Multiplicative-Weight Update Method

Anil Maheshwari

anil@scs.carleton.ca
School of Computer Science
Carleton University
Canada

Outline

Motivating Problem

How to be an Expert?

Task: Predict whether the stock index will go $\uparrow$ or $\downarrow$ at the end of each day.

Model:

1. Access to n experts (newspapers, stock briefs, $\ldots$)
2. Correct Prediction: Reward of \$$0$ for that day.
3. Wrong Prediction: Costs us \$$1$ for that day.

Problem: Devise an algorithm that predicts for each day.
Suppose we are at day t, where $t \in \{1, \ldots, T\}$.
Algorithm can utilize previous predictions for $t-1$ days $+$ experts advises for days $1, \ldots, t$

Objective: At the end of T days, we want to be competitive with respect to any expert. Our cost (loss) is not significantly higher than the cost of any expert (including the best expert).

WHY?

Introduction to Algorithms, by **C**ormen, **L**eiserson, **R**ivest and **S**tein
4th Edition, MIT Press, 2022.
Chapter 33: Machine Learning
33.1 Clustering
33.2 Multiplicative Weights Algorithm
33.3 Gradient Descent

Numerous Applications:
- Replaces solving linear programs for important class of problems.
- Applications in game theory
- Online Learning
- Approximating numerous combinatorial optimization problems
- Fastest algorithm for Network Flow

Warmup: ≥ 1 Real Experts

Extra Knowledge

- Among the n experts there is at least one expert who is always correct.
- We don't know their identity.

≥ 1 Real Experts

Extra Knowledge: The best expert(s) always makes correct prediction.

```
Let the set of experts be $E = \{1, \ldots, n\}$

For each day $t := 1$ to $T$ do:
    Step 1: Among all the remaining experts in $E$, poll them to find the
            prediction of the majority of them for that day. Record that as
            the prediction of the algorithm.
    Step 2: Observe the true outcome at the end of the day. Discard all
            those experts that predicted wrong from $E$ from future
            considerations.
```

Warmup (contd.)

#Wrong Predictions

Algorithm makes at most $\lceil \log_2 n \rceil$ wrong predictions

Proof.

At least half of the remaining experts are discarded for future consideration when the algorithm makes an incorrect prediction.

# mistakes	0	1	2	3	$\cdots$
# remaining experts	n	$\leq n/2$	$\leq n/4$	$\leq n/8$	$\cdots$

$\square$

MWU Method

Multiplicative-Weight Update Method - Idea

Extra Knowledge

- Among the n experts there is at least one expert who makes fewer than m mistakes over the period of T days.
- We call them the best experts.
- We don't know the identity of the best experts.

Multiplicative-Weight Update Method - Idea

Extra Knowledge: The best expert(s) makes $\leq m$ mistakes over the period of T days.

```
Let the set of experts be $E = \{1, \ldots, n\}$
For each expert $i$, set its weight $w_i^1 = 1$

For each day $t := 1$ to $T$ do:
        Step 1: Find the weighted majority prediction of the experts. Sum
                total the weights of all the experts that predict $\uparrow$ (respectively,
                $\downarrow$). Whichever of the two sums is higher is the prediction for
                the day $t$.
        Step 2: Observe the true outcome at the end of the day $t$
        Step 3: For all the experts $i$ that predicted wrongly, their weight is set
                to $w_i^{t+1} = w_i^t/2$. For all others, $w_i^{t+1} = w_i^t$.
```

Potential Function

Potential Function

Potential Function Φ^t

Define the potential function Φ^t for day $t \in \{1, \ldots, T\}$ to be the sum total of the weights of all the experts at the start of the day t, i.e. $\Phi^t = \sum_{i=1}^{n} w_i^t$

Observations:

1. $\Phi^1 = \sum_{i=1}^{n} w_i^1 = \sum_{i=1}^{n} 1 = n$
2. If the algorithm makes a wrong prediction on day t, $\Phi^{t+1} \leq \frac{3}{4}\Phi^t$
3. If the algorithm has made M mistakes in T days, its total weight at the end of day T, $\Phi^{T+1} \leq \left(\frac{3}{4}\right)^M \Phi^1 = \left(\frac{3}{4}\right)^M n$

Observation 2

Bounding Potential Function

If the algorithm makes a wrong prediction on day t, $\Phi^{t+1} \leq \frac{3}{4}\Phi^t$.

Proof.

Define $\Phi^t(\uparrow) = \sum_{\text{expert } i \text{ predicts } \uparrow} w_i^t$, and

$\Phi^t(\downarrow) = \sum_{\text{expert } i \text{ predicts } \downarrow} w_i^t$.

We know, $\Phi^t = \Phi^t(\uparrow) + \Phi^t(\downarrow)$.

Assume $\Phi^t(\uparrow) \geq \Phi^t(\downarrow) \implies \Phi^t(\uparrow) \geq \frac{1}{2}\Phi^t$ and $\Phi^t(\downarrow) \leq \frac{1}{2}\Phi^t$.

Since the algorithm makes a wrong prediction on the day t,
$\Phi^{t+1} = \frac{1}{2}\Phi^t(\uparrow) + \Phi^t(\downarrow)$.

Observe that Φ^{t+1} is maximized when $\Phi^t(\uparrow) = \Phi^t(\downarrow)$.
$\implies \Phi^{t+1} \leq \frac{1}{4}\Phi^t + \frac{1}{2}\Phi^t = \frac{3}{4}\Phi^t$. $\square$

Bounding Potential Function

Assume that the Best Expert is i (we don't know its identity).

Bounds on Potential Function

Weight of the expert i at the end of day T is $\geq (\frac{1}{2})^m$.
Therefore, $\left(\frac{1}{2}\right)^m \leq \Phi^{T+1} \leq \left(\frac{3}{4}\right)^M n$

Proof.

Weight of expert i can't go lower than $\left(\frac{1}{2}\right)^m$, as it can make at most m mistakes.
Since Φ^{T+1} is sum of weights, including the best experts weight, $\left(\frac{1}{2}\right)^m \leq \Phi^{T+1}$.

Observe, $\Phi^{T+1} \leq \frac{3}{4}\Phi^T \leq \left(\frac{3}{4}\right)^2 \Phi^{T-1} \leq \cdots \leq \left(\frac{3}{4}\right)^M \Phi^1 = \left(\frac{3}{4}\right)^M n$.

