NIKA HAGHTALAB AND MICHAEL JORDAN

CS294: DECISIONS, LEARNING AND GAMES

Preliminaries

- **Domain (Instance space):** An arbitrary set $\mathcal{X}$ that includes all possible instances.
- **Labels:** A set $\mathcal{Y}$ that includes all possible labels or predictions for a single instance, such as $\{0, 1\}$, $\{-1, 1\}$, $\{false, true\}$, etc.
- **Labeled instance:** An instance-label pair $(x, y) \in \mathcal{X} \times \mathcal{Y}$ is called a labeled instance.
- **Hypothesis:** A *hypothesis* (a.k.a classifier or predictor) is a function $h : \mathcal{X} \to \mathcal{Y}$.
- **Hypothesis class:** A concept class $\mathcal{H}$ is a pre-determined set of concepts.
- **Consistency:** We say that a hypothesis $h \in \mathcal{H}$ is *consistent* with a set $\{(x_1, y_1), \ldots, (x_m, y_m)\}$, if for all $i \in [m]$, $h(x_i) = y_i$.
- Finite number of players: $[k] = \{1, 2, 3, ..., k\}$
- **Set of (pure) actions/strategies:** S_i
- **Cost function** for $i \in [k] : C_i : \{s_1 \times s_2 \times ... \times s_k\} \to [0, 1]$ (Note that we mostly care about boundedness of the output space)
- **Strategy profile:** $(s_1, ..., s_k)$, where $s_i \in S_i \forall i$
- Strategy of i s_i and strategies of every one except i: s_{-i}
- **Mixed strategy**: distribution σ_i over S_i
- **Mixed strategy profile**: $(\sigma_1, ..., \sigma_k)$

Lecture 1

Some Examples

In this section, we give examples of how even simple decisions have consequential implications, have to rely on learning in difficult and complex situations, and contribute to rational behavior that emerges in groups and the society. A common theme in these examples is they involve uncertainty, learning, and decentralized agency.

Example (Which route to use for your daily commute?). Everyday we need to decide on the route P^t we take to work. We want to choose a route that is the shortest (accounting for traffic), but we don't know the traffic ahead of time. In fact, in every day t, the total driving time for route P, denoted by $\text{cost}^t(P)$, may be different from what the traffic looked like in the past, e.g., due to road accidents, first day of school, and changes in the routes other drivers take. So, we want to have a strategy for picking the route P^t that leads to a short commute on average.

The average time spent on commute over T days is $\frac{1}{T}\sum_{t=1}^{T}\text{cost}^t(P^t)$. After T days, we can look back at the historical traffic data and compute the (fixed) route that would have had the best average commute time. The average commute time of the best fixed route in hindsight is $\min_{P*}\frac{1}{T}\sum_{t=1}^{T}\text{cost}^t(P^*)$. Can we have an algorithm for picking P^t that is almost as good as the best route in hindsight, i.e.,

$$\frac{1}{T}\sum_{t=1}^{T}\text{cost}^t(P^t) \leq \min_{P^*}\frac{1}{T}\sum_{t=1}^{T}\text{cost}^t(P^*) + \epsilon,$$

for an $\epsilon \to 0$?

Challenges: When the commute length (traffic) is unpredictable and possibly designed adversarially, we don't know which paths may be good in the future. It's possible that historically well-performing paths don't do well in the next round. So, we need to hedge our bets and *"explore" paths that might have not performed well historically*. We need to balance this off with "exploiting" paths that did perform well, because their average performance will remain good for some time.

Example (Route planners). You could use a route planner, like Google Maps, Apple Maps, or Waze to plan your commute. These planners run a similar "explore-exploit" algorithm in the background not only to hedge their bets but also to gain more information about the traffic as they monitor drivers in action. If this exploration suggest routes that a driver believes to be longer than other alternatives, the driver may benefit from switching his path and not following the instructions. Over the long run, this affects the quality of observations the app may receive. The apps may even lose market share if they use algorithms that explore too carelessly.

Challenges: Can we design explore-exploit algorithms that don't suggest obviously bad routes? Does the competition for market-share between several apps impact their explore-exploit tradoffs and overall learning quality?

Example (Impact of route planning at scale). The commute time and traffic is impacted by the decisions of all other drivers. Often, the more drivers on one road, the longer the commute time. So, cost(P) is formed by the impact of all driver's choices. Assume that 1000s of drivers, which we refer to as one unit of traffic, want to go from A to B. And the congestion cost of an edge, denoted by $c_e(x)$, is a function of the fraction of drivers x that that take edge e.

A social planner, who tells each person which route to take, can plan an optimal routing, i.e., the least time the society spends on commute! In the above example, when half of the drivers take AVB and the other half take AWB, then commute time is just 1.5. This allocation is also individually optimal to each player. That is, no driver can switch to a different route and reduce their own commute time. Such an allocation is called an "equilibrium".

What if the city adds a bridge between V and W, with congestion cost of $c(x) = 0$? While the previously socially optimal solution is still a valid solution with commute time of 1.5, this solution is no longer an equilibrium because players benefit from deviating from their current path and taking the bridge. In fact, the only equilibrium in this case is a solution where all drivers take the bridge and pay the commute cost of 2.

Challenges: Computing socially optimal solutions and equilibria need a lot of information and coordination. Can these be achieved by simple learning algorithms? e.g., what if each diver deploys an online learning algorithm (from Example 1) or simply picks the (instantaneous) shortest route? Can learning dynamics help us arrive at good solutions or are they inherently different from the type of coordination needed for finding these solutions?

Other examples of applications our theoretical framework needs to

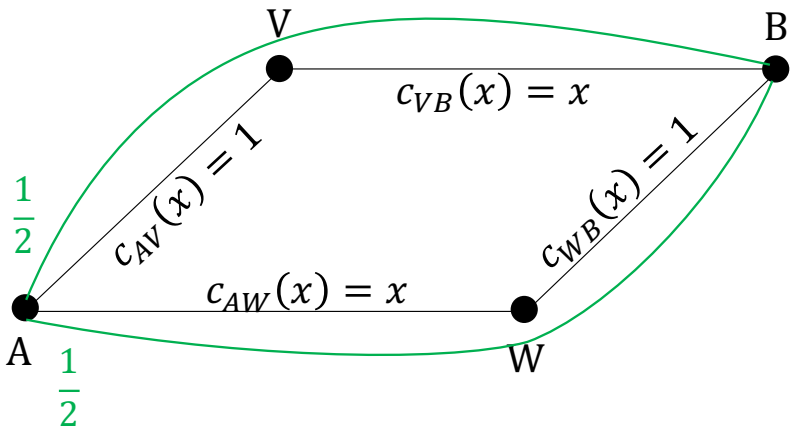


Figure 1: The green allocation is the social optima and an equilibrium. It has a cost of 1.5

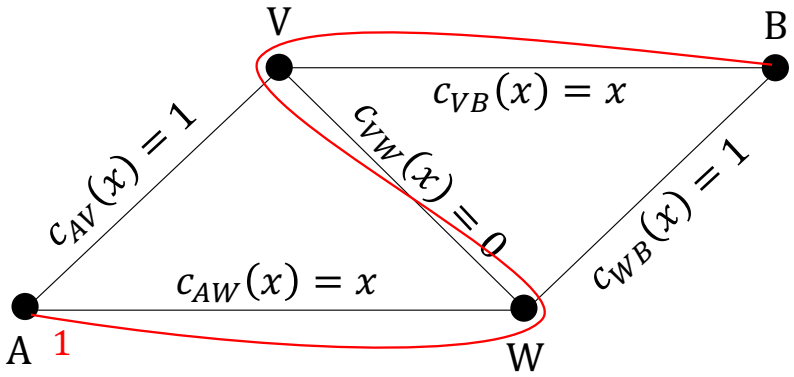


Figure 2: The red allocation is the only equilibrium and costs 2. The allocation in Fig 1 is still a valid solution with cost 1.5.

address include matching drivers and riders in rideshare programs and deciding how to split costs between friends who are sharing a ride.

Online Model

We start with a formal model of learning and learnability in presence of unpredictable, even possibly adversarial, sequences.

Consider a scenario where we are faced with a sequence of incoming instances and we need to make an irrevocable decision about the instance as soon as we see it. To provide robust learning guarantees, we assume that an all-powerful adaptive adversary is choosing the instances and therefore we do not know how well our decisions will perform a priori. An example of this scenario is deciding which route to take to arrive on campus everyday without knowing the traffic en route a priori. At the end of the day, we can look back at the traffic in the city (on all roads) and decide on the route we will be taking tomorrow. What you want is that, over a long period of time, we don't do much worse than if we took the single best route in hindsight.

In another example, consider the setting where everyday we guess whether the stock market index is going to go up or down. You may rely on some recommendation sources: Wall Street Journal, CNN, your coworker, your psychic(!), etc. At the end of the day, you observe whether you were right or wrong. Your goal is not to make many more mistakes compared to your best advisor in hindsight.

This type of decision making will be formalized here and in the next few lectures.

Mistake Bound Model

A simple model for formalizing online decisions is the Mistake Bound model. Given a class of hypotheses $\mathcal{H}$, for $t = 1, 2, 3, ...$

- Receive an $x^t \in \mathcal{X}$.
- Predict $\hat{y}^t \in \{-, +\}$.
- Receive the true label y^t. A mistake is made if $\hat{y}^t \neq y^t$

The goal is to bound the number of mistakes you make, assuming that the sequence of labeled instances (even if produced by an adversary) is consistent with respect to an unknown hypothesis $h^* \in \mathcal{H}$, i.e., for all t, $y^t = h^*(x^t)$. At every round t, the algorithm has access to the sequence of past experience, i.e., $\{(x^1, y^1), (x^2, y^2), \cdots, (x^{t-1}, y^{t-1})\}$.

Definition 1. An algorithm $\mathcal{A}$ learns the hypothesis class $\mathcal{H}$ with mistake bound M, if M is the minimum integer for which the follow-

ing statement holds: For *any* online sequence $(x^1, y^1), (x^2, y^2), ...$ that is consistent with respect to some unknown $h^* \in \mathcal{H}$, $\mathcal{A}$ makes at most M mistakes.

Next, we provide a general-purpose algorithm for learning in the mistake bound model.

One way to look at Mistake Bound model is is that our hypotheses in $\mathcal{H}$ are like "experts". Each expert will make a prediction at every time step t. There is at least one expert who is always right (by consistency) but we don't know who it is. We can observe the history of predictions by every expert to make a decision at time t. Our goal is not to figure out who this perfect expert is, rather, to not make more than M mistakes. In fact, there is a simple and universal algorithm for learning any finite class of hypotheses in the mistake bound model.

Majority/Halving Algorithm Algorithm: At time t, consult all h_i that have yet to make a mistake. Go with their majority vote.

Theorem 1. *The Majority Algorithm has mistake bound* $M \leq \lfloor \log_2 |\mathcal{H}| \rfloor$.

Proof. Note that the mistake bound model assumes that there is some consistent $h^* \in \mathcal{H}$ such that $\forall t, h^*(x^t) = y^t$. Let $S_t = \{h_i : h_i \text{ has not made a mistake so far}\}$ and let $W_t = |S_t|$. If the majority make a mistake, then at least half of S_t were wrong. Thus, $W_{t+1} \leq 0.5 W_t$. Furthermore, we know $W_t \geq 1$ for all t since we assume there exists a "perfect" expert that is always right. This implies that you can halve W_t at most $\lfloor \log_2(|\mathcal{H}|) \rfloor$ times. □

The Majority algorithm relies heavily on the existence of a perfect expert, i.e., a consistent hypothesis. What if there is no consistent $h^* \in \mathcal{H}$? Can we still provide good guarantees on the mistake bound? This is what we will study in the next lecture.

Lecture 2

Learning under Misspecification

In the last lecture, we saw how the Majority Algorithm can have at most $\lfloor \log_2 |\mathcal{H}| \rfloor$ mistakes regardless of the time horizon we run the algorithm for. However, a crucial assumption used in Theorem 1 is that the sequence $\{(x^i, y^i)\}_{i=1}^{\infty}$ is consistent with respect to some unknown $h^* \in \mathcal{H}$. What if there is no such consistent h^*?

We can instead compare our algorithm to the function in $\mathcal{H}$ that performs the best. Let **OPT** denote the number of mistakes made by the best function in $\mathcal{H}$, and denote the function as h^*, i.e.,

$$\textbf{OPT} = \min_{h \in \mathcal{H}} \sum_t \mathbb{I}\{h(x^t) \neq y^t\}, \tag{1}$$

$$h^* = \arg\min_{h \in \mathcal{H}} \sum_t \mathbb{I}\{h(x^t) \neq y^t\}. \tag{2}$$

Weighted Majority Algorithm

Algorithm 1: Weighted Majority (ϵ)

procedure WEIGHTED MAJORITY(ϵ)
  $w_1^1, \ldots, w_n^1 \leftarrow 1$ initial weights ▷ w_i^t is expert i weight @ time t
  for $t = 1, \ldots$ **do**
    $W^t \leftarrow \sum_{i=1}^n w_i^t$
    if $\sum_{i:h_i(x_t)=1} w_i^t \geq W^t/2$ **then**
      Predict $\hat{y}^t \leftarrow 1$
    else
      Predict $\hat{y}^t \leftarrow 0$
    end if ▷ Observe y^t after prediction
    $E^t \leftarrow \{i : h_i(x_t) \neq y^t\}$
    $w_i^{t+1} = w_i^t \left[(1-\epsilon)^{\mathbb{I}\{i \in E^t\}}\right] \forall i$
  end for
end procedure

For convenience, define $n = |\mathcal{H}|$.

Theorem 2. *Weighted Majority(ϵ) algorithm makes at most* $\frac{2}{1-\epsilon}\mathbf{OPT} + \frac{2}{\epsilon}\ln n$. *The described bound is weaker version, in particular, for* $\epsilon = 0.5$, *the bound becomes* $2.4(\mathbf{OPT} + \log_2 n)$.

Proof. We shall derive the bound for $\epsilon = 0.5$. Intuitively, W^t is like the credibility of all the experts combined, which decreases as the algorithm makes mistakes. Let i^* refer to the index of optimal expert, then, we have the bound

$$W^t \geq w_{i^*}^t \geq (\frac{1}{2})^{\mathbf{OPT}}.$$

The above makes use of the fact that the weight $w_{i^*}^t$ of the optimal expert will be at least $(\frac{1}{2})^{\mathbf{OPT}}$ since the algorithm halves the weights of the experts when a mistake is made for $\epsilon = 0.5$.

If the algorithm makes mistake at time t, we have

$$W^{t+1} = \frac{1}{2}\sum_{i \in E^t} w_i^t + \sum_{i \notin E^t} w_i^t \leq \frac{3}{4}W^t. \tag{3}$$

The last inequality follows from $\sum_{i \in E^t} w_i^t \geq W^t/2$. Thus, denoting M^t as number of mistakes made by the algorithm till time t and M as the total number of mistakes by the algorithm, we obtain

$$W^t \leq \left(\frac{3}{4}\right)^{M_t} W^1 \tag{4}$$

$$= \left(\frac{3}{4}\right)^{M_t} n. \tag{5}$$

From (3) and (5), obtain

$$M_t \leq \frac{\mathbf{OPT} + \log_2 n}{\log_2 \frac{4}{3}} \quad \forall t \tag{6}$$

$$\implies M \leq \frac{\mathbf{OPT} + \log_2 n}{\log_2 \frac{4}{3}} \tag{7}$$

□

The weighted majority algorithm has a shortcoming. In case the weights of the functions which predict 0 and 1 are nearly equal, even then the algorithm confidently goes ahead with one of the labels as a prediction. This can be exploited by the adversary. If we consider a randomized algorithm, we can achieve a slightly better mistake bound. Before we dive into this algorithm, let us define some terms to work at a higher level of generality.

Randomized Weighted Majority

There are n experts/options and at time t, each expert i incurs a 'cost' $c_t(i)$, where $c_t : [n] \to [0, 1]$. Let i_t denote the expert chosen by

the algorithm at t, so the overall cost of the algorithm till time T is $\sum_{t=1}^{T} c^t(i^t)$. The optimal cost is given by $\mathbf{OPT} = \min_{i\in[n]} \sum_{t=1}^{T} c^t(i)$, and let i^* denote the optimal expert.

Algorithm 2: Randomized Weighted Majority (ϵ)

procedure RWM(ϵ)
  $w_1^1, \ldots, w_n^1 \leftarrow 1$ initial weights ▷ w_i^t is expert i weight @ time t
  for $t = 1, \ldots$ **do**
    $W^t \leftarrow \sum_{i=1}^{n} w_i^t$
    $p_i^t \leftarrow w_i^t / W^t$
    Sample $i^t \sim p^t$
    Predict i^t and observe $c^t(\cdot)$
    $w_i^{t+1} = w_i^t (1-\epsilon)^{c^t(i)} \quad \forall i$
  end for
end procedure

Theorem 3. *Under RWM(ϵ), for $T > 0$,*

$$\mathbb{E}\left[\sum_{i=1}^{T} c^t(i^t)\right] \leq \frac{1}{1-\epsilon}\mathbb{E}\left[\min_{i\in[n]} \sum_{t=1}^{T} c^t(i)\right] + \frac{\ln n}{\epsilon}$$

Proof.

$$W^{t+1} \geq w_{i^*}^{t+1} = (1-\epsilon)^{\sum_{i=1}^{t} c^t(i^*)}$$

It is important to note that $W^{t+1}, w_{i^*}^{t+1}, i^*, c^t(\cdot)$ are all random variables so the above inequality is technically an almost sure statement. Taking *log* and then expectation over all the randomness,

$$\mathbb{E}[\ln W^{t+1}] \geq \ln(1-\epsilon)\mathbb{E}[\sum_{i=1}^{t} c^t(i^*)]. \tag{8}$$

Next, let us upper bound $\mathbb{E}[\ln W^{t+1}]$.

$$W^{t+1} = \sum_{i\in[n]} (1-\epsilon)^{c^t(i)} w_i^t \tag{9}$$

$$\leq W^t - \epsilon \sum_{i\in[n]} c^t(i) w_i^t, \tag{10}$$

where we used the identity $(1-\epsilon)^c \leq 1 - c\epsilon$ for $c \in [0,1]$ in (9). Let H_t refer to all the history and observations made by the algorithm before loop at time t. Observe that $\sum_{i\in[n]} c^t(i) w_i^t = W^t \sum_{i\in[n]} p_i^t c^t(i) = W^t \mathbb{E}[c^t(i^t)|H_t]$. Again, it is important to note that the above is an almost sure statement involving random variables. Using it in (10) and the fact that $1 - x \leq e^{-x}\ \forall x \geq 0$, get

$$W^{t+1} \leq W^t(1 - \epsilon\mathbb{E}[c^t(i^t)|H_t]) \tag{11}$$

$$\leq W^t \exp(-\epsilon\mathbb{E}[c^t(i^t)|H_t]). \tag{12}$$

Recursing (12), we get $W^{t+1} \leq n \exp(-\epsilon \sum_{k=1}^{t} \mathbb{E}[c^k(i^k)|H_k])$. Taking *log* and then expectation, $\mathbb{E}[\ln W^{t+1}] \leq \ln n - \epsilon \sum_{k=1}^{t} \mathbb{E}\left[\mathbb{E}[c^k(i^k)|H_k]\right]$. By the tower property of conditional expectation, $\mathbb{E}\left[\mathbb{E}[c^k(i^k)|H_k]\right] = \mathbb{E}[c^k(i^k)]$. Combining the above with (8), we get

$$\mathbb{E}\left[\sum_{i=1}^{T} c^t(i^t)\right] \leq -\frac{\ln(1-\epsilon)}{\epsilon}\mathbb{E}[\sum_{i=1}^{t} c^t(i^*)] + \frac{\ln n}{\epsilon} \tag{13}$$

$$\leq \frac{1}{1-\epsilon}\mathbb{E}[\sum_{i=1}^{t} c^t(i^*)] + \frac{\ln n}{\epsilon} \tag{14}$$

□

An immediate corollary of the theorem is as follows.

Corollary 1. *Set* $\epsilon = \min\left\{\frac{1}{2}, \sqrt{\frac{\ln n}{2\mathbf{OPT}}}\right\}$ *and get regret* $O(\sqrt{\mathbf{OPT} \ln n})$.

One problem with the above result is that setting ϵ as described is impractical since we do not know the value of **OPT**, so we can instead upper bound it by T.

Corollary 2. *Set* $\epsilon = \min\left\{\frac{1}{2}, \sqrt{\frac{\ln n}{2T}}\right\}$ *and get regret* $O(\sqrt{T \ln n})$.

What if we do not know T either before starting the algorithm? One smart way of dealing with it is to assume the horizon is some value T_0 and set appropriate ϵ. If during the run of the algorithm, we observe time exceeding T_0, then from that point on, act as if the horizon is $2T_0$ and change ϵ, and so on. It can be shown that for unknown horizon length T, this modified algorithm still gets regret $O(\sqrt{T \ln n})$.

Lecture 3

Last lecture, we talked about the WeightedMajority, and observed that randomness improved the performance, i.e., Randomized-WeightedMajority. We defined the notion of *regret* as below:

$$\mathsf{Regret}_T(\mathcal{A}, c) = \mathbb{E}\left[\underbrace{\sum_{t=1}^{T} c^t(i^t)}_{\text{alg. output}} - \underbrace{\min_{i\in[n]} \sum_{t=1}^{T} c^t(i)}_{\text{opt}}\right],$$

$$\mathsf{Regret}_T(\mathcal{A}) = \inf_c \mathsf{Regret}_T(\mathcal{A}, c).$$

We proved that

$$\mathsf{Regret}_T(\textsc{RandomizedWeightedMajority}) \le 2\sqrt{2T\log n}.$$

Today we will discuss the notion of online learnability, and ask what classes of problems are online learnable. We will derive an upper bound and lower bound on online learnability, and describe an algorithm for online learning called SOA.

We noted that the regret of randomized weighted majority does not grow quite as fast as T does. More formally, the average regret

$$\lim_{T\to\infty} \frac{\mathsf{Regret}_T(\textsc{RandomizedWeightedMajority})}{T} = 0.$$

This is what we want for online learnability.

Definition 2 (Online Learning Algorithm)**.** Algorithm $\mathcal{A}$ learns $\mathcal{H}$ in the *online adversarial model* if

$$\lim_{T\to\infty} \frac{\mathsf{Regret}_T(\mathcal{A})}{T} = 0,$$

i.e., there exists a function $T_{\mathcal{H}} : (0,1) \to \mathbb{N}$ such that for any $\varepsilon \in (0,1)$ and $T \ge T_{\mathcal{H}}(\varepsilon)$,

$$\frac{\mathsf{Regret}_T(\mathcal{A})}{T} \le \varepsilon.$$

This function $T_{\mathcal{H}}$ is analogous to *sample complexity*, i.e., how many samples we need to achieve a low enough regret. It is ideal to find

$T_{\mathcal{H}}$ which gives us a *non-asymptotic* sample complexity. Alternatively one can show that the limit is 0 without finding such a $T_{\mathcal{H}}$ — this is called an *asymptotic* result — but it is not ideal. Also, such $\mathcal{A}$ are called *no-regret learners* for $\mathcal{H}$.

