Lecture 4

Lecturer: Constantinos Daskalakis *Scribe: Jason Biddle, Alan Deckelbaum*

NOTE: The content of these notes has not been formally reviewed by the lecturer. It is recommended that they are read critically.

1 Introduction

In this lecture we cover the following topics:

- Hedging
- Using Learning in Zero-Sum Games

This is our last lecture on zero-sum games. The reason we have spent several lectures on zero-sum games is that zero-sum games are one of the few cases in the social sciences where we're fairly confident about our mathematical predictions of how agents behave—recall Aumann's quote from the first lecture. In this lecture, we begin by finishing our discussion on hedging and expert algorithms, and then relate this topic to convergence of distributed algorithms to equilibria in zero-sum games.

2 Expert Algorithms

Recall our setup from last lecture. There are n experts, and at each time-step $t = 1, 2, 3 \ldots$ there is an associated loss vector $\boldsymbol{\ell}^t \in [0,1]^n$ which assigns a loss value to each expert. In each time-step, we pick a probability distribution $\mathbf{P}^t$ over the n experts. Our loss obtained up to T is given by

$$L^T := \sum_{t=1}^{T} \boldsymbol{\ell}^t \cdot \mathbf{P}^t.$$

As a benchmark for our algorithm's performance, we compare L^T against the performance of the single best expert over these rounds. That is, our benchmark is

$$\min_i \sum_{t=1}^{T} \ell_i^t.$$

For notational purposes, we will refer to the quantity $\sum_{t=1}^{T} \ell_i^t$ as L_i^T. As mentioned in the last lecture, the follow-the-leader algorithm (at each time τ, choose an expert which has minimal $L_i^{\tau-1}$ value) can be worse than our benchmark by a factor of n. In this lecture, we will show that the "multiplicative weights update algorithm" or "hedging algorithm" has a significantly better performance guarantee.

2.1 Hedging Algorithm (Multiplicative Weights Update Algorithm)

We now recall the hedging algorithm introduced last lecture. At each time-step, we will maintain a weight vector $\mathbf{w}^t$ assigning a weight to each expert. Our probability distribution $\mathbf{P}^t$ will simply assign probabilities to experts proportional to their weights:

$$\mathbf{P}^t = \frac{\mathbf{w}^t}{\mathbf{w}^t \cdot \mathbb{1}}.$$

To update the weights each round, we simply set

$$w_i^t \leftarrow w_i^t \cdot u_b(\ell_i^t)$$

where u_b is any function which satisfies the following conditions:

- $b \in (0, 1)$
- For all $x \in [0, 1]$, we have $b^x \leq u_b(x) \leq 1 - (1 - b)x$

We now have the following performance guarantee of the multiplicative weights update algorithm:

Theorem 1. *For all $\boldsymbol{\ell}^1, \boldsymbol{\ell}^2, \ldots, \boldsymbol{\ell}^t, \ldots$ and all t, we have*

$$L^t \leq (\min_i L_i^t) \cdot \frac{\ln(1/b)}{1-b} + \frac{\ln n}{1-b}.$$

Proof: We will define a potential function at time t to be $\ln(\sum_{i=1}^n w_i^t)$. We compute

$$\sum_{i=1}^n w_i^{t+1} = \sum_{i=1}^n (w_i^t \cdot u_b(\ell_i^t)) \leq \sum_{i=1}^n (w_i^t(1 - (1-b)\ell_i^t)).$$

We now note that $w_i^t = p_i^t \cdot \sum_i w_i^t$, and hence the right-hand side above is just

$$= \left(\sum_i w_i^t\right) \cdot \sum_i (p_i^t(1 - (1-b)\ell_i^t)).$$

We now take the natural log of both sides to obtain

$$\ln\left(\sum_{i=1}^n w_i^{t+1}\right) \leq \ln\left(\sum_{i=1}^n w_i^t\right) + \ln\left(1 - (1-b)\mathbf{P}^t \cdot \boldsymbol{\ell}^t\right).$$

Since $ln(1-x) \leq -x$, we have:

$$\ln\left(\sum_{i=1}^n w_i^{t+1}\right) \leq \ln\left(\sum_{i=1}^n w_i^t\right) - (1-b)\mathbf{P}^t \cdot \boldsymbol{\ell}^t.$$

Summing both sides of the inequality from $t = 1$ to T and cancelling the terms which appear on both sides yields:

$$\ln\left(\sum_{i=1}^n w_i^{T+1}\right) \leq \ln\left(\sum_{i=1}^n w_i^1\right) - (1-b)L^T.$$

