Notes on Follow-the-Regularized-Leader

The Multiplicative Weights algorithm and its analysis are very powerful but seem to come out of nowhere. How would you design such an algorithm, and how would you come up with such an analysis? In these notes we discuss a broader framework in which we get Multiplicative Weights and its analysis as a special case, and we are also able to derive the *Gradient Descent* algorithm and its analysis as another special case.

Before introducing this framework, let us make our problem more general. Now at each time step t, our algorithm is required to output an admissible solution $x^{(t)} \in K$, where $K \subseteq \mathbb{R}^n$ is a convex[1] set of admissible solutions known to the algorithm, and then the algorithm is told a convex cost function[2] $f_t : K \to \mathbb{R}$, and the algorithm incurs the loss $f_t(x^{(t)})$. After T steps, the regret of the algorithm is defined as

$$\sum_{t=1}^{T} f_t(x^{(t)}) - \min_{x \in K} \sum_{t=1}^{T} f_t(x)$$

The setup in which we analyzed the Multiplicative Weights algorithm corresponds to the case in which Δ is the set of probability distributions and the loss functions $f_t(\cdot)$ are of the form $f_t(x) = \sum_{i=1}^{n} \ell_i^{(t)} x_i$. (Note that the set of probability distributions is convex and that all linear functions are convex.)

In designing an algorithm for this problem, perhaps the first idea that would come to mind is that, at every step, we pick the solution that *would have been optimal for the previous steps*, that is, we define:

$$x^{(t)} = \arg\min_{x \in K} \sum_{k=1}^{t-1} f_k(x) \tag{1}$$

This is algorithm, called *Follow The Leader* (abbreviated FTL), is actually a bad idea. If $K = \Delta$ is the set of probability distributions and the f_t are linear, for example, FTL will "overfit" on a single coordinate, and then the adversary can easily create a sequence of loss functions $f_t(\cdot)$ that will be terrible for the algorithm.

The next idea is to look for a solution $x^{(t)}$ that is optimal for an expression similar to (1), but that contains an additional term, called a "regularizer," that penalizes "overfitting" solutions, by charging a higher cost to "concentrated" solutions than to "spread-out" solutions:

[1] Recall that a set K is convex if for every two points $x, y \in K$ the segment connecting x and y is entirely contained in K. This is a useful property when discussing optimization. Why the letter "K"? It is a common letter to use for convex bodies. I think it comes from German: in German "convex body" is "konvex körper."

[2] A function is convex if the set of points (x, y) such that $f(x) \leq y$, that is, the set of points "above the graph" of the function is convex. We will talk about two additional characterizations later on. This is also a useful property for optimization. In particular, a local minimum of a convex function on a a convex set is also a global minimum

$$x^{(t)} = \arg\min_{x \in K} \sum_{k=1}^{t-1} f_k(x) + R(x) \tag{2}$$

The above algorithm is called *Follow the Regularized Leader.* It is actually a family of algorithms, one for every choice of the function $R(\cdot)$.

We have the following remarkable analysis, that holds with no assumptions whatsoever on K, on the family of allowed loss functions $f_t(\cdot)$, or on the regularizer $R(\cdot)$.

THEOREM 1
For every $x \in K$, and $x^{(t)}$ chosen according to the FTRL algorithm with cost functions f_t and regularizer $R(\cdot)$, we have

$$\sum_{t=1}^{T} f_t(x^{(t)}) - \sum_{t=1}^{T} f_t(x) \leq \left(\sum_{t=1}^{T} f_t(x^{(t)}) - f_t(x^{(t+1)}) \right) + R(x) - R(x^{(1)}) \tag{3}$$

and so the regret $\mathrm{Regr}_T := \sum_{t=1}^{T} f_t(x^{(t)}) - \min_{x \in K} \sum_{t=1}^{T} f_t(x)$ *satisfies*

$$\mathrm{Regr}_T \leq \left(\sum_{t=1}^{T} f_t(x^{(t)}) - f_t(x^{(t+1)}) \right) + \max_{x \in K} R(x) - R(x^{(1)}) \tag{4}$$

We write the regret at time T as Regr_T because we want to keep the symbol R for the regularizer.

