Minimax Theorem via No-Regret Learning

Lecture 19

We consider repeated decision-making problems with uncertainty. For example, this could be a person commuting to work every day deciding which route, out of several possible options, to take, or a news station deciding which meteorologist's report to trust for their prediction of the weather. The *regret* of the decision maker is the improvement in performance that would have been possible if they had known the underlying truth, like which route is, on average, the fastest or which meteorologist is, on average, most accurate. We will see *no-regret learning* algorithms that make the average regret of the decision maker approach zero.

Using these algorithms, we will fulfill a promise from earlier in the semester and prove von Neumann's Minimax Theorem.

1 No-Regret Learning

Definition 1 (Regret in Online Decision Making)**.** In *online decision making*, an agent has to choose one of the n actions in a set A repeatedly for T rounds. After picking an action $a_t \in A$ in round t, they observe the incurred *cost* (or loss) $c_t(a) \in [0,1]$ in round t for all possible choices $a \in A$ they could have made. They have to pay $c_t(a_t)$, the cost of the action they picked.

The *regret* of the agent after T rounds is how much better they could have performed in hindsight by picking the *best fixed* action. That is,

$$\text{Regret}(T) = \sum_{t=1}^{T} c_t(a_t) - \min_{a \in A} \sum_{t=1}^{T} c_t(a),$$

where the first term is the total cost of the agent over T rounds from the actions they picked while the second term is the lowest cost possible for the agent from choosing the same, fixed action in every round.[1] The *average regret* after T rounds is $1/T \cdot \text{Regret}(T)$.

We can model this learning process through the interaction of the agent with an adversary on a matrix C. In this matrix, there is one row for every action the agent can take, and one column (potentially infinitely many) for every possible cost function c_t they could face. In each round $t = 1, 2, \cdots, T$,

- first, the agent selects a row i_t (an action from A),
- second, the adversary selects a column j_t (the environment, i.e., the cost function), and
- finally, the agent incurs the cost $C(i_t, j_t)$, the entry of C in the chosen column and row.

Thus, the average regret of the agent after T rounds is

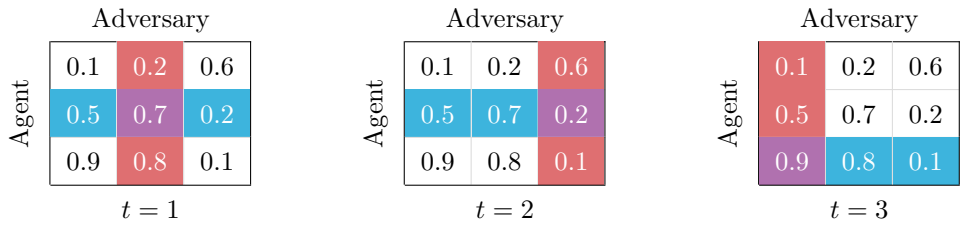
$$\frac{1}{T} \cdot \text{Regret}(T) = \frac{1}{T} \sum_{t=1}^{T} C(i_t, j_t) - \min_{\text{row } i} \left(\frac{1}{T} \sum_{t=1}^{T} C(i, j_t) \right).$$

Example 1 (Regret)**.** Consider the matrix shown below

		Adversary		
Agent	0.1	0.2	0.6
	0.5	0.7	0.2
	0.9	0.8	0.1

[1] In problems where the agent in each round t doesn't pick actions to minimize an (undesired) cost c_t but instead to maximize a (desired) reward r_t, we define the regret over T rounds as $\text{Regret}(T) = \max_{a \in A} \sum_{t=1}^{T} r_t(a) - \sum_{t=1}^{T} r_t(a_t)$

The choices of the agent (blue) and the adversary (red) for the first three rounds are shown below. For example, in round $t = 1$, the agent chooses the middle row and the adversary reacts by choosing the middle column. Thus, the agent incurs a cost of 0.7 in this round.



The total cost of the agent incurred over the $T = 3$ rounds is $0.7 + 0.2 + 0.9 = 1.8$. Knowing the choices of the adversary, the best fixed action the agent could have picked is the top row, which would have led to a cost of $0.2 + 0.6 + 0.1 = 0.9$. Thus, the average regret of the agent in this case is $\frac{1}{3}(1.8 - 0.9) = 0.3$ — they could have on average incurred 0.3 less cost per round.

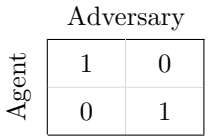
Importantly, note that regret is not measured with respect to the optimal row in every round independently, but with respect to the best *fixed* row.

A desirable property for an algorithm used by the agent in this setting would be to have negligible regret as the number of rounds is large enough:

Definition 2 (No-Regret Algorithm)**.** An algorithm is a *no-regret algorithm* if the average regret goes to 0, $1/T \cdot \text{Regret}(T) \to 0$, as $T \to \infty$.