We can initially distribute the weights uniformly, so that (for example) $w_i^1 = \frac{1}{n}$ for each i. We notice that in this case, $\ln(\sum_i w_i^1) = 0$. Therefore, we have

$$L^T \leq \frac{-\ln(\sum_{i=1}^n w_i^{T+1})}{1-b}.$$

By monotonicity of the negative log function, we see that the for any particular i, we have

$$L^T \leq \frac{-\ln(w_i^{T+1})}{1-b}.$$

We now observe that our update rule $w_i^{t+1} \leftarrow w_i^t u_b(\ell_i^t)$ combined with the inequality $u_b(x) \geq b^x$ implies that $w_i^{T+1} \geq w_i^1 b^{\ell_i^1} \cdot b^{\ell_i^2} \cdots b^{\ell_i^T}$ and hence $w_i^{T+1} \geq w_i^1 b^{L_i^T} = \frac{1}{n} b^{L_i^T}$. Therefore, we see that for all i, we have

$$L^T \leq \frac{-\ln(\frac{1}{n} b^{L_i^t})}{1-b} = \frac{\ln(n)}{1-b} - \frac{L_i^t \ln(b)}{1-b}.$$

Since i was arbitrary, the proof is complete.

□

If we set $b = 1 - \epsilon$ for some $\epsilon \in (0, 1/2)$, our above bound becomes

$$L^T \leq (\min_i L_i^T) \frac{\ln(\frac{1}{1-\epsilon})}{\epsilon} + \frac{\ln(n)}{\epsilon}.$$

Using the standard inequality $-\ln(1-z) \leq z + z^2$ for all $z \in (0, 1/2)$, we obtain

$$L^T \leq \min_i L_i^T (1+\epsilon) + \frac{\ln(n)}{\epsilon}.$$

Suppose that we know the time horizon T in advance. Then we can set

$$\epsilon = \min\left(\sqrt{\frac{\ln(n)}{n}}, \frac{1}{2}\right)$$

to obtain the bound

$$L^T \leq \min_i L_i^T + 2\sqrt{T \cdot \ln(n)}.$$

Therefore, we can bound the average loss by

$$\frac{L_T}{T} - \min_i \frac{L_i^T}{T} \leq \frac{\sqrt{4 \ln(n)}}{\sqrt{T}}.$$

Even if we do not know the final time horizon T in advance, we can use a "doubling trick" to obtain a similar bound. The idea behind this trick is to start by choosing ϵ for a T of 2. If the time horizon exceeds 2, we now select a new ϵ corresponding to $T = 4$. If this time horizon is surpassed, we select a new ϵ corresponding to $T = 8$, and so on.

Instead of the above "doubling trick," we could also change ϵ with each step. By setting ϵ to have a form such as $\epsilon_t = \min\left(\sqrt{\frac{\ln(n)}{t}}, \frac{1}{2}\right)$, we can do slightly better than we did with the doubling trick.

2.2 Tightness of the Multiplicative Weights Bound

We now look at how close the performance of the multiplicative weights update method is to the optimal learning algorithm. We will argue that our bound is asymptotically close to optimal, by giving two examples.

2.2.1 Example 1

Suppose we have n experts. In each round, an expert will either receive a loss of 0 or a loss of 1 (that is, $\boldsymbol{\ell}^t \in \{0,1\}^n$.) The losses are assigned according to the following random process:

- At $t = 1$, select a random subset $S_1 \subset [n]$ of size $n/2$. Assign loss 0 to the experts in S_1, and assign loss 1 to the experts in $\bar{S_1}$.
- At $t = 2$, select a random subset $S_2 \subset S_1$ of size $n/4$. Assign loss 0 to the experts in S_2, and assign loss 1 to the remaining experts.

- At $t = 3$, select a random subset $S_3 \subset S_2$ of size $n/8$. Assign loss 0 to the experts in S_3, and assign loss 1 to the remaining experts.
- Continue the above process up until $t = \log_2 n$. At each step t a total of $n/(2^t)$ experts will have loss 0.

It is clear that, after $t = \log_2 n$, the best expert will have loss 0. Furthermore, it is clear that any learning algorithm A will have expected performance $E[\ell_A^t] \geq \frac{1}{2}$. (All of the experts from $\bar{S}_{t-1}$ will have loss 1, and every expert from S_{t-1} will have expected loss 1/2 at time t.) Therefore, for any learning algorithm A, we see that $E[L_A^T] \geq T/2$, where $T \approxeq \log_2 n$. Since $T \approxeq \log_2 n$, our above bound for the multiplicative weights learning algorithm of $2\sqrt{T \ln n}$ is within a constant factor of the best possible performance of $\log_2 n$.

