# Choosing Experts via Multiplicative Weights Update

April 10, 2023

- We continue the theme of "**how do you optimize when the future is unknown or even adversarial?**"

  1. **Last week: Online algorithms (competitive analysis)**
     - Example: Ski rental, Online set cover.
  2. **Today (and next week): Minimizing regret**
     - Given a fixed set of "strategies" for some thing.
     - Each day, you must choose a strategy but don't know which one is good.
     - **Goal:** perform as good as the best strategy at time goes by.

## 1 Warm up: The 2-Action Setting

- There are $n$ experts $\mathcal{E}$.
- At day $t$, the following happen *in order*:
  1. Every expert $i$ predicts whether a stock price is "up" or "down"
  2. The **algorithm** chooses "up" or "down"
  3. Then, the **adversary** reveals the actual outcome.
     - The actual outcome can depend on our choice today.
- Define **the total loss** $L_A$ as the total number of mistakes the algorithm makes.

### 1.1 Assume Perfect Expert

- Suppose there exists a perfect expert (never wrong).
- Consider this algorithm.
- **The Halving algorithm:**
  - Consider all experts $\mathcal{E}'$ with no mistakes so far.
  - Each round, follow the majority of $\mathcal{E}'$.

- What is the total lost?

**Lemma 1.1.** *If there exists a perfect expert, then the Halving algorithm guarantees*

$$L_A \leq \log n.$$

- Every time we make a mistake, the size of $\mathcal{E}'$ reduces by a factor of 2.
- So we can make at most $\lceil \log_2 n \rceil$ mistakes.

## 1.2 No Perfect Expert

- But what if there is no perfect expert?
- So... we will just compare our loss with the best expert.
- Let $L_\star$ be the total loss of the best expert (i.e. the number of mistakes made by the best expert).
- Consider this algorithm.
- **The Iterated Halving Algorithm**:
  - Divide the time into "epochs"
  - In each epoch, run **the halving algorithm**:
    * Keep track of all experts $\mathcal{E}'$ with no mistake in this epoch.
    * When $\mathcal{E}'$ is empty, start a new epoch.
- Can you bound $L_A$ in term of $L_\star$?

**Lemma 1.2.** *The Iterated Halving Algorithm guarantees*

$$L_A \leq \log(n) \cdot L_\star + \log(n).$$

- **Analysis:**
  - When we start a new epoch, all experts must make at least one mistake.
    * $L_\star \geq \#\text{epochs} - 1$
  - For each epoch, we made at most $\log n$ mistakes.
    * $L_A \leq \log n \cdot \#\text{epochs}$
- How much small can $L_A$ be compared to $L_\star$?

**Exercise 1.3.** Show an algorithm that guarantees $L_A \leq (2+\epsilon)L_\star + O(\frac{\log n}{\epsilon})$.

- So, with small additive factor, you can match the best expert up to the factor of 2 (the number of possible actions).

## 1.3 Lower Bounds

- Can we make even less mistakes?
  - For example, $L_A \leq 1.99 L_\star +$ some small things.
- Let's say there are only 2 experts:
  - one always say "Up".
  - another always down "Down".
- What would you do if you are an adversary?

**Lemma 1.4.** *There exists an adversary that guarantees that* $L_A \geq 2L_\star$.

- Whatever algorithm chooses, the adversary just reveals the opposite outcome.
  - If algorithm chooses "Up", reveal "Down"
  - If algorithm chooses "Down", reveal "Up"
- After $T$ days, $L_A = T$.
- But $L_\star \leq T/2$
  - If we choose "Up" less often, then the Up-expert makes $T/2$ mistakes.

## 1.4 Conclusion on the 2-Action Setting

- Recall the setting
  - There are $n$ experts.
  - On day $t$,
    - each expert $i$ experiences a loss $\ell_i^t \in \{0, 1\}$.
      - $\ell_i^t = 1$ if he makes a mistake
      - $\ell_i^t$ is chosen by the adversary **after the algorithm chose the action.**
    - algorithm's loss is $\ell_A^t \in \{0, 1\}$.
  - Let $L_i = \sum_t \ell_i^t$ denote the total loss of expert $i$. So $L_\star = \min_i L_i$.
  - Let $L_A = \sum_t \ell_A^t$ denote our total loss.
- The conclusion:
  - When there are 2 distinct actions to choose each day ("Up" or "Down"):
  - There is an algorithm with total loss $L_A \leq (2 + \epsilon) L_\star + O(\frac{\log n}{\epsilon})$.
  - There exists an adversary that ensures $L_A \geq 2L_\star$.
- So basically the factor of 2 is tight (when we must choose 1 out of 2 actions)

