No-Regret Online Learning Algorithms

Joseph Chuang-Chieh Lin

Department of Computer Science and Information Engineering,
Tamkang University

Lecture Notes

27 December 2021 – 11 Jan 2022

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

And we want to minimize the cost.

The difficulty

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

What's the regret?

- The offline optimum: After T steps,

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

- The regret after T steps:

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

What's the regret?

- The offline optimum: After T steps,

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

- The regret after T steps:

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

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

What's the regret?

- The offline optimum: After T steps,

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

- The regret after T steps:

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

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

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 $\mathcal{K}$:

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

Convex set

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

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

Prerequisites (2/5)

Convex function

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

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

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

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

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 $\boldsymbol{x}$, then for all $\boldsymbol{y} \in \mathbb{R}^d$,

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

Subgradient

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

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

Prerequisites (4/5)

Projection

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

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

Pythagoras Theorem

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

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

Prerequisites (5/5)

Minimum vs. zero gradient

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

Karush-Kuhn-Tucker (KKT) Theorem

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

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

Convex losses to linear losses

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

Convex losses to linear losses

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

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

Convex losses to linear losses

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

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

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, $\boldsymbol{x}_1 \in \mathcal{K}$, step size $\{\eta_t\}$.
2. **for** $t \leftarrow 1$ to T **do**:
   1. Play $\boldsymbol{x}_t$ and observe cost $f_t(\boldsymbol{x}_t)$.
   2. Update and Project:

$$
\begin{aligned}
\boldsymbol{y}_{t+1} &= \boldsymbol{x}_t - \eta_t \nabla f_t(\boldsymbol{x}_t) \\
\boldsymbol{x}_{t+1} &= \Pi_{\mathcal{K}}(\boldsymbol{y}_{t+1})
\end{aligned}
$$

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 $T \geq 1$:

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

Proof of Theorem A (1/3)

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

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

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

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

Proof of Theorem A (2/3)

- Hence

$$||\boldsymbol{x}_{t+1} - \boldsymbol{x}^*||^2 \leq ||\boldsymbol{x}_t - \boldsymbol{x}^*||^2 + \eta_t^2 ||\nabla f_t(\boldsymbol{x}_t)||^2 - 2\eta_t (\nabla f_t(\boldsymbol{x}_t))^\top (\boldsymbol{x}_t - \boldsymbol{x}^*)$$

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

- 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)

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

The Lower Bound

Theorem B

Let $\mathcal{K} = \{\boldsymbol{x} \in \mathbb{R}^d : ||\boldsymbol{x}||_\infty \leq r\}$ 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 $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_T$ with $||\boldsymbol{g}_t||_2 \leq L$ and $\boldsymbol{u} \in \mathcal{K}$ such that the regret of A satisfies

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

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

Proof of Theorem B (1/2)

- The approach:

  For any random variable $\boldsymbol{z}$ with domain $\mathcal{V}$ and any function f,

$$\sup_{\boldsymbol{x} \in V} f(\boldsymbol{x}) \geq E[f(\boldsymbol{z})].$$

Proof of Theorem B (1/2)

- The approach:

  For any random variable $\boldsymbol{z}$ with domain $\mathcal{V}$ and any function f,

$$\sup_{\boldsymbol{x} \in V} f(\boldsymbol{x}) \geq E[f(\boldsymbol{z})].$$

- $\mathrm{regret}_T = \max_{\boldsymbol{u} \in \mathcal{K}} \mathrm{regret}_T(\boldsymbol{u})$.

Proof of Theorem B (1/2)

- The approach:

  For any random variable $\boldsymbol{z}$ with domain $\mathcal{V}$ and any function f,

$$\sup_{\boldsymbol{x} \in V} f(\boldsymbol{x}) \geq E[f(\boldsymbol{z})].$$

- $\text{regret}_T = \max_{\boldsymbol{u} \in \mathcal{K}} \text{regret}_T(\boldsymbol{u})$.
- Let $\boldsymbol{v}, \boldsymbol{w} \in \mathcal{K}$ such that $||\boldsymbol{v} - \boldsymbol{w}|| = D$.