2.2.2 Example 2

The above example had a bounded time horizon of $T = \log_2 n$. However, we can also provide an example with an unbounded time horizon. In this example, we have 2 experts. At every time t, we choose $\boldsymbol{\ell}^t$ to be either $(0, 1)$ or $(1, 0)$ uniformly at random. (That is, we uniformly at random select one expert to receive 1 point of loss, and the other expert receives no points of loss.)

It is obvious that every learning algorithm A will have $E[L_A^T] \geq T/2$ (since both experts have an expected loss of 1/2 at each round).

Our benchmark (the loss of the best expert at time T) will be, with constant probability, $\frac{T}{2} - \Omega(\sqrt{T})$. (This bound comes from the thought experiment of flipping a fair coin T times and estimating the minimum of the number of heads and the number of tails. We know that, with significant probability, the average number of heads will be within a few standard deviations of the average number of tails. Thus, the number of heads and the number of tails should each be, with high probability, within $\pm\sqrt{T}$ from $T/2$.) Thus, we see that the $\sqrt{T}$ term in our performance guarantee is necessary.

3 Back to Zero-sum Games

Recall the definition of a two-player zero-sum game defined by a pair of $m \times n$ payoff matrices (R, C) where $R + C = 0$. For the remainder of this section, we assume w.l.o.g. that $m = n$.

Now suppose row player and column player both use a multiplicative weights update (MWU) experts algorithm to generate their respective mixed strategies, x^t and y^t for the zero-sum game at time step t. Each player could use a different MWU algorithm, however we assume both algorithms are of low regret (i.e. the algorithm achieves the MWU bound discussed in the previous section).

Since the game is zero-sum, row player's loss at time step t is determined by column player's strategy,

$$\ell_{row}^t = Cy^t.$$

Similiarly, column player's loss at time step t is determined by row player's strategy,

$$(\ell_{col}^t)^T = (x^t)^T R \quad \Leftrightarrow \quad \ell_{col}^t = R^T x^t.$$

Recall our assumption of bounded losses, $\ell^t \in [0, 1]^n$ in the experts algorithm setting. Here, however, the losses of row and column player are not restricted to that range since the payoffs in R and C can take on any values as long as $R + C = 0$. In order to use the proved MWU bounds, we must first normalize R and C so that $\ell_{row}^t, \ell_{col}^t \in [0, 1]^n$.

Before proceeding, let us introduce the following scalar term:

$$M = \max_{i,j} |R_{i,j}| = \max_{i,j} |C_{i,j}| \qquad \text{since } R + C = 0.$$

Now we apply an affine transformation to R and C to produce normalized payoff matrices R' and C'

$$\begin{aligned} R' &= \frac{1}{2M}[R + M\,\mathbb{1}] \\ C' &= \frac{1}{2M}[C + M\,\mathbb{1}], \end{aligned}$$

where $\mathbb{1}$ denotes the matrix of ones. Next we derive the cumulative loss for row and column player's experts algorithms for time horizon T using the normalized payoff matrices. To avoid confusion between the time horizon T and the tranpose operator, we henceforth use the conjugate transpose A^* to denote the transpose of A. Entries in the strategy vectors and payoff matrices are real, so A^* coincides with the transpose of A.

$$\begin{aligned} L_{row}^T &= \sum_{t=1}^{T} (x^t)^* \ell_{row}^t = \sum_{t=1}^{T} (x^t)^* C' y^t = \sum_{t=1}^{T} (x^t)^* \left(\frac{1}{2M}[C + M\,\mathbb{1}] \right) y^t \\ &= \frac{1}{2M} \left[\sum_{t=1}^{T} (x^t)^* C y^t + M \sum_{t=1}^{T} (x^t)^* \mathbb{1} y^t \right] = \frac{1}{2M} \left[\sum_{t=1}^{T} (x^t)^* C y^t + MT \right], \end{aligned} \tag{1}$$

since $\forall t$, $(x^t)^* \mathbb{1} y^t = 1$. Using the same construction for column player, we find the following:

$$L_{col}^T = \sum_{t=1}^{T} (\ell_{col}^t)^* y^t = \sum_{t=1}^{T} (x^t)^* R' y^t = \frac{1}{2M} \left[\sum_{t=1}^{T} (x^t)^* R y^t + MT \right]. \tag{2}$$

