

No-Regret Online Learning Algorithms

Joseph Chuang-Chieh Lin

Department of Computer Science & Engineering,
National Taiwan Ocean University

5th March 2025



Credits for the resource

The slides are based on the lectures of Prof. Luca Trevisan:
<https://lucatrevisan.github.io/40391/index.html>

the lectures of Prof. Shipra Agrawal:
<https://ieor8100.github.io/mab/>

the lectures of Prof. Francesco Orabona:
<https://parameterfree.com/lecture-notes-on-online-learning/>

and also Elad Hazan's textbook:
Introduction to Online Convex Optimization, 2nd Edition.



Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)
- 3 Multiplicative Weight Update (MWU)
- 4 Follow The Leader (FTL)
- 5 Follow The Regularized Leader (FTRL)
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)
 - Greedy Algorithms
 - Upper Confidence Bound (UCB)
 - Time-Decay ϵ -Greedy



Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)
- 3 Multiplicative Weight Update (MWU)
- 4 Follow The Leader (FTL)
- 5 Follow The Regularized Leader (FTRL)
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)
 - Greedy Algorithms
 - Upper Confidence Bound (UCB)
 - Time-Decay ϵ -Greedy



Online Convex Optimization

Goal: Design an algorithm such that

- At discrete time steps $t = 1, 2, \dots$, output $\mathbf{x}_t \in \mathcal{K}$, for each t .
 - \mathcal{K} : a convex set of feasible solutions.
- After \mathbf{x}_t is generated, a convex cost function $f_t : \mathcal{K} \mapsto \mathbb{R}$ is revealed.
- Then the algorithm suffers the loss $f_t(\mathbf{x}_t)$.

And we want to minimize the cost.



The difficulty

- The cost functions f_t is unknown before t .
- $f_1, f_2, \dots, f_t, \dots$ are not necessarily fixed.
 - Can be generated dynamically by an adversary.



What's the regret?

- The **offline optimum**: After T steps,

$$\min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}).$$

- The **regret** after T steps:

$$\text{regret}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}).$$



What's the regret?

- The **offline optimum**: After T steps,

$$\min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}).$$

- The **regret** after T steps:

$$\text{regret}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}).$$

- The rescue: $\text{regret}_T \leq o(T)$.



What's the regret?

- The **offline optimum**: After T steps,

$$\min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}).$$

- The **regret** after T steps:

$$\text{regret}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}).$$

- The rescue: $\text{regret}_T \leq o(T)$. \Rightarrow **No-Regret** in average when $T \rightarrow \infty$.
 - For example, $\text{regret}_T / T = \frac{\sqrt{T}}{T} \rightarrow 0$ when $T \rightarrow \infty$.



Prerequisites (1/5)

Diameter

Let $\mathcal{K} \subseteq \mathbb{R}^d$ be a bounded convex and closed set in Euclidean space. We denote by D an upper bound on the **diameter** of \mathcal{K} :

$$\forall \mathbf{x}, \mathbf{y} \in \mathcal{K}, \|\mathbf{x} - \mathbf{y}\| \leq D.$$

Convex set

A set \mathcal{K} is **convex** if for any $\mathbf{x}, \mathbf{y} \in \mathcal{K}$, we have

$$\forall \alpha \in [0, 1], \alpha \mathbf{x} + (1 - \alpha) \mathbf{y} \in \mathcal{K}.$$



Prerequisites (2/5)

Convex function

A function $f : \mathcal{K} \mapsto \mathbb{R}$ is **convex** if for any $\mathbf{x}, \mathbf{y} \in \mathcal{K}$,

$$\forall \alpha \in [0, 1], f((1 - \alpha)\mathbf{x} + \alpha\mathbf{y}) \leq (1 - \alpha)f(\mathbf{x}) + \alpha f(\mathbf{y}).$$

Equivalently, if f is differentiable (i.e., $\nabla f(\mathbf{x})$ exists for all $\mathbf{x} \in \mathcal{K}$), then f is convex if and only if for all $\mathbf{x}, \mathbf{y} \in \mathcal{K}$,

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}).$$



Prerequisites (3/5)

Theorem [Rockafellar 1970]

Suppose that $f : \mathcal{K} \mapsto \mathbb{R}$ is a convex function and let $x \in \text{int dom}(f)$. If f is differentiable at x , then for all $y \in \mathbb{R}^d$,

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle.$$

Subgradient

For a function $f : \mathbb{R}^d \mapsto \mathbb{R}$, $g \in \mathbb{R}^d$ is a **subgradient** of f at $x \in \mathbb{R}^d$ if for all $y \in \mathbb{R}^d$,

$$f(y) \geq f(x) + \langle g, y - x \rangle.$$



Prerequisites (4/5)

Projection

The closest point of \mathbf{y} in a convex set \mathcal{K} in terms of norm $\|\cdot\|$:

$$\Pi_{\mathcal{K}}(\mathbf{y}) := \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}\|.$$

Pythagoras Theorem

Let $\mathcal{K} \subseteq \mathbb{R}^d$ be a convex set, $\mathbf{y} \in \mathbb{R}^d$ and $\mathbf{x} = \Pi_{\mathcal{K}}(\mathbf{y})$. Then for any $\mathbf{z} \in \mathcal{K}$, we have

$$\|\mathbf{y} - \mathbf{z}\| \geq \|\mathbf{x} - \mathbf{z}\|.$$



Prerequisites (5/5)

Minimum vs. zero gradient

$$\nabla f(\mathbf{x}) = 0 \text{ iff } \mathbf{x} \in \arg \min_{\mathbf{x} \in \mathbb{R}^d} \{f(\mathbf{x})\}.$$

Karush-Kuhn-Tucker (KKT) Theorem

Let $\mathcal{K} \subseteq \mathbb{R}^d$ be a convex set, $\mathbf{x}^* \in \arg \min_{\mathbf{x} \in \mathcal{K}} f(\mathbf{x})$. Then for any $\mathbf{y} \in \mathcal{K}$ we have

$$\nabla f(\mathbf{x}^*)^\top (\mathbf{y} - \mathbf{x}^*) \geq 0.$$



Convex losses to linear losses

- We have the convex loss function $f_t(\mathbf{x}_t)$ at time t .
- Say we have subgradients \mathbf{g}_t for each \mathbf{x}_t .
- $f(\mathbf{x}_t) - f(\mathbf{u}) \leq \langle \mathbf{g}, \mathbf{x}_t - \mathbf{u} \rangle$ for each $\mathbf{u} \in \mathbb{R}^d$.



Convex losses to linear losses

- We have the convex loss function $f_t(\mathbf{x}_t)$ at time t .
- Say we have subgradients \mathbf{g}_t for each \mathbf{x}_t .
- $f(\mathbf{x}_t) - f(\mathbf{u}) \leq \langle \mathbf{g}_t, \mathbf{x}_t - \mathbf{u} \rangle$ for each $\mathbf{u} \in \mathbb{R}^d$.
- Hence, if we define $\tilde{f}_t(\mathbf{x}) := \langle \mathbf{g}_t, \mathbf{x} \rangle$, then for any $\mathbf{u} \in \mathbb{R}^d$,

$$\sum_{t=1}^T f_t(\mathbf{x}_t) - f(\mathbf{u}) \leq \sum_{t=1}^T \langle \mathbf{g}_t, \mathbf{x}_t - \mathbf{u} \rangle = \sum_{t=1}^T \tilde{f}_t(\mathbf{x}_t) - \tilde{f}(\mathbf{u}).$$



Convex losses to linear losses

- We have the convex loss function $f_t(\mathbf{x}_t)$ at time t .
- Say we have subgradients \mathbf{g}_t for each \mathbf{x}_t .
- $f(\mathbf{x}_t) - f(\mathbf{u}) \leq \langle \mathbf{g}_t, \mathbf{x}_t - \mathbf{u} \rangle$ for each $\mathbf{u} \in \mathbb{R}^d$.
- Hence, if we define $\tilde{f}_t(\mathbf{x}) := \langle \mathbf{g}_t, \mathbf{x} \rangle$, then for any $\mathbf{u} \in \mathbb{R}^d$,

$$\sum_{t=1}^T f_t(\mathbf{x}_t) - f(\mathbf{u}) \leq \sum_{t=1}^T \langle \mathbf{g}_t, \mathbf{x}_t - \mathbf{u} \rangle = \sum_{t=1}^T \tilde{f}_t(\mathbf{x}_t) - \tilde{f}(\mathbf{u}).$$

OCO \rightarrow OLO.



Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)**
- 3 Multiplicative Weight Update (MWU)
- 4 Follow The Leader (FTL)
- 5 Follow The Regularized Leader (FTRL)
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)
 - Greedy Algorithms
 - Upper Confidence Bound (UCB)
 - Time-Decay ϵ -Greedy



Online Gradient Descent (GD)

① **Input:** convex set \mathcal{K} , T , $\mathbf{x}_1 \in \mathcal{K}$, step size $\{\eta_t\}$.

② **for** $t \leftarrow 1$ to T **do:**

① Play \mathbf{x}_t and observe cost $f_t(\mathbf{x}_t)$.

