15-888 Computational Game Solving (Fall 2025) Monday, September 8th, 2025

Lecture 4
Equilibrium Computation in Normal-form Games: Linear Programming and Online Learning

Ioannis Anagnostides

Today's lecture takes a deeper dive into normal-form (aka. strategic-form) games and some basic algorithms for computing equilibria.

In more detail, this lecture will cover the following topics. To begin with, we will consider the special case of (two-player) zero-sum games (Section 1). We will see how to phrase the problem of computing minimax strategies—equivalently, Nash equilibria—as a linear program. While this reduction provides a polynomial-time algorithm for computing minimax strategies, more often than not it will not be the most scalable avenue. This motivates considering simpler iterative algorithms for solving zero-sum games (Section 2), leading to the framework of online learning and regret minimization (Section 3). As we shall see as a recurrent theme throughout the course, many of the recent breakthrough results in solving large-scale games rely on the latter class of algorithms.

1 Minimax equilibria through linear programming

The main subject of this section is the reduction from minimax equilibria in zero-sum games to linear programming; in fact, the other direction also holds—zero-sum games are equivalent to linear programming [von Stengel, 2024]—although we will not cover it here. In what follows, we will focus on normal-form games.

Normal-form games can be thought of as modeling simultaneous-move games, although the underlying strategic interaction can itself be sequential. Any finite game can be cast in normal form, but, as we shall discuss more in the next lecture, the normal-form representation is often inefficient.

Formally, we have a set of n players. Each player $i \in [n]$ has a finite set of available actions $\mathcal{A}_i$; we will denote by $m_i \coloneqq |\mathcal{A}_i|$ the number of actions. Every player $i \in [n]$ has a *utility function* u_i mapping any joint action profile $(a_1, \dots, a_n) \in \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$ to a real value $u_i(a_1, \dots, a_n)$. One way of representing a normal-form game is by explicitly providing each player's payoff tensor, containing the utility for each possible action combination; this grows exponentially with the number of players. Player can randomize by specifying a probability distribution over their available actions. Under a joint strategy $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_n) \in \Delta(\mathcal{A}_1) \times \cdots \times \Delta(\mathcal{A}_n)$, the *expected utility* of player $i \in [n]$ reads $u_i(\boldsymbol{x}_1, \dots, \boldsymbol{x}_n) \coloneqq \mathbb{E}_{(a_1,\dots,a_n)\sim(\boldsymbol{x}_1,\dots,\boldsymbol{x}_n)} u_i(a_1, \dots, a_n) = \sum_{(a_1,\dots,a_n)\in\mathcal{A}_1\times\cdots\times\mathcal{A}_n} \prod_{i'=1}^{n} \boldsymbol{x}_{i'}[a_{i'}] u_i(a_1, \dots, a_n)$. It is assumed that players strive to maximize their expected utility.

Zero-sum games A (two-player) zero-sum game is the special case in which there are only two players with exactly opposing interests: $u_1(a_1, a_2) = -u_2(a_1, a_2)$ for all $(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2$; that is, whatever one players wins, the other player loses. We will now see how to solve a zero-sum game.

Let $\mathbf{A} \in \mathbb{R}^{m_1 \times m_2}$ be the payoff matrix of the game. Taking the perspective of the row player, let's consider the following optimization problem.

$$\min_{\boldsymbol{x} \in \Delta(\mathcal{A}_1)} \max_{\boldsymbol{y} \in \Delta(\mathcal{A}_2)} \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y} = \sum_{a_1 \in \mathcal{A}_1} \sum_{a_2 \in \mathcal{A}_2} \boldsymbol{x}[a_1] \mathbf{A}[a_1, a_2] \boldsymbol{y}[a_2]. \tag{1}$$

This is to be interpreted as follows. The row player first specifies a strategy $\boldsymbol{x}$, whereupon the column player responds optimally to that strategy. The key point here is that the row player anticipates that the column player will respond optimally to $\boldsymbol{x}$. This makes solutions to (1) particularly robust, as it makes no difference whether the row player announces its strategy in advance. The famous minimax theorem of von Neumann [1928] shows that, under optimal play, playing second does not confer an advantage.

Theorem 1.1 (von Neumann, 1928)**.** *For any matrix* $\mathbf{A} \in \mathbb{R}^{m_1 \times m_2}$,

$$\min_{\boldsymbol{x} \in \Delta(\mathcal{A}_1)} \max_{\boldsymbol{y} \in \Delta(\mathcal{A}_2)} \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y} = \max_{\boldsymbol{y} \in \Delta(\mathcal{A}_2)} \min_{\boldsymbol{x} \in \Delta(\mathcal{A}_1)} \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y} = v. \tag{2}$$

More general versions of the minimax theorem have been proven (*e.g.*, Sion, 1958). v, as defined in (1), is called the *value* of the game. Theorem 1.1 shows that zero-sum games have a well-defined value; this is not so in general-sum games.