Definition 3 (Online Learnable Hypotheses)**.** Class $\mathcal{H}$ is *online adversarial learnable* if there exists an algorithm $\mathcal{A}$ which learns $\mathcal{H}$ in the online adversarial model.

Example. The RandomizedWeightedMajority regret bound says that all finite hypothesis classes $\mathcal{H}$ of size n are online adversarial learnable, using the RandomizedWeightedMajority algorithm and

$$T_{\mathcal{H}}(\varepsilon) = 8\frac{\log n}{\varepsilon^2}$$

for some absolute constant c_1. Indeed, we have, for $T \geq T_{\mathcal{H}}(\varepsilon)$, that

$$\frac{\mathsf{Regret}_T(\textsc{RandomizedWeightedMajority})}{T} = \frac{2\sqrt{2\log n}}{\sqrt{T}} \leq \frac{2\sqrt{2\log n}}{\sqrt{T_{\mathcal{H}}(\varepsilon)}} \leq \varepsilon.$$

Here is the question we will try to answer soon. Every time we deal with a finite $\mathcal{H}$, we know the answer as above. But the above bounds are increasing logarithmically in n, and it is very natural to study $\mathcal{H}$ which are infinite, i.e., the binary threshold example from earlier. We want some characterization, well-defined for infinite hypothesis classes, that captures online learnability.

The first question is whether the size of $\mathcal{H}$ is the correct measure. The answer turns out to be no. The reasoning is the following. If we are given a set of functions, we can copy and perturb each function by a tiny amount, and get essentially the same set of functions and the same learnability property. More formally, we can get a lot of functions which will disagree on small parts of the domain.

Now that we're thinking of the domain $\mathcal{X}$, let's ask whether the size of the domain is the correct characterization. Is it possible to keep padding $\mathcal{X}$ without changing the difficulty of regret minimization? The answer is that it is of course possible by adding redundant coordinates.

So the problem is about the "richness" of $\mathcal{H}$ on $\mathcal{X}$.

Example (Lack of Online Learnability). When $\mathcal{H}$ and $\mathcal{X}$ are infinite, online learnability may not be possible. The idea is to use our familiar one-dimensional thresholds:

$$\mathcal{H} = \left\{h_a \colon [0,1] \to \{-1,1\} \text{ given by } h_a(x) \doteq 2 \cdot \mathbb{I}(x \geq a) - 1 \;\middle|\; a \in [0,1]\right\}.$$

Suppose we are given $x^1 = \frac{1}{2}$. Our algorithm says that $h(x^1) = +1$, but the adversary hands us $y^1 = -1$, giving us a mistake. The

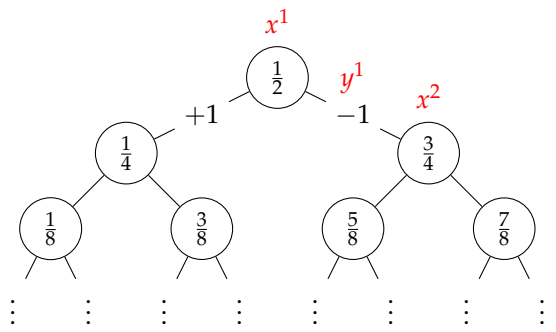


Figure 3: Online learning for thresholds on $[0, 1]$.

adversary's choice is then $x^2 = \frac{3}{4}$, and we construct an infinite binary tree in this fashion. See Figure 3.

More generally, as an adversary, our strategy is very simple; we pick an edge at random at each iteration, or equivalently, we pick a path at random in the infinite binary tree. Since we picked an answer at every node at random, the algorithm can do no better than random guessing, and after T iterations will make roughly $\frac{T}{2}$ mistakes. There will exist some $h \in \mathcal{H}$ which is consistent with the path we chose, and so the optimal $h \in \mathcal{H}$ will achieve loss 0. Thus the regret is $\frac{T}{2}$, which is not sublinear, and so this class is not online learnable.

There is an interesting criticism of this counterexample, where we don't really care about small enough differences between functions in $\mathcal{H}$ in practice — e.g., in our example, we only care about determining the threshold up to a given precision — but formally this model is still not online learnable. Even as we narrow down the threshold to a smaller interval, the adversary can exploit the fact that there is some uncertainty to cause large amounts of regret.

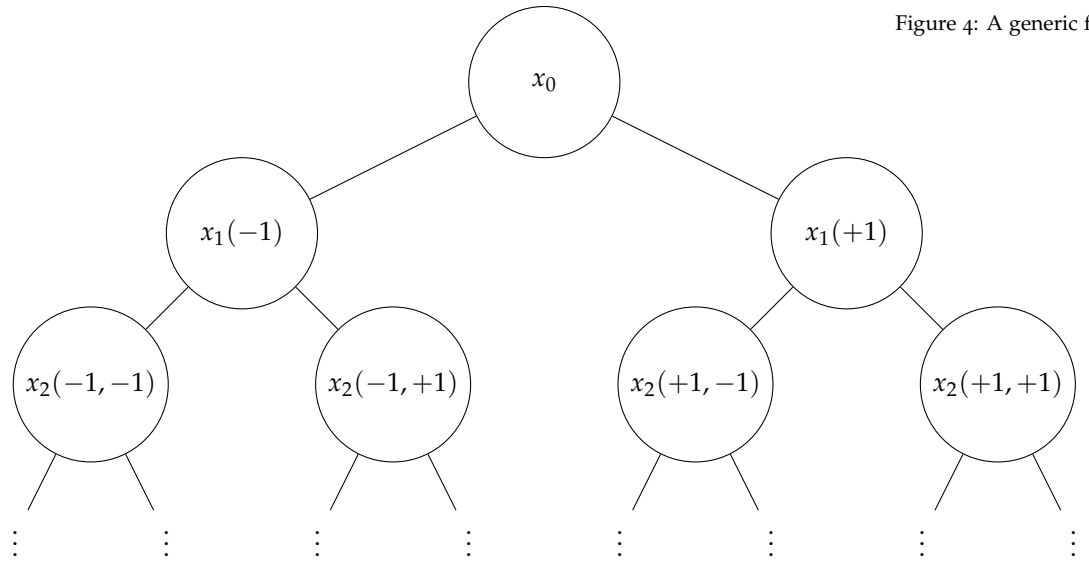
There are two ways to deal with this discrepancy. The first is to make everything stochastic instead of adversarial, in which case smaller regions of uncertainty are assigned smaller probabilities, so the probability that a mistake is made in this region goes to 0. This definition is too naive; producing a satisfying mechanism to weld together stochastic and adversarial behavior has been the topic of recent research.

If this tree for our problem is infinite, then the problem is not learnable. The relevant quantity in determining online learnability is the depth of this tree.

More formally, consider a full binary tree of depth d. The nodes of this tree are $x \in \mathcal{X}$. For every $y = (y_1, \ldots, y_d) \in \{+1, -1\}^d$, define $x_t(y)$ as the node in the tree at depth t if we start at the root and, at time $i \in [t]$, go left if $y_i = -1$ and right if $y_i = +1$. See Figure 4

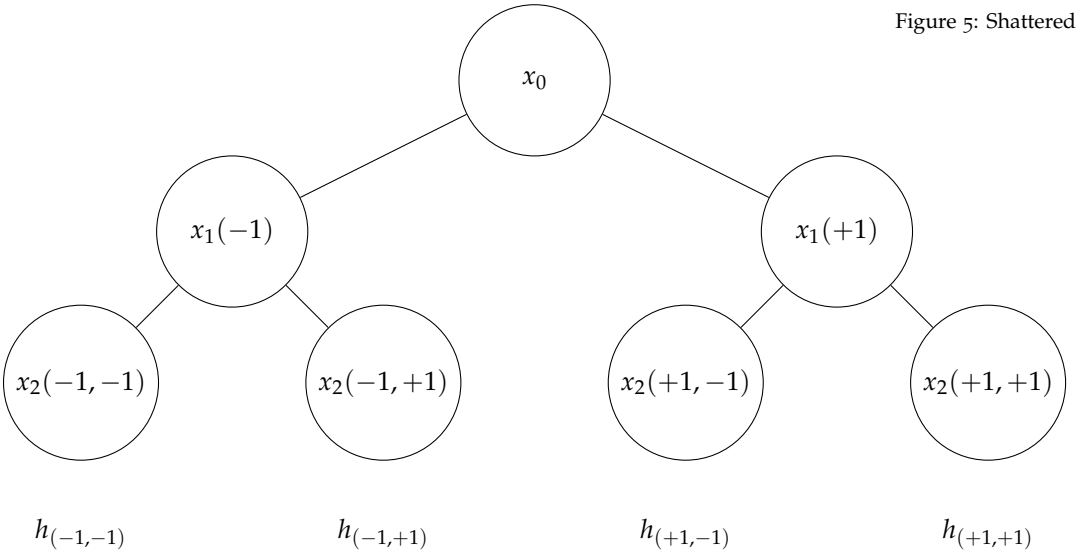
Such a tree is called a *shatter tree* if it has the following property: for all $y = (y_1, \ldots, y_d) \in \{-1, +1\}^d$ there exists $h \in \mathcal{H}$ such that

Figure 4: A generic full binary tree.



$h(x_i(y)) = y_i$ for all $i \in [d]$. See Figure 5.

Figure 5: Shattered tree.



As in the above figure, a shattered tree conceptually assigns a $h \in \mathcal{H}$ to every leaf, which predicts the response to all samples on the path from the root to the leaf correctly. Note that in this model we can send $d \to \infty$.

We formalize a general notion that captures the above intuition .

Definition 4 (Littlestone Dimension [Lit87])**.** The Littlestone dimension of class $\mathcal{H}$, denoted $\mathsf{LD}(\mathcal{H})$, is the maximal depth of any shatter tree for $\mathcal{H}$.

Now the answer to our online learnability question is that a hy-

pothesis class $\mathcal{H}$ is online learnable if and only if $\mathsf{LD}(\mathcal{H})$ is finite.

Many other quantities are formalized by Littlestone dimension; for example, differential privacy and game theoretic dynamics.

Theorem 4 (Regret Lower Bound)**.** *The regret of any algorithm $\mathcal{A}$ is lower bounded by*

$$\mathsf{Regret}_T(\mathcal{A}) \geq \Omega(\sqrt{T \cdot \min\{T, \mathsf{LD}(\mathcal{H})\}}).$$

We will not prove this, but the argument is similar to what we already proved.

Corollary 3. *If $\mathsf{LD}(\mathcal{H}) = \infty$ then $\mathcal{H}$ is not online adversarially learnable.*

Proof. The same game we played above in the linear thresholding case can prove linear regret in infinite-dimensional shatter trees. □

Theorem 5 (Regret Upper Bound [BDR21])**.** *There exists an algorithm $\mathcal{A}$ such that*

$$\mathsf{Regret}_T(\mathcal{A}) \leq O(\sqrt{T \cdot \min\{T, \mathsf{LD}(\mathcal{H})\}}).$$

While this is the product of recent research, the similar upper bound

$$\mathsf{Regret}_T(\mathcal{A}) \leq O(\sqrt{T \log T \cdot \min\{T, \mathsf{LD}(\mathcal{H})\}})$$

was known more than 10 years ago[1].

[1] Shai Ben-David, Dávid Pál, and Shai Shalev-Shwartz. Agnostic online learning. In *Proceedings of the Conference on Learning Theory (COLT)*, 2009

Theorem 6 (Regret Upper Bound via SOA [Lit87])**.** *In the mistake bound model, with the "promise" that there exists $h \in \mathcal{H}$ which is perfect, there is an algorithm called "Standard Optimal Algorithm (SOA)" that makes $\leq \mathsf{LD}(\mathcal{H})$ mistakes.*

Proof. We want to cook up an algorithm which implicitly or explicitly uses the Littlestone dimension. The idea is very similar to the halving algorithm. In the halving algorithm we kept track of the "credibility" of each hypothesis, and at each timestep took the majority of all hypotheses weighted by credibility, reweighting the credibility if a mistake is made.

The SOA will transplant this credibility to the Littlestone dimension world. At each mistake, we will remove the more credible, or complex part of $\mathcal{H}$ from consideration.

This is the halving algorithm, except instead of taking into account the credibility we use the Littlestone dimension.

It suffices to show that if we make a mistake at time t, then

$$\mathsf{LD}(\mathcal{H}_{t+1}) \leq \mathsf{LD}(\mathcal{H}_t) - 1.$$

What this is saying is that at each timestep t, if we make a mistake then we move to the smaller sub-tree corresponding to $\mathcal{H}_{t+1}$. Since

Algorithm 3: Standard Online Algorithm

$\mathcal{H}_1 \doteq \mathcal{H}$.
for times $t \in \{1, 2, \dots\}$ **do**
  Receive x^t
  Define $\mathcal{H}_t^{+1} \doteq \{h \in \mathcal{H}_t \colon h(x^t) = +1\}$
  Define $\mathcal{H}_t^{-1} \doteq \{h \in \mathcal{H}_t \colon h(x^t) = -1\}$
  Predict $\widehat{y}^t \doteq \arg\max_{y \in \{-1,1\}} \mathsf{LD}(\mathcal{H}_t^y)$
  Observe y^t
  Update $\mathcal{H}_{t+1} = \mathcal{H}_t^{y^t}$
end for

we always go to a smaller sub-tree at a mistake, we make at most $\mathsf{LD}(\mathcal{H})$ mistakes.

It remains to prove the above inequality. Suppose for the sake of contradiction that $\mathsf{LD}(\mathcal{H}_{t+1}) = \mathsf{LD}(\mathcal{H}_t)$. Since we chose the $\mathcal{H}_t^{\pm}$ with the highest Littlestone dimension, and we made a mistake, we must have

$$\mathsf{LD}(\mathcal{H}_{t+1}) = \mathsf{LD}(\mathcal{H}_t) = \mathsf{LD}(\mathcal{H}_t^{+1}) = \mathsf{LD}(\mathcal{H}_t^{-1}).$$

But if the trees corresponding to $\mathcal{H}_t^{+1}$ and $\mathcal{H}_t^{-1}$ are shattered and have depth d, the tree corresponding to $\mathcal{H}_t$, with root x^t, must be shattered by one of the functions in $\mathcal{H}_t^{y^t}$, and has depth $d + 1$. Thus we must have

$$\mathsf{LD}(\mathcal{H}_t^{+1}) = \mathsf{LD}(\mathcal{H}_t^{-1}) = d \implies \mathsf{LD}(\mathcal{H}_t) = d + 1.$$

This is a contradiction. □

Lecture 4

Today, we will introduce game theory, drawing a connection to the no-regret learning algorithms discussed in the past few lectures.

Game theory as a field was introduced by John von Neumann in 1928 as a means to reason about how strategic agents interact with each other. Most courses on this topic consider strategic agents to be competitive or non-cooperative; however, there are many cases where we are interested in the behavior of systems with cooperative agents. Later on in the class, we will learn about Shapley values, which measure the incentive for coalitions to cooperate.

A game is defined by a set of K agents, each with (possible different) actions sets A_j. Each agent j selects an action from their action set $a_j \in A_j$. These actions can be selected sequentially or simultaneously. To measure the outcome of the actions chosen, each agent possesses their own loss function $\ell_j(a_j, a_{-j})$, where a_{-j} represents the set of actions chosen by all other agents. Each agent seeks to strategically select their action to minimize their own loss function.

Example (Chicken). Consider two drivers approaching each other at high speed. Each driver has the choice to swerve or drive straight ahead. In the best outcome, a driver would continue straight ahead while causing their opponent to swerve, so that their opponents would be a "chicken". If both drivers choose to swerve, both are equally cowardly. However, if both refuse to swerve, a horrible collision ensues.

We can model the set of possible outcomes and losses the following matrix:

	Swerve	Drive Ahead
Swerve	0, 0	-1, +1
Drive Ahead	+1, -1	-10, -10

For two player games, we use the notation that in cell i, j, we list the losses as $\ell_r(a_i, a_j), \ell_c(a_i, a_j)$, where r represents the row player and c represents the column player.

Now, consider the case where the row player rips off their steering wheel, committing to driving straight ahead. The best response for

the column player would be to swerve. This would be an instance of a sequential game. We will study sequential games such as Stackelberg games later on in the semester.

Example (Hide/Choose). One person (called the hider) has the option to either place one coin in their left hand or two coins in their right hand, and then places both hands behind their back. Another person (called the chooser) then selects left or right, and receives the coins (if any) from that respective hand of the hider.

With the chooser as the row player and the hider as the column player, this game can be represented by the following matrix, where we use utilities instead of losses:

	Left	Right
Left	1, -1	0, 0
Right	0, 0	2, -2

Rather than committing to a fixed action (or strategy), each player may find it beneficial to instead select a distribution over their actions. The best distribution for the hider is to hide it in their left hand with probability $\frac{2}{3}$ and in their right with probability $\frac{1}{3}$; conversely, the choser finds it optimal to select left with probability $\frac{1}{3}$ and right with probability $\frac{2}{3}$.

With these random strategies, both players choose left with probability $\frac{4}{9}$ and both players choose right with probability $\frac{1}{9}$. This means that the expected reward under these (optimal) random strategies is $\frac{2}{3}$, while the expected loss for the hider is $\frac{2}{3}$.

Since the utilities sum to zero in each cell, this is an instance of a zero-sum game. Zero-sum games have a value, which is the expected return (or loss) while playing in a game. In the example above, the value of the game is $\frac{2}{3}$.

Example (Rock Paper Scissors). We can model the popular game of rock paper scissors as a zero-sum game between two players. It induces the following game matrix:

	Rock	Paper	Scissors
Rock	0, 0	-1, 1	1, -1
Paper	1, -1	0, 0	-1, 1
Scissors	-1, 1	1, -1	0, 0

If one player was forced to reveal their strategy before the other player selected theirs, committing to a single strategy (such as always choosing rock) would make the game extremely easy for the other player. The best thing for the first player to do would be to commit to a strategy of selecting each action with probability $\frac{1}{3}$. If so, the best thing for the second player would be to commit to the same strategy. This leads to a value of zero for the game.

Intuitively, it would seem that when the first player is forced to commit to a strategy, the second player would have an advantage. However, in 1928, Von Neumann proved that in two player zero-sum games with finite action spaces, there is no advantage.

Before stating this formally, we first introduce the following procedure:

- The row player first selects a distribution P over their action set
- The column player then selects their own distribution Q that minimizes their loss
- Sample $i \sim P$ and $j \sim Q$
- Calculate the expected loss of the row player by, where L coalesces $\ell_r(i, j)$ into a matrix

$$\mathcal{L}(P, Q) = \sum_i \sum_j P_i Q_j \ell_r(a_i, a_j) = P^\intercal L Q$$

When the row player is forced to commit first, they will try to minimize their expected loss; when the column player responds, they try to maximize the loss of the row player, thereby increasing their utility. We can represent this by:

$$\max_{Q \in \Delta(A_c)} \min_{P \in \Delta(A_r)} \mathcal{L}(P, Q)$$

On the other hand, we can represent the column player committing first as:

$$\min_{P \in \Delta(A_r)} \max_{Q \in \Delta(A_c)} \mathcal{L}(P, Q)$$

This leads us to our primary result of the lecture.

Theorem 7 (The Minimax Theorem)**.** *Every two player, zero-sum game with finite action spaces satisfies:*

$$\min_{P \in \Delta(A_r)} \max_{Q \in \Delta(A_c)} \mathcal{L}(P, Q) = \max_{Q \in \Delta(A_c)} \min_{P \in \Delta(A_r)} \mathcal{L}(P, Q)$$

Proof. We will prove this by showing the inequality holds in both directions. Showing $\geq$ is simple, as it follows from first principles. To show the more difficult direction $\leq$, we will leverage the no-regret online learning strategies discussed over the past few lectures.

First, to show $\geq$, consider the relationship below, which we know to be true by the definition of a maximum.

$$\max_{Q \in \Delta(A_c)} \mathcal{L}(P, Q) \geq \mathcal{L}(P, Q) \quad \forall P, Q \tag{15}$$

Then, it must also be true that

$$\min_{P \in \Delta(A_r)} \max_{Q \in \Delta(A_c)} \mathcal{L}(P, Q) \geq \min_{P \in \Delta(A_r)} \mathcal{L}(P, Q) \quad \forall Q \tag{16}$$

Note that the left hand side is greater than the right hand side for any choice of Q. Then, it must be true that

$$\min_{P \in \Delta(A_r)} \max_{Q \in \Delta(A_c)} \mathcal{L}(P, Q) \geq \max_{Q \in \Delta(A_c)} \min_{P \in \Delta(A_r)} \mathcal{L}(P, Q) \tag{17}$$

Now, it remains to show $\leq$. Over the years, this has been shown through many different methods, but many rely on the use of duality. Instead, we will model this as a repeated game, where each player is free to choose P_t and Q_t at each iteration, and adjust their strategies according to the outcomes of past iterations.

Formally, consider a repeated game where the row player chooses P_t according to a no-regret algorithm that uses the history of distributions played by the column player $Q_1, \ldots, Q_{t-1}$. At each time t, the column player plays Q_t, a best response to P_t (i.e. minimizes their expected loss). We can define the time-averaged distribution of strategies for the row and column player as $\bar{P} = \frac{1}{T} \sum_{t=1}^{T} P_t$ and $\bar{Q} = \frac{1}{T} \sum_{t=1}^{T} Q_t$ respectively.

Remember that we can express $\mathcal{L}(P, Q)$ as $P^\intercal L Q$, where $[L]_{i,j} = \ell_r(a_i, a_j)$. We will be working with this matrix expression for the remainder of the proof.

