ECE 8803: Online Decision Making in Machine Learning **Fall 2022**

Lecture 9: September 27

Lecturer: Bhuvesh Kumar

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the Instructor.*

9.1. Online linear optimization (OLO)

While the *decision-making using expert advice* (DEA) framework has a very diverse set of applications, its use is limited in certain contexts. For example, it only allows us to choose among a *finite* set of options, while several applications in practice may need us to make decisions from an infinite or continuous set. This motivates the much more general *online optimization* paradigm. We now look at an explicit example of this paradigm, which is also motivated by finance.

9.1.1 An example: Buying/selling stocks

A common problem in finance applications is to decide which stocks to buy, and which to sell, over time. The intuition is roughly as follows: if a stock is likely to *increase* in value in the future, then we would like to buy more of it now so that we can later sell it at a profit. On the other hand, if a stock likely to *decrease* in value in the future, then we would like to sell it now rather than later to avoid a loss. Our overall goal, of course, is to "play the market" in order to maximize long-run profit.

We can model this problem as an online optimization problem. Suppose we are playing the stock market for T days, and considering d stocks that fluctuate in value over time. Each morning when the market opens, we go over this list of stocks, and choose to either buy or sell some amount of each one. This decision is represented by a vector $\boldsymbol{w}_t \in [-D, D]^d$, whose definitely we now unpack. Here, t indexes the day, and $w_{t,i}$ represents the amount corresponding to stock i. If $w_{t,i}$ is positive, then we have chosen to buy stock; if negative, we have chosen to sell or "short" stock. For example, if $w_{t,i} = 3$, we decided to buy 3 shares of stock on the morning of day t; if $w_{t,i} = -2$, we decided to short 2 shares of stock on the morning of day t. Finally, D represents our limit on overall buying power on each day: we cannot buy/short unlimited amounts of stock. Concretely, we will assume a limited buying power of the form[1] $\sqrt{\sum_{i=1}^d w_{t,i}^2} \leq D$.

We make this decision $\boldsymbol{w}_t$ on the morning of day t. At the end of day t, we sell whatever stock we bought, and we buy back the stock that we shorted. We make or lose

1. Note that there are several ways in which we could restrict our overall buying power: we could also constrain the buying power of each stock more directly (i.e. $|w_{t,i}| \leq D$), or the buying power of the sums (i.e. $\sum_{i=1}^t |w_{t,i}| \leq D$). All of these constraints will turn out to yield highly similar answers to the problem. We choose the ℓ_2-norm constraint for relative simplicity.

Loss incurred	Value increased by $\ell_{t,i}$	Value decreased by $\ell_{t,i}$
Buy k units	$-k\ell_{t,i}$	$k\ell_{t,i}$
Short k units	$k\ell_{t,i}$	$-k\ell_{t,i}$

money depending on how much the value of the stock changed over the course of the day. Qualitatively, there are four cases:

- If we bought stock and its value decreases, then we make a loss. For example, if we bought k units of stock i and the value of the stock decreases by ℓ_i, we make a *loss* of $k\ell_{t,i}$ on that stock.
- If we bought stock and its value increases, then we make a profit. For example, if we bought k units of stock i and the value of the stock increases by $\ell_{t,i}$, we make a *profit* of $k\ell_{t,i}$ on that stock; thus a loss of $-k\ell_{t,i}$.
- If we shorted stock and its value decreases, then we make a profit. For example, if we shorted k units of stock i and the value of the stock decreases by $\ell_{t,i}$, we make a *profit* of $k\ell_{t,i}$ on that stock; thus a loss of $-k\ell_{t,i}$.
- If we shorted stock and its value increases, then we make a loss. For example, if we shorted k units of stock i and the value of the stock increases by $\ell_{t,i}$, we make a *loss* of $k\ell_{t,i}$ on that stock.