Remark 1.2 (Minimax equilibria are Nash equilibria). Let $\boldsymbol{x} \in \operatorname{argmin}_{\boldsymbol{x}' \in \Delta^{m_1}} \max_{\boldsymbol{y}' \in \Delta^{m_2}} \langle \boldsymbol{x}', \mathbf{A}\boldsymbol{y}' \rangle$ and $\boldsymbol{y} \in \operatorname{argmax}_{\boldsymbol{y}' \in \Delta^{m_2}} \min_{\boldsymbol{x}' \in \Delta^{m_1}} \langle \boldsymbol{x}', \mathbf{A}\boldsymbol{y}' \rangle$ be a pair of minimax strategies. It readily follows that $(\boldsymbol{x}, \boldsymbol{y})$ is a Nash equilibrium. Indeed, Theorem 1.1 implies that $\langle \boldsymbol{x}, \mathbf{A}\boldsymbol{y}' \rangle \leq v = \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y}$ for any $\boldsymbol{y}'$ and $\langle \boldsymbol{x}', \mathbf{A}\boldsymbol{y} \rangle \geq v = \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y}$ for any $\boldsymbol{x}'$; the other direction also holds. We will simply refer to $(\boldsymbol{x}, \boldsymbol{y})$ as an equilibrium.

Remark 1.3 (Generic zero-sum games have a unique equilibrium). A zero-sum game can admit multiple equilibria. Yet, the set of games with a unique equilibrium has measure one [van Damme, 1991]; this informally means that if one samples a game from a distribution of games, that game will almost surely have a unique equilibrium.

Minimax equilibria via linear programming In this context, the main algorithmic question is how to efficiently solve (1). We will now see how to do so through the use of linear programming.

The first point is that (1) is equivalent to $\min_{\boldsymbol{x} \in \Delta(\mathcal{A}_1)} \max_{a_2 \in \mathcal{A}_2} \boldsymbol{x}^\top \mathbf{A}[:, a_2]$, where $\mathbf{A}[:, a_2]$ denotes the column of $\mathbf{A}$ corresponding to a_2; this holds because the column player can always best respond optimally through a pure strategy. We introduce an auxiliary variable $t \in \mathbb{R}$ that satisfies the constraints $t \geq \boldsymbol{x}^\top \mathbf{A}[:, a_2]$ for all $a_2 \in \mathcal{A}_2$. This is intended to implement $\max_{a_2 \in \mathcal{A}_2} \boldsymbol{x}^\top \mathbf{A}[:, a_2]$, but so far we have only guaranteed $t \geq \max_{a_2 \in \mathcal{A}_2} \boldsymbol{x}^\top \mathbf{A}[:, a_2]$. This can be resolved by making the

objective of the linear program to be $\min t$, so that an optimal solution t is always forced to be exactly equal to $\max_{a_2 \in \mathcal{A}_2} \boldsymbol{x}^\top \mathbf{A}[:, a_2]$. In summary, the induced linear program reads

$$\begin{aligned} \text{minimize} \quad & t \\ \text{subject to} \quad & t \geq \boldsymbol{x}^\top \mathbf{A}[:, a_2] \text{ for all } a_2 \in \mathcal{A}_2, \\ & \sum_{a_1 \in \mathcal{A}_1} \boldsymbol{x}[a_1] = 1, \\ & \boldsymbol{x} \geq 0. \end{aligned} \tag{P}$$

We claim that if $(t, \boldsymbol{x})$ is an optimal solution to (P), then $\boldsymbol{x}$ constitutes a solution to (1). We can apply the same reasoning to cast $\max_{\boldsymbol{y} \in \Delta(\mathcal{A}_2)} \min_{\boldsymbol{x} \in \Delta(\mathcal{A}_1)} \boldsymbol{x}^\top \mathbf{A} \boldsymbol{y}$, which is the problem faced by the column player, into the following linear program.

$$\begin{aligned} \text{maximize} \quad & t \\ \text{subject to} \quad & t \leq \boldsymbol{y}^\top \mathbf{A}[a_1, :] \text{ for all } a_1 \in \mathcal{A}_1, \\ & \sum_{a_2 \in \mathcal{A}_2} \boldsymbol{y}[a_2] = 1, \\ & \boldsymbol{y} \geq 0. \end{aligned} \tag{D}$$

The upshot now is that (D) is the dual of (P), which immediately implies Theorem 1.1. In other words, the minimax theorem is a consequence of linear programming duality.

Theorem 1.4. *There is a polynomial-time algorithm for finding equilibria in zero-sum games.*

2 Simple iterative algorithms

Notwithstanding Theorem 1.4, algorithms for solving linear programs tend to scale poorly in very large instances. We thus turn our attention to simpler iterative algorithms. Iterative algorithms for solving zero-sum games have a long history from the early stages of game theory, going back to von Neumann himself [Brown and von Neumann, 1950].

2.1 Best-response dynamics

Perhaps the most natural approach for solving a zero-sum game consists of letting the players best respond to each other's strategy in an alternating fashion. However, in general, this will fail to converge. Indeed, some zero-sum games do not even admit pure equilibria, so this algorithm is doomed to fail in such games. Let's look at a concrete example in the rock-paper-scissors game. We denote "rock" by R, "paper" by P, and "scissors" by S. Suppose we start from (R, P). Then a sequence of best-response dynamics is (S, P) → (S, R) → (P, R) → (P, S) → (R, S) → (R, P). So we ended up at the same point as the one we started at, and this will cycle *ad infinitum*.

