Optimization for Machine Learning 50.579

Instructor: Ioannis Panageas Scribed by: Foo Lin Geng, Iosif Sakos, Ryann Sim

Lecture 3. Online Optimization and Learning.

1 Multiplicative Weights Update

1.1 The Simplified Expert's Game

Definition 1.1 (The Simplified Expert's Game) *For each day $t = 1, \ldots, T$, you have to choose between alternatives A, B (e.g., rain or not rain).*

1) *Choose A or B according to some rule.*
2) *One of the alternatives is realized.*
3) *If you choose correctly you are not penalized otherwise you lose one point.*
4) *Imagine that there are n experts who on each day t, recommend either A or B.*

Our objective is to perform close to the best expert! We define the following algorithm:

Algorithm 1: Weighted Majority

```
Initialize w_i^0 = 1 for all i ∈ [n].
for t = 1 ... T do
    if ∑_{i choose A} w_i^{t-1} ≥ ∑_{i choose B} w_i^{t-1} then
        choose A, otherwise B.
    end
    for expert i that made a mistake do
        w_i^t = (1 − ε) w_i^{t-1}.
    end
    for expert i that did not make any mistakes do
        w_i^t = w_i^{t-1}.
    end
end
```

Remarks: The above algorithm performs almost as well as the 'best' expert, which can also be interpreted as the best "choice" in hindsight. ϵ is the step-size (a small positive number), which can be chosen later.

Theorem 1.1 (Weighted Majority) *Let M_T, M_T^B be the total number of mistakes the algorithm and best expert make until step T. It holds that*

$$M_T < 2\left[(1+\epsilon)M_T^B + \frac{\log(n)}{\epsilon}\right] \tag{1}$$

Proof: Let M_t^i be the total number of mistakes of the i-th expert at time t. We define

$$\Delta M_t^i := \begin{cases} M_t^i - M_{t-1}^i, & \text{if } t > 0 \\ 0, & \text{otherwise} \end{cases}$$

Notice that ΔM_t^i is a indicator of whether the i-th expert made a mistake at time t, and as such $\Delta M_t^i \in \{0, 1\}$ for all $t \geq 1$. Using this notation we can define $\mathcal{M}_t := \{i : \Delta M_t^i = 1\}$ as the set of all experts that did a mistake at time t. We are going to refer to the equivalent notions in regards to our own mistakes with M_t and ΔM_t, respectively.

We can then describe the algorithm's steps succinctly as follows:

$$w_i^t = \begin{cases} (1 - \Delta M_t^i \epsilon) w_i^{t-1}, & \text{if } t > 0 \\ 1, & \text{if } t = 0 \end{cases} \tag{2}$$

and

$$\Delta M_t = \begin{cases} 1, & \text{if } \sum_{i \in \mathcal{M}_t} w_i^{t-1} \geq \sum_{i \notin \mathcal{M}_t} w_i^{t-1} \text{ and } t \geq 1 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Notice that by Equation 2 we have:

$$0 < w_i^t = (1 - \Delta M_t^i \epsilon) w_i^{t-1} < w_i^{t-1} < \ldots < w_0 < 1, \text{ for all } t \geq 0$$

Now, we are going to define the potential function $\Phi_t := \sum_{i=1}^n w_i^t$, and observe that:

i) $\Phi_0 = \sum_{i=1}^n w_i^0 = \sum_{i=1}^n 1 = n$, and

ii) $\Phi_t = \sum_{i=1}^n w_i^t < \sum_{i=1}^n w_i^{t-1} = \Phi_{t-1}$ for all $t > 0$.

Suppose $\Delta M_t = 1$ for some $t \geq 1$. Then by Equation 3 we also have:

$$\begin{aligned} \sum_{i \in \mathcal{M}_t} w_i^{t-1} \geq \sum_{i \notin \mathcal{M}_t} w_i^{t-1} &\implies 2 \sum_{i \in \mathcal{M}_t} w_i^{t-1} \geq \sum_{i=1}^n w_i^{t-1} = \Phi_{t-1} \\ &\implies \sum_{i \in \mathcal{M}_t} w_i^{t-1} \geq \frac{\Phi_{t-1}}{2} \end{aligned}$$

and, hence

$$\Phi_t = \sum_{i=1}^n w_i^t = \sum_{i=1}^n (1 - \Delta M_t^i \epsilon) w_i^{t-1} = \Phi_{t-1} - \epsilon \sum_{i \in \mathcal{M}_t} w_i^{t-1} \leq \Phi_{t-1} - \frac{\epsilon}{2}\Phi_{t-1} = (1 - \frac{\epsilon}{2})\Phi_{t-1}$$

By combining the above we finally have the following relationship about Φ_t:

$$\Phi_t \leq \Delta M_t(1-\frac{\epsilon}{2})\Phi_{t-1} + (1-\Delta M_t)\Phi_{t-1} = (1-\Delta M_t\frac{\epsilon}{2})\Phi_{t-1} \text{ for all } t > 0$$

Which at time T yields the following product form:

$$\begin{aligned}
\Phi_T &\leq \prod_{t=1}^{T}(1-\Delta M_t\frac{\epsilon}{2})\Phi_0 \\
&= (1-\frac{\epsilon}{2})^{\sum_{t=1}^{T}\Delta M_t} n \\
&= (1-\frac{\epsilon}{2})^{\sum_{t=1}^{T}(M_t - M_{t-1})} n \\
&= (1-\frac{\epsilon}{2})^{M_T - M_0} n \\
&= (1-\frac{\epsilon}{2})^{M_T} n
\end{aligned}$$

