Notes for CPSC 6120: Topics in Algorithmic Game Theory

Spring 2026, Yale University

January 2026

Contents

1 Lecture 1 – Jan 13

Many existing successes in machine learning are examples of **single agent learning**. In single agent problems, there is a single objective function to be optimized, though it may be complicated and is typically

nonconvex. We have theoretical guarantees that gradient descent can efficiently local minima to these problems, even if they are nonconvex. Empirically, we have found that these solutions are usually good.

Games show up implicitly in ML: examples include Generative Adversarial Networks (GANs), adversarial training, and alignment. What about solving **multi agent games**, or **multi agent learning**? Here, there is no longer a single objective. There are different **agents**, each equipped with its own objective functions. These may be aligned or competing. **What counts as a solution to a system with multiple agents?**

Game theory tries to answer these types of questions, aiming for "equilibrium" (loosely speaking). When is equilibrium computation tractable? And can we simply transfer single agent methods to multi agent problems?

1.1 Implicit GAN Game: GD vs GD

An example of why simply applying agent methods doesn't work: gradient descent vs. gradient descent. Here, we train each agent independently via gradient descent and let them play against each other.

Consider GD vs. GD for GANs. The goal is to train a deep generative model such that sampling $Z \sim N(0, I) \to G_x(\cdot)$ and applying the generator NN G_x results in an image $\sim P_{\text{interesting}}$. The Goodfellow setup for this problem is a two-player game between a player tuning the parameters x of a DNN (**generator**) and a player tuning the parameters y of a DNN (**discriminator**). The generator DNN generates images with the goal of fooling the discriminator. The discriminator DNN receives either hallucinated images from the generator or real images from a dataset, with the goal of distinguishing real images from fake ones.

The Wasserstein GAN loss function is:

$$l(x, y) = E_{Z \sim P_{real}}[D_y(Z)] - E_{Z \sim N(0,I)}[D_y(G_x(Z))].$$

The discriminator leanrs to produce a score for each image $D_y(Z)$. The loss is then the difference between (expected discriminator score on a real image) and (expected discriminator score on fake image). The generator wants to train such that this loss is low (i.e., scores for real and fake images are similar), whereas the discriminator wants to train such that the same loss is high (real and fake images have very different scores).

We wish to find an "equilibrium" between these two goals. Theoretically speaking, such an equilibrium would meant the generator finds a distribution which perfectly matches the true one, and the discriminator only has random accuracy. Standard game theory typically assumes utilities which are quasi-concave, which is crucial for the existence of an equilibrium. In practice, loss functions are typically nonconvex.

If we ignore this, we can attempt to solve the GAN game with simultaneous gradient ascent dynamics:

- $u_y(x, y) = -u_x(x, y) = l(x, y)$
- $x_{t+1} = x_t + \eta \cdot \nabla_x u_x(x_t, y_t) = x_t + \eta \nabla_x l(x_t, y_t)$
- $y_{t+1} = y_t + \eta \cdot \nabla_y u_y(x_t, y_t) = y_t - \eta \nabla_y l(x_t, y_t)$

Two empirical experiments with MNIST and a Gaussian mixture yield poor results. We will later see the issue lies with the algorithm (GD vs GD), as opposed to the game not having an equilbrium (after all, GANs work well for image generation!). In fact, simultaneous gradient ascent provably diverges and exhibits chaotic behavior.

1.2 More Explicit Games: Algorithmic Agents in Economy

Examples of algorithmic agents in economy include **autobidders** and **algorithmic pricing**.

Notably, algorithmic pricing has faced controversy as it has been accused of collusion or antitrust law. These algorithms do not need to explicitly collude to exhibit undesirable behavior for consumers. Consider the Calvano, et al. paper "Artificial Intelligence, Algorithmic Pricing, and Collusion."

In traditional equilibrium theory, the **Bertrand duoply**, where two symmetric firms (i.e., producing the same product with the same production cost) compete for the same customer, is solved by both companies setting to the production cost. However, this does not hold when algorithms set the prices. The Calvano paper shows that in uncoupled learning, where the firms cannot communicate other than setting prices, observing profit, and adjusting their own prices using the q-learning algorithm, both agents will converge to a value above the Nash equilibrium (though lower than the monopoly price). This behavior looks like collusion. When the price for one agent is manually lowered, both agents end up walking back to the stable (greater than Nash) price point: it is robust!

Examples like these motivate us to understand multi-agent games more, as running single-agent algorithms against each other clearly yield unexpected behavior. Standard game theory is often inadequate for answering these questions.

1.3 Online Convex Optimization (OCO)

Let the **action space** Ω be a compact, convex subset of $\mathbb{R}^d$ with the Euclidean metric. A **game** is played for T rounds. For each round $t = 1, \ldots, T$:

- Learner takes an action $x_t \in \Omega$
- Environment picks a convex loss function $f^t : \Omega \to \mathbb{R}$
- Note that selection of this loss function can be adaptive, or it can be oblivious to the learner action x_t
- The learner observes $f^t(\cdot)$ and suffers loss $f^t(x^t)$

Define **external regret** or simply **regret** as

$$\text{ExtReg}(T) = sup_{f^1,\ldots,f^T} \left(\sum_{t=1}^{T} f^t(x^t) - \min_{x \in \mathbb{R}} \sum_{t=1}^{T} f^t(x) \right).$$

The first term is the cumulative loss incurred across T rounds given a sequence of actions $x^1, \ldots, x^T$. The second term is the minimum possible loss given a single fixed action x picked with hindsight for the same sequence of loss functions. External regret is their largest possible difference.

What does it meant to make ExtReg small? Intuitively, this formulation encourages an online learning algorithm to perform almost as well as the **single best static action** picked with hindsight (the knowledge of what action would have minimized loss incurred).

A decision making algorithm is said to be **regret minimizing** if ExtReg is sublinear in T, that is, $o(T)$. To motivate minimizing regret, note that **a regret minimizing algorithm can be shown to solve a convex optimization problem:**

Theorem 1. *Let $f^1 = f^2 = \cdots = f^T = f$. Then $\frac{1}{T}\sum_{t=1}^{T} x^t$ is an approximate minimum for f.*

Proof. Since the case where all loss functions are the same f is a special case of all possible f^t sequences, we must have that

$$\sum_{t=1}^{T} f(x^t) - \min_x \sum_{t=1}^{T} f(x) \leq \sup\left(\sum f^t(x^t) - \min_x \sum_{t=1}^{T} f(x)\right) = \text{ExtReg}(T).$$

Note that $\min_x \sum_t f(x) = T \min_x f(x) = T f(x_{\text{true min}})$.

Then

$$\frac{1}{T}\sum_{t=1}^{T} f(x^t) = f(x_{\text{true min}}) + \frac{1}{T}\text{ExtReg}(T)$$

By Jensen's inequality,

$$f\left(\frac{1}{T}\sum_{t=1}^{T} x^t\right) \leq \frac{1}{T}\sum_{t=1}^{T} f(x^t).$$

And since $x_{\text{true min}}$ is where f achieves its true minimum,

$$f(x_{\text{true min}}) \leq f\left(\frac{1}{T}\sum_{t=1}^{T} x^t\right) \leq f(x_{\text{true min}}) + \frac{1}{T}\text{ExtReg}(T).$$

Since ExtReg is sublinear in T, the second term approaches 0 as $T \to \infty$. □

1.4 Example Problems

Previously, we said that the learner would observe $f^t(\cdot)$ each round, which is somewhat hand-wavy. Generally, there are two more realistic types of observations: either the gradient $\nabla f^t(x^t)$ or the value $f^t(x^t)$ is observed.

Some more motivating examples of minimizing external regret are as follows:

1.4.1 Expert Problem

Consider having n experts, each of which recommends an action to take at each round. We can imagine this as n stocks, which we can choose to buy at each round. However, we do not hold these stocks across rounds; choosing an expert/stock immediately yields some loss for the round, which we directly observe (the second observation type). The next round, we again choose a new stock to buy.

The action space $\Omega = \Delta^n$ is the **simplex** in n. This means we are placing some probability distribution over the experts to randomly decide whose advice to take (which of n stocks to buy) each round.

Side note: setting the action space as the simplex over n is basically equivalent to defining a probability distribution over all the experts. This is because the $n-1$ simplex in $\mathbb{R}^n$ is the space of possible probability distributions on a finite set with n possible outcomes.

For each round t, given that x is some distribution over n experts, the expected loss is

$$f^t(x) = ^t, x\rangle,$$

i.e., the x-weighted average of loss function evaluated for each expert choice, normalized since x a probability distribution.

Then the external regret for a sequence of loss functions $\{l^t\}$ is:

$$\text{ExtReg}(T) = \sum_{t=1}^{T} \langle l^t, x\rangle - \min_{x\in\Omega} \sum_{t=1}^{T} \langle l^t, x\rangle.$$

If the external regret is sublinear, the average loss over time performs at least as well as picking the best single fixed expert/stock, even though the decision making algorithm doesn't have the kind of hindsight to pick a single fixed expert!

1.4.2 Online Spam Filtering

A simple example of online spam filtering: suppose we have an email which is mapped to a high dimensional embedding vector $a^t \in \mathbb{R}^d$. Say we also have a filter vector, $x^t \in \mathbb{R}^d$. Let a positive $\text{sign}(^t, x^t\rangle)$ denote spam, whereas negative denotes a legitimate email. The loss function is

$$f^t(x^t) = \frac{1}{4}\left(^t, x^t\rangle - b^t\right)^2,$$

where b^t is the true label of email a^t.

1.4.3 Portfolio Selection

This is similar to the expert problem, except this time assets persist over rounds/days. Let there be n assets to choose from. At each round, an adversary picks $r^t \in \mathbb{R}^n$, a vector where the i-th entry, r_i^t, denote the price ratio for asset $i \in [n]$ at between round t and $t+i$.

For example, suppose we have 1 dollar on round t, which we split amongst the n assets by a distribution x^t. Then on round $t+1$, we would start with

$$\langle x^t, r^t\rangle = x_1^t r_1^t + x_2^t r_2^t + \cdots + x_n^t r_n^t$$

dollars, and the total return ratio is

$$\prod_{t=1}^{T} \langle x^t, r^t \rangle.$$

This is nonconvex, but we can take the log returns for the convex log loss:

$$\sum_{t=1}^{T} \log^t, r^t \rangle.$$

Side note: I think this would only necessarily be convex if we force the returns to always be positive?

In this case, the baseline $\min_x \sum_{t=1}^{T} \log\langle x, r^t \rangle$ refers to the best fixed portfolio, which means that at each round we must rebalance back to the same distribution x. It does not mean we buy according to x on round 1 and hold.

2 Lecture 2 – Jan 20

More motivation on OCO: when we are playing a game, there is often limited information. The setup of OCO doesn't ask that we know a lot about the environment; pessimistically, we assume the worst case: that it is **adversarial**. What sorts of actions should we take? OCO provides us with some nontrivial guarantees in these situations.

2.1 Relaxation to Linear Loss

Recall our formulation of external regret,

$$\text{ExtReg}(T) = sup_{f^1,\ldots,f^T} \left(\sum_{t=1}^{T} f^t(x^t) - \min_{x \in \mathbb{R}} \sum_{t=1}^{T} f^t(x) \right).$$

Since each of the f^t loss functions chosen by the adversary was assumed to be convex, we note that the worst case must be linear loss. To see this, consider the difference of concern in ExtReg,

$$f^t(x^t) - f^t(x).$$

Since f^t is convex, we have

$$f^t(x^t) - f^t(x) \leq \langle x^t - x, \nabla f^t(x^t) \rangle.$$

Note that f^t does not have to be differentiable. We can take the **subgradient** of f instead, which is essentially any vector replacing ∇f^t that satisfies the expression above.

Summing on both sides,

$$\sum_{t=1}^{T}\Big(f^t(x^t) - f^t(x)\Big) \leq \sum_{t=1}^{T}\langle x^t - x, \nabla f^t(x^t)\rangle$$

implies

$$\max_{x\in\Omega}\sum_{t=1}^{T}\big(f^t(x^t) - f^t(x)\big) \leq \max_{x\in\Omega}\sum_{t=1}^{T}\langle x^t - x, \nabla f^t(x^t)\rangle.$$

The left hand side is precisely the external regret for a sequence of loss functions $f^1, ..., f^T$. Consider the right hand side as the external regret for loss functions $g^t, ..., g^T$, where $g^t(x) = \langle \nabla f^t(x^t), x\rangle$. This comes from bilinearity/symmetry, i.e.,

$$g^t(x^t) - g^t(x) = \langle \nabla f^t(x^t), x^t\rangle - \langle \nabla f^t(x^t), x\rangle = \langle \nabla f^t(x^t), x^t - x\rangle = \langle x^t - x, \nabla f^t(x^t)\rangle.$$

Therefore, the external regret for any sequence of convex loss functions chosen by the adversary is upper bounded by the external regret for a linear loss. If we can compete effectively against a linear adversary, then the same will hold for convex loss, which simplifies our analysis of OCO.

