Lecture 3

Algorithms for single shot and repeated 0-sum games

April 8, 2005
Lecturer: Anna Karlin
Notes: Chris Ré

3.1 Overview

So far we have discussed different solution concepts for games. Each of these solution concepts gives a value of a game under assumptions about how players will play. In this lecture, we will examine how players find the solutions that correspond to these different solution concepts. A solution's existence does us little good if finding how to play it is intractable.

We show how to find Nash equilibria by solving a linear program. Along the way we will examine some basic properties of games, for example: In a one shot game does it matter who goes first? Last we examine repeated games. We give an algorithm that gets the value of the game over the long term.

3.2 0 or Constant Sum Games

Notice it is without loss of generality that we say 0 sum games, since adding a constant or multiplying by a constant factor would change little. In what follows, we will be analyzing payoffs from the point of view of the row player.

3.2.1 Some Definitions

We start with a discussion of a classical result for 0 sum games. These games are among the simplest and nicest games so we will do a quick review of them. Here an $n \times n$ matrix, M represents the payoffs for the game. Specifically, M_{ij} is the payoff to the row player, if the row player plays strategy i and the column player strategy j. The game is called 0 sum because the column players payoff is $-M_{ij}$ that is their sum is 0.

Definition 3.1. Given $\Delta = \{x \in \mathbb{R}^n | \sum x_i = 1, x_i \geq 0\}$. An $\boldsymbol{x} \in \Delta$ is called a *strategy distribution*

A vector $\boldsymbol{x} \in \Delta$ is an assignment of weights to each of the n strategies.

Let $\boldsymbol{p}$ denote the row player's strategy and $\boldsymbol{q}$ denote the column player's stategy. We can calculate the expected payoff of two strategies in the obvious way $\sum_{i=1}^{n} \sum_{j=1}^{n} p_i q_j M_{ij}$. Sometimes we will denote this by $M(\boldsymbol{p}, \boldsymbol{q})$.

3.2.2 Going first

Intuitively, it seems like a disadvantage to go first because your opponent can then tailor their strategy to yours. To make this more precise, suppose that you received payoff M_{ij} as a result of playing in the first round. Now suppose that your opponent played some strategy j' to start with and you again play strategy i. Then your payoff is $M_{ij'}$. You know that $M_{ij'} \geq M_{ij}$ because your opponent should be trying to minimize along the row i. This is clear in a zero sum game because your opponent should seek to minimize your payoff thus maximizing his own. As a result, it is clearly at least as good to go second. The natural question now is: Is it better to go second?

3.2.3 Von Neumann's Theorem

We will answer this question by transforming our problem into a linear program and then appealing to strong duality to tell us that the answer is no. To do this we need some notation, let A^* be the row players payoff if row player goes first and plays optimally. Let B^* be the row players payoff if the row player goes second. The discussion in section 3.2.2 makes clear that $A^* \leq B^*$. We can write formulae for these terms explicitly:

$$A^* = \max_{\boldsymbol{p}} \min_{\boldsymbol{q}} \sum_i \sum_j p_i q_j M_{ij}$$

$$B^* = \min_{\boldsymbol{q}} \max_{\boldsymbol{p}} \sum_i \sum_j p_i q_j M_{ij}$$

Remark. Notice there is a pure strategy that is a best response when going second. This is because it never makes sense to put weight on a non maximium valued strategy index after the opponents strategy is given. As a result, you just pick any one of the maximium responses which is clearly at least as good.

Notice that a similiar remark about going first does not hold. For example think of the tossing coins game we saw. In this game if you go first and select a pure strategy, say always heads, then you will lose every time while the mixed solution has you winning about half the time.

Theorem 3.1 (von Nuemann). *MinMax Theorem* $A^* = B^*$[1]

First let us write the column player goes first as a linear programming. From our above remark it suffices to consider the case where the second player plays a pure stategy. This allows to wrie the following linear program in $n+1$ variables, the stategy indicies and the payoff.

Let $B = \max_i \sum M_{ij} q_j$, we seek to minimize B.

$$\begin{array}{ll} B - \sum M_{ij} q_j & \geq 0 \\ \vdots & \vdots \\ B - \sum M_{nj} q_j & \geq 0 \\ \sum q_j & = 1 \\ \forall i \; q_i & q_i \geq 0 \end{array}$$

The program for row player goes first is $A = min_j \sum M_{ij} p_i$, we seek to maximize A.

