Today: The multiplicative weights algorithm.

If periods indicate good.

Which Channel?
Multiplicative Weights Algorithm.
Framework: \( n \) experts, each loses different amount every day.
Perfect Expert: \( \log n \) mistakes.
Imperfect Expert: best makes \( m \) mistakes.
Deterministic Strategy: \( 2(1 + \varepsilon)m + \frac{\log n}{\varepsilon} \)
Real numbered losses: Best loses \( L^* \) total.
Randomized Strategy: \( (1 + \varepsilon)L^* + \frac{\log n}{\varepsilon} \)

Strategy:
Choose proportional to weights
multiply weight by \( (1 - \varepsilon)^{\text{loss}} \).

Multiplicative weights framework! Algorithm appears in many settings!
Applications next!
Learning.

Which stock do you buy?
Which weather station is most accurate?
Which road do you take?
How should I behave?

Today: Do what seems to work!

    Softly.
    Avoid a little what doesn’t work.
    Do something a little more that does.
Experts framework.

$n$ experts.

Every day, each offers a prediction.

“Rain” or “Shine.”

<table>
<thead>
<tr>
<th></th>
<th>Day 1</th>
<th>Day 2</th>
<th>Day 3</th>
<th>...</th>
<th>Day T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert 1</td>
<td>Shine</td>
<td>Rain</td>
<td>Shine</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>Expert 2</td>
<td>Shine</td>
<td>Shine</td>
<td>Shine</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>Expert 3</td>
<td>Rain</td>
<td>Rain</td>
<td>Rain</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td></td>
<td>:</td>
<td>:</td>
<td>:</td>
<td>Shine</td>
<td>...</td>
</tr>
</tbody>
</table>

Rained! Shined! Shined! ... 

Whose advice do you follow?

“The one who is correct most often.”

Sort of.

How well do you do?
Infallible expert.

One of the experts is infallible!

Your strategy?

Choose any expert that has not made a mistake!

How long to find perfect expert?

Maybe..never! Never see a mistake.

Better model?

How many mistakes could you make? **Mistake Bound.**

(A) 1
(B) 2
(C) $\log n$
(D) $n - 1$

Adversary designs setup to watch who you choose, and make that expert make a mistake.

$n - 1!$
Note.

Adversary:
  makes you want to look bad.
  "You could have done so well"...
  but you didn’t! aha.ha ...ha ha!

Analysis of Algorithms: do as well as possible!

Minimize Regret:
  Regret \equiv \text{Difference between Loss/Gain compared to the best.}
Back to mistake bound.

Infallible Experts.

Alg: Choose one of the perfect experts.

**Mistake Bound: \( n - 1 \)**
- Lower bound: adversary argument.
- Upper bound: every mistake finds fallible expert.

Better Algorithm?

Making decision, not trying to find expert!

Algorithm: Go with the majority of previously correct experts.

What you would do anyway!
Alg 2: find majority of the perfect

How many mistakes could you make?

(A) 1
(B) 2
(C) \( \log n \)
(D) \( n - 1 \)

At most \( \log n \)!

When alg makes a mistake, |“perfect” experts| drops by a factor of two.

Initially \( n \) perfect experts

\[
\begin{align*}
\text{mistake } &\rightarrow \leq n/2 \text{ perfect experts} \\
\text{mistake } &\rightarrow \leq n/4 \text{ perfect experts} \\
\vdots \\
\text{mistake } &\rightarrow \leq 1 \text{ perfect expert}
\end{align*}
\]

\( \geq 1 \) perfect expert \( \rightarrow \) at most \( \log n \) mistakes!
Goal?
Do as well as the best expert!

Algorithm. Suggestions?
Go with majority?
Penalize inaccurate experts?
Best expert is penalized the least.

1. Initially: $w_i = 1$.
2. Predict with weighted majority of experts.
3. $w_i \rightarrow w_i/2$ if wrong.
Analysis: weighted majority

Goal: Best expert makes $m$ mistakes.

Potential function: $\sum w_i$. Initially $n$.