2.2 Algorithms for OCO

What is an algorithm with small regret over the above $g^t(x)$? The adversary can pick a loss that depends on the action the learner takes (i.e., the adversary is **adaptive**). We will answer some basic questions with the d-expert problem.

2.2.1 External Regret vs. Dynamic Regret

Why does the baseline of external regret compare against the best fixed action, instead of the best at each turn? Intuitively, we imagine that the best at each turn is impossible to compete against. We will see with the expert problem that this is indeed true.

Define the **dynamic regret** for an algorithm A and a game with T rounds as

$$\mathrm{DReg}_A(T) = \sup_{\ell^1,\ldots,\ell^T} L_A^T - L_*^T,$$

where $L_A^T := \sum_{t=1}^T \langle x^t, \ell^t\rangle$ and $L_*^T := \sum_{t=1}^T \min_{i\in[d]} \ell_i^t$. This compares against the loss incurred by the best possible action per turn, whereas the external regret expression has the min outside the sum. **We cannot minimize DReg**, in the sense that it is **linear in T.**

Proof. Consider the expert problem with two possible actions, *High* and *Low*, and $0-1$ loss. A random algorithm picks actions each round. The adversary places a loss of 1 on whichever action has > 0.5 probability of occurring, and 0 on the other. Then the best case loss in hindsight is $L_*^T = 0$. However, the algorithm will achieve an expected loss of $\geq T/2$, because the loss is always placed on actions with $\geq 1/2$ probability of occurring. Therefore, DReg must grow linearly in T. □

Remark 1. *Dynamic regret minimization is possible for more stable environments, such as a game where the adversary is attempting to maximize their own utility.*

Claim 1. *No deterministic algorithm can achieve $o(T)$ external regret.*

Proof. Not really a proof, more like intuition. Again consider the *High* vs. *Low* 2-expert problem. A deterministic algorithm plays an action each round. The adversary places 1 loss on the action played, and 0 otherwise. The cumulative loss $L_A^T = T$, and the best *fixed* action in hindsight (recall we are thinking about ExtReg is at most $T/2$. Therefore ExtReg must be linear in T. □

This gives us two key ideas: we cannot minimize dynamic regret, and we cannot use a deterministic algorithm to minimize external regret. How can we design good algorithms?

2.3 Follow the Leader (FTL) vs. Be the Leader (BTL)

One of the most basic algorithms to consider is **Follow the Leader (FTL)**. In FTL, at each round, we look at the history of the game and play the action which would do best w.r.t. previous data:

$$\text{FTL Action at Round t} := x^t \in \text{argmin}_{x\in\Omega} \sum_{\tau=1}^{t-1} f^\tau(x),$$

where x is a fixed action.

Remark 2. *FTL will not work for our purposes in the expert problem, but it works in other scenarios, e.g. if f^τ is picked from a fixed distribution instead of adversarially. In the 2-player 0-sum game where both players use FTL, they will converge to Nash.*

FTL is weak in this scenario. What is strong? We compare FTL to **Be the Leader (BTL)**:

$$\text{BTL Action at Round t} := \hat{x}_t \in \text{argmin}_{x\in\Omega} \sum_{\tau=1}^{t} f^\tau(x).$$

BTL is like FTL, except it also takes the current round's loss into consideration. This is impossible when we are playing. However, it is well defined, and in some sense, it is ideal. We would like to analyze and find ways to approximate BTL.

Lemma 1. $ExtReg_{BTL}(T) \leq 0$.

Proof. Induction: we want to show

$$\sum_{t=1}^{k} f^t(\hat{x}^t) \leq \sum_{t=1}^{k} f^t(x).$$

Let $k = 1$. The base case trivially holds. Assume true for k and consider

$$\sum_{t=1}^{k+1} f^t(\hat{x}^t) \leq_? \sum_{t=1}^{k+1} f^t(x).$$

Let us analyze the RHS, $\sum_{t=1}^{k+1} f^t(x)$. By definition of BTL, it must be that

$$\sum_{t=1}^{k+1} f^t(\hat{x}^{k+1}) \leq \sum_{t=1}^{k+1} f^t(x).$$

Then we can show instead that

$$\sum_{t=1}^{k+1} f^t(\hat{x}^t) \leq_? \sum_{t=1}^{k+1} f^t(\hat{x}^{k+1}).$$

The final terms of both sums where $t = k+1$ are clearly equal. From the inductive hypothesis,

$$\sum_{t=1}^{k} f^t(\hat{x}^t) \leq_? \sum_{t=1}^{k} f^t(\text{any action}).$$

Since $\hat{x}^{k+1}$ is one of these possible actions, we certainly have

$$\sum_{t=1}^{k} f^t(\hat{x}^t) \leq_? \sum_{t=1}^{k} f^t(\hat{x}^{k+1}).$$

Therefore,

$$\sum_{t=1}^{k+1} f^t(\hat{x}^t) \leq \sum_{t=1}^{k+1} f^t(\hat{x}^{k+1}) \leq \sum_{t=1}^{k} f^t(x).$$

□

Can we modify FTL to be more like BTL?

2.4 Perturbed Algorithms: FTPL vs. BTPL

Normalize the loss functions such that $||\ell||_\infty \leq 1$. Then

$$\begin{aligned}\text{Reg}_{\text{FTL}}(T) - \text{Reg}_{\text{BTL}}(T) &\leq \sum_{t=1}^{T} \langle x^t - \hat{x}^t, \ell^t \rangle \\ &= \sum_{t=1}^{T} \langle x^t - x^{t+1}, \ell^t \rangle \\ &\leq \sum_{t=1}^{T} ||x^t - x^{t+1}||_1,\end{aligned}$$

which is $\Omega(T)$ in the worst case. This is because FTL will play the best move w.r.t. the game history. Then the adversary can alternate losses at each round such that the minimizer jumps between extreme points, and FTL will be extremely unstable. Then, in the 2 dimensional *High* vs. *Low* case,

$$||x^t - x^{t+1}||_1 = 2$$

at each round, so $\sum_{t=1}^{T} 2$ is at best linear in T.

This motivates us to define a more stable version of FTL. Consider modifying FTL by regularizing the loss with a noise term:

$$x^t \in \text{argmin}_{x \in \Omega} \sum_{\tau=1}^{t-1} f^\tau(x) + f^0(x),$$

where f^0 has an appropriately small range. The same modification to BTL has $\text{Reg}_{\text{BTPL}} \left(\max_{x,x'} |f^0(x) - f^0(x')| \right)$.

More formally, define the **Follow the Perturbed Leader (FTPL)** and **Be the Perturbed Leader (BTPL)** algorithms:

Definition 1 (Follow the Perturbed Leader)**.** *Let $N \sim U(0, 1/\epsilon)^d$ and $f^0(x) = \langle N, x \rangle$. At round t, the action taken by FTPL is*

$$x^t = \textit{argmin}_{x \in \Omega} \langle x, \sum_{\tau=1}^{t-1} \ell^\tau + N \rangle.$$

Definition 2 (Be the Perturbed Leader)**.** *At round t, the action taken by BTPL is*

$$\tilde{x}(t) = \textit{argmin}_{x \in \Omega} \langle x, \sum_{\tau=1}^{t} \ell^\tau + N \rangle.$$

Note that sampling the noise term N uniformly is not the most optimal method, but we will analyze it version because it is illustrative.

Claim 2. *Under certain conditions, $Reg_{FTPL}(T) \approx Reg_{BTPL}(T)$.*

Proof. Denote $\sum_{\tau=1}^{t-1} f^\tau(x) =: F^{t-1}(x)$. and define

$$x^t := \arg\min_{x\in\Omega} F^{t-1}(x) + f^0(x), \qquad \hat{x}^t := \arg\min_{x\in\Omega} F^t(x) + f^0(x).$$

Then

$$\mathrm{Reg}_{\mathrm{FTPL}}(T) - \mathrm{Reg}_{\mathrm{BTPL}}(T) \;\le\; \sum_{t=1}^{T} \big(f^t(x^t) - f^t(\hat{x}^t)\big).$$

For linear losses $f^t(x) = \langle \ell^t, x\rangle$,

$$f^t(x^t) - f^t(\hat{x}^t) = \langle \ell^t, x^t - \hat{x}^t\rangle \;\le\; \|\ell^t\|_\infty \, \|x^t - \hat{x}^t\|_1.$$

Assume the mapping

$$g \;\mapsto\; \arg\min_{x\in\Omega} \langle g, x\rangle + f^0(x)$$

is L-Lipschitz from $\|\cdot\|_\infty$ to $\|\cdot\|_1$. Since

$$\hat{x}^t = M\left(\sum_{\tau<t} \ell^\tau + \ell^t + \ell^0\right), \qquad x^t = M\left(\sum_{\tau<t} \ell^\tau + \ell^0\right),$$

we obtain

$$\|x^t - \hat{x}^t\|_1 \;\le\; L\|\ell^t\|_\infty.$$

Therefore,

$$\mathrm{Reg}_{\mathrm{FTPL}}(T) - \mathrm{Reg}_{\mathrm{BTPL}}(T) \;\le\; L\sum_{t=1}^{T} \|\ell^t\|_\infty^2 \;\le\; LT.$$

The perturbation f^0 is chosen to ensure this Lipschitz stability (deterministically or in expectation). □

This is motivating! If we add noise to FTPL, it performs closely to BTPL. Furthermore, FTPL is minimizable:

$$\mathrm{Reg}_{\mathrm{FTPL}}(T) = \sum_{t\le T} \mathbb{E}\left(\langle x^t, \ell^t\rangle\right) - \min_x \sum_{t\le T} \langle x, \ell^t\rangle,$$

where the expectation is taken over FTPL because N is a random variable. We only have control over ϵ, so for sublinearity, we could select $\epsilon = \frac{1}{G\sqrt{T}}$ s.t. the above equals $2DG\sqrt{T}$.

We will now derive an explicit upper bound for $\mathrm{Reg}_{\mathrm{FTPL}}$ that relates $\mathrm{Reg}_{\mathrm{BTPL}}$ and the bounded difference between FTPL and BTPL.

Theorem 2.

$$Reg_{FTPL}(T) \le D/\epsilon + DG^2T\epsilon,$$

where D is the diameter of Ω, i.e.,

$$D = max_{x,x'\in\Omega}||x - x'||_1$$

and

$$G = max_t||\ell^t||_1.$$

To prove 2, we first need an intermediate result:

Lemma 2. *Let $\ell^{<k} := \sum_{t=1}^{k-1} \ell^t$. Then BTPL has constant regret:*

$$Reg_{BTPL}(T) \leq D/\epsilon.$$

Proof. BTPL is BTL with a fake "day 0" where the loss is $f^0(x) = \langle N, x\rangle$. Therefore, the loss incurred using BTPL is

$$\sum_{t=0}^{T} \langle \tilde{x}^t, \ell^t\rangle \leq \sum_{t=0}^{T} \langle x, \ell^t\rangle.$$

We are only interested in the loss from $t = 1, ..., T$. Subtracting the left day 0 term over, the true regret for BTPL is

$$\sum_{t=1}^{T} \langle \tilde{x}^t, \ell^t\rangle \leq \sum_{t=1}^{T} \langle x, \ell^t\rangle + \langle x - \tilde{x}^0, \ell^0\rangle.$$

$x - \tilde{x}^0$ is bounded by $D = \text{diam}(\Omega)$, and ℓ^0 is bounded since $||\ell||_\infty \leq 1/\epsilon$. Therefore, Hölder's inequality implies that the rightmost term must be $\leq D/\epsilon$. For fixed ϵ, BTPL has constant regret. □

We can now prove 2.