Nevertheless, if we look at the average strategies in the previous example, we find that it does converge to the equilibrium—each action is asymptotically played with probability 1/3. Is

this the case more generally? Not so. For example, one can tweak rock-paper-scissors so that best-response dynamics behave the same—by maintaining the *relative* order of the actions—but the equilibrium is different.

Proposition 2.1. *There are zero-sum games in which the average strategies under best-response dynamics fail to converge to an equilibrium.*

2.2 Fictitious play

A more sophisticated, classic algorithm is *fictitious play*. Originally proposed by Brown [1951], it was shown to converge by Robinson [1951]. (This is the same Julia Robinson who contributed to the resolution of Hilbert's 10th problem.)

Fictitious play proceeds as follows. Let's say we initialize from some arbitrary strategies $(\boldsymbol{x}^{(1)}, \boldsymbol{y}^{(1)})$. For a time $t \geq 1$, let $\bar{\boldsymbol{x}}^{(t)} = \sum_{\tau=1}^{t} \boldsymbol{x}^{(\tau)}/t$ be the *average* strategy of the row player and $\bar{\boldsymbol{y}}^{(t)}$ the average strategy of the column player. At every time t, the row player selects a best-response action against the *average opponent* up to that point; namely, $\boldsymbol{x}^{(t)} \in \operatorname{argmin}_{\boldsymbol{x} \in \Delta(\mathcal{A}_1)} \langle \boldsymbol{x}, \mathbf{A}\bar{\boldsymbol{y}}^{(t-1)} \rangle$. Similarly, $\boldsymbol{y}^{(t)} \in \operatorname{argmax}_{\boldsymbol{y} \in \Delta(\mathcal{A}_2)} \langle \bar{\boldsymbol{x}}^{(t-1)}, \mathbf{A}\boldsymbol{y} \rangle$. One can think of this as basic opponent modeling: we don't know the opponent's next action, but we imagine a "fictitious" opponent whose next action adheres to the historical average. Unlike best-response dynamics, the time average of fictitious play converges in every zero-sum game.

Theorem 2.2 (Robinson, 1951)**.** *The time average of fictitious play converges in zero-sum games.*

The original paper of Robinson only proved the convergence asymptotically. The proof relies on a delicate inductive argument, from which one can derive a rate of $O(t^{-\frac{1}{m_1+m_2-2}})$. Daskalakis and Pan [2014] later proved that the rate of convergence can indeed be exponentially slow: for a certain $m \times m$ payoff matrix—in fact, the identity one—fictitious play will converge at a rate of $\Omega(t^{-\frac{1}{m}})$. One caveat of that result is that it only holds for adversarial tie-breaking. The convergence rate of fictitious play for non-adversarial tie-breaking remains an open problem (for some recent progress, see Wang, 2025).

3 Online learning, regret minimization, and self-play

While fictitious play always converges to an equilibrium (Theorem 2.2), this holds only when both players follow that algorithm. On the flip side, fictitious play is *exploitable*: if one's opponent knows that one is using fictitious play, it is easy to exploit it; we will formalize this in Proposition 3.2. This section introduces the framework of online learning and *regret minimization* to reason about such questions. Along the way, we will also develop algorithms with superior convergence guarantees compared to fictitious play. As we shall see in Sections 3.2 and 3.3, the key connection between online learning and games is that regret minimization in *self-play* converges to different types of equilibria.

3.1 Online learning and regret

Consider a *learner* who is to make a sequence of decisions over T rounds. The learner interacts with an *environment*, which intends to model a potentially adversarial player. In each round $t \in [T]$, the learner specifies a mixed strategy $\boldsymbol{x}^{(t)} \in \Delta(\mathcal{A})$. (More broadly, the learner could select a strategy from a general convex and compact set.) The environment thereupon selects a *utility vector* $\boldsymbol{u}^{(t)}$, so that the utility obtained by the learner at that round is $\langle \boldsymbol{x}^{(t)}, \boldsymbol{u}^{(t)} \rangle$. An online algorithm produces a sequence of strategies based on the feedback observed so far. The goal of the learner is to maximize the collected reward.

What's a sensible way of measuring the performance of the learner in this online environment? There are different notions of *hindsight rationality*. Perhaps the most common performance benchmark is *regret*, defined as

$$\mathrm{Reg}^{(T)} \coloneqq \max_{\boldsymbol{x} \in \Delta(\mathcal{A})} \left\{ \sum_{t=1}^{T} \langle \boldsymbol{x}, \boldsymbol{u}^{(t)} \rangle \right\} - \sum_{t=1}^{T} \langle \boldsymbol{x}^{(t)}, \boldsymbol{u}^{(t)} \rangle. \tag{3}$$

The second term in (3) is the cumulative utility of the learner through the T rounds, whereas the first term in (3) is the optimal utility that could have been obtained in hindsight *through a fixed strategy*. Naturally, the learner strives to minimize regret. In a later course, we will introduce more powerful notions of regret.