Proof of Theorem B (1/2)

- The approach:

  For any random variable $\boldsymbol{z}$ with domain $\mathcal{V}$ and any function f,

$$\sup_{\boldsymbol{x} \in V} f(\boldsymbol{x}) \geq E[f(\boldsymbol{z})].$$

- $\text{regret}_T = \max_{\boldsymbol{u} \in \mathcal{K}} \text{regret}_T(\boldsymbol{u})$.
- Let $\boldsymbol{v}, \boldsymbol{w} \in \mathcal{K}$ such that $||\boldsymbol{v} - \boldsymbol{w}|| = D$.
- Let $\boldsymbol{z} := \frac{\boldsymbol{v} - \boldsymbol{w}}{||\boldsymbol{v} - \boldsymbol{w}||}$

Proof of Theorem B (1/2)

- The approach:

  For any random variable $\boldsymbol{z}$ with domain $\mathcal{V}$ and any function f,

$$\sup_{\boldsymbol{x}\in V} f(\boldsymbol{x}) \geq E[f(\boldsymbol{z})].$$

- $\text{regret}_T = \max_{\boldsymbol{u}\in\mathcal{K}} \text{regret}_T(\boldsymbol{u})$.
- Let $\boldsymbol{v}, \boldsymbol{w} \in \mathcal{K}$ such that $||\boldsymbol{v} - \boldsymbol{w}|| = D$.
- Let $\boldsymbol{z} := \frac{\boldsymbol{v}-\boldsymbol{w}}{||\boldsymbol{v}-\boldsymbol{w}||} \Rightarrow \langle \boldsymbol{z}, \boldsymbol{v} - \boldsymbol{w}\rangle = D$.
- Let $\epsilon_1, \epsilon_2, \ldots, \epsilon_T$ be i.i.d. random variables such that $\Pr[\epsilon_t = 1] = \Pr[\epsilon_t = -1] = 1/2$ for each t.

Proof of Theorem B (1/2)

- The approach:

  For any random variable $\boldsymbol{z}$ with domain $\mathcal{V}$ and any function f,

$$\sup_{\boldsymbol{x} \in V} f(\boldsymbol{x}) \geq E[f(\boldsymbol{z})].$$

- $\text{regret}_T = \max_{\boldsymbol{u} \in \mathcal{K}} \text{regret}_T(\boldsymbol{u})$.
- Let $\boldsymbol{v}, \boldsymbol{w} \in \mathcal{K}$ such that $||\boldsymbol{v} - \boldsymbol{w}|| = D$.
- Let $\boldsymbol{z} := \frac{\boldsymbol{v}-\boldsymbol{w}}{||\boldsymbol{v}-\boldsymbol{w}||} \Rightarrow \langle \boldsymbol{z}, \boldsymbol{v} - \boldsymbol{w} \rangle = D$.
- Let $\epsilon_1, \epsilon_2, \ldots, \epsilon_T$ be i.i.d. random variables such that $\Pr[\epsilon_t = 1] = \Pr[\epsilon_t = -1] = 1/2$ for each t.
- We choose the losses $\boldsymbol{g}_t = L\epsilon_t \boldsymbol{z}$.

Proof of Theorem B (1/2)

- The approach:

  For any random variable $\boldsymbol{z}$ with domain $\mathcal{V}$ and any function f,

$$\sup_{\boldsymbol{x} \in V} f(\boldsymbol{x}) \geq E[f(\boldsymbol{z})].$$

- $\text{regret}_T = \max_{\boldsymbol{u} \in \mathcal{K}} \text{regret}_T(\boldsymbol{u})$.
- Let $\boldsymbol{v}, \boldsymbol{w} \in \mathcal{K}$ such that $||\boldsymbol{v} - \boldsymbol{w}|| = D$.
- Let $\boldsymbol{z} := \frac{\boldsymbol{v}-\boldsymbol{w}}{||\boldsymbol{v}-\boldsymbol{w}||} \Rightarrow \langle \boldsymbol{z}, \boldsymbol{v} - \boldsymbol{w} \rangle = D$.
- Let $\epsilon_1, \epsilon_2, \ldots, \epsilon_T$ be i.i.d. random variables such that $\Pr[\epsilon_t = 1] = \Pr[\epsilon_t = -1] = 1/2$ for each t.
- We choose the losses $\boldsymbol{g}_t = L\epsilon_t \boldsymbol{z}$.
  - The cost at t: $\langle L\epsilon_t \boldsymbol{z}, \boldsymbol{x}_t \rangle$.
  - $||g_t|| = \sqrt{L^2 \epsilon_t^2} \cdot ||\boldsymbol{z}|| \leq L$.

