CS 726 Lecture: Setting the Stage

Jelena Diakonikolas

A generic optimization problem is of the form

$$\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}). \tag{P}$$

To give problem (P) a meaning, we need to specify (i) the *vector space* in which the vector of variables $\mathbf{x}$ lives, (ii) the *feasible set* $\mathcal{X}$, which is a subset of our vector space and determines what choices of vectors $\mathbf{x}$ we are allowed to output, and (iii) the *objective function* f, which maps each vector of variables $\mathbf{x}$ to a real value. Here, we think about the objective function as some cost or loss that we want to minimize. Alternatively, in some problems, the goal may be to maximize a reward $g(\mathbf{x})$. This setup is no different than (P), as we can take $f(\mathbf{x}) = -g(\mathbf{x})$.

Another thing that we need to consider here is what it means to "solve" (P). In general, unless we make strong assumptions about the objective function and the feasible set (the set $\mathcal{X}$ of "allowed" solutions $\mathbf{x}$), we cannot hope to find the exact vector $\mathbf{x}^*$ that minimizes f over $\mathcal{X}$. Most of the time, this is also not necessary, and we can settle for points that are "good enough." What that means will become clear once we introduce the definitions for different points we call *local minima* or *local solutions*, and once we discuss different notions of approximation for such points.

1 Vector Space: Where Optimization Variables Live

Our main objects of interest are optimization problems in $\mathbb{R}^d$ whose length is measured using one of the standard ℓ_p norms for $p \geq 1$. This means that the vector $\mathbf{x}$ from (P) is a d-dimensional column vector $[x_1, x_2, \dots x_d]^T$ and that $\mathcal{X} \subseteq \mathbb{R}^d$. We will use $\langle \cdot, \cdot \rangle$ to denote any inner product on $\mathbb{R}^d$. Most frequently, it suffices to consider the standard inner product, which, given vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, is defined as:

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{d} x_i y_i. \tag{1}$$

The simplest ℓ_p norm is the Euclidean norm, obtained for $p = 2$. It is denoted as $\| \cdot \|_2$ and defined by

$$\|\mathbf{x}\|_2 = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle} = \sqrt{\sum_{i=1}^{d} {x_i}^2}. \tag{2}$$

More generally, ℓ_p norms for $p \geq 1$ are defined by

$$\|\mathbf{x}\|_p = \Big(\sum_{i=1}^{d} |x_i|^p \Big)^{1/p}. \tag{3}$$

The most commonly used ℓ_p norms are the ℓ_2 (Euclidean) norm, the ℓ_1 norm, and the ℓ_∞ norm. The latter two are just

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{d} |x_i| \qquad \text{and} \qquad \|\mathbf{x}\|_\infty = \max_{1 \leq i \leq d} |x_i|. \tag{4}$$

Most of the time, it will suffice to think about Euclidean norms, though we will sometimes also consider other ℓ_p norms. Once we fix the ℓ_p norm we want to use, we have defined our normed vector space $(\mathbb{R}^d, \| \cdot \|_p)$. The pair $(\mathbb{R}^d, \| \cdot \|_p)$ tells us that the optimization variables are vectors in $\mathbb{R}^d$ and that their length is measured using the ℓ_p norm. We will also call this space the *primal space*. This is it to contrast it with something called the *dual space*. By

definition, the dual space is the space of all linear functions $\mathbf{z}$ acting on $\mathbf{x}$. In our case, these are just d-dimensional vectors, while the value of the linear function $\mathbf{z}$ is simply $\langle \mathbf{z}, \mathbf{x} \rangle$. An important property of the dual space is that the vectors in dual space are measured with respect to the *dual norm*. The definition of a norm $\|\cdot\|_*$ that is dual to $\|\cdot\|$ is

$$\|\mathbf{z}\|_* = \sup_{\mathbf{x} \in \mathbb{R}^d,\ \|\mathbf{x}\|=1} \langle \mathbf{z}, \mathbf{x} \rangle . \tag{5}$$

An immediate consequence of the definition of the dual norm is the inequality that relates the inner product of two vectors $\mathbf{x}, \mathbf{z}$ and their primal and dual norms, respectively, as stated in the following proposition. This inequality can be seen as a generalization of the classical Cauchy-Schwarz inequality, and it will come in handy when we analyze the convergence of different algorithms.