These four cases are represented pictorially in Table 9.1.1. It is then easy to verify that we can write a *loss function* in terms of the overall decision vector $\boldsymbol{w}_t$: we write $\boldsymbol{\ell}_t := \begin{bmatrix} \ell_{t,1} & \dots & \ell_{t,d} \end{bmatrix}$ as the vector that represents how much each stock depreciated, and the overall loss incurred on that day will be represented as $\langle \boldsymbol{w}_t, \boldsymbol{\ell}_t \rangle$. From the above discussion, note that the elements of $\boldsymbol{\ell}_t$ can also be either positive or negative. We will also assume them to be bounded, i.e. $||\boldsymbol{\ell}_t||_2 \leq G$; this simply represents the fact that the overall aggregate value of the stocks cannot astronomically crash or rise[2].

Our formulation now becomes an *online linear optimization* formulation. In other words, we wish to design decisions $\{\boldsymbol{w}_t\}_{t=1}^T$ to minimize the total loss, given by

$$H_T := \sum_{t=1}^{T} \langle \boldsymbol{w}_t, \boldsymbol{\ell}_t \rangle. \tag{9.1}$$

We wish to do this subject to the constraint that $|w_{t,i}| \leq D$, i.e. $||\boldsymbol{w}_t||_\infty \leq D$. Thus, we can write our decision set $\mathcal{B}$ as the ℓ_∞ ball. (It will turn out that the methodology that we develop for this problem extends more generally to any compact decision set.)

Finally, just like in the case of sequence prediction, this is a setting in which we may not be able to make any assumptions on the evolution of the loss vectors $\{\ell_t\}_{t\geq 1}$. This is actually a very realistic concern in the case of financial applications, as the stock market is

2. Like for the constraints on decisions, we could model this boundedness through an alternative set of constraints, such as $||\boldsymbol{\ell}_t||_\infty \leq G$ or $||\boldsymbol{\ell}_t||_1 \leq G$. These will turn out to yield similar answers to the one provided here.

in fact highly unpredictable. Accordingly, our goal is to minimize regret with respect to the best *fixed* decision on stock trading in hindsight, i.e.

$$R_T := H_T - \min_{\boldsymbol{w} \in \mathcal{B}} \sum_{t=1}^{T} \langle \boldsymbol{w}, \boldsymbol{\ell}_t \rangle \tag{9.2}$$

Think about what the best fixed decision is doing: essentially, it gets to see the *overall* evolution of the value of each stock, and take a decision accordingly. So if stock i overall increased in value, it would be better to keep buying and selling it. On the other hand, if stock i overall decreased in value, it would be better to keep shorting and buying it.

Just like in online sequence prediction, our goal is to obtain sublinear in T regret with respect to this benchmark. We will now review basic intuition and design an algorithm that achieves this.

9.1.2 The equivalent of FTL and why it doesn't work

We can now define the natural equivalent of FTL, and provide an intuitive example to show why it doesn't work. Let's begin by introducing a general version of the Follow the Leader (FTL) algorithm without making the linearity assumption on the loss functions.

Definition 1 *The Follow-the-Leader algorithm in online optimization simply chooses.*

$$\boldsymbol{w}_t := \arg\min_{\boldsymbol{w} \in \mathcal{B}} \left[L_{t-1}(\boldsymbol{w}) \right] := \sum_{s=1}^{t-1} \ell_s(\boldsymbol{w})$$

at every round t.

In case of linear optimization with linear losses, i.e, $\ell_s(\boldsymbol{w}) = \langle \boldsymbol{w}, \boldsymbol{\ell}_s \rangle$, the FTL iterate in round t is given by $\boldsymbol{w}_t = \arg\min_{\boldsymbol{w} \in \mathcal{B}} \sum_{s=1}^{t-1} \langle \boldsymbol{w}, \boldsymbol{\ell}_s \rangle$. Let us look at our finance example for the case $d = 1$ (i.e. only one stock) to get a closer look at what FTL is doing. Recall that our buying/selling power every day is D units, and ℓ_t denotes the depreciation in the stock value on day t. We can then verify that FTL will make the following binary decision:

- If the stock overall depreciates over time, i.e. $\sum_{s=1}^{t-1} \ell_t > 0$, then the FTL algorithm will decide to sell *all* the stock, i.e. decide $w_t = -D$.
- If the stock increases in value over time, i.e. $\sum_{s=1}^{t-1} \ell_t < 0$, then the FTL algorithm will decide to buy *all* the stock, i.e. decide $w_t = D$.