Proof of Theorem B (2/2)

$$\begin{aligned}
\sup_{\boldsymbol{g}_1,\ldots,\boldsymbol{g}_T} \text{regret}_T &\geq E\left[\sum_{t=1}^{T} L\epsilon_t \langle \boldsymbol{z}, \boldsymbol{x}_t \rangle - \min_{\boldsymbol{u}\in\mathcal{K}} \sum_{t=1}^{T} L\epsilon_t \langle \boldsymbol{z}, \boldsymbol{u} \rangle\right] \\
&= E\left[-\min_{\boldsymbol{u}\in\mathcal{K}} \sum_{t=1}^{T} L\epsilon_t \langle \boldsymbol{z}, \boldsymbol{u} \rangle\right] = E\left[\max_{\boldsymbol{u}\in\mathcal{K}} \sum_{t=1}^{T} L\epsilon_t \langle \boldsymbol{z}, \boldsymbol{u} \rangle\right] \\
&\geq E\left[\max_{\boldsymbol{u}\in\{\boldsymbol{v},\boldsymbol{w}\}} \sum_{t=1}^{T} L\epsilon_t \langle \boldsymbol{z}, \boldsymbol{u} \rangle\right] \\
&= E\left[\frac{1}{2}\sum_{t=1}^{T} L\epsilon_t \langle \boldsymbol{z}, \boldsymbol{v}+\boldsymbol{w} \rangle + \frac{1}{2}\left|\sum_{t=1}^{T} L\epsilon_t \langle \boldsymbol{z}, \boldsymbol{v}-\boldsymbol{w} \rangle\right|\right] \\
&\geq \frac{L}{2} E\left[\left|\sum_{t=1}^{T} L\epsilon_t \langle \boldsymbol{z}, \boldsymbol{v}-\boldsymbol{w} \rangle\right|\right] = \frac{LD}{2} E\left[\left|\sum_{t=1}^{T} \epsilon_t\right|\right] \\
&\geq \frac{\sqrt{2}LD\sqrt{T}}{4}. \quad \text{(by Khintchine inequality)}
\end{aligned}$$

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

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.
  - $\boldsymbol{x}_t = (\boldsymbol{x}_t(1), \boldsymbol{x}_t(2), \ldots, \boldsymbol{x}_t(n))$, where $\boldsymbol{x}_t(i) \in [0, 1]$ and $\sum_i \boldsymbol{x}_t(i) = 1$.

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.
  - $\boldsymbol{x}_t = (\boldsymbol{x}_t(1), \boldsymbol{x}_t(2), \ldots, \boldsymbol{x}_t(n))$, where $\boldsymbol{x}_t(i) \in [0, 1]$ and $\sum_i \boldsymbol{x}_t(i) = 1$.
- The loss of following expert i at time t: $\ell_t(i)$.
- The expected loss of the algorithm at time t:

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

The regret of listening to the experts...

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

- The set of feasible solutions $K = \Delta \subseteq \mathbb{R}^n$, probability distributions over $\{1, \ldots, n\}$.
- $f_t(\boldsymbol{x}) = \sum_i \boldsymbol{x}(i)\ell_t(i)$: 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 $\boldsymbol{w}_t = (\boldsymbol{w}_t(1), \ldots, \boldsymbol{w}_t(n))$ where $\boldsymbol{w}_1 := (1, 1, \ldots, 1)$.
- Update the weights at time t by
  - $\boldsymbol{w}_t(i) := \boldsymbol{w}_{t-1}(i) \cdot e^{-\beta \boldsymbol{\ell}_{t-1}(i)}$.
  - $\boldsymbol{x}_t := \frac{\boldsymbol{w}_t(i)}{\sum_{j=1}^{n} \boldsymbol{w}_t(j)}$.

