An initiative of the study of Algorithmic Game Theory and Computational Economics is to justify equilibrium concepts. To understand what is meant by “justification”, from a pragmatic perspective, we are not satisfied by just knowing that equilibrium exists, but we want to know how equilibrium is reached. There are two aspects of reaching an equilibrium:

- The first aspect is computational efficiency. If we know an equilibrium exists, but we have to spend a century of time to find/compute/reach it, then it is barely satisfiable to say we really reach the equilibrium. This poses a challenge to the concepts of Nash Equilibrium in general-sum games and unsplittable selfish routing games, since we know in general, finding the NE in these games is PLS-complete or PPAD-complete, indicating that efficient algorithms are unlikely to exist.

- The second aspect is method/algorithms to reach an equilibrium in real games/markets. So far, we have seen a number of algorithms for equilibrium computation. For two-person zero-sum and general-sum games, the algorithms involve steps of solving linear systems and linear programs, and also polyhedral walking. Common sense is telling us that these methods are not possible to be executed in real games/markets. Put in in perspective, in real games/markets, we can view them as limited computers, which means they have various unavoidable constraints that prohibit them from performing any kind of computations which we ideally want. Such constraints include
  - incomplete/partial information;
  - highly distributed environment, which prohibits centralized computations;
  - implausibility of complicated arithmetic.

These motivated researchers to investigate the performance of simple and distributed dynamic (which belongs to the broader category of natural algorithm/dynamic), which uses only local information, in real games/markets. These dynamics are our focus in the next few lectures.

1 Learning from History, Choosing the Future — Multiplicative Weight

At this moment, let’s not focus on game for a while, but consider a simpler scenario. A person is having \( n \) choices which she has to pick one of them every day. The choices might be different methods/routes travel from home to office, or which restaurant for lunch. Based on how she was satisfied with the past choices, it is natural that she will adjust her choice from time to time. A perhaps natural way to do is best response.

**Example 1: Best Response Dynamic**

Suppose the person has 2 choices \( \alpha \) and \( \beta \). We assume that every day, she knows the scores of both choices in the past day, but \textit{not} the scores in the current day. If she chooses by best response, then every day her choice is the choice with the higher total score in the past days. In Day 1, her choice is arbitrary. E.g., see the first table below (S stands for Score; TS stands for Total Score):

<table>
<thead>
<tr>
<th>Day</th>
<th>( \alpha ) S</th>
<th>( \beta ) S</th>
<th>( \alpha ) TS</th>
<th>( \beta ) TS</th>
<th>Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.0</td>
<td>8.0</td>
<td>2.0</td>
<td>8.0</td>
<td>( \alpha )</td>
</tr>
<tr>
<td>2</td>
<td>10.0</td>
<td>3.0</td>
<td>12.0</td>
<td>11.0</td>
<td>( \beta )</td>
</tr>
<tr>
<td>3</td>
<td>8.0</td>
<td>7.0</td>
<td>20.0</td>
<td>18.0</td>
<td>( \alpha )</td>
</tr>
<tr>
<td>4</td>
<td>7.0</td>
<td>10.0</td>
<td>27.0</td>
<td>28.0</td>
<td>( \alpha )</td>
</tr>
<tr>
<td>5</td>
<td>1.0</td>
<td>2.0</td>
<td>28.0</td>
<td>30.0</td>
<td>( \beta )</td>
</tr>
</tbody>
</table>

Future is not known in advance. Thus, based on history, we cannot expect that we can always make the \textit{best} choice every day. If she is unlucky, she can make really bad choice every day — see the second table above. Her earned score every day is 0.0.
Another natural way is making choice randomly every day, and the score she earns every day is the expected score w.r.t. the probability distribution which she chooses. It somewhat remedies the bad situation in Example 1 since if the scores 0.0 and 10.0 do alternate daily, it is quite natural for her to choose uniform distribution daily, earning an expected score of 5.0 per day.

Now we are ready to introduce the Multiplicative Weight (MW) Algorithm. Suppose that there are $N$ choices, and the score of every choice is within the interval $[0, 1]$. For simplicity, here we assume that the person knows in advance that she needs to make choices for $T$ days — in the exercise, we will see how to handle the case when $T$ is not known in advance. The parameter $0 < \epsilon < 1$ in the algorithm below will be determined later.