② Update and Project:

$$\mathbf{y}_{t+1} = \mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t)$$

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{K}}(\mathbf{y}_{t+1})$$

③ **end for**



GD for online convex optimization is of no-regret

Theorem A

Online gradient descent with step size $\{\eta_t = \frac{D}{G\sqrt{t}}, t \in [T]\}$ guarantees the following for all $T \geq 1$:

$$\text{regret}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x}^* \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}^*) \leq \frac{3}{2} GD\sqrt{T}.$$



Proof of Theorem A (1/3)

- Let $\mathbf{x}^* \in \arg \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x})$.
- Since f_t is convex, we have

$$f_t(\mathbf{x}_t) - f_t(\mathbf{x}^*) \leq (\nabla f_t(\mathbf{x}_t))^\top (\mathbf{x}_t - \mathbf{x}^*).$$

- By the updating rule for \mathbf{x}_{t+1} and the Pythagorean theorem, we have

$$\|\mathbf{x}_{t+1} - \mathbf{x}^*\|^2 = \|\Pi_{\mathcal{K}}(\mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t)) - \mathbf{x}^*\|^2 \leq \|\mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t) - \mathbf{x}^*\|^2.$$



Proof of Theorem A (2/3)

- Hence

$$\begin{aligned} \|\mathbf{x}_{t+1} - \mathbf{x}^*\|^2 &\leq \|\mathbf{x}_t - \mathbf{x}^*\|^2 + \eta_t^2 \|\nabla f_t(\mathbf{x}_t)\|^2 - 2\eta_t (\nabla f_t(\mathbf{x}_t))^\top (\mathbf{x}_t - \mathbf{x}^*) \\ 2(\nabla f_t(\mathbf{x}_t))^\top (\mathbf{x}_t - \mathbf{x}^*) &\leq \frac{\|\mathbf{x}_t - \mathbf{x}^*\|^2 - \|\mathbf{x}_{t+1} - \mathbf{x}^*\|^2}{\eta_t} + \eta_t G^2. \end{aligned}$$

- Summing above inequality from $t = 1$ to T and setting $\eta_t = \frac{D}{G\sqrt{t}}$ and $\frac{1}{\eta_0} := 0$ we have :

Proof of Theorem A (3/3)

$$\begin{aligned}
 2 \left(\sum_{t=1}^T f_t(\mathbf{x}_t) - f_t(\mathbf{x}^*) \right) &\leq 2 \sum_{t=1}^T (\nabla f_t(\mathbf{x}_t))^\top (\mathbf{x}_t - \mathbf{x}^*) \\
 &\leq \sum_{t=1}^T \frac{\|\mathbf{x}_t - \mathbf{x}^*\|^2 - \|\mathbf{x}_{t+1} - \mathbf{x}^*\|^2}{\eta_t} + G^2 \sum_{t=1}^T \eta_t \\
 &\leq \sum_{t=1}^T \|\mathbf{x}_t - \mathbf{x}^*\|^2 \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) + G^2 \sum_{t=1}^T \eta_t \\
 &\leq D^2 \sum_{t=1}^T \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) + G^2 \sum_{t=1}^T \eta_t \\
 &\leq D^2 \frac{1}{\eta_T} + G^2 \sum_{t=1}^T \eta_t \\
 &\leq 3DG\sqrt{T}.
 \end{aligned}$$



The Lower Bound

Theorem B

Let $\mathcal{K} = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\|_\infty \leq r\}$ be a convex subset of \mathbb{R}^d . Let A be any algorithm for Online Convex Optimization on \mathcal{K} . Then for any $T \geq 1$, there exists a sequence of vectors $\mathbf{g}_1, \dots, \mathbf{g}_T$ with $\|\mathbf{g}_t\|_2 \leq L$ and $\mathbf{u} \in \mathcal{K}$ such that the regret of A satisfies

$$\text{regret}_T(\mathbf{u}) = \sum_{t=1}^T \langle \mathbf{g}_t, \mathbf{x}_t \rangle - \sum_{t=1}^T \langle \mathbf{g}_t, \mathbf{u} \rangle \geq \frac{\sqrt{2}LD\sqrt{T}}{4}.$$

- The diameter D of \mathcal{K} is at most $\sqrt{\sum_{i=1}^d (2r)^2} \leq 2r\sqrt{d}$.
- $\|\mathbf{x}\|_\infty \leq r \Leftrightarrow |\mathbf{x}(i)| \leq r$ for each $i \in [n]$.



Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)
- 3 Multiplicative Weight Update (MWU)**
- 4 Follow The Leader (FTL)
- 5 Follow The Regularized Leader (FTRL)
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)
 - Greedy Algorithms
 - Upper Confidence Bound (UCB)
 - Time-Decay ϵ -Greedy



Listen to the experts?

- Let's say we have n experts.
- We want to make best use of the advices coming from the experts.



Listen to the experts?

- Let's say we have n experts.
- We want to make best use of the advices coming from the experts.
- The idea: at each time step, decide the probability distribution (i.e., weights) of the experts to follow their advice.
 - $\mathbf{x}_t = (\mathbf{x}_t(1), \mathbf{x}_t(2), \dots, \mathbf{x}_t(n))$, where $\mathbf{x}_t(i) \in [0, 1]$ and $\sum_i \mathbf{x}_t(i) = 1$.



Listen to the experts?

- Let's say we have n experts.
- We want to make best use of the advices coming from the experts.
- The idea: at each time step, decide the probability distribution (i.e., weights) of the experts to follow their advice.
 - $\mathbf{x}_t = (\mathbf{x}_t(1), \mathbf{x}_t(2), \dots, \mathbf{x}_t(n))$, where $\mathbf{x}_t(i) \in [0, 1]$ and $\sum_i \mathbf{x}_t(i) = 1$.
- The loss of following expert i at time t : $\ell_t(i)$.
- The expected loss of the algorithm at time t :

$$\langle \mathbf{x}_t, \ell_t \rangle = \sum_{i=1}^n \mathbf{x}_t(i) \ell_t(i).$$



The regret of listening to the experts...

$$\text{regret}_T^* = \sum_{t=1}^T \langle \mathbf{x}_t, \ell_t \rangle - \min_i \sum_{t=1}^T \ell_t(i).$$

- The set of feasible solutions $K = \Delta \subseteq \mathbb{R}^n$, probability distributions over $\{1, \dots, n\}$.
- $f_t(\mathbf{x}) = \sum_i \mathbf{x}(i) \ell_t(i)$: linear function.
- ★ Assume that $|\ell_t(i)| \leq 1$ for all t and i .

The MWU Algorithm

- The spirit: “Hedge”.
- Well-known and frequently rediscovered.



The MWU Algorithm

- The spirit: “Hedge”.
- Well-known and frequently rediscovered.

Multiplicative Weight Update (MWU)

- Maintain a vector of weights $\mathbf{w}_t = (\mathbf{w}_t(1), \dots, \mathbf{w}_t(n))$ where $\mathbf{w}_1 := (1, 1, \dots, 1)$.
- Update the weights at time t by
 - $\mathbf{w}_t(i) := \mathbf{w}_{t-1}(i) \cdot e^{-\beta \ell_{t-1}(i)}$.
 - $\mathbf{x}_t := \frac{\mathbf{w}_t(i)}{\sum_{j=1}^n \mathbf{w}_t(j)}$.

β : a parameter which will be optimized later.



The MWU Algorithm

- The spirit: “Hedge”.
- Well-known and frequently rediscovered.

Multiplicative Weight Update (MWU)

- Maintain a vector of weights $\mathbf{w}_t = (\mathbf{w}_t(1), \dots, \mathbf{w}_t(n))$ where $\mathbf{w}_1 := (1, 1, \dots, 1)$.
- Update the weights at time t by
 - $\mathbf{w}_t(i) := \mathbf{w}_{t-1}(i) \cdot e^{-\beta \ell_{t-1}(i)}$.
 - $\mathbf{x}_t := \frac{\mathbf{w}_t(i)}{\sum_{j=1}^n \mathbf{w}_t(j)}$.

β : a parameter which will be optimized later.

The weight of expert i at time t : $e^{-\beta \sum_{k=1}^{t-1} \ell_k(i)}$.



MWU is of no-regret

Theorem 1 (MWU is of no-regret)

Assume that $|\ell_t(i)| \leq 1$ for all t and i . For $\beta \in (0, 1/2)$, the regret of MWU after T steps is bounded as

$$\text{regret}_T^* \leq \beta \sum_{t=1}^T \sum_{i=1}^n \mathbf{x}_t(i) \ell_t^2(i) + \frac{\ln n}{\beta} \leq \beta T + \frac{\ln n}{\beta}.$$

In particular, if $T > 4 \ln n$, then

$$\text{regret}_T^* \leq 2\sqrt{T \ln n}$$

by setting $\beta = \sqrt{\frac{\ln n}{T}}$.

Proof of Theorem 1

Let $W_t := \sum_{i=1}^n \mathbf{w}_t(i)$.

The idea:

- If the algorithm incurs a large loss after T steps, then W_{T+1} is small.
- And, if W_{T+1} is small, then even the best expert performs quite badly.



Proof of Theorem 1

Let $W_t := \sum_{i=1}^n \mathbf{w}_t(i)$.

The idea:

- If the algorithm incurs a large loss after T steps, then W_{T+1} is small.
- And, if W_{T+1} is small, then even the best expert performs quite badly.

Let $L^* := \min_i \sum_{t=1}^T \ell_t(i)$.



The proof (contd.)

Lemma 1 (W_{T+1} is SMALL $\Rightarrow L^*$ is LARGE)

$$W_{T+1} \geq e^{-\beta L^*}.$$

Proof.

Let $j = \arg \min L^* = \arg \min_i \sum_{t=1}^T \ell_t(i)$.

$$W_{T+1} = \sum_{i=1}^n e^{-\beta \sum_{t=1}^T \ell_t(i)} \geq e^{-\beta \sum_{t=1}^T \ell_t(j)} = e^{-\beta L^*}.$$



The proof (contd.)

Lemma 2 (MWU brings large loss $\Rightarrow W_{T+1}$ is SMALL)

$$W_{T+1} \leq n \prod_{t=1}^n (1 - \beta \langle \mathbf{x}_t, \ell_t \rangle + \beta^2 \langle \mathbf{x}_t, \ell_t^2 \rangle),$$

Proof.

Note: $W_1 = n$.

$$\frac{W_{t+1}}{W_t} = \sum_{i=1}^n \frac{\mathbf{w}_{t+1}(i)}{W_t} = \sum_{i=1}^n \frac{\mathbf{w}_t(i) \cdot e^{-\beta \ell_t(i)}}{W_t}$$

The proof (contd.)

Lemma 2 (MWU brings large loss $\Rightarrow W_{T+1}$ is SMALL)

$$W_{T+1} \leq n \prod_{t=1}^n (1 - \beta \langle \mathbf{x}_t, \ell_t \rangle + \beta^2 \langle \mathbf{x}_t, \ell_t^2 \rangle),$$

Proof.

Note: $W_1 = n$.

$$\begin{aligned} \frac{W_{t+1}}{W_t} &= \sum_{i=1}^n \frac{\mathbf{w}_{t+1}(i)}{W_t} = \sum_{i=1}^n \frac{\mathbf{w}_t(i) \cdot e^{-\beta \ell_t(i)}}{W_t} = \sum_{i=1}^n \mathbf{x}_t(i) \cdot e^{-\beta \ell_t(i)} \\ &\leq \sum_{i=1}^n \mathbf{x}_t(i) \cdot (1 - \beta \ell_t(i) + \beta^2 \ell_t^2(i)) \end{aligned}$$

The proof (contd.)

Lemma 2 (MWU brings large loss $\Rightarrow W_{T+1}$ is SMALL)

$$W_{T+1} \leq n \prod_{t=1}^n (1 - \beta \langle \mathbf{x}_t, \ell_t \rangle + \beta^2 \langle \mathbf{x}_t, \ell_t^2 \rangle),$$

Proof.

