Course Notes for EE227C (Spring 2018): Convex Optimization and Approximation

Instructor: Moritz Hardt

Email: hardt+ee227c@berkeley.edu

Graduate Instructor: Max Simchowitz

Email: msimchow+ee227c@berkeley.edu

February 8, 2018

Abstract

These are course notes for EE227C (Spring 2018): Convex Optimization and Approximation, taught at UC Berkeley. For further information, see the course page at:

https://ee227c.github.io/.

Contents

1 Lecture 1: Convexity

This lecture provides the most important facts about convex sets and convex functions that we'll heavily make use of. These are often simple consequences of Taylor's theorem.

1.1 Convex sets

Definition 1.1 (Convex set)**.** A set $K \subseteq \mathbb{R}^n$ is *convex* if it the line segment between any two points in K is also contained in K. Formally, for all $x, y \in K$ and all scalars $\gamma \in [0, 1]$ we have $\gamma x + (1 - \gamma)y \in K$.

Theorem 1.2 (Separation Theorem)**.** *Let $C, K \subseteq \mathbb{R}^n$ be convex sets with empty intersection $C \cap K = \emptyset$. Then there exists a point $a \in \mathbb{R}^n$ and a number $b \in \mathbb{R}$ such that*

1. *for all* $x \in C$, *we have* $\langle a, x \rangle \geqslant b$.

2. *for all* $x \in K$, *we have* $\langle a, x \rangle \leqslant b$.

If C and K are closed and at least one of them is bounded, then we can replace the inequalities by strict inequalities.

The case we're most concerned with is when both sets are compact (i.e., closed and bounded). We highlight its proof here.

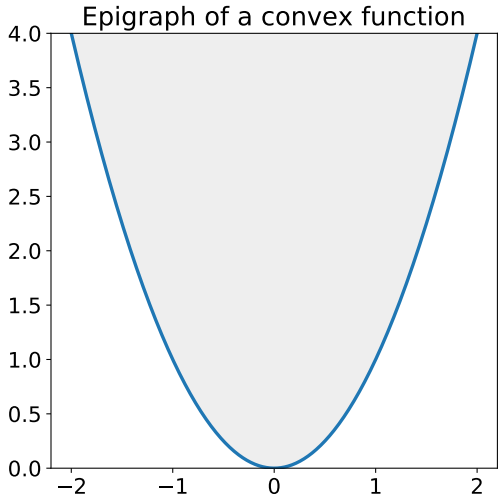
Proof of Theorem 1.2 for compact sets. In this case, the Cartesian product $C \times K$ is also compact. Therefore, the distance function $\|x - y\|$ attains its minimum over $C \times K$. Taking p, q to be two points that achieve the minimum. A separating hyperplane is given by the hyperplane perpendicular to $q - p$ that passes through the midpoint between p and q. That is, $a = q - p$ and $b = (\langle a, q \rangle - \langle a, p \rangle)/2$. For the sake of contradiction, suppose there is a point r on this hyperplane contained in one of the two sets, say, C. Then the line segment from p to r is also contained in C by convexity. We can then find a point along the line segment that is close to q than p is, thus contradicting our assumption. ■

1.1.1 Notable convex sets

- Linear spaces $\{x \in \mathbb{R}^n \mid Ax = 0\}$ and halfspaces $\{x \in \mathbb{R}^n \mid \langle a, x \rangle \geqslant 0\}$
- Affine transformations of convex sets. If $K \subseteq \mathbb{R}^n$ is convex, so is $\{Ax + b \mid x \in K\}$ for any $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. In particular, affine subspaces and affine halfspaces are convex.
- Intersections of convex sets. In fact, every convex set is equivalent to the intersection of all affine halfspaces that contain it (a consequence of the separating hyperplane theorem).
- The cone of positive semidefinite matrices, denotes, $S_+^n = \{A \in \mathbb{R}^{n \times n} \mid A \succeq 0\}$. Here we write $A \succeq 0$ to indicate that $x^\top A x \geqslant 0$ for all $x \in \mathbb{R}^n$. The fact that S_+^n is convex can be verified directly from the definition, but it also follows from what we already knew. Indeed, denoting by $S_n = \{A \in \mathbb{R}^{n \times n} \mid A^\top = A\}$ the set of all $n \times n$ symmetric matrices, we can write S_+^n as an (infinite) intersection of halfspaces $S_+^n = \bigcap_{x \in \mathbb{R}^n \setminus \{0\}} \{A \in S_n \mid x^\top A x \geqslant 0\}$.
- See Boyd-Vandenberghe for lots of other examples.

1.2 Convex functions

Definition 1.3 (Convex function)**.** A function $f : \Omega \to \mathbb{R}$ is *convex* if for all $x, y \in \Omega$ and all scalars $\gamma \in [0, 1]$ we have $f(\gamma x + (1 - \gamma)y) \leqslant \gamma f(x) + (1 - \gamma) f(y)$.



Jensen (1905) showed that for continuous functions, convexity follows from the "midpoint" condition that for all $x, y \in \Omega$,

$$f\left(\frac{x+y}{2}\right) \leqslant \frac{f(x)+f(y)}{2}.$$

This result sometimes simplifies the proof that a function is convex in cases where we already know that it's continuous.

Definition 1.4. The *epigraph* of a function $f\colon \Omega \to \mathbb{R}$ is defined as

$$\operatorname{epi}(f) = \{(x,t) \mid f(x) \leqslant t\}.$$

Fact 1.5. *A function is convex if and only if its epigraph is convex.*

Convex functions enjoy the property that local minima are also global minima. Indeed, suppose that $x \in \Omega$ is a local minimum of $f\colon \Omega \to \mathbb{R}$ meaning that any point in a neighborhood around x has larger function value. Now, for every $y \in \Omega$, we can find a small enough γ such that

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

Therefore, $f(x) \leqslant f(y)$ and so x must be a global minimum.

1.2.1 First-order characterization

It is helpful to relate convexity to Taylor's theorem, which we recall now. We define the *gradient* of a differentiable function $f\colon \Omega \to \mathbb{R}$ at $x \in \Omega$ as the vector of partial derivatives

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_i}\right)_{i=1}^{n}.$$

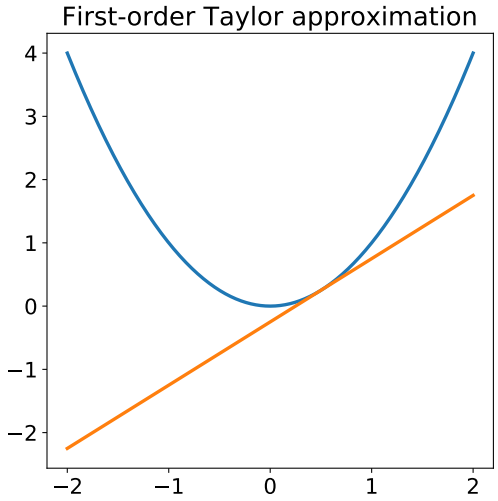


Figure 1: Taylor approximation of $f(x) = x^2$ at 0.5.

We note the following simple fact that relates linear forms of the gradient to a one-dimensional derivative evaluated at 0. It's a consequence of the multivariate chain rule.

Fact 1.6. *Assume $f : \Omega \to \mathbb{R}$ is differentiable and let $x, y \in \Omega$. Then,*

$$\nabla f(x)^\top y = \left.\frac{\partial f(x + \gamma y)}{\partial \gamma}\right|_{\gamma=0}.$$

Taylor's theorem implies the following statement.

Proposition 1.7. *Assume $f : \Omega \to \mathbb{R}$ is continuously differentiable along the line segment between two points x and y. Then,*

$$f(y) = f(x) + \nabla f(x)^\top (y - x) + \int_0^1 (1 - \gamma) \frac{\partial^2 f(x + \gamma(y - x))}{\partial \gamma^2} \, \mathrm{d}\gamma$$

Proof. Apply a second order Taylor's expansion to $g(\gamma) = f(x + \gamma(y - x))$ and apply Fact 1.6 to the first-order term. ■

Among differentiable functions, convexity is equivalent to the property that the first-order Taylor approximation provides a global lower bound on the function.

Proposition 1.8. *Assume $f : \Omega \to \mathbb{R}$ is differentiable. Then, f is convex if and only if for all $x, y \in \Omega$ we have*

$$f(y) \geqslant f(x) + \nabla f(x)^\top (y - x). \tag{1}$$

Proof. First, suppose f is convex, then by definition

$$\begin{aligned} f(y) &\geqslant \frac{f((1-\gamma)x+\gamma y)-(1-\gamma)f(x)}{\gamma} \\ &\geqslant f(x)+\frac{f(x+\gamma(y-x))-f(x)}{\gamma} \\ &\to f(x)+\nabla f(x)^\top (y-x) \quad \text{as } \gamma \to 0 \qquad \text{(by Fact 1.6.)} \end{aligned}$$

On the other hand, fix two points $x, y \in \Omega$ and $\gamma \in [0,1]$. Putting $z = \gamma x + (1-\gamma)y$ we get from applying Equation 1 twice,

$$f(x) \geqslant f(z) + \nabla f(z)^\top (x-z) \quad \text{and} \quad f(y) \geqslant f(z) + \nabla f(z)^\top (y-z)$$

Adding these inequalities scaled by γ and $(1-\gamma)$, respectively, we get $\gamma f(x) + (1-\gamma)f(y) \geqslant f(z)$, which establishes convexity. ∎

A direct consequence of Proposition 1.8 is that if $\nabla f(x) = 0$ vanishes at a point x, then x must be a global minimizer of f.

