Optimization for Machine Learning

Reading group, 2017-2018

March 15, 2018

Contents

Part I

Part 1: Fall 2017

Chapter 1

Introduction

Speaker: Patrick Rebeschini, 12/10/2017.

Many problems in statistics and machine learning can be formulated as the problem of computing a solution to:

$$\begin{aligned} &\text{minimize} \quad && r(x) := \mathbf{E}\ell(x^T\Phi(W), Y) \\ &\text{subject to} \quad && x \in \mathcal{X}, \end{aligned} \tag{1.1}$$

given only data in the form of m i.i.d. samples $(W_1, Y_1), \ldots, (W_m, Y_m) \sim (W, Y) \in \mathcal{W} \times \mathcal{Y}$, i.e., without knowledge of the distribution of (W, Y) (in particular, without knowledge of the function r that we want to minimize!). Here, the function $\Phi : \mathcal{W} \to \mathbb{R}^n$ (known) is a feature map, the function $x \in \mathcal{X} \subseteq \mathbb{R}^n \to x^T\Phi(W)$ is a linear predictor for the label Y (the domain/constraint set $\mathcal{X}$ is known), the function $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ (known) is a loss function characterizing the penalty in committing a wrong prediction, and $r(x)$ is the *test cost* or *expected risk* associated to the parameter x. In this setting, one wants to find $x^\star$ that yields the best linear predictor over unseen data using the data in the training set, under the assumption that the unseen data share the same (unknown) distribution of observed samples.

In binary classification, one has $\mathcal{Y} = \{-1, 1\}$ and loss functions are typically of the form $\ell(z, y) = \varphi(-zy)$ for a given $\varphi : \mathbb{R} \to \mathbb{R}$. A few examples are:

- Zero-One loss (a.k.a. the true loss): $\varphi(u) = \mathbf{1}_{u\geq 0}$.
- Exponential loss: $\varphi(u) = \exp(u)$.
- Hinge loss: $\varphi(u) = \max\{0, 1 + u\}$ (giving SVM).
- Logistic loss: $\varphi(u) = \log_2(1 + \exp(u))$.

In regression, one has $\mathcal{Y} = \mathbb{R}$, and the typical loss is:

- Least-squares loss: $\ell(u, y) = (u - y)^2$.

Perhaps the most intuitive paradigm for solving problem (1.1) is the *empirical risk minimization*, that is, the idea to minimize the *training cost* or *empirical risk*:

$$\begin{aligned} &\text{minimize} \quad && R(x) := \frac{1}{m}\sum_{i=1}^{m} R_i(x) \\ &\text{subject to} \quad && x \in \mathcal{X}, \end{aligned} \tag{1.2}$$

where $R_i(x) := \ell(x^T\Phi(W_i), Y_i)$. This approach is motivated by the fact that for each $x \in \mathcal{X}$ the law of large number gives $r(x) \approx R(x)$ so that one might expect that $x^\star \approx X^\star$, where $X^\star \in \arg\min_{x\in\mathcal{X}} R(x)$. This intuition can be made precise, and Proposition 1.0.1 below shows that the estimation error $r(X^\star) - r(x^\star) \geq 0$ is controlled by suprema of random processes.

Proposition 1.0.1. We have

$$r(X^\star) - r(x^\star) \leq \sup_{x\in\mathcal{X}}(r(x) - R(x)) + \sup_{x\in\mathcal{X}}(R(x) - r(x)).$$

Proof. By adding and subtracting terms, we get

$$r(X^\star) - r(x^\star) = r(X^\star) - R(X^\star) + R(X^\star) - R(x^\star) + R(x^\star) - r(x^\star).$$

As $X^\star$ is a minimizer of R, we have $R(X^\star) - R(x^\star) \leq 0$. The proof follows by taking the supremum over $x \in \mathcal{X}$. □

This analysis assumes that one can actually compute an exact minimizer $X^\star$ for the training cost R. With a finite amount of computational resources available, however, typically one can only compute an approximate minimizer. Let $\hat{X}_t \in \mathcal{X}$ denote an approximate solution to $\arg\min_{x\in\mathcal{X}} R(x)$ that is computed after t iterations of a given algorithmic procedure. Proceeding as above we have the following result.

Proposition 1.0.2. We have

$$r(\hat{X}_t) - r(x^\star) \leq \underbrace{\sup_{x\in\mathcal{X}}(r(x) - R(x)) + \sup_{x\in\mathcal{X}}(R(x) - r(x))}_{\text{STATISTICS}} + \underbrace{R(\hat{X}_t) - R(X^\star)}_{\text{OPTIMIZATION}}. \tag{1.3}$$

Proof. By adding and subtracting terms, we get

$$r(\hat{X}_t) - r(x^\star) = r(\hat{X}_t) - R(\hat{X}_t) + R(\hat{X}_t) - R(X^\star) + R(X^\star) - R(x^\star) + R(x^\star) - r(x^\star).$$

As $X^\star$ is a minimizer of R, we have $R(X^\star) - R(x^\star) \leq 0$. The proof follows by taking the supremum over $x \in \mathcal{X}$. □

The problem is now to estimate how close the expected risk of the approximate empirical risk minimizer $r(\hat{X}_t)$ is to the minimal expected risk $r(x^\star)$ in terms of the number of training samples m, the dimension of the feature map n, the number of iterates t for the optimization routine, and the other parameters that define the model at hand (for example, the radius of the parameter set $\mathcal{X}$, properties of the loss function ℓ, and of the feature map Φ). To this end, as Proposition 1.0.2 shows, we need to control two random terms: the *STATISTICS* term $\sup_{x\in\mathcal{X}}(r(x) - R(x)) + \sup_{x\in\mathcal{X}}(R(x) - r(x))$ and the *OPTIMIZATION* term $R(\hat{X}_t) - R(X^\star)$.

For the most part this reading group will focus on algorithms that can be used to approximately minimize the empirical risk and hence control the *OPTIMIZATION* term in (1.3), making various assumptions for the loss function ℓ and the parameter space $\mathcal{X}$, starting from the fruitful assumption of convexity.[1] Along with convexity, we state some of the main properties that a generic function f can have that are used to assess the quality of algorithms to minimize f. Here we assume that f is differentiable, but analogous notions can be defined for non-differentiable functions using the notion of subgradients. We also mention in brackets equivalent (local) definitions, which apart from L-Lipschitz hold when f is twice-differentiable.

- Convexity: $f(x) - f(y) \leq \nabla f(x)^T(x - y)$ for any $x, y \in \mathbb{R}^n$ ($\nabla^2 f(x) \succcurlyeq 0$ for any $x \in \mathbb{R}^n$).
- L-Lipschitz: $|f(x) - f(y)| \leq L\|x - y\|_2$ for any $x, y \in \mathbb{R}^n$ ($\|\nabla f(x)\|_2 \leq L$ for any $x \in \mathbb{R}^n$).
- β-Smoothness: $\|\nabla f(x) - \nabla f(y)\|_2 \leq \beta\|x - y\|_2$ for any $x, y \in \mathbb{R}^n$ ($\nabla^2 f(x) \preccurlyeq \beta I$ for any $x \in \mathbb{R}^n$).

[1]Convexity is only needed to perform the analysis of the algorithms that we introduce, but it is not needed to run the algorithms themselves; in fact, in practice the algorithms that we define are run also on non-convex problems.

- α-Strong convexity: $f(x) - f(y) \leq \nabla f(x)^T(x-y) - \frac{\alpha}{2}\|x-y\|_2^2$ for any $x, y \in \mathbb{R}^n$ ($\nabla^2 f(x) \succcurlyeq \alpha I$ for any $x \in \mathbb{R}^n$).

Going back to the losses defined above, it is easy to check that the convex losses below satisfy the following:

- Exponential loss: $\varphi(u) = \exp(u)$: Lipschitz NO, Smooth NO, strongly-convex NO.
- Hinge loss: $\varphi(u) = \max\{0, 1+u\}$: Lipschitz YES, Smooth NO, strongly-convex NO.
- Logistic loss: $\varphi(u) = \log_2(1+\exp(u))$: Lipschitz YES, Smooth YES, strongly-convex NO.
- Least-squares loss: $\ell(u, y) = (u-y)^2$: Lipschitz NO, Smooth YES, strongly-convex YES.

Note that as far as the empirical risk minimization goes, what we care about are the (almost sure) properties of the function $x \in \mathcal{X} \to R(x) := \frac{1}{m}\sum_{i=1}^m R_i(x) = \frac{1}{m}\sum_{i=1}^m \ell(x^T\Phi(W_i), Y_i)$. In particular, if $\mathcal{X}$ and Φ are bounded, then $x^T\Phi(W_i)$ is bounded and we can restrict the definitions above to hold on a bounded set. Henceforth, we say that a function $x \in \text{Dom}(f) \subseteq \mathbb{R}^n \to f(x)$ is L-Lipschitz if $|f(x) - f(y)| \leq L\|x-y\|_2$ for any $x, y \in \text{Dom}(f)$ ($\|\nabla f(x)\|_2 \leq L$ for any $x \in \text{Dom}(f)$). Assuming boundedness provides extra flexibility, allowing one to directly use these properties in setting where otherwise they would not apply (take the exponential loss, for instance, which is only Lipschitz on a bounded interval).

Even if we assume boundedness, assuming that the empirical risk R is strongly convex is typically a strong assumption. In fact, note that *for any* $x \in \mathcal{X}$ (that is, regardless of the properties of $\mathcal{X}$) the Hessian

$$\nabla^2 R(x) = \frac{1}{m}\sum_{i=1}^m \frac{\partial^2}{\partial z^2}\ell(x^T\Phi(W_i), Y_i)\,\Phi(W_i)\Phi(W_i)^T \in \mathbb{R}^{n\times n}$$

is not invertible if $m < n$, being a sum of $m < n$ rank-1 matrices. So in this case $x \to \nabla^2 R(x)$ can not be strongly convex. In applications where $m \geq n$, the empirical covariance can be invertible, but it is typically the case that the strong convexity parameter α is very small, of order $1/m$, effectively $\alpha \approx 0$ (or close to machine precision) for applications involving large datasets. For this reason, in this reading group we will only focus on results about Lipschitz and smooth functions, which are more natural assumptions for the empirical risk R, as we now show.

First, note that if we assume that ℓ is L_{loss}-Lipschitz in the first coordinate, namely, $z \to \ell(z, y)$ is L_ℓ-Lipschitz for any $y \in \mathcal{Y}$, and that the feature map Φ is bounded in ℓ_2, namely, $\|\Phi\|_2 \leq G$, then R is GL_ℓ-Lipschitz:

$$|R(x) - R(y)| \leq \frac{1}{m}\sum_{i=1}^m |\ell(x^T\Phi(W_i), Y_i) - \ell(y^T\Phi(W_i), Y_i)| \leq \frac{L_\ell}{m}\sum_{i=1}^m \|(x-y)^T\Phi(W_i))\|_2 \leq GL_\ell\|x-y\|_2,$$

where for the last inequality we used Cauchy-Schwarz. Second, note that if we assume that ℓ is β_ℓ-Lipschitz in the first coordinate, namely, $z \to \ell(z, y)$ is β_{loss}-Lipschitz for any $y \in \mathcal{Y}$, and that $\|\Phi\|_2 \leq G$, then R is $G^2\beta_{\text{loss}}$-smooth:

$$\begin{aligned}\|\nabla R(x) - \nabla R(y)\|_2 &\leq \frac{1}{m}\sum_{i=1}^m \left\|\left(\frac{\partial}{\partial z}\ell(x^T\Phi(W_i), Y_i) - \frac{\partial}{\partial z}\ell(y^T\Phi(W_i), Y_i)\right)\Phi(W_i)\right\|_2 \\ &\leq \frac{1}{m}\sum_{i=1}^m \beta_\ell|(x-y)^T\Phi(W_i)|\|\Phi(W_i)\|_2 \leq G^2\beta_\ell\|x-y\|_2.\end{aligned}$$

Before we embark on the adventure of minimizing the empirical risk and bound the *OPTIMIZATION* term in (1.3), we should try to develop some basic understanding of what to expect from the behavior of the *STATISTICS* term as well. This understanding will give us insights on the type of precision that we should strive for to control the *OPTIMIZATION* term. To this end, we now give a brief detour on statistical learning theory and derive an explicit bound in the case when the loss function ℓ is Lipschitz (no assumption on convexity is made for this result), the feature map is bounded, and also the parameter space $\mathcal{X}$ is bounded. The proof is instructive and relies on classical machinery of concentration inequalities and Rademacher complexity.

1.1 Statistics term

Statistical learning theory focuses on understanding how the estimation error $r(X^\star) - r(x^\star)$ converges to zero as a function of the sample size m, the loss function ℓ, the parameter space $\mathcal{X}$, and possibly some properties of the distribution of $\Phi(W)$ and Y (recall that the distribution of the data is not known). We will prove the following results, inspired by the presentation in [8].

Proposition 1.1.1 (Mean). Let $z \to \ell(z,y)$ be L_ℓ-Lipschitz for any $y \in \mathcal{Y}$, $\sup_{x\in\mathcal{X}} \|x\|_2 \le B$ and $\|\Phi(W)\|_2 \le G$ a.s.. Then,

$$\mathbf{E}[\text{STATISTICS}] = \mathbf{E}\sup_{x\in\mathcal{X}}(r(x) - R(x)) + \mathbf{E}\sup_{x\in\mathcal{X}}(R(x) - r(x)) \le 4\frac{BGL_\ell}{\sqrt{m}}.$$

Proposition 1.1.2 (High probability). Let $z \to \ell(z,y)$ be L_ℓ-Lipschitz for any $y \in \mathcal{Y}$, $\sup_{x\in\mathcal{X}} \|x\|_2 \le B$ and $\|\Phi(W)\|_2 \le G$ a.s.. Then, with probability at least $1-\delta$ we have

$$\text{STATISTICS} = \sup_{x\in\mathcal{X}}(r(x) - R(x)) + \sup_{x\in\mathcal{X}}(R(x) - r(x)) \le 2\frac{\ell_0 + BGL_\ell}{\sqrt{m}}\left(2 + \sqrt{2\log\frac{1}{\delta}}\right),$$

where $\ell_0 := \sup_{y\in\mathcal{Y}} |\ell(0,y)|$.

The quality of the bound in Proposition 1.1.2 is typical in statistical learning theory. In particular, it is typically the case that the *slow* rate $O(1/\sqrt{m})$ can not be improved unless other assumptions are taken into consideration that allow to prove the *fast* rate $O(1/m)$. These upper bounds suggest[2] that we only need to approximate the *OPTIMIZATION* term up to precision $O(1/\sqrt{m})$ (resp. $O(1/m)$).

Bound on the mean

We prove Proposition 1.1.1. An important tool to bound the mean of a supremum of a random process is the Rademacher complexity. The Rademacher complexity of a set is defined as follows.

Definition 1.1.1. *The Rademacher complexity of a set $T \subseteq \mathbb{R}^m$ is*

$$\mathcal{R}(T) := \mathbf{E}\sup_{t\in T}\frac{1}{m}\sum_{i=1}^m \epsilon_i t_i,$$

where $\epsilon_1, \dots, \epsilon_m$ are i.i.d. random variables uniform in $\{-1, 1\}$.

The quantity $\mathcal{R}(T)$ is a measure of how complex the set T is, as $\sup_{t\in T}\sum_{i=1}^m \epsilon_i t_i$ describes how well elements in T can replicate the sign pattern of a random signal $\epsilon = (\epsilon_1, \dots, \epsilon_m)$. One way of seeing this is to restrict to the case $T \subseteq [-1,1]^n$. If $T = [-1,1]^n$ then $\mathcal{R}(T) = 1$, as for any realization of the signal ϵ we can find $t \in T$ that has its same sign pattern. More generally, if $T \subseteq [-1,1]^n$ with k-sparse components, then $\mathcal{R}(T) = k/m$.

One reason why the Rademacher complexity is a useful notion is given by the following lemma, that shows how this quantity behaves with respect to composition with Lipschitz functions.

Lemma 1.1.3 (Contraction property). For each $i \in \{1, \dots, m\}$, let $\varphi_i : \mathbb{R} \to \mathbb{R}$ be a L-Lipschitz function. Let $(\varphi_1, \dots, \varphi_m) \circ T = \{(\varphi_1(t_1), \dots, \varphi_m(t_m))^T : t \in T\}$. Then,

$$\mathcal{R}((\varphi_1, \dots, \varphi_m) \circ T) \le L\mathcal{R}(T),$$

or, explicitly,

$$\mathbf{E}\sup_{t\in T}\sum_{i=1}^m \epsilon_i \varphi_i(t_i) \le L\mathbf{E}\sup_{t\in T}\sum_{i=1}^m \epsilon_i t_i.$$

[2]Caveats are in order: after all the result in (1.3) is only an upper-bound, and it is often the case that in practice one does not know the constants sitting in front of the error bounds that one can prove for both the *OPTIMIZATION* term and the *STATISTICS* term, i.e., the constants G, L, R in our case.

Proof. First, note that for $S \subseteq \mathbb{R}^2$ and $\varphi : \mathbb{R} \to \mathbb{R}$ 1-Lipschitz we have

$$\begin{aligned}\sup_{s\in S}(s_1 + \varphi(s_2)) + \sup_{s\in S}(s_1 - \varphi(s_2)) &= \sup_{s,s'\in S}(s_1 + s_1' + \varphi(s_2) - \varphi(s_2')) = \sup_{s,s'\in S}(s_1 + s_1' + |\varphi(s_2) - \varphi(s_2')|) \\ &\le \sup_{s,s'\in S}(s_1 + s_1' + |s_2 - s_2'|) = \sup_{s,s'\in S}(s_1 + s_1' + s_2 - s_2') \\ &= \sup_{s\in S}(s_1 + s_2) + \sup_{s\in S}(s_1 - s_2).\end{aligned}$$

Then, if the functions φ_i's are 1-Lipschitz we have, for any $k \in \{1, \dots, m\}$,

$$\begin{aligned}\mathbf{E}\sup_{t\in T}\sum_{i=1}^m \epsilon_i\varphi_i(t_i) &= \mathbf{E}\mathbf{E}\Big[\sup_{t\in T}\sum_{i=1}^m \epsilon_i\varphi_i(t_i)\Big|\epsilon_1, \dots, \epsilon_{i-1}, \epsilon_{i+1}, \dots, \epsilon_m\Big] \\ &= \frac{1}{2}\mathbf{E}\Big[\sup_{t\in T}\Big(\sum_{i\ne k}\epsilon_i\varphi_i(t_i) + \varphi_k(t_k)\Big) + \sup_{t\in T}\Big(\sum_{i\ne k}\epsilon_i\varphi_i(t_i) - \varphi_k(t_k)\Big)\Big] \\ &\le \frac{1}{2}\mathbf{E}\Big[\sup_{t\in T}\Big(\sum_{i\ne k}\epsilon_i\varphi_i(t_i) + t_k\Big) + \sup_{t\in T}\Big(\sum_{i\ne k}\epsilon_i\varphi_i(t_i) - t_k\Big)\Big] \\ &= \mathbf{E}\sup_{t\in T}\Big(\sum_{i\ne k}\epsilon_i\varphi_i(t_i) + \epsilon_k t_k\Big),\end{aligned}$$

and by iterating one finds

$$\mathbf{E}\sup_{t\in T}\sum_{i=1}^m \epsilon_i\varphi_i(t_i) \le \mathbf{E}\sup_{t\in T}\sum_{i=1}^m \epsilon_i t_i.$$

If the functions φ_i's are L-Lipschitz, then clearly,

$$\mathbf{E}\sup_{t\in T}\sum_{i=1}^m \epsilon_i\varphi_i(t_i) = L\mathbf{E}\sup_{t\in T}\sum_{i=1}^m \epsilon_i\frac{\varphi_i(t_i)}{L} \le L\mathbf{E}\sup_{t\in T}\sum_{i=1}^m \epsilon_i t_i.$$

□

Given the contraction property for the Rademacher complexity of a set, we can derive the following result.

Proposition 1.1.4. Let $z \to \ell(z, y)$ be L_ℓ-Lipschitz for any $y \in \mathcal{Y}$. Then,

$$\mathbf{E}\sup_{x\in\mathcal{X}}(r(x) - R(x)) \le 2\mathbf{E}\sup_{x\in\mathcal{X}}\frac{1}{m}\sum_{i=1}^m \epsilon_i R_i(x) \le 2L_\ell\mathbf{E}\sup_{x\in\mathcal{X}}\frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i),$$

where $\epsilon_1, \dots, \epsilon_n$ are i.i.d. random variables uniform in $\{-1, 1\}$, independent of $(W_1, Y_1), \dots, (W_m, Y_m)$.

Proof. The proof uses the standard machinery of symmetrization. Let $Z_i = (W_i, Y_i)$ for any $i \in \{1, \dots, m\}$. Let $Z_1', \dots, Z_m'$ be an independent copy of the data $Z_1, \dots, Z_m$, and let $R_i'(x) := \ell(x^T\Phi(W_i'), Y_i')$ for each $i \in \{1, \dots, m\}$.

Using that $r(x) = \mathbf{E}R_i'(x) = \mathbf{E}[R_i'(x)|Z_1, \dots, Z_m]$, we have

$$\begin{aligned}
\mathbf{E} \sup_{x\in\mathcal{X}} \left(r(x) - \frac{1}{m}\sum_{i=1}^m R_i(x) \right) &= \mathbf{E} \sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m \mathbf{E}[R_i'(x) - R_i(x))|Z_1, \dots, Z_m] \\
&\leq \mathbf{E}\mathbf{E}\left[\sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m (R_i'(x) - R_i(x)) \middle| Z_1, \dots, Z_m \right] \\
&= \mathbf{E} \sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m (R_i'(x) - R_i(x)) \\
&= \mathbf{E} \sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m \epsilon_i (R_i'(x) - R_i(x)) \quad \text{by symmetrization} \\
&\leq 2\mathbf{E} \sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m \epsilon_i R_i(x) = 2\mathbf{E} \sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m \epsilon_i \ell(x^T\Phi(W_i), Y_i).
\end{aligned}$$

Conditioning on $Z_1, \dots, Z_m$ we can apply the contraction property for the Rademacher complexity of a set, applying Lemma 1.1.3 with the choice $\varphi_i(z) = \ell(z, Y_i)$ and $T = \{(x^T\Phi(W_1), \dots, x^T\Phi(W_m))^T : x \in \mathcal{X}\}$:

$$\mathbf{E}\left[\sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m \epsilon_i \ell(x^T\Phi(W_i), Y_i) \middle| Z_1, \dots, Z_m \right] \leq L_\ell \mathbf{E}\left[\sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i) \middle| Z_1, \dots, Z_m \right], \tag{1.4}$$

and the result follows by taking the expectation. □

Remark 1.1.1 (Emipirical Rademacher complexity)**.** *The quantity*

$$\mathbf{E} \sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i)$$

on the right hand side of the bound in Proposition 1.1.4 can be interpreted as the expected value of the Rademacher complexity of a random set. In fact, by conditioning on the data $W_1, \dots, W_m$ we get

$$\mathbf{E} \sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i) = \mathbf{E}\mathbf{E}\left[\sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i) \middle| W_1, \dots, W_m \right] = \mathbf{E}\mathbf{E}\left[\sup_{t\in T_{W_1,\dots,W_m}} \frac{1}{m}\sum_{i=1}^m \epsilon_i t_i \middle| W_1, \dots, W_m \right],$$

where $T_{W_1,\dots,W_m} := \{(x^T\Phi(W_1), \dots, x^T\Phi(W_m))^T : x \in \mathcal{X}\}$. The Rademacher complexity that we get when we condition on the data, as at the end of the proof of Proposition 1.1.4, is typically referred to as the empirical Rademacher complexity*.*

We now present a result to control the Rademacher complexity when we have boundedness assumptions with respect to the Euclidean norm.

Proposition 1.1.5. Let $\sup_{x\in\mathcal{X}} \|x\|_2 \leq B$ and $\|\Phi(W)\|_2 \leq G$ a.s.. Then,

$$\mathbf{E} \sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i) \leq \frac{BG}{\sqrt{m}}.$$

Proof. We have

$$\begin{aligned}
\mathbf{E}\sup_{x\in\mathcal{X}}\frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i) &\le \sup_{x\in\mathcal{X}}\|x\|_2\mathbf{E}\Big\|\frac{1}{m}\sum_{i=1}^m \epsilon_i\Phi(W_i)\Big\|_2 \quad \text{by Cauchy-Schwarz}\\
&\le B\mathbf{E}\sqrt{\Big\|\frac{1}{m}\sum_{i=1}^m \epsilon_i\Phi(W_i)\Big\|_2^2} \le B\sqrt{\mathbf{E}\Big\|\frac{1}{m}\sum_{i=1}^m \epsilon_i\Phi(W_i)\Big\|_2^2} \quad \text{by Jensen's inequality}\\
&= \frac{B}{m}\sqrt{\mathbf{E}\sum_{j=1}^n\Big(\sum_{i=1}^m \epsilon_i\Phi(W_i)_j\Big)^2} = \frac{B}{m}\sqrt{\mathbf{E}\sum_{j=1}^n\sum_{i=1}^m(\epsilon_i\Phi(W_i)_j)^2} \quad \text{by the independence of the } \epsilon\text{'s}\\
&= \frac{B}{m}\sqrt{\mathbf{E}\sum_{i=1}^m\|\Phi(W_i)\|_2^2} \le \frac{GB}{\sqrt{m}} \quad \text{as } \|\Phi(W)\|_2\le G \text{ a.s.}
\end{aligned}$$

□

Proposition 1.1.4 and Proposition 1.1.5 immediately yields the following result, from which Proposition 1.1.1 follows immediately.

Proposition 1.1.6. Let $z \to \ell(z,y)$ be L_ℓ-Lipschitz for any $y\in\mathcal{Y}$, $\sup_{x\in\mathcal{X}}\|x\|_2\le B$ and $\|\Phi(W)\|_2\le G$ a.s.. Then,

$$\mathbf{E}\sup_{x\in\mathcal{X}}(r(x)-R(x)) \le 2\frac{BGL_\ell}{\sqrt{m}}.$$

Bound with high probability

We prove Proposition 1.1.2. Let us consider the following quantity:

$$h(Z_1,\dots,Z_m) := \sup_{x\in\mathcal{X}}(r(x)-R(x)), \tag{1.5}$$

where we use the notation $Z_i := (W_i, Y_i)$. Proposition 1.1.6 yields a bound on $\mathbf{E}h(Z_1,\dots,Z_m)$. A classical way to derive bounds in high probability is to use concentration inequalities, which are tools in probability theory to bound the deviation of a function of many i.i.d. random variables from its mean. One of the classical concentration inequalities is the Bounded Difference, of which we now state a simple version.

Proposition 1.1.7 (Bounded Difference Inequality). Let $Z_1,\dots,Z_m\in\mathcal{Z}$ be m i.i.d. random variables, and let $h:\mathcal{Z}^m\to\mathbb{R}$ be a function satisfying, for every $k\in\{1,\dots,m\}$ and for all $z_1,\dots,z_m,z'_k\in\mathcal{Z}$:

$$|h(z_1,\dots,z_{k-1},z_k,z_{k+1},\dots,z_m) - h(z_1,\dots,z_{k-1},z'_k,z_{k+1},\dots,z_m)| \le c.$$

Then,

$$\mathbf{P}(h(Z_1,\dots,Z_m)\ge \mathbf{E}h(Z_1,\dots,Z_m)+t) \le \exp\Big(-\frac{2t^2}{mc^2}\Big),$$

or, equivalently, with probability at least $1-\delta$ we have

$$h(Z_1,\dots,Z_m) \le \mathbf{E}h(Z_1,\dots,Z_m) + c\sqrt{\frac{m}{2}\log\frac{1}{\delta}}.$$

To apply the Bounded Difference inequality to the function h defined in (1.5), we need to find c that satisfy the assumption of Proposition 1.1.7 to control the magnitude of the difference in the function h upon changing only one coordinate. The next proposition shows how to do this we have boundedness assumptions with respect to the Euclidean norm.

Proposition 1.1.8. Let h be the function defined in (1.5). Let $z \to \ell(z, y)$ be L_ℓ-Lipschitz for any $y \in \mathcal{Y}$, $\sup_{x\in\mathcal{X}} \|x\|_2 \le B$ and $\|\Phi(W)\|_2 \le G$ a.s.. Then, $c \in \mathbb{R}$ satisfying the requirement of the Bounded Difference inequality is given by

$$c = \frac{2}{m}\Big(\ell_0 + BGL_\ell\Big),$$

where $\ell_0 := \sup_{y\in\mathcal{Y}} |\ell(0, y)|$.

Proof. Fix $k \in \{1, \dots, m\}$ and let $z = (z_1, \dots, z_{k-1}, z_k, z_{k+1}, \dots, z_m)$ and $z' = (z_1, \dots, z_{k-1}, z'_k, z_{k+1}, \dots, z_m)$. Then,

$$|h(z) - h(z')| = \left|\sup_{x\in\mathcal{X}}\Big(r(x) - \frac{1}{m}\sum_{i=1}^m R_i^z(x)\Big) - \sup_{x\in\mathcal{X}}\Big(r(x) - \frac{1}{m}\sum_{i=1}^m R_i^{z'}(x)\Big)\right|,$$

where $R_i^z(x) := \ell(x^T\Phi(w_i), y_i)$. If $h(z) - h(z') \ge 0$ and we let $\tilde{x} \in \mathcal{X}$ be the maximizer of $\sup_{x\in\mathcal{X}}(r(x) - \frac{1}{m}\sum_{i=1}^m R_i^z(x))$ (note that the supremum is attained by the Extreme Value Theorem), we have

$$\begin{aligned} h(z) - h(z') &= \Big(r(\tilde{x}) - \frac{1}{m}\sum_{i=1}^m R_i^z(\tilde{x})\Big) - \sup_{x\in\mathcal{X}}\Big(r(x) - \frac{1}{m}\sum_{i=1}^m R_i^{z'}(x)\Big) \\ &\le \Big(r(\tilde{x}) - \frac{1}{m}\sum_{i=1}^m R_i^z(\tilde{x})\Big) - \Big(r(\tilde{x}) - \frac{1}{m}\sum_{i=1}^m R_i^{z'}(\tilde{x})\Big) \\ &= \frac{1}{m}(R_k^z(\tilde{x}) - R_k^{z'}(\tilde{x})) \le \frac{2}{m}\sup_{x\in\mathcal{X}, z\in\mathcal{Z}} |R_k^z(x)|. \end{aligned}$$

Proceeding analogously in the case $h(z) - h(z') \le 0$, we finally find

$$|h(z) - h(z')| \le \frac{2}{m}\sup_{x\in\mathcal{X}, z\in\mathcal{Z}} |R_k^z(x)| = \frac{2}{m}\sup_{x\in\mathcal{X}, w\in\mathcal{W}, y\in\mathcal{Y}} |\ell(x^T\Phi(w), y)|.$$

Using, that $z \to \ell(z, y)$ is L_ℓ-Lipschitz for any $y \in \mathcal{Y}$ and Cauchy-Schwarz we get

$$|\ell(x^T\Phi(w), y)| \le |\ell(0, y)| + |\ell(x^T\Phi(w), y) - \ell(0, y)| \le |\ell(0, y)| + L_\ell\|x^T\Phi(w)\|_2 \le |\ell(0, y)| + L_\ell\|x\|_2\|\Phi(w)\|_2.$$

As $\|\Phi(W)\|_2 \le G$ a.s. and $\sup_{x\in\mathcal{X}} \|x\|_2 \le B$ we have $|\ell(x^T\Phi(w), y)| \le |\ell(0, y)| + BGL_\ell$, so that

$$|h(z) - h(z')| \le \frac{2}{m}\Big(\sup_{y\in\mathcal{Y}} |\ell(0, y)| + BGL_\ell\Big).$$

□

The proof of Proposition 1.1.2 follows by putting everything together, namely, Proposition 1.1.7, Proposition 1.1.8, and Proposition 1.1.6, noting that the bounds in high probability that we have obtained hold *simultaneously* for $\sup_{x\in\mathcal{X}}(r(x) - R(x))$ and $\sup_{x\in\mathcal{X}}(R(x) - r(x))$. Here are the details.