Let $B \in \{1, \ldots, n\}$ be the best expert at time T. Then, it also holds:

$$\begin{aligned}
\Phi_T &= \sum_{i=1}^{n} w_i^T \\
&> w_B^T \\
&= (1-\epsilon\Delta M_T^B) w_i^{T-1} \\
&\vdots \\
&= \prod_{t=1}^{T}(1-\Delta M_t^B \epsilon) w_i^0 \\
&= (1-\epsilon)^{\sum_{t=1}^{T} M_t^B} \\
&= (1-\epsilon)^{M_T^B}
\end{aligned}$$

And, thus, we have:

$$(1-\frac{\epsilon}{2})^{M_T} n \geq \Phi_T > (1-\epsilon)^{M_T^B} \implies M_T \log(1-\frac{\epsilon}{2}) + \log(n) > M_T^B \log(1-\epsilon) \tag{4}$$

To complete the proof we are going to rely on the following claim:

Claim 1.2 *The following inequality holds if $x \in [0,\, 0.684]$:*

$$-x - x^2 < \log(1-x) < -x$$

Proof: We first consider the the roots of the equation $-x - x^2 = \log(1-x)$. We find that the inequality $-x - x^2 < \log(1-x)$ holds when $x \in [0,\, 0.684]$. We then take the Taylor's expansion of the log term, which is $-x - \frac{x^2}{2} - \frac{x^3}{3} - \ldots$. The right inequality holds due to the non-negativity of $\frac{x^2}{2} + \frac{x^3}{3} + \frac{x^4}{4} \ldots$.

Using the above claim, and for sufficiently small ϵ, Inequality 4 yields:

$$-\frac{\epsilon}{2}M_T + \log(n) > M_T \log(1 - \frac{\epsilon}{2}) + \log(n) > M_T^B \log(1 - \epsilon) > (-\epsilon - \epsilon^2)M_T^B$$
$$\implies M_T < 2\left[(1+\epsilon)M_T^B + \frac{\log(n)}{\epsilon}\right]$$

■

1.2 The Randomized Experts Game

Definition 1.2 (The Randomized Expert's Game) *For each day $t = 1, \dots, T$, you have to choose between alternatives A, B (e.g., rain or not rain).*

1) Choose A or B with some probability.

2) One of the alternatives is realized.

3) If you choose correctly you are not penalized otherwise you lose one point.

4) Imagine that there are n experts who on each day t, recommend either A or B.

The 'right' objective now is to perform close to the best expert — in expectation. The Randomized Weighted Majority algorithm is defined in Algorithm 2.

Algorithm 2: Randomized Weighted Majority

```
Initialize w_i^0 = 1 for all i ∈ [n].
for t = 1 ... T do
    choose expert's i recommendation with probability proportional to w_i^{t-1}.
    for expert i that made a mistake do
        w_i^t = (1 - ε) w_i^{t-1}.
    end
    for expert i that did not make any mistakes do
        w_i^t = w_i^{t-1}.
    end
end
```

Remarks: This algorithm is also known as Multiplicative Weights Update. Similar to the simplified version, the "randomized" algorithm performs almost as well as the 'best' expert (fewest mistakes). The only difference is that We now choose action i with probability $p_i^t = \frac{w_i^{t-1}}{\sum_j w_j^{t-1}}$. This random choice of actions allows us to improve our bounds by a little.

Theorem 1.3 (Randomized Weighted Majority) *Let M_T, M_T^B be the total number of mistakes the algorithm and best expert make until step T, respectively. It holds that*

$$\mathbb{E}[M_T] < (1+\epsilon)M_T^B + \frac{\log(n)}{\epsilon} \tag{5}$$

Proof: The proof is similar to that of Theorem 1.1 and, hence, we are going to rely on the same notation. To cope with algorithm's probabilistic nature, we consider ΔM_t to be random variables, such as:

$$\Delta M_t = \Delta M_t^i \text{ w.p. } p_i^t$$

Notice, since $\Delta M_t \coloneqq M_t - M_{t-1},\ t > 0$, it follows that M_t are considered to be random variables as well; however, note that $\Delta M_t^i,\ i \in \{1, \ldots, n\}$ are still known variables. Furthermore, by definition, we have:

$$p_i^t = \frac{w_i^{t-1}}{\sum_{j=1}^n w_j^{t-1}} = \frac{w_i^{t-1}}{\Phi_{t-1}}$$

The RHS of Inequality 4 holds by the same reasoning; thus, we are going to concern ourselves with the LHS. Specifically, we are going to prove that $\Phi_T < e^{\epsilon \mathbb{E}[M_T]} n$ which will conclude the proof, since for sufficiently small ϵ we have:

$$e^{\epsilon \mathbb{E}[M_T]} n > (1-\epsilon)^{M_T^B} > (-\epsilon - \epsilon^2) M_T^B \implies \mathbb{E}[M_T] < (1+\epsilon)M_T^B + \frac{\log(n)}{\epsilon}$$

By definition we have:

$$\begin{aligned}
\Phi_t &= \sum_{i=1}^n w_i^t \\
&= \sum_{i=1}^n (1 - \Delta M_t^i \epsilon) w_i^{t-1} \\
&= \sum_{i=1}^n (1 - \Delta M_t^i \epsilon) \Phi_{t-1} p_i^t \\
&= (1 - \epsilon \sum_{i=1}^n \Delta M_t^i p_i^t) \Phi_{t-1} \\
&= (1 - \epsilon \mathbb{E}[\Delta M_t]) \Phi_{t-1}
\end{aligned}$$

By Claim 1.2 we then have that for sufficiently small ϵ:

$$\log(1 - \epsilon \mathbb{E}[\Delta M_t]) < -\epsilon \mathbb{E}[\Delta M_t] \implies \Phi_t < e^{-\epsilon \mathbb{E}[\Delta M_t]} \Phi_{t-1} \text{ for all } t > 0$$