□

Number of mistakes

Bounding number of mistakes

The algorithm makes at most $2.41(m + \log n)$ mistakes.

Proof.

We had $\left(\frac{1}{2}\right)^m \leq \Phi^{T+1} \leq \left(\frac{3}{4}\right)^M n$.

Take $\log$'s:

$$\begin{aligned} -m &\leq M \log\left(\frac{3}{4}\right) + \log n \\ -M \log\left(\frac{3}{4}\right) &\leq m + \log n \\ M \log\left(\frac{4}{3}\right) &\leq m + \log n \\ \mathbf{M} &\leq \mathbf{2.41(m + \log n)} \end{aligned}$$

□

Replacing $\frac{1}{2}$

Choose $\eta \in (0, \frac{1}{2}]$

```
Let the set of experts be $E = \{1, \dots, n\}$
For each expert $i$, set its starting weight $w_i^1 = 1$

For each day $t := 1$ to $T$ do:
    Step 1: Find the weighted majority prediction of the experts. Sum
            total the weights of all the experts that predict $\uparrow$ (respectively,
            $\downarrow$). Whichever of the two sums is higher is the prediction for
            the day $t$.
    Step 2: Observe the true outcome at the end of the day $t$
    Step 3: For all the experts $i$ that predicted wrongly, their weight is set
            to $w_i^{t+1} = (1 - \eta)w_i^t$. For all others, $w_i^{t+1} = w_i^t$.
```

Upper Bounding the Potential Function

Suppose the algorithm made a mistake on the day t, i.e., the majority weighted prediction was incorrect.

An Upper Bound

$\Phi^{t+1} \leq (1 - \frac{\eta}{2})\Phi^t$.

Proof.

Assume weighted majority prediction was in favor of $\uparrow$.

Note $\Phi^t(\uparrow) \geq \frac{1}{2}\Phi^t \geq \Phi^t(\downarrow)$.

Given the assumption, we have

$$\begin{aligned}
\Phi^{t+1} &= \Phi^t(\downarrow) + (1-\eta)\Phi^t(\uparrow) \\
&= \Phi^t(\downarrow) + \Phi^t(\uparrow) - \eta\Phi^t(\uparrow) \\
&\leq \Phi^t - \frac{\eta}{2}\Phi^t \\
&= (1 - \frac{\eta}{2})\Phi^t
\end{aligned}$$

$\square$

Analysis

We have assumed that the algorithm makes at most M mistakes and that the best expert makes at most m mistakes over T days.

New Bound

For any $\eta \in (0, \frac{1}{2}]$, we have $M \leq 2(1+\eta)m + \frac{2}{\eta} \log n$

Proof.

We have $(1-\eta)^m \leq \Phi^{T+1} \leq (1-\frac{\eta}{2})^M n$.

Take $\log$'s and use $-\eta - \eta^2 \leq \ln(1-\eta) \leq -\eta$ for $\eta \in [0, \frac{1}{2}]$.

$$\begin{aligned}
m\ln(1-\eta) &\leq M\ln(1-\frac{\eta}{2}) + \ln n \\
-m(\eta+\eta^2) &\leq -M\frac{\eta}{2} + \ln n \\
M\frac{\eta}{2} &\leq \ln n + (\eta+\eta^2)m \\
\mathbf{M} &\leq \frac{\mathbf{2}}{\eta}\ln \mathbf{n} + \mathbf{2}(\mathbf{1}+\eta)\mathbf{m}
\end{aligned}$$

□

An Example

For any $\eta \in (0, \frac{1}{2}]$, we have $M \leq 2(1+\eta)m + \frac{2}{\eta}\log n$

Example: Two experts A and B.
A predicts $(\uparrow, \downarrow)$ correctly only on even numbered days
B predicts $(\uparrow, \downarrow)$ correctly only on odd numbered days

Best Expert: Wrong half the time.

How many days the Algorithm makes wrong predictions?

Randomization

Improvements (contd.)

How to remove the multiplicative factor of 2 in $M \leq \mathbf{2}(1+\eta)m + \frac{2}{\eta}\log n$?

Use Randomization

Generalizations

- So far, we have had two actions/events/outcomes/possibilities $(\uparrow, \downarrow)$.

- Assume that we have Q possible events.

- For any day t, one of the possible Q events occur.

- For each expert i, and for each possible event $q \in Q$ for day t, the loss (cost) incurred to expert i on day t is given by $m_i^t(q) \in [0, 1]$.

- Note that the losses don't have to be 0 or 1, but can be any real number between 0 and 1.

- To keep the notation simple, we will say that the loss of expert i on the day t is m_i^t.

Objectives

Let $m_i^t \in [0,1]$ denote the loss of expert $i \in \{1,\ldots,n\}$ on day $t \in \{1,\ldots,T\}$

Objective: Algorithm to be competitive against the cost incurred by the best expert. Assume i is the (unknown) best expert.
Its total cost over T days is $\sum\limits_{t=1}^{T} m_i^t$.

$M^t =$ Expected cost that the algorithm incurs on the day t.

Our algorithm should have the following guarantee:

$$\sum_{i=1}^{T} M^t \leq \frac{\ln n}{\eta} + (1+\eta)\sum_{t=1}^{T} m_i^t$$

Total expected cost over T days is within an additive factor $\frac{\ln n}{\eta}$ and a multiplicative factor $(1+\eta)$ of the best expert.

Randomized MWU

Multiplicative Weight Update Method

Set of experts $E = \{1, \dots, n\}$.
Let η be any real number in $[0, \frac{1}{2}]$
For each expert i, set its initial weight $w_i^1 = 1$.

```
For each day $t := 1$ to $T$ do:
    Step 1: Define $\Phi^t = \sum_{i=1}^{n} w_i^t$
            For each expert $i$, compute the probability $p_i^t = \frac{w_i^t}{\Phi^t}$.
    Step 2: Choose an expert based on their probabilities and predict
            according to its chosen action.
    Step 3: Update Weights: For each expert $i$ set $w_i^{t+1} = w_i^t(1 - \eta m_i^t)$.
            Note $m_i^t = m_i^t(q) \in [0, 1]$.
```

Analysis - Upper Bound

Expected loss M^t

The expected loss M^t that the algorithm incurs on the day t is given by

$$M^t = \sum_{i=1}^{n} p_i^t m_i^t = \langle p^t \cdot m^t \rangle, \tag{1}$$

where $p^t = (p_1^t, p_2^t, \ldots, p_n^t)$ and $m^t = (m_1^t, m_2^t, \ldots, m_n^t)$ and their dot product is $\langle p^t \cdot m^t \rangle$

Analysis (contd.)