Remark 3.1. An observation that will be relevant in a later lecture is that *regret can be negative*, which means that *all fixed* strategies yield inferior utility in hindsight than what obtained by the learner. A simple example where this is so occurs when $\boldsymbol{x}^{(t)} = \boldsymbol{u}^{(t)} = (1, 0)$ for $T = 0 \mod 2$ and $\boldsymbol{x}^{(t)} = \boldsymbol{u}^{(t)} = (0, 1)$ for $T = 1 \mod 2$.

Is fictitious play no-regret? An online algorithm is referred to as *no-regret* if $\mathrm{Reg}^{(T)} = o(T)$ *under any sequence of utilities*, which can be potentially adversarially chosen; we will be mostly interested in algorithms that produce strongly sublinear regret of the form $T^{1-\alpha}$ for a constant $\alpha \in (0, 1]$. Now, fictitious play, as introduced in Section 2.2, is not just an algorithm for solving a zero-sum game; it can also be thought of as an online learning algorithm—in that literature, it mostly goes by *follow the leader*. Specifically, in every round $t \in [T]$, it prescribes choosing

$$\boldsymbol{x}^{(t)} \in \operatorname*{argmax}_{\boldsymbol{x} \in \Delta(\mathcal{A})} \sum_{\tau=1}^{t-1} \langle \boldsymbol{x}, \boldsymbol{u}^{(\tau)} \rangle. \tag{4}$$

But does it have the no-regret property? As alluded to, the answer is no.

Proposition 3.2. *There is a sequence of utilities $(\boldsymbol{u}^{(t)})_{t=1}^{T}$ such that fictitious play incurs $\Omega(T)$ regret.*

In proof, let's suppose that fictitious play is initialized at the first action. Consider then the following sequence of utilities.

$$\boldsymbol{u}^{(t)} = \begin{cases} (\epsilon, 0) & \text{if } t = 1, \\ (0, 1) & \text{if } t = 0 \mod 2, \text{ and} \\ (1, 0) & \text{otherwise.} \end{cases}$$

It's not hard to see that fictitious play incurs linear regret under this sequence of utilities: it only obtains a nonzero utility of ϵ in the first round, whereas *any fixed strategy* in hindsight accrues roughly $T/2$.

Follow the regularized leader To address this deficiency of fictitious play, we will introduce a small twist to (4); namely,

$$\boldsymbol{x}^{(t)} = \operatorname*{argmax}_{\boldsymbol{x}\in\Delta(\mathcal{A})} \left\{ \sum_{\tau=1}^{t-1} \langle \boldsymbol{x}, \boldsymbol{u}^{(\tau)} \rangle - \frac{1}{\eta}\mathcal{R}(\boldsymbol{x}) \right\}, \tag{5}$$

where $\eta > 0$ is the *learning rate* and $\mathcal{R}$ is a strictly convex regularizer. The online algorithm (5) is known as *follow the regularized leader (*FTRL*)* [Kalai and Vempala, 2005]; the strategies are well-defined since the underlying maximization problem is strictly concave. The basic intuition of FTRL is that it counterbalances fictitious play with a regularizer that incentives the strategies to be more mixed; injecting more randomness into one's strategy should make it harder for the adversary to exploit the learner. The learning rate η regulates how much the algorithm should be mixing. At the extreme where $\eta = +\infty$ in (5), one recovers fictitious play.

Example 3.3 (Multiplicative weights update). Let $\mathcal{R} : \boldsymbol{x} \mapsto \sum_{a\in\mathcal{A}} \boldsymbol{x}[a] \ln \boldsymbol{x}[a]$ be the (negative) *entropy regularizer*. It's not hard to show that the solution to (5) is $\boldsymbol{x}^{(t)}[a] \propto \exp\left(\eta \sum_{\tau=1}^{t-1} \boldsymbol{u}^{(\tau)}[a]\right)$. This is known as *multiplicative weights update (*MWU*)* (also known as exponential weights or randomized weighted majority), and is perhaps the most well-studied online algorithm.

Example 3.4 (Euclidean regularizer). Let $\mathcal{R} : \boldsymbol{x} \mapsto \frac{1}{2}\sum_{a\in\mathcal{A}}(\boldsymbol{x}[a])^2 = \frac{1}{2}\|\boldsymbol{x}\|_2^2$ be the Euclidean regularizer. The solution to (5) is $\boldsymbol{x}^{(t)} = \Pi_{\Delta(\mathcal{A})}\left(\eta \sum_{\tau=1}^{t-1} \boldsymbol{u}^{(\tau)}\right)$, where $\Pi_{\Delta(\mathcal{A})}(\cdot)$ denotes the Euclidean projection operator.

Let $R \coloneqq \max_{\boldsymbol{x}\in\Delta(\mathcal{A})} \mathcal{R}(\boldsymbol{x}) - \min_{\boldsymbol{x}\in\Delta(\mathcal{A})} \mathcal{R}(\boldsymbol{x})$ be the *range* of the regularizer $\mathcal{R}$. We further assume that $\mathcal{R}$ is 1-strongly convex with respect to a norm $\|\cdot\|$. We will denote by $\|\cdot\|_*$ the dual norm of $\|\cdot\|$; that is, $\|\boldsymbol{u}\|_* \coloneqq \sup_{\|\boldsymbol{x}\|\le 1}\langle \boldsymbol{x}, \boldsymbol{u}\rangle$. For example, the dual norm of $\|\cdot\|_1$ is $\|\cdot\|_\infty$, while the dual norm of $\|\cdot\|_2$ is itself. More broadly, $\|\cdot\|_p$ is dual to $\|\cdot\|_q$ when $1/p + 1/q = 1$. In this context, FTRL enjoys the following no-regret guarantee.

