THEORETICAL MACHINE LEARNING COS 511

LECTURE #11

APRIL 21, 2015

LECTURER: ELAD HAZAN SCRIBE: KIRAN VODRAHALLI

Contents

1 Introduction

Some notes about the final project: due May $17, 2015$, you can work with a partner, there is no lower bound on the size, but an upper bound on the length of the paper.

We are going to relate what we've learned to game theory. There's a lot of interesting work going on, and in fact the regret approach comes from game theory. We are going to talk about solution concepts in games - equilibrium - and I will also relate it to linear optimization, particularlyl the concept of duality. We also will discuss duality theory.

2 Linear Programming

So there is kind of a history of optimization - in the $50s$, Dantzig was thinking about a model for operations during the war, and he came up with a mathematical model called linear programming.

In linear programming, we have variables (which can be thought of as vectors in $\mathbb{R}^n$), and we have constraints - which are linear. In general, we can write this as a matrix inequality

$$A\mathbf{x} \leq b$$

where $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$. Also, it makes sense to assume here that $m \geq n$ - otherwise it's too easy - it's underdetermined, and you can always solve it and get an unconstrained minimization problem, you get infinite solutions.

Example 2.1. Max-flow.
Routing in a graph, or maximizing flow in a graph can be written as a linear program. Here we will write the max-flow problem. Given $G = (V, E)$, $\mathbf{x} \in \mathbb{R}^{|E|}$, $0 \leq x_e \leq 1$ (think of $\mathbf{x}$ as the flow in the graph, then this is a capacity constraint). The source and sink are given by s, t. Then flow conservation is given by $\forall v \notin \{s, t\}$ $\sum_{v \in e} x_e = 0$. We also have $\sum_{s \in e} \leq 1$, $\sum_{t \in e} x_e \leq 1$ (we don't need these, this is just saying the amount of flow you can pass is at most 1), and we are trying to find max $\sum_{e \in S} x_e$.

We can re-write this as

$$\max \mathbf{c}^\top \mathbf{x} \text{ for } \mathbf{x} \in \mathbb{R}^{|E|}$$

$$A\mathbf{x} \leq \mathbf{b}$$

In fact, there was a Nobel prize awarded for simply using linear programming for economics. When I learned it in undergrad, it seemed very complicated, because it's a lot of linear algebra. So there is a geometric interpretation of what goes on here. All these linear inequalities are essentially an intersection of half-spaces (each $\mathbf{a}^\top \mathbf{x} \leq \mathbf{b}$, and taking the intersection of all of them gives you a polyhedral region). We have some direction $\mathbf{c}^\top \mathbf{x}$. We want to maximize over this surface in that direction. Now, the vertices are where the inequalities are tight (they are intersections of equations). What Dantzig proposed was a very nice algorithm on the simplex, where you move in the direction where it increases, until you reach the maximum. His algorithm worked very very well, even though theoretically, it could take exponential time. An upper bound on the number of vertices is $\leq \binom{m}{n} \leq m^n$, which is exponential. The simplex algorithm in the worst case can visit all of them. This is still an active area research, a while ago a faculty candidate gave a talk about new algorithms for linear programming; very complicated and beautiful.

When Dantzig invented this problem and the simplex algorithm, he gave a talk about it at Princeton. von Neumann had thought about the theory of games for 20 years, and he immediately said this is related to something he'd been thinking about, and starting writing the duality theorem on the board.

Linear programs have a dual. What is a dual? First note that we can transform any linear program to the standard form

$$\max \mathbf{c}^\top \mathbf{x}$$

$$A\mathbf{x} \leq \mathbf{b}$$

$$\mathbf{x} > \mathbf{0}$$

Let us say the solution is λ^*. We can always guarantee $\mathbf{x}$ non-negative? Why? Because we can always write any number as the difference of two non-negative numbers, and insist that both are non-negative.

We can also write $\mathbf{c}^T \mathbf{x} \geq \lambda$, and see if it feasible at all. Now how do you prove a linear program is infeasible? How do we show it is not larger than something? Duality theorem says there is a mechanized way of showing this. We can take linear combinations of the rows, and show that the reverse inequality is well. I am going to much simpler concept which will immediately imply this.