Begin by considering the following expression. This is true by the definition of minimum, since $\bar{P}$ is a valid distribution in $\Delta(A_r)$.

$$\min_{P \in \Delta(A_r)} \max_{Q \in \Delta(A_c)} P^\intercal L Q \leq \max_{Q \in \Delta(A_c)} \bar{P}^\intercal L Q \quad \forall \bar{P} \tag{18}$$

Now, we can substitute $\bar{P}$ with its definition

$$\min_{P \in \Delta(A_r)} \max_{Q \in \Delta(A_c)} P^\intercal L Q \leq \max_{Q \in \Delta(A_c)} \frac{1}{T} \sum_{t=1}^{T} P_t^\intercal L Q \quad \forall P_t \tag{19}$$

The maximum of a sum is less than or equal to the sum of maximums. This is a consequence of Jensen's inequality.

$$\min_{P \in \Delta(A_r)} \max_{Q \in \Delta(A_c)} P^\intercal L Q \leq \max_{Q \in \Delta(A_c)} \frac{1}{T} \sum_{t=1}^{T} P_t^\intercal L Q \leq \frac{1}{T} \sum_{t=1}^{T} \max_{Q \in \Delta(A_c)} P_t^\intercal L Q \quad \forall P_t \tag{20}$$

When P_t is selected, the column player selects Q_t as their best response. We can replace the maximum in the summation with this.

$$\min_{P \in \Delta(A_r)} \max_{Q \in \Delta(A_c)} P^\intercal L Q \leq \frac{1}{T} \sum_{t=1}^{T} P_t^\intercal L Q_t \quad \forall Q_t \tag{21}$$

Recall from lecture 2 how we define average regret. We expressed average regret as

$$\frac{\text{Regret}}{T} = \underbrace{\frac{1}{T} \sum_{t=1}^{T} P_t^\intercal L Q_t}_{\text{output of the algorithm}} - \underbrace{\min_{P \in \Delta(A_r)} P^\intercal L \bar{Q}}_{\text{optimal } P \text{ in hindsight}} \quad \forall \bar{Q}, Q_t \tag{22}$$

Rearranging, we find

$$\frac{1}{T}\sum_{t=1}^{T} P_t^\intercal L Q_t = \min_{P\in\Delta(A_r)} P^\intercal L\bar{Q} + \frac{\text{Regret}}{T} \quad \forall \bar{Q}, Q_t \tag{23}$$

Thus, we can replace the right hand side of (21) with this expression

$$\min_{P\in\Delta(A_r)} \max_{Q\in\Delta(A_c)} P^\intercal L Q \leq \min_{P\in\Delta(A_r)} P^\intercal L\bar{Q} + \frac{\text{Regret}}{T} \quad \forall \bar{Q} \tag{24}$$

By the definition of maximum, we can upper bound the right hand side by

$$\min_{P\in\Delta(A_r)} \max_{Q\in\Delta(A_c)} P^\intercal L Q \leq \max_{Q\in\Delta(A_c)} \min_{P\in\Delta(A_r)} P^\intercal L Q + \frac{\text{Regret}}{T} \tag{25}$$

Since we are playing a no-regret algorithm, for any ϵ, we can find a T such that $\frac{\text{Regret}}{T} < \epsilon$. We find that as T approaches infinity, $\frac{\text{Regret}}{T}$ converges to 0. Thus, we are justified in stating

$$\min_{P\in\Delta(A_r)} \max_{Q\in\Delta(A_c)} P^\intercal L Q \leq \max_{Q\in\Delta(A_c)} \min_{P\in\Delta(A_r)} P^\intercal L Q \tag{26}$$

This completes our proof of the Minimax Theorem. □

Lecture 5

Up to now we have gone over online learning with a no regret algorithm and last lecture we talked about the minmax equilibrium. In this lecture we will look beyond zero-sum games and expand on notions of equilibria, online learning, and decision making.

Let's start with some basic notation.(also found in the preliminaries section). In a cost minimization game, we define:

- Finite number of players: $[k] = \{1, 2, 3, ..., k\}$
- Set of (pure) actions/strategies: S_i
- Cost function for $i \in [k] : C_i : \{s_1 \times s_2 \times ... \times s_k\} \to [0, 1]$ (Note that we mostly care about boundedness of the output space)
- Strategy profile: $(s_1, ..., s_k)$, where $s_i \in S_i \forall i$
- Strategy of i s_i and strategies of every one except i: s_{-i}
- Mixed strategy: distribution σ_i over S_i
- Mixed strategy profile: $(\sigma_1, ..., \sigma_k)$

Pure Nash Equilibrium (PNE): A pure strategy profile $(s_1, ..., s_k)$ is a Pure Nash Equilibrium if:

$$\forall i \in [k] \quad C_i(s_1, ..., s_k) \leq C_i(s_i', s_{-i}) \quad \forall s_i' \in S_i$$

In other words, we call a PNE a set of pure strategies, which no agent has an incentive to deviate from in order to decrease their cost.

Mixed Nash Equilibrium (MNE): A mixed strategy profile $(\sigma_1, ..., \sigma_k)$ is a Mixed Nash Equilibrium if for $\sigma = \sigma_1 \times \sigma_2 \times \cdots \times \sigma_k$:

$$\forall i \in [k] \quad \mathbb{E}_{\forall i s_i \sim \sigma_i}[C_i(s)] \leq \mathbb{E}_{\forall i s_{-i} \sim \sigma_{-i}}[C_i(s_i', s_{-i})] \qquad \forall s_i' \in S_i$$

In other words, given everyone's mixed strategies, no agent can deviate from their own to decrease their expected cost.

Here it is important to note that for a zero-sum game, a minmax equilibrium is equivalent with a Nash equilibrium. Notice that by the

definition of a minmax equilibrium.

$$\min_{\sigma_1} \max_{\sigma_2} \mathbb{E}_{s_1 \sim \sigma_1, s_2 \sim \sigma_2}[C(s_1, s_2)] = \max_{\sigma_2} \min_{\sigma_1} \mathbb{E}_{s_1 \sim \sigma_1, s_2 \sim \sigma_2}[C(s_1, s_2)] \tag{27}$$

Conditioning on player 1 not changing strategy, player 2 does not have incentive to deviate and vice versa.

Let's think about an example that we have seen before: rock paper scissors. In this game, we have established that the optimal strategy would be a uniform distribution over the three possible actions for each player, in contrast to a pure strategy of picking the same action every time . This prompts us to think that Pure Nash equilibria might not always exist. Let's formalize our intuition through the following theorem.

Theorem 8. *Any game with finite strategy sets has a Mixed Nash equilibrium (but not necessarily a Pure Nash equilibrium).*

Before we delve into the proof of above theorem, we will first need to use the following:

Theorem 9 (Brouwer's Fixed Point Theorem)**.** *For any compact and convex domain X and a continuous function $f : X \to X$,*

$$\exists x_0 \in X \quad x_0 = f(x_0).$$

Note that a compact set in $\mathbb{R}^d$ is a closed and bounded set (the abstract definition is more general but we will not focus on this). In particular, it includes all limiting values of its points (e.g. $(0, 1)$ is not compact but $[0, 1]$ is compact). Let's see an example of this. Take $f(x) = \frac{x+1}{2}$ on $(0, 1)$. This fullfils two of the three criteria: continuous function on a convex set. But the domain is not compact; therefore, there is no fixed point. Now, that we have seen Brouwer's fixed Point Theorem let's consider a first attempt toward proving Theorem 8.

Proof. Consider X to be the set of all mixed strategy profiles $(\sigma_1, ..., \sigma_k)$, where $\sigma_i \in \Delta(s_i)$. Then let $f(\sigma) = (\phi_1(\sigma), ..., \phi_k(\sigma))$, where we define:

$$\phi_i = \arg\min_{a_i} \mathbb{E}_{s_{-i} \sim \sigma_{-i}}[C_i(a_i, s_{-i})]$$

We consider ϕ_i to be the best response of player i to the strategies of all other players.

Notice, however, that Brower's theorem might not hold here as the arg min introduces discontinuity. To ensure that the function is continuous, we want to "move" from our current strategy toward the best-response action in "tiny steps". Therefore, we introduce a

regularization term that penalizes the distance between our current and new strategy. In particular, we define:

$$\phi_i(\sigma) = \arg\min_{\rho_i} C(\rho_i, s_{-i}) + \|\rho_i - \sigma_i\|_2^2$$

Notice that the function that is being minimized above is strongly convex, so it has a unique minimizer $\phi_i(\sigma)$. Moreover, ϕ_i is also continuous. Notice also that the domain X is convex and compact. Therefore, using Brouwer's theorem we can say that there is σ^* such that

$$f(\sigma^*) = \sigma^* \quad \text{or equivalently} \quad \forall\, i \quad \phi_i(\sigma^*) = \sigma_i^*$$

Now, we claim that $\sigma^* = (\sigma_1^*, \cdots, \sigma_k^*)$ is a Mixed Nash equilibrium.

Assume for the sake of contradiction that it is not. Then:

$$\exists\, i \in [k] \quad \exists\, \rho_i \quad \text{s.t.} \quad C_i(\sigma^*) - C_i(\sigma_{-i}^*, \rho_i) = \delta > 0$$

For some $\alpha \in [0, 1]$:

$$\begin{aligned} \hat{\rho}_i &= (1-a)\sigma_i^* + a\rho_i \\ &= \sigma_i^* + \alpha(\rho_i - \sigma_i^*) \\ \implies \hat{\rho}_i - \sigma_i^* &= \alpha(\rho_i - \sigma_i^*) \end{aligned}$$

Now we claim that:

$$C_i(\sigma^*) > C_i(\sigma_{-i}^*, \hat{\rho}_i) + \|\hat{\rho}_i - \sigma_i^*\|_2^2$$

We have that:

$$\begin{aligned} C_i(\sigma_{-i}^*, \hat{\rho}_i) + \|\hat{\rho}_i - \sigma_i^*\|_2^2 &= C_i(\sigma^*) + (\hat{\rho}_i - \sigma_i^*) \cdot C_i(\sigma_i^*, \cdot) + \alpha^2 \|\rho_i - \sigma_i^*\|_2^2 \\ &= C_i(\sigma^*) + \alpha(\rho_i - \sigma_i^*) \cdot C_i(\sigma_i^*, \cdot) + \alpha^2 \|\rho_i - \sigma_i^*\|_2^2 \\ &= C_i(\sigma^*) \underbrace{-\alpha\delta + \alpha^2 \|\rho_i - \sigma_i^*\|_2^2}_{<0} \qquad \text{(by definition of Nash)} \\ &< C_i(\sigma^*) \end{aligned}$$

□

Lecture 6

In the last lecture, we showed that mixed Nash equilibrium (MNE) exists in any finite game by applying Brouwer fixed-point theorem. Recall that a mixed Nash equilibrium is a product distribution σ such that

$$\mathbb{E}_{s\sim\sigma}[C_i(\sigma)] \leq \mathbb{E}_{s_{-i}\sim\sigma_{-i}}[C_i(s_i', s_{-i})],\ \forall i \in [K],\ s_i' \in S_i.$$

In today's lecture, we will introduce two different notions of equilibrium, one different notion of regret, and study two learning dynamics that converge to these equilibriums.

Correlated Equilibrium

Definition 5 (Correlated Equilibrium (CE))**.** A correlated equilibrium σ is a joint distribution over $\prod_{i=1}^{n} S_i$ such that

$$\mathbb{E}_{s\sim\sigma}[C_i(\sigma)|s_i] \leq \mathbb{E}_{s_{-i}\sim\sigma_{-i}}[C_i(s_i', s_{-i})|s_i],\ \forall i \in [K],\ s_i' \in S_i. \tag{28}$$

Remark. A few remarks are in order:

- To understand correlated equilibria, one may imagine a mediater who draws actions from σ and gives them as suggestions to each agent. No agent, looking at the action suggested by the mediator, can increase its utility by unilaterally switching to other strategies.
- Most useful equilibia in the real world are correlated equilibria. For example, the traffic light presents a correlated equilibrium where drivers maximize the utility by following the suggestion given by the traffic lights. We will illustrate this in the next Example.

Example (Traffic lights). Consider two-player general-sum game given by the following loss matrix:

	Stop	Go
Stop	1, 1	1, 0
Go	0, 1	5, 5

It is not hard to see that (Stop, Go) and (Go, Stop) are two pure Nash Equilibria.

To find a mixed Nash equilibrium, we set player 1's probability of playing Stop such that player 2 is indifferent to the choice of Stop or Go. More precisely, let p denote the player 1's probability of playing Stop, we notice that player 2's loss of playing Stop is given by 1, and player 2's loss of playing Stop is given by $5(1-p)$. Letting $1 = 5(1-p)$ yields $p = \frac{4}{5}$. Therefore we find a MNE $(\underbrace{4/5}_{\text{Stop}}, \underbrace{1/5}_{\text{Go}}) \times (\underbrace{4/5}_{\text{Stop}}, \underbrace{1/5}_{\text{Go}})$.

However, letting the drivers have $\frac{4}{5}$ probability of stopping at a crossing is both inefficient (wasting $4/5$ of the time) and dangerous (having $1/25$ probability of clashing). Indeed, this MNE gives an expected loss of $8/5$ for both players. One may find that $\frac{1}{2}(\text{Stop}, \text{Go}) + \frac{1}{2}(\text{Go}, \text{Stop})$ is a correlated equilibrium that gives an expected loss of $1/2$ for both players. Therefore, this correlated equilibrium achieves more social welfare.

It is not hard to verify that $\frac{1}{3}(\text{Stop}, \text{Go}) + \frac{1}{3}(\text{Go}, \text{Stop}) + \frac{1}{3}(\text{Stop}, \text{Stop})$ is also a correlated equilibrium.

We have the following equivalent definition of CE. The proof of equivalence is left as an exercise to the reader.

Definition 6 (Equivalent Definition of Correlated Equilibrium)**.** A correlated equilibrium σ is a joint distribution over $\prod_{i=1}^{n} S_i$ such that for any $i \in [K]$ and any *swap function* $\delta : S_i \to S_i$,

$$\mathbb{E}_{s\sim\sigma}[C_i(\sigma)] \leq \mathbb{E}_{s\sim\sigma}[C_i(\delta(s_i), s_{-i})]. \tag{29}$$

Here, the *swap function* δ is a mapping from actions to actions, possibly not subjective.

Coarse Correlated Equilibrium

Definition 7 (Coarse Correlated Equilibrium (CCE))**.** A coarse correlated equilibrium σ is a joint distribution over $\prod_{i=1}^{n} S_i$ such that

$$\mathbb{E}_{s\sim\sigma}[C_i(\sigma)] \leq \mathbb{E}_{s\sim\sigma}[C_i(s_i', s_{-i})],\ \forall i \in [K],\ s_i' \in S_i. \tag{30}$$

Figure 6 displays the relation between MNE, CE, and CCE. Since the definition of CE allows σ to be product measure, it is not hard to see that any NE is a CE. If σ is a CE, using the tower property of conditional expectation and applying Eq. (28) for all conditional expectation on s_i confirms that Eq. (30) also holds for σ. It follows that any CE is also a CCE.

The next example shows that CCE sometimes gives counter-intuitive equilibria.

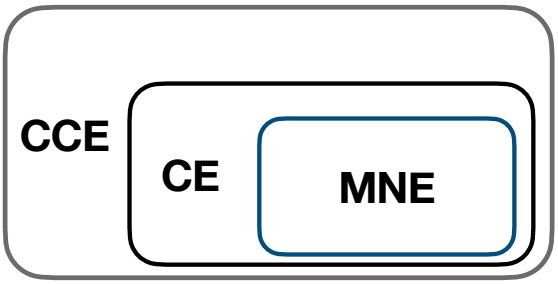


Figure 6: Venn-diagram visualization of different notions of equilibria we've seen so far. All inclusion relations shown in the figure are strict.

Example (Counter-intuitive CCE). Consider two-player general-sum game given by the following loss matrix where $\epsilon \in (0, 1)$ is sufficiently small

	A	B	C
A	-1, -1	1, 1	0, 0
B	1, 1	-1, -1	0, 0
C	0, 0	0, 0	$1+\epsilon$, $1+\epsilon$

Since action C is strictly dominated, no CE would be supported on C. However, we have the following CCE that is supported on C:

$$\sigma = \frac{1}{3}(\mathrm{A}, \mathrm{A}) + \frac{1}{3}(\mathrm{B}, \mathrm{B}) + \frac{1}{3}(\mathrm{C}, \mathrm{C}).$$

To see that it is indeed a CCE, we notice that for any $i \in [3]$

$$\mathbb{E}_\sigma[C_i(\sigma)] = \frac{\epsilon - 1}{3}$$
$$\mathbb{E}_{s_{-i} \sim \sigma_{-i}}[C_i(s_i', s_{-i})] = 0 > \mathbb{E}_\sigma[C_i(\sigma)], \ \forall\, s_i' = \text{ A or B.}$$

No-regret Dynamics

We consider the following dynamics associated to a general-sum game where every agent follows a no-regret algorithm and receives the cost as the expected value of the cost function under other agents' (mixed) strategies.

Algorithm 4: No-regret Dynamics

for $t \in \{1, 2, \ldots, T\}$ **do**

$\forall\ i \in [n]$, player i simultaneously and independently choose strategy p_i^t using a no-regret algorithm.

$\forall\ i \in [n]$, player i observes the cost vector c_i^t given by

$$c_i^t(s_i) = \mathbb{E}_{s_{-i} \sim p_{-i}^t}[C_i(s_i, s_{-i})]$$

end for

The main result in this section is that a no-regret dynamic converges to the coarse correlated equilibrium of the corresponding general-sum game.

Theorem 10 (No-regret dynamics converge to CCE)**.** *Consider the no-regret dynamics given above, with each player incurring regret $T \cdot \epsilon$. Then σ defined by*

$$\sigma = \frac{1}{T} \sum_{t=1}^{T} \prod_{j=1}^{n} p_j^t$$

is an ϵ-approximate CCE, i.e.,

$$\underbrace{\mathbb{E}_{s \sim \sigma}[C_i(\sigma)]}_{(1)} \leq \underbrace{\mathbb{E}_{s_{-i} \sim \sigma_{-i}}[C_i(s_i', s_{-i})]}_{(2)} + \epsilon,\ \forall i \in [K],\ s_i' \in S_i.$$

Proof. We use the linearity of expectation and the definition of c_i^t,

$$\begin{aligned}(1) = \mathbb{E}_{s \sim \sigma}[C_i(\sigma)] &= \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}_{s^t \sim \sigma^t}[C_i(s^t)] \\ &= \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}_{s_i^t \sim p_i^t}[c_i^t(s_i^t)]\end{aligned}$$

$$\begin{aligned}(2) = \mathbb{E}_{s \sim \sigma}[C_i(s_i', s_{-i})] &= \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}_{s^t \sim \sigma^t}[C_i(s_i', s_{-i})] \\ &= \frac{1}{T} \sum_{t=1}^{T} c_i^t(s_i') \\ &= \mathbb{E}_{s_{-i} \sim \sigma_{-i}}[C_i(s_i', s_{-i})]\end{aligned}$$

By the definition of no-regret algorithm:

$$\begin{aligned}\frac{1}{T}\ \text{Alg's cost} &\leq \frac{1}{T}\ \text{Cost of any fixed action} + \frac{1}{T}\ \text{Regret} \\ (1) &\leq (2) + \epsilon\end{aligned}$$

The above inequality holds for any player $i \in [K]$ and any pure strategy deviation $s_i' \in S_i$. □

No-swap-regret Dynamics

We have demonstrated that if all players employ a no-regret algorithm, their no-regret dynamic will converge to a CCE. Furthermore, if all players use a more powerful no-swap-regret algorithm, their dynamic will converge to a CE. We begin by defining the notion of swap-regret.

Definition 8 (Swap-Regret)**.** For a sequence of costs $c^1, c^2, ..., c^T$ and played strategy $s^1, s^2, ..., s^T$,

$$\text{Swap-Regret} = \sum_{t=1}^{T} c^t(s^t) - \min_{\delta:\mathcal{S}\to\mathcal{S}} \sum_{t=1}^{T} c^t(\delta(s^t))$$

Remark: The min is taken over all possible swap functions. By restricting the swap function δ to be constant function $\delta_i(s) = i$, we recover the definition of (External) Regret.

Definition 9 (no-swap-regret)**.** An algorithm is a no-swap-regret algorithm iff for any sequence $c^1, c^2, ..., c^T$,

$$\frac{\text{Swap-Regret}}{T} \to 0 \quad \text{as} \quad T \to \infty$$

We state the following theorem that no-swap-regret dynamics converge to CE, the proof is analogous to the proof for no-regret dynamics.

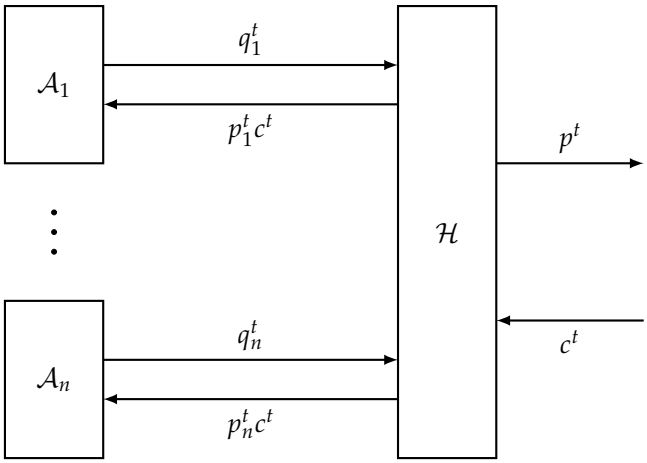
Theorem 11 (No-swap-regret dynamics converge to CE)**.** *Consider the no-swap-regret dynamics with each player incurring swap-regret $T \cdot \epsilon$. Then σ defined by*

$$\sigma = \frac{1}{T} \sum_{t=1}^{T} \prod_{j=1}^{n} p_j^t$$

is an ϵ-approximate CE, i.e.,

$$\mathbb{E}_{s\sim\sigma}[C_i(\sigma)] \leq \mathbb{E}_{s\sim\sigma}[C_i(\delta(s_i), s_{-i})] + \epsilon, \ \forall i \in [K], \forall \delta_i : S_i \to S_i.$$

Now, all that remains is to prove that there exists an algorithm that is no-swap-regret. The first no-swap-regret algorithms (and in fact something closely related called a no-internal-regret) algorithms and their relationship to CEs were in [FV97]. We provide the following blackbox reduction from any no-regret algorithms to no-swap-regret algorithms due to [BM05]. We also encourage readers to read Chap 4 of [NRTV07] for more details.



High level idea: Consider n copies of a no-regret algorithm, $\mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_n$, where n is the number of actions.

- $\mathcal{A}_i$ is responsible for one pure strategy $i \in [n]$
- q_i^t is the vector probability distribution over action $[n]$

  $q_i^t(j)$: probability of playing j as recommended by $\mathcal{A}_i$ at time t

- p^t: a clever way of combinning $q_1^t, q_2^t, ..., q_n^t$

  $p^t(j)$: probability of playing j at time t by algorithm $\mathcal{H}$

- c^t : cost vector experienced in real life deploying $\mathcal{H}$
- $p_i^t c^t$: scaling experienced by $\mathcal{A}_i$

$p^t = p^t Q$, Markov chain of algorithms' recommendations

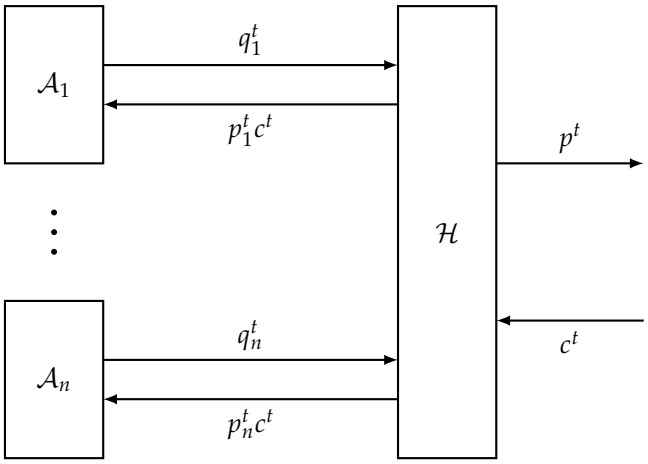
Lecture 7

In the last lecture, we learned how Coarse correlated equilibria (CCE) and Correlated equilibria (CE) emerge from the dynamics of algorithms with no-(external)-regret and no-SWAP-regret respectively.