Theorem 3.5 (Kalai and Vempala, 2005)**.** *For any sequence of utilities* $(\boldsymbol{u}^{(t)})_{t=1}^T$, FTRL *incurs regret bounded as*

$$\mathrm{Reg}^{(T)} \le \frac{R}{\eta} + \eta \sum_{t=1}^{T} \|\boldsymbol{u}^{(t)}\|_*^2. \tag{6}$$

In particular, if $\|\boldsymbol{u}^{(t)}\|_* \le B$ *for all* $t \in [T]$ *and* $\eta \coloneqq \frac{1}{B}\sqrt{R/T}$*, we have* $\mathrm{Reg}^{(T)} \le 2B\sqrt{RT}$.

More specifically, let's consider MWU (Example 3.3). The range of the entropic regularizer is $\log m$. So, if the utilities are such that $\|\boldsymbol{u}^{(t)}\|_\infty \le 1$ for all t, MWU guarantees $\mathrm{Reg}^{(T)} \le 2\sqrt{T \ln m}$. This is known to be optimal when facing an adversarial sequence of utilities. While the tuning of the learning rate in Theorem 3.5 assumes that the time horizon T is fixed and known in advance, that's not necessary; one can employ the so-called doubling trick [Shalev-Shwartz, 2012].

Mirror descent Another class of algorithms closely related to FTRL is *(online) mirror descent (*MD*)*. For a regularizer $\mathcal{R}$, we let $\mathcal{B}_{\mathcal{R}}(\boldsymbol{x}, \boldsymbol{x}') \coloneqq \mathcal{R}(\boldsymbol{x}) - \mathcal{R}(\boldsymbol{x}') - \langle \nabla\mathcal{R}(\boldsymbol{x}'), \boldsymbol{x} - \boldsymbol{x}' \rangle$ be the *Bregman divergence* induced by $\mathcal{R}$. For example,

- the Euclidean regularizer $\mathcal{R} : \boldsymbol{x} \mapsto \frac{1}{2}\|\boldsymbol{x}\|_2^2$ generates the squared Euclidean distance $\mathcal{B}_{\mathcal{R}}(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2$.
- The (negative) entropy regularizer $\mathcal{R} : \boldsymbol{x} \mapsto \sum_{a \in \mathcal{A}} \boldsymbol{x}[a] \ln \boldsymbol{x}[a]$ generates the Kullback-Leibler divergence $\mathcal{B}_{\mathcal{R}}(\boldsymbol{x}, \boldsymbol{x}') = \sum_{a \in \mathcal{A}} \boldsymbol{x}[a] \ln\left(\frac{\boldsymbol{x}[a]}{\boldsymbol{x}'[a]}\right)$.

In this context, the update rule of MD is as follows.

$$\boldsymbol{x}^{(t)} \coloneqq \operatorname*{argmax}_{\boldsymbol{x} \in \Delta(\mathcal{A})} \left\{ \langle \boldsymbol{x}, \boldsymbol{u}^{(t-1)} \rangle - \frac{1}{\eta} \mathcal{B}_{\mathcal{R}}(\boldsymbol{x}, \boldsymbol{x}^{(t-1)}) \right\}. \tag{7}$$

MD balances between optimizing with respect to the previously observed utility $\boldsymbol{u}^{(t-1)}$ while being close—measured by the Bregman divergence $\mathcal{B}_{\mathcal{R}}$—to the previous strategy $\boldsymbol{x}^{(t-1)}$. The extreme case where $\eta = +\infty$ in (7) amounts to best responding to the previously observed utility.

Under some assumptions on the regularizer $\mathcal{R}$—namely, that is a *Legendre regularizer* [Cesa-Bianchi and Lugosi, 2006]—it is known that MD is equivalent to FTRL. For example, this is the case when $\mathcal{R}$ is the (negative) entropy regularizer—both FTRL and MD produce MWU.

Example 3.6 (Online gradient descent). Let $\mathcal{R} : \boldsymbol{x} \mapsto \frac{1}{2}\|\boldsymbol{x}\|_2^2$. The update rule of MD in (7) then becomes $\boldsymbol{x}^{(t)} = \Pi_{\Delta(\mathcal{A})}(\boldsymbol{x}^{(t-1)} + \eta \boldsymbol{u}^{(t-1)})$. This is known as *online gradient descent*.

Let $R \coloneqq \max_{\boldsymbol{x} \in \Delta(\mathcal{A})} \mathcal{B}_{\mathcal{R}}(\boldsymbol{x}, \boldsymbol{x}^{(1)})$. MD enjoys the following no-regret guarantee, which closely echoes Theorem 3.5 pertaining to FTRL.