Remark 1.9 (Subgradients)**.** *Of course, not all convex functions are differentiable. The absolute value $f(x) = |x|$, for example, is convex but not differentiable at 0. Nonetheless, for every x, we can find a vector g such that*

$$f(y) \geqslant f(x) + g^\top (y-x)\,.$$

Such a vector is called a subgradient *of f at x. The existence of subgradients is often sufficient for optimization.*

1.2.2 Second-order characterization

We define the *Hessian* matrix of $f\colon \Omega \to \mathbb{R}$ at a point $x \in \Omega$ as the matrix of second order partial derivatives:

$$\nabla^2 f(x) = \left(\frac{\partial^2 f}{\partial x_i \partial x_j}\right)_{i,j\in[n]}.$$

Schwarz's theorem implies that the Hessian at a point x is symmetric provided that f has continuous second partial derivatives in an open set around x.

In analogy with Fact 1.6, we can relate quadratic forms in the Hessian matrix to one-dimensional derivatives using the chain rule.

Fact 1.10. *Assume that $f\colon \Omega \to \mathbb{R}$ is twice differentiable along the line segment from x to y. Then,*

$$y^\top \nabla^2 f(x+\gamma y) y = \frac{\partial^2 f(x+\gamma y)}{\partial \gamma^2}\,.$$

Proposition 1.11. *If f is twice continuously differentiable on its domain Ω, then f is convex if and only if $\nabla^2 f(x) \succeq 0$ for all $x \in \Omega$.*

Proof. Suppose f is convex and our goal is to show that the Hessian is positive semidefinite. Let $y = x + \alpha u$ for some arbitrary vector u and scalar α. Proposition 1.8 shows

$$f(y) - f(x) - \nabla f(x)^\top (y - x) \geqslant 0$$

Hence, by Proposition 1.7,

$$\begin{aligned} 0 &\leqslant \int_0^1 (1-\gamma) \frac{\partial^2 f(x + \gamma(y-x))}{\partial \gamma^2} \, \mathrm{d}\gamma \\ &= (1-\gamma) \frac{\partial^2 f(x + \gamma(y-x))}{\partial \gamma^2} \quad \text{for some } \gamma \in (0,1) \qquad \text{(by the mean value theorem)} \\ &= (1-\gamma)(y-x)^\top \nabla^2 f(x + \gamma(y-x))(y-x)\,. \qquad \text{(by Fact 1.10)} \end{aligned}$$

Plugging in our choice of y, this shows $0 \leqslant u^\top \nabla^2 f(x + \alpha\gamma u) u$. Letting α tend to zero establishes that $\nabla^2 f(x) \succeq 0$. (Note that γ generally depends on α but is always bounded by 1.)

Now, suppose the Hessian is positive semidefinite everywhere in Ω and our goal is to show that the function f is convex. Using the same derivation as above, we can see that the second-order error term in Taylor's theorem must be nonnegative. Hence, the first-oder approximation is a global lower bound and so the function f is convex by Proposition 1.8. ■

1.3 Convex optimization

Much of this course will be about different ways of minimizing a convex function $f : \Omega \to \mathbb{R}$ over a convex domain Ω :

$$\min_{x \in \Omega} f(x)$$

Convex optimization is not necessarily easy! For starters, convex sets do not necessarily enjoy compact descriptions. When solving computational problems involving convex sets, we need to worry about how to represent the convex set we're dealing with. Rather than asking for an explicit description of the set, we can instead require a computational abstraction that highlights essential operations that we can carry out. The Separation Theorem motivates an important computational abstraction called *separation oracle*.

Definition 1.12. A *separation oracle* for a convex set K is a device, which given any point $x \notin K$ returns a hyperplane separating x from K.

Another computational abstraction is a *first-order oracle* that given a point $x \in \Omega$ returns the gradient $\nabla f(x)$. Similarly, a *second-order oracle* returns $\nabla^2 f(x)$. A function value oracle or *zeroth-order oracle* only returns $f(x)$. First-order methods are algorithms that make do with a first-order oracle.

1.3.1 What is efficient?

Classical complexity theory typically quantifies the resource consumption (primarily running time or memory) of an algorithm in terms of the bit complexity of the input. This approach can be cumbersome in convex optimization and most textbooks shy away from it. Instead, it's customary in optimization to quantify the cost of the algorithm in terms of how often it accesses one of the oracles we mentioned.

The definition of "efficient" is not completely cut and dry in optimization. Typically, our goal is to show that an algorithm finds a solution x with $f(x) = \min_{x\in\Omega} f(x) + \epsilon$ for some additive error $\epsilon > 0$. The cost of the algorithm will depend on the target error. Highly practical algorithms often have a polynomial dependence on ϵ, such as $O(1/\epsilon)$ or even $O(1/\epsilon^2)$. Other algorithms achieve $O(\log(1/\epsilon))$ steps in theory, but are prohibitive in their actual computational cost. Technically, if we think of the parameter ϵ as being part of the input, it takes only $O(\log(1/\epsilon))$ bits to describe the error parameter. Therefore, an algorithm that depends more than logarithmically on $1/\epsilon$ may not be polynomial time algorithm in its input size.

In this course, we will make an attempt to highlight both the theoretical performance and practical appeal of an algorithm. Moreover, we will discuss other performance criteria such as robustness to noise. How well an algorithm performs is rarely decided by a single criterion, and usually depends on the application at hand.

2 Lecture 2: Gradient method

In this lecture we encounter the fundamentally important *gradient method* and a few ways to analyze its convergence behavior. The goal here is to solve a problem of the form

$$\min_{x\in\Omega} f(x)$$

where we'll make some additional assumptions on the function $f : \Omega \to \mathbb{R}$. The technical exposition closely follows the corresponding chapter in Bubeck's text [Bub15].

2.1 Gradient descent

For a differentiable function f, the basic gradient method startingf from an initial point x_0 is defined by the iterative description

$$x_{t+1} = x_t - \eta \nabla_t f(x_t) \qquad\qquad (t \geqslant 0)$$

where η_t is the so-called *step size* that may vary with t.

The first assumption that leads to a convergence analysis is that the gradients of the function aren't too big over the domain.

Definition 2.1 (L-Lipschitz)**.** A differentiable function f is said to be *L-Lipschitz* over the domain Ω if for all $x \in \Omega$, we have

$$\|\nabla f(x)\| \leqslant L\,.$$

Fact 2.2. *If a function f is L-Lipschitz, it implies that the difference between two points in the range is bounded,*

$$|f(x) - f(y)| \leqslant L\|x - y\|$$

How can we ensure that $x_{t+1} \in \Omega$? One natural approach is to "project" each iterate back onto the domain Ω.

Definition 2.3 (Projection)**.** The *projection* of a point x onto a set Ω is defined as

$$\Pi_\Omega(x) = \operatorname*{argmin}_{y \in \Omega} \|x - y\|$$

Example 2.4. A projection onto the Euclidean ball B_2 is just normalization:

$$\Pi_{B_2}(x) = \frac{x}{\|x\|}$$

A crucial property of projections is that when $x \in \Omega$, we have for any y (possibly outside Ω):

$$\|\Pi_\Omega(y) - x\|^2 \leqslant \|y - x\|^2$$

That is, the projection of y onto a convex set containing x is closer to x. See Figure 2 for a geometric picture.

Lemma 2.5.

$$\|\Pi_\Omega(y) - x\|^2 \leqslant \|y - x\|^2 - \|y - \Pi_\Omega(y)\|^2$$

Which follows from the Pythagorean theorem. Note that this lemma implies the above property.

2.1.1 Modifying gradient descent with projections

So now we can modify our original procedure to use two steps.

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

$$x_{t+1} = \Pi_\Omega(y_{t+1})$$

And we are guaranteed that $x_{t+1} \in \Omega$. Note that computing the projection may be the hardest part of your problem, as you are computing an argmin. However, there are convex sets for which we know explicitly how to compute the projection (see Example 2.4).

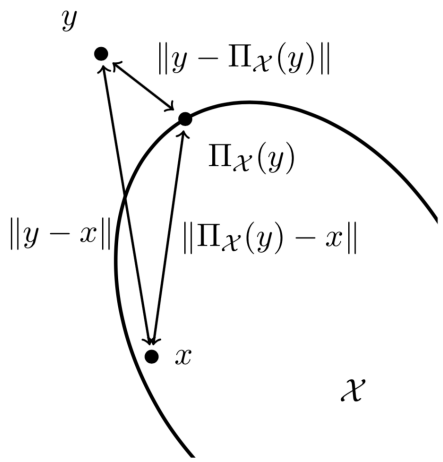


Figure 2: Projection of y onto set $\mathcal{X}$. (Figure taken from *S. Bubeck. Convex Optimization: Algorithms and Complexity [Bub15])*

2.2 Convergence rate of gradient descent for Lipschitz functions

Theorem 2.6 (Projected Gradient Descent for L-Lipschitz Functions)**.** *Assume that function f is convex, differentiable, and closed with bounded gradients. Let L be the Lipchitz constant of f over the convex domain Ω. Let R be the upper bound on the distance $\|x_1 - x^*\|_2$ from the initial point x_1 to the optimal point $x^* = \arg\min_{x\in\Omega} f(x)$. Let t be the number of iterations of project gradient descent. If the learning rate η is set to $\eta = \frac{R}{L\sqrt{(t)}}$, then*

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

This means that the difference between the functional value of the average point during the optimization process from the optimal value is bounded above by a constant proportional to $\frac{1}{\sqrt{t}}$.