Which at time T yields:

$$\begin{aligned}
\Phi_T &< e^{-\epsilon\mathbb{E}[\Delta M_T]}\Phi_{T-1} \\
&\vdots \\
&< \prod_{t=1}^{T} e^{-\epsilon\mathbb{E}[\Delta M_t]}\Phi_0 \\
&= \exp\left(-\epsilon\sum_{t=1}^{T}\mathbb{E}[\Delta M_t]\right)n \\
&= \exp\left(-\epsilon\mathbb{E}\left[\sum_{t=1}^{T}\Delta M_t\right]\right)n \\
&= \exp\left(-\epsilon\mathbb{E}\left[\sum_{t=1}^{T}(M_t - M_{t-1})\right]\right)n \\
&= e^{-\epsilon\mathbb{E}[M_T - M_0]}n \\
&= e^{-\epsilon\mathbb{E}[M_T]}n
\end{aligned}$$

■

1.3 The General Setting

Definition 1.3 (The General Setting) *At each time step $t = 1 \ldots T$.*

1) ***Player*** *chooses $x_t \in \mathcal{K} \subset \mathbb{R}^n$ (some closed convex set).*
2) ***Adversary*** *chooses $\ell_t \in \mathcal{F}$ (set of convex functions).*
3) ***Player*** *suffers loss $\ell_t(x_t)$ and observes feedback.*

The Player's goal is to minimize the (time average) **Regret**, that is:

$$\frac{1}{T}\left[\sum_{t=1}^{T}\ell(x_t) - \min_{u\in\mathcal{K}}\sum_{t=1}^{T}\ell(u)\right] \tag{6}$$

If Regret $\to 0$ as $T \to \infty$, the algorithm is called **no-regret**.

1.3.1 Convex Optimization as Special Case

Observe that if the adversary chooses a single $\ell \in \mathcal{F}$ function such that $\ell_1 = \cdots = \ell_T = \ell$ then Regret can be reduced to:

$$\begin{aligned}
\frac{1}{T}\left[\sum_{t=1}^{T} \ell_t(x_t) - \min_{u\in\mathcal{K}} \sum_{t=1}^{T} \ell_t(u)\right] &= \frac{1}{T}\left[\sum_{t=1}^{T} \ell(x_t) - \min_{u\in\mathcal{K}} \sum_{t=1}^{T} \ell(u)\right] \\
&= \frac{1}{T}\left[\sum_{t=1}^{T} \ell(x_t) - \sum_{t=1}^{T} \min_{u\in\mathcal{K}} \ell(u)\right] \\
&= \frac{1}{T}\sum_{t=1}^{T} \ell(x_t) - \frac{1}{T}\sum_{t=1}^{T} \ell(x^*) \\
&= \frac{1}{T}\sum_{t=1}^{T} \ell(x_t) - \ell(x^*)
\end{aligned}$$

Where $x^* \coloneqq \min_{u\in\mathcal{K}} \ell(u)$ is the global minimizer of ℓ—which is well-defined, since ℓ is a convex function. Since $\mathcal{K}$ is convex we also have that $\frac{1}{T}\sum_{t=1}^{T} x_t \in \mathcal{K}$ Hence, by Jensen's Inequality, it follows that:

$$\frac{1}{T}\sum_{t=1}^{T} \ell(x_t) - \ell(x^*) \geq \ell\left(\frac{1}{T}\sum_{t=1}^{T} x_t\right) - \ell(x^*) \geq 0, \text{ (Since } x^* \text{ is a global minimizer of } \ell)$$

Assuming $x_t,\ t \in \{0, \ldots, T\}$ are generated from a no-regret algorithm it follows that Regret $\to 0$ as $T \to \infty$; which implies that:

$$\begin{aligned}
0 = \lim_{T\to\infty}\left[\frac{1}{T}\sum_{t=1}^{T} \ell(x_t) - \ell(x^*)\right] &\geq \lim_{T\to\infty}\left[\ell\left(\frac{1}{T}\sum_{t=1}^{T} x_t\right) - \ell(x^*)\right] \geq 0 \\
\implies \lim_{T\to\infty} \ell\left(\frac{1}{T}\sum_{t=1}^{T} x_t\right) = \ell(x^*)
\end{aligned} \tag{7}$$

Thus, the problem of optimizing a convex function ℓ can be reduced to a regret-minimization problem (if we are using a no-regret algorithm).

1.3.2 Regret for Expert's Problem

We now go back to the Expert's Problem as defined earlier. What can we say about the Regret of our previous algorithms? The goal here is once again to minimize the time average Regret, which in this case is:

$$\frac{\mathbb{E}[M_T] - \#\text{mistakes best expert}}{T} \tag{8}$$

Explanation: We choose x_t as the probability distribution at time t over the experts and ℓ_t is their probability of making a mistake. Recall that we have:

$$\mathbb{E}[M_T] \leq (1+\epsilon)M_T^B + \frac{\log(n)}{\epsilon} \tag{9}$$

By choosing $\epsilon = \sqrt{\frac{\log(n)}{T}}$, we get average regret $2\sqrt{\frac{\log(n)}{T}}$. Can we do better? Nope. This is because of the probabilistic argument below (we argue that we can do no better, because we decide our move (A or B) in hindsight).

Lemma 1.4 *Consider just two experts that choose one A and B respectively at all times. The adversary chooses uniformly at random A or B. The expected number of mistakes of an online algorithm is $\frac{T}{2}$. One of the two fixed strategies will have, with a fixed high probability: $\frac{T}{2} - \Theta(\sqrt{T})$.*

Proof: Let T be the number of times that the experts make choices, and let $A(T)$ be the number of times A is chosen. The probability of an expert choosing A is $p = \frac{1}{2}$. We also note that the expected number of A chosen, $E[A(T)] = pT$.

