Advanced Algorithms (Fall 2025)

Multiplicative Weight Update

Lecturers: 尹一通，刘景铖，栗师

Nanjing University

- Focus of this lecture: learning with experts online

  - how to dynamically choose from among a set of "experts" in a way that compares favorably to the best expert

- Use it to solve 0-sum game and linear programs approximately

Outline

1. Online Learning with Experts
   - Two-outcome case
   - A more general setting
2. Multiplicative Weight Update Algorithm to Solve 0-Sum Game
3. Approximate LP feasibility using Multiplicative Weights

Outline

1. Online Learning with Experts
   - Two-outcome case
   - A more general setting
2. Multiplicative Weight Update Algorithm to Solve 0-Sum Game
3. Approximate LP feasibility using Multiplicative Weights

Learning with Experts Online

- m experts, indexed by $[m]$
- a two-outcome event on each of following T days: up or down
  - Example: stock goes up or down? rain or not?

Learning with Experts Online

- m experts, indexed by $[m]$
- a two-outcome event on each of following T days: up or down
  - Example: stock goes up or down? rain or not?
- on each day t:
  - m experts make predictions about day t
  - algorithm makes a prediction, knowing the predictions of the m experts
  - the outcome of day t reveals

Learning with Experts Online

- m experts, indexed by $[m]$
- a two-outcome event on each of following T days: up or down
  - Example: stock goes up or down? rain or not?
- on each day t:
  - m experts make predictions about day t
  - algorithm makes a prediction, knowing the predictions of the m experts
  - the outcome of day t reveals
- Goal: minimize the number of mistakes
  - Ideally, not too bad compared to the best expert.

When There Is a Perfect Expert

Lemma There is an algorithm that makes at most $\lceil \log_2 m \rceil$ mistakes, assuming there is a perfect expert, i.e., an expert that makes no mistakes.

When There Is a Perfect Expert

Lemma There is an algorithm that makes at most $\lceil \log_2 m \rceil$ mistakes, assuming there is a perfect expert, i.e., an expert that makes no mistakes.

Proof.

- The algorithm: only keep the experts who made no mistakes so far.
- Among all the experts, follow the majority.
- observation: when we made a mistake on a day, at least half of the remaining experts made a mistake on that day. $\square$

General Case

- What if there is no perfect expert?

General Case

- What if there is no perfect expert?
- Weighted majority: give weights to experts. When an expert made a mistake, halve his weight. In each step, follow the weighted majority.

General Case

- What if there is no perfect expert?
- Weighted majority: give weights to experts. When an expert made a mistake, halve his weight. In each step, follow the weighted majority.

Weighted Majority

```
w_i^0 ← 1, ∀i ∈ [m]
for t = 1, 2, ··· , T do
    if ∑_{i:predicts up} w_i^{t−1} ≥ ∑_{i:predicts down} w_i^{t−1} then
        predict "up"
    else
        predict "down"
    for every expert i do
        if i make a mistake then w_i^t ← w_i^{t−1}/2 else w_i^t ← w_i^{t−1}
```

Analysis

- $\Phi^t := \sum_{i=1}^m w_i^t$.
- $M_i^t \in \{0,1\}, i \in [m], t \in [T]$: whether expert i made a mistake on day t
- $M^t \in \{0,1\}, t \in [T]$: whether our algorithm made a mistake

Analysis

- $\Phi^t := \sum_{i=1}^m w_i^t$.
- $M_i^t \in \{0,1\}, i \in [m], t \in [T]$: whether expert i made a mistake on day t
- $M^t \in \{0,1\}, t \in [T]$: whether our algorithm made a mistake

Obs. If $M^t = 1$ for some t, we have

$$\Phi^t \leq \Phi^{t-1} - \frac{\Phi^{t-1}}{4} = \frac{3}{4}\Phi^{t-1}$$

Analysis

- $\Phi^t := \sum_{i=1}^m w_i^t$.
- $M_i^t \in \{0,1\}, i \in [m], t \in [T]$: whether expert i made a mistake on day t
- $M^t \in \{0,1\}, t \in [T]$: whether our algorithm made a mistake

Obs. If $M^t = 1$ for some t, we have

$$\Phi^t \leq \Phi^{t-1} - \frac{\Phi^{t-1}}{4} = \frac{3}{4}\Phi^{t-1}$$

- So, $\Phi^T \leq (\frac{3}{4})^{\sum_{t=1}^T M^t}\Phi_0 = (\frac{3}{4})^{\sum_{t=1}^T M^t} m$.