Before proving the theorem, recall the "Fundamental Theorem of Optimization", which is that an inner product can be written as a sum of norms: $u^\top v = \frac{1}{2}(\|u\|^2 + \|v\|^2 - \|u - v\|^2)$. This property can be seen from $\|u - v\|^2 = \|u\|^2 + \|v\|^2 - 2u^\top v$.

Proof of Theorem 2.6. The proof begins by first bounding the difference in function values $f(x_s) - f(x^*)$.

$$\begin{aligned}
f(x_s) - f(x^*) &\leqslant \nabla f(x_s)^\top (x_s - x^*) && (2)\\
&= \frac{1}{\eta}(x_s - y_{s+1})^\top (x_s - x^*) && (3)\\
&= \frac{1}{2\eta}\left(\|x_s - x^*\|^2 + \|x_s - y_{s+1}\|^2 - \|y_{s+1} - x^*\|^2\right) && (4)\\
&= \frac{1}{2\eta}\left(\|x_s - x^*\|^2 - \|y_{s+1} - x^*\|^2\right) + \frac{\eta}{2}\|\nabla f(x_s)\|^2 && (5)\\
&\leqslant \frac{1}{2\eta}\left(\|x_s - x^*\|^2 - \|y_{s+1} - x^*\|^2\right) + \frac{\eta L^2}{2} && (6)\\
&\leqslant \frac{1}{2\eta}\left(\|x_s - x^*\|^2 - \|x_{s+1} - x^*\|^2\right) + \frac{\eta L^2}{2} && (7)
\end{aligned}$$

Equation 2 comes from the definition of convexity. Equation 3 comes from the update rule for projected gradient descent. Equation 4 comes from the "Fundamental Theorem of Optimization." Equation 5 comes from the update rule for projected gradient descent. Equation 6 is because f is L-Lipchitz. Equation 7 comes from Lemma 2.5.

Now, sum these differences from $s = 1$ to $s = t$:

$$\sum_{s=1}^{t} f(x_s) - f(x^*) \leqslant \frac{1}{2\eta} \sum_{s=1}^{t} \left(\|x_s - x^*\|^2 - \|x_{s+1} - x^*\|^2 \right) + \frac{\eta L^2 t}{2} \tag{8}$$

$$= \frac{1}{2\eta} \left(\|x_1 - x^*\|^2 - \|x_t - x^*\|^2 \right) + \frac{\eta L^2 t}{2} \tag{9}$$

$$\leqslant \frac{1}{2\eta} \|x_1 - x^*\|^2 + \frac{\eta L^2 t}{2} \tag{10}$$

$$\leqslant \frac{R^2}{2\eta} + \frac{\eta L^2 t}{2} \tag{11}$$

Equation 9 is because Equation 8 is a telescoping sum. Equation 10 is because $\|x_t - x^*\|^2 \geqslant 0$. Equation 11 is by the assumption that $\|x_1 - x^*\|^2 \leqslant R^2$.

Finally,

$$f\left(\frac{1}{t}\sum_{s=1}^{t} x_s\right) - f(x^*) \leqslant \frac{1}{t}\sum_{s=1}^{t} f(x_s) - f(x^*) \qquad \text{(by convexity)}$$

$$\leqslant \frac{R^2}{2\eta t} + \frac{\eta L^2}{2} \qquad \text{(by Equation 11)}$$

$$\leqslant \frac{RL}{\sqrt{t}} \qquad \text{(for } \eta = R/L\sqrt{t}.\text{)}$$

■

2.3 Convergence rate for smooth functions

The next property we'll encounter is called *smoothness* and it often leads to stronger convergence guarantees.

Definition 2.7 (β-smoothness)**.** A continuously differentiable function f is β smooth if the gradient ∇f is β-Lipschitz, i.e

$$\|\nabla f(x) - \nabla f(y)\| \leqslant \beta \|x - y\|$$

Lemma 2.8. *Let f be a β-smooth function on $\mathbb{R}^n$. Then for any $x, y \in \mathbb{R}^n$, one has*

$$|f(x) - f(y) - \nabla f(y)^\top (x - y)| \leqslant \frac{\beta}{2} \|x - y\|^2$$

Proof. First represent $f(x) - f(y)$ as an integral, apply Cauchy-Schwarz and then β-smoothness:

$$
\begin{aligned}
&|f(x) - f(y) - \nabla f(y)^\top (x-y)| \\
&\qquad = \left| \int_0^1 \nabla f(y + t(x-y))^\top (x-y) dt - \nabla f(y)^\top (x-y) \right| \\
&\qquad \leqslant \int_0^1 \|\nabla f(y + t(x-y)) - \nabla f(y)\| \cdot \|x-y\| dt \\
&\qquad \leqslant \int_0^1 \beta t \|x-y\|^2 dt \\
&\qquad = \frac{\beta}{2} \|x-y\|^2
\end{aligned}
$$

■

We also need the following lemma.

Lemma 2.9. *Let f be a β-smooth function, then for any $x, y \in \mathbb{R}^n$, one has*

$$
f(x) - f(y) \leqslant \nabla f(x)^\top (x-y) - \frac{1}{2\beta} \|\nabla f(x) - \nabla f(y)\|
$$

Proof. Let $z = y - \frac{1}{\beta}(\nabla f(y) - \nabla f(x))$. Then one has

$$
\begin{aligned}
&f(x) - f(y) \\
&= f(x) - f(z) + f(z) - f(y) && (12) \\
&\leqslant \nabla f(x)^\top (x-z) + \nabla f(y)^\top (z-y) + \frac{\beta}{2} \|z-y\|^2 && (13) \\
&= \nabla f(x)^\top (x-y) + (\nabla f(x) - \nabla f(y))^\top (y-z) + \frac{1}{2\beta} \|\nabla f(x) - \nabla f(y)\|^2 && (14) \\
&= \nabla f(x)^\top (x-y) - \frac{1}{2\beta} \|\nabla f(x) - \nabla f(y)\|^2 && (15)
\end{aligned}
$$

■

We will show that gradient descent with the update rule

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

attains a faster rate of convergence under the smoothness condition.

Theorem 2.10. *Let f be convex and β-smooth on $\mathbb{R}^n$ then gradient descent with $\eta = \frac{1}{\beta}$ satsifies*

$$f(x_t) - f(x^*) \leqslant \frac{2\beta\|x_1 - x^*\|^2}{t-1}$$

To prove this we will need the following two lemmas.

Proof. By the update rule and lemma 2.8 we have

$$f(x_{s+1}) - f(x_s) \leqslant -\frac{1}{2\beta}\|\nabla f(x_s)\|^2$$

In particular, denoting $\delta_s = f(x_s) - f(x^*)$ this shows

$$\delta_{s+1} \leqslant \delta_s - \frac{1}{2\beta}\|\nabla f(x_s)\|^2$$

One also has by convexity

$$\delta_s \leqslant \nabla f(x)s)^\top (x_s - x^*) \leqslant \|x_s - x^*\| \cdot \|\nabla f(x_s)\|$$

We will prove that $\|x_s - x^*\|$ is decreasing with s, which with the two above displays will imply

$$\delta_{s+1} \leqslant \delta_s - \frac{1}{2\beta\|x_1 - x^*\|^2}\delta_s^2$$

We solve the recurrence as follows. Let $w = \frac{1}{2\beta\|x_1 - x^*\|^2}$, then

$$w\delta_s^2 + \delta_{s+1} \leqslant \delta_s \iff w\frac{\delta_s}{\delta_{s+1}} + \frac{1}{\delta_s} \leqslant \frac{1}{\delta_{s+1}} \implies \frac{1}{\delta_{s+1}} - \frac{1}{\delta_s} \geqslant w \implies \frac{1}{\delta_t} \geqslant w(t-1)$$

To finish the proof it remains to show that $\|x_s - x^*\|$ is decreasing with s. Using lemma 2.9 one immediately gets

$$(\nabla f(x) - \nabla f(y))^\top (x - y) \geqslant \frac{1}{\beta}\|\nabla f(x) - \nabla f(y)\|^2$$

We use this and the fact that $\nabla f(x^*) = 0$

$$\begin{aligned}
\|x_{s+1} - x^*\|^2 &= \|x_s - \frac{1}{\beta}\nabla f(x_s) - x^*\|^2 && (16)\\
&= \|x_s - x^*\|^2 - \frac{2}{\beta}\nabla f(x_s)^\top (x_s - x^*) + \frac{1}{\beta^2}\|\nabla f(x_s)\|^2 && (17)\\
&\leqslant \|x_s - x^*\|^2 - \frac{1}{\beta^2}\|\nabla f(x_s)\|^2 && (18)\\
&\leqslant \|x_s - x^*\|^2 && (19)
\end{aligned}$$

which concludes the proof. ■

3 Lecture 3: Strong convexity

This lecture introduces the notion of α-strong convexity and combines it with β-smoothness to develop the concept of condition number. Adding an assumption of, respectively, strong convexity or conditioning improves the rates of error decay for gradient descent proved in the previous lecture from $O(1/\sqrt{t})$ to $O(1/t)$ and $O(1/t)$ to $O(\mathrm{e}^{-t})$.

