Prediction and Learning: It's Only a Game - Homework #3

Jacob Abernethy Due: 11/27/2013

Homework Policy: Working in groups is fine. Please write the members of your group on your solutions. There is no strict limit to the size of the group but we may find it a bit suspicious if there are more than 4 to a team. Questions labelled with **(Challenge)** are not strictly required, but you'll get some participation credit if you have something interesting to add, even if it's only a partial answer.

1) **Generalized Minimax Theorem. (20 pts)** Let $X \subset \mathbb{R}^n$ and $Y \subset \mathbb{R}^m$ be convex compact sets. Let $f : X \times Y \to \mathbb{R}$ be some differentiable function with bounded gradients, where $f(\cdot, \mathbf{y})$ is convex in its first argument for all fixed $\mathbf{y}$, and $f(\mathbf{x}, \cdot)$ is concave in its second argument for all fixed $\mathbf{x}$.

Prove that

$$\inf_{\mathbf{x}\in X} \sup_{\mathbf{y}\in Y} f(\mathbf{x}, \mathbf{y}) = \sup_{\mathbf{y}\in Y} \inf_{\mathbf{x}\in X} f(\mathbf{x}, \mathbf{y}).$$

And, furthermore, give an efficient algorithm for finding an ϵ-optimal pair $(\mathbf{x}^*, \mathbf{y}^*)$.

2) **Online Non-Convex Optimization.** Sometimes our nice assumptions don't always hold. AWWW SHUCKS!! But maybe things will still work out just fine. For the rest of this problem assume that $X \subset \mathbb{R}^n$ is the learner's decision set, and the learner observes a sequence of functions $f_1, f_2, \ldots, f_T$ mapping $X \to \mathbb{R}$. The regret of an algorithm choosing a sequence of $\mathbf{x}_1, \mathbf{x}_2, \ldots$ is defined in the usual way:

$$\text{Regret}_T := \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x}\in X} \sum_{t=1}^{T} f_t(\mathbf{x})$$

(a) (15 pts) It would really suck if X were not actually convex, wouldn't it? Prove that, as long as X is bounded and the functions f_t are convex, there exists a randomized algorithm that guarantees an expected regret bound that scales as $\sqrt{T}$. (Hint: You might want to think about the convex hull of X. And let's not concern ourselves with efficiency.)

UPDATE: First, one student asked what it means to be "convex when the domain X is not convex", and this is a good point. To clarify, here you can assume that f_t is defined as the restriction of some convex function defined on all of $\mathbb{R}^n$.
Second, the convex analysis issues here are somewhat more difficult that I had intended. To get full credit you can solve one of two special cases: (i) assume the functions f_t are linear on all $\mathbb{R}^n$ or (ii) assume that $n = 1$ and we are in a 1-dimensional setting, and the f_ts are convex although not necessarily linear. I will leave the general problem as a nice (**Challenge**).

(b) (15 pts) Wouldn't it ruin your lovely day if the functions f_t were not convex? Maybe the only two conditions you can guarantee is that the functions f_t are bounded (say in $[0, 1]$) and are 1-*Lipschitz*: they satisfy that $|f_t(\mathbf{x}) - f_t(\mathbf{x}')| \leq \|\mathbf{x} - \mathbf{x}'\|_2$. Prove that, assuming X is convex and bounded, there exists a randomized algorithm with a decent expected regret bound. Something like $\sqrt{nT \log T}$ would be admirable. (Hint: Always good to ask the *experts* for ideas. And you needn't worry about efficiency.)

3) **Information-theoretic Lower Bound for EXP3.** **(25 pts)** Someone hands you N coins and tells you that all BUT ONE are symmetrically weighted: each will come up heads exactly half the time. But there's one odd coin $I \in [N]$ that lands heads with probability $\frac{1}{2} + \epsilon$ for some $\epsilon > 0$. Your task is to sequentially select coins to flip, in any manner you choose, and then to make a guess $\hat{I} \in [N]$ which is the odd coin.

CLAIM(*): If you perform fewer than $\frac{cN}{\epsilon^2}$ coin flips, where $c > 0$ is some constant, then $\Pr(\hat{I} = I) \leq 3/4$. In other words, no algorithm will have a very good chance to guess the odd coin with so few coin tosses.

