Lecture 13: Matrix multiplicative weights

November 10, 2025

1 Introduction

In this lecture, we will present some classic results in online learning, specifically, the *mulitplicative weights* update, as well as some of its matrix-valued generalizations, and in the next lecture we'll see how these ideas can be used to obtain fast algorithms for problems in robust statistics.

We will be interested in the following online learning setting: there is a set of n experts which we will label with $1, \ldots, n$. Then, we will interact with an "adversary" for T timesteps. We consider the following sequence of interactions: for each $t = 1, \ldots, T$, we perform the following actions, *in order*:

- The algorithm A chooses a distribution $p^{(t)} = (p_1^{(t)}, \ldots, p_n^{(t)})$ over the n experts.
- The adversary then reveals a *gain vector* $g^{(t)} = (\ell_1^{(t)}, \ldots, \ell_n^{(t)})$, where $\ell_i^{(t)} \in [-1, 1]$ for all t, i. We interpret this to mean that expert i gains $\ell_i^{(t)}$ money at time t: so if this is negative, the expert is actually losing money. Crucially, this function *can depend on* $p^{(t)}$ (and indeed, on everything revealed up to this point), in any arbitrary way!
- The algorithm achieves the expected gain at time t

$$G_{A,t} = \sum_{i=1}^{n} p_i^{(t)} g_i^{(t)} = \langle p^{(t)}, g^{(t)} \rangle \; .$$

At first glance, it may seem surprising that anything is possible in this setting, given the strength of the adversary. But it turns out that there are in fact strong possible guarantees possible: in particular, one can provably compete with the performance of the best expert! Formally, for all $i = 1, \ldots, T$, let expert i's total gain be given by

$$G_i = \sum_{t=1}^{T} g_i^{(t)} \; ,$$

and let $G_A = \sum_{t=1}^{T} G_{A,t}$ and $G_* = \max_{i \in [n]} G_i$. Finally, we define the *regret* of the algorithm to be this difference:

$$\text{Regret}(A) = G_* - G_A \; .$$

We will say that an algorithm has *zero regret* if

$$\frac{\text{Regret}(A)}{T} \to 0 \quad \text{as} \quad T \to \infty \; .$$

Example 1.1. A natural first attempt at an algorithm is to just always choose the expert that has experienced the smallest loss so far, which corresponds to the strategy where $p_i^{(t)} = (0, 0, \ldots, 1, 0, \ldots, 0)$. However, this "follow-the-leader" strategy, while intuitive, is very bad: indeed, one can easily show that in the worst case, this strategy could end up with regret being $1 - o(1)$. Since regret 1 is the worst possible, this strategy achieves essentially no non-trivial guarantees.

Algorithm 1 The multiplicative weights update (MWU) algorithm

procedure MW(ε)
  Let $w_i^{(1)} = 1$ for all $i = 1, \ldots, n$.
  for $t = 1, \ldots, T$ **do**
    Let the probability we choose expert i in round t be given by

$$p_i^{(t)} = \frac{w_i^{(t)}}{\sum_{j=1}^n w_j^{(t)}} .$$

    When $g^{(t)}$ is revealed, let

$$w_i^{(t+1)} = w_i^{(t)} \cdot \exp\left(\varepsilon g_i^{(t)}\right) .$$

  end for
end procedure

2 The multiplicative weights update

It turns out that there is a very nice strategy which is just a slight tweak of this strategy which achieves zero regret. Intuitively, rather than only following the leader, we should choose a "soft"strategy where we are more likely to choose experts which are correct, and the way we do so is in a multiplicative fashion. Formally, we consider the following update strategy, given in Algorithm 1. Intuitively, the idea is that whenever an expert makes a mistake, we should downweight their significance, so over time our weights should converge to only being over experts which have been historically good. Indeed, this is what we will show in a second.

Historical note The multiplicative weights update and variants thereof have been discovered and rediscovered many times throughout history. See [1] for a more complete history. The exact version we're covering today is often known as the *hedge* algorithm [2], but it is (in large part) completely interchangeable with any of the other multiplicative weights updates.