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 $\boldsymbol{w}_t = (\boldsymbol{w}_t(1), \ldots, \boldsymbol{w}_t(n))$ where $\boldsymbol{w}_1 := (1, 1, \ldots, 1)$.
- Update the weights at time t by
  - $\boldsymbol{w}_t(i) := \boldsymbol{w}_{t-1}(i) \cdot e^{-\beta \boldsymbol{\ell}_{t-1}(i)}$.
  - $\boldsymbol{x}_t := \frac{\boldsymbol{w}_t(i)}{\sum_{j=1}^{n} \boldsymbol{w}_t(j)}$.

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

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

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

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

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)

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

Proof.

Note: $W_1 = n$.

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

The proof (contd.)

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

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

Proof.

Note: $W_1 = n$.

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

$$\leq \sum_{i=1}^{n} \boldsymbol{x}_t(i) \cdot (1 - \beta \boldsymbol{\ell}_t(i) + \beta^2 \boldsymbol{\ell}_t^2(i))$$

The proof (contd.)

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

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

Proof.

Note: $W_1 = n$.

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

The proof (contd.)

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

$$W_{T+1} \le n \prod_{t=1}^{n} (1 - \beta\langle \boldsymbol{x}_t, \boldsymbol{\ell}_t \rangle + \beta^2 \langle \boldsymbol{x}_t, \boldsymbol{\ell}_t^2 \rangle),$$

Proof.

Note: $W_1 = n$.

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

□

The proof (contd.)

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

$$W_{T+1} \le n \prod_{t=1}^{n} e^{-\beta\langle x_t, \ell_t\rangle + \beta^2\langle x_t, \ell_t^2\rangle}.$$

Proof.

Note: $W_1 = n$.

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

□

The proof (contd.)

Hence

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

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

The proof (contd.)

Hence

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

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

Thus,

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

The proof (contd.)

Hence

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

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

Thus,

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

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 $f_1, \ldots, f_{t-1}$, and then give the solution

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

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 $f_1, \ldots, f_{t-1}$, and then give the solution

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

  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 $f_1, \ldots, f_{t-1}$, and then give the solution

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

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

FTL leads to "overfitting"

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

FTL leads to "overfitting"

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

FTL leads to "overfitting"

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

FTL leads to "overfitting"

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

FTL leads to "overfitting"

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

FTL leads to “overfitting”

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

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 $f_1, \ldots, f_t$ and any number of time steps T, the FTL algorithm satisfies

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

Analysis of FTL

Theorem 2 (Analysis of FTL)

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

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

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

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

Proof of Theorem 2

Recall that

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

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

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

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

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

Proof of Theorem 2

Recall that

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

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

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

Assume that it holds up to T. Then:

Proof of Theorem 2

Recall that

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

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

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

Assume that it holds up to T. Then:

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

Proof of Theorem 2

Recall that

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

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

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

Assume that it holds up to T. Then:

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

where

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

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

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

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 $\boldsymbol{x}$, the regret with respect to $\boldsymbol{x}$ after T steps of the FTRL algorithm is bounded as

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

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

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: $R, f_1, f_2, \ldots, f_T$.
    - Use $\boldsymbol{x}_1$ as the first solution.
  - The solutions: $\boldsymbol{x}_1, \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$.

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, $\ldots$, f_T.
    - Use $\boldsymbol{x}_1$ as the first solution.
  - The solutions: $\boldsymbol{x}_1$, $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\ldots$, $\boldsymbol{x}_T$.
- The regret:

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

Proof of Theorem 3

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

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

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, . . . , f_T.
    - Use $\boldsymbol{x}_1$ as the first solution.
  - The solutions: $\boldsymbol{x}_1$, $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, . . . , $\boldsymbol{x}_T$.
- The regret:

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

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 "concentralized" 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 "concentralized" distributions.
  - *negative*-entropy: a good measure of how centralized a distribution is.

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

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 "concentralized" distributions.
  - *negative*-entropy: a good measure of how centralized a distribution is.

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

- So our FTRL gives

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

Using negative entropy regularization

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

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

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

Using negative entropy regularization

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

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

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

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

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

Rediscover MWU?

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

Rediscover MWU?

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

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

Rediscover MWU?

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

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

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

Rediscover MWU?

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

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

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

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

Rediscover MWU?