Analysis

- $\Phi^t := \sum_{i=1}^m w_i^t$.
- $M_i^t \in \{0,1\}, i \in [m], t \in [T]$: whether expert i made a mistake on day t
- $M^t \in \{0,1\}, t \in [T]$: whether our algorithm made a mistake

Obs. If $M^t = 1$ for some t, we have

$$\Phi^t \leq \Phi^{t-1} - \frac{\Phi^{t-1}}{4} = \frac{3}{4}\Phi^{t-1}$$

- So, $\Phi^T \leq (\frac{3}{4})^{\sum_{t=1}^T M^t}\Phi_0 = (\frac{3}{4})^{\sum_{t=1}^T M^t} m$.
- On the other hand, let k be the best expert

$$\Phi^T = \sum_{i=1}^m w_i^T \geq w_k^T \geq \left(\frac{1}{2}\right)^{\sum_{t=1}^T M_k^t}$$

$$\Phi^T \leq (\frac{3}{4})^{\sum_{t=1}^{T} M^t} m \qquad \Phi^T \geq \left(\frac{1}{2}\right)^{\sum_{i=1}^{T} M_k^t}$$

$$\Phi^T \leq (\frac{3}{4})^{\sum_{t=1}^{T} M^t} m \qquad \Phi^T \geq \left(\frac{1}{2}\right)^{\sum_{i=1}^{T} M_k^t}$$

$$\begin{aligned}
\left(\frac{1}{2}\right)^{\sum_{i=1}^{T} M_i^t} &\leq \Phi^T \leq \left(\frac{3}{4}\right)^{\sum_{i=1}^{T} M^t} m \\
(-\ln 2)\sum_{t=1}^{T} M_i^t &\leq (-\ln\frac{4}{3})\sum_{t=1}^{T} M^t + \ln m \quad \text{By taking logarithm} \\
\sum_{t=1}^{T} M^t &\leq \frac{\ln 2}{\ln 4/3}\sum_{t=1}^{T} M_k^t + \frac{\ln m}{\ln 4/3} \\
&\leq 2.41\sum_{t=1}^{T} M_k^t + 3.47\ln m
\end{aligned}$$

$$\sum_{t=1}^{T} M^t \leq 2.41 \sum_{t=1}^{T} M_k^t + 3.47 \ln m$$

Make the first constant arbitrarily close to 2

- when expert i makes a mistake on day t: $w_i^t \leftarrow w_i^{t-1} \cdot (1-\epsilon)$
- when algorithm makes a mistake on day t: $\Phi^t \leq \Phi^{t-1} \cdot (1 - \frac{\epsilon}{2})$

$$\Phi_T \leq (1-\epsilon/2)^{\sum_{t=1}^{T} M^t} \cdot m$$

- $\Phi_T \geq (1-\epsilon)^{\sum_{t=1}^{T} M_k^t}$

$$\sum_{t=1}^{T} M^t \leq \frac{\ln(1-\epsilon)}{\ln(1-\epsilon/2)} \sum_{t=1}^{T} M_k^t - \frac{\ln m}{\ln(1-\epsilon/2)}$$

$$= (2 + O(\epsilon)) \sum_{t=1}^{T} M_k^t + O\left(\frac{1}{\epsilon}\right) \ln m.$$

Lemma For any constant $\epsilon > 0$ and any function $f(m)$, no deterministic algorithm can achieve a multiplicative factor of $2 - \epsilon$ and additive factor of $f(m)$.

Lemma For any constant $\epsilon > 0$ and any function $f(m)$, no deterministic algorithm can achieve a multiplicative factor of $2 - \epsilon$ and additive factor of $f(m)$.

Proof.