Potential Function

For any $t \in \{1, \ldots, T\}$: $\Phi^{t+1} \leq \Phi^t e^{-\eta M^t}$.

Moreover, $\Phi^{T+1} \leq n e^{-\eta \sum_{t=1}^{T} M^t}$

Proof.

$$\begin{aligned} \Phi^{t+1} &= \sum_{i=1}^{n} w_i^{t+1} = \sum_{i=1}^{n} w_i^t (1 - \eta m_i^t) = \sum_{i=1}^{n} w_i^t - \eta \sum_{i=1}^{n} w_i^t m_i^t \\ &= \Phi^t - \eta \sum_{i=1}^{n} \Phi^t p_i^t m_i^t = \Phi^t (1 - \eta \langle p^t \cdot m^t \rangle) \\ &\leq \Phi^t e^{-\eta \langle p^t \cdot m^t \rangle} = \Phi^t e^{-\eta M^t} \end{aligned}$$

Using induction on t, we obtain

$$\Phi^{T+1} \leq \Phi^0 e^{-\eta \sum_{t=1}^{T} M^t} = n e^{-\eta \sum_{t=1}^{T} M^t}$$

□

Analysis - Lower Bound

Lower Bound

For any expert i, $\Phi^{T+1} \geq (1-\eta)^{\sum_{t=1}^{T} m_i^t}$

Proof.

Since all $m_i^t \in [0,1]$, $w_i^t \geq 0$.

$\implies \Phi^{t+1} \geq w_i^{t+1}$ for any individual weight as $\Phi^{t+1} = \sum_{i=1}^{n} w_i^{t+1}$

Note: For $\epsilon \in [0,1]$, $1 - \epsilon x \geq (1-\epsilon)^x$ if $x \in [0,1]$

Using the update rule of w_i^{t+1}, we have

$$\Phi^{T+1} \geq w_i^{T+1} = w_i^T(1-\eta m_i^T) = w_i^1 \prod_{t=1}^{T}(1-\eta m_i^t) \geq (1-\eta)^{\sum_{t=1}^{T} m_i^t} \qquad \square$$

Putting Lower & Upper Bounds Together

Putting upper and lower bounds for Φ^{T+1} we have

$$ne^{-\eta \sum_{t=1}^{T} M^t} \geq \Phi^{T+1} \geq (1-\eta)^{\sum_{t=1}^{T} m_i^t} \tag{2}$$

Take $\log$'s and divide by η:

$$\frac{\ln n}{\eta} - \sum_{t=1}^{T} M^t \geq \frac{\ln(1-\eta)}{\eta} \sum_{t=1}^{T} m_i^t \tag{3}$$

This is equivalent to

$$\sum_{t=1}^{T} M^t \leq \frac{\ln n}{\eta} - \frac{\ln(1-\eta)}{\eta} \sum_{t=1}^{T} m_i^t \tag{4}$$

Putting Lower & Upper Bounds Together (contd.)

Use $\eta \in [0, \frac{1}{2}]$, $-\eta - \eta^2 \leq \ln(1-\eta)$ and we obtain:

$$\sum_{t=1}^{T} M^t \leq \frac{\ln n}{\eta} + \frac{\eta + \eta^2}{\eta} \sum_{t=1}^{T} m_i^t \tag{5}$$

Equivalently,

$$\boxed{\sum_{t=1}^{T} M^t \leq \frac{\ln n}{\eta} + (1+\eta) \sum_{t=1}^{T} m_i^t}$$

± 1 Costs

MWU with costs in $[-1, 1]$

The costs of each expert can be positive or negative, i.e. $m_i^t \in [-1, 1]$. Use the same algorithm as for the cost $[0, 1]$:

```
MWU with costs in $[-1, 1]$

Set of experts $E = \{1, \dots, n\}$.
Let $\eta$ be any real number in $[0, \frac{1}{2}]$.
For each expert $i$, set its initial weight $w_i^1 = 1$.

For each day $t := 1$ to $T$ do:
    Step 1: Define $\Phi^t = \sum_{i=1}^{n} w_i^t$.
            For each expert $i$, compute $p_i^t = \frac{w_i^t}{\Phi^t}$.
    Step 2: Choose an expert based on their probabilities and predict
            according to the chosen expert.
    Step 3: Update Weights: For each expert $i$ set $w_i^{t+1} = w_i^t(1 - \eta m_i^t)$.
```

Analysis - Upper Bound

Upper Bound

$\Phi^{T+1} \leq n e^{-\eta \sum_{t=1}^{T} M^t}$

Proof.

The expected cost of the algorithm on day t: $M^t = \sum_{i=1}^{n} p_i^t m_i^t = \langle p^t \cdot m^t \rangle$

The upper bound for Φ^{t+1} follows the same analysis:
$\Phi^{t+1} = \sum_{i=1}^{n} w_i^{t+1} = \sum_{i=1}^{n} w_i^t (1 - \eta m_i^t) \leq \Phi^t e^{-\eta M^t}$

And using induction on t we have $\Phi^{T+1} \leq n e^{-\eta \sum_{t=1}^{T} M^t}$ $\square$

Analysis - Lower Bound

Lower Bound

$$\Phi^{T+1} \geq (1-\eta)^{\sum\limits_{m_i^t \geq 0} m_i^t} (1+\eta)^{-\sum\limits_{m_i^t < 0} m_i^t}$$

Proof.

Since $m_i^t \in [-1, 1]$, we have $1 - \eta m_i^t \geq 0$.
Thus, $w_i^t \geq 0 \implies \Phi^{t+1} \geq w_i^{t+1}$ for any individual weight.

From the update rule of w_i^{t+1}: $\Phi^{T+1} \geq w_i^{T+1} = \prod\limits_{t=1}^{T} (1 - \eta m_i^t)$.

