CS787: Advanced Algorithms

Topic: Online Learning **Presenters:** David He, Chris Hopman

17.3.1 Follow the Perturbed Leader

17.3.1.1 Prediction Problem

Recall the prediction problem that we discussed in class. In that problem, there are n experts and on day t expert i makes prediction $p_{i,t} \in \{-1, 1\}$. Based upon the experts' predictions, we decide to predict -1 or 1. We then observe the state of the day, s_t. The cost of the algorithm is the total number of mistakes that it makes. Our goal is to bound the number of mistakes against those of the best expert.

Weighted Majority Algorithm

In class we analyzed the weighted majority algorithm. This algorithm runs as follows:

1. Initialize $w_{i,0} = 1$ for each expert i.
2. On day t, do:
   (a) If $\sum_i w_{i,t} p_{i,t} \geq 0$ predict 1, otherwise predict -1.
   (b) For each i, $w_{i,t+1} = \begin{cases} w_{i,t} \text{ if expert } i \text{ made a correct prediction.} \\ (1-\varepsilon) w_{i,t} \text{ otherwise.} \end{cases}$

Let m be the total number of mistakes that our algorithm makes and L^* be the number of mistakes that the best expert makes. We analysed this algorithm with $\varepsilon = 0.5$ and found that

$$m \leq 2.41(L^* + \log_2 n).$$

Also, we noted that, in general,

$$m \leq (1-\varepsilon)L^* + O(\frac{1}{\varepsilon} \log n).$$

Follow the Perturbed Leader

Let's consider another algorithm. On each day we will add a random perturbation to the total cost so far of each expert i, and then choose the prediction of the expert of minimum cost. That is:

1. Initialize $c_{i,0} = 0$ for each expert i.

2. On day t, do:

   (a) For each i, select $p_{i,t} \geq 0$ from exp. distribution $d\mu(x) = \varepsilon e^{-\varepsilon x}$.

   (b) Make the same prediction as the expert with minimal $c_{i,t} - p_{i,t}$.

   (c) For each i, $c_{i,t+1} = \begin{cases} c_{i,t} \text{ if expert } i \text{ made a correct prediction.} \\ c_{i,t} + 1 \text{ otherwise.} \end{cases}$

We will show that this algorithm gives

$$E[m] \leq (1-\varepsilon)L^* + O(\frac{1}{\varepsilon} \log n).$$

17.3.1.2 Linear Generalization

Consider a generalization of the prediction problem. We must make a series of decisions $d_1, d_2, \ldots$ each from a set $\mathcal{D} \subset \mathbb{R}^n$. After making the tth decision, we observe the the state $s_t \in \mathcal{S} \subset \mathbb{R}^n$. The total cost of our decisions is $\sum d_t \cdot s_t$.
In the prediction problem, n is the number of experts, choosing expert i corresponds to the decision vector d with a 1 in position i and 0 everywhere else, and the state of each period is the vector of experts' costs (that is, a 1 in position i if expert i predicted wrong, and 0 otherwise).

Our goal is then to have a total cost $\sum d_t \cdot s_t$ close to $\min_{d \in \mathcal{D}} \sum d \cdot s_t$, the cost of the best single decision in hindsight. Let M be a function that computes the best single decision in hindsight, $\arg\min_{d \in \mathcal{D}} \sum d \cdot s_t$. Since costs are additive, we can consider M as a function,

$$M(s) = \arg\min_{d \in \mathcal{D}} d \cdot s.$$

FPL and FPL*

We present two versions of the generalized Follow the Perturbed Leader algorithm.

FPL(ε): On each period t,

1. Choose p_t uniformly at random from the cube $\left[0, \frac{1}{\varepsilon}\right]^n$.
2. Make decision $M(s_1 + s_2 + \ldots + s_{t-1} + p_t)$.

FPL*(ε): On each period t,

1. Choose p_t as follows: Independently for each coordinate, choose $\pm(r/\varepsilon)$ for r from a standard exponential distribution.
2. Make decision $M(s_1 + s_2 + \ldots + s_{t-1} + p_t)$.