Proof of Proposition 1.1.2. Using, respectively, Proposition 1.1.7 and Proposition 1.1.6, we have that with probability at least $1 - \delta$ the following holds:

$$\sup_{x\in\mathcal{X}}(r(x) - R(x)) \le \mathbf{E}\Big[\sup_{x\in\mathcal{X}}(r(x) - R(x))\Big] + c\sqrt{\frac{m}{2}\log\frac{1}{\delta}} \le \frac{2BGL_\ell}{\sqrt{m}} + c\sqrt{\frac{m}{2}\log\frac{1}{\delta}},$$

with $c = \frac{2}{m}(\ell_0 + BGL_\ell)$ by Proposition 1.1.8, which yields

$$\mathbf{P}\left(\sup_{x\in\mathcal{X}} (r(x) - R(x)) \le c' \right) \ge 1 - \delta,$$

where $c' := (\ell_0 + BGL_\ell)(2 + \sqrt{2\log(1/\delta)})/\sqrt{m}$. As the bounds we have derived holds also for $\sup_{x\in\mathcal{X}}(R(x) - r(x))$, we have

$$\mathbf{P}\left(\sup_{x\in\mathcal{X}} (r(x)-R(x)) + \sup_{x\in\mathcal{X}} (R(x)-r(x)) \le 2c' \right) \ge \mathbf{P}\left(\left\{ \sup_{x\in\mathcal{X}} (r(x)-R(x)) \le c' \right\} \bigcap \left\{ \sup_{x\in\mathcal{X}} (R(x)-r(x)) \le c' \right\} \right) \ge 1-\delta.$$

□

1.2 Overview of the reading group

Given the analysis of the previous section, we can now describe the plan for the reading group this term. While we motivated our interest in optimization techniques by looking at the classical setting of supervised machine learning and empirical risk minimization, only in Week 8 (hopefully!) we will see how to develop specialized algorithms to minimize R (that is, algorithms that can take advantage of the specific structure of R). In fact, most of our time will be devoted to reviewing classical algorithms to optimize a *general* convex function f over a convex set $\mathcal{X}$, namely, to solve:

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & x \in \mathcal{X}. \end{aligned}$$

Below is the current plan of what we aim to cover, along with references to the corresponding literature.

IMPORTANT NOTE FOR SPEAKERS.
The results that are quoted in bold are the ones we will be focusing on. When presenting your section and typing up notes, focus only on covering these results, along with introducing the quantities/proofs the are needed. You might comment on the other results if you have time, or just to give an overview of the bigger picture. In other words, instead of trying to cover as much as possible content-wise, try to cover as little as possible with as much explanation as possible (pictures, intuition, etc.).
As we move along in the reading group, the plan below might change. Please make sure that you have the latest version of these lecture notes before you prepare your section.

Week 2: Convexity. Black-box model. Projected gradient descent methods.

Readings: Parts of Chapter 1 and Chapter 3 in [13]. Details are below.

We introduce convexity, and some of its basic properties, such as the existence of subgradients (**Proposition 1.1 in [13]**) and the first order optimality condition (**Proposition 1.3 in [13]**).

We introduce the black-box model of computation, where we assume that the constraint set $\mathcal{X} \subseteq \mathbb{R}^n$ is known and the objective function f is unknown but can be accessed through queries to first order oracles: given $x \in \mathcal{X}$, a first order oracle yields back a subgradient of f at x.

We study how the different assumptions of L-Lipschitz, β-Smoothness, and α-Strong convexity, lead to different convergence rates for projected subgradient descent methods. Upon different conditions, different projected subgradient descent methods achieve the following rates:

	L-Lipschitz	β-smooth
Convex	$O(BL/\sqrt{t})$	$O(B^2\beta/t)$
α-strongly convex	$O(L^2/(\alpha t))$	$O(B^2 e^{-t\alpha/\beta})$

where B is the radius of the Euclidean ball that contains $\mathcal{X}$, namely, $\sup_{x\in\mathcal{X}} \|x\|_2 \le B$. As we discuss earlier, for applications in machine learning the assumption of α-strong convexity is typically too strong. This is why we only prove the rates for the convex case, both for L-Lipschitz and for β-Smooth. For completeness, here is where to find all the results mentioned above:

- Convex & L-Lipschitz: $O(BL/\sqrt{t})$ (**Theorem 3.2 in [13]**).
- Convex & β-Smoothness: $O(B^2\beta/t)$ (**Theorem 3.7 and in [13]**).
- α-Strong convexity & L-Lipschitz: $O(L^2/(\alpha t))$ (Theorem 3.9 and in [13]).
- α-Strong convexity & β-Smoothness: $O(B^2 \exp(-t\alpha/\beta))$ (Theorem 3.10 and in [13]).

These results are sometimes called *dimension-free*, since as presented the rates do not depend explicitly on the ambient dimension n. However, the dependence on the dimension enters implicit in the constants: note that B is the radius of the constraint set $\mathcal{X} \subseteq \mathbb{R}^n$, and that the parameters L, α, and β, are defined in terms of the Euclidean norm in $\mathbb{R}^n$. The terminology dimension-free is used in [13] to differentiate the rates achieved by subgradient descent methods from the rates obtained by ellipsoid methods (Chapter 2 in [13]), which we will not cover in this reading group.

We can also phrase these results in terms of *oracle complexity*, namely, the number of queries to the oracle that are *sufficient* to find an ϵ-approximate minima of a convex function:

	L-Lipschitz	β-smooth
Convex	$O(B^2L^2/\epsilon^2)$	$O(B^2\beta/\epsilon)$
α-strongly convex	$O(L^2/(\alpha\epsilon))$	$O((\beta/\alpha)\log(B^2/\epsilon))$

Remark 1.2.1. *To conclude, let us go back to our motivating example of minimizing the empirical risk function* $R(x) = \frac{1}{m}\sum_{i=1}^m \ell(x^T\Phi(W_i), Y_i)$. *Under the assumptions of Proposition 1.1.1, R is GL_ℓ-Lipschitz (see Section 1). Assuming that the loss function ℓ is convex, the subgradient descent method then yields:*

$$\mathit{OPTIMIZATION} := R(\hat{X}_t) - R(X^\star) \le \frac{BGL_\ell}{\sqrt{t}}.$$

Here the constants are explicit, and they match the ones in Proposition 1.1.1 and Proposition 1.1.2 for the STATISTICS term. In particular, following the rationale of optimizing the OPTIMIZATION *term in* (1.3) *up to the accuracy given by the* STATISTICS *term in* (1.3)*, one finds that it suffices to run the algorithm for $t \sim m$ number of steps.*

Week 3: Lower bounds for oracle complexity.

Readings: Parts of Chapter 3 in [13]. Details are below.

We see that within the first order oracle model one can prove lower-bounds for the amount of calls to the oracle that are *needed* to achieve a certain accuracy ϵ. In the non-smooth case we show that the rates achieved by subgradient descent methods, i.e., $O(BL/\sqrt{t})$ for convex functions and $O(L^2/(\alpha t))$ for α-strongly convex functions, can not be improved (**Theorem 3.13 in [13]; we will only cover the proof of the first statement for convex L-Lipschitz functions**). On the other hand, in the smooth case we prove oracle-complexity lower bounds that are better than the ones achieved by subgradient methods, namely, $O(D^2\beta/t^2)$ for smooth functions (**Theorem 3.14 in [13]**) and $O(D^2\exp(-t\sqrt{\alpha/\beta}))$ for smooth and strongly convex functions (Theorem 3.15 in [13]), where D is the diameter of $\mathcal{X}$, i.e., $D := \max_{x,y\in\mathcal{X}} \|x-y\|_2$. To recap, the optimal rates are:

	L-Lipschitz	β-smooth
Convex	$O(BL/\sqrt{t})$	$O(D^2\beta/t^2)$
α-strongly convex	$O(L^2/(\alpha t))$	$O(D^2 e^{-t\sqrt{\alpha/\beta}})$

Week 4: Application: Boosting.

Reading: Parts of Part II, Section 1.4 and Part II, Section 2.2 in [21].

We apply the projected subgradient descent algorithm to solve an important example in machine learning: Boosting. For a given loss function φ (recall, $\ell(z,y)=\varphi(-zy)$ in classification), Boosting can be written as the problem:

$$\begin{aligned}\text{minimize}\quad & r(x)=\mathbf{E}\varphi(-Yx^T\Phi(W))\\ \text{subject to}\quad & x\in\Delta_n,\end{aligned}$$

where $\Delta_n := \{x\in[0,1]^n : \sum_{k=1}^n x_k = 1\}$ is the n-dimensional probability simplex. Here the feature map $\Phi:\mathcal{W}\to\{-1,1\}^n$ encodes the prediction of the n base classifiers on the given data point it is applied to, and $x^T\Phi$ is a convex combination of these classifiers. Following the error decomposition given in Proposition 1.0.2, we show that in the case of Boosting the *STATISTICS* term is $O(\sqrt{\log n/m})$, which only grows logarithmically with the number of base classifiers n. This is a nice feature in applications where the number n of base classifiers is very large, possibly exponential. Hence, one would hope to be able to design an algorithmic procedure that only uses $\log n$ calls to the first order oracle in order to match the accuracy of the *STATISTICS* term to find an approximate solution to the empirical risk minimization problem:

$$\begin{aligned}\text{minimize}\quad & R(x)=\frac{1}{m}\sum_{i=1}^m\varphi(-Y_ix^T\Phi(W_i))\\ \text{subject to}\quad & x\in\Delta_m.\end{aligned}$$

Note that if φ is L_φ-Lipschitz on $[-1,1]$, then the empirical risk R is $L_\varphi\sqrt{n}$-Lipschitz on $[-1,1]$:

$$\begin{aligned}|R(x)-R(y)| &\le \frac{1}{m}\sum_{i=1}^m|\varphi(-Y_ix^T\Phi(W_i))-\varphi(-Y_iy^T\Phi(W_i))| \le \frac{1}{m}\sum_{i=1}^m L_\varphi\|Y_i(x-y)^T\Phi(W_i)\|_2\\ &\le L_\varphi\|\Phi(W_i)\|_2\|x-y\|_2 \le \sqrt{n}L_\varphi\|x-y\|_2.\end{aligned}$$

Hence, projected subgradient descent yields a rate $O(L_\varphi\sqrt{n/t})$. Imposing the condition $\sqrt{n/t}\lesssim\sqrt{\log n/m}$, we find that $t\gtrsim nm/\log n$, which does *not* scale logarithmically with n as we hoped for!

Boosting is one example where one would like to apply subgradient descent methods on a non-Euclidean space: the probability simplex Δ_n. However, subgradient descent methods are only designed for the Euclidean geometry. Note, in fact, that all the definitions we gave for L-Lipschitz, β-smoothness, and α-strong convexity, as well as the bounds for the constraint set $\mathcal{X}$, are expressed in terms of the Euclidean norm $\|\cdot\|_2$. This motivates the design of a new class of algorithms (in fact, a generalization of subgradient descent) that can adapt to the geometry of the problem at hand, as we will see next.

Week 5: Non-Euclidean setting: mirror descent.

Readings: Section 4.1, 4.2, and 4.3 in [13]. Details are below.

We introduce the mirror descent algorithm and show that this algorithm adapts also to non-Euclidean geometries. We say that f is L-Lipschitz in some norm $\|\cdot\|$ if $|f(x)-f(y)|\le L\|x-y\|$ for any $x,y\in\mathbb{R}^n$. We show that for such convex functions mirror descent yields rates that scale like $O(LB/\sqrt{t})$ (**Theorem 4.2 in [13]**). This allows mirror descent to achieve a rate $O(\sqrt{\log n/t})$ in the example of boosting, where one wants to minimize a function with subgradient bounded in the ℓ_∞-norm over the probability simplex (**Section 4.3 in [13]**), in contrast to the rate $O(\sqrt{n/t})$ achieved by subgradient descent, as shown last time. We also show that in the case of the Euclidean norm, the algorithm is equivalent to projected subgradient descent.

Week 6: Acceleration by coupling gradient descent and mirror descent.

Readings: [4].

We show that mirror descent can be coupled with gradient descent to yield an *accelerated* algorithm that can achieve the lower bound for smooth functions that we proved in Week 3. The acceleration technique is linear coupling, from the paper [4].

Week 7: Non-Euclidean setting: Frank-Wolfe.

Readings: Section 3.3 in [13]. Details are below.

In many applications, the computational bottleneck in gradient descent methods is given by the projection step on the constraint set $\mathcal{X}$. To address this issue, we study the Frank-Wolfe algorithm (a.k.a. conditional gradient descent), which replaces the projection step with a linear optimization over $\mathcal{X}$, which in some cases can be a much simpler problem. For smooth objective functions, the Frank-Wolfe algorithm achieves rate $O(B^2\beta/t)$ (**Theorem 3.8 in [13]**). While this rate is slow and it does not match the oracle complexity lower bound $O(D^2\beta/t^2)$, the saving in the computational complexity makes this algorithm convenient in some applications. Other advantages of this algorithm over gradient descent methods is that this algorithm can apply to smoothness in any norm (f is β-smooth in some norm $\|\cdot\|$ if $\|\nabla f(x) - \nabla f(y)\|_* \leq \beta\|x-y\|$ for any $x, y \in \mathbb{R}^n$, where the dual norm $\|\cdot\|_*$ is defined as $\|g\|_* = \sup_{x\in\mathbb{R}^n:\|x\|\leq 1} g^T x$), and that the algorithm computes sparse iterates. We will discuss a concrete application where these benefits are substantials, the least square regression with structured sparsity (**in Section 3.3 in [13]**).

Week 8: Stochastic oracle model.

Readings: Section 6.1, 6.2, and 6.3 in [13].

The first order oracle model that we have investigated so far had allowed us to produce a complete theory of convex optimization, in the sense that for various classes of convex functions we can design algorithms with an *oracle complexity* that matches the lower bounds. However, the black-box model does not tell us anything about the *computational complexity*, namely, the number of elementary computations that an algorithm needs to do to solve the problem (indeed, to address this question we should "open" the black-box). Going back to the original problem in machine learning that we set to solve, as described in Section 1, we note that the first order oracle model is too general for our needs, and that there might be computational savings to be gained in working with a model of computation that is more fined-tuned to our problem. This consideration motivates us to consider the *stochastic* first order oracle model, where for any point $x \in \mathcal{X}$ the oracle gives back an unbiased estimator of the gradient at x. Within this framework, two approaches have attracted a lot of attention lately, due to the computational savings they yield.

Multiple passes over the data. Within the setting of empirical risk minimization, one notices that we know the *global* structure of the function we want to minimize, namely, $R = \frac{1}{m}\sum_{i=1}^m R_i$. Hence, for instance, given $x \in \mathcal{X}$ we can have access to $\nabla R_i(x)$ for a specific $i \in \{1, \dots, m\}$, which is something that is not allowed in the oracle model previously discussed where only access to $\nabla R(x)$ is granted. Note that computing $\nabla R(x)$ costs $O(m)$ operations as R is the sum of m terms, while computing $\nabla R_i(x)$ costs $O(1)$.

Single pass over the data. The empirical risk minimization approach described in Section 1 has allowed us to break the original problem we care about, namely, problem (1.1), into a *STATISTICS* component and an *OPTIMIZATION* component, see Proposition 1.0.2. We show that within the stochastic oracle model there is a way to directly address problem (1.1) and yield a bound for $r(\hat{X}_t) - r(x^\star)$, *combining* both statistics and optimization. Recall that we do not know the function r as we assume that we do not know the distribution of the random variables W and Y; as a consequence, we also do not know the gradient of r, so we can not minimize r within the classical first order oracle model. On the other hand, we can treat the $\nabla R_i(x)$'s as unbiased estimators of ∇r.

Chapter 2

Convexity. Black-box model. Projected gradient descent methods.

Speaker: Anthony Caterini, 19/10/2017.

We want to focus first on introducing the notion of convexity, which will admit some very nice properties for optimization. In particular, convex functions always have subgradients, and any local minimum is also a global minimum. We will also introduce projected subgradient descent methods and prove convergence rates in the cases where f is convex and either L-Lipschitz continuous or β-smooth. In these two cases, the essence of the method is the same, although the particulars are slightly different.

The methods in this section relate to controlling the *OPTIMIZATION* term in empirical risk minimization. We note that although the empirical risk R may not be convex, we can still run the algorithms that we discuss to find local optima.

Note that in this section, we rely heavily on [13], although the extension of projected gradient descent to time-varying step sizes comes from [21, Lecture 11, Page 5].

2.1 Convexity

We begin by recalling the definition of a convex function and convex set.

Definition 2.1.1 (Convex function and set)**.** *A function $f : \mathcal{X} \subset \mathbb{R}^n \to \mathbb{R}$ is* convex *if, for all $(x, y, \gamma) \in \mathcal{X} \times \mathcal{X} \times [0, 1]$,*

$$f((1-\gamma)x + \gamma y) \leq (1-\gamma)f(x) + \gamma f(y).$$

In other words, f always lies below its chords. Similarly, a set *$\mathcal{X} \subset \mathbb{R}^n$ is* convex *if, for all $(x, y, \gamma) \in \mathcal{X} \times \mathcal{X} \times [0, 1]$,*

$$(1-\gamma)x + \gamma y \in \mathcal{X}.$$

In other words, $\mathcal{X}$ contains all of its line segments.

Existence of subgradients

We will also define the notion of a subgradient – useful when f is not differentiable – and show that a convex function always admits subgradients. This will be important when we introduce the projected subgradient method, as we will not assume differentiability of f. We will instead assume that f is convex, and that we have access to a *first order oracle* that can give us a subgradient of f at each point.

Definition 2.1.2 (Subgradient)**.** *Let $\mathcal{X} \subset \mathbb{R}^n$ and $f : \mathcal{X} \to \mathbb{R}$. Then, $g \in \mathbb{R}^n$ is a* subgradient *of f at $x \in \mathcal{X}$ if, for any $y \in \mathcal{X}$, one has*

$$f(x) - f(y) \leq g^T(x - y).$$

The set of subgradients of f at x is denoted $\partial f(x)$.

Note that we can rewrite the above inequality as $f(y) \geq f(x) + g^T(y - x)$. Thus, each subgradient defines a plane that supports f. Also notice the set of subgradients at a point at which f is non-differentiable can be infinite. Conversely, we will see that, for convex and differentiable functions, the standard gradient is also a subgradient. We first mention the supporting hyperplane theorem and the definition of the epigraph before showing this.

Theorem 2.1.1 (Supporting Hyperplane Theorem)**.** *Let $\mathcal{X}$ be a convex set, and $x_0 \in \partial\mathcal{X}$ (the boundary of $\mathcal{X}$). Then, there exists $w \in \mathbb{R}^n$, $w \neq 0$, such that*

$$\forall x \in \mathcal{X}, w^T x \geq w^T x_0.$$

Definition 2.1.3 (Epigraph)**.** *The* epigraph *of a function $f : \mathcal{X} \to \mathbb{R}$ is*

$$epi(f) = \{(x, t) \in \mathcal{X} \times \mathbb{R} : t \geq f(x)\}.$$

Also, a function is convex if and only if its epigraph is convex.

Proposition 2.1.2 (Existence of subgradients). Let $\mathcal{X}$ be convex and $f : \mathcal{X} \to \mathbb{R}$. If $\forall x \in \mathcal{X}, \partial f(x) \neq \emptyset$, then f is convex. Conversely, if f is convex, then for any $x \in \text{int}(\mathcal{X}), \partial f(x) \neq \emptyset$. Furthermore, if f is convex and differentiable at x, then $\nabla f(x) \in \partial f(x)$.

Proof. For the first claim, let $g \in \partial f((1-\gamma)x + \gamma y)$, for any $x, y \in \mathcal{X}$ and $\gamma \in [0, 1]$. Then,

$$f((1-\gamma)x + \gamma y) \leq f(x) + \gamma g^T(y - x), \tag{2.1}$$
$$f((1-\gamma)x + \gamma y) \leq f(y) + (1-\gamma) g^T(x - y). \tag{2.2}$$

Then, adding $(1-\gamma)*(2.1)$ with $\gamma*(2.2)$ clearly shows that f is a convex function.

Now let us suppose that f is convex. Let $x \in \mathcal{X}$. Clearly, $(x, f(x)) \in \partial\text{epi}(f)$, and $\text{epi}(f)$ is a convex set. Thus, by the Supporting Hyperplane Theorem, there exists $(a, b) \in \mathbb{R}^n \times \mathbb{R}$, $(a, b) \neq 0$, such that for all $(y, t) \in \text{epi}(f)$,

$$a^T x + b f(x) \geq a^T y + bt. \tag{2.3}$$

By allowing $t \to \infty$, we can see that $b \leq 0$. Also, if we assume $x \in \text{int}(\mathcal{X})$, then $y = x + \varepsilon a \in \mathcal{X}$ for small $\varepsilon > 0$. Plugging this into (2.3), we see that $b = 0$ implies $a = 0$, which is a contradiction; therefore, $b < 0$. We can now rewrite (2.3) with $t = f(y)$ as

$$f(x) - f(y) \leq \frac{1}{|b|} a^T(x - y),$$

i.e. $a/|b|$ is a subgradient of f.

Finally, if f is convex and differentiable, it is not hard to show that $\nabla f(x) \in \partial f(x)$ by definition. $\square$

First order optimality condition

We also have a nice first order optimality condition when dealing with convex functions.

Proposition 2.1.3 (First order optimality condition). Let f be convex, and $\mathcal{X}$ be a closed set on which f is differentiable. Then,

$$x^* \in \arg\min_{x \in \mathcal{X}} f(x)$$

if and only if for all $y \in \mathcal{X}$,

$$\nabla f(x^*)^T (x^* - y) \leq 0.$$

Proof. If we first assume that $\nabla f(x^*)^T (x^* - y) \leq 0$ for all $y \in \mathcal{X}$, then since the gradient is a subgradient,

$$f(x^*) - f(y) \leq \nabla f(x^*)^T (x^* - y) \leq 0,$$

i.e. $f(x^*) \leq f(y)$ for all $y \in \mathcal{X}$.

Now, if we assume $x^* \in \arg\min_{x \in X} f(x)$, we know that f is locally non-decreasing around x^*. Define $h(t) = f(x^* + t(y - x^*))$ for all $y \in \mathcal{X}$ – the rate of change of f along the line from x to y. We require $h'(0) \geq 0$ since x^* is a minimizer. Thus,

$$h'(0) = \nabla f(x^*)^T (y - x^*) \geq 0,$$

i.e. $\nabla f(x^*)^T (x^* - y) \leq 0$ for all $y \in \mathcal{X}$. □

We will see that the above proposition proves to be very useful when proving the convergence rates of projected gradient descent methods. Of course the above proposition holds with equality when x^* is an interior point, as $\nabla f(x^*) = 0$ in this case. However, it may also be the case that x^* is on the boundary of $\mathcal{X}$ and not necessarily a minimum in the space in which $\mathcal{X}$ is embedded, and this is the less-intuitive case that Proposition 2.1.3 handles well.

2.2 Black-box model

In the black-box model of computation, we assume that we know the constraint set $\mathcal{X}$ but not the objective function $f : \mathcal{X} \to \mathbb{R}$. Nonetheless, we assume that we can make queries to an *oracle* and receive some information about f as output. Of particular interest to us is a **first order oracle**, which takes a point $x \in \mathcal{X}$ as input and outputs a subgradient of f at x. Of course, if f is convex, we will always be able to find a subgradient. We are interested in understanding the *oracle complexity* of convex optimization – the number of necessary and sufficient queries to the oracle to find an ε-approximate minima.

2.3 Projected gradient descent methods

We will now move into developing projected gradient descent algorithms for constrained optimization problems of the form

$$\min_{x \in \mathcal{X}} f(x),$$

where $f : \mathcal{X} \to \mathbb{R}$, and $\mathcal{X} \subset \mathbb{R}^n$ is the constraint set. If f is differentiable and $\mathcal{X} = \mathbb{R}^n$, we can write out the basic gradient descent method as

$$x_{t+1} = x_t - \eta \nabla f(x_t),$$

for some initial point $x_1 \in \mathbb{R}^n$ and step size $\eta > 0$. Methods of this type can obtain an *oracle complexity* that is independent of the dimension and are thus attractive in the high-dimensional setting.

However, we are interested in cases where the constraint set is a strict subset of $\mathbb{R}^n$ and f may not be differentiable. We thus turn to a *projected subgradient descent* method instead, where each iteration performs the following: take a step in the direction of a subgradient of f, and then project this new point back onto the constraint set $\mathcal{X}$. We therefore begin by defining the projection operator and describe one of its properties.

Note that throughout this section, we will assume that the constraint set $\mathcal{X}$ is compact and convex, and is contained in a Euclidean ball of radius B centred at $x_1 \in \mathcal{X}$. Furthermore, $\|\cdot\|$ denotes the Euclidean norm.

Definition 2.3.1 (Projection operator). *For all $y \in \mathbb{R}^n$, the* projection operator *on $\mathcal{X}$, $\Pi_{\mathcal{X}}$, is defined by*

$$\Pi_{\mathcal{X}}(y) = \arg\min_{x \in \mathcal{X}} \|x - y\|.$$

Lemma 2.3.1. Let $x \in \mathcal{X}$ and $y \in \mathbb{R}^n$. Then,

$$(\Pi_{\mathcal{X}}(y) - x)^T (\Pi_{\mathcal{X}}(y) - y) \leq 0,$$

which also implies $\|\Pi_{\mathcal{X}}(y) - x\|^2 + \|y - \Pi_{\mathcal{X}}(y)\|^2 \leq \|y - x\|^2$.

Proof. This is a direct consequence of Proposition 2.1.3 since $\Pi_{\mathcal{X}}(y)$ is a minimizer of $h_y(z) = \|y - z\|$, and $\nabla h_y(z) = (z - y)/\|z - y\|$. □

L-Lipschitz functions

We now introduce projected subgradient descent for the case where the subgradients of $f(x)$ satisfy $\|g\| \leq L$, for all $x \in \mathcal{X}$. Note that this immediately implies f is L-Lipschitz continuous. The algorithm consists of two update steps at every iteration $t \geq 1$:

$$\begin{aligned} y_{t+1} &= x_t - \eta_t g_t, \text{where } g_t \in \partial f(x_t), \\ x_{t+1} &= \Pi_{\mathcal{X}}(y_{t+1}). \end{aligned}$$

We state and prove the convergence of this method in the L-Lipschitz case below.

Theorem 2.3.2 (L-Lipschitz Projected Subgradient Descent). *Suppose $\mathcal{X}$ is contained in a Euclidean ball of radius B centred at x_1. Suppose also that for every $x \in \mathcal{X}$, the subgradients $g \in \partial f(x)$ satisfy $\|g\| \leq L$ and that f is convex. Then, the projected subgradient descent method with $\eta_s \equiv \eta = \frac{B}{L\sqrt{t}}$ satisfies*

$$f\left(\frac{1}{t}\sum_{s=1}^{t} x_s\right) - f(x^*) \leq \frac{LB}{\sqrt{t}} \quad \textit{and} \quad f(x^o) - f(x^*) \leq \frac{LB}{\sqrt{t}}, \tag{2.4}$$

where $x^o \in \arg\min_{x \in \{x_1,\ldots,x_t\}} f(x)$. Moreover, with $\eta_s = \frac{B}{L\sqrt{s}}$, then $\exists c > 0$ (for instance, $c = 2(1 + \log 2)$) such that

$$f\left(\left(\sum_{s=\lceil t/2\rceil+1}^{t} \eta_s\right)^{-1} \sum_{s=\lceil t/2\rceil+1}^{t} \eta_s x_s\right) - f(x^*) \leq c\frac{LB}{\sqrt{t}} \quad \textit{and} \quad f(x^o) - f(x^*) \leq c\frac{LB}{\sqrt{t}}. \tag{2.5}$$

Proof. Recall that $2a^Tb = \|a\|^2 + \|b\|^2 - \|a-b\|^2$. Then, for any $1 \le s \le t$,

$$\begin{aligned} f(x_s) - f(x^*) &\le g_s^T(x_s - x^*) \\ &= \frac{1}{\eta}(x_s - y_{s+1})^T(x_s - x^*) \\ &= \frac{1}{2\eta}\left(\|x_s - x^*\|^2 + \|x_s - y_{s+1}\|^2 - \|y_{s+1} - x^*\|^2\right) \\ &= \frac{1}{2\eta}\left(\|x_s - x^*\|^2 - \|y_{s+1} - x^*\|^2\right) + \frac{\eta}{2}\|g_s\|^2. \end{aligned}$$

Now, from Lemma 2.3.1, we know that $\|y_{s+1} - x^*\| \ge \|x_{s+1} - x^*\|$. Therefore, summing from $s = 1$ to t, we get

$$\frac{1}{t}\sum_{s=1}^{t}(f(x_s) - f(x^*)) \le \frac{1}{2\eta t}\left(\|x_1 - x^*\|^2 - \|x_{t+1} - x^*\|^2\right) + \frac{\eta}{2}L^2 \le \frac{B^2}{2\eta t} + \frac{\eta L^2}{2}.$$

Selecting $\eta = \frac{B}{L\sqrt{t}}$ to minimize the right-hand side of the above inequality gives the first result in (2.4) (since $f\left(\frac{1}{t}\sum_{s=1}^{t} x_s\right) \le \frac{1}{t}\sum_{s=1}^{t} f(x_s)$ by Jensen's inequality). Clearly, $f(x^o) \le \frac{1}{t}\sum_{s=1}^{t} f(x_s)$, proving the second result of (2.4).