Today's lecture explains how a no-SWAP-regret algorithm can be constructed using no-(external)-regret algorithms as black boxes. Then, we also discuss online algorithms for the partial information case, i.e when the cost of only the action picked is observed.

No-SWAP-regret algorithm

To recap, we considered the following black box in the last class.



The black box has n copies of a no-regret algorithm, with $\mathcal{A}_i$ corresponding to the i^{th} pure strategy. q_i^t is the probability distribution of actions suggested by algorithm $\mathcal{A}_i$ at time t and Q^t is a matrix with its i^{th} column as q_i^t. Algorithm $\mathcal{H}$ "cleverly" combines these q_i^ts into p^t and observes a cost vector c^t. Scaled cost vector $p_i^t c^t$ is then sent back to algorithm $\mathcal{A}_i$.

Theorem 12 (No-SWAP-regret algorithm [NRTV07])**.** *In the above setup, algorithm $\mathcal{H}$ has no-SWAP-regret if p^t is set as the stationary distribution of the Markov chain with probability distribution matrix Q^t, i.e p^t*

satisfies $p^t = Q^t p^t$.

Proof. We'll do a constructive proof to find the relation between p^t and q_i^ts.

The cost incurred by algorithm $\mathcal{H}$ is $\sum_{t=1}^{T} \mathbf{E}_{i\sim p^t}[c^t_{(i)}] = \sum_{t=1}^{T}\sum_{i=1}^{n} p^t_{(i)} c^t_{(i)}$. Here, subscript indices are indicated in parenthesis to indicate they are scalars. Further, algorithm $\mathcal{H}$'s competing benchmark is a swap function $\delta : [n] \to [n]$ with cost $\sum_{t=1}^{T}\sum_{i=1}^{n} p^t_{(i)} c^t_{(\delta(i))}$. Notice that to prove algorithm $\mathcal{H}$ has no-SWAP-regret, we need to show:

$$\sum_{t=1}^{T} \mathbf{E}_{i\sim p^t}[c^t_{(i)}] \leq \sum_{t=1}^{T}\sum_{i=1}^{n} p^t_{(i)} c^t_{(\delta(i))} + o(T) \quad \forall \delta : [n] \to [n]. \tag{31}$$

Consider algorithm $\mathcal{A}_j$. Its cost would be $\sum_{t=1}^{T} \mathbf{E}_{i\sim q_j^t}[p^t_{(j)} c^t_{(i)}] = \sum_{t=1}^{T}\sum_{i=1}^{n} p^t_{(j)} c^t_{(i)} q^t_{j,(i)}$, where $q^t_{j,(i)}$ is the i^{th} component of vector q^t_j. Algorithm $\mathcal{A}_j$'s competing benchmark is a fixed action k with cost $\sum_{t=1}^{T} p^t_{(j)} c^t_{(k)}$. Since $\mathcal{A}_j$ is a no-regret algorithm, the following holds:

$$\sum_{t=1}^{T} \mathbf{E}_{i\sim q_j^t}[p^t_{(j)} c^t_{(i)}] \leq \sum_{t=1}^{T} p^t_{(j)} c^t_{(k)} + \text{Regret}(\mathcal{A}_j) \quad \forall k \in [n] \tag{32}$$

Now, take any swap function $\delta : [n] \to [n]$. Summing over inequality 32 for every $\mathcal{A}_j$ and setting $k = \delta(j)$, we get:

$$\sum_{j=1}^{n}\sum_{t=1}^{T}\sum_{i=1}^{n} p^t_{(j)} c^t_{(i)} q^t_{j,(i)} \leq \sum_{j=1}^{n}\sum_{t=1}^{T} p^t_{(j)} c^t_{(\delta(j))} + \sum_{j=1}^{n} \text{Regret}(\mathcal{A}_j).$$

Notice that the RHS of the above inequality is the RHS of inequality 31 that we want to prove. Further, imposing $\sum_{j=1}^{n} p^t_{(j)} q^t_{j,(i)} = p^t_{(i)}$ sets the LHS of both inequalities to be equal too, proving 31. Note that this can always be possible from the existence of a stationary distribution for a stochastic matrix. □

Remark. p^t is the stationary distribution of Q^t but the algorithm $\mathcal{H}$ doesn't necessarily reach a stationary distribution over time. In fact, we would want an adversarial algorithm to be able to make big jumps and not stagnate at a particular distribution.

Partial information scenario

In real life, agents might not know who or what other agents do. In this case, the best thing an agent could do is to minimize its cost assuming the worst would happen. This is the idea behind min-max equilibrium. But a min-max equilibrium needn't be a Nash equilibrium in non-zero-sum games. Also, there is no straightforward algorithmic way of calculating Nash equilibria of games. This makes us

wonder about the usefulness of these concepts. How would people act in real life? What is the right notion of equilibrium?

We learned that no-regret algorithms reach Coarse correlated equilibria (CCE). But CCE can be a convex set and convergence to it doesn't mean convergence to a point. Also, how do we interpret a CCE? Neither agent possesses the CCE distribution but somehow their empirical distribution converges to it in the long run?

The scenarios we discussed so far in the course can be unrealistic for yet another reason. When an agent picks an action, they might only learn the cost of it and not the cost of alternate actions. For instance, when a driver picks a route, they only observe the time it takes on that route. This is the case of partial information when the costs of all actions aren't revealed at each time step.

Multiarm bandits

Imagine a lottery machine with multiple arms. As an agent, you get to pick an arm and observe the reward associated with the arm. The aim is to learn to pick the best arms over time while also collecting as much reward as possible. Here comes the exploration-exploitation tradeoff, wherein you want to try new arms so you don't miss out on the best but also want to pick the best arms you know of more often.

Formally, at time t an agent selects a probability distribution $p^t = (p_1^t, \cdots, p_n^t)$ over their actions. The adversary chooses a cost $c_t = (c_1^t, \cdots, c_n^t)$ corresponding to the actions. An action $i^t \sim p^t$ is chosen at random by the agent and a cost $c_{i_t}^t$ associated with the action is observed. Notice, the difference from the previous cases where the entire cost c^t is observed.

Before discussing an online algorithm for the partial information case, consider the *exponential weights (hedge)* algorithm for the full information case.

Algorithm 5: Exponential Weights (Hedge)

Maintain weights $(w_1^t, \cdots, w_n^t)$ associated with each of the actions.
Initialize weights $w_i^1 = 1 \; \forall i \in [n]$.
for $t \in \{1, 2, \ldots, T\}$ **do**
  Calculate distribution p^t s.t $p_i^t = \frac{w_i^t}{Z^t} \quad \forall \; i \in [n]$, where $Z^t = \sum_{i=1}^n w_i^t$.
  Pick $i^t \sim p^t$ and incur cost $c_{i_t}^t$.
  Update weights $w_i^{t+1} = w_i^t e^{-\epsilon c_i^t} \; \forall i \in [n]$.
end for

Notice that though the cost incurred depends on the action picked, the costs of taking all alternate actions are also observed. Consider the partial information case where these alternate costs are unknown.

This algorithm can be changed as follows:

Algorithm 6: Exp 3

Maintain weights $(w_1^t, \cdots, w_n^t)$ associated with each of the actions.
Initialize weights $w_i^1 = 1 \; \forall i \in [n]$.
for $t \in \{1, 2, \ldots, T\}$ **do**
Calculate distribution p^t s.t $p_i^t = \frac{w_i^t}{Z^t} \;\; \forall \, i \in [n]$, where $Z^t = \sum_{i=1}^n w_i^t$.
Pick $i^t \sim p^t$ and incur cost $c_{i_t}^t$.
Update weight $w_i^{t+1} = w_i^t e^{-\epsilon \hat{c}_i^t} \; \forall i \in [n]$, where

$$\hat{c}_i^t = \begin{cases} \frac{c_{i_t}^t}{p_{i_t}^t} & i = i_t \\ 0 & i \neq i_t \end{cases}$$

end for

Interestingly, the name *Exp 3* for the algorithm is an acronym for exploration, exploitation, and exponential. Notice that though the weight corresponding to only the action picked is changed, the relative weights for all actions still change. If we update the relative weights of all actions despite not observing all costs, why should this algorithm work? The intuition lies in the observations that using exponential weights helps to aggregate costs over time and that $\mathbf{E}[\hat{c}_i^t] = c_i^t$ i.e the expected cost for every action is unbiased. So one could expect the weights to converge to what's expected with unbiased costs over time.

Consider the *exponential weights* algorithm again. Its expected regret is defined as $\mathbf{E}[R] \triangleq \sum_{t=1}^T p^t \cdot c^t - \min_i \sum_{t=1}^T c_i^t$. The following bound holds on this regret:

$$\mathbf{E}[R] \leq \epsilon \sum_{t=1}^T p^t \cdot (c^t \circ c^t) + \frac{\log n}{\epsilon}$$

So if the costs $c_i^t \in [0,1] \;\; \forall i \in [n]$ then $(c^t \circ c^t)_i \leq 1 \;\; \forall i$ and $\mathbf{E}[R] \leq \epsilon + \frac{\log n}{\epsilon}$. Minimizing the RHS of the later inequality w.r.t ϵ gives $\mathbf{E}[R] \leq 2\sqrt{T \log n}$. Hence, the regret for the full information case is $o(\sqrt{T \log n})$. Turns out the regret for partial information case using *Exp 3* is $o(\sqrt{nT \log n})$. Both these regret bounds will be proved in the next class.

Lecture 8

In the last lecture, we firstly completed the proof of the blackbox construction of a no-SWAP-regret algorithm using no-(external)-regret algorithms. Then, we started to discuss online learning for partial information case. We presented the *exponential weights (Hedge)* algorithm and the *Exp3* algorithm and stated the regret bounds of them.

In this Lecture, we will prove the regret bounds of the two algorithms. Then, we will introduce stochastic multi-armed bandits and present the *Upper Confidence Bound (UCB)* algorithm.

Regret Analysis of Multi-arm Bandit Algorithms

Theorem 13 (*Exponential Weights (Hedge)* is no-regret)**.** *The expected regret of the Exponential Weights (Hedge) algorithm is bounded as follows*

$$\mathbb{E}[R(T)] \leq \epsilon \sum_{t=1}^{T} p^t \cdot (c^t \circ c^t) + \frac{\log n}{\epsilon}.$$

Furthermore, if $c_i^t \in [0,1]$, then there exists ϵ with which the algorithm's expected regret can be bounded as follows

$$\mathbb{E}[R(T)] \leq 2\sqrt{T \log n}.$$

Proof. Recall the definition of Z^t

$$\begin{aligned}
Z^{t+1} &= \sum_{i\in[n]} w_i^t e^{-\epsilon c_i^t} && \text{(definition)} \\
&= Z^t \sum_{i\in[n]} p_i^t e^{-\epsilon c_i^t} && \text{(definition)} \\
&\leq_{(A)} Z^t \sum_{i\in[n]} p_i^t (1 - \epsilon c_i^t + \epsilon^2 (c_i^t)^2) && (33) \\
&= Z^t (1 - \epsilon p^t \cdot c^t + \epsilon^2 p^t \cdot (c^t \circ c^t))) && \text{(definition)} \\
&\leq_{(B)} Z^t \exp\{-\epsilon p^t \cdot c^t + \epsilon^2 p^t \cdot (c^t \circ c^t)\}, && (34)
\end{aligned}$$

where (A) is from the fact that $e^{-x} \leq 1 - x + x^2, \forall x > 0$ and (B) is from the fact that $e^{-x} \geq 1 - x, \forall x$. Using the above calculation

iteratively, we have

$$Z^{T+1} \leq Z^1 \exp\{-\epsilon \sum_{t=1}^{T} p^t \cdot c^t + \epsilon^2 \sum_{t=1}^{T} p^t \cdot (c^t \circ c^t)\} \tag{35}$$

$$= n \exp\{-\epsilon \sum_{t=1}^{T} p^t \cdot c^t + \epsilon^2 \sum_{t=1}^{T} p^t \cdot (c^t \circ c^t)\}. \tag{36}$$

Consider $w_k^{T+1}, \forall k \in [n]$. On the one hand,

$$w_k^{T+1} = \exp\{-\epsilon \sum_{t=1}^{T} c_k^t\}.$$

On the other hand,

$$w_k^{T+1} \leq Z^{T+1} \leq n \exp\{-\epsilon \sum_{t=1}^{T} p^t \cdot c^t + \epsilon^2 \sum_{t=1}^{T} p^t \cdot (c^t \circ c^t)\}.$$

Combining these two, we have

$$-\epsilon \sum_{t=1}^{T} c_k^t \leq \log n - \epsilon \sum_{t=1}^{T} p^t \cdot c^t + \epsilon^2 \sum_{t=1}^{T} p^t \cdot (c^t \circ c^t).$$

Thus, we bound the expected regret

$$\mathbb{E}[R(T)] = \sum_{t=1}^{T} p^t \cdot c^t - \min_{k \in [n]} \sum_{t=1}^{T} c_k^t \leq \epsilon \sum_{t=1}^{T} p^t \cdot (c^t \circ c^t) + \frac{\log n}{\epsilon}.$$

If $c_i^t \in [0,1]$, then $\mathbb{E}[R(T)] \leq \epsilon T + \frac{\log n}{\epsilon}$. Furthermore, minimizing the expected regret over ϵ, we have $\epsilon = \sqrt{\frac{\log n}{T}}$ and thus $\mathbb{E}[R(T)] \leq 2\sqrt{T \log n}$. □

Theorem 14 (*Exp3* is no-regret)**.** *The expected regret of the Exp3 algorithm is bounded as follows*

$$\mathbb{E}[R(T)] \leq \epsilon \sum_{t=1}^{T} \sum_{i=1}^{n} (c_i^t)^2 + \frac{\log n}{\epsilon}.$$

Furthermore, if $c_i^t \in [0,1]$, then there exists ϵ with which the algorithm's expected regret can be bounded as follows

$$\mathbb{E}[R(T)] \leq 2\sqrt{Tn \log n}.$$

Proof. Recall the definition of Z^t

$$Z^{t+1} = \sum_{i \in [n]} w_i^t e^{-\epsilon \hat{c}_i^t} \qquad \text{(definition)}$$

$$= Z^t \sum_{i \in [n]} p_i^t e^{-\epsilon \hat{c}_i^t} \qquad \text{(definition)}$$

$$\leq_{(A)} Z^t \sum_{i\in[n]} p_i^t(1-\epsilon\hat{c}_i^t+\epsilon^2(\hat{c}_i^t)^2) \tag{37}$$

$$= Z^t(1-\epsilon p^t\cdot\hat{c}^t+\epsilon^2 p^t\cdot(\hat{c}^t\circ\hat{c}^t))) \qquad \text{(definition)}$$

$$\leq_{(B)} Z^t\exp\{-\epsilon p^t\cdot\hat{c}^t+\epsilon^2 p^t\cdot(\hat{c}^t\circ\hat{c}^t)\}, \tag{38}$$

where (A) is from the fact that $e^{-x}\leq 1-x+x^2, \forall x>0$ and (B) is from the fact that $e^{-x}\geq 1-x, \forall x$. Using the above calculation iteratively, we have

$$Z^{T+1}\leq Z^1\exp\{-\epsilon\sum_{t=1}^T p^t\cdot\hat{c}^t+\epsilon^2\sum_{t=1}^T p^t\cdot(\hat{c}^t\circ\hat{c}^t)\} \tag{39}$$

$$= n\exp\{-\epsilon\sum_{t=1}^T p^t\cdot\hat{c}^t+\epsilon^2\sum_{t=1}^T p^t\cdot(\hat{c}^t\circ\hat{c}^t)\}. \tag{40}$$

Consider $w_k^{T+1}, \forall k\in[n]$. On the one hand,

$$w_k^{T+1}=\exp\{-\epsilon\sum_{t=1}^T\hat{c}_k^t\}.$$

On the other hand,

$$w_k^{T+1}\leq Z^{T+1}\leq n\exp\{-\epsilon\sum_{t=1}^T p^t\cdot\hat{c}^t+\epsilon^2\sum_{t=1}^T p^t\cdot(\hat{c}^t\circ\hat{c}^t)\}.$$

Combining these two, we have

$$-\epsilon\sum_{t=1}^T\hat{c}_k^t\leq\log n-\epsilon\sum_{t=1}^T p^t\cdot\hat{c}^t+\epsilon^2\sum_{t=1}^T p^t\cdot(\hat{c}^t\circ\hat{c}^t).$$

Taking the expectation of both sides and using the following three facts that

$$\mathbb{E}[\hat{c}_k^t]=c_k^t,$$

$$\mathbb{E}[p^t\cdot\hat{c}^t]=\sum_{i=1}^n\mathbb{E}[p_i^t c_i^t]=\sum_{i=1}^n p_i^t\cdot p_i^t\cdot\frac{1}{p_i^t}c_i^t=\sum_{i=1}^n p_i^t c_i^t=p^t\cdot c^t,$$

$$\mathbb{E}[p^t\cdot(\hat{c}^t\circ\hat{c}^t)]=\sum_{i=1}^n\mathbb{E}[p_i^t(c_i^t)^2]=\sum_{i=1}^n p_i^t p_i^t\frac{1}{(p_i^t)^2}(c_i^t)^2=\sum_{i=1}^n(c_i^t)^2,$$

we have

$$-\sum_{t=1}^T c_k^t\leq-\sum_{t=1}^T p^t\cdot c^t+\epsilon\sum_{t=1}^T\sum_{i=1}^n(c_i^t)^2+\frac{\log n}{\epsilon}.$$

Since it is true for any $k\in[n]$, taking the supremum of LHS, we have

$$-\inf_{k\in[n]}\sum_{t=1}^T c_k^t\leq-\sum_{t=1}^T p^t\cdot c^t+\epsilon\sum_{t=1}^T\sum_{i=1}^n(c_i^t)^2+\frac{\log n}{\epsilon},$$

which implies

$$\mathbb{E}[R(T)]=\sum_{t=1}^T p^t\cdot c^t-\inf_{k\in[n]}\sum_{t=1}^T c_k^t\leq\epsilon\sum_{t=1}^T\sum_{i=1}^n(c_i^t)^2+\frac{\log n}{\epsilon}.$$

If $c_i^t \in [0,1]$, then $\mathbb{E}[R(T)] \leq \epsilon T + \frac{\log n}{\epsilon}$. Furthermore, minimizing the expected regret over ϵ, we have $\epsilon = \sqrt{\frac{\log n}{T}}$ and thus $\mathbb{E}[R(T)] \leq 2\sqrt{T \log n}$. □

Stochastic multi-armed bandits

Now we turn to another bandit setting and present an algorithm called *UCB*.

Setup

- n arms, each associated with a distribution P_i.
- Denote $X_{i,t}$ to be the reward from the i-th arm at time t. (i.e. $\{X_{i,t}\}_t \overset{\text{iid}}{\sim} P_i$.)
- Let $\mu_i = \mathbb{E}[X_{i,t}]$, $\mu^* = \max_{j \in [n]} \mu_j$ and $\Delta_i = \mu^* - \mu_i, \forall i \in [n]$.
- Denote $I_t \in [n]$ to be agent's choice at time t. Let $T_i = \sum_{t=1}^T \mathbf{1}_{\{I_t = i\}}$.

Regret

- (Pseudo-) Regret:
$$\begin{aligned} R(T) &= \max_{j \in [n]} \left(\sum_{t=1}^T \left(\mathbb{E}[X_{j,t}] - \mathbb{E}[X_{I_t,t}] \right) \right) \\ &= T\mu^* - \sum_{t=1}^T \mu_{I_t} \\ &= \sum_{t=1}^T (\mu^* - \mu_{I_t}) \\ &= \sum_{t=1}^T \sum_{i=1}^n \mathbf{1}_{\{I_t = i\}} (\mu^* - \mu_i) \\ &= \sum_{t=1}^T \sum_{i=1}^n \mathbf{1}_{\{I_t = i\}} \Delta_i \\ &= \sum_{i=1}^n \Delta_i \sum_{t=1}^T \mathbf{1}_{\{I_t = i\}} \\ &= \sum_{i=1}^n \Delta_i T_i. \end{aligned}$$
- Expected (Pseudo-) regret:
$$\mathbb{E}[R(T)] = \sum_{i=1}^n \Delta_i \mathbb{E}[T_i].$$

Upper Confidence Bound (UCB) Algorithm At each time T, form

$$\hat{\mu}_{i,T_i} = \frac{\sum_{i:I_t=i} X_{t,i}}{T_i}.$$

Choose

$$I_t \in \arg\max_{i\in[n]} \left(\hat{\mu}_{i,T_i} + \sqrt{\frac{1}{2T_i}\log(\frac{1}{\delta})} \right),$$

where δ is a parameter of the algorithm. We will give a concrete example of δ and discuss the intuition of the algorithm in the next lecture.

A clarificatioon of different notions of regret

When coming to stochastic bandits, the randomness is over two things, not only the algorithm but also the loss (reward) functions. The different orders of taking expectations and the infimum give us different notions of regret. Even though the lecture didn't discuss this part, I think it's good to clarify them here.

Let ℓ_t be the loss function in round t. Say there are k actions and the action set is $[K]$. Let a_t be the algorithm's choice of action at round t.

- (Realized) Regret:

$$R(T) = \sum_{t=1}^{T} \ell_t(a_t) - \inf_{a\in[K]} \sum_{t=1}^{T} \ell_t(a).$$

  This is the notion we used in adversarial bandit setting, where there is no randomness over loss.

- Expected Regret:

$$\mathbb{E}[R(T)] = \mathbb{E}_{\ell,\text{Alg}} \left[\sum_{t=1}^{T} \ell_t(a_t) - \inf_{a\in[K]} \sum_{t=1}^{T} \ell_t(a) \right].$$

  The above two notions are usually considered in solving games like CCE.

- (Realized) Pseudo-regret:

$$\sum_{t=1}^{T} \mathbb{E}_\ell[\ell_t(a_t)] - \inf_{a\in[K]} \mathbb{E}_\ell[\sum_{t=1}^{T} \ell_t(a)].$$

  This is usually considered in stochastic setting, which considers the mean reward of the arms as follows

$$R(T) = \sup_{a\in[K]} \sum_{t=1}^{T} \mu(a) - \sum_{t=1}^{T} \mu(a_t) = T\mu^* - \sum_{t=1}^{T} \mu(a_t) = \sum_{a\in[K]} \Delta_a \cdot T_a,$$

  where the last equality has been shown above.