1 Analysis

The proof of Theorem 1 is simple and it comes down to two ideas: reduce to the case $R(\cdot) \equiv 0$ and use induction on T.

So suppose that there is no regularizer, or that, equivalently, the regularizer is identically equal to zero. Then the update rule is simply (1), and we are running the *Follow the Leader* algorithm. (In this case, the analysis will work for an arbitrary starting point, because any starting point is a minimizer for a regularizer that is a constant function.) In this setup, proving (3) reduces to proving that, for every x,

$$\sum_{t=1}^{T} f_t(x) \geq \sum_{t=1}^{T} f_t(x^{(t+1)})$$

which we will do by induction on T. For $T = 1$ it's just the definition of $x^{(2)}$, and, for the inductive step, see that

$$\sum_{t=1}^{T+1} f_t(x) \geq \sum_{t=1}^{T+1} f_t(x^{(T+2)})) \geq \sum_{t=1}^{T+1} f_t(x^{(t+1)})$$

where the first step applies the definition of $x^{(T+2)}$ and the second step applies the inductive hypothesis to get

$$\sum_{t=1}^{T} f_t(x^{(T+2)}) \geq \sum_{t=1}^{T} f_t(x^{(t+1)})$$

Now we just have to interpret the general case as an instance of Follow the Leader in which there is a zeroth time step in which the cost function is $R(\cdot)$. In more detail, suppose that $x^{(1)}, \ldots, x^{(T)}$ are the solutions produced by the FTRL algorithm with regularizer $R(\cdot)$ when run against the cost functions $f_1, \ldots, f_T$, and let $x \in K$ be an arbitrary other point. Consider the behavior of the FTL algorithm that uses $x^{(0)} = x^{(1)}$ as a starting point in a game in which there is a zeroth time step in which the cost function is $g_0 = R$ and then T additional time steps with functions are $g_t = f_t$ for $t = 1, \ldots, T$ Notice that, at time $t = 1, \ldots, T$, the FTL algorithm will produce the solution $x^{(t)}$. The regret bound for FTL implies that

$$\sum_{t=0}^{T} g_t(x^{(t)}) - \sum_{t=0}^{T} g_t(x) \leq \sum_{t=0}^{T} g_t(x^{(t)}) - g_t(x^{(t+1)})$$

that is

$$\left(R(x^{(1)}) + \sum_{t=1}^{T} f_t(x^{(t)})\right) - \left(R(x) + \sum_{t=1}^{T} f_t(x)\right) \leq R(x^{(1)}) - R(x^{(1)}) + \sum_{t=1}^{T} f_(x^{(t)}) - f_t(x^{(t+1)})$$

which is equivalent to the Theorem.

2 FTRL with Negative-Entropy Regularizer

Now we go back to the setting of x being a probability distribution and the loss functions f_t being linear $f_t(x) = \sum_i \ell_i^{(t)} x_i = \langle \ell^{(t)}, x \rangle$, where the losses $\ell_i^{(t)}$ are in $[0, 1]$. By reasoning from first principles, we will "rediscover" the Multiplicative Weight algorithms.

We said that we want to choose a regularizer that tends to "spread out" the distribution x, that is, make it "more random" and less concentrated on a few possible events. A standard measure of "how much randomness" there is in a distribution is the *entropy* of a distribution. If x is a distribution, its entropy is defined as

$$H(x) = \sum_i x_i \ln \frac{1}{x_i}$$

where we follow the convention that $0 \log 0 = 0$. (The entropy function is usually defined using a logarithm in base 2, not base e, but this only changes it by a fixed multiplicative constant. Using natural logarithms will make some of the calculations that we will do later cleaner.) Entropy is maximized by the uniform distribution $\left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right)$, on which it equals $\ln n$, and it is minimized by distributions entirely concentrated on one outcome, such as $(1, 0, \ldots, 0)$, on which it equals zero. In particular, $H(x) \geq 0$ for every distribution x. If we look at the function $-H(x)$, then it is convex (a fact that we will not prove) and, as the above examples suggest, minimizing $-H(x)$, which is the same as maximizing $H(x)$, makes x as random and "spread out" as possible.