Now consider the case of an adaptive step size η_s. The above derivation yields

$$f(x_s) - f(x^*) \le \frac{1}{2\eta_s}\left(\|x_s - x^*\|^2 - \|y_{s+1} - x^*\|^2\right) + \frac{\eta_s}{2}\|g_s\|^2.$$

If we want to apply the same argument as above and get a telescoping sum with cancellations, we need to take a weighted sum weighted by η_s. Namely, replacing $t^{-1}\sum_{s=1}^{t}$ with $(\sum_{s=1}^{t}\eta_s)^{-1}\sum_{s=1}^{t}\eta_s$ we get

$$\left(\sum_{s=1}^{t}\eta_s\right)^{-1}\sum_{s=1}^{t}\eta_s\left(f(x_s) - f(x^*)\right) \le \left(\sum_{s=1}^{t}\eta_s\right)^{-1}\left(\frac{B^2}{2} + \left(\sum_{s=1}^{t}\eta_s^2\right)\frac{L^2}{2}\right),$$

which reduces to the previous result when $\eta_s \equiv \eta$. We want the right-hand-side to go to zero as t increases, so we need both $\sum \eta_s \to \infty$ and $\frac{\sum \eta_s^2}{\sum \eta_s} \to 0$. This is the case if we take $\eta_s = \frac{K}{\sqrt{s}}$, as $\sum_{s=1}^{t}\eta_s \ge c_1 K\sqrt{t}$ (e.g., $c_1 = 1$) and $\sum_{s=1}^{t}\eta_s^2 \le c_2 K^2 \log t$ (e.g., $c_2 = 1 + 1/\log 2$ if $t \ge 2$, using that $\sum_{s=1}^{t} 1/s \le 1 + \int_{s=0}^{t}\frac{1}{s}ds = 1 + \log t \le c_2 \log t$). We can then choose $K = \frac{B}{L}$ such that

$$\left(\sum_{s=1}^{t}\eta_s\right)^{-1}\sum_{s=1}^{t}\eta_s\left(f(x_s) - f(x^*)\right) \le \frac{B^2}{2c_1 K\sqrt{t}} + \frac{c_2 K L^2 \log t}{2c_1\sqrt{t}} \le \left(\frac{1+c_2}{2c_1}\right)LB\sqrt{\frac{\log t}{t}}.$$

To get rid of the log term, we note that if we only sum from $\lceil t/2\rceil + 1$ to t, for $t \ge 3$, we have $\sum_{s=\lceil t/2\rceil+1}^{t}\eta_s \ge c_1' K\sqrt{t}$ (e.g., $c_1' = 1/4$, as $\sum_{s=\lceil t/2\rceil+1}^{t} 1/\sqrt{s} \ge \frac{t-1}{2\sqrt{t}} = \frac{\sqrt{t}}{2}(1 - 1/t) \ge \frac{\sqrt{t}}{4}$) and $\sum_{s=\lceil t/2\rceil+1}^{t}\eta_s^2 \le c_2' K^2$ (e.g., $c_2' = \log 2$ using that $\sum_{s=\lceil t/2\rceil+1}^{t}\frac{1}{s} \le \int_{t/2}^{t}\frac{1}{s}ds = \log 2$). Therefore, we finally have that

$$\min_{1\le s\le t} f(x_s) - f(x^*) \le \min_{\lceil t/2\rceil+1\le s\le t} f(x_s) - f(x^*) \le \left(\sum_{s=\lceil t/2\rceil+1}^{t}\eta_s\right)^{-1}\sum_{s=\lceil t/2\rceil+1}^{t}\eta_s\left(f(x_s) - f(x^*)\right) \le c\frac{LB}{\sqrt{t}},$$

with $c = \left(\frac{1+c_2'}{2c_1'}\right)$, where we used that the minimum element of a set is always less or equal to a convex combination of the elements. The proof of both results in (2.5) follows again by Jensen's inequality. $\square$

The result of (2.5) is promising since we would rather use an adaptive step size than one that depends on the total number of iterations t, to be set and fixed in advance, i.e., prior to running the algorithm. Unfortunately, both step sizes still depend on B and L – quantities which may not be known. The results in Theorem 2.3.2 demonstrate that the projected subgradient method for convex and L-Lipschitz functions exhibits a convergence rate of $O\left(\frac{BL}{\sqrt{t}}\right)$. We can also phrase this in terms of *oracle complexity*: as at each iteration we make a constant (in fact, one) number of calls to the oracle, then to achieve ε-convergence, we require $\frac{BL}{\sqrt{t}} \leq \varepsilon$, or $t \geq \frac{B^2L^2}{\varepsilon^2}$.

β-smooth functions

Now, we move to the case of a β-smooth convex function. Recall a function $f : \mathcal{X} \to \mathbb{R}$ is β-smooth if, for all $x, y \in \mathcal{X}$, we have $\|\nabla f(x) - \nabla f(y)\| \leq \beta\|x - y\|$. We show some auxiliary results first for convex and β-smooth functions – deferring to [13] for the proofs – before deriving the convergence rate of projected subgradient descent with $\eta = \frac{1}{\beta}$.

Lemma 2.3.3. Let $f : \mathcal{X} \to \mathbb{R}$ be a convex and β-smooth function. Then, for all $x, y \in \mathcal{X}$, we have

$$f(x) - f(y) - \nabla f(y)^T(x - y) \leq \frac{\beta}{2}\|x - y\|^2.$$

Proof. Refer to the proof in [13, Lemma 3.4]. □

Lemma 2.3.4. Let $x, y \in \mathcal{X}$, $x^+ = \Pi_{\mathcal{X}}\left(x - \frac{1}{\beta}\nabla f(x)\right)$, and $g_{\mathcal{X}}(x) = \beta(x - x^+)$. Then, we have the following:

$$f(x^+) - f(y) \leq g_{\mathcal{X}}(x)^T(x - y) - \frac{1}{2\beta}\|g_{\mathcal{X}}(x)\|^2.$$

Proof. Refer to the proof in [13, Lemma 3.6]. □

Theorem 2.3.5. *[β-Smooth Projected Subgradient Descent] Let $f : \mathcal{X} \to \mathbb{R}$ be convex and β-smooth on $\mathcal{X}$. Also, suppose $\mathcal{X}$ is contained in a Euclidean ball of radius B centred at x_1. Then, the projected subgradient descent method with $\eta = \frac{1}{\beta}$ satisfies*

$$f(x_t) - f(x^*) \leq \frac{3\beta B^2 + f(x_1) - f(x^*)}{t}. \tag{2.6}$$

Proof. From Lemma 2.3.4 and since $x_{s+1} = \Pi_{\mathcal{X}}(x_s - \frac{1}{\beta}\nabla f(x_s))$, we have the following:

$$\begin{aligned} f(x_{s+1}) - f(x_s) &\leq g_{\mathcal{X}}(x_s)^T(x_s - x_s) - \frac{1}{2\beta}\|g_{\mathcal{X}}(x_s)\|^2 = -\frac{1}{2\beta}\|g_{\mathcal{X}}(x_s)\|^2, \text{ and} \\ f(x_{s+1}) - f(x^*) &\leq g_{\mathcal{X}}(x_s)^T(x - x^*) \leq \|g_{\mathcal{X}}(x_s)\| \cdot \|x_s - x^*\|, \end{aligned}$$

since $f(x_{s+1}) \geq f(x^*)$. Also, we can show that $\|x_s - x^*\|$ is decreasing with s: from Lemma 2.3.4, we have $g_{\mathcal{X}}(x_s)^T(x_s - x^*) \geq \frac{1}{2\beta}\|g_{\mathcal{X}}(x_s)\|$, and thus

$$\begin{aligned} \|x_{s+1} - x^*\|^2 &= \|x_s - \frac{1}{\beta}g_{\mathcal{X}}(x_s) - x^*\|^2 \\ &= \|x_s - x^*\|^2 - \frac{2}{\beta}g_{\mathcal{X}}(x_s)^T(x_s - x^*) + \frac{1}{\beta^2}\|g_{\mathcal{X}}(x_s)\|^2 \leq \|x_s - x^*\|^2. \end{aligned}$$

We can therefore bound the difference $f(x_{s+1}) - f(x^*)$ in terms of the difference at the previous iterate:

$$\begin{aligned} f(x_{s+1}) - f(x^*) &= (f(x_{s+1}) - f(x_s)) + (f(x_s) - f(x^*)) \\ &\leq f(x_s) - f(x^*) - \frac{1}{2\beta}\|g_{\mathcal{X}}(x_s)\|^2 \\ &\leq f(x_s) - f(x^*) - \frac{\|g_{\mathcal{X}}(x_s)\|^2 \|x_s - x^*\|^2}{2\beta\|x_1 - x^*\|^2} \\ &\leq f(x_s) - f(x^*) - \frac{(f(x_{s+1}) - f(x^*))^2}{2\beta\|x_1 - x^*\|^2}. \end{aligned}$$

Thus, we can prove (2.6) using induction. It is trivial to show the base case $t = 1$. Then, assuming it is true for $t = s$,

$$\begin{aligned} f(x_{s+1}) - f(x^*) &\leq f(x_s) - f(x^*) - \frac{(f(x_{s+1}) - f(x^*))^2}{2\beta\|x_1 - x^*\|^2} \\ &\leq \frac{3\beta B^2 + f(x_1) - f(x^*)}{s}, \text{ by the inductive hypothesis} \\ &\leq \frac{3\beta B^2 + f(x_1) - f(x^*)}{s+1}. \end{aligned}$$

Therefore, (2.6) is true for all $t \in \mathbb{N}$. □

The above theorem demonstrates that for β-smooth convex functions, the error in estimating the optimal function value $O\left(\frac{\beta B^2}{t}\right)$, corresponding to an oracle complexity of $O\left(\frac{\beta B^2}{\varepsilon}\right)$. We notice that the error rate decreasing faster in t in this case than in the Lipschitz case, but the dependence on B^2 instead of B is more loose, and we will see in section 2.4 that we cannot compare β and L directly. We also note here that the step size η depends on β, which can be difficult to find in practice. As in the Lipschitz case, we again have no explicit reference to the underlying dimension n, although both B and β implicitly depend on n.

2.4 Remark on convexity, strong convexity, smoothness, and Lipschitz continuity

Finally, having shown convergence results for convex β-smooth and L-Lipschitz functions, it is important to note the geometric connection between these types of functions and convex functions. Recall the Taylor series expansion of $f(y)$ around x:

$$f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(x)(y - x) + \cdots$$

Convex functions are uniformly bounded *below* by an hyperplane that can be constructed at any point x:

$$f(y) \geq f(x) + \nabla f(x)^T (y - x),$$

and α-strongly convex functions are uniformly bounded *below* by this hyperplane and a quadratic:

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{\alpha}{2}\|y - x\|^2.$$

On the other hand, β-smooth functions are uniformly bounded *above* by this hyperplane and a quadratic:

$$f(y) \leq f(x) + \nabla f(x)^T (y - x) + \frac{\beta}{2}\|y - x\|^2.$$

Finally, L-Lipschitz functions are uniformly bounded *above and below* by a double cone:

$$f(x) - L\|y - x\| \leq f(y) \leq f(x) + L\|y - x\|.$$

Another important point to note is the relative independence of β-smoothness and Lipschitz continuity. Consider the case where $f(x) = x^2$ – clearly, this is β-smooth for $\beta = 2$ (and α-strongly convex with $\alpha = 2$). However, it is not Lipschitz continuous, as we cannot hope to uniformly upper-bound a quadratic function by a line. On the other hand, consider the hinge function $f(x) = \max(0, 1 + x)$. It is easy to see that this is Lipschitz continuous with $L = 1$, but not β-smooth for any β. Thus, neither condition implies the other.

Chapter 3

Lower bounds for oracle complexity

Speaker: Dominic Richards, 26/10/2017.

3.1 Week 2 Recap

Recall the objective is to minimise some convex function $f : \mathcal{X} \to \mathbb{R}$, being phrased as the following optimisation problem

$$\min f(x) \tag{3.1}$$
$$\text{s.t. } x \in \mathcal{X} \tag{3.2}$$

where $\mathcal{X}$ has the assumption $\sup_{x\in\mathcal{X}} ||x||_2 \leq R$, that is we can enclose the optimisation space inside a Euclidean ball of size R. The optimal will be denoted $x^* = \arg\min_{x\mathcal{X}} f(x)$.

An assortment of assumptions can be placed upon f, the following 3 are of most interest

1. L-Lipschitz: $|f(x) - f(y)| \leq L||x - y||_2$ for any $x, y \in \mathbb{R}^n$ $(||\nabla f(x)|_2 \leq L)$.
2. $\beta-$smooth: $||\nabla f(x) - \nabla f(x)||_2 \leq \beta||x - y||_x$ for any $x, y \in \mathbb{R}^n$ $(\nabla^2 f(x) \preccurlyeq \beta I$ for any $x, y \in \mathbb{R}^n)$
3. $\alpha-$Strong convexity: $f(x) - f(y) \leq \nabla f(x)^T(x - y) - \frac{\alpha}{2}||x - y||_2^2$ for any $x, y \in \mathbb{R}^n$ $(\nabla^2 f(x) \succcurlyeq \alpha I)$ for any $x \in \mathbb{R}^n)$.

Recall that $\alpha-$Strong convexity may not be reasonable in applied problems where the sample size is small, due to is aligning implying the empirical covariance matrix is invertible. For more information on this look to proposition 2 from the notes accompanying this reading group.

Now we recall some results from the previous chapter regarding projected gradient descent. The routine, denoting the set of sub gradients for $f(x)$ as $\partial f(x)$, is the following

$$y_{t+1} = x_t - \eta g_t \quad \text{where } g_t \in \partial f(x_t) \tag{3.3}$$
$$x_{t+1} = \Pi_{\mathcal{X}}(y_{t+1}) \tag{3.4}$$

where the function $\Pi_{\mathcal{X}}(y_{t+1})$ projects the point y_{t+1} back into the constraint set $\mathcal{X}$. The rate at the above routine finds x^* depends upon which of the assumptions above f satisfies. We now list down the bounds between the the sequence of points produced from the above routine and the optimal x^* under different conditions on f.

L-Lipschitz and Convex from Theorem 3.2 from [13] with $\eta = \frac{R}{L\sqrt{t}}$

$$f\left(\frac{1}{t}\sum_{s=1}^{t} x_s\right) - f(x^*) \leq \frac{RL}{\sqrt{t}} \tag{3.5}$$

giving a complexity of $O(RL/\sqrt{t})$

$\beta-$**Smooth and Convex** from Theorem 3.7 in [13] with $\eta = \frac{1}{\beta}$

$$f(x_t) - f(x^*) \leq \frac{3\beta||x_1 - x^*||^2 + f(x_1) - f(x^*)}{t} \tag{3.6}$$

giving a complexity of $O\left(\frac{R^2\beta}{t}\right)$.

L-Lipschitz and $\alpha-$strongly Convex from Theorem 3.9 in [13] with $\eta_s = \frac{2}{\alpha(s+1)}$, that is the gradient step now depends upon time, we get

$$f\left(\sum_{s=1}^{t} \frac{2s}{t(t+1)} x_s\right) - f(x^*) \leq \frac{2L^2}{\alpha(t+1)} \tag{3.7}$$

giving a complexity of $O\left(\frac{L^2}{\alpha(t+1)}\right)$.

$\beta-$**Smooth and $\alpha-$strongly Convex** from Theorem 3.10 in [13] with $\eta = \frac{1}{\beta}$ we have

$$||x_{t+1} - x^*||^2 \leq \exp\left(-\frac{\alpha t}{\beta}\right)||x_1 - x^*||^2 \tag{3.8}$$

giving a complexity of $O\left(R^2 \exp\left(-\frac{\alpha t}{\beta}\right)\right)$.

3.2 Week 3 - Lower Bounds

To prove lower complexity bounds a "sufficiently hard" f, satisfying the various conditions, must be found, that, when giving a black-box procedure a fixed number of first order oracle calls, will remain a fixed distance away from its optimum. An example of a black-box procedure being the projected gradient descent of previously - taking a position and sub gradient (x_t, g_t) $g_t \in \partial f(x_t)$ and returning the next position $x_{t+1} = \Pi_{\mathcal{X}}(x_t - \eta g_t)$. Recalling that a first order oracle call at $f(x)$ returns to us a sub-gradient $g \in \partial f(x)$.

Generalising this, given a history $(x_1, g_1, \ldots, x_t, g_t)$ such that $g_s \in \partial f(x_s)$, a black-box procedure will be a mapping $(x_1, g_1, \ldots, x_t, g_t) \to x_{t+1}$. Letting $x_1 = 0$, we will assume that the black-box procedure, for any $t \geq 0$, returns a position in the span of the gradients

$$x_{t+1} \in \mathrm{Span}(g_1, \ldots, g_t). \tag{3.9}$$

Additionally we denote the standard basis of $\mathbb{R}^n$ as $e_1, \ldots, e_n$, and the Euclidean ball of radius R as $B_2(R) = \{x \in \mathbb{R}^n : ||x|| < R\}$.

Now for any $t \leq n$ we will show there exists an f that will remain bounded away from its optimal within the first t calls to a black-box procedure satisfying (3.9). Noting that if the number of oracle calls grows beyond the dimension $(t > n)$ Vaidya method from Chapter 2 becomes viable, achieving ε accuracy in $n \log(\frac{Rn}{r\varepsilon})$ first order oracle calls.

The remainder of this section is as follows: first lower oracle complexity bounds for various f are listed, after which the proof for the $\beta-$Smooth and Convex case is provided. The proof for the $L-$Lipschitz $\alpha-$Strong Convex case being omitted due to it being similar in spirit to the $\beta-$Smooth case. After which we will find out there a gap between the rates achieved by projected gradient descent and the proven complexity lower bounds, suggesting new methods can be found in order to improve rates. Before moving on we highlight an additional source [8, 75-76] which included some interesting relevant points.

Theorem 3.2.1. *Lower Bound for L-Lipschitz and Convex / $\alpha-$Convex*

Let $t \leq n$, $L, R > 0$. There exists an L-Lipschitz and Convex f such that for any black-box procedure satisfying (3.9)

$$\min_{1\leq s\leq t} f(x_s) - \min_{x\in B_2(R)} f(x) \geq \frac{RL}{2(1+\sqrt{t})} \tag{3.10}$$

thus giving oracle complexity of $\Omega\left(\frac{1}{\varepsilon^2}\right)$. Moreover if f is $\alpha-$strongly Convex

$$\min_{1\leq s\leq t} f(x_s) - \min_{x\in B_2(R)} f(x) \geq \frac{L^2}{8\alpha t} \tag{3.11}$$

giving oracle complexity of $\Omega\left(\frac{1}{\varepsilon}\right)$.

Remark: These rates are optimal in the case of projected gradient descent.

Proof: See Theorem 3.13 *[13].*

Theorem 3.2.2. *Lower Bound for $\beta-$Smooth and Convex*

Let $t \leq (n-1)/2, \beta > 0$. There exists a $\beta-$smooth convex f such that for any black-box procedure satisfying (3.9)

$$\min_{1\leq s\leq t} f(x_s) - \min_{x\in B_2(R)} f(x) \geq \frac{3\beta}{32}\frac{||x_1 - x^*||^2}{(t+1)^2} \tag{3.12}$$

giving a lower oracle complexity bound of $\Omega\left(\frac{1}{\sqrt{\varepsilon}}\right)$.

Proof Adapted from [13] Theorem 3.14:

Let $A_k \in \mathbb{R}^{n\times n}$ be a matrix defined by

$$(A_k)_{i,j} = \begin{cases} 2, & i=j, i\leq k \\ -1, & j\in\{i-1,i+1\}, i\leq k, j\neq k+1 \\ 0, & \text{Otherwise} \end{cases}$$

which can alternatively be represented in the standard basis as

$$A_k = 2\sum_{i=1}^{k} e_i e_i^T - \sum_{i=1}^{k-1} e_i e_{i+1}^T - \sum_{i=2}^{k-1} e_{i+1} e_i^T \tag{3.13}$$

noting that $0 \preccurlyeq A_k \preccurlyeq 4I_n$ since

$$x^T A_k x = 2\sum_{i=1}^{k} x(i)^2 - 2\sum_{i=1}^{k-1} x(i)x(i+1) = x(1)^2 + x(k)^2 + \sum_{i=1}^{k-1}(x(i) - x(i+1))^2.$$

Consider now the $\beta-$smooth convex function

$$f(x) = \frac{\beta}{8}x^T A_{2t+1} x - \frac{\beta}{4}x^T e_1$$

which has the sub gradient $\partial f(x) = \frac{\beta}{4}A_{2t+1}x - \frac{\beta}{4}e_1$. As $x_1 = 0$ we have $\partial f(x_1) = -\frac{\beta}{4}e_1$, and due to (3.9), $x_2 = \eta_{21}e_1$ for $\eta_{21} \in \mathbb{R}$. Next iteration we have $\partial f(x_2) = \frac{\beta}{4}\left(A_{2t+1}x_2 - e_1\right)$ which, using the representation (3.13), is now in the direction of e_2. To see this consider the following

$$\begin{aligned} A_{2t+1}x_2 = \eta_{21}A_k e_1 &= \eta_{21}\left(2\sum_{i=1}^{2t+1} e_i e_i^T - \sum_{i=1}^{2t} e_i e_{i+1}^T - \sum_{i=2}^{2t} e_{i+1}e_i^T\right)e_1 \\ &= \eta_{21}\left(2e_1 - e_2\right) \end{aligned}$$

as $e_i^T e_1 = 0$ if $i \neq 1$. Therefore we can deduce that $\mathrm{Span}(e_1, \partial f(x_2)) = \mathrm{Span}(e_1, e_2)$.

Proceeding inductively for iterations $2 \leq s \leq t$, we have $x_s \in \mathrm{Span}(e_1, \dots, e_{s-1})$, and therefore $x_s = \sum_{i=1}^{s-1} \eta_{s-1,i}e_i$. Once again $\partial f(x_s)$ will contain

$$\begin{aligned} A_{2t+1}x_s &= \left(2\sum_{i=1}^{2t+1} e_i e_i^T - \sum_{i=1}^{2t} e_i e_{i+1}^T - \sum_{i=2}^{2t} e_{i+1}e_i^T\right)\left(\sum_{j=1}^{s-1}\eta_{s-1,j}e_j\right) \\ &= \sum_{j=1}^{s-1}\eta_{s-1,j}\left(2\sum_{i=1}^{2t+1} e_i e_i^T - \sum_{i=1}^{2t} e_i e_{i+1}^T - \sum_{i=2}^{2t} e_{i+1}e_i^T\right)e_j \\ &= \sum_{j=1}^{s-1}\eta_{s-1,j}\left(2e_j - e_{j-1} - e_{j+1}\right) \end{aligned}$$

which most importantly is now in the direction of e_s due to element e_{j+1} under the sum, meaning that $\mathrm{Span}(e_1, \dots, e_{s-1}, \partial f(x_s)) = \mathrm{Span}(e_1, \dots, e_s)$.

Intuitively, due to the right most sum in (3.13), one extra co-ordinate is explored per iteration. This means, for some $s \leq t$, x_s will be zero in the co-ordinates not yet to explore, that is $x_s(i) = 0$ for $i = s, \dots, n$. Therefore $x_s^T A_{2t+1}x_s = x_s^T A_s x_s$, which means the best we can do in the first s iterations is find the optimal of a restricted version of f, namely $f_s(x)$, defined for some k as

$$f_k(x) = \frac{\beta}{8}xA_kx - \frac{\beta}{4}x^T e_1. \tag{3.14}$$

The objective is to bound the difference between the optimal values of the restricted f_s and global f, thus proving the black box method can only do so well in t iterations. Denoting $f^* = \inf_{x\in\mathcal{X}} f(x)$, consider the following set of inequalities

$$f(x_s) - f^* = f_s(x_s) - f_{2t+1}^* \geq f_s^* - f_{2t+1}^* \geq f_t^* - f_{2t+1}^* \tag{3.15}$$

To see $f_s^* \geq f_t^*$ observe that

$$\begin{aligned} f_t(x) &= \sum_{i=1}^{s} x(i) - 2\sum_{i=1}^{s-1} x(i)x(i+1) + 2\sum_{i=s+1}^{t} x(i)^2 - 2\sum_{i=s}^{t-1} x(i)x(i+1) \\ &= f_s(x) + x(s+1)^2 + x(t)^2 + \sum_{i=s+1}^{t-1}\left(x(i) - x(i+1)\right)^2 - 2x(s)x(s+1) \end{aligned}$$

which when plugging in the optimal point for f_s can attain $f_t(x_s^*) = f_s(x_s^*) = f_s^*$ as $x_s^*(i) = 0$ for $i = s+1, \dots, n$.

We must now explicitly find the minimiser x_k^* for f_k, its norm, and the corresponding objective f_k^*. To find x_k^* we have $\partial f_k(x) = 0 \implies A_k x = e_1$, and since $x_k^* \in \text{Span}(e_1, \dots, e_k)$, the solution becomes $x_k^*(i) = 1 - \frac{i}{k+1}$ for $i = 1, \dots, k$. We then immediately get

$$f_k^* = -\frac{\beta}{8}\left(1 - \frac{1}{k+1}\right). \tag{3.16}$$

With the norm of x_s^* then becoming

$$||x_k^*|| = \sum_{i=1}^{k}\left(1 - \frac{i}{k+1}\right)^2 = \sum_{i=1}^{k}\left(\frac{i}{k=1}\right)^2 \leq \frac{k+1}{3}. \tag{3.17}$$

Finally bringing together (3.15), (3.16), (3.17) we get

$$f(x_s) - f^* \geq f_t^* - f_{2t+1}^* = \frac{\beta}{8}\left(\frac{1}{t+1} - \frac{1}{2t+2}\right) \geq \frac{3\beta}{32}\frac{||x_{2t+1}^*||^2}{(t+1)^2}$$

noting that as the right hand side does not depending upon x_s, a minimum can be taken. □

To make comparison between the lower bound rates as well as those obtained by projected gradient descent, we look to oracle complexities contained in Figure 3.1.

		Upper Bound (PGD)	Lower Bound
Convex	$L-$Lipschitz	$O\left(\frac{1}{\varepsilon^2}\right)$	$\Omega\left(\frac{1}{\varepsilon^2}\right)$
	$\beta-$Smooth	$O\left(\frac{1}{\varepsilon}\right)$	$\Omega\left(\frac{1}{\sqrt{\varepsilon}}\right)$
$\alpha-$Strong Convex	$L-$Lipschitz	$O\left(\frac{1}{\varepsilon}\right)$	$\Omega\left(\frac{1}{\varepsilon}\right)$
	$\beta-$Smooth	$O\left(\frac{\beta}{\alpha}\log\left(\varepsilon^{-1}\right)\right)$	$\Omega\left(\sqrt{\frac{\beta}{\alpha}}\log\left(\varepsilon^{-1}\right)\right)$

Figure 3.1: Oracle complexity- order of iterations t required for $f(x_t) - f^* \leq \varepsilon$. Source for lower bound in the $\beta-$Smooth $\alpha-$Strong Convex case from Theorem 3.15 [13]. Upper Bound is oracle complexity of (PGD) Projected Gradient Descent.

Observe that there is a gap in the upper and lower bounds in the case of $\beta-$smooth functions, specifically an order $\frac{1}{\sqrt{\varepsilon}}$ and $\sqrt{\frac{\beta}{\alpha}}$ for Convex and $\alpha-$Strong Convex functions respectively. This suggests there are more optimal black box procedure over projected gradient descent that could close the complexity gap.

Chapter 4

Application: Boosting

Speaker: Patrick Rebeschini, 02/11/2017.

In this section we go back to machine learning. We consider the same setting introduced in Section 1, and apply the projected subgradient descent algorithm to minimize the empirical risk in an important example: Boosting. The setting is that of binary classification, where given m i.i.d. labeled data points $(W_1, Y_1), \ldots, (W_m, Y_m) \in \mathcal{W} \times \mathcal{Y}$, $\mathcal{Y} = \{-1, 1\}$, coming from an unknown distribution, one wants to construct a classifier $h_{\text{hard}} : \mathcal{W} \to \{-1, 1\}$ that minimizes the probability of mistakes over the unseen data, i.e., that minimize the expected risk with respect to the "true" loss $\mathbf{E}\varphi_{\text{true}}(-Y h_{\text{hard}}(W)) = \mathbf{P}(h_{\text{hard}}(W) \neq Y)$, where $\varphi_{\text{true}}(u) = \mathbf{1}_{u \geq 0}$, and where (W, Y) is a random variable (independent of everything else) coming from the same unknown distribution as the m data points we are given.

As stated the problem is discrete. To get a continuous problem where we can apply gradient descent methods, we relax the setting above in two ways. First, instead of the true loss φ_{true} we consider a convex "surrogate" φ, i.e., a convex function φ such that $\varphi_{\text{true}}(u) \leq \varphi(u)$ for any $u \in \mathbb{R}$. Possible choices are:

- Exponential loss: $\varphi(u) = \exp(u)$.
- Hinge loss: $\varphi(u) = \max\{0, 1 + u\}$.
- Logistic loss: $\varphi(u) = \log_2(1 + \exp(u))$.

Second, instead of *hard* classifiers $h_{\text{hard}} : \mathcal{W} \to \{-1, 1\}$, we consider *soft* classifiers $h : \mathcal{W} \to [-1, 1]$. The problem we want to solve is then

$$\begin{aligned} \text{minimize} \quad & \mathbf{E}\varphi(-Y h(W)) \\ \text{subject to} \quad & h \in \mathcal{H}, \end{aligned}$$

where $\mathcal{H}$ is a given class of soft classifiers. In the setting of Boosting, one assumes to have access to a set of n *base* (soft or hard, it does not matter here; what matters for what we develop today is that each Φ_k is bounded) classifiers encoded in a vector map $\Phi : \mathcal{W} \to [-1, 1]^n$, where $\Phi(W)_k \equiv \Phi_k(W)$ represents the outcome of the k-th base classifier on the data point W, and one wants to find the best convex combinations of these base classifiers. If we let $\Delta_n := \{x \in [0, 1]^n : \sum_{k=1}^n x_k = 1\}$ be the n-dimensional probability simplex, we want to find $x \in \Delta_n$ so that $x^T \Phi = \sum_{k=1}^n x_k \Phi_k$ minimizes the expected risk. In other words, we consider the family $\mathcal{H} = \{h : h = x^T \Phi \text{ for some } x \in \Delta_n\}$, and the problem we want to solve reads:

$$\begin{aligned} \text{minimize} \quad & r(x) = \mathbf{E}\varphi(-Y x^T \Phi(W)) \\ \text{subject to} \quad & x \in \Delta_n. \end{aligned}$$

This problem is a particular instance of the general formulation given in (1.1), with the choice $\ell(z, y) = \varphi(-zy)$ for the loss function and $\mathcal{X} = \Delta_n$ for the constraint set. We can then work within the framework of empirical risk minimization as introduced in Section 1, and use the error decomposition given in Proposition 1.0.2 to bound the *STATISTICS* term $\sup_{x\in\mathcal{X}}(r(x) - R(x)) + \sup_{x\in\mathcal{X}}(R(x) - r(x))$ and the *OPTIMIZATION* term $R(\hat{X}_t) - R(X^\star)$, respectively.

4.1 Statistics term

Note that in the case of Boosting, as $x \in \Delta_n$ and $\Phi \in [-1, 1]^n$ and $\mathcal{Y} = \{-1, 1\}$, the loss function φ is only evaluated in the interval $[-1, 1]$, as $-1 \leq yx^T\Phi(w) \leq 1$ for any $x \in \Delta_n, w \in \mathcal{W}$, and $y \in \mathcal{Y}$. So, we can restrict to this interval as far as the Lipschitz property of φ goes. On $[-1, 1]$ we have the following Lipschitz constants:

- Exponential loss: $\varphi(u) = \exp(u)$, $L_\varphi = e$.
- Hinge loss: $\varphi(u) = \max\{0, 1 + u\}$, $L_\varphi = 1$.
- Logistic loss: $\varphi(u) = \log_2(1 + \exp(u))$, $L_\varphi = \log_2(e)\frac{e}{1+e} \approx 1.05$.

A direct application of Proposition 1.1.1 immediately yields the following result.

Corollary 4.1.1. Let φ be L_ℓ-Lipschitz on $[-1, 1]$. In the case of Boosting, we have

$$\mathbf{E}[\text{STATISTICS}] \leq 4\frac{L_\varphi}{\sqrt{m}}\sqrt{n}.$$

Proof. The proof follows from Proposition 1.1.1 noticing that in the setting of Boosting we have $\|\Phi(W)\|_2 \leq \sqrt{n}$ and $\sup_{x\in\Delta_n} \|x\|_2 \leq 1$. $\square$

We could also directly apply Proposition 1.1.2 to obtain a bound in high probability, and we would similarly get a bound that depends *polynomially* on n. Upon a closer look, we note that the assumptions of Proposition 1.1.1 and Proposition 1.1.2 are with respect to the Euclidean norm $\|\cdot\|_2$. However, while $\|\Phi(W)\|_2 \leq \sqrt{n}$ and $\sup_{x\in\Delta_n} \|x\|_2 \leq 1$, one has $-1 \leq x^T\Phi(W) \leq 1$ for each $x \in \Delta_n$, so perhaps one can avoid the linear dependence on $\sqrt{n}$ by developing a new version of Proposition 1.1.1 and Proposition 1.1.2 that can avoid the use of Cauchy Schwarz $x^T\Phi(W) \leq \|x\|_2\|\Phi(W)\|_2$ and directly use that $-1 \leq x^T\Phi(W) \leq 1$. This is our goal. We prove the following results.