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

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

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

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

Regret of FTRL + Negative-Entropy Regularization

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

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

Regret of FTRL + Negative-Entropy Regularization

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

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

$\because$ weights are non-increasing

Regret of FTRL + Negative-Entropy Regularization

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

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

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

Regret of FTRL + Negative-Entropy Regularization

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

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

Regret of FTRL + Negative-Entropy Regularization

- By Theorem 3, for any $\boldsymbol{x}$,

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

Regret of FTRL + Negative-Entropy Regularization

- By Theorem 3, for any $\boldsymbol{x}$,

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

$\because$ max entropy for uniform distribution

Regret of FTRL + Negative-Entropy Regularization

- By Theorem 3, for any $\boldsymbol{x}$,

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

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 $\boldsymbol{x}$,

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

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.

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

The regularizer of 2-norm tells us...

- $\boldsymbol{x}_1 = \mathbf{0}$.
- $\boldsymbol{x}_{t+1} = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} c||\boldsymbol{x}||^2 + \sum_{k=1}^{t}\langle \boldsymbol{\ell}_k, \boldsymbol{x}\rangle$.

- Compute the gradient:

$$2c\boldsymbol{x} + \sum_{k=1}^{t} \boldsymbol{\ell}_k = 0$$

$$\Rightarrow \quad \boldsymbol{x} = -\frac{1}{2c}\sum_{k=1}^{t} \boldsymbol{\ell}_k.$$

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

The regularizer of 2-norm tells us...

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

$$
\begin{aligned}
& 2c\boldsymbol{x} + \sum_{k=1}^{t} \boldsymbol{\ell}_k = 0 \\
\Rightarrow \quad & \boldsymbol{x} = -\frac{1}{2c} \sum_{k=1}^{t} \boldsymbol{\ell}_k .
\end{aligned}
$$

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

The regularizer of 2-norm tells us...

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

$$2c\boldsymbol{x} + \sum_{k=1}^{t} \boldsymbol{\ell}_k = 0$$

$$\Rightarrow \quad \boldsymbol{x} = -\frac{1}{2c} \sum_{k=1}^{t} \boldsymbol{\ell}_k.$$

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

The regularizer of 2-norm tells us...

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

$$2c\boldsymbol{x} + \sum_{k=1}^{t} \boldsymbol{\ell}_k = 0$$

$$\Rightarrow \quad \boldsymbol{x} = -\frac{1}{2c}\sum_{k=1}^{t}\boldsymbol{\ell}_k.$$

Hence, $\boldsymbol{x}_1 = \mathbf{0}, \boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \frac{1}{2c}\boldsymbol{\ell}_t$.
$\rightarrow$ penalize the experts that performed badly in the past!

The regret of FTRL with 2-norm regularization

- First, we have

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

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

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

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

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

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

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

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

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

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

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

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

Unconstrained optimization + projection

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

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

Unconstrained optimization + projection

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

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

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

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

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

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

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

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

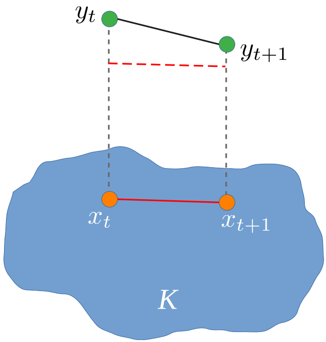
$$
\begin{aligned}
\boldsymbol{x}'_{t+1} &= \arg\min_{\boldsymbol{x}\in\mathcal{K}} ||\boldsymbol{x} - \boldsymbol{y}_{t+1}|| = \arg\min_{\boldsymbol{x}\in\mathcal{K}} ||\boldsymbol{x} - \boldsymbol{y}_{t+1}||^2 \\
&= \arg\min_{\boldsymbol{x}\in\mathcal{K}} ||\boldsymbol{x}||^2 - 2\langle \boldsymbol{x},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

To bound the regret

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

To bound the regret

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



To bound the regret

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

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

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

To bound the regret

$$f_t(\boldsymbol{x}_t) - f_t(\boldsymbol{x}_{t+1}) = \langle \boldsymbol{\ell}_t, \boldsymbol{x}_t - \boldsymbol{x}_{t+1} \rangle \leq ||\boldsymbol{\ell}_t|| \cdot ||\boldsymbol{x}_t - \boldsymbol{x}_{t+1}|| \leq ||\boldsymbol{\ell}_t|| \cdot ||\boldsymbol{y}_t - \boldsymbol{y}_{t+1}|| \leq \frac{1}{2c} ||\boldsymbol{\ell}_t||^2.$$

