Lecture 1: October 31, 2011

Lecturer: Yishay Mansour *Scribe: Shai Hertz, Tamar Lavee*

1.1 Regret Minimization: An Online Learning Model

In the regret minimzation model, an agent performs an action a (out of N possible actions), for which he receives some value $loss(a)$. We can also think of a full information model, in which the agent gets for each action a function f that is used to calculate the loss for every possible action: $loss(a) = f(a)$. This process is repeated T times, where T is unknown in advance to the agent.
The target of the agent is to minimize the sum of losses over all T iterations. Specifically, we'll use adverserial model, in which some adversary creates our series of losses, that is unknown in advance. In this model the absolute loss is not bounded, so we would like to compare it to the minimal possible loss. We define the regret as the difference between the absolute loss of our agent and the loss of an agent who always performs a constant 'minimal loss' action a_j:

$$\sum_{t=1}^{T} loss(a_i^t) \leq \left(\min_j \sum_{t=1}^{T} loss(a_j^t) \right) + Regret$$

We can use regret minimization to model online learning: In every step the agent receives an input x_t, chooses a hypothesis h_t and outputs $h_t(x_t)$. Than he sees the loss $l(h_t, x_t)$.
For example, let's think of our way from home to work as a graph in which our home is the source s, our work is the target t and each edge has a weight that corresponds to the traffic jam on this specific route (see figure 1). The action is choosing a specific path from s to t (note that there's an exponential number of such actions), and the loss is the sum of the edges on this path. If d_t is the weight function for day t, we can write our loss as:

$$loss(p_t, d_t) = \sum_{e \in p_t} d_t(e)$$

In this example, the benchmark to which we'll compare our loss is a constant path for all days, which minimizes the sum of losses. Obviously, we can outperform this benchmark by choosing a different (minimal) path for each day.
Typically, algorithms in this model are greedy, either randomized or regularized.

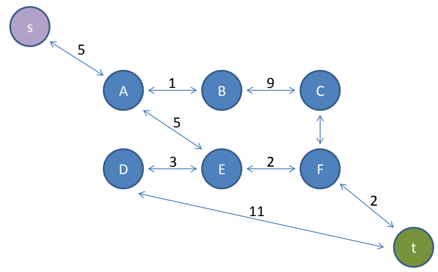

Figure 1.1: A graph representing all possible routes from home to work. The weights of the edges correspond to the traffic between two nodes. In this particular day, the best way from home to work is: s-A-E-F-t.

1.2 Online Convex Optimization

In this model, our actions belong to a convex set K (e.g. the interval $[0,1]$). In each step:

1. The learner chooses $x_t \in K$.
2. The learner gets a convex cost function $f_t : K \to \mathbb{R}$.
3. The loss is $f_t(x_t)$.

Since each f_t is convex, so the sum $\sum_t f_t$ is also convex and therefore its minimum is well defined.

1.2.1 Follow the Leader (FTL)

The naive algorithm for this setting is the "Follow the Leader" (FTL) algorithm. In this algorithm, in each step we choose the best x_t so far:

$$x_t = \operatorname*{argmin}_{x \in K} \sum_{\tau=1}^{t-1} f_\tau(x)$$

However, this approach can lead to unbounded loss. For example, consider K to be the real line segment between -1 and 1, and $f_0 = \frac{1}{2}x$, and let f_i alternate between $-x$ and x. The FTL strategy will keep shifting between -1 and 1, always making the wrong choice. FTL

final loss will be T, while the loss of any fixed x will be 0! Thus, the high fluctuation is a major drawback of FTL.
A possible solution to this problem is discretization. Note that this approach does not use the convexity assumption.
Assume that the derivative of every f_t is bounded, and let $G = \max_t |\frac{\partial}{\partial x} f_t|$. We would like our grid to be fine enough such that for each point, the value of f_t on its neighboring grid points will be 'close enough' (ϵ-wise) to its actual value. For example, for $K = [0,1]^d$, we will need $\left(\frac{G}{\epsilon}\right)^d$ grid points (the number of actions in our model). Now we can use Randomized Weighted Majority algorithm (for example) which guarantees a bound of $\sqrt{T \log N}$. Hence for this discretization the regret will be bounded by $\sqrt{T \cdot d \cdot \log\left(\frac{G}{\epsilon}\right)}$.