Theorem 17.3.1.1 *Let $|x|_1$ denote the L_1 norm. Let*

$$D \geq |d - d'|_1, \text{ for all } d, d' \in \mathcal{D},$$

$$R \geq |d \cdot s|, \text{ for all } d \in \mathcal{D}, s \in \mathcal{S},$$

$$A \geq |s|_1, \text{ for all } s \in \mathcal{S}.$$

Then, (a) FPL with parameter $\varepsilon \leq 1$ gives,

$$E[cost of FPL(\varepsilon)] \leq \text{min-cost}_T + \varepsilon RAT + \frac{D}{\varepsilon},$$

(b) for nonnegative $\mathcal{D}, \mathcal{S} \subset \mathbb{R}^n_+$, FPL^* gives,

$$E[cost of FPL^*(\varepsilon/2A)] \leq (1+\varepsilon)\text{min-cost}_T + \frac{4AD(1+ln\ n)}{\varepsilon}.$$

Our proof of 17.3.1.1 follows that given in [3].

Definition 17.3.1.2 *For convenience, we define: $s_{1:t} = s_1 + s_2 + \ldots + s_t$.*

Lemma 17.3.1.3

$$\sum_{t=1}^{T} M(s_{1:t}) \cdot s_t \leq M(s_{1:t}) \cdot s_{1:t}$$

Proof: By induction on T. T = 1 is trivial, for the induction step from T to T + 1,

$$\begin{aligned}\sum_{t=1}^{T+1} M(s_{1:t}) \cdot s_t &\leq M(s_{1:T}) \cdot s_{1:T} + M(s_{1:T+1}) \cdot s_{T+1} \\ &\leq M(s_{1:T+1}) \cdot s_{1:T} + M(s_{1:T+1}) \cdot s_{T+1} \\ &= M(s_{1:T+1}) \cdot s_{1:T+1}\end{aligned}$$

■

This lemma says that if at each step t we could make the decision that is the optimum single decision in hindsight up to step t then we would have no regret. Next, we show that adding perturbations does not add too much cost.

Lemma 17.3.1.4 *For any state sequence $s_1, s_2, \ldots$, any $T > 0$, and any vectors $p_0 = 0, p_1, p_2, \ldots, p_T \in \mathbb{R}^n$,*

$$\sum_{t=1}^{T} M s_{1:t} + p_t \cdot s_t \leq M(s_{1:T}) \cdot s_{1:T} + D \sum_{t=1}^{T} |p_t - p_{t-1}|_\infty$$

Proof: By 17.3.1.3, we have

$$[rl] \sum_{t=1}^{T} M(s_{1:t} + p_t) \cdot (s_t + p_t - p_{t-1}) \leq M(s_{1:T} + p_T) \cdot (s_{1:T} + p_T) \tag{17.3.1.1}$$

$$\leq M(s_{1:T}) \cdot (s_{1:T} + p_T) \tag{17.3.1.2}$$

$$= M(s_{1:T}) \cdot s_{1:T} + \sum_{t=1}^{T} M(s_{1:T}) \cdot (p_t - p_{t-1}). \tag{17.3.1.3}$$

$$\sum_{t=1}^{T} M(s_{1:t} + p_t) \cdot s_t \leq M(s_{1:T}) \cdot s_{1:T} + \sum_{t=1}^{T} (M(s_{1:T}) - M(s_{1:t} + p_t)) \cdot (p_t - p_{t-1}). \tag{17.3.1.4}$$

Recall that $D \geq |d - d'|_1$ for any d, d'. Also, note that $u \cdot v \leq |u|_1 |v|_\infty$.

■

Note that the distributions over $s_{1:t} + p_t$ and $s_{1:t-1} + p_t$ where p_t is chosen as in $FPL(\varepsilon)$, are similar. That is, they are both cubes. We now prove a bound on the overlap of the two distributions.