There are situations in which FTL can indeed be verified to do well. Suppose, for example, that the depreciation of the stock followed the following pattern:

$$\begin{bmatrix} -5 & 10 & -5 & 10 & \dots & 10 \end{bmatrix}. \tag{9.3}$$

Then, while the stock value is fluctuating, it overall drifts in the direction of increase over time. It is clear that our best bet should be to buy (and sell at a profit) all the stock, and this is exactly what FTL does. This is a situation in which FTL can be verified to incur sublinear regret, and is somewhat analogous to the Bernoulli sequences that we studied in previous lectures.

On the other hand, the stock's overall amount of depreciation/increase could fluctuate around 0 in the following way:

$$\begin{bmatrix}10 & -10 & 10 & -10 & \dots & -10\end{bmatrix}, \tag{9.4}$$

and this is a situation in which FTL would do very poorly. To see this, note that on all *even-numbered* days, FTL would decide to buy D units of stock as the value appears to be increasing until then; however, the stock would then depreciate by 10 points, leading to a loss of $10D$. Overall, FTL will incur a loss of $10D \cdot \frac{T}{2} = 2.5DT$. On the other hand, with the benefit of hindsight, we would see that this stock is heavily fluctuating and we should neither have bought or sold it. The best decision in hindsight would be to neither buy nor sell on each round, which would incur no loss.

Putting these together, we see that FTL would incur a regret exactly equal to $5DT$ in this case. In the above example, the algorithm keeps "overfitting" to the past history: if an expert is a bit better than the others, the algorithm puts all its probability mass on that expert, and the algorithm keeps changing its mind at every step. Interestingly, this is the main reason FTL fails, which is reflected in its regret guarantee shown below.

Lemma 2 *The regret of FTL on a loss sequence* $\{\ell_t(\boldsymbol{w})\}_{t=1}^T$ *is upper bounded by*

$$R_T \leq \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}_{t+1}).$$

Proof This proof is borrowed from Lecture 2 of the notes: `https://courses.cs.washington.edu/courses/cse599s/14sp/scribes/lecture2/scribeNote.pdf`. Recall that the regret of FTL is given by

$$R_T := \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_t) - \min_{\boldsymbol{w}\in\mathcal{B}} \left[\sum_{t=1}^{T} \ell_t(\boldsymbol{w})\right].$$

Thus, it suffices to upper bound

$$R_T(\boldsymbol{w}) := \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_t) - \sum_{t=1}^{T} \ell_t(\boldsymbol{w})$$

for any $\boldsymbol{w} \in \mathcal{B}$. We now use induction to prove that:

$$\sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{t+1}) \leq \sum_{t=1}^{T} \ell_t(\boldsymbol{w}) \tag{9.5}$$

for any $\boldsymbol{w} \in \mathcal{B}$. Clearly, this suffices to prove Lemma 4.

Base case: $T = 1$ Note that $\boldsymbol{w}_2$ minimizes $\ell_1(\cdot)$, and so we get $\ell_1(\boldsymbol{w}_2) \leq \ell_1(\boldsymbol{w})$ for any $\boldsymbol{w} \in \mathcal{B}$.