Proof. $\text{Reg}_{\text{FTPL}}(T) - \text{Reg}_{\text{BTPL}}(T) \leq \sum_{t=1}^{T} \big(f^t(x^t) - f^t(\hat{x}^t)\big)$. Since BTPL regret is bounded by D/ϵ, rearrange for

$$\text{Reg}_{\text{FTPL}}(T) \leq D/\epsilon + \sum_{t=1}^{T} \big(f^t(x^t) - f^t(\hat{x}^t)\big).$$

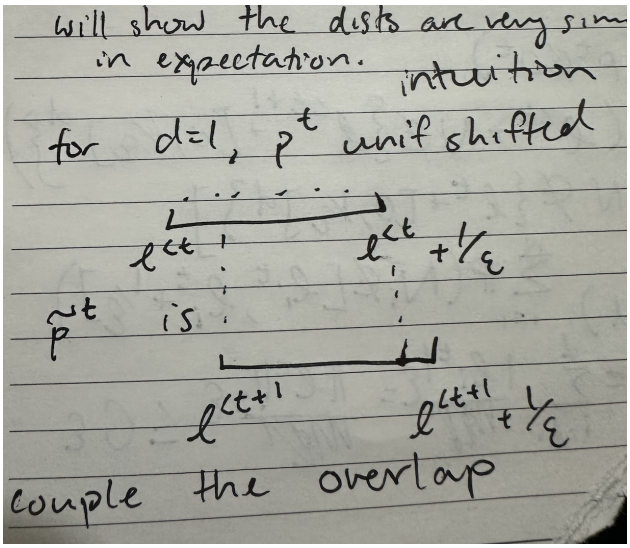
Therefore, we need to bound the loss differences between FTPL and BTPL actions $x^t, \hat{x}^t$. We will use a coupling argument between N and $\tilde{N}$, where $\tilde{N}$ is the random variable used for BTPL.

Let $N, \tilde{N} \sim U(0, 1/\epsilon)^d$. For round t, the FTPL action is

$$p_t = \ell^{<t} + N$$

and the BTPL action is

$$\tilde{p}^t = \ell^{<t+1} + \tilde{N}.$$



(Coupling argument intuition.)

Define the region $O^t = \{\ell^t + [0, 1/\epsilon]^d\} \cap \{\ell^{<t+1} + [0, 1/\epsilon]^d\}$. Then

$$\mathbb{P}(p^t \notin O^t) \leq G\epsilon, \text{ and } \mathbb{P}(\tilde{p}^t \notin O^t) \leq G\epsilon.$$

To see this, note that

$$\begin{aligned}
\mathbb{P}(p^t \notin^t) \; & \mathbb{P}(\ell^{<t} + N \notin \{\ell^{<t+1} + [0, 1/\epsilon]^d\}) \\
&= \mathbb{P}(N \notin \{\ell^t + [0, 1/\epsilon]^d\}) \\
&= \sum_{i=1}^{d} \mathbb{P}(N_i \notin [\ell_i^t, \ell_i^t + 1/\epsilon]) \\
&= \sum_{i=1}^{d} |\ell_i^t|\epsilon = ||\ell^t||_1 \epsilon \\
&\leq G\epsilon.
\end{aligned}$$

Furthermore, when both actions are in the overlap region O^t, we can couple them perfectly because the two distributions have the same densities (due to uniform dist).

proof continues... control the difference to get the $DG^2T\epsilon$ term in theorem. □

3 Lecture 3 – Jan 27

As we learn about more algorithms, it's important to remark on their selection. Different OCO algorithms are generally separated by their parameters and ambient space, but the time dependence (asymptotically

speaking) tends to be similar. One common piece of advice is to pick for the "geometry" of the problem. We will introduce another general class of algorithms (FTRL) and discuss what this means.

3.1 Follow the Regularized Leader (FTRL)

Recall our modification for FTL to FTPL: we want to pick the action

$$x^t = \text{argmin}_{x\in\Omega} \sum_{\tau=1}^{t-1} f^\tau(\cdot) + f^0(\cdot),$$

where f^0 is small and the minimizer of

$$\sum_{\tau=1}^{t-1} f^\tau(\cdot) + f^0(\cdot) = F^{t-1}(\cdot) + f^0(\cdot)$$

is Lipschitz.

Alternative methods of finding the regularizer f^0 give rise to **Follow the Regularized Leader (FTRL)**.

Theorem 3 (A bound for any regularizer)**.** *For any non negative regularizer* Φ,

$$\sum_{t=1}^{T} \left(\langle x^t, \ell^t\rangle - \langle x, \ell^t\rangle\right) \leq \sum_{t=1}^{T} \langle x^t - x^{t+1}, \ell^t\rangle + \Phi(x) - \min_{x'} \Phi(x')$$

$\forall x \in \Omega$.

Proof. For the RHS of 3, let the sum of the inner product be denoted as (B) and the remaining terms $\Phi(x) - \min_{x'} \Phi(x')$ be (A).

Notice immediately that the (A) term is entirely determined by the range of Φ. Note that

$$\text{Reg}_{\text{FTRL}}(T) = \text{Reg}_{\text{BTRL}} + L_{\text{FTRL}}(T) - L_{BTRL}(T).$$

We want to bound Reg_{BTRL} by (A) and bound the L terms by (B).

Let $\Phi(x)$ be $\frac{1}{2\eta}||x||^2$, where η is the step size. Then

$$(A) = \Phi(x) - \min_{x'}\Phi(x') \leq \frac{1}{2\eta}||x||^2 \leq \frac{1}{2\eta}\max_{x\in\Omega}||x||^2.$$

Next, we claim that the (regularized) action x^t is equal to the following:

$$x^t = \text{argmin}_{x\in\Omega}\langle x, \ell^{<t}\rangle + \frac{1}{2\eta}||x||^2 \underset{(claim)}{=} \text{argmin}_{x\in\Omega}||x + y\ell^{<t}||^2,$$

where we can interpret $||x + y\ell^{<t}||^2$ as the projection of $-y\ell^{<t}$ into the convex action set Ω. To see this intuitively, imagine why x satisfying the argmin over Ω should be the x which minimizes distance $||\cdot||$ to $-y\ell^{<t}$, i.e., the projection.

More formally, let $\pi_s(x) := \text{argmin}_{y \in S} ||y - x||^2$ be the Euclidean projection into S. Then the action of FTRL, x^t, equals $\pi_\Omega(-y\ell^{<t}) = \text{argmin}_{x \in \Omega} ||x - y\ell^{<t}||^2$.

We claim that $||x^t - x^{t+1}|| \leq \eta ||\ell||^t$ from convexity; that is, that the distance between the two actions is less than the step size η times the magnitude of the loss. To see this, note that

$$||\pi_s(x) - \pi_s(y) \leq ||x - y||$$

for all x, y and convex set S. Then

$$||x^t - x^{t+1}|| \leq || - y\ell^{<t+1} + y\ell^{<t}|| = y||\ell^t||.$$

So we can bound (B) by

$$\leq \sum_{t=1}^{T} ||x^t - x^{t+1}||\ ||\ell^t|| \leq y \sum_{t=1}^{T} ||\ell^t||^2.$$

We have upper bounded both (A) and (B) in the $\text{Reg}_{FTRL}(T)$ expression. Now, if we simply let $G = \max_t ||\ell^t||$ and choose η, we have that

$$\min_\eta \frac{1}{2\eta} ||x||^2 + y \sum_{t=1}^{T} ||\ell^t||^2 = O(\max_{x \in \Omega} ||x|| G\sqrt{T}).$$

□

.

3.2 Examples of FTRL: Online Regression/Spam Filtering

Recall the spam filtering problem introduced in Lecture 1. Here, the action space is

$$\Omega = \{x \in \mathbb{R}^d : ||x||_2 \leq 1\},$$

where x are the coefficients of a linear classifier.

The loss incurred at round t is $f^t(x) = \frac{1}{2}(\langle x^t, \ell^t \rangle - yt)^2$. Then

$$\nabla f^t(x) = \langle x^t, \ell^t \rangle \ell^t - y^t \ell^t,$$

which is Reg_{FTRL} regularized by $\mathcal{L}^2$, with regret bound $O\left(\sqrt{T} \max_\tau ||\ell||^2\right) = O(T \cdot G)$.

3.3 Bregman Divergence

The **Bregman divergence** is defined for any convex function. We can think of it as a generalization of distance, though it is not itself a distance: it is not symmetric.

Define the Bregman divergence as a function $D_\psi : \Omega \times \Omega \to \mathbb{R}$ that maps pairs of actions as follows:

$$D_\psi(x, y) = \Psi(x) - \Psi(y) - \langle \nabla \Psi(y), x - y \rangle.$$

If Ψ is linear, then $D_\Psi(x,y) = 0$. Note when we use the $\mathcal{L}^2$ regularizer, $\Psi = ||x-y||^2 \implies D_\Psi(x,y) = ||x-y||^2$. And when we use entropy as the regularizer, the KL-divergence is the Bregman divergence.

(primal, dual drawing)

Therefore, we can consider FTRL as

$$\begin{aligned}
&\text{argmin}_{x\in\Omega}\langle \ell^t, x\rangle + \Psi(x) \\
&= \text{argmin}_{x\in\Omega} - \langle \nabla\Psi(w^t), x\rangle + \Psi(x) \\
&= \text{argmin}_{x\in\Omega}\Psi(x) - \Psi(w^t) - \langle \nabla\Psi(w^t), x - w^t\rangle \\
&= D_\Psi(x, w^t),
\end{aligned}$$

the Bregman divergence wrt Ψ of the best fixed action x and the FTRL action w^t. (CHECK THIS)

3.4 Stability Lemma

Lemma 3. *Let $x^* = argmin_{x\in\Omega}\langle x, L\rangle + \frac{1}{y}\Psi(x)$ and $y^* = argmin_{y\in\Omega}\langle y, \bar{L}\rangle + \frac{1}{y}\Psi(y)$. Suppose Ψ is 1-strongly convex with respect to some norm $||\cdot||$, i.e.,*

$$\Psi(w) \geq \Psi(z) + \langle \nabla\Psi(z), w - z\rangle + \frac{1}{2}||w-z||^2.$$

Then

$$\begin{aligned}
||x^* - y^*||^2 &\leq y\langle y^* - x^*, L - \bar{L}\rangle \\
&\leq y||x^* - y^*||\ ||L - \bar{L}||_* \\
&\implies ||x^* - y^*|| \leq y||L - \bar{L}||_*,
\end{aligned}$$

where the $*$ norms are different from the one used in the definition of strong convexity. It is

$$||w||_* = max_{u\ s.t.\ ||u||\leq 1}\langle u, w\rangle$$

If Ψ is strongly convex wrt $\mathcal{L}^2$, then the $*$ norm is also $\mathcal{L}^2$. And if strong convexity is wrt $\mathcal{L}^1$, the $*$ norm is $\mathcal{L}^\infty$. In general, for $\mathcal{L}^p$ norms, the $*$ norm is q such that $\frac{1}{p} + \frac{1}{q} = 1$.

3.5 Examples of FTRL: Expert Problem and MWU

In the expert problem, $\Omega = \Delta^d$ and $\ell = [0,1]^d$. With the $\mathcal{L}^2$ regularizer, $\text{Reg}_{FTRL}(T) = O(d\sqrt{T})$. With different regularizers, we get different regret bounds. For example, **Hedge**, also called **Multiplicative Weight Update (MWU)**, uses negative entropy as the regularizer:

Definition 3 (MWU)**.** *The regularizer for the Multiplicative Weight Update version of FTRL is:*

$$\Psi(x) = \sum_{i=1}^{d} x_i \log(x_i).$$

In the expert problem, the action taken is

$$x^t = \text{argmin}_{x \in \Delta^d} \langle x, \ell^{<t} \rangle + \frac{1}{\eta} \sum_{i=1}^{d} x_i \log(x_i).$$

The Lagrangian $L(x, \lambda)$ is

$$L(x, \lambda) = \langle x, \ell \rangle + \frac{1}{\eta} \sum_i x_i \log x_i + \lambda \left(\sum_{i=1}^{N} x_i - 1 \right),$$

and

$$\frac{\partial L}{\partial x_i} = 0 \implies \ell_i + \frac{1}{\eta}(\ln x_i + 1) + \lambda = 0 \tag{1}$$

$$\frac{\partial L}{\partial \lambda} = 0 \implies \sum_{i=1}^{d} x_i = 1 \tag{2}$$

From (1),

$$x_i = e^{-\eta(\lambda + \ell_i) - 1} \implies x_i \propto e^{-} \eta \ell_i.$$

So x_i is in fact

$$\frac{e^{-\eta \ell_i}}{\sum_i e^{-\eta \ell_i}},$$

which is actually the softmax function.

