# Exponential Weights Algorithms for Online Learning

Yoav Freund

January 9, 2020

slides in Hedge/talk2.handout.pdf on:
https://github.com/yoavfreund/2020-Online-Learning

# Outline

**Hedge**$(\eta)$Algorithm

Hedging vs. Halving

# Outline

**Hedge**$(\eta)$Algorithm

- Hedging vs. Halving

Bound on total loss

- Upper bound on $\sum_{i=1}^{N} w_i^{T+1}$
- Lower bound on $\sum_{i=1}^{N} w_i^{T+1}$
- Combining Upper and Lower bounds

# Outline

**Hedge**($\eta$)Algorithm
- Hedging vs. Halving

Bound on total loss
- Upper bound on $\sum_{i=1}^{N} w_i^{T+1}$
- Lower bound on $\sum_{i=1}^{N} w_i^{T+1}$
- Combining Upper and Lower bounds

tuning $\eta$

# Outline

- **Hedge**$(\eta)$Algorithm
  - Hedging vs. Halving
- Bound on total loss
  - Upper bound on $\sum_{i=1}^{N} w_i^{T+1}$
  - Lower bound on $\sum_{i=1}^{N} w_i^{T+1}$
  - Combining Upper and Lower bounds
- tuning $\eta$
- Lower Bounds

# Outline

- **Hedge**$(\eta)$Algorithm
  - Hedging vs. Halving
- Bound on total loss
  - Upper bound on $\sum_{i=1}^{N} w_i^{T+1}$
  - Lower bound on $\sum_{i=1}^{N} w_i^{T+1}$
  - Combining Upper and Lower bounds
- tuning $\eta$
- Lower Bounds
- Repeated Matrix Games

# The hedging problem

- $N$ possible actions

# The hedging problem

- $N$ possible actions
- At each time step $t = 1, 2, \ldots, T$:

# The hedging problem

- $N$ possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.

# The hedging problem

- $N$ possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.

# The hedging problem

- $N$ possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss $\mathbf{p}^t \cdot \ell^t$

# The hedging problem

- $N$ possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \le \ell_i^t \le 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss $\mathbf{p}^t \cdot \ell^t$
- **Goal:** minimize total expected loss

# The hedging problem

- $N$ possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss $\mathbf{p}^t \cdot \boldsymbol{\ell}^t$
- **Goal:** minimize total expected loss
- Here we have stochasticity - but only in **algorithm**, not in **outcome**

# The hedging problem

- $N$ possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss $\mathbf{p}^t \cdot \boldsymbol{\ell}^t$
- **Goal:** minimize total expected loss
- Here we have stochasticity - but only in **algorithm**, not in **outcome**
- Fits nicely in game theory

# Hedging vs. Halving

- Like halving - we want to zoom into best action (expert).

# Hedging vs. Halving

- Like halving - we want to zoom into best action (expert).
- Unlike halving - no action is perfect.

# Hedging vs. Halving

- Like halving - we want to zoom into best action (expert).
- Unlike halving - no action is perfect.
- Basic idea - reduce probability of lossy actions,
  but not all the way to zero.

# Hedging vs. Halving

- Like halving - we want to zoom into best action (expert).
- Unlike halving - no action is perfect.
- Basic idea - reduce probability of lossy actions, but not all the way to zero.
- **Modified Goal:** minimize difference between expected total loss and minimal total loss of repeating one action.

$$\sum_{t=1}^{T} \mathbf{p}^t \cdot \ell^t - \min_i \left( \sum_{t=1}^{T} \ell_i^t \right)$$

# Using hedge to generalize halving alg.

- Suppose that there is no perfect expert.

# Using hedge to generalize halving alg.

- Suppose that there is no perfect expert.
- action $i$ = use prediction of expert $i$

# Using hedge to generalize halving alg.

- Suppose that there is no perfect expert.
- action $i$ = use prediction of expert $i$
- Now each iteration of game consistst of three steps:

# Using hedge to generalize halving alg.

- Suppose that there is no perfect expert.
- action $i$ = use prediction of expert $i$
- Now each iteration of game consistst of three steps:
  - Experts make predictions $e_i^t \in \{0, 1\}$

# Using hedge to generalize halving alg.

- Suppose that there is no perfect expert.
- action $i$ = use prediction of expert $i$
- Now each iteration of game consistst of three steps:
  - Experts make predictions $e_i^t \in \{0, 1\}$
  - Algorithm predicts 1 with probability $\sum_{i:e_i^t=1} p_i^t$.