To start, we use Hoeffding's inequality to bound the probability of obtaining extreme values. For some $\epsilon > 0$,

$$\mathbf{Pr}(A(T) \leq (p-\epsilon)T) \leq e^{-2\epsilon^2 n}$$

Taking the complement of the set, we get:

$$\mathbf{Pr}(A(T) > (p-\epsilon)T) > 1 - e^{-2\epsilon^2 n}$$

As we want a fixed value for the probability, we set the RHS to 0 by setting $\epsilon = \frac{1}{\sqrt{T}}$.

$$\mathbf{Pr}(A(T) > pT - \sqrt{T}) > 1 - e^{-2}$$

We observe that, with a fixed probability, $A(T)$ is lower-bounded by $(\Theta(pT - \sqrt{T}))$. Since pT is constant, we conclude by saying that with a fixed probability, $A(T) = pT - \Theta(\sqrt{T})$. ■

Remarks: This means that the long run average regret, $\frac{A(T)}{T}$ is equal to $p - \Theta(\frac{1}{\sqrt{T}})$. This scales similarly to the regret of the Randomized Expert's algorithm described above. Hence the expression for the regret found above is the best we can do for the Expert's problem.

2 Online Gradient Descent

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function that is differentiable and L-Lipschitz in some compact convex set $\mathcal{X}$ of diameter D. Online GD is defined:

Algorithm 3: Online Gradient Descent

Initialize at some x_0.
for $t \coloneqq 1$ *to* T **do**
  choose x_t and **observe** $\ell_t(x_t)$.
  $y_t = x_t - \alpha_t \nabla \ell_t(x_t)$.
  $x_{t+1} = \prod_{\mathcal{X}}(y_t)$.
end
Regret: $\frac{1}{T}\left(\sum_{t=1}^{T} \ell_t(x_t) - \min_x \sum_{t=1}^{T} \ell_t(x)\right)$

Theorem 2.1 (Online Gradient Descent) *Let $\ell_t : \mathbb{R}^n \to \mathbb{R},\ t = 1, \ldots, T$ be convex functions that are differentiable and L-Lipschitz in some compact convex set $\mathcal{X}$ of diameter D. It holds that:*

$$\frac{1}{T}\left(\sum_{t=1}^{T} \ell_t(x_t) - \min_{x\in\mathcal{X}} \sum_{t=1}^{T} \ell_t(x)\right) \leq \frac{3}{2}\frac{LD}{\sqrt{T}} \tag{10}$$

by appropriately choosing $\alpha_t = \frac{D}{L\sqrt{t}}$.

Proof: Let $x^* = \arg\min_{x\in\mathcal{X}} \sum_{t=1}^{T} \ell_t(x)$, and note x^* is well-defined for any compact set $\mathcal{X}$. We, now, choose $t \in \{1, \ldots, T\}$. Then, by Online Gradient Descent's definition, we have:

$$x_t = \prod_{\mathcal{X}}(y_t) \implies x_t \in \mathcal{X}$$

Hence, since ℓ_t is differentiable in $\mathcal{X}$ and $\mathcal{X}$ is a convex set, it follows by the First-Order Condition of Convexity that:

$$\ell_t(x^*) \geq \ell_t(x_t) + \nabla \ell_t(x_t)^\intercal (x^* - x_t) \implies \ell_t(x_t) - \ell_t(x^*) \leq \nabla \ell_t(x_t)^\intercal (x_t - x^*)$$

Furthermore, by Online Gradient Descent's definition, we also have that $y_t = x_t - \alpha_t \nabla \ell_t(x_t)$ for $\alpha_t = \frac{D}{L\sqrt{t}} > 0$; thus, by substitution on above, we have:

$$\begin{aligned}
\ell_t(x_t) - \ell_t(x^*) &\leq \frac{1}{\alpha_t}(x_t - y_t)^\intercal (x_t - x^*) \\
&= \frac{1}{2\alpha_t}(\|x_t - y_t\|_2^2 + \|x_t - x^*\|_2^2 - \|y_t - x^*\|_2^2) \\
&= \frac{1}{2\alpha_t}(\|x_t - x^*\|_2^2 - \|y_t - x^*\|_2^2) + \frac{1}{2\alpha_t}\|x_t - (x_t - \alpha_t \nabla \ell_t(x_t))\|_2^2 \\
&= \frac{1}{2\alpha_t}(\|x_t - x^*\|_2^2 - \|y_t - x^*\|_2^2) + \frac{\alpha_t}{2}\|\nabla \ell_t(x_t)\|_2^2
\end{aligned}$$

We should continue by applying the following facts, which can be verified in previous notes:

i) $\|y_t - x^*\|_2 \geq \|\prod_{\mathcal{X}}(y_t) - x^*\|_2$, and

ii) $\|\nabla \ell_t(x_t)\|_2 \leq L$.

Where the former follows by the convexity of $\mathcal{X}$, while the later holds due the Lipschitz property of ℓ_t. Thus, the previous inequality now yields:

$$\begin{aligned}
\ell_t(x^t) - \ell_t(x^*) &\leq \frac{1}{2\alpha_t}(\|x_t - x^*\|_2^2 - \|\textstyle\prod_{\mathcal{X}}(y_t) - x^*\|_2^2) + \frac{\alpha_t L^2}{2} \\
&= \frac{1}{2\alpha_t}(\|x_t - x^*\|_2^2 - \|x_{t+1} - x^*\|_2^2) + \frac{\alpha_t L^2}{2}
\end{aligned}$$