Lemma 17.3.1.5 *For any $v \in \mathbb{R}^n$, the cubes $[0, \frac{1}{\varepsilon}]^n$ and $v + [0, \frac{1}{\varepsilon}]^n$ overlap in at least a $(1 - \varepsilon|v|_1)$ fraction.*

Proof: Take a random point $x \in [0, \frac{1}{\varepsilon}]^n$. If $x \notin v + [0, \frac{1}{\varepsilon}]^n$, then for some i, $x_i \notin v_i + [0, \frac{1}{\varepsilon}]$, which happens with probability at most $\varepsilon |v_i|$. By the union bound, we are done. ■

Proof of 17.3.1.1(a): In terms of expected performance, it does not matter if we choose a new p_t each day or if $p_t = p_1$ for all t. So, assume that is the case. Then, by 17.3.1.4

$$\sum_{t=1}^{T} M(s_t + p_1) \cdot s_t \leq M(s_{1:T}) \cdot s_{1:T} + D|p_1|_\infty \leq M(s_{1:T}) \cdot s_{1:T} + \frac{D}{\varepsilon}.$$

Assume that the distributions over $s_{1:t} + p_1$ and $s_{1:t-1} + p_1$ overlap on a fraction f of their volumes. Then,

$$E[M(s_{1:t} - 1) + p_t) \cdot s_t] \leq E[M(s_{1:t} + p_t) \cdot s_t] + (1 - f)R$$

This is because on the fraction that they overlap, the expectation is identical, and on the fraction that they don't, the difference can be at most R.

By 17.3.1.5, $1 - f \leq \varepsilon |s_t|_1 \leq \varepsilon A$. And so,

$$E[costofFPL(\varepsilon)] \leq \text{min-cost}_T + \varepsilon RAT + \frac{D}{\varepsilon},$$

■

A proof of 17.3.1.1(b) can be found in [3].

17.3.2 Application of FPL and FPL*

Prediction problem

For the prediction problem it would seem that $D = 1$ and $A = n$, but, in fact, the worst case for the algorithm is when each period only one expert incurs cost. Thus, we may as well imagine that A = 1, and we get the bound:

$$E[m] \leq (1 - \varepsilon)L^* + O(\frac{1}{\varepsilon} \log n).$$

Online shortest paths

In the online shortest paths problem, we have a directed graph and a fixed pair of nodes (s, t). Each period we pick a path from s to t, and then the length of each edge are revealed. The cost of each period is the length of the chosen path.

Follow the perturbed leading path On each period t,

1. For each edge e, pick $p_{e,t}$ randomly from an exponential distribution.
2. Use the shortest path in the graph with lengths $c_{e,t} + p_{e,t}$ on edge e, where $c_{e,t}$ = sum of the lengths of edge e so far.

If all edge lengths are in $[0, 1]$ we have $D \leq 2n$, since any shortest path will have at most n - 1 edges, $A \leq 2m$ and so by 17.3.1.1(b) we have

$$E[cost] \leq (1 + \varepsilon)(\text{best-time in hindsight}) + \frac{O(mn \log n)}{\varepsilon}$$

17.3.3 Online Convex Programming

In convex programming, we are asked the following: Given a convex set $F \subseteq \mathbb{R}^n$ and a convex cost function $c : F \to \mathbb{R}$, find an $x \in F$ at which $c(x)$ is minimal. In this formulation, F is the **feasible region**, and x is the **optimal solution**.

Convex programming is a generalization of linear programming where the feasible region is allowed to be any convex subset of $\mathbb{R}^n$ (not necessarily polytopal) and the cost function is allowed to be any convex function (not necessarily linear). Convex programming also generalizes semi-definite

programming: semidefinite programming is convex programming with the additional constraint that the Hessian of the cost function is constant over F.

Convex programming is appealing because it is able to encode significantly more problems than linear and semi-definite programming, while being sufficiently similar so that techniques for solving linear and semi-definite programming (gradient descent, interior point, ellipsoid methods, etc) more or less apply to convex programming.

The online version of convex programming occurs over an infinite number of time steps, consisting of a convex feasible set $F \subseteq \mathbb{R}^n$ and a infinite sequence of cost functions $\{c^1, c^2, \dots\}$, where c^t is the cost function at time t. At each time step t, we are asked to select solution $x^t \in F$ without knowing the cost function c^t in advance; c^t is only revealed after we select x^t.