Group for each day the positive m_i^t's and the negative m_i^t's:

$$\Phi^{T+1} \geq (1-\eta)^{\sum\limits_{m_i^t \geq 0} m_i^t} (1+\eta)^{-\sum\limits_{m_i^t < 0} m_i^t} \qquad \square$$

Upper+Lower Bound

Note: For $\epsilon \in [0,1]$:
If $x \in [0,1]$, $(1-\epsilon)^x \leq 1 - \epsilon x$.
If $x \in [-1,0]$, $(1+\epsilon)^{-x} \leq 1 - \epsilon x$.

$$ne^{-\eta \sum_{t=1}^{T} M^t} \geq (1-\eta)^{\sum_{m_i^t \geq 0} m_i^t} (1+\eta)^{-\sum_{m_i^t < 0} m_i^t}$$

Take $\log$'s and divide by η:

$$\frac{\ln n}{\eta} - \sum_{t=1}^{T} M^t \geq \frac{\ln(1-\eta)}{\eta} \sum_{m_i^t \geq 0} m_i^t - \frac{\ln(1+\eta)}{\eta} \sum_{m_i^t < 0} m_i^t$$

$$\sum_{t=1}^{T} M^t \leq \frac{\ln n}{\eta} - \frac{\ln(1-\eta)}{\eta} \sum_{m_i^t \geq 0} m_i^t + \frac{\ln(1+\eta)}{\eta} \sum_{m_i^t < 0} m_i^t$$

Upper+Lower Bound (contd.)

Use, $\eta + \eta^2 \geq -\ln(1-\eta)$ and $\ln(1+\eta) \geq \eta - \eta^2$, for $\eta \in [0, \frac{1}{2}]$.

$$\sum_{t=1}^{T} M^t \leq \frac{\ln n}{\eta} + (1+\eta) \sum_{m_i^t \geq 0} m_i^t + (1-\eta) \sum_{m_i^t < 0} m_i^t$$

Note: $(\eta - \eta^2) \sum_{m_i^t<0} m_i^t \geq \ln(1+\eta) \sum_{m_i^t<0} m_i^t$ because of negative values.

On expanding, we obtain

$$\sum_{t=1}^{T} M^t \leq \frac{\ln n}{\eta} + \eta \sum_{t=1}^{T} |m_i^t| + \sum_{t=1}^{T} m_i^t$$

Since $|m_i^t| \leq 1$, we have

$$\boxed{\sum_{t=1}^{T} M^t \leq \frac{\ln n}{\eta} + \eta T + \sum_{t=1}^{T} m_i^t}$$

Observations

Cost of MWU

By setting $\eta = \sqrt{\frac{\ln n}{T}}$ in $\sum_{t=1}^{T} M^t \leq \frac{\ln n}{\eta} + \eta T + \sum_{t=1}^{T} m_i^t$, we obtain $\sum_{t=1}^{T} M^t \leq 2\sqrt{T \ln n} + \sum_{t=1}^{T} m_i^t$

Interpretation: The cost of the MWU algorithm is off by an additive factor that is proportional to the square root of the product of the number of days and the number of experts compared to the best expert.

Average Error: Consider the average error on each day (divided by T):

$$\boxed{\frac{1}{T} \sum_{t=1}^{T} M^t \leq 2\sqrt{\frac{\ln n}{T}} + \frac{1}{T} \sum_{t=1}^{T} m_i^t}$$

Observe that as T increases, the average error drops down. Therefore, the MWU method can learn from experts reasonably well when executed over several days.

Generalizations

MWU with costs in $[-l, \rho]$

The costs of each expert is in $m_i^t \in [-l, \rho]$, where $l < \rho$.

```
**MWU with costs in** $[-l, \rho]$
Set of experts $E = \{1, \dots, n\}$.
Let $\eta$ be any real number in $[0, \frac{1}{2}]$
For each expert $i$, set its weight $w_i^1 = 1$

For each day $t := 1$ to $T$ do:
    **Step 1:** Define $\Phi^t = \sum_{i=1}^{n} w_i^t$. For each expert $i$, compute $p_i^t = \frac{w_i^t}{\Phi^t}$
    **Step 2:** Choose an expert based on their probabilities and predict
                according to the chosen expert
    **Step 3:** Update Weights: For each expert $i$ set
                $w_i^{t+1} = \begin{cases} w_i^t (1-\eta)^{\frac{m_i^t}{\rho}}, \text{ if } m_i^t \geq 0 \\ w_i^t (1+\eta)^{-\frac{m_i^t}{\rho}}, \text{ if } m_i^t < 0 \end{cases}$
```

Analysis

As $m_i^t/\rho \in [-1, +1]$, we have

Upper Bound

$$\Phi^{T+1} \leq ne^{-\eta \sum_{t=1}^{T} \frac{M^t}{\rho}}$$

Lower Bound

$$\Phi^{T+1} \geq (1-\eta)^{\sum_{m_i^t \geq 0} \frac{m_i^t}{\rho}} (1+\eta)^{-\sum_{m_i^t < 0} \frac{m_i^t}{\rho}}$$

Results

Theorem

$$\sum_{t=1}^{T} M^t \leq \frac{\rho \ln n}{\eta} + (1+\eta) \sum_{m_i^t \geq 0} m_i^t + (1-\eta) \sum_{m_i^t < 0} m_i^t$$

Corollary

If $\eta \leq \min(\frac{1}{2}, \frac{\epsilon}{4\rho})$ for some error parameter ϵ, after $T = \frac{16\rho^2 \ln n}{\epsilon^2}$ days, the average expected loss is given by

$$\frac{1}{T} \sum_{t=1}^{T} M^t \leq \epsilon + \frac{1}{T} \sum_{t} m_i^t$$

Applications

Applications

Numerous applications including:

1. 2-Player Zero-Sum Games
2. Is the linear system $Ax \geq b$, $x \geq 0$, feasible?
3. Multi-commodity Flow Problems: Give k source-sink pairs, and capacities on edges of a graph, maximize the total flow from each s_i to t_i, respecting the capacity constraint on each edge.
4. LP for bounded number of variables.
5. Approximate set cover.
6. Ada Boost algorithm in ML.
7. Winnow Algorithm - Hedge Algorithm - Weighted Majority Algorithm.

2-Player Zero Sum Games

Matching Pennies

2-Players: Row and Column. Players simultaneously place coin on the table. If both the coins show the same face (HH,TT), Row player wins \$1 from the Column player. Otherwise, loses a dollar to the Column player.