3.5.1 MWU Regret Bound

Recall from before that

$$\text{Reg}_{\text{FTRL}}(T) = \text{Reg}_{\text{BTRL}} + L_{\text{FTRL}}(T) - L_{BTRL}(T).$$

What is the regret bound for MWU? We will bound the regret terms (A) and the loss terms (B) separately.

(A) is $\leq \frac{1}{\eta} \log d$, since the largest possible entropy is over the uniform distribution and is $\log d$. Now we will bound (B).

Claim 3. *$\Psi(x) = \sum_{i \in [d]} x_i \ln x_i$ is 1-strongly convex with respect to the $\mathcal{L}^1$ norm.*

Proof. From the Taylor expansion on $\Psi(w)$ definition of strong convexity, it suffices to show for all $x, y \in \Delta^d$,

$$(x - y)^T \nabla^2 \Psi(w)(x - y) \geq ||x - y||_1^2,$$

where $w = \lambda x + (1 - \lambda) y$.

The Hessian, $\nabla^2 \Psi(w)$, is a diagonal matrix because each term in Ψ is independent of the others.

Then

$$(x-y)^T \nabla^2 \Psi(w)(x-y) = \sum_{i \in [d]} \frac{(x_i - y_i)^2}{w_i} \geq \sum_{i \in [d]} |x_i - y_i|^2 = ||x-y||_1^2.$$

□

Then (B) is bounded

$$\leq \sum_{t=1}^{T} ||x^t - x^{t+1}||_1 ||\ell^t||_\infty \leq \eta T,$$

since $||\ell^t||_\infty$ is 1 for the expert problem and from the stability lemma,

$$||x^t - x^{t+1}|| \leq \eta ||\ell^t||_{\star=\infty}.$$

Therefore,

$$\text{Reg}_{MWU}(T) \leq \eta T + \frac{\log d}{\eta}.$$

4 Lecture 4 – Feb 3

In Lecture 3, we learned about FTRL. Today, we will compare the different algorithms we've covered thus far, then transition into talking about games.

4.1 Comparing Algorithms with the Expert Problem

Let us generalize the expert problem such that every expert now corresponds to a combinatorial action.

More formally, let $S = \{v_1, \ldots, v_M\}$ denote the set of experts. Each $v_k \in \{0,1\}^N$, and N may or may not be large. Restrict $\max_j ||v_j||_1 \leq m$, where $m \ll N$.

We can therefore interpret the action set as any distribution over the m experts/vectors, i.e.,

$$\Omega = \left\{ \sum_{j=1}^{M} x_j v_j : x \in \Delta^M \right\} \subseteq [0,1]^N.$$

check the subset above.

The loss incurred at time t due to action x is $f^t(x) = \langle x, \ell^t \rangle$ where $\ell^t \in [0,1]^N$.

In the standard expert problem, $m = 1$: choices are not combinatorial, and S is the set of standard basis vectors.

4.1.1 Online Shortest Path Problem

One special case of the combinatorial expert problem is the **online shortest path problem**.

In this setup, we have a graph $G = (V, E)$. We want to travel from node s to node t, and the time it takes to travel along a particular edge changes at each round.

To see how this maps experts to parameters in the graph, consider the following:

- We have N edges. Each (simple) path from s to t corresponds to a $\{0,1\}^N$ vector in S: if you take a certain edge, the entry at index i is 1, and 0 otherwise.
- M is the total number of s-t paths, which is potentially very large.
- m is the length of the longest path.

We know from previous lectures that the algorithms discussed thus far have asymptotic time dependence on the order of $\sqrt{T}$. But there are other parameters in this problem: $N, M,$ and m, where M is very large: assume only $\leq (N)^m$.

We will investigate the differences between $\mathcal{L}^2$ FTRL, MWU, and FTPL. The following analysis assumes we have already tuned the learning rate appropriately.

4.1.2 OSP with L2 FTRL

The regret bound is

$$\text{Reg}(T) = O(\sqrt{T} \cdot max_{x \in \Omega} ||x||_2 \cdot max_t ||\ell^t||_2).$$

We already know the $\mathcal{L}^1$ norm is bounded by m, so the same must be true of the $\mathcal{L}^2$ term on x.

The problematic term is $\max_t ||\ell^t||_2$. We haven't assumed anything on the structure of the loss (other than restricting each element to $[0, 1]$). The dimension could be very large, so we conservatively bound this by $\sqrt{N}$.

4.1.3 OSP with MWU

We can think of this as the standard expert problem, where the loss is potentially up to m. The regret bound is

$$\text{Reg}(T) = O(\sqrt{T}\sqrt{\log M} \cdot m) = O(\sqrt{T} \cdot m \cdot \sqrt{m \log N}),$$

where the second equality comes from the large bound on M.

Note this doesn't have polynomial dependence on N, which allows N to be quite large. However, there is still a problem with MWU here.

Recall that running the algorithm itself relies on N because we must maintain a distribution over the experts. For the online shortest path problem, the support of this distribution becomes massive, and explicitly keeping track of it becomes very expensive[1]. In fact, the **per iteration runtime** is $\Theta(M)$.

[1] There is a strategy to run MWU here, but we won't cover it in this lecture.

4.1.4 OSP with FTPL

The regret bound is

$$\text{Reg}_{FTPL}(T) = O(m \cdot \sqrt{NT}).$$

At first glance, the polynomial dependence on N isn't very encouraging. However, running the algorithm for the online shortest path problem is particularly easy: at each iteration, we literally find the shortest path (compute smallest loss with some noise).

FTPL is a good algorithm to run if you can efficiently minimize a linear objective. In these situations, we will simply say the per iteration runtime is "efficient," which, for example, MWU may not give us.

Recall also that our proof of FTPL did not choose noise optimally (we used the Uniform distribution), but with Laplace noise we can actually get a better bound on Reg, resulting in $O(m\sqrt{m \log N \cdot T})$.

4.2 Strong Convexity & Online Gradient Descent (OGD)

Before we transition to discussing games, we remark that our reduction from convex to linear loss may not always be the best method. If we know stronger properties about the loss sequence (e.g., it is strongly convex), we can achieve tighter bounds, which linear loss would sacrifice.

Definition 4 (Strong Convexity)**.** *f is α-strongly convex w.r.t. norm $||\cdot||_2$ if*

$$f(x) \geq f(x') + \langle x - x', \nabla f(x) \rangle + \frac{\alpha}{2}||x - x'||^2.$$

Equivalently, convexity of $f(x) - \frac{\alpha}{2}||x||^2$ implies $f(x)$ is α-strongly convex.

Theorem 4 (Regret Bound for Strongly Convex Loss Functions)**.** *If $f^1, \ldots, f^T$ are all α-strongly convex w.r.t. $\mathcal{L}^2$, then **online gradient descent (OGD)** with step size[2] $\eta^t = \frac{1}{\alpha \cdot t}$ achieves the following regret:*

$$\textit{Reg}(T) \leq \frac{G^2}{2\alpha} \cdot (1 + \log T),$$

where G is $max_{t,x}||\nabla f^t(x)||$.

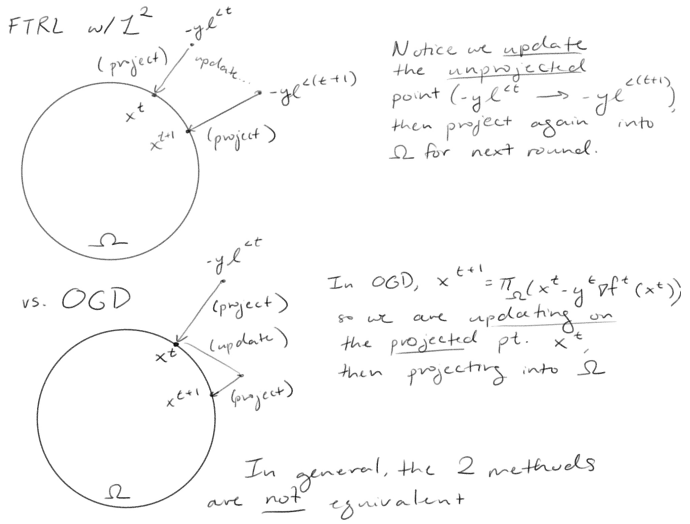
4 is an important theorem. All of the algorithms we saw previously achieved $\sqrt{T}$ regret bounds, but OGD with strong convexity isn't even polynomial[3] in T.

Definition 5 (Online Gradient Descent (OGD))**.** *For every round $t = 1, \ldots, T$ do the following:*

- *Choose x^t and observe $f^t(x^t)$*
- *Update and project into Ω: Choose $x^{t+1} = \pi_\Omega(x^t - y^t \nabla f^t(x^t))$.*

[2]Note that η^t is an **adaptive** step size: it denotes η at time t, not exponentiation. This should make sense: steps start large and become more conservative as t grows.

[3]For general convex functions, both $\mathcal{L}^2$ FTRL and OGD give $\sqrt{T}$ regret bounds.



Proof of 4. Let $x^* \in \text{argmin}_{x\in\Omega} \sum_{t=1}^T f^t(x)$ and $g^t := \nabla f^t(x^t)$.

From strong convexity, we have:

$$2(f^t(x^t) - f^t(x^*)) - 2\langle x^t - x^*, g^t\rangle \leq -\alpha||x^* - x^t||^2 \tag{1}$$

Intuitively, if we want the regret to be small, we would hope that the average of x^t gets close to x^*. Therefore, it makes sense to define a "potential" function of sorts on the distance between x^{t+1}, x^*:

$$\begin{aligned}
||x^{t+1} - x^*||^2 &= ||\pi_\Omega(x^t - y^t g^t) - x^*||^2 \\
&\leq ||x^t - y^t g^t - x^*||^2 \\
&= ||x^t - x^*||^2 - 2y^t\langle x^t - x^*, g^t\rangle + (y^t)^2||g^t||^2.
\end{aligned}$$

Rearranging,

$$2\langle x^t - x^*, g^t\rangle \leq \frac{||x^t - x^*||^2 - ||x^{t+1} - x^*||^2}{y^t} + y^t||g^t||^2. \tag{2}$$

Therefore, by (1) and (2),

$$2\sum_{t=1}^T \left(f^t(x^t) - f^t(x^*)\right) \leq \sum_{t=1}^T \left[\frac{||x^t - x^*||^2 - ||x^{t+1} - x^*||^2}{y^t} + y^t||g^t||^2 - \alpha||x^* - x^t||^2\right] \tag{3}$$

Pulling all the terms involving the distance $||x^* - x^t||$ out, we simplify the RHS of (5) to:

$$\sum_{t=1}^{T}\left(\frac{1}{\eta^t}-\frac{1}{\eta^{t-1}}-\alpha\right)||x^t-x^*||^2+G^2\sum_{t=1}^{T}\eta^t.$$

The $\frac{1}{\eta^t}-\frac{1}{\eta^{t-1}}$ term is from the telescoping sum on the RHS of (3), since step sizes are not uniform.

As a convention, we set the $\frac{1}{\eta^0}$ in the first term of the left sum as 0. By choosing $\eta^t=\frac{1}{\alpha t}$, the first term at $t=1$ is $\frac{1}{1/\alpha t}-0-\alpha=\alpha(1)-\alpha=0$. This reduces to

$$G^2\sum_{t=1}^{T}\frac{1}{\alpha\cdot t}\leq\frac{G^2}{\alpha}(1+\log T),$$

where the upper bound follows from harmonic series.[4] □

4.3 Games

In previous sections we focused on discussing algorithms. What happens if we use these algorithms as strategies in **games**?

Consider **fictitious play**, where each player uses FTL for every round. It has been shown that the time-average of this strategy achieves Nash equilibrium[5] The proven convergence rate to Nash is exponential, but empirically, faster rates have been observed.

We will show that any algorithm with $\sqrt{T}$ external regret will yield a $1/\sqrt{T}$ convergence rate to Nash equilibrium.

4.3.1 2-Player Zero-Sum Games

Definition 6 (2-Player Zero-Sum Games)**.** *In a **2P0SG**, player 1 takes actions $x\in X\subseteq\mathbb{R}^m$ with reward function $-f(x,y)$. Player 2 takes actions $y\in Y\subseteq\mathbb{R}^n$ with reward function $f(x,y)$.*

Let $f(\cdot,\cdot)$ be convex-concave:

- For fixed y, f is convex in x
- For fixed x, f is concave in y.