# Using hedge to generalize halving alg.

- Suppose that there is no perfect expert.
- action $i$ = use prediction of expert $i$
- Now each iteration of game consistst of three steps:
  - Experts make predictions $e_i^t \in \{0, 1\}$
  - Algorithm predicts 1 with probability $\sum_{i:e_i^t=1} p_i^t$.
  - outcome $o_i^t$ is revealed. $\ell_i^t = 0$ if $e_i^t = o_i^t$, $\ell_i^t = 1$ otherwise.

# The Hedge($\eta$)Algorithm

Consider action $i$ at time $t$

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

# The **Hedge**($\eta$)Algorithm

Consider action $i$ at time $t$

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$w_i^t = w_i^1 e^{-\eta L_i^t}$$

Note freedom to choose initial weight ($w_i^1$) $\sum_{i=1}^n w_i^1 = 1$.

# The **Hedge**($\eta$)Algorithm

Consider action $i$ at time $t$

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$w_i^t = w_i^1 e^{-\eta L_i^t}$$

Note freedom to choose initial weight ($w_i^1$) $\sum_{i=1}^{n} w_i^1 = 1$.

- $\eta > 0$ is the learning rate parameter. Halving: $\eta \to \infty$

# The **Hedge**($\eta$)Algorithm

Consider action $i$ at time $t$

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$w_i^t = w_i^1 e^{-\eta L_i^t}$$

Note freedom to choose initial weight ($w_i^1$) $\sum_{i=1}^{n} w_i^1 = 1$.

- $\eta > 0$ is the learning rate parameter. Halving: $\eta \to \infty$
- Probability:

$$p_i^t = \frac{w_i^t}{\sum_{j=1}^{N} w_i^t},$$

# The **Hedge**($\eta$)Algorithm

Consider action $i$ at time $t$

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$w_i^t = w_i^1 e^{-\eta L_i^t}$$

Note freedom to choose initial weight ($w_i^1$) $\sum_{i=1}^n w_i^1 = 1$.

- $\eta > 0$ is the learning rate parameter. Halving: $\eta \to \infty$
- Probability:

$$p_i^t = \frac{w_i^t}{\sum_{j=1}^N w_i^t},$$

# The Hedge($\eta$)Algorithm

Consider action $i$ at time $t$

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$w_i^t = w_i^1 e^{-\eta L_i^t}$$

Note freedom to choose initial weight ($w_i^1$) $\sum_{i=1}^{n} w_i^1 = 1$.

- $\eta > 0$ is the learning rate parameter. Halving: $\eta \to \infty$
- Probability:

$$p_i^t = \frac{w_i^t}{\sum_{j=1}^{N} w_i^t}, \quad \mathbf{p}^t = \frac{\mathbf{w}^t}{\sum_{j=1}^{N} w_i^t}$$

# Choosing the initial weights

- Giving an action high initial weight makes alg perform well if that action performs well.

# Choosing the initial weights

- Giving an action high initial weight makes alg perform well if that action performs well.
- If good action has low initial weight, our total loss will be larger.

# Choosing the initial weights

- Giving an action high initial weight makes alg perform well if that action performs well.
- If good action has low initial weight, our total loss will be larger.
- As $\sum_{i=1}^{n} w_i^1 = 1$ increasing one weight implies decreasing some others.

# Choosing the initial weights

- Giving an action high initial weight makes alg perform well if that action performs well.
- If good action has low initial weight, our total loss will be larger.
- As $\sum_{i=1}^{n} w_i^1 = 1$ increasing one weight implies decreasing some others.
- Plays a similar role to prior distribution in Bayesian algorithms.