Suppose row (column) player announces their startegy first. Will column (row) player have any advantage?

The fundamental theorem of game theory states that

Minmax Theorem [von Neuman 1928]

No player has an advantage/disadvantage in any finite, zero-sum, two-person game provided they play their optimal mixed strategies.

Traditionally shown using LP Duality.
Alternatively, a straightforward proof using MWU that also provides the Nash Equilibria.

Set Cover Problem

Input: A universe $U = \{1, \dots, n\}$ consisting of n elements.
A set of m subsets $\mathcal{S} = \{S_1, S_2, \dots, S_m\}$ of U such that $\cup_{i=1}^{m} S_i = U$.

Output: Find a minimum number of subsets of $\mathcal{S}$ such that their union covers all the elements of U.

An Example:

$U = \{1, 2, 3, 4\}$.

$S_1 = \{1, 2\}$

$S_2 = \{2, 4\}$

$S_3 = \{1, 4\}$

$S_4 = \{2, 3, 5\}$

$S_5 = \{3, 4\}$

$S_6 = \{1, 3, 4\}$

Possible Set Covers: $\{S_1, S_3, S_4\}, \{S_4, S_5\}, \{S_3, S_4\}$.

Computational Complexity of Set Cover

- $\mathbf{NP}$-Complete.

- $O(\log n)$ approximation using the greedy method - pick the set that covers the most uncovered elements.

- New: Analysis using MWU method.

MWU Setup

- We will have n experts, where expert i is responsible for the coverage of element $i \in U$.

- In each round of MWU, we will choose a set. Let $S_j \in \mathcal{S}$ be the set chosen in the current round. The penalty for the expert i in this round is given by

$$M(i, S_j) = \begin{cases} 1 \text{ if } i \in S_j \\ 0 \text{ otherwise} \end{cases}$$

- Choose $\eta = 1$ (though we had assumed that $\eta < 1/2$ in MWU analysis).

- MWU update rule in round t, where the set S_j is chosen, is given by $w_i^{t+1} = w_i^t(1 - \eta M(i, S_j)) = w_i^t(1 - M(i, S_j))$, for each $i \in \{1, \ldots, n\}$.

- Initially the weight of each element is 1, i.e., $w_i^1 = 1$.

- Note that after the t-th round, all the elements that have been covered by the sets chosen so far, will have a weight of 0.

MWU Setup Contd.

- Given weights $w_1, \ldots, w_n$, we define the distribution $p_i = w_i / \sum_j w_j$.

- For any set S_j, $\sum_{i=1}^{n} p_i M(i, S_j) = \sum_{i \in S_j} p_i =$ proportion of the sum of the weights associated to the uncovered elements in S_j.

- Greedy (or MWU) algorithm picks the set that has the maximum weight. Let S_j be the set chosen in this step.

- Let OPT be the number of sets in an optimal set cover. Observe that there is a set in an optimal solution that covers at least $1/\text{OPT}$ fraction of the weight.

- Thus, by Greedy choice, $w(S_j) \geq 1/\text{OPT}$.

- Hence, the change in the potential in round t, $\Phi^{t+1} < \Phi^t (1 - 1/\text{OPT}) \leq \Phi^t e^{-1/\text{OPT}}$.

MWU Analysis

- We execute the MWU update method till all the elements of U are covered.

- Let $T = \lceil \ln n \rceil \text{OPT}$.

- $\Phi^{T+1} < \Phi^{T} e^{-1/\text{OPT}} < \Phi^{1} e^{-T/\text{OPT}} = ne^{-\lceil \ln n \rceil \text{OPT}/\text{OPT}} = ne^{-\lceil \ln n \rceil} \leq 1$.

- Thus, after $T = \lceil \ln n \rceil \text{OPT}$ rounds, $\Phi^{T+1} = 0$.

- Hence there are no elements left to be covered in U after T rounds. I.e., T is the last round.

- The above MWU based analysis of the greedy algorithm for the set cover problem provides an $O(\log n)$ approximation.

Linear Classifiers using MWU

Given

- $\epsilon > 0$

- k labelled examples as tuples (v_j, l_j), where $v_j \in \Re^n$ is a point (or vector) in n dimensional space, and $l_j \in \{-1, +1\}$, for $j = 1, \ldots, k$.

- There exists a vector $x^* \in \Re^n_+$ (i.e., each coordinate of x^* is non-negative) such $l_j(v_j \cdot x^*) \geq \epsilon$, for $j = 1, \ldots, k$. Moreover $||x^*||_1 = \sum\limits_{i=1}^{n} x_i^* = 1$.

Task:

To find a vector $x \in \Re^n_+$ such that $||x||_1 = 1$ and $l_j(v_j \cdot x) \geq 0$, for $j = 1, \ldots, k$.

MWU Setup

- Let ρ be the absolute value of the largest coordinate among $v_1, \ldots, v_k$.

- Each coordinate of each vector v_i acts as an expert.

- Initialze the weight vector $w^1 = (1, \ldots, 1) \in \Re^n$.

- Set $p^1 = (1/n, \ldots, 1/n) \in \Re^n$. Note that $p_i^1 = w_i^1 / \sum_{j=1}^{n} w_j^1$.

- In each round, pick a vector v_j that satisfies $l_j(p^t \cdot v_j) < 0$.

- Note that if no such vector v_j exists, p^t satisfies the constraints. I.e., for all $j = 1, \ldots, k$, $l_j(p^t \cdot v_j) \geq 0$, and we output $x = p^t$.

- Assume that such a v_j exists, where $l_j(p^t \cdot v_j) < 0$.

- Define the loss function $m^t = -\frac{l_j}{\rho} v_j$.

- Because of the choice of ρ, each coordinate of m^t is in $[-1, +1]$.

Bounds on T

Next we show that after we perform $T = 1, \ldots, 4\frac{\rho^2}{\epsilon^2}\ln n$ rounds, p^T satisfies $l_j(p^T \cdot v_j) \geq 0$ for all $j = 1, \ldots, k$.

- By assumption, there exists an $x^* \in \Re^n_+$ such that $\forall j$: $l_j(v_j \cdot x^*) \geq \epsilon$.

- Since $m^t = -\frac{l_j}{\rho}v_j$, we have $m^t \cdot x^* = -\frac{l_j}{\rho}v_j \cdot x^* \leq -\frac{\epsilon}{\rho}$.

