Lecture Notes Online Learning, Part 2

Online Convex Optimization

March the 15th, 2018

The lecture notes will cover three sections.

- First we are going to state the setup of Online Convex Optimization (OCO)
- Since OCO is based on convex methods, we quickly sketch two interesting properties of convex sets and convex functions
- We then formulate the our strategy to get good regret bounds in the OCO setting

1 The OCO Setup

In the expert setting of the previous lecture we played a game, where in each round we could chose a distribution over those experts. In the OCO setup we want to get rid of the notion of experts, in some settings we might not have access to those. For this consider the following example:

Example: Spam Filter We want to classify a mail as spam or not spam. For each mail, we get a d dimensional feature-vector $x \in \mathcal{X}$, together with a label $y \in \{-1, 1\}$, where -1 means that the mail is spam, and 1 that the mail is not spam. We chose a simple linear bounded classifier $a \in [-1, 1]^d$ and try to minimize the squared error $l(a, x, y) = (a^T x - y)^2$. If this error is small we classify the mail correctly. Moving to the online setting, we can update our classifier $a_{t+1} \in \mathcal{A}$ after each round t, after seeing a new mail $z_t = (x_t, y_t)$ in round t.

Similar to the expert setting we play an ongoing game against an adversary. The difference is that we do not have experts anymore. So our actions are not distributions over experts anymore, but more abstract points in a convex and closed set $\mathcal{A}$. We thus play the following game.

- At time t we pick an action $a_t \in \mathcal{A}$
- An adversary picks an action $z_t \in \mathcal{Z}$
- We observe the loss $l(a_t, z_t)$

For the loss function $l(a_t, z_t)$ we ask that it is convex and differentiable with respect to a_t. Going back to the spam filter example we see that $\mathcal{A} = [-1, 1]^d$ was the set of linear classifiers, the adversary space $Z = \mathcal{X} \times \{-1, 1\}$ is the space of possible mails together with the label (spam, no spam) and $l(a_t, z_t)$ is the squared loss.

Example: Mix Loss The OCO setting is actually a generalization of the expert setting. Given d experts, the action set $\mathcal{A}$ is given by $\mathcal{A} = \Delta_d$, while the adversary set $\mathcal{Z}$ is given by $Z = (-\infty, \infty]^d$. The loss function is given by the mix loss. The only thing that changes here is our performance measure In the expert setting we measured the performance of a strategy a_t by the expert regret. Since we do not have experts any more, we compare our strategy now to the best fixed action instead to the best expert, and define this way the OCO regret.

Definition 1 *Let $\mathcal{A}$ be a convex and closed action space and $l : \mathcal{A} \times \mathcal{Z} \to \mathbb{R}$ be a convex and differentiable loss function. The* ***OCO regret*** *after n rounds for a given player strategy $a_t \in \mathcal{A}$ and adversary strategy $z_t \mathcal{Z}$ with $1 \leq t \leq n$ is given by*

$$R_n = \sum_{t=1}^{n} l(a_t, z_t) - \min_{a \in \mathcal{A}} \sum_{t=1}^{n} l(a, z_t).$$

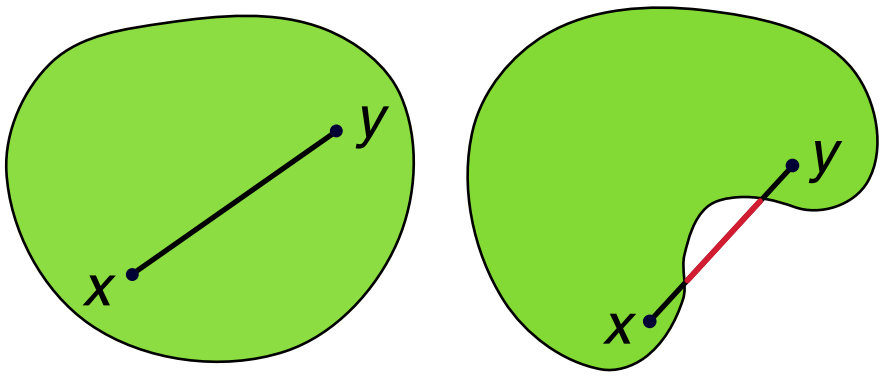
2 Convex Sets and convex Functions

In the OCO setup we assumed that the action space $\mathcal{A}$ is convex (and closed) and the loss function is convex and differentiable. This restriction is necessary to still be able to get good regret bounds. In the following subsections we will point out the specific properties that we are going to use.