Proposition 1.1. *For any two vectors* $\mathbf{x}, \mathbf{z} \in \mathbb{R}^d$:

$$\langle \mathbf{z}, \mathbf{x} \rangle \leq \|\mathbf{z}\|_* \cdot \|\mathbf{x}\|. \tag{2.1}$$

Proof. Fix any two vectors $\mathbf{x}, \mathbf{z} \in \mathbb{R}^d$. The inequality from the statement of the proposition holds trivially if either $\mathbf{x} = \mathbf{0}$ or $\mathbf{z} = \mathbf{0}$, where $\mathbf{0}$ denotes the vector with all coordinates equal to zero, so assume $\mathbf{x} \neq \mathbf{0}$ and $\mathbf{z} \neq \mathbf{0}$. Let $\hat{\mathbf{x}} = \frac{\mathbf{x}}{\|\mathbf{x}\|}$. Then $\|\hat{\mathbf{x}}\| = 1$, and using the definition of the dual norm, we have:

$$\|\mathbf{z}\|_* = \sup_{\mathbf{x} \in \mathbb{R}^d,\ \|\mathbf{x}\|=1} \langle \mathbf{z}, \mathbf{x} \rangle \geq \langle \mathbf{z}, \hat{\mathbf{x}} \rangle = \frac{\langle \mathbf{z}, \mathbf{x} \rangle}{\|\mathbf{x}\|}. \tag{6}$$

Multiplying both sides of the last inequality by $\|\mathbf{x}\|$, we get $\langle \mathbf{z}, \mathbf{x} \rangle \leq \|\mathbf{z}\|_* \cdot \|\mathbf{x}\|$, as claimed. □

A standard fact about ℓ_p norms is that the norm dual to $\|\cdot\|_p$ is the norm $\|\cdot\|_q$ where q is such that $\frac{1}{p} + \frac{1}{q} = 1$. Thus, the space $(\mathbb{R}^d, \|\cdot\|_q)$ is dual to $(\mathbb{R}^d, \|\cdot\|_p)$ whenever $\frac{1}{p} + \frac{1}{q} = 1$ and $p \geq 1$.

When we are solving an optimization problem, we are typically not told which norm to use. The performance of any algorithm that we will see in this class depends on certain properties of the feasible set $\mathcal{X}$ and the objective function f that are measured with respect to the norm that we select. If, for one choice of the norm, we get that these quantities are finite, then they will be finite with respect to every ℓ_p norm. This is because ℓ_p norms are related by the following (tight) inequalities

$$(\forall \mathbf{x} \in \mathbb{R}^d)(\forall p \geq 1)(\forall r > p): \qquad \|\mathbf{x}\|_r \leq \|\mathbf{x}\|_p \leq d^{\frac{1}{p}-\frac{1}{r}} \|\mathbf{x}\|_r. \tag{2.2}$$

However, the choice of the norm will make a big difference in terms of how fast the algorithm that we apply to our optimization problem will be. This is because the speed with which our algorithm converges will scale with the quantities that we measure with respect to the selected ℓ_p norm, and, thus, selecting a wrong norm may mean we end up with an algorithm that is slower by factor polynomial in d. For large-scale problems that we are interested in here, this difference determines whether we would be able to use our algorithm in practice or not.

2 Feasible Set

The feasible set determines what choices of vectors $\mathbf{x}$ are acceptable as possible solutions to (P). If $\mathcal{X} \equiv \mathbb{R}^d$, we say that the problem (P) is *unconstrained*. Otherwise, we are dealing with a *constrained* optimization problem. This distinction between constrained and unconstrained problems is not always completely clear or even necessary, as we will see in the next section.

We typically use one of the following two ways of thinking about and expressing feasible sets: (i) as *abstract* sets $\mathcal{X}$ that we view as some geometric bodies, such as, e.g., a ball, a box, or a polyhedron; and (ii) as *constraint-based* sets described by functional constraints of the form $f_i(\mathbf{x}) \leq 0$, for $i \in \{1, 2, \ldots, m\}$ and $h_j(\mathbf{x}) = 0$, for $j \in \{1, 2, \ldots, p\}$. Of course, the main difference here is in how we express or think about the feasible set; for example, if our feasible set is a unit Euclidean ball centered at the origin, then we could describe it by the constraint $\|\mathbf{x}\|_2^2 - 1 \leq 0$ or just think about it as an abstract geometric body. Observe here that the two types of constraints we wrote in (ii) are without loss of generality, as, for example, constraints of the form $\bar{f}_i(\mathbf{x}) \geq C$ can equivalently be written as $f_i \leq 0$ for $f_i(\mathbf{x}) = -\bar{f}_i(\mathbf{x}) + C$.