2.1 Analysis of MWU

Before we begin the analysis, we note that as a reuslt of the specific form of the update we use, there's a nice form for the weights we end up using. Namely, by linearity, we actually have that

$$w_i^{(t)} = \exp\left(\varepsilon \cdot \sum_{k=1}^{t-1} g_i^{(k)}\right) = \exp\left(\varepsilon G_i\right) .$$

Our main guarantee is the following:

Theorem 2.1. *For $\varepsilon \leq 1/2$, the matrix multiplicative weights algorithm satisfies:*

$$G_* \leq G_A + \varepsilon T + \frac{\log n}{\varepsilon} .$$

Therefore, if we set $\varepsilon = \sqrt{T^{-1} \log n}$, we obtain that

$$\text{Regret}(A) \leq 2\sqrt{\frac{\log n}{T}} ,$$

which in particular is vanishing as $T \to \infty$. Thus this algorithm achives zero regret.

In fact, we will show something slightly stronger: in particular, we can obtain a bound that is in the form of a "local" norm, in the following sense: let

$$\lambda^{(t)} = \|g^{(t)}\|_\infty \cdot \langle p^{(t)}, |g^{(t)}| \rangle \ ,$$

where for a vector v, we let $|v|$ denote the vector of entrywise absolute values of v. Then, we will show that for any gain vectors g (not just bounded ones), we have:

$$G_* \leq G_A + \varepsilon \sum_{i=1}^{T} \lambda^{(t)} + \frac{\log n}{\varepsilon} \ .$$

Note that if $g_i^{(t)} \in [-1, 1]$, then $\lambda^{(t)} \in [0, 1]$ for all $t = 1, \ldots, T$. Thus, this bound is strictly better than the bound in the theorem. Throughout this proof, we will often require the following standard estimates:

Fact 2.2. *For all x, it holds that $1 + x \leq e^x$. Moreover, for all x satisfying $|x| \leq 0.6$, it holds that $e^x \leq 1 + x + x^2$.*

Proof of Theorem 2.1. The key will be to analyze the following potential function:

$$\Phi^{(t)} = \sum_{i=1}^{n} w_i^{(t)} \ .$$

First, we prove an upper bound on Φ. We note that for all t:

$$\begin{aligned}
\Phi^{(t+1)} &= \sum_{i=1}^{n} w_i^{(t+1)} \\
&= \sum_{i=1}^{n} w_i^{(t)} \cdot \exp\left(\varepsilon g_i^{(t)}\right) \\
&\leq \sum_{i=1}^{n} w_i^{(t)} \cdot \left(1 + \varepsilon g_i^{(t)} + \varepsilon^2 \left(g_i^{(t)}\right)^2\right) && \text{(by Fact 2.2)} \\
&= \Phi^{(t)} \sum_{i=1}^{n} p_i^{(t)} \cdot \left(1 + \varepsilon g_i^{(t)} + \varepsilon^2 \left(g_i^{(t)}\right)^2\right) && \text{(since } p^{(t)} = w^{(t)}/\Phi^{(t)}\text{)} \\
&= \Phi^{(t)} \left(1 + \varepsilon G_{A,t} + \varepsilon^2 \lambda^{(t)}\right) \\
&\leq \Phi^{(t)} \exp\left(\varepsilon G_{A,t} + \varepsilon^2 \lambda^{(t)}\right) \ . && \text{(by Fact 2.2)}
\end{aligned}$$

In particular, since $\Phi^1 = n$, by induction, we obtain that

$$\Phi^{T+1} \leq n \cdot \exp\left(\varepsilon G_A + \varepsilon^2 \sum_{t=1}^{T} \lambda^{(t)}\right) \ .$$

On the other hand, we have that for all $i = 1, \ldots, n$,

$$\Phi^{T+1} \geq w_i^{(T)} = \exp\left(\varepsilon G_i\right) \ .$$

Combining these two bounds and taking logs, we obtain that

$$\varepsilon G_i \leq \log n + \varepsilon G_A + \varepsilon^2 \sum_{t=1}^{T} \lambda^{(t)} \ ,$$

from which the desired expression follows from rearranging, and taking a supremum over $i = 1, \ldots, n$. ☐