Inductive step: We assume that Equation (9.5) holds for $T-1$, i.e. we have

$$\sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}_{t+1}) \leq \sum_{t=1}^{T-1} \ell_t(\boldsymbol{w})$$

for any $\boldsymbol{w} \in \mathcal{B}$. We will now use this to show that Equation (9.5) holds. We show the following sequence of equivalent inequalities:

$$\begin{aligned} \sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}_{t+1}) &\leq \sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}) \\ \iff \sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}_{t+1}) + \ell_T(\boldsymbol{w}_{T+1}) &\leq \sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}) + \ell_T(\boldsymbol{w}_{T+1}) \\ \implies \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{t+1}) &\leq \sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}) + \ell_T(\boldsymbol{w}_{T+1}). \end{aligned}$$

Since the last inequality holds for all $\boldsymbol{w}$, we can take $\boldsymbol{w} := \boldsymbol{w}_{T+1}$, giving us

$$\sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{t+1}) \leq \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{T+1}).$$

But clearly, $\boldsymbol{w}_{T+1}$ minimizes $\sum_{t=1}^{T} \ell_t(\boldsymbol{w})$! Thus, we get

$$\sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{t+1}) \leq \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{T+1}) \leq \sum_{t=1}^{T} \ell_t(\boldsymbol{w}),$$

which completes our proof.

■

9.1.3 How to achieve stability: Regularization

The above example demonstrates that the instability issue of FTL that we first observed in binary sequence prediction persists for the case of online optimization! In binary sequence prediction, we saw that adding randomization *stabilizes* the updates and makes them less vulnerable, on average, to unpredictability in the process. How do we achieve stability here? We first provide a mathematical explanation, and then motivate it through the stock market example. Notice, first, that the decision vector $\boldsymbol{w}_t = \mathbf{0}$ is included in the decision set $\mathcal{B}$ (for any norm, not just the ℓ_2-norm). Also observe that regardless of the loss sequence $\{\boldsymbol{\ell}_t\}_{t=1}^{T}$, we would have $\langle \boldsymbol{w}_t, \boldsymbol{\ell}_t \rangle = 0$ for all t. Thus, such a simple decision is extremely stable to perturbations in the loss vectors.

In the context of the stock market example, this represents a very conservative or "low-risk" decision: we simply decide not to play the market! This is in stark constrast to the case of FTL, which goes "all-in" by maximally exploiting the information present in past observations of the stock market. Of course, this kind of low-risk decision-making is optimal in some settings, such as the example of highly fluctuating stock values in Equation (9.4). However, it is also unnecessarily conservative in a more favorable situation where the stock value drifts in one direction or the other, such as in Equation (9.3). In this scenario, we really should be taking advantage of the drift and deciding to buy the stock. Here, purely low-risk decision-making would incur a linear regret, as the best-loss-in-hindsight would be given by $5DT$ (the overall loss incurred by deciding to buy the stock on every day).

9.1.4 The Follow-the-Regularized-Leader algorithm

The discussion above makes it clear that we want to balance high-risk elements (FTL) and low-risk elements in our decision making. We can easily achieve this by adding a level of *regularization* to FTL! Let us describe this algorithm below.

Definition 3 *The Follow-the-Regularized-Leader (FTRL) algorithm chooses*

$$\boldsymbol{w}_t := \arg\min_{\boldsymbol{w}\in\mathcal{B}} \left[L_{t-1}(\boldsymbol{w}) + R(\boldsymbol{w})\right],$$

where $R(\cdot)$ is a regularization function [3].

For example, if we use l_2 norm of the iterate as the regularization, i.e $R(\boldsymbol{w}) = \frac{1}{\eta}||\boldsymbol{w}||_2^2$, FTLR in round t chooses

$$\boldsymbol{w}_t := \arg\min_{\boldsymbol{w}\in\mathcal{B}} \left[L_{t-1}(\boldsymbol{w}) + \frac{1}{\eta}||\boldsymbol{w}||_2^2\right]$$

where $\eta > 0$ is a **learning rate** parameter, analogous to the one that we picked in binary prediction.