Definition 2.2. Dual linear program.

$$\min \mathbf{b}^\top \mathbf{y} \text{ where } \mathbf{y} \in \mathbb{R}^m$$

$$A^\top \mathbf{y} \geq \mathbf{c}$$

$$\mathbf{y} \geq \mathbf{0}$$

Let us say the solution is μ^*. The duality theorem says that $\lambda^* = \mu^*$.

To prove this, Dantzig used Farkas' Lemma, and von Neumann used the Brouwer fixed point theorem. Both are more complicated than the method we will use to show duality.

I am going to do this proof, give these algorithms and so on by relating linear programming to game theory, in particular zero-sum games.

3 Game Theory

We are trying to model economic interaction between people. The fundamental definition that we want to discuss is a zero-sum, two player game.

3.1 Zero-Sum Games

Definition 3.1. Zero-sum, two-player game.
We have $S_1 = n$ strategies of Player 1, and $S_2 = m$ strategies of Player 2. We also have a function $M : S_1 \times S_2 \to [0, 1]$, which can be thought of as the pay-off function.

Example 3.2. Rock-paper-scissors.
We have three strategies for each player; thus $m = n = 3$ and $S_1 = S_2 = \{\text{rock}, \text{paper}, \text{scissors}\}$. We can represent the payoff as a matrix:

$$0\ 1\ -1$$
$$-1\ 0\ 1$$
$$1\ -1\ 0$$

It's called a zero-sum game because whatever one player wins, another loses. There are more general games that are not zero-sum which Nash talked about. Economists work on modeling various types of games as matrices, many are not zero-sum.

What does it mean to play a game? Let us assume for now that we have only one round in a game. This is game theory, so we no longer have the assumption that he is an adversary, and that he is just as smart as me. We're not going to assume anything about what the other player is doing. What can I hope to attain. I am going to play the best assuming he does the worst possible thing. Under any circumstances, I am going to guarantee at least something for myself.

Definition 3.3. The min-max optimal strategy.
Let $i^* = \arg\left(\max_{i \in S_1} \min_{j \in S_2}\right)$. This is our strategy.

According to this definition, all of the strategies are equally good in rock-paper-scissors.
We define the von Neumann equilibrium.

Definition 3.4. von Neumann equilibrium.
It is a pair of strategies $(i, j) \in S_1 \times S_2$ such that $i = \text{argmax}_{k \in S_1} M(k, j)$ and $j = \text{argmin}_{k \in S_2} M(i, k)$.

This is different from optimization. von Neumann asked, what will they play? This is a reasonable solution. If neither player is incentivized to move away from their strategy, they are in equilibrium.

In RPS, there is no equilibrium - it's a never-ended cycle, there is no pure equilibrium. For some games, there is. You can think of some games that are very very large. There might be a pure equilibrium.

Example 3.5.

$$1\ 1\ 1$$
$$0\ 1\ 1$$
$$0\ 0\ 0.5$$

Here, 0.5 is an equilibrium - neither is incentivized to change.

von Neumann generalized to mixed strategies, distributions over S_1. Let $\mathbf{p} \in \triangle_{S_2}, \mathbf{q} \in \triangle_{S_2}$. We have $\mathbf{E}[\text{gain of } \mathbf{p}_2] = \sum_i \mathbf{p}_i \sum_j \mathbf{q}_j M(i, j) = \mathbf{p}^\top M \mathbf{q}$.

Definition 3.6. Mix Strategy profile.
$\mathbf{p}, \mathbf{q}$ is a von Neumann equilibrium iff $\forall i \in S_1, \forall j \in S_2$:

$$\mathbf{p}^\top M \mathbf{q} \geq \mathbf{e}_i^\top M \mathbf{q}$$
$$\mathbf{p}^\top M \mathbf{q} \geq \mathbf{p}^\top M \mathbf{e}_j$$

where $\mathbf{e}_i, \mathbf{e}_j$ are the unit vectors.

The optimal strategy that a row player can play will be $\mathbf{p}^* = \text{argmin}_{\mathbf{p}\in\triangle_{S_1}} \max_{\mathbf{q}} \mathbf{p}^\top M\mathbf{q}$, and $\mathbf{q}^* = \text{argmax}_{\mathbf{q}\in\triangle_{S_2}} \min_{\mathbf{p}} \mathbf{p}^\top M\mathbf{q}$.