- Expected (Pseudo-)regret:

$$\mathbb{E}_{\text{Alg}}\left[\sum_{t=1}^{T}\mathbb{E}_{\ell}[\ell_t(a_t)] - \inf_{a\in[K]}\mathbb{E}_{\ell}[\sum_{t=1}^{T}\ell_t(a)]\right].$$

  People sometimes omit the Pseudo here, especially in stochastic bandit setting. For stochastic bandit setting, we have

$$\mathbb{E}[R(T)] = \mathbb{E}_{\text{Alg}}[T\mu^* - \sum_{t=1}^{T}\mu(a_t)] = \sum_{a\in[K]}\Delta_a \cdot \mathbb{E}[T_a].$$

Lecture 9

In the last lecture, we introduced Upper Confidence Bound (UCB) algorithm that can be used to solve the multi-arm bandit problem. In this lecture, we show the upper bound of regret of UCB algorithm. We start by reminding the notations. We have total n arms and reward of each arm $X_{i,t}$ follows the distribution P_i for every time period $t \in [T]$. Define $\mu_i = \mathbb{E}[X_i]$ and I_t be an index that is selected by the algorithm at time t. $T_i = \sum_{t=1}^{T} \mathbb{1}\{I_t = i\}$ denote the number iterations where index i selected by the algorithm and $\hat{\mu}_{i,T_i} = \frac{1}{T_i}\sum_{t:I_t=i} X_{i,t}$ denote the sample mean of the i-th arm. We define the regret $\mathcal{R}(T) := \max_{j\in[n]} \sum_{t=1}^{T}\Big(\mathbb{E}[X_{j,t}] - \mathbb{E}[X_{I_t,t}]\Big)$. Then UCB algorithm chooses index I_t by the following rule:

$$I_t \in \operatorname*{argmax}_{i\in[n]} \hat{\mu}_{i,T_i} + \sqrt{\frac{\log 1/\delta}{2T_i}}.$$

δ is the parameter that defines the degree of confidence of the algorithm and we choose $\delta = t^{-\alpha}$ at time t. $\alpha > 1$ is the parameter that we need to choose and inputting this, we get

$$I_t \in \operatorname*{argmax}_{i\in[n]} \hat{\mu}_{i,T_i} + \sqrt{\frac{\alpha \log t}{2T_i}}.$$

We can see that T_i is the exploitation part of this algorithm as less we visit the i-th arm $\frac{1}{T_i}$ will increase. Also, $\alpha \log t$ is the exploration part of this algorithm. It is natural to ask where $\sqrt{\frac{\log 1/\delta}{2T_i}}$ comes from. This term comes from Hoeffding's inequality and we will prove this concentration inequality first by introducing Hoeffding's lemma. We define $\Delta_i = \mu^* - \mu_i$, where $\mu^* = \max_{j\in[n]} \mu_j$. Under this notation, we can write the regret as $\mathcal{R}(T) = \sum_{i=1}^{n} \Delta_i \mathbb{E}(T_i)$. Now, we introduce Chernoff-Hoeffding strategy.

Let Z be a random variable and $\lambda \geq 0$ be given. Then we have

$$\mathbb{P}(Z \geq t) = \mathbb{P}\left(e^{\lambda Z} \geq e^{\lambda t}\right) \leq \frac{\mathbb{E}\left[e^{\lambda Z}\right]}{e^{\lambda t}}$$

where the last inequality is from Markov inequality. As $\lambda \geq 0$ is given, we can choose λ to minimize the RHS of the inequality. That is,

$$\mathbb{P}(Z \geq t) \leq \inf_{\lambda \geq 0} \frac{\mathbb{E}[e^{\lambda Z}]}{e^{\lambda t}} = e^{-\sup_{\lambda \geq 0}(\lambda t - \psi(\lambda))}$$

where we define $\psi(\lambda) = \log(\mathbb{E}[e^{\lambda Z}])$. Note that $\sup_{\lambda \geq 0}(\lambda t - \psi(\lambda))$ is a convex conjugate function of $\psi(\cdot)$ and can be expressed as $\psi^*(t)$. To upper bound $-\sup_{\lambda \geq 0}(\lambda t - \psi(\lambda))$, we need to know how $\psi(\lambda)$ behaves and we can do this by using second-order Taylor expansion which is $\psi(\lambda) = \psi(0) + \lambda\psi'(0) + \frac{\lambda^2}{2}\psi''(\theta)$ for some $\theta \in [0, \lambda]$. Notice that $\psi(0) = 0$ and $\psi'(0) = 0$, so we can rewrite this as $\psi(\lambda) = \frac{\lambda^2}{2}\psi''(\theta)$ for some $\theta \in [0, \lambda]$. So, we only need to bound $\psi''(\theta)$ and this can be done by the following lemma.

Lemma 15 (Hoeffding's Lemma)**.** *Let X be a random variable with $\mathbb{E}[X] = 0$ and $X \in [a, b]$. Then we have*

1. $\psi''(\lambda) \leq \frac{(b-a)^2}{4}$
2. $\psi(\lambda) \leq \frac{\lambda^2(b-a)^2}{8}$.

Proof of 1. Notice that for any bounded random variable $X \in [a, b]$, we have $|X - \frac{b-a}{2}| \leq \frac{b-a}{2}$ and $Var(X) \leq \frac{(b-a)^2}{4}$. Let P denote the distribution of X and P_λ denote the distribution with density $e^{-\psi(\lambda)}e^{\lambda X}$ with respect to P. Then, we have

$$\begin{aligned}\psi''(\lambda) &= e^{-\psi(\lambda)}\mathbb{E}[X^2 e^{\lambda X}] - e^{-2\psi(\lambda)}\left(\mathbb{E}[Xe^{\lambda X}]^2\right) \\ &= \mathrm{Var}_{P_\lambda}(X) \leq \frac{(b-a)^2}{4}.\end{aligned}$$

Notice that the second equation comes from changing base distribution from P to P_λ and the last inequality comes from the fact that $\mathrm{Var}(X) \leq \frac{(b-a)^2}{4}$ for $X \in [a, b]$. □

Using Hoeffding's Lemma, we can further bound

$$\mathbb{P}(Z \geq t) \leq e^{-\sup_{\lambda \geq 0}\{\lambda t - \frac{\lambda^2(b-a)^2}{8}\}} \leq e^{-\frac{2t^2}{(b-a)^2}}.$$

Note that the second inequality comes from solving maximization of the quadratic function of λ. This concentration inequality is called Hoeffding's inequality and we give the formal explanation below.

Theorem 16 (Hoeffding's Inequality)**.** *Let X be a random variable with $\mathbb{E}[X] = 0$ and $X \in [a, b]$. Then for given $t \in \mathbb{R}$, we have*

$$\mathbb{P}(Z \geq t) \leq e^{-\frac{2t^2}{(b-a)^2}}.$$

We can use Hoeffding's inequality get some high probability guarantee of random variables.

Example. Let $X \in [0,1]$ and for given $\delta \in (0,1)$ we want to guarantee $\mathbb{P}(Z \geq t) \leq \delta$. By Hoeffding's inequality, we have $\mathbb{P}(Z \geq t) \leq e^{-2t^2}$. We choose $\delta = e^{-2t^2}$ and we have $t = \sqrt{\frac{1}{2}\log(\frac{1}{\delta})}$.

Example. Let $X_1, \cdots, X_n$ be random variables such that $X_i \in [a_i, b_i]$ and independent to each other. We define $Z = \sum_{i=1}^n (X_i - \mathbb{E}X_i)$. By Hoeffding's inequality, we have $\mathbb{P}(Z \geq t) \leq e^{-\frac{2t^2}{\sum_{i=1}^n (b_i - a_i)^2}}$. This result directly comes from $\psi(\lambda) \leq \frac{\lambda^2}{8}\sum_{i=1}^n (b_i - a_i)^2$. If we customize to $(a_i, b_i) = (0,1)$ for all $i \in [n]$, then we have $\mathbb{P}(Z \geq t) \leq e^{-\frac{2t^2}{n}}$. Dividing both side in side $\mathbb{P}(\cdot)$ by n and using $\bar{Z}$ to denote the sample mean and replacing $\frac{t}{n}$ with ϵ, we get

$$\mathbb{P}\left(\bar{Z} \geq \epsilon\right) \leq e^{-2n\epsilon^2}.$$

If we want to guarantee $\mathbb{P}\left(\bar{Z} \geq \epsilon\right) \leq \delta$, then we choose $\epsilon = \sqrt{\frac{\log(1/\delta)}{2n}}$ and get the desired result.

Notice that this choice of $\epsilon = \sqrt{\frac{\log(1/\delta)}{2n}}$ appears in the UCB algorithm. Indeed, we have $\mathbb{P}\left(\mu_i \leq \hat{\mu}_{i,T_i} + \sqrt{\frac{\alpha \log t}{2T_i}}\right) \geq 1 - \delta$ by Hoeffding's inequality. We say that UCB algorithm uses $1 - \delta$ confidence bound of μ_i. Now, we show the bound of the regret by UCB algorithm.

Theorem 17. *Let $\mathcal{R}(T)$ be the regret of the solution that is generated by the UCB algorithm. Then, we have*

$$\mathcal{R}(T) \leq \sum_{i:\Delta_i > 0} \left(\frac{2\alpha \log T}{\Delta_i} + \frac{2\alpha}{\alpha - 1}\Delta_i\right).$$

Proof. WLOG, assume that arm 1 is optimal. We define $c_i = \sqrt{\frac{\alpha \log t}{2T_i}}$. Notice that $\mathcal{R}(T) = \sum_{i:\Delta_i > 0} \Delta_i \mathbb{E}[T_i]$, so we only need bound the term $\mathbb{E}[T_i]$. Also, we define "*succeed*" in the following way. We say Hoeffding succeeds for optimal arm 1 if $\mu_1 \leq \hat{\mu}_{1,T_1} + c_1$ and say Hoeffding succeeds for non-optimal arm $i \neq 1$ if $\mu_i \geq \hat{\mu}_{i,T_i} - c_i$. Arm $i \neq 1$ will be played if either

1. Hoeffding fails for arm i or arm 1.
2. Hoeffding succeeds for both arm i and 1, but the means μ_i and μ_1 are too close to be distinguished with the current amount of data.

We first consider the second case. The bad choice can be made if $\hat{\mu}_{1,T_1} + c_1 < \hat{\mu}_{i,T_i} + c_i$. As we assumed Hoeffding succeeded in both arms i and 1, this implies

$$\mu_1 \leq \hat{\mu}_{1,T_1} + c_1 < \hat{\mu}_{i,T_i} + c_i \leq \mu_i + 2c_i.$$

This bad choice stops happening when $\mu_i + 2c_i \leq \mu_1$. This is equivalent to

$$2c_i \leq \Delta_i \iff 2\sqrt{\frac{\alpha \log t}{2T_i}} \leq \Delta_i \iff T_i \geq \frac{2\alpha \log t}{\Delta_i^2}.$$

Now we consider the first case. We denote F_1 to denote a set $\{$Hoeffding fails for arm 1$\}$ and F_i to denote a set $\{$Hoeffding fails for arm $i\}$. Then the expected number of first case to happen is $\mathbb{E}[\mathbb{1}\{F_1 \cup F_i\}]$ and we can bound this term by using union bound and Hoeffding's inequality;

$$\mathbb{E}[\mathbb{1}\{F_1 \cup F_i\}] \leq \mathbb{P}(F_1) + \mathbb{P}(F_i) \leq t^{-\alpha}.$$

Combining these the two results above, we can upper bound $\mathbb{E}[T_i]$ as

$$\begin{aligned}
\mathbb{E}[T_i] = \sum_{t=1}^{T} \mathbb{E}\mathbb{1}\{I_t = i\} &\leq \sum_{t=1}^{T} \mathbb{E}[\mathbb{1}\{F_1 \cup F_i\}] + \frac{2\alpha \log T}{\Delta_i^2} \\
&\leq \sum_{i=1}^{T} t^{-\alpha} + \frac{2\alpha \log T}{\Delta_i^2} \\
&\leq \frac{2\alpha}{\alpha - 1} + \frac{2\alpha \log T}{\Delta_i^2}.
\end{aligned}$$

The first inequality comes from the fact that $i \neq 1$ will be played if either the two cases that we considered above. The first term comes from case 1 and the second term comes from case 2. Also, we can further bound $\mathbb{E}[\mathbb{1}\{F_1 \cup F_i\}]$ by using the fact the UCB algorithm uses the $1 - \delta$ Hoeffding confidence bound. As this holds for any $i \neq 1$, multiplying both side by Δ_i and adding all $i \neq 1$, we have

$$\mathcal{R}(T) = \sum_{i>1} \Delta_i \mathbb{E}[T_i] \leq \sum_{i>1} \left(\frac{2\alpha \log T}{\Delta_i} + \frac{2\alpha}{\alpha - 1} \Delta_i \right).$$

□

Lecture 10

In this lecture, we study statistical learning theory (SLT) also known as empirical process theory (EPT). Suppose that we take samples by repeating i.i.d. process. If we get enough samples, we are very likely to do well for things coming afterward. However, in complicated situations, fitting regressions of arbitrary functions is hopeless. We need to restrict to some set of functions to learn.

In SLT, we have training data with labels $\{(x_1, y_1), \ldots, (x_n, y_n)\} \sim \mathcal{X} \times \mathcal{Y}$ and a hypothesis class $\mathcal{H}$ of maps $\mathcal{X} \to \mathcal{Y}$. We have i.i.d. assumption but nonparametric, that is, $(x_i, y_i) \overset{i.i.d.}{\sim} \mathbb{P}$ over $\mathcal{X} \times \mathcal{Y}$. Then, for a loss function $l(y, h(x))$ s.t. $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, the risk is defined as

$$R(h) = \mathbb{E}\left[l(Y, h(X)\right]$$

and the empirical risk is

$$\hat{R}(h) = \frac{1}{n}\sum_{i=1}^{n} l(y_i, h(x_i)).$$

Empirical risk minimization (ERM)

$$\inf_{h \in \mathcal{H}} \hat{R}(h) \to \hat{h}$$

The basic idea is that if for each h, $\hat{R}(h)$ is a good estimator of $R(h)$, then ERM might find a good h that minimizes the risk. Using LLN on h, $\hat{R}(h)$ converges to $R(h)$ pointwise. However, by Gibb's Phenomenon, there are oscillations around jumps of the Fourier series of a piecewise continuously differentiable function. Large peaks are produced around jumps that result in large approximation errors. Therefore, we need uniform LLN.

The starting point is using Hoeffding.

$$\mathbb{P}\left(\hat{R}(h) - R(h) \geq \epsilon\right) \leq e^{-n\epsilon^2}$$

as $\hat{R}(h)$ is sum of i.i.d. random variables. We want

$$\mathbb{P}\left(\sup_{h\in\mathcal{H}}\left|\hat{R}(h)-R(h)\right|\geq\epsilon\right)\leq C\left(\mathcal{H}\right)e^{-n\epsilon^2},$$

where $C\left(\mathcal{H}\right)$ is a measure of the complexity of $\mathcal{H}$. For example, when $\mathcal{H}$ is finite, $C\left(\mathcal{H}\right)$ can be the cardinality of $\mathcal{H}$. However, consider a class of step functions. Although it is an infinite class, it is actually not complex. Thus, cardinality is not a good measure of complexity in general. We call the measure of the complexity as **growth function** $\Phi(n)=C\left(\mathcal{H}\right)$. If $\Phi(n)$ is a polynomial function in VC dimension, it is okay since we can use SLLN. For our purpose, growth function should grow slower than $e^{n\epsilon^2}$.

Tools we will use for ERM are as follows:

- Hoeffding and other concentration inequalities
- Growth function
- VC dimension
- Rademacher complexity
- Covering numbers

We will focus on the first three in this course.

Let $R^*=R(h)$ for some h that can even be outside of $\mathcal{H}$ which is really smart and doing well. Then, we can decompose the error of our estimator as

$$R(\hat{h})-R^*=\overbrace{\left[R(\hat{h})-\inf_{h\in\mathcal{H}}R(h)\right]}^{\text{estimation error}}+\overbrace{\left[\inf_{h\in\mathcal{H}}R(h)-R^*\right]}^{\text{approximation error}}$$

We will focus on the estimation error.

$$\begin{aligned}
R(\hat{h})-\inf_{h\in\mathcal{H}}R(h)&=\left(R(\hat{h})-\hat{R}(\hat{h})\right)+\left(\hat{R}(\hat{h})-\inf_{h\in\mathcal{H}}R(h)\right)\\
&=\left(R(\hat{h})-\hat{R}(\hat{h})\right)+\sup_{h\in\mathcal{H}}\left(\hat{R}(\hat{h})-R(h)\right)\\
&\leq\left(R(\hat{h})-\hat{R}(\hat{h})\right)+\sup_{h\in\mathcal{H}}\left(\hat{R}(h)-R(h)\right)\\
&\leq\left(R(\hat{h})-\hat{R}(\hat{h})\right)+\sup_{h\in\mathcal{H}}\left|\hat{R}(h)-R(h)\right|\\
&\leq 2\sup_{h\in\mathcal{H}}\left|\hat{R}(h)-R(h)\right|
\end{aligned}$$

where the first inequality holds since $\hat{h}$ minimizes $\hat{R}$. Thus, our goal is to control $\sup_{h\in\mathcal{H}}\left|\hat{R}(h)-R(h)\right|$. Note that $\left(\hat{R}(h)-R(h)\right)_{h\in\mathcal{H}}$ is a stochastic process indexed by $\mathcal{H}$, which we call **empirical process**.

Given samples $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, we want to derive

$$\mathbb{P}_S \left(\sup_{h \in \mathcal{H}} \left| \hat{R}(h) - R(h) \right| \geq \epsilon \right) \leq \delta$$

Definition 10 (Growth Function)**.** For any subset $\mathcal{C} \subseteq \mathcal{X}$, define

$$\mathcal{H}_{\mathcal{C}} = \{ f \in \mathcal{F} : \exists h \in \mathcal{H} \text{ s.t. } \forall x \in \mathcal{C},\ f(x) = h(x) \}$$

where $\mathcal{F}$ is the set of all measurable maps from $\mathcal{C}$ to $\mathcal{Y}$. Then, the growth function is

$$\Phi(n) = \max_{\mathcal{C} \subseteq \mathcal{X},\ |\mathcal{C}| \leq n} |\mathcal{H}_{\mathcal{C}}|.$$

e.g. $\mathcal{Y} = \{0, 1\} \Rightarrow \Phi(n) \leq 2^n$, naively, but not good enough for SLLN.
e.g. $\mathcal{H} \subseteq \{h : \mathbb{R} \to \{-1, 1\}\}$ such as $\mathcal{H} = \{\mathcal{X} \to sgn(x - b), b \in \mathbb{R}\}$.
Consider points $\mathcal{C} = \{x_1, x_2, \ldots, x_n\}$. Since $sgn(x - b)$ is a step function, we only have $n + 1$ discriminations. Thus, $\Phi(n) = n + 1$.

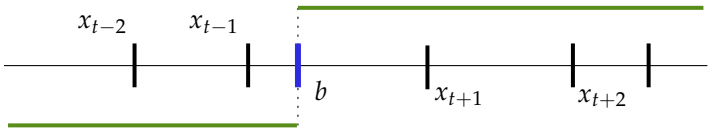


Figure 7: Step Function $sgn(x - b)$

Theorem 18. *Assume bounded loss. WLOG, $l \in [0, 1]$. With probability at least $1 - \delta$ on the training set w.r.t. repeated sampling of training data of size n, we have*

$$\left| R(h) - \hat{R}(h) \right| \leq \sqrt{\frac{8 \ln \left(\frac{4}{\delta} \Phi(2n) \right)}{n}} \quad \textit{uniformly on } \mathcal{H}.$$

To prove Theorem 18, we first introduce symmetrization.

Symmetrization

Define a shadow sample S'. If $\left| R(h) - \hat{R}(h) \right| \geq \epsilon$ and $\left| R(h) - \hat{R}'(h) \right| \leq \frac{\epsilon}{2}$, then, $\left| \hat{R}'(h) - \hat{R}(h) \right| \geq \frac{\epsilon}{2}$, using the triangle inequality. Then,

$$\underbrace{\mathbb{1}\left\{ \left| R(h) - \hat{R}(h) \right| \geq \epsilon \right\}}_{\text{I}} \cdot \underbrace{\mathbb{1}\left\{ \left| R(h) - \hat{R}'(h) \right| \leq \frac{\epsilon}{2} \right\}}_{\text{II}} \leq \underbrace{\mathbb{1}\left\{ \left| \hat{R}'(h) - \hat{R}(h) \right| \geq \frac{\epsilon}{2} \right\}}_{\text{III}}.$$

Take expectation w.r.t. S' on II: Using Hoeffding,

$$\mathbb{E}_{S'} \left[\mathbb{1}\left\{ \left| R(h) - \hat{R}'(h) \right| \leq \frac{\epsilon}{2} \right\} \right] \geq 1 - 2 \exp\left(-\frac{n\epsilon^2}{2} \right) \geq \frac{1}{2} \qquad (41)$$

where the last inequality holds for $n \geq \frac{4\ln 2}{\epsilon^2}$. Similarly taking expectation w.r.t. S' on III:

$$\mathbb{E}_{S'}\left[\mathbb{1}\left\{\left|\hat{R}'(h)-\hat{R}(h)\right| \geq \frac{\epsilon}{2}\right\}\right] \leq \mathbb{P}_{S'}\left(\sup_h \left|\hat{R}'(h)-\hat{R}(h)\right| \geq \frac{\epsilon}{2}\right). \tag{42}$$

Using (41) and (42), we get

$$\mathbb{1}\left\{\left|R(h)-\hat{R}(h)\right| \geq \epsilon\right\} \leq 2\mathbb{P}_{S'}\left(\sup_h \left|\hat{R}'(h)-\hat{R}(h)\right| \geq \frac{\epsilon}{2}\right). \tag{43}$$

Take expectation w.r.t. S on (43),

$$\mathbb{P}_S\left(\left|R(h)-\hat{R}(h)\right| \geq \epsilon\right) \leq 2\mathbb{P}_{S,S'}\left(\sup_h \left|\hat{R}'(h)-\hat{R}(h)\right| \geq \frac{\epsilon}{2}\right)$$

Let $\sigma = \Pi_{i=1}^n \sigma_i$, where σ_i's are i.i.d. random variables with Rademacher distribution, i.e., $\sigma_i = \begin{cases} 1, & \text{w.p. } \frac{1}{2} \\ -1, & \text{w.p. } \frac{1}{2} \end{cases}$. Then,

$$\begin{aligned} &2\mathbb{P}_{S,S'}\left(\sup_h \left|\hat{R}'(h)-\hat{R}(h)\right| \geq \frac{\epsilon}{2}\right) \\ &= 2\mathbb{E}_{S,S'}\left[\mathbb{P}_\sigma\left(\sup_h \frac{1}{n}\sum_{i=1}^n \left(l\left(Y_i, h(X_i)\right) - l\left(Y_i', h(X_i')\right)\right)\sigma_i \geq \frac{\epsilon}{2}\right)\right] \end{aligned}$$

If we condition on samples S and S', $\sup_h \frac{1}{n}\sum_{i=1}^n \left(l\left(Y_i, h(X_i)\right) - l\left(Y_i', h(X_i')\right)\right)$ is just a constant. Additionally, since S, S' come from the same distribution, $l_S - l_{S'}$ and $l_{S'} - l_S$ have the same distribution. Thus, $\sup_h \frac{1}{n}\left(l\left(Y_i, h(X_i)\right) - l\left(Y_i', h(X_i')\right)\right)\sigma_i$ is some number $\times$ coin toss. Therefore, conditioning on S, S', the randomness w.r.t. $\{\sigma_i\}$ is Hoeffding and is bounded by $\exp\left(-\frac{n\epsilon^2}{n}\right)$, which is independent of S, S'. Thus, using union bound, we get

$$2\mathbb{P}_{S,S'}\left(\sup_h \left|\hat{R}'(h)-\hat{R}(h)\right| \geq \frac{\epsilon}{2}\right) \leq 4\Phi(2n)\exp\left(-\frac{n\epsilon^2}{8}\right).$$

Note that the random variable in Hoeffding bound is the loss, which we assumed that it is in $[0, 1]$.