1.2.2 Regularized FTL

In this solution, we add regularization to the naive FTL in order to reduce its fluctuation. Our input will be:

- $\eta > 0$ - Learning rate
- R - Regularization function, assumed to be strongly convex and smooth (has a continuous second derivative)
- K - Action space, a convex set

The RTFL Algorithm is shown in algorithm 1. Note that we assume that f_t is linear, and therefore we use it as a vector.

Algorithm 1 RFTL

$x_1 = \underset{x \in K}{\operatorname{argmin}} R(x)$
for $t = 1$ to T **do**
  predict x_t
  observe f_t
  update x_{t+1}:

$$x_{t+1} = \underset{x \in K}{\operatorname{argmin}} \underbrace{\left[\eta \sum_{s=1}^{t} f_s^{\mathrm{T}} x + R(x) \right]}_{\Phi_t(x)}$$

end for

Special Cases

The following examples are known algorithms that can be seen as special cases of the RFTL algorithm.

- Multiplicative Updates:
  $K = \Delta_n = \{x \in [0,1]^n : \sum x_i = 1\}$
  $R(x) = x \cdot \log x = \sum x_i \log x_i$
  We get that the update function is:

$$\Phi_t(x) = \sum_{i=1}^{n} \left[\eta x_i \left(\sum_{s=1}^{t} f_{s,i} \right) + x_i \log x_i \right]$$

  We can find analytically the x that minimizes Φ_t:

$$\frac{\partial}{\partial x_i} \Phi_t(x) = \eta \cdot \sum_{s=1}^{t} f_{s,i} + \log x_i + 1 = 0$$

$$x_{t+1,i} = C \cdot \exp(-\eta \cdot \sum_{s=1}^{t} f_{s,i})$$

  Where C is a constant, that ensures that $x_{t+1} \in \Delta_n$:

$$C = \frac{1}{\sum_j \exp(-\eta \cdot \sum_{s=1}^{t} f_{s,j})}$$

- Gradient Descent:
  $K = \{x \in (0,1)^n : \|x\| = 1\}$ (Unit ball)
  $R(x) = \frac{1}{2}\|x\|_2^2$
  As before, let's look at the derivative of $\Phi_t(x)$:

$$\frac{\partial}{\partial x_i} \Phi_t(x) = \eta \cdot \sum_{s=1}^{t} f_{s,i} + x_i = 0$$

  Therefore:

$$x_{t+1,i} = -C\eta \sum_{s=1}^{t} f_{s,i} = -C \left(\eta \sum_{s=1}^{t-1} f_{s,i} - \eta f_{t,i} \right) = -C(x_{t,i} - \eta f_{t,i})$$

  Where C is a constant that normalizes x_{t+1}. Thus, the updating rule is:

$$x_{t+1,i} = \frac{x_t - \eta f_t}{\|x_t - \eta f_t\|_2}$$

Regret Bound for the RFTL Algorithm

Definition For a positive semi-definite matrix A, the A-norm of x will be defined as $\|x\|_A = \sqrt{x^{\mathrm{T}} A x}$, and the dual norm of this matrix norm will be denoted by $\|x\|_A^* = \|x\|_{A^{-1}}$.

Let's denote the diameter of K as measured by the regularization function R by D:

$$D = \max_{u \in K} (R(u) - R(x_1)).$$

We'll set λ to be:

$$\lambda = \max_{t, x \in K} f_t^{\mathrm{T}} [\nabla^2 R(x)]^{-1} f_t.$$

So, λ is a measure of the magnitude of the cost function f, using the dual $\nabla^2 R(x)$-norm.

Theorem 1.1 *The RTFL algorithm guarantees that for every u:*

$$\mathit{Regret} \leq \sum_{t=1}^{T} f_t^{\mathrm{T}} \cdot (x_t - u) \leq 2\sqrt{2\lambda D T}$$

For example, in multiplicative updates: $R(x) = x \cdot \log x$, $\nabla R(x) = \log x + 1$, $\nabla^2 R(x)$ is a matrix whose diagonal is $\frac{1}{x}$.
Therefore:

$$\lambda = f_t^{\mathrm{T}} [\nabla^2 R(x)]^{-1} f_t = \sum_{i=1}^{N} f_i^2 x_i \leq \sum_{i=1}^{N} x_i = 1$$

(We assume that every $|f_t| \leq 1$, and since K is a simplex)
We also know that $D = R(u) - R(x_1) \leq \log n$.
Hence from the theorem we get that the regret bound for multiplicative updates is $2\sqrt{2T \log n}$.