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

$$\text{regret}_T \leq c||\boldsymbol{x}^*||^2 - c||\boldsymbol{x}_1||^2 + \frac{1}{2c} \sum_{t=1}^{T} ||\boldsymbol{\ell}_t||^2 \leq cD^2 + \frac{1}{2c} TL^2 \leq DL\sqrt{2T}.$$

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

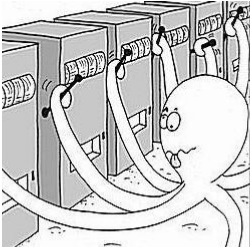


Fig.: Image credit: Microsoft Research

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: $I^* := \arg\max_{i \in \{1,\ldots,N\}} \mu_i$.
  - 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

1. For $t \leq cN$, select a random arm with probability $1/N$ and pull it.
2. For $t > cN$, pull the arm $I_t := \arg\max_{i=1,\ldots,N} \hat{\mu}_{i,t}$.

- Here c is a constant.

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 $I_t := \arg\max_{i=1,\ldots,N} \hat{\mu}_{i,t}$.

- 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 $I_t := \arg\max_{i=1,\ldots,N} \hat{\mu}_{i,t}$.

- 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

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 $\hat{\mu}_{1,cN} < \hat{\mu}_{2,cN}$.
  - If this is the case, the algorithm will keep pulling arm 2 and will never change!

ϵ-Greedy Algorithm

ϵ-Greedy Algorithm

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

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

ϵ-Greedy Algorithm

ϵ-Greedy Algorithm

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

- With probability $1 - \epsilon$, pull arm $I_t := \arg\max_{i=1,\ldots,N} \hat{\mu}_{i,t}$.
- 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 $t = 1, 2, \ldots, N$:

- With probability $1 - \epsilon$, pull arm $I_t := \arg\max_{i=1,\ldots,N} \hat{\mu}_{i,t}$.
- 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 $t = 1, 2, \ldots, N$:

- With probability $1 - \epsilon$, pull arm $I_t := \arg\max_{i=1,\ldots,N} \hat{\mu}_{i,t}$.
- 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 $t = 1, 2, \ldots, N$:

- With probability $1 - \epsilon$, pull arm $I_t := \arg\max_{i=1,\ldots,N} \hat{\mu}_{i,t}$.
- 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