The technical part follows the corresponding chapter in Bubeck's text [Bub15].

3.1 Reminders

Recall that we had (at least) two definitions apiece for convexity and smoothness: a general definition for all functions and a more compact definition for (twice-)differentiable functions.

A function f is convex if, for each input, there exists a globally valid *linear* lower bound on the function: $f(y) \geqslant f(x) + g^\top(x)(y - x)$. For differentiable functions, the role of g is played by the gradient.

A function f is β-smooth if, for each input, there exists a globally valid *quadratic* upper bound on the function, with (finite) quadratic parameter β: $f(y) \leqslant f(x) + g^\top(x)(y - x) + \frac{\beta}{2}\|x - y\|^2$. More poetically, a smooth, convex function is "trapped between a parabola and a line". Since β is covariant with affine transformations, e.g. changes of units of measurement, we will frequently refer to a β-smooth function as simply smooth.

For twice-differentiable functions, these properties admit simple conditions for smoothness in terms of the Hessian, or matrix of second partial derivatives. A $\mathcal{D}^2$ function f is convex if $\nabla^2 f(x) \succeq 0$ and it is β-smooth if $\nabla^2 f(x) \preceq \beta I$.

We furthermore defined the notion of L-Lipschitzness. A function f is L-Lipschitz if the amount that it "stretches" its inputs is bounded by L: $|f(x) - f(y)| \leqslant L\|x - y\|$. Note that for differentiable functions, β-smoothness is equivalent to β-Lipschitzness of the gradient.

3.2 Strong convexity

With these three concepts as sword, shield, and slightly larger shield, we were able to prove two error decay rates for gradient descent (and its projective, stochastic, and subgradient flavors). However, these rates were substantially slower than what's observed in certain settings in practice.

Noting the asymmetry between our linear lower bound (from convexity) and our quadratic upper bound (from smoothness) we introduce a new, more restricted function class by upgrading our lower bound to second order.

Definition 3.1 (α-Strong Convexity)**.** A function $f : \Omega \to \mathbb{R}$ is *α-strongly convex* if, for all

$x, y \in \Omega$, the following inequality holds for some $\alpha > 0$:

$$f(y) \geqslant f(x) + g(x)^\top (y - x) + \frac{\alpha}{2} \|x - y\|^2$$

As with smoothness, we will often shorten "α-strongly convex" to "strongly convex". A strongly convex, smooth function is one that can be "squeezed between two parabolas". If β-smoothness is a good thing, then α-convexity guarantees we don't have too much of a good thing.

Once again, twice-differentiable functions afford a quick condition: a D^2 function is $alpha$-strongly convex if $\nabla^2 f(x) \succeq \alpha I$.

Once again, note that α changes under affine transformations. Conveniently enough, for α-strongly convex, β-smooth functions, we can define a basis-independent quantity that combines these properties:

Definition 3.2 (Condition Number)**.** An α-strongly convex, β-smooth function f has *condition number* $\frac{\alpha}{\beta}$.

For a positive-definite quadratic function f, this definition of the condition number corresponds with the perhaps more familiar definition of the condition number of a matrix.

3.3 A look back and ahead

The following table summarizes the results from the previous lecture and the results to be obtained in this lecture. In both, ϵ is the difference between f at some value x' computed from the outputs of gradient descent and f calculated at an optimizer x^*.

	Convex	Strongly Convex
Lipschitz	$\epsilon \leqslant O(1/\sqrt{t})$	$\epsilon \leqslant O(1/t)$
Smooth	$\epsilon \leqslant O(1/t)$	$\epsilon \leqslant O(\mathrm{e}^{-t})$

Table 1: Bounds on error ϵ as a function of number of steps taken t for gradient descent applied to various classes of functions.

The rate for conditioned functions is frequently observed in practice, so we have reason to believe this bound is tight for a relevant class of functions. Since a rate that is exponential in terms of the magnitude of the error is linear in terms of the bit precision, this rate of convergence is termed *linear*. We now move to prove these rates.

3.4 Convergence rate strongly convex functions

Theorem 3.3. *Assume* $f \colon \Omega \to \mathbb{R}$ *is* α-strongly convex *and* L-Lipschitz. *Let* x^* *be an optimizer of* f*, and let* x_s *be the updated point at step* s *using projected gradient descent. Let the*

max number of iterations be t with an adaptive step size $\eta_s = \frac{2}{\alpha(s+1)}$, then

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

This implies the convergence rate of projected gradient descent for α-strongly convex functions is similar to that of β-smooth functions with a bound on error $\epsilon \leqslant O(1/t)$.

In order to prove Theorem 3.3, we need the following proposition.

Proposition 3.4 (Jensen's inequality). *Assume $f : \Omega \to \mathbb{R}$ is a convex function and $x_1, x_2, ..., , x_n, \sum_{i=1}^n \gamma_i x_i / \sum_{i=1}^n \gamma_i \in \Omega$ with weights $\gamma_i > 0$, then*

$$f\left(\frac{\sum_{i=1}^n \gamma_i x_i}{\sum_{i=1}^n \gamma_i}\right) \leqslant \frac{\sum_{i=1}^n \gamma_i f(x_i)}{\sum_{i=1}^n \gamma_i}$$

For a graphical "proof" follow this link.

Proof of Theorem 3.3. Recall the two steps update rule of projected gradient descent

$$\begin{aligned} y_{s+1} &= x_s - \eta_s \nabla f(x_s) \\ x_{s+1} &= \Pi_\Omega(y_{s+1}) \end{aligned}$$

First, the proof begins by exploring an upper bound of difference between function values $f(x_s)$ and $f(x^*)$.

$$\begin{aligned} f(x_s) - f(x^*) &\leqslant \nabla f(x_s)^\top (x_s - x^*) - \frac{\alpha}{2}\|x_s - x^*\|^2 \\ &= \frac{1}{\eta_s}(x_s - y_{s+1})^\top (x_s - x^*) - \frac{\alpha}{2}\|x_s - x^*\|^2 && \text{(by update rule)} \\ &= \frac{1}{2\eta_s}(\|x_s - x^*\|^2 + \|x_s - y_{s+1}\|^2 - \|y_{s+1} - x^*\|^2) - \frac{\alpha}{2}\|x_s - x^*\|^2 \\ & && \text{(by "Fundamental Theorem of Optimization")} \\ &= \frac{1}{2\eta_s}(\|x_s - x^*\|^2 - \|y_{s+1} - x^*\|^2) + \frac{\eta_s}{2}\|\nabla f(x_s)\|^2 - \frac{\alpha}{2}\|x_s - x^*\|^2 \\ & && \text{(by update rule)} \\ &\leqslant \frac{1}{2\eta_s}(\|x_s - x^*\|^2 - \|x_{s+1} - x^*\|^2) + \frac{\eta_s}{2}\|\nabla f(x_s)\|^2 - \frac{\alpha}{2}\|x_s - x^*\|^2 \\ & && \text{(by Lemma 2.5)} \\ &\leqslant (\frac{1}{2\eta_s} - \frac{\alpha}{2})\|x_s - x^*\|^2 - \frac{1}{2\eta_s}\|x_{s+1} - x^*\|^2 + \frac{\eta_s L^2}{2} && \text{(by Lipschitzness)} \end{aligned}$$

By multiplying s on both sides and substituting the step size η_s by $\frac{2}{\alpha(s+1)}$, we get

$$s(f(x_s) - f(x^*)) \leqslant \frac{L^2}{\alpha} + \frac{\alpha}{4}(s(s-1)\|x_s - x^*\|^2 - s(s+1)\|x_{s+1} - x^*\|^2)$$

Finally, we can find the upper bound of the function value shown in Theorem 3.3 obtained using t steps projected gradient descent

$$\begin{aligned}
f\left(\sum_{s=1}^{t}\frac{2s}{t(t+1)}x_s\right) &\leqslant \sum_{s=1}^{t}\frac{2s}{t(t+1)}f(x_s) && \text{(by Proposition 3.4)}\\
&\leqslant \frac{2}{t(t+1)}\sum_{s=1}^{t}\left(sf(x^*)+\frac{L^2}{\alpha}+\frac{\alpha}{4}(s(s-1)\|x_s-x^*\|^2-s(s+1)\|x_{s+1}-x^*\|^2)\right)\\
&= \frac{2}{t(t+1)}\sum_{s=1}^{t}sf(x^*)+\frac{2L^2}{\alpha(t+1)}-\frac{\alpha}{2}\|x_{t+1}-x^*\|^2\\
& && \text{(by telescoping sum)}\\
&\leqslant f(x^*)+\frac{2L^2}{\alpha(t+1)}
\end{aligned}$$

This concludes that solving an optimization problem with a strongly convex objective function with projected gradient descent has a convergence rate is of the order $\frac{1}{t+1}$, which is faster compared to the case purely with Lipschitzness. ■

3.5 Convergence rate for smooth and strongly convex functions

Theorem 3.5. *Assume $f\colon \mathbb{R}^n \to \mathbb{R}$ is α-*strongly convex *and β-*smooth*. Let x^* be an optimizer of f, and let x_t be the updated point at step t using gradient descent with a constant step size $\frac{1}{\beta}$, i.e. using the update rule $x_{t+1} = x_t - \frac{1}{\beta}\nabla f(x_t)$. Then*

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

In order to prove Theorem 3.5, we require use of the following lemma.