To prove the above theorem, we'll first use a lemma that will relate the regret to the 'stability' in prediction:

Lemma 1.2 *For every $u \in K$ the RFTL algorithm guarantees the following:*

$$\sum_{t=1}^{T} [f_t(x_t) - f_t(u)] \leq \sum_{t=1}^{T} [f_t(x_t) - f_t(x_{t+1})] + \frac{1}{\eta} [R(u) - R(x_1)]$$

Proof: For simplicity, we'll use $f_0(x) = \frac{1}{\eta} R(x)$, and then we can write the update rule as:

$$x_{t+1} = \operatorname*{argmin}_{x \in K} \sum_{x=0}^{t} f_s(x)$$

We'll prove by induction on T that:

$$\sum_{t=0}^{T}[f_t(x_t) - f_t(u)] \leq \sum_{t=0}^{T}[f_t(x_t) - f_t(x_{t+1})]$$

Induction base: By definition we have $x_1 = \underset{x \in K}{\operatorname{argmin}} R(x)$. Therefore we are certain that for all u:

$$f_0(x_1) \leq f_0(u)$$

Thus:

$$f_0(x_0) - f_0(u) \leq f_0(x_0) - f_0(x_1)$$

Indcution step: We'll assume that for T we have:

$$\sum_{t=0}^{T}[f_t(x_t) - f_t(u)] \leq \sum_{t=0}^{T}[f_t(x_t) - f_t(x_{t+1})]$$

Now we'll look at $T+1$, i.e., the choice of X_{T+2}. Since $x_{T+2} = \underset{x \in K}{\operatorname{argmin}} \sum_{t=0}^{T+1} f_t(x)$ we have that for every u:

$$\begin{aligned}
\sum_{t=0}^{T+1} f_t(x_t) - \sum_{t=0}^{T+1} f_t(u) &\leq \sum_{t=0}^{T+1} f_t(x_t) - \sum_{t=0}^{T+1} f_t(x_{T+2}) \\
&= \sum_{t=0}^{T}[f_t(x_t) - f_t(x_{T+2})] + f_{T+1}(x_{T+1}) - f_{T+1}(x_{T+2}) \\
&\leq \sum_{t=0}^{T}[f_t(x_t) - f_t(x_{t+1})] + f_{T+1}(x_{T+1}) - f_{T+1}(x_{T+2}) \\
&= \sum_{t=0}^{T+1}[f_t(x_t) - f_t(x_{t+1})]
\end{aligned}$$

where the inequality in the third line follows form the induction assumption.
Now we can conclude:

$$\begin{aligned}
\sum_{t=0}^{T}[f_t(x_t) - f_t(u)] &= \sum_{t=1}^{T}[f_t(x_t) - f_t(u)] + f_0(x_0) - f_0(u) \\
&\leq \sum_{t=0}^{T}[f_t(x_t) - f_t(x_{t+1})] \\
&= \sum_{t=1}^{T}[f_t(x_t) - f_t(x_{t+1})] + f_0(x_0) - f_0(x_1)
\end{aligned}$$

Therefore:

$$\begin{aligned}\sum_{t=1}^{T}[f_t(x_t)-f_t(u)] &\le \sum_{t=1}^{T}[f_t(x_t)-f_t(x_{t+1})]+[-f_0(x_0)+f_0(u)+f_0(x_0)-f_0(x_1)]\\ &= \sum_{t=1}^{T}[f_t(x_t)-f_t(x_{t+1})]+\frac{1}{\eta}[R(u)-R(x_1)]\end{aligned}$$

□

Now we can return to our theorem.

Proof: From the Taylor expansion of Φ_t and since K and R are convex, there exists a point $z_t \in [x_{t+1}, x_t]$ for which:

$$\begin{aligned}\Phi_t(x_t) &= \Phi_t(x_{t+1}) + (x_t - x_{t+1})^{\mathrm{T}}\nabla\Phi_t(x_{t+1}) + \frac{1}{2}\|x_t - x_{t+1}\|_{z_t}^2\\ &= \Phi_t(x_{t+1}) + \frac{1}{2}\|x_t - x_{t+1}\|_{z_t}^2\end{aligned}$$

(Since x_{t+1} minimizes Φ_t, we have $\nabla\Phi_t(x_{t+1}) = 0$)

Therefore:

$$\begin{aligned}\|x_t - x_{t+1}\|_{z_t}^2 &= 2[\Phi_t(x_t) - \Phi_t(x_{t+1})]\\ &= 2[\Phi_{t-1}(x_t) - \Phi_{t-1}(x_{t+1})] + 2\eta f_t^{\mathrm{T}}(x_t - x_{t+1})\\ &\le 2\eta f_t^{\mathrm{T}}(x_t - x_{t+1})\end{aligned}$$

since x_t minimizes $\Phi_{t-1}(\cdot)$.

