CSE599s Spring 2014 - Online Learning
Theoretical Homework Exercise 1 - due 4/24/14

Online optimization is the following repeated game:

> player and adversary agree on the terms of the game: a set of points $\mathcal{W}$, a set of loss functions $\mathcal{F}$, and the length of the game T
> **for** $t = 1, 2, \ldots, T$ **do**
> player chooses a point $w_t \in \mathcal{W}$
> adversary chooses a function $f_t \in \mathcal{F}$
> player incurs a loss of $f_t(w_t)$ and observes the function f_t
> **end for**

The player's *regret* after T rounds is defined as

$$\sum_{t=1}^{T} f_t(w_t) \;-\; \min_{w \in \mathcal{W}} \sum_{t=1}^{T} f_t(w) \quad .$$

A *regret upper-bound* is a function $R(T)$ such that for any T and any sequence $f_1, \ldots, f_T$ of functions in $\mathcal{F}$ it holds that

$$\sum_{t=1}^{T} f_t(w_t) \;-\; \min_{w \in \mathcal{W}} \sum_{t=1}^{T} f_t(w) \;\leq\; R(T) \quad .$$

1 Conservative Updates

a. Prove that we can assume, without loss of generality, that $\min_x f_t(x) = 0$ for each t.

b. Making the above assumption, an online optimization algorithm is *conservative* if

$$f_t(w_t) = 0 \quad \Rightarrow \quad w_{t+1} = w_t \quad .$$

In other words, a conservative algorithm keeps playing the same point as long as it doesn't suffer any loss. Let A be an online optimization algorithm (not necessarily conservative) with a regret bound of $R(T)$. Use A as a black-box to construct a conservative online optimization algorithm A' with the same regret bound.

2 The Doubling Trick

You are given an online optimization algorithm A that guarantees a regret upper-bound of $R(T) = T^p$, for some $p \in (0,1)$, as long as its parameters are set as a function of T. We will use A as a black-box to construct another online optimization algorithm A', which guarantees a regret upper-bound of $\mathcal{O}(T^p)$, and which does not need to know the length of the game in advance. In other words, the regret upper-bound of A' holds simultaneously for all lengths T. In particular, define A' as follows:

```
for epoch m = 0, 1, 2, ... do
  Reset A with parameters chosen for T = 2^m
  for rounds t = 2^m, ..., 2^{m+1} − 1 do
    Run A
  end for
end for
```

Essentially, the algorithm initially guesses $T = 1$, and when it observes this guess was too low, it doubles it's initial guess and re-starts A. This is called the *doubling trick*.

a. Prove that, for any T, the regret up to time T is upper-bounded by $\sum_{m=0}^{\lceil \log_2(T) \rceil} R(2^m)$ (keep in mind that regret can be negative).

b. Prove that the regret of A' up to time T is $\mathcal{O}(T^p)$ (hint: for any $x \neq 1$, it holds that $\sum_{k=0}^{n} x^k = \frac{x^{n+1}-1}{x-1}$).

3 An Application

You are working on a team developing a smartphone app that every morning predicts how many emails the phone's owner will receive that day. You construct feature vectors $x_t \in \mathbb{R}^n$ with $\|x_t\|_2 = 1$ such that you can predict the number of emails as a linear function of a x_t using a model $w \in \mathbb{R}^n$, that is, the prediction is $w \cdot x_t$. The app can observe the actual number of emails received, y_t, at the end of the day, so the problem fits nicely into the online model. You apply Online Gradient Descent with convex loss function $f_t(w) = |w \cdot x_t - y_t|$. Assume we know T, and we take $W = \{w \mid \|w\|_2 \leq D\}$.

a. Give a bound G such that $\|g\|_2 \leq G$ for all $g \in \partial f_t(w_t)$ for any $w_t \in W$.

b. Using the above G, what learning rate do we need in order to get a regret bound of $GD\sqrt{2T}$ for Online Gradient Descent?

Thinking about the problem, someone on your team suggests: "It would be nice to get low regret with respect to a comparator strategy that can use one vector $w^a \in W$ on weekdays, and a different model $w^b \in W$ on weekend days."

c. Describe a transformation that produces a single online convex optimization problem (which can be solved using a single instance of OGD as a subroutine) that gives a regret guarantee against the best pair of models (w^a, w^b). Hint: You will need to transform both the loss functions and the points played, and the dimensionality of the problem will change.

d. What learning rate does the theory recommend for the transformed problem? What is the regret bound against the stronger comparator class used in the transformed problem?

e. The above regret bounds do not imply that one approach or the other will have less cumulative loss. Describe an adversary (some process for generating examples (x_t, y_t)) for which you would expect the original approach to have lower cumulative loss. Describe a scenario where you would expect the transformed approach to have lower cumulative loss. Make these as realistic as possible; they can be described in general terms as long as it is clear what the impact on the cumulative loss would be.

f. An alternative approach is to simply use two separate copies of OGD: for predictions on weekends you predict and train with one instance of OGD, and for predictions on weekdays you use a separate OGD instance. Suppose 5/7 of the T examples are for weekdays, and 2/7 are for weekends (assume T is divisible by 7). As a function of the total T, how would you set the learning rates for the weekend and weekday algorithms (again, based on the theory)? (Keep in mind neither algorithm will see a all T examples). Give an overall bound on the total cumulative regret for all T predictions against the comparator class that can use a separate $w \in W$ for weekdays and weekends. Compare this bound to the one from part d. Is this bound better or worse? Why?