2.1 Convex Sets

Convex sets have the property that if we take two points in them and form the intersecting line between them, then all points on that line are again in this set as illustrated in Figure 1. The formal definition is the following.

Definition 2 *A set $A \subset \mathbb{R}^d$ is called convex if for every $a, b \in A$ and $\lambda \in [0, 1]$ the point $\lambda a + (1 - \lambda)b$ is also in A.*



(a) Convex Set (b) Non-Convex Set

Figure 1: [1]Illustration of a convex and a non-convex subset of $\mathbb{R}^2$

For the online gradient descent algorithm introduced in this lecture we will need the following notation. Let $A \subset \mathbb{R}^d$ be a closed and convex set, and $w \in \mathbb{R}^d$. We want to be able to project the point w onto the convex set A. We do that by replacing w with the closest point in the convex set A. Formally we introduce the projection as follows.

Definition 3 *The projection $\Pi_A(w)$ of an point $w \in \mathbb{R}^d$ onto a closed, convex set $A \subset \mathbb{R}^d$ is defined as*

$$\Pi_A(w) = \arg\min_{a \in A} ||w - a||_2$$

Where $||.||_2$ is the euclidean norm in $\mathbb{R}^d$.

[1]pictures by CheCheDaWaff, distributed under a CC-BY 4.0 license.

The reason we use convex sets for our algorithm is a property known as Pythagoras theorem. It basically states that the distance from any point in a convex set is smaller to the projected point in comparison to the not projected point.

Theorem 1 ***(Pythagoras)***
Let $A \subset \mathbb{R}^d$ be a closed, convex set. Then for all points $a \in A$ and $w \in \mathbb{R}^d$ the following inequality holds.

$$||\Pi_A(w) - a||_2 \leq ||w - a||_2$$

2.2 Convex Functions

If a function $l : \mathbb{R} \to \mathbb{R}$ is differentiable then it is convex if the line defined by the gradient in each point is below the function. Formally this function is convex if the following inequality holds.

$$l(b) \geq l(a) + \frac{\partial}{\partial a} l(a)(b - a) \tag{1}$$

If we fix a and keep b variable then we can think of the right hand side of this inequality as the line segment through $l(a)$ and with a slope defined by the gradient. This intuition is captured in Figure 2.

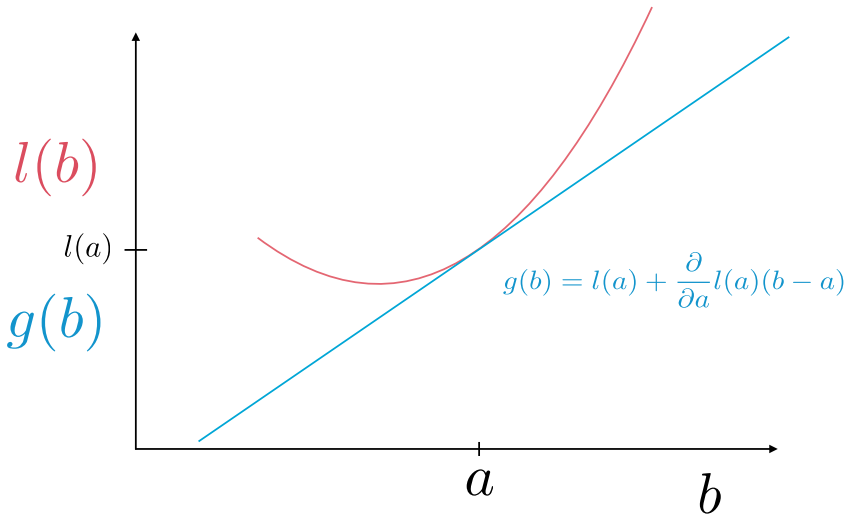


Figure 2: The red graph illustrates a convex and differentiable one dimensional function. The blue graph depicts the gradient in point a and we see that it is always below the convex function.