Let's choose the negative entropy as a regularizer, with a multiplicative factor of c that we will optimize later:

$$R(x) := -cH(x) = c \sum_i x_i \ln x_i$$

Having described the regularizer, we have completely described our FTRL algorithm (up to the choice of c). Now let us try to understand what it does. How does the algorithm get started?

$$x^{(1)} = \arg\min_{x \in \Delta} \quad c \sum_i x_i \log x_i$$

that is, we are looking for the distribution of maximum entropy, and that is the uniform distribution, so we have

$$x^{(1)} = \left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n} \right)$$

The value of the regularizer at $x^{(1)}$ is

$$R(x^{(1)}) = -c \ln n$$

and we have $R(x) \leq 0$ for every probability distribution x, so, by applying Theorem 1, our regret will be

$$\begin{aligned} \mathrm{Regr}_T &:= \sum_{t=1}^{T} f_t(x^{(t)}) - \min_{x \in \Delta} \sum_{t=1}^{T} f_t(x) && (5) \\ &\leq \sum_{t=1}^{T} f_t(x^{(t)}) - f_t(x^{(t+1)}) \quad + c \ln n && (6) \end{aligned}$$

Now we need to understand what the update rule does.

Recall that, if we write $f_t(x) = \langle \ell^{(t)}, x \rangle$, then the update rule for our loss functions and regularizer is

$$x^{(t+1)} = \arg\min_{x \in \Delta} \sum_{k=1}^{t} \langle \ell^{(k)}, x \rangle + c \sum_i x_i \log x_i \tag{7}$$

To find the above minimum, we will use the method of Lagrange multipliers. It is not essential to understand this calculation in order to follow the rest of this section, so the reader may jump ahead to the solution given below in (8).

Applying the method of Lagrange multipliers, we want to look for points with positive coordinates at which the gradient of

$$f_\lambda(x) := \sum_{k=1}^{t} \langle \ell_k, x \rangle + c \sum_i x_i \ln x_i + \lambda \cdot \left(\sum_i x_i - 1 \right)$$

is zero for some possible values of λ. We see that

$$\nabla f_\lambda(x) = \sum_{k=1}^{t} \ell_k + c \cdot (\ln x_1, \ldots, \ln x_n) + (c + \lambda) \cdot (1, \ldots, 1)$$

which means that the stable points are of the form

$$x_i = e^{-\frac{1}{c}\sum_{k=1}^{t}\ell_i^{(k)}} \cdot e^{-1-\frac{\lambda}{c}}$$

and the unique feasible such point is the one that makes the x_i add up to one, so we have

$$x_i^{(t+1)} = \frac{e^{-\frac{1}{c}\sum_{k=1}^{t}\ell_i^{(k)}}}{\sum_j e^{-\frac{1}{c}\sum_{k=1}^{t}\ell_j^{(k)}}} \tag{8}$$

which is exactly the multiplicative weight update if we define ϵ so that $(1-\epsilon) = e^{-1/c}$.