Lecture 11

The goal of this lecture is to induce an important concept of VC Dimension, and introduce a way to go between statistical learning and online learning and interpolate between the two.

The setting considered in the lecture is identical to the last lecture:

1. Dataset $(x_i, y_i) \sim D$ for some unknown distribution D
2. Loss function l. It doesn't matter what l is, we just assume it to be bounded, convex, and Lipschitz function.
3. Hypothesis class $\mathcal{H}$

We aim to find a function $\hat{h}$ such that

$$\hat{h} = \arg\min_{h \in \mathcal{H}} \mathbb{E}_D \left[l(h(x), y) \right] \triangleq \arg\min_{h \in \mathcal{H}} R(h)$$

In empirical risk minimization (ERM), given the dataset of samples $S = \{(x_i, y_i)\}$, instead of the true expectation, we just evaluate the empirical loss and minimize it:

$$\tilde{h} = \arg\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} l(y_i, h(x_i)) \triangleq \arg\min_{h \in \mathcal{H}} \hat{R}_n(h)$$

Definition 11 (Uniform convergence)**.** The sequence of the gap between the empirical risk and the true risk, indexed by the number of samples, *converges uniformly* if

$$\forall h \in \mathcal{H}, |\hat{R}_n(h) - R(h)| \leq \epsilon \quad \text{w.p.} \quad 1 - \delta \quad \text{as long as } n \text{ is large}$$

Also, recall the definition of growth function from the previous lecture:

Definition 12 (Growth function)**.**

$$\Phi(n) = \max_{|C|=n} \left| H|_C \right| = \left| \left\{ f : C \to R \middle| \exists \hat{f} \in \mathcal{H} : f(x) = \hat{f}(x) \forall x \in C \right\} \right| \tag{44}$$

The above definition just expresses the number of functions that can be expressed by $\mathcal{H}$, as a function of the size of the dataset. The result of the last lecture (Theorem 18) is as follows:

$$\left|\hat{R}_n(h) - R(h)\right| \leq C \cdot \sqrt{\frac{\log(\Phi(2n)/\delta}{n}}$$

The main challenge now is how do we bound the growth function, so that the uniform convergence of the risk gap is achieved. We seek to bound it using some natural quantity of a hypothesis class, one of which is the VC dimension.

Definition 13 (Shattering)**.** Let $T = \{x_1, \ldots, x_{|T|}\}$. T is *shattered* if $\forall y \in \{-1, 1\}^{|T|}$, there exists a function $h \in \mathcal{H}$ such that $h(x_i) = y_i$.

The concept of shattering relates to growth functions, in that: If shattered set has size n, then $\phi(k) = 2^k \; \forall k \leq n$. This holds trivially and gets us nowhere if we want the uniform convergence of the risk bound.

Definition 14 (Vapnik–Chervonenkis (VC) dimension)**.** Given $\mathcal{H}$, the VC dimension of $\mathcal{H}$ is the maximum cardinality of a shattered set.

Example (1-D Thresholding).

Let $\mathcal{H} = \{sgn(x - b) : b \in \mathbb{R}\}$, where $sgn(x)$ is a step function, mapping to 1 if $x > 0$ or -1 if $x \leq 0$. Can $\mathcal{H}$ shatter a set of two samples? No. Consider $S = \{(x_1, 1), (x_2, -1)\}$ with $x_1 < x_2$. Therefore, the VC dimension of 1D thresholding is 1.

Theorem 19 (Sauer-Shelah Lemma)**.** *If you take a class $\mathcal{H}$ of VC dimension d, then*

$$\Phi(n) \leq \sum_{i=0}^{d} \binom{n}{i}.$$

Here, the RHS denotes the size of the set of subsets of size up to d. For large enough d, it holds that $\sum_{i=0}^{d} \binom{n}{i} \lesssim n^{O(d)}$. Plugging this back in to Theorem 18 gives

$$\left|\hat{R}_n(h) - R(h)\right| \leq C\sqrt{\frac{d\log(n) - \log(\delta)}{n}} \to 0 \text{ as } n \to \infty$$

Two Notions of Learning

So far in the lectures, we have covered both statistical learning (SL) and online learning (OL). We note their differences:

1. In SL, we assume that the data is independent and identically distributed. This is not necessarily the case in OL, as we even consider an adversarial sequence of data.

2. The rate of learning depend on VC dimension and Littlestone dimension, in SL and OL respectively.

3. The issue with SL is that it makes a very strong assumption of i.i.d. distribution of the dataset. However in real life, we know that the decisions affect which data we collect, as in the application of self-driving for example. On the other hand in OL, even in a very simple learning environment, we may not be able to learn anything.

We will exemplify the third bullet point by 1-D thresholding again. We've seen that the VC dimension of 1-D thresholding problem is 1. However, in online learning, its Littlestone dimension is ∞, so you can't get any rate even in the simplest setting like this. In some sense, the online learning model is too expressive, so that you shouldn't be able to do much.

What we want to see today is a model that interpolates between the two and get the best of both worlds. There are a few things to capture at this point:

1. When x_{t+1} depends on x_t

2. Does noise help?

Let us now consider a smoothed adversary model. As similar in the online learning model, we assume that the adversary picks a distribution D_t and throws at us $x_t \sim D_t$. The learner observes x_t, and responds with $\hat{y}_t$, and the adversary reveals the correct answer y_t. This model captures:

1. The dependence of x_{t+1} on x_t, by letting the distribution D_t to be a function of $\{x_k\}_{1:t-1}$, $\{y_k\}_{1:t-1}$, and $\{\hat{y}_k\}_{1:t-1}$

2. Noise by letting D_t be noisy.

Definition 15 ($\sigma-$smoothness)**.** A distribution D is σ-smooth with respect to a distribution μ, if for all $x \sim \mu$,

$$D(x) \leq \frac{\mu(x)}{\sigma}$$

We assume σ just just some positive parameter smaller than 1.

In the smooth adversary setting, the adversary picks x_t, in that it picks D_t that is σ-smooth, and then the learner observes x_t and then the game goes on. The only constraint for the adversary is that the distribution has to be σ-smooth, and the learner knows what σ is. It rules out cases such as when the adversary is allowed to put too much mass at any single point, by bounding a mass at any given point.

Remark. As $\frac{1}{\sigma} \to \infty$, it is actually the adversarial model, and as σ goes to 0, this is statistical learning with the knowledge of the marginal distribution of x_t.

So we would like to evaluate the rate for the smoothed adversary model. First we note what our benchmark is by the below, which is just the regret:

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} l(\hat{y}, y) - \inf_{h\in\mathcal{H}} \frac{1}{n} l(h(x_i), y_i)\right]$$

In online learning and statistical learning, we've shown that

$$\text{Regret} \leq \sqrt{\frac{LD\log(n)}{n}} \text{ and } \text{Regret} \leq \sqrt{\frac{VC\log(n)}{n}},$$

respectively.

Theorem 20 (HRS'22)**.** *In smoothed online learning,*

$$\text{Regret} \leq \sqrt{\frac{VC\log(\frac{n}{\sigma})}{n}}$$

The interpretation of this theorem is that smoothed online learning is a nice interpolation of statistical learning and online learning, by noting that it is always the class that $VC(\mathcal{H}) \leq LD(\mathcal{H})$. In particular, for any reasonably small $\frac{1}{\sigma}$ this gives back the result for statistical learning. For many classes of interest, usually VC dimension is small and Littlestone dimension is large.

Proof. Recall the definition of regret:

$$\frac{1}{n}\sum_{i=1}^{n} l(\hat{y}_i, y_i) - \inf_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} l(h(x_i), y_i)$$

Let $\mathcal{H}' \subseteq \mathcal{H}$, where $\mathcal{H}'$ is finite. Then we decompose the regret by adding and subtracting the optimal risk under $\mathcal{H}'$:

$$\underbrace{\frac{1}{n}\sum_{i=1}^{n} l(\hat{y}_i, y_i) - \inf_{h'\in\mathcal{H}'} \frac{1}{n}\sum_{i=1}^{n} l(h'(x_i), y_i)}_{\leq\sqrt{T\log|\mathcal{H}'|}} + \underbrace{\inf_{h'\in\mathcal{H}'} \frac{1}{n}\sum_{i=1}^{n} l(h(x_i), y_i) - \inf_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} l(h(x_i), y_i)}_{\text{Approximation error } A_n}$$

Since $\mathcal{H}$ is finite, we know by Theorem 5 that the first two terms are upper bounded by $\sqrt{T\log|\mathcal{H}'|}$. The last two terms is known as approximation error (approximation of $\mathcal{H}$ by $\mathcal{H}'$), and we note that as $\mathcal{H}'$ gets larger, the approximation error gets smaller but the upper bound of the first two terms gets more loose. So the main challenge is to understand the tradeoff from the size of $\mathcal{H}'$, between the approximation error and the regret from finite $\mathcal{H}'$ and try to balance it.

We leave the remainder of the proof as an exercise for the next scribe. □

Lecture 12

The goal of this lecture will be to be finish the proof for the regret bound in smoothed online learning. Recall the theorem we are trying to prove,

Theorem 21 (HRS'22)**.** *In smoothed online learning,*

$$\text{Regret} \leq \sqrt{\frac{d \log(\frac{n}{\sigma})}{n}}$$

,

where d is the VC dimension of the function class $\mathcal{H}$ and the VC dimension is a notion of complexity for the offine i.i.d. data setting. We will now prove a slighter looser claim for computational ease.

Proof. Recall the definition of regret:

$$\frac{1}{n}\sum_{i=1}^{n} l(\hat{y}_i, y_i) - \inf_{h \in \mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} l(h(x_i), y_i)$$

Let $\mathcal{H}' \subseteq \mathcal{H}$, where $\mathcal{H}'$ is finite. Then we decompose the regret by adding and subtracting the optimal risk under $\mathcal{H}'$:

$$\underbrace{\frac{1}{n}\sum_{i=1}^{n} l(y_i, \hat{y}_i) - \inf_{h' \in \mathcal{H}'} \frac{1}{n}\sum_{i=1}^{n} l(y_i, h'(x_i))}_{\leq \sqrt{T \log |\mathcal{H}'|}} + \underbrace{\inf_{h' \in \mathcal{H}'} \frac{1}{n}\sum_{i=1}^{n} l(y_i, h'(x_i)) - \inf_{h \in \mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} l(y_i, h(x_i))}_{\text{Approximation error } A_n}$$

We will look at the normalized version of this regret bound and take the expectation to get:

$$\leq \sqrt{\frac{\log |\mathcal{H}'|}{n}} + \underbrace{\mathbb{E}\left[\overbrace{\inf_{h' \in \mathcal{H}'} \frac{1}{n}\sum_{i=1}^{n} l(y_i, h'(x_i))}^{\text{Best expert in finite subclass}} - \overbrace{\inf_{h \in \mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} l(y_i, h(x_i))}^{\text{Best expert}}\right]}_{\text{Approximation error } A_n}$$

In the last lecture, we proved the bound on the first term. Now we will focus on A_n. First, we can bring out the negative infimum operation to get a supremum upper bound on A_n.

$$A_n \leq \frac{1}{n}\mathbb{E}\left[\sup_{h\in\mathcal{H}}\inf_{h'\in\mathcal{H}'}\sum_{i=1}^{n} l(y_i, h'(x_i)) - l(y_i, h(x_i))\right]$$

Next, we can take the loss function to be 0/1-loss.

$$\leq \frac{1}{n}\mathbb{E}\left[\sup_{h\in\mathcal{H}}\inf_{h'\in\mathcal{H}'}\sum_{i=1}^{n} \mathrm{I}(y_i \neq h'(x_i)) - \mathrm{I}(y_i \neq h(x_i))\right]$$
$$\leq \frac{1}{n}\mathbb{E}\left[\sup_{h\in\mathcal{H}}\inf_{h'\in\mathcal{H}'}\sum_{i=1}^{n} \mathrm{I}(h'(x_i) \neq h(x_i))\right]$$

We fix a single x_i and by the definition of a smoothed distribution, we have

$$\mathbb{E}_{x_i\sim D_i}\left[\mathrm{I}(h(x_t) \neq h'(x_t)\right] \leq \frac{1}{\sigma}\mathbb{E}_{x\sim\mu}\left[\mathrm{I}(h(x) \neq h'(x))\right]$$

Before we continue with the proof, let's define a cover (or net) as it will be useful to finishing the proof.

Definition 16 (Cover/Net)**.** $\mathcal{H}'$ is an ϵ-cover for $\mathcal{H}$ under μ, if for all $h \in \mathcal{H} \exists h' \in \mathcal{H}'$,

$$\mathbb{E}_{x\sim\mu}\left[I(h(x) \neq h'(x))\right] \leq \epsilon$$

The last part of this proof requires us to take $\mathcal{H}'$ to be a cover for $\mathcal{H}$ under base measure μ. Let

$$G = \{x \mapsto h\Delta h' : h' \text{is close to } h \text{ under } \mu'\}$$

$h\Delta h'$ is the symmetric difference between hypothesis classes: $\mathrm{I}(h(x) \neq h'(x))$. Note that G is a set of functions.

$$A_n \leq \frac{1}{n}\mathbb{E}_{x_i\sim D_i}\left[\sup_{g\in\mathcal{G}}\sum_{i=1}^{n} g(x_i)\right]$$

We want to decompose the above supremum and we will use the coupling lemma to help us do it.

Lemma 22 (Coupling Lemma)**.** *For any smoothed distribution D and any k,*

$$\exists \Pi \text{ on } X^{k+1} \text{ such that } (X, Z_1, \ldots, Z_k) \sim \Pi$$

satisfies

- $X \sim D$
- $(Z_1, \ldots, Z_k) \overset{iid}{\sim} \mu_k$
- *With probability* $1 - (1-\sigma)^k$, $X \in \{Z_1, \ldots, Z_p\}$

$$\begin{aligned}
\frac{1}{n}\mathbb{E}_{x_i\sim D_i}\left[\sup_{g\in\mathcal{G}}\sum_{i=1}^{n} g(x_i)\right] &= \frac{1}{n}\mathbb{E}_{\Pi_i}\left[\sup_{g\in\mathcal{G}}\sum_{i=1}^{n} g(x_i)\right] \\
&\leq \frac{1}{n}\mathbb{E}_{\Pi_i}\left[\sup_{g\in\mathcal{G}}\sum_{i=1}^{n}\sum_{j=1}^{k} g(z_j^{(i)})\right] + (1-\sigma)^k\cdot n \\
&\leq \sup_{g\in\mathcal{G}}\frac{1}{n}\mathbb{E}_{\Pi_i}\left[\sum_{t=1}^{n} g(x_i)\right] + (1-\sigma)^k\cdot n \\
&\leq \frac{\epsilon}{\sigma} + (1-\sigma)^k\cdot n
\end{aligned}$$

□

Now we will state two important facts to help us put all the pieces together to get a bound on the regret.

Fact 23. *$\mathcal{G}$ has a VC dimension O(d), thus by the VC theorem, we have that*

$$A_n \leq \frac{\epsilon}{\sigma} + \frac{1}{n}\sqrt{n\cdot k\cdot d\cdot\log(n)}$$

Fact 24 (Hausser's Theorem)**.** *For all VC class's $\mathcal{H}$, $\exists$ finite $\mathcal{H}'$ s.t.*

$$|\mathcal{H}'| \leq \left(\frac{100}{\epsilon}\right)^d.$$

This is true for all ϵ, where $\mathcal{H}'$ is an ϵ-cover.

Getting back to our bound for the regret, we have

$$\begin{aligned}
\text{Regret} &\leq \sqrt{\frac{\log|\mathcal{H}'|}{n}} + \frac{1}{n}\mathbb{E}\left[\inf_{h'\in\mathcal{H}'}\sum_{i=1}^{n} l(y_i,h'(x_i)) - \inf_{h\in\mathcal{H}}\sum_{i=1}^{n} l(y_i,h(x_i))\right] \\
&\leq \sqrt{\frac{d\log(1/\epsilon)}{n}} + \frac{\epsilon}{\sigma} + \frac{1}{n}\sqrt{nkd\log(n)}
\end{aligned}$$

If we set $\epsilon = \frac{1}{n}$ and $k \sim \frac{1}{\sigma}$, then we see that last term is greater than the first term and so the first term goes away in our bound. What is left is to do is to set ϵ such that $\frac{\epsilon}{\sigma} = \sqrt{\frac{d\log(n)}{n\sigma}}$. By setting $\epsilon = \sqrt{\frac{d\sigma\log(n)}{n}}$, we get the following regret bound

$$\text{Regret} \leq \sqrt{\frac{d\log(n/\sigma)}{n\sigma}}.$$

This is a simpler bound, but we can obtain the true regret bound but applying Berstein's inequality to **Fact 23** instead of agnostic bounds.

A Brief Introduction to Calibration

We will conclude this lecture by briefly introducing the notion of calibration. A common example starts with a meteorologist who is trying to predict the weather, where it will either rain or not rain on a given day. There is also a predictor that uses past data to model the weather for the next day. In principle, we might try to use a logistic regression as our predictor but that makes strong assumptions that we might not be willing to make. A simple algorithm for forecasting the weather would be to use the weather from today as the predicted weather for tomorrow (i.e. if it rained yesterday, then predict it will rain again tomorrow). While naive, this algorithm at least matches the means. But what if we want more from our algorithm? Let's put our predictions into bins from 0 to 1. We can look at single bin, for example the days in which we predicted a 30% chance of ran. We can then count up the days that it actually rained. We want to ensure that it actually rained for 30% of the days in which we predicted 30%. We can do this for every bin and this is called calibration. We will see later how to develop algorithms that enforce calibration and later we discuss what it means to be multicalibrated.

Lecture 13

In this lecture, we will formally introduce the notion of calibration that enables us to test whether a learning algorithm makes consistent predictions. At a high level, the idea is to ask for *prediction conditioned average consistency* for all prediction buckets. A recent article by Foster and Hart is the main source for the material in this lecture [FH21]. In their own words, they describe calibration as follows:

> Calibration means that forecasts and average realized frequencies are close.

Calibration is not a learning algorithm on its own, instead, the calibration strategies are used for fixing learning algorithms.

Background

One of the earliest techniques used to address similar questions is the notion of approachability. In his seminal work in 1956, David Blackwell introduced approachability as a generalization of zero-sum game theory to vector payoffs [Bla56]. In this game, the row player aims to get payoffs inside a target set while the column player tries to prevent payoffs from being inside this set. Later on, Hart and Mas-Colell showed that calibration can be considered a special case of approachability in their regret-based approach to calibration [HMC00].

Most of the early works on calibration focused on Bayesian settings. However, starting in the 90s, the focus in the literature shifted towards distribution-free approaches for calibration. A surprising result by Foster and Vohra showed that one may always generate forecasts that are guaranteed to be calibrated, no matter what the weather will actually be [FV98]. They showed that these forecasts must necessarily be stochastic; i.e., in each period the forecast is chosen by randomization that may depend on the history.

More recently, the focus is on designing multi-calibration algorithms that can achieve calibration with respect to multiple criteria.

Setup

We write down the model (prediction game) as follows. In each round $t = 1, 2, \ldots, T$:

- The prediction player predicts some probability $p_t \in [0, 1]$.
- The outcome $y_t \in \mathcal{Y} = \{0, 1\}$ is revealed.

Remark. For ease of exposition, we will assume that the predictions are quantized to the set of buckets $\mathcal{C} = \{1/N, 2/N, \ldots, (N-1)/N, 1\}$ for some N we will choose later.

In the example of the weather prediction game,

- The prediction p_t corresponds to a forecast for the probability of rain.
- The outcome y_t corresponds to whether it rains ($y_t = 1$) or it does not ($y_t = 0$).

To define the objective of calibration, we continue with some definitions.

Definition 17. Given a sequence of predictions and outcomes, let

$$n_t(z) = \sum_{s=1}^{t} \mathbb{1}_z(p_s) \tag{45}$$

be the number of rounds up to t on which the prediction was in bucket z. Then, let

$$\bar{y}_t(z) = \frac{1}{n_t(z)} \sum_{s=1}^{t} \mathbb{1}_z(p_s) y_s \tag{46}$$

be the average prediction for a bucket z up to time t. Then, the error for a bucket z is defined as

$$e_t(z) = \bar{y}_t(z) - z. \tag{47}$$

Using these preliminary definitions, we can define ϵ-calibration as follows.

Definition 18. A prediction strategy satisfies ϵ-calibration if

$$\limsup_{t \to \infty} \left(\sup_{y_1, \ldots, y_t} \mathbb{E}[K_t] \right) \leq \epsilon, \tag{48}$$

where

$$K_t = \sum_{z \in \mathcal{C}} \frac{n_t(z)}{t} |e_t(z)| \tag{49}$$

is a weighted sum of prediction errors for all buckets.

Remark. In this definition, the expectation is over the randomness of the algorithm.