We have

Theorem 3.7. *(von Neumann).*
Every zero-sum game has an equilibrium value d in particular where $(\boldsymbol{p}^*, \boldsymbol{q}^*)$ are in equilibrium and both achieve d.

We will not use duality to prove this. Historically, von Neumann proved this by looking at something looking at a mapping from two strategies to the best response. This equilibrium is a fixed point for this mapping. Then you can use fixed-point theory to show something exists - this requires topology to show this.

First, let us argue about the equivalence between linear programming and zero-sum games. I would like to write an equilibrium as a linear program. At least I would like to write the computation of $\mathbf{p}^*$ and $\mathbf{q}^*$ as a linear program. How can we compute $\mathbf{p}^*$ through linear programming?

When is the max obtained? It is obtained at the vertices. So for $\mathbf{p}^*$, we can write

$$\max \lambda \text{ s.t. } \forall i \in [n]$$
$$\mathbf{p}^\top M\mathbf{e}_i \leq \lambda$$
$$\sum \mathbf{p}_i = 1, \mathbf{p}_i \geq 0$$

where $\mathbf{p} \in \mathbb{R}^n$. Similarly, we can write

$$\max \mu \text{ s.t. } \forall j \in [n]$$
$$\mathbf{e}_i^\top M\mathbf{q} \geq \mu$$
$$\sum \mathbf{q}_i = 1$$
$$\mathbf{q}_i \geq 0$$

These are dual programs, and $\lambda = \mu$. Given any matrix M, we can sketch this form. We take any linear program, ensure variables are non-negative (split into two nonnegative variables), ensure things sum up to 1 (if they don't, add some extra thing, and split it into too if less than 1. If greater than 1, scale it.)

Minimax theorem implies duality, duality implies minmax.

I have to say, I prefer the game view to duality. For games, both of you can play and you will obtain the same value. It's a much clearer interpretation for me than the geometric.

Proof. Von-Neumann theorem (from a proof in Freund-Schapire).
Recall $\lambda = \min_p \max_q \mathbf{p}^\top M\mathbf{q}$ and $\mu = \max_q \min_p \mathbf{p}^\top M\mathbf{q}$.

1. $\lambda \geq \mu$. We have $\lambda = \max_q \mathbf{p}^{*\top} M\mathbf{q} \geq \max_q \min_p \mathbf{p}^\top M\mathbf{q} = \mu$. This is called weak duality, and basically uses nothing.

2. $\mu \geq \lambda$. This is the main part. Now we will use low regret algorithms. Consider a repeated game (OCO setting). For $t = 1, 2, \cdots, T$, our decision set $K = \triangle_n$. A player plays $\mathbf{p}_t$ according to some low regret algorithm, we define $\mathbf{q}_t$ by the $\text{argmax}_{\mathbf{q}\in\triangle_n} \mathbf{p}_t^\top M\mathbf{q}$. We define a loss function according to $f_t(\mathbf{p}) = \mathbf{p}^\top A\mathbf{q}_t$.

   Denote by $\bar{\mathbf{p}_t}$ and $\bar{\mathbf{q}_t}$ the averages of all strategies up to time t. $\lambda = \min_p \max_q \mathbf{p}^\top M\mathbf{q} \leq \max_q \bar{\mathbf{p}_t} M\mathbf{q} = \max_q \frac{1}{T}\sum_t \mathbf{p}_t^\top M\mathbf{q} \leq \frac{1}{T}\sum_t \mathbf{p}_t M\mathbf{q}_t = \frac{1}{T}\sum_t f_t(\mathbf{p}_t) \leq \frac{1}{T}\sum_t f_t(\mathbf{p}^*) + \frac{\text{Regret}}{T} \leq \frac{1}{T}\sum_t \mathbf{p}_t^* M\mathbf{q} + \frac{\sqrt{2\log(n)}}{\sqrt{T}} = \min_p \mathbf{p}^\top M\mathbf{q} + \sqrt{\frac{2\log(n)}{T}} \leq \max_q \min_p \mathbf{p}^\top M\mathbf{q} + \sqrt{\frac{2\log(n)}{T}} \to \mu$. $\mathbf{q}$ is chosen to maximize

the payoff, so we can replace by $\mathbf{q}_t$. Also we suppose we are using randomized weighted majority. Finally, we are taking the limit as $T \to \infty$.