Theorem 3.7. *For any sequence of utilities* $(\boldsymbol{u}^{(t)})_{t=1}^T$, MD *incurs regret bounded as*

$$\mathrm{Reg}^{(T)} \le \frac{R}{\eta} + \eta \sum_{t=1}^{T} \|\boldsymbol{u}^{(t)}\|_*^2.$$

Regret matching Returning to Example 3.3, the update rule of MWU can be equivalently expressed as $\boldsymbol{x}^{(t)}[a] \propto \exp\left(\eta \sum_{\tau=1}^{t-1} (\boldsymbol{u}^{(\tau)}[a] - \langle \boldsymbol{x}^{(t)}, \boldsymbol{u}^{(t)} \rangle)\right) = \exp(\eta \boldsymbol{r}^{(t-1)}[a])$, where $\boldsymbol{r}^{(t)} \coloneqq \sum_{\tau=1}^{t} (\boldsymbol{u}^{(\tau)} - \langle \boldsymbol{x}^{(\tau)}, \boldsymbol{u}^{(\tau)} \rangle \mathbf{1})$; $\mathbf{1}$ here is the all-ones vector. Similarly, the update rule of FTRL with Euclidean regularization (Example 3.4) can be equivalently expressed as $\boldsymbol{x}^{(t)} \coloneqq \Pi_{\Delta(\mathcal{A})}(\eta \boldsymbol{r}^{(t-1)})$. From this viewpoint, another natural algorithm suggests itself: $\boldsymbol{x}^{(t)} \propto \max(\boldsymbol{r}^{(t-1)}, \mathbf{0}) \coloneqq [\boldsymbol{r}^{(t-1)}]^+$; this is precisely *regret matching* [Hart and Mas-Colell, 2000], given in Algorithm 1. This online algorithm also has the no-regret property, as shown below.

Theorem 3.8. *For any sequence of utilities* $(\boldsymbol{u}^{(t)})_{t=1}^T$ *in* $[0,1]^{\mathcal{A}}$, RM *incurs regret bounded as* $\sqrt{mT}$.

While this regret bound is inferior compared to the one we saw for MWU, RM is *parameter-free* and *scale invariant*, and tends to perform remarkably well in practice. (More on that in the upcoming lectures.)

Proof of Theorem 3.8. For any t, we have

$$\|[\boldsymbol{r}^{(t)}]^+\|_2^2 \le \|[\boldsymbol{r}^{(t-1)}]^+ + \boldsymbol{g}^{(t)}\|_2^2 \le \|[\boldsymbol{r}^{(t-1)}]^+\|_2^2 + \|\boldsymbol{g}^{(t)}\|_2^2 + 2\langle [\boldsymbol{r}^{(t-1)}]^+, \boldsymbol{g}^{(t)}\rangle,$$

where $\boldsymbol{g}^{(t)} \coloneqq \boldsymbol{u}^{(t)} - \langle \boldsymbol{x}^{(t)}, \boldsymbol{u}^{(t)}\rangle \mathbf{1}$. Given that $\langle \boldsymbol{x}^{(t)}, \boldsymbol{g}^{(t)}\rangle = 0$, it follows that $\langle [\boldsymbol{r}^{(t-1)}]^+, \boldsymbol{g}^{(t)}\rangle = 0$, where we used the fact that $\boldsymbol{x}^{(t)} \propto [\boldsymbol{r}^{(t-1)}]^+$ (by definition of RM). As a result,

$$\|[\boldsymbol{r}^{(t)}]^+\|_2^2 \le \|[\boldsymbol{r}^{(t-1)}]^+\|_2^2 + \|\boldsymbol{g}^{(t)}\|_2^2.$$

The resulting summation telescopes, so we have $\|[\boldsymbol{r}^{(T)}]^+\|_2^2 \le \sum_{t=1}^T \|\boldsymbol{g}^{(t)}\|_2^2 \le mT$ since $\|\boldsymbol{g}^{(t)}\|_\infty \le 1$ for all $t \in [T]$. □

A minor tweak to RM, known as RM$^+$, is given in Algorithm 2. It was introduced by Tammelin [2014]. The only difference is that RM$^+$ truncates the negative regrets to 0 in every update (Line 10). An identical proof to Theorem 3.8 implies that RM$^+$ enjoys a similar regret guarantee as RM. Yet, RM$^+$ is typically far superior in practice, as we shall see more in the upcoming lectures. A neat connection, shown recently by Farina et al. [2021], is that if one operates over a certain lifted space, RM$^+$ is an instance of MD while RM an instance of FTRL; this is despite the fact that, unlike FTRL and MD, RM and RM$^+$ are both parameter-free. In particular, this connection revolves around the framework of *Blackwell approachability* [Blackwell, 1956]. While RM$^+$ was originally put forward as a simple heuristic to speed up RM, the result of Farina et al. [2021] reveals that the difference between RM and RM$^+$ is fundamental.