- For $i \in [N]$, set $W_i \leftarrow 1$.
- For Day $t = 1$ to $T$ do:
  - Compute
    $$\Gamma^t \leftarrow \sum_{i=1}^{N} W_i^t.$$  
  - Choose a probability distribution $(p_1^t, p_2^t, \cdots, p_N^t)$ such that for each $i \in [n]$,
    $$p_i^t = \frac{W_i^t}{\Gamma^t}.$$  
  - Observe the score of every choice, and denote them by $s_1^t, s_2^t, \cdots, s_N^t$ respectively.
  - For each $i \in [n]$, set
    $$W_i^{t+1} \leftarrow W_i^t \cdot (1 - \epsilon)^{1-s_i^t}.$$  

We make several observations before going into the analysis. If $\epsilon = 0$, then every day the person chooses a uniform distribution. It would be fine if overall every choice is roughly equally good. However, if some choices are very good overall while the others are very bad, the performance of the algorithm with $\epsilon = 0$ is not satisfactory.

On the other hand, if $\epsilon$ is very large, say when $\epsilon$ tends to $+\infty$, it is getting close to the best response dynamic we discussed in Example 1. Due to the example, we know that setting $\epsilon$ large is not a good choice either.

There is a larger problem: what is our benchmark? This is a problem we deliberately avoid to address until now. We want you to have a better understanding of the situation before seeking and sticking to a reasonable benchmark. Based on our discussion so far, a reasonable benchmark is: suppose that after $T$ rounds, how good is

(a) the total earned expected score of the person

compared to

(b) the total earned expected score if the person hypothetically stick to any fixed probability distribution in all the $T$ rounds

? The fixed probability distribution that leads to the optimal (b) is called the offline optimal. The message is: If the earned score (a), which is achieved by an online algorithm, is quite good compared to the earned score if we were allowed to look back the history and select a fixed optimal choice$^2$, then the online algorithm seems satisfactory.

$^1$As you will see in the analysis, the initial weights $\{W_i\}_{i \in [n]}$ need not be identical. Thus, if the person believes some choices are much better than the others, she might put more weights to such choices initially.

$^2$This is equivalent to selecting a fixed probability distribution.
2 Analysis of Multiplicative Weight Algorithm — No Regret

**Definition 1: Earned Expected Score, Cumulative Score and Regret**

Suppose that the person chooses probability distribution \((p_1^t, p_2^t, \cdots, p_N^t)\) in round \(t\), while the scores of the \(N\) choices are \(s_1^t, s_2^t, \cdots, s_N^t\). Then her **earned expected score** in round \(t\) is

\[
S(t) := \sum_{i=1}^{N} p_i^t \cdot s_i^t,
\]

while her **total earned expected score** in the first \(T'\) rounds is \(\overline{S}(T') := \sum_{t=1}^{T'} S(t)\).

Let the **Cumulative Score** of Choice \(i\) be \(C_i(T') := \sum_{t=1}^{T'} s_i^t\), and let \(C^*(T') := \max_{i \in [N]} C_i(T')\).

An algorithm that chooses \((p_1^t, p_2^t, \cdots, p_N^t)\) in each round \(t\) is said to achieve **regret** \(R(T)\) if the algorithm guarantees that for any sufficiently large \(T\), \(\overline{S}(T) \geq C^*(T) - R(T)\). If \(\lim_{T \to \infty} \frac{R(T)}{T} = 0\), then the algorithm is a **no-regret** algorithm.

