Exponential Weights Algorithms for Online Learning

Yoav Freund

- slides in `https://github.com/yoavfreund/2025-online-learning/blob/main/2.Hedge/talk2.handout.pdf`

Exponential Weights Algorithms for Online Learning

Yoav Freund

- slides in `https://github.com/yoavfreund/2025-online-learning/blob/main/2.Hedge/talk2.handout.pdf`
- Paper: Freund, Schapire "A decision-theoretic generalization of on-line learning and an application to boosting”

Exponential Weights Algorithms for Online Learning

Yoav Freund

- slides in `https://github.com/yoavfreund/2025-online-learning/blob/main/2.Hedge/talk2.handout.pdf`
- Paper: Freund, Schapire "A decision-theoretic generalization of on-line learning and an application to boosting"
- In PLG: pages 12-25

Outline

Decision Theoretic Online learning

- Hedging vs. Halving
- Failure of Follow the leader

Outline

Outline

Decision Theoretic Online learning

- Hedging vs. Halving
- Failure of Follow the leader

Hedge(η)Algorithm

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

Decision Theoretic Online learning

- Hedging vs. Halving
- Failure of Follow the leader

Hedge(η)Algorithm

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 η

Outline

Decision Theoretic Online learning

Hedging vs. Halving

Failure of Follow the leader

Hedge(η)Algorithm

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 η

Lower Bounds

The hedging problem

- AKA “Decision theoretic Online Learning” (DTOL)

The hedging problem

- AKA “Decision theoretic Online Learning” (DTOL)
- N possible actions

The hedging problem

- AKA "Decision theoretic Online Learning" (DTOL)
- N possible actions
- At each time step $t = 1, 2, \ldots, T$:

The hedging problem

- AKA "Decision theoretic Online Learning" (DTOL)
- N possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.

The hedging problem

- AKA "Decision theoretic Online Learning" (DTOL)
- 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

- AKA “Decision theoretic Online Learning” (DTOL)
- 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

- AKA "Decision theoretic Online Learning" (DTOL)
- 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 $\mathbf{p}^t \cdot \ell_t$
- **Goal:** minimize total expected loss

The hedging problem

- AKA "Decision theoretic Online Learning" (DTOL)
- 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 $\mathbf{p}^t \cdot \ell_t$
- **Goal:** minimize total expected loss
- Here we have stochasticity - but only in **algorithm**, not in **outcome**

The hedging problem

- AKA “Decision theoretic Online Learning” (DTOL)
- 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 $\mathbf{p}^t \cdot \ell_t$
- **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.

Failure of follow the leader

Failure of follow the leader

expert1 loss
expert1 cumul

expert2 loss
expert2 cumul

FTL cumul

Failure of follow the leader

	$t = 1$
expert1 loss	0.5
expert1 cumul	0.5
expert2 loss	0.0
expert2 cumul	0.0
FTL cumul	0.0

Failure of follow the leader

	$t = 1$	$t = 2$
expert1 loss	0.5	0.0
expert1 cumul	0.5	0.5
expert2 loss	0.0	1.0
expert2 cumul	0.0	1.0
FTL cumul	0.0	1.0

Failure of follow the leader

	$t = 1$	$t = 2$	$t = 3$
expert1 loss	0.5	0.0	1.0
expert1 cumul	0.5	0.5	1.5
expert2 loss	0.0	1.0	0.0
expert2 cumul	0.0	1.0	1.0
FTL cumul	0.0	1.0	2.0

Failure of follow the leader

	$t = 1$	$t = 2$	$t = 3$	$t = 4$
expert1 loss	0.5	0.0	1.0	0.0
expert1 cumul	0.5	0.5	1.5	1.5
expert2 loss	0.0	1.0	0.0	1.0
expert2 cumul	0.0	1.0	1.0	2.0
FTL cumul	0.0	1.0	2.0	3.0

Failure of follow the leader

	$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$
expert1 loss	0.5	0.0	1.0	0.0	1.0
expert1 cumul	0.5	0.5	1.5	1.5	2.5
expert2 loss	0.0	1.0	0.0	1.0	0.0
expert2 cumul	0.0	1.0	1.0	2.0	2.0
FTL cumul	0.0	1.0	2.0	3.0	4.0

Failure of follow the leader

	$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$	$t = 6$
expert1 loss	0.5	0.0	1.0	0.0	1.0	0.0
expert1 cumul	0.5	0.5	1.5	1.5	2.5	2.5
expert2 loss	0.0	1.0	0.0	1.0	0.0	1.0
expert2 cumul	0.0	1.0	1.0	2.0	2.0	3.0
FTL cumul	0.0	1.0	2.0	3.0	4.0	5.0

Failure of follow the leader

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	$t=7$
expert1 loss	0.5	0.0	1.0	0.0	1.0	0.0	1.0
expert1 cumul	0.5	0.5	1.5	1.5	2.5	2.5	3.5
expert2 loss	0.0	1.0	0.0	1.0	0.0	1.0	0.0
expert2 cumul	0.0	1.0	1.0	2.0	2.0	3.0	3.0
FTL cumul	0.0	1.0	2.0	3.0	4.0	5.0	6.0