Note: $W_1 = n$.

$$\begin{aligned} \frac{W_{t+1}}{W_t} &= \sum_{i=1}^n \frac{\mathbf{w}_{t+1}(i)}{W_t} = \sum_{i=1}^n \frac{\mathbf{w}_t(i) \cdot e^{-\beta \ell_t(i)}}{W_t} = \sum_{i=1}^n \mathbf{x}_t(i) \cdot e^{-\beta \ell_t(i)} \\ &\leq \sum_{i=1}^n \mathbf{x}_t(i) \cdot (1 - \beta \ell_t(i) + \beta^2 \ell_t^2(i)) \\ &= 1 - \beta \langle \mathbf{x}_t, \ell_t \rangle + \beta^2 \langle \mathbf{x}_t, \ell_t^2 \rangle \end{aligned}$$

The proof (contd.)

Lemma 2 (MWU brings large loss $\Rightarrow W_{T+1}$ is SMALL)

$$W_{T+1} \leq n \prod_{t=1}^n (1 - \beta \langle \mathbf{x}_t, \ell_t \rangle + \beta^2 \langle \mathbf{x}_t, \ell_t^2 \rangle),$$

Proof.

Note: $W_1 = n$.

$$\begin{aligned} \frac{W_{t+1}}{W_t} &= \sum_{i=1}^n \frac{w_{t+1}(i)}{W_t} = \sum_{i=1}^n \frac{w_t(i) \cdot e^{-\beta \ell_t(i)}}{W_t} = \sum_{i=1}^n \mathbf{x}_t(i) \cdot e^{-\beta \ell_t(i)} \\ &\leq \sum_{i=1}^n \mathbf{x}_t(i) \cdot (1 - \beta \ell_t(i) + \beta^2 \ell_t^2(i)) \\ &= 1 - \beta \langle \mathbf{x}_t, \ell_t \rangle + \beta^2 \langle \mathbf{x}_t, \ell_t^2 \rangle \leq e^{-\beta \langle \mathbf{x}_t, \ell_t \rangle + \beta^2 \langle \mathbf{x}_t, \ell_t^2 \rangle}. \end{aligned}$$



The proof (contd.)

Lemma 2 (MWU brings large loss $\Rightarrow W_{T+1}$ is SMALL)

$$W_{T+1} \leq n \prod_{t=1}^n e^{-\beta \langle \mathbf{x}_t, \ell_t \rangle + \beta^2 \langle \mathbf{x}_t, \ell_t^2 \rangle}.$$

Proof.

Note: $W_1 = n$.

$$\begin{aligned} \frac{W_{t+1}}{W_t} &= \sum_{i=1}^n \frac{w_{t+1}(i)}{W_t} = \sum_{i=1}^n \frac{w_t(i) \cdot e^{-\beta \ell_t(i)}}{W_t} = \sum_{i=1}^n x_t(i) \cdot e^{-\beta \ell_t(i)} \\ &\leq \sum_{i=1}^n x_t(i) \cdot (1 - \beta \ell_t(i) + \beta^2 \ell_t^2(i)) \\ &= 1 - \beta \langle \mathbf{x}_t, \ell_t \rangle + \beta^2 \langle \mathbf{x}_t, \ell_t^2 \rangle \leq e^{-\beta \langle \mathbf{x}_t, \ell_t \rangle + \beta^2 \langle \mathbf{x}_t, \ell_t^2 \rangle}. \end{aligned}$$



The proof (contd.)

Hence

$$\ln W_{T+1} \leq \ln n - \left(\sum_{i=1}^T \beta \langle \ell_t, \mathbf{x}_t \rangle \right) + \left(\sum_{i=1}^T \beta^2 \langle \ell_t^2, \mathbf{x}_t \rangle \right)$$

and $\ln W_{T+1} \geq -\beta L^*$.



The proof (contd.)

Hence

$$\ln W_{T+1} \leq \ln n - \left(\sum_{i=1}^T \beta \langle \ell_t, \mathbf{x}_t \rangle \right) + \left(\sum_{i=1}^T \beta^2 \langle \ell_t^2, \mathbf{x}_t \rangle \right)$$

and $\ln W_{T+1} \geq -\beta L^*$.

Thus,

$$\left(\sum_{t=1}^T \langle \ell_t, \mathbf{x}_t \rangle \right) - L^* \leq \frac{\ln n}{\beta} + \beta \sum_{t=1}^T \langle \ell_t^2, \mathbf{x}_t \rangle.$$



The proof (contd.)

Hence

$$\ln W_{T+1} \leq \ln n - \left(\sum_{i=1}^T \beta \langle \ell_t, \mathbf{x}_t \rangle \right) + \left(\sum_{i=1}^T \beta^2 \langle \ell_t^2, \mathbf{x}_t \rangle \right)$$

and $\ln W_{T+1} \geq -\beta L^*$.

Thus,

$$\left(\sum_{t=1}^T \langle \ell_t, \mathbf{x}_t \rangle \right) - L^* \leq \frac{\ln n}{\beta} + \beta \sum_{t=1}^T \langle \ell_t^2, \mathbf{x}_t \rangle.$$

Take $\beta = \sqrt{\frac{\ln n}{T}}$, we have $\text{regret}_T \leq 2\sqrt{T \ln n}$.



Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)
- 3 Multiplicative Weight Update (MWU)
- 4 Follow The Leader (FTL)**
- 5 Follow The Regularized Leader (FTRL)
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)
 - Greedy Algorithms
 - Upper Confidence Bound (UCB)
 - Time-Decay ϵ -Greedy



Why so complicated?

- How about just *following the one with best performance*?



Why so complicated?

- How about just *following the one with best performance*?
 - Follow The Leader (FTL) Algorithm.



Why so complicated?

- How about just *following the one with best performance*?
 - Follow The Leader (FTL) Algorithm.
- First, we assume to make no assumptions on \mathcal{K} and $\{f_t : L \mapsto \mathbb{R}\}$.
- At time t , we are given previous cost functions f_1, \dots, f_{t-1} , and then give the solution

$$\mathbf{x}_t := \arg \min_{\mathbf{x} \in \mathcal{K}} \sum_{k=1}^{t-1} f_k(\mathbf{x}).$$



Why so complicated?

- How about just *following the one with best performance*?
 - Follow The Leader (FTL) Algorithm.
- First, we assume to make no assumptions on \mathcal{K} and $\{f_t : L \mapsto \mathbb{R}\}$.
- At time t , we are given previous cost functions f_1, \dots, f_{t-1} , and then give the solution

$$\mathbf{x}_t := \arg \min_{\mathbf{x} \in \mathcal{K}} \sum_{k=1}^{t-1} f_k(\mathbf{x}).$$

That is, the best solution for the previous $t-1$ steps.



Why so complicated?

- How about just *following the one with best performance*?
 - Follow The Leader (FTL) Algorithm.
- First, we assume to make no assumptions on \mathcal{K} and $\{f_t : L \mapsto \mathbb{R}\}$.
- At time t , we are given previous cost functions f_1, \dots, f_{t-1} , and then give the solution

$$\mathbf{x}_t := \arg \min_{\mathbf{x} \in \mathcal{K}} \sum_{k=1}^{t-1} f_k(\mathbf{x}).$$

That is, the best solution for the previous $t-1$ steps.

- It seems reasonable and makes sense, doesn't it?



FTL leads to “overfitting”

$$t: \quad 1$$

$$\mathbf{x}_t: \quad (0.5, 0.5)$$

$$\ell_t: \quad (0, 0.5)$$

$$f_t(\mathbf{x}_t): \quad 0.25$$

$$\arg \min_{\mathbf{x}} \sum_{k=1}^t f_k(\mathbf{x}): \quad (1, 0)$$

FTL leads to “overfitting”

t :	1	2
\mathbf{x}_t :	(0.5, 0.5)	(1, 0)
ℓ_t :	(0, 0.5)	(1, 0)
$f_t(\mathbf{x}_t)$:	0.25	1
$\arg \min_{\mathbf{x}} \sum_{k=1}^t f_k(\mathbf{x})$:	(1, 0)	(0, 1)

FTL leads to “overfitting”

t :	1	2	3
\mathbf{x}_t :	(0.5, 0.5)	(1, 0)	(0, 1)
ℓ_t :	(0, 0.5)	(1, 0)	(0, 1)
$f_t(\mathbf{x}_t)$:	0.25	1	1
$\arg \min_{\mathbf{x}} \sum_{k=1}^t f_k(\mathbf{x})$:	(1, 0)	(0, 1)	(1, 0)

FTL leads to “overfitting”

t :	1	2	3	4
\mathbf{x}_t :	(0.5, 0.5)	(1, 0)	(0, 1)	(1, 0)
ℓ_t :	(0, 0.5)	(1, 0)	(0, 1)	(1, 0)
$f_t(\mathbf{x}_t)$:	0.25	1	1	1
$\arg \min_{\mathbf{x}} \sum_{k=1}^t f_k(\mathbf{x})$:	(1, 0)	(0, 1)	(1, 0)	(0, 1)

FTL leads to “overfitting”

t :	1	2	3	4	5
\mathbf{x}_t :	(0.5, 0.5)	(1, 0)	(0, 1)	(1, 0)	(0, 1)
ℓ_t :	(0, 0.5)	(1, 0)	(0, 1)	(1, 0)	(0, 1)
$f_t(\mathbf{x}_t)$:	0.25	1	1	1	1
$\arg \min_{\mathbf{x}} \sum_{k=1}^t f_k(\mathbf{x})$:	(1, 0)	(0, 1)	(1, 0)	(0, 1)	(1, 0)

FTL leads to “overfitting”

t :	1	2	3	4	5	...
\mathbf{x}_t :	(0.5, 0.5)	(1, 0)	(0, 1)	(1, 0)	(0, 1)	...
ℓ_t :	(0, 0.5)	(1, 0)	(0, 1)	(1, 0)	(0, 1)	...
$f_t(\mathbf{x}_t)$:	0.25	1	1	1	1	...
$\arg \min_{\mathbf{x}} \sum_{k=1}^t f_k(\mathbf{x})$:	(1, 0)	(0, 1)	(1, 0)	(0, 1)	(1, 0)	...

FTL leads to “overfitting”

t :	1	2	3	4	5	...
\mathbf{x}_t :	(0.5, 0.5)	(1, 0)	(0, 1)	(1, 0)	(0, 1)	...
ℓ_t :	(0, 0.5)	(1, 0)	(0, 1)	(1, 0)	(0, 1)	...
$f_t(\mathbf{x}_t)$:	0.25	1	1	1	1	...
$\arg \min_{\mathbf{x}} \sum_{k=1}^t f_k(\mathbf{x})$:	(1, 0)	(0, 1)	(1, 0)	(0, 1)	(1, 0)	...

optimum loss: $\approx T/2$.