Therefore, in a 2P0SG, player 1 seeks to maximize concave $-f$ (since f for fixed y is convex), while player 2 seeks to maximize their reward of concave f (fixed x).

We can write the action spaces as

$$X=\Delta^m, Y=\Delta^n$$

[4] We remark that this regret expression isn't just a free $\log T$; there is dependence on the strength of the convexity (α). For certain problems, one might want to simply add some regularizer such that the function is strongly convex, but this affects alpha (that is, no free lunch).

[5] Consider a game of rock-paper-scissors where players use FTL. This obviously leads to a loop, which is Nash over time.

and denote $A(i,j)$ as the payoff for the action i,j: where player 1 plays the i-th action and player 2 plays the j-th action. Then we can write f as a bilinear function

$$f(x,y) = -x^T Ay.$$

Definition 7 (Nash Equilibrium for 2P0SG)**.** *The pair (x^*, y^*) is a **Nash equilibrium** iff the following is true:*

$$f(x^*, y) \leq f(x^*, y^*) \ \forall y$$
$$f(x, y^*) \geq f(x^*, y^*) \ \forall x$$

Furthermore, to understand "convergence" to Nash, we must define a way to measure how "close" a strategy is to a Nash equilibrium.

Definition 8 (Duality Gap)**.** *We define the **duality gap** dg as*

$$dg(x,y) := max_{y'} f(x,y') - min_{x'} f(x',y)$$

Implicitly, we should think of the duality gap expression as

$$dg(x,y) := max_{y'} f(x,y') - f(x,y) - min_{x'} f(x',y) + f(x,y),$$

where the first two terms measure how much more we can gain by adjusting y to y' when x is fixed, and the second two terms analogously measure gain from adjusting x in response to fixed y. It should be clear that dg is therefore a nonnegative function. A smaller duality gap implies we are "closer" to Nash equilibrium.

4.3.2 Convergence to Nash Corresponds to Algorithm Regret

We will now prove the theorem prefaced earlier in this section. Formally,

Theorem 5. *Let $(x^1, y^1), \ldots, x(^T, y^T)$ be the history of play. Suppose player 1's algorithm has a regret of $R^x(T)$, and player 2's algorithm has regret $R^y(T)$.*

Then the time average over x and y, $(\bar{x}, \bar{y})$, is upper bounded by

$$dg(\bar{x}, \bar{y}) \leq \frac{1}{T}(R^x(T) + R^y(T)).$$

This implies if both players use regret-minimizing algorithms, the duality gap goes to 0, i.e., the time average of the actions will reach Nash equilibrium.[6]

Proof. First define the time averages as

$$\bar{x} = \frac{1}{T}\sum_{t=1}^{T} x^t \text{ and } \bar{y} = \frac{1}{T}\sum_{t=1}^{T} y^t.$$

[6]From von Neumann minimax theorem, we also get strong duality for linear programming and more exciting things, yay!

For player 1, the loss for round t is $f(\cdot, y^t)$.[7] Similarly, for player 2, the loss for round t is $-f(x^t, \cdot)$.[8]

$$R^x(T) \geq \sum_t f(x^t, y^t) - min_x \sum_t f(x, y^t) \geq \sum_t f(x^t, y^t) - min_x T \cdot f(x, \bar{y})$$

and

$$R^y(T) \geq -\sum_t f(x^t, y^t) + max_y \sum_t f(x^t, y) \geq max_y T \cdot f(\bar{x}, y) - \sum_t f(x^t, y^t)).$$

Summing the two inequalities, we have the duality gap times T:

$$R^x(T) + R^y(T) \geq T \cdot dg(\bar{x}, \bar{y})$$

□

The above result shows why $\sqrt{T}$ regret algorithms yield a $1/\sqrt{T}$ convergence rate for dg.

Some remarks: the regret bound is adversarial, but we may not be playing against an adversarial opponent (e.g., they could be maximizing their own rewards). Therefore, this bound this is somewhat pessimistic. In fact, we can show that the two regrets will be constant (with no dependence on T) if you play against a copy of yourself (**self-play**).[9]

Secondly, we are looking at the average $\bar{x}, \bar{y}$ over time. But the convergence of the average-iterate does not guarantee anything for the final iterate. This motivates analyzing x^T, y^T instead, which would tell us whether play actually stabilizes.[10]

5 Lecture 5 – Feb 10

Suppose we have a 2 player game with a driver and a pedestrian. They each have two strategies: to cross and to wait. The Nash equilibrium is either (C, W) or (W, C); otherwise, they will either both cross and lose, or both wait and gain nothing.

Consider mixing the strategies: for example, crossing with probability 1/100 and waiting with probability 99/100. This is inefficient. In fact, the most efficient method for both players would be (C, W) half the time and (W, C) the other half. But this isn't a strategy profile that can be implemented independently: it requires an agreement or coordination between the players.

Now consider the case with n players. The action for the i-th player is $x_i \in X_i$, a convex and compact set in $\mathbb{R}^{n_i}$. Each player has a utility function $u_i(x)$ such that

- The function is concave in x_i for all $x_{-i} \in X_{-i}$, where $-i$ denotes all indices j excluding i.
- The function is linear in x_i when every other player's action is held fixed.

[7] As in, after seeing the action of the opponent at round t, compute the loss. Obviously, the player has to commit to their action that round before actually being able to see the opponent's action/loss.

[8] Signs are because in our 2P0SG setup, we denoted f as the reward, and we are now dealing with losses.

[9] This result requires a stronger algorithm (not a general one like in the proof).

[10] This also requires a specific algorithm and will result in constant regret.

5.1 Definitions of Equilibria

5.1.1 Nash and Epsilon Nash

Definition 9 (Nash equilibrium)**.** *$x^* \in X$ is a **Nash equilibrium** if and only if for all i and $x_i \in X_i$,*

$$U_i(x_i, x^*_{-i}) \leq U_i(x^*).$$

That is, no player benefits from changing its action independently from x^*. Often, Nash equilibrium must be approximated because the probabilities are not necessarily rational numbers. In this case, we have an **Epsilon-Nash equilibrium**, which simply says that there is no improvement over some small ϵ, i.e. a strategy profile which closely approximates Nash.

5.1.2 Coarse Correlated Equilibrium (CCE)

Definition 10. *A **coarse correlated equilibrium (CCE)** is a distribution σ^* over X[11] such that*

$$\mathbb{E}_{x\sim\sigma^*}(U_i(s_i, x_i)) \leq \mathbb{E}_{x\sim\sigma^*}(U_i(x))$$

for all $i \in [n]$ and $s_i \in X_i$[12]*.*

To reach an approximate CCE, we will see that running any no-regret algorithm for T rounds is sufficient. Denote the loss incurred on day t as $-U_i(\cdot, x^*_{-i})$. Then for some sequence of play $(x^1_1, ..., x^1_n), ..., (x^T_1, ..., x^T_n)$, the regret is

$$\sum_{t=1}^{T} -U_i(x^t_i, x^{t)}_{-i} - \left(\min_{s_i\in X_i} \sum_{t=1}^{T} -U_i(s_i, x^t_{-i})\right) \leq \text{Reg}_i(T). \tag{1}$$

Since

$$-\left(\min_{s_i\in X_i} \sum_{t=1}^{T} -U_i(s_i, x^t_{-i})\right)$$

is just maximizing

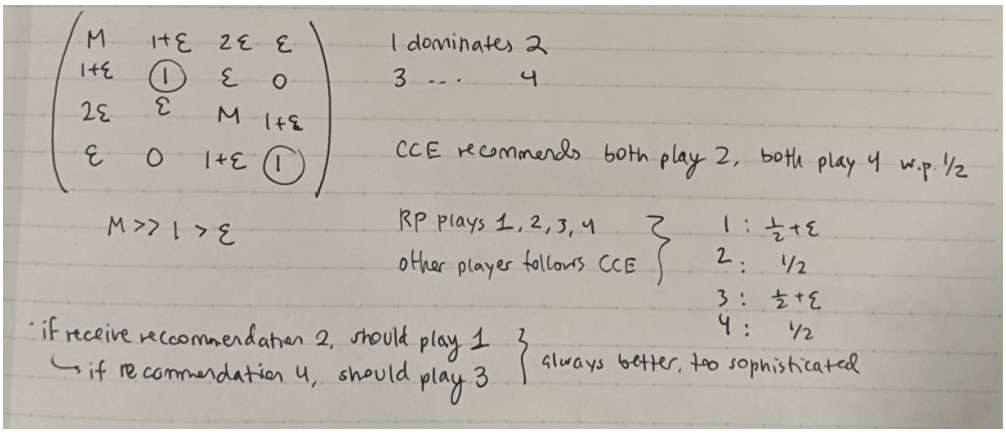
$$\max_{s_i\in X_i} \sum_{t=1}^{T} U_i(s_i, x^t_i),$$

dividing both sides of (1) by T, we have that

$$\max_i \left(\frac{\text{Reg}_i(T)}{T}\right) - \text{CCE} \sim 1/\sqrt{T}.$$

[11] σ^* is a distribution over joint actions

[12] We can understand the LHS of this inequality as the deviation from performing action s_i and ignoring the recommendation.



5.1.3 Correlated Equilibrium

Definition 11. *A distribution σ^* is a **correlated equilibrium** if*

$$\mathbb{E}_{x\sim\sigma^*}(U_i(\phi(x_i), x_{-i})) \leq \mathbb{E}_{x\sim\sigma^*}(U_i(x))$$

for all $i \in [n]$ and all $\phi_i : X_i \to X_i$, which are functions that map actions to any other actions.

Nash equilibrium is a special case of correlated equilibrium, and correlated equilibria are a subset of coarse correlated equilibria. For any reasonable game, these categories are nonempty.

5.1.4 Phi Equilibrium

We can generalize these ideas to Φ **equilibrium**, which represents a strategy profile where players have low regret regarding a specific set of deviations Φ. CCE and CE are special cases of Φ equilibrium:

Definition 12. *A distribution σ over X is a Φ **equilibrium** if*

$$\mathbb{E}_{x\sim\sigma}(U_i(\phi(x_i), x_{-i})) \leq \mathbb{E}_{x\sim\sigma}(U_i(x))$$

for all $i \in [n]$ and all $\phi_i \in \Phi_i \subseteq X_i^{X_i}$. In particular,

- *If Φ_i contains exactly all constant functions, then this is a coarse correlated equilibrium*[13].
- *If $\Phi_i = X_i^{X_i}$, the set of all functions from $X^i \to X^i$, then this is a correlated equilibrium*[14].