---

**Theorem 1: Key Theorem about Multiplicative Weight Algorithm**

If \(T \geq 4 \ln N\), by setting \(\epsilon = \sqrt{\frac{\ln N}{T}} \leq \frac{1}{2}\),

\[
\overline{S}(T) \geq C^*(T) - 2\sqrt{T \ln N},
\]

i.e., the MW Algorithm achieves regret \(2\sqrt{T \ln N}\). Equivalently,

\[
\frac{\overline{S}(T)}{T} \geq \frac{C^*(T)}{T} - 2\sqrt{\ln N} \cdot \frac{1}{\sqrt{T}}.
\]

The second inequality in the theorem takes average over the \(T\) rounds. To appreciate the power of MW Algorithm, observe that if the person does not do anything clever (e.g., if she stays with the uniform distribution all the time), then it is not hard to see that \(\frac{C^*(T)}{T} \geq \Omega(1)\) in the worst case. The theorem states that the MW Algorithm ensures \(\frac{C^*(T)}{T} - \overline{S}(T) \leq O(1/\sqrt{T}) = o(1)\). The message is: by employing the MW algorithm, when \(T\) is large enough, in an average/amortized sense, the person is guaranteed that her earned expected score is no worse than the offline optimal minus a small term. In the proof, we will use two elementary inequalities:

\[
\forall 0 \leq x \leq 1, \quad 1 - x \leq e^{-x} \quad \text{and} \quad \forall 0 \leq x \leq \frac{1}{2}, \quad 1 - x \geq e^{-x - x^2}.
\]

**Proof:** First, write

\[
1 - S(t) = \sum_{i=1}^{N} \frac{W_i}{\Gamma^t} \cdot (1 - s_i^t). \tag{1}
\]

Our target is to show that \(\sum_{t=1}^{T} S(t)\) is large, or equivalently, \(\sum_{t=1}^{T} (1 - S(t)) = T - \sum_{t=1}^{T} S(t)\) is small.

Note that

\[
\Gamma^{t+1} \leq \sum_{i=1}^{N} W_i \cdot (1 - \epsilon)^{1-s_i^t} = \sum_{i=1}^{N} W_i \cdot [1 - \epsilon (1 - s_i^t)] \quad \text{for any } 0 \leq \epsilon, a \leq 1, \ (1 - \epsilon)^a \leq 1 - \epsilon a
\]

\[
= \sum_{i=1}^{N} W_i - \epsilon \cdot \sum_{i=1}^{N} W_i \cdot (1 - s_i(t))
= \Gamma^t - \epsilon \cdot \Gamma^t \cdot (1 - S(t)) \quad \text{(by (1))}
= \Gamma^t \cdot [1 - \epsilon \cdot (1 - S(t))].
\]
Hence,
\[ \frac{\Gamma^{t+1}}{\Gamma^t} \leq 1 - \epsilon \cdot (1 - S(t)) \leq e^{-\epsilon \cdot (1 - S(t))} \]
and
\[ \Gamma^{T+1} = \frac{\Gamma^{T+1}}{\Gamma^T} \cdot \frac{\Gamma^T}{\Gamma^{T-1}} \cdot \ldots \cdot \frac{\Gamma^2}{\Gamma^1} \leq e^{-\epsilon \cdot (T - S(T))} \cdot N. \] (2)

On the other hand,
\[ \Gamma^{T+1} = \sum_{i=1}^{N} W_i^T \geq (1 - \epsilon)^T - C^*(T) \geq (e^{\epsilon - \epsilon^2})^T - C^*(T) = e^{-(\epsilon + \epsilon^2)(T - C^*(T))}. \] (3)

Combining (2) and (3) yields
\[ - (\epsilon + \epsilon^2)(T - C^*(T)) \leq - \epsilon \cdot (T - S(T)) + \ln N \]
\[ \epsilon \cdot C^*(T) - \epsilon^2 \cdot (T - C^*(T)) \leq \epsilon \cdot S(T) + \ln N \]
\[ C^*(T) - \epsilon \cdot (T - C^*(T)) \leq S(T) + \frac{\ln N}{\epsilon} \]

Rearranging terms yields
\[ S(T) \geq C^*(T) - \epsilon \cdot (T - C^*(T)) - \frac{\ln N}{\epsilon} \geq C^*(T) - \epsilon \cdot T - \frac{\ln N}{\epsilon}. \]

By setting \( \epsilon = \sqrt{\frac{\ln N}{T}} \), the above inequality implies that
\[ S(T) \geq C^*(T) - 2\sqrt{T \ln N}. \]

## 3 Multiplicative Weight Algorithm in Two-Person Zero-Sum Games

As a first application of MW Algorithm, we look at two-person zero-sum games. Suppose that both players Alice and Bob employ the MW Algorithm to play the game. There are two models — the first has highly informational requirement and is less plausible (but easier to analyze), while the second has low informational requirement and is more plausible (but more technical to analyze). In this lecture, we will use the first model.