By the generalized Cauchy-Schwartz inequality,

$$f_t^{\mathrm{T}}(x_t - x_{t+1}) \le \|f_t\|_{z_t}^* \cdot \|x_t - x_{t+1}\|_{z_t}$$

Combining the two together:

$$f_t^{\mathrm{T}}(x_t - x_{t+1}) \le \|f_t\|_{z_t}^* \cdot \sqrt{2\eta f_t^{\mathrm{T}}(x_t - x_{t+1})}$$

We'll square both sides and use our defined λ to get:

$$f_t^{\mathrm{T}}(x_t - x_{t+1}) \le 2\eta(\|f_t\|_{z_t}^*)^2 \le 2\eta\lambda$$

Now we shall sum over t and use the lemma to get:

$$\sum_{t=1}^{T} f_t^{\mathrm{T}}(x_t - u) \le T\cdot(2\eta\lambda) + \frac{1}{\eta}[R(u) - R(x_1)] \le 2\eta\lambda T + \frac{1}{\eta}D$$

We will choose $\eta = \sqrt{\frac{D}{2\lambda T}}$ to achieve the desired regret bound:

$$\sum_{t=1}^{T} f_t^{\mathrm{T}}(x_t - u) \le 2\sqrt{2\lambda DT}$$

□

1.2.3 Primal-Dual Algorithm

We now turn to look at another algorithm which uses Bregman Divergence. We can think of Bregman divergence as a generalized metric, which is not necessarily symmetric and for which the triangle inequality doesn't necessarily hold.

Definition Bregman divergence B^R is defined as:

$$B^R(x\|y) = R(x) - R(y) - (x-y)^{\mathrm{T}} \cdot \nabla R(y)$$

Examples:

1. $R(x) = \frac{1}{2}\|x\|_2^2 \Rightarrow \nabla R(y) = y$
   The Bregman divergence for this function:

$$\begin{aligned} B^R(x\|y) &= \frac{1}{2}\|x\|_2^2 - \frac{1}{2}\|y\|_2^2 - (x-y)^{\mathrm{T}} \cdot y \\ &= \frac{1}{2}\|x\|_2^2 + \frac{1}{2}\|y\|_2^2 - xy \\ &= \frac{1}{2}\|x-y\|_2^2 \end{aligned}$$

   We got that the Bregman divergence of this regularizer is proportional to the squared Euclidean distance.

2. $R(p) = \sum p_i \cdot \ln p_i - \sum p_i \Rightarrow \nabla R(q) = \ln q$
   The Bregman divergence for this function:

$$\begin{aligned} B^R(p\|q) &= \sum p_i \cdot \ln p_i - \sum p_i - \sum q_i \cdot \ln q_i + \sum q_i - \sum (p_i - q_i) \ln q_i \\ &= \sum p_i \ln \frac{p_i}{q_i} + \sum q_i - \sum p_i \end{aligned}$$

   Note that the first term of this expression is the Kullback-Leibler divergence between p and q.

Algorithm 2 Primal-Dual

Let K be a convex set, $\eta > 0$, $R(x)$ a convex regularization function
for $t = 1$ to T **do**
  predict x_t
  if $t = 1$ **then**
    choose y_1 such that $\nabla R(y_1) = 0$
  else
    choose y_t such that $\nabla R(y_t) = \nabla R(x_{t-1}) - \eta \nabla f_{t-1}(x_{t-1})$
  end if
  Update using a projection according to B^R:

$$x_t = \underset{x \in K}{\operatorname{argmin}} B^R(x \| y_t)$$

end for

The Primal-Dual algorithm, like RFTL, computes the next prediction using a simple update rule. We will use Bregman divergence to define a dual space in which the algorithm will search for the best choice of x_{t+1}. The specific algorithm is shown in algorithm 2.

For the special case of linear cost functions, the Primal-Dual and the RFTL algorithms are identical - they will produce an identical set of points x_t.