5.1.5 Phi Regret Minimization

We define the Φ-regret for action set Ω, loss functions f^t, and $\Phi \subseteq \Omega^\Omega$ as

[13] A constant function $\phi_i \in \Phi_i$ represents player i committing to a fixed action.
[14] Φ_i represents all possible swap functions for player i.

$$\Phi\text{-Reg}_A(T) := \sup_{f^{\leq T}} \left(\sum_{t=1}^{T} f^t(x^t) - \inf_{\phi \in \Phi} \sum_{t=1}^{T} f^t(\phi(x^t)) \right).$$

We can use a general recipe to minimize Φ-Regret:

Theorem 6 (GGM08)**.** *For all $\phi \in \Phi$, there exists an action $x \in \Omega$ such that $\phi(x) = x$.*

Furthermore, for a loss sequence $f^1, ..., f^T$ and action sequence $x^1, ..., x^T$, there exists a no-external-regret of Φ algorithm A^Φ that, given $f^{<t}$, $x^{<t}$, produces this fixed point ϕ at iteration t, denoted ϕ^t.

$$Reg_{A^\Phi}(T) := \sup_{f^{\leq T}, x^{\leq T}} \left(\sum_{t=1}^{T} f^t(\phi^t(x^t)) - \inf_{\phi \in \Phi} \sum_{t=1}^{T} f^t(\phi(x^t)) \right)$$

The algorithm to minimize Φ-regret is as follows:

- Feed $f^{<t}, x^{<t}$ to A^Φ and receive ϕ^t
- Compute x^t such that $\phi^t(x^t) = x^t$
- Play x^t

Proof.

$$\begin{aligned}
\Phi\text{-Reg}_A(T) &= \sum_{t=1}^{T} f^t(x^t) - \inf_{\phi \in \Phi} \sum_{t=1}^{T} f^t(\phi(x^t)) \\
&=_{\text{fixed pt}} \sum_{t=1}^{T} f^t(\phi^t(x^t)) - \inf_{\phi} \sum_{t=1}^{T} f^t(\phi(x^t)) \\
&\leq \text{Reg}_{A^\Phi}(T).
\end{aligned}$$

□

5.2 Blum-Mansour

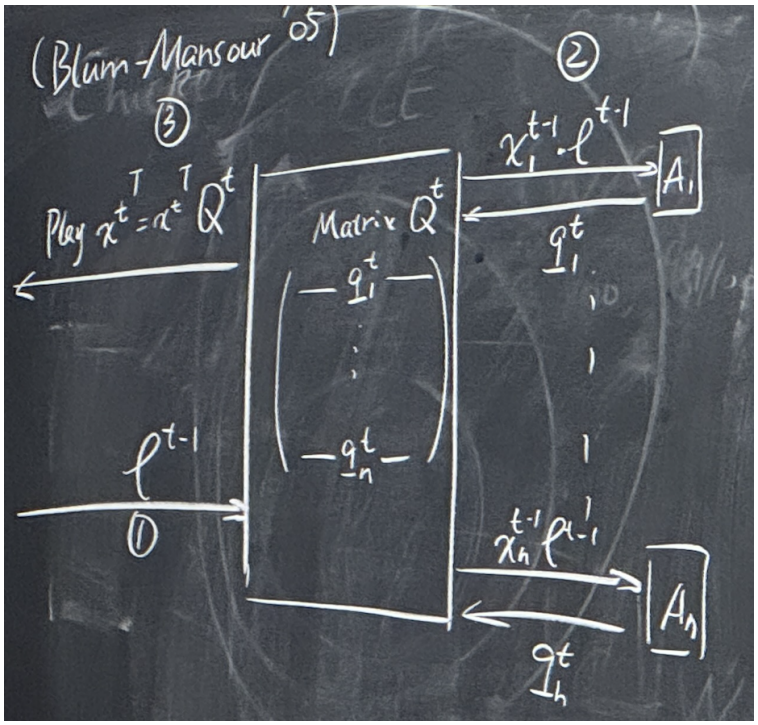
For $\Omega = \Delta^n$, $f^t(x) = \langle \ell^t, x \rangle$ and Φ the set of all linear transformations from $\Omega \to \Omega$, or equivalently

$$\equiv \{Q \in \mathbb{R}^{n \times n}, Q \text{ is row stochastic}\},$$

define $\phi^Q(x) = (x^T Q)^T$. The regret under this choice of Φ is known as the **swap regret**[15].

We can think of swap regret with a simple analogy: something like, "every time you bought IBM stock, you should have bought Microsoft instead." There is a reduction from Blum and Mansour (2005) that converts any low external regret algorithm to a low swap regret algorithm.

[15]In the case of pure actions, since Q is row stochastic and maps actions to other actions, it will map pure actions to other pure actions. This is the generalization where x is a choice from a simplex.



Blum-Mansour instantiates n copies $A_1, ..., A_n$ (image right) of an external regret minimization algorithm. Each A_j is responsible for the expected regret ove rthe number of times we play action j instead of a better action. Each algorithm A_j will propose a distribution q_j over the actions.

Q is a matrix where each row is q_j. In the first step, all proposed distributions q_j are uniform and equal.

Given Q, compute p such that $p = pQ$. Since Q is row stochastic we can interpret it as a transition matrix for a Markov chain, so p exists as its stationary distribution. This p is the distribution to select an algorithm A_j to select an action.

After playing distribution p, we receive some cost c. The cost is split up and fed to each A_j by giving it $p_j c$ and we repeat the process[16]

$$\begin{aligned}\sum_{t=1}^{T}\langle \ell^t, (x^t)^T Q^t\rangle - \sum_{t=1}^{T}\langle \ell^t, (x^t)^T Q\rangle &= \sum_{t=1}^{T}\langle \ell^t, \sum_i x_i^t q_{-i}^t\rangle - \sum_t \langle \ell^t, \sum_i x_i^t q_{-i}\rangle \\ &= \sum_i \left(\sum_t \langle x_i^t \ell^t, q_{-i}^t\rangle - \sum_t \langle x_i^t \ell^t, q_i\rangle\right)^i \\ &\leq \sum_i \mathrm{Reg}_{A_i}(T).\end{aligned}$$

Therefore, the regret is $O(n\sqrt{\log n \cdot T}$.

[16]The above explanation was taken from this CMU article and the Blum Mansour paper.

6 Lecture 6 – Feb 17

6.1 Swap Regret and Manipulation

There are different notions of regret in games. We usually look at external regret, which benchmarks against the best static action in hindsight. Algorithms like Hedge FTRL, FTPL, OGD, and more can minimize external regret.

There is also **swap regret**, where we benchmark against the best "swap function" which maps actions ot other actions. Algorithms that minimize swap regret are generally more complex than those for minimizing external regret. Therefore, why should we care about swap regret at all? This lecture will discuss perspectives on swap regret.

The canonical answer to the above question is that **minimizing swap regret leads to convergence to correlated equilibrium (CE)**. We will see that there are two other motivating reasons to care about swap regret:

- Low swap regret $\Leftrightarrow$ "non manipulability."
- Swap regret is "geometrically minimal."

With regard to the second bullet point, we will see that learning algorithms can be associated to convex sets. In this view, swap regrets algorithms have particularly nice properties.

6.2 Learning in Games

Say we have a two player game with the **Learner (L)** as one player and the **Optimizer (O)** as the other. The game is repeated for T rounds, and in each round t,

- The learner takes some action $x_t \in \Delta_m$
- The optimizer plays $y_t \in \Delta_n$.

Each player has a bilinear utility function:

$$U_L(x, y), U_O(x, y)$$

can be understood as $m \times n$ matrices that provide the corresponding utility for each possible x, y action pair.

The learner wants to maximize $\sum_{t=1}^{T} U_L(x_t, y_t)$.

A short tangent: why do we pitch the setup as a **learner vs optimizer** problem? The learner picks their action x_t by following a learning algorithm A, i.e.,

$$x_t = A(y_1, y_2, ..., y_{t-1}.$$

The optimizer will pick y_t to maximize $\sum_{t=1}^{T} U_O(x_t, y_t)$, and we can understand the optimizer as knowing what A the learner is using. The optimizer is smart and chooses actions y_t to maximize its total payoff.

Meanwhile, the learner is looking at the actions the optimizer has played and is using some algorithm to decide what to play next.

The regrets in this setup are

$$\text{ExtReg}(x_{1:T}, y_{1:T}) = \max_{x^* \in \Delta_m} \left(\sum_{i=1}^{T} (U_L(x^*, y^*) - U_L(x_t, y_t)) \right)$$

$$\text{SwapReg}(x_{1:T}, y_{1:T}) = \max_{\Pi:[m]\to[m]} \left(\sum_{i=1}^{T} (U_L(\Pi(x_t), y_t) - U_L(x_t, y_t)) \right).$$

In zero sum games, $U_O(x, y) = -U_L(x, y)$. These games have nice properties due to **Minimax Theorem**:

Theorem 7 (Minimax)**.** *For a zero sum game, there exists a value V and $x^* \in \Delta_m, y^* \in \Delta_n$ such that*

- *For any $y \in \Delta_n$, $U_L(x^*, y) \geq V$.*
- *For any $x \in \Delta_m$, $U_O(x, y^*) \leq V$.*

Corollary 1. *If L runs a no external regret algorithm, then*

$$\sum U_L(x_t, y_t) \geq VT - o(T).$$

However, these properties will totally fail for **general sum games.**

Definition 13 (General Sum Game)**.** *A **general sum game** is a game where the utilities for learner L and optimizer O are not necessarily $U_O = -U_L$.*

What happens when L runs a no external regret algorithm in a general sum game?

- O will play $y_t = y^*$ for all t.
- L will play $x^* \in \text{argmax}_{x \in \Delta_m} U_L(x, y^*)$, which we will also denote $BR_L(y^*)$, i.e., the best response to O playing y^*.
- O gets $U_O(x^*, y^*)$.

But if O chooses y^* optimally, they will get the **Stackelberg regret**,

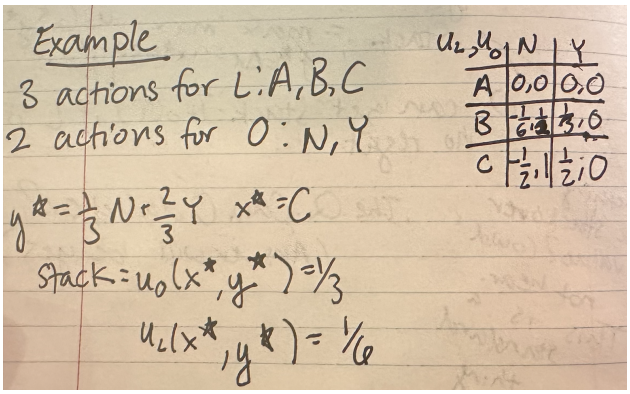
$$\text{Stack} = \max_{y^* \in \Delta_n} \max_{x^* \in BR_L(y^*)} U_O(x^*, y^*).$$

We will see that O can actually do better than this. Therefore, we cannot rely on the Minimax Theorem, and O can in fact extract almost all the utility via a different method.

6.3 Failure of Regret Minimization

Theorem 8 (BMSW18)**.** *In a repeated auction, if the bidders run "mean-based" no-regret algorithms (i.e., Hedge), then the auctioneer can extract the full welfare of the bidders.*

The auction setup in the theorem is complex to walk through, so we will illustrate with a simpler example.



There exists a dynamic strategy for O that will get them more utility when used against Hedge (or similar algorithm). In this example, suppose) plays

- $\frac{1}{3}N + \frac{2}{3}Y$ for the first $\frac{T}{2}$ rounds.
- N for the last $\frac{T}{2}$ rounds.

In the second half of the game, O is giving no utility to the learner. But for L running Hedge,

- In the first $\frac{T}{2}$ rounds, L will play action C.
- In the second $\frac{T}{2}$ rounds, L will play action B.

This will result in L rapidly losing utility in the second half of the game. This sort of phenomenon is a fundamental issue with mean-based no regret algorithms. So what learning algorithms don't have these flaws?

6.4 Non Manipulability

Definition 14 (Non Manipulability)**.** *A learning algorithm is non manipulable if any O playing against the algorithm gets at most Stack $\cdot\, T - o(T)$ utility.*

Theorem 9. *A learning algorithm A is* ***non manipulable*** *if and only if A is a no swap regret algorithm.*

To better understand the above theorem, we return to the learning algorithm-convex set correspondence we prefaced in the start of section 6. These convex set representations are called **menus**.

6.4.1 Menus

Definition 15 (Correlated Strategy Profile (CSP))**.** *Consider an action sequence $x_1, x_2, ..., x_T$ and $y_1, y_2, ..., y_T$. The* ***correlated strategy profile (CSP)*** *is*

$$\Phi = \frac{1}{T}\sum_{t=1}^{T} x_t \otimes y_t,$$

where $\otimes$ is the tensor product, so

$$x \otimes y = xy^T = \begin{pmatrix} x_1y_1 & x_1y_2 & \dots & x_1y_T \\ x_2y_1 & x_2y_2 & \dots & x_2y_T \\ \vdots & \vdots & \ddots & \vdots \\ x_Ty_1 & x_Ty_2 & \dots & x_Ty_T \end{pmatrix}$$

The quantities $U_L(\Phi), U_O(\Phi), \text{Reg}(\Phi), \text{SwapReg}(\Phi)$ are all defined. For example, we can understand $U_L(\Phi)$ as the average learner utility over the course of the game.

Φ describes the transcript of the game through utilities. How can we transform this into a convex set?