FTL's loss: $\approx T$.

regret: $\approx T/2$ (linear).



Analysis of FTL

Theorem 2 (Analysis of FTL)

For any sequence of cost functions f_1, \dots, f_t and any number of time steps T , the FTL algorithm satisfies

$$\text{regret}_T \leq \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})).$$



Analysis of FTL

Theorem 2 (Analysis of FTL)

For any sequence of cost functions f_1, \dots, f_t and any number of time steps T , the FTL algorithm satisfies

$$\text{regret}_T \leq \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})).$$

Implication: If $f_t(\cdot)$ is Lipschitz w.r.t. to some distance function $\|\cdot\|$, then \mathbf{x}_t and \mathbf{x}_{t+1} are close $\Rightarrow \|f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})\|$ can't be too large.

Modify FTL: \mathbf{x}_t 's shouldn't change too much from step by step.



Proof of Theorem 2

Recall that

$$\text{regret}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x})$$

Proof of Theorem 2

Recall that

$$\text{regret}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}) \leq \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})).$$

The theorem $\Leftrightarrow \sum_{t=1}^T f_t(\mathbf{x}_{t+1}) \leq \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x})$.

Proof of Theorem 2

Recall that

$$\text{regret}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}) \leq \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})).$$

The theorem $\Leftrightarrow \sum_{t=1}^T f_t(\mathbf{x}_{t+1}) \leq \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x})$.

Prove by induction. $T = 1$: The definition of \mathbf{x}_2 .



Proof of Theorem 2

Recall that

$$\text{regret}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}) \leq \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})).$$

The theorem $\Leftrightarrow \sum_{t=1}^T f_t(\mathbf{x}_{t+1}) \leq \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x})$.

Prove by induction. $T = 1$: The definition of \mathbf{x}_2 .

Assume that it holds up to T . Then:

Proof of Theorem 2

Recall that

$$\text{regret}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}) \leq \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})).$$

The theorem $\Leftrightarrow \sum_{t=1}^T f_t(\mathbf{x}_{t+1}) \leq \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x})$.

Prove by induction. $T = 1$: The definition of \mathbf{x}_2 .

Assume that it holds up to T . Then:

$$\sum_{t=1}^{T+1} f_t(\mathbf{x}_{t+1}) = \sum_{t=1}^T f_t(\mathbf{x}_{t+1}) + f_{T+1}(\mathbf{x}_{T+2}) \leq \sum_{t=1}^{T+1} f_t(\mathbf{x}_{T+2}) = \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^{T+1} f_t(\mathbf{x}),$$

Proof of Theorem 2

Recall that

$$\text{regret}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}) \leq \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})).$$

$$\text{The theorem} \Leftrightarrow \sum_{t=1}^T f_t(\mathbf{x}_{t+1}) \leq \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}).$$

Prove by induction. $T = 1$: The definition of \mathbf{x}_2 .

Assume that it holds up to T . Then:

$$\sum_{t=1}^{T+1} f_t(\mathbf{x}_{t+1}) = \sum_{t=1}^T f_t(\mathbf{x}_{t+1}) + f_{T+1}(\mathbf{x}_{T+2}) \leq \sum_{t=1}^{T+1} f_t(\mathbf{x}_{T+2}) = \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^{T+1} f_t(\mathbf{x}),$$

where

$$\sum_{t=1}^T f_t(\mathbf{x}_{t+1}) \leq \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}) \leq \sum_{t=1}^T f_t(\mathbf{x}_{T+2}).$$



Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)
- 3 Multiplicative Weight Update (MWU)
- 4 Follow The Leader (FTL)
- 5 Follow The Regularized Leader (FTRL)**
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)
 - Greedy Algorithms
 - Upper Confidence Bound (UCB)
 - Time-Decay ϵ -Greedy



Introducing REGULARIZATION

- You might have already been using regularization for quite a long time.



Introducing REGULARIZATION

```
from keras import regularizers  
model.add(Dense(64, input_dim=64,  
                kernel_regularizer=regularizers.l2(0.01))
```



Introducing REGULARIZATION

```
# L1 data (only 5 informative features)
X_1, y_1 = datasets.make_classification(n_samples=n_samples,
                                       n_features=n_features, n_informative=5,
                                       random_state=1)

# L2 data: non sparse, but less features
y_2 = np.sign(.5 - rnd.rand(n_samples))
X_2 = rnd.randn(n_samples, n_features // 5) + y_2[:, np.newaxis]
X_2 += 5 * rnd.randn(n_samples, n_features // 5)

clf_sets = [(LinearSVC(penalty='l1', loss='squared_hinge', dual=False,
                       tol=1e-3),
              np.logspace(-2.3, -1.3, 10), X_1, y_1),
             (LinearSVC(penalty='l2', loss='squared_hinge', dual=True),
              np.logspace(-4.5, -2, 10), X_2, y_2)]
```

The regularizer

At each step, we compute the solution

$$\mathbf{x}_t := \arg \min_{\mathbf{x} \in \mathcal{K}} \left(R(\mathbf{x}) + \sum_{k=1}^{t-1} f_k(\mathbf{x}) \right).$$

This is called **Follow the Regularized Leader (FTRL)**.

In short,

$$\text{FTRL} = \text{FTL} + \text{Regularizer}.$$



Analysis of FTRL

Theorem 3 (Analysis of FTRL)

For

- every sequence of cost function $\{f_t(\cdot)\}_{t \geq 1}$ and
- every regularizer function $R(\cdot)$,

for every \mathbf{x} , the regret with respect to \mathbf{x} after T steps of the FTRL algorithm is bounded as

$$\text{regret}_T(\mathbf{x}) \leq \left(\sum_{t=1}^T f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) \right) + R(\mathbf{x}) - R(\mathbf{x}_1),$$

where $\text{regret}_T(\mathbf{x}) := \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}))$.

Proof of Theorem 3

- Consider a *mental* experiment:

Proof of Theorem 3

- Consider a *mental* experiment:
 - We run the FTL algorithm for $T + 1$ steps.
 - The sequence of cost functions: R, f_1, f_2, \dots, f_T .
 - Use x_1 as the first solution.
 - The solutions: $x_1, x_1, x_2, \dots, x_T$.



Proof of Theorem 3

- Consider a *mental* experiment:
 - We run the FTL algorithm for $T + 1$ steps.
 - The sequence of cost functions: R, f_1, f_2, \dots, f_T .
 - Use \mathbf{x}_1 as the first solution.
 - The solutions: $\mathbf{x}_1, \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T$.
- The regret:

$$R(\mathbf{x}_1) - R(\mathbf{x}) + \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}))$$



Proof of Theorem 3

- Consider a *mental* experiment:
 - We run the FTL algorithm for $T + 1$ steps.
 - The sequence of cost functions: R, f_1, f_2, \dots, f_T .
 - Use \mathbf{x}_1 as the first solution.
 - The solutions: $\mathbf{x}_1, \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T$.
- The regret:

$$R(\mathbf{x}_1) - R(\mathbf{x}) + \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x})) \leq R(\mathbf{x}_1) - R(\mathbf{x}_1) + \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}))$$

minimizer of $R(\cdot)$



Proof of Theorem 3

- Consider a *mental* experiment:
 - We run the FTL algorithm for $T + 1$ steps.
 - The sequence of cost functions: R, f_1, f_2, \dots, f_T .
 - Use \mathbf{x}_1 as the first solution.
 - The solutions: $\mathbf{x}_1, \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T$.
- The regret:

$$R(\mathbf{x}_1) - R(\mathbf{x}) + \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x})) \leq R(\mathbf{x}_1) - R(\mathbf{x}_1) + \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}))$$

output of FTRL at $t + 1$



Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)
- 3 Multiplicative Weight Update (MWU)
- 4 Follow The Leader (FTL)
- 5 Follow The Regularized Leader (FTRL)**
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)
 - Greedy Algorithms
 - Upper Confidence Bound (UCB)
 - Time-Decay ϵ -Greedy



Using negative-entropy regularization

- We have seen an example that FTL tends to put all probability mass on one expert (it's bad!)



Using negative-entropy regularization

- We have seen an example that FTL tends to put all probability mass on one expert (it's bad!)
- **Idea:** penalize over “concentrated” distributions.
 - *negative-entropy*: a good measure of how centralized a distribution is.



Using negative-entropy regularization

- We have seen an example that FTL tends to put all probability mass on one expert (it's bad!)
- **Idea:** penalize over “concentrated” distributions.
 - *negative-entropy*: a good measure of how centralized a distribution is.

$$R(\mathbf{x}) := c \cdot \sum_{i=1}^n \mathbf{x}(i) \ln \mathbf{x}(i).$$



Using negative-entropy regularization

- We have seen an example that FTL tends to put all probability mass on one expert (it's bad!)
- **Idea:** penalize over “concentrated” distributions.
 - *negative-entropy*: a good measure of how centralized a distribution is.

$$R(\mathbf{x}) := c \cdot \sum_{i=1}^n x(i) \ln x(i).$$

- So our FTRL gives

$$\mathbf{x}_t = \arg \min_{\mathbf{x} \in \Delta} \left(\sum_{k=1}^{t-1} \langle \ell_k, \mathbf{x} \rangle + c \cdot \sum_{i=1}^n x(i) \ln x(i) \right).$$



Using negative entropy regularization

$$\mathbf{x}_t = \arg \min_{\mathbf{x} \in \Delta} \left(\sum_{k=1}^{t-1} \langle \ell_k, \mathbf{x} \rangle + c \cdot \sum_{i=1}^n \mathbf{x}(i) \ln \mathbf{x}(i) \right).$$

- The constraint $\mathbf{x} \in \Delta \Rightarrow \sum_i \mathbf{x}_i = 1$.
- So we use **Lagrange multiplier** to solve

$$\mathcal{L} = \left(\sum_{k=1}^{t-1} \langle \ell_k, \mathbf{x} \rangle \right) + c \cdot \left(\sum_{i=1}^n \mathbf{x}(i) \ln \mathbf{x}(i) \right) + \lambda \cdot (\langle \mathbf{x}, \mathbf{1} \rangle - 1).$$

Using negative entropy regularization

$$\mathbf{x}_t = \arg \min_{\mathbf{x} \in \Delta} \left(\sum_{k=1}^{t-1} \langle \ell_k, \mathbf{x} \rangle + c \cdot \sum_{i=1}^n \mathbf{x}(i) \ln \mathbf{x}(i) \right).$$

- The constraint $\mathbf{x} \in \Delta \Rightarrow \sum_i \mathbf{x}_i = 1$.
- So we use **Lagrange multiplier** to solve

$$\mathcal{L} = \left(\sum_{k=1}^{t-1} \langle \ell_k, \mathbf{x} \rangle \right) + c \cdot \left(\sum_{i=1}^n \mathbf{x}(i) \ln \mathbf{x}(i) \right) + \lambda \cdot (\langle \mathbf{x}, \mathbf{1} \rangle - 1).$$

- The partial derivative $\frac{\partial \mathcal{L}}{\partial \mathbf{x}(i)}$:

$$\left(\sum_{k=1}^{t-1} \ell_k(i) \right) + c \cdot (1 + \ln \mathbf{x}_i) + \lambda$$



Rediscover MWU?