Consequently by taking the sum for $t \in \{1, \ldots, T\}$ we have:

$$\begin{aligned}
&\sum_{t=1}^{T}(\ell_t(x^t) - \ell_t(x^*)) \\
&\leq \sum_{t=1}^{T}\left[\frac{1}{2\alpha_t}(\|x_t - x^*\|_2^2 - \|x_{t+1} - x^*\|_2^2)\right] + \frac{L^2}{2}\sum_{t=1}^{T}\alpha_t \\
&= \frac{1}{2\alpha_1}\|x_1 - x^*\|_2^2 + \sum_{t=2}^{T}\left[\|x_t - x^*\|_2^2\left(\frac{1}{2\alpha_t} - \frac{1}{2\alpha_{t-1}}\right)\right] - \frac{1}{2\alpha_{T+1}}\|x_{T+1} - x^*\|_2^2 + \frac{L^2}{2}\sum_{t=1}^{T}\alpha_t \\
&\leq \frac{1}{2\alpha_1}\|x_1 - x^*\|_2^2 + \sum_{t=2}^{T}\left[\|x_t - x^*\|_2^2\left(\frac{1}{2\alpha_t} - \frac{1}{2\alpha_{t-1}}\right)\right] + \frac{L^2}{2}\sum_{t=1}^{T}\alpha_t
\end{aligned}$$

However, since $\mathcal{X}$ has diameter D, it holds $\|x_t - x^*\|_2 \leq D$ for all $t \in \{1, \ldots, T\}$, and, hence:

$$\begin{aligned}
\sum_{t=1}^{T}(\ell_t(x^t) - \ell_t(x^*)) &\leq \frac{1}{2\alpha_1}D^2 + \sum_{t=2}^{T}\left[D^2\left(\frac{1}{2\alpha_t} - \frac{1}{2\alpha_{t-1}}\right)\right] + \frac{L^2}{2}\sum_{t=1}^{T}\alpha_t \\
&= D^2\left[\frac{1}{2\alpha_1} + \sum_{t=2}^{T}\left(\frac{1}{2\alpha_t} - \frac{1}{2\alpha_{t-1}}\right)\right] + \frac{L^2}{2}\sum_{t=1}^{T}\alpha_t \\
&= D^2\left(\frac{1}{2\alpha_1} + \frac{1}{2\alpha_T} - \frac{1}{2\alpha_1}\right) + \frac{L^2}{2}\sum_{t=1}^{T}\alpha_t \\
&= \frac{D^2}{2\alpha_T} + \frac{L^2}{2}\sum_{t=1}^{T}\alpha_t \\
&= \frac{D^2}{2\frac{D}{L\sqrt{T}}} + \frac{L^2}{2}\sum_{t=1}^{T}\frac{D}{L\sqrt{t}} \\
&= \frac{LD}{2}\left(\sqrt{T} + \sum_{t=1}^{T}\frac{1}{\sqrt{t}}\right)
\end{aligned}$$

To complete the proof we are going to rely on the following Lemma:

Lemma 2.2

$$\sum_{t=1}^{T}\frac{1}{\sqrt{t}} \leq 2\sqrt{T}$$

Proof: We first observe that:

$$\int_{t=0}^{T} \frac{1}{\sqrt{t}} = 2\sqrt{T}$$

We see that $\sum_{t=1}^{T} \frac{1}{\sqrt{t}}$ is just the Riemann sum approximation of the integral $\int_{t=0}^{T} \frac{1}{\sqrt{t}}$ by taking value to be the right of the intervals. Since $\frac{1}{\sqrt{t}}$ is a decreasing function in t, the Riemann sum approximation is a lower bound to the actual value, hence:

$$\sum_{t=1}^{T} \frac{1}{\sqrt{t}} \leq \int_{t=0}^{T} \frac{1}{\sqrt{t}} = 2\sqrt{T}$$

Thus, the above inequality yields:

$$\begin{aligned} \sum_{t=1}^{T} (\ell_t(x^t) - \ell_t(x^*)) \leq \frac{3LD}{2}\sqrt{T} &\implies \sum_{t=1}^{T} \ell_t(x^t) - \sum_{t=1}^{T} \ell_t(x^*) \leq \frac{3LD}{2}\sqrt{T} \\ &\implies \frac{1}{T}\left(\sum_{t=1}^{T} \ell_t(x_t) - \min_{x\in\mathcal{X}} \sum_{t=1}^{T} \ell_t(x)\right) \leq \frac{3}{2}\frac{LD}{\sqrt{T}} \end{aligned}$$

■

Remarks: If we want error ϵ, we need $T = \Theta(\frac{L^2D^2}{\epsilon^2})$ iterations (same as GD for L-Lipschitz).

3 MWUA General Setting

As defined in Algorithm 2, the MWUA algorithm is a generalized version of the Randomized Weighted Majority Algorithm, which is defined in Algorithm 4:

Algorithm 4: MWUA

Initialize $w_i^0 = 1$ for all $i \in [n]$.
for $t = 1 \dots T$ **do**
  choose action i with probability proportional to w_i^{t-1}.
  for *each action* i **do**
    $w_i^t = (1-\epsilon)^{c_i^t} w_i^{t-1}$.
  end
end

Remarks: This is a generalized form of the Randomized Weighted Majority Algorithm, where 2 things are slightly different. Firstly, instead of a cost of 1/0 for correct/incorrect actions, each action i at time t has a cost c_i^t (which is chosen by the adversary). Secondly, we set $\epsilon := \sqrt{\frac{\log(n)}{T}}$.