A Calibration Algorithm

Start by defining the number of positive predictions

$$r_t(z) = \sum_{s=1}^{t} \mathbb{1}_z(p_s) y_s \tag{50}$$

and the excess number of positive predictions

$$G_t(z) = r_t(z) - z n_t(z) \tag{51}$$
$$= n_t(z) e_t(z) \tag{52}$$

for any bin $z \in \mathcal{C}$. Then, we can write

$$K_t = \frac{1}{t} \sum_{z \in \mathcal{C}} |G_t(z)| \tag{53}$$
$$= \frac{1}{t} \sum_{z \in \mathcal{C}} \sqrt{G_t(z)^2} \tag{54}$$
$$\leq \frac{1}{t} \sqrt{\sum_{z \in \mathcal{C}} G_t(z)^2}. \tag{55}$$

Therefore, if an algorithm can minimize

$$S_t := \sum_{z \in \mathcal{C}} G_t(z)^2, \tag{56}$$

it will imply an upper bound for K_t. To minimize S_t, we consider a telescoping decomposition of $S_t = \sum_{s=1}^{t} (S_s - S_{s-1})$ and aim for minimizing each difference term $S_t - S_{t-1}$ individually. Using the definition of S_t, we can write

$$S_t - S_{t-1} = (G_{t-1}(p_t) + y_t - p_t)^2 - G_{t-1}(p_t)^2 \tag{57}$$
$$= \underbrace{2 G_{t-1}(p_t)(y_t - p_t)}_{=:\Delta_t} + \underbrace{(y_t - p_t)^2}_{\leq 1} \tag{58}$$

Since the second term on the right-hand side is upper bounded by 1, its effect will be negligible compared to Δ_t as we will show later in this lecture. For this reason, we continue our analysis by minimizing the Δ_t term only. Our goal is to achieve $\mathbb{E}\Delta_t \leq 0$ regardless of the selection of y_t. To be able to achieve this goal, we need to choose p_t randomly. We consider a simple form of randomness given as

$$p_t = \begin{cases} z_1 \text{ w.p. } \lambda \\ z_2 \text{ w.p. } 1 - \lambda \end{cases} \tag{59}$$

for some $z_1, z_2 \in \mathcal{C}$ and $\lambda \in [0, 1]$. With this choice of distribution for p_t, we obtain

$$\mathbb{E}\Delta_t = \mathbb{E}\left[2 G_{t-1}(p_t)(y_t - p_t)\right]$$

$$
\begin{aligned}
&= 2\lambda G_{t-1}(z_1)(y_t - z_1) + 2(1-\lambda)G_{t-1}(z_2)(y_t - z_2) \\
&= 2\left(\lambda G_{t-1}(z_1) + (1-\lambda)G_{t-1}(z_2)\right)(y_t - z_2) \\
&\quad + 2\lambda G_{t-1}(z_1)(z_2 - z_1)
\end{aligned}
$$

To achieve $\mathbb{E}\Delta_t \leq 0$ for any y_t, we want choose λ such that

$$\lambda G_{t-1}(z_1) + (1-\lambda)G_{t-1}(z_2) = 0. \tag{60}$$

Then, to minimize $\mathbb{E}\Delta_t$, we pick z_1 and z_2 next to each other such that they satisfy $z_2 - z_1 = 1/N$.

As we will show later in the lecture, the function $z \to G_{t-1}(z)$ always has a down-crossing. Therefore, we can always find λ such that (60) is satisfied and $G_{t-1}(z_1) \leq 0$.

As a result, by making a probabilistic prediction, we are able to hedge against the adversary. With this probabilistic choice of p_t, we are able to achieve $\mathbb{E}\Delta_t \leq 0$.

Minimax-based proof

Set up a zero-sum game between a learner and adversary with costs

$$\ell_t(p_t, y_t) = 2G_{t-1}(p_t)(y_t - p_t) + 1. \tag{61}$$

The learner wants to minimize this cost and the adversary aims for maximizing. The actions y_t of the adversary are from $\mathcal{Y} = \{0, 1\}$ while the actions p_t of the learner are from $\mathcal{C}$.

Start with calculating the value of the game where the adversary acts first:

$$\max_{q_1 \in \Delta_\mathcal{Y}} \min_{q_2 \in \Delta_\mathcal{C}} \mathbb{E}_{y_t \sim q_1, p_t \sim q_2} \left[\ell_t(p_t, y_t)\right] \tag{62}$$

Note that given the adversary's choice, the learner can compute $\mathbb{E}_{Y \sim q_1} Y$. So, the learner outputs a p_t such that $|p_t - \mathbb{E}_{Y \sim q_1} Y| \leq 1/N$. Under this choice of p_t,

$$
\begin{aligned}
\max_{q_1 \in \Delta_\mathcal{Y}} \min_{q_2 \in \mathcal{C}} \mathbb{E}_{y_t \sim q_1, p_t \sim q_2} \left[\ell_t(p_t, y_t)\right] &\leq 2\frac{G_{t-1}}{N} + 1 && (63) \\
&\leq \frac{2t}{N} + 1 && (64)
\end{aligned}
$$

Then, by the minimax theorem,

$$\min_{q_2 \in \Delta_\mathcal{C}} \max_{q_1 \in \Delta_\mathcal{Y}} \mathbb{E}_{y_t \sim q_1, p_t \sim q_2} \left[\ell_t(p_t, y_t)\right] \leq \frac{2t}{N} + 1 \tag{65}$$

Noting that $\ell_t(p_t, y_t) \geq S_t - S_{t-1}$ by construction, we can write

$$\mathbb{E}\left[S_t - S_{t-1}\right] \leq \frac{2t}{N} + 1 \tag{66}$$

As a result,

$$\mathbb{E}\left[S_t\right] \leq \sum_{s=1}^{T}\left(\frac{2s}{N}+1\right) \leq T\left(\frac{2T}{N}+1\right) \tag{67}$$

By choosing $N = T$, we can achieve $\mathbb{E}S_t \in O(T)$. Therefore,

$$\mathbb{E}K_T = \frac{1}{T}\mathbb{E}[\sqrt{S_T}] \tag{68}$$

$$\leq \frac{1}{T}\sqrt{\mathbb{E}S_T} \tag{69}$$

$$\in O\left(\frac{1}{\sqrt{T}}\right) \tag{70}$$

Lecture 14

Much of what we have studied so far was concerned with obtaining good performance guarantees for a single population or situation. The meaning of population has changed across the different settings in which we've been working. For example, when we were talking about statistical learning theory, the population was a distribution $\mathcal{D}$ and the goal was to learn some function h such that the error on the population was small. In online learning, there was one sequence for which we were interested in the average performance from time 1 to T and the loss on (x^t, y^t). Referring to this sequence as a population might be slightly misleading, but the idea is the same.

In either of these cases, we wanted good performance on every person that appears in $\mathcal{X}$ or on each x_t in a sequence. A similar idea holds in both offline and online calibration, where we wanted to achieve a certain level of performance for any member of the population. More formally,

- Statistical learning: Given population $\mathcal{D}$, learn h s.t.

$$L_{\mathcal{D}}(h) \leq \epsilon + \min_{h^* \in \mathcal{H}} L_{\mathcal{D}}(h^*).$$

- Online learning: Given $\{(x^t, y^t)\}_{t=1}^T$, minimize

$$\frac{1}{T} \sum_{t=1}^{T} \ell(h^t, x^t, y^t).$$

- Calibration for predictor p: Given population $\mathcal{D}$, achieve

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}[y|p(x) = \nu] = \nu \quad \forall \nu.$$

The question we ask today is, are these kinds of guarantees good enough to justify the deployment of these algorithms on populations that are potentially composed of different sub-populations? If not, how should we modify them so that they're more robust and more useful for different individuals or groups that are represented in that population?

The issue with these guarantees is that they apply to the average case. For an example of why this can be undesirable, assume that we have a distribution of people such that 10% are elderly and 90% are younger people. Suppose we are predicting a heart condition and have 5% error on the whole population.

One possible situation is that our classifier has 5% error in either of these populations. That would be pretty good if true. However, it could be that our classifier achieves 0% error on the large population, younger people, and 50% error on the smaller population comprised of the elderly. If we use this classifier to make any decisions for the elderly, who are most at risk, our predictions will be terrible.

Next, we consider an example from calibration. Refer to the above definition for what it means for a predictor to be calibrated for a distribution $\mathcal{D}$. One way that a predictor p could be calibrated is by defining

$$p(x) := \mathbb{E}_{(x,y)\sim\mathcal{D}}[y].$$

This predictor is trivially calibrated because it only makes a single prediction, the expected value of y. However, it considers only the average case and may actually perform terribly on all sub-populations. For both calibration and prediction, it would be ideal to have

$$p(x) := \mathbb{E}_{(x,y)\sim\mathcal{D}}[y|x]$$

Unfortunately, this is far too fine-grained. If the distribution itself doesn't have a really nice structure, we're not going to be able to learn this predictor from our samples. We want to find a middle ground between averaging and point-wise guarantees.

For the next several lectures, we will be discussing the *multi-objective learning* setting, wherein we seek to learn a function h such that it performs well for several (sub-)populations, perhaps even as captured by different loss functions. We'll start by discussing two different ways that we might want to describe this phenomenon.

Two ways to capture multiple (sub-)populations

1. Multiple distributions $\mathcal{D}_1, ..., \mathcal{D}_k$ and respective loss functions $L_{\mathcal{D}_i}(h)$.

2. Single distribution $\mathcal{D}$ and groups of features $\mathcal{G} = \{\mathcal{G}_1, ..., \mathcal{G}_r\}$, with $\mathcal{G}_i \in 2^{\mathcal{X}}$.

In the first perspective, the population distributions are unknown, but we can directly sample from each $\mathcal{D}_i$. For example, our two distributions might be concerned with people in Berkeley and Boston.

If we want more information regarding one or the other, we can directly go to these cities and be sure of which distribution the data is coming from.

The second perspective leads to implicit conditional distributions $\mathcal{D}'_1, ..., \mathcal{D}'_k$, as opposed to the explicit distributions seen from the first perspective. These implicit distributions are given by $\mathcal{D}'_j = \mathcal{D}|\{x|x \in \mathcal{G}_j\}$. In order to sample from $\mathcal{D}'_j$, we must sample $(x, y) \sim \mathcal{D}$ and reject $x \notin \mathcal{G}_j$. As a result, the two perspectives have different interpretations of sample complexity.

Next, we take a closer look at the loss functions. In the first regime, the loss functions $L_{\mathcal{D}_j}$ are provided along with their respective distributions. In the second regime, we are interested in $L_{\mathcal{D}'_j}(h)$, which is more useful when there is no direct access to $\mathcal{D}_j$. This perspective more naturally gives rise to different interpretation of loss functions. Let us define

$$L^j_{\mathcal{D}}(h) := \mathbb{E}_{(x,y)\sim\mathcal{D}}[\underbrace{\ell(h, x, y) \cdot \mathbf{1}_{\{x\in\mathcal{G}_j\}}}_{\ell^j(h,x,y)}] = \underbrace{\mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(h, x, y)|x \in \mathcal{G}_j]}_{L_{\mathcal{D}'_j}(h)} \cdot \mathcal{P}[x \in \mathcal{G}_j].$$

With this in hand, we can describe the two equivalent learning problems more formally.

1. Given unknown $\mathcal{D}_1, ..., \mathcal{D}_k$, learn h s.t. w.h.p.

$$\max_{i\in[k]} L_{\mathcal{D}_i}(h) < \epsilon + \min_{h^*\in\mathcal{H}} \max_{i\in[k]} L_{\mathcal{D}_i}(h^*).$$

2. Given unknown $\mathcal{D}$ and losses $\ell_1, ..., \ell_r$, learn h s.t. w.h.p.

$$\max_{j\in[r]} L^j_{\mathcal{D}}(h) < \epsilon + \min_{h^*\in\mathcal{H}} \max_{j\in[r]} L^j_{\mathcal{D}}(h^*).$$

We can see that there is a very subtle difference between the settings with multiple losses versus multiple distributions. In fact, sometimes the best way to represent a multi-objective problem is to assume that you have both multiple losses and multiple distributions. With this in mind, let's define what multi-objective learning is really about.

Multi-objective learning

Given unknown $\mathcal{D}_1, ..., \mathcal{D}_k$ and known $\ell^1, ..., \ell^r$ find h s.t.

$$\max_{j\in[r]} \max_{i\in[k]} L^j_{\mathcal{D}_i}(h) < \epsilon + \min_{h^*\in\mathcal{H}} \max_{j\in[r]} \max_{i\in[k]} L^j_{\mathcal{D}_i}(h^*),$$

where $L^j_{\mathcal{D}_i}(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}_i}[\ell^j(h, x, y)]$.

In online or offline settings with singe distributions, we were primarily concerned with the rate at which our performance improved.

In other words, we cared about how fast our regret would vanish or how many samples we needed to get our error to fall under ϵ. We're asking exactly the same fundamental question here. How many samples from $\mathcal{D}_1, ..., \mathcal{D}_k$ are sufficient? As a complementary question, what learning algorithm would actually find an h that achieves the above guarantees?

Before we directly address these questions, we briefly revisit the single-distribution setting. Recall the sample complexity of learning from a single distribution $\mathcal{D}$ for a class $\mathcal{H}$ with $VC(\mathcal{H}) = d$. If given

$$S \subset \mathcal{H} \text{ s.t. } |S| = m_{\epsilon,\delta}^{single} = \mathcal{O}\left(\frac{d + \log(1/\delta)}{\epsilon^2}\right),$$

then uniform convergence guarantees that w.p. $1 - \delta$,

$$|L_{\mathcal{D}}(h) - L_S(h)| < \frac{\epsilon}{2} \ \ \forall h \in \mathcal{H}.$$

Hence, for any output of an empirical risk minimizing algorithm, $h_S = \underset{h \in \mathcal{H}}{\operatorname{argmin}} L_S(h)$, we have

$$L_{\mathcal{D}}(h_S) < \epsilon + \min_{h^* \in \mathcal{H}} L_{\mathcal{D}}(h_S).$$

Going forward into the multi-objective setting, we make two assumptions to facilitate the process:

1. Simplifying: single loss L but different distributions $\mathcal{D}_1, ..., \mathcal{D}_k$.
2. Realizability: $\exists h^* \in \mathcal{H}$ such that $\forall i \in [k]$, $L_{\mathcal{D}_i}(h^*) = 0$.

The following results can easily be extended to situations with multiple losses. In the single distribution case, the second assumption gives us a new sample complexity:

$$S \subset \mathcal{H} \text{ s.t. } |S| = m_{\epsilon,\delta}^{single} = \mathcal{O}\left(\frac{d \log(1/\epsilon) + \log(1/\delta)}{\epsilon}\right)$$
$$\Rightarrow \ \text{ w.p. } 1 - \delta, \ \forall h \in \mathcal{H} \text{ s.t. } L_S(h) = 0, \ L_{\mathcal{D}}(h) < \frac{\epsilon}{2}.$$

We also consider a notion of consistency that is somewhat strong. Namely, that there exists a function that is an optimal classifier for every distribution.

For our initial attempt at multi-objective learning, we will just copy what we did for single distributions. Consider sample sets $S_1, ..., S_k$ s.t. $S_i \sim \mathcal{D}_i^{m_i}$. Define the reunion by

$$\overline{S} = \bigcup_{i=1}^{k} S_i.$$

In this setting, we let the algorithm return any $h_{\overline{S}}$ such that $L_{\overline{S}}(h_{\overline{S}}) = 0$. Observe that this is still consistent because our hypothesis achieves

zero loss on all k empirical loss functions. Recall that in the single distribution case, we had $S \sim \mathcal{D}^m$ and selected h_S s.t. $L_S(h_S) = 0$.

We claim that if $m_i = m^{single}_{\epsilon,\frac{\delta}{k}}$, then w.p. $1-\delta$,

$$L_{\mathcal{D}_i}(h_{\overline{S}}) \leq \epsilon \ \ \forall i.$$

Proof. By nature of our sample complexity $m^{single}_{\epsilon,\frac{\delta}{k}}$, we know that for each distribution, we will observe $L_{\mathcal{D}_i}(h) > \epsilon$ with probability at most δ/k. Hence, a union bound argument gives us

$$\begin{aligned}\mathbb{P}\Big(\bigcup_{i=1}^{k}\{L_{\mathcal{D}_i}(h) > \epsilon\}\Big) &\leq \sum_{i=1}^{k}\mathbb{P}(L_{\mathcal{D}_i}(h) > \epsilon) \\ &< \sum_{i=1}^{k}\frac{\delta}{k} = \delta.\end{aligned}$$

More succinctly, with probability at most δ,

$$\exists i \in [k] \text{ s.t. } L_{\mathcal{D}_i}(h_{\overline{S}}) > \epsilon,$$

which is the negation of our claim. □

The result of this approach is that the sample complexity is the same as doing each sub-population independently:

$$m = km^{single}_{\epsilon,\frac{\delta}{k}} \approx \frac{kd}{\epsilon}...$$

This is too large due to the fact that we are paying for the complexity of two things at the same time: the complexity of the class ($d = VC(\mathcal{H})$) and the number of sub-populations k. For example, consider a consulting problem where we are solving a universal problem among several different cities, each with their own data set. We would hope that if they collaborated with another, they would use fewer samples than if we addressed each individually.

Next time, we will show that it is possible to attain

$$m \approx \frac{\log(k)d}{\epsilon}.$$

This is faster than k individual distributions, yet slower than the setting with a single distribution.

With the former approach, we are suffering from a lack of adaptivity. We took the same number of samples no matter how difficult a distribution was compared to the average or the worst case. If we want to save samples, we should not ignore this aspect of the problem. Instead, we should consider each distribution's difficulty in the context of every other distribution, which necessitates a more adaptive, interactive way of taking samples.

We've previously discussed such notions in the online learning setting, where both our algorithm and the adversary are adapted. Instead of talking about adaptivity, we can talk about the thing we use adaptivity for. Recall that if we could learn an adaptive setting, that meant that we knew how to design a no-regret algorithm, and then we could solve the min-max problem. In fact, we solved the minimized problem using a no-regret algorithm with another no-regret, or with a best-response. With this in mind, we look to our definition of a multi-objective learning problem and ask, is there something there that looks like min-max? Indeed, we can already see both a minimum and maximum.

$$\min_{i\in[k]} L_{\mathcal{D}_i}(h) \leq \underbrace{\min_{h^*\in\mathcal{H}}}_{(i)} \underbrace{\max_{i\in[k]}}_{(ii)} L_{\mathcal{D}_i}(h) + \epsilon.$$

The minimizer, (i), can be viewed as the best-response algorithm, and the maximizer, (ii), can be seen as a no-regret algorithm. We are essentially trying to find an $\epsilon-$approximate notion of an equilibrium; any h that satisfies this inequality is an ϵ-optimal min-max strategy for the learner. In essence, our goal is now the min-max equilibrium of a game defined by the loss table $L_{\mathcal{D}_i}$, where $\mathcal{D}_i$ is unknown. Hence, the game is unknown, but we can sample $(x, y) \sim \mathcal{D}_i$ to get a sense of the game. The next question is, how many samples do we need to find the min-max equilibria of a game? We will address this in the next lecture.

Lecture 15

Recall that we were looking for a hypothesis h that satisfies the following inequality.

$$\max_{j\in[r]}\max_{i\in[k]} L^j_{\mathcal{D}_i}(h) \le \min_{h^*\in\mathcal{H}}\left(\max_{j\in[r]}\max_{i\in[k]} L^j_{\mathcal{D}_i}(h^*)\right)+\epsilon$$

In this lecture, we focus on multi-distribution setting where we only have a single loss function. This reduces the above inequality into the following condition.

$$\max_{i\in[k]} L_{\mathcal{D}_i}(h) \le \underbrace{\min_{h^*\in\mathcal{H}}}_{(i)}\underbrace{\max_{i\in[k]}}_{(ii)} L_{\mathcal{D}_i}(h^*)+\epsilon$$

From the last lecture, we saw that, under realizable setting, $m^{multiple}_{\epsilon,\delta} = km^{single}_{\epsilon,\delta/k} \approx \frac{kd}{\epsilon}$. In fact, this complexity is unavoidable with non-adaptive algorithm. Today, we introduce adaptive algorithms using the game dynamics we discussed at the end of the last class to reduce the sample complexity. In particular, we want the sample complexity to be depend on $\log k$ by a logarithmic factor, $\log k \cdot VC/poly(\epsilon)$. There are three ways to introduce game dynamic into this problem.

Player 1 (min player)	Player 2 (max player)
Best Response Alg.	No-regret Alg.
No-regret Alg.	No-regret Alg.
No-regret Alg.	Best response Alg.

For all the cases, player 1 is a min player who is minimizing the loss by choosing a hypothesis and player 2 is a max player who is maximizing the loss by choosing a distribution. Before we start, we should note that $L_{\mathcal{D}_i}(h)$ is unknown, which indicates that we don't have a game table. This means that we should take samples to estimate these values. In this sense, our sample complexity is a product of the following two factors.

- How quickly no-regret/best response dynamics to converge to $\epsilon-$ approximated equillibria $\left(\frac{Regret_1}{T}+\frac{Regret_2}{T}\right)$

- Per step number of samples needed for estimating the loss $L_{\mathcal{D}_i}(h)$

Best response - No regret scheme

Player 1 use empirical risk minimization (ERM) for best-response algorithm and player 2 will use random weighted majority (RWM) or hedge algorithm for no-regret algorithm. For simplicity, we work with the realizability assumption (although the analysis carry through without this assumptions too), which means there exists h^* such that $\min_{h^* \in \mathcal{H}} L_{\mathcal{D}_i}(h^*) = 0$ for all $i \in [k]$. The details of the algorithm are as follows.

Algorithm 7: Best response (ERM) - No regret (Hedge)

Initialize weights $\beta^0 = (\frac{1}{k}, \frac{1}{k}, ..., \frac{1}{k})$ where each entry is associated with each of the distributions $\mathcal{D}_i$, $i \in [k]$.

for $t \in \{1, 2, \ldots, T\}$ **do**

Define $P^t := \sum_{i=1}^{k} \beta_i^{t-1} \mathcal{D}_i$, a mixture distribution over $\mathcal{X} \times \mathcal{Y}$.

Player 1 (min)

Uses $\Theta(\frac{VC(\mathcal{H})}{\epsilon'^2})$ samples from P^t to find the best response h^t that satisfies $L_{P^t}(h^t) \leq \min_{h^* \in \mathcal{H}} L_{P^t}(h^*) + \epsilon' \underset{Realizable}{=} \epsilon'$.

Player 2 (max)

For each $i \in [k]$, using $\Theta(\frac{1}{\epsilon'^2})$ samples, estimate $L_{\mathcal{D}_i}(h^t)$ empirically. We get $|L_{\mathcal{D}_i}(h^t) - \widehat{L}_{\mathcal{D}_i}(h^t)| \leq \epsilon'$ w.h.p from Hoeffding's inequality.