Let's examine two natural algorithms for regret minimization and see if they are no-regret algorithms.

Example 2 (Learning Algorithms)**.** Consider the following cost matrix C:



Let's first consider the algorithm where the agent alternates between choosing the top row and the bottom row. The adversary reacts to each action picked by the agent with the column that gives the agent a cost of 1 — left if the agent chose up and right if the agent chose down. Thus, the average cost of the agent will be $1/T \cdot \sum_{t=1}^{T} C(i_t, j_t) = 1$. At the same time, the cost from picking any fixed action is $1/T \cdot \sum_{t=1}^{T} C(r, j_t) \approx 1/2$. Thus, the average regret after T rounds is $1/T \cdot \text{Regret}(T) \approx 1/2$, so does not go to 0.

Alternatively, the agent may in any round choose the next action greedily to pick the row that has led to less cost so far (and breaking ties in favor of the top row). However, if the adversary reacts, again, with the action that gives the agent a cost of 1, this indeed results in the exact same algorithm as considered before.

Example 2 illustrates the limitations of deterministic algorithms in an adversarial learning setting. In fact, in the example, the average regret of *any* deterministic algorithm is at least $1/2$: In each round t, regardless of the algorithm's choice, the adversary can select the column for which the agent has a cost of 1, while the other action has no cost. In that case, at least one of the two possible fixed actions incurs a cost in at most $1/2 \cdot T$ rounds. Thus, the average regret is at least $1/2$.

2 Expert Advice and the Weighted Majority Algorithm

We'll now explore a slightly more structured version of this general setting in which the adversary is a bit more restricted. We'll find that a randomized no-regret algorithm exists. In fact, it turns out that this algorithm can be generalized to even be no-regret in the general setting described above.

Definition 3 (Expert Advice Model)**.** In each round t, the agent receives binary *advice* (say, $+$ or $-$) from each of n experts. Seeing this advice, the agent has to make a binary *prediction*. Then, the adversary decides on the true binary outcome. The goal of the agent is to predict outcomes almost as accurately as the best expert in hindsight. Thus, their regret is

$$\text{Regret}(T) = M^T - \min_{i \in [n]} m_i^T,$$

where M^T is the number of times the prediction of the agent was wrong (up to round T) and m_i^T is the number of times the advice of expert i was wrong (up to round T).

Example 3 (Expert Advice Model)**.** Assume there are 4 experts that give binary advice in $T = 3$ rounds as shown in the table below. In each round, based on the advice, the agent needs to make a prediction, which may be correct (as in $t = 1$ and $t = 2$) or incorrect (as in $t = 3$). The colors, green and red, encode whether a piece of advice or the prediction was correct or not.

	Exp. 1	Exp. 2	Exp. 3	Exp. 4	Prediction	Outcome
$t = 1$	$+$	$-$	$-$	$+$	$+$	$+$
$t = 2$	$-$	$-$	$+$	$-$	$-$	$-$
$t = 3$	$+$	$-$	$-$	$-$	$+$	$-$

Intuitively, when making a prediction, the agent should trust experts that were right many times in the past more. We'll first examine a deterministic algorithm following the principle:

Algorithm 1 Weighted Majority Algorithm

1: Initially, give each expert a weight of 1.
2: Make predictions according to the weighted majority.
3: After each round, penalize each incorrect expert by halving their weights.

Example 4 (Weighted Majority Algorithm)**.** Assume there are 4 experts. In round $t = 1$, all experts have weight 1. They give respective advice $-, +, +, +$. Since the weighted majority of advice, 3 over 1, is $+$, the weighted majority algorithm chooses $+$ as its prediction. If this turns out to be correct, the weight of the wrong expert, expert 1, is halved to $1/2$. In the next round, $t = 2$, the respective advice is $+, +, -, -$. Since the weight of expert 1 is only $1/2$, the weighted majority of advice, 2 over 1.5, is $-$, so $-$ is chosen as the prediction. If this turns out to be wrong, the weight of the wrong experts, expert 3 and expert 4, are halved. This process is shown below.

		Exp. 1	Exp. 2	Exp. 3	Exp. 4	Prediction	Outcome
$t = 1$	Weight	1	1	1	1	3 : 1	
	Advice	$-$	$+$	$+$	$+$	$+$	$+$
$t = 2$	Weight	~~1~~ $1/2$	1	1	1	1.5 : 2	
	Advice	$+$	$+$	$-$	$-$	$-$	$+$
$t = 3$	Weight	$1/2$	1	~~1~~ $1/2$	~~1~~ $1/2$		
	Advice						

Theorem 1. *The number of mistakes made by the weighted majority algorithm is at most* $2.5(m + \log_2 n)$*, where* m *is the number of mistakes made by the best expert.*

Proof. Let M be the number of mistakes by the algorithm and let W be the total weight of all experts. Initially, $W = n$. Each time the algorithm makes a mistake, at least half of the weighted vote was incorrect.