It turns out that the above claim is true, but we will not concern ourselves with proving it here. Instead we will use it to construct a lower bound. Recall that the EXP3 algorithm has a regret bound on the order of $\sqrt{TN \log N}$ where N is the number of arms and T is the length of the sequence. Prove that if we could find an algorithm with a slightly better regret bound, say $O((NT)^{\frac{1}{2}-\delta})$ for some $\delta > 0$, we could then violate **CLAIM(*)**.

UPDATE: The version of this problem that I intended is slightly more difficult: I *should have* said that if you can find an algorithm with regret bound of the form $O(T^{\frac{1}{2}} N^{\frac{1}{2}-\delta})$ then you can violate **CLAIM(*)**. You are welcome to solve this as originally stated for full credit, but the latter case is more interesting since it shows precisely that the $\sqrt{N}$ dependence is fundamental to the bandits problem.

4) **Competing Against Changing CRPs.** **(25 pts)** Recall that we had a modification to the EWA algorithm earlier in the course in which we were able to compete against the best *changing sequence* of experts, as long as the expert we were competing against didn't switch too many times. We achieved a regret bound on the order of $\sqrt{kT(\log N + \log T)}$ where k was the number of switches.

Let's now try to do the same thing for universal portfolios! The inputs to the algorithm are a sequence of price relative vectors for n assets, $\mathbf{b}^1, \mathbf{b}^2, \ldots \in (0, \infty)^n$ and the algorithm must select a sequence of portfolios $\mathbf{w}^1, \mathbf{w}^2, \ldots \in \Delta_n$. The basic UCPR strategy is as follows: if $V_t(\mathbf{w}) := \prod_{s=1}^{t} \mathbf{w} \cdot \mathbf{b}^s$ is the (multiplicative) wealth earned by CRP $\mathbf{w}$, then universal CRP algorithm should choose a portfolio on day $t+1$ defined by

$$\mathrm{UCRP}(\mathbf{b}^1, \ldots, \mathbf{b}^t) := \frac{\int_{\mathbf{w} \in \Delta_n} \left(\prod_{s=1}^{t} \mathbf{w} \cdot \mathbf{b}^s \right) \mathbf{w} d\mathbf{w}}{\int_{\mathbf{w} \in \Delta_n} \left(\prod_{s=1}^{t} \mathbf{w} \cdot \mathbf{b}^s \right) d\mathbf{w}}$$

In other words, the portfolio should be the average of all portfolios $\mathbf{w}$ in the simplex, but weighted by the function $V_t(\mathbf{w})$, the wealth earned by $\mathbf{w}$.

Here's a proposed strategy for competing against changing CRPs: be forgetful! Mix together a sequence of UCRP algorithms, each of which has forgotten some set of days in the past. Specifically, define

$$\widehat{\mathrm{UCRP}}(\mathbf{b}^1, \ldots, \mathbf{b}^t) := \frac{1}{t+1} \sum_{s=0}^{t} \mathrm{UCRP}(\mathbf{b}^{s+1}, \mathbf{b}^{s+2}, \ldots, \mathbf{b}^t).$$

where the last term in the sum is the just the initial UCRP strategy $\mathrm{UCRP}(\emptyset)$.

Prove that $\widehat{\mathrm{UCRP}}$ can compete against any sequence of best-in-hindsight CRPs $\mathbf{w}_*^1, \mathbf{w}_*^2, \ldots, \mathbf{w}_*^T$, as long as $\sum_{t=1}^{T} \mathbf{1}[\mathbf{w}_*^t \neq \mathbf{w}_*^{t+1}] \leq k$, with a regret bound that is roughly $O(kn \log T)$. Recall that the regret here is defined as the log wealth ratio; that is, if the algorithm chooses a sequence of portfolios $\mathbf{w}^1, \ldots, \mathbf{w}^T$ then

$$\mathrm{Regret}_T := \log \left(\frac{\prod_{t=1}^{T} \mathbf{w}_*^t \cdot \mathbf{b}^t}{\prod_{t=1}^{T} \mathbf{w}^t \cdot \mathbf{b}^t} \right)$$