# Bound on the loss of **Hedge**$(\eta)$Algorithm

# Bound on the loss of **Hedge**($\eta$)Algorithm

- Theorem (main theorem)

  *For any sequence of loss vectors* $\ell^1, \ldots, \ell^T$*, and for any* $i \in \{1, \ldots, N\}$*, we have*

$$L_{\textbf{Hedge}(\eta)} \leq \frac{-\ln(w_i^1) + \eta L_i}{1 - e^{-\eta}}.$$

# Bound on the loss of **Hedge**($\eta$)Algorithm

- Theorem (main theorem)

  *For any sequence of loss vectors $\ell^1, \ldots, \ell^T$, and for any $i \in \{1, \ldots, N\}$, we have*

$$L_{\textbf{Hedge}(\eta)} \leq \frac{-\ln(w_i^1) + \eta L_i}{1 - e^{-\eta}}.$$

  - Proof: by combining upper and lower bounds on $\sum_{i=1}^{N} w_i^{T+1}$

# Upper bound on $\sum_{i=1}^N w_i^{T+1}$

## Lemma (upper bound)

*For any sequence of loss vectors* $\ell^1, \ldots, \ell^T$ *we have*

$$\ln\left(\sum_{i=1}^{N} w_i^{T+1}\right) \leq -(1-e^{-\eta})L_{\mathbf{Hedge}(\eta)}.$$

# Proof of upper bound (slide 1)

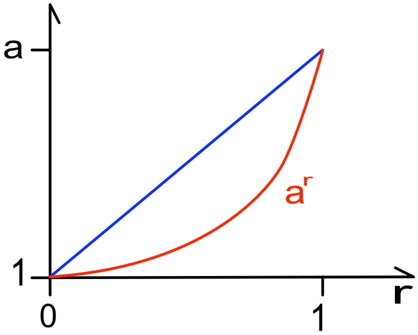
- If $a \geq 0$ then $a^r$ is convex.

# Proof of upper bound (slide 1)

- If $a \geq 0$ then $a^r$ is convex.
- For $r \in [0, 1]$, $a^r \leq 1 - (1 - a)r$

# Proof of upper bound (slide 1)

- If $a \geq 0$ then $a^r$ is convex.
- For $r \in [0, 1]$, $a^r \leq 1 - (1 - a)r$




-

# Proof of upper bound (slide 2)

Applying $a^r \leq 1 - (1-a)^r$ where $a = e^{-\eta}, r = \ell_i^t$

$$\sum_{i=1}^{N} w_i^{t+1} = \sum_{i=1}^{N} w_i^t e^{-\eta \ell_i^t}$$

# Proof of upper bound (slide 2)

Applying $a^r \leq 1 - (1-a)^r$ where $a = e^{-\eta}, r = \ell_i^t$

$$\begin{aligned}\sum_{i=1}^{N} w_i^{t+1} &= \sum_{i=1}^{N} w_i^t e^{-\eta \ell_i^t} \\ &\leq \sum_{i=1}^{N} w_i^t \left(1 - (1 - e^{-\eta})\ell_i^t\right)\end{aligned}$$

# Proof of upper bound (slide 2)

Applying $a^r \leq 1 - (1-a)^r$ where $a = e^{-\eta}, r = \ell_i^t$

$$
\begin{aligned}
\sum_{i=1}^{N} w_i^{t+1} &= \sum_{i=1}^{N} w_i^t e^{-\eta \ell_i^t} \\
&\leq \sum_{i=1}^{N} w_i^t \left(1 - (1 - e^{-\eta}) \ell_i^t\right) \\
&= \left(\sum_{i=1}^{N} w_i^t\right) \left(1 - (1 - e^{-\eta}) \frac{\mathbf{w}^t}{\sum_{i=1}^{N} w_i^t} \cdot \ell^t\right)
\end{aligned}
$$