1. Initially: $w_i = 1$.
   For best expert, $b$, $w_b \geq \frac{1}{2^m}$.

2. Predict with weighted majority of experts.
   Each mistake:
   - total weight of incorrect experts reduced by $-1$? $-2$? factor of $\frac{1}{2}$?
   - each incorrect expert weight multiplied by $\frac{1}{2}$!
   - total weight decreases by factor of $\frac{1}{2}$? factor of $\frac{3}{4}$?
   - mistake $\rightarrow \geq$ half weight with incorrect experts.
   - $(\geq \frac{1}{2}$ total.) $\times 1/2$

   1 So reduction by $\geq 1/4$.

Mistake $\rightarrow$ potential function decreased by $\frac{3}{4}$.

We have

$$\frac{1}{2^m} \leq \sum w_i \leq \left( \frac{3}{4} \right)^M n.$$ 

where $M$ is number of algorithm mistakes.
Analysis: continued.

\[ \frac{1}{2^m} \leq \sum_i w_i \leq \left( \frac{3}{4} \right)^M n. \]

\( m \) - best expert mistakes  \( M \) - algorithm mistakes.

\[ \frac{1}{2^m} \leq \left( \frac{3}{4} \right)^M n. \]

Take log of both sides.

\[ -m \leq -M \log(4/3) + \log n. \]

Solve for \( M \).

\[ M \leq (m + \log n)/\log(4/3) \leq 2.4(m + \log n) \]
Analysis: \((1 - \varepsilon)\) penalty

Algorithm: Multiply by \(1 - \varepsilon\) for incorrect experts...

\[(1 - \varepsilon)^m \leq (1 - \frac{\varepsilon}{2})^M n.\]

Massage...

\[
\ln(1 - \varepsilon)^m \leq \ln((1 - \frac{\varepsilon}{2})^M n) \implies m\ln(1 - \varepsilon) \leq M\ln(1 - \varepsilon/2) + \ln n
\]

\[
\implies (-\varepsilon - \varepsilon^2)m \leq M(-\varepsilon/2) + \ln n
\]

\[
\implies -\varepsilon(1 + \varepsilon)m \leq M(-\varepsilon/2) + \ln n \implies 2(1 + \varepsilon)m \geq M - \ln n
\]

Penultimate step from:

\[-\varepsilon - \varepsilon^2 \leq \ln(1 - \varepsilon) \leq -\varepsilon, \varepsilon \leq 1/2\]

\[M \leq 2(1 + \varepsilon)m + \frac{2\ln n}{\varepsilon}\]

Approaches a factor of two of best expert performance as \(m \to \infty\).
Best Analysis?

Consider two experts: A, B

Adversary: A correct even days, B correct odd days

Best expert performance: $T/2$ mistakes.

Pattern (A): $T - 1$ mistakes.

Factor of (almost) two worse!
Better approach?
Use?
   Randomization!
That is, choose expert $i$ with prob $\propto w_i$
Bad example: A,B,A,B,A...
After a bit, A and B make nearly the same number of mistakes.
Choose each with approximately the same probability.
Make a mistake around $1/2$ of the time.
Best expert makes $T/2$ mistakes.
Roughly optimal!
Randomized analysis.

Some formulas:

For \( \varepsilon \leq 1, x \in [0, 1] \),

\[
(1 + \varepsilon)^x \leq (1 + \varepsilon x)
\]

\[
(1 - \varepsilon)^x \leq (1 - \varepsilon x)
\]

For \( \varepsilon \in [0, \frac{1}{2}] \),

\[-\varepsilon - \varepsilon^2 \leq \ln(1 - \varepsilon) \leq -\varepsilon\]

\[\varepsilon - \varepsilon^2 \leq \ln(1 + \varepsilon) \leq \varepsilon\]

Proof Idea: \( \ln(1 + x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \cdots \)
Randomized algorithm

Expert $i$ loses $\ell_i^t \in [0, 1]$ in round $t$.