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}(i)} = 0 \quad \Rightarrow \quad \mathbf{x}(i) = \exp \left(-1 - \frac{\lambda}{c} - \frac{1}{c} \sum_{k=1}^{t-1} \ell_k(i) \right)$$



Rediscover MWU?

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}(i)} = 0 \quad \Rightarrow \quad \mathbf{x}(i) = \exp \left(-1 - \frac{\lambda}{c} - \frac{1}{c} \sum_{k=1}^{t-1} \ell_k(i) \right)$$

Take the value of λ to make the solution a probability distribution.

Thus,



Rediscover MWU?

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}(i)} = 0 \Rightarrow \mathbf{x}(i) = \exp \left(-1 - \frac{\lambda}{c} - \frac{1}{c} \sum_{k=1}^{t-1} \ell_k(i) \right)$$

Take the value of λ to make the solution a probability distribution.

Thus,

$$\mathbf{x}(i) = \frac{\exp \left(-\frac{1}{c} \sum_{k=1}^{t-1} \ell_k(i) \right)}{\sum_j \exp \left(-\frac{1}{c} \sum_{k=1}^{t-1} \ell_k(j) \right)}.$$



Rediscover MWU?

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}(i)} = 0 \quad \Rightarrow \quad \mathbf{x}(i) = \exp \left(-1 - \frac{\lambda}{c} - \frac{1}{c} \sum_{k=1}^{t-1} \ell_k(i) \right)$$

Take the value of λ to make the solution a probability distribution.

Thus,

$$\mathbf{x}(i) = \frac{\exp \left(-\frac{1}{c} \sum_{k=1}^{t-1} \ell_k(i) \right)}{\sum_j \exp \left(-\frac{1}{c} \sum_{k=1}^{t-1} \ell_k(j) \right)}.$$

Exactly the solution of MWU if we take $c = 1/\beta!$



Rediscover MWU?

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}(i)} = 0 \quad \Rightarrow \quad \mathbf{x}(i) = \exp \left(-1 - \frac{\lambda}{c} - \frac{1}{c} \sum_{k=1}^{t-1} \ell_k(i) \right)$$

Take the value of λ to make the solution a probability distribution.

Thus,

$$\mathbf{x}(i) = \frac{\exp \left(-\frac{1}{c} \sum_{k=1}^{t-1} \ell_k(i) \right)}{\sum_j \exp \left(-\frac{1}{c} \sum_{k=1}^{t-1} \ell_k(j) \right)}.$$

Exactly the solution of MWU if we take $c = 1/\beta$!

- Now it remains to bound the deviation of each step.



Regret of FTRL + Negative-Entropy Regularization

- At each step,
$$f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) = \langle \ell_t, \mathbf{x}_t - \mathbf{x}_{t+1} \rangle$$
- Let's go back to use the notation of MWU.
 - $\mathbf{w}_1(i) = 1$ (initialization).
 - $\mathbf{w}_{t+1}(i) = \mathbf{w}_t(i) \cdot e^{-\ell_t(i)/c}$.



Regret of FTRL + Negative-Entropy Regularization

- At each step,
 $f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) = \langle \ell_t, \mathbf{x}_t - \mathbf{x}_{t+1} \rangle$.
- Let's go back to use the notation of MWU.
 - $\mathbf{w}_1(i) = 1$ (initialization).
 - $\mathbf{w}_{t+1}(i) = \mathbf{w}_t(i) \cdot e^{-\ell_t(i)/c}$.

- So, $\mathbf{x}_t = \frac{\mathbf{w}_t(i)}{\sum_j \mathbf{w}_t(j)}$.

- Then,

$$\begin{aligned} \mathbf{x}_{t+1}(i) &= \frac{\mathbf{w}_{t+1}(i)}{\sum_j \mathbf{w}_{t+1}(j)} = \frac{\mathbf{w}_t(i)e^{-\ell_t(i)/c}}{\sum_j \mathbf{w}_{t+1}(j)} \geq \frac{\mathbf{w}_t(i)e^{-\ell_t(i)/c}}{\sum_j \mathbf{w}_t(j)} \\ &\geq \mathbf{x}_t(i) \cdot e^{-1/c} \geq (1 - 1/c)\mathbf{x}_t(i). \end{aligned}$$



Regret of FTRL + Negative-Entropy Regularization

- At each step,

$$f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) = \langle \ell_t, \mathbf{x}_t - \mathbf{x}_{t+1} \rangle$$
- Let's go back to use the notation of MWU.

- $\mathbf{w}_1(i) = 1$ (initialization).
- $\mathbf{w}_{t+1}(i) = \mathbf{w}_t(i) \cdot e^{-\ell_t(i)/c}$.

- So, $\mathbf{x}_t = \frac{\mathbf{w}_t(i)}{\sum_j \mathbf{w}_t(j)}$.

- Then,

$$\begin{aligned} \mathbf{x}_{t+1}(i) &= \frac{\mathbf{w}_{t+1}(i)}{\sum_j \mathbf{w}_{t+1}(j)} = \frac{\mathbf{w}_t(i)e^{-\ell_t(i)/c}}{\sum_j \mathbf{w}_{t+1}(j)} \geq \frac{\mathbf{w}_t(i)e^{-\ell_t(i)/c}}{\sum_j \mathbf{w}_t(j)} \\ &\geq \mathbf{x}_t(i) \cdot e^{-1/c} \geq (1 - 1/c)\mathbf{x}_t(i). \end{aligned}$$

\therefore weights are non-increasing



Regret of FTRL + Negative-Entropy Regularization

- At each step,
$$f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) = \langle \ell_t, \mathbf{x}_t - \mathbf{x}_{t+1} \rangle$$
- Let's go back to use the notation of MWU.

- $\mathbf{w}_1(i) = 1$ (initialization).

- $\mathbf{w}_{t+1}(i) = \mathbf{w}_t(i) \cdot e^{-\ell_t(i)/c}$.

- So, $\mathbf{x}_t = \frac{\mathbf{w}_t(i)}{\sum_j \mathbf{w}_t(j)}$.

- Then,

$$\begin{aligned} \mathbf{x}_{t+1}(i) &= \frac{\mathbf{w}_{t+1}(i)}{\sum_j \mathbf{w}_{t+1}(j)} = \frac{\mathbf{w}_t(i)e^{-\ell_t(i)/c}}{\sum_j \mathbf{w}_{t+1}(j)} \geq \frac{\mathbf{w}_t(i)e^{-\ell_t(i)/c}}{\sum_j \mathbf{w}_t(j)} \\ &\geq \mathbf{x}_t(i) \cdot e^{-1/c} \geq (1 - 1/c)\mathbf{x}_t(i). \end{aligned}$$

assume $0 \leq \ell_t(i) \leq 1$



Regret of FTRL + Negative-Entropy Regularization

- At each step,

$$f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) = \langle \ell_t, \mathbf{x}_t - \mathbf{x}_{t+1} \rangle \leq \sum_i \ell_t(i) \cdot \frac{1}{c} \mathbf{x}_t(i) \leq \frac{1}{c}.$$
- Let's go back to use the notation of MWU.
 - $\mathbf{w}_1(i) = 1$ (initialization).
 - $\mathbf{w}_{t+1}(i) = \mathbf{w}_t(i) \cdot e^{-\ell_t(i)/c}.$
- So, $\mathbf{x}_t = \frac{\mathbf{w}_t(i)}{\sum_j \mathbf{w}_t(j)}.$
- Then,

$$\begin{aligned} \mathbf{x}_{t+1}(i) &= \frac{\mathbf{w}_{t+1}(i)}{\sum_j \mathbf{w}_{t+1}(j)} = \frac{\mathbf{w}_t(i) e^{-\ell_t(i)/c}}{\sum_j \mathbf{w}_{t+1}(j)} \geq \frac{\mathbf{w}_t(i) e^{-\ell_t(i)/c}}{\sum_j \mathbf{w}_t(j)} \\ &\geq \mathbf{x}_t(i) \cdot e^{-1/c} \geq (1 - 1/c) \mathbf{x}_t(i). \end{aligned}$$

Regret of FTRL + Negative-Entropy Regularization

- By Theorem 3, for any \mathbf{x} ,

$$\text{regret}_T(\mathbf{x}) \leq \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})) + R(\mathbf{x}) - R(\mathbf{x}_1) \leq \frac{T}{c} + c \ln n.$$



Regret of FTRL + Negative-Entropy Regularization

- By Theorem 3, for any \mathbf{x} ,

$$\text{regret}_T(\mathbf{x}) \leq \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})) + R(\mathbf{x}) - R(\mathbf{x}_1) \leq \frac{T}{c} + c \ln n.$$

\therefore max entropy for uniform distribution



Regret of FTRL + Negative-Entropy Regularization

- By Theorem 3, for any \mathbf{x} ,

$$\text{regret}_T(\mathbf{x}) \leq \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})) + R(\mathbf{x}) - R(\mathbf{x}_1) \leq \frac{T}{c} + c \ln n.$$

Again, we have $\text{regret}_T \leq 2\sqrt{T \ln n}$ by choosing $c = \sqrt{\frac{T}{\ln n}}$.



Regret of FTRL + Negative-Entropy Regularization

- By Theorem 3, for any \mathbf{x} ,

$$\text{regret}_T(\mathbf{x}) \leq \sum_{t=1}^T (f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1})) + R(\mathbf{x}) - R(\mathbf{x}_1) \leq \frac{T}{c} + c \ln n.$$

Again, we have $\text{regret}_T \leq 2\sqrt{T \ln n}$ by choosing $c = \sqrt{\frac{T}{\ln n}}$.

- Note the slight difference b/w regret and regret*.



Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)
- 3 Multiplicative Weight Update (MWU)
- 4 Follow The Leader (FTL)
- 5 Follow The Regularized Leader (FTRL)**
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)
 - Greedy Algorithms
 - Upper Confidence Bound (UCB)
 - Time-Decay ϵ -Greedy



L2 Regularization

- Let's try to apply the FTRL to the case that the regularizer is of L2 norm!
- Consider also linear cost functions but $\mathcal{K} = \mathbb{R}^n$ first.
- What kind of problem we might encounter?



L2 Regularization

- Let's try to apply the FTRL to the case that the regularizer is of L2 norm!
- Consider also linear cost functions but $\mathcal{K} = \mathbb{R}^n$ first.
- What kind of problem we might encounter?
- The offline optimum could be $-\infty$.
- FTL will also tend to find a solution of “big” size, too.



L2 Regularization

- Let's try to apply the FTRL to the case that the regularizer is of L2 norm!
- Consider also linear cost functions but $\mathcal{K} = \mathbb{R}^n$ first.
- What kind of problem we might encounter?
- The offline optimum could be $-\infty$.
- FTL will also tend to find a solution of “big” size, too.
- To fight this tendency, it makes sense to use a regularizer which penalizes the size of a solution.

$$R(\mathbf{x}) := c\|\mathbf{x}\|^2.$$



The regularizer of 2-norm tells us...

- $\mathbf{x}_1 = \mathbf{0}$.
- $\mathbf{x}_{t+1} = \arg \min_{\mathbf{x} \in \mathbb{R}^n} c \|\mathbf{x}\|^2 + \sum_{k=1}^t \langle \ell_k, \mathbf{x} \rangle$.
- Compute the gradient:

$$2c\mathbf{x} + \sum_{k=1}^t \ell_k = 0$$
$$\Rightarrow \mathbf{x} = -\frac{1}{2c} \sum_{k=1}^t \ell_k.$$

Hence, $\mathbf{x}_1 = \mathbf{0}$, $\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{1}{2c} \ell_t$.



The regularizer of 2-norm tells us...

- $\mathbf{x}_1 = \mathbf{0}$.
- $\mathbf{x}_{t+1} = \arg \min_{\mathbf{x} \in \mathbb{R}^n} c \|\mathbf{x}\|^2 + \sum_{k=1}^t \langle \ell_k, \mathbf{x} \rangle$.
convex
- Compute the gradient:

$$2c\mathbf{x} + \sum_{k=1}^t \ell_k = 0$$
$$\Rightarrow \mathbf{x} = -\frac{1}{2c} \sum_{k=1}^t \ell_k.$$

Hence, $\mathbf{x}_1 = \mathbf{0}$, $\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{1}{2c} \ell_t$.



The regularizer of 2-norm tells us...

- $\mathbf{x}_1 = \mathbf{0}$.
- $\mathbf{x}_{t+1} = \arg \min_{\mathbf{x} \in \mathbb{R}^n} c \|\mathbf{x}\|^2 + \sum_{k=1}^t \langle \ell_k, \mathbf{x} \rangle$.
- Compute the gradient:

$$2c\mathbf{x} + \sum_{k=1}^t \ell_k = 0$$
$$\Rightarrow \mathbf{x} = -\frac{1}{2c} \sum_{k=1}^t \ell_k.$$

Hence, $\mathbf{x}_1 = \mathbf{0}$, $\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{1}{2c} \ell_t$.

The regularizer of 2-norm tells us...

- $\mathbf{x}_1 = \mathbf{0}$.
- $\mathbf{x}_{t+1} = \arg \min_{\mathbf{x} \in \mathbb{R}^n} c \|\mathbf{x}\|^2 + \sum_{k=1}^t \langle \ell_k, \mathbf{x} \rangle$.
- Compute the gradient:

$$2c\mathbf{x} + \sum_{k=1}^t \ell_k = 0$$
$$\Rightarrow \mathbf{x} = -\frac{1}{2c} \sum_{k=1}^t \ell_k.$$

Hence, $\mathbf{x}_1 = \mathbf{0}$, $\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{1}{2c} \ell_t$.

→ penalize the experts that performed badly in the past!



The regret of FTRL with 2-norm regularization

- First, we have

$$f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) = \langle \ell_t, \mathbf{x}_t - \mathbf{x}_{t+1} \rangle = \left\langle \ell_t, \frac{1}{2c} \ell_t \right\rangle = \frac{1}{2c} \|\ell_t\|^2.$$

- So, with respect to a solution \mathbf{x} ,

$$\begin{aligned} \text{regret}_T(\mathbf{x}) &\leq R(\mathbf{x}) - R(\mathbf{x}_1) + \sum_{t=1}^T f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) \\ &= c \|\mathbf{x}\|^2 + \frac{1}{2c} \sum_{t=1}^T \|\ell_t\|^2. \end{aligned}$$

- Suppose that $\|\ell_t\| \leq L$ for each t and $\|\mathbf{x}\| \leq D$. Then by optimizing $c = \sqrt{\frac{T}{2D^2L^2}}$, we have

$$\text{regret}_T(\mathbf{x}) \leq DL\sqrt{2T}.$$



Dealing with constraints

- Let's deal with the constraint that \mathcal{K} is an arbitrary convex set instead of \mathbb{R}^n .
- Using the same regularizer, we have our FTRL which gives

$$\mathbf{x}_1 = \arg \min_{\mathbf{x} \in \mathcal{K}} c \|\mathbf{x}\|^2,$$

$$\mathbf{x}_{t+1} = \arg \min_{\mathbf{x} \in \mathcal{K}} c \|\mathbf{x}\|^2 + \sum_{k=1}^t \langle \ell_k, \mathbf{x} \rangle.$$



Dealing with constraints

- Let's deal with the constraint that \mathcal{K} is an arbitrary convex set instead of \mathbb{R}^n .
- Using the same regularizer, we have our FTRL which gives

$$\mathbf{x}_1 = \arg \min_{\mathbf{x} \in \mathcal{K}} c \|\mathbf{x}\|^2,$$

$$\mathbf{x}_{t+1} = \arg \min_{\mathbf{x} \in \mathcal{K}} c \|\mathbf{x}\|^2 + \sum_{k=1}^t \langle \ell_t, \mathbf{x} \rangle.$$

- The idea:** First solve the unconstrained optimization and then project the solution on \mathcal{K} .



Unconstrained optimization + projection

$$\mathbf{y}_{t+1} = \arg \min_{\mathbf{y} \in \mathbb{R}^n} c \|\mathbf{y}\|^2 + \sum_{k=1}^t \langle \ell_k, \mathbf{y} \rangle.$$

$$\mathbf{x}'_{t+1} = \Pi_{\mathcal{K}}(\mathbf{y}_{t+1}) = \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}_{t+1}\|.$$

Unconstrained optimization + projection

$$\mathbf{y}_{t+1} = \arg \min_{\mathbf{y} \in \mathbb{R}^n} c \|\mathbf{y}\|^2 + \sum_{k=1}^t \langle \ell_k, \mathbf{y} \rangle.$$

$$\mathbf{x}'_{t+1} = \Pi_{\mathcal{K}}(\mathbf{y}_{t+1}) = \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}_{t+1}\|.$$

- **Claim:** $\mathbf{x}'_{t+1} = \mathbf{x}_{t+1}$.



Proof of the claim: $\mathbf{x}'_{t+1} = \mathbf{x}_{t+1}$

- First, we already have that $\mathbf{y}_{t+1} = -\frac{1}{2c} \sum_{k=1}^t \ell_t$.
- Then,

$$\begin{aligned}\mathbf{x}'_{t+1} &= \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}_{t+1}\| = \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}_{t+1}\|^2 \\ &= \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x}\|^2 - 2\langle \mathbf{x}, \mathbf{y}_{t+1} \rangle + \|\mathbf{y}_{t+1}\|^2\end{aligned}$$

Proof of the claim: $\mathbf{x}'_{t+1} = \mathbf{x}_{t+1}$

- First, we already have that $\mathbf{y}_{t+1} = -\frac{1}{2c} \sum_{k=1}^t \ell_t$.
- Then,

$$\begin{aligned}\mathbf{x}'_{t+1} &= \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}_{t+1}\| = \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}_{t+1}\|^2 \\&= \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x}\|^2 - 2\langle \mathbf{x}, \mathbf{y}_{t+1} \rangle + \|\mathbf{y}_{t+1}\|^2 \\&= \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x}\|^2 - 2\langle \mathbf{x}, \mathbf{y}_{t+1} \rangle \\&= \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x}\|^2 - 2 \left\langle \mathbf{x}, -\frac{1}{2c} \sum_{k=1}^t \ell_t \right\rangle \\&= \arg \min_{\mathbf{x} \in \mathcal{K}} c\|\mathbf{x}\|^2 + \left\langle \mathbf{x}, \sum_{k=1}^t \ell_t \right\rangle \\&= \mathbf{x}_{t+1}.\end{aligned}$$



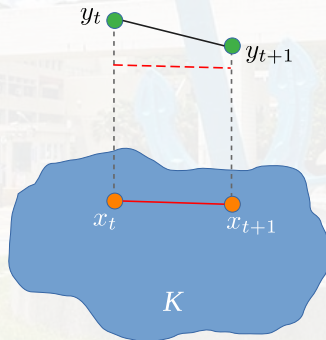
To bound the regret

$$f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) = \langle \ell_t, \mathbf{x}_t - \mathbf{x}_{t+1} \rangle \leq \|\ell_t\| \cdot \|\mathbf{x}_t - \mathbf{x}_{t+1}\|$$



To bound the regret

$$\begin{aligned} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) &= \langle \ell_t, \mathbf{x}_t - \mathbf{x}_{t+1} \rangle \leq \|\ell_t\| \cdot \|\mathbf{x}_t - \mathbf{x}_{t+1}\| \\ &\leq \|\ell_t\| \cdot \|\mathbf{y}_t - \mathbf{y}_{t+1}\|. \end{aligned}$$



To bound the regret

$$\begin{aligned} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) &= \langle \ell_t, \mathbf{x}_t - \mathbf{x}_{t+1} \rangle \leq \|\ell_t\| \cdot \|\mathbf{x}_t - \mathbf{x}_{t+1}\| \\ &\leq \|\ell_t\| \cdot \|\mathbf{y}_t - \mathbf{y}_{t+1}\| \\ &\leq \frac{1}{2c} \|\ell_t\|^2. \end{aligned}$$