Now we bound the cumulative loss for row player's experts algorithm given a known time horizon T,

$$\begin{aligned} L_{row}^T &\le \min_i L_i^T + 2\sqrt{T \ln(n)} \le L_i^T + 2\sqrt{T \ln(n)}, \quad \forall i \\ L_i^T &= \sum_{t=1}^{T} e_i^* \ell_{row}^t = \sum_{t=1}^{T} e_i^* C' y^t = \sum_{t=1}^{T} e_i^* \left(\frac{1}{2M}[C + M\,\mathbb{1}] \right) y^t \\ &= \frac{1}{2M} \left[\sum_{t=1}^{T} e_i^* C y^t + M \sum_{t=1}^{T} e_i^* \mathbb{1} y^t \right] = \frac{1}{2M} \left[\sum_{t=1}^{T} e_i^* C y^t + MT \right] \\ \Rightarrow L_{row}^T &\le \frac{1}{2M} \left[\sum_{t=1}^{T} e_i^* C y^t + MT \right] + 2\sqrt{T \ln(n)}, \qquad \forall i, \end{aligned}$$

which we combine with equation 1 and simplify,

$$\begin{aligned} \frac{1}{2M} \left[\sum_{t=1}^{T} (x^t)^* C y^t + MT \right] &\le \frac{1}{2M} \left[\sum_{t=1}^{T} e_i^* C y^t + MT \right] + 2\sqrt{T \ln(n)}, \quad \forall i \\ (\times 2M) \quad \Rightarrow \quad \sum_{t=1}^{T} (x^t)^* C y^t &\le \sum_{t=1}^{T} e_i^* C y^t + 4M\sqrt{T \ln(n)}, \qquad \forall i. \end{aligned} \tag{3}$$

Similiarly, we bound the cumulative loss for column player's experts algorithm,

$$\begin{aligned}
L_{col}^T &\leq \min_j L_j^T + 2\sqrt{T\ln(n)} \leq L_j^T + 2\sqrt{T\ln(n)}, \qquad \forall j \\
L_j^T &= \sum_{t=1}^T (\ell_{col}^t)^* e_j = \sum_{t=1}^T (x^t)^* R' e_j = \sum_{t=1}^T (x^t)^* \left(\frac{1}{2M}[R + M\mathbb{1}]\right) e_j \\
&= \frac{1}{2M}\left[\sum_{t=1}^T (x^t)^* R e_j + M\sum_{t=1}^T (x^t)^* \mathbb{1} e_j\right] = \frac{1}{2M}\left[\sum_{t=1}^T (x^t)^* R e_j + MT\right] \\
\Rightarrow L_{col}^T &\leq \frac{1}{2M}\left[\sum_{t=1}^T (x^t)^* R e_j + MT\right] + 2\sqrt{T\ln(n)}, \qquad \forall j,
\end{aligned}$$

which we combine with equation 2 and simplify,

$$\begin{aligned}
\frac{1}{2M}\left[\sum_{t=1}^T (x^t)^* R y^t + MT\right] &\leq \frac{1}{2M}\left[\sum_{t=1}^T (x^t)^* R e_j + MT\right] + 2\sqrt{T\ln(n)}, \quad \forall j \\
(\times 2M) \quad \Rightarrow \quad \sum_{t=1}^T (x^t)^* R y^t &\leq \sum_{t=1}^T (x^t)^* R e_j + 4M\sqrt{T\ln(n)}, \qquad \forall j. \qquad (4)
\end{aligned}$$

Equations 3 and 4 tell us that the cumulative losses of the row and column player's experts algorithms are bounded by the no-regret loss times a multiplicative term, which is linear in the maximum absolute value M in the payoff tables. In the following theorem, we see what these bounds tell us about the average payoff of the game.

Theorem 2. *If $(x^1, x^2, ..., x^T)$ and $(y^1, y^2, ..., y^T)$ are the sequences of strategies generated for the row player and column player respectively by the MWU algorithm, then*

$$\left(\frac{1}{T}\sum_{t=1}^T x^t, \frac{1}{T}\sum_{t=1}^T y^t\right) \textit{ is a } \left(8M\sqrt{\frac{\ln(n)}{T}}\right) - \textit{approximate Nash Equilibrium.}$$

Proof: For an ϵ-approximate N.E., we must show the following:

$$\begin{aligned}
\left(\frac{1}{T}\sum_{t=1}^T x^t\right)^* R\left(\frac{1}{T}\sum_{t=1}^T y^t\right) &\geq e_i^* R\left(\frac{1}{T}\sum_{t=1}^T y^t\right) - \epsilon, \quad \forall i \in \{1, ..., m\} \\
\left(\frac{1}{T}\sum_{t=1}^T x^t\right)^* C\left(\frac{1}{T}\sum_{t=1}^T y^t\right) &\geq \left(\frac{1}{T}\sum_{t=1}^T x^t\right)^* C e_j - \epsilon, \quad \forall j \in \{1, ..., n\}.
\end{aligned}$$