- In each round, each player declares a probability distribution over her/his own strategy sets. Then Alice’s payoff in that round is \( V_A(\vec{p}, \vec{q}) \), and Bob’s payoff in that round is \( V_B(\vec{p}, \vec{q}) \).

They can observe the expected payoff of each of her/his strategy, i.e., in each round, Alice observes \( V_A^i(\vec{q}) \) for all \( i \in [n] \), and Bob observes \( V_B^j(\vec{p}) \) for all \( j \in [n] \).

Note that Alice does not observe \( \vec{q} \), and Bob does not observe \( \vec{p} \).

- In each round, each player has a probability distribution over her/his own strategy sets in her/his mind, but each round each player only declares one pure strategy, and the payoff to each player is according to their choices of pure strategies — this payoff is also the only thing they can observe.

### Theorem 2: No-Regret Algorithm in Two-Person Zero-Sum Game

If both players in a two-person zero-sum game employ a no-regret algorithm, then
\[ \lim_{T \to \infty} \frac{1}{T} \cdot \sum_{t=1}^{T} V_A(\vec{p}^t, \vec{q}^t) \]
equals to the game value of the zero-sum game.
Proof: Recall that the game value is equal to the lower game value \( u \), by the Minimax Theorem. By the definition of lower game value, there exists a probability distribution that Alice can choose to guarantee her a payoff of at least \( u \) in each round. Thus, \( C^*(T) \geq u \cdot T \), and hence a no-regret algorithm we guarantee her that

\[
\frac{1}{T} \cdot \sum_{t=1}^{T} V_A(p^t, q^t) \geq u - o_T(1).
\]

Symmetrically, consider from the Bob’s perspective yields

\[
\frac{1}{T} \cdot \sum_{t=1}^{T} V_B(p^t, q^t) \geq -u - o_T(1).
\]

Since \( V_A(p^t, q^t) = -V_B(p^t, q^t) \) for all \( t \), we have

\[
u - o_T(1) \leq \frac{1}{T} \cdot \sum_{t=1}^{T} V_A(p^t, q^t) = \frac{1}{T} \cdot \sum_{t=1}^{T} \left[ -V_B(p^t, q^t) \right] \leq u + o_T(1). \quad \square
\]

Before stating the next theorem about two-person zero-sum games, we introduce the following notion of approximated NE.

**Definition 2: Approximated Nash Equilibrium**

The pair \((\vec{p}, \vec{q})\) forms a \(\delta\)-Nash Equilibrium (\(\delta\)-NE) if each player cannot raise her/his own payoff by more than \(\delta\) by changing her/his own mixed strategy, while the other player remains fixed on his/her choice of mixed strategy. Mathematically, it is formally defined as:

\[
\forall \vec{p}, \ V_A(\vec{p}, \vec{q}) \geq V_A(\vec{p}, \vec{q}) - \delta \quad \text{and} \quad \forall \vec{q}, \ V_B(\vec{p}, \vec{q}) \geq V_B(\vec{p}, \vec{q}) - \delta.
\]

**Theorem 3: No-Regret Algorithm in Two-Person Zero-Sum Game Converges Empirically to \(\delta\)-NE**

If both players in a two-person zero-sum game employ a no-regret algorithm, then let

\[
\vec{p}^T = \frac{1}{T'} \cdot \sum_{t=1}^{T'} p^t \quad \text{and} \quad \vec{q}^T = \frac{1}{T'} \cdot \sum_{t=1}^{T'} q^t
\]

be the empirical average at time \(T'\).

For any \(\delta > 0\), there exists a time \(T_{\delta}\) such that for all \(T' > T_{\delta}\), \(\left(\vec{p}^T, \vec{q}^T\right)\) is a \(\delta\)-NE.