So, assume $\max_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x}\| \leq D$ and $\|\ell_t\| \leq L$ for all t , we have

$$\begin{aligned} \text{regret}_T &\leq c\|\mathbf{x}^*\|^2 - c\|\mathbf{x}_1\|^2 + \frac{1}{2c} \sum_{t=1}^T \|\ell_t\|^2 \\ &\leq cD^2 + \frac{1}{2c} TL^2 \end{aligned}$$



To bound the regret

$$\begin{aligned} f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) &= \langle \ell_t, \mathbf{x}_t - \mathbf{x}_{t+1} \rangle \leq \|\ell_t\| \cdot \|\mathbf{x}_t - \mathbf{x}_{t+1}\| \\ &\leq \|\ell_t\| \cdot \|\mathbf{y}_t - \mathbf{y}_{t+1}\| \\ &\leq \frac{1}{2c} \|\ell_t\|^2. \end{aligned}$$

So, assume $\max_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x}\| \leq D$ and $\|\ell_t\| \leq L$ for all t , we have

$$\begin{aligned} \text{regret}_T &\leq c\|\mathbf{x}^*\|^2 - c\|\mathbf{x}_1\|^2 + \frac{1}{2c} \sum_{t=1}^T \|\ell_t\|^2 \\ &\leq cD^2 + \frac{1}{2c} TL^2 \leq DL\sqrt{2T}. \end{aligned}$$



Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)
- 3 Multiplicative Weight Update (MWU)
- 4 Follow The Leader (FTL)
- 5 Follow The Regularized Leader (FTRL)
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)**
 - Greedy Algorithms
 - Upper Confidence Bound (UCB)
 - Time-Decay ϵ -Greedy



Multi-Armed Bandit

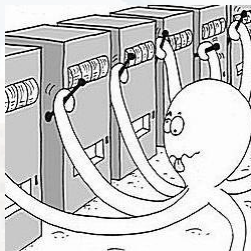


Fig.: Image credit: Microsoft Research

The setting

- We can see N arms as N experts.
- Arms give are independent.
- We can only pull an arm and observe the reward of it.
 - It's NOT possible to observe the reward of pulling the other arms...
- Each arm i has its own reward $r_i \in [0, 1]$.



The setting

- We can see N arms as N experts.
- Arms give are independent.
- We can only pull an arm and observe the reward of it.
 - It's NOT possible to observe the reward of pulling the other arms...
- Each arm i has its own reward $r_i \in [0, 1]$.
 - μ_i : the mean of reward of arm i
 - $\hat{\mu}_i$: the empirical mean of reward of arm i
 - μ^* : the mean of reward of the BEST arm.
 - $\Delta_i : \mu^* - \mu_i$.
 - Index of the best arm: $I^* := \arg \max_{i \in \{1, \dots, N\}} \mu_i$.
 - The associated highest expected reward: $\mu^* = \mu_{I^*}$.



The regret formulation for MAB

Let I_t be the arm played by the algorithm at time t .
The regret of the algorithm in T rounds is

$$\text{regret}_T = \sum_{t=1}^T (\mu^* - \mu_{I_t})$$



The regret formulation for MAB

Let I_t be the arm played by the algorithm at time t .
The regret of the algorithm in T rounds is

$$\text{regret}_T = \sum_{t=1}^T (\mu^* - \mu_{I_t}) = \sum_{i=1}^N \sum_{t: I_t=i} (\mu^* - \mu_i)$$



The regret formulation for MAB

Let I_t be the arm played by the algorithm at time t .
The regret of the algorithm in T rounds is

$$\begin{aligned}\text{regret}_T &= \sum_{t=1}^T (\mu^* - \mu_{I_t}) = \sum_{i=1}^N \sum_{t: I_t=i} (\mu^* - \mu_i) \\ &= \sum_{i=1}^N n_{i,T} \Delta_i\end{aligned}$$



The regret formulation for MAB

Let I_t be the arm played by the algorithm at time t .

The regret of the algorithm in T rounds is

$$\begin{aligned}\text{regret}_T &= \sum_{t=1}^T (\mu^* - \mu_{I_t}) = \sum_{i=1}^N \sum_{t: I_t=i} (\mu^* - \mu_i) \\ &= \sum_{i=1}^N n_{i,T} \Delta_i \\ &= \sum_{i: \mu_i < \mu^*} n_{i,T} \Delta_i.\end{aligned}$$

Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)
- 3 Multiplicative Weight Update (MWU)
- 4 Follow The Leader (FTL)
- 5 Follow The Regularized Leader (FTRL)
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)**
 - Greedy Algorithms**
 - Upper Confidence Bound (UCB)
 - Time-Decay ϵ -Greedy



A Naïve Greedy Algorithm

Greedy Algorithm

- ➊ For $t \leq cN$, select a random arm with probability $1/N$ and pull it.
 - ➋ For $t > cN$, pull the arm $I_t := \arg \max_{i=1,\dots,N} \hat{\mu}_{i,t}$.
- Here c is a constant.



A Naïve Greedy Algorithm

Greedy Algorithm

- ➊ For $t \leq cN$, select a random arm with probability $1/N$ and pull it.
 - ➋ For $t > cN$, pull the arm $I_t := \arg \max_{i=1,\dots,N} \hat{\mu}_{i,t}$.
- Here c is a constant.
 - This algorithm is of **linear** regret, hence is not a no-regret algorithm.



A Naïve Greedy Algorithm

Greedy Algorithm

- 1 For $t \leq cN$, select a random arm with probability $1/N$ and pull it.
 - 2 For $t > cN$, pull the arm $I_t := \arg \max_{i=1,\dots,N} \hat{\mu}_{i,t}$.
- Here c is a constant.
 - This algorithm is of **linear** regret, hence is not a no-regret algorithm.
 - For example,
 - Arm 1: 0/1 reward with mean $3/4$.
 - Arm 2: Fixed reward of $1/4$.
 - After $cN = 2c$ steps, with constant probability, we have $\hat{\mu}_{1,cN} < \hat{\mu}_{2,cN}$.



A Naïve Greedy Algorithm

Greedy Algorithm

- ➊ For $t \leq cN$, select a random arm with probability $1/N$ and pull it.
 - ➋ For $t > cN$, pull the arm $I_t := \arg \max_{i=1,\dots,N} \hat{\mu}_{i,t}$.
- Here c is a constant.
 - This algorithm is of **linear** regret, hence is not a no-regret algorithm.
 - For example,
 - Arm 1: 0/1 reward with mean $3/4$.
 - Arm 2: Fixed reward of $1/4$.
 - After $cN = 2c$ steps, with constant probability, we have $\hat{\mu}_{1,cN} < \hat{\mu}_{2,cN}$.
 - If this is the case, the algorithm will keep pulling arm 2 and will never change!



ϵ -Greedy Algorithm

ϵ -Greedy Algorithm

For all $t = 1, 2, \dots, N$:

- With probability $1 - \epsilon$, pull arm $I_t := \arg \max_{i=1, \dots, N} \hat{\mu}_{i,t}$.
- With probability ϵ , select an arm uniformly at random (i.e., each with probability $1/N$).



ϵ -Greedy Algorithm

ϵ -Greedy Algorithm

For all $t = 1, 2, \dots, N$:

- With probability $1 - \epsilon$, pull arm $I_t := \arg \max_{i=1, \dots, N} \hat{\mu}_{i,t}$.
- With probability ϵ , select an arm uniformly at random (i.e., each with probability $1/N$).
- It looks good.



ϵ -Greedy Algorithm

ϵ -Greedy Algorithm

For all $t = 1, 2, \dots, N$:

- With probability $1 - \epsilon$, pull arm $I_t := \arg \max_{i=1, \dots, N} \hat{\mu}_{i,t}$.
 - With probability ϵ , select an arm uniformly at random (i.e., each with probability $1/N$).
-
- It looks good.
 - Unfortunately, this algorithm still incurs **linear** regret.



ϵ -Greedy Algorithm

ϵ -Greedy Algorithm

For all $t = 1, 2, \dots, N$:

- With probability $1 - \epsilon$, pull arm $I_t := \arg \max_{i=1, \dots, N} \hat{\mu}_{i,t}$.
 - With probability ϵ , select an arm uniformly at random (i.e., each with probability $1/N$).
-
- It looks good.
 - Unfortunately, this algorithm still incurs **linear** regret.
 - Indeed,
 - Each arm is pulled in average $\epsilon T/N$ times.



ϵ -Greedy Algorithm

ϵ -Greedy Algorithm

For all $t = 1, 2, \dots, N$:

- With probability $1 - \epsilon$, pull arm $I_t := \arg \max_{i=1, \dots, N} \hat{\mu}_{i,t}$.
 - With probability ϵ , select an arm uniformly at random (i.e., each with probability $1/N$).
-
- It looks good.
 - Unfortunately, this algorithm still incurs **linear** regret.
 - Indeed,
 - Each arm is pulled in average $\epsilon T/N$ times.
 - Hence the (expected) regret will be at least $\frac{\epsilon T}{N} \sum_{i: \mu_i < \mu^*} \Delta_i$.



Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)
- 3 Multiplicative Weight Update (MWU)
- 4 Follow The Leader (FTL)
- 5 Follow The Regularized Leader (FTRL)
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)**
 - Greedy Algorithms
 - Upper Confidence Bound (UCB)**
 - Time-Decay ϵ -Greedy



The upper confidence bound algorithm (UCB)

- At each time step (round), we simply pull the arm with the highest “empirical reward estimate + high-confidence interval size”.
- The empirical reward estimate of arm i at time t :

$$\hat{\mu}_{i,t} = \frac{\sum_{s=1}^t I_{s,i} \cdot r_s}{n_{i,t}}.$$

$n_{i,t}$: the number of times arm i is played.

$I_{s,i}$: 1 if the choice of arm is i at time s and 0 otherwise.

- Reward estimate + confidence interval:

$$\text{UCB}_{i,t} := \hat{\mu}_{i,t} + \sqrt{\frac{\ln t}{n_{i,t}}}.$$



Algorithm UCB

UCB Algorithm

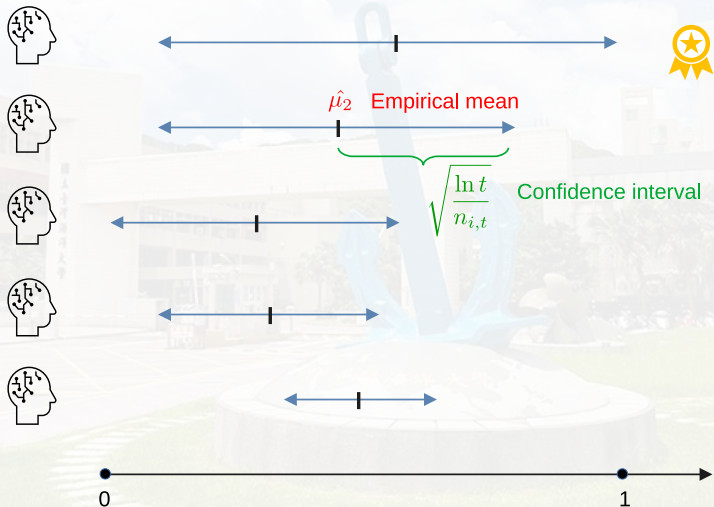
N arms, T rounds such that $T \geq N$.