Regret Bound for the Primal-Dual Algorithm

Theorem 1.3 *Suppose that R is such that $B^R(x\|y) \geq \|x - y\|^2$ for some norm.*
Denote $\|f_t(x_t)\|^* \leq G_*$ and $B^R(x\|x_1) \leq D^2$.
Then, applying the Primal-Dual algorithm with $\eta = \frac{D}{2G_*\sqrt{T}}$ will guarantee:

$$Regret \leq DG_*\sqrt{T}$$

Proof: Since the functions f_t are convex, for any $x^* \in K$,

$$f_t(x_t) - f_t(x^*) \leq \nabla f_t(x_t)^{\mathrm{T}} \cdot (x_t - x^*)$$

A useful property of the Bregman Divergence is:

$$\begin{aligned}
&B^R(x\|y) - B^R(x\|z) + B^R(y\|z) \\
&\quad = R(x) - R(y) - (x-y)^{\mathrm{T}}\nabla R(y) - R(x) + R(z) + (x-z)^{\mathrm{T}}\nabla R(z) + R(y) - R(z) - (y-z)^{\mathrm{T}}\nabla R(z) \\
&\quad = -(x-y)^{\mathrm{T}}\nabla R(y) + (x-z)^{\mathrm{T}}\nabla R(z) - (y-z)^{\mathrm{T}}\nabla R(z) \\
&\quad = (x-y)^{\mathrm{T}}(\nabla R(z) - \nabla R(y))
\end{aligned}$$

From the definition of the Primal-Dual algorithm:

$$\nabla f_t(x_t) = \frac{1}{\eta}(\nabla R(x_t) - \nabla R(y_{t+1}))$$

Combining these properties:

$$\begin{aligned}
[f_t(x_t) - f_t(x^*)] &\leq \nabla f_t(x_t)^{\mathrm{T}}(x_t - x^*) \\
&= \frac{1}{\eta}[\nabla R(x_t) - \nabla R(y_{t+1})](x_t - x^*) \\
&= \frac{1}{\eta}(\nabla R(y_{t+1}) - \nabla R(y_t))^{\mathrm{T}}(x^* - x_t) \\
&= \frac{1}{\eta}[B^R(x^* \| x_t) - B^R(x^* \| y_{t+1}) + B^R(x_t \| y_{t+1}) \\
&\leq \frac{1}{\eta}[B^R(x^* \| x_t) - B^R(x^* \| x_{t+1}) + B^R(x_t \| y_{t+1})
\end{aligned}$$

The last inequality follows from the fact that x_{t+1} is the projection w.r.t the Bregman divergence of y_{t+1} and $x^* \in K$ which is convex.
Summing over all iterations we get:

$$\begin{aligned}
2Regret &\leq \frac{1}{\eta}[B^R(x^* \| x_1) - B^R(x^* \| x_T)] + \frac{1}{\eta}\sum_{t=1}^{T} B^R(x_t \| y_{t+1}) \\
&\leq \frac{1}{\eta}D^2 + \frac{1}{\eta}\sum_{t=1}^{T} B^R(x_t \| y_{t+1})
\end{aligned}$$

We proceed to bound $B^R(x_t \| y_{t+1})$. By definition of Bregman divergence, we get:

$$\begin{aligned}
B^R(x_t \| y_{t+1}) + B^R(y_{t+1} \| x_t) &= (\nabla R(x_t) - \nabla R(y_{t+1})^{\mathrm{T}}(x_t - y_{t+1}) \\
&= \eta \nabla f_t(x_t)^{\mathrm{T}}(x_t - y_{t+1}) \\
&\leq \eta(\|\nabla f_t(x_t)\|)^* \cdot \|x_t - y_{t+1}\| \\
&\leq (\eta \|\nabla f_t(x_t)\|^*)^2 + \|x_t - y_{t+1}\|^2 \\
&\leq \eta^2 G_*^2 + \|x_t - y_{t+1}\|^2
\end{aligned}$$

Where the inequality in the third row follows from the generalized Cauchy-Schwartz inequality, and the fourth row follows from the simple fact: $2xy \leq x^2 + y^2$.
Thus, by our assumption that $B^R(x \| y) \geq \|x - y\|^2$ we get:

$$\begin{aligned}
B^R(x_t \| y_{t+1}) &\leq \eta^2 G_*^2 + \|x_t - y_{t+1}\|^2 - B^R(y_{t+1} \| x_t) \\
&\leq \eta^2 G_*^2
\end{aligned}$$

Plugging that into the regret expression, and using $\eta = \frac{D}{2G_*\sqrt{T}}$, we conclude:

$$Regret \leq \frac{1}{2}\left[\frac{1}{\eta}D^2 + \eta T G_*^2\right] \leq D G_* \sqrt{T}$$

$\square$