Since at least half the total weight is held by incorrect experts and their weights are halved, the total weight drops by at least $1/4$ every time the algorithm makes a mistake. Thus

$$W \leq n \cdot \left(\frac{3}{4}\right)^M.$$

Meanwhile, the best expert, who has made m mistakes, has weight $w = (1/2)^m$. Since the best expert's weight is a part of the total weight, it has to hold that

$$\left(\frac{1}{2}\right)^m \leq n \cdot \left(\frac{3}{4}\right)^M.$$

Taking logarithms of both sides and solving for M, we get that

$$M \leq \frac{m \log_2 2 + \log_2 n}{2 \log_2 2 - \log_2 3} \leq 2.5(m + \log_2 n) \qquad \square$$

This is some progress in the right direction! We know that under the weighted majority algorithm, the agent won't make more than 2.5 times the number of mistakes of the best expert (plus a term of $\log_2 n$ that is negligible as $T \to \infty$). However, this is not a no-regret algorithm as m may be on the order of T, causing the regret $M - m$ to be on the order of T (so the average regret to be at least a non-zero constant).

We may hope that modifying the constant of $1/2$ used in the weighted majority algorithm may help:

Algorithm 2 Modified Weighted Majority Algorithm

1: Initially, give each expert a weight of 1.
2: Make predictions according to the weighted majority.
3: After each round, penalize each incorrect expert by taking away an ε-fraction of their weight.

However, also this algorithm is not a no-regret algorithm: In fact, no deterministic algorithm can have average regret less than $1/2$ in the worst case. To see this, consider the case of there being two experts, one whose advice always is $+$ and one whose advice always is $-$. Thus, exactly one expert is right in each round and we want to determine which; this is equivalent to Example 2, for which we argued above that no deterministic algorithm can have average regret less than $1/2$.

Therefore, we need to consider randomized algorithms to find a no-regret algorithm. Intuitively, the worst case scenario for the weighted majority algorithm is when the advice of the experts is split roughly evenly: If we make a mistake, only about a quarter of the total weight is lost, so, the number of 'good' experts that made a mistake is not very large. In contrast, if the weight is split 90% : 10%, this is a lot better for us: If we make a mistake we know that almost half of the total weight is lost, so, that the number of 'good' experts that made a mistake is also large. To alleviate the bad cases, we try to 'smooth them out' by using randomization.

Algorithm 3 Randomized Weighted Majority Algorithm (for expert setting)

1: Initially, give each expert a weight of 1.
2: In each round, let w_+ and w_- be the total weight of experts with advice $+$ and $-$, respectively.
3: Predict $+$ with probability $\frac{w_+}{w_+ + w_-}$; predict $-$ with probability $\frac{w_-}{w_+ + w_-}$.
4: After each round, penalize each incorrect expert by taking away an ε-fraction of their weight.

Theorem 2. *For suitable ε, the randomized weighted majority algorithm for the expert advice model has (expected) regret at most $2\sqrt{T \ln n}$. Since the average regret $\frac{2\sqrt{T \ln n}}{T} \to 0$ as $T \to \infty$, the randomized weighted majority algorithm is a no-regret algorithm.*

The proof idea is similar to the proof of Theorem 1. It lower- and upper-bounds the remaining weight with the number of mistakes by the best expert and the algorithm to bound the difference between those two.

It turns out that the randomized weighted majority algorithm can be applied to the more general learning setting we described at the beginning of the lecture. The idea in the general setting is the same: Each action (formerly, an expert) has a weight and we sample the action to take (formerly, the expert to trust) according to this weight. We then update the weight of *each* action based on the cost if we had taken it (formerly, the cost was 1 if and only if the expert was wrong):

Algorithm 4 Randomized Weighted Majority Algorithm (for general setting)

1: Initially, give each action $a_i \in A$ a weight of $w_i = 1$.
2: In each round, pick action a_i with probability $\frac{w_i}{\sum_{a_j \in A} w_j}$
3: After each round, penalize each action $a_i \in A$ according to the cost it would have caused, by setting its weight to $w_i(1 - \varepsilon \cdot c_t(a_i))$ where c_t is the cost function the adversary picked in this round.

Theorem 3. *For suitable ε, the randomized weighted majority algorithm has (expected) regret at most $2\sqrt{T \ln n}$ in the general online decision making setting. Since average regret $\frac{2\sqrt{T \ln n}}{T} \to 0$ as $T \to \infty$, the randomized weighted majority algorithm is a no-regret algorithm.*

This is a surprising and strong result: Even though there are no restrictions on the adversary (except, that the cost of any action in any round is in a bounded range, like $[0, 1]$), this *online* algorithm has negligible regret to the best fixed action *in hindsight!*

3 Proving the Minimax Theorem via No-Regret Learning

As promised at the beginning of the semester, we now prove the Minimax Theorem, using Theorem 3.