- Each day $\frac{m}{2}$ experts predict "up", $\frac{m}{2}$ experts predict "down".
- Our algorithm always makes a mistake.
- Our algorithm made T mistakes, and the best expert makes at most $T/2$.
- If $T \gg f(m)$, we have $T > (2 - \epsilon) \cdot \frac{T}{2} + f(m)$. □

Lemma For any constant $\epsilon > 0$ and any function $f(m)$, no deterministic algorithm can achieve a multiplicative factor of $2 - \epsilon$ and additive factor of $f(m)$.

Proof.

- Each day $\frac{m}{2}$ experts predict "up", $\frac{m}{2}$ experts predict "down".
- Our algorithm always makes a mistake.
- Our algorithm made T mistakes, and the best expert makes at most $T/2$.
- If $T \gg f(m)$, we have $T > (2 - \epsilon) \cdot \frac{T}{2} + f(m)$. □

- However, if randomness is allowed, we can make multiplicative factor $1 + \epsilon$.

Outline

1. Online Learning with Experts
   - Two-outcome case
   - A more general setting
2. Multiplicative Weight Update Algorithm to Solve 0-Sum Game
3. Approximate LP feasibility using Multiplicative Weights

A more general setting

- no predictions: experts pay penalties
- algorithm chooses to follow experts

A more general setting

- no predictions: experts pay penalties
- algorithm chooses to follow experts
- with randomness: follow a distribution over experts
- algorithm pays the weighted average penalty of experts

A more general setting

- no predictions: experts pay penalties
- algorithm chooses to follow experts
- with randomness: follow a distribution over experts
- algorithm pays the weighted average penalty of experts
- goal: compare to the best expert

A more general setting

- no predictions: experts pay penalties
- algorithm chooses to follow experts
- with randomness: follow a distribution over experts
- algorithm pays the weighted average penalty of experts
- goal: compare to the best expert

The General Setting

```
for t ← 1, 2, ..., T do
    algorithm chooses a distribution p^t = (p_1^t, p_2^t, ..., p_n^t) over experts
    the penalty vector M^t ∈ [-1, 1]^m is revealed
    each expert i pays penalty M_i^t
    algorithm pays penalty ⟨p^t, M^t⟩ = Σ_{i=1}^m p_i^t M_i^t
```

for $t \leftarrow 1, 2, \cdots, T$ **do**

algorithm chooses a distribution $p^t = (p_1^t, p_2^t, \cdots, p_n^t)$ over experts

the penalty vector $M^t \in [-1, 1]^m$ is revealed

each expert i pays penalty M_i^t

algorithm pays penalty $\langle p^t, M^t \rangle = \sum_{i=1}^{m} p_i^t M_i^t$

A more general setting

- no predictions: experts pay penalties
- algorithm chooses to follow experts
- with randomness: follow a distribution over experts
- algorithm pays the weighted average penalty of experts
- goal: compare to the best expert

The General Setting

for $t \leftarrow 1, 2, \cdots, T$ **do**

  algorithm chooses a distribution $p^t = (p_1^t, p_2^t, \cdots, p_n^t)$ over experts

  the penalty vector $M^t \in [-1, 1]^m$ is revealed

  each expert i pays penalty M_i^t

  algorithm pays penalty $\langle p^t, M^t \rangle = \sum_{i=1}^m p_i^t M_i^t$

- Note: penalty can be negative: negative penalty = reward

Theorem Let $\epsilon \in (0,1]$. If $T \geq \frac{\ln m}{\epsilon^2}$, then there is an algorithm that satisfies:

$$\frac{1}{T}\sum_{t=1}^{T}\langle p^t, M^t\rangle \leq \frac{1}{T}\sum_{t=1}^{T} M_i^t + 2\epsilon, \forall i \in [m].$$

Theorem Let $\epsilon \in (0,1]$. If $T \geq \frac{\ln m}{\epsilon^2}$, then there is an algorithm that satisfies:

$$\frac{1}{T}\sum_{t=1}^{T}\langle p^t, M^t\rangle \leq \frac{1}{T}\sum_{t=1}^{T} M_i^t + 2\epsilon, \forall i \in [m].$$