To complete the analysis in this framework, we have to upper bound the regret bound

$$\sum_t f_t(x^{(t)}) - f_t(x^{(t+1)}) = \sum_t \langle \ell^{(t)}, x^{(t)} - x^{(t+1)} \rangle$$

Define, as in the notes on the Multiplicative Weight Algorithm, $w_i^{(t)} = e^{-\frac{1}{c}\sum_{k=1}^{t-1}\ell_i^{(k)}}$ and $W_t = \sum_{i=1}^{n} w_i^{(t)}$, so that we have $x_i^{(t)} = w_i^{(t)}/W_t$. Since $\ell_i^{(t)} \in [0,1]$, we see that W_t decreases with t. Thus, we have

$$x_i^{(t)} - x_i^{(t+1)} = x_i^{(t)} \cdot \left(1 - \frac{x_i^{(t+1)}}{x_i^{(t)}}\right) = x_i^{(t)} \cdot \left(1 - \frac{w_i^{(t+1)}}{W_{t+1}} \cdot \frac{W_t}{w_i^{(t)}}\right) = x_i^{(t)} \cdot \left(1 - e^{-\frac{1}{c}\ell_i^{(t)}} \cdot \frac{W_t}{W_{t+1}}\right) \leq \frac{1}{c} \cdot x_i^{(t)}$$

because we have $\frac{W_t}{W_{t+1}} \geq 1$ and $e^{-\ell_i^{(t)}/c} \geq 1 - \ell_i^{(t)}/c \geq 1 - 1/c$.

Overall, the bound on the regret is

$$\text{Regr}_T \leq \sum_{t=1}^{T} \langle \ell^{(t)}, x^{(t)} - x^{(t+1)} \rangle + c \ln n \leq \sum_{t=1}^{T} \sum_i \frac{1}{c} x_i^{(t)} + c \ln n = \frac{T}{c} + c \ln n$$

which is the same regret we computed before, if we replace c with $1/\epsilon$. Thus we have rediscovered the same algorithm with the same regret bound.

3 FTRL With Length-Square Regularization

Another interesting example is when $K = \mathbb{R}^n$ (meaning that have no constraints on the solutions provided by the algorithm), the loss functions $f_t(x) = \langle \ell^{(t)}, x \rangle$ are linear, and the regularizer is

$$R(x) = c||x||^2 = c \cdot \sum_i x_i^2$$

proportional to the square of the length of the solution vector x, where c is a parameter that we can choose and that we will select later.

Note that if x is a probability distribution over a sample space of n possibilities, then $||x||^2$ is always between $1/n$ and 1, with $1/n$ achieved by the uniform distribution and 1 achieved by distributions that assign probability 1 to one element of the sample space and 0 to all others. Thus this regularizer, like negative entropy, prefers spread-out distributions to concentrated ones.

Let us understand what the algorithm will do this time.

$$x^{(1)} = \arg\min_{x \in \mathbb{R}^n} R(x) = (0, 0, \dots 0)$$

and the update rule is

$$x^{(t+1)} = \arg\min_{x \in \mathbb{R}^n} f_1(x) + \cdots + f_t(x) + c||x||^2$$

The minimum must be achieved at a point in which the gradient of the right-hand side is zero, so let us compute the gradient of the right-hand side: it is

$$\sum_{k=1}^{t} \ell^{(k)} + 2cx$$

and the only way to make it zero (which is then the unique optimum) is to set

$$x^{(t+1)} = -\frac{1}{2c} \sum_{k=1}^{t} \ell^{(k)}$$

Another way to think about the behavior of the algorithm is that it sets

$$x^{(0)} = 0$$

$$x^{(t+1)} = x^{(t)} - \frac{1}{2c} \ell^{(t)}$$

We cannot prove an absolute regret bound, but we have the following bound in term of a particular fixed solution x:

$$\forall x \in \mathbb{R}^n \quad \sum_{t=1}^{T} f_t(x^{(t)}) - f_t(x) \leq \sum_{t=1}^{t} \langle \ell^{(t)}, x^{(t)} - x^{(t+1)} \rangle + c||x||^2 = \sum_{t=1}^{T} \frac{1}{2c} ||\ell^{(t)}||^2 + c||x||^2$$