Algorithm 1: Regret matching (RM)

1 Initialize cumulative regrets $\boldsymbol{r}^{(0)} \coloneqq \mathbf{0}$;
2 **for** $t = 1, \dots, T$ **do**
3 Define $\boldsymbol{\theta}^{(t)} \coloneqq \max(\boldsymbol{r}^{(t-1)}, \mathbf{0})$;
4 **if** $\boldsymbol{\theta}^{(t)} = \mathbf{0}$ **then**
5 Let $\boldsymbol{x}^{(t)} \in \Delta(\mathcal{A})$ be arbitrary
6 **else**
7 Compute $\boldsymbol{x}^{(t)} \coloneqq \boldsymbol{\theta}^{(t)} / \|\boldsymbol{\theta}^{(t)}\|_1$;
8 Output strategy $\boldsymbol{x}^{(t)} \in \Delta(\mathcal{A})$;
9 Observe utility $\boldsymbol{u}^{(t)} \in [0,1]^{\mathcal{A}}$;
10 $\boldsymbol{r}^{(t)} \coloneqq \boldsymbol{r}^{(t-1)} + \boldsymbol{u}^{(t)} - \langle \boldsymbol{x}^{(t)}, \boldsymbol{u}^{(t)}\rangle \mathbf{1}$;

Algorithm 2: Regret matching$^+$ (RM$^+$)

1 Initialize cumulative regrets $\boldsymbol{r}^{(0)} \coloneqq \mathbf{0}$;
2 **for** $t = 1, \dots, T$ **do**
3 Define $\boldsymbol{\theta}^{(t)} \coloneqq \boldsymbol{r}^{(t-1)}$;
4 **if** $\boldsymbol{\theta}^{(t)} = \mathbf{0}$ **then**
5 Let $\boldsymbol{x}^{(t)} \in \Delta(\mathcal{A})$ be arbitrary
6 **else**
7 Compute $\boldsymbol{x}^{(t)} \coloneqq \boldsymbol{\theta}^{(t)} / \|\boldsymbol{\theta}^{(t)}\|_1$;
8 Output strategy $\boldsymbol{x}^{(t)} \in \Delta(\mathcal{A})$;
9 Observe utility $\boldsymbol{u}^{(t)} \in [0,1]^{\mathcal{A}}$;
10 $\boldsymbol{r}^{(t)} \coloneqq [\boldsymbol{r}^{(t-1)} + \boldsymbol{u}^{(t)} - \langle \boldsymbol{x}^{(t)}, \boldsymbol{u}^{(t)}\rangle \mathbf{1}]^+$;

3.2 Self-play in zero-sum games

There is a celebrated connection between regret minimization and equilibrium computation in games. In particular, let's first consider zero-sum games. The basic idea is to have both players employ a regret minimization algorithm. The row player is to observe the sequence of utilities $(-\mathbf{A}\boldsymbol{y}^{(t)})_{t=1}^T$ while the column player the sequence of utilities $(\mathbf{A}^\top \boldsymbol{x}^{(t)})_{t=1}^T$. So long as both players have no-regret, we get convergence to an equilibrium [Freund and Schapire, 1999]; one typically uses the same regret minimization algorithm for both players, which is why this paradigm is often referred to as "self-play."

Theorem 3.9. *Suppose that the row player incurs regret* $\mathrm{Reg}_1^{(T)}$ *under the sequence of utilities* $(-\mathbf{A}\boldsymbol{y}^{(t)})_{t=1}^T$ *while the column player incurs regret* $\mathrm{Reg}_2^{(T)}$ *under the sequence of utilities* $(\mathbf{A}^\top \boldsymbol{x}^{(t)})_{t=1}^T$. *Then* $(\bar{\boldsymbol{x}}^{(T)}, \bar{\boldsymbol{y}}^{(T)})$ *is an* ϵ*-equilibrium with* $\epsilon = \frac{1}{T}(\mathrm{Reg}_1^{(T)} + \mathrm{Reg}_2^{(T)})$.

Above, we say that a strategy profile $(\boldsymbol{x}, \boldsymbol{y})$ is an *ϵ-equilibrium* if

$$\max_{\boldsymbol{y}' \in \Delta(\mathcal{A}_2)} \langle \boldsymbol{y}', \mathbf{A}^\top \boldsymbol{x} \rangle - \min_{\boldsymbol{x}' \in \Delta(\mathcal{A}_1)} \langle \boldsymbol{x}', \mathbf{A}\boldsymbol{y} \rangle \le \epsilon; \tag{8}$$

The left-hand side of (8) is referred to as the *duality gap*.

Proof of Theorem 3.9. The regret of the row player can be expressed as

$$\mathrm{Reg}_1^{(T)} = \max_{\boldsymbol{x}' \in \Delta(\mathcal{A}_1)} \sum_{t=1}^T \langle \boldsymbol{x}' - \boldsymbol{x}^{(t)}, -\mathbf{A}\boldsymbol{y}^{(t)} \rangle = \sum_{t=1}^T \langle \boldsymbol{x}^{(t)}, \mathbf{A}\boldsymbol{y}^{(t)} \rangle - T \min_{\boldsymbol{x}' \in \Delta(\mathcal{A}_1)} \langle \boldsymbol{x}', \mathbf{A}\bar{\boldsymbol{y}}^{(T)} \rangle. \tag{9}$$