Algorithm for the General Setting

$w_i^0 \leftarrow 1$ for every $i \in [m]$
for $t \leftarrow 1, 2, \cdots, T$ **do**
    choose $p^t \leftarrow \frac{w^{t-1}}{|w^{t-1}|_1}$
    the penalty vector $M^t \in [-1,1]^m$ is revealed
    each expert i pays penalty M_i^t
    algorithm pays penalty $\langle p^t, M^t\rangle = \sum_{i=1}^m p_i^t M_i^t$
    for every $i \in [m]$ **do**: $w_i^t \leftarrow w_i^{t-1} \cdot e^{-\epsilon \cdot M_i^t}$

Analysis of Algorithm

- the strategy: $p^t = \frac{w^{t-1}}{|w^{t-1}|_1}$ $\qquad w_i^t = w_i^{t-1} \cdot e^{-\epsilon \cdot M_i^t}$

Analysis of Algorithm

- the strategy: $p^t = \frac{w^{t-1}}{|w^{t-1}|_1}$ $\qquad w_i^t = w_i^{t-1} \cdot e^{-\epsilon \cdot M_i^t}$
- let $\Phi^t := |w^t|_1 = \sum_{i=1}^m w_i^t$ for all $t \in [0, T]$

Analysis of Algorithm

- the strategy: $p^t = \frac{w^{t-1}}{|w^{t-1}|_1}$ $\qquad w_i^t = w_i^{t-1} \cdot e^{-\epsilon \cdot M_i^t}$
- let $\Phi^t := |w^t|_1 = \sum_{i=1}^m w_i^t$ for all $t \in [0, T]$

$$\Phi^t = \sum_{i=1}^m w_i^t = \sum_{i=1}^m e^{-\epsilon \cdot M_i^t} \cdot w_i^{t-1}$$

$$\leq \sum_{i=1}^m (1 - \epsilon \cdot M_i^t + (\epsilon \cdot M_i^t)^2) \cdot w_i^{t-1}$$

as $e^x \leq 1 + x + x^2, \forall x \in [-1, 1]$

$$\leq \sum_{i=1}^m (1 - \epsilon \cdot M_i^t + \epsilon^2) \cdot w_i^{t-1} \qquad \text{as } |M_i^t| \leq 1$$

$$= (1 + \epsilon^2) \sum_{i=1}^m w_i^{t-1} - \epsilon \cdot \langle w^{t-1}, M^t \rangle$$

$$= (1 + \epsilon^2) \Phi^{t-1} - \epsilon \cdot \Phi^{t-1} \cdot \langle p^t, M^t \rangle \qquad \text{as } \Phi^{t-1} \cdot p^t = w^{t-1}$$

Analysis of Algorithm

$$\begin{aligned}\Phi^t &\leq (1+\epsilon^2)\Phi^{t-1} - \epsilon \cdot \Phi^{t-1} \cdot \langle p^t, M^t \rangle \\ &= \left(1+\epsilon^2 - \epsilon \cdot \langle p^t, M^t \rangle\right) \cdot \Phi^{t-1} \\ &\leq \exp\left(-\epsilon \cdot \langle p^t, M^t \rangle + \epsilon^2\right) \cdot \Phi^{t-1} \qquad \text{as } 1 - x \leq e^{-x}, \forall x \in \mathbb{R}\end{aligned}$$

$$\begin{aligned}\Phi^T &\leq \exp\left(\sum_{t=1}^{T}\left(-\epsilon \cdot \langle p^t, M^t \rangle + \epsilon^2\right)\right) \cdot \Phi^0 \\ &= \exp\left(-\epsilon \cdot \sum_{t=1}^{T} \langle p^t, M^t \rangle + T\epsilon^2\right) \cdot \Phi^0\end{aligned}$$

$$\Phi^T \leq \exp\Big(- \epsilon \cdot \sum_{t=1}^{T} \langle p^t, M^t \rangle + T\epsilon^2 \Big) \cdot \Phi^0$$

$$\Phi^T \leq \exp\Big(- \epsilon \cdot \sum_{t=1}^{T} \langle p^t, M^t \rangle + T\epsilon^2 \Big) \cdot \Phi^0$$