What good is any of this? Suppose that we have a convex function $g : \mathbb{R}^n \to \mathbb{R}$ that we want to minimize, and suppose that we have scaled the function so that we always have $||\nabla g|| \leq 1$. Recall that a function $g : \mathbb{R}^n \to \mathbb{R}$ is convex if for every two possible inputs x and y, the value of the function in the interval $[x, y]$ is smaller that the value of the linear interpolation of g between x and y, that is, for every $\lambda \in [0, 1]$:

$$g((1 - \lambda)x + \lambda y) \leq (1 - \lambda)g(x) + \lambda g(y)$$

If g is differentiable, then an equivalent characterization of convexity is that g is always bigger than its first-order approximation, that is, for every two points x and y we have

$$g(x) \geq g(y) + \langle \nabla g(y), x - y \rangle$$

In order to minimize g, we are going to run the following game against the algorithm that we just described: when the algorithm proposes the solution $x^{(t)}$, we choose the cost function given by the gradient of g at $x^{(t)}$ that is we define the cost function

$$f_t(x) := \langle \nabla g(x^{(t)}), x \rangle$$

After running the algorithm for T steps, and using the assumption that the norm of the gradient is at most 1, our regret bound is

$$\forall x \in \mathbb{R}^n \quad \sum_{t=1}^{T} f_t(x^{(t)}) - f_t(x) \leq \frac{T}{2c} + c||x||^2$$

where

$$\sum_{t=1}^{T} f_t(x^{(t)}) - f_t(x) = \sum_{t=1}^{T} \langle \nabla g(x^{(t)}), x^{(t)} - x \rangle$$

Since g is convex, it is always bigger than its first-order approximation so for every t we have

$$\forall x \in \mathbb{R}^n \quad g(x) \geq g(x^{(t)}) + \langle \nabla g(x^{(t)}), x - x^{(t)} \rangle$$

Putting things together we have

$$\forall x \in \mathbb{R}^n \quad \sum_{t=1}^{T} g(x^{(t)}) - g(x) \leq \frac{T}{2c} + c||x||^2$$

Dividing by T and letting x^* be a point where g is minimized we have

$$\frac{1}{T} \sum_{i=1}^{T} g(x^{(t)}) \leq g(x^*) + \frac{1}{2c} + \frac{c}{T}||x^*||^2$$

Using the convexity of $g(\cdot)$ we have

$$g\left(\frac{1}{T} \sum_{i=1}^{T} x^{(t)}\right) \leq \frac{1}{T} \sum_{i=1}^{T} g(x^{(t)}) \leq \min_{x \in \mathbb{R}^n} g(x) + \frac{1}{2c} + \frac{c}{T}||x^*||^2$$

If we know an upper bound $||x^*|| \leq B$ we can choose $c = \frac{1}{B} \cdot \sqrt{\frac{T}{2}}$ and we have

$$g\left(\frac{1}{T} \sum_{i=1}^{T} x^{(t)}\right) \leq \min_{x \in \mathbb{R}^n} g(x) + \frac{\sqrt{2}B}{\sqrt{T}}$$

That is, the average of the points found by the algorithm has a value of g that is close to the optimum, and we can get arbitrarily close by running for more steps.

This is just the *gradient descent* algorithm! The algorithm starts with an initial solution

$$x^{(0)} = 0$$

and then at each step it computes a new solution

$$x^{(t+1)} = x^{(t)} - \frac{1}{2c} \nabla g(x^{(t)})$$

by moving by a "step" of length $1/2c$ in the direction opposite to the gradient at the previous solution (because the gradient is the direction in which $g(\cdot)$ grows the fastest and the opposite direction is the one in which $g(\cdot)$ decreases the most). After several steps, we have a guarantee that the average of the solutions found so far is a good approximation of the optimum.

4 Summary

We have demonstrated that both Multiplicative Weights and Gradient Descent can be seen as special cases of the Follow-The-Regularized-Leader framework, by using different regularizers.