Theorem 3.1 (MWUA) *Let* $OPT = min_i \sum_{t=1}^{T} c_i^t$

$$\mathbb{E}[cost_{MWUA}] \leq OPT + \epsilon T + \frac{\log n}{\epsilon}$$

Proof: We first define the **potential** function $\phi_t = \sum_i w_i^t$. Let the best action (in hindsight) be i^*. Then, we have:

$$\phi_T > w_{i^*}^T = (1-\epsilon)^{OPT} \tag{11}$$

Now, we look at ϕ_{t+1}:

$$\begin{aligned}
\phi_{t+1} = \sum_i w_i^{t+1} &= \sum_i w_i^t (1-\epsilon)^{c_i^t} \\
&= \sum_i \phi_t p_i^{t+1} (1-\epsilon)^{c_i^t} \\
&= \phi_t \sum_i p_i^{t+1} (1-\epsilon)^{c_i^t} \\
&\leq \phi_t \sum_i p_i^{t+1} (1-\epsilon \cdot c_i^t) \\
&= \phi_t (1-\epsilon \cdot \mathbb{E}[\text{cost}(t)_{MWUA}]) \\
&\leq \phi_t e^{-\epsilon \mathbb{E}[\text{cost}(t)_{MWUA}]}
\end{aligned}$$

Note that for the first inequality, we used the fact that $(1-\epsilon)^x \leq 1-\epsilon x$ for $x \in [0,1]$, $\epsilon \in [0,1/2]$. Then, similarly to previous proofs, we can take the telescopic product of the above inequality to obtain:

$$\phi_T \leq \phi_1 e^{-\epsilon \mathbb{E}[\text{cost}(t)_{MWUA}]} \tag{12}$$

Therefore, by substituting in 11, we obtain:

$$\begin{aligned}
(1-\epsilon)^{OPT} &\leq n e^{-\epsilon \mathbb{E}[\text{cost}(t)_{MWUA}]} \\
OPT \log(1-\epsilon) &\leq \log(n) - \epsilon \mathbb{E}[\text{cost}(t)_{MWUA}]
\end{aligned}$$

By 1.2, we can express the above inequality as:

$$\begin{aligned}
OPT(-\epsilon - \epsilon^2) &\leq \log(n) - \epsilon \mathbb{E}[\text{cost}(t)_{MWUA}] \\
\mathbb{E}[\text{cost}(t)_{MWUA}] &\leq \frac{\log(n)}{\epsilon} + (1+\epsilon) OPT
\end{aligned}$$

Plugging in $\epsilon = \sqrt{\frac{\log(n)}{T}}$ gives

$$\begin{aligned}
\frac{1}{T}\left(\mathbb{E}[\text{cost}(t)_{MWUA}] - OPT\right) &\leq \frac{1}{T}\frac{\log(n)}{\epsilon} + \epsilon \frac{OPT}{T} \\
&= \sqrt{\frac{\log(n)}{T}} + \sqrt{\frac{\log(n)}{T}} \cdot \frac{OPT}{T}
\end{aligned}$$

However, we note that OPT is upper bounded by T. Hence:

$$\frac{1}{T}\left(\mathbb{E}[\text{cost}(t)_{MWUA}] - OPT\right) \leq 2\sqrt{\frac{\log(n)}{T}}$$

$\blacksquare$

3.1 Examples

3.1.1 Solving Linear Programs

Suppose we are given a linear program in the standard form

$$\begin{aligned} Ax &\geq b \\ \text{s.t. } x &\geq 0 \end{aligned}$$

Goal: Check Feasibility. Compute a vector $x^* \geq 0$ such that for some $\epsilon > 0$ we get

$$a_i^\top x^* \geq b_i - \epsilon, \text{ for all } i$$

Oracle Access: Given a vector c and scalar d, does there exist a $x \geq 0$ such that $c^\top x \geq d$? This is a condition that can be used to check for feasibility.

Claim 3.2 *Using the above and binary search, one can solve any linear program using MWUA!*

Setting: Consider every constraint $a_i^\top x - b_i$ as an action.

- Choose $c_i(x) = \frac{a_i^\top x - b_i}{\rho}$ with ρ chosen such that $|c_i| \leq 1$.
- Initialize $w_i^0 = 1$ (uniform distribution).
- For each $t = 1 \ldots T$, ask the oracle if there exists a point $x \geq 0$ such that $c^\top x \geq d$ where

  $$c = \sum_i p_i^t a_i, \quad d = \sum_i p_i^t b_i$$

  .

If the answer is no, the linear problem is infeasible. If the answer is yes (the oracle returns an x^t), each action suffers cost $c_i^t = c_i(x^t)$.

From Theorem 3.1, we obtain the following inequality:

$$0 \leq \frac{1}{T}\sum_t \sum_i p_i^t (a_i^\top x_i^t - b_i) \leq \frac{1}{T}\sum_t \sum_i p_i^* (a_i^\top x_i^t - b_i) + 2\rho\sqrt{\frac{\log(m)}{T}} \tag{13}$$

where p^* is the optimal hindsight. Note that the RHS is at most (for all i)

$$\frac{1}{T}\sum_t \left(a_i^\top x_i^t - b_i\right) + 2\rho\sqrt{\frac{\log(m)}{T}} = \frac{1}{T}a_i^\top \sum_t x_i^t - b_i + 2\rho\sqrt{\frac{\log(m)}{T}} \tag{14}$$

Therefore, by choosing $T = \frac{4\rho^2 \log(m)}{\epsilon^2}$ and $\tilde{x} = \frac{1}{T}\sum_t x^t$ we get that $a_i^\top \tilde{x} - b_i + \epsilon \geq 0$ for all i. In other words, we have found a feasible solution $\tilde{x}$ to the linear program!

3.1.2 MWUA and Zero-Sum Games

Definition 3.1 (Payoff Matrix) *Consider a matrix A (called a payoff matrix). A_{ij} denotes the amount of money player x pays to player y. For example, Rock-Paper-Scissors has the following payoff matrix:*

$$A = \begin{pmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{pmatrix}$$

Definition 3.2 (Nash Equilibrium) *A vector (x^*, y^*) is called a NE if*

$$x^{*\top} A y^* \geq x^{*\top} A \tilde{y} \text{ for all } \tilde{y} \in \Delta \text{ and } x^{*\top} A y^* \leq \tilde{x}^\top A y^* \text{ for all } \tilde{x} \in \Delta$$

To compute the Nash Equilibrium of a game, we can let the players run MWUA against each other!