- For any expert $i \in [m]$, $\Phi^T \geq w_i^T = \exp\Big(- \epsilon \sum_{t=1}^{T} M_i^t \Big)$.

$$\Phi^T \le \exp\Big(-\epsilon \cdot \sum_{t=1}^{T} \langle p^t, M^t \rangle + T\epsilon^2 \Big) \cdot \Phi^0$$

- For any expert $i \in [m]$, $\Phi^T \ge w_i^T = \exp\Big(-\epsilon \sum_{t=1}^{T} M_i^t \Big)$.
- So, for every $i \in [m]$, we have (note that $\Phi^0 = m$)

$$\begin{aligned}
-\epsilon \cdot \sum_{t=1}^{T} M_i^t \quad &\le \quad -\epsilon \cdot \sum_{t=1}^{T} \langle p^t, M^t \rangle + T\epsilon^2 + \ln m \\
\sum_{t=1}^{T} \langle p^t, M^t \rangle \quad &\le \quad \sum_{t=1}^{T} M_i^t + T\epsilon + \frac{\ln m}{\epsilon} \\
\frac{1}{T} \sum_{t=1}^{T} \langle p^t, M^t \rangle \quad &\le \quad \frac{1}{T} \sum_{t=1}^{T} M_i^t + \epsilon + \frac{\ln m}{T\epsilon} \\
&\le \quad \frac{1}{T} \sum_{t=1}^{T} M_i^t + 2\epsilon \qquad \Big[T \ge \frac{\ln m}{\epsilon^2} \Big]
\end{aligned}$$

Theorem Let $\epsilon \in (0,1]$. If $T \geq \frac{\ln m}{\epsilon^2}$, then there is an algorithm that satisfies:

$$\frac{1}{T}\sum_{t=1}^{T}\langle p^t, M^t\rangle \leq \frac{1}{T}\sum_{t=1}^{T} M_i^t + 2\epsilon, \forall i \in [m].$$

Theorem Let $\epsilon \in (0,1]$. If $T \geq \frac{\ln m}{\epsilon^2}$, then there is an algorithm that satisfies:

$$\frac{1}{T}\sum_{t=1}^{T}\langle p^t, M^t\rangle \leq \frac{1}{T}\sum_{t=1}^{T} M_i^t + 2\epsilon, \forall i \in [m].$$

Coro. Suppose each penalty in the game is in $[-\rho, \rho]$ (instead of $[-1,1]$) for some $\rho > 0$. Let $\epsilon \in \big(0, 2\rho\big]$. If $T \geq \frac{4\rho^2 \ln m}{\epsilon^2}$, then there is an algorithm that satisfies

$$\frac{1}{T}\sum_{t=1}^{T}\langle p^t, M^t\rangle \leq \frac{1}{T}\sum_{t=1}^{T} M_i^t + \epsilon, \forall i \in [m].$$

Theorem Let $\epsilon \in (0, 1]$. If $T \geq \frac{\ln m}{\epsilon^2}$, then there is an algorithm that satisfies:

$$\frac{1}{T}\sum_{t=1}^{T}\langle p^t, M^t\rangle \leq \frac{1}{T}\sum_{t=1}^{T} M_i^t + 2\epsilon, \forall i \in [m].$$

Coro. Suppose each penalty in the game is in $[-\rho, \rho]$ (instead of $[-1, 1]$) for some $\rho > 0$. Let $\epsilon \in \big(0, 2\rho\big]$. If $T \geq \frac{4\rho^2 \ln m}{\epsilon^2}$, then there is an algorithm that satisfies

$$\frac{1}{T}\sum_{t=1}^{T}\langle p^t, M^t\rangle \leq \frac{1}{T}\sum_{t=1}^{T} M_i^t + \epsilon, \forall i \in [m].$$

Proof.