Proof: Observe that the function \(V_A^i(\vec{q})\) is a linear function of \(\vec{q}\); thus,

\[
V_A^i(\vec{q}^T) = \frac{1}{T'} \cdot \sum_{t=1}^{T'} V_A^i(\vec{q}),
\]

which, when written using the regret-side notation introduced in this lecture, is \(\frac{1}{T'} \cdot C_i(T')\) for Alice. Thus, \(\bar{V}_A(\vec{q}^T) = \frac{1}{T'} \cdot C^*(T')\).

Since Alice employs a no-regret algorithm, by the definition of no-regret algorithm and by Theorem 2 for any \(\delta > 0\), there exists a \(T_{\delta}\) such that for any \(T' > T_{\delta}\),

\[
u + \delta/4 \geq \frac{1}{T'} \cdot \sum_{t=1}^{T'} V_A(p^t, q^t) \geq \bar{V}_A(\vec{q}^T) - \delta/4,
\]
i.e., $V_A(\vec{q}^T) \leq u + \delta/2$. In other words,

$$V_A(\vec{p}, \vec{q}^T) \leq u + \delta/2.$$  \hspace{1cm} (4)

Symmetrically, for any $\vec{q}$, $V_B(\vec{p}^T, \vec{q}) \leq -u + \delta/2$, and hence $V_A(\vec{p}^T, \vec{q}) \geq u - \delta/2$. In particular,

$$V_A(\vec{p}^T, \vec{q}) \geq u - \delta/2.$$  \hspace{1cm} (5)

Inequalities (4) and (5) complete the proof.

The empirical convergence in Theorem 8 might not strike you as a natural convergence notion. It is reasonable to ask whether stronger convergence notion like “point-wise convergence” holds in the same setting. Unfortunately, the answer is no.

**Example 2: MW Algorithm in a Two-Person Zero-Sum Game**

Alice and Bob are playing the following two-person zero-sum game:

$$
\begin{bmatrix}
1 & 2 & 3 \\
1 & 1.0 & 0.1 & 0.1 \\
2 & 0.0 & 0.8 & 0.5 \\
3 & 0.5 & 0.4 & 1.0
\end{bmatrix}
$$

Suppose that both of them employ the MW Algorithm to play this game with $\epsilon = 0.005$, for $T = 50000$ times. Note that the payoff of Bob is within the range $[-1, 0]$ instead of the range $[0, 1]$ required in the MW Algorithm; thus Bob needs to add one to the payoffs for using the MW Algorithm.

There is a unique NE, at

$$\vec{p} = \left( \frac{5}{14}, \frac{6}{14}, \frac{3}{14} \right) \approx (0.357, 0.429, 0.214) \quad \text{and} \quad \vec{q} = \left( \frac{51}{126}, \frac{70}{126}, \frac{5}{126} \right) \approx (0.405, 0.556, 0.040).$$

The six plots in Figure 4 are the values of $p_1, p_2, p_3$ and $q_1, q_2, q_3$ during the process.

The empirical averages after $T$ rounds are

$$\hat{p} \approx (0.3278, 0.4005, 0.2717) \quad \text{and} \quad \hat{q} \approx (0.4033, 0.5579, 0.0388).$$

Anyway, Theorem 8 provides a very simple algorithm for computing a $\delta$-NE of a zero-sum game, namely executing the MW algorithms for sufficiently long time, and then take the empirical average. This avoids the relatively heavy machinery of linear programming which we used in previous lectures to compute an exact NE.

However, one should beware that a $\delta$-NE for some small $\delta > 0$ is not necessarily close to an exact NE.

4 Multiplicative Weight Algorithms in General Games, and Coarsely Correlated Equilibrium

We have seen a few interesting results about MW algorithms employed in two-person zero-sum games. Naturally, the next question to ask is about general games.

Recall that we discuss in the beginning of this lecture that NE is challenged due to its intrinsic computational hardness. This motivates researchers to look at weaker but more computationally tractable equilibrium (or solution) concepts. One such well received concept is called correlated equilibrium, proposed by Aumann in 1974. Very briefly, it concerns scenarios in which players receive advice from a trusted
Figure 1: The values of $p_1, p_2, p_3$ (left column) and $q_1, q_2, q_3$ (right column) when both players in the game in Example 2 employ MW Algorithm with $\epsilon = 0.005$, and repeat the game for $T = 50000$ times.
coordinator about what strategy to play. The coordinator’s advice forms a correlated equilibrium if no individual player has an incentive to deviate from the advice she has received, if she believes the other players are following the coordinator’s advice. It is known that correlated equilibrium can be efficiently computable via linear programming.

