DS 320 Algorithms for Data Science | **Lecture #23**
Spring 2023 | Prof. Kira Goldner

Multiplicative Weight Update

Online Decision-Making

At each time step $t = 1, 2, \dots, T$:

a decision-maker picks a probability distribution $\mathbf{p}^t$ over her experts or actions $i = 1, \dots, N$

an adversary picks a **loss** vector $\ell^t : A \to [0,1]$

an action i^t is chosen according to the distribution $\mathbf{p}^t$, and the decision-maker receives loss ℓ_i^t

the decision-maker learns ℓ^t, the entire **loss** vector

The input arrives "one piece at a time."

What should we compare to?

- We can't compare to the best action sequence $\sum_{i=1}^{T} \min_{i=1}^{N} \ell_i^t$ or we may get no approximation.
- Instead, we compare to the best fixed action $\min_{i=1}^{N} \sum_{i=1}^{T} \ell_i^t$.
- Specifically, our goal is to minimize *regret*. For fixed loss vectors $\ell^1, \dots, \ell^T$. The regret of the action sequence $a^1, \dots, a^T$ is

$$\underbrace{\sum_{t=1}^{T} \ell^t(a^t)}_{\text{our algorithm}} - \underbrace{\min_{i=1}^{N} \sum_{t=1}^{T} \ell_i^t}_{\text{best fixed action}} \quad . \tag{1}$$

Example 1 (Randomization Is Necessary for No Regret)**.** Fix a deterministic online decision-making algorithm. At each time step t, the algorithm commits to a single action a_t. The obvious strategy for the adversary is to set the loss of action at to 1, and the loss of every other action to 0. Then, the cumulative loss of the algorithm is 1 while the cumulative loss of the best action in hindsight is at least $T(1 - \frac{1}{n})$. Even when there are only 2 actions, for arbitrarily large T, the worst-case regret of the algorithm is at least $\frac{T}{2}$.

For randomized algorithms, the next example limits the rate at which regret can vanish as the time horizon T grows.

Example 2 ($\sqrt{(\ln n)/T}$ Regret Lower Bound)**.** Suppose there are $n = 2$ actions, and that we choose each loss vector ℓ^t independently and equally likely to be $(0, 1)$ or $(1, 0)$. No matter how smart or dumb an online decision-making algorithm is, with respect to this random choice of loss vectors, its expected loss at each time step is exactly $\frac{1}{2}$ and its expected cumulative loss is thus $\frac{T}{2}$. The expected cumulative loss of the best fixed action in hindsight is $b\sqrt{T}$, where b is some constant independent of T. This follows from the fact that if a fair coin is flipped T times, then the expected number of heads is $\frac{T}{2}$ and the standard deviation is $\frac{1}{2}\sqrt{T}$.

Fix an online decision-making algorithm $\mathcal{A}$. A random choice of loss vectors causes $\mathcal{A}$ to experience expected regret at least $b\sqrt{T}$, where the expectation is over both the random choice of loss vectors and the action realizations. At least one choice of loss vectors induces an adversary that causes $\mathcal{A}$ to have expected regret at least $b\sqrt{T}$, where the expectation is over the action realizations.

A similar argument shows that, with n actions, the expected regret of an online decision-making algorithm cannot grow more slowly than $b\sqrt{T \ln n}$, where $b > 0$ is some constant independent of n and T.

The Multiplicative Weights Algorithm

Multiplicative Weights (MW) Algorithm

```
initialize weights $w_i^1 = 1$ for every expert $i = 1 \ldots, N$
**for** each time step $t = 1, 2, \ldots, T$ **do**
    let $W^t = \sum_{i=1}^{N} w_k^t$ be the sum of the weights
    choose expert $k$ with probability $p_k^t = w_k^t / W^t$
    for each expert $k$, update weights
        $w_k^{t+1} = w_k^t \cdot (1 - \varepsilon\, \ell_k^t)$
```

Formally, we want an algorithm that works in the following framework:

1. In rounds $1, \ldots, T$, the algorithm chooses some expert i^t.

2. Each expert i experiences a loss $\ell_i^t \in [0, 1]$. The algorithm experiences the loss of the expert it chooses: $\ell_A^t = \ell_{i^t}^t$.

3. The total loss of expert i is $L_i^T = \sum_{t=1}^{T} \ell_i^t$, and the total loss of the algorithm is $\mathbb{E}_{\mathbf{p}}\left[L_{MWU}\right] = \mathbb{E}_{\mathbf{p}}\left[\sum_{t=1}^{T} \ell_{i^t}^t\right]$. The goal of the algorithm is to obtain loss not much worse than that of the best expert: $\min_i L_i^T$.