1. Initially $w_i = 1$ for expert $i$.
2. Choose expert $i$ with prob $\frac{w_i}{W}$, $W = \sum_i w_i$.
3. $w_i \leftarrow w_i(1 - \varepsilon)\ell_i^t$

$W(t)$ sum of $w_i$ at time $t$. $W(0) = n$

Best expert, $b$, loses $L^*$ total. $\rightarrow W(T) \geq w_b \geq (1 - \varepsilon)^{L^*}$.

$L_t = \sum_i \frac{w_i \ell_i^t}{W}$ expected loss of alg. in time $t$.

Claim: For $\varepsilon \leq 1/2$, $W(t+1) \leq W(t)(1 - \varepsilon L_t)$ Loss $\rightarrow$ weight loss.

Proof:

$W(t+1) = \sum_i (1 - \varepsilon)\ell_i^t w_i \leq \sum_i (1 - \varepsilon \ell_i^t) w_i$

$= \sum_i w_i - \varepsilon \sum_i w_i \ell_i^t$

$= \sum_i w_i \left(1 - \varepsilon \frac{\sum_i w_i \ell_i^t}{\sum_i w_i}\right)$

$= W(t)(1 - \varepsilon L_t)$
Analysis

\[(1 - \varepsilon)^{L^*} \leq W(T) \leq n \sum_t (1 - \varepsilon L_t)\]

Take logs

\[(L^*) \ln(1 - \varepsilon) \leq \ln n + \sum \ln(1 - \varepsilon L_t)\]

Use \(-\varepsilon - \varepsilon^2 \leq \ln(1 - \varepsilon) \leq -\varepsilon\)

\[-(L^*) (\varepsilon + \varepsilon^2) \leq \ln n - \varepsilon \sum L_t\]

And

\[\sum_t L_t \leq (1 + \varepsilon) L^* + \frac{\ln n}{\varepsilon}\]

\(\sum_t L_t\) is total expected loss of algorithm.

Within \((1 + \varepsilon)\) ish of the best expert!

No factor of 2 loss!
Gains.

Why so negative?
Each day, each expert gives gain in $[0, 1]$.

Multiplicative weights with $(1 + \varepsilon)^{g_i^t}$.

$$G \geq (1 - \varepsilon)G^* - \frac{\log n}{\varepsilon}$$

where $G^*$ is payoff of best expert.

Scaling:
Not $[0, 1]$, say $[0, \rho]$.

$$L \leq (1 + \varepsilon)L^* + \frac{\rho \log n}{\varepsilon}$$
Other Models

What if you only know the advice of the expert you pick?

Bandit Problems.
   Model for studying exploration versus exploitation.

Reinforcement Learning.
   Get reward with certain action.
   Don’t know anything about action not taken.

Algorithms for Reinforcement Learning.

Same as the Bandit model in Learning with the Regret Framework.
Optimization (Online and offline).

Minimize a convex function $f(x)$. $x^* \in \mathbb{R}^n$.

$x^*$ minimizes $f(x)$.

Gradients: $\nabla f(\cdot)$.

Loss on experts corresponding to coordinates $x_i$.

Linear combination of gradients linear lower bound on $f(x)$.

Expert’s says average value of loss is low relative to lower bound,

$\Rightarrow$ average difference of $f(x_t)$ from $f(x^*)$ is low.

Expert’s algorithm is a dual version of gradient descent.

Online: works with just local gradients.
Multiplicative Weights Algorithm.
Framework: $n$ experts, each loses different amount every day.
  Perfect Expert: $\log n$ mistakes.
  Imperfect Expert: best makes $m$ mistakes.
Deterministic Strategy: $2(1 + \varepsilon)m + \frac{\log n}{\varepsilon}$
Real numbered losses: Best loses $L^*$ total.
Randomized Strategy: $(1 + \varepsilon)L^* + \frac{\log n}{\varepsilon}$

Strategy:
  Choose proportional to weights
  multiply weight by $(1 - \varepsilon)^{\text{loss}}$.

Multiplicative weights framework! Algorithm appears in many settings!
Applications next!