- Since $x^* \in \Re^n_+$ acts like a probability vector, by MWU gurantees, we have

$$\begin{aligned} \sum_{t=1}^{T} m^t \cdot p^t &\leq \sum_{t=1}^{T} m^t \cdot x^* + \frac{\ln n}{\eta} + \eta T \\ &\leq -\frac{\epsilon}{\rho}T + \frac{\ln n}{\eta} + \eta T \end{aligned}$$

- In each step we have been choosing v_j such that $l_j(v_j \cdot p^t) < 0$. Thus, $m^t \cdot p^t = -\frac{l_j}{\rho}(v_j \cdot p^t) > 0$.

$\implies -\frac{\epsilon}{\rho}T + \frac{\ln n}{\eta} + \eta T > 0$.

Bounds on T contd.

- Set $\eta = \epsilon/2\rho$ in $-\frac{\epsilon}{\rho}T + \frac{\ln n}{\eta} + \eta T > 0$ and we obtain

$$-\frac{\epsilon}{\rho}T + \frac{2\rho \ln n}{\epsilon} + \frac{\epsilon}{2\rho}T > 0$$

- Isolating T, we obtain

$$T < \frac{4\rho^2}{\epsilon^2} \ln n$$

- Thus, within $T < \frac{4\rho^2}{\epsilon^2} \ln n$ steps all the inequalities are satisfied and we find the required $x = p^t \in \Re^n_+$.

Linear Programming

Let A be a $n \times m$ matrix, b is a vector of length m,
$\mathcal{P}$ is the convex feasibility region.

Approximate Abstract Feasibility Problem

Let $\epsilon \geq 0$ be an error parameter.

If $z \in \mathcal{P}$ and $Az \geq b$ is feasible
- Report $x \in \mathcal{P}$ such that $\forall i \in \{1, \dots, n\}$: $A_i x \geq b_i - \epsilon$

Else, report infeasibility.

Typically, LP problems have an objective function to minimize/maximize.

There are standard methods that can turn an optimization problem to a decision problem (i.e. the feasibility problem).

Moreover, we assume, we are given an ρ-bounded oracle.

ρ-bounded oracle

The ρ-bounded oracle takes as input a probability distribution $p = (p_1, \ldots, p_n)$, where $\sum_{i=1}^{m} p_i = 1$, on the rows of A and returns the following:

ρ-bounded oracle

If $x \in \mathcal{P}$ and $p^T Ax \geq p^T b$ is feasible,
return $x^* \in \mathcal{P}$ such that $\forall i : |A_i x^* - b_i| \leq \rho$.
Otherwise, return that the system is infeasible.

Note: $p^T Ax \geq p^T b$ is a single inequality.
Its a linear combination of rows of A given by the vector p.
Finding $x \in \mathcal{P}$ that satisfies a single constraint $p^T Ax \geq p^T b$ is an easier problem than satisfying all the constraints of $Ax \geq b$.

Next, we apply the MWU method for solving LPs.
Each constraint (i.e., row i of A) is associated to an expert i, and its cost/regret for day t is given by $m_i^t = \frac{1}{\rho}(A_i x^t - b_i)$.

MWU Method for LP

Step 1: Fix an $\eta \in [0, \min(\frac{1}{2}, \frac{\epsilon}{2\rho})]$ and set $w^1 = (1, \ldots, 1)$.

Step 2: For $t = 1$ to $T = 4\frac{\rho^2 \ln n}{\epsilon^2}$ days do:

1. Compute $p^t = (\frac{w_1^t}{\Phi^t}, \ldots, \frac{w_n^t}{\Phi^t})$, where $\Phi^t = \sum_{i=1}^{n} w_i^t$.
2. Execute the ρ-bounded oracle for p^t. It either returns that the system is infeasible and we STOP or returns the vector x^t.
3. Compute the costs of each expert i by evaluating $m_i^t = \frac{1}{\rho}(A_i x^t - b_i)$. (Note $m_i^t \in [-1, 1]$.)
4. Update weights for each expert i: $w_i^{t+1} = w_i^t(1 - \eta m_i^t)$.

Step 3: If we didn't report infeasibility during the T days of execution, return $\bar{x} = \frac{1}{T}\sum_{t=1}^{T} x^t$ as an approximation to the LP.

Observations

1. If $A_i x^t \geq b_i$, then $m_i^t \geq 0$ and the i-th constraint is satisfied.
   If $A_i x^t < b_i$, then $m_i^t < 0$.
   $\implies$ for the rows of A for which the constraints are satisfied their weights will be smaller compared to the rows for which the constraints are not satisfied.
2. In the next round the unsatisfied rows (experts) will get higher probabilities compared to the satisfied rows. The more unsatisfied the row is, higher its probability, and bigger its proportion in $p^T Ax \geq p^T b$ in the call to ρ-bounded oracle. Hence it has higher chances to get satisfied in future rounds.
3. Suppose we didn't report infeasibility on any of the T days of execution of MWU Method. We claim that $\bar{x} = \frac{1}{T} \sum_{t=1}^{T} x^t$, which is a convex combination (average) of T feasible vectors, is within the polytope $\mathcal{P}$.

Analysis (Contd.)

Claim

Assume that in Step 2, the MWU method didn't report infeasibility for the entire run of the algorithm. Then, $\bar{x} \in \mathcal{P}$

Proof.

By definition, $\bar{x} = \frac{1}{T}\sum_{t=1}^{T} x^t$, where each $x^t \in \mathcal{P}$ by construction.

Convex combination of $x^1, \ldots, x^T$ is also in the convex region $\mathcal{P}$. $\square$

From the analysis of MWU Method we know that the expected cost of this algorithm is bounded with respect to the cost of any expert i by

$$\sum_{t=1}^{T} M^t = \sum_{t=1}^{T} (p^t)^T \cdot m^t \leq \frac{\ln n}{\eta} + \eta T + \sum_{t=1}^{T} m_i^t$$

First we show that M^t's are non-negative.

Analysis (Contd.)

Claim

For each $t = 1, \ldots, T$, $M^t \geq 0$. Moreover, $\sum_{t=1}^{T} M^t \geq 0$.

Proof.