- scale penalties by $\frac{1}{\rho}$ so that each penalty is in $[-1, 1]$
- $+\epsilon$ before scaling $= +\frac{\epsilon}{\rho}$ after scaling

Theorem Let $\epsilon \in (0,1]$. If $T \geq \frac{\ln m}{\epsilon^2}$, then there is an algorithm that satisfies:

$$\frac{1}{T}\sum_{t=1}^{T}\langle p^t, M^t\rangle \leq \frac{1}{T}\sum_{t=1}^{T} M_i^t + 2\epsilon, \forall i \in [m].$$

Coro. Suppose each penalty in the game is in $[-\rho, \rho]$ (instead of $[-1,1]$) for some $\rho > 0$. Let $\epsilon \in \big(0, 2\rho\big]$. If $T \geq \frac{4\rho^2 \ln m}{\epsilon^2}$, then there is an algorithm that satisfies

$$\frac{1}{T}\sum_{t=1}^{T}\langle p^t, M^t\rangle \leq \frac{1}{T}\sum_{t=1}^{T} M_i^t + \epsilon, \forall i \in [m].$$

Proof.

- scale penalties by $\frac{1}{\rho}$ so that each penalty is in $[-1,1]$
- $+\epsilon$ before scaling $= +\frac{\epsilon}{\rho}$ after scaling
- Need $T \geq \frac{\ln m}{(\epsilon/2\rho)^2} = \frac{4\rho^2 \ln m}{\epsilon^2}$ and $\frac{\epsilon}{2\rho} \leq 1 \quad \Longleftrightarrow \quad \epsilon \leq 2\rho$ $\square$

Outline

Recall:

0-Sum Game

Input: a payoff matrix $M \in \mathbb{R}^{m \times n}, m, n \geq 1$,
two players: row player R, column player C

Output: R plays a row $i \in [m]$, C plays a column $j \in [n]$
payoff of game is M_{ij}
R wants to minimize M_{ij}, C wants to maximize M_{ij}

Recall:

0-Sum Game

Input: a payoff matrix $M \in \mathbb{R}^{m \times n}, m, n \geq 1$,
two players: row player R, column player C

Output: R plays a row $i \in [m]$, C plays a column $j \in [n]$
payoff of game is M_{ij}
R wants to minimize M_{ij}, C wants to maximize M_{ij}

Rock-Scissor-Paper Game

payoff	R	S	P
R	0	-1	1
S	1	0	-1
P	-1	1	0

- By scaling, we assume $M \in [-1, 1]^{m \times n}$.

player	objective	game term	number	distribution
row player	minimize	expert	m	y
column player	maximize	event	n	x

- By scaling, we assume $M \in [-1, 1]^{m \times n}$.

player	objective	game term	number	distribution
row player	minimize	expert	m	y
column player	maximize	event	n	x

Multiplicative weight update for 0-sum games

```
let $w_i^0 = 1$ for every $i \in [m]$
for $t \leftarrow 1$ to $T$, where $T = \left\lceil \frac{4 \ln m}{\epsilon^2} \right\rceil$ do
    algorithm chooses distribution $y^t = \frac{w^{t-1}}{|w^{t-1}|_1}$
    let $j^t$ be the $j \in [n]$ that maximizes $M(y^t, j)$
    event $j^t$ happens:
        expert $i \in [m]$ pays penalty $M(i, j_t)$
        algorithm pays penalty $M(y^t, j_t)$
    $w_i^t \leftarrow w_i^{t-1} \cdot e^{-\epsilon \cdot M(i, j_t)/2}$ for every $i \in [m]$
```

player	objective	game term	number	distribution
row player	minimize	expert	m	y
column player	maximize	event	n	x

- Since $T \geq \frac{4 \ln m}{\epsilon^2}$, we have

$$\frac{1}{T}\sum_{t=1}^{T} M(p^t, j^t) \quad \leq \quad \min_{i \in [m]} \frac{1}{T}\sum_{t=1}^{T} M(i, j^t) + \epsilon$$

player	objective	game term	number	distribution
row player	minimize	expert	m	y
column player	maximize	event	n	x

- Since $T \geq \frac{4 \ln m}{\epsilon^2}$, we have

$$\frac{1}{T}\sum_{t=1}^{T} M(p^t, j^t) \quad \leq \quad \min_{i \in [m]} \frac{1}{T}\sum_{t=1}^{T} M(i, j^t) + \epsilon$$