Lemma 3.6. *Assume f as in Theorem 3.5. Then $\forall x, y \in \mathbb{R}^n$ and an update of the form $x^+ = x - \frac{1}{\beta}\nabla f(x)$,*

$$f(x^+)-f(y) \leqslant \nabla f(x)^\top(x-y)-\frac{1}{2\beta}\|\nabla f(x)\|^2-\frac{\alpha}{2}\|x-y\|^2$$

Proof of Lemma 3.6.

$$\begin{aligned}
f(x^+)-f(x)+f(x)-f(y) &\leqslant \nabla f(x)^\top(x^+-x)+\frac{\beta}{2}\|x^+-x\|^2 && \text{(Smoothness)}\\
&\quad+\nabla f(x)^\top(x-y)-\frac{\alpha}{2}\|x-y\|^2 && \text{(Strong convexity)}\\
&= \nabla f(x)^\top(x^+-y)+\frac{1}{2\beta}\|\nabla f(x)\|^2-\frac{\alpha}{2}\|x-y\|^2\\
& && \text{(Definition of } x^+\text{)}\\
&= \nabla f(x)^\top(x-y)-\frac{1}{2\beta}\|\nabla f(x)\|^2-\frac{\alpha}{2}\|x-y\|^2\\
& && \text{(Definition of } x^+\text{)}
\end{aligned}$$

■

Now with Lemma 3.6 we are able to prove Theorem 3.5.

Proof of Theorem 3.5.

$$\begin{aligned}
\|x_{t+1} - x^*\|^2 &= \|x_t - \frac{1}{\beta}\nabla f(x_t) - x^*\|^2 \\
&= \|x_t - x^*\|^2 - \frac{2}{\beta}\nabla f(x_t)^\top (x_t - x^*) + \frac{1}{\beta^2}\|\nabla f(x_t)\|^2 \\
&\leqslant (1 - \frac{\alpha}{\beta})\|x_t - x^*\|^2 \qquad \text{(Use of Lemma 3.6 with } y = x^*, x = x_t) \\
&\leqslant (1 - \frac{\alpha}{\beta})^t \|x_1 - x^*\|^2 \\
&\leqslant \exp(-t\frac{\alpha}{\beta})\|x_1 - x^*\|^2
\end{aligned}$$

■

We can also prove the same result for the constrained case using projected gradient descent.

Theorem 3.7. *Assume $f\colon \Omega \to \mathbb{R}$ is α-*strongly convex *and β-*smooth*. Let x^* be an optimizer of f, and let x_t be the updated point at step t using projected gradient descent with a constant step size $\frac{1}{\beta}$, i.e. using the update rule $x_{t+1} = \Pi_\Omega(x_t - \frac{1}{\beta}\nabla f(x_t))$ where Π_Ω is the projection operator. Then*

$$\|x_{t+1} - x^*\|^2 \leqslant \exp(-t\frac{\alpha}{\beta})\|x_1 - x^*\|^2$$

As in Theorem 3.5, we will require the use of the following Lemma in order to prove Theorem 3.7.

Lemma 3.8. *Assume f as in Theorem 3.5. Then $\forall x, y \in \Omega$, define $x^+ \in \Omega$ as $x^+ = \Pi_\Omega(x - \frac{1}{\beta}\nabla f(x))$ and the function $g\colon \Omega \to \mathbb{R}$ as $g(x) = \beta(x - x^+)$. Then*

$$f(x^+) - f(y) \leqslant g(x)^\top (x - y) - \frac{1}{2\beta}\|g(x)\|^2 - \frac{\alpha}{2}\|x - y\|^2$$

Proof of Lemma 3.8. The following is given by the Projection Lemma, for all x, x^+, y defined as in Theorem 3.7.

$$\nabla f(x)^\top (x^+ - y) \leqslant g(x)^\top (x^+ - y)$$

Therefore, following the form of the proof of Lemma 3.6,

$$\begin{aligned} f(x^+) - f(x) + f(x) - f(y) &\leqslant \nabla f(x)^\top (x^+ - y) + \frac{1}{2\beta}\|\nabla g(x)\|^2 - \frac{\alpha}{2}\|x - y\|^2 \\ &\leqslant \nabla g(x)^\top (x^+ - y) + \frac{1}{2\beta}\|\nabla g(x)\|^2 - \frac{\alpha}{2}\|x - y\|^2 \\ &= \nabla g(x)^\top (x - y) - \frac{1}{2\beta}\|\nabla g(x)\|^2 - \frac{\alpha}{2}\|x - y\|^2 \end{aligned}$$

■

The proof of Theorem 3.7 is exactly as in Theorem 3.5 after substituting the appropriate projected gradient descent update in place of the standard gradient descent update, with Lemma 3.8 used in place of Lemma 3.6.

4 Lecture 4: Some applications of gradient methods

This lecture was a sequence of code examples that you can find here:

Lecture 4
(opens in your browser)

5 Lecture 5: Conditional Gradient (Frank-Wolfe)

In this lecture we discuss the conditional gradient method, also known as the Frank-Wolfe (FW) algorithm [FW56]. The motivation for this approach is that the projection step in projected gradient descent can be computationally inefficient in certain scenarios.

5.1 The algorithm

Conditional gradient side steps the projection step using a clever idea.

We start from some point $x_0 \in \Omega$. Then, for time steps $t = 1$ to T, where T is our final time step, we set

$$x_{t+1} = x_t + \eta_t(\bar{x}_t - x_t)$$

where

$$\bar{x}_t = \arg\min_{x \in \Omega} f(x_t) + \nabla f(x_t)^\top (x - x_t).$$

This expression simplifies to:

$$\bar{x}_t = \arg\min_{x \in \Omega} \nabla f(x_t)^\top x$$

Note that we need step size $\eta_t \in [0, 1]$ to guarantee $x_{t+1} \in \Omega$.

So, rather than taking a gradient step and projecting onto the constraint set. We optimize a liner function (defined by the gradient) inside the constraint set.

5.2 Conditional gradient convergence analysis

As it turns out, conditional gradient enjoys a convergence gurantee similar to the one we saw for projected gradient descent.

Theorem 5.1 (Convergence Analysis)**.** *Assume we have a function $f : \Omega \to \mathbb{R}$ that is convex, β-smooth and attains its global minimum at a point $x^* \in \Omega$. Then, Frank-Wolfe achieves*

$$f(x_t) - f(x^*) \leqslant \frac{2\beta D^2}{t+2}$$

with step size

$$\eta_t = \frac{2}{t+2}.$$

Here, D is the diameter of Ω, defined as $D = \max_{x-y\in\Omega} \|x - y\|$.

Note that we can trade our assumption of the existence of x^* for a dependence on L, the Lipschitz constant, in our bound.

Proof of Theorem 5.1. By smoothness and convexity, we have

$$f(y) \leqslant f(x) + \nabla f(x)^\top (x - x_t) + \frac{\beta}{2}\|x - y\|^2$$

Letting $y = x_{t+1}$ and $x = x_t$, combined with the progress rule of conditional gradient descent, the above equation yields:

$$f(x_{t+1}) \leqslant f(x_t) + \eta_t \nabla f(x_t)^\top (\bar{x}_t - x_t) + \frac{\eta_t^2 \beta}{2}\|\bar{x}_t - x_t\|^2$$

We now recall the definition of D from Theorem 5.1 and observe that $\|\bar{x}_t - x_t\|^2 \leqslant D^2$. Thus, we rewrite the inequality:

$$f(x_{t+1}) \leqslant f(x_t) + \eta_t \nabla f(x_t)^\top (x_t^* - x_t) + \frac{\eta_t^2 \beta D^2}{2}$$

Because of convexity, we also have that

$$\nabla f(x_t)^\top (x^* - x_t) \leqslant f(x^*) - f(x_t)$$

Thus,

$$f(x_{t+1}) - f(x^*) \leqslant (1 - \eta_t)(f(x_t) - f(x^*)) + \frac{\eta_t^2 \beta D^2}{2} \tag{20}$$

We use induction in order to prove $f(x_t) - f(x^*) \leqslant \frac{2\beta D^2}{t+2}$ based on Equation 20 above.
First step: Base Case $t = 0$
Since $f(x_{t+1}) - f(x^*) \leqslant (1-\eta_t)(f(x_t) - f(x^*)) + \frac{\eta_t^2 \beta D^2}{2}$, when t=0, $\eta_t = \frac{2}{0+2} = 1, then$

$$\begin{aligned} f(x_1) - f(x^*) &\leqslant (1-\eta_t)(f(x_t) - f(x^*)) + \frac{\beta}{2}\|x_1 - x^*\|^2 \\ &= (1-1)(f(x_t) - f(x^*)) + \frac{\beta}{2}\|x_1 - x^*\|^2 \\ &\leqslant \frac{\beta D^2}{2} \\ &\leqslant \frac{2\beta D^2}{3} \end{aligned}$$

Thus, the induction hypothesis holds for our base case.