Since $m^t = \frac{1}{\rho}(Ax^t - b)$, we have

$$M^t = (p^t)^T \cdot m^t = \frac{1}{\rho}\left((p^t)^T \cdot (Ax^t - b)\right) = \frac{1}{\rho}((p^t)^T \cdot Ax^t - (p^t)^T \cdot b) \geq 0$$

The last inequality holds as the system is satisfied, i.e. $(p^t)^T Ax^t \geq p^t b$, because of the ρ-bounded oracle in Step 2.. $\square$

Analysis (contd.)

Claim

For each $i = 1, \ldots, n$, $A_i \bar{x} \geq b_i - \epsilon$, where $\bar{x} = \frac{1}{T} \sum_{t=1}^{T} x^t$.

Proof.

- For $t = 1, \ldots, T$, each $M^t \geq 0 \implies \frac{\ln n}{\eta} + \eta T + \sum_{t=1}^{T} m_i^t \geq \sum_{t=1}^{T} M^t \geq 0$.

- Substitute $m_i^t = \frac{1}{\rho}(A_i x^t - b_i)$, we obtain: $\frac{\ln n}{\eta} + \eta T + \frac{1}{\rho} \sum_{t=1}^{T} (A_i x^t - b_i) \geq 0$.

- This is equivalent to $\frac{\ln n}{\eta} + \eta T + \frac{1}{\rho} \sum_{t=1}^{T} A_i x^t - \frac{T}{\rho} b_i \geq 0$.

- Multiply by $\frac{\rho}{T}$ and use $\bar{x} = \frac{1}{T} \sum_{t=1}^{T} x^t$:

$$\frac{\rho \ln n}{T\eta} + \rho\eta + A_i \bar{x} - b_i \geq 0$$

- Substitute $T = 4\frac{\rho^2 \ln n}{\epsilon^2}$ and $\eta \in [0, \min(\frac{1}{2}, \frac{\epsilon}{2\rho})]$ we obtain

$$\epsilon + A_i \bar{x} - b_i \geq 0$$

$\square$

Computational Complexity

- We run the algorithm for T days.

- For each day we make a call to the ρ-bounded oracle.

- The overall time complexity is bounded by the time it takes to run $O(\frac{\rho^2 \ln n}{\epsilon^2})$ calls to the oracle.

- Note that we didn't execute a simplex or any algorithm for LP!

- Except the MWU steps, the main work is carried in designing an approriate ρ-bounded oracle. This depends on the specific LP problem.

ρ-Bounded Oracle for Set Cover LP

Input: A universe U consisting of n elements.
A set of m subsets $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$ of U such that $\cup_{i=1}^{m} S_i = U$.

Output: Find a minimum number of subsets of $\mathcal{S}$ such that their union covers all elements of U.

Let is look at the integer linear program for set cover. Define a $0-1$ indicator variable x_S for each set $S \in \mathcal{S}$, where $x_S = 1$ if and only if S is included in the set cover.

The ILP formulation is:

$$\begin{aligned} \min \sum_{S \in \mathcal{S}} x_S \\ \forall u \in U : \sum_{u \in S} x_S \geq 1 \\ \forall S \in \mathcal{S} : x_S \in \{0,1\} \end{aligned}$$

The fractional linear program (LP) is where we replace the integrality constraint $x_S \in \{0,1\}$ by $0 \leq x_S \leq 1$. Moreover, we can replace $0 \leq x_S \leq 1$ by $x_S \geq 0$ as this is a minimization problem.

Optimization Problem $\rightarrow$ Feasibility Problem

- The value of the objective function in ILP formulation is one of the numbers $\{1, \ldots, m\}$.

- Suppose we guess that the size of the set cover is $\beta \in \{1, \ldots, m\}$.

- If the following feasibility inequality can be satisfied, we know that the optimal value is at most β.

$$\begin{aligned} \forall u \in U : \sum_{u \in S} x_S &\geq 1 \\ \sum_{S \in \mathcal{S}} x_S &\leq \beta \\ \forall S \in \mathcal{S} : x_S &\geq 0 \end{aligned}$$

- We can perform a binary search to find the true optimal value.

Fractional Set Cover Feasibility Problem

Let $\mathcal{P} = \{x \in \Re^m | \sum_{i=1}^{m} x_i \leq \beta \bigwedge \forall i \in \{1, \ldots, m\}, x_i \geq 0\}$.

$\mathcal{P}$ is a convex polytope that defines the feasible region for LP.

We can express the feasibility problem succinctly as follows:

Fractional Set Cover Feasibility Problem

Report $x \in \mathcal{P}$ such that $\forall u \in U : \sum_{u \in S} x_S \geq 1$,

Otherwise, report infeasibility.

We define an approximate abstract feasibility problem as:

Approximate Abstract Feasibility Problem

Let $\epsilon \geq 0$ be an error parameter.

If $z \in \mathcal{P}$ and $Az \geq b$ is feasible

- Report $x \in \mathcal{P}$ such that $\forall i \in \{1, \ldots, n\}$: $A_i x \geq b_i - \epsilon$

Else, report infeasibility.

Approximate Set Cover Feasibility Problem

Define a $0-1$ characteristic matrix A of size $n \times m$. Elements of $\mathcal{U} = \{u_1, \ldots, u_n\}$ form rows and subsets in $\mathcal{S} = \{S_1, \ldots, S_m\}$ form columns.

$$A[i,j] = \begin{cases} 1, & \text{if } u_i \in S_j \\ 0, & \text{otherwise} \end{cases}$$

Approximate Set Cover Feasibility Problem

Input: For a universe U of size n and m-subsets of U, we have
- characteristic matrix A of size $n \times m$,
- vector b of length n consisting of 1's,
- Feasibility region $\mathcal{P} = \{x \in \Re^m | \sum_{i=1}^{m} x_i \leq \beta \wedge \forall i \in \{1, \ldots, m\}, x_i \geq 0\}$.
- Error parameter $\epsilon \geq 0$.

Output:
If $z \in \mathcal{P}$ and $Az \geq b$ is feasible
- report $x \in \mathcal{P}$ such that $A_i x \geq 1 - \epsilon, \forall i \in \{1, \ldots, n\}$,
Else, report infeasibility.

Note: A_i represents the i-th row of matrix A.