Similarly,

$$\mathrm{Reg}_2^{(T)} = \max_{\boldsymbol{y}' \in \Delta(\mathcal{A}_2)} \sum_{t=1}^T \langle \boldsymbol{y}' - \boldsymbol{y}^{(t)}, \mathbf{A}^\top \boldsymbol{x}^{(t)} \rangle = T \max_{\boldsymbol{y}' \in \Delta(\mathcal{A}_2)} \langle \boldsymbol{y}', \mathbf{A}^\top \bar{\boldsymbol{x}}^{(T)} \rangle - \sum_{t=1}^T \langle \boldsymbol{x}^{(t)}, \mathbf{A}\boldsymbol{y}^{(t)} \rangle. \tag{10}$$

Summing (9) and (10), the claim follows. □

3.3 Self-play in general-sum games

We now extend the previous connection to multi-player general-sum games. While regret minimizing algorithms do not necessarily converge to Nash equilibria, they do converge to what's known as a *coarse correlated equilibrium*.

Definition 3.10 (Coarse correlated equilibrium)**.** A correlated distribution $\boldsymbol{\mu} \in \Delta(\mathcal{A}_1 \times \cdots \times \mathcal{A}_n)$ is an *ϵ-coarse correlated equilibrium* if for any player $i \in [n]$ and deviation $a_i' \in \mathcal{A}_i$,

$$\mathbb{E}_{(a_1,\ldots,a_n)\sim\mu} u_i(a_1, \ldots, a_n) \ge \mathbb{E}_{(a_1,\ldots,a_n)\sim\mu} u_i(a_i', a_{-i}) - \epsilon.$$

A Nash equilibrium is always a coarse correlated equilibrium, but not *vice versa*; a Nash equilibrium can be thought of as an *uncorrelated* (coarse) correlated equilibrium—that is, $\boldsymbol{\mu}$ is a product distribution. We are ready to state the general connection between regret minimization and coarse correlated equilibria.

Theorem 3.11. *Suppose that each player* $i \in [n]$ *incurs regret* $\mathrm{Reg}_i^{(T)}$ *under the sequence of utilities* $(\boldsymbol{u}_i(\boldsymbol{x}_{-i}^{(t)}))_{t=1}^T$. *Then the distribution* $\boldsymbol{\mu} \coloneqq \frac{1}{T}\sum_{t=1}^T \boldsymbol{x}_1^{(t)} \otimes \cdots \otimes \boldsymbol{x}_n^{(t)}$ *is an* ϵ*-coarse correlated equilibrium with* $\epsilon = \frac{1}{T}\max_{1\le i\le n} \mathrm{Reg}_i^{(T)}$.

A few comments are in order. First, $\boldsymbol{x}_1^{(t)} \otimes \cdots \otimes \boldsymbol{x}_n^{(t)}$ is the product distribution induced by $(\boldsymbol{x}_1^{(t)}, \ldots, \boldsymbol{x}_n^{(t)})$; $\otimes$ denotes the tensor product. As such, the correlated distribution $\boldsymbol{\mu}$ produced by Theorem 3.11 is a mixture of T product distributions. We have also denoted by $\mathbb{R}^{\mathcal{A}_i} \ni \boldsymbol{u}_i(\boldsymbol{x}_{-i}^{(t)}) \coloneqq (u_i(a_i, \boldsymbol{x}_{-i}^{(t)}))_{a_i \in \mathcal{A}_i}$, with an overload in the notation.

References

Bernhard von Stengel. Zero-sum games and linear programming duality. *Mathematics of Operations Research*, 49(2):1091–1108, 2024.

John von Neumann. Zur Theorie der Gesellschaftsspiele. *Mathematische Annalen*, 100:295–320, 1928.

Maurice Sion. On general minimax theorems. *Pacific Journal of Mathematics*, 8(1):171 – 176, 1958.

Eric van Damme. *Stability and perfection of Nash equilibria*, volume 339. Springer, 1991.

George W Brown and John von Neumann. Solutions of games by differential equations. 1950.

George W Brown. Iterative solution of games by fictitious play. *Act. Anal. Prod Allocation*, 13(1): 374, 1951.

Julia Robinson. An iterative method of solving a game. *Annals of Mathematics*, 54:296–301, 1951.

Constantinos Daskalakis and Qinxuan Pan. A counter-example to karlin's strong conjecture for fictitious play. In *Foundations of Computer Science (FOCS)*, 2014.

Yuanhao Wang. Tie-breaking agnostic lower bound for fictitious play. *arXiv:2507.09902*, 2025.

Adam Tauman Kalai and Santosh S. Vempala. Efficient algorithms for online decision problems. *Journal of Computer System Sciences*, 71(3):291–307, 2005.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Found. Trends Mach. Learn.*, 4(2):107–194, 2012.

Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

Sergiu Hart and Andreu Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68:1127–1150, 2000.

Oskari Tammelin. Solving large imperfect information games using CFR+. *arXov: 1407.5042*, 2014.

Gabriele Farina, Christian Kroer, and Tuomas Sandholm. Faster game solving via predictive blackwell approachability: Connecting regret matching and mirror descent. In *Conference on Artificial Intelligence (AAAI)*, 2021.

David Blackwell. An analog of the minmax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6:1–8, 1956.

Yoav Freund and Robert Schapire. Adaptive game playing using multiplicative weights. 29:79–103, 1999.