Theorem 1. *For any sequence of losses, over the randomness of our algorithmic choices* $\mathbf{p}$*,*

$$\mathbb{E}_{\mathbf{p}}[\textsc{Regret}_{\mathrm{MWU}}] \leq 2\sqrt{\ln(N)T} + \varepsilon T.$$

That is, for any expert k

$$\frac{1}{T}\mathbb{E}_{\mathbf{p}}\left[\sum_{t=1}^{T}\ell_{i^t}^t\right] \le \frac{1}{T}\left[\sum_{t=1}^{T}\ell_k^t\right] + \varepsilon + \frac{\ln(N)}{\varepsilon \cdot T}$$

In particular, by setting $\varepsilon = \sqrt{\frac{\ln(N)}{T}}$ *we get:*

$$\frac{1}{T}\mathbb{E}_{\mathbf{p}}\left[\sum_{t=1}^{T}\ell_{i^t}^t\right] - \frac{1}{T}\left[\min_{i=1}\sum_{t=1}^{T}\ell_i^t\right] \le \varepsilon + 2\sqrt{\frac{\ln(N)}{T}}.$$

In other words, the average loss of the algorithm quickly approaches the average loss of the best expert exactly, at a rate of $1/\sqrt{T}$. Note that this works against an arbitrary sequence of losses, which might be chosen adaptively by an adversary. This is pretty incredible. And it will be the source of the power of this framework in applications: we (the algorithm designer) can play the role of the adversary to get the results that we want.

Corollary 2. *There is an online decision-making algorithm that, for every adversary and $\varepsilon > 0$, has expected time-averaged regret*[1] *at most ε after at most $(4\ln n)/\varepsilon^2$ time steps.*

Recap of notation:

- N: the number of experts (actions)
- i, k: index of a specific expert (action)
- w: weights assigned to experts, a vector for each expert, indexed for each time step t and expert i
- W^t: the sum over all experts of weights at time t—$W^t = \sum_{i=1}^{N} w_i^t$.
- p: a probability distribution over experts, indexed for each time step t and expert i, equal to weights w normalized by the sum W—$p_i^t = w_i^t/W^t$.
- ε: update parameter
- ℓ: adversary's loss assignments for each time step and expert, $\ell_i^t \in [0, 1]$.
- F: *expected* loss. $F^t = \sum_{i=1}^{N} p_i^t \ell_i^t$.

Proof. Let F^t denote the expected loss of the MWU algorithm at time t. By linearity of expectation, we have $\mathbb{E}[L_{MWU}^T] = \sum_{t=1}^{T} F^t$. We also know that:

$$F^t = \sum_{i=1}^{N} p_i^t \cdot \ell_i^t = \sum_{i=1}^{N} \frac{w_i^t}{W^t} \cdot \ell_i^t. \tag{2}$$

Thus we want to lower bound the sum of the F^t's.

[1]Time-averaged regret just means the regret, divided by T.

How does W^t change between rounds? We know that $W^1 = N$, and looking at the algorithm, we derive W^{t+1} as a function of W^t and the expected loss (2)

$$\begin{aligned} W^{t+1} &= \sum_{i=1}^{N} w_i^{t+1} \\ &= \sum_{i=1}^{N} w_i^t \cdot (1 - \varepsilon \ell_i^t) \\ &= W^t (1 - \varepsilon F^t). \end{aligned}$$

So by induction, we can write:

$$W^{T+1} = \underbrace{W^1}_{=N} \prod_{t=1}^{T} (1 - \varepsilon F^t).$$

Taking the log, and using the fact that $\ln(1-x) \le -x$, we can write:

$$\begin{aligned} \ln(W^{T+1}) &= \ln(N) + \sum_{t=1}^{T} \ln(1 - \varepsilon F^t) && \ln(ab) = \ln(a) + \ln(b) \\ &\le \ln(N) - \varepsilon \sum_{t=1}^{T} F^t && \text{let } \varepsilon F^t = x \\ &= \ln(N) - \varepsilon \mathbb{E}[L_{MWU}^T] \end{aligned}$$

Similarly, we can unroll the update rule for our weights

$$w_k^{t+1} = w_k^t \cdot (1 - \varepsilon\, \ell_k^t) \qquad \Longrightarrow \qquad w_k^{T+1} := \underbrace{w_i^1}_{=1} \prod_{t=1}^{T} (1 - \varepsilon \ell_i^t) \tag{3}$$

(using the fact that $\ln(1-x) \ge -x - x^2$ for $0 < x < \frac{1}{2}$), we know that for every expert k:

$$\begin{aligned} \ln(W^{T+1}) &\ge \ln(w_k^{T+1}) && W^{T+1} = \sum_{i=1}^{N} w_i^{T+1} \ge w_k^{T+1} \\ &= \sum_{t=1}^{T} \ln(1 - \varepsilon \ell_k^t) && \text{by (3)} \\ &\ge \sum_{t=1}^{T} \varepsilon \ell_k^t - \sum_{t=1}^{T} (\varepsilon \ell_k^t)^2 && \text{let } \varepsilon \ell_k^t = x \\ &\ge -\varepsilon L_k^T - \varepsilon^2 T \end{aligned}$$

Combining these two bounds, we get that for all k:

$$\ln(N) - \varepsilon \mathbb{E}[L_{MWU}^T] \ge -\varepsilon L_k^T - \varepsilon^2 T$$

for all k. Dividing by ε and rearranging, we get:

$$\mathbb{E}[L_{MWU}^T] \le \min_k L_k^T + \varepsilon T + \frac{\ln(N)}{\varepsilon}.$$

□