We begin by replacing C with $-R$ (since $R+C=0$) in equation 3,

$$\begin{aligned}
\sum_{t=1}^{T}(x^t)^*(-R)y^t &\le \sum_{t=1}^{T} e_i^*(-R)y^t + 4M\sqrt{T\ln(n)}, \quad \forall i \\
\Rightarrow \quad -\sum_{t=1}^{T}(x^t)^*Ry^t &\le -\sum_{t=1}^{T} e_i^*Ry^t + 4M\sqrt{T\ln(n)}, \quad \forall i \\
\Rightarrow \quad \sum_{t=1}^{T}(x^t)^*Ry^t &\ge \sum_{t=1}^{T} e_i^*Ry^t - 4M\sqrt{T\ln(n)}, \quad \forall i \\
\Rightarrow \quad \sum_{t=1}^{T}(x^t)^*Ry^t &\ge e_i^*R\sum_{t=1}^{T} y^t - 4M\sqrt{T\ln(n)}, \quad \forall i.
\end{aligned}$$

Next we rearrange the terms in equation 4 and apply the above bound,

$$\begin{aligned}
\sum_{t=1}^{T}(x^t)^*Re_j &\ge \sum_{t=1}^{T}(x^t)^*Ry^t - 4M\sqrt{T\ln(n)}, \quad \forall j \\
\Rightarrow \quad \min_j\left[\left(\sum_{t=1}^{T}x^t\right)^*Re_j\right] &\ge \sum_{t=1}^{T}(x^t)^*Ry^t - 4M\sqrt{T\ln(n)} \\
\Rightarrow \quad \min_j\left[\left(\sum_{t=1}^{T}x^t\right)^*Re_j\right] &\ge e_i^*R\sum_{t=1}^{T}y^t - 8M\sqrt{T\ln(n)}, \quad \forall i.
\end{aligned}$$

Finally we introduce the average strategy $\frac{1}{T}\sum_t y^t$ which is a probability distribution over the deterministic strategies e_j. We use the fact that any linear function applied to $\frac{1}{T}\sum_t y^t$ must have value at least as large as the minimum value it takes over the deterministic strategies e_j

$$\begin{aligned}
\left(\sum_{t=1}^{T}x^t\right)^*R\left(\frac{1}{T}\sum_{t=1}^{T}y^t\right) &\ge \min_j\left[\left(\sum_{t=1}^{T}x^t\right)^*Re_j\right] \\
\Rightarrow \quad \left(\sum_{t=1}^{T}x^t\right)^*R\left(\frac{1}{T}\sum_{t=1}^{T}y^t\right) &\ge e_i^*R\sum_{t=1}^{T}y^t - 8M\sqrt{T\ln(n)}, \quad \forall i \\
\left(\times\frac{1}{T}\right)\Rightarrow \left(\frac{1}{T}\sum_{t=1}^{T}x^t\right)^*R\left(\frac{1}{T}\sum_{t=1}^{T}y^t\right) &\ge e_i^*R\left(\frac{1}{T}\sum_{t=1}^{T}y^t\right) - 8M\sqrt{\frac{ln(n)}{T}}, \quad \forall i.
\end{aligned}$$

Following the same construction for column player, that is by replacing R by $-C$ in equation 4 and rearranging the terms in equation 3, we find the following:

$$\left(\frac{1}{T}\sum_{t=1}^{T}x^t\right)^*C\left(\frac{1}{T}\sum_{t=1}^{T}y^t\right) \ge \left(\frac{1}{T}\sum_{t=1}^{T}x^t\right)^*Ce_j - 8M\sqrt{\frac{ln(n)}{T}}, \quad \forall j$$

□

Remark 1. *Theorem 2 demonstrates two important facts about the two-player zero-sum game as the time horizon increases (i.e. $T\to\infty$):*
1) The average payoffs obtained by the row and column player of the game converge to their values in the game.
2) (Informal) The histogram of play, or average strategies of the players, converges to an equilbrium of the game.

Remark 2. *In Thereom 2, the* ϵ *error term decreases as the inverse square root of time. It is an open question whether there exists a simple no-regret algorithm for zero-sum games converging faster (e.g. exponentially fast).*