3 Matrix multiplicative weight update

There is a very rich theory of multiplicative weights and many generalizations of it, in particular, via the language of *mirror descent* [3]. One specialization of this turns out to be for a natural matrix-valued analog of this problem. Specifically, consider the following "quantum" analog of the learning from experts problem (it turns out there is a formal sense in which this can be thought of as a quantum analog!): for each $t = 1, \ldots, T$, we perform the following actions:

- The algorithm A chooses a *policy matrix* $P^{(t)}$ over the n experts satisfying $\mathrm{tr}(P^{(t)}) = 1$ and $P^{(t)} \succeq 0$.
- The adversary then reveals a *gain matrix* $G^{(t)}$. Once again, this can depend on everything revealed up to this point. Typically, we will consider the setting where $\|G^{(t)}\|_\infty \leq 1$.
- The algorithm achieves the expected gain at time t

$$G_{A,t} = \langle P^{(t)}, G^{(t)} \rangle \ .$$

As before, let $G_A = \sum_{t=1}^{T} G_{A,T}$. In other words, we've replaced the entrywise constraints on the probability vectors and the gain vectors with entrywise constraints *on the eigenvalues of the corresponding matrices.* The competitive advantage is now naturally phrased in a slightly different way. Previously, an equivalent way to phrase the optimal value of the best expert is the following: let $G = (G_1, \ldots, G_n)$ denote the vector of total gain vectors observed over time. Then, $G_* = \|G\|_\infty$. In the spirit of turning entrywise constraints into spectral ones, we can similarly define $G = \sum_{i=1}^{T} G^{(T)}$, and define the value of the best offline policy to be $G_* = \|G\|_\infty$. Indeed, if we knew G ahead of time, this is the best value our algorithm could achieve (why?). Therefore, we once again define the regret of an algorithm to be

$$R(A) = G_* - G_A \ .$$

For this problem, as before, it turns out there is a good policy, once again using a notion of multiplicative weights. Specifically, for a parameter $\varepsilon > 0$ as before, define

$$P^{(t)} = \frac{\exp\left(\varepsilon \sum_{i=1}^{t} G^{(t)}\right)}{\mathrm{tr}\exp\left(\varepsilon \sum_{i=1}^{t} G^{(t)}\right)} \ . \tag{1}$$

By definition, $P^{(t)}$ is a valid policy matrix. Then, we have the following guarantee (which is much more nontrivial to prove!):

Theorem 3.1 ([4])**.** *Suppose ε is chosen so that $\varepsilon G^{(t)} \preceq I$ for all iterations $t = 1, \ldots, T$. Then,*

$$G_* \leq G_A + \varepsilon \sum_{t=1}^{T} \left\langle P^{(t)}, \left| G^{(t)} \right| \right\rangle \cdot \|G^{(t)}\|_\infty + \frac{\log n}{\varepsilon} \ ,$$

where for a symmetric matrix M, we let $|M|$ denote the matrix with all of its eigenvalues replaced with their absolute values.

Using this guarantee One can think of this guarantee as an *algorithmic certificate* on the spectral norm of G. Typically, this is how such guarantees will be used in practice. That is, we will now play the role of the adversary, and use this guarantee to feed in a sequence of (very adaptively) chosen gain matrices A, so that in the end, by using the multiplicative update, we can guarantee that the resulting matrix is bounded spectrally. In the next lecture, we will see how to do this to obtain very efficient algorithms for robust statistics.

References

[1] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[2] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[3] Arkadij Semenovič Nemirovskij and David Borisovich Yudin. Problem complexity and method efficiency in optimization. 1983.

[4] Zeyuan Allen-Zhu, Zhenyu Liao, and Lorenzo Orecchia. Spectral sparsification and regret minimization beyond matrix multiplicative updates. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 237–245. ACM, 2015.