# Proof of upper bound (slide 2)

Applying $a^r \leq 1 - (1-a)^r$ where $a = e^{-\eta}, r = \ell_i^t$

$$
\begin{aligned}
\sum_{i=1}^N w_i^{t+1} &= \sum_{i=1}^N w_i^t e^{-\eta \ell_i^t} \\
&\leq \sum_{i=1}^N w_i^t \left(1 - (1 - e^{-\eta})\ell_i^t\right) \\
&= \left(\sum_{i=1}^N w_i^t\right)\left(1 - (1 - e^{-\eta})\frac{\mathbf{w}^t}{\sum_{i=1}^N w_i^t} \cdot \ell^t\right) \\
&= \left(\sum_{i=1}^N w_i^t\right)(1 - (1 - e^{-\eta})\mathbf{p}^t \cdot \ell^t)
\end{aligned}
$$

# Proof of upper bound (slide 3)

- Combining

$$\sum_{i=1}^{N} w_i^{t+1} \leq \left(\sum_{i=1}^{N} w_i^t\right) \left(1 - (1 - e^{-\eta})\mathbf{p}^t \cdot \ell^t\right)$$

# Proof of upper bound (slide 3)

- Combining

$$\sum_{i=1}^{N} w_i^{t+1} \leq \left(\sum_{i=1}^{N} w_i^t\right)(1-(1-e^{-\eta})\mathbf{p}^t \cdot \ell^t)$$

- for $t = 1, \ldots, T$

# Proof of upper bound (slide 3)

- Combining

$$\sum_{i=1}^{N} w_i^{t+1} \leq \left(\sum_{i=1}^{N} w_i^t\right) \left(1 - (1 - e^{-\eta})\mathbf{p}^t \cdot \ell^t\right)$$

- for $t = 1, \ldots, T$
- yields

$$\sum_{i=1}^{N} w_i^{T+1} \quad \leq \quad \prod_{t=1}^{T} \left(1 - (1 - e^{-\eta})\mathbf{p}^t \cdot \ell^t\right)$$

# Proof of upper bound (slide 3)

- Combining

$$\sum_{i=1}^{N} w_i^{t+1} \leq \left(\sum_{i=1}^{N} w_i^t\right)\left(1-(1-e^{-\eta})\mathbf{p}^t \cdot \ell^t\right)$$

- for $t = 1, \ldots, T$
- yields

$$\sum_{i=1}^{N} w_i^{T+1} \quad \leq \quad \prod_{t=1}^{T}\left(1-(1-e^{-\eta})\mathbf{p}^t \cdot \ell^t\right)$$

# Proof of upper bound (slide 3)

- Combining

$$\sum_{i=1}^{N} w_i^{t+1} \leq \left(\sum_{i=1}^{N} w_i^t\right) \left(1 - (1 - e^{-\eta})\mathbf{p}^t \cdot \ell^t\right)$$

- for $t = 1, \ldots, T$
- yields

$$\begin{aligned} \sum_{i=1}^{N} w_i^{T+1} &\leq \prod_{t=1}^{T}\left(1 - (1 - e^{-\eta})\mathbf{p}^t \cdot \ell^t\right) \\ &\leq \exp\left(-(1 - e^{-\eta})\sum_{t=1}^{T} \mathbf{p}^t \cdot \ell^t\right) \end{aligned}$$

since $1 + x \leq e^x$ for $x = -(1 - e^{-\eta})$.

# Lower bound on $\sum_{i=1}^N w_i^{T+1}$

For any $j = 1, \ldots, N$:

$$\sum_{i=1}^{N} w_i^{T+1} \geq w_j^{T+1} = w_j^1 e^{-\eta L_j}$$

# Combining Upper and Lower bounds

- Combining bounds on $\ln\left(\sum_{i=1}^{N} w_i^{T+1}\right)$

$$\ln w_j^1 - \eta L_j \leq \ln \sum_{i=1}^{N} w_i^{T+1} \leq -(1 - e^{-\eta}) \sum_{t=1}^{T} \mathbf{p}^t \cdot \ell^t$$