Proceeding by induction, we assume that $f(x_t) - f(x^*) \leqslant \frac{2\beta D^2}{t+2}$ holds for all integers up to t and we show the claim for $t+1$.

By Equation 20,

$$\begin{aligned} f(x_{t+1}) - f(x^*) &\leqslant \left(1 - \frac{2}{t+2}\right)(f(x_t) - f(x^*)) + \frac{4}{2(t+2)}\beta D^2 \\ &\leqslant \left(1 - \frac{2}{t+2}\right)\frac{2\beta D^2}{t+2} + \frac{4}{2(t+2)}\beta D^2 \\ &= \beta D^2 \left(\frac{2t}{(t+2)^2} + \frac{2}{(t+2)^2}\right) \\ &= 2\beta D^2 \cdot \frac{t+1}{(t+2)^2} \\ &= 2\beta D^2 \cdot \frac{t+1}{t+2} \cdot \frac{1}{t+2} \\ &\leqslant 2\beta D^2 \cdot \frac{t+2}{t+3} \cdot \frac{1}{t+2} \\ &= 2\beta D^2 \frac{1}{t+3} \end{aligned}$$

Thus, the inequality also holds for the $t+1$ case.

■

5.3 Application to nuclear norm optimization problems

The code for the following examples can be found here.

5.3.1 Nuclear norm projection

The *nuclear norm* (sometimes called *Schatten* 1*-norm* or *trace norm*) of a matrix A, denoted $\|A\|_*$, is defined as the sum of its singular values

$$\|A\|_* = \sum_i \sigma_i(A)\,.$$

The norm can be computed from the singular value decomposition of A. We denote the unit ball of the nuclear norm by

$$B_*^{m\times n} = \{A \in \mathbb{R}^{m\times n} \mid \|A\|_* \leqslant 1\}.$$

How can we project a matrix A onto B_*? Formally, we want to solve

$$\min_{X\in B_*} \|A - X\|_F^2$$

Due to the rotational invariance of the Frobenius norm, the solution is obtained by projecting the singular values onto the unit simplex. This operation corresponds to shifting all singular values by the same parameter θ and clipping values at 0 so that the sum of the shifted and clipped values is equal to 1. This algorithm can be found in [DSSSC08].

5.3.2 Low-rank matrix completion

Suppose we have a partially observable matrix Y, of which the missing entries are filled with 0 and we would like to find its completion form projected on a nuclear norm ball. Formally we have the objective function

$$\min_{X\in B_*} \frac{1}{2}\|Y - P_O(X)|_F^2$$

where P_O is a linear projection onto a subset of coordinates of X specified by O. In this example $P_O(X)$ will generate a matrix with corresponding observable entries as in Y while other entries being 0. We can have $P_O(X) = X \odot O$ where O is a matrix with binary entries. Calculate the gradient of this function we will have

$$\nabla f(X) = Y - X \odot O$$

We can use projected gradient descent to solve this problem but it is more efficient to use Frank-Wolfe algorithm. We need to solve the linear optimization oracle

$$\bar{X}_t \in \operatorname*{argmin}_{X\in B_*} \nabla f(X_t)^\top X$$

To simplify this problem, we need a simple fact that follows from the singular value decomposition.

Fact 5.2. *The unit ball of the nuclear norm is the convex hull of rank-1 matrices*

$$\text{conv}\{uv^\top \,|\, \|u\| = \|v\| = 1, u \in \mathbb{R}^m, v \in \mathbb{R}^n\} = \{X \in \mathbb{R}^{m\times n} \mid \|X\|_* = 1\}\,.$$

From this fact it follows that the minimum of $\nabla f(X_t)^\top X$ is attained at a rank-1 matrix $uv^\top$ for unit vectors u and v. Equivalently, we can maximize $-\nabla f(X_t)^\top uv^\top$ over all unit vectors u and v. Put $Z = -\nabla f(X_T)$ and note that

$$Z^\top uv^\top = \text{tr}(Z^\top uv^\top) = \text{tr}(u^\top Zv) = u^\top Zv\,.$$

Another way to see this is to note that the dual norm of a nuclear norm is operator norm,

$$\|Z\| = \max_{\|X\|_* \leqslant 1} \langle Z, X\rangle\,.$$

Either way, we see that to run Frank-Wolfe over the nuclear norm ball we only need a way to compute the top left and singular vetors of a matrix. One way of doing this is using the classical power method:

- Pick a random unit vector x_1 and let $y_1 = A^\top x / \|A^\top x\|$.
- From $k = 1$ to $k = T - 1$:
  - Put $x_{k+1} = \frac{Ay_k}{\|Ay_k\|}$
  - Put $y_{k+1} = \frac{A^\top x_{k+1}}{\|A^\top x_{k+1}\|}$
- Return x_T and y_T as approximate top left and right singular vectors.

6 Lecture 6: Discovering acceleration

In this lecture, we seek to find methods that converge faster than those discussed in previous lectures. To derive this accelerated method, we start by considering the special case of optimizing quadratic functions.

6.1 Quadratics

Definition 6.1 (Quadratic function)**.** A quadratic function $f\colon \mathbb{R}^n \to \mathbb{R}$ takes the form:

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c,$$

where $A \in S^n$, $b \in \mathbb{R}^n$ and $c \in \mathbb{R}$.

Note that substituting $n = 1$ into the above definition recovers the familiar univariate quadratic function $f(x) = ax^2 + bx + c$ where $a, b, c \in \mathbb{R}$, as expected. There is one subtlety in this definition: we restrict A to be symmetric. In fact, we could allow $A \in \mathbb{R}^{n \times n}$ and this would define the same class of functions, since for any $A \in \mathbb{R}^{n \times n}$ there is a symmetric matrix $\tilde{A} = \frac{1}{2}\left(A + A^T\right)$ for which:

$$x^T A x = x^T \tilde{A} x \; \forall x \in \mathbb{R}^n.$$

Restricting $A \in S^n$ ensures each quadratic function has a *unique* representation.

The gradient and Hessian of a general quadratic function take the form:

$$\nabla f(x) = Ax - b$$
$$\nabla^2 f(x) = A.$$

Note provided A is non-singular, the quadratic has a unique critical point at:

$$x^* = A^{-1} b.$$

When $A \succ 0$, the quadratic is *strictly convex* and this point is the unique global minima.

6.2 Gradient descent on a quadratic

In this section we will consider a quadratic $f(x)$ where A is positive definite, and in particular that:

$$\alpha I \preceq A \preceq \beta I,$$

for some $0 < \alpha < \beta$. This implies that f is α-strongly convex and β-smooth.

From Theorem 3.7 we know that under these conditions, gradient descent with the appropriate step size converges linearly at the rate $\exp\left(-t\frac{\alpha}{\beta}\right)$. Clearly the size of $\frac{\alpha}{\beta}$ can dramatically affect the convergence guarantee. In fact, in the case of a quadratic, this is related to the *condition number* of the matrix A.

Definition 6.2 (Condition number)**.** Let A be a real matrix. Its *condition number* (with respect to the Euclidean norm) is:

$$\kappa(A) = \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)},$$

the ratio of its largest and smallest eigenvalues.

So in particular, we have that $\kappa(A) \leqslant \frac{\beta}{\alpha}$; henceforth, we will assume that α, β correspond to the minimal and maximal eigenvalues of A so that $\kappa(A) = \frac{\beta}{\alpha}$. It follows that gradient descent with a constant step size $\frac{1}{\beta}$ converges at:

$$||x_{t+1} - x^*\|^2 \leqslant \exp\left(-t\frac{1}{\kappa}\right)\|x_1 - x^*\|^2.$$

In many cases, f is ill-conditioned and κ can easily take values in the hundreds or thousands. In this case, case convergence could be very slow: note that at $t > \kappa$, the error may have been reduced by only a factor of $3\times$. Can we do better than this?

In the case of a quadratic, we can of course use the analytic solution $x^* = A^{-1}b$. However, it will prove instructive to consider applying gradient descent to quadratic functions, and derive the convergence bound that we previously proved for any strongly convex smooth functions. This exercise will show us where we are losing performance, and suggest a method that can attain better guarantees.