If the cost functions $\{c^i\}$ have any regularity, then we can attempt to learn from the already known cost functions $c^1, c^2, \dots, c^{t-1}$. Hopefully, this allows us to to select solutions which are not far from optimal.

In the following sections, we show that the GREEDY-PROJECTION algorithm, a variation of gradient descent, achieves $O(\sqrt{T})$ competitiveness, where T is the number of time steps.

17.3.3.1 Definitions

The benchmark for an online convex programming algorithm is the *offline convex programming algorithm*. This offline algorithm is given the first T cost functions in advance but can only select a *static solution* that is fixed over all time steps. The offline algorithm selects $x \in F$ as to minimize the cost function $\gamma(x) = \sum_{1 \le i \le T} c^i(x)$.

Definition 17.3.3.1 *Given an online algorithm A and an online convex programming problem $(F, \{c^i\})$, let $x^1, x^2, \dots$ be the solutions selected by an online algorithm A. The **cost** of algorithm A at time T is*

$$C_A(T) = \sum_{1 \le t \le T} c^t(x^t).$$

Definition 17.3.3.2 *Given an online algorithm A and an online convex programming problem $(F, \{c^i\})$, the **competitive difference** of algorithm A up to time T is*

$$R_A(T) = C_A(T) - \gamma(x^*),$$

where $x^* = \min_{x \in F} \gamma(x)$ is the optimal static solution selected by the offline algorithm.

Algorithm 17.3.3.3 (Greedy-Projection) *Let $(F, \{c^i\})$ be an online convex programming problem. Let $\eta_t = t^{-1/2}$ for $t \ge 1$ be a sequence of learning rates. Select $x^1 \in F$ arbitrarily. After receiving cost function c^i in time step i, select x^{i+1} according to*

$$x^{i+1} = \mathbf{Proj}\left(x^i - \eta_i \nabla c^t(x^t)\right),$$

where $\mathbf{Proj}(x) = \arg\min_{x \in F} d(x, y)$ is the projection of x onto the point in F closest to x.

GREEDY-PROJECTION performs gradient descent weighted by learning rates $\eta_t = t^{-1/2}$, and projects out of bounds solutions back into the feasible set if necessary.

Theorem 17.3.3.4 *Let $(F, \{c^i\})$ be an online convex programming problem. Assume*

- *F is nonempty, bounded, and closed;*
- *c^t is differentiable for all t;*
- *$\|\nabla c^t(x)\|$ is bounded above over all t, $x \in F$,*
- *for all t, there exists an algorithm which produces $\nabla c^t(x)$ given $x \in F$; and*
- *for all $x \in \mathbb{R}^n$, there exists an algorithm which produces $\mathbf{Proj}(x)$.*

Let $G = \sup_{t \geq 1, x \in F} \|\nabla c^t\|$ be an upper bound to $\|\nabla c^t\|$, and let D be the diameter of F.
Then the competitive difference of Greedy-Projection *at time T is bounded above:*

$$R_{\mathrm{GP}}(T) \leq (D^2 + G^2)\sqrt{T}.$$

In particular, the average competitive difference approaches 0:

$$R_{\mathrm{GP}}(T)/T \leq 0.$$

17.3.3.2 Proof of competitive difference bound

Proof: Define

$$y^{t+1} = x^t - \eta_t \nabla c^t(x^t)$$

so that $x^{t+1} = \mathbf{Proj}(y^{t+1})$. Let $x^* = \min_{x \in F} \gamma(x)$ be the optimal static solution selected by the offline algorithm. Then

$$\begin{aligned} y^{t+1} - x^* &= (x^t - x^*) - \eta_t \nabla c^t(x^t) \\ (y^{t+1} - x^*)^2 &= (x^t - x^*)^2 - 2\eta_t (x^t - x^*) \cdot \nabla c^t(x^t) + \frac{\eta_t}{2} \|\nabla c^t(x^t)\|^2. \end{aligned}$$