# Combining Upper and Lower bounds

- Combining bounds on $\ln\left(\sum_{i=1}^{N} w_i^{T+1}\right)$

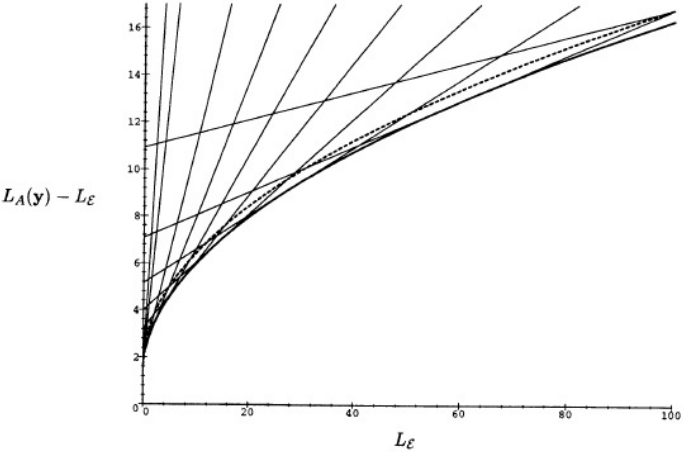
$$\ln w_j^1 - \eta L_j \leq \ln \sum_{i=1}^{N} w_i^{T+1} \leq -(1 - e^{-\eta}) \sum_{t=1}^{T} \mathbf{p}^t \cdot \ell^t$$

- Reversing signs, using $L_{\mathbf{Hedge}(\eta)} = \sum_{t=1}^{T} \mathbf{p}^t \cdot \ell^t$ and reorganizing we get

$$L_{\mathbf{Hedge}(\eta)} \leq \frac{-\ln(w_i^1) + \eta L_i}{1 - e^{-\eta}}$$

# Tuning $\eta$




$L_A(\mathbf{y}) - L_\mathcal{E}$
$L_\mathcal{E}$
0
2
4
6
8
10
12
14
16
0
20
40
60
80
100

# Tuning $\eta$

- Suppose $\min_i L_i \leq \tilde{L}$

# Tuning $\eta$

- Suppose $\min_i L_i \le \tilde{L}$
- set

$$\eta = \ln\left(1 + \sqrt{\frac{2\ln N}{\tilde{L}}}\right) \approx \sqrt{\frac{2\ln N}{\tilde{L}}}$$

# Tuning $\eta$

- Suppose $\min_i L_i \leq \tilde{L}$
- set

$$\eta = \ln\left(1 + \sqrt{\frac{2\ln N}{\tilde{L}}}\right) \approx \sqrt{\frac{2\ln N}{\tilde{L}}}$$

- use uniform initial weights $\mathbf{w}^1 = \langle 1/N, \ldots, 1/N \rangle$

# Tuning $\eta$

- Suppose $\min_i L_i \leq \tilde{L}$
- set

$$\eta = \ln\left(1 + \sqrt{\frac{2\ln N}{\tilde{L}}}\right) \approx \sqrt{\frac{2\ln N}{\tilde{L}}}$$

- use uniform initial weights $\mathbf{w}^1 = \langle 1/N, \ldots, 1/N \rangle$
- Then

$$L_{\mathbf{Hedge}(\eta)} \leq \frac{-\ln(w_i^1) + \eta L_i}{1 - e^{-\eta}} \leq \min_i L_i + \sqrt{2\tilde{L}\ln N} + \ln N$$

## Tuning $\eta$ as a function of $T$

- trivially $\min_i L_i \leq T$, yielding

$$L_{\textbf{Hedge}(\eta)} \leq \min_i L_i + \sqrt{2T \ln N} + \ln N$$

# Tuning $\eta$ as a function of $T$

- trivially $\min_i L_i \leq T$, yielding

$$L_{\mathbf{Hedge}(\eta)} \leq \min_i L_i + \sqrt{2T \ln N} + \ln N$$

- per iteration we get:

$$\frac{L_{\mathbf{Hedge}(\eta)}}{T} \leq \min_i \frac{L_i}{T} + \sqrt{\frac{2 \ln N}{T}} + \frac{\ln N}{T}$$

# How good is this bound?

- Very good! There is a closely matching lower bound!