Proposition 4.1.2 (Mean). Let φ be L_ℓ-Lipschitz on $[-1, 1]$. In the case of Boosting we have

$$\mathbf{E}[\text{STATISTICS}] \leq 4\frac{L_\varphi}{\sqrt{m}}\sqrt{2\log n}.$$

Proposition 4.1.3 (High probability). Let φ be L_ℓ-Lipschitz on $[-1, 1]$. In the case of Boosting we have

$$\text{STATISTICS} \leq 2\frac{L_\varphi}{\sqrt{m}}(2\sqrt{2\log n} + \sqrt{2\log(1/\delta)}).$$

Bound on the mean

We prove Proposition 4.1.2. To bound the mean of the *STATISTICS* term in the case of Boosting, we follow the same plan described in Section 1.1 and use the notion of Rademacher complexity. This time, however, we avoid the use of Cauchy Schwarz to bound the Rademacher complexity. First, following the proof of Proposition 1.1.4 it is immediate to derive a new version of this result for a loss function of the type $\ell(z, y) = \varphi(-yz)$.

Proposition 4.1.4. Let $yx^T\Phi(w) \in [d_-, d_+]$ for each $x \in \mathcal{X}, w \in \mathcal{W}$, and $y \in \mathcal{Y}$. Let φ be L_φ-Lipschitz on $[d_-, d_+]$. Then,

$$\mathbf{E} \sup_{x\in\mathcal{X}} (r(x) - R(x)) \le 2L_\varphi \mathbf{E} \sup_{x\in\mathcal{X}} \frac{1}{m} \sum_{i=1}^m \epsilon_i x^T \Phi(W_i) Y_i,$$

where $\epsilon_1, \ldots, \epsilon_n$ are i.i.d. random variables uniform in $\{-1, 1\}$, independent of $(W_1, Y_1), \ldots, (W_m, Y_m)$.

Proof. The proof is exactly the same as the proof of Proposition 1.1.4 up to equation (1.4), which now reads

$$\mathbf{E}\left[\sup_{x\in\mathcal{X}} \frac{1}{m} \sum_{i=1}^m \epsilon_i \varphi(-x^T\Phi(W_i)Y_i) \middle| Z_1, \ldots, Z_m \right] \le L_\ell \mathbf{E}\left[\sup_{x\in\mathcal{X}} \frac{1}{m} \sum_{i=1}^m \epsilon_i x^T\Phi(W_i)Y_i \middle| Z_1, \ldots, Z_m \right].$$

□

We now derive a new version of Proposition 1.1.5 to bound $\mathbf{E}\sup_{x\in\mathcal{X}} \frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i)Y_i$, where instead of using the Cauchy Schwarz inequality and get a bound that depends on the Euclidean norms of x and Φ, we use that in the case of Boosting where $\mathcal{X} = \Delta_n$ the supremum inside the expectation is achieved in a vertex of the simplex, and hence, conditioning on the data, we are left with the (empirical) Rademacher complexity of a *finite* set, which only grows *logarithmically* with the size of the set. We first state the result about the behaviour of the Rademacher complexity $\mathcal{R}(T)$ when the set T has finite cardinality.

Lemma 4.1.5. Let $T \subseteq \mathbb{R}^m$ with $|T| < \infty$. We have

$$\mathcal{R}(T) \le \max_{t\in T} \|t\|_2 \frac{\sqrt{2\log|T|}}{m}.$$

Proof. Recall from Definition 1.1.1 that $\mathcal{R}(T) := \mathbf{E}\sup_{t\in T} \frac{1}{m}\sum_{i=1}^m \epsilon_i t_i$. If we use Cauchy Schwarz, we would get

$$\mathcal{R}(T) \le \sup_{t\in T} \|t\|_2 \frac{\mathbf{E}\|\epsilon\|_2}{m} = \sup_{t\in T} \|t\|_2 \frac{1}{\sqrt{m}},$$

which is too general for our case. In particular, this result does not use the fact that T is a finite set. To get the $1/m$ rate in the bound in the case when $|T| < \infty$, we adopt two usual tricks in probability: first, we take exponentials and use Jensen's inequality; second, we bound a maximum over a set of positive numbers by its sum. For the first step, note that for any real-valued random variable X and any $s > 0$, Jensen's inequality yields

$$\mathbf{E}X = \frac{1}{s}\log e^{s\mathbf{E}X} \le \frac{1}{s}\log \mathbf{E}e^{sX}.$$

For the second step, note that if $X = \max_{t\in T} X_t$, then

$$\mathbf{E}e^{sX} = \mathbf{E}\max_{t\in T} e^{sX_t} \le \sum_{t\in T} \mathbf{E}e^{sX_t}.$$

If we choose $X_t = \sum_{i=1}^m \epsilon_i t_i$, then

$$\mathbf{E}e^{sX_t} = \prod_{i=1}^m \mathbf{E}e^{s\epsilon_i t_i} \le \prod_{i=1}^m e^{s^2 t_i^2/2} = e^{s^2\|t\|_2^2/2},$$

where we used Hoeffding's Lemma that says that for a random variable Z that is bounded in the interval $[a, b]$, i.e., $a \le Z \le b$ a.s., one can bound its moment generating function as follows: $\mathbf{E}e^{sZ} \le e^{s^2(b-a)^2/8}$. Putting everything together, we get

$$\mathbf{E}X \le \frac{1}{s}\log \mathbf{E}e^{sX} \le \frac{1}{s}\log\sum_{t\in T}\mathbf{E}e^{sX_t} \le \frac{1}{s}\log\sum_{t\in T} e^{s^2\|t\|_2^2/2} \le \frac{1}{s}\log|T| + \frac{s}{2}\max_{t\in T}\|t\|_2^2.$$

Optimizing this bounds over $s > 0$, one finds $\mathbf{E}X \le \max_{t\in T}\|t\|_2\sqrt{2\log|T|}$, and the proof follows. □

Armed with the previous bound on the Rademacher complexity of a finite set, we can bound the object $\mathbf{E}\sup_{x\in\mathcal{X}}\frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i)Y_i$ as follows.

Proposition 4.1.6. In the case of Boosting, we have

$$\mathbf{E}\sup_{x\in\Delta_n}\frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i)Y_i \le \sqrt{\frac{2\log n}{m}}.$$

Proof. Let us define the (random) function

$$x\in\Delta_n \to G(x) := \frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i)Y_i = \sum_{j=1}^n x_j A_j,$$

where $A_j := \frac{1}{m}\sum_{i=1}^m \epsilon_i\Phi(W_i)_j Y_i$. As G is a linear function, its supremum over the simplex Δ_n is achieved in at least one of the n vertices of the simplex. Note that

$$\sup_{x\in\Delta_n} G(x) = \sup_{x\in\Delta_n}\sum_{j=1}^n x_j A_j \le \max\{A_1,\dots,A_n\}.$$

If we let $e_1,\dots,e_n\in\mathbb{R}^n$ be the base vectors, then it is immediate to see that, for instance, the bound above is achieved with equality at the vertex $e_{j^\star}$, where $j^\star := \min\{k\in\{1,\dots,n\} : A_k = \max\{A_1,\dots,A_n\}\}$. Hence,

$$\sup_{x\in\Delta_n} G(x) = \max_{x\in\{e_1,\dots,e_n\}} G(x),$$

which yields

$$\mathbf{E}\sup_{x\in\Delta_n}\frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i)Y_i = \mathbf{E}\max_{x\in\{e_1,\dots,e_n\}}\frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i)Y_i.$$

Conditioning on the data we get that the empirical Rademacher complexity is the Rademacher complexity over a finite set. Using the notation $Z_i = (W_i, Y_i)$, in the spirit of Remark 1.1.1, we can use Lemma 4.1.5 to get

$$\begin{aligned}\mathbf{E}\left[\max_{x\in\{e_1,\dots,e_n\}}\frac{1}{m}\sum_{i=1}^m \epsilon_i x^T\Phi(W_i)Y_i\middle| Z_1,\dots,Z_m\right] &= \mathbf{E}\left[\max_{t\in T_{Z_1,\dots,Z_m}}\frac{1}{m}\sum_{i=1}^m \epsilon_i t_i\middle| Z_1,\dots,Z_m\right]\\ &\le \max_{t\in T_{Z_1,\dots,Z_m}}\|t\|_2\frac{\sqrt{2\log|T_{Z_1,\dots,Z_m}|}}{m} \le \sqrt{\frac{2\log n}{m}},\end{aligned}$$

where $T_{Z_1,\dots,Z_m} := \{(x^T\Phi(W_1)Y_1,\dots,x^T\Phi(W_m)Y_m)^T : x\in\{e_1,\dots,e_n\}\}$ contains n vectors, i.e., $|T_{Z_1,\dots,Z_m}| = n$, and clearly $\max_{t\in T_{Z_1,\dots,Z_m}}\|t\|_2 \le \sqrt{m}$ (here we use $x^T\Phi\in[-1,1]$ instead of Cauchy Schwarz). □

Proposition 4.1.4 (with $d_- = -1$, $d_+ = 1$) and Proposition 4.1.6 immediately yields the following result, from which Proposition 4.1.2 follows immediately.

Proposition 4.1.7. Let φ be L_ℓ-Lipschitz on $[-1,1]$. In the case of Boosting we have

$$\mathbf{E}\sup_{x\in\mathcal{X}}(r(x)-R(x)) \le 2\frac{L_\varphi}{\sqrt{m}}\sqrt{2\log n}.$$

Bound with high probability

We prove Proposition 4.1.3. To derive a bound in high probability for the $STATISTICS$ term in the case of Boosting, we follow the same plan described in Section 1.1 and use the Bounded Difference concentration inequality. This time, however, we avoid the use of Cauchy Schwarz to find the constant c that bounds the variation of a single component in the Bounded Difference inequality. Below is a new version of Proposition 1.1.8, where instead of using Cauchy Schwarz to get the generic bound $x^T\Phi(W) \leq \|x\|_2\|\Phi(W)\|_2$, we use the assumption that $x^T\Phi(W) \leq [d_-, d_+]$. This assumption is satisfied in the case of Boosting with $d_- = -1$ and $d_+ = 1$. This allows to save a factor n as opposed to what one would have by directly using Proposition 1.1.8 (Proposition 1.1.8 yields $c = \frac{2}{m}(|\varphi(0)| + \sqrt{n}L_\ell)$).

Proposition 4.1.8. Let h be the function defined in (1.5). Let $\mathcal{X}$ be bounded, and let $yx^T\Phi(w) \in [d_-, d_+]$ for each $x \in \mathcal{X}, w \in \mathcal{W}$, and $y \in \mathcal{Y}$. Let φ be L_φ-Lipschitz on $[d_-, d_+]$. Then, $c \in \mathbb{R}$ satisfying the requirement of the Bounded Difference inequality is given by

$$c = (d_+ - d_-)\frac{L_\varphi}{m}.$$

Proof. Fix $k \in \{1, \dots, m\}$ and let $z = (z_1, \dots, z_{k-1}, z_k, z_{k+1}, \dots, z_m)$ and $z' = (z_1, \dots, z_{k-1}, z'_k, z_{k+1}, \dots, z_m)$. Then,

$$|h(z) - h(z')| = \left| \sup_{x\in\mathcal{X}} \left(r(x) - \frac{1}{m}\sum_{i=1}^m R_i^z(x) \right) - \sup_{x\in\mathcal{X}} \left(r(x) - \frac{1}{m}\sum_{i=1}^m R_i^{z'}(x) \right) \right|,$$

where $R_i^z(x) := \ell(x^T\Phi(w_i), y_i) = \varphi(-y_i x^T\Phi(w_i))$. If $h(z) - h(z') \geq 0$ and we let $\tilde{x} \in \mathcal{X}$ be the maximizer of $\sup_{x\in\mathcal{X}}(r(x) - \frac{1}{m}\sum_{i=1}^m R_i^z(x))$ (note that the supremum is attained by the Extreme Value Theorem), we have

$$\begin{aligned}
h(z) - h(z') &= \left(r(\tilde{x}) - \frac{1}{m}\sum_{i=1}^m R_i^z(\tilde{x}) \right) - \sup_{x\in\mathcal{X}} \left(r(x) - \frac{1}{m}\sum_{i=1}^m R_i^{z'}(x) \right) \\
&\leq \left(r(\tilde{x}) - \frac{1}{m}\sum_{i=1}^m R_i^z(\tilde{x}) \right) - \left(r(\tilde{x}) - \frac{1}{m}\sum_{i=1}^m R_i^{z'}(\tilde{x}) \right) \\
&= \frac{1}{m}(R_k^z(\tilde{x}) - R_k^{z'}(\tilde{x})) = \frac{1}{m}(\varphi(-y_k\tilde{x}^T\Phi(w_k)) - \varphi(-y'_k\tilde{x}^T\Phi(w'_k))) \\
&\leq \frac{1}{m}(\sup_{d_-\leq u\leq d_+} \varphi(u) - \inf_{d_-\leq u\leq d_+} \varphi(u))) \leq (d_+ - d_-)\frac{L_\varphi}{m},
\end{aligned}$$

where the last but one inequality comes as $d_- \leq yx^T\Phi(w) \leq d_+$ for each $y \in \mathcal{Y}, x \in \mathcal{X}, w \in \mathcal{W}$. Proceeding analogously in the case $h(z) - h(z') \leq 0$, we obtain the result. □

The proof of Proposition 4.1.3 follows the same lines as the proof of Proposition 1.1.2 in Section 1.1.

Proof of Proposition 4.1.3. Using, respectively, Proposition 1.1.7 and Proposition 4.1.7, we have that with probability at least $1 - \delta$ the following holds:

$$\sup_{x\in\mathcal{X}}(r(x) - R(x)) \leq \mathbf{E}\left[\sup_{x\in\mathcal{X}}(r(x) - R(x)) \right] + c\sqrt{\frac{m}{2}\log\frac{1}{\delta}} \leq 2L_\varphi\sqrt{\frac{2\log n}{m}} + c\sqrt{\frac{m}{2}\log\frac{1}{\delta}},$$

with $c = 2L_\varphi/m$ by Proposition 4.1.8, which yields

$$\mathbf{P}\left(\sup_{x\in\mathcal{X}}(r(x) - R(x)) \leq c' \right) \geq 1 - \delta,$$

where $c' := L_\ell(2\sqrt{2\log n} + \sqrt{2\log(1/\delta)})/\sqrt{m}$. As the bounds we have derived holds also for $\sup_{x\in\mathcal{X}}(R(x) - r(x))$, we have

$$\mathbf{P}\Big(\sup_{x\in\mathcal{X}}(r(x)-R(x))+\sup_{x\in\mathcal{X}}(R(x)-r(x)) \le 2c'\Big) \ge \mathbf{P}\Big(\Big\{\sup_{x\in\mathcal{X}}(r(x)-R(x)) \le c'\Big\}\bigcap\Big\{\sup_{x\in\mathcal{X}}(R(x)-r(x)) \le c'\Big\}\Big) \ge 1-\delta.$$

$\square$

4.2 Optimization term?

Following the error decomposition given in Proposition 1.0.2, Proposition 4.1.2 and Proposition 4.1.3 show that in the case of Boosting the *STATISTICS* term is $O(\sqrt{\log n/m})$, which only grows logarithmically with the number of base classifiers n. This is a nice feature in applications where the number n of base classifiers is very large, possibly exponential. Hence, one would hope to be able to design an algorithmic procedure that only uses $\sim \log n$ calls to the first order oracle in order to match the accuracy of the *STATISTICAL* term to find an approximate solution to the empirical risk minimization problem:

$$\begin{aligned} &\text{minimize} \quad && R(x) = \frac{1}{m}\sum_{i=1}^m \varphi(-Y_i x^T\Phi(W_i)) \\ &\text{subject to} \quad && x \in \Delta_m. \end{aligned}$$

Given Theorem 2.3.2, we can assess the guarantees given by the projected subgradient descent method to solve this optimization problem. Note that if φ is L_φ-Lipschitz on $[-1,1]$, then the empirical risk R is $L_\varphi\sqrt{n}$-Lipschitz on $[-1,1]$:

$$\begin{aligned} |R(x) - R(y)| &\le \frac{1}{m}\sum_{i=1}^m |\varphi(-Y_i x^T\Phi(W_i)) - \varphi(-Y_i y^T\Phi(W_i))| \le \frac{1}{m}\sum_{i=1}^m L_\varphi \|Y_i(x-y)^T\Phi(W_i)\|_2 \\ &\le L_\varphi\|\Phi(W_i)\|_2\|x-y\|_2 \le \sqrt{n}L_\varphi\|x-y\|_2. \end{aligned}$$

As $B = \sup_{x\in\Delta_n}\|x\|_2 = 1$, projected subgradient descent yields a rate $O(L_\varphi\sqrt{n/t})$. Imposing the condition $\sqrt{n/t} \lesssim \sqrt{\log n/m}$, we find that $t \gtrsim nm/\log n$, which does *not* scale logarithmically with n as we hoped for!

Boosting is one example where one would like to apply subgradient descent methods on a non-Euclidean space: the probability simplex Δ_n. However, subgradient descent methods are only designed for the Euclidean geometry. Note, in fact, that all the definitions we gave for L-Lipschitz, β-smoothness, and α-strong convexity, as well as the bounds for the constraint set $\mathcal{X}$, are expressed in terms of the Euclidean norm $\|\cdot\|_2$. This motivates the design of a new class of algorithms (in fact, a generalization of subgradient descent) that can adapt to the geometry of the problem at hand, as we will see next.

Chapter 5

Non-Euclidean setting: mirror descent

Let $f\colon \mathcal{X} \to \mathbb{R}$ denote a convex function and suppose we are interested in the following optimisation problem

$$\min_{x \in \mathcal{X}} f(x)$$

The projected subgradient descent algorithm proceeds as follows

$$\begin{aligned} y_{t+1} &= x_t - \eta g_t, \qquad g_t \in \partial f(x_t) \\ x_{t+1} &= \Pi_{\mathcal{X}}(y_{t+1}), \end{aligned}$$

where $\Pi_{\mathcal{X}}(x) = \arg\min_{y \in \mathcal{X}} \|x - y\|_2$ is the projection to $\mathcal{X}$. In week 2 we saw that a projected subgradient descent algorithm with $\|x\| \leq B$ and step size $\eta = \frac{B}{L\sqrt{t}}$ has bounded approximation error

$$f\left(\frac{1}{t}\sum_{s=1}^{t} x_s\right) - f(x^*) \leq \frac{BL}{\sqrt{t}}.$$

Example. Let $\mathcal{B}_n \subset X = \mathbb{R}^k$ denote the euclidean ball with unit radius and $f\colon \mathcal{B}_n \to \mathbb{R}$ a convex function with $\|\nabla f(x)\|_\infty \leq 1$. In the notation of Theorem 2.3.2 we have $L = \|\nabla f(x)\|_2 \leq \sqrt{n}$, $B = 1$ and therefore

$$f\left(\frac{1}{t}\sum_{s=1}^{t} x_s\right) - f(x^*) \leq \frac{LB}{\sqrt{t}} = \sqrt{\frac{n}{t}}.$$

Previously, we established that in case of boosting the STATISTICS term is of order $O(\sqrt{\log(n)/m})$. As discussed already in section 4.2, we would like to match this rate with an optimisation algorithm that scales with $\log(n)$ whereas projected subgradient descent scales with n. Similar to above, the solution lies in replacing the euclidean norm with a problem specific notion of distance. This can be achieved, for example, by describing the geometry of the problem with a different norm (and hence also a different notion of distance). This is the idea of the *mirror descent* algorithm.

However, as a caveat, note that replacing the norm in gradient descent is not entirely straightforward. To see this we briefly review some basics about derivatives on normed vector spaces. Denote X a normed vector space and $\|\cdot\|_X$ an associated norm. The *dual space* X^* of X is defined as the vector space of all linear maps $A\colon X \to \mathbb{R}$, equipped with the *operator* or *dual norm*

$$\|A\|_* := \sup\left\{|Ax|; x \in X, \|x\|_X = 1\right\},$$

which is the maximal stretch of the unit ball in X.

Remark 5.0.1. *If $X = \mathbb{R}^k$ and $\|\cdot\|$, then linear functionals can be written as a scalar product, i.e. for some $g \in \mathbb{R}^n$, we have $A(x) = g^\mathsf{T} x$ and the dual norm reads*

$$\|g\|_* := \|A\|_* := \sup\left\{|g^\mathsf{T} x|; x \in \mathbb{R}^n, \|x\| = 1\right\}.$$

Definition 5.0.1. *For general normed spaces $(X, \|\cdot\|_X)$ and $(Y, \|\cdot\|_Y)$, we say that an operator $T\colon U \to Y, U \subset X$ open is (Fréchet-)differentiable in some point $x \in U$ if there exists a bounded linear operator $D_T(x)\colon X \to Y$, such that*

$$\frac{\|T(x+h) - T(x) - D_T(x)h\|_Y}{\|h\|_X} \to 0$$

as $\|h\|_X \to 0$. The operator $D_T(x)$ is called the (Fréchet-)derivative of T in x.

It is immediate that for $f\colon X \to \mathbb{R}$ the operator $D_f(x)$ is an element of the dual X^*.

Going back to our gradient descent algorithm this means that in general the equation $x - \eta\nabla f(x)$ doesn't make sense, because $\nabla f(x)$ is an element of the dual and x is an element of the primal. Note that the exception in case of the euclidean norm arises because we can write linear functionals as an appropriate scalar product, see Remark 5.0.1. (More generally, the Riesz representation theorem implies that the dual of a Hilbert space is isometrically isomorph to its primal.)

In mirror descent allows us to circumvent this problem by mapping the current point of our descent algorithm to its dual, perform the gradient descent step and map back to our primal space. In general there is no guarantee that our new point in the primal space is in our restriction set $\mathcal{X}$ and, hence, an additional projection may be required. Summarised we get the following algorithm.

Procedure 5.0.1 Mirror Descent

```
Set $x_1 \in \arg\min_{x \in \mathcal{X} \cap \mathcal{C}} \Phi(x)$
For $t \geq 1$ find $y_{t+1}$ such that
    $\nabla\Phi(y_{t+1}) = \nabla\Phi(x_t) - \eta g_t, \quad g_t \in \partial f(x_t)$
    $x_{t+1} \in \Pi^\Phi_\mathcal{X}(y_{t+1}),$
```

where we used Φ to denote the *mirror map* that helps map x to the dual space and $\Pi^\Phi_\mathcal{X}$ denotes an appropriate projection. We will develop the steps in this algorithm more carefully in the following.

5.1 Mirror Maps

We need the following two concepts.

Definition 5.1.1 (Mirror Map)**.** *Let $\mathcal{D} \subset \mathbb{R}^n$ open and $\mathcal{X} \subset \overline{\mathcal{D}}$ with $\mathcal{X} \cap \mathcal{D} \neq \emptyset$. A* mirror map *is a map $\Phi\colon \mathcal{D} \to \mathbb{R}$ if the following properties hold*

i) Φ is strictly convex and differentiable.

ii) The gradient is a surjective map, i.e.

$$\begin{aligned}\nabla\Phi\colon \mathcal{D} &\to \mathbb{R}^n\\ x &\mapsto \nabla\Phi(x).\end{aligned}$$

is onto.

iii) The gradient diverges on the boundary

$$\lim_{x \to \partial \mathcal{D}} \|\nabla \Phi(x)\| = \infty$$

Definition 5.1.2 (Bregman Divergence)**.** *The Bregman divergence associated with a function f is defined as*

$$D_f(x, y) = f(x) - f(y) - \nabla f(y)^\mathsf{T}(x - y).$$

Hence, the Bregman divergence is measuring the "error" of the local linear approximation. We have the following identity

$$D_f(x, y) + D_f(z, x) - D_f(z, y) = \left(\nabla f(x) - \nabla f(y)\right)^\mathsf{T}(x - z)\,, \tag{5.1}$$

which follows directly collecting terms.

A note on these conditions. We will use the Bregman divergence to project $y \in \mathcal{D}$ back to $\mathcal{X}$, i.e. we

$$\Pi^\Phi_\mathcal{X}(y) = \arg\min_{x \in \mathcal{X} \cap \mathcal{D}} D_\Phi(x, y).$$

Condition $iii)$ implies the existence of this projection. In addition, this implies that $x \mapsto D_\Phi(x, y)$ is locally increasing on the boundary of $\mathcal{D}$. Together the compactness of $\mathcal{X}$ and the fact that $x \mapsto D_\Phi(x, y)$ is strictly convex also implies the uniqueness. Condition $ii)$ ensures that for every point in the dual $\mathcal{D}$ we need a preimage in $\mathcal{D}$.

5.2 Mirror Descent: Examples

We are now able to describe the algorithm in detail for two important examples. Our first example illustrates the generality of the algorithm and its connection to projected subgradient descent. With our second example we will go back to our boosting example and we will show in Section 5.3 that by applying this mirror descent with an appropriate mirror map, we will recover the promised $\log(n)$ rate, thus matching the approximation quality of the STATISTICS term.

Ball setup and projected subgradient descent

We can recover the original projected subgradient descent by considering $\mathcal{D}$ to be the whole of $\mathbb{R}^n$ and the mirror map half of the squared Euclidian norm, i.e.

$$\Phi(x) = \frac{1}{2}\|x\|_2^2$$

The associated Bregman divergence is

$$\begin{aligned} D_\Phi(x, y) &= \Phi(x) - \Phi(y) - \nabla \Phi(y)^\mathsf{T}(x - y) \\ &= \frac{1}{2}\|x\|_2^2 - \frac{1}{2}\|y\|_2^2 - y^\mathsf{T}x + y^\mathsf{T}y \\ &= \frac{1}{2}\|x - y\|_2^2 \end{aligned}$$

which implies that he Bregman projection coincides with our standard projection $\Pi_\mathcal{X}$.

Entropy setup and boosting I

Consider as the restriction set the simplex $\mathcal{X} = \Delta_n := \{x \in [0,1]^n : \sum_{i=1}^n x_i = 1\}$, i.e. discrete finite space probability measures or probability simplex. We choose the mirror map to be the negative entropy

$$\Phi(x) = \sum_{i=1}^n x_i \log x_i,$$

where $\mathcal{D} = \mathbb{R}^n_{>0} := \{x \in \mathbb{R}^n : x_i > 0, i = 1, \dots, n\}$. Further, $\nabla\Phi(x) = (1 + \log(x_i))_{i=1,\dots,n}$ so our update step becomes

$$\log(y_{i,s+1}) = \log(x_{i,s}) - \eta g_s$$

or equivalently

$$y_{i,s+1} = x_{i,s} \exp(-\eta g_s)$$

hence mirror descent here amounts to an "‘exponential gradient descent algorithm"’. The Bregman divergence is

$$\begin{aligned} D_\Phi(x, y) &= \Phi(x) - \Phi(y) - \nabla\Phi(y)^\mathsf{T}(x - y) \\ &= \sum_{i=1}^n x_i \log x_i - \sum_{i=1}^n y_i \log y_i - \sum_{i=1}^n \log(y_i)(x_i - y_i) - \sum_{i=1}^n (x_i - y_i) \\ &= \sum_{i=1}^n x_i \log\left(\frac{x_i}{y_i}\right) \end{aligned}$$

since $\sum_{i=1}^n (x_i - y_i) = 0$ on Δ_n. Hence, the Bregman divergence coincides with the relative entropy or Kullback-Leibler divergence. Moreover, the above calculations imply that Φ is 1-strongly convex with respect to $\|\cdot\|_1$

$$\begin{aligned} \Phi(x) - \Phi(y) - \nabla\Phi(y)^\mathsf{T}(x - y) &= \sum_{i=1}^n x_i \log\left(\frac{x_i}{y_i}\right) \\ &\geq \frac{1}{2}\left(\sum_{i=1}^n |x_i - y_i|\right)^2 \\ &= \frac{1}{2}\|x - y\|_1^2, \end{aligned}$$

where we used Pinsker's inequality

$$\|x - y\|_{\mathrm{tv}} = \frac{1}{2}\sum_{i=1}^n |x_i - y_i| \leq \sqrt{\frac{1}{2}\sum_{i=1}^n x_i \log\left(\frac{x_i}{y_i}\right)},$$

where $\|\cdot\|_{\mathrm{tv}}$ denotes the total variation norm on the space of probability measures. Furthermore, we show that the Bregman projection in this case amounts to renormalising the vector $y \mapsto x = \frac{y}{\|y\|_1}$ (unsurprisingly). Recall we want to minmimize

$$\Pi^\Phi_\mathcal{X}(y) = \operatorname*{arg\,min}_{x \in \mathcal{X} \cap \mathcal{D}} D_\Phi(y, x).$$

Note that Bregman divergence in not symmetrical so swapping the components might result in different behaviour. In this case we want to project by minimizing over the second coordinate and our input is y. Using Lagrange multipliers, minimizing the Kullback-Leibler divergence under the restriction Δ_n we have

$$\mathcal{L}(x) = \sum_{i=1}^n y_i \log\frac{y_i}{x_i} + \lambda \sum_{i=1} (x_i - 1).$$

The first order condition yields

$$\begin{aligned} \frac{y_i}{x_i} &= \lambda, \qquad i = 1, \dots, n \\ \sum_{i=1}^n x_i &= 1. \end{aligned}$$

Hence, $y_i = \lambda x_i$, $\sum_{i=1}^n y_i = \lambda$ and $x_i = y/\|y\|_1$.

5.3 Oracle complexity

We start with a useful Lemma.

Lemma 5.3.1. Let $x \in \mathcal{X} \cap \mathcal{D}$ and $y \in \mathcal{D}$, then we have the following two inequalities.

$$\left(\nabla\Phi\left(\Pi_{\mathcal{X}}^{\Phi}(y)\right) - \nabla\Phi(y)\right)^{\mathsf{T}}\left(\Pi_{\mathcal{X}}^{\Phi}(y) - x\right) \leq 0 \tag{5.2}$$

and as a consequence

$$D_{\Phi}(x, \Pi_{\mathcal{X}}^{\Phi}(y)) + D_{\Phi}(\Pi_{\mathcal{X}}^{\Phi}(y), y) \leq D_{\Phi}(x, y). \tag{5.3}$$

Proof. Note that for x, y

$$\begin{aligned} \nabla_x D_{\Phi}(x, y) &= \nabla_x\left(\Phi(x) - \Phi(y) - \nabla_y\Phi(y)^{\mathsf{T}}(x - y)\right) \\ &= \nabla_x\Phi(x) - \nabla_y\Phi(y). \end{aligned}$$

Hence, taking $x^* = \Pi_{\mathcal{X}}(y) \in \arg\min_{x \in \mathcal{X} \cap \mathcal{D}} D_{\Phi}(x, y)$, then by the first order optimality condition (Proposition 2.1.3) for any given y and all $z \in \mathcal{X}$

$$\begin{aligned} 0 &\geq \nabla_x D_{\Phi}(x^*, y)^{\mathsf{T}}(x^* - z) \\ &= \left(\nabla_x\Phi(x^*) - \nabla_y\Phi(y)\right)^{\mathsf{T}}(x^* - z), \end{aligned}$$

which shows (5.2). An application of the identity (5.1) yields

$$\left(\nabla_x\Phi(x^*) - \nabla_y\Phi(y)\right)^{\mathsf{T}}(x^* - y) = D_{\Phi}(x^*, y) + D_{\Phi}(z, x^*) - D_{\Phi}(z, y),$$

which shows (5.3). □

We are now able to prove the main result of this section.

Theorem 5.3.2. *Fix a norm $\|\cdot\|$ on $\mathbb{R}^n$. Let Φ denote a ρ-strongly convex mirror map on $\mathcal{X} \cap \mathcal{D}$. Denote f a Lipschitz function with Lipschitz constant L. Then mirror descent with $\eta = \frac{R}{L}\sqrt{\frac{2\rho}{t}}$ satisfies the following approximation guarantee*

$$f\left(\frac{1}{t}\sum_{s=1}^{t} x_s\right) - f(x^*) \leq RL\sqrt{\frac{2}{\rho t}},$$

where $R^2 = \sup_{x \in \mathcal{X} \cap \mathcal{D}} \Phi(x) - \Phi(x_1)$.