This idea can be generalized for functions in multiple variables $l : \mathbb{R}^d \to \mathbb{R}$. We then change the derivative $\frac{\partial}{\partial a} l(a) : \mathbb{R} \to \mathbb{R}$ for the gradient $\nabla l(a) : \mathbb{R}^d \to \mathbb{R}^d$. Generalizing to multiple dimensions and reordering (1) we get the following property of convex, differentiable functions.

Lemma 1 *If $l : \mathbb{R}^d \to \mathbb{R}$ is a convex and differentiable function then the following inequality holds.*

$$l(a) - l(b) \leq \nabla^T l(a)(a - b) \tag{2}$$

Here $\nabla^T l(b)$ denotes the transpose of the gradient $\nabla l(b)$.

3 Online Gradient Descent

Our goal is now to achieve a bound for the OCO regret R_n that grows slower than linear in n. We could try to use a similar strategy as in in the expert setting by replacing the discrete objects in there with continuous ones. This can be done indeed, as showed for example in [Bub11] in chapter 3. It turns out, however, that we can define a more efficient algorithm with a better OCO regret bound. The way to achieve this is a very simple algorithm called Online Gradient Descent (OGD). The method of gradient descent is a well known technique in convex optimization for finding extremum points. The basic idea is to start with an initial guess for an extremum and then follow the direction of the negative gradient as illustrated in Figure 3. The gradient will provide us the right direction to walk into, but we have to decide how far we go that direction. This will depend on a parameter $\eta \in \mathbb{R}$ set by us, the learning rate. Note that the actions we are

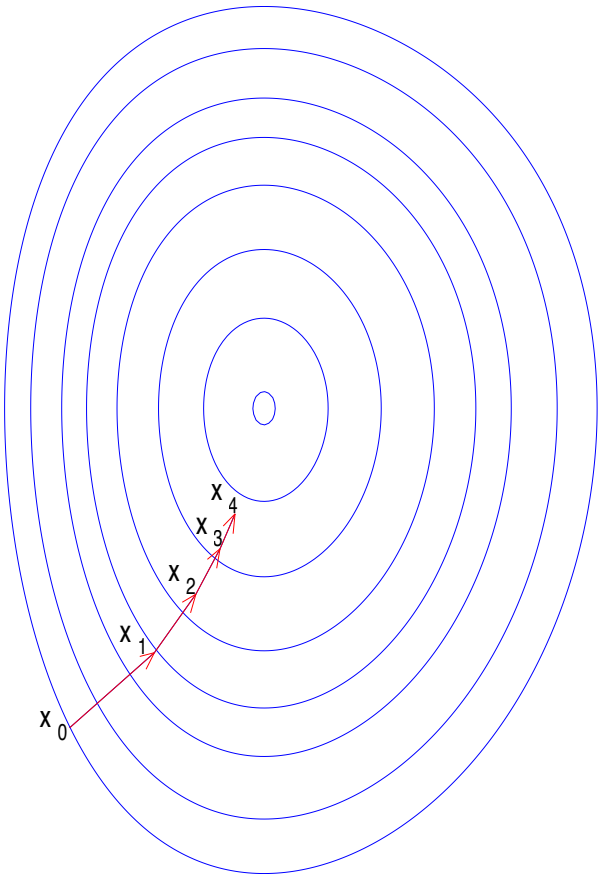


Figure 3: This picture shows the level sets of a three dimensional graph, so basically the projection onto the x-y plane. The values of the function decrease with each circle going closer to the center. The minimum is thus found in the little most inner circle. The sequence from x_0 to x_5 depicts the behavior of the gradient descent algorithm.

allowed to take are defined by the action space $\mathcal{A}$. When we use gradient descent it can happen that we end up with a new solution that is not in $\mathcal{A}$ anymore. To solve this problem we project this new solution then back to $\mathcal{A}$. So formally OGD with learning rate η iterates the following two steps.

Definition 4 *Fix a convex, closed action space $\mathcal{A}$, a convex, differentiable loss function $l : \mathcal{A} \times$*

$\mathcal{Z} \to \mathbb{R}$ *and an initial action* $a_1 \in \mathcal{A}$*. The strategy given by* ***Online Gradient Descent*** *with learning rate* η *is defined by the following two iterating steps.*

- *Set* $w_{t+1} = a_t - \eta \nabla_a l(a_t, z_t)$
- *Set* $a_{t+1} = \Pi_A(w_{t+1})$

The term $\Pi_A(w)$ *here is the projection of* w *onto the set* A*.*

This strategy achieves the following regret bound.