One of the most important properties of sets is convexity. Except for some special cases, we often need to make an assumption that the feasible set is convex to be able to guarantee that any algorithm we select converges to an acceptable solution (defined later), and that it does so at some reasonable speed.

Definition 2.1. We say that a set $\mathcal{X}$ is convex if for any pair of points from the set $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, the line between $\mathbf{x}$ and $\mathbf{y}$ is fully contained in $\mathcal{X}$. In other words, $\mathcal{X}$ is convex if for any pair of points $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ and any $\alpha \in (0, 1)$:

$$(1 - \alpha)\mathbf{x} + \alpha\mathbf{y} = \mathbf{x} + \alpha(\mathbf{y} - \mathbf{x}) \in \mathcal{X}. \tag{7}$$

If a set is not convex, we say that it is nonconvex.

One of the reasons why we do not consider completely arbitrary nonconvex sets is that we would very quickly run into computational bottlenecks. For example, a simple nonconvex constraint of the form $x_i(1 - x_i) = 0$ would force the i^{th} coordinate of $\mathbf{x}$ to only take one of the two possible values: 0 or 1. This would enable us to write optimization problems that are not too large (e.g., have size polynomial in d) but are not solvable with any known polynomial-time algorithms. In particular, we would be able to express problems that are from the computational class known as NP-complete, for which existing provably convergent algorithms can only handle very small problem instances, with d at the order of 10-100.

We will always assume that the feasible set is *closed*. Of course, to talk about closed (or open) sets, we need to define a topology. Since we will only be working in $\mathbb{R}^d$, whenever we talk about open or closed sets, we will be assuming the *standard topology on* $\mathbb{R}^d$.

Since we are only concerned with finite-dimensional optimization problems, Heine-Borel theorem implies that every closed and bounded set is compact. Compactness, in turn, guarantees that every continuous function defined and continuous on this set attains its extremal values on it, by the well-known Weierstrass's theorem. Thus, compact sets are of particular interest, as we do not need to worry about whether a minimizer of the function exists – as already noted, this is guaranteed by the theorem of Weierstrass. On the other hand, whenever we work with unconstrained optimization problems, to obtain any meaningful convergence guarantees, we will always assume that the objective function f is bounded below on its domain (even though it might not attain its minimum on $\mathbb{R}^d$).

3 Basic Properties of the Objective Function

We will never work with completely arbitrary objective functions. The reason is that, similarly as for feasible sets, we would quickly run into computationally intractable problems.

Recall that the domain of a function is the set of points on which the function is well-defined. We will always assume that the domain $\mathcal{D}$ of the objective function f is a subset of $\mathbb{R}^d$. It will also be convenient to assume that f maps $\mathcal{D}$ to the extended real line $\bar{\mathbb{R}} \stackrel{\text{def}}{=} \mathbb{R} \cup \{-\infty, +\infty\}$, i.e., that f is *extended real valued*. Working with extended real valued functions is particularly useful, as we can define each function $f : \mathcal{D} \to \bar{\mathbb{R}}$ on the entire $\mathbb{R}^d$ by assigning it value $+\infty$ on all points that lie outside the domain $\mathcal{D}$. Note that we do not lose anything by doing this, as our goal is to minimize f. Once we have defined f on the entire $\mathbb{R}^d$, we only need to consider the *effective domain* of f, defined as the set of points on which f takes values lower than $+\infty$, i.e., $\text{dom}(f) = \{\mathbf{x} \in \mathbb{R}^d : f(\mathbf{x}) < \infty\}$. In the sequel, whenever we talk about the domain of f, we will be referring to the effective domain of f, as this simplifies the discussion and is without loss of generality. We say that a function f is *proper* if it is not equal to $\pm\infty$ everywhere, i.e., if it takes at least some real values. Note that a necessary condition for f to be proper is that its effective domain is non-empty.

The field of linear and nonlinear optimization is often referred to as the "continuous" optimization, typically to contrast it with discrete, or combinatorial, optimization. Thus, it seems natural that the minimum assumption we would make about the objective function is that it be *continuous*. In most settings, this will indeed be the minimum assumption we make about the objective function. However, continuity can be relaxed slightly, if additional structure such as convexity (defined later in this section) and certain "simplicity" structure can be assumed. In those cases we may allow the objective function to be *lower semicontinuous*, defined as follows.