Definition 16 (Menus)**.** *The **menu** of a learning algorithm A is*

$$\textit{Menu}(A) = \{\Phi | O \textit{ can pick } y_t \implies \Phi\},$$

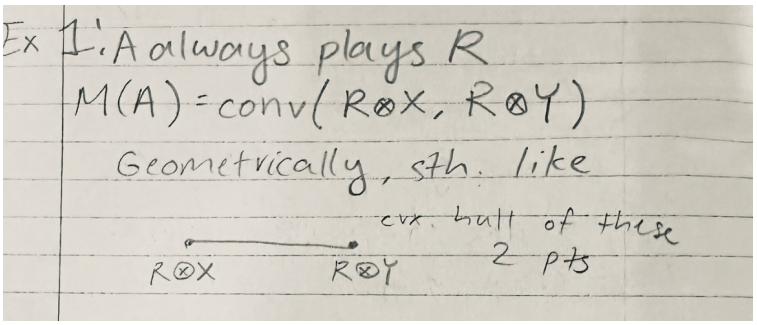
that is, the set of all possible correlated strategy profiles that the optimizer can induce under its choice of action, which we can restrict to

$$M(A) = \overline{conv}(\{\Phi | y_1, ..., y_T \in \Delta_n, x_t = A(y_1, ..., y_{t-1}\ \forall t\}).$$

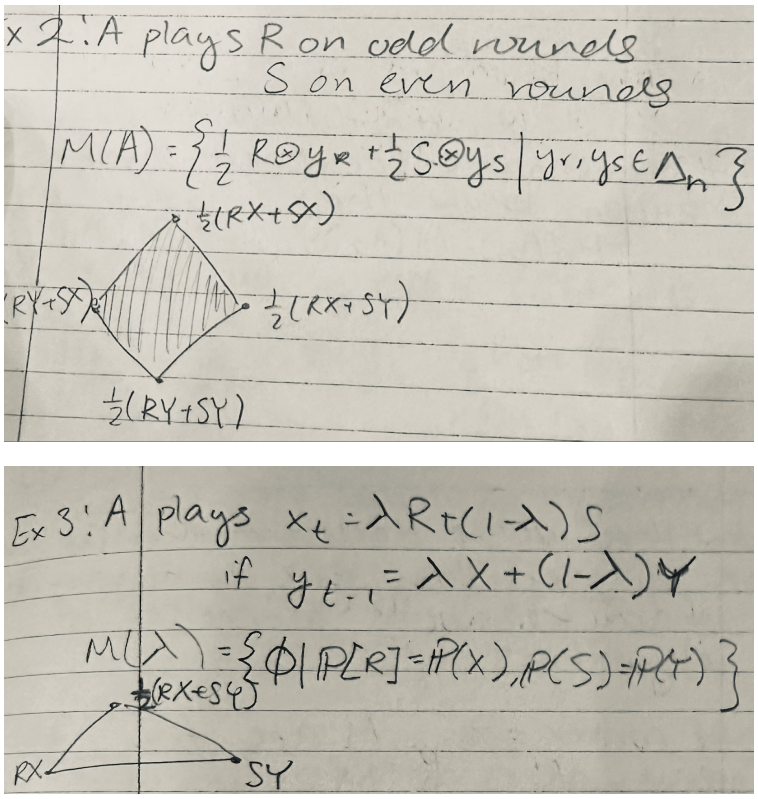
This convexification is valid because the optimizer is at most concerned about linear utilities (convex to linear reduction)[17].

Intuitively, the menu is the convex set of the possible joint distributions that the optimizer can induce when facing an algorithm A. Therefore, the menu of an algorithm A is more like a menu of options that the *optimizer* can choose to induce through its own action when playing *against* A.

Some example menus for very simple games where L has 2 actions $\{R, S\}$ and O has 2 actions $\{X, Y\}$:



[17]Also note that the above menus have been defined for finite T round games. For generic time horizons, there is some sort of argument for convergence $M(A_1), M(A_2), ..., \to M(A)$ that can be made rigorous.



6.5 Characterization Theorems

Now that we've discussed menus, we will see that they give us nontrivial characterizations of classes of algorithms.

What convex sets M are the menus of

- Some learning algorithm?
- Some no regret algorithm?
- Some no swap regret algorithm?

Theorem 10. *M is the menu of a learning algorithm iff $\forall y \in \Delta_n, \exists x \in \Delta_m$ such that $x \otimes y \in M$.*

Proof. ($\rightarrow$). What if O plays only y? Then $(\frac{1}{T}\sum_t x_t) \otimes y \in M$.

($\leftarrow$). Blackwell's approachability theorem. □

Theorem 11. *M is the menu of a no swap regret learning algorithm if and only if*

$$M = conv\{BR(y) \otimes (y) | y \in \Delta_n\}.$$

Proof. (Show conv $\subseteq M$). If O plays y every round any no regret L must play $BR(y)$ in most rounds.

(Show $M \subseteq$ conv). Pick $\phi \in M$. Decompose ϕ as

$$\phi = \sum_{i=1}^{m} \lambda_i (e_i \otimes y_i),$$

where e_i the standard basis vector and y_i the average action of the optimizer when the learner plays action i.

Then

$$\text{SwapReg}(\phi) = \sum_{i=1}^{m} \lambda_i \cdot \max_{i^*}(U_L(i^*, y_i) - U_L(i, y_i)),$$

where $i^* = \Pi^*(i)$, the output of the swap. By the assumption of no swap regret, $\text{SwapReg}(\phi) = 0$ implies

$$\forall i \in [m], U_L(i, y_i) = \max_{i^*} UL(i^*, y_i)$$

and

$$i \in BR(y_i).$$

□

Thus, we have seen a connection between learning algorithms and the set of convex sets. Furthermore, we can see that algorithms with no swap regret correspond with the convex set of distributions of any optimizer action y and the *best response* to it for the learner.

7 Lecture 7 – Feb 24

Recall Φ regret minimization where Φ is the set of swap functions $\Omega \to \Omega$

$$\Phi - \text{Reg}_A(T) = \sum_{t=1}^{T} f^t(x^t) - \min_{\phi \in \Phi} \sum_{t=1}^{T} f^t(\phi(x^t)),$$

i.e., the cumulative loss minus the regret for the best possible swap function.

From Gordon et al., given

1. $\forall \phi \in \Phi, \exists x \in \Omega$ such that $\phi(x) = x$ (a fixed point)
2. and an external regret minimizer over Φ,

we can obtain an algorithm that also has low **swap regret**. A concrete example of this for finite action sets is Blum-Mansour. (remark: in recent work, condition 1 is relaxed and new conclusions can be derived)

Today, we will discuss *a black box method that reduces the problem from swap regret minimization to external regret minimization.*

Recall that Blum-Mansour gives regret bound $\sqrt{n \log(n) \cdot T}$ where n is the number of actions. This regret bound isn't great because of the polynomial n dependence on n, and n can be exponentially large. In

particular, this is the case for extensive form games, which can be flattened to normal form games before applying Blum-Mansour. However, this may take a long time because

$$\frac{\sqrt{n\log(n)\cdot T}}{T} = \epsilon \implies T = \frac{n\log(n)}{\epsilon^2}.$$

We can achieve better dependence on n with other methods, but it will require accepting a worse dependence on T: there is sadly no free lunch.

7.1 Treeswap DDFG'24, PR'24

The **Treeswap algorithm** gives us regret bound

$$T = \left(\frac{\log n}{\epsilon^2}\right)^{1/\epsilon}.$$

For every fixed ϵ, the above is polynomial, but this isn't particularly fast, either. At the current moment, taking the minimum of (Treeswap, Blum-Mansour) is the best regret bound we can hope for.

7.1.1 Treeswap Setup

- Construct a depth d tree which is m-ary. Let $T = M^d$ and A be an algorithm that minimizes external regret.
- Fix ϵ and run the Treeswap algorithm for M^d rounds, and the average regret will be $\leq \epsilon$, i.e.,

$$\frac{\mathrm{Reg}_A(M)}{M} \leq \epsilon.$$

We can imagine the Treeswap algorithm as maintaining a copy of A at every node and controlling when to activate and deactivate each node.

7.1.2 Treeswap Algorithm

1. Each instance of A at level i is used for M^{d-i} rounds. We can fix an order for nodes to be used; WLOG, let's say we're going from left to right on each row.
2. For A at level i, we update every M^{d-i-1} rounds. A is updated with the average loss from the previous M^{d-i-1} rounds.
3. At each round t, output a uniform mixture over the d distributions of the active heads.

Intuition: this is essentially an update rule s.t. at any given moment, the only active nodes are a path from root to leaf node. This path, over the course of the algorithm, will slowly traverse from (root to the leftmost leaf node) to connecting (root to the rightmost leaf node). At all times there will be one active path, and it will be the shortest one.

Proof. At each level i, examine 2 quantities R_i, S_i. R_i is the average loss of all algorithms at level i over T rounds. S_i is defined for each block of size M^{d-i}, where it is the loss of the best fixed action in hindsight, averaged over T.

Note that for all i, $R_i - S_i \leq \epsilon$. Secondly, for all $i \leq d-1$, we have that

$$\frac{R_i - S_{i+1}}{d}$$

is level i's total contribution to the swap regret. This is because S_{i+1} corresponds to level $i+1$, where algorithms update every M^{d-i} rounds instead of M^{d-i-1} at level i. So for any M^{d-i} block of S_{i+1}, we are looking at the algorithms of level i, had they updated (check this?)

$$\text{SwapReg}_{\text{Treeswap}}(T) \leq \frac{1}{d}\sum_{i=0}^{d-1}(R_i - S_{i+1}) = \frac{1}{d}\left(\sum_{i=0}^{d-1}(R_i - S_i)\right) + \frac{S_0 - S_d}{d}.$$

Assume the losses are bounded between $[0,1]$. Then S_0, S_1 are both bounded between $[0,1]$ and $\frac{S_0 - S_1}{d} \leq \frac{1}{d}$. Therefore, the above is less than $\epsilon + \frac{1}{d}$.

For example, suppose A is MWU. Then $M = \frac{\log(n)}{\epsilon^2}$ and $T = M^d = \frac{\log(n)}{\epsilon^2}^{1/\epsilon}$.

□

Remark 3. *We use $M = \frac{\log n}{\epsilon^2}$ to compare against Blum-Mansour, but in general, there may not be a finite number of actions n. All we actually need from A is that it minimizes ExtReg, and minimal assumptions are made on Φ, as well. Therefore, Treeswap works for any Φ.*

7.2 Calibration

(definition of calibration and rainmaker example of calibrated forecasts)

For all $x \in [0,1]$, define

$$p_\epsilon^t(x) = \frac{\sum_{\tau=1}^t y^\tau \mathbb{1}\{q^\tau \in [x-\epsilon, x+\epsilon]\}}{\sum_{\tau=1}^t \mathbb{1}\{q^\tau \in [x-\epsilon, x+\epsilon]\}},$$

where y^τ is the outcome on round τ. In the rainmaker example, the above is essentially (# times model predicted rain and it rained)/(# number of times it rained in general).

Definition 17 (ϵ-calibration)**.** *A model is ϵ-calibrated if $\forall x \in [0,1]$ for which*

$$limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=1}^t \mathbb{1}\{q^\tau \in [x-\epsilon, x+\epsilon]\} > 0,$$

we have that

$$limsup_{t\to\infty} |p_\epsilon^t(x) - x| \leq \epsilon.$$

7.2.1 An algorithm for calibration

Let

$$q^t = I^t/N,$$

where $I^t \in [N]$. We use the loss function $f^t(i) = (y^t - i/N)^2$.

Define the $\mathcal{L}^2$ **calibration error** as

$$C(T) = \sum_{i=0}^{N} \left(p_\epsilon^T(i/N) - i/N\right)^2 \left(\frac{1}{T}\sum_{\tau=1}^{T} p_i^t\right).$$

Note that each $\rho^t \in \Delta^N$.

We can express the per-round regret of swapping from i to j as

$$\begin{aligned} r_{ij}^t &= p_i^t \left(\left(y^t - \frac{i}{N}\right)^2 - \left(y^t - \frac{j}{N}\right)^2\right) \\ &= p_i^t \frac{j-i}{N}\left(2y^t - \frac{i+j}{N}\right). \end{aligned}$$

Then the cumulative swap regret over T rounds is

$$\begin{aligned} R_{i,j}^T &= \sum_{i=1}^{T} r_{ij}^t \\ &= \frac{2(j-i)}{N}\sum_{t=1}^{T} p_i^t \left(y^t - \frac{i+j}{2N}\right) \\ &= \frac{2(j-i)}{N}\left(\sum_{t=1} p_i^t\right)\left(p^T\left(\frac{i}{N}\right) - \frac{i+j}{N}\right) \\ &= \left(\sum_{t=1}^{T} p_i^t\right)\left(\left(p^T\left(\frac{i}{N}\right) - \frac{i}{N}\right)^2 - \left(p^T\left(\frac{i}{N}\right) - \frac{j}{N}\right)^2\right). \end{aligned}$$

Claim 4. *In the continuous setting, swap regret equals calibration error.*

In our case, this is not true, because we are confined to picking discrete actions $1/N$. Therefore, we have some error, which we can see as a gap in the following bound:

$$\begin{aligned} C(T) &\le \frac{1}{T}\left(\sum_{i=0}^{N} \max_j \left(R_{i,j}^T + \min_j \left(p^T\left(\frac{i}{N}\right) - \frac{j}{N}\right)^2\right)\right) \\ &\le \frac{1}{T}\sum_{i=0}^{N} \max_j R_{i,j}^T + \frac{N+1}{N^2}. \end{aligned}$$

7.3 Treecal

The goal of the **Treecal algorithm** is to get efficient calibration by running Treeswap with a smarter base algorithm than MWU.