Proof. Let $x \in \mathcal{X} \cap \mathcal{D}$. Using the definition of the gradient step of mirror descent

$$\nabla\Phi(y_{t+1}) = \nabla\Phi(x_t) - \eta g_t$$

we get

$$\begin{aligned} f(x_s) - f(x) &\leq g_s^{\mathsf{T}}(x_s - x) \\ &= \frac{1}{\eta}\left(\nabla\Phi(x_s) - \nabla\Phi(y_{s+1})\right)^{\mathsf{T}}(x_s - x) \\ &= \frac{1}{\eta}\left(D_{\Phi}(x_s, y_{s+1}) + D_{\Phi}(x, x_s) - D_{\Phi}(x, y_{s+1})\right) \\ &\leq \frac{1}{\eta}\left(D_{\Phi}(x_s, y_{s+1}) + D_{\Phi}(x, x_s) - D_{\Phi}(x, x_{s+1}) - D_{\Phi}(x_{s+1}, y_{s+1})\right), \end{aligned} \tag{5.4}$$

where we used identity (5.1) in line 3 and inequality (5.3) applied to $D_\Phi(x, y_{s+1})$ in line 4. We bound the difference $D_\Phi(x_s, y_{s+1}) - D_\Phi(x_{s+1}, y_{s+1})$ as follows

$$\begin{aligned}
&D_\Phi(x_s, y_{s+1}) - D_\Phi(x_{s+1}, y_{s+1}) \\
&= \Phi(x_s) - \Phi(y_{s+1}) - \nabla\Phi(y_{s+1})^\mathsf{T}(x_s - y_{s+1}) - \Phi(x_{s+1}) + \Phi(y_{s+1}) + \nabla\Phi(y_{s+1})^\mathsf{T}(x_{s+1} - y_{s+1}) \\
&= \Phi(x_s) - \Phi(x_{s+1}) - \nabla\Phi(y_{s+1})^\mathsf{T}(x_s - x_{s+1}) \\
&= \Phi(x_s) - \Phi(x_{s+1}) - \nabla\Phi(x_s)^\mathsf{T}(x_s - x_{s+1}) + \left(\nabla\Phi(x_s)^\mathsf{T} - \nabla\Phi(y_{s+1})^\mathsf{T}\right)(x_s - x_{s+1}) \\
&\le \left(\nabla\Phi(x_s)^\mathsf{T} - \nabla\Phi(y_{s+1})^\mathsf{T}\right)(x_s - x_{s+1}) - \frac{\alpha}{2}\|x_s - x_{s+1}\|^2 \\
&= \eta g_s^\mathsf{T}(x_s - x_{s+1}) - \frac{\alpha}{2}\|x_s - x_{s+1}\|^2 \qquad (5.5) \\
&\le \eta\|g_s\|_* \|x_s - x_{s+1}\| - \frac{\alpha}{2}\|x_s - x_{s+1}\|^2 \\
&\le \frac{(\eta\|g_s\|_*)^2}{2\rho} \\
&\le \frac{(\eta L)^2}{2\rho},
\end{aligned}$$

where we used the inequality $az - bz^2 \le a^2/4b$ for all $z \in \mathbb{R}$ in the last line and α-strong convexity in line 4. We compute

$$\begin{aligned}
&\sum_{s=1}^{t} f(x_s) - f(x) \\
&\le \frac{1}{\eta}\left(\sum_{s=1}^{t} D_\Phi(x, x_s) - D_\Phi(x, x_{s+1}) + \frac{1}{\eta}\frac{(\eta L)^2}{2\rho}\right) \\
&= \frac{1}{\eta}\left(D_\Phi(x, x_1) - D_\Phi(x, x_{s+1})\right) + \frac{t}{\eta}\frac{(\eta L)^2}{2\rho} \\
&\le \frac{D_\Phi(x, x_1)}{\eta} + \frac{t\eta L^2}{2\rho}, \qquad (5.6)
\end{aligned}$$

where we used that

$$D_\Phi(x, x_{s+1}) = \Phi(x) - \Phi(x_{s+1}) - \nabla\Phi(x_{s+1})^\mathsf{T}(x - x_{s+1}) \ge 0.$$

Note that by construction $D_\Phi(x, x_1) \le R^2$. Thus, plugging in $\eta = \frac{R}{L}\sqrt{\frac{2\rho}{t}}$ we get

$$\begin{aligned}
(5.6) &\le \frac{R^2}{\frac{R}{L}\sqrt{\frac{2\rho}{t}}} + \frac{R}{L}\sqrt{\frac{2\rho}{t}}\frac{tL^2}{2\rho} \\
&= LR\sqrt{\frac{t}{2\rho}} + LR\sqrt{\frac{t}{2\rho}} \\
&= LR\sqrt{\frac{2t}{\rho}}.
\end{aligned}$$

This implies the required result. □

Remark 5.3.1. *The inequality*

$$g_s^\mathsf{T}(x_s - x_{s+1}) \le L\|x_s - x_{s+1}\|$$

that we used in the above proof is somewhat crucial in our case because L bounds the dual or operator norm, i.e. $L \geq \|g\|_* = \sup\{g^\top x : \|x\| = 1\}$. Using the fact that for $1 \leq p, q \leq \infty$ and $\frac{1}{p} + \frac{1}{q} = 1$ the norms ℓ^p and ℓ^q are dual, we see that we can get much finer control by adjusting the norm we are using to the problem hat hand. Seen from a different point, we can now use any appropriate form of Hölder's inequality and are not confined to using Cauchy-Schwarz.

Remark 5.3.2. *If we do not use the previous inequality in the proof we get that for any $\eta > 0$*

$$\eta g_s^\top (x_s - x) \leq \frac{\eta^2}{2\rho}\|g_s\|_*^2 + D_\Phi(x, x_s) - D_\Phi(x, x_{s+1}).$$

We will use this fact in the next section, when we explain linear coupling.

Entropy setup and boosting II

We are finally able to show how to achieve the $\log(n)$ rate in the case of boosting. Note, that for the negative entropy mirror map $\Phi(x) = \sum_{i=1}^n x_i \log x_i$ and $x \in \Delta_n$

$$-\log(n) \leq \sum_{i=1}^n x_i \log x_i \leq 0$$

and hence $R^2 = \sup_{x \in \mathcal{X} \cap \mathcal{D}} \Phi(x) - \Phi(x_1) = \log(n)$. We already established that Φ is 1-strongly convex with respect to $\|\cdot\|_1$.

Corollary 5.3.3. Let f be a convex function on Δ_n with

$$\|g\|_\infty \leq L, \quad \forall g \in \partial f(x), \forall x \in \Delta_n.$$

Then, mirror descent with $\Phi(x) = \sum_{i=1}^n x_i \log x_i$ and $\eta = \frac{R}{L}\sqrt{\frac{2\rho}{t}} = \frac{1}{L}\sqrt{\frac{2\log(n)}{t}}$ yields

$$f\left(\frac{1}{t}\sum_{s=1}^t x_s\right) - f(x^*) \leq L\sqrt{\frac{2\log(n)}{t}}.$$

Proof. Apply Theorem 5.3.2. □

In order to achieve a predefined bound $\epsilon > 0$ we thus neeed to take $t \geq \frac{2L^2\log(n)}{\epsilon^2}$. Recall the optimisation problem in boosting

$$\begin{aligned} \text{minimize} \quad & R(x) = \frac{1}{m}\sum_{i=1}^m \varphi(-Y_i x^T \Phi(W_i)) \\ \text{subject to} \quad & x \in \Delta_m. \end{aligned}$$

In this case our gradient is

$$g(x) = \nabla R(x) = \frac{1}{m}\sum_{i=1}^m \varphi'\left(Y_i x^\top \Phi(W_i)\right)\Phi(W_i)(-Y_i).$$

By construction we obtain $\|\Phi\|_\infty \leq 1$, $\|Y_i\|_\infty \leq 1$ and thus $\|g\|_\infty \leq L_\varphi$, the Lipschitz constant of φ.

Chapter 6

Acceleration by coupling gradient descent and mirror descent

We have seen as a consequence of Theorem 2.3.5 that if f is β-smooth then projected gradient descent needs $T = O\left(\frac{\beta R^2}{\varepsilon}\right)$ iterations to obtain an ε-minimizer. However, we derived in Theorem 3.2.2 a lower bound in which the dependence on β and ε was $O\left(\sqrt{\frac{\beta}{\varepsilon}}\right)$. Nesterov in [20] designed a gradient descent method whose complexity matches this lower bound. This method only works for $\|\cdot\|_2$. Later, in [19] he generalized his method to allow arbitrary norms. Algorithms whose complexity is optimal are said to be accelerated. Nesterov's accelerated gradient descent has always been regarded as an obscure and unintuitive method whose proof uses "magical" algebra tricks. Since he published his seminal work in 1983, several authors have proposed other accelerated methods, in order to give more intuition about acceleration and to show this from other points of view. For example, in [13] we can find a geometric interpretation of acceleration. Linear Coupling [4] (2014) is one of these methods. We think that it is one of the best methods to understand acceleration. We start giving some intuition about how gradient and mirror descent can be combined to obtain these accelerated method. We present first a simplified version of linear coupling that also achieves acceleration but under more restrictive assumptions and finally, we present linear coupling. This method works for any norm and for each of them, the analysis of Linear Coupling is the same. We will only work with convex,, β-smooth functions f, we note that [4], the paper in which this section is based, contains other analysis like the α-strictly convex and β-smooth case.

6.1 Intuition

To understand why combining gradient and mirror descent makes sense and why doing so is a good idea we will note some properties about both methods. As we have been doing in previous sections we use the notation x^* for a minimizer of f restricted to $\mathcal{X}$. We have seen in 2.3.5 that projected gradient descent is defined by

$$\begin{aligned} y_{k+1} &= x_k - \frac{1}{\beta} g_k, \text{ where } g_k \in \partial f(x_k) \\ x_{k+1} &= \Pi_{\mathcal{X}}(y_{k+1}). \end{aligned}$$

One should not be surprised that in the case that $\mathcal{X} = \mathbb{R}^n$ regular gradient descent in unconstrained optimization is defined by the rule $x_{k+1} = x_k - \frac{1}{\beta} g_k$. But, why is gradient descent, constrained or not, defined in that way? Of course, it works and it provides a non trivial convergence rate, but the proofs of this rate usually only define the method and prove convergence without explaining where the method comes from, specially regarding the choice of the learning rate. It can be useful to see gradient descent from the following point of view. Let's do it first for unconstrained gradient descent with $\|\cdot\|_2$ only. If we are at point $x_k \in \mathbb{R}^n$ and we compute the next point by moving against the gradient, the choice of $\frac{1}{\beta}$ is the best choice for the learning rate, in the sense that we can guarantee maximal local decrease for that choice of the learning rate. This is not difficult to see using the assumptions we have at hand. The smoothness assumption tells us that along the line defined by x_k and $\nabla f(x_k)$, f is lower bounded by a parabola with leading coefficient $\frac{\beta}{2}$ (blue graph in Figure 6.1) whose derivative in x_k coincides with the one of f (restricted to the line, i.e. it is $\|\nabla f(x_k)\|$). Therefore, the derivative of $x^2\beta/2$ is $x\beta = \|\nabla f(x_k)\|$ so the distance to the minimum in the parabola is $\mathtt{x} = \frac{1}{\beta}\|\nabla f(x_k)\|$ and it is clear now that maximal guaranteed progress is the evaluation of $x^2\beta/2$ at $\mathtt{x}$, i.e. $\frac{1}{2\beta}\|\nabla f(x_k)\|^2$. And we have just proved that the guaranteed decrease is maximal for that choice of the learning rate and we have computed how much. All these arguments can be written using inequalities, but hopefully this can be considered cleaner by some people. Proving the rate of convergence of gradient descent given the guaranteed progress at each step is straightforward. With this picture in mind, it is also very easy to derive the rate of convergence of gradient descent in the case that f is also μ-strongly convex. Since this assumption lower bounds f by another parabola with leading coefficient $\mu/2$, we can see that the guaranteed progress is proportional to $1/2\beta$ and $f(x_k) - f(x^*)$ is upper bounded by something proportional to $1/2\mu$ so after one step the value $f(x_k) - f(x^*)$ decreases to at least $(f(x_k) - f(x^*))\left(1 - \frac{\|\nabla f(x_k)\|/2\beta}{\|\nabla f(x_k)\|/2\mu}\right) = (f(x_k) - f(x^*))\left(1 - \frac{\mu}{\beta}\right)$. And therefore $f(x_T) - f(x^*) \leq (f(x_0) - f(x^*))\left(1 - \frac{\mu}{\beta}\right)^T$, so an ε-minimizer is found in $O\left((f(x_0) - f(x^*))\log\frac{1}{\varepsilon}\right)$ iterations.

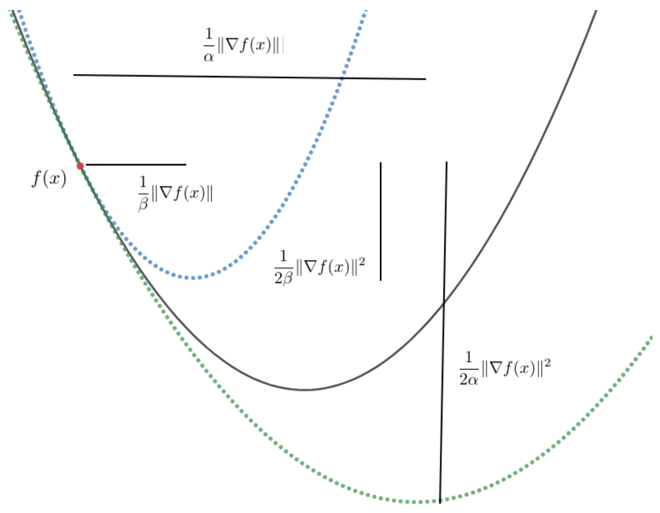
$\frac{1}{\alpha}\|\nabla f(x)\|$
$f(x)$
$\frac{1}{\beta}\|\nabla f(x)\|$
$\frac{1}{2\beta}\|\nabla f(x)\|^2$
$\frac{1}{2\alpha}\|\nabla f(x)\|^2$

Figure 6.1: Visualization of the smoothness bound (blue) and the strong convexity bound (green) of a function f (black).

We make two important remarks about the previous analysis. Firstly, gradient descent uses the assumption of β-smoothness to guarantee maximal local decrease and secondly, the guaranteed decrease is better if the norm of the gradient is large. We will note later that the regret of mirror descent is lower when the norm of the gradient is low, and this, along with the second remark, is what we can leverage to combine gradient and mirror descent. But let's focus first on the first remark. The maximal decrease on the objective we can guarantee from x_k occurs when we minimize, as we did before in our toy example, the bound that is given by the β-smoothness assumption, which is

$$f(y) \leq f(x_k) + \langle \nabla f(x_k), y - x_k \rangle + \frac{\beta}{2}\|y - x_k\|^2,$$

for every $y \in \mathcal{X}$. Note that for enough regular functions, the smoothness condition can be derived by upper bounding a second order multivariate Taylor expansion using that the Hessian's eigenvalues are upper bounded by β. It is a simple way to remember the inequality. So we can define the next point as

$$\begin{aligned} x_{k+1} &:= \arg\min_{y\in\mathcal{X}} \left\{ f(x_k) + \langle \nabla f(x_k), y - x_k \rangle + \frac{\beta}{2}\|y - x_k\|^2 \right\} \\ &= \arg\min_{y\in\mathcal{X}} \left\{ \langle \nabla f(x_k), y - x_k \rangle + \frac{\beta}{2}\|y - x_k\|^2 \right\}. \end{aligned} \tag{6.1}$$

If we take $\|\cdot\|_2$ and $\mathcal{X} = \mathbb{R}^n$ we are searching for the minimizer in a quadratic function ay^ty+b^ty+c, $(a \in \mathbb{R}; b, c \in \mathbb{R}^n)$ which is $-\frac{b}{2a}$ or in our case

$$-\frac{\nabla f(x_k) - 2x_k\beta/2}{2\beta/2} = x_k - \frac{1}{\beta}\nabla f(x_k).$$

which matches our previous analysis. Maybe the following, for a general convex set $\mathcal{X}$, is more interesting (we subtract constant terms inside the arg min's):

$$\arg\min_{y\in\mathcal{X}} \left\{ \left\| \left(x_k - \frac{1}{\beta}\nabla f(x_k) \right) - y \right\|^2 \right\} \overset{?}{=} \arg\min_{y\in\mathcal{X}} \left\{ \langle \nabla f(x_k), y - x_k \rangle + \frac{\beta}{2}\|y - x_k\|^2 \right\}$$

$$\Leftrightarrow \arg\min_{y\in\mathcal{X}} \left\{ \langle y, y \rangle - 2\left\langle y, \left(x_k - \frac{1}{\beta}\nabla f(x_k) \right) \right\rangle \right\} \overset{?}{=} \arg\min_{y\in\mathcal{X}} \left\{ \langle \nabla f(x_k), y \rangle + \frac{\beta}{2}(\langle y, y \rangle - 2\langle x_k, y \rangle) \right\}.$$

It is clear that the two arg min's of the last expression are the same, since we can obtain the left hand side by dividing by $\frac{\beta}{2}$ in the arg min of the right hand side. This means that our rule for projected gradient descent (left hand side) computes the point in $\mathcal{X}$ whose decrease guarantee given by the β-smoothness of f is maximal among all the points in $\mathcal{X}$ (right hand side). It is natural now to define gradient descent for general norms by rule (6.1). We denote

$$\texttt{Prog}(x) := -\min_{y\in\mathcal{X}} \left\{ \langle \nabla f(x), y - x \rangle + \frac{\beta}{2}\|y - x\|^2 \right\} \geq 0.$$

By the definition of x_{k+1} it is clear that $f(x_{k+1}) \leq f(x_k) - \texttt{Prog}(x)$ (and $\texttt{Prog}(x) = \frac{1}{2\beta}\|\nabla f(x)\|_*^2$ if $\mathcal{X} = \mathbb{R}^n$). We will call $\texttt{Grad}(x)$ to the arg min of the previous expression.

In short, we can say that **gradient descent at each iteration maximizes the guaranteed local decrease**.

Our second remark was that with $\mathcal{X} = \mathbb{R}^n$ and $\|\cdot\|_2$ the decrease is better if $\|\nabla f(x)\|$ is larger. An intuition that we will formalize later is that mirror descent for $\mathcal{X} = \mathbb{R}^n$ and $\|\cdot\|_2$ suffers from a small loss if $\|\nabla f(x)\|$ is small. In general we will prove that a bound for the mirror descent loss is going to have a term that will be easy to control and something proportional to $\texttt{Prog}(x)$. So when mirror descent suffers from a large loss, gradient descent decreases the objective a lot, and when gradient descent does not have a large guaranteed decrease, mirror descent's loss will be small. This is the key idea of linear coupling.

We saw last week that mirror descent tackles the dual optimization problem by constructing lower bounds to the optimum. Recall that each queried gradient $\nabla f(x)$ can be viewed as a hyperplane lower bounding the objective f, that is, $f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle$ for all y. Mirror-descent methods attempt to carefully construct a convex combination of these hyperplanes in order to yield even a stronger lower bound. From this point of view our claimed intuition about mirror descent having a small loss when $\|\nabla f(x)\|_2$ is small should be clear, because the planes are closer to being horizontal. We will denote

$$\texttt{Mirr}_{x_k}(\xi) := \arg\min_{x\in\mathcal{X}} \left\{D(x, x_k) + \langle \xi, x - x_k\rangle\right\}.$$

Note that a mirror descent step is $x' = \texttt{Mirr}_{x_k}(\eta\partial f(x_k))$. Here, D is the Bregman divergence of a mirror map Φ which for simplicity we will assume that is 1-strongly convex. [1]

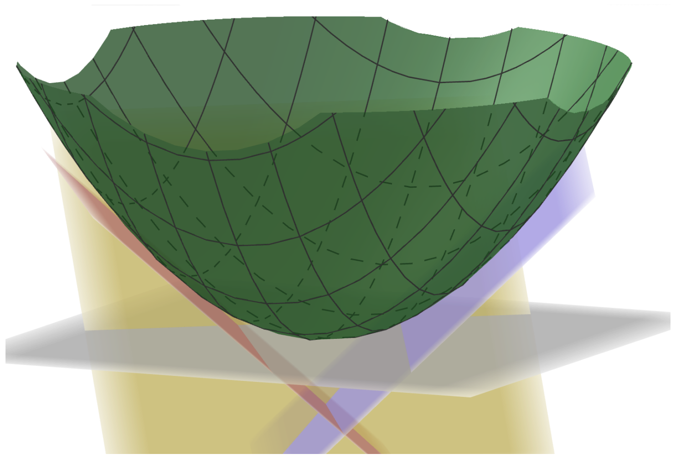

Figure 6.2: Mirror combines several linear lower bounds obtained by convexity to give a lower bound on the optimum.

Thought experiment. For the sake of demonstrating the idea, suppose $\|\nabla f(x)\|_2$, the norm of the observed gradient, is **either** always $\geq K$, or always $\leq K$, where K will be determined later. Under such "wishful assumption", we can propose the following algorithm: if the norm of the gradient is always $\geq K$ perform T gradient descent steps. Otherwise perform T mirror descent steps. To analyze such an algorithm, suppose without loss of generality we start with some point x_0 whose objective distance $f(x_0) - f(x^*)$ is at most 2ε, and we want to find some x so that $f(x) - f(x^*) \leq \varepsilon$.

If T gradient descent steps are performed, the objective decreases by at least $\frac{\|\nabla f(\cdot)\|_2^2}{2\beta} \geq \frac{K^2}{2\beta}$ per step and we only need $T \geq \Omega\left(\frac{\varepsilon\beta}{K^2}\right)$ steps to achieve an ε accuracy. If T mirror descent steps are performed, we need $T \geq \Omega\left(\frac{K^2}{\varepsilon^2}\right)$ steps according to the mirror descent convergence (Remark 5.3.2). In sum, we need $T \geq \Omega\left(\max\left\{\frac{\varepsilon\beta}{K^2}, \frac{K^2}{\varepsilon^2}\right\}\right)$ steps to converge to an ε-minimizer. Setting K to be the magic number to balance the two terms, we only need $T \geq \Omega\left(\sqrt{\frac{\beta}{\varepsilon}}\right)$ iterations. This means that in the general case in which $f(x_0) - f(x^*) \leq d$ we can repeat this procedure to obtain points such that the distances of their images to $f(x^*)$ halve after each iteration. So we only need $T \geq \Omega\left(\sqrt{\frac{\beta}{\varepsilon}}\left(\sqrt{\frac{2\varepsilon}{d}} + \sqrt{\frac{4\varepsilon}{d}} + \cdots + \sqrt{\frac{1}{2}} + 1\right)\right) = \Omega\left(\sqrt{\frac{\beta}{\varepsilon}}\right)$.

[1] Note that this is equivalent to the mirror descent step in Algorithm 5.0.1. We defined x_{k+1} to be in $\Pi^{\Phi}_{\mathcal{X}}(y_{k+1}) = \arg\min_{x\in\mathcal{X}} D(x, y_{k+1}) = \arg\min_{x\in\mathcal{X}} \{\Phi(x) - \Phi(y_{k+1}) - \langle\nabla\Phi(y_{k+1}), x\rangle\} = \arg\min_{x\in\mathcal{X}} \{\Phi(x) - \langle\nabla\Phi(x_k) - \eta\nabla f(x_k), x\rangle\}$ where in the last step we have removed constant terms from the arg min and used the definition of $\nabla\Phi(y_{k+1})$. On the other hand $\arg\min_{x\in\mathcal{X}}\{D_(x, x_k) + \langle\eta\nabla f(x_k), x - x_k\rangle\} = \arg\min_{x\in\mathcal{X}}\{\Phi(x) - \Phi(x_k) - \langle\nabla\Phi(x_k), x - x_k\rangle + \langle\eta\nabla f(x_k), x - x_k\} = \arg\min_{x\in\mathcal{X}} \{\Phi(x) - \langle\nabla\Phi(x_k) - \eta\nabla f(x_k), x\rangle\}$ where in the last step we have also removed constant terms. We observe that thus the two expressions lead to the same result. However, the second one (or the simplified version of both for that matter) does not need to obtain y_{k+1} from $\nabla\Phi(y_{k+1})$.

6.2 Warm-Up Method with Fixed Step Length

The key ideas of this method will be the same as the ones for the final method. As we mentioned previously, at each iteration linear coupling performs a gradient descent step and a mirror descent step and the point for the next iteration is a convex combination of the points obtained by gradient and mirror descent. The key of the proof is to see that the loss in which mirror descent incurs is something proportional to the gradient descent guaranteed decrease (plus a couple of Bregman divergences that will telescope) and using this fact to see that a particular convex combination of mirror and gradient descent incurs in a similar loss.

Formally, we start with $x_0 = y_0 = z_0$. In each iteration $k = 0, 1, \dots, T-1$ we define $x_{k+1} \leftarrow \tau z_k + (1-\tau) y_k$ and then we perform:

- A gradient step $y_{k+1} \leftarrow \texttt{Grad}(x_{k+1})$.
- A mirror step $z_{k+1} \leftarrow \texttt{Mirr}_{z_k}(\eta \nabla f(x_{k+1}))$.

The choices of η and τ will become clear at the end of this section, but from a high level η is determined by the mirror descent analysis, similarly to what we saw last week, and τ will be determined as the best parameter to balance gradient and mirror descent, similar to K in the previous thought experiment.

The two key ideas mentioned before materialize in the following two lemmas:

Lemma 6.2.1. For every $u \in \mathcal{X} = \mathbb{R}^n$,

$$\begin{aligned}\eta\langle \nabla f(x_{k+1}), z_k - u\rangle &\le \frac{\eta^2}{2}\|\nabla f(x_{k+1})\|_*^2 + D(u, z_k) - D(u, z_{k+1}) \\ &\le \eta^2 \beta (f(x_{k+1}) - f(y_{k+1})) + D(u, z_k) - D(u, z_{k+1}).\end{aligned} \tag{6.2}$$

Proof. The first inequality is Remark 5.3.2, which is the key lemma of the mirror descent analysis. The second inequality is from the gradient step guarantee $f(x_{k+1}) - f(y_{k+1}) \ge \frac{1}{2\beta}\|\nabla f(x_{k+1})\|_*^2$. □

The second lemma bounds the actual regret at x_{k+1} and picks τ so that we can telescope the sum of the regrets.

Lemma 6.2.2 (Coupling). Letting $\tau \in (0,1)$ satisfy that $\frac{1-\tau}{\tau}\eta\beta$, we have that

$$\forall u \in \mathcal{X} = \mathbb{R}^n, \quad \eta\langle \nabla f(x_{k+1}), x_{k+1} - u\rangle \le \eta^2\beta\left(f(y_k) - f(y_{k+1})\right) + \left(D(u, z_k) - D(u, z_{k+1})\right).$$

Proof. We can compute easily the difference between the regrets at x_{k+1} and at z_k using the definition of x_{k+1}.

$$\begin{aligned}\eta\langle \nabla f(x_{k+1}), x_{k+1} - u\rangle - \eta\langle \nabla f(x_{k+1}), z_k - u\rangle &= \eta\langle \nabla f(x_{k+1}), x_{k+1} - z_k\rangle \\ = \frac{(1-\tau)\eta}{\tau}\langle \nabla f(x_{k+1}), y_k - x_{k+1}\rangle &\le \frac{(1-\tau)\eta}{\tau}(f(y_k) - f(x_{k+1})).\end{aligned} \tag{6.3}$$

Above, we used the fact that $\tau(x_{k+1} - z_k) = (1-\tau)(y_k - x_{k+1})$, as well as the convexity of f. It is immediate that by choosing $\frac{1-\tau}{\tau} = \eta\beta$ and combining (6.2) and (6.3) we have the result. □

If we telescope 6.2.2 for $k = 0, 1, \dots, T-1$ and setting $\bar{x} := \frac{1}{T}\sum_{k=0}^{T-1} x_k$ and $u = x^*$, we have

$$\eta T\left(f(\bar{x}) - f(x^*)\right) \le \sum_{k=0}^{T-1} \eta\langle \nabla f(x_k), x_k - x^*\rangle \le \eta^2\beta\left(f(y_0) - f(y_T)\right) + D(x^*, x_0) - D(x^*, x_T). \tag{6.4}$$

Suppose our initial point is of error at most d, that is $f(y_0) - f(x^*) \le d$, and suppose $D(x^*, x_0) \le \Theta$, then (6.4) gives $f(\bar{x}) - f(x^*) \le \frac{1}{T}\left(\eta\beta d + \Theta/\eta\right)$. Choosing $\eta = \sqrt{\Theta/(\beta d)}$ to be the value that balances the above two terms (it is essentially the same way of choosing η in mirror descent), we obtain that $f(x) - f(x^*) \le \frac{\sqrt{\beta\Theta d}}{T}$. In other words, in $T = 4\sqrt{\beta\Theta/d}$ steps, we can obtain some $\bar{x}$ satisfying $f(\bar{x}) - f(x^*) \leq d/2$, halving the distance to the optimum. If we restart this entire entire procedure a few number of times, halving the distance for every run, then we obtain an ε-approximate solution in

$$T = O\left(\sqrt{\beta\Theta/\varepsilon} + \sqrt{\beta\Theta/2\varepsilon} + \sqrt{\beta\Theta/4\varepsilon} + \dots\right) = O\left(\sqrt{\beta\Theta/\varepsilon}\right)$$

iterations. The value of $\eta = \sqrt{\Theta/(\beta d)}$ increases as time goes and therefore $\tau = \frac{1}{\eta\beta+1}$ decreases as time goes. This tells us that gradient descent is given more weight the mirror step when we are close to the minimum. This fact can be counter-intuitive because when it is closer to the optimum, the observed gradients will become smaller, and therefore mirror steps should perform well according to the thought experiment. This understanding is incorrect, the reason is that when it is closer to the optimum, the threshold between large and small gradients also becomes smaller, so one cannot rely only on mirror steps.

6.3 Final Method with Variable Step Lengths

This final method will change η and τ gradually to obtain an algorithm that does not need to know the bounds Θ and d. This approach will also work for any convex restriction set $\mathcal{X}$. The final algorithm is the following:

Procedure 6.3.1 AGM (f, Φ, x_0, T)

```
Input: $f$ a differentiable and convex function on $\mathcal{X}$ that is $\beta$-smooth with respect to $\|\cdot\|$, $\Phi$ the $DGF$ function that is 1-strongly convex with
    respect to the same $\|\cdot\|$ over $\mathcal{X}$. $x_0$ some initial point and $T$ the number of iterations.
Output: $y_T$ such that $f(y_T) - f(x^*) \leq \frac{4\Theta\beta}{T^2}$.
1: $D(y,x) := \Phi(y) - \langle \nabla\Phi(x), y - x\rangle - \Phi(x)$.
2: $y_0 \leftarrow x_0$,   $z_0 \leftarrow x_0$.
3: for $k \leftarrow 0$ to $T-1$ do
4:     $\eta_{k+1} \leftarrow \frac{k+2}{2\beta}$, and $\tau_k \leftarrow \frac{1}{\eta_{k+1}\beta} = \frac{2}{k+2}$
5:     $x_{k+1} \leftarrow \tau_k z_k + (1-\tau_k) y_k$.
6:     $y_{k+1} \leftarrow \mathtt{Grad}(x_{k+1})$          $\left(:= \arg\min_{y\in\mathcal{X}} \left\{\frac{\beta}{2}\|y - x_{k+1}\|^2 + \langle \nabla f(x_{k+1}), y - x_{k+1}\rangle\right\}\right)$
7:     $z_{k+1} \leftarrow \mathtt{Mirr}_{z_k}(\eta_{k+1}\nabla f(x_{k+1}))$          $(:= \arg\min_{z\in\mathcal{X}} \{D(z,z_k) + \langle \eta_{k+1}\nabla f(x_{k+1}), z - z_k\rangle\})$
8: end for
9: return $y_T$
```

We will prove that this method is accelerated. In particular, we have the following theorem.