6.2.1 Convergence analysis

Theorem 6.3. *Assume $f : \mathbb{R}^n \to \mathbb{R}$ is a quadratic where the quadratic coefficient matrix has a condition number κ. Let x^* be an optimizer of f, and let x_t be the updated point at step t using gradient descent with a constant step size $\frac{1}{\beta}$, i.e. using the update rule $x_{t+1} = x_t - \frac{1}{\beta}\nabla f(x_t)$. Then:*

$$\|x_{t+1} - x^*\|^2 \leqslant \exp\left(-t\frac{1}{\kappa}\right)\|x_1 - x^*\|^2.$$

Proof. Write:

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c,$$

where $A \in S^n$, $b \in \mathbb{R}^n$ and $c \in \mathbb{R}$. A gradient descent update with step size η_t takes the form:

$$x_{t+1} = x_t - \eta_t \nabla f(x_t) = x_t - \eta_t\left(Ax_t - b\right)$$

Subtracting x^* from both sides of this equation and using the property that $Ax^* - b = \nabla f(x^*) = 0$:

$$\begin{aligned} x_{t+1} - x^* &= \left(x_t - \eta_t\left(Ax_t - b\right)\right) - \left(x^* - \eta_t\left(Ax^* - b\right)\right) \\ &= (I - \eta_t A)(x_t - x^*). \end{aligned}$$

S Thus:

$$\begin{aligned} \|x_{t+1} - x^*\|_2 &\leqslant \|(I - \eta_t A)\|_2 \|(x_t - x^*)\|_2 \\ &\leqslant \left(\prod_{k=1}^{t}\|I - \eta_k A\|_2\right)\|x_1 - x^*\|_2. \end{aligned}$$

Set $\eta_k = \frac{1}{\beta}$ for all k. Note that $\frac{\alpha}{\beta}I \preceq \frac{1}{\beta}A \preceq I$, so:

$$\max_{A\in\mathbb{R}^{n\times n}}\left\|I - \frac{1}{\beta}A\right\|_2 = 1 - \frac{\alpha}{\beta} = 1 - \frac{1}{\kappa}.$$

It follows that:

$$\begin{aligned}\|x_{t+1} - x^*\|_2 &\leqslant \left(1 - \frac{1}{\kappa}\right)^t \|x_1 - x^*\|_2 \\ &\leqslant \exp\left(-\frac{t}{\kappa}\right) \|x_1 - x^*\|_2 .\end{aligned}$$

■

6.3 Accelerated gradient descent

In the previous section, we proved an upper bound on the convergence rate. In this section, we would like to improve on this. Consider whether there was any point where we were careless in the proof above? One obvious candidate is that our choice of step size, $\eta_k = \frac{1}{\beta}$, was chosen rather arbitrarily. In fact, by choosing the sequence η_k we can select *any* degree-t polynomial of the form:

$$p(A) = \prod_{k=1}^{t} (I - \eta_K A) .$$

Note that:

$$\|p(A)\|_2 = \max_{x \in \lambda(A)} |p(x)|$$

where $p(A)$ is a matrix polynomial, and $p(t)$ is the corresponding scalar polynomial. In general, we may not know the set of eigenvalues $\lambda(A)$, but we do know that all eigenvalues are in the range $[\alpha, \beta]$. Relaxing the inequality accordingly:

$$\|p(A)\| = \max_{x \in [\alpha, \beta]} |p(x)| .$$

We want a polynomial $p(t)$ that takes on small values in $[\alpha, \beta]$. We impose an additional normalization constraint that $p(0) = 1$ (otherwise we could 'cheat' by scaling the polynomial down everywhere on its domain), but allow it to take on arbitrary values elsewhere.

6.3.1 A naive polynomial solution

A naive solution chooses a uniform step size $\eta_t = \frac{2}{\alpha + \beta}$. Note that:

$$\max_{x \in [\alpha, \beta]} \left| 1 - \frac{2}{\alpha + \beta} x \right| = \frac{\beta - \alpha}{\alpha + \beta} \leqslant \frac{\beta}{\alpha} = \kappa,$$

recovering the same convergence rate we proved previously.

The resulting polynomial $p_t(x)$ is plotted in figure 3 for degrees $t = 3$ and $t = 6$, with $\alpha = 1$ and $\beta = 10$. Note that doubling the degree from three to six only halves the maximum absolute value the polynomial attains in $[\alpha, \beta]$, explaining why convergence is so slow.

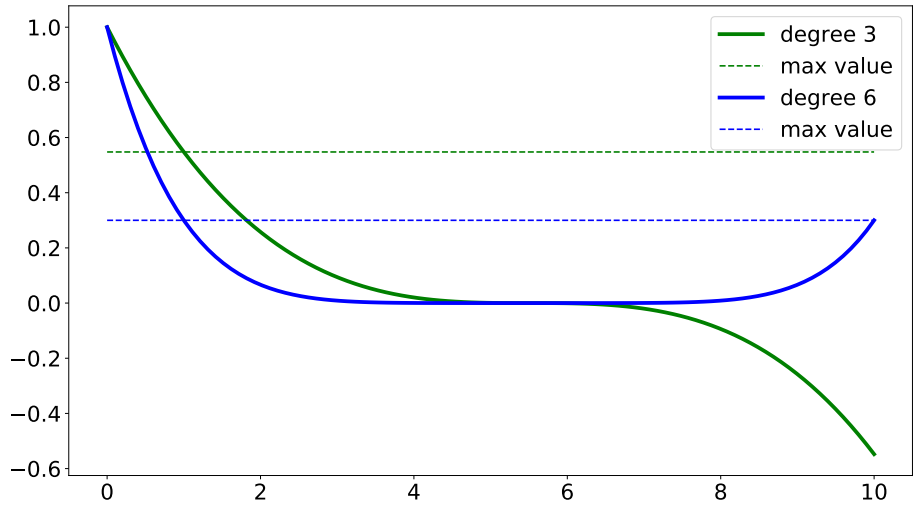


Figure 3: Naive Polynomial

6.4 Chebyshev polynomials

Fortunately, we can do better than this by speeding up gradient descent using Chebyshev polynomials. The Chebyshev polynomials are a sequence of orthogonal polynomials which are related to de Moivre's formula and which can be defined recursively. We will use Chebyshev polynomials of the first kind which are donated T_n.

The Chebyshev polynomials of the first kind are defined by the recurrence relation:

$$T_0(a) = 1, \quad T_1(a) = a$$
$$T_k(a) = 2aT_{k-1}(a) - T_{k-2}(a), \text{ for } k \geqslant 2.$$

And figure 4 is the plot of the Chebyshev polynomials for $i = 0, 1, 2, 3, 4$.

Why Chebyshev polynomials? Suitably rescaled, they minimize the absolute value in a desired interval $[\alpha, \beta]$ while satisfying the normalization constraint of having value 1 at the origin.

Recall that the eigenvalues of the matrix we consider are in the interval $[\alpha, \beta]$. We need to rescale the Chebyshev polynomials so that they're supported on this interval and still attain value 1 at the origin. This is accomplished by the polynomial

$$P_k(a) = \frac{T_k\left(\frac{\alpha+\beta-2a}{\beta-\alpha}\right)}{T_k\left(\frac{\alpha+\beta}{\beta-\alpha}\right)}.$$

We see on figure 5 that doubling the degree has a much more dramatic effect on the magnitude of the polynomial in the interval $[\alpha, \beta]$.

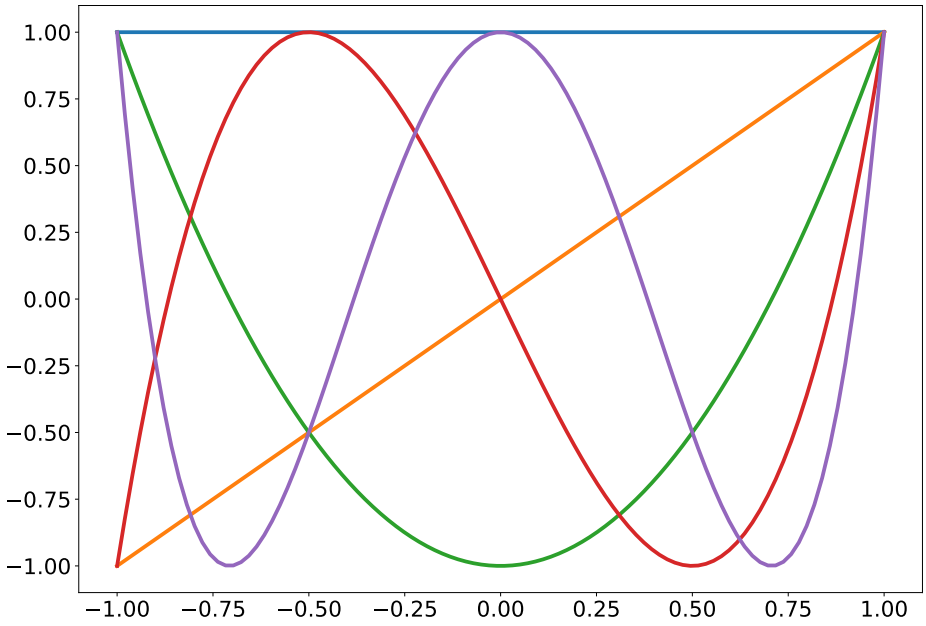


Figure 4: Chebychev polynomials

Let's compare on figure 6 this beautiful Chebyshev polynomial side by side with the naive polynomial we saw earlier. The Chebyshev polynomial does much better: at around 0.3 for degree 3 (needed degree 6 with naive polynomial), and below 0.1 for degree 6.

6.4.1 Accelerated gradient descent

The Chebyshev polynomial leads to an accelerated version of gradient descent. Before we describe the iterative process, let's first see what error bound comes out of the Chebyshev polynomial.