Notice that η naturally measures the amount of risk we take: a higher value of η leads to less regularization, and more weight placed on the past observations (therefore, higher risk), while a lower value of η leads to more regularization, and less weight placed on the past observations (therefore, lower risk). Two extreme cases are below:

- If $\eta \to \infty$, FTRL reduces to the highest-risk option, FTL.
- If $\eta \to 0$, FTRL reduces to the lowest-risk option of picking the most stable vector $\boldsymbol{w}_t = \boldsymbol{0}$.

Adding the regularization terms ensures that the iterate doesn't change as often leads to the following regret bound

Lemma 4 *For any sequence of loss functions $\{\ell_t(\boldsymbol{w})\}_{t=1}^T$ and regularizer $R(\boldsymbol{w})$, the regret of FTRL is upper bounded by*

$$R_T \le \sum_{t=1}^{T} [\ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}_{t+1})] + R(\boldsymbol{w}^*) - R(\boldsymbol{w}_1).$$

where $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}\in\mathcal{B}} \sum_{t=1}^T \ell_t(\boldsymbol{w})$ and $R_T := \sum_{t=1}^T \ell_t(\boldsymbol{w}_t) - \sum_{t=1}^T \ell_t(\boldsymbol{w}^*)$.

---

3. Note that although we use R as on overloaded notation for regret and regularization, whenever we use R in the context of regret for this lecture, we always subscript with time horizon T, i.e. R_T

Proof Let us run FTRL for T rounds with regularizer $R(\cdot)$ and losses $\ell_1, \ell_2, \cdots, \ell_T$ and the iterates be $\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_T$ where we let $\boldsymbol{w}_1 = \arg\min_{\boldsymbol{w} \in \mathcal{B}} R(\boldsymbol{w})$.

Now consider another instance where we run the Follow the leader algorithm for $T+1$ rounds with sequence of losses $R(\cdot), \ell_1, \ell_2, \cdots, \ell_T$. If we let $\boldsymbol{w}_1$ be the first iterate, then we can see that the $T+1$ iterates chosen by FTL would be exactly $\boldsymbol{w}_1, \boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_T$.

Regret bound of FTL implies that

$$\begin{aligned} R(\boldsymbol{w}_1) + \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_t) - \min_{\boldsymbol{w} \in \mathcal{B}} \left[\sum_{t=1}^{T} \ell_t(\boldsymbol{w}) + R(\boldsymbol{w}_t) \right] &\leq R(\boldsymbol{w}_1) + \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_t) - (R(\boldsymbol{w}_1) + \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{t+1})) \\ &= \sum_{t=1}^{T} (\ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}_{t+1})) \end{aligned}$$

$$\begin{aligned} \implies R(\boldsymbol{w}_1) + \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_t) - (\sum_{t=1}^{T} \ell_t(\boldsymbol{w}^*) + R(\boldsymbol{w}^*) &\leq \sum_{t=1}^{T} (\ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}_{t+1})) \\ \implies R_T &\leq \sum_{t=1}^{T} (\ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}_{t+1})) + R(\boldsymbol{w}^*) - R(\boldsymbol{w}_1) \end{aligned}$$

■

In the next lecture we will see how this regret guarantee of FTRL leads to a $O(\sqrt{T}))$ regret when the regularizer is the l_2 norm.

9.2. Additional references

- The proof structure that we have used for OLO is inspired in part by Lecture 12 of the following course: `https://lucatrevisan.github.io/40391/lecture12.pdf`. A much more general version of this statement holds (for arbitrary regularizers and constraints on decisions), and is stated and proved in Theorem 5.2 of Hazan (2016). This is advanced reading (assumes convex analysis background), but worthwhile if you are interested.

- Multiplicative weights can actually be written in this FTRL framework! HW 2, Problem 1 explores this formally. Moreover, FTPL with various noise distributions can also be written as various instances of FTRL: see the book chapter `http://dept.stat.lsa.umich.edu/~tewaria/research/abernethy16perturbation.pdf` for more details on this connection.

References

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.