Theorem 6.3.1. *If $f(x)$ is β-smooth w.r.t. $\|\cdot\|$ on $\mathcal{X}$, and $\Phi(x)$ is 1-strongly convex w.r.t. $\|\cdot\|$ on $\mathcal{X}$, then Algorithm 6.3.1 outputs y_T satisfying $f(y_T) - f(x^*) \leq 4\Theta\beta/T^2$, where Θ is any upper bound on $D(x^*, x_0)$.*

We will use two lemmas, that are the analogous to Lemma 6.2.1 and Lemma 6.2.2, that we will prove after the proof of the theorem.

Lemma 6.3.2. If $\tau_k = \frac{1}{\eta_{k+1}\beta}$, then linear coupling satisfies that for every $u \in \mathcal{X}$,

$$\begin{aligned}\eta_{k+1}\langle \nabla f(x_{k+1}), z_k - u\rangle &\leq \eta_{k+1}^2\beta\mathtt{Prog}(x_{k+1}) + (D(u,z_k) - D(u,z_{k+1})) \\ &\leq \eta_{k+1}^2\beta\left(f(x_{k+1}) - f(y_{k+1})\right) + (D(u,z_k) - D(u,z_{k+1}))\,.\end{aligned}$$

Lemma 6.3.3 (Coupling). For any $u \in \mathcal{X}$,

$$\left(\eta_{k+1}^2\beta\right) f(y_{k+1}) - \left(\eta_{k+1}^2\beta - \eta_{k+1}\right) f(y_k) + (D(u,z_{k+1}) - D(u,z_k)) \leq \eta_{k+1} f(u).$$

Proof of Theorem 6.3.1. In order to telescope Lemma 6.3.3 we only need to set the sequence of η_k so that $\eta_k^2\beta \approx \eta_{k+1}^2\beta - \eta_{k+1}$ as well as $\tau_k = 1/\eta_{k+1}\beta \in (0,1]$. In Algorithm 6.3.1 $\eta_k = \frac{k+1}{2\beta}$ so that $\eta_k^2\beta = \eta_{k+1}^2\beta - \eta_{k+1} + \frac{1}{4\beta}$.

Summing up Lemma 6.3.3 for $k = 0, 1, \ldots, T-1$, we obtain

$$\eta_T^2 \beta f(y_T) + \sum_{k=1}^{T-1} \frac{1}{4\beta} f(y_k) + (D(u, z_T) - D(u, z_0)) \le \sum_{k=1}^{T} \eta_k f(u).$$

By choosing $u = x^*$, we notice that $\sum_{k=1}^{T} \eta_k = \frac{T(T+3)}{4\beta}$, $f(y_k) \ge f(x^*)$, $D(u, z_T) \ge 0$ and $D(x^*, z_0) \le \Theta$. Therefore, we obtain

$$\frac{(T+1)^2}{4\beta^2} \beta f(y_T) \le \left(\frac{T(T+3)}{4\beta} - \frac{T-1}{4\beta} \right) f(x^*) + \Theta,$$

which after simplification implies $f(y_T) \le f(x^*) + \frac{4\Theta\beta}{(T+1)^2}$. □

Proof of Lemma 6.3.2. The second inequality of the lemma is again by the gradient descent guarantee $f(x_{k+1}) - f(y_{k+1})) \ge \texttt{Prog}(x_{k+1})$. To prove the first one, we first write down the key inequality of mirror-descent analysis (whose proof is identical to the one given in the previous section).

$$\begin{aligned}
\eta_{k+1} \langle \nabla f(x_{k+1}), z_k - u \rangle &= \langle \eta_{k+1} \nabla f(x_{k+1}), z_k - z_{k+1} \rangle + \langle \eta_{k+1} \nabla f(x_{k+1}), z_{k+1} - u \rangle \\
&\le \langle \eta_{k+1} \nabla f(x_{k+1}), z_k - z_{k+1} \rangle + \langle -\nabla D(u, z_k), z_{k+1} - u \rangle \\
&= \langle \eta_{k+1} \nabla f(x_{k+1}), z_k - z_{k+1} \rangle + D(u, z_k) - D(u, z_{k+1}) + D((z_{k+1}, z_k) \\
&\le \left(\langle \eta_{k+1} \nabla f(x_{k+1}), z_k - z_{k+1} \rangle - \frac{1}{2} \|z_k - z_{k+1}\|^2 \right) + (D(u, z_k) - D(u, z_{k+1}))
\end{aligned}$$

The first inequality is due to the minimality of $z_{k+1} = \arg\min_{z \in \mathcal{X}} \{ D(u, z_k) + \langle \eta_{k+1} \nabla f(x_{k+1}), z \rangle \}$, which implies that $\langle \nabla D(z_{k+1}, z_k) + \eta_{k+1} \nabla f(x_{k+1}), u - z_{k+1} \rangle \ge 0$ for all $u \in \mathcal{X}$. The second inequality is because $D(y, x) \ge \frac{1}{2} \|x - y\|^2$ by the strong convexity of the $\Phi(\cdot)$. If we apply Cauchy-Schwarz to the first summand of the last expression above and the fact that $az - bz^2 \le \frac{a^2}{4b}$, $\forall z \in \mathbb{R}$ we get

$$\left(\langle \eta_{k+1} \nabla f(x_{k+1}), z_k - z_{k+1} \rangle - \frac{1}{2} \|z_k - z_{k+1}\|^2 \right) \le (\eta_{k+1} \|\nabla f(x_{k+1})\|_*) \|z_k - z_{k+1}\| - \frac{1}{2} \|z_k - z_{k+1}\|^2 \le \frac{\eta_{k+1}^2}{2} \|\nabla f(x_{k+1})\|_*^2$$

and thus we have the result for $\mathcal{X} = \mathbb{R}^n$. For the general constrained we need to use the special choice of $\tau_k = 1/\eta_{k+1}\beta$ as follows. Letting $v := \tau_k z_{k+1} + (1 - \tau_k) y_k \in \mathcal{X}$ so that $x_{k+1} - v = (\tau_k z_k + (1 - \tau_k) y_k) - v = \tau_k (z_k - z_{k+1})$, we have

$$\begin{aligned}
&\langle \eta_{k+1} \nabla f(x_{k+1}), z_k - z_{k+1} \rangle - \frac{1}{2} \|z_k - z_{k+1}\|^2 \\
=& \left\langle \frac{\eta_{k+1}}{\tau_k} \nabla f(x_{k+1}), x_{k+1} - v \right\rangle - \frac{1}{2\tau_k^2} \|x_{k+1} - v\|^2 \\
=& \eta_{k+1}^2 \beta \left(\langle \nabla f(x_{k+1}), x_{k+1} - v \rangle - \frac{\beta}{2} \|x_{k+1} - v\|^2 \right) \le \eta_{k+1}^2 \beta \texttt{Prog}(x_{k+1})
\end{aligned}$$

where the last inequality is from the definition of $\texttt{Prog}(x_{k+1})$.

□

Proof of Lemma 6.3.3. We can derive the lemma from the following inequalities

$$\begin{aligned}
&\eta_{k+1}(f(x_{k+1}) - f(u)) \\
&\leq \eta_{k+1}\langle \nabla f(x_{k+1}), x_{k+1} - u\rangle \\
&= \eta_{k+1}\langle \nabla f(x_{k+1}), x_{k+1} - z_k\rangle + \eta_{k+1}\langle \nabla f(x_{k+1}), z_k - u\rangle \\
&= \frac{(1-\tau_k)\eta_{k+1}}{\tau_k}\langle \nabla f(x_{k+1}), y_k - x_{k+1}\rangle + \eta_{k+1}\langle \nabla f(x_{k+1}), z_k - u\rangle \\
&\leq \frac{(1-\tau_k)\eta_{k+1}}{\tau_k}(f(y_k) - f(x_{k+1})) + \eta_{k+1}\langle \nabla f(x_{k+1}), z_k - u\rangle \\
&\leq \frac{(1-\tau_k)\eta_{k+1}}{\tau_k}(f(y_k) - f(x_{k+1})) + \eta_{k+1}^2\beta(f(x_{k+1}) - f(y_{k+1})) + D(u, z_k) - D(u, z_{k+1}) \\
&= (\eta_{k+1}^2\beta - \eta_{k+1})f(y_k) - (\eta_{k+1}^2\beta)f(y_{k+1}) + \eta_{k+1}f(x_{k+1}) + (D(u, z_k) - D(u, z_{k+1}))
\end{aligned} \tag{6.5}$$

where the lines $4, 5, 6$ and 7 are obtained, respectively, by (4) the choice of x_{k+1} that satisfies $\tau_k(x_{k+1} - z_k) = (1-\tau_k)(y_k - x_{k+1})$; (5) is by the convexity of f and $1 - \tau_k \geq 0$; (6) uses Lemma 6.3.2 and (7) uses the choice of $\tau_k = 1/\eta_{k+1}\beta$. □

Conclusion. Linear coupling describes a family of methods whose oracle complexity matches the lower bound that we had proved for these kinds of algorithms. However, this is not the end of the story. Linear coupling provides intuition about acceleration as any other method before. The authors of the paper advocate thinking about any kind of first order optimization problem in terms of linear coupling. Provided one proves similar lemmas for the guaranteed progress of the primal approach (gradient descent) and for the loss of the dual approach (mirror descent) for your particular problem, one can try to use linear coupling to obtain a fast algorithm. Since Linear Coupling was published, there have been several proposals of algorithms to solve optimization problems whose complexity is lower than the one of the best previous known algorithm. Some of these are: the first accelerated stochastic gradient descent algorithm [1], a faster algorithm for matrix scaling [3], a faster algorithm of accelerated coordinate descent using non-uniform sampling [7], nearly-linear time packing and covering LP solvers [5], an algorithm to solve positive LPs faster in parallel [6], an algorithm for non-convex optimization faster than stochastic gradient descent [2].

Chapter 7

Non-Euclidean setting: Frank-Wolfe

Speaker: Chris Carmona, 23/10/2017.

The *Conditional Gradient Descent* method, also known as *Frank-Wolfe* is an iterative first-order optimization algorithm for constrained convex optimization, originally proposed in [15] by Marguerite Frank and Philip Wolfe, then post-docs working in the Princeton logistics project led by Kuhn and Tucker. In recent years, Frank-Wolfe-type methods have re-gained interest, started by the valuable insights provided by [16], and fuelled by its good scalability and convergence properties properties (eg. convergence invariance under affine transformations).

The method addresses general constrained convex optimization problems of the form $\min_{x\in\mathcal{X}} f(x)$. It assumes that the objective function f is convex and continuously differentiable, and that the domain $\mathcal{X}$ is a compact convex subset of any vector space. Procedure 7.0.1 describes the method. In each iteration, the algorithm considers a linear approximation of the objective function, and moves towards a minimizer of this linear function, taken over the domain $\mathcal{X}$. Figure 7.1 illustrates one iteration of the algorithm (the domain $\mathcal{X}$ is denoted by $\mathcal{D}$ here).

Procedure 7.0.1 Conditional Gradient Descent (Frank-Wolfe)

Input: f a differentiable and convex function on $\mathcal{X}$ that is β-smooth w.r.t. some norm $\|\cdot\|$.

Output: x_T such that $f(x_T) - f(x^*) \leq \frac{4\beta R^2}{T+1}$ with $R = \sup_{x,y\in\mathcal{X}} \|x-y\|$

```
1: Let $x_0 \in \mathcal{X}$
2: for $t \in 0, \dots, T$ do
3:     Set $\gamma_t = \frac{2}{t+2}$
4:     Compute
           $s_t := \arg\min_{s\in\mathcal{X}} \langle s, \nabla f(x_t)\rangle$     (7.1)
5:     Update $x_{t+1} := (1-\gamma_t)x_t + \gamma_t s_t = x_t + \gamma_t(s_t - x_t)$
6: end for
7: return $x_T$
```

From a computational perspective, a key property of this scheme is that it replaces the projection step of projected gradient descent by a linear optimization over $\mathcal{X}$, which in some cases can be a much simpler problem.

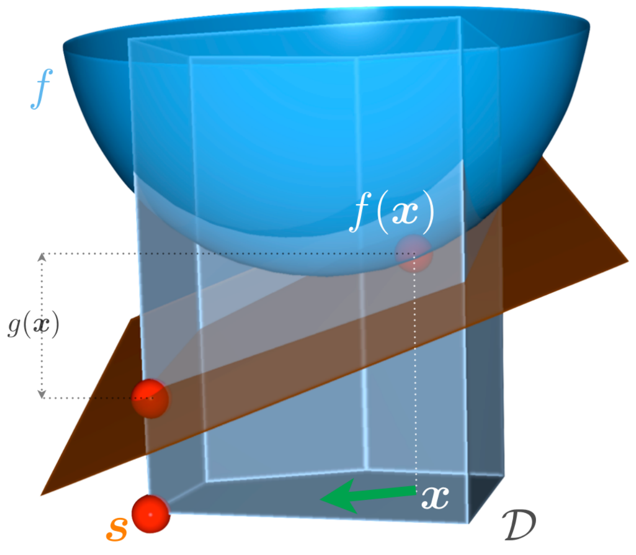

Figure 7.1: Graphical representation of the Conditional Gradient Descent algorithm [16]

7.1 Convergence analysis.

A major advantage of conditional gradient descent over projected gradient descent is that the former can adapt to smoothness in an arbitrary norm.

The following theorem establishes a bound of convergence. Note that the number of iterations needed to have $f(x_t) - f(x^*) \le \varepsilon$ is $O(\frac{1}{\varepsilon})$.

Theorem 7.1.1. *Let f be a convex and β-smooth function w.r.t. some norm $\|\cdot\|$, $R = \sup_{x,y\in\mathcal{X}} \|x-y\|$, and $\gamma_t = \frac{2}{t+1}$ for $t \ge 1$. Then for any $t \ge 2$, one has:*

$$f(x_t) - f(x^*) \le \frac{4\beta R^2}{t+1} \tag{7.2}$$

Proof. First, note that the following inequalities hold:

$$\begin{aligned}
f(x_{t+1}) - f(x_t) &\le \nabla f(x_t)^\top (x_{t+1} - x_t) + \frac{\beta}{2}\|x_{t+1} - x_t\|^2 && (f \text{ is } \beta\text{-smooth}) \\
&\le \gamma_t \nabla f(x_t)^\top (s_t - x_t) + \frac{\beta}{2}\gamma_t^2 R^2 && (x_{t+1} = x_t + \gamma(s_t - x_t)) \\
&\le \gamma_t \nabla f(x_t)^\top (x^* - x_t) + \frac{\beta}{2}\gamma_t^2 R^2 && (\text{by definition } s_t \le x^*) \\
&\le \gamma_t (f(x^*) - f(x_t)) + \frac{\beta}{2}\gamma_t^2 R^2 && (\text{convexity of } f)
\end{aligned}$$

If we define $\delta_t = f(x_t) - f(x^*)$ then we get:

$$\delta_{t+1} \le (1-\gamma_t)\delta_t + \frac{\beta}{2}\gamma_t^2 R^2 \tag{7.3}$$

Finally, the result is proved by induction using $\gamma_t = \frac{2}{t+1}$. We initialize at step 2, $t = 1$, with the above inequality yielding $\delta_2 \leq \frac{\beta}{2} R^2$. $\square$

7.2 Sparse iterates property.

In addition to being projection-free and "norm-free", the conditional gradient descent satisfies a perhaps even more important property: it produces *sparse iterates*.

Consider the situation where $\mathcal{X} \subset \mathbb{R}^n$ is a polytope, that is the convex hull of a finite set of points (these points are called the vertices of $\mathcal{X}$). Then Caratheodory's theorem states that any point $x \in \mathcal{X}$ can be written as a convex combination of at most $n+1$ vertices of $\mathcal{X}$. On the other hand, by definition of the conditional gradient descent, one knows that the t-th iterate x_t can be written as a convex combination of t vertices (assuming that x_1 is a vertex). Thanks to the dimension-free rate of convergence one is usually interested in the regime where $t << n$, and thus we see that the iterates of conditional gradient descent are very sparse in their vertex representation.

7.3 Application: Regularized least-squares regression.

The three properties of conditional gradient descent (projection-free, norm-free, and sparse iterates) are critical to develop a computationally efficient procedure in solving this kind of problems.

Consider the regularized least-squares regression problem (which includes the LASSO). We would like to approximate a signal $Y \in \mathbb{R}^n$ by using a combination of few of the columns $x_j; j \in 1, \ldots, p$ in the matrix $X \in \mathbb{R}^{n \times p}$. Then, for a fixed λ we have the penalized problem in dimension p:

$$\min_{\beta \in \mathbb{R}^p} \|Y - X\beta\|_2^2 + \lambda \|\beta\|_1$$

Instead of considering the penalized version of the problem one could look at the following constrained problem, with $s \in \mathbb{R}$ fixed:

$$\min_{\beta \in \mathbb{R}^p} \|Y/s - X\beta\|_2^2 \tag{7.4}$$
$$\text{subject to} \|\beta\|_1 \leq 1$$

We are interested in situations where $n << p$, that is the number of variables p in X can be very large, potentially exponential in the dimension n. Nonetheless we want to restrict our attention to algorithms that run in reasonable time with respect to n, that is we want polynomial time algorithms in n. Of course in general this is impossible, and we need to assume that the dictionary has some structure that can be exploited. Here we make the assumption that one can solve in time $p(n)$ (where p is polynomial) the following problem for any $y \in \mathbb{R}^n$:

$$\min_{1 \leq i \leq p} y^\top x_i$$

Finally, for normalization issues, we assume that the ℓ_2-norm of the x_i's are controlled by some $m > 0$, that is $\|x_j\|_2 \leq m, \forall j \in 1, \ldots, p$

Our problem of interest 7.4 corresponds to minimizing the function $f(\beta) = \frac{1}{2}\|Y - X\beta\|_2^2$ on the ℓ_1-ball of $\mathbb{R}^p$ in polynomial time in n. At first sight this task may seem completely impossible indeed one is not even allowed to write down entirely a vector $\beta \in \mathbb{R}^b$ (since this would take time linear in p). The key property that will save us is that this function admits sparse minimizers as we discussed above, and this will be exploited by the conditional gradient descent method.

Let us study the computational complexity of the t-th step of conditional gradient descent. First, observe that:

$$\nabla f(\beta) = X^\top (X\beta - Y)$$

Now assume that $z_t = X\beta_t - Y \in \mathbb{R}^n$ is already computed, then to compute the update 7.1 one needs to find the coordinate $i_t \in 1, \dots, p$ that maximizes $|[\nabla f(\beta_t)](i)|$ which can be done by maximizing $x_i^\top z_t$ and $-x_i^\top z_t$. Thus the update 7.1 takes time $O(p(n))$

$$x_{t+1} = (1 + \gamma_t)x_t + \gamma_t \cdot sign(\nabla f(x_t)_{i_t}) \cdot e_{i_t} \tag{7.5}$$

Computing β_{t+1} from β_t and i_t takes time $O(t)$ since $\|\beta_t\|_0 \leq t$, and computing z_{t+1} from z_t and i_t takes time $O(n)$. Thus the overall time complexity of running t steps is (we assume $p(n) = \Omega(n)$)

$$O(tp(n) + t^2) \tag{7.6}$$

To derive a rate of convergence it remains to study the smoothness of f . This can be done as follows:

$$\begin{aligned}\|\nabla f(\beta) - \nabla f(\gamma)\|_\infty &= \|X^\top X(\beta - \gamma)\|_\infty \\ &= \max_{1 \leq j \leq p} |x_j^\top \sum_{k=1}^{p} x_k(\beta(k) - \gamma(k))| \\ &\leq m^2 \|\beta - \gamma\|_1\end{aligned} \tag{7.7}$$

which means that f is m^2-smooth with respect to the ℓ_1-norm.

Thus we get the following rate of convergence:

$$f(\beta_t) - f(\beta^*) \leq \frac{8m^2}{t+1} \tag{7.8}$$

Putting together 7.6 and 7.8 we proved that one can get an ε-optimal solution to 7.4 with a computational effort of $O(m^2 p(n)/\varepsilon + m^4/\varepsilon^2)$ using the conditional gradient descent.

7.4 Frank-Wolfe variants.

Line search.

Instead of using the pre-defined step-sizes $\gamma = \frac{2}{k+2}$ a modified algorithm the best point on the line segment between the current iterate x(k) and s.

$$\gamma_t = \arg\min_{\gamma \in [0,1]} f(x_{t-1} + \gamma(s_{t-1} - x_{k-1}))$$

Fully corrective.

This is a harder-working variant of the Frank-Wolfe method, which after the addition of a new *atom* (or search direction) s re-optimizes the objective f over all previously used atoms. The change consists on computing x_{t+1} using a convex combination of all the previous atoms s_t. This is:

$$x_{t+1} = \arg\min_{x \in \text{conv}(s_0, \dots, s_t)} f(x) \tag{7.9}$$

Comparing to the original Frank-Wolfe method, the idea is that the variant here will hopefully make more progress per iteration, and therefore result in iterates x being combinations of even fewer atoms (i.e. better sparsity). This however comes at a price, namely that the internal problem in each iteration can now become as hard to solve as the original optimization problem, implying that no global run-time guarantees can be given for the algorithm in general.

Away steps.

Another important variant is the use of *away-steps.* The idea is that in each iteration, we not only add a new atom s, but potentially also remove an old atom (provided it is bad with respect to our objective). This requires that the iterate x is represented as a convex combination of the current atoms. This variant can improve the sparsity of the iterates. Using away-steps, a faster linear convergence can be obtained for some special problem class.

Inexact updates.

Jaggi (2011) also analyzes inexact Frank-Wolfe updates.

Suppose we choose s_t with some "small" inaccuracy so that:

$$\langle s_t, \nabla f(x_t)\rangle = \min_{s\in\mathcal{X}} \langle s, \nabla f(x_t)\rangle + \delta\frac{\gamma_{t+1}C_f}{2} \tag{7.10}$$

where δ is our *inaccuracy parameter.* C_f is known as the *curvature constant*, measures how far f is from being linear, getting $C_f = 0$ when f is linear, and defined as:

$$C_f := \sup_{\substack{x,s\in\mathcal{X},\\ \gamma\in[0,1],\\ y=x+\gamma(s-x)}} \frac{2}{\gamma^2}(f(y) - f(x) - \nabla f(x)^\top (y-x)) \tag{7.11}$$

In [16] it is shown that the attained rate of convergence is basically the same under this *inexact updates.*

Theorem 7.4.1. *The Conditional Gradient Method using fixed step* $\gamma_t = \frac{2}{t+2}$ *sizes and inaccuracy parameter* $\delta \geq 0$, *satisfies*

$$f(x_t) - f(x^*) \leq \frac{2C_f}{t+2}(1+\delta) \tag{7.12}$$

Chapter 8

Stochastic oracle model

Speaker: Adam Foster, 30/11/2017.

Assume that we want to solve the following problem:

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \mathcal{X}, \end{array} \tag{8.1}$$

where f is a convex function, possibly not known. In previous work, we assumed that we had a first order oracle which, given $x \in \mathcal{X}$, would yield a $g \in \partial f(x)$. When analysing our algorithms in terms of oracle complexity, it was the *number of calls* to the oracle that was important. In many applications, an oracle may not exist, or a single call to the oracle may be very expensive. In such cases, we accept randomness in our oracle but require it to be unbiased.

Formally, a first order *stochastic* oracle model defines the framework where given $x \in \mathcal{X}$ an oracle yields back a random variable G that is an unbiased estimator of a subgradient of f at x, namely, $\mathbf{E}[G] \in \partial f(x)$. If X is a random variable, the oracle yields back a random variable G that is an unbiased estimator of a subgradient of f at X conditionally on X, namely, $\mathbf{E}[G|X] \in \partial f(X)$.

8.1 Projected gradient descent with a stochastic oracle

We analyse the behaviour of the projected gradient descent algorithm to solve problem (8.1) when we replace exact knowledge of subgradients of f with unbiased estimates of them. For a given initial point $X_1 \in \mathcal{X}$, possibly random, and a given collection of step sizes $\eta_1, \eta_2, \ldots$, the stochastic projected gradient descent is defined by the sequence of random variables generated according to the following update:

$$\begin{aligned} Y_{s+1} &= X_s - \eta_s G_s, \\ X_{s+1} &= \Pi_{\mathcal{X}}(Y_{s+1}). \end{aligned}$$

where G_s is obtained from a stochastic oracle with $\mathbf{E}[G_s|X_s] \in \partial f(X_s)$.

Remarkably, for f a L-Lipschitz function we can recover the rate of convergence of the deterministic oracle (cf. Theorem 2.3.2). When f is additionally α-strongly convex we recover the unaccelerated rate of convergence of [13] Theorem 3.9.

Theorem 8.1.1 (L-Lipschitz Stochastic Projected Subgradient Descent)**.** *Let f be convex. Suppose $\mathcal{X}$ is contained in a Euclidean ball of radius B centred at X_1. Suppose that for every $X \in \mathcal{X}$ the stochastic oracle yields an unbiased* *estimator of the subgradient of f at X bounded in L_2, namely, $\mathbf{E}[G|X] \in \partial f(X)$ and $\mathbf{E}[\|G\|^2] \leq L^2$. Then, the projected subgradient descent method with $\eta_s \equiv \eta = \frac{B}{L\sqrt{t}}$ satisfies*

$$\mathbf{E}f\left(\frac{1}{t}\sum_{s=1}^{t} X_s\right) - f(x^*) \leq \frac{LB}{\sqrt{t}}. \tag{8.2}$$

Proof. For any $1 \leq s \leq t$ we have

$$f(X_s) - f(x^*) \leq \mathbf{E}[G_s|X_s]^T(X_s - x^*) = \mathbf{E}[G_s^T(X_s - x^*)|X_s].$$

Proceeding as in the proof of Theorem 2.3.2, we find

$$G_s^T(X_s - x^*) \leq \frac{1}{2\eta}\left(\|X_s - x^*\|^2 - \|X_{s+1} - x^*\|^2\right) + \frac{\eta}{2}\|G_s\|^2.$$

Taking the expectation, we get

$$\mathbf{E}f(X_s) - f(x^*) \leq \mathbf{E}[G_s^T(X_s - x^*)] \leq \frac{1}{2\eta}\left(\mathbf{E}\|X_s - x^*\|^2 - \mathbf{E}\|X_{s+1} - x^*\|^2\right) + \frac{\eta}{2}\mathbf{E}[\|G_s\|^2],$$

and using the assumption $\mathbf{E}[\|G_s\|^2] \leq L^2$ we get

$$\frac{1}{t}\sum_{s=1}^{t}(\mathbf{E}f(X_s) - f(x^*)) \leq \frac{1}{2\eta t}\left(\mathbf{E}\|X_1 - x^*\|^2 - \mathbf{E}\|X_{t+1} - x^*\|^2\right) + \frac{\eta}{2}L^2 \leq \frac{B^2}{2\eta t} + \frac{\eta L^2}{2}.$$

Selecting $\eta = \frac{B}{L\sqrt{t}}$ to minimize the right-hand side of the above inequality gives the first result in (2.4) (since $f\left(\frac{1}{t}\sum_{s=1}^{t} X_s\right) \leq \frac{1}{t}\sum_{s=1}^{t} f(X_s)$ by Jensen's inequality). □

Theorem 8.1.2 (α-strongly convex Stochastic Projected Subgradient Descent)**.** *Let f be α-strongly convex. Suppose that for every $X \in \mathcal{X}$ the stochastic oracle yields an unbiased estimator of the subgradient of f at X bounded in L_2, namely, $\mathbf{E}[G|X] \in \partial f(X)$ and $\mathbf{E}[\|G\|^2] \leq L^2$. Then, the projected subgradient descent method with $\eta_s = \frac{2}{\alpha(s+1)}$ satisfies*

$$\mathbf{E}f\left(\sum_{s=1}^{t} \frac{2s}{t(t+1)} X_s\right) - f(x^*) \leq \frac{2L^2}{\alpha(t+1)}. \tag{8.3}$$

Proof. For any $1 \leq s \leq t$ we have

$$f(X_s) - f(x^*) \leq \mathbf{E}[G_s|X_s]^T(X_s - x^*) - \frac{\alpha}{2}\|X_s - x^*\|^2 = \mathbf{E}\left[G_s^T(X_s - x^*) - \frac{\alpha}{2}\|X_s - x^*\|^2|X_s\right].$$

Proceeding as before, we find

$$G_s^T(X_s - x^*) - \frac{\alpha}{2}\|X_s - x^*\|^2 \leq \left(\frac{1}{2\eta_s} - \frac{\alpha}{2}\right)\|X_s - x^*\|^2 - \frac{1}{2\eta_s}\|X_{s+1} - x^*\|^2 + \frac{\eta_s}{2}\|G_s\|^2.$$

Taking the expectation and applying $\mathbf{E}[\|G_s\|^2] \leq L^2$ gives

$$\mathbf{E}f(X_s) - f(x^*) \leq \left(\frac{1}{2\eta_s} - \frac{\alpha}{2}\right)\mathbf{E}[\|X_s - x^*\|^2] - \frac{1}{2\eta_s}\mathbf{E}[\|X_{s+1} - x^*\|^2] + \frac{\eta_s L^2}{2}.$$

Multiplying by s and using $\eta_s = \frac{2}{\alpha(s+1)}$ we get

$$s(\mathbf{E}f(X_s) - f(x^*)) \le \frac{\alpha}{4}s(s-1)\mathbf{E}[\|X_s - x^*\|^2] - \frac{\alpha}{4}s(s+1)\mathbf{E}[\|X_{s+1} - x^*\|^2] + \frac{L^2}{\alpha},$$

which upon summing over s leads to

$$\sum_{s=1}^{t} s(\mathbf{E}f(X_s) - f(x^*)) \le \frac{tL^2}{\alpha} + (\text{telescopes}).$$

By multiplying through by $\frac{2}{t(t+1)}$ and applying Jensen's inequality to f we have (8.3). $\square$

Theorem 8.1.1 shows that, in expectation, the stochastic projected subgradient descent method yields the same convergence guarantees as the deterministic counterpart analysed in Theorem 2.3.2. In particular, the oracle complexity is the same[1]: to get an accuracy ϵ, both methods requires $O(1/\epsilon^2)$ calls to their respective oracles. The main advantage of the stochastic version lies in the fact that in some applications the *computational* complexity involved in having access to a stochastic oracle is much cheaper than in the deterministic case. We now show that the stochastic model yields substantial computational saving in machine learning.

8.2 Expected risk minimization and empirical risk minimization revisited

Let us recall the original problem that motivates us in Section 1, namely, the expected risk minimization:

$$\begin{aligned} \text{minimize} \quad & r(x) = \mathbf{E}\ell(x^T\Phi(W), Y) \\ \text{subject to} \quad & x \in \mathcal{X}. \end{aligned}$$

The assumption here is that we know the loss function ℓ (indeed, we can choose it!) and the constraint set $\mathcal{X}$, but we do not know the distribution of (W, Y). We only have access to m i.i.d. samples $(W_1, Y_1), \ldots, (W_m, Y_m)$ from this unknown distribution.

With the concept of a stochastic oracle in hand, one can now attack the expected risk minimization problem directly because $G \in \partial R_i$ is an unbiased estimator of the subgradient of r where $R_i(x) = \ell(x^T\Phi(W_i), Y_i)$. Using this method, one can hope to solve the problem exactly. However, to ensure unbiasedness it is necessary to make only a *single* gradient step with each independent (W_i, Y_i), leading to a single pass through the data. Such an approach is well-suited to an online learning algorithm in which a data is processed and discarded in a stream.