## 2 The General Setting

- We want to generalize the setting as follows:
  - Each expert has its own independent suggestion.
  - (So there are $n$ distinct actions to choose each day, not just "up" or "down").
  - Each expert can *lose or gain*.
  - Our choice on each day
    - not just "up" or "down".
    - choose which expert to follow
    - can be *randomized*. (We can choose a distribution of experts.)
- More formally, there are $n$ experts and $T$ days.
- At each day $t = 1, 2, \ldots, T$ , the following happen *in order*:
  1. We choose a distribution $p^t = (p_1^t, \ldots, p_n^t)$ of experts
     - meaning that we follow expert $i$ with probability $p_i^t$
  2. Then, the **adversary** reveals the **loss vector** $\ell^t = (\ell_1^t, \ldots, \ell_n^t)$
     - meaning that expert $i$ **losses** money $\ell_i^t \in [-1, 1]$.
       - If $\ell_i^t < 0$, this means that expert $i$ gains money.
     - Important: $\ell^t$ can depend on $p^t$ (and the entire history) in an adversarial way.
  3. Our (expected) loss is $\ell_A^t = \langle p^t, \ell^t \rangle = \sum_i p_i^t \ell_i^t$.
- **Goal:**
  - Our total loss $L_A = \sum_{t=1}^T \ell_A^t$
  - Total loss of the best expert $L_\star = \min_i \sum_{t=1}^T \ell_i^t$.
  - Want $L_A$ as small as $L_\star$.
- What is a natural algorithm?
  - Since we absolutely have no control of the future, maybe we can rely on the past information?
  - **Following the Leader**: Just choose the single best expert so far.
  - Is this a good algorithm?

### 2.1 Choosing One Expert cannot be Good

- Here, we answer the above question negatively, in a very strong way.
- Suppose each day we must choose only 1 expert $i$.
- That is, each day $t$, set $p^t = e_i$ where $e_i$ is the elementary unit vector.

**Lemma 2.1.** *When $\ell_i^t \in [0,1]$ for all $t$ (no gain). There exists an adversary such $L_A \geq T$ and $L_\star \leq T/n$. (Factor $n$ worse!).*

- The proof is easy
  - If the algorithm chooses expert $i$ (i.e. $p^t = e_i$), then the adversary says that only expert $i$ losses. (i.e. $\ell^t = e_i$).
  - So $L_A \geq T$.
  - There must be an expert that we choose at most $T/n$ times.
  - So $L_\star \leq T/n$.

**Lemma 2.2.** *In general ($\ell_i^t \in [-1,1]$), there exists an adversary such that $L_A \geq T$ and $L_\star \leq -(1-\frac{2}{n})T$. (Positive vs. negative!!)*

- Therefore,
  - as long as we deterministically choose an expert, the bound must be bad.
  - So "Follow the Leader" cannot be good.
- To get good bounds, we will be randomized.
  - In other words, choose a distribution of experts.

# 3 The Multiplicative Weights Update algorithm

- It turns out that something very close to "Follow the Leader" is very good.
- The algorithm is called **Multiplicative Weights Update (MWU).**

---

**Algorithm 1** The Multiplicative Weights Update (MWU) algorithm

MWU($\epsilon$):

- $w_i^1 \leftarrow 1 \quad \forall i$
- For $t = 1, \ldots, T$
  - Follows expert $i$ with probability $p_i^t = \frac{w_i^t}{\sum_j w_j^t}$
  - After $\ell^t$ is revealed, $w_i^{t+1} \leftarrow w_i^t \cdot (1 - \epsilon \ell_i^t) \quad \forall i$

---

## 3.1 Superficial Interpretation

- The weight $w_i^t$ of expert $i$ is its "importance".
- We follow experts according to their importance $p_i^t \sim w_i^t$.
- Initially, all experts have the same importance.
- If expert $i$ losses a lot, its importance decreases a lot $w_i^{t+1} \leftarrow w_i^t \cdot \exp(-\epsilon \ell_i^t)$. (It increases when it gains.)
- "$\epsilon$" is how aggressive we decrease the weights.
- This is a "softer" version of Follow-the-leader.
  - If $w_i^t \approx w_j^t$ for all $i, j$, then we choose random expert.
  - If expert $i$ performs much better than everyone ($w_i^t \gg w_j^t$ for all $j$), then we almost surely follow expert $i$.