[1]Prof. Beame recommends a freely avaialable book by a similiar name. The book is a=b available from http://www.cis.upenn.edu/ wilf/AeqB.html

$$
\begin{array}{ll}
\sum M_{i1} p_i & \geq A \\
\vdots & \vdots \\
\sum M_{in} p_i & \geq A \\
\sum p_i & = 1 \\
\forall i p_i & p_i \geq 0
\end{array}
$$

Proof. By Linear Programming Strong Duality We need to show that these programs are duals. Their optimization terms are dual. Notice that we get $\sum p_i \leq 1$ as the dual condition which is not exactly what we wanted. If there is a non-zero p_i solution, then since we are maximizing the A we can without loss assume there is one where $\sum p_i = 1$ since it can only improve. Now we have shown

$$\max_{\boldsymbol{p}} \min_{j} \sum M_{ij} p_i = \min_{\boldsymbol{q}} \max_{\boldsymbol{p}} \sum_i \sum_j p_i q_j M_{ij}$$

which suffices to show the theorem. □

3.2.4 Solutions and Nash Equilibria

If you solve the program and its dual from section 3.2.3, you get a pair of vectors that are strategies we call these p^*, q^*. We will denote by $M(p^*, q^*)$ the expected value of the game under these stategies. We call the tuple $(p^*, q^*, M(p^*, q^*))$ a solution to the game. Now, we are going to relate this solution to our notion of Nash equilibrium.

Theorem 3.2. *$(p^*, q^*, M(p^*, q^*))$ is a solution to a game iff p^*, q^* is a mixed Nash Equilibrium.*

Proof.

$$\max_{\boldsymbol{p}} M(\boldsymbol{p}, q^*) \leq M(p^*, q^*) \leq \min_{\boldsymbol{q}} M(p^*, \boldsymbol{q})$$

These inequalities hold if (p^*, q^*) is a Nash Equilibrium. This precisely says that neither player has incentive to deviate.

Now notice that $\min_{\boldsymbol{q}} \max_{\boldsymbol{p}} M(p, q) \leq \max_{\boldsymbol{p}} M(p, q^*)$ and similiarily $\min_{\boldsymbol{q}} M(p^*, \boldsymbol{q}) \leq \max_{\boldsymbol{p}} \min_{\boldsymbol{q}} M(\boldsymbol{p}, \boldsymbol{q})$.

By theorem 3.1, these two quantities are equal. This proves the theorem in both directions since all inequalities collapse to equalities. □

There are still some things we would like to deal with: What if this M matrix is very large or unknown to the players? What if one of the players is dumb (i.e. not playing optimally)?

3.3 Repeated Games

Up until now, we have just considered one shot games. Now we want to play the same game repeatedly, more formally here is the setup.

- Row Player chooses a mixed strategy $\boldsymbol{p}_t$ for time step t.

- Column Player chooses $\boldsymbol{q}_t$ for the same step.
- Row Player observes the payoffs $\forall i \ M(i, \boldsymbol{q}_t)$
- Row Player gets $M(\boldsymbol{p}_t, \boldsymbol{q}_t)$

We would like to do as well as the best fixed strategy against the column players choices. Notice, we are not considering how we do against the best dynamic algorithm - it is against a fixed choice of strategy for all plays.

Remark. Notice that we are learning all payoffs against that strategy - not just the one we receive. This is for simplification and there are extensions that deal with these more general cases.

3.3.1 Learning from Experts

A choice of strategies at each step, your choice corresponds to with what weight you choose to listen to one of the experts. After each stage you experience the gain of your expert. In this case we will give an algorithm such that:

$$E[\text{gain}] \geq G_{opt}(1 - \frac{\epsilon}{2}) - \frac{g_{\max}}{\epsilon} \log n$$

3.3.2 Weighted Majority

There are many variations of this algorithm but, they seem to follow the same flavor.

Let $w_i(t)$ denote the weight given to expert i at time t. Let $w(t)$ denote $\sum_i^n w_i(t)$.