Algorithm 5: MWUA for Zero-Sum Games

Initialize $p_{i,x}^0 = 1/n$, $p_{i,y}^0 = 1/n$ for all i (both players, uniform)
for t=1... T **do**
  Player x chooses i with probability $p_{i,x}^t$ and y with $p_{i,y}^t$ respectively
  for *each action* i **do**
    $p_{i,x}^t = p_{i,x}^{t-1} \frac{(1-\epsilon)^{(Ap_y^{t-1})_i}}{Z_x}$
    $p_{i,y}^t = p_{i,y}^{t-1} \frac{(1+\epsilon)^{(A^\top p_x^{t-1})_i}}{Z_y}$
  end
end

Theorem 3.3 (MWUA) *Let $\tilde{x} = \frac{1}{T}\sum_t p_x^t$ and $\tilde{y} = \frac{1}{T}\sum_t p_y^t$. Assume that A has entries in $[-1, 1]$ and $T = \Theta\left(\frac{\log n}{\epsilon^2}\right)$. It holds that $(\tilde{x}, \tilde{y})$ is an ϵ-approximate NE that is*

$$\tilde{x}^\top A\tilde{y} \leq x'^\top A\tilde{y} + \epsilon \text{ and } \tilde{x}^\top A\tilde{y} \geq x^\top A y' - \epsilon$$

Proof: Exercise 6 ■

Remark: The above theorem is not true in general for last iterates! Indeed, in standard Matching Pennies game with the payoff matrix shown below, the strategy profile is shown to diverge almost surely to the boundary, as seen in Figure 1.

$$A = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}$$

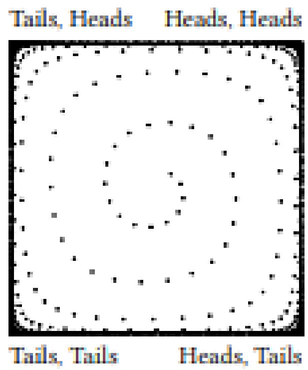

Figure 1: Phase Portrait of Matching Pennies using MWUA

4 General Family of No-Regret Algorithms

4.1 Follow the Leader

Definition 4.1 (Follow The Leader (FTL)) *Let $f_k : \mathbb{R}^n \to \mathbb{R},\ k = 0, \ldots, T$ be convex functions, differentiable in some convex set $\mathcal{K} \subset \mathbb{R}^n$. FTL is defined in Algorithm 6.*

Algorithm 6: Follow the Leader

Initialize at some x_0.
for $t :=$ *to* T **do**
  Choose $x_t = \arg\min_{x \in \mathcal{K}} \sum_{k=0}^{t-1} f_k(x)$
end

Remark 4.2 *FTL is possible to perform poorly under certain conditions. Specifically FTL is not a no-regret algorithm. To see this consider the following example:*

Example 4.1 *Let us consider the case of $\mathcal{K} = \Delta_2$ (the 2-dimensional Simplex $\subset \mathbb{R}^2$) and $f_k(x) =$*

$X^\intercal \ell_k$ *for all* $k \in \{0, \ldots, T\}$, *where* ℓ_k *are chosen as by an adversary as follows:*

$$
\begin{aligned}
\ell_0 &= (0, \frac{1}{2}) \\
\ell_1 &= (1, 0) \\
\ell_2 &= (0, 1) \\
&\vdots \\
\ell_T &= \begin{cases} (1, 0), & \text{if } T \text{ is odd} \\ (0, 1), & \text{otherwise} \end{cases}
\end{aligned}
$$

Then given $x_0 = (\frac{1}{2}, \frac{1}{2})$, *FTL yields the following:*

$$
\begin{aligned}
x_1 &= \arg\min_{x \in \Delta_2} x^\intercal \ell_0 = (1, 0) \\
x_2 &= (0, 1) \\
&\vdots \\
x_T &= \begin{cases} (1, 0), & \text{if } T \text{ is odd} \\ (0, 1), & \text{otherwise} \end{cases}
\end{aligned}
$$

Recall, that the regret's formula:

$$
\frac{1}{T}\left[\sum_{k=0}^{T} f_k(x_k) - \min_{u \in \Delta_2} \sum_{k=0}^{T} f_k(u)\right] = \frac{1}{T}\left[\sum_{k=0}^{T} x_k^\intercal \ell_k - \min_{u \in \Delta_2} \sum_{k=0}^{T} u^\intercal \ell_k\right] \approx \frac{1}{T}\left[T - \frac{T}{2}\right] = \frac{1}{2}
$$

where we used symmetry of $\{\ell_0, \ldots, \ell_T\}$ *to derive that* $\arg\min_{u \in \Delta_2} \sum_{k=0}^{T} u^\intercal \ell_k = (\frac{1}{2}, \frac{1}{2})$. *Notice that Regret* $\to \frac{1}{2}$ *as* $T \to \inf$, *which implies that FTL is not a no-regret algorithm.*

4.2 Follow the Regularized Leader

Definition 4.3 (Follow the Regularized Leader) *Let* $f_k : \mathbb{R}^n \to \mathbb{R}$ *be convex functions for all* k, *differentiable in some convex set* $\mathcal{K}$. *Moreover, let* R *be a strongly convex function. FTRL is defined:*

Algorithm 7: Follow the Regularized Leader

```
1 Initialize at some x_0.
2 for t := to T do
3     Choose x_t = argmin_{x∈K} { ε_{t−1} · Σ_{k=0}^{t−1} f_k(x) + R(x) }
4 end
```