An outline of the alg:

1. Calibration error with respect to the Bregman divergence equals the swap regret with respect to the Bregman divergence
2. If the Bregman divergence D_R is 1-strongly convex with respect to some norm, then D_R will dominate the norm squared
3. Use Treeswap where the algorithm is follow the leader. Take

$$\text{argmin}_x \sum_{\tau=1}^{t} D_R(y^t, x) = \frac{1}{t} \sum_{\tau=1}^{t} y^\tau.$$

8 Lecture 8 – Mar 24

8.1 Prediction/Optimism for No-Regret Algorithms

Optimism is a variant in online convex optimization in which the player makes some sort of prediction about the loss they will incur in the oncoming round. Roughly, an optimistic no-regret algorithm proceeds as follows:

1. Learner obtains a prediction about the loss at round t, $m_t \in \mathbb{R}^d$.
2. Learner picks an action $x^t \in \Omega$.
3. Environment picks ℓ^t.
4. Learner sees ℓ^t and suffers loss $\langle x^t, \ell^t \rangle$.

Definition 18 (Optimistic Follow the Regularized Leader)**.** *Recall that the player's action in FTRL is*

$$x^t = \textit{argmin} \langle x, \ell^{<t} \rangle + \frac{1}{\eta} \phi(x),$$

where ϕ is a strongly convex function.

In optimistic FTRL (OFTRL), after receiving a prediction m^t, the player's action at round t is

$$x^t = \textit{argmin} \langle x, \ell^{<t} + m^t \rangle + \frac{1}{\eta} \phi(x).$$

This is a simple change; essentially, we want to minimize the loss we think we will incur over the history *and* the coming round. We will show that OFTRL is good if m^t is good, that is, we will obtain regret bounds that depend on the quality of m^t.[18]

First, we seek to prove the following theorem.

[18] This is intuitive. For one, if m^t is perfect, OFTRL is BTRL.

Theorem 12. *OFTRL ensures that, for any $u \in \Omega$,*

$$\sum_{t\le T}\langle x^{t-u,\ell^t\rangle} \le \frac{\phi(u)-\phi(\hat{x}^1)}{\eta} + \sum_{t\le T}\langle x^t - \hat{x}^{t+1}, \ell^t - m^t\rangle - \frac{1}{\eta}\sum_{t\le T}(D_\phi(x^t,\hat{x}^t) + D_\phi(\hat{x}^{t+1},x^t))$$

where $\hat{x}^t = argmin\langle x, \ell^{<t}\rangle + \frac{1}{\eta}\phi(x)$ and D_ϕ is the Bregman divergence.

Before we prove 12, we will need the following.

Fact 1. *If $F : \Omega \to \mathbb{R}$ is convex and differentiable and $z^* \in \Omega$ minimizes F, then $F(z^*) \le F(z) - D_F(z, z^*)$.*

Lemma 4. *Define the following:*

$$\begin{aligned} x^* &\in argmin_{x\in\Omega}\langle x, L\rangle + \phi(x)\\ \phi &= min_{x\in\Omega}\langle x, L\rangle + \phi(x)\\ \bar{x}^* &\in argmin_{x\in\Omega}\langle x, \bar{L}\rangle + \phi(x)\\ \bar{\phi} &= min_{x\in\Omega}\langle x, \bar{L}\rangle + \phi(x)\end{aligned}$$

Then $\bar{\phi} - \phi \le \langle x^*, \bar{L} - L\rangle - D_\phi(x^*, \bar{x}^*)$.

Proof of 4. Take F to be $\langle x, \bar{L}\rangle + \phi(x)$ and apply 1. □

Now we can prove 12.

Proof. Define

$$\hat{\phi}^t = \langle \hat{x}^t, \ell^{<t}\rangle + \frac{1}{\eta}\phi(\hat{x}^t)$$

and

$$\phi^t = \langle x^t, m^t + \ell^{<t}\rangle + \frac{1}{\eta}\phi(x^t).$$

By 4, we have

$$\hat{\phi}^t - \phi^t + \langle x^t, m^t\rangle \le -\frac{1}{\eta}D_\phi(x^t,\hat{x}^t) \tag{1}$$

and

$$\phi^t - \hat{\phi}^{t+1} + \langle \hat{x}^{t+1}, \ell^t - m^t\rangle \le -\frac{1}{\eta}D_\phi(\hat{x}^{t+1},x^t). \tag{2}$$

Combining the two, we have

$$\langle x^t, \ell^t \rangle \leq \hat{\phi}^{t+1} - \hat{\phi}^t + \langle x^t - \hat{x}^{t+1}, \ell^t - m^t \rangle - \frac{1}{\eta}(D_\phi(x^t, \hat{x}^t) + D_\phi(\hat{x}^{t+1}, x^t)). \tag{3}$$

Summing both sides over $\sum_{t=1}^{T}$, we have a telescoping sum that yields a $\hat{\phi}^{T+1}$ term, which we bound

$$\hat{\phi}^{T+1} \leq \langle u, \ell^{<t} \rangle + \frac{1}{\eta}\phi(u).$$

Therefore, we have

$$\sum_{t=1}^{T}\langle x^t, \ell^t \rangle \leq \sum_{t=1}^{T}\langle u, \ell^t \rangle + \frac{1}{\eta}(\phi(u) - \phi(x^1)) + \sum_{t=1}^{T}\langle x^t - \hat{x}^{t+1}, \ell^t - m^t \rangle - \frac{1}{\eta}\sum_{t \leq T}(D_\phi(x^t, \hat{x}^t) + D_\phi(\hat{x}^{t+1}, x^t)).$$

□

We can abbreviate this regret bound to the **regret bound by variation of utilities (RVU)** for ease of memorization. Roughly speaking, for ϕ a 1-strongly convex function with respect to some norm (*) and $B_\phi = \max_{x \in \Omega}\phi(x) - \min_{x \in \Omega}\phi(x)$,

$$\text{Reg}_{\text{OFTRL}} \leq B_\phi/\eta + \eta \sum_{t \leq T} ||\ell^t - m^t||_*^2 - \frac{1}{4\eta}\sum_{t=2}^{T} ||x^t - x^{t-1}||^2.$$

A common choice of m^t is simply ℓ^{t-1}, hence the term "optimism": we believe that "today" will be the same as "yesterday." A common step size choice is the adaptive

$$\eta^t = \sqrt{B_\phi / \sum_{\tau \leq t} ||\ell^\tau - m^\tau||_*^2}.$$

We will now see how this regret bound can be applied to different games.

8.2 OFTRL for 2P0SG

Suppose we have a 2 player 0 sum game with payoff matrix A that is N rows by M columns, with entries $A_{i,j}$ normalized to be between $[0, 1]$.

The loss is $x^T A y$, where $x \in \Delta^N$ and $y \in \Delta^M$. One player incurs loss $\ell^\tau = Ay^\tau$ and the other incurs $g^\tau = -A^T x^\tau$. Let

$$x_i^t \propto \exp(-\eta(\ell_i^{t-1} + \ell_\tau^{<t}))$$

for all i and

$$y_i^t \propto \exp(-\eta(g_j^{t-1} + g_\tau^{<t}))$$

for all j.

Theorem 13. *Let $\eta = 1/4$. Define the **social regret** as*

$$Reg_R(T) + Reg_C(T) = O(\ln NM),$$

where R and C denote the row and column player w.r.t. the payoff matrix. Note that the social regret has no time dependence.

Since the duality gap

$$dg(\bar{x}, \bar{y}) = \frac{Reg_R(T) + Reg_C(T)}{T},$$

$(\bar{x}, \bar{y})$ is $\frac{O(\ln NM)^T}{T}$, which is approximately Nash equilibrium.

We will use the following fact in the proof.

Fact 2.

$$\begin{aligned}\|\ell^t - \ell^{t-1}\|_\infty &= \|Ay^t - Ay^{t-1}\|_\infty \\ &= max_{i\in[N]}\langle A_i, y^t - y^{t-1}\rangle \\ &\leq max_i\|A_i\|_\infty \|y^t - y^{t-1}\|.\end{aligned}$$

Proof.

$$\begin{aligned}\mathrm{Reg}_R(T) &\leq \frac{\ln N}{\eta} + \eta\sum_{t=1}^T \|y^t - y^{t-1}\|^2 - \frac{1}{4\eta}\sum_{t=1}^T \|x^t - x^{t-1}\|_1^2, \\ \mathrm{Reg}_C(T) &\leq \frac{\ln M}{\eta} + \eta\sum_{t=1}^T \|x^t - x^{t-1}\|_1^2 - \frac{1}{4\eta}\sum_{t=1}^T \|y^t - y^{t-1}\|^2.\end{aligned}$$

Adding the two inequalities for $\eta = \frac{1}{4}$:

$$\mathrm{Reg}_R(T) + \mathrm{Reg}_C(T) \leq 4\ln(NM) - \frac{3}{4}\left(\sum_{t=1}^T \|x^t - x^{t-1}\|_1^2 + \|y^t - y^{t-1}\|^2\right).$$

Note that $\mathrm{Reg}_R(T) + \mathrm{Reg}_C(T)$ is always non-negative; it is the duality gap.[19]

Therefore,

$$\frac{3}{4}\sum_{t=1}^T \left(\|x^t - x^{t-1}\|_1^2 + \|y^t - y^{t-1}\|^2\right) \leq \ln(NM).$$

□

[19]This is true for 2P0SG but not necessarily all games.

8.3 OFTRL for Bimatrix Games

In a bimatrix game, we have two N by M matrices, A and B. Now, $\ell^\tau = Ay^\tau$ and $g^\tau = B^T x^\tau$. Suppose both players use optimistic Hedge.

Theorem 14 (Syrgkanis et al.)**.** *Using the above setup,*

$$max\{\text{Reg}_R(T), \text{Reg}_C(T)\} = \mathcal{O}\Big(T^{1/4}\left(\ln^{3/4}(MN)\right)\Big).$$

Proof of Thm 3.

$$\text{Reg}_R(T) \leq \frac{\ln N}{\eta} + \eta \sum_{t=1}^{T} \|y^t - y^{t-1}\|^2 - \frac{1}{4\eta} \sum_{t=1}^{T} \|x^t - x^{t-1}\|_1^2,$$
$$\text{Reg}_C(T) \leq \frac{\ln N}{\eta} + \eta \sum_{t=1}^{T} \|x^t - x^{t-1}\|_1^2 - \frac{1}{4\eta} \sum_{t=1}^{T} \|y^t - y^{t-1}\|^2.$$

We can simplify these inequalities with a crude bound,

$$\|y^t - y^{t+1}\|_1 \leq \eta \|2g^t - g^{t-1}\|_\infty \leq 3\eta$$

$$\text{Reg}_R(T) \leq \frac{\ln N}{\eta} + 4\eta^3 T.$$

Setting $\eta \approx T^{-1/4}$, we get the desired result.

8.4 Results on Optimism

In 2020, Chen-Peng showed $\tilde{O}(T^{1/6})$ external regret bound for 2 players and an $O(T^{1/4})$ bound for swap regret.

In 2021, Daskalakis et al. showed $O(\text{polylog}(T))$ for both external and swap regret.

These results, as well as today's theorems, also work for > 2 players, we simply showed proofs for 2-player games. These results are all for $m^t = \ell^{t-1}$. In some sense, the literature's understanding of behavior with more complex predictions is a bit weak.

It is currently an open problem to look for better than $\log(T)$ regret bounds.

9 Lecture 9 – Mar 31

10 Lecture 10 – Apr 9

11 Lecture 11, 12

Final presentations.