IIIS 2018 Spring: ATCS - Learning and Optimization - theory and practice
Lecture date: Mar 12, 2018
Instructor: Jian Li **Scribe: Yu Shi**

1 Introduction

In this lecture, we first introduce the Universal Portfolios by Thomas Cover, which is an online algorithm providing good allocation of wealth in a market. Then, we show that an online learning algorithm can be used in supervised learning with the generalization error bounded by the regret. This technique is known as Online to Batch Conversion. Then, we give a brief introduction to Fenchel conjugate. Finally, we introduce some concepts which will be used in Online Mirror Descent for next lecture.

2 Thomas Cover's Universal Portfolios

We want a weighted combination of stocks with promising return. A combination of stocks is known as *portfolio*. More generally, a portfolio is an allocation of wealth among various financial assets, such as stocks and bonds. The Universal Portfolio algorithm learns a good portfolio every day. Such portfolios approximate the return of optimal fixed portfolio in the long term. Formally, with m stocks, let $\mathbf{x}_i = (x_{i,1}, ..., x_{i,m})^T \in \mathbb{R}^m$ be the vector of stock *return* in day i. Note that on day i when we decide our portfolio, we only know the price of previous days, instead of $\mathbf{x}_i$. Here $x_{i,j}$ is the *return* of stock j in day i, i.e. $x_{i,j} = \frac{\text{price of stock } j \text{ on day } i}{\text{price of stock } j \text{ on day } i-1}$. On each day i, we use a vector $\mathbf{b}_i = (b_{i,1}, ..., b_{i,m})^T \in \Delta^m$ to represent the portfolio. Here Δ^m is the simplex $\{\mathbf{b} \in \mathbb{R}^m | \sum_{j=1}^m b_j = 1\}$. On day i we allocate $b_{i,j}$ of our current total wealth to stock j. Thus our return on day i is $\mathbf{b}_i^T \mathbf{x}_i$, compared with day $i-1$. The total return after n days is $T_n = \prod_{i=1}^n \mathbf{b}_i^T \mathbf{x}_i$.

With Universal Portfolios, we can approximate the return of optimal Constant Rebalancing Portfolio (CRP). A Constant Rebalancing Portfolio is a fixed wealth allocation throughout all days. Given a fixed wealth allocation $\mathbf{b} \in \Delta^m$, the return of CRP on day n is $S_n(\mathbf{b}) = \prod_{i=1}^n \mathbf{b}^T \mathbf{x}_i$. With Universal Portfolio algorithm, we can achieve

$$\frac{T_n}{S_n(\mathbf{b}^*)} \geq \frac{1}{(n+1)^{m-1}} \tag{1}$$

Here $\mathbf{b}^*$ is the optimal CRP. Considering the average return per day, we have

$$\left(\frac{T_n}{S_n(\mathbf{b}^*)}\right)^{\frac{1}{n}} \geq \frac{1}{(n+1)^{\frac{m-1}{n}}} \tag{2}$$

As $n \to \infty$, the right hand side approach 1. Thus we get an average daily return comparable with optimal CRP.

2.1 The Algorithm

On each day, the portfolio of Universal Portfolio algorithm is a weighted sum of all portfolios in Δ^m. The weight for each $\mathbf{b} \in \Delta^m$ is its return on previous day. Formally, on day i, the portfolio is

$$\hat{\mathbf{b}_i} = \frac{\displaystyle\int_{\Delta^m} S_{i-1}(\mathbf{b}) \cdot \mathbf{b} \, \mathrm{d}u(\mathbf{b})}{\displaystyle\int_{\Delta^m} S_{i-1}(\mathbf{b}) \, \mathrm{d}u(\mathbf{b})} \tag{3}$$

Here $u(\mathbf{b})$ is the CDF of uniform distribution on Δ^m.

2.2 Guarantee of Universal Portfolio

Theorem 1: Let $\mathbf{b}^*$ be the CRP that produce best total return by day n. With the universal portfolios in (3), $T_n = \prod_{i=1}^n \hat{\mathbf{b}_i}^T \mathbf{x}_i$. We have

$$\frac{T_n}{S_n(\mathbf{b}^*)} \geq \frac{e^{-1}}{(n+1)^{m-1}} \tag{4}$$

For the sake of simplicity, we only prove the bound with constant e^{-1} here. For proof of the better bound in (1), please refer to [1].