ρ-bounded oracles - review

Recall that the ρ-bounded oracle takes as input a probability distribution $p = (p_1, \ldots, p_n)$, where $\sum_{i=1}^{m} p_i = 1$, on the rows of A and returns the following:

ρ-bounded oracle

If $x \in \mathcal{P}$ and $p^T Ax \geq p^T b$ is feasible,
return $x^* \in \mathcal{P}$ such that $\forall i : |A_i x^* - b_i| \leq \rho$.
Otherwise, return that the system is infeasible.

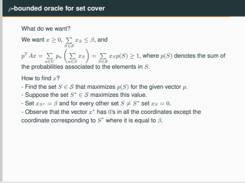
For the set cover problem $p^T b = 1$, as b is the vector of all 1s, and p is a vector of probabilities that add to 1.

ρ-bounded oracle for set cover

What do we want?

We want $x \geq 0$, $\sum\limits_{S \in \mathcal{S}} x_S \leq \beta$, and
$p^T Ax = \sum\limits_{u \in U} p_u \left(\sum\limits_{u \in S} x_S \right) = \sum\limits_{S \in \mathcal{S}} x_S p(S) \geq 1$, where $p(S)$ denotes the sum of the probabilities associated to the elements in S.

How to find x?
- Find the set $S \in \mathcal{S}$ that maximizes $p(S)$ for the given vector p.
- Suppose the set $S^* \in \mathcal{S}$ maximizes this value.
- Set $x_{S^*} = \beta$ and for every other set $S \neq S^*$ set $x_S = 0$.
- Observe that the vector x^* has 0's in all the coordinates except the coordinate corresponding to S^* where it is equal to β.



ρ-bounded oracle for set cover

What do we want?

We want $x \geq 0$, $\sum_{S \in \mathcal{S}} x_S \leq \beta$, and

$p^T Ax = \sum_{u \in U} p_u \left(\sum_{u \in S} x_S \right) = \sum_{S \in \mathcal{S}} x_S p(S) \geq 1$, where $p(S)$ denotes the sum of the probabilities associated to the elements in S.

How to find x?
- Find the set $S \in \mathcal{S}$ that maximizes $p(S)$ for the given vector p.
- Suppose the set $S^* \in \mathcal{S}$ maximizes this value.
- Set $x_{S^*} = \beta$ and for every other set $S \neq S^*$ set $x_S = 0$.
- Observe that the vector x^* has 0's in all the coordinates except the coordinate corresponding to S^* where it is equal to β.

- What are the two equalities in $p^T Ax = \sum_{u \in U} p_u \left(\sum_{u \in S} x_S \right) = \sum_{S \in \mathcal{S}} x_S p(S)$?
- We will interpret the product $p^T Ax$ in two different ways
- Think of each element $u_i \in \mathcal{U}$ has an associated probability p_i.

1st interpretation: Product Ax is a vector of dimension n, where its i-th entry is the number of sets in $\mathcal{S}$ that contain the element u_i.
- $p^T Ax$ is the dot-product of vectors p^T and Ax, where the i-th entry in Ax is multiplied by the probability p_i.
- Thus, $p^T Ax$ is the sum of the products of the probability p_i of element u_i times the number of occurrences of u_i in $\mathcal{S}$.

2nd Interpretation: For each set $S \in \mathcal{S}$ sum up the probabilities associated to each element in that set (this is the quantity $p(S)$).
- We take the sum $p(S)$ over all sets so that for each element we take into account the number of times it occurs in the sets of $\mathcal{S}$.

-Therefore, $p^T Ax = \sum_{u \in U} p_u \left(\sum_{u \in S} x_S \right) = \sum_{S \in \mathcal{S}} x_S p(S)$.

ρ-bounded oracle for set cover (contd.)

x^* is feasible

The vector $x^* = (0, 0, \ldots, \beta, 0, \ldots, 0) \in \mathcal{P}$, as each of its coordinates is ≥ 0 and the sum of the coordinates is $\leq \beta$.

Consider $\sum\limits_{S \in \mathcal{S}} x^*_S p(S)$.

If $\sum\limits_{S \in \mathcal{S}} x^*_S p(S) \geq 1$, we have the x^* that we are looking for.

Claim

If $\sum_{S \in \mathcal{S}} x^*_S p(S) < 1$, then no other $x \in \mathcal{P}$ can satisfy the inequality $\sum\limits_{S \in \mathcal{S}} x_S p(S) \geq 1$.

Proof.

Note that under the constraints ($x \geq 0$, $\sum\limits_{S \in \mathcal{S}} x_S \leq \beta$) the choice of x (=x^*) that maximizes the expression $\sum\limits_{S \in \mathcal{S}} x_S p(S)$ didn't satisfy the inequality. Any other assignment will have a value at most the max value. $\square$

ρ-bounded oracle for set cover (contd.)

What should be the value of ρ?

Find the smallest value ρ such that for all $i \in \{1, \ldots, n\}$, $|A_i x^* - b_i| \leq \rho$.
- Due to the choice of x^* and the matrix A being a $0-1$ matrix, the product $A_i x^*$ is either 0 or β.

- Thus $|A_i x^* - b_i| = |A_i x^* - 1| \leq |\beta - 1| \leq \beta \leq m$.

- Note that $\beta \geq 1$ as the set cover consists of one or more sets to cover $\mathcal{U}$.

- Set $\rho = \min\{\beta, m\}$.

Now we have the required ρ-bounded oracle for the set cover.

ρ-bounded oracle
If $x \in \mathcal{P}$ and $p^T Ax \geq p^T b$ is feasible,
we return $x^* = (0, 0, \ldots, \beta, 0, \ldots, 0) \in \Re^m$ such that $\forall i : |A_i x^* - b_i| \leq \rho$.
Otherwise, we return that the system is infeasible.

Conclusions

Algorithmic Techniques

- Divide-and-Conquer
- Greedy
- Graph Traversal
- Randomization
- Dynamic Programming
- Linear Programming
- $\cdots$
- $\cdots$
- Multiplicative-Weight Update Method

References

- Arora, Hazan and Kale, The multiplicative weights update method: a meta-algorithm and applications, Theory of Computing 8(1): 121-164, 2012.

- Google Scholar's citations for Arora, Hazan and Kale's Survey Paper: (≈ 1000 citations)

- Several Lecture Notes

- MWU is latest addition in CLRS Algorithms textbook's 4th ed. (falls under the Machine Learning chapter.)