All we used here is really the inequality that follows from randomized weighted majority, i.e. a regret bound from a low-regret OCO algorithm. It could even be the worst rate, the rate does not even come into the picture. I think this is the simplest known proof for the duality theorem and the minmax theorem.

□

We know we can optimize over a general convex ball. Let $f : S \times T \to \mathbb{R}$ such that f is convex-concave. We have a bilinear function essentially (linear in both p and q). Convex-concave means $f(\cdot, t)$ is convex and $f(s, \cdot)$ is concave for all $s \in S$, $t \in T$. Furthermore suppose that S, T are convex.

We can show that

Theorem 3.8. *Sion's minimax theorem.*

$$\min_{s \in S} \max_{t \in T} f(s,t) = \max_t \min_s f(s,t)$$

This is convex duality, and is stronger than LP duality. A variant of the proof we just did proves the statement.

4 A Few More Remarks About Games

4.1 Non-zero sum Games

One example of a non-zero sum game is the Prisoner's Dilemma.

Example 4.1. Prisoner's dilemma.
Two convicts have to decide whether they rat out the other guy, or if they cooperate with each other. The function is in terms of the number of years in jail. In non-zero-sum games, we must write what they both get. If they both rat each other out, they each get 5 years. If they both cooperate with each other, they get 3 years each. If one cooperates and the other rats out, the one who rats out gets out of jail and the other stays 10 years.

One matrix with two numbers doesn't make sense, so what we do is write out different matrices for different players. In this case, we have

$$A = \begin{bmatrix} 5 & 0 \\ 10 & 3 \end{bmatrix}$$

$$B = \begin{bmatrix} 5 & 0 \\ 10 & 3 \end{bmatrix}$$

No matter what happens, I have an incentive to rat. From a static point of view, the intuitive sense is that taking the minmax is not satisfactory. The $\mathbf{p}^*$ and $\mathbf{q}^*$ are no longer in equilibrium. How do we define an equilibrium in this setting. We now have two matrices.

Definition 4.2. Nash equilibrium.
It is defined to be a pair of strategies $(\mathbf{p}, \mathbf{q}) \in \triangle_n \times \triangle_n$ such that $\forall i$ $\mathbf{e}_i^\top A \mathbf{q} \leq \mathbf{p}^\top A \mathbf{e}_i$, and $\forall j$, $\mathbf{p}^\top B \mathbf{e}_j \geq \mathbf{p}^\top B \mathbf{q}$.

We can prove that Nash equilbrium exists. The picture is not so nice as for zero-sum games. Just a slight change, and things get harder. Equilibrium can be prove by using the Brouwer fixed-point theorem.

Remark 4.3. What happens in a repeated game (A, B) if both players play no-regret algorithm? The idea is that these optimal strategies will converge to an equilibrium. But there is no such value in non-zero sum games. So what happens in a general game? This is kind of an open question, and the field is called no regret dynamics. Some papers say that players will keep on shifting their strategies forever. For some games, (convex-concave games) they do converge to Nash equilibrium. Sometimes it is known that nothing happens. There are other concepts, called correlate equilibrium. There is a very rich area in the subject and an active area of research.

Remark 4.4. Regarding Prisoner's Dilemma, there is a Nash equilibrium - it is for both to rat. If one rats, there is no reason for the other to cooperate. But if we repeat the game, it is very interesting. Let us say that I give you the following task: you have to define a computer program, and play them against each other. They will play Prisoner's Dilemma 1000 times. There could be one program that always rats. Is this the best program? Not for sure. There could be one that cooperates. If there are many programs that you guys create, maybe on average there will be one that does better than the one that always rats. What is going to happen in this kind of question? It has been empirically shown that the following strategy is optimal. It is called Tit for tat. You cooperate: if the other guy is also ratting, you also rat. If he stops cooperating, you also stop. It is like three lines of code. This three line algorithm is sort of going to win all of them. You cooperate until the other person rats, and then you rat until the other guy starts cooperating again. People have written books about this - we have to punish the guy who is not cooperating. If it plays against itself, it cooperates forever. If it plays against a guy who is evil, it is evil. It can forgive, too! If you start behaving well, then I will forgive you. Is there a formal explanation of why this does well? It turns out that there is. This is called evolutionary game theory. This has a biological explanation. Let's say there is a large population, and we will divide the population into algorithms. Maybe 1% are evil, 50% are nice guys, maybe 2% are tit-for-tat. We will do the following experiment. We will increase or decrease the population according to the average payoff - the population of the strategies will increase/decrease. Teams that win more resources get more people on their team. Maybe they start killing off each other after the environment changes, so there is an inherent dynamics to this.