Update $\beta_i^t \propto \beta_i^{t-1} \cdot \exp(\widehat{L}_{\mathcal{D}_i}(h^t) \cdot \eta)$ (Hedge update)

end for

Output $\bar{h} \sim \text{Unif}(h^1, h^2, ..., h^T)$

Now we analyze the algorithm and find proper ϵ' that can lead to the following objective.

$$\max_{i \in [k]} \underbrace{L_{\mathcal{D}_i}(\bar{h})}_{\mathbb{E}_{h \sim \bar{h}}[L_{\mathcal{D}_i}(h)]} \leq \min_{h^* \in \mathcal{H}} \max_{i \in [k]} L_{\mathcal{D}_i}(h^*) + \epsilon \underset{Realizable}{=} \epsilon \tag{71}$$

From the properties of ERM and hedge algorithm, we obtain the following inequality.

$$\epsilon' \underset{ERM}{\geq} \frac{1}{T} \sum_{t=1}^{T} L_{P^t}(h^t) \underset{Hedge}{\geq} \max_{i \in [k]} \underbrace{\frac{1}{T} \sum_{t=1}^{T} L_{\mathcal{D}_i}(h^t)}_{\mathbb{E}_{h \sim \bar{h}}[L_{\mathcal{D}_i}(h)]} - \frac{Regret}{T} - \epsilon'$$

Since $Regret \leq \sqrt{T \log k}$, setting $T = \frac{1}{\epsilon'^2} \log k$ and $\epsilon' = \frac{\epsilon}{3}$ lead to the desired inequality (71). Hence, the sample complexity of the

algorithm is as follows.

$$m_\epsilon^{multiple} = \frac{1}{\epsilon'^2} \cdot \frac{k + VC(\mathcal{H})}{\epsilon'^2} = \mathcal{O}\left(\frac{VC(\mathcal{H}) \log k}{\epsilon^4} + \frac{k}{\epsilon^4}\right) \tag{72}$$

It is worth noting that under the realizability assumption, the dependency on ϵ can be reduced since ERM sample complexity under realizability is $\mathcal{O}(\frac{VC(\mathcal{H})}{\epsilon})$ instead of $\mathcal{O}(\frac{VC(\mathcal{H})}{\epsilon^2})$. Moreover, under the assumption, we can make $\bar{h}$ deterministic instead of random. Majority Vote$(h^1, h^2, ..., h^T)$ will also satisfy the inequality (71).

However, in this algorithm, we are drawing too many samples each time. In fact per step closedness, i.e. every function we are estimating in each step is $\epsilon-$accurate. As this need not to be achieved, we can opt out from per step closedness and take less samples at each step. In this sense, we introduce the algorithm in which both player play with no-regret algorithms.

No regret - No regret scheme

Algorithm 8: No regret (Hedge) - No regret (Exp3)

Initialize weights $\alpha^1 = (\frac{1}{|\mathcal{H}|}, \frac{1}{|\mathcal{H}|}, ..., \frac{1}{|\mathcal{H}|})$ and $\beta^1 = (\frac{1}{k}, \frac{1}{k}, ..., \frac{1}{k})$ where each entry is associated with each hypotheses $h \in \mathcal{H}$ and distributions $\mathcal{D}_i$, $i \in [k]$, respectively.

for $t \in \{1, 2, \ldots, T\}$ **do**

Player 1 (min)
Output α^t, which only depends on $\beta^1, ..., \beta^{t-1}$.
Sample $i_t \sim \beta^t$, $z_{i_t}^t := (x_{i_t}^t, y_{i_t}^t) \sim \mathcal{D}_{i_t}$.
Update $\alpha_h^{t+1} \propto \alpha_h^t \exp(-\eta \cdot l(h, z_{i_t}^t))$ for all $h \in \mathcal{H}$.

Player 2 (max)
Output β^t, which only depends on $\alpha^1, ..., \alpha^{t-1}$.
Sample $i_t \sim \beta^t$, $z_{i_t}^t := (x_{i_t}^t, y_{i_t}^t) \sim \mathcal{D}_{i_t}$.
Update $\beta_{i_t}^{t+1} \propto \beta_{i_t}^t \exp(\eta \cdot l(\alpha^t, z_{i_t}^t)/\beta_{i_t}^t)$. [2]

end for

Output $\bar{h} \sim \text{Unif}(h^1, h^2, ..., h^T)$

[2] Technically, to get a high probability regret bound for Exp3, unlike Hedge, we need a variant of Exp3 with some additional regularizing terms. One option is to utilize Exp3-IX from [Neu15]

The major difference between the players is that player 1 is playing the game with full information scheme while player 2 is playing the game with partial information scheme. This comes from the fact that player 1 can still estimate the loss for all hypotheses by only using a single sample $z_{i_t}^t$, but player 2 cannot estimate the loss for a distribution $\mathcal{D}_j$ if $i^t \neq j$ and hence $z_{i^t}^t$ was not sampled from $\mathcal{D}_j$. Therefore, we only update one β_{i_t} per step with additional factor

$1/\beta^t_{i^t}$, which is likewise the Exp 3 algorithm. Meanwhile, note that α^t is chosen independent of β^t and β^t is chosen independent of α^t. This allows us to use Hoeffding's inequality or martingales concentration inequality to prove the regret bound.

Since we have regret bounds for population regret which we learned in the previous lectures. It remains to prove that the regret of the single-sample scheme actually converges to the population regret. For simplicity, we denote $\mathbb{E}_{h\sim\alpha^t}[l(h, z^t_{i_t})]$ and $\mathbb{E}_{i\sim\beta^t, z^t_{i_t}\sim\mathcal{D}_i}[l(h, z^t_{i_t})]$ by $l(\alpha^t, z^t_{i_t})$ and $L_{\beta^t}(h)$, respectively. Empirical regret and population regret can be written as the following.

$$Reg_1\left(\{\alpha^t, \beta^t\}_{t=1}^T\right) = \sum_{t=1}^T L_{\beta^t}(\alpha^t) - \min_{h^*\in\mathcal{H}} \sum_{t=1}^T L_{\beta^t}(h^*) \leq \sqrt{T\log(\mathcal{H})}$$

$$Reg_2\left(\{\alpha^t, \beta^t\}_{t=1}^T\right) = \max_{i^*\in[k]} \sum_{t=1}^T L_{D_{i^*}}(\alpha^t) - \sum_{t=1}^T L_{\beta^t}(\alpha^t) \leq \sqrt{kT\log(k)}$$

$$\widehat{Reg}_1\left(\{\alpha^t, \beta^t, z^t_i\}_{t\in[T], i\in[k]}\right) = \sum_{t=1}^T l(\alpha^t, z^t_{i_t}) - \min_{h^*\in\mathcal{H}} \sum_{t=1}^T l(h^*, z^t_{i_t})$$

$$\widehat{Reg}_2\left(\{\alpha^t, \beta^t, z^t_i\}_{t\in[T], i\in[k]}\right) = \max_{i^*\in[k]} \sum_{t=1}^T l(\alpha^t, z^t_{i^*}) - \sum_{t=1}^T l(\alpha^t, z^t_{i_t})$$

From these, we would like to prove that with probability $1-\delta$, the following holds.

$$\left|\widehat{Reg}_1\left(\{\alpha^t, \beta^t, z^t\}_{t\in[T], i\in[k]}\right) - Reg_1\left(\{\alpha^t, \beta^t\}_{t=1}^T\right)\right| \leq \sqrt{T\log\frac{|\mathcal{H}|}{\delta}}$$

$$\left|\widehat{Reg}_2\left(\{\alpha^t, \beta^t, z^t\}_{t\in[T], i\in[k]}\right) - Reg_2\left(\{\alpha^t, \beta^t\}_{t=1}^T\right)\right| \leq \sqrt{T\log\frac{k}{\delta}}$$

Only a sketch of proof was discussed in the lecture. For player 1, we will bound the following two terms separately.

$$\left|\sum_{t=1}^T \left(l(\alpha^t, z^t_{i_t}) - L_{\beta^t}(\alpha^t)\right)\right| \text{ and } \left|\min_{h^*\in\mathcal{H}} \sum_{t=1}^T l(h^*, z^t_{i_t}) - \min_{h^*\in\mathcal{H}} \sum_{t=1}^T L_{\beta^t}(h^*)\right|$$

For the first term, note that the player 1 chose $z^t_{i_t}$ *after* α^t was chosen. This implies that $l(\alpha^t, z^t_{i_t})$ is an unbiased estimator of $L_{\beta^t}(\alpha^t)$, i.e. $\mathbb{E}_{i_t\sim\beta^t, z^t_{i_t}\sim\mathcal{D}_{i_t}}[l(\alpha^t, z^t_{i_t})] = L_{\beta^t}(\alpha^t)$. Thus, we can use Hoeffding's inequality to bound the first term. For the second term, we will use a union bound argument along with Hoeffding's inequality used for the first term. We can bound the two terms for player 2 in a similar way. Martingale concentration might be required to bound the second term for player 2.

Putting all the results together, we want our $T \approx \frac{2}{\epsilon^2}(k + \log|\mathcal{H}|)$ to satisfy $Reg_1 \leq \sqrt{T\log(\mathcal{H})} \leq \frac{T\epsilon}{2}$ and $Reg_2 \leq \sqrt{kT\log(k)} \leq \frac{T\epsilon}{2}$.

Since at each time step we take 2 samples, this leads to the sample complexity of

$$m_{\epsilon}^{multiple} \approx \frac{2}{\epsilon^2}(k + \log|\mathcal{H}|).$$

The lecture concluded by introducing an open problem for further exploration. On the first game scheme where each player deploys best-response algorithm and no-regret algorithm, respectively, the sample complexity is $\mathcal{O}\left(\frac{1}{\epsilon^4}(VC(\mathcal{H})\log k + k)\right)$. On the second game scheme, the sample complexity is $\mathcal{O}\left(\frac{1}{\epsilon^2}(\log|\mathcal{H}| + k)\right)$. The second sample complexity is useless if $\mathcal{H}$ is infinite, but we can use $\epsilon-$net of $\mathcal{H}$ to further refine the argument. However, we might lose some factor of ϵ while dealing with the ϵ-net. In this regard, we can think of following open problem which hasn't been solved: Is there any way to equalize two sample complexity by restricting our problem onto a specific $\mathcal{H}$ or onto a specific setting?

Lecture 16

In the last lecture, we work on the single loss function version of the multi-objective learning:

$$\max_{j\in[r]}\max_{i\in[k]} L^j_{\mathcal{D}_i}(h) \le \min_{h^*\in\mathcal{H}}\left(\max_{j\in[r]}\max_{i\in[k]} L^j_{\mathcal{D}_i}(h^*)\right)+\epsilon.$$

Recall in the last lecture, there are three ways to introduce game dynamic into this problem:

1. The min player uses Best-response, and the max player uses No-regret. The sample complexity of multi-distribution learning is $\frac{VCD(\mathcal{H})\cdot\log(k)}{\epsilon^4}+\frac{k\log(k)}{\epsilon^2}$.

2. The min player uses No-regret, and the max player uses No-regret. The sample complexity of multi-distribution learning is $\frac{\log(\mathcal{H})+k}{\epsilon^2}$.

3. The min player uses No-regret, and the max player uses Best-response.

In this lecture, we will first investigate the incentives in multi-distribution learning, which is a relatively under-explored direction. Secondly, we will turn our focus from multi-objective learning to multi-calibration.

So far, the motivation of collaboration we see from previous lectures is achieving the low per-agent loss in the sense of small total number of samples used. Yet, it is equivalently important that we also have small number of samples per-agent. The importance of this perspective arrives in applications such as any physical domain, privacy, and data processing. More precisely, this means that

- Every agent i receives low cost, i.e., $L_{\mathcal{D}_i}(h)\le\epsilon$.

- The number of samples provided by $\mathcal{D}_i$ is "reasonably/desirably small".

Mathematically, we seek to find an algorithm Alg that solves the

following optimization problem

$$\begin{aligned} &\min_i \sum_{i=1}^{k} m_i \\ &\text{s.t. } \mathbb{E}_{m_i, S_i \sim \mathcal{D}_i}\left[L_{\mathcal{D}_i}\left(\text{Alg}(S_1 \cup \cdots \cup S_k)\right)\right] \leq \epsilon, \forall i \\ &\quad \text{Agent } i \text{ is "happy" with } (m_i, m_{-i}), \forall i, \end{aligned} \tag{73}$$

where m_i denotes the number of samples needed from $\mathcal{D}_i$. We denote the expected loss for agent i in (73) as $\text{Err}_i(m_i, m_{-i})$. There are two questions we are interested in:

1. **Feasibility:** Does an allocation $m = (m_1, \ldots, m_k)$ exists that satisfies program (73)?

2. What's the impact of requiring m to satisfy the per-agent happiness condition?

We first define two conditions when the players are obviously unhappy:

1. Not happy with (m_i, m_{-i}) if $\exists \overline{m} \leq m_i$ and $\text{Err}_i(\overline{m}, 0) \leq \epsilon$.

2. Not happy with (m_i, m_{-i}) if $\exists \overline{m} \leq m_i$ and $\text{Err}_i(\overline{m}, m_{-i}) \leq \epsilon$.

Based on the above two criteria, we define two concepts: *individual rationality* and *stable equilibrium*.

Definition 19 (Individual Rationality)**.** Allocation $\mathbf{m} = (m_1, \ldots, m_k)$ is individual rational (IR) if $\forall i$ and $\overline{m_i} \leq m_i$, we have $\text{Err}_i(\overline{m_i}, 0) > \epsilon$.

Individual rationality refers to the idea that joining the collaboration never worse for an agent than solving the task independently. This makes sure that the agent has no incentive not to join a collaboration. Further, we would also like to guarantee that no agent has an incentive to deviate from the prescribed protocol. This is formally defined in the concept of stable equilibrium, which resembles the ides of Nash equilibrium.

Definition 20 (Stable Equilibrium)**.** Allocation $\mathbf{m} = (m_1, \ldots, m_k)$ is stable if $\forall i$ and $\overline{m_i} \leq m_i$, we have $\text{Err}_i(\overline{m_i}, m_{-i}) > \epsilon$.

Stable equilibrium essentially says that while fixing everyone else, agent i has no incentive to deviate.

These new constraints lead to the following questions:

1. Does an IR allocation always exists? Yes. Let m_i = number of samples D_i needed independently with realization assumption: $\exists h^\star$ such that $L_{D_i}(h^\star) = 0, \ \forall i$.

2. Does a stable allocation always exist? Holds under mild assumption.

Before presenting the following theorem, we define three notations: m^{opt}, m^{IR}, and m^{stable} as follows:

$$\begin{aligned} m^{opt} &= \min_{\mathbf{m}} \sum_{i=i}^{k} m_i \text{ such that Eq. (73) holds.} \\ m^{IR} &= \min_{\mathbf{m}} \sum_{i=i}^{k} m_i \text{ such that Eq. (73) and Definition 19 hold.} \\ m^{stable} &= \min_{\mathbf{m}} \sum_{i=i}^{k} m_i \text{ such that Eq. (73) and Definition 20 hold.} \end{aligned} \tag{74}$$

Theorem 25. *There exists distributions $\mathcal{D}_1, \cdots, \mathcal{D}_k$ and ϵ such that*

$$\frac{m^{IR}}{m^{opt}} \geq \Omega(\sqrt{k}) \text{ and } \frac{m^{stable}}{m^{opt}} \geq \Omega(\sqrt{k}). \tag{75}$$

Proof. **Distributions**: Let D_0 be the total k points, i.e., $D_0 = B_1 \cup B_2 \cup, \ldots, \cup B_k$, and $D_1, \ldots, D_k$ be as demonstrated in Figure 8. The total k points are partitioned into $\sqrt{k}$ blocks, with each block having $\sqrt{k}$ points and denoted by $B_1, \ldots, B_{\sqrt{k}}$. There are $\sqrt{k}$ agents supported on B_i, e.g., $D_1, \ldots, D_{\sqrt{k}}$ contains all points in B_1, and for each D_i for $i \in \{1, 2, \ldots, \sqrt{k}\}$, there are $k - \sqrt{k}$ additional unique points.

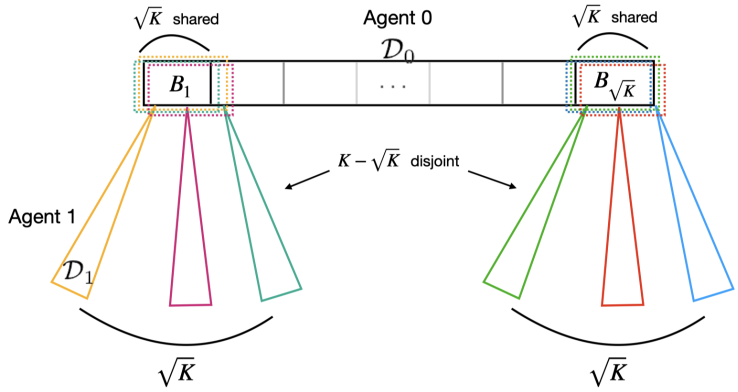


Figure 8: Construction of Distributions

Labelling We label the points according to a function sampled uniformly random from all functions of the type $\{-1, 1\}^{\mathcal{X}}$. That is to say for each $x \in \mathcal{X}$, $f^\star$ sets $f^\star(x) \sim_{\text{unif}} \{-1, +1\}$.

Error Based on the labelling function, the expected error of $D_i = \frac{1}{2} - \frac{\text{the number of uniform data points have seen}}{2k}$. If we set $\epsilon = \frac{1}{2} - \mathcal{O}\left(\frac{1}{k}\right)$. Then we have $\text{Err}_i(\overline{m}, m_{-i}) \leq \epsilon \Leftrightarrow$ observe 1 sample point per distribution.

What makes this construction especially bad for the IR is that, from a purely collaborative standpoint, it makes sense for D_0 to

sample all the time. However, it is not individually ration for it to take any more than one sample. Thus, $m^{IR} \geq \mathcal{O}(k)$.

Formally, we claim that m^{opt} achieves when we sample $\sqrt{k}$ point from D_0. This is because each time D_0 samples, it draws sample from some block B_i with probability $\frac{1}{\sqrt{k}}$. When we sample D_0 for $\sqrt{k}$ times, the expected number if times it lands in block B_i is 1. Hence, the expected error for each D_i satisfies the per agent accuracy (refer to Eq.(73)) with $\sqrt{k}$ samples.

Hence, $\frac{m^{IR}}{m^{opt}} = \Omega(\sqrt{k})$. Since stable is a stricter condition than IR, it naturally holds that $\frac{m^{stable}}{m^{opt}} = \Omega(\sqrt{k})$. □

Multi-calibration

$\mathcal{X}$ is domain/features. $\mathcal{S} \subseteq \{0,1\}^{\mathcal{X}}$ is a subset of the feature space, known as sub-populations.

Predictor $p : \mathcal{X} \to \{0,1\}$ is $(\mathcal{S}, \lambda, \epsilon)$ is multi-calibrated(MC), if $\forall S \in \mathcal{S}, v \in \{0, \lambda, 2\lambda, \ldots, 1\}$ buckets (i.e., $p(x) \in v$ if $v - \frac{\lambda}{2} \leq p(x) \leq v + \frac{\lambda}{2}$), we have

$$-\epsilon \leqslant \mathbb{E}[(p(x) - y) \cdot \mathbb{1}_{(p(x) \in v, x \in S)}] \leqslant \epsilon \Rightarrow \forall S, v, \sigma \underset{(x,y)\sim D}{\mathbb{E}}[l^{S,v,\sigma}(p,x,y)] \leq \epsilon. \tag{76}$$

Multi-calibration is an instance of multi-objective learning: $p : \mathcal{X} \to [0,1]$ is MC is its a ϵ-optimal solution (defined in Eq. (76)) to the following multi-objective setting:

1. Single distribution: D
2. Multiple loss function: $l^{S,v,\sigma}(p,(x,y)) = \sigma(p(x) - y) \cdot \mathbb{1}\{p(x) \in v, x \in S\}$ for $\forall S \in \mathcal{S}, v \in \{0, \lambda, 2\lambda, \ldots, 1\},$ and $\omega \in \{-1, 1\}$.

Simple but not optimal approach:

- Max player: No-regret to choose mixture over $l^{S,v,\sigma}$, need single sample $z^t \sim D$.
- Min player: Best-response to choose p^t, need zero sample.

Then $\overline{p} = \frac{1}{T}\sum_t p^t$ is $(S, \lambda, \epsilon)-$multi-calibration.

Theorem 26. *For any ϵ, there is explicit construction for p^t that is independent of D such that*

$$\mathbb{E}_{S,v,\epsilon\sim\beta}\left[\mathbb{E}_{(x,y)\sim D}\left[\ell^{(s}(p,x,y)\right] \mid\right] \leq \epsilon. \tag{77}$$

Proof idea: Implied by Lecture 13, S doesn't play a big role since S

is known. The real deal is to handle unknown $D|x \in S$.

$$\begin{aligned} \max_{\beta \in D(y)} \min_{p^\star} L'\left(p^\star, \beta\right) &= \min_{\alpha \in D(x \to [0,1])} \max_{y} L'(\alpha, y) \\ \max_{D \in D(x,y)} \min_{p^\star} L'\left(p^\star, D\right) &= \min_{\alpha} \max_{(x,y)} L'(\alpha, (x,y)), \end{aligned} \tag{78}$$

where we notice that α is independent of D.

Lecture 17

Lecture 18

Lecture 19

Lecture 20

Lecture 21

Lecture 22

Lecture 23

Lecture 24

Lecture 25

Lecture 26

Bibliography

[BDPSS09] Shai Ben-David, Dávid Pál, and Shai Shalev-Shwartz. Agnostic online learning. In *Proceedings of the Conference on Learning Theory (COLT)*, 2009.

[BDR21] Adam Block, Yuval Dagan, and Alexander Rakhlin. Majorizing measures, sequential complexities, and online learning. In Mikhail Belkin and Samory Kpotufe, editors, *Conference on Learning Theory, COLT 2021, 15-19 August 2021, Boulder, Colorado, USA*, volume 134 of *Proceedings of Machine Learning Research*, pages 587–590. PMLR, 2021.

[Bla56] David Blackwell. An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6(1):1–8, Mar 1956.

[BM05] Avrim Blum and Yishay Mansour. From external to internal regret. In *Learning Theory: 18th Annual Conference on Learning Theory, COLT 2005, Bertinoro, Italy, June 27-30, 2005. Proceedings 18*, pages 621–636. Springer, 2005.

[FH21] Dean P. Foster and Sergiu Hart. Forecast hedging and calibration. *Journal of Political Economy*, 129(12):3447–3490, Dec 2021.

[FV97] Dean P Foster and Rakesh V Vohra. Calibrated learning and correlated equilibrium. *Games and Economic Behavior*, 21(1-2):40, 1997.

[FV98] Dean P. Foster and Rakesh V. Vohra. Asymptotic calibration. *Biometrika*, 85(2):379–390, 1998.

[HMC00] Sergiu Hart and Andreu Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68(5):1127–1150, Sep 2000.

[Lit87] Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Mach. Learn.*, 2(4):285–318, 1987.

[Neu15] Gergely Neu. Explore no more: Improved high-probability regret bounds for non-stochastic bandits. *Advances in Neural Information Processing Systems*, 28, 2015.

[NRTV07] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic game theory*. Cambridge university press, 2007.