## 3.2 The Guarantee: No Regret

- We can define **regret:** Regret $= L_A - L_\star$.
  - This is, the cost of not knowing the best expert from the beginning.
- An algorithm as a **no-regret property** if Regret $= o(T)$.
  - That is, $\frac{\text{Regret}}{T} \to 0$ when $T \to \infty$
  - As time goes by, you have no regret on average.

**Theorem 3.1.** *For* $0 < \epsilon \leq 1/2$, $\text{MWU}(\epsilon)$ *guarantees that*

$$L_A \leq L_\star + \epsilon T + \frac{\ln n}{\epsilon}.$$

*Moreover, if* $\ell^t \in [0,1]^n$ *for all* $t$*, then*

$$L_A \leq (1+\epsilon) L_\star + \frac{\ln n}{\epsilon}.$$

- **Take-away**:
  - MWU has no-regret property (by setting $\epsilon = \sqrt{\frac{\ln n}{T}}$, Regret $\leq \sqrt{T \ln n}$).
  - The second bound is stronger ($L_\star \leq T$).
    * When there is no gain, total loss of MWU *does not depend on time.*
    * Depend on the best expert up to a $(1+\epsilon)$-factor and a small $O(\frac{\ln n}{\epsilon})$-additive factor.
- Super simple algorithm, yet super strong guarantee.
  - This looks magical...
  - To see how one can derive this, we need some background in calculus.

### 3.3 Proof

- Let the total weight at round $t$ be the potential function

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

- Plan:

1. Upper bound: $\Phi^{T+1} \leq n \cdot \exp(-\epsilon L_A)$
2. Lower bound: $w_i^{T+1} \geq \exp(-\epsilon L_i - \epsilon^2 T)$.
3. Comparing the two bounds: for all $i$

$$e^{-\epsilon L_i - \epsilon^2 T} \leq \Phi^{T+1} \leq n \cdot e^{-\epsilon L_A}$$

Taking log and implying by $1/\epsilon$:

$$-L_i - \epsilon T \leq \frac{\ln n}{\epsilon} - L_A$$

Rearranging:

$$L_A \leq L_i + \epsilon T + \frac{\ln n}{\epsilon}$$

and so we are done.

- Upper bound: $\Phi^{T+1} \leq n \cdot \exp(-\epsilon L_A)$ because

$$\begin{aligned}
\Phi^{t+1} &= \sum_{i=1}^{n} w_i^{t+1} \\
&= \sum_{i=1}^{n} w_i^t (1 - \epsilon \ell_i^t) \\
&= \Phi^t \cdot \sum_{i=1}^{n} p_i^t (1 - \epsilon \ell_i^t) \\
&= \Phi^t \cdot (1 - \epsilon \langle p^t, \ell^t \rangle) \\
&\leq \Phi^t \cdot \exp(-\epsilon \langle p^t, \ell^t \rangle) \qquad \text{as } 1 - x \leq e^{-x}.
\end{aligned}$$

  - That is, the total weight at round $t$ is decreased by factor of $\exp(\epsilon \langle p^t, \ell^t \rangle)$ where $\langle p^t, \ell^t \rangle$ is our loss,
  - By unfolding

$$\Phi^{T+1} \leq \Phi^1 \exp(-\epsilon \sum_{t=1}^{T} \langle p^t, \ell^t \rangle) = n \cdot \exp(-\epsilon L_A).$$

- Lower bound: $w_i^{T+1} \geq \exp(-\epsilon L_i - \epsilon^2 T)$.

$$\begin{aligned}
w_i^{T+1} &= w_i^T(1-\epsilon\ell_i^T) \\
&= \prod_{t=1}^{T}(1-\epsilon\ell_i^t) \\
&\geq \prod_{t=1}^{T}\exp(-\epsilon\ell_i^t - (\epsilon\ell_i^t)^2) && \text{as } 1-x \geq e^{-x-x^2} \text{ for all } x \leq 0.6 \\
&= \exp(-\epsilon\sum_{t=1}^{T}\ell_i^t - \epsilon^2\sum_{t=1}^{T}(\ell_i^t)^2) \\
&\geq \exp(-\epsilon L_i - \epsilon^2 T) && \text{as } |\ell_i^t| \leq 1
\end{aligned}$$