Definition 4.5. Evolutionary stable equilibrium.
If you play around, the portion plays the same.

Tit-for-tat is good in the sense that it will increase and increase until it is the dominating strategy. There are maybe a few people that look like tit-for-tat, but also defect at the end and do a little better. If they propagate, they will kill each other off, so it is not stable - but they can do better.

There are many interesting questions here. What happens if you play a no-regret algorithm,etc. ? The literature unfortuantely is insufficient in this field. They often talk about differential equations and dynamics.

5 How to Solve a Linear Program

We will solve a linear program in the game-theoretic formulation. We are given min λ, $A\mathbf{x} \leq \lambda$, $\mathbf{x} \geq \mathbf{0}$. We can reduce every linear program to this formulation by at most doubling the number of variables, and maybe adding a few more. We have the simplex method, the ellipsoid method (theoretically very well, but not very good in practice), and interior point methods (which are based on Newton's method). They are good in practice, and in theory, but the theory is very heavy-weight. You could do a whole course on interior point methods.

Then, there is a very easy method which I will show today. We will basically copy the proof of the minmax theorem.

Theorem 5.1. *The Lagrangian-relaxation returns $\bar{\boldsymbol{x}}$ such that $\bar{\boldsymbol{x}} \in \triangle_n$. For all i, $A_i\bar{\boldsymbol{x}} \leq \lambda^* + \sqrt{\frac{2\log(n)}{T}}$*

Algorithm 1 Lagrangian Relaxtion

$\mathbf{x} \leftarrow [1, 0, 0, \cdots, 0]$
for $t = 1, 2, \cdots, T$ **do**
    $\mathbf{q}_t \leftarrow \max_q \mathbf{p}_t^\top A \mathbf{q}$
    $\mathbf{x}_{t+1}(i) = \mathbf{x}_t(i) e^{-\eta A_i \mathbf{q}_t}$ and scale to 1.
end for
Return $\bar{\mathbf{x}} = \frac{1}{T} \sum_t \mathbf{x}_t$

Then, the runtime to get an ϵ-approximate solution is $T = \mathcal{O}\left(\frac{\log(n)}{\epsilon^2}\right)$, and then each step takes mn (matrix multiplication). So in total, the runtime is $\mathcal{O}\left(\frac{mn\log(n)}{\epsilon^2}\right)$, which is linear time if $\epsilon \in \mathcal{O}(1)$.

Proof. $\max_q \bar{\mathbf{x}} A \mathbf{q} \leq \frac{1}{T} \sum_t \mathbf{x}_t A \mathbf{q}_t \leq \frac{1}{T} \min_{\mathbf{x}} \sum_t \mathbf{x}^\top A \mathbf{q}_t + \sqrt{\frac{2\log(n)}{T}} = \min_{\mathbf{x}} \mathbf{x}^\top A \bar{\mathbf{q}} + \sqrt{\cdot} \leq \max_q \min_x \mathbf{x}^\top A \mathbf{q} + \sqrt{\cdot} = \lambda^* + \sqrt{\frac{2\log(n)}{T}}$. So this implies that for all i, $\bar{\mathbf{x}} A \mathbf{e}_i \leq \lambda^* + \sqrt{\frac{2\log(n)}{T}}$. □

It is not used too much, because many times you want ϵ to be small, in which case it is not too good.. but for the linear programs which arise in machine learning, you have noise in the matrix (maybe 1% in the data), and you don't need the ϵ being that small. So this sort of method is much better for machine learning, because the other methods run much worse in m, n, but better in ϵ. This other algorithm is the opposite, because we care less about precision.

There is an algorithm that does it in $\Theta\left(\frac{m+n}{\epsilon^2}\log(n)\right)$, which is actually sublinear! This is much better than what you would get than any other solver. This is given by Clarkson, me (Hazan), and Woodruff. This again allows for bad ϵ when you do not care about precision, and is the best you can do.

There are things called LP-rescaling algorithms, and they relate a lot to what was discussed today by Dunagan-Vempala. They use rescaling to get $\frac{1}{\epsilon}$ guarantee, but this happens in quadratic time.