Proof: Follow the definitions, it is easy to show that $T_n = \mathbb{E}_{\mathbf{b}\in\Delta^m}[S_n(\mathbf{b})]$. Here $\mathbf{b} \in \Delta^m$ follows the uniform distribution. Let $\Delta_\alpha^m = \{(1-\alpha)\mathbf{b}^* + \alpha\mathbf{z} : \mathbf{z} \in \Delta^m\}$, then Δ_α^m is a subregion in Δ^m. $\forall \mathbf{b} \in \Delta_\alpha^m$ we have

$$S_n(\mathbf{b}) = \prod_{i=1}^n \mathbf{b}^T\mathbf{x}_i = \prod_{i=1}^n \Big[(1-\alpha)\mathbf{b}^{*T}\mathbf{x}_i + \alpha\mathbf{z}^T\mathbf{x}_i\Big] \geq \prod_{i=1}^n \Big[(1-\alpha)\mathbf{b}^{*T}\mathbf{x}_i\Big] = (1-\alpha)^n S_n(\mathbf{b}^*)$$

Note that

$$\int_{\mathbf{b}\in\Delta_\alpha^m} \mathrm{d}u(\mathbf{b}) = \int_{\mathbf{z}\in\Delta^m} \mathrm{d}u(\alpha\mathbf{z}) = \alpha^{m-1} \int_{\mathbf{z}\in\Delta^m} \mathrm{d}u(\mathbf{z}) = \alpha^{m-1}$$

$$\left(\int_{\mathbf{b}\in\Delta_\alpha^m} du(\mathbf{b}) = \text{Volumn of } \Delta_\alpha^m = \alpha^{m-1}\right)$$

Thus

$$\begin{aligned}
T_n = \mathop{\mathbb{E}}_{\mathbf{b}\in\Delta^m}[S_n(\mathbf{b})] &= \int_{\mathbf{b}\in\Delta^m} S_n(\mathbf{b}) \, \mathrm{d}u(\mathbf{z}) \geq \int_{\mathbf{b}\in\Delta_\alpha^m} S_n(\mathbf{b}) \, \mathrm{d}u(\mathbf{b}) \\
&= \int_{\mathbf{b}\in\Delta_\alpha^m} S_n(\mathbf{b}) \, \mathrm{d}u(\mathbf{b}) \geq \int_{\mathbf{b}\in\Delta_\alpha^m} (1-\alpha)^n S_n(\mathbf{b}^*) \, \mathrm{d}u(\mathbf{b}) \\
&= (1-\alpha)^n S_n(\mathbf{b}^*) \int_{\mathbf{b}\in\Delta_\alpha^m} \mathrm{d}u(\mathbf{b}) \\
&= (1-\alpha)^n \alpha^{m-1} S_n(\mathbf{b}^*)
\end{aligned}$$

Let $\alpha = \frac{1}{n+1}$,

$$T_n \geq \left(\frac{n}{n+1}\right)^n \frac{1}{(n+1)^{m-1}} S_n(\mathbf{b}^*) \geq \frac{e^{-1}}{(n+1)^{m-1}} S_n(\mathbf{b}^*)$$

3 Application of Online Learning to Supervised Learning

Given a set of training data $T = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_m, y_m)\}$, where each $(\mathbf{x}_i, y_i)$ is sampled from some distribution $\mathcal{D}$. We want to learn a model with parameters θ to minimize the loss $\sum_{i=1}^m l(\theta; (\mathbf{x}_i, y_i))$. This is the standard setting in supervised learning. However, if we see the data points in T as a sequence, we can convert it to an online learning problem. At step k, we have $(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{k-1}, y_{k-1})$, and want to learn parameters θ_k to predict the label of $\mathbf{x}_k$. The regret of this online learning problem at step T is

$$\text{reg}_T = \sum_{i=1}^T l(\theta_i; (\mathbf{x}_i, y_i)) - \inf_\theta \sum_{i=1}^T l(\theta; (\mathbf{x}_i, y_i)).$$

We call $\sum_{i=1}^m l(\theta; (\mathbf{x}_i, y_i))$ the *empirical risk*, which is the loss evaluated on training data. Since we want to use the model for prediction, we care more about the performance on unseen data points from distribution $\mathcal{D}$, which is the *population risk*: $\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[l(\theta; (\mathbf{x}, y))\right]$. Usually there will be a gap between empirical risk and population risk, due to overfitting on the training data. This gap is called *generalization error*:

$$\left| \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \left[l(\theta; (\mathbf{x}, y))\right] - \frac{1}{m}\sum_{i=1}^m l(\theta; (\mathbf{x}_i, y_i)) \right|.$$

In this section, we'll show that with a good online learning algorithm we can always get a small population risk, which is bounded by the regret of the algorithm. Before that, we need to introduce Martingale and its properties.