# How good is this bound?

- Very good! There is a closely matching lower bound!
- There exists a stochastic adversarial strategy such that with high probability for any hedging strategy **S** after $T$ trials

$$L_{\mathbf{S}} - \min_i L_i \geq (1 - o(1))\sqrt{2T \ln N}$$

# How good is this bound?

- Very good! There is a closely matching lower bound!
- There exists a stochastic adversarial strategy such that with high probability for any hedging strategy **S** after $T$ trials

$$L_{\mathbf{S}} - \min_i L_i \geq (1 - o(1))\sqrt{2T \ln N}$$

- The adversarial strategy is random, extremely simple, and does not depend on the hedging strategy!

# The adversarial strategy

- Adversary sets each loss $\ell_i^t$ indepedently at random to 0 or 1 with equal probabilities $(1/2, 1/2)$.

# The adversarial strategy

- Adversary sets each loss $\ell_i^t$ indepedently at random to 0 or 1 with equal probabilities $(1/2, 1/2)$.
- Obviously, nothing to learn !
  $L_{\mathbf{S}} \approx T/2$.

# The adversarial strategy

- Adversary sets each loss $\ell_i^t$ indepedently at random to 0 or 1 with equal probabilities $(1/2, 1/2)$.
- Obviously, nothing to learn !
  $L_{\mathbf{S}} \approx T/2$.
- On the other hand $\min_i L_i \approx T/2 - \sqrt{2T \ln N}$

# The adversarial strategy

- Adversary sets each loss $\ell_i^t$ indepedently at random to $0$ or $1$ with equal probabilities $(1/2, 1/2)$.
- Obviously, nothing to learn !
  $L_{\mathbf{S}} \approx T/2$.
- On the other hand $\min_i L_i \approx T/2 - \sqrt{2T \ln N}$
- The difference $L_{\mathbf{S}} - \min_i L_i$ is due to unlearnable random fluctuations!

# The adversarial strategy

- Adversary sets each loss $\ell_i^t$ indepedently at random to 0 or 1 with equal probabilities $(1/2, 1/2)$.
- Obviously, nothing to learn !
  $L_{\mathbf{S}} \approx T/2$.
- On the other hand $\min_i L_i \approx T/2 - \sqrt{2T \ln N}$
- The difference $L_{\mathbf{S}} - \min_i L_i$ is due to unlearnable random fluctuations!
- Detailed proof quite involved. See games paper.

# Summary

- Given learning rate $\eta$ the **Hedge**$(\eta)$algorithm satisfies

$$L_{\textbf{Hedge}(\eta)} \leq \frac{\ln N + \eta L_i}{1 - e^{-\eta}}$$

# Summary

- Given learning rate $\eta$ the **Hedge**($\eta$)algorithm satisfies

$$L_{\textbf{Hedge}(\eta)} \leq \frac{\ln N + \eta L_i}{1 - e^{-\eta}}$$

- Setting $\eta \approx \sqrt{\frac{2 \ln N}{T}}$ guarantees

$$L_{\textbf{Hedge}(\eta)} \leq \min_i L_i + \sqrt{2T \ln N} + \ln N$$

# Summary

- Given learning rate $\eta$ the **Hedge**($\eta$)algorithm satisfies

$$L_{\textbf{Hedge}(\eta)} \leq \frac{\ln N + \eta L_i}{1 - e^{-\eta}}$$

- Setting $\eta \approx \sqrt{\frac{2 \ln N}{T}}$ guarantees

$$L_{\textbf{Hedge}(\eta)} \leq \min_i L_i + \sqrt{2T \ln N} + \ln N$$

- A trivial random data, in which there is nothing to be learned forces any algorithm to suffer this total loss

# Some loose threads

- Total Loss of best action usually scales linearly with time $T$, but we need to know the **horizon** $T$ ahead of time to choose $\eta$ correctly.

# Some loose threads

- Total Loss of best action usually scales linearly with time $T$, but we need to know the **horizon** $T$ ahead of time to choose $\eta$ correctly.
- $T$ is tight only when the loss of experts at each iteration is either 0 or 1. If the loss of the best expert is $o(T)$ then we would like to have a tighter bound.

# Some loose threads

- Total Loss of best action usually scales linearly with time $T$, but we need to know the **horizon** $T$ ahead of time to choose $\eta$ correctly.
- $T$ is tight only when the loss of experts at each iteration is either 0 or 1. If the loss of the best expert is $o(T)$ then we would like to have a tighter bound.
- Observing only the loss of chosen action - the multi-armed bandit problem. Will get to that later in the course.