Alternatively, one could minimize the empirical risk $R(x) = \frac{1}{m}\sum R_i(x)$ (see (1.2)). Proposition 1.0.2 shows that the error can be controlled by breaking the resulting error into $STATISTICS$ and $OPTIMIZATION$ terms. Whilst R and its subgradients may be tractable, such computations are typically $O(m)$. Stochastic oracle methods can reduce this to $O(1)$. One could select $I \sim \text{Uniform}(1, ..., m)$ and use $G \in \partial R_I$ as an unbiased estimator of the subgradient of R. Since unlimited i.i.d. samples of I are available, multiple passes over the data can be made. The key difference is that the functions being optimized in the two cases are different.

8.3 Single pass over the data

In the expected risk minimization problem we do not know the function r that we want to minimize, and in particular we can not operate in the deterministic first order oracle model discussed in the previous weeks as for a given $x \in \mathcal{X}$

[1]Note that different notions of accuracy are used, however, as we consider the expected value of the stochastic method.

we do not have access to a subgradient in $\partial r(x)$. On the other hand, given $x \in \mathcal{X}$ we can have access to an unbiased estimator of a subgradient of r evaluated at x. In fact, given the function $x \mapsto R_i(x) := \ell(x^T\Phi(W_i), Y_i)$, which is known to us, we can compute a subgradient of R_i at x. This is a random variable $G_i \in \partial R_i(x)$ that satisfies $\mathbf{E}G_i \in \partial r(x)$. The same is true if we want to get an estimate of the subgradient of r at $X \in \mathcal{X}$ when X is a random variable *independent* of (W_i, Y_i). In fact, in this case we can evaluate a subgradient of R_i at X, and this random variable $G_i \in \partial R_i(X)$ satisfies $\mathbf{E}[G_i|X] \in r(X)$.[2] Hence, we satisfy the assumption of the first order stochastic oracle model. At the same time, as we have m independent data points at out disposal, we can have access to at most m independent unbiased estimators of subgradients evaluated at possibly different locations in $\mathcal{X}$. In other words, the requirement of independence restricts us to a *single pass* over the data, which is not as general as in the stochastic block model defined above where we can have how many queries to the oracle as we want.

It is easy to check that under the assumptions of Proposition 1.1.2, Theorem 8.1.1 yields the following convergence guarantees for the stochastic projected descent method:

$$\mathbf{E}r(\hat{X}_m) - r(x^*) \le \frac{GL_{\text{loss}}B}{\sqrt{m}},$$

with $\hat{X}_m = \frac{1}{m}\sum_{i=1}^m X_i$. The computational savings are clear. If computing a subgradient for each functions R_i costs $O(1)$, then the computational complexity of stochastic gradient descent to achieve precision ϵ (in expectation) is of order $O(1/\epsilon^2)$. On the other hand, if we apply the deterministic gradient descent method to minimize the empirical risk R up to precision $1/\sqrt{m}$, as discussed in Section 1 and in Remark 1.2.1, then we see that the computational complexity is $O(m/\epsilon^2)$, as we need $O(1/\epsilon^2)$ calls to the deterministic oracle but each call costs $O(m)$ base iterations, as $R(x) := \frac{1}{m}\sum_{i=1}^m R_i(x)$.

8.4 Multiple passes over the data

In the unconstrained empirical risk minimization problem

$$\text{minimize} \qquad R(x) = \frac{1}{m}\sum_{i=1}^m R_i(x)$$

we assume henceforth that the R_i are β-smooth and that R is α-strongly convex. Let $\kappa = \beta/\alpha$ which is typically very large. The problem is amenable to basic gradient descent

$$x_{t+1} = x_t - \frac{\eta_t}{m}\sum_{i=1}^m \nabla R_i(x),$$

and stochastic gradient descent

$$x_{t+1} = x_t - \eta_t \nabla R_{I_t}(x),$$

where I_t are i.i.d Uniform$(1, ..., m)$. In [13] Theorem 3.10 it is shown that basic gradient descent can achieve an exponential rate $O(m\kappa\log(1/\epsilon))$ whereas Theorem 8.1.2 showed that stochastic gradient descent achieves $O(1/(\alpha\epsilon))$.

This motivates Stochastic Variance Reduced Gradient descent (SVRG) algorithm. This is a stochastic oracle method which achieves $O((m+\kappa)\log(1/\epsilon))$.

The main idea of the algorithm is to introduce a reference y and to compute the deterministic gradient $\nabla R(y)$. In place of $\nabla R_I(x)$ one uses $\nabla R_I(x) - \nabla R_I(y) + \nabla R(y)$. Lemma 8.4.1 is a preliminary result. The main algorithm is presented in Procedure 8.4.1.

[2]Note that if X is random, then there are two sources of randomness in G_i: one source is X itself, the other is the data point (W_i, Y_i). The statement $\mathbf{E}[G_i|X] \in \partial r(X)$ holds if X and (W_i, Y_i) are independent.

Lemma 8.4.1. Let $f_1, ..., f_m$ be β-smooth convex functions on $\mathbb{R}^n$ and $I \sim \text{Uniform}(1, ..., m)$. Then for all $x \in \mathbb{R}^n$

$$\mathbf{E}\|\nabla f_I(x) - \nabla f_I(x^*)\|^2 \leq 2\beta(f(x) - f(x^*))$$

Proof. Let

$$g_I(x) = f_I(x) - f_I(x^*) - \nabla f_I(x^*)^T(x - x^*) \geq 0$$

by convexity. Note g_I is a β-smooth function. For β-smooth g, it can be shown that

$$g\left(x - \frac{1}{\beta}\nabla g(x)\right) - g(x) \leq -\frac{1}{2\beta}\|\nabla g(x)\|^2,$$

and since $g_I \geq 0$ we can discard $g_I\left(x - \frac{1}{\beta}\nabla g_I(x)\right)$ to obtain

$$-g_I(x) \leq -\frac{1}{2\beta}\|\nabla g_I(x)\|^2.$$

From the definition of g_I this gives

$$\|\nabla f_I(x) - \nabla f_I(x^*)\|^2 \leq 2\beta\left(f_I(x) - f_I(x^*) - \nabla f_I(x^*)^T(x - x^*)\right).$$

When we take expectations with respect to I we note that $\mathbf{E}\nabla f_I(x^*) = 0$. The result follows.

□

Procedure 8.4.1 Stochastic Variance Reduced Gradient descent (SVRG)

Input: R is an α-strongly convex average of β-smooth convex functions $R_1, ..., R_m$.

```
1: Let $y^{(1)} \in \mathbb{R}^n$ be an arbitrary initial point
2: for $s = 1, 2, ...$ do
3:     $x_1^{(s)} = y^{(s)}$
4:     for $t = 1, ..., T$ do
5:         $I_t^{(s)} \sim \text{Uniform}(1, ..., m)$ independently
6:         $x_{t+1}^{(s)} = x_t^{(s)} - \eta\left(\nabla R_{I_t^{(s)}}(x_t^{(s)}) - \nabla R_{I_t^{(s)}}(y^{(s)}) + \nabla R(y^{(s)})\right)$
7:     end for
8:     $y^{(s+1)} = \frac{1}{T}\sum_{t=1}^{T} x_t^{(s)}$
9: end for
```

We now show that the errors in SVRG decrease exponentially.

Theorem 8.4.2 (Stochastic Variance Reduced Gradient descent). *Let $R_1, ..., R_m$ be β-smooth convex functions on $\mathbb{R}^n$ and R be α-strongly convex. Let $0 < \delta < 1$. Then SVRG with $\eta = \delta/2\beta$ and $T = 4\kappa/\delta$ satisfies*

$$\mathbf{E}R(y^{(s+1)}) - R(x^*) \leq \left(\frac{1 + 2\delta}{2(1 - \delta)}\right)^s \left(R(y^{(1)}) - R(x^*)\right)$$

Proof. Fix an epoch $s \geq 1$.

Aim

$$\mathbf{E}[R(y^{(s+1)})|y^{(s)}] - R(x^*) \leq \left(\frac{1 + 2\delta}{2(1 - \delta)}\right)(R(y^{(s)}) - R(x^*)) \tag{8.4}$$

which gives the main result by Tower Law and induction.

Recall

$$y^{(s+1)} = \frac{1}{T}\sum_{t=1}^{T} x_t^{(s)}$$

To simplify notation, we omit the s in the rest of the proof.

To begin, it follows from our assumptions on R and $R_1, , ..., R_m$ and [13] Theorem 3.10 that

$$\|x_{t+1} - x^*\|^2 = \|x_t - x^*\|^2 - 2\eta v_t^T(x_t - x^*) + \eta^2\|v_t\|^2 \tag{8.5}$$

where

$$v_t = \nabla R_{I_t}(x_t) - \nabla R_{I_t}(y) + \nabla R(y)$$

We now upper bound $\mathbf{E}_{I_t}\|v_t\|^2$ as follows

$$\begin{aligned}\mathbf{E}_{I_t}\|v_t\|^2 &\le 2\mathbf{E}\|\nabla R_{I_t}(x_t) - \nabla R_{I_t}(x^*)\|^2 + 2\mathbf{E}\|\nabla R_{I_t}(y) - \nabla R_{I_t}(x^*) + \nabla R(y)\|^2 \\ &\quad \text{Triangle inequality and } (a+b)^2 \le 2(a^2+b^2) \\ &\le 2\mathbf{E}\|\nabla R_{I_t}(x_t) - \nabla R_{I_t}(x^*)\|^2 + 2\mathbf{E}\|\nabla R_{I_t}(y) - \nabla R_{I_t}(x^*)\|^2 \\ &\quad \text{Since } \mathbf{E}\|X - \mathbf{E}X\|^2 \le \mathbf{E}\|X\|^2 \text{ and } \nabla R(y) \text{ is the expectation} \\ &\le 4\beta(R(x_t) - R(x^*) + R(y) - R(x^*)) \\ &\quad \text{by Lemma 8.4.1}\end{aligned}$$

Also by convexity

$$\mathbf{E}_{I_t} v_t^T(x_t - x^*) = \nabla R(x_t)^T(x_t - x^*) \ge R(x_t) - R(x^*),$$

when we plug this into (8.5) along with the previous result we obtain

$$\mathbf{E}_{I_t}\|x_{t+1} - x^*\|^2 \le \|x_t - x^*\|^2 - 2\eta(1 - 2\beta\eta)(R(x_t) - R(x^*)) + 4\beta\eta^2(R(y) - R(x^*)).$$

We now sum this inequality over $t = 1, ..., T$ to give

$$\mathbf{E}\|x_{T+1} - x^*\|^2 \le \|x_1 - x^*\|^2 - 2\eta(1 - 2\beta\eta)\mathbf{E}\sum_{t=1}^{T}(R(x_t) - R(x^*)) + 4\beta\eta^2 T(R(y) - R(x^*)).$$

We also have $x_1 = y$ and by α-strong convexity of R, $R(x) - R(x^*) \le \frac{\alpha}{2}\|x - x^*\|^2$. Applying this along with Jensen's Inequality then gives (upon rearranging)

$$\mathbf{E}R\left(\frac{1}{T}\sum_{t=1}^{T} x_t\right) - R(x^*) \le \left(\frac{1}{\alpha\eta(1 - 2\eta\beta)T} + \frac{2\beta\eta}{1 - 2\beta\eta}\right)(R(y) - R(x^*))$$

Using $\eta = \delta/2\beta$ and $T = 4\kappa/\delta$ we have (8.4) and the result follows.

□

Part II

Part 2: Spring 2018

Chapter 9

Introduction

This term we plan to cover the following material.

Online Optimization (week 2 and week 3)

Prediction with expert advice. Follow the perturbed leader. Stochastic bandits.
Lecture 15, 16 and 18 in [21]. See also relevant material in [9].

Support Vector Machines and Kernels (week 3 and week 4)

Reproducing Kernel Hilbert Spaces, Support Vector Machines. Generalization bounds for for SVM using Rademacher complexity.
Lecture 9 (Support Vector Machines), Lecture 10, Lecture 12 (only Example 2.3.3) in [21]. See also relevant material in [9].

Adaboost (week 5, week 6, week 7)

AdaBoost and universal consistency.
Lecture 17, 18, 19 in [10].

Kernel boosting algorithm. Generalization bounds via early stopping.
Paper [22].

Robust Optimization (week 8, possibly also later)

Robust empirical risk minimization. Generalization bounds using localized Rademacher complexities.
Paper [14].

Neural Networks (moving forward)

Generalization bounds for Neural Networks using margin theory.
Paper [11].

Chapter 10

Online Optimization

Speakers: Dominic Richards and Anthony Caterini, 26/01/2018.

The objective in this context is to make a sequence of predictions say, given $(X_1, Y_1), \dots, (X_{t-1}, Y_{t-1})$ of iid random variables as well as X_t, to then predict Y_t. We will cover an adversarial method which can be viewed as playing against an opponent to which we are trying to minimise losses against. The first method considered is expert advice where by we must select an action from a series of experts.

10.1 Expert Advice

Note that this section is based on [21, Lecture 15].

10.1.1 Cumulative Regret

Let $\mathcal{A}$ be a convex set of actions we can take and $\mathcal{Z}$ the adversary's set of moves. At time t simultaneously reveal $a_t \in \mathcal{A}$ and $z_t \in \mathcal{Z}$ with $\ell(a_t, z_t)$ being the loss associated to the player/decision maker and adversary choosing a_t and z_t respectively. The game will go on for n time steps, the total loss accumulated being $\sum_{t=1}^n \ell(a_t, z_t)$.

Due to $\sum_{t=1}^n \ell(a_t, z_t)$ being arbitrarily large, compare ourselves to the loss of an **expert**. An expert simply being $b \in \mathcal{A}^n$, $b = (b_1, \dots, b_t, \dots, b_n)^T$. Choosing K experts $b^{(1)}, \dots, b^{(n)}$ in a sense restricts us to a subspace of $\mathcal{A}^n$, avoiding the need to consider all possible strategies. This gives us one possible benchmark, the minimum loss from sticking to a single expert

$$\text{benchmark} = \min_{1 \le j \le K} \sum_{t=1}^n \ell(b_t^{(j)}, z_t).$$

A notion of **cumulative regret** can be defined to be

$$R_n := \sum_{t=1}^n \ell(a_t, z_t) - \min_{1 \le j \le K} \sum_{t=1}^n \ell(b_t^{(j)}, z_t)$$

which we hope to minimise.
The information we have at time t is the following

- All previous moves $a_1, \dots, a_{t-1}$
- All adversary's previous moves $z_1, \dots, z_{t-1}$

- All experts' strategies $b^{(1)}, \dots, b^{(K)}$.

Ultimately our strategy is going to consist of taking convex combinations of experts that have previously performed well. Noting that simply picking b^* the best performing expert thus far would be too easily exploitable from the adversary's point of view, as he can maximise the loss with respect to that choice.

Actions are now $a_t = p_t b_t^T = \sum_{k=1}^K p_{t,k} b_t^{(k)}$ where $p_t \in \Delta^K$ is the K-dimensional simplex. Denote the associated loss $\ell(a_t, z_t) = \ell(p_t, z_t)$ with regret

$$R_n = \sum_{t=1}^n \ell(p_t, z_t) - \min_{1 \le j \le K} \sum_{t=1}^n \ell(e_j, z_t)$$

e_j denoting the standard basis with zeros everywhere but the jth entry. Within this setting [21] states that our goal has reduced to the following optimisation problem

$$\min_{\theta \in \Delta^K} \sum_{j=1}^K \theta_j \sum_{t=1}^n l(e_j, z_t)$$

which, unmentioned in the notes, upper bounds the Regret when ℓ is convex in p_t. Specifically if ℓ is convex we get

$$\begin{aligned}\sum_{t=1}^n \ell(p, z_t) = \sum_{t=1}^n \ell\left(\sum_{k=1}^K p_k e_k, z_t\right) &\le \sum_{t=1}^n \sum_{k=1}^K p_k \ell(e_k, z_t) \\ &= \sum_{k=1}^K p_k \sum_{t=1}^n \ell(e_k, z_t)\end{aligned}$$

.

The aforementioned optimisation problem can be attacked with standard project gradient descent, updates being

$$q_{t+1} = p_t - \eta\left(\ell(e_1, z_t), \dots, \ell(e_K, z_t)\right)^T$$

and then $p_{t+1} = \Pi_{\Delta^K}(p_t)$, a projection back onto the simplex. But the topology gives a natural opportunity to apply mirror descent, which in this setting is interpreted as exponential weights.

10.1.2 Exponential Weights

The **Exponential Weighting (EW) strategy**, or equivalently mirror descent steps with Φ = negative entropy, becomes

$$q_{t+1,j} = p_{t,j} \exp(-\eta \ell(e_j, z_t)), \quad p_{t+1,j} = \frac{q_{t+1,j}}{\sum_{l=1}^K q_{t+1,l}}$$

which equivalently can be written as

$$p_t = \frac{1}{W_t} \sum_{j=1}^K (w_{t,j} e_j), \quad W_t := \sum_{j=1}^K w_{t,j}, \quad w_{t,j} := \exp\left(-\eta \sum_{s=1}^{t-1} \ell(e_j, z_s)\right).$$

Intuitively observe that those experts that have a low loss previously $\sum_{s=1}^{t-1} \ell(e_j, z_s)$ gain extra weight. Now we have the following theorem to bound the regret.

Theorem 10.1.1. *Assume $\ell(\cdot, z)$ is convex for all $z \in \mathcal{Z}$ and that $\ell(p, z) \in [0,1]$ for all $p \in \Delta^L, z \in \mathcal{Z}$. Then the EW strategy has regret*

$$R_n \leq \frac{\log K}{\eta} + \frac{\eta n}{2}$$

Optimising $\eta^* = \sqrt{\frac{2 \log K}{n}}$ gives

$$R_n \leq \sqrt{2n \log K}$$

From here two proofs can be given: a shorter non-optimal proof using results from mirror descent, or a longer version that gives optimal constants. We favour the latter to save deriving mirror descent results.

Proof. Observe that we have the following recursion for the normalising constants

$$\begin{aligned} W_{t+1} = \sum_{j=1}^{K} \exp\left(-\eta \sum_{s=1}^{t} \ell(e_j, z_s)\right) &= \sum_{j=1}^{K} \underbrace{\exp\left(-\eta \sum_{s=1}^{t-1} \ell(e_j, z_s)\right)}_{\propto p_{t-1,j}} \exp\left(-\eta \ell(e_j, z_t)\right) \\ &= W_t E_{J\sim p_{t-1}}\left[\exp\left(-\eta \ell(e_J, z_t)\right)\right] \end{aligned}$$

Now Hoeffding's lemma states that if $a \leq X \leq b$ and $E[X] = 0$ then

$$E[\exp(\lambda X)] \leq \exp(\lambda^2 (a-b)^2/8).$$

Therefore with $X = -\ell(e_J, z_t) + E[\ell(e_J, z_t)]$ and $a = E[\ell(e_J, z_t)] - 1, b = E[\ell(e_J, z_t)]$ as well as $\lambda = \eta$ the expectation becomes bounded

$$\begin{aligned} E_{J\sim p_{t-1}}\left[\exp\left(-\eta \ell(e_J, z_t)\right)\right] &\leq \exp\left(\eta^2/8 - \eta E_J[\ell(e_J, z_t)]\right) \\ &\leq \exp\left(\eta^2/8 - \eta \ell(E_J[e_J], z_t)\right) \\ &= \exp\left(\eta^2/8 - \eta \ell(p_t, z_t)\right) \end{aligned}$$

the later inequality arising due to convexity and equality due to $E_J e_J = \sum_{j=1}^{K} p_{t,j} e_j$. Recursively applying this bound gives with $W_1 = K$ due to $w_{1,j} = 1$

$$W_{t+1} \leq K \exp(n\eta^2/8) \exp\left(-\eta \sum_{s=1}^{n} \ell(p_s, z_s)\right)$$

Observe, that the right most quantity is the running regret, the quantity we wish to upper bound. Therefore we must lower bound W_{t+1}. Specifically we have

$$\begin{aligned} W_{t+1} &= \sum_{j=1}^{K} \exp\left(-\eta \sum_{s=1}^{n} \ell(e_j, z_s)\right) \\ &\geq \max_{j=1,\ldots,K} \exp\left(-\eta \sum_{s=1}^{n} \ell(e_j, z_s)\right) \end{aligned}$$

Combining the two bounds and taking the log of both sides gives

$$\max_{j=1,\dots,K} -\eta \sum_{s=1}^{n} \ell(e_j, z_s) \leq \log(K) + \frac{n\eta^2}{8} - \eta \sum_{s=1}^{n} \ell(p_s, z_s)$$

Therefore rearranging and turn max into a min by pulling out a minus gives

$$\sum_{s=1}^{n} \ell(p_s, z_s) - \min_{j=1,\dots,K} \sum_{s=1}^{n} \ell(e_j, z_s) \leq \frac{\log(K)}{\eta} + \frac{n\eta}{8}.$$

□

10.2 Follow the Perturbed Leader

We would now like to consider another strategy for online learning known as *Follow the Perturbed Leader* (FPL), as described (mostly) in [21, Lecture 16].

10.2.1 New Notation

In this section, it will be more convenient to *vectorize* our notation from the previous section[1]. To this end, let us define the vector Z_t as the loss associated to all experts at time t:

$$Z_t = \begin{bmatrix} \ell(e_1, z_t) \\ \vdots \\ \ell(e_K, z_t) \end{bmatrix}.$$

Then, we can rewrite the upper bound on regret as as

$$R_n = \sum_{t=1}^{n} \ell(p_t, z_t) - \min_{1\leq j\leq K} \sum_{t=1}^{n} \ell(e_j, z_t) \leq \sum_{t=1}^{n} p_t^T Z_t - \min_{p\in\Delta^K} \sum_{t=1}^{n} p^T. \tag{10.1}$$

Note that the first term is from Jensen's inequality, i.e.

$$\sum_{t=1}^{n} p_t^T Z_t \geq \sum_{t=1}^{n} \ell(p_t, z_t),$$

and the second term comes from the fact that minimizing a linear function over the probability simplex corresponds to selecting one of the vertices.

10.2.2 Follow the Leader

Before describing FPL, we will describe the associated *unperturbed* version simply called Follow the Leader (FL). In FL, at time t, we select a strategy that would have minimized the cumulative loss over the previous $t-1$ timesteps. Mathematically, if we call p_t our strategy at time t, then we select

$$p_t = \arg\min_{p\in\Delta^K} \sum_{s=1}^{t-1} p^T Z_s.$$

[1]The notation in [21, Lecture 16] is quite difficult to follow, as terms are redefined in terms of themselves and used in multiple places to mean the same thing.

This greedy strategy can be hazardous, however. Consider the following example with $K = 2$. Suppose that $Z_1 = (\varepsilon, 0)^T$, and for $t \geq 2$,

$$Z_t = \begin{bmatrix} \mathbf{1}_{t \text{ odd}} \\ \mathbf{1}_{t \text{ even}} \end{bmatrix}.$$

In this example, the FL strategy would be to first choose p_1 arbitrarily ($p_1 = (0,1)^T$ in the best case), and then for $t \geq 2$,

$$p_t = \begin{bmatrix} \mathbf{1}_{t \text{ odd}} \\ \mathbf{1}_{t \text{ even}} \end{bmatrix}.$$

Thus, with the FL strategy, we have

$$n - 1 \leq \sum_{t=1}^{n} p_t^T Z_t \leq n - 1 + \epsilon,$$

On the other hand, the optimal choice of p to minimize this sum incurs cost equal to

$$\min_{p \in \Delta^K} \sum_{t=1}^{n} p^T Z_t = \begin{cases} \frac{n-1}{2}, & \text{for } n \text{ odd}, \\ \frac{n}{2} - (1 - \epsilon), & \text{for } n \text{ even}. \end{cases}$$

Thus, we can only bound the regret R_n from (10.1) linearly. However, we would like to do better, which is where the FPL strategy comes in.

10.2.3 FPL Algorithm and Analysis

FPL can be considered to be a *regularization* of FL, as we add a small amount of noise to the selection of the strategy, but we can guarantee that the expected cumulative regret increases no faster than the order of $\sqrt{n}$ in the case of an oblivious adversary[2]. The exact FPL strategy is as follows: for some $\eta > 0$, we first draw ξ from a uniform distribution on $[0, \frac{1}{\eta}]^K$. Then, at time t, we select p_t according to

$$p_t = \arg\min_{p \in \Delta^K} \left\{ p^T \left(\sum_{s=1}^{t-1} Z_s + \xi \right) \right\}. \tag{10.2}$$

We will show that the FPL strategy incurs expected regret at worst proportionally to $\sqrt{n}$, but we first describe an intermediate result.

Lemma 10.2.1 (Be The Leader). For any loss function $\mathcal{L} : \Delta^K \times \mathbb{R}^K \to \mathbb{R}$, let

$$q_t^* = \arg\min_{q \in \Delta^K} \sum_{s=1}^{t} \mathcal{L}(q, Z_s).$$

Then, we have

$$\sum_{t=1}^{n} \mathcal{L}(q_t^*, Z_t) \leq \sum_{t=1}^{n} \mathcal{L}(q_n^*, Z_t).$$

[2]This means the sequence $\{Z_t\}_{t=1}^n$ is chosen ahead of time

Proof. This can be proven by induction on n. It is clearly true for the base case $n = 1$. Then, assuming it is true for $n = k$, we have that

$$\begin{aligned}\sum_{t=1}^{k+1} \mathcal{L}(q_t^*, Z_t) &= \sum_{t=1}^{k} \mathcal{L}(q_t^*, Z_t) + \mathcal{L}(q_{k+1}^*, Z_{k+1}) \\ &\leq \sum_{t=1}^{k} \mathcal{L}(q_k^*, Z_t) + \mathcal{L}(q_{k+1}^*, Z_{k+1}) && \text{by the induction hypothesis} \\ &\leq \sum_{t=1}^{k} \mathcal{L}(q_{k+1}^*, Z_t) + \mathcal{L}(q_{k+1}^*, Z_{k+1}) && \text{by the definition of } q_k^* \\ &= \sum_{t=1}^{k+1} \mathcal{L}(q_{k+1}^*, Z_t),\end{aligned}$$

which completes the proof of the claim. □

Note that the result of the above lemma indeed makes perfect sense: using q_t^* at each step, as opposed to just q_n^*, affords us more freedom to optimize the sum. Also note that the use of q instead of p in the above was deliberate: we will generally use q for strategies that can *cheat* and perform optimally at time t given information only available after time t (as in the proof of the theorem below).

Now we present and prove the main result, which is the bound on the expected regret of the FPL algorithm.

Theorem 10.2.2. *Suppose that each component of Z_t is between 0 and 1 for all $t \in \{1, \dots, n\}$.*[3] *Then, FPL with $\eta = \sqrt{\frac{2}{nK}}$ yields expected regret*

$$\mathbf{E}_\xi[R_n] \leq 2\sqrt{2nK}. \tag{10.3}$$

Proof. Let us first define the quantity q_t as the optimal strategy given the adversary's moves from time $s = 1$ to $s = t$:

$$q_t = \operatorname*{arg\,min}_{p \in \Delta^K} p^T \left(\xi + \sum_{s=1}^{t} Z_s \right).$$

Then, invoking Lemma 10.2.1 with the loss function

$$\mathcal{L}(p, Z_t) = p^T Z_t + p^T \xi \, \mathbf{1}_{t=1},$$

we have

$$\sum_{t=1}^{n} \mathcal{L}(q_t, Z_t) = q_1^T \xi + \sum_{t=1}^{n} q_t^T Z_t \leq \sum_{t=1}^{n} \mathcal{L}(q_n, Z_t) = \min_{q \in \Delta^K} q^T \left(\xi + \sum_{t=1}^{n} Z_t \right).$$

Thus, for any $q \in \Delta^K$,

$$\begin{aligned} q_1^T \xi + \sum_{t=1}^{n} q_t^T Z_t &\leq q^T \left(\xi + \sum_{t=1}^{n} Z_t \right) \\ \implies \sum_{t=1}^{n} \left(q_t^T Z_t - q^T Z_t \right) &\leq \left(q^T - q_1^T \right) \xi \leq \|q - q_1\|_1 \, \|\xi\|_\infty \leq \frac{2}{\eta} \end{aligned}$$

[3] We can rescale ℓ if this is not the case.

by Hölder's inequality and the fact that $q, q_1 \in \Delta^K$ (so that the largest 1-norm distance between them occurs in the case when they lie on different vertices of Δ^K, which would be a distance of 2 away).

We can now bound part of the expected regret term:

$$\begin{aligned}
\mathbf{E}[R_n] &\leq \mathbf{E}\left[\sum_{t=1}^{n} p_t^T Z_t\right] - \sum_{t=1}^{n} p^{*T} Z_t \\
&= \mathbf{E}\left[\sum_{t=1}^{n} (p_t - q_t)^T Z_t\right] + \mathbf{E}\left[\sum_{t=1}^{n} \left(q_t^T Z_t - p^{*T} Z_t\right)\right] \\
&\leq \sum_{t=1}^{n} \mathbf{E}\left[(p_t - q_t)^T Z_t\right] + \frac{2}{\eta},
\end{aligned} \tag{10.4}$$

where $p^* = \arg\min_{p \in \Delta^K} \sum_{t=1}^{n} p^T Z_t$.

To handle the other term, we first define a function $h_t : [0, \frac{1}{\eta}]^K$ as

$$h_t(\xi) = Z_t^T \left(\arg\min_{p \in \Delta^K} p^T \left[\xi + \sum_{s=1}^{t-1} Z_s\right]\right).$$

Then, we can easily see that

$$\mathbf{E}\left[Z_t^T (p_t - q_t)\right] = \mathbf{E}\left[h_t(\xi)\right] - \mathbf{E}\left[h_t(\xi + Z_t)\right].$$

We can therefore bound this term according to

$$\begin{aligned}
\mathbf{E}\left[Z_t^T (p_t - q_t)\right] &= \eta^K \int_{\xi \in [0, \frac{1}{\eta}]^K} h_t(\xi) d\xi - \eta^K \int_{\xi \in \left\{Z_t + [0, \frac{1}{\eta}]^K\right\}} h_t(\xi) d\xi \\
&\leq \eta^K \int_{\xi \in [0, \frac{1}{\eta}]^K \setminus \left\{Z_t + [0, \frac{1}{\eta}]^K\right\}} h_t(\xi) d\xi, && \text{since } h_t(\xi) \geq 0 \\
&\leq \eta^K \int_{\xi \in [0, \frac{1}{\eta}]^K \setminus \left\{Z_t + [0, \frac{1}{\eta}]^K\right\}} d\xi, && \text{since } h_t(\xi) \leq 1 \\
&= \mathbf{P}\left[\exists i \in \{1, \ldots, K\} : \xi^i \leq Z_t^i\right] && \text{by the geometry of the integrated set} \\
&\leq \sum_{i=1}^{K} \mathbf{P}\left(\xi^i \leq Z_t^i\right) && \text{union bound} \\
&\leq \eta K Z_t^i \leq \eta K && \text{since } Z_t^i \leq 1.
\end{aligned}$$

Substituting this result into (10.4) gives us the bound

$$\mathbf{E}[R_n] \leq \eta K n + \frac{2}{\eta}.$$

This is optimized when $\eta = \sqrt{\frac{2}{nK}}$, which gives us (10.3). □

10.2.4 Closing Remarks on FPL

The first thing to note about Theorem 10.2.2 is that we assume that we know n at the start so that we can choose η optimally. In many applications, however, this is not the case: we do not always know for how many periods we will participate. In practice, it may be possible to assume that there may be some small number of periods, and then update η accordingly if we exceed this number with a new, larger number of periods.

We also have selected a sub-optimal noise component in the above theorem. In the original paper on FPL, the authors show that they can recover the $\sqrt{\log K}$ dependency in the bound – as seen in the exponential weighting strategy – using an exponential noise term [17]. However, the proof is more complicated than the one presented above. There may also be theoretical gains realized by resampling ξ at each iteration as opposed to at the start of the optimization.