3.1 Martingale

Definition 1 (Martingale): A sequence of random variables $X_1, ..., X_m$ is called Martingale if it satisfies:

1. $\mathbb{E}\left[|X_i|\right] < \infty$
2. $\mathbb{E}\left[X_i | X_1, ..., X_{i-1}\right] = X_{i-1}$

for any $i \in [m]$. More generally, we should define the second expectation with *filtration*[2]. Filtration can be interpreted as all historical information. We use F_i to represent the filtration that contains all information needed to measure X_i. Thus we can think F_i as the history of the first i steps. Then the second equation becomes $\mathbb{E}\left[X_i | F_{i-1}\right] = X_{i-1}$. From now on, we will use F_i to represent the filtration.

Definition 2 (Martingale Difference): A sequence of random variables $Z_1, ..., Z_m$ is called Martingale Difference if it satisfies:

1. $\mathbb{E}\left[|Z_i|\right] < \infty$
2. $\mathbb{E}\left[Z_i | F_{i-1}\right] = 0$

for any $i \in [m]$.

We have two straightforward conclusions for Martingale and Martingale Difference:

1. If sequence $X_1, ..., X_m$ is Martingale, then $Z_1, ..., Z_m$ is Martingale Difference, where $Z_i = X_i - X_{i-1}$.
2. If $Z_1, ..., Z_m$ is Martingale Difference, then $\mathbb{E}[Z_i] = 0$, $\forall i \in [m]$.

3.2 Online to Batch

In this section, we show that with a good online learning algorithm, we can always get small population risk for a supervised learning task. This application of online learning to supervised learning is also known as *Online to Batch* conversion. We use $\mathcal{L}(\theta)$ to represent the population risk of parameter θ. Let $\theta^* = \text{argmin}_\theta \mathcal{L}(\theta)$, which is the optimal parameter. Recall that, with an online learning algorithm, we learn parameter θ_i at step i with $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{i-1}, y_{i-1})\}$. $\forall i \in [m]$, let

$$Z_i = [l(\theta_i; (\mathbf{x}_i, y_i)) - \mathcal{L}(\theta_i)] - [l(\theta^*; (\mathbf{x}_i, y_i)) - \mathcal{L}(\theta^*)].$$

Lemma 1: Sequence $Z_1, ..., Z_m$ is Martingale Difference.
Proof:

$$\begin{aligned}\mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D}}[Z_i|F_{i-1}] &= \mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D}}[l(\theta_i; (\mathbf{x}_i, y_i)) - \mathcal{L}(\theta_i)|F_{i-1}] - \mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D}}[l(\theta^*; (\mathbf{x}_i, y_i)) - \mathcal{L}(\theta^*)|F_{i-1}] \\ &= [\mathcal{L}(\theta_i) - \mathcal{L}(\theta_i)] - [\mathcal{L}(\theta^*) - \mathcal{L}(\theta^*)] = 0\end{aligned}$$

Lemma 2: $\sum_{i=1}^m \mathcal{L}(\theta_i) \leq m\mathcal{L}(\theta^*) + \text{reg}_m - \sum_{i=1}^m Z_i$
Proof:

$$\begin{aligned}\sum_{i=1}^m(\mathcal{L}(\theta_i) - \mathcal{L}(\theta^*)) &= \sum_{i=1}^m (l(\theta_i; (\mathbf{x}_i, y_i)) - l(\theta^*; (\mathbf{x}_i, y_i)) - Z_i) \\ &= \sum_{i=1}^m l(\theta_i; (\mathbf{x}_i, y_i)) - \sum_{i=1}^m l(\theta^*; (\mathbf{x}_i, y_i)) - \sum_{i=1}^m Z_i \\ &\leq \sum_{i=1}^m l(\theta_i; (\mathbf{x}_i, y_i)) - \inf_\theta \sum_{i=1}^m l(\theta; (\mathbf{x}_i, y_i)) - \sum_{k=1}^m Z_i \\ &= \text{reg}_m - \sum_{i=1}^m Z_i\end{aligned}$$

Thus $\sum_{i=1}^m \mathcal{L}(\theta_i) \leq m\mathcal{L}(\theta^*) + \text{reg}_m - \sum_{i=1}^m Z_i$.