# Homework

- Due thursday January 16, 2020

# Homework

- Due thursday January 16, 2020
- Turn in a latex-generated PDF printout.

# Homework

- Due thursday January 16, 2020
- Turn in a latex-generated PDF printout.
- From Prediction, Learning and Games by Cesa-Bianchi and Lugosi.

# Homework

- Due thursday January 16, 2020
- Turn in a latex-generated PDF printout.
- From Prediction, Learning and Games by Cesa-Bianchi and Lugosi.
- **2.1** Assume that you have to predict a sequence $Y_1, Y_2, \ldots \in \{0, 1\}$ of i.i.d. random variables with unknown distribution, your decision space is $[0, 1]$, and the loss function is $\ell(\widehat{p}, y) = |\widehat{p} - y|$. How would you proceed? Try to estimate the cumulative loss of your forecaster and compare it to the cumulative loss of the best of the two experts, one of which always predicts 1 and the other always predicts 0. Which are the most "difficult" distributions? How does your (expected) regret compare to that of the weighted average algorithm (which does not "know" that the outcome sequence is i.i.d.)?

# Homework

- Due thursday January 16, 2020
- Turn in a latex-generated PDF printout.
- From Prediction, Learning and Games by Cesa-Bianchi and Lugosi.
- **2.1** Assume that you have to predict a sequence $Y_1, Y_2, \ldots \in \{0, 1\}$ of i.i.d. random variables with unknown distribution, your decision space is $[0, 1]$, and the loss function is $\ell(\widehat{p}, y) = |\widehat{p} - y|$. How would you proceed? Try to estimate the cumulative loss of your forecaster and compare it to the cumulative loss of the best of the two experts, one of which always predicts 1 and the other always predicts 0. Which are the most "difficult" distributions? How does your (expected) regret compare to that of the weighted average algorithm (which does not "know" that the outcome sequence is i.i.d.)?
- Observing only the loss of chosen action - the multi-armed bandit problem. Will get to that later in the course.

# Zero sum games in matrix form

- Game between two players.

# Zero sum games in matrix form

- Game between two players.
- Defined by $n \times m$ matrix $\mathbf{M}$

# Zero sum games in matrix form

- Game between two players.
- Defined by $n \times m$ matrix $\mathbf{M}$
- Row player chooses $i \in \{1, \ldots, n\}$

# Zero sum games in matrix form

- Game between two players.
- Defined by $n \times m$ matrix $\mathbf{M}$
- Row player chooses $i \in \{1, \ldots, n\}$
- Column player chooses $j \in \{1, \ldots, m\}$

# Zero sum games in matrix form

- Game between two players.
- Defined by $n \times m$ matrix $\mathbf{M}$
- Row player chooses $i \in \{1, \ldots, n\}$
- Column player chooses $j \in \{1, \ldots, m\}$
- Row player gains $\mathbf{M}(i, j) \in [0, 1]$

# Zero sum games in matrix form

- Game between two players.
- Defined by $n \times m$ matrix $\mathbf{M}$
- Row player chooses $i \in \{1, \ldots, n\}$
- Column player chooses $j \in \{1, \ldots, m\}$
- Row player gains $\mathbf{M}(i, j) \in [0, 1]$
- Column player looses $\mathbf{M}(i, j)$

# Zero sum games in matrix form

- Game between two players.
- Defined by $n \times m$ matrix $\mathbf{M}$
- Row player chooses $i \in \{1, \ldots, n\}$
- Column player chooses $j \in \{1, \ldots, m\}$
- Row player gains $\mathbf{M}(i, j) \in [0, 1]$
- Column player looses $\mathbf{M}(i, j)$
- Game repeated many times.

# Pure vs. mixed strategies

- Choosing a single action = pure strategy.

# Pure vs. mixed strategies

- Choosing a single action = pure strategy.
- Choosing a Distribution over actions = mixed strategy.