- $\hat{t}$: the $t \in [T]$ with minimum $M(y^t, j^t)$ $\qquad \hat{y} := y^{\hat{t}}$

player	objective	game term	number	distribution
row player	minimize	expert	m	y
column player	maximize	event	n	x

- Since $T \geq \frac{4 \ln m}{\epsilon^2}$, we have

$$\frac{1}{T}\sum_{t=1}^{T} M(p^t, j^t) \quad \leq \quad \min_{i \in [m]} \frac{1}{T}\sum_{t=1}^{T} M(i, j^t) + \epsilon$$

- $\hat{t}$: the $t \in [T]$ with minimum $M(y^t, j^t)$ $\qquad \hat{y} := y^{\hat{t}}$
- $\hat{x}$: uniform distribution over multi-set $\{j^1, j_2, \cdots, j^T\}$

$$\max_j M(\hat{y}, j) = M(\hat{y}, j^{\hat{t}}) \leq \frac{1}{T}\sum_{t=1}^{T} M(p^t, j^t)$$

$$\leq \min_i \frac{1}{T}\sum_{t=1}^{T} M(i, j^t) + \epsilon = \min_i M(i, \hat{x}) + \epsilon$$

player	objective	game term	number	distribution
row player	minimize	expert	m	y
column player	maximize	event	n	x

$$\max_j M(\hat{y}, j) \leq \min_i M(i, \hat{x}) + \epsilon$$

player	objective	game term	number	distribution
row player	minimize	expert	m	y
column player	maximize	event	n	x

$$\max_j M(\hat{y}, j) \leq \min_i M(i, \hat{x}) + \epsilon$$

- λ^*: value of game

player	objective	game term	number	distribution
row player	minimize	expert	m	y
column player	maximize	event	n	x

$$\max_j M(\hat{y}, j) \leq \min_i M(i, \hat{x}) + \epsilon$$

- λ^*: value of game

$$\lambda^* \leq \max_j M(\hat{y}, j) \leq \min_i M(i, \hat{x}) + \epsilon \leq \lambda^* + \epsilon$$

player	objective	game term	number	distribution
row player	minimize	expert	m	y
column player	maximize	event	n	x

$$\max_j M(\hat{y}, j) \leq \min_i M(i, \hat{x}) + \epsilon$$

- λ^*: value of game

$$\lambda^* \leq \max_j M(\hat{y}, j) \leq \min_i M(i, \hat{x}) + \epsilon \leq \lambda^* + \epsilon$$

- Therefore $\hat{y}$ and $\hat{x}$ are approximately the optimum strategies for the row and column players.

Outline

1. Online Learning with Experts
   - Two-outcome case
   - A more general setting
2. Multiplicative Weight Update Algorithm to Solve 0-Sum Game
3. Approximate LP feasibility using Multiplicative Weights

Linear Program: Exact Version

Input: An "easy" polytope $K \subseteq \mathbb{R}^n$ (e.g., $K = [0,1]^n$)
normal linear constraints $Ax \geq b$, $\quad A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$

Output: decide if $\{x \in K : Ax \geq b\} = \emptyset$,
if not, then output $x \in K$ with $Ax \geq b$

Linear Program: Exact Version

Input: An "easy" polytope $K \subseteq \mathbb{R}^n$ (e.g., $K = [0,1]^n$)
normal linear constraints $Ax \geq b, \quad A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$

Output: decide if $\{x \in K : Ax \geq b\} = \emptyset$,
if not, then output $x \in K$ with $Ax \geq b$

Linear Program: Approximate Version

Input: An "easy" polytope $K \subseteq \mathbb{R}^n$ (e.g., $K = [0,1]^n$)
normal linear constraints $Ax \geq b, \quad A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$

Output: either claim $\{x \in K : Ax \geq b\} = \emptyset$,
or output $x \in K$ with $Ax \geq b - \epsilon \cdot \mathbf{1}$

Linear Program: Exact Version

Input: An "easy" polytope $K \subseteq \mathbb{R}^n$ (e.g., $K = [0,1]^n$)

normal linear constraints $Ax \geq b$, $\quad A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$