- ① For $t = 1, \dots, N$, play arm t .
- ② For $t = N + 1, \dots, T$, play arm

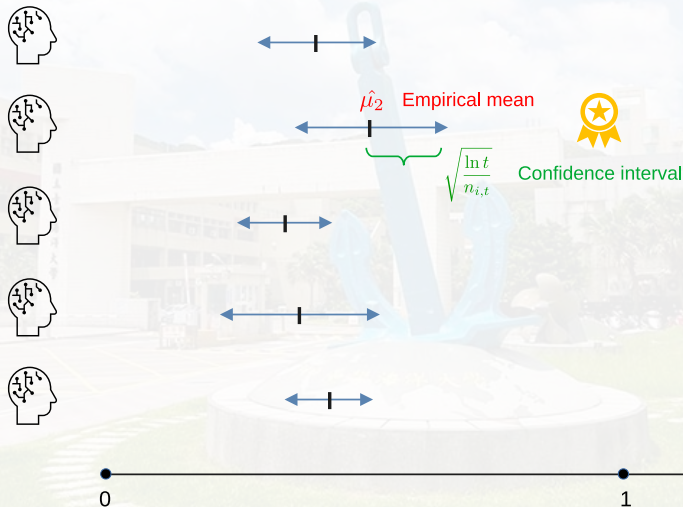
$$A_t = \arg \max_{i \in \{1, \dots, N\}} \text{UCB}_{i,t-1}.$$



Algorithm UCB



Algorithm UCB (after more time steps...)



From the Chernoff bound (proof skipped)

For each arm i at time t , we have

$$|\hat{\mu}_{i,t} - \mu_i| < \sqrt{\frac{\ln t}{n_{i,t}}}$$

with probability $\geq 1 - 2/t^2$.

Immediately, we know that

- with prob. $\geq 1 - 2/t^2$, $\text{UCB}_{i,t} := \hat{\mu}_{i,t} + \sqrt{\frac{\ln t}{n_{i,t}}} > \mu_i$.
- with prob. $\geq 1 - 2/t^2$, $\hat{\mu}_{i,t} < \mu_i + \frac{\Delta_i}{2}$ when $n_{i,t} \geq \frac{4 \ln t}{\Delta_i^2}$.



From the Chernoff bound (proof skipped)

For each arm i at time t , we have

$$|\hat{\mu}_{i,t} - \mu_i| < \sqrt{\frac{\ln t}{n_{i,t}}}$$

with probability $\geq 1 - 2/t^2$.

To understand why, please take my Randomized Algorithms course. :)
Immediately, we know that

- with prob. $\geq 1 - 2/t^2$, $\text{UCB}_{i,t} := \hat{\mu}_{i,t} + \sqrt{\frac{\ln t}{n_{i,t}}} > \mu_i$.
- with prob. $\geq 1 - 2/t^2$, $\hat{\mu}_{i,t} < \mu_i + \frac{\Delta_i}{2}$ when $n_{i,t} \geq \frac{4 \ln t}{\Delta_i^2}$.

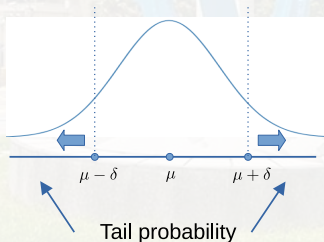


Appendix: Tail probability by the Chernoff/Hoeffding bound

The Chernoff/Hoeffding bound

For independent and identically distributed (i.i.d.) samples $x_1, \dots, x_n \in [0, 1]$ with $\mathbb{E}[x_i] = \mu$, we have

$$\Pr \left[\left| \frac{\sum_{i=1}^n x_i}{n} - \mu \right| \geq \delta \right] \leq 2e^{-2n\delta^2}.$$



Very unlikely to play a suboptimal arm

Lemma 3

At any time step t , if a suboptimal arm i (i.e., $\mu_i < \mu^*$) has been played for $n_{i,t} \geq \frac{4 \ln t}{\Delta_i^2}$ times, then $\text{UCB}_{i,t} < \text{UCB}_{I^*,t}$ with probability $\geq 1 - 4/t^2$. Therefore, for any t ,

$$\Pr \left[I_{t+1,i} = 1 \mid n_{i,t} \geq \frac{4 \ln t}{\Delta_i^2} \right] \leq \frac{4}{t^2}.$$

Proof of Lemma 3

With probability $< 2/t^2 + 2/t^2$ (union bound) that

$$\begin{aligned} \text{UCB}_{i,t} = \hat{\mu}_{i,t} + \sqrt{\frac{\ln t}{n_{i,t}}} &\leq \hat{\mu}_{i,t} + \frac{\Delta_i}{2} \\ &< \left(\mu_i + \frac{\Delta_i}{2} \right) + \frac{\Delta_i}{2} \\ &= \mu^* < \text{UCB}_{i^*,t} \end{aligned}$$

does NOT hold.

Playing suboptimal arms for very limited number of times

Lemma 4

For any arm i with $\mu_i < \mu^*$,

$$\mathbb{E}[n_{i,T}] \leq \frac{4 \ln T}{\Delta_i^2} + 8.$$

$$\begin{aligned}\mathbb{E}[n_{i,T}] &= 1 + \mathbb{E} \left[\sum_{t=N}^T \mathbb{1} \{I_{t+1,i} = 1\} \right] \\ &= 1 + \mathbb{E} \left[\sum_{t=N}^T \mathbb{1} \left\{ I_{t+1,i} = 1, n_{i,t} < \frac{4 \ln t}{\Delta_i^2} \right\} \right] \\ &\quad + \mathbb{E} \left[\sum_{t=N}^T \mathbb{1} \left\{ I_{t+1,i} = 1, n_{i,t} \geq \frac{4 \ln t}{\Delta_i^2} \right\} \right]\end{aligned}$$

Proof of Lemma 4 (contd.)

$$\begin{aligned}\mathbb{E}[n_{i,T}] &\leq \frac{4 \ln T}{\Delta_i^2} + \mathbb{E} \left[\sum_{t=N}^T \mathbb{1} \left\{ I_{t+1,i} = 1, n_{i,t} \geq \frac{4 \ln t}{\Delta_i^2} \right\} \right] \\&= \frac{4 \ln T}{\Delta_i^2} + \sum_{t=N}^T \Pr \left[I_{t+1,i} = 1, n_{i,t} \geq \frac{4 \ln t}{\Delta_i^2} \right] \\&= \frac{4 \ln T}{\Delta_i^2} + \sum_{t=N}^T \Pr \left[I_{t+1,i} = 1 \mid n_{i,t} \geq \frac{4 \ln t}{\Delta_i^2} \right] \cdot \Pr \left[n_{i,t} \geq \frac{4 \ln t}{\Delta_i^2} \right] \\&\leq \frac{4 \ln T}{\Delta_i^2} + \sum_{t=N}^T \frac{4}{t^2} \\&\leq \frac{4 \ln T}{\Delta_i^2} + 8.\end{aligned}$$

The regret bound for the UCB algorithm

Theorem 4

For all $T \geq N$, the (expected) regret by the UCB algorithm in round T is

$$\mathbb{E}[\text{regret}_T] \leq 5\sqrt{NT \ln T} + 8N.$$

Proof of Theorem 4

- Divide the arms into two groups:

- ① Group ONE (G_1): “almost optimal arms” with $\Delta_i < \sqrt{\frac{N}{T} \ln T}$.
- ② Group TWO (G_2): “bad” arms with $\Delta_i \geq \sqrt{\frac{N}{T} \ln T}$.

$$\sum_{i \in G_1} n_{i,T} \Delta_i \leq \left(\sqrt{\frac{N}{T} \ln T} \right) \sum_{i \in G_1} n_{i,T} \leq T \cdot \sqrt{\frac{N}{T} \ln T} = \sqrt{NT \ln T}.$$

By Lemma 4,

$$\begin{aligned} \sum_{i \in G_2} \mathbb{E}[n_{i,T}] \Delta_i &\leq \sum_{i \in G_2} \frac{4 \ln T}{\Delta_i} + 8 \Delta_i \leq \sum_{i \in G_2} 4 \sqrt{\frac{T \ln T}{N}} + 8 \\ &\leq 4 \sqrt{NT \ln T} + 8N. \end{aligned}$$



Outline

- 1 Introduction
- 2 Gradient Descent for Online Convex Optimization (GD)
- 3 Multiplicative Weight Update (MWU)
- 4 Follow The Leader (FTL)
- 5 Follow The Regularized Leader (FTRL)
 - MWU Revisited
 - FTRL with 2-norm regularizer
- 6 Multi-Armed Bandit (MAB)**
 - Greedy Algorithms
 - Upper Confidence Bound (UCB)
 - Time-Decay ϵ -Greedy



Time Decaying ϵ -Greedy Algorithm

What if the horizon T is known in advance when we run ϵ -Greedy?

Time-Decaying ϵ -Greedy Algorithm

For all $t = 1, 2, \dots, N$, set $\epsilon := N^{1/3} / T^{1/3}$:

- With probability $1 - \epsilon$, pull arm $I_t := \arg \max_{i=1, \dots, N} \hat{\mu}_{i,t}$.
- With probability ϵ , select an arm uniformly at random (i.e., each with probability $1/N$).



Time Decaying ϵ -Greedy Algorithm

What if the horizon T is known in advance when we run ϵ -Greedy?

Time-Decaying ϵ -Greedy Algorithm

For all $t = 1, 2, \dots, N$, set $\epsilon := N^{1/3} / T^{1/3}$:

- With probability $1 - \epsilon$, pull arm $I_t := \arg \max_{i=1, \dots, N} \hat{\mu}_{i,t}$.
- With probability ϵ , select an arm uniformly at random (i.e., each with probability $1/N$).

Theorem

Time-Decaying ϵ -Greedy Algorithm gets roughly $O(N^{1/3} T^{2/3})$ regret.





Thank you.