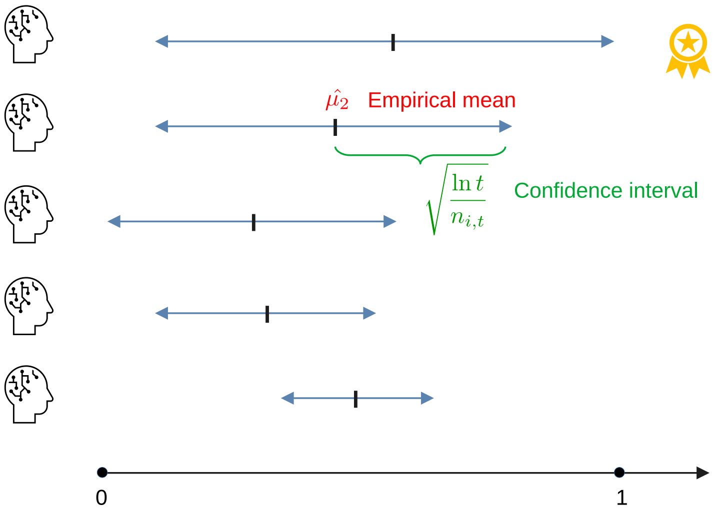
UCB Algorithm

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

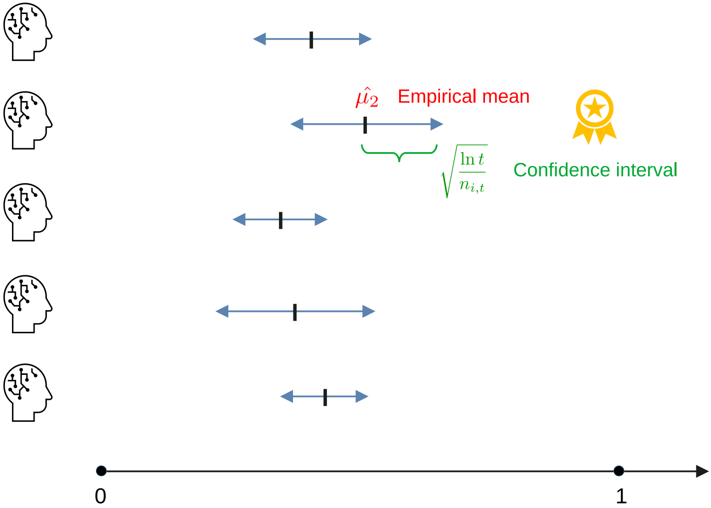
1. For $t = 1, \ldots, N$, play arm t.
2. For $t = N + 1, \ldots, T$, play arm

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

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. $\geq 1 - 2/t^2$, $\mathrm{UCB}_{i,t} := \hat{\mu}_{i,t} + \sqrt{\frac{\ln t}{n_{i,t}}} > \mu_i$.
- 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

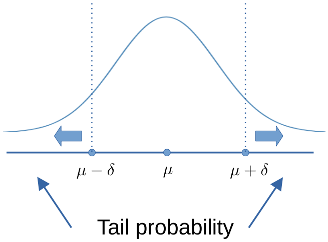
- with prob. $\geq 1 - 2/t^2$, $\mathrm{UCB}_{i,t} := \hat{\mu}_{i,t} + \sqrt{\frac{\ln t}{n_{i,t}}} > \mu_i$.
- with prob. $\geq 1 - 2/t^2$, $\hat{\mu}_{i,t} < \mu_i + \frac{\Delta_i}{2}$ when $n_{i,t} \geq \frac{4 \ln t}{\Delta_i^2}$.

Appendix: Tail probability by the Chernoff/Hoeffding bound

The Chernoff/Hoeffding bound

For independent and identically distributed (i.i.d.) samples $x_1, \ldots, x_n \in [0, 1]$ 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 $\mathrm{UCB}_{i,t} < \mathrm{UCB}_{I^*,t}$ with probability $\geq 1 - 4/t^2$. Therefore, for any t,

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

Proof of Lemma 3

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

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

does NOT hold.

Playing suboptimal arms for very limited number of times

Lemma 4

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

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

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

Proof of Lemma 4 (contd.)

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

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

- With probability $1 - \epsilon$, pull arm $I_t := \arg\max_{i=1,\ldots,N} \hat{\mu}_{i,t}$.
- 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 $t = 1, 2, \ldots, N$, set $\epsilon := N^{1/3}/T^{1/3}$:

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

Claim

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

Sketch of proving the claim

- The expected regret $\mathrm{E}[R(T)] = \sum_{t=1}^{T} \mathrm{E}[\mu^* - \mu_{T_t}]$.
- Using the greedy choice that $\hat{\mu}_{I_t} \geq \hat{\mu}_{I^*}$, we have

$$
\begin{aligned}
\mathrm{E}[R(T)] &\leq \sum_{t=1}^{T} (1-\epsilon)\mathrm{E}[(\mu_{I^*} - \hat{\mu}_{I^*} + \hat{\mu}_{I_t} - \mu_{I_t}) \mid \text{greedy choice of } I_t] + \epsilon T \\
&\leq \sum_{t=1}^{T} \left(\sqrt{\frac{\ln T}{n_{I^*,t}}} + \sqrt{\frac{\ln T}{n_{I_t,t}}} \right) + \frac{1}{T} \cdot 1 \cdot T + \epsilon T \quad \text{(Chernoff)} \\
&\approx\leq \sum_{t=1}^{T} \left(\sqrt{\frac{\ln T}{\epsilon t / N}} + \sqrt{\frac{\ln T}{\epsilon t / N}} \right) + \epsilon T + 1 \\
&\leq \sqrt{\frac{N}{\epsilon}} \sqrt{T \log T} + \epsilon T + 1 = O(N^{1/3} T^{2/3} \sqrt{\log T}).
\end{aligned}
$$

Thank you.