# Pure vs. mixed strategies

- Choosing a single action = pure strategy.
- Choosing a Distribution over actions = mixed strategy.
- Row player chooses dist. over rows **P**

# Pure vs. mixed strategies

- Choosing a single action = pure strategy.
- Choosing a Distribution over actions = mixed strategy.
- Row player chooses dist. over rows **P**
- Column player chooses dist. over columns **Q**

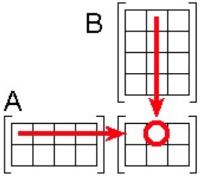
# Pure vs. mixed strategies

- Choosing a single action = pure strategy.
- Choosing a Distribution over actions = mixed strategy.
- Row player chooses dist. over rows $\mathbf{P}$
- Column player chooses dist. over columns $\mathbf{Q}$
- Row player gains $\mathbf{M}(\mathbf{P}, \mathbf{Q})$.

# Pure vs. mixed strategies

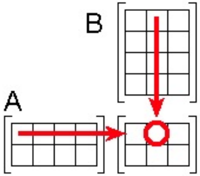
- Choosing a single action = pure strategy.
- Choosing a Distribution over actions = mixed strategy.
- Row player chooses dist. over rows **P**
- Column player chooses dist. over columns **Q**
- Row player gains $\mathbf{M}(\mathbf{P}, \mathbf{Q})$.
- Column player looses $\mathbf{M}(\mathbf{P}, \mathbf{Q})$.

# Mixed strategies in matrix notation




$$(A \times B)_{12} = \sum_{r=1}^{4} a_{1r} b_{r2} = a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32} + a_{14}b_{42}$$

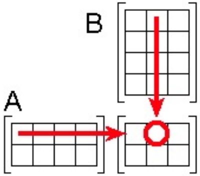
# Mixed strategies in matrix notation




$$(A \times B)_{12} = \sum_{r=1}^{4} a_{1r} b_{r2} = a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32} + a_{14}b_{42}$$

**Q** is a column vector. **P**$^T$ is a row vector.

# Mixed strategies in matrix notation




$$(A \times B)_{12} = \sum_{r=1}^{4} a_{1r} b_{r2} = a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32} + a_{14}b_{42}$$

**Q** is a column vector. $\mathbf{P}^T$ is a row vector.

$$\mathbf{M}(\mathbf{P}, \mathbf{Q}) = \mathbf{P}^T \mathbf{M} \mathbf{Q} = \sum_{i=1}^{n} \sum_{j=1}^{m} \mathbf{P}(i)\mathbf{M}(i,j)\mathbf{Q}(j)$$

# The minmax Theorem

When using pure strategies, second player has an advantage.

# The minmax Theorem

When using pure strategies, second player has an advantage.

John von Neumann, 1928.

$$\min_{\mathbf{P}} \max_{\mathbf{Q}} \mathbf{M}(\mathbf{P}, \mathbf{Q}) = \max_{\mathbf{Q}} \min_{\mathbf{P}} \mathbf{M}(\mathbf{P}, \mathbf{Q})$$

In words: for mixed strategies, choosing second gives no advantage.

# Minmax is weaker than diminishing regret

- The minmax theorem proves the existence of an Equilibrium.

# Minmax is weaker than diminishing regret

- The minmax theorem proves the existence of an Equilibrium.
- Learning guarantees no regret with respect to the past.

# Minmax is weaker than diminishing regret

- The minmax theorem proves the existence of an Equilibrium.
- Learning guarantees no regret with respect to the past.
- If all sides use learning, then game will converge to minmax equilibrium.

# Minmax is weaker than diminishing regret

- The minmax theorem proves the existence of an Equilibrium.
- Learning guarantees no regret with respect to the past.
- If all sides use learning, then game will converge to minmax equilibrium.
- If opponent is not optimally adversarial (limited by knowledge, computationa power...) then learning gives better performance than min-max.

# Minmax is weaker than diminishing regret

- The minmax theorem proves the existence of an Equilibrium.
- Learning guarantees no regret with respect to the past.
- If all sides use learning, then game will converge to minmax equilibrium.
- If opponent is not optimally adversarial (limited by knowledge, computationa power...) then learning gives better performance than min-max.
- Our goal is to minimize regret.