Claim 4.2 *When $R(x) = \frac{1}{2}\|x\|^2$, $f_k(x) = x^\top c_k$ (linear in x), and $\mathcal{K} = \mathcal{R}^n$, we have Online GD where step-size α is fixed.*

Proof: We first substitute the terms into FTRL.

$$x_t = argmin_{x\in\mathcal{K}} \left\{ \epsilon_{t-1} \cdot \sum_{k=0}^{t-1} f_k(x) + R(x) \right\}$$
$$= argmin_{x\in\mathcal{K}} \left\{ \epsilon_{t-1} \cdot \sum_{k=0}^{t-1} x^T c_k + \frac{1}{2}||x||^2 \right\}$$

Then, we take the derivative and set to zero. This will give us a minimum point, because the term is convex.

$$\frac{\partial}{\partial x_i} \left\{ \epsilon_{t-1} \cdot \sum_{k=0}^{t-1} x^T c_k + \frac{1}{2}||x||^2 \right\} = 0$$
$$\epsilon_{t-1} \cdot \sum_{k=0}^{t-1} c_{k,i} + x_i = 0$$
$$x_i = -\epsilon_{t-1} \cdot \sum_{k=0}^{t-1} c_{k,i}$$

Thus, for all i, we choose $x_{t,i} = \prod_{\mathcal{K}} \left\{ -\epsilon_{t-1} \cdot \sum_{k=0}^{t-1} c_{k,i} \right\}$, which is the projection of the minimum point into the set $\mathcal{K}$.

For online GD, we have the stepwise iteration $x_t = \prod_{\mathcal{K}} \{x_{t-1} - \alpha_{t-1}\nabla f_{t-1}(x_{t-1})\}$. Substituting $f_k(x) = x^T c_k$, we get $x_t = \prod_{\mathcal{K}} \{x_{t-1} - \alpha_{t-1}(c_{t-1})\}$.

Taking the telescopic sum, we get $x_t = \prod_{\mathcal{K}} \left\{ x_0 - \sum_{k=0}^{t-1} \alpha_k c_k \right\}$. This is allowed because $\mathcal{K} = \mathcal{R}^n$, and is thus a linear subspace. $\prod_{\mathcal{K}} A + \prod_{\mathcal{K}} B = \prod_{\mathcal{K}}(A + B)$.

If we set $x_{0,i} = 0$ for all i and α_k to some constant ϵ for all k, we get:

$$x_{t,i} = \prod_{\mathcal{K}} \left\{ -\epsilon \cdot \sum_{k=0}^{t-1} c_{k,i} \right\} \tag{15}$$

This is similar to the result obtained from FTRL. ■

Claim 4.3 *When $R(x) = -\sum x_i log x_i$ (entropy) and $f_k(x) = x^\top c_k$ (linear in x), we have MWUA.*

Proof: First, we describe MWUA. For MWUA, the stepwise iteration for each action i is:

$$w_{t,i} = (1 - \delta)^{c_{(t-1),i}} w_{(t-1),i}$$

$$\log w_{t,i} = c_{(t-1),i} \cdot \log(1-\delta) + \log w_{(t-1),i}$$

Taking the telescopic sum and since $w_{0,i} = 1$:

$$\begin{aligned}
\log w_{t,i} &= \log(1-\delta) \cdot \sum_{k=0}^{t-1} c_{k,i} + \log w_{0,i} \\
&= \log(1-\delta) \cdot \sum_{k=0}^{t-1} c_{k,i}
\end{aligned}$$

Next, we substitute the terms into FTRL.

$$\begin{aligned}
x_t &= argmin_{x\in\mathcal{K}} \left\{ \epsilon_{t-1} \cdot \sum_{k=0}^{t-1} f_k(x) + R(x) \right\} \\
&= argmin_{x\in\mathcal{K}} \left\{ \epsilon_{t-1} \cdot \sum_{k=0}^{t-1} x^T c_k - \sum_{j=0}^{n} x_j \log(x_j) \right\}
\end{aligned}$$

Then, we take the derivative and set to zero. Because entropy is a convex function, the root of this derivative will give us a minimum point.

$$\frac{\partial}{\partial x_i} \left\{ \epsilon_{t-1} \cdot \sum_{k=0}^{t-1} x^T c_k - \sum_{j=0}^{n} x_j \log(x_j) \right\} = 0$$

$$\epsilon_{t-1} \cdot \sum_{k=0}^{t-1} c_{k,i} - \log(x_i) - 1 = 0$$

$$\log(e \cdot x_i) = \epsilon_{t-1} \cdot \sum_{k=0}^{t-1} c_{k,i}$$

If we set $\epsilon_{t-1} = \log(1-\delta)$ for all t, we get $\log(e \cdot x_{t,i}) = \log(1-\delta) \cdot \sum_{k=0}^{t-1} c_{k,i}$ which is similar to the MWUA formula, except with a linear scaling of $\frac{1}{e}$.

For instance, if $\sum_{k=0}^{t-1} c_{k,i} = 0$, we observe that no cost has been incurred, and $w_{t,i} = e \cdot x_{t,i} = 1$. If $\sum_{k=0}^{t-1} c_{k,i} = s$, then $w_{t,i} = e \cdot x_{t,i} = s \cdot \log(1-\delta)$

Hence, we conclude that MWUA is similar to FTRL with initial states $x_{0,i} = \frac{1}{e}$ and step sizes $\epsilon_t = \log(1-\delta)$ for all t. ■

References

[1] Dimitris Bertsimas and John N Tsitsiklis. *Introduction to linear optimization*, volume 6. Athena Scientific Belmont, MA, 1997.

[2] Stephen Boyd and Lieven Vandenberghe. *Convex optimization.* Cambridge university press, 2004.

[3] Laurence A Wolsey and George L Nemhauser. *Integer and combinatorial optimization.* John Wiley & Sons, 2014.