Theorem 4 (von Neumann, 1928)**.** *Every (finite) two-player zero-sum game has a unique value v such that*

- *player 1 (the row player) can guarantee a utility of at least v, and*
- *player 2 (the column player) can guarantee a utility of at least $-v$,*

if this player has to commit to a strategy first and the other player reacts optimally.

This theorem implies that a Nash equilibrium always exists in (finite) zero-sum games.

Proof. Given any zero-sum game, let V_R be the largest utility the row player can guarantee if they commit first. Let $-V_C$ be the largest utility the column player can guarantee if they commit first, so that V_C is the smallest utility of the row player that the column player can guarantee if they commit first.

First, we can see that $V_C \geq V_R$, since going second cannot harm a player. To confirm this, assume towards a contradiction that $V_C < V_R$. Then, after the column player commits to the strategy that guarantees them V_C, the row player could play the strategy they committed to to guarantee V_R to get utility at least V_R. Thus, the row player gets utility strictly greater than V_C — a contradiction to the column player being able to guarantee that the row player's utility is at most V_C.

To show the minimax theorem, we need to prove that $V_C = V_R$. We do this by assuming $V_C > V_R$ and showing that this leads to a contradiction:

Given a zero-sum game with $V_C > V_R$, we first shift and scale the entries in the payoff matrix of the game (we can do this without changing the relative order of the entries, so $V_C > V_R$ still holds) so that the utility of the row player in the game under any pair of strategies is in $[-1, 0]$. Thus, we can now interpret the (negative) utility of the row player as a cost in online decision making. Furthermore, let $\delta = V_C - V_R > 0$.

Let's assume that the zero-sum game is played repeatedly. In each round, the row player commits to a mixed strategy, to which the column player responds optimally (after observing the mixed strategy but before seeing the outcome of the randomness). Let's assume that the row player picks their strategy in each round according to the randomized weighted majority algorithm, where each possible pure strategy is one action and the cost of an action is the negative utility the row player gets from playing it.

By Theorem 3, we know that after T rounds of this online decision making process (T rounds of the game), it holds that

$$\text{(total exp. cost of agent under algorithm)} - \text{(total cost from best fixed action)} \leq 2\sqrt{T \ln n},$$

or equivalently, if U_{ALG} is the total utility of the row player over the T rounds and $U_{\text{best action in hindsight}}$ is the utility of the best fixed action (given the observed responses of the column player),

$$U_{\text{ALG}} \geq U_{\text{best action in hindsight}} - 2\sqrt{T \ln n}.$$

At the same time, it holds that $U_{\text{ALG}} \leq T \cdot V_R$ since V_R is the largest utility the row player can guarantee if they commit to any mixed strategy first and the column player responds optimally, so an upper bound on the expected utility of the row player in any round when they commit to the randomized weighted majority algorithm strategy.

We'll now show that $U_{\text{best action in hindsight}} \geq T \cdot V_C$. Suppose the column player played strategies $s_1, s_2, \ldots, s_T$ in the T rounds. Define the mixed strategy y for the column player that plays each s_t with probability $1/T$ (adding the probabilities up if a strategy appears multiple times in $s_1, s_2, \ldots, s_T$). Let x be the row player's best response to y. Then it holds that

$$V_C \leq u_1(x, y) = \frac{1}{T}\left(u_1(x, s_1) + \cdots + u_1(x, s_T)\right),$$

since the column player is committing first to y and the row player is best responding with x. Thus, by definition, the utility of the column player is at most $-V_C$ and, thus, the utility of the row player is at least V_C. Moreover, since there always exists a pure best response, we can assume that x is a pure strategy, so that x itself is an action. In other words,

$$u_1(x, s_1) + \cdots + u_1(x, s_T) = U_{\text{best action in hindsight}}.$$

Now, putting the two inequalities together proves the claim.

Since we know that $T \cdot V_R \geq U_{\text{ALG}}$ and $U_{\text{ALG}} \geq T \cdot V_C - 2\sqrt{T \ln n}$, it follows that

$$V_R \geq V_C - \frac{2\sqrt{T \ln n}}{T} \quad \Longrightarrow \quad \frac{2\sqrt{T \ln n}}{T} \geq V_C - V_R > \delta.$$

Since $\frac{2\sqrt{T \ln n}}{T} \to 0$ as $T \to \infty$, we get a contradiction for any constant δ for T large enough. Thus, we can conclude that $V_R = V_C$ □