So, just how large is the polynomial in the interval $[\alpha, \beta]$? First, note that the maximum value is attained at α. Plugging this into the definition of the rescaled Chebyshev polynomial we get the upper bound for any $a \in [\alpha, \beta]$,

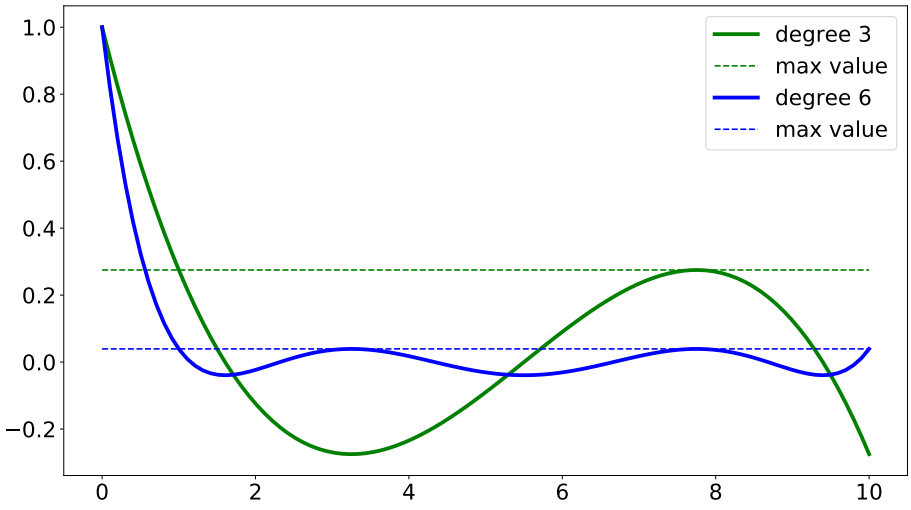


Figure 5: Rescaled Chebyshev

$$\begin{aligned}|P_k(a)| &\leqslant |P_k(\alpha)| \\ &= \frac{|T_k(1)|}{|T_K\left(\frac{\beta+\alpha}{\beta-\alpha}\right)|} \\ &= |T_K\left(\frac{\beta+\alpha}{\beta-\alpha}\right)^{-1}|.\end{aligned}$$

Recalling the condition number $\kappa = \beta/\alpha$, we have

$$\frac{\beta+\alpha}{\beta-\alpha} = \frac{\kappa+1}{\kappa-1}.$$

Typically κ is large, so this is $1+\epsilon$, $\epsilon \approx \frac{2}{\kappa}$. Therefore, we have

$$|P_k(a)| \quad \leqslant |T_k(1+\epsilon)^{-1}|.$$

To upper bound $|P_k|$, we need to lower bound $|T_k(1+\epsilon)|$.
Fact: for $a > 1$, $T_k(a) = \cosh\left(k \cdot \operatorname{arccosh}(a)\right)$ where:

$$\cosh(a) = \frac{e^a + e^{-a}}{2}, \quad \operatorname{arccosh}(a) = \ln\left(x + \sqrt{x^2-1}\right).$$

Now, letting $\phi = \operatorname{arccosh}(1+\epsilon)$:

$$e^{\phi} = 1 + \epsilon + \sqrt{2\epsilon+\epsilon^2} \geqslant 1 + \sqrt{\epsilon}.$$

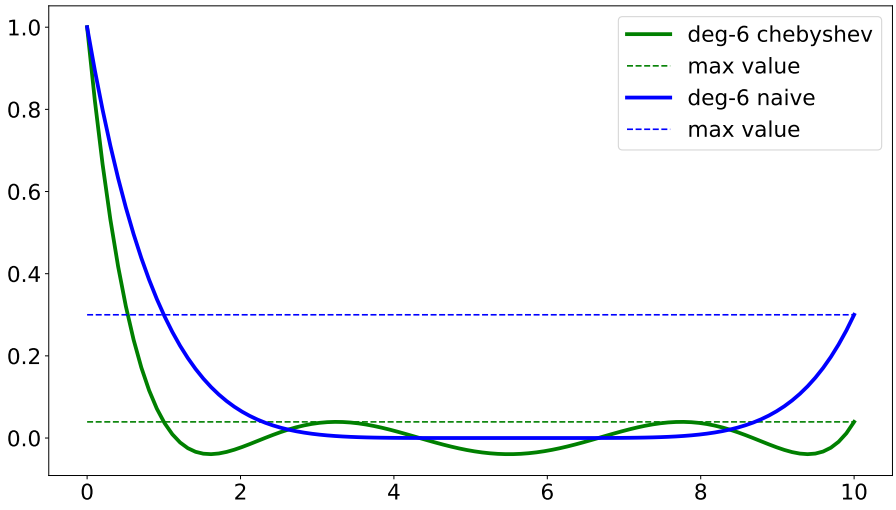


Figure 6: Rescaled Chebyshev VS Naive Polynomial

So, we can lower bound $|T_k(1+\epsilon)|$:

$$\begin{aligned}
|T_k(1+\epsilon)| &= \cosh\left(k\text{arccosh}(1+\epsilon)\right) \\
&= \cosh(k\phi) \\
&= \frac{(e^{\phi})^k + (e^{-\phi})^k}{2} \\
&\geqslant \frac{(1+\sqrt{\epsilon})^k}{2}.
\end{aligned}$$

Then, the reciprocal is what we needed to upper bound the error of our algorithm, so we have:

$$|P_k(a)| \leqslant |T_k(1+\epsilon)^{-1}| \leqslant 2(1+\sqrt{\epsilon})^{-k}.$$

Thus, this establishes that the Chebyshev polynomial achieves the error bound:

$$\begin{aligned}
\|X_{t+1} - X^*\| &\leqslant 2(1+\sqrt{\epsilon})^{-t} \|X_0 - X^*\| \\
&\approx 2(1+\sqrt{\frac{2}{\kappa}})^{-t} \|X_0 - X^*\| \\
&\leqslant 2\exp\left(-t\sqrt{\frac{2}{\kappa}}\right) \|X_0 - X^*\|.
\end{aligned}$$

This means that for large κ, we get quadratic savings in the degree we need before the error drops off exponentially. Figure 7 shows the different rates of convergence, we clearly see that the

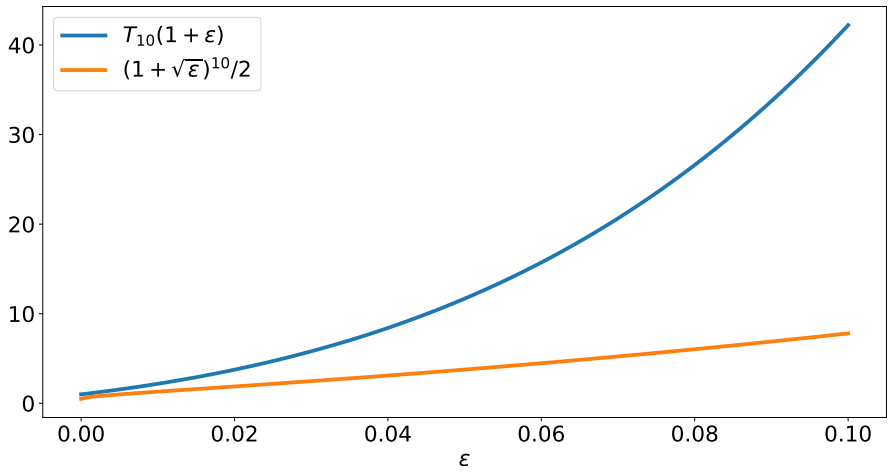


Figure 7: Convergence for naive polynomial and Chebyshev

6.4.2 The Chebyshev recurrence relation

Due to the recursive definition of the Chebyshev polynomial, we directly get an iterative algorithm out of it. Transferring the recursive definition to our rescaled Chebyshev polynomial, we have:

$$P_{K+1}(a) = (\eta_k a + \gamma_k)P_k(a) + \mu_k P_{k-1}(a).$$

where we can work out the coefficients η_k, γ_k, μ_k from the recurrence definition. Since $P_k(0) = 1$, we must have $\gamma_k + \mu_k = 1$. This leads to a simple update rule for our iterates:

$$\begin{aligned} X_{k+1} &= (\eta_k A + \gamma_k)X_k + (1-\gamma_K)X_{k-1} - \eta_k b \\ &= (\eta_K A + (1-\mu_k))X_k + \mu_k X_{k-1} - \eta_k b \\ &= X_k - \eta_k(AX_k - b) + \mu_k(X_k - X_{k-1}). \end{aligned}$$

We see that the update rule above is actually very similar to plain gradient descent except for the additional term $\mu_k(x_k - x_{k-1})$. This term can be interpreted as a *momentum* term, pushing the algorithm in the direction of where it was headed before. In the next lecture, we'll dig deeper into momentum and see how to generalize the result for quadratics to general convex functions.

7 List of contributors

Many thanks to the students of EE227C for their generous help in creating these lecture notes.

Lecture 2:

Lecture 3:

Lecture 5: Victoria Cheng, Kun Qian, Zeshi Zheng

Lecture 6: Adam Gleave, Andy Deng, Mathilde Badoual

8 Acknowledgments

These notes build on an earlier course by Ben Recht, as well as an upcoming textbook by Recht and Wright. Some chapters also closely follow Bubeck's monograph on the topic [Bub15].

References

[Bub15] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3-4):231–357, 2015.

[DSSSC08] John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proc.* 25*th ICML*, pages 272–279. ACM, 2008.

[FW56] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, mar 1956.