Definition 3.1. We say that a function $f : \mathbb{R}^d \to \bar{\mathbb{R}}$ is lower semicontinuous at $\bar{\mathbf{x}} \in \mathbb{R}^d$ if $\liminf_{\mathbf{x} \to \bar{\mathbf{x}}} \geq f(\bar{\mathbf{x}})$ and lower semicontinuous on $\mathbb{R}^d$ if it is lower semicontinuous at every point $\bar{\mathbf{x}} \in \mathbb{R}^d$.

The main usefulness of lower semicontinuous functions (and likely the main reason they are even considered in nonlinear optimization) is that the indicator function of a closed set $\mathcal{X}$, defined as

$$I_{\mathcal{X}}(\mathbf{x}) = \begin{cases} 0, & \text{if } \mathbf{x} \in \mathcal{X} \\ +\infty, & \text{otherwise,} \end{cases} \tag{8}$$

is a lower semicontinuous function (verify this!). Thus, we can view constrained and unconstrained optimization problems in a unified way: minimizing f over a closed set $\mathcal{X}$ is the same as minimizing $f + I_{\mathcal{X}}$ over the entire $\mathbb{R}^d$!

Of course, while lower semicontinuous functions are useful for abstracting away constraints, we cannot work with arbitrary such functions. To see this, consider a function $f(\cdot) = I_{\mathcal{X}}(\cdot)$, where $\mathcal{X}$ is some closed set that is *not* known to us. If we access f by only querying its local information – e.g., for any candidate point $\mathbf{x} \in \mathbb{R}^d$, we get back the value of f at $\mathbf{x}$ and all of f's derivatives that exist, if any, at $\mathbf{x}$ (most optimization algorithms will work in this way) – then unless we are lucky and query a point inside $\mathcal{X}$, we get no clue in which direction to move. Thus, whenever we make an assumption that some function $f(\mathbf{x})$ is lower semicontinuous, we will also assume that it is convex (defined below) and that problems of the form $\min_{\mathbf{x}\in\mathbb{R}^d} g(\mathbf{x}) + f(\mathbf{x})$ are easily solvable for any function g from a given class of "simple" functions, such as linear or quadratic functions.

Most often, the minimum assumption we will make about a function in terms of its continuity is that it is Lipschitz-continuous.

Definition 3.2. We say that a function $f : \mathbb{R}^d \to \bar{\mathbb{R}}$ is Lipschitz-continuous on a set $\mathcal{X} \subseteq \mathbb{R}^d$ if there exists a constant $M < \infty$ such that for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ we have:

$$|f(\mathbf{x}) - f(\mathbf{y})| \leq M\|\mathbf{x} - \mathbf{y}\|.$$

Observe that, as we are only considering finite-dimensional normed vector spaces, if a function is Lipschitz-continuous w.r.t. some norm $\|\cdot\|$ then it is Lipschitz-continuous w.r.t. every other norm. However, the Lipschitz constant M depends on the selected norm, and this is crucial to the performance of the algorithms in different setups.

Lipschitz-continuity of a function on its own is not sufficient for obtaining efficient optimization algorithms even if the function is bounded below; we always need to assume more structure such as, e.g., convexity. Thus, without convexity, we make additional assumptions about the objective function, such as that it is differentiable and its gradients are Lipschitz-continuous. Such functions are referred to as *smooth* functions.

Definition 3.3. We say that a differentiable function $f : \mathbb{R}^d \to \bar{\mathbb{R}}$ is smooth on a set $\mathcal{X} \subseteq \mathbb{R}^d$ if there exists a constant $L < \infty$ such that for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$:

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|_* \leq L\|\mathbf{x} - \mathbf{y}\|,$$

where $\|\cdot\|, \|\cdot\|_*$ is a pair of dual norms.

Functions that are convex are of particular interest in nonlinear optimization, due to their rich structure that enables the design of efficient optimization algorithms.

Definition 3.4. We say that a function $f : \mathbb{R}^d \to \bar{\mathbb{R}}$ is convex, if for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ and any $\alpha \in (0, 1)$:

$$f((1-\alpha)\mathbf{x} + \alpha\mathbf{y}) \leq (1-\alpha)f(\mathbf{x}) + \alpha f(\mathbf{y}).$$

It is also possible to equivalently define convex functions via the definition of convex sets, using the notion of an epigraph, as follows.

Lemma 3.5. *Let $f : \mathbb{R}^d \to \bar{\mathbb{R}}$. Then f is convex if and only if its epigraph, defined as*

$$\mathrm{epi}(f) = \{(\mathbf{x}, a) : \mathbf{x} \in \mathbb{R}^d, a \in \mathbb{R}, f(\mathbf{x}) \leq a\},$$

is convex.

The proof is left as an exercise.