Theorem 2 (Online to Batch): Let $\theta_1, ..., \theta_m$ be the outputs of the online learning algorithm. Then

$$\mathbb{E}\left[\frac{1}{m}\sum_{i=1}^m \mathcal{L}(\theta_i)\right] \leq \mathcal{L}(\theta^*) + \frac{1}{m}\mathbb{E}[\text{reg}_m]. \tag{5}$$

When $\mathcal{L}$ is convex, we have

$$\mathbb{E}\left[\mathcal{L}\left(\frac{1}{m}\sum_{i=1}^m \theta_i\right)\right] \leq \mathcal{L}(\theta^*) + \frac{1}{m}\mathbb{E}[\text{reg}_m]. \tag{6}$$

Proof: Since $Z_1, ..., Z_m$ is Martingale Difference, we have $\mathbb{E}[Z_i] = 0, \forall i \in [m]$. Take expectation on both sides of Lemma 2, we get (5). With the convexity of $\mathcal{L}$,

$$\mathbb{E}\left[\mathcal{L}\left(\frac{1}{m}\sum_{i=1}^{m}\theta_i\right)\right] \leq \mathbb{E}\left[\frac{1}{m}\sum_{i=1}^{m}\mathcal{L}(\theta_i)\right].$$

Thus (6) follows from (5).

4 Dual Norm and Fenchel Conjugate

In this section, we provide some preliminary knowledge for Online Mirror Descent, including dual norm and Fenchel Conjugate.

4.1 Dual Norm

In a *vector space* $\mathcal{X}$, a *norm* is a function mapping from $\mathcal{X}$ to $\mathbb{R}$. A norm in $\mathcal{X}$ measures the length of vectors in $\mathcal{X}$. For a formal definition of norm, you can refer to [**?**]. For example, in vector space $\mathbb{R}^n$, we have 2-norm: $\|\mathbf{x}\|_2 = \sqrt{\mathbf{x}^T\mathbf{x}}$, 1-norm: $\|\mathbf{x}\|_1 = \sum_{i=1}^{n}|x_i|$, and ∞-norm $\|\mathbf{x}\|_\infty = \max_i |x_i|$, $\forall \mathbf{x} = (x_1, ..., x_n) \in \mathbb{R}^n$.

Given a norm $\|\cdot\|$ in vector space $\mathbb{R}^n$, we can define its *dual norm* $\|\cdot\|_*$

$$\|\mathbf{x}\|_* = \sup_{\|\mathbf{y}\|\leq 1} \mathbf{x}^T\mathbf{y}.$$

For example, it is easy to verity that the dual norm of $\|\cdot\|_1$ in $\mathbb{R}^n$ is $\|\cdot\|_\infty$. With $1 < p < \infty$, we can define p-norm: $\|\mathbf{x}\|_p = (\sum_{i=1}^{n}|x_i|^p)^{\frac{1}{p}}$. If $1 < q < \infty$ and $\frac{1}{p} + \frac{1}{q} = 1$, then $\|\cdot\|_q$ is the dual norm of $\|\cdot\|_p$, and vice versa.

4.2 Fenchel Conjugate

Given a function defined in $\mathbb{R}^n$, we define its *Fenchel Conjugate* function.
Definition 3 (Fenchel Conjugate/Dual): Let $h : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, its conjugate(dual) function is $h^* : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$,

$$h^*(\theta) = \sup_{\mathbf{x}\in\mathbb{R}^n} \{\theta^T\mathbf{x} - h(\mathbf{x})\}.$$

(For functions not defined in the whole $\mathbb{R}^n$ space, we can define its value as $+\infty$ out of the domain of definition, and extend the domain of definition to $\mathbb{R}^n$.)

We have three important observations about conjugate functions:

1. Conjugate function is always convex. Note that for any function h and a fixed $\mathbf{x} \in \mathbb{R}^n$, $\theta^T\mathbf{x} - h(\mathbf{x})$ is a linear of θ. Thus the conjugate function is a point-wise supremum over a set of linear functions, which is convex.