10.3 Stochastic Bandits

Speakers: Dominic Richards and Anthony Caterini, 02/02/2018.

This week, we will be looking at stochastic bandits. This will be strongly based off the material in [21, Lecture 18]. The stochastic multi-armed bandit is a well-known and well-studied model of decision making. We define the model as follows: suppose our bandit has K *arms* (i.e. possible actions), and at each time $t \in \{1, \dots, n\}$, we receive reward $X_{k,t}$ for selecting action $k \in \{1, \dots, K\}$. The sequence $\{X_{k,1}, \dots, X_{k,t}, \dots, X_{k,n}\}$ is i.i.d. for each choice of k, with $\mathbf{E}[X_{k,t}] = \mu_k$. We define $\mu_* = \max_k \mu_k$, and we define a policy $\pi = \{\pi_1, \dots, \pi_n\}$ such that π_t specifies the action at time t and is only dependent on information available before time t.

We then define the regret in our policy π as

$$\begin{aligned}
R_n &= \max_k \mathbf{E}\left[\sum_{t=1}^{n} X_{k,t}\right] - \mathbf{E}\left[\sum_{t=1}^{n} X_{\pi_t,t}\right] \\
&= n\mu_* - \sum_{t=1}^{n}\sum_{k=1}^{K} \mathbf{E}\left[X_{\pi_t,t} | \pi_t = k\right] \mathbf{P}[\pi_t = k] \qquad \text{by the law of total probability} \\
&= n\mu_* - \sum_{t=1}^{n}\sum_{k=1}^{K} \mu_k \mathbf{E}\left[\mathbf{1}_{\pi_t=k}\right] \\
&= n\mu_* - \sum_{k=1}^{K} \mu_k \mathbf{E}\left[\sum_{t=1}^{n} \mathbf{1}_{\pi_t=k}\right] \\
&= \sum_{k=1}^{K} \Delta_k \mathbf{E}[T_k(n)], \qquad (10.5)
\end{aligned}$$

where $\Delta_k = \mu_* - \mu_k$ and $T_k(n) = \sum_{t=1}^{n} \mathbf{1}_{\pi_t=k}$.

10.3.1 Warm Up: Full Info Case

10.3.2 Upper Confidence Bound

Now we consider the case where we only observe a realization of $X_{k,t}$ if $\pi_t = k$. Given that we no longer have full information, what is a good strategy? A naïve idea is to pull all arms once, and from then on select action π_t at time t satisfying

$$\pi_t \in \arg\max_{k \in [K]} \frac{1}{T_k(t)} \sum_{s=1}^{t-1} \mathbf{1}_{\pi_s=k} X_{k,s},$$

i.e. choose the action that currently has the highest empirical mean, where we slightly redefine $T_k(t)$ as

$$T_k(t) = \sum_{s=1}^{t-1} \mathbf{1}_{\pi_s=k}.$$

Unfortunately, this strategy suffers from a major pitfall: it is entirely possible that we will completely ignore actions having high reward because of unlucky observations early on. To alleviate this issue, we instead select a strategy which maxmimizes some probabilistic upper bound on the average return of action k. In this way, actions which have not been chosen will have a higher-variance estimate of the mean, and thus a higher upper bound. We call this algorithm the Upper Confidence Bound (UCB) algorithm, and we present it in Algorithm 10.3.1.

Before proceeding, it is important to note the following: without loss of generality, we now assume that $X_{k,t}$ is subgaussian with variance proxy 1 in this section. Also, we will use the notation

$$\hat{\mu}_{k,t} = \frac{1}{T_k(t)} \sum_{s=1}^{t-1} \mathbf{1}_{\pi_s = k} X_{k,s}$$

to denote the empirical mean of action k at time t, and $\hat{\mu}_{*,t}$ to denote the empirical mean of the optimal action.

Procedure 10.3.1 Upper Confidence Bound (UCB)

```
Input: $X_{k,t}$ is 1-subgaussian for all $t \in [n], k \in [K]$.
1: for $t = 1$ to $K$ do
2:     $\pi_t = t$
3: end for
4: for $t = K + 1$ to $n$ do
5:     $T_k(t) = \sum_{s=1}^{t-1} \mathbf{1}_{\pi_s = k}$
6:     $\hat{\mu}_{k,t} = \frac{1}{T_k(t)} \sum_{s=1}^{t-1} \mathbf{1}_{\pi_s = k} X_{k,s}$
7:     $\pi_t \in \arg\max_{k \in [K]} \left( \hat{\mu}_{k,t} + 2\sqrt{\frac{2 \log t}{T_k(t)}} \right)$
8: end for
```

The UCB strategy is quite good: we will show below that we can bound our total regret by a $\mathcal{O}(\log n)$ function. **NOTE: I am not sure about the factor of 2 in line 7 of Algorithm 10.3.1 – it is messing up some later computations and does not seem to be standard in other references.**

Theorem 10.3.1. *The UCB policy has regret satisfying*

$$R_n \leq 32 \sum_{k:\Delta_k} \frac{\log n}{\Delta_k} + \left(1 + \frac{\pi^2}{3}\right) \sum_{k=1}^{K} \Delta_k.$$

Proof. First, a brief note: this is not the exact same result as the one in [21, Lecture 18] – the 32 here is replaced by an 8 there. However, that proof is riddled with errors and very tough to follow. In fact, some of the lines written below are provided with no justification because I could not determine why they were true. I provided some additional information, as I could find it, from multiple sources, but mainly from [12, Theorem 2.1], with $\alpha = 4$ and $\psi^*(\epsilon) = 2\epsilon^2$.

That being said, let us proceed with the proof. Our goal is to bound $\mathbf{E}[T_k(n)]$, and then invoke (10.5) to bound the regret R_n. We can restrict ourselves to k satisfying $\Delta_k > 0$, since when $\Delta_k = 0$, we do not need to control $\mathbf{E}[T_k(n)]$ to control the regret. We can start by saying that

$$\mathbf{E}[T_k(n)] = 1 + \sum_{t=K+1}^{n} \mathbf{P}[\pi_t = k],$$

since action k will be chosen once in the first K iterations of UCB.

Now, for $t > K$, we have that $\pi_t = k$ implies $\hat{\mu}_{k,t} + 2\sqrt{\frac{2\log t}{T_k(t)}} \geq \hat{\mu}_{*,t} + 2\sqrt{\frac{2\log t}{T_*(t)}}$, by virtue of the UCB strategy. This, in turn, implies that one of the following inequalities holds:

$$\hat{\mu}_{k,t} > \mu_k + 2\sqrt{\frac{2\log t}{T_k(t)}}, \tag{10.6}$$

$$\mu_* \geq \hat{\mu}_{*,t} + 2\sqrt{\frac{2\log t}{T_*(t)}}, \tag{10.7}$$

$$\mu_* \leq \mu_k + 4\sqrt{\frac{2\log t}{T_k(t)}}. \tag{10.8}$$

To prove this, suppose all of the above inequalities were false. Then,

$$\begin{aligned}
\hat{\mu}_{k,t} + 2\sqrt{\frac{2\log t}{T_k(t)}} &\leq \mu_k + 4\sqrt{\frac{2\log t}{T_k(t)}} && \text{since (10.6) is false}\\
&< \mu_* && \text{since (10.8) is false}\\
&< \hat{\mu}_{*,t} + 2\sqrt{\frac{2\log t}{T_*(t)}} && \text{since (10.7) is false}
\end{aligned}$$

This is a contradiction, however, since we assume that $\pi_t = k$.

We can also bound the probability that equations (10.6) and (10.7) are true:

$$\begin{aligned}
\mathbf{P}\left[(10.7) \text{ holds}\right] &\leq \mathbf{P}\left[\exists s \in \{1, \dots, t\} : \hat{\mu}_{*,t} + 2\sqrt{\frac{2\log t}{s}} \leq \mu_* \middle| T_*(t) = s\right]\\
&\leq \sum_{s=1}^{t} \mathbf{P}\left[\mu_* - \hat{\mu}_{*,t} \geq 2\sqrt{\frac{2\log t}{s}} \middle| T_*(t) = s\right] && \text{by the union bound}\\
&\leq \sum_{s=1}^{t} \exp\left(\frac{-s}{2}\frac{8\log t}{s}\right) && \text{by the subgaussian assumption, since } \hat{\mu}_{*,t} \text{ has } s \text{ terms}\\
&= \frac{1}{t^3}.
\end{aligned}$$

Similarly, we can show $\mathbf{P}\left[(10.6) \text{ holds}\right] \leq \frac{1}{t^3}$. Now, we can finally bound the item we are after:

$$\begin{aligned}
\sum_{t=K+1}^{n} \mathbf{P}\left[\pi_t = k\right] &\leq \sum_{t=1}^{n} \left(\mathbf{P}\left[(10.6) \text{ holds}, \pi_t = k\right] + \mathbf{P}\left[(10.7) \text{ holds}, \pi_t = k\right] + \mathbf{P}\left[(10.8) \text{ holds}, \pi_t = k\right]\right)\\
&\leq 2\sum_{t=1}^{\infty} \frac{1}{t^3} + \sum_{t=1}^{n} \mathbf{P}\left[\mu_* \leq \mu_k + 4\sqrt{\frac{2\log t}{T_k(t)}}, \pi_t = k\right]\\
&\leq 2\sum_{t=1}^{\infty} \frac{1}{t^2} + \sum_{t=1}^{n} \mathbf{P}\left[T_k(t) \leq \frac{32\log t}{\Delta_k^2}, \pi_t = k\right] && \text{note the extra factor of } 2^2 = 4\\
&\leq \frac{\pi^2}{3} + \sum_{t=1}^{n} \mathbf{P}\left[T_k(t) \leq \frac{32\log n}{\Delta_k^2}, \pi_t = k\right]
\end{aligned}$$

$$\begin{aligned}
&\leq \frac{\pi^2}{3} + \sum_{t=1}^{n} \mathbf{E}\left[\mathbf{1}_{T_k(t) \leq \frac{32 \log n}{\Delta_k^2}}\right] \\
&= \frac{\pi^2}{3} + \mathbf{E}\left[\sum_{t=1}^{n} \mathbf{1}_{T_k(t) \leq \frac{32 \log n}{\Delta_k^2}}\right] \\
&\leq \frac{\pi^2}{3} + \frac{32 \log n}{\Delta_k^2} \qquad \textbf{Not exactly clear on this step -- what if } T_k(t) \textbf{ was always low?}
\end{aligned}$$

Thus, we have that

$$R_n = \sum_{k=1}^{K} \Delta_k \mathbf{E}[T_k(n)] \leq \sum_{k, \Delta_k > 0} \Delta_k \left(1 + \frac{\pi^2}{3} + \frac{32 \log n}{\Delta_k^2}\right),$$

completing the proof. □

We will make a quick comment on this result before moving on. If we consider the case that $K = 2$, we know that $R_n \sim \frac{\log n}{\Delta}$, which is consistent with the idea that we will have a tough time deciding between the two actions if the difference in reward (Δ) is small. However, R_n is guaranteed to be less than $n\Delta$, so a small Δ does not generally imply large regret. Indeed, we can bound R_n by $\min\{\frac{\log n}{\Delta}, n\Delta\}$, up to constants.

10.3.3 Bounded Regret

Chapter 11

Kernels

Speaker: ???

Chapter 12

AdaBoost

Speaker: Adam Foster, 16/03/2018.

In ensemble methods for machine learning we assume that

$$(W_1, Y_1), ..., (W_n, Y_n)$$

are i.i.d. with $W_i \in \mathcal{W}$. For binary classification, we have $Y_i \in \{\pm 1\}$. We have access to a set $\mathcal{G}$ of base (also called weak) procedures

$$g : \mathcal{W} \to \{\pm 1\} \text{ for } g \in \mathcal{G}$$

The aim is to construct an aggregate of base procedures which solves the following optimization problem

$$\begin{aligned} \min_f \quad & \mathbf{E}\ell(f(W), Y) \\ \text{subject to} \quad & f = \sum_{t=1}^{T} \alpha_t g_t \text{ with } \alpha_t \geq 0, g_t \in \mathcal{G} \text{ for } t = 1, ..., T. \end{aligned}$$

As in Section 1, ℓ is a loss functions typically of the form $\ell(z, y) = \varphi(-zy)$ for a given $\varphi : \mathbb{R} \to \mathbb{R}$. A natural loss function in the classification regime is the Zero-One loss

$$\begin{aligned} \ell_{01}(f(w), y) &= \mathbf{1}[\text{sign}(f(w)) \neq y] \\ \varphi_{01}(u) &= \mathbf{1}[u \geq 0] \end{aligned}$$

but regrettably this is neither convex nor differentiable. In AdaBoost, we use the Exponential Loss

$$\varphi_{\exp}(u) = \exp(u)$$

which is an upper bound on φ_{01}.

Since the distribution of (W, Y) is unknown, we optimize the **empirical** loss

$$\mathbf{E}_n \ell(f(W), Y) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(W_i), Y_i)$$

12.1 The AdaBoost algorithm

AdaBoost is a recursive algorithm that greedily minimizes the empirical exponential loss, $\ell_{\exp}$, at each step.

Suppose that f_{t-1} is given. Greedy minimization means that we choose α_t and g_t to minimize

$$\mathbf{E}_n \ell_{\exp}[f_{t-1}(W) + \alpha_t g_t(W), Y] = \frac{1}{n}\sum_{i=1}^{n} \exp[-Y_i f_{t-1}(W_i)] \exp[-\alpha_t Y_i g_t(W_i)]$$

define the normalized weights $D_t(i) \propto \exp[-Y_i f_{t-1}(W_i)]$. The objective is then proportional to

$$\sum_{i=1}^{n} D_t(i) \exp[-\alpha_t Y_i g_t(W_i)] = e^{-\alpha_t} + (e^{\alpha_t} - e^{-\alpha_t}) \sum_{i=1}^{n} D_t(i)\mathbf{1}[Y_i \neq g_t(W_i)] \tag{12.1}$$

Recall $\alpha_t \geq 0$, so $e^{\alpha_t} \geq e^{-\alpha_t}$. Thus we can first choose g_t to minimize the weighted misclassification error

$$\epsilon_t = \sum_{i=1}^{n} D_t(i)\mathbf{1}[Y_i \neq g_t(W_i)]$$

then the optimal α_t is simply

$$\alpha_t = \tfrac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right) \tag{12.2}$$

We have just derived the following algorithm

Procedure 12.1.1 AdaBoost

```
1: Let $f_0 \equiv 0$
2: Let $D_1(i) = \frac{1}{n}$
3: for $t = 1$ to $T$ do
4:     Choose $g_t \in \mathcal{G}$ to minimize the weighted misclassification error $\epsilon_t = \sum_{i=1}^{n} D_t(i)\mathbf{1}[Y_i \neq g_t(W_i)]$
5:     Let $\alpha_t = \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$
6:     Let $f_t = f_{t-1} + \alpha_t g_t$
7:     Reweight $D_{t+1}(i) \propto D_t(i)\exp(-\alpha_t Y_i g_t(W_i))$
8: end for
9: Normalize $f = f_T / \sum_{t=1}^{T} \alpha_t$
```

12.1.1 Choice of $\mathcal{G}$

In classification, it is extremely common to choose the base procedures to be some form of decision tree. Two common cases are:

- Decision stumps. For $\mathcal{W} = \mathbb{R}^k$, the (i, c, s)-decision stump assigns $+s$ to $\{w : w_i \geq c\}$ and $-s$ to the rest. (We need $s = \pm 1, 1 \leq i \leq k, c \in \mathbb{R}$.)

- Decision trees. For $\mathcal{W} = \mathbb{R}^k$, decision trees recursively split the input space into a number of (hyper)rectangles and assign $+1$ or -1 to each rectangle. They are a generalization of decision stumps.

12.1.2 Benefits of exponential loss

A useful property of the Exponential Loss ℓ_{exp} is that it has the same population minimizer as ℓ_{01}.

To see this, first apply the Law of Total Expectation

$$\mathbf{E}[\exp(-Yf(W))] = \mathbf{E}[\mathbf{E}[\exp(-Yf(W))|W]].$$

It suffices to minimize the conditional expectation pointwise

$$\mathbf{E}[\exp(-Yf(w))] = \mathbf{E}[e^{-f(w)}\mathbf{P}(Y=1|W=w) + e^{f(w)}\mathbf{P}(Y=-1|W=w)]$$

we then differentiate

$$\frac{\partial}{\partial f(w)}\left(e^{-f(w)}\mathbf{P}(Y=1|W=w) + e^{f(w)}\mathbf{P}(Y=-1|W=w)\right) = -e^{-f(w)}\mathbf{P}(Y=1|W=w)+e^{f(w)}\mathbf{P}(Y=-1|W=w)$$

and set the derivative equal to zero

$$\begin{aligned} e^{-f^*(w)}\left(-\mathbf{P}(Y=1|W=w) + e^{2f^*(w)}\mathbf{P}(Y=-1|W=w)\right) &= 0 \\ f^*(w) &= \tfrac{1}{2}\log\frac{\mathbf{P}(Y=1|W=w)}{\mathbf{P}(Y=-1|W=w)} \end{aligned}$$

which indeed gives the Zero-One minimizer.

12.1.3 A geometric perspective on AdaBoost

Recall the **Kullback-Leibler** (KL) divergence for distributions p and q

$$KL(p,q) = \sum_{i=1}^{n} p(i)\log\frac{p(i)}{q(i)}$$

Let u denote the uniform distribution on $1, ..., n$. Consider the following optimization problem

$$\begin{aligned} \min_{p} \quad & KL(p,u) \\ \text{subject to} \quad & p \in \Delta_n, \\ & \sum_{i=1}^{n} p(i)Y_i g(W_i) = 0 \text{ for } g \in \mathcal{G} \end{aligned}$$

The Lagrangian for this problem is

$$L(p,\alpha,\lambda) = \sum_{i=1}^{n} p(i)\log(np(i)) + \sum_{g\in\mathcal{G}}\alpha_g\left(\sum_{i=1}^{n} p(i)Y_i g(W_i)\right) + \lambda\left(\sum_{i=1}^{n} p(i) - 1\right)$$

(we have ignored the non-negativity constraints but it turns out we get them for free).

We differentiate L w.r.t. $p(i)$

$$\frac{\partial L}{\partial p(i)} = \log(np(i)) + 1 + \sum_{g\in\mathcal{G}}\alpha_g Y_i g(W_i) + \lambda$$

and solve equal to 0

$$p(i) = \exp\left(-\sum_{g\in\mathcal{G}} \alpha_g Y_i g(W_i)\right) / Z$$

where Z is a normalization constant. The Lagrangian dual function is

$$g(\alpha, \lambda) = \log n - \log Z = \log n - \log \sum_{i=1}^{n} \exp\left(-\sum_{g\in\mathcal{G}} \alpha_g Y_i f(W_i)\right)$$

and the dual problem $\max_{\alpha,\lambda} g(\alpha, \lambda)$ is equivalent to

$$\min_{\alpha} \frac{1}{n}\sum_{i=1}^{n} \exp\left(-Y_i \sum_{g\in\mathcal{G}} \alpha_g g(W_i)\right)$$

which is the AdaBoost objective.

12.2 Weak learning implies strong learning

The set $\mathcal{G}$ of base procedures must have sufficient flexibility to give an edge over random guessing. More formally

Definition 12.2.1 (Weak learners of parameter γ)**.** *Set $\mathcal{G}$ is a family of γ-**weak learners** if for every distribution on (W, Y) there exists $g \in \mathcal{G}$ such that*

$$\mathbf{P}_{(W,Y)}[Yg(W) \le 0] \le \frac{1-\gamma}{2}$$

With this definition, we can begin a theoretical analysis of AdaBoost.

Theorem 12.2.1 (AdaBoost with weak learners)**.** *The optimized empirical exponential loss of AdaBoost is*

$$\prod_{t=1}^{T} 2\sqrt{\epsilon_t(1-\epsilon_t)}.$$

Suppose now that $\exists\gamma > 0$ such that $\mathcal{G}$ is a family of γ-weak learners. Then the empirical misclassification rate decreases to zero at the following exponential rate

$$\mathbf{P}_n[Yf(W) \le 0] \le (1-\gamma^2)^{T/2}$$

and equals zero after at most $2\log n/\gamma^2$ iterations.

Proof. From equation (12.1) taking $t = T$ we first note that the constant of proportionality is $\frac{1}{n}\sum D_T(i)(W_i) = \mathbf{E}_n[\exp(-Yf_{T-1}(W))]$. Thus,

$$\mathbf{E}_n[\exp(-Yf_T(W)] = \left(e^{-\alpha_T} + \epsilon_T(e^{\alpha_T} - e^{-\alpha_T})\right)\mathbf{E}_n[\exp(-Yf_{T-1}(W))]$$

Using equation (12.2) we have $e^{\pm\alpha_T} = \left(\frac{1-\epsilon_T}{\epsilon_T}\right)^{\pm 1/2}$, giving

$$\begin{aligned}\mathbf{E}_n[\exp(-Yf_T(W)] &= 2\sqrt{\epsilon_T(1-\epsilon_T)}\mathbf{E}_n[\exp(-Yf_{T-1}(W))] \\ &= \prod_{t=1}^{T} 2\sqrt{\epsilon_t(1-\epsilon_t)}\end{aligned}$$

Now suppose $\mathcal{G}$ is a family of γ-weak learners. Then (considering the empirical law on (W, Y) in the definition of weak learners) we must have each $\epsilon_t \leq \frac{1-\gamma}{2}$, since the choice of g_t is optimal. Hence $2\sqrt{\epsilon_t(1-\epsilon-t)} \leq \sqrt{1-\gamma^2}$

Finally, since $\varphi_{\exp} \geq \varphi_{01}$, we have

$$\begin{aligned}\mathbf{P}_n(Yf_T(W) \leq 0) &\leq \prod_{t=1}^{T} 2\sqrt{\epsilon_t(1-\epsilon_t)} \\ &\leq (1-\gamma^2)^{T/2}\end{aligned}$$

as claimed.

Now, after $T = 2\log n/\gamma^2$ iterations, we have $(1-\gamma^2)^{T/2} < e^{-\gamma^2 T/2} = 1/n$. But $\mathbf{P}_n(Yf_T(W))$ takes a minimum positive value $1/n$ and hence must be 0.

□

12.3 Gradient boosting

AdaBoost can be generalized in a number of ways. Within the realm of classification, we can consider new loss functions like the logistic loss $\varphi(u) = \log_2(1 + \exp(u))$. We can extend AdaBoost to multi-class classification. One way to do this is to let $Y \in \mathbb{R}^k$ with one component equal to 1 and the others equal to $-1/(k-1)$. The Exponential Loss function becomes $\ell_{\exp}(z, y) = \exp(-y^T z)$. The multi-class AdaBoost algorithm can then be derived exactly as in Section 12.1.

Boosting is also applicable to regression, where $Y \in \mathbb{R}$. In this context, one typically uses the least-squares loss $\ell(z, y) = (z - y)^2$. In the regression context, the possible choices for $\mathcal{G}$ become much larger. For instance:

- Decision trees. Like classification trees, regression trees split the space recursively into rectangles and assign a real number to each rectangle.
- Single-hidden-layer neural networks. These take as output $\sigma(b + \langle x, W\rangle)$ where σ is a nonlinear function, $b \in \mathbb{R}, x \in \mathcal{W}$.
- Wavelets.
- Regression splines.
- Reproducing kernel Hilbert spaces (RHKS). If $k : \mathcal{W} \times \mathcal{W} \to \mathbb{R}$ is a positive definite kernel, we may take $\mathcal{G}$ as the set $\{k(\cdot, w) : w \in A\}$ where $A \subseteq \mathcal{W}$ is a collection of points

In the last three cases, suitable choices of $\mathcal{G}$ ensure that span$(\mathcal{G})$ consists of all (appropriate) real functions on $\mathcal{W}$.

12.3.1 Boosting as variational gradient descent

(This section is based on [18])

We can phrase the optimization problem that boosting targets as

$$\min_{f \in \mathcal{F}} C(f) \tag{12.3}$$

where $\mathcal{F} = \text{span}(\mathcal{G})$ consists of finite linear combinations of base procedures and $C(f) = \mathbf{E}_n \ell(f(W), Y)$ is the empirical loss. Under suitable conditions[1], one could approach this problem directly using gradient descent methods

$$f_t = f_{t-1} - \eta_t \partial C(f_{t-1}).$$

[1]for instance, if $\mathcal{F}$ is a Hilbert space and C a differentiable functional

As an approximation, one could find

$$g_t = \arg\max_{g\in\mathcal{G}} \; -\langle \partial C(f_{t-1}), g\rangle \tag{12.4}$$

and then

$$f_t = f_{t-1} + \alpha_t g_t$$

optimizing the non-negative coefficient α_t separately.

Whilst $\partial C(f_{t-1})$ may seem a daunting functional derivative, if we take the inner product

$$\langle f_1, f_2\rangle = \frac{1}{n}\sum_{i=1}^{n} f_1(W_i) f_2(W_i) \tag{12.5}$$

we have a more explicit form for the objective in (12.4)

$$-\langle \partial C(f_{t-1}), g\rangle = -\frac{1}{n^2}\sum_{i=1}^{n} \frac{\partial \ell}{\partial z}(f_{t-1}(W_i), Y_i)\cdot g(W_i)$$

There are two interesting special cases. If $Y \in \{\pm 1\}$ and $\ell(z,y) = \varphi(-yz)$ then we have

$$\begin{aligned}-\langle \partial C(f_{t-1}), g\rangle &= \frac{1}{n^2}\sum_{i=1}^{n} Y_i \varphi'(-Y_i f_{t-1}(W_i))\cdot g(W_i)\\ &= \frac{1}{n^2}\sum_{i=1}^{n}\varphi'(-Y_i f_{t-1}(W_i)) - \frac{2}{n^2}\sum_{i=1}^{n}\varphi'(-Y_i f_{t-1}(W_i))\mathbf{1}[Y_i \neq g(W_i)]\end{aligned}$$

assuming that $\varphi' \leq 0$ (sensible for classification) we see that g is chosen to minimize the weighted empirical misclassification error.

If $Y \in \mathbb{R}$ and $\ell(z,y) = (z-y)^2$ then we have

$$-\langle \partial C(f_{t-1}), g\rangle = -\frac{1}{n^2}\sum_{i=1}^{n} 2(f_{t-1}(W_i) - Y_i)\cdot g(W_i)$$

so g is chosen to align maximally with the residuals left by f_{t-1}.

The gradient perspective gives rise to a very large number of boosting algorithms.

Procedure 12.3.1 Gradient Boosting

```
1: Let $f_0 \equiv 0$
2: for $t = 1$ to $T$ do
3:     Choose $g_t \in \mathcal{G}$ to maximize $-\langle \partial C(f_{t-1}), g\rangle$, using the inner product of (12.5)
4:     Choose $\alpha_t$ to minimize $C(f_{t-1} + \alpha_t g_t)$
5:     Let $f_t = f_{t-1} + \alpha_t g_t$
6: end for
7: Normalize $f = f_T / \sum_{t=1}^{T}\alpha_t$
```

12.4 Universal consistency of AdaBoost

References:

- `https://bcourses.berkeley.edu/courses/1409209/files/67384944`
- `http://statistics.berkeley.edu/sites/default/files/tech-reports/722.pdf`

We begin with a number of definitions. Recall the **classification risk**

$$R(f) = \mathbf{P}[f(W) \neq Y]$$

Let $R^* = \inf_f R(f)$ where the inf is taken over all classification rules[2] $f : \mathcal{W} \to \{\pm 1\}$.

A sequence (f_n) of classification rules is **universally consistent** if

$$R(f_n) \to R^* \text{ a.s. as } n \to \infty \tag{12.6}$$

for every distribution of (W, Y).

A proof that AdaBoost is universally consistent can be found in (cite Bartlett).

12.4.1 Classification calibration

We want

$$R_\varphi(f_n) \to R_\varphi^* \implies R(f_n) \to R^*$$

12.4.2 Sieves

Let $\mathcal{F}_n$ be all classification rules obtainable with a sample of size n. Let $\bar{f}_n$ be the optimizer of R_φ over this family. Then we want to prove that

$$R_\varphi(\bar{f}_n) \to R^* \tag{12.7}$$

and that the empirical risk and population risk get close

$$|R_\varphi(\bar{f}_n) - R_{\varphi,n}(\bar{f}_n)| \to 0 \tag{12.8}$$

The concentration inequalities we seek to use require bounded φ. Hence, we clip them using π_C.

[2]Note that classification rules are derived from training data. Classification rules generated from n examples are therefore measurable w.r.t. the σ-algebra, Σ_n, generated by $(W_1, Y_1), ..., (W_n, Y_n)$. The class considered here is all classification rules measurable under $\Sigma = \cup_n \Sigma_n$.

Chapter 13

Robustness

Speaker: ???

Chapter 14

Neural Networks

Speaker: ???

Bibliography

[1] Zeyuan Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. *arXiv preprint arXiv:1603.05953*, 2016.

[2] Zeyuan Allen-Zhu. Natasha 2: Faster non-convex optimization than sgd. *arXiv preprint arXiv:1708.08694*, 2017.

[3] Zeyuan Allen-Zhu, Yuanzhi Li, Rafael Oliveira, and Avi Wigderson. Much faster algorithms for matrix scaling. *arXiv preprint arXiv:1704.02315*, 2017.

[4] Zeyuan Allen-Zhu and Lorenzo Orecchia. Linear coupling: An ultimate unification of gradient and mirror descent. *arXiv preprint arXiv:1407.1537*, 2014.

[5] Zeyuan Allen-Zhu and Lorenzo Orecchia. Nearly-linear time packing and covering lp solvers. *arXiv preprint arXiv:1411.1124*, 2014.

[6] Zeyuan Allen-Zhu and Lorenzo Orecchia. Using optimization to solve positive lps faster in parallel. *arXiv preprint arXiv:1407.1925*, 2014.

[7] Zeyuan Allen-Zhu, Zheng Qu, Peter Richtárik, and Yang Yuan. Even faster accelerated coordinate descent using non-uniform sampling. In *International Conference on Machine Learning*, pages 1110–1119, 2016.

[8] Francis Bach. Statistical machine learning and convex optimization.

[9] Peter Bartlett. Cs 281b / stat 241b, spring 2008:. *http://people.eecs.berkeley.edu/~bartlett/courses/281b-sp08/*, 2008.

[10] Peter Bartlett. Cs 281b / stat 241b, 2016:. *https://bcourses.berkeley.edu/courses/1409209/pages/lectures*, 2016.

[11] Peter L. Bartlett, Dylan J. Foster, and Matus Telgarsky. Spectrally-normalized margin bounds for neural networks. *CoRR*, abs/1706.08498, 2017.

[12] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[13] Sébastien Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.

[14] John C. Duchi and Hongseok Namkoong. Variance-based regularization with convex objectives. *https://arxiv.org/pdf/1610.02581.pdf*, 2017.

[15] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 3 1956.

[16] Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages I–427–I–435. JMLR.org, 2013.

[17] Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

[18] Llew Mason, Jonathan Baxter, Peter L Bartlett, and Marcus R Frean. Boosting algorithms as gradient descent. In *Advances in neural information processing systems*, pages 512–518, 2000.

[19] Yu Nesterov. Smooth minimization of non-smooth functions. *Mathematical programming*, 103(1):127–152, 2005.

[20] Yurii Nesterov. A method of solving a convex programming problem with convergence rate o (1/k2).

[21] Philippe Rigollet. 18.657 mathematics of machine learning. fall 2015. *https://ocw.mit.edu/courses/mathematics/18-657-mathematics-of-machine-learning-fall-2015/lecture-notes/*, 2015.

[22] Yuting Wei, Fanny Yang, and Martin J. Wainwright. Early stopping for kernel boosting algorithms: A general analysis with localized complexities. *https://arxiv.org/pdf/1707.01543.pdf*, 2017.