### 3.4 Proof: a stronger bound when $\ell_i^t \in [0,1]$

Suppose that $\ell_i^t \in [0,1]$ for all $i$ (i.e. only loss money).

$$\begin{aligned}
w_i^{T+1} &= w_i^T(1-\epsilon\ell_i^T) \\
&= \prod_{t=1}^{T}(1-\epsilon\ell_i^t) \\
&\geq \prod_{t=1}^{T}(1-\epsilon)^{\ell_i^t} && \text{as } (1-\epsilon\ell_i^t) \geq (1-\epsilon)^{\ell_i^t} \text{ for } \ell_i^t \in [0,1] \\
&\geq \exp(-\epsilon\sum_{t=1}^{T}\ell_i^t - \epsilon^2\sum_{t=1}^{T}\ell_i^t) && \text{as } 1-x \geq e^{-x-x^2} \text{ for } x \leq 0.6 \\
&\geq \exp(-\epsilon L_i - \epsilon^2 L_i)
\end{aligned}$$

- So
  - We get $w_i^{T+1} \geq \exp(-\epsilon L_i - \epsilon^2 L_i)$
  - instead of $w_i^{T+1} \geq \exp(-\epsilon L_i - \epsilon^2 T)$.
- By comparing upper and lower bounds as before, we can conclude $L_A \leq L_i + \epsilon L_i + \frac{\ln n}{\epsilon}$.

## 4 Analyzing Gain instead of Loss

- Sometimes, it is more convenient to think about Gain instead of Loss.
- Consider the same setting where each day **adversary** reveals the **gain vector** $g^t = (g_1^t, \ldots, g_n^t)$ where $g^t = -\ell^t$.
- Let $G_A = \sum_{t=1}^{T}\langle p^t, g^t\rangle = -L_A$ be our total gain.
- Let $G_\star = \max_i \sum g_i^t = -L_\star$ be the total gain of the best expert.

- We can rewrite the MWU algorithm in term as gain as follows:
  - Replace $w_i^{t+1} \leftarrow w_i^t \cdot (1 - \epsilon \ell_i^t)$ by $w_i^{t+1} \leftarrow w_i^t \cdot (1 + \epsilon g_i^t)$.

---

**Algorithm 2** The Multiplicative Weights Update (MWU) algorithm

MWU($\epsilon$):

- $w_i^1 \leftarrow 1 \quad \forall i$
- For $t = 1, \ldots, T$
  - Follows expert $i$ with probability $p_i^t = \frac{w_i^t}{\sum_j w_j^t}$
  - After $\ell^t$ is revealed, $w_i^{t+1} \leftarrow w_i^t \cdot (1 + \epsilon g_i^t) \quad \forall i$

---

- With the same proof, we can show that:

**Theorem 4.1.** *For* $0 < \epsilon \leq 1/2$, MWU($\epsilon$) *guarantees that*

$$G_A \geq G_\star - \epsilon T - \frac{\ln n}{\epsilon}.$$

*Moreover, if* $g^t \in [0,1]^n$ *for all* $t$, *then*

$$G_A \geq G_\star - \epsilon G_\star - \frac{\ln n}{\epsilon}.$$

# 5 Discussion: History

- MWU actually follows from a more general class of algorithm called "Follow the Regularized Leader", which generalizes gradient descent too!
  - See the fantastic lecture notes by Luca Travisan if you are interested.[1]
- MWU has been rediscovered independently many times.
  - 1950's Game theory (for solving zero-sum game)
  - 1980's Machine learning (AdaBoost: boost weaker learners to strong learners)
  - 1990's LP solvers and flow algorithms
- Philosophers ask whether math is *inverted* or *discovered*.
- Some algorithms are clearly inverted/engineered/very artificial.
- I think MWU is discovered. It was there...
  - In biology, Genes even update their strategies using MWU.[2]

---

[1] https://lucatrevisan.github.io/40391/lecture11.pdf and https://lucatrevisan.github.io/40391/lecture12.pdf

[2] https://cacm.acm.org/magazines/2016/11/209128-sex-as-an-algorithm/fulltext