2. Conjugate function encodes all the gradients, or tangent planes of the original function. Consider the piecewise linear function $h(x)$ and its dual function $h^*(\theta)$,

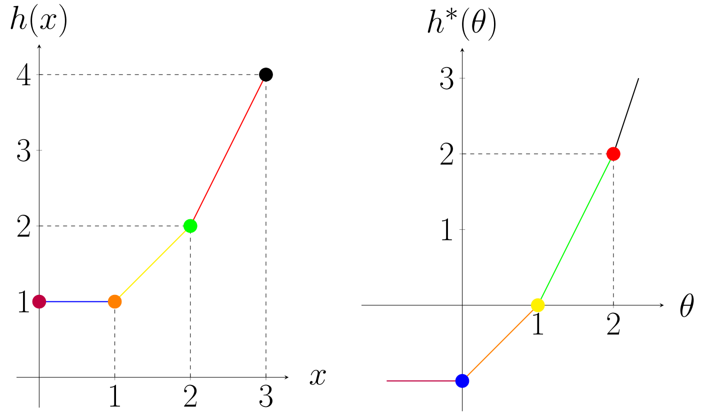

Figure 1: Conjugate Function Encodes the Tangent Planes

$$h(x) = \begin{cases} 1, & 0 \le x \le 1 \\ x, & 1 < x \le 2 \\ 2x-2, & 2 < x \le 3 \\ +\infty, & \text{otherwise} \end{cases} \qquad h^*(\theta) = \begin{cases} -1, & x \le 0 \\ x-1, & 0 < x \le 1 \\ 2x-2, & 1 < x \le 2 \\ 3x-4, & x > 2 \end{cases}$$

In Figure 1, both h^* and h are piece-wise linear functions. The plot of h^* is segmented at $\theta = 0, 1, 2$. And the gradients of h in three segments are $0, 1$ and 2, respectively. Thus, each position where gradients of h^* changes, corresponds to a gradient value of h. Also, the gradients of h^* are encoded by h. The plot of h is segmented at $x = 0, 1, 2, 3$. And the gradients of h^* in four segments are $0, 1, 2$ and 3, respectively. The encoding relationship is indicated by the colors of dots and line segments in Figure 1.

3. Suppose h is defined and derivable in $\mathbb{R}^n$. Fix $\theta_1 \in \mathbb{R}^n$, let $\mathbf{x}_1 = \text{argmax}\{\mathbf{x}^T\theta_1 - h(\mathbf{x})\}$. Then we have

$$\theta_1 = \nabla h(\mathbf{x}_1), \quad \mathbf{x}_1 = \nabla h^*(\theta_1)$$

This is a relatively formal description of the *encoding* illustrated in Figure 1. In underivable cases like Figure 1, we replace the gradients with *subgradients*[4]. Let $\partial h(\mathbf{x}_1)$ be the subgradients of h at $\mathbf{x}_1$ and $\partial h^*(\theta_1)$ be the subgradients of h^* at θ, then we have $\theta_1 \in \partial h(\mathbf{x}_1)$ and $\mathbf{x}_1 \in \partial h^*(\theta_1)$.

Since $\mathbf{x}_1 = \text{argmax}\{\mathbf{x}^T\theta_1 - h(\mathbf{x})\}$, $\mathbf{x}_1$ is a stationary point of $\mathbf{x}^T\theta_1 - h(\mathbf{x})$. Thus

$$\nabla_{\mathbf{x}} \left(\mathbf{x}^T\theta_1 - h(\mathbf{x})\right)|_{\mathbf{x}=\mathbf{x}_1} = \theta_1 - \nabla h(\mathbf{x}_1) = 0$$

So $\theta_1 = \nabla h(\mathbf{x}_1)$. By definition of h^*, $h^*(\theta_1) = \mathbf{x}_1^T\theta_1 - h(\mathbf{x}_1)$. $\forall \theta_2 \in \mathbb{R}^n$, we have $h^*(\theta_2) \ge \mathbf{x}_1^T\theta_2 - h(\mathbf{x}_1)$. Thus $\mathbf{x}_1^T\theta_1 - h^*(\theta_1) = h(\mathbf{x}_1) \ge \mathbf{x}_1^T\theta_2 - h^*(\theta_2)$. So θ_1 is a maximum point (thus a stationary point) of function $f(\theta) = \mathbf{x}_1^T\theta - h^*(\theta)$. Thus $\nabla f(\theta_1) = \mathbf{x}_1 - \nabla h^*(\theta_1) = 0$. So $\mathbf{x}_1 = \nabla h^*(\theta_1)$. (We assume both h and h^* are derivable here. But we can generalize the idea to underivable case using subgradient.)

