \ell_\tau^L\right) + \frac{1}{\eta}\left(p\log(p) + (1-p)\log(1-p)\right) \\
= & \operatorname*{argmin}_{p\in[0,1]} \; p \cdot \sum_{\tau=1}^{t-1} (\ell_\tau^H - \ell_\tau^L) + \frac{1}{\eta}\left(p\log(p) + (1-p)\log(1-p)\right)
\end{aligned}$$

By taking the first order conditions to the above minimization problem we get the solution:

$$\tilde{p}_t = \frac{\exp\{-\eta \cdot \sum_{\tau=1}^{t-1}(\ell_\tau^H - \ell_\tau^L)\}}{1 + \exp\{-\eta \cdot \sum_{\tau=1}^{t-1}(\ell_\tau^H - \ell_\tau^L)\}} = \frac{\exp\{-\eta \cdot \sum_{\tau=1}^{t-1} \ell_\tau^H\}}{\exp\{-\eta \sum_{\tau=1}^{t-1} \ell_\tau^L\} + \exp\{-\eta \cdot \sum_{\tau=1}^{t-1} \ell_\tau^H\}}$$

A nice way of looking at the latter algorithm, is that at each iteration we keep a weight w_t^i for each action $i \in \{H, L\}$, starting from $w_0^i = \frac{1}{2}$. Then at each iteration we update the weight of each algorithm as:

$$w_{t+1}^i = w_t^i \cdot \exp\{-\eta \cdot \ell_t^i\} \tag{16}$$

and at each iteration we play each action with probability proportional to its weight! Hence, this algorithm gradually penalizes actions that incur higher losses, by dropping their weight faster.

This algorithm is a very well-known algorithm, known under various names, such as *exponential weight updates*, *EXP*, *multiplicative weight updates*, *Hedge*, ..., and has found many applications in many areas of computer science even ones not directly related to online learning.

6 Historical Remarks

The online learning framework dates back to the very early work of Hannan [Han57] on consistency and of Blackwell [B$^+$56] on approachability, which are both terms that are very closely related to achieving the no-regret condition. The exponential weights algorithm dates back to the early and very influential work of Littlestone and Warmuth [LW94] and Freund and Schapire [FS97] and Kivinen and Warmuth [KW97]. The analysis of Hedge, through the Follow-the-Regularized-Leader lens is due to Shalev-Shwartz and Singer [SSS07a, SSS07b]. A very comprehensive survey of Follow-the-Regularized-Leader and related algorithms can be found in [SS12].

References

[B$^+$56] David Blackwell et al. An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6(1):1–8, 1956.

[FS97] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, August 1997.

[Han57] James Hannan. Approximation to bayes risk in repeated play. *Contributions to the Theory of Games*, 3:97–139, 1957.

[KW97] Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Inf. Comput.*, 132(1):1–63, January 1997.

[LW94] N. Littlestone and M.K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212 – 261, 1994.

[SS12] Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012.

[SSS07a] Shai Shalev-Shwartz and Yoram Singer. Online learning: Theory, algorithms, and applications. 2007.

[SSS07b] Shai Shalev-Shwartz and Yoram Singer. A primal-dual perspective of online learning algorithms. *Mach. Learn.*, 69(2-3):115–142, December 2007.