Output: decide if $\{x \in K : Ax \geq b\} = \emptyset$,

if not, then output $x \in K$ with $Ax \geq b$

Linear Program: Approximate Version

Input: An "easy" polytope $K \subseteq \mathbb{R}^n$ (e.g., $K = [0,1]^n$)

normal linear constraints $Ax \geq b$, $\quad A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$

Output: either claim $\{x \in K : Ax \geq b\} = \emptyset$,

or output $x \in K$ with $Ax \geq b - \epsilon \cdot \mathbf{1}$

- Note: in case there is no exact solution, but an approximate solution, algorithm can respond either way.

Approximate LP Solver using MWU

row of A	constraint	expert	dual solution y	m
column of A	variable		primal solution x	n

- event $=$ a point in K

Approximate LP Solver using MWU

row of A	constraint	expert	dual solution y	m
column of A	variable		primal solution x	n

- event $=$ a point in K

$w_i^0 \leftarrow 1$ for every $i \in [m]$
for $t \leftarrow 1$ to T, for some T to be decided later **do**
    $y^t \leftarrow \frac{w^{t-1}}{|w^{t-1}|}$
    if $\exists x^t \in K$ s.t $\langle y^t, Ax\rangle \geq \langle y^t, b\rangle$ **then** ▷ event x^t happens
        for every $i \in [m]$ **do**
            expert i gets penalty $A_i x^t - b_i$
            $w_i^t \leftarrow w_i^{t-1} \cdot e^{-\epsilon\cdot(A_i x^t - b_i)/2}$
        our algorithm gets penalty $\langle y^t, Ax^t - b\rangle$
    else return "empty"
return $\hat{x} = \frac{1}{T}\sum_{i=1}^{T} x^t$

- Counter-intuitive: the more satisfied a constraint is, the more penalty it gets.
- In every iteration, we only need to focus on one “aggregated” linear constraint
- If algorithm returns “empty”, then the LP is not feasible

- Counter-intuitive: the more satisfied a constraint is, the more penalty it gets.
- In every iteration, we only need to focus on one "aggregated" linear constraint
- If algorithm returns "empty", then the LP is not feasible
- $\rho := \sup_{x \in K} \max_i |A_i x - b_i|$, $\quad \epsilon \in (0, 2\rho] \quad T := \left\lceil \frac{4\rho^2 \ln m}{\epsilon^2} \right\rceil$

- Counter-intuitive: the more satisfied a constraint is, the more penalty it gets.
- In every iteration, we only need to focus on one "aggregated" linear constraint
- If algorithm returns "empty", then the LP is not feasible

- $\rho := \sup_{x \in K} \max_i |A_i x - b_i|$, $\quad \epsilon \in (0, 2\rho] \quad T := \left\lceil \frac{4\rho^2 \ln m}{\epsilon^2} \right\rceil$
- $\forall i \in [m]$:

$$0 \leq \frac{1}{T} \sum_{t=1}^{T} \langle y^t, Ax^t - b \rangle \leq \frac{1}{T} \sum_{t=1}^{T} (A_i x^t - b_i) + \epsilon = A_i \hat{x} - b_i + \epsilon$$

- Counter-intuitive: the more satisfied a constraint is, the more penalty it gets.
- In every iteration, we only need to focus on one "aggregated" linear constraint
- If algorithm returns "empty", then the LP is not feasible

- $\rho := \sup_{x \in K} \max_i |A_i x - b_i|$, $\quad \epsilon \in (0, 2\rho] \quad T := \left\lceil \frac{4\rho^2 \ln m}{\epsilon^2} \right\rceil$
- $\forall i \in [m]$:

$$0 \leq \frac{1}{T} \sum_{t=1}^{T} \langle y^t, Ax^t - b \rangle \leq \frac{1}{T} \sum_{t=1}^{T} (A_i x^t - b_i) + \epsilon = A_i \hat{x} - b_i + \epsilon$$

- Therefore, $A_i x^* \geq b_i - \epsilon, \forall i \in [m] \quad \Longleftrightarrow \quad Ax^* \geq b - \epsilon \cdot \mathbf{1}$