At time zero, initialize the weights for $i \in \{1, \dots, n\}$ set $w_i(0) = 1$. At each stage we pick an expert with probability $\frac{w_i}{w}$.

Given gains $(g_1, \dots, g_n)$ at time t let $\widehat{g_i} = \frac{g_i}{\sum_j g_j}$. We update the weights as $w_i(t+1) \leftarrow w_i(t)(1+\epsilon)^{\widehat{g_i}}$.

First two inequalities for use in the main theorem:

Claim 3.1. $(1+\epsilon)^x \leq 1 + \epsilon x$ when $\epsilon \geq 0$ and $x \in [0, 1]$.

Proof. Let $f(x) = (1+\epsilon)^x - 1 - x\epsilon$. The inequality holds if $f(x) \leq 0$ on $[0, 1]$. Notice its second derivative, $(1+\epsilon)^x \log(1+\epsilon)^2$, is strictly positive on $[0, 1]$. This is because $\epsilon > 0$. $f(0) = f(1) = 0$ and it is negative in between, so the inequality holds. $\square$

Claim 3.2. for $x \geq 0$ $\log(1+x) \in [x - \frac{x^2}{2}, x]$

Proof. Clearly this holds at 0, they all equal 0.

Examine the derivative of $\log(1+x) - x = \frac{1}{1+x} - 1$ which is always non-positive on $[0, \infty)$. This shows $\log(1+x) \leq x$.

Now, $\frac{1}{1+x} - 1 + x = \frac{-x+x+x^2}{1+x} = \frac{x^2}{1+x}$ which is always non-negative on $[0, \infty)$. This shows $\log(1+x) \geq x - \frac{x^2}{2}$. $\square$

Theorem 3.3.
$$E[gain] \geq G_{opt}(1-\frac{\epsilon}{2}) - \frac{g_{\max}}{\epsilon}\log n$$

Proof. Let $E_t = \sum_i \frac{w_i}{w} g_i(t)$, this is our expected gain in the t^{th} round.
Let $\widehat{E_t} = \frac{E_t}{g_{\max}}$. Consider the t^{th} stage of the algorithm.

$$w(t+1) = \sum_i w_i(1+\epsilon)^{\widehat{g_i}} \leq \sum_i (1+\widehat{g_i}\epsilon) \text{ by claim 3.1}$$
$$= w(t) + w(t) * E_t * \epsilon = w(t)(1+\epsilon E_t)$$

Let t_{end} be the index of the final game. Now, $(1+\epsilon)^{\widehat{g_{opt}}} \leq w(t_{end})$ because $\widehat{g_{opt}}$ represents a single weight used in the update and $w(t_{end})$ is a sum over all of them.

$w(t_{end}) \leq w(0)\prod_t^{t_{end}}(1+\epsilon\widehat{E_t}) = n\prod_t(1+\epsilon\widehat{E_t})$ holds by the above since $g_{max} \leq 1$ and the weights are initialized to 1. So we have the following situation.

$$\widehat{g_{opt}}\log(1+\epsilon) \leq \log n + \sum_t^{t_{end}} \log(1+\epsilon\widehat{E_t})$$

We can use claim 3.2 to estimate this and get:

$$\widehat{g_{opt}}(\epsilon - \frac{\epsilon^2}{2}) \leq \log n + \sum_t^{t_{end}} \epsilon\widehat{E_t}$$

$$\Rightarrow \sum_t^{t_{end}} \widehat{E_t} \geq G_{opt}(1-\frac{\epsilon}{2}) - \frac{\log n}{\epsilon}$$

□

3.3.3 Application of Weighted Majority to Learning from Experts

We can directly apply this to our situation. In our case the experts are rows $g_i(t) = M(i, \boldsymbol{q_t})$. Our goal is to show that in the 'long term', we get the value of the game. In this case, by long term we mean the expectation of our strategy is equal to the value of the game.

Let's examine the average per round payoff, let T be the number of rounds:

$$\max_i \sum_j M_{ij}(\frac{1}{T}\sum_t^T q_j(t)) \geq \min_{\boldsymbol{q}} \max_i \sum_j M_{ij} q_j(t)$$

The minimium value of $\boldsymbol{q}$ is independent of time, so clearly this is lower than the average value over time. Notice that this right hand side is the value of the game at each step - so we have our desired solution.