The **Hedge**(η)Algorithm

Consider action i at time t

- Total loss:

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

The **Hedge**(η)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**(η)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**(η)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**(η)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**(η)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**(η)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*

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

Bound on the loss of **Hedge**(η)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*

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

- Note effect of the limits $\eta \to 0$ and $\eta \to \infty$

Bound on the loss of **Hedge**(η)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*

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

- Note effect of the limits $\eta \to 0$ and $\eta \to \infty$
- 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*

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

Proof of upper bound (slide 1)

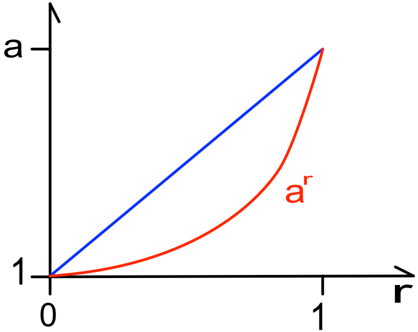
- If $a \geq 0$ then a^r is convex.

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 $a = e^{-\eta}, r = \ell_i^t$

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

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

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

- 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

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

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

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

Proof of upper bound (slide 3)

- Combining

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

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

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

since $1 + x \le e^x$ 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 $\ln\left(\sum_{i=1}^{N} w_i^{T+1}\right)$

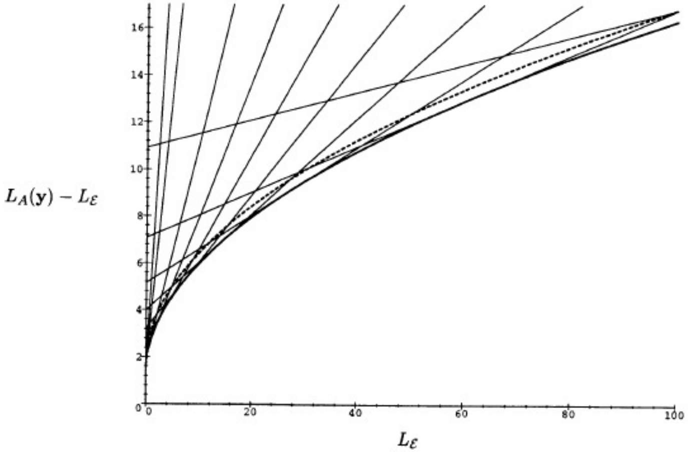
$$\ln w_j^1 - \eta L_j \leq \ln \sum_{i=1}^{N} w_i^{T+1} \leq -(1 - e^{-\eta}) \sum_{t=1}^{T} \mathbf{p}^t \cdot \ell_t$$

- 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 η

How to Use Expert Advice 451



Tuning η

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

Tuning η

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

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

Tuning η

- 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 η

- 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

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

Exact tuning of η

2.2 How to choose β

So far, we have analyzed **Hedge**(β) for a given choice of β, and we have proved reasonable bounds for any choice of β. In practice, we will often want to choose β so as to maximally exploit any prior knowledge we may have about the specific problem at hand.

The following lemma will be helpful for choosing β using the bounds derived above.

Lemma 4 *Suppose* $0 \leq L \leq \tilde{L}$ *and* $0 < R \leq \tilde{R}$. *Let* $\beta = g(\tilde{L}/\tilde{R})$ *where* $g(z) = 1/(1 + \sqrt{2/z})$. *Then*

$$\frac{-L \ln \beta + R}{1 - \beta} \leq L + \sqrt{2\tilde{L}\tilde{R}} + R.$$

Proof: (Sketch) It can be shown that $-\ln \beta \leq (1 - \beta^2)/(2\beta)$ for $\beta \in (0, 1]$. Applying this approximation and the given choice of β yields the result. ■

Lemma 4 can be applied to any of the bounds above since all of these bounds have the form given in the lemma. For example, suppose we have N strategies, and we also know a prior bound $\tilde{L}$ on the loss of the best strategy. Then, combining Equation (9) and Lemma 4, we have

$$L_{\mathbf{Hedge}(\beta)} \leq \min_i L_i + \sqrt{2\tilde{L} \ln N} + \ln N \tag{11}$$

Tuning η 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 η as a function of T

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

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

- per iteration we get:

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

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 ℓ_i^t indepedently at random to 0 or 1 with equal probabilities $(1/2, 1/2)$.

The adversarial strategy

- Adversary sets each loss ℓ_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 ℓ_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 ℓ_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 ℓ_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 section 3.7 in PLG.

Summary

- Given learning rate η the **Hedge**(η)algorithm satisfies

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

Summary

- Given learning rate η the **Hedge**(η)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 η the **Hedge**(η)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 η 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 η 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 η 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, 2025

Homework

- Due thursday January 16, 2025
- Upload to gradescope, entry code: EV8RWE

Homework

- Due thursday January 16, 2025
- Upload to gradescope, entry code: EV8RWE
- From PLG.

Homework

- Due thursday January 16, 2025
- Upload to gradescope, entry code: EV8RWE
- From PLG.
- **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.)?