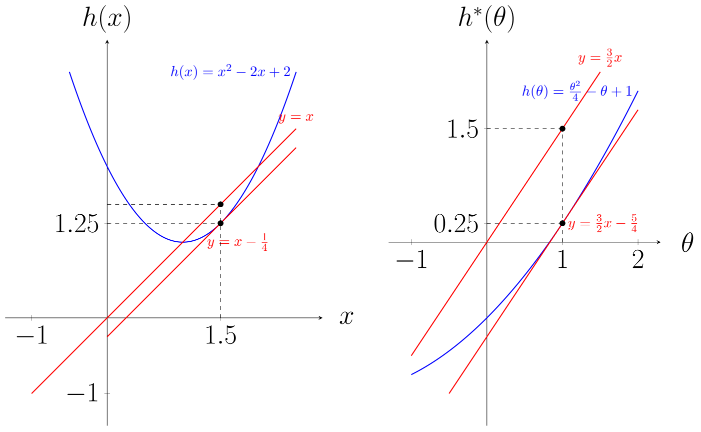

Figure 2: $\theta_1 = \nabla h(\mathbf{x}_1)$ and $\mathbf{x}_1 = \nabla h^*(\theta_1)$

Let $h(x) = x^2 - 2x + 2$, thus $h^*(\theta) = \max_x(\theta x - x^2 + 2x - 2) = \frac{\theta^2}{4} + \theta - 1$. We plot h and h^* in Figure 2. Let $\theta_1 = 1$, let $x_1 = \text{argmax}_x\{\theta_1 x - h(x)\} = \text{argmax}_x\{-x^2 + 3x - 2\} = \frac{3}{2}$. $h^{*\prime}(\theta) = \frac{\theta}{2} + 1, h'(x) = 2x - 2$. Thus $h'(x_1) = 1 = \theta_1$, and $h^{*\prime}(\theta_1) = \frac{3}{2} = x_1$.

Some examples of Fenchel conjugate functions:

$$(1)\;\; f(x) = ax - b,\; f^*(\theta) = \begin{cases} b, & \theta = a \\ +\infty, & \theta \neq a \end{cases}$$

$$(2)\;\; f(x) = |x|,\; f^*(\theta) = \begin{cases} 0, & -1 < \theta < 1 \\ +\infty, & |x| \geq 1 \end{cases}$$

5 Some Concepts

Definition 4 (L-Lipschitz): Given a norm $\|\cdot\|$ and $L > 0$, a function $f : \mathcal{X} \to \mathbb{R}$ is called L-Lipschitz if $\|\nabla f(x)\|_* \leq L, \forall x \in \mathcal{X}$. L-Lipschitz depicts the continuity of f.
Definition 5 (β-Smooth): Given a norm $\|\cdot\|$ and $\beta > 0$, a function $f : \mathcal{X} \to \mathbb{R}$ is called β-smooth if $\|\nabla f(x) - \nabla f(y)\|_* \leq \beta\|x - y\|, \forall x, y \in \mathcal{X}$. β-smooth depicts the smoothness of f.
Definition 6 (Strongly Convex): Given $\alpha > 0$, a function f is called α-strongly convex if

$$f(x) - f(y) \leq \langle \nabla f(x), x - y\rangle - \frac{\alpha}{2}\|x - y\|^2, \quad \forall x, y.$$

Definition 7 (Bregman Divergence): Given a function f (typically a strongly convex function), the Bregman Divergence D_f of f is defined as

$$D_f(x, y) = f(x) - f(y) - \langle \nabla f(y), x - y\rangle, \quad \forall x, y.$$

References

[1] Blum, Avrim, and Adam Kalai. ”Universal portfolios with and without transaction costs.” Machine Learning 35.3 (1999): 193-205.

[2] Wikipedia contributors. Martingale (probability theory) wikipedia, the free encyclopedia, 2018. [Online; accessed 18-March-2018].

[3] Wikipedia contributors. Norm (mathematics) wikipedia, the free encyclopedia, 2018. [Online; accessed 18-March-2018].

[4] Wikipedia contributors. Subderivative wikipedia, the free encyclopedia, 2018. [Online; accessed 19-March-2018].