Theorem 2 *Assume the setting from the previous definition. Assume further that* $||a||_2 \leq R$ *for all* $a \in \mathcal{A}$ *and* $||\nabla l(a, z)||_2 \leq G$ *for all* $(a, z) \in \mathcal{A} \times \mathcal{Z}$*. The strategy given by the Online Gradient Descent with learning rate* η *and* $a_1 = 0$ *achieves a bound of the regret given by*

$$R_n \leq \frac{R^2}{2\eta} + \frac{\eta G n}{2}. \tag{3}$$

For a learning rate $\eta = \frac{R}{G\sqrt{n}}$ *we can thus bound*

$$R_n \leq RG\sqrt{n} \tag{4}$$

The proof of this Theorem is found in [Bub11].

Proof Let $a \in \mathcal{A}$ be an arbitrary action. We will show that the OCO regret bound holds for this a. This is enough, since if the bound holds for an arbitrary a, then it also holds for the minimizing a as defined in the OCO regret.
First we use the inequality of convex and differentiable functions as in 2.

$$l(a_t, z_t) - l(a, z_t) \leq \nabla_{a_t}^T l(a_t, z_t)(a_t - a)$$

This holds for all t, so we can sum this over n rounds:

$$R_n^a := \sum_{t=1}^{n} l(a_t, z_t) - l(a, z_t) \leq \sum_{t=1}^{n} \nabla_{a_t}^T l(a_t, z_t)(a_t - a) \tag{5}$$

The left hand side side is the OCO regret, expect that we compare to a and not to the best a. So we can continue now bounding the right hand side. First look at the following inequality.

$$||a_{t+1} - a||^2 = ||\Pi_{\mathcal{A}}(a_t - \eta \nabla_{a_t} l(a_t, z_t)) - a||^2 \leq ||a_t - \eta \nabla_{a_t} l(a_t, z_t) - a||^2 = ||a_t - a||^2 - 2\eta \nabla_{a_t}^T l(a_t - z_t) + ||\eta \nabla_{a_t} l(a_t, z_t)||^2$$

The first equality follows from the definition of the OGD. The inequality follows from the Pythagoras Theorem. The last equality is just expanding the squared norm. This can be re-written as follows:

$$2\eta \nabla_{a_t}^T l(a_t - z_t) \leq ||a_t - a||^2 - ||a_{t+1} - a||^2 + ||\eta \nabla_{a_t} l(a_t, z_t)||^2$$

If we sum that now over n turns we get:

$$\sum_{t=1}^{n} 2\eta \nabla_{a_t}^T l(a_t - z_t) \leq \sum_{t=1}^{n} ||a_t - a||^2 - ||a_{t+1} - a||^2 + ||\eta \nabla_{a_t} l(a_t, z_t)||^2 \tag{6}$$

The sum on the right hand side telescopes, meaning that in each round several terms cancel. So we can rewrite this sum as follows:

$$\sum_{t=1}^{n} ||a_t - a||^2 - ||a_{t+1} - a||^2 + ||\eta \nabla_{a_t} l(a_t, z_t)||^2$$

$$= ||a_1 - a||^2 - ||a_{n+1} - a||^2 + \sum_{t=1}^{n} \eta^2 ||\nabla_{a_t} l(a_t, z_t)||^2 \leq R^2 + n\eta^2 G^2 \tag{7}$$

For the inequality we just used the bounds $||a|| \leq R$ and $||\nabla_a l(a, z)|| \leq G$. Combining Inequality (6) and (7) we derive by dividing with 2η:

$$\sum_{t=1}^{n} \nabla_{a_t}^T l(a_t, z_t)(a_t - a) \leq \frac{R^2 + n\eta^2 G^2}{2\eta} \tag{8}$$

Combining Inequality (5) and (8) we derive the final result:

$$\sum_{t=1}^{n} l(a_t, z_t) - l(a, z_t) \leq \frac{R^2 + n\eta^2 G^2}{2\eta}$$

References

[Bub11] Sébastien Bubeck. Introduction to online optimization. *Lecture Notes*, pages 1–86, 2011.