Note that

$$\begin{aligned} (x^{t+1} - x^*)^2 &= (\mathbf{Proj}(y^{t+1}) - x^*)^2 \\ &\leq (y^{t+1} - x^*)^2 \end{aligned}$$

because otherwise we can find $z = x^* + (y^{t+1} - x^*) \cdot (x^{t+1} - x^*)/\|x^{t+1} - x^*\|$ which resides in F and is strictly closer to y^{t+1} than x^{t+1} is, which violates the fact that **Proj** chooses the point in F closest to y^{t+1}. Then, we have

$$(x^{t+1} - x^*)^2 \leq (x^t - x^*)^2 - 2\eta_t (x^t - x^*) \cdot \nabla c^t(x^t) + \|\nabla c^t(x^t)\|^2.$$

Next, because c^t is convex, we use the first order Taylor approximation at x^*

$$c^t(x^*) \geq \nabla c^t(x^t) \cdot (x^* - x^t) + c^t(x^t),$$

obtaining

$$\begin{aligned} c^t(x^t) - c^t(x^*) &\leq c^t(x^t) - \left(\nabla c^t(x^t) \cdot (x^* - x^t) + c^t(x^t)\right) \\ &= (x^t - x^*) \cdot \nabla c^t(x^t) \\ &\leq \frac{1}{2\eta_t}\left((x^t - x^*)^2 - (x^{t+1} - x^*)^2\right) + \frac{\eta_t}{2}\|\nabla c^t(x^t)\|^2. \end{aligned}$$

Note that $c^t(x^t) - c^t(x^*)$ is the contribution to cost of the algorithm from time interval t. We find the competitive difference by summing this differential cost over all time steps:

$$\begin{aligned} R_{\mathrm{GP}}(T) &= C_{\mathrm{GP}}(T) - \gamma(x^*) \\ &= \sum_{1 \leq t \leq T} \left(c^t(x^t) - c^t(x^*)\right) \\ &\leq \sum_{1 \leq t \leq T} \left[\frac{1}{2\eta_t}\left((x^t - x^*)^2 - (x^{t+1} - x^*)^2\right) + \frac{\eta_t}{2}\|\nabla c^t(x^t)\|^2\right]. \end{aligned}$$

Expanding the telescoping sum and using the gradient upper bound $\|\nabla c^t(x^t)\| \leq G$ and diameter D gives

$$\begin{aligned} R_{\mathrm{GP}}(T) &\leq \frac{1}{2\eta_1}(x^1 - x^*)^2 - \frac{1}{2\eta_T}(x^{T+1} - x^*)^2 + \frac{1}{2}\sum_{2 \leq t \leq T}\left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right)(x^t - x^*)^2 + \frac{G^2}{2}\sum_{1 \leq t \leq T}\eta_t \\ &\leq D\left(\frac{1}{2\eta_1} - \frac{1}{2\eta_T} + \sum_{2 \leq t \leq T}\left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right)\right) + \frac{G^2}{2}\sum_{1 \leq t \leq T}\eta_t \\ &\leq D^2\frac{1}{2\eta_T} + \frac{G^2}{2}\sum_{1 \leq t \leq T}\eta_t \end{aligned}$$

Because $\eta_t = t^{-1/2}$,

$$\begin{aligned} \sum_{1 \leq t \leq T}\eta_t = \sum 1 \leq t \leq T\frac{1}{\sqrt{t}} &\leq 1 + \int_{t=1}^{T}\frac{dt}{\sqrt{t}} \\ &\leq 1 + \left[2\sqrt{t}\right]_1^T \\ &\leq 2\sqrt{T}, \end{aligned}$$

and thus

$$\begin{aligned} R_{\mathrm{GP}}(T) &\leq \frac{D^2}{2\eta_T} + \frac{G^2}{2}\sum_{1 \leq t \leq T}\eta_t \\ &\leq \frac{D^2}{2}\sqrt{T} + G^2\sqrt{T} \\ &\leq (D^2 + G^2)\sqrt{T}. \end{aligned}$$

■

References

[1] M. Zinkevich. Online Convex Programming and Generalized Gradient Ascent. In ICML, 2003

[2] E. Hazan, A. Kalai, S. Kale, A. Agarwal. Logarithmic Regret Algorithms for Online Convex Optimization.

[3] A. Kalai, S. Vempala Efficient algorithms for online decision problems. In *Journal of Computer and System Sciences*, 2005.