Our focus in this section, however, is a even weaker equilibrium concept, called coarsely correlated equilibrium. In the definition of Mixed NE, every player has a probability distribution over her own strategy set \( S_i \). In the definition of coarsely correlated equilibrium, there is a joint probability distribution over \( \prod_i S_i \). The joint probability distribution might be thought as the coordinator we just mentioned.

**Definition 3: Coarsely Correlated Equilibrium**

In a game of \( n \) players, let \( S_i \) denote the strategy set of Player \( i \). Suppose that all payoff values are bounded in \([0, 1]\). For any \( \delta > 0 \), a probability distribution \( P \) over \( \prod_{i=1}^n S_i \) is a \( \delta \)-coarsely correlated equilibrium (\( \delta \)-CCE) if for every Player \( i \) and for any pure strategy \( \tilde{s}_i \in S_i \),

\[
E_{\vec{s} \sim P} [u_i(\vec{s})] \geq E_{\vec{s} \sim P} [u_i(\tilde{s}_i, \vec{s}_{-i})] - \delta.
\]

\( P \) is said to be a coarsely correlated equilibrium if it is a 0-CCE.

**Theorem 4: No-Regret Algorithm in General Game Converges Empirically to \( \delta \)-CCE**

In a game of \( n \) players, suppose all players employ a no-regret algorithm and play the game for \( T \) times. Let \( p_{t,i}^j \) denote the probability distribution declared by Player \( i \) at time \( t \), and for any \( s_i \in S_i \), let \( \tilde{p}_{i,s_i}^j \) denote the probability that \( s_i \) is chosen w.r.t. the probability distribution \( p_{t,i}^j \).

The empirical joint probability distribution \( P \) is given as below. For any \( \vec{s} \in \prod_{i=1}^n S_i \), let

\[
P_{\vec{s}}^{T} := \frac{\sum_{t=1}^{T} \prod_{i=1}^n p_{t,i,s_i}^j}{T^n}.
\]

Then for any \( \delta > 0 \), there exists a time \( T_\delta \) such that for all \( T' > T_\delta \), \( P_{\vec{s}}^{T'} \) is a \( \delta \)-CCE.

Theorem 4 is almost a direct consequence of Definitions 1 and 3. We leave its proof as an exercise to you. To end this lecture, we give a definition of correlated equilibrium, and then compare it with CCE.

**Definition 4: Correlated Equilibrium**

In a game of \( n \) players, let \( S_i \) denote the strategy set of Player \( i \). Suppose that all payoff values are bounded in \([0, 1]\). For any \( \delta > 0 \), a probability distribution \( P \) over \( \prod_{i=1}^n S_i \) is a \( \delta \)-correlated equilibrium (\( \delta \)-CE) if for every Player \( i \) and for any pure strategy \( s^\#, \tilde{s}_i \in S_i \),

\[
E_{\vec{s} \sim P} \left[ u_i(\vec{s}) \left| s_i = s^\# \right. \right] \geq E_{\vec{s} \sim P} \left[ u_i(\tilde{s}_i, \vec{s}_{-i}) \left| s_i = s^\# \right. \right] - \delta.
\]

\( P \) is said to be a correlated equilibrium if it is a 0-CE.

At first glance, it is not easy to understand what the definitions of CCE and CE really mean. Here is a less quantitative intuition. Suppose that in both case, \( P \) is known commonly.

- For CE, the concern is the following question: If Player \( i \) receives an advice to choose \( s^\# \in S_i \), but she does not know what advices the other players get, should Player \( i \) follow or deviate?

- For CCE, the concern is the following question: Now, the information available to Player \( i \) is even more restricted. Before receiving any advice, she has to choose to follow the advice, or deviate to some other (mixed) strategy irrespective to the advice she will receive.

We leave to you as